API Reference

Reference Guide

Changepoint 2021

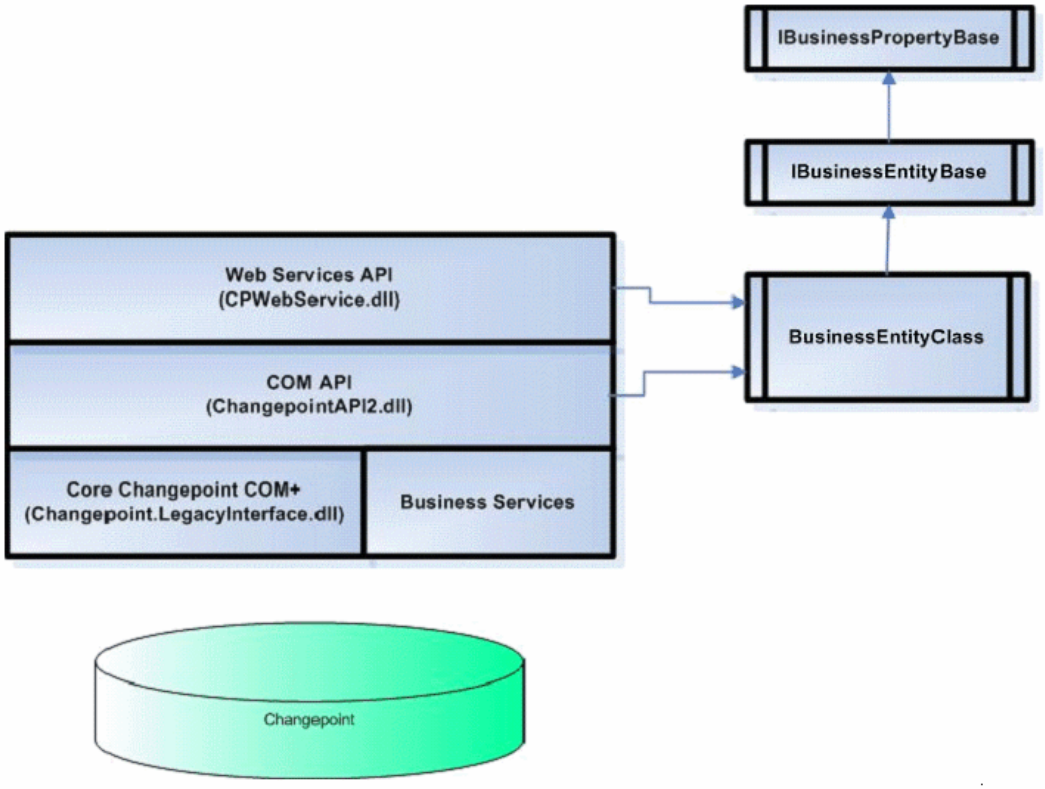
Contents

1. Changepoint COM API Objects and Methods

Changepoint COM API overview

The Changepoint API is a COM+ interface. The Changepoint API can only be run on computers that support COM+ (Windows 2000 or above). This allows the API to use the core Changepoint COM+ business logic (Changepoint.LegacyInterface.dll).

The COM API and Web Services API share the business entity object definition. Although they share some common properties, they have their own instances and their own functional interfaces.



The Changepoint API is not designed to check for and enforce business relations as is done in the user interface. Note the following limitations:

- If a GUID is passed as a parameter to a method, the API attempts to save it as is (for example, .Project.Id or .Product.Id when creating a Request object)
- You must pass the correct GUID to the API methods.

For information on requirements and installation, refer to the *Changepoint API Installation Guide*.

Client applications

All COM API 'client applications' require an app.config file, which must contain the following:

```
<?xml version="1.0"?>
<configuration>
   <appSettings>
    <add key="GlobalDebug" value="False" />
    <add key="GlobalCommandTimeout" value="0" />
    <add key="SQLSettings" value="server=SERVERNAME;database=DATABASENAME;trusted_
connection=no;uid=USERID;pwd=PASSWORD"/>
   </appSettings>
</configuration>
```

The SQLSettings value must be encrypted. Use the APITest Kit to generate the encrypted connection string.

Parameters

In the COM API methods, some parameters are mandatory and some are optional. The required or optional status is indicated in the method syntax as well as in the descriptions of the parameters.

Usage notes

- The Changepoint API supports the use of floating point and extended ASCII characters, except for the pipe character (|).
- Customers/clients, projects, contracts/initiatives, opportunities/candidates, requests, products/applications, resources, tasks and task assignment support an unlimited number of configurable fields. Configurable fields must be defined in System Manager.
- The Changepoint API does not validate conditional values and properties for configurable fields.

About APIs and XML

Many of the Changepoint APIs have a corresponding XML structure and methods for working with the XML structure.

Naming conventions for XML elements

The XML elements have the same name as the object properties except for the following:

- API object properties with the signature data type contain both identity and timestamp information. An object property with the signature data type (for example, CreatedBy) is usually mapped to two separate XML elements (for example, createdbyid and createdon).
- The sxmlUDF object property corresponds to the <udf> XML element.
- An API object property with the identity data type usually maps an XML element of the same name, and the identifying information is usually stored in the child nodes (id, name, alternatename, etc.). There are a few exceptions to this general rule. If child nodes are not used in the XML, the identifying information is stored in different XML elements. For example, the customer property of ApiEngSplitBillingRule maps to the customerid and customername XML elements.

Mandatory properties and elements

If an API object property is mandatory, the corresponding XML element is mandatory. Mandatory properties are marked with an asterisk (*) in the object property tables.

Child nodes and lookups

Many of the XML elements have child nodes, such as id, name, alternatename, userdefinedid, firstname, lastname, and email1. The value of the id child node is required if the parent element is mandatory or if an add or update is necessary. The CreateByXML or UpdateByXML method can look up the id if the appropriate information is provided. The following table outlines the basic requirements for looking up an id based on a given set of child nodes. The numbering in the right column is the lookup order. For example, if a lookup can be performed based on userdefinedid or name, and if both are specified but have different values, then the lookup will be based on the userdefinedid because it is first in the lookup order.

Child nodes	Value required to look up an id
• id • name • alternatename	name
• id • name • alternatename • userdefinedid	1. userdefinedid 2. name
• id • name • alternatename • firstname • lastname • email1	1. email1 2. firstname and lastname 3. name
• id • name • alternatename • firstname • lastname • userdefinedid	1. userdefinedid 2. firstname and lastname 3. name

ApiConnection

The ApiConnection object holds the database connection and the login user information. Before doing anything to the API, you must first call the ApiConnection.Login() method, then assign the (logged in) ApiConnection object to each Entity Object that needs to be called.

Namespace

Changepoint.ChangepointAPI2.ApiConnection

Methods

Properties

Property (*=required)	Type	Description
ADOConnectionString	String	Read-only. The connection string used in COM+.
ApiVersion	String	Read-only. Retrieves the API version.
ChangepointUserId	String	Read-only. If the login is successful, returns the login ResourceId in the Changepoint database.
Connected	Boolean	Read-only. The login status of the ApiConnection.
ConnectionString	String	The encrypted connection string.
CPException	ApiException	Read-only. The ApiException object which holds the exception information.
LoginId	String	The Changepoint login ID of the user who is connecting to Changepoint. The login ID should be the user's email address.
LoginPassword	String	Write-only. The password associated with the user's login ID.
LoginUser	ApiUser	The ApiUser object that will hold the login user information after the connection is established.

Property (*=required)	Type	Description
LogLevel	Byte	Specifies the level of error information to be returned in COM API log file if the test result shows a problem with the connection. The values are as follows: 0 = NoLog 1 = Source 2 = ErrorMessages 3 = InputParameters 4 = Returns 8 = Checkpoint.
LogPath	String	The log file path, default is: C:\Program Files\Changepoint\Changepoint\ApiLogs\

ApiConnection: GenerateConnectionStr

```
Public Shared Function GenerateConnectionStr(ByVal serverName As String, ByVal databaseName As String, ByVal userId As String, ByVal password As String) As String
```

Purpose

Return encrypted connection string by given server information.

Parameters

Parameter (*=required)	Description
*serverName	Database server name
*databaseName	Database name
*userId	Database login ID
*password	Login password

Returns

An encrypted connection string.

Remarks

It is a shared (static) function.

Example

```
Dim sConnstr As String = ApiConnection.GenerateConnection("myserver",
"Changepoint", "myuser", "mypassword")
```

Related information

"ApiConnection" on page 36

ApiConnection: Login

```
Public Overloads Function Login() As ApiUser

Public Overloads Function Login(ByVal loginId As String, ByVal loginPassword
As String) As ApiUser

Public Overloads Function Login(ByVal dbServerName As String, ByVal dbName As
String, ByVal dbUserName As String, ByVal dbUserPassword As String, ByVal
cpLoginId As String, ByVal cpLoginPassword As String) As ApiUser
```

Purpose

Enables login into the database with the given login ID and login password.

Parameters

In the overloads function with the full set of parameters, all parameters are required. See the remarks and example for more information.

Parameter (*=required)	Description
*loginId	The login ID for the Changepoint user (should be the user's email address).
*loginPassword	The login password for the Changepoint user associated with the login ID.
*dbServerName	Server name.

Parameter (*=required)	Description
*dbName	Database name.
*dbUserName	The login user name for the database.
*dbUserPassword	The login password for the database associated with the dbUserName.
*cpLoginId	The login ID for the Changepoint user (should be the user's email address).
*cpLoginPassword	The login password for the Changepoint user associated with cpLoginId.

Returns

ApiUser if the login succeeded.

Remarks

The loginId and loginPassword parameters must be preset in the overloads function Login() with no parameters.

In the overloads function with no paramenters and in the overloads function with the loginID and loginPassword parameters, the connection string must be set before the function is called.

In the overloads function with the full set of parameters, the function string is set internally

Example

Example 1:

```
Dim myCon as New ApiConnection()
myCon.LoginId=myId
myCon.LoginPassword=myPassword
myCon.ConnectionString=myConStr
Dim myUser as ApiUser = myCon.Login()
```

Example 2:

```
Dim myCon as New ApiConnection()
Dim myUser as ApiUser = myCon.Login("YKZCPDEVX64", "Shangrila", "sqladmin",
"sqladmin","cpadmin", "cpadmin")
```

Related information

"ApiConnection" on page 36

ApiConnection: New

```
Public Sub New()
Public Sub New(ByVal appType As CPAppType)
```

Purpose

ApiConnection object constructor.

Parameters

Parameter (*=required)	Description
*appType (Byte)	Values available with CPAppType.COMAPI = 0 CPAppType.WebService = 1

Returns

None

Remarks

The default appType is CPAppType.COMAPI.

Example

```
Dim myCon as New ApiConnection(CPAppType.COMAPI)
```

Related information

"ApiConnection" on page 36

ApiConnection: ValidateConnection

```
Public Function ValidateConnection() As Int32
```

Purpose

Validates connection to the database server.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

The connection string must be set before the function is called. Checks the log file upon error.

Example

```
Dim myCon as New ApiConnection()
myCon.ConnectionString=myConStr
Dim i as Int32 = myCon.ValidateConnection ()
```

Related information

"ApiConnection" on page 36

ApiConnection: ValidateLoginUser

```
Public Function ValidateLoginUser(Optional ByVal loginId As String = "") As
ApiUser
```

Purpose

Validates login user in the database.

Parameters

Parameter (*=required)	Description
loginId	Optional, the login ID for the Changepoint user (should be the user's email address).

Returns

Returns ApiUser if login is successful.

Remarks

The function will check that the login user is valid in the database. The connection string must be set before the function is called.

Example

```
Dim myCon as New ApiConnection()
myCon.LoginId=myId
myCon.ConnectionString=myConStr
Dim myUser as ApiUser = myCon.ValidateLoginUser ()
```

Related information

"ApiConnection" on page 36

ApiContact

The ApiContact object allows users to create, retrieve and update customer contacts within Changepoint.

Namespace

Changepoint.ChangepointAPI2.ApiContact

Methods

Properties

Property (*=required)	Type	Description
Anniversary	DateTime	Contact's wedding anniversary.
AssistantName	String	Name of the contact's assistant.
AssistantPhone	String	Phone number of the contact's assistant.
AvailableToAll	Boolean	Available for public viewing
Birthday	DateTime	Contact's birthday.
BusinessAddress	String	First line of the contact's business address.
BusinessAddressLine2	String	Second line of the contact's business address.
BusinessAddressLine3	String	Third line of the contact's business address.
BusinessCity	String	City for the contact's business address.
BusinessCountry	Identity	Identifier of the country for the contact's business address.
BusinessFax	String	Fax number for the contact's business.
BusinessPhone	String	Business phone number for the contact.
BusinessPhone2	String	Alternate business phone number for the contact.
BusinessPostal	String	Postal code or zip code for the contact's business address.

Property (*=required)	Type	Description
BusinessProvince	Identity	Identifier of the province or state where the contact's business is located.
BusinessWorkLocation Group	Identity	Identifier of the work location group associated to the contact's business address.
BusinessWorkLocation	Identity	Identifier of the work location associated to the contact's business address.
BypassMetadataCheck	CPMetadataCheck	Determines the level of MetaData checking when adding or updating data. (default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
CarPhone	String	Contact's mobile/cell phone number.
ContactCode1	Identity	Identifier of the first additional code field, if configured.
ContactCode2	Identity	Identifier of the second additional code field, if configured.
ContactCode3	Identity	Identifier of the third additional code field, if configured.
ContactExists	Boolean	A contact exists. Used internally, set if contact exists. If set by calling program, value is ignored.

Property (*=required)	Type	Description
ContactId	String	Identification for the contact. The contact object generates the contactid if a create function (Add or CreateByXml) is called.
ContactText1	String	First additional text field, if configured.
ContactText2	String	Second additional text field, if configured.
ContactText3	String	Third additional text field, if configured.
ContactType	Identity	Identifier of the type of contact or contact relationship.
*CPConnection	ApiConnection	Connection to the Changepoint database
CreatedBy	Signature	Creator of the record.
*Customer	Identity	Identifier of the contact's customer.
Deleted	Boolean	Flag that indicates if the object has been deleted.
Department	String	Business department of which the contact is a member.
Description	String	Additional comments or description concerning the contact.
Email1	String	Primary e-mail address for the contact.
Email2	String	Secondary e-mail address for the contact.
Email3	String	Third email address for the contact.
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.

Property (*=required)	Type	Description
*FirstName	String	First name of the contact.
History	String	History of changes/additions to the contact record.
HistoryUpdates	String	Comment for the history updates section of the contact record.
HomeAddress	String	First line of the contact's home address.
HomeAddressLine2	String	Second line of the contact's home address.
HomeAddressLine3	String	Third line of the contact's home address.
HomeCity	String	City for the contact's home address.
HomeCountry	Identity	Identifier of the country for the contact's home address.
Homefax	String	Contact's home fax number.
Homephone	String	Home phone number for the contact.
Homephone2	String	Alternate home phone number for the contact.
HomePostal	String	Postal code or zip code for the contact's home address.
HomeProvince	Identity	Identifier of the province or state where the contact's home is located.
HomeWorkLocationGroup	Identity	Identifier of the work location group associated to the contact's home address.

Property (*=required)	Type	Description
HomeWorkLocation	Identity	Identifier of the work location associated to the contact's home address.
IgnoreSomeMetaData Checks	Boolean	The purpose of this flag is to disable most of the metadata checks when adding or updating records. This is useful if the objective is just to load records and then clean them up later. If this flag is set to true then only those fields where the metadata is set to mandatory will be checked and only to ensure they exist. All other checks for these fields will be ignored.
*LastName	String	Last name of the contact.
ManagerName	String	Name of the contact's manager.
ManagerPhone	String	Phone number of the contact's manager.
MiddleName	String	Middle name of the contact.
mVersion	String	Read-only. Current version number for internal version control.
MobilePhone	String	Contact's mobile/cell phone number.
Name	String	Contact name
NickName	String	Name by which the contact prefers to be addressed.
Office	String	Name of the office in which the contact works.
OtherAddress	String	Other address

Property (*=required)	Type	Description
OtherAddressLine2	String	Second line of other address
OtherAddressLine3	String	Third line of other address
OtherCity	String	Other city
OtherCountry	Identity	Identifer of the other country
OtherFax	String	Other fax
OtherPhone	String	Other phone
OtherPostal	String	Postal or zip code for other address
OtherProvince	Identity	Identifier of the other province or state
OtherWorkLocationGroup	Identity	Identifier of the work location group for the other address
OtherWorkLocation	Identity	Identifier of the work location for the other address
Pager	String	Pager number
Prefix	String	Courtesy title of the contact, for example, Mr., Ms., Dr., etc.
PrimaryPhone	String	Primary phone number
Profession	String	Contact's profession.
Reference	Boolean	Indicates whether or not the contact is willing to serve as a reference.
SpouseName	String	Name of the contact's spouse.
Suffix	String	Educational or professional designations, such as MBA or Ph.D.
Title	String	Contact's job title.

Property (*=required)	Type	Description
UpdatedBy	Signature	Updater of the record
WebAddress	String	Web Address

Related information

"ApiContact XML" on page 50

ApiContact XML

```
<root>
   <contact>
      <bypassmetadatacheck>checkall</bypassmetadatacheck>
      <createdbyid />
      <updatedbyid />
      <contactid />
      <name />
      <firstname />
      <lastname />
      <middlename />
      <nickname />
      <customer>
         <id />
         <name />
         <alternatename />
         <userdefinedid />
      </customer>
      <anniversary/>
      <assistantname />
      <assistantphone />
      <availabletoall>true</availabletoall>
      <birthday/>
      <businessaddress />
      <businessaddressline2 />
      <businessaddressline3 />
      <businesscity />
      <businesscountry>
         <id />
         <name />
         <alternatename />
      </businesscountry>
      <businessfax />
      <businessphone />
      <businessphone2 />
      <businesspostal />
```

```
<businessprovince>
   <id />
   <name />
   <alternatename />
</businessprovince>
<businessworklocationgroup>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</businessworklocationgroup>
<businessworklocation>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</businessworklocation>
<carphone />
<contactcode1>
   <id />
   <name />
   <alternatename />
</contactcode1>
<contactcode2>
   <id />
   <name />
   <alternatename />
</contactcode2>
<contactcode3>
   <id />
   <name />
   <alternatename />
</contactcode3>
<contacttext1 />
<contacttext2 />
<contacttext3 />
<contacttype>
   <id />
   <name />
   <alternatename />
</contacttype>
<department />
<description />
<email1 />
<email2 />
<email3 />
<history />
<historyupdates />
```

```
<homeaddress />
<homeaddressline2 />
<homeaddressline3 />
<homecity />
<homecountry>
   <id />
   <name />
   <alternatename />
</homecountry>
<homefax />
<homephone />
<homephone2 />
<homepostal />
<homeprovince>
   <id />
   <name />
   <alternatename />
</homeprovince>
<homeworklocationgroup>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</homeworklocationgroup>
<homeworklocation>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</homeworklocation>
<managername />
<managerphone />
<mobilephone />
<office />
<otheraddress />
<otheraddressline2 />
<otheraddressline3 />
<othercity />
<othercountry>
   <id />
   <name />
   <alternatename />
</othercountry>
<otherfax />
<otherphone />
<otherpostal />
<otherprovince>
   <id />
```

```
            <name />
            <alternatename />
         </otherprovince>
         <otherworklocationgroup>
            <id />
            <name />
            <alternatename />
            <userdefinedid />
         </otherworklocationgroup>
         <otherworklocation>
            <id />
            <name />
            <alternatename />
            <userdefinedid />
         </otherworklocation>
         <pager />
         <prefix />
         <primaryphone />
         <profession />
         <reference />
         <spousename />
         <suffix />
         <title />
         <webaddress />
      </contact>
   </root>
```

Comments

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

Example

Not available

Related information

"ApiContact" on page 43

ApiContact: AddContact

```
Public Function AddContact(ByVal mContact As ApiContact, Optional ByRef
sContactId As String = "") As Int32
```

Purpose

Adds a new contact to a customer in Changepoint.

Parameters

Parameter (*=required)	Description
*mContact	The populated contact object.
sContactId	Returns the ContactId, if successful.

Returns

0 = Success

Non-zero = Error

Remarks

Use the GetList function of the ApiCustomer object to find the customer ID to which to attach the contact.

The error returns are documented in the log file.

Example

```
Dim myContact As New ChangepointAPI2.ApiContact
Dim mContact As New ChangepointAPI2.ApiContact
Dim iRet As Int32
Dim sContactId As String

myContact.CPConnection = myCon

With mContact
   .Reference = False
   .History = "History"
   .Customer.Id = "{D0674B9C-BBF7-4BB9-8D38-689EB1DC2BDC}"
   .prefix = "Mr."
   .FirstName = "Jane"
   .LastName = "Smith"
   .MiddleName = "Rose"
   .ContactType.Id = "{9146E6A3-7A3D-11D2-80ED-0060975AEC0F}"
   .Description = "Main contact"
   .Title = "Title"
   .Department = "0"
```

```
.Office = "Office"
.Profession = "Profession"
.ManagerName = "ManagerName__random__"
.AssistantName = "Assistant Name"
.SpouseName = "Spouse Name"
.Birthday = "1980/10/27"
.Anniversary = "2006/1/1"
.BusinessAddress = "Business Address"
.BusinessAddressLine2 = "Business Address Line2"
.BusinessAddressLine3 = "Business Address Line3"
.BusinessCity = "Business City"
.BusinessProvince.Id = "{04B6E8AE-79C6-4402-87B2-1BBC5D8914CB}"
.BusinessPostal = "Business Postal"
.HomeAddress = "Home Address"
.HomeAddressLine2 = "Home Address Line2"
.HomeAddressLine3 = "Home Address Line3"
.HomeCity = "Home City"
.HomeProvince.Id = "{00000000-0000-0000-0000-000000000000}"
.HomePostal = "Home Postal"
.IgnoreSomeMetaDataChecks = False
.OtherAddress = "OtherAddress"
.OtherAddressLine2 = "Other Address Line2"
.OtherAddressLine3 = "Other Address Line3"
.OtherCity = "Other City"
.OtherProvince.Id = "{00000000-0000-0000-0000-000000000000}"
.OtherCountry.Id = "CAN"
.OtherPostal = "Other Postal"
.AssistantPhone = "Assistant Phone"
.BusinessPhone = "Business Phone"
.BusinessPhone2 = "Business Phone2"
.BusinessFax = "Business Fax"
.CarPhone = "Car Phone"
.HomePhone = "Home Phone"
.HomePhone2 = "Home Phone2"
.HomeFax = "Home Fax"
.ManagerPhone = "Manager Phone"
.MobilePhone = "Mobile Phone"
.OtherPhone = "Other Phone"
.OtherFax = "Other Fax"
.Pager = "123"
.PrimaryPhone = "321123123"
.Email1 = "Email1"
.Email2 = "Email2"
.Email3 = "Email3"
.WebAddress = "Web Address"
.HomeWorkLocationGroup.Id = "{00000000-0000-0000-0000-000000000000}"
.HomeWorkLocation.Id = "{00000000-0000-0000-0000-000000000000}"
.BusinessWorkLocationGroup.Id = "{6AAF1A2D-2535-40E3-8D03-07801AE85E41}"
```

```
   .BusinessWorkLocation.Id = "{0175FA25-7CAA-4DC3-BB95-62A37A36134F}"
   .OtherWorkLocationGroup.Id = "{00000000-0000-0000-0000-000000000000}"
   .OtherWorkLocation.Id = "{00000000-0000-0000-0000-000000000000}"
   .NickName = "Nick Name"
   .AvailableToAll = True
   .ContactExists = False
   .ContactCode1.Id = "{AF4F4AEF-36F4-11D4-B418-0050DA7707C1}"
   .ContactCode2.Id = "{AF4F4AF1-36F4-11D4-B418-0050DA7707C1}"
   .ContactCode3.Id = "{AF4F4AF7-36F4-11D4-B418-0050DA7707C1}"
   .ContactText1 = "Contact Text1"
   .ContactText2 = "Contact Text2"
   .ContactText3 = "Contact Text3"
End With

iRet = myContact.AddContact(mContact, sContactId)
```

Related information

"ApiContact" on page 43

ApiContact: CheckCustomerReference

```
Public Function CheckCustomerReference(ByVal sCustomerId As String) As Boolean
```

Purpose

Check whether this customer can be referenced or not

Parameters

Parameter (*=required)	Description
*sCustomerId	The contact ID to retrieve.

Returns

A boolean value to indicate if the customer is willing to serve as a reference.

Remarks

None

Example

Not available

Related information

"ApiContact" on page 43

ApiContact: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, Optional ByRef sId As String
= "")
```

Purpose

Creates a contact using an XML string of the Contact object in Changepoint.

Parameters

Parameter (*=required)	Description
*sXML	A new contact XML string which is passed based on the contact XML schema
ByRef	ByRef parameter that will return the new ContactId after the contact has been added.

Returns

0 = Success

Nonzero = Error

Remarks

The ApiContact XML structure can be obtained by the GetXMLStructure method. The ByPassMetadataCheck switch stops any metadata validation in ApiContact.

Example

Not available

Related information

"ApiContact" on page 43

"ApiContact XML" on page 50

"ApiContact: GetXMLStructure" on page 62

ApiContact: DeleteContact

```
Public Function DeleteContact(ByVal contactId As String) As Int32
```

Purpose

Deletes a contact in Changepoint.

Parameters

Parameter (*=required)	Description
*ContactId	The Contact ID

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

Not available

Related information

"ApiContact" on page 43

ApiContact: Exists

```
Public Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Check if the contact exists.

Parameters

Parameter (*=required)	Description
sId	ID of the contact

Returns

True if the contact exists, else False.

Remarks

If sId is not provided, the value of the property ContactId is used.

Example

```
Not available
```

Related information

"ApiContact" on page 43

ApiContact: GetByXML

```
Public Function GetByXML(Optional ByVal sXML As String = "", Optional ByVal
sContactId As String = "") As String
```

Purpose

Takes the XML string passed in the sXML parameter and returns the string filled with data for the Contact ID specified in the sContactId parameter.

Parameters

Parameter (*=required)	Description
sXML	XML string of the Contact object, with the fields specified
sContactId	Contact ID • If no Contact ID is passed in, the XML string (sXML) is examined • if there is no Contact ID in sXML then the object's Contact ID is selected. • If there still is no valid Contact ID, the method returns an error.

Returns

An XML string mirroring sXML with data inserted, or the entire XML of the Contact object including data.

Remarks

If sXML = "" then the XML string provided by GetXMLStructure is used.

Example

Not available

Related information

"ApiContact" on page 43

"ApiContact XML" on page 50

ApiContact: GetContact

```
Public Function GetContact(ByVal sContactId As String) As ApiContact
```

Purpose

Retrieve a contact by contact ID

Parameters

Parameter (*=required)	Description
*sContactId	The contact ID to retrieve.

Returns

An ApiContact object.

Remarks

None

Example

Not available

Related information

"ApiContact" on page 43

ApiContact: GetContacts

```
Public Function GetContacts(ByVal sCustomerId As String) As DataSet
```

Purpose

Retrieve contact records for a customer

Parameters

Parameter (*=required)	Description
*sCustomerId	The customer ID to retrieve the contacts for.

Returns

A dataset of contact information for a particular customer.

Remarks

The fields returned are FirstName, MiddleName, LastName, Name, and ContactId.

Example

Not available

Related information

"ApiContact" on page 43

ApiContact: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As String
```

Purpose

Returns the ContactId based on the UDF Text field and value.

Parameters

Parameter (*=required)	Description
*sUDFField	UDF text field name (for example, Text1)
*sUDFValue	UDF text value

Returns

ContactId, or an empty string if nothing is found.

Remarks

None

Related information

"ApiContact" on page 43

ApiContact: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Return the XML structure of the ApiContact object

Parameters

None

Returns

An XML string of the ApiContact object

Remarks

None

Example

Not available

Related information

"ApiContact" on page 43

"ApiContact XML" on page 50

ApiContact: SetPropertiesByXML

```
Public Function SetPropertiesByXML(ByVal sXML As String, Optional ByVal
bNotInitialize As Boolean = True) As Int32
```

Purpose

Set the properties of the object via an XML string.

Parameters

Parameter (*=required)	Description
*sXML	XML string of the Contact object with data to load into the object.
bNotInitialize	Switch that initializes the object. Normal use is to set as True.

Returns

0 = Success

Nonzero = Error

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods. It is recommended to use the GetXMLStructure method to get the correct XML structure.

Example

Not Available

Related information

"ApiContact" on page 43

"ApiContact XML" on page 50

"ApiContact: GetXMLStructure" on page 62

"ApiContact: CreateByXML" on page 57

"ApiContact: UpdateByXML" on page 64

ApiContact: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, Optional ByVal sContactId As
String = "") As Int32
```

Purpose

Update the database using an XML string of the Contact object containing update information.

Parameters

Parameter (*=required)	Description
*sXML	A contact XML string with updated fields.
sContactId	Contact ID of the contact being updated.

Returns

0 = Success

Nonzero = Error

Remarks

The method uses the following sequence to find the contact ID:

1. If the sContactId parameter is passed in, the method uses this value for the contact ID.
2. If this fails, the method attempts to extract the contact ID from <contactid> in the XML.
3. If this fails, the contact ID is taken from the object properties.
4. If this fails, an attempt is made to look up the contact ID using <email1> in the XML. If <email1> has a value, but the value is invalid or duplicated, then you will get an error and no further attempts are made to look up the contact ID.
5. If <email1> is empty, an attempt is made to look up the contact ID using <name> in the XML.

Example

Not available

Related information

"ApiContact" on page 43

"ApiContact XML" on page 50

ApiContact: UpdateContact

```
Public Function UpdateContact(ByVal mContact As ApiContact) As Int32
```

Purpose

Updates a contact in Changepoint.

Parameters

Parameter (*=required)	Description
*mContact	The populated contact object.

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

```
Dim myContact As New ChangepointAPI2.ApiContact
Dim mContact As New ChangepointAPI2.ApiContact
Dim iRet As Int32

myContact.CPConnection = myCon

mContact = myContact.GetContact("{7C2EC28B-4066-11D4-A17D-00010228881E}")

With mContact
   .Title = "Account Manager"
   .ManagerName = "John Smith"
   .BusinessPhone = "416-123-4567"
End With

myContact.CPConnection = myCon

iRet = myContact.UpdateContact(mContact)
```

Related information

"ApiContact" on page 43

ApiCreditNote

The ApiCreditNote object allows users to retrieve credit notes in the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiCreditNote

Methods

Properties

Property (*=required)	Type	Description
BillingOfficeId	String	Identifier for the Billing office associated with the credit note.
*CPConnection	ApiConnection	Connection to the Changepoint database
CreditAmount	Decimal	Credit amount
CreditNoteDate	Date	Date the credit note was created.
CreditNoteId	String	Identifier for the credit note.
CreditTax1	Decimal	Tax 1 credit amount
CreditTax2	Decimal	Tax 2 credit amount
Description	String	Description of the credit note.
EngagementId	String	Identifier for the contract associated with the credit note.
InvoiceId	String	Identifier for the Invoice associated with the credit note.
Reason	String	Reason for credit note.

ApiCreditNote: GetCreditNote

```
Public Function GetCreditNote(ByVal sCreditNoteNumber As String) As
ApiCreditNote
```

Purpose

Retrieve a credit note record by credit note number

Parameters

Parameter (*=required)	Description
*sCreditNoteNumber	The credit note number.

Returns

An ApiCreditNote object

Remarks

The CreditNoteId is looked up and then it calls GetCreditNoteById to return the ApiCreditNote object.

Example

Not available

Related information

"ApiCreditNote" on page 66

ApiCreditNote: GetCreditNoteById

```
Public Function GetCreditNoteById(ByVal sCreditNoteId As String) As
ApiCreditNote
```

Purpose

Retrieve a credit note record by Credit Note ID

Parameters

Parameter (*=required)	Description
*sCreditNoteId	Credit Note ID.

Returns

A single CreditNote object.

Remarks

None

Example

Not available

Related information

"ApiCreditNote" on page 66

ApiCreditNote: GetCreditNotes

```
Public Function GetCreditNotes(ByVal Status As ApiInvoice.CPInvoiceStatus) As
DataSet
```

Purpose

Retrieve credit note records by invoice status

Parameters

Parameter (*=required)	Description
*Status	Status of the invoice

Returns

A dataset of credit note information. The fields returned by the dataset are: CreditNoteId, InvoiceId, InvoiceNumber (from DisplayInvoiceId), and CreditNote number (from DisplayInvoiceId followed by 'CN', for example, "00005CN").

Remarks

The following statuses apply to the credit note:

- cpInvDraft
- cpInvPending_Approval
- cpInvPending_SecondLevel_Approval
- cpInvApproved
- cpInvCommited
- cpInvSent
- cpInvPartialPaid
- cpInvPaid

Example

```
Dim myCreditNote As New ChangepointAPI2.ApiCreditNote
Dim oRet As DataSet
myCreditNote.CPConnection = myCon
oRet = myCreditNote.GetCreditNotes(ApiInvoice.CPInvoiceStatus.cpInvDraft)
```

Related information

"ApiCreditNote" on page 66

ApiCustomer

The ApiCustomer object allows users to create, retrieve and update customers in the Changepoint database. Used with the ApiCustomer object, the ApiCustomerAddress object is also exposed by the ChangepointAPI2

Namespace

Changepoint.ChangepointAPI2.ApiCustomer

Methods

Properties

Property (*=required)	Type	Description
AccountManager	Identity	Identifier of the account manager associated with the customer.
*AccountType	Identity	Identifier of the account type associated with the customer.
AllowEngagement	Boolean	Flag determines if contracts are allowed to be created for the customer.
AllowReportCard	Boolean	Boolean value to determine if the customer Report Card report can be run for this customer.
AlternateName	String	Alternate customer name. May be used to the store name in another language.
AnnualRevenue	Identity	Identifier of the annual revenue associated with the customer.
AvailableToAll	Boolean	Allows everyone to view the customer.
BillingOffices	Generic.List(Of Identity)	List of the identifiers of the billing offices available to this customer.
BusinessAddress1	ApiCustomerAddress	First line of the business address
BusinessAddress2	ApiCustomerAddress	Second line of the business address
BusinessAddress3	ApiCustomerAddress	Third line of the business address

Property (*=required)	Type	Description
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
Campaign	Identity	Identifier of the campaign associated with the customer.
*CPConnection	ApiConnection	Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CreatedBy	Signature	Creator of the record.
CustomerId	String	Changepoint ID of the customer. The customer object generates the customerid if a create function (Add or CreateByXml) is called. Mandatory only if an update function (Update or UpdateByXml) is called.
CustomerServicePhone	String	Customer Service Phone
CustomerStatus	Identity	Identifier of the customer status. The identifier is code from CustomerStatus table.

Property (*=required)	Type	Description
Deleted	Boolean	Flag that indicates if the object has been deleted.
Description	String	Description of the customer.
Entity	CPEntity	The Changepoint entity that the object represents.
Employees	Identity	Employees that exist in Employees table.
ExecutiveOfficePhone	String	Executive office phone
FaxPhone	String	Fax phone
History	String	A string containing customer history entries for this customer.
HistoryUpdates	String	Comments that can be added to the customer history
HRPhone	String	HR Phone
IndustryType	Identity	Identifier of the industry type associated with the customer.
MainPhone	String	Main phone
MarketingPhone	String	Marketing phone
mVersion	String	Read-only. The current version number for internal version control.
*Name	String	Customer name (maximum 200 characters)
OtherPhone	String	Other phone
ParentCustomer	Identity	Identifier of the parent customer if it exists.

Property (*=required)	Type	Description
Reference	Boolean	A boolean value to determine if the customer is willing to serve as a reference
SalesPhone	String	Sales Phone
SalesRegion	Identity	Identifier of the sales region associated with the customer.
SalesRep	Identity	Identifier of the sales representative associated with the customer
*SalesStatus	Identity	Identifier of the sales status of the customer.
SANSurveyCustomer	Boolean	Boolean value to determine if the customer can receive surveys
SecurityPhone	String	Security phone
sxmlUDF	String	UDF XML string. Includes all information of UDF code and UDF text
TaxExempt	Boolean	Indicates whether or not this customer is tax exempt when billing
TaxExemptId	String	Tax Exemption ID
TechnicalSupportPhone	String	Technical support phone
TeleMarketer	Identity	Identifier of the telemarketer responsible for customer
TollfreePhone	String	Toll free phone
UpdatedBy	Signature	The updater of the record and when it was updated.
UserDefinedCustomerId	String	User defined Customer ID

Property (*=required)	Type	Description
VATRegNumber	String	VAT registration number
WebSite	String	Web site URL including protocol, for example, http://www.changepoint.com

Related information

"ApiCustomer XML" on page 75

"ApiCustomerAddress" on page 96

ApiCustomer XML

```
<root>
   <customer>
      <bypassmetadatacheck>checkall</bypassmetadatacheck>
      <createdbyid />
      <updatedbyid />
      <customerid/>
      <name/>
      <alternatename/>
      <description/>
      <userdefinedcustomerid/>
      <industrytype>
         <id/>
         <name/>
         <alternatename/>
      </industrytype>
      <annualrevenue>
         <id/>
         <name/>
         <alternatename/>
      </annualrevenue>
      <employees>
         <id/>
         <name/>
         <alternatename/>
      </employees>
      <accountmanager>
         <id/>
         <name/>
         <alternatename/>
         <firstname />
         <lastname />
         <userdefinedid />
```

```
</accountmanager>
<salesrep>
   <id/>
   <name/>
   <alternatename/>
   <firstname />
   <lastname />
   <userdefinedid />
</salesrep>
<accounttype>
   <id/>
   <name/>
   <alternatename/>
</accounttype>
<customerstatus>
   <id/>
   <name/>
   <alternatename/>
</customerstatus>
<salesstatus>
   <id/>
   <name/>
   <alternatename/>
</salesstatus>
<salesregion>
   <id/>
   <name/>
   <alternatename/>
</salesregion>
<campaign>
   <id/>
   <name/>
   <alternatename/>
</campaign>
<telemarketer>
   <id/>
   <name/>
   <alternatename/>
   <firstname />
   <lastname />
   <userdefinedid />
</telemarketer>
<parentcustomer>
   <id/>
   <name/>
   <alternatename/>
</parentcustomer>
<userdefinedid />
```

```
      <taxexempt/>
      <taxexemptid/>
      <allowengagement>true</allowengagement>
      <availabletoall>true</availabletoall>
      <reference>false</reference>
      <allowreportcard>false</allowreportcard>
      <sansurveycustomer>false</sansurveycustomer>
      <customerservicephone/>
      <executiveofficephone/>
      <faxphone/>
      <hrphone/>
      <mainphone/>
      <marketingphone/>
      <otherphone/>
      <salesphone/>
      <securityphone/>
      <technicalsupportphone/>
      <tollfreephone/>
      <website/>
      <vatregnumber/>
      <udf/>
      <customeraddresses>
      ...
      </customeraddresses>
   </customer>
</root>
```

Comments

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

The XML for the ApiCustomer object includes the XML for the ApiCustomerAddress object and may contain XML for UDFs (configurable fields). For more information on the XML details for included objects, see the documentation for the included objects.

Example

Not available

Related information

"ApiCustomer" on page 70

"ApiCustomerAddress XML" on page 97

"UDF XML" on page 742

ApiCustomer: Add

```
Public Overrides Function Add(Optional ByRef sId As String = "") As Integer
```

Purpose

Add a new customer to Changepoint database.

Parameters

Parameter (*=required)	Description
sId	CustomerId

Returns

0 = Success

Nonzero = Error

Remarks

Ensure all mandatory properties are set.

Example

```
Dim myCus as New Customer
Dim sId As String
With myCus
   .Name = "Changepoint Canada"
   .AccountManager.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
   .AccountType.Id = "{9146E6A9-7A3D-11D2-80ED-0060975AEC0F}"
   .AllowEngagement = True
   .CustomerStatus.Id = "{9146E69E-7A3D-11D2-80ED-0060975AEC0F}"
   .SalesStatus.Id = "C"
   .SalesRep.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
   .BusinessAddress1.AddressLine = "30 Leek Crescent, Suite 400"
   .BusinessAddress1.City = "Richmond Hill"
   .BusinessAddress1.StateProvince.Id = "{27044F87-175F-4F45-A8AD-
12750600EE49}"
   .BusinessAddress1.Country.Id = "CAN"
   .BusinessAddress1.PostalCode = "L4B 4N4"
   .BillingOffices.Add(New Identity("{5757A330-3476-11D3-807A-00105A0B7C01}"))
   ...
   iRet = .Add(sId)
End With
```

Related information

"ApiCustomer" on page 70

ApiCustomer: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, Optional ByRef sId As String
= "") As Int32
```

Purpose

Create a customer using an XML string of the Customer object in Changepoint.

Parameters

Parameter (*=required)	Description
*sXML	A new customer XML string which can be passed based on the Customer xml schema
sId	ByRef parameter that will return the new CustomerId after the customer has been added.

Returns

0 = Success

Nonzero = Error

Remarks

The ApiCustomer XML structure can be obtained by the GetXMLStructure method.

The ByPassMetadataCheck switch will stop any metadata validation in Customer and also in Customer UDFs.

For details of the use of UDF default values, see "UDF default values logic for CreateByXML" on page 746.

Example

Not available

Related information

"ApiCustomer" on page 70

"ApiCustomer XML" on page 75

"ApiCustomer: GetXMLStructure" on page 91

ApiCustomer: Delete

```
Public Overrides Function Delete(Optional ByVal sId As String = "") As Integer
```

Purpose

Deletes the Customer from Changepoint database.

Parameters

Parameter (*=required)	Description
sId	ID of the customer to be deleted.

Returns

0 = Success

Nonzero = Error

Remarks

Customers with active contracts will not be deleted.

If sId is not provided, the value of the property CustomerId is used.

Example

```
Dim myCus as New ApiCustomer()
myCus.CPConnection = myCon
Dim retId as String = String.Empty
Dim ret as Int32= myCus.Delete(retId)
```

Related information

"ApiCustomer" on page 70

ApiCustomer: Exists

```
Public Overrides Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Check if the customer exists.

Parameters

Parameter (*=required)	Description
sId	ID of the customer

Returns

True if the customer exists, else False.

Remarks

If sId is not provided, the value of the property CustomerId is used.

Example

```
Dim myCus as New ApiCustomer()
myCus.CPConnection = myCon
Dim ret as Boolean = myCus.Exists("{CA8AC6E5-5F9A-41CA-BD57-9A35D28A7E76}")
```

Related information

"ApiCustomer" on page 70

ApiCustomer: GetById

```
Public Overrides Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Fills the object with customer information of the specified sId passed in the parameter or of the property CustomerId.

Parameters

Parameter (*=required)	Description
sId	Id of customer that will be retrieved.

Returns

0 = Success

Nonzero = Error

Remarks

If optional parameter sId is not provided, the property CustomerId is used.

Example

```
Dim myCus as New ApiCustomer()
Dim iRet As Int32
myCus.CPConnection = myCon
iRet = myCus.GetById("{CA8AC6E5-5F9A-41CA-BD57-9A35D28A7E76}")
```

Related information

"ApiCustomer" on page 70

ApiCustomer: GetByUserDefinedId

```
Public Function GetByUserDefinedId(ByVal userDefinedCustumerId As String,
Optional ByRef retRows As Int32 = 0) As DataSet
```

Purpose

Fills the object with customer information of the specified user defined customer ID.

Parameters

Parameter (*=required)	Description
*userDefinedCustumerId	User Defined Customer ID.
retRows	The number of rows returned by the method.

Returns

A list of customers in a dataset upon success and nothing in a dataset upon error.

Remarks

The object is filled with the first customer from the list.

Returns all rows if parameter retRows is not provided.

Returns the following columns: CustomerId, Name

Example

```
Dim myCus as New ApiCustomer()
Dim dNumOfRowsReturned As Int32
Dim dsRet As DataSet
myCus.CPConnection = myCon
dsRet = myCus.GetByUserDefinedId("Cust-2008-0001",dNumOfRowsReturned)
```

Related information

"ApiCustomer" on page 70

ApiCustomer: GetByXML

```
Public Function GetByXML(Optional ByVal sXML As String = "", Optional ByVal
sCustomerId As String = "") As String
```

Purpose

Takes the XML string passed in the sXML parameter and returns the string filled with data for the Customer ID specified in the sCustomerId parameter.

Parameters

Parameter (*=required)	Description
sXML	XML string of the Customer object, with the fields specified
sCustomerId	Customer ID • If no Customer ID is passed in, the XML string (sXML) is examined • if there is no Customer ID in sXML then the object's Customer ID is selected. • If there still is no valid CustomerId, the method returns an error.

Returns

An XML string mirroring sXML with data inserted, or the entire XML of the Customer object including data.

Remarks

If sXML = "" then the XML string provided by GetXMLStructure is used.

Example

Not available

Related information

"ApiCustomer" on page 70

"ApiCustomer XML" on page 75

ApiCustomer: GetCampaigns

```
Public Function GetCampaigns() As DataSet
```

Purpose

Retrieve a list of campaigns.

Parameters

None

Returns

A list of campaigns in a dataset upon success and nothing in a dataset upon error.

Remarks

Returns the following columns: CampaignId, Name

Example

```
Dim myCus as New ApiCustomer()
Dim dsRet As DataSet
myCus.CPConnection = myCon
dsRet = myCus.GetCampaigns()
```

Related information

"ApiCustomer" on page 70

ApiCustomer: GetContactsByCustomerId

```
Public Function GetContactsByCustomerId(ByVal sCustomerId As String) As
DataSet
```

Purpose

Retrieve a list of contacts by CustomerId.

Parameters

Parameter (*=required)	Description
*sCustomerId	ID of Customer

Returns

A list of contacts in a dataset upon success and nothing in a dataset upon error.

Remarks

Returns the following columns: ContactId, Name

Example

```
Dim myCus as New ApiCustomer()
Dim dsRet As DataSet
myCus.CPConnection = myCon
```

```
dsRet = myCus.GetContactsByCustomerId("{CA8AC6E5-5F9A-41CA-BD57-
9A35D28A7E76}")
```

Related information

"ApiCustomer" on page 70

ApiCustomer: GetContactsbyCustomerName

```
Public Function GetContactsbyCustomerName(ByVal customerName As String,
Optional ByVal contactFirstName As String = "", Optional ByVal contactLastName
As String = "") As DataSet
```

Purpose

Retrieve a list of contacts by filters provided by parameters.

Parameters

Parameter (*=required)	Description
*customerName	Name of Customer
contactFirstName	First name of contact
contactLastName	Last name of contact

Returns

A list of contacts in a dataset upon success and nothing in a dataset upon error.

Remarks

Returns the following columns: ContactId, Name

Example

```
Dim myCus as New ApiCustomer()
Dim dsRet As DataSet
myCus.CPConnection = myCon
dsRet = myCus.GetContactsbyCustomerName("Changepoint Corporation")
```

Related information

"ApiCustomer" on page 70

ApiCustomer: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As String
```

Purpose

Returns the CustomerId based on the UDF Text field and value.

Parameters

Parameter (*=required)	Description
*sUDFField	UDF text field name (for example, Text1)
*sUDFValue	UDF text value

Returns

CustomerId, or an empty string if nothing is found.

Remarks

None

Related information

"ApiCustomer" on page 70

ApiCustomer: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows As Int16 = -1) As
DataSet
```

Purpose

Retrieve a list of customers.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).

Returns

Customer records in a dataset upon success and nothing in a dataset upon error.

Returns all rows with no parameter.

Remarks

Returns the following columns:

CustomerId, CustomerName, UserDefinedCustomerId

Example

```
Dim myCus as New ApiCustomer()
Dim dsRet As DataSet
myCus.CPConnection = myCon
dsRet = myCus.GetList()
```

Related information

"ApiCustomer" on page 70

ApiCustomer: GetResourcesByUserDefinedId

```
Public Function GetResourcesByUserDefinedId(ByVal sUserDefinedId As String) As
DataSet
```

Purpose

Retrieve a list of customer resources for a specified user-defined ID.

Parameters

Parameter (*=required)	Description
sUserDefinedId	User-defined ID for which to look up resources

Returns

Dataset with 2 columns (ResourceId, Name) upon success and nothing in a dataset upon error.

Remarks

None

Example

Not available

Related information

"ApiCustomer" on page 70

ApiCustomer: GetUDF

```
Public Function GetUDF(Optional ByVal retOption As CPUDFReturnType =
CPUDFReturnType.OnlyValues, Optional ByVal actionResourceId As String = "") As
String
```

Purpose

Retrieve UDF (configurable field) information for customer

Parameters

Parameter (*=required)	Description
retOption	The CPUDFReturnType Values are: • WithStructure = 0 – returns UDF field data with full field structure • OnlyValues = 1 – returns only UDF fields with data, do not include any field structure or options • OnlyStructure = 2 – returns only UDF XML structure, no field data • OnlyStructureAndOptions = 3 – returns UDF structure and option data without field data • WithStructureAndOptions = 4 – returns UDF field data with full structure and all field options (except entity based and conditional codes)
actionResourceId	The user ID of the user that called the method. Default is the login user id.

Returns

An XML string of UDFs.

Remarks

If CustomerId is empty, the retOption is only available on OnlyStructure (2) and OnlyStructureAndOptions (3).

Example

```
Dim myCus as New ApiCustomer()
Dim sRet As String
myCus.CPConnection = myCon
myCus.CustomerId = "{CA8AC6E5-5F9A-41CA-BD57-9A35D28A7E76}"
sRet = myCus.GetUDF(CPUDFReturnType.WithStructureAndOptions)
```

Related information

"ApiCustomer" on page 70

ApiCustomer: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal codeName As String, Optional ByVal
searchString As String = "") As String
```

Purpose

Retrieve UDF code options for the specified customer UDF code.

Parameters

Parameter (*=required)	Description
*codeName	The name of the UDF code, for example, "Code3"
SearchString	Returns the options that start with this string.

Returns

UDF code options in XML format.

Remarks

If the customerId is not provided, returns default UDF values.

Example

```
Dim myCus as New ApiCustomer()
Dim sRet As String
myCus.CPConnection = myCon
```

```
myCus.CustomerId = "{CA8AC6E5-5F9A-41CA-BD57-9A35D28A7E76}"
sRet = myCus.GetUDFCodeOptions("Code2", "")
```

Related information

"ApiCustomer" on page 70

ApiCustomer: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Return the XML structure of the ApiCustomer object

Parameters

None

Returns

An XML string of the ApiCustomer object

Remarks

None

Example

```
Dim myCus as New ApiCustomer()
Dim sRet As String
myCus.CPConnection = myCon
sRet = myCus.GetXMLStructure()
```

Related information

"ApiCustomer" on page 70

"ApiCustomer XML" on page 75

ApiCustomer: SaveUDF

```
Public Function SaveUDF(Optional ByVal sUDF As String = "") As Int32
```

Purpose

Saves (Insert/Update) UDF data for a customer.

Parameters

Parameter (*=required)	Description
sUDF	UDF string in XML format.

Returns

0 = Success

Nonzero = Error

Remarks

The Customer.Id is mandatory field, so this property should be populated before calling SaveUDF method. If sUDF is not provided, take the value of property sxmlUDF.

It's recommended to call GetUDF method to obtain correct UDF XML format.

Example

```
Dim myCus as New ApiCustomer()
Dim iRet As Int32
Dim strXMLUDF as string ="<root></root>"
With myCus
   .CPConnection = myCon
   .CustomerId = "{CA8AC6E5-5F9A-41CA-BD57-9A35D28A7E76}"
   iRet = .SaveUDF(strXMLUDF)
End With
```

Related information

"ApiCustomer" on page 70

"ApiCustomer XML" on page 75

"ApiCustomer: GetXMLStructure" on page 91

"UDF XML" on page 742

ApiCustomer: SetPropertiesByXML

```
Public Function SetPropertiesByXML(ByVal sXML As String, Optional ByVal
bNotInitialize As Boolean = True) As Int32
```

Purpose

Set the properties of the object via an XML string.

Parameters

Parameter (*=required)	Description
*sXML	XML string of the Customer object with data to load into the object.
bNotInitialize	Switch that initializes the object. Normal use is to set as True.

Returns

0 = Success

Nonzero = Error

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods. It is recommended to use the GetXMLStructure method to get the correct XML structure.

Example

```
Dim myCus as New ApiCustomer()
Dim iRet As Int32
Dim strXmlCustomer as string ="<root></root>"
myCus.CPConnection = myCon
iRet = myCus.SetPropertiesByXML(strXmlCustomer, true)
If iRet = 0 Then
   iRet = myCus.Update()
End If
```

Related information

"ApiCustomer" on page 70

"ApiCustomer XML" on page 75

"ApiCustomer: GetXMLStructure" on page 91

"ApiCustomer: CreateByXML" on page 79

"ApiCustomer: UpdateByXML" on page 95

ApiCustomer: Update

```
Public Overrides Function Update() As Integer
```

Purpose

Update a customer record.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

It's recommended to populate object with the data to be updated by the GetById method.

Example

```
Dim myCus as New ApiCustomer()
Dim iRet As Int32
myCus.CPConnection = myCon
iRet = myCus.GetById("{CA8AC6E5-5F9A-41CA-BD57-9A35D28A7E76}")
With myCus
   .SalesStatus.Id = "C"
   .BusinessAddress1.AddressLine = "30 Leek Crescent, Suite 400"
   .BusinessAddress1.City = "Richmond Hill"
   .BusinessAddress1.StateProvince.Id = "{27044F87-175F-4F45-A8AD-
12750600EE49}"
   .BusinessAddress1.Country.Id = "CAN"
   .BusinessAddress1.PostalCode = "L4B 4N4"
   ...
   iRet = .Update()
End With
```

Related information

"ApiCustomer" on page 70

"ApiCustomer: GetById" on page 81

ApiCustomer: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, Optional ByVal sCustomerId
As String = "") As Int32
```

Purpose

Update the database using an XML string of the Customer object containing update information.

Parameters

Parameter (*=required)	Description
*sXML	A customer XML string with updated fields.
sCustomerId	Customer ID of the customer being updated.

Returns

0 = Success

Nonzero = Error

Remarks

The method uses the following sequence to find the customer ID:

1. If the sCustomerId parameter is passed in, the method uses this value for the customer ID.
2. If this fails, the method attempts to extract the customer ID from <customerid> in the XML.
3. If this fails, the customer ID is taken from the object properties.
4. If this fails, an attempt is made to look up the customer ID using <userdefinedcustomerid> in the XML. If <userdefinedcustomerid> has a value, but the value is invalid or duplicated, then you will get an error and no further attempts are made to look up the customer ID.
5. If <userdefinedcustomerid> is empty, an attempt is made to look up the customer ID using <name> in the XML.

Example

Not available

Related information

"ApiCustomer" on page 70

"ApiCustomer XML" on page 75

ApiCustomerAddress

The ApiCustomerAddress object contains address information for the customer.

Namespace

Changepoint.ChangepointAPI2.ApiCustomerAddress

Methods

None

Properties

Property (*=required)	Type	Description
AddressLine	String	First line of the address
AddressLine2	String	Second line of the address
AddressLine3	String	Third line of the address
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
City	String	City or town for the address
Country	Identity	Identifier of the country for the address
PostalCode	String	Zip or postal code for the address
StateProvince	Identity	Identifier of the state or province for the address

Property (*=required)	Type	Description
WorkLocation	Identity	Identifier of the work location
WorkLocationGroup	Identity	Identifier of the work location group

Related information

"ApiCustomer" on page 70

"ApiCustomerAddress XML" on page 97

ApiCustomerAddress XML

```
</root>
   </customer>
      <customeraddresses>
         <customeraddress id="1">
            <addressline />
            <addressline2 />
            <addressline3 />
            <city />
            <stateprovince>
               <id />
               <name />
               <alternatename />
            </stateprovince>
            <country>
               <id />
               <name />
               <alternatename />
            </country>
            <postalcode />
            <workLocation>
               <id />
               <name />
               <alternatename />
               <userdefinedid />
            </workLocation>
            <worklocationgroup>
               <id />
               <name />
               <alternatename />
               <userdefinedid />
            </worklocationgroup>
            <bypassmetadatacheck />
         </customeraddress>
```

```
            <customeraddress id="2">
            ...
            </customeraddress>
            <customeraddress id="3">
            ...
            </customeraddress>
         </addresses>
      </customer>
   </root>
```

Comments

The XML for the ApiCustomerAddress object is included in the XML for the ApiCustomer object. The following container elements are mandatory:

```
<root>
   <customer>
      <customeraddresses>
         <customeraddress id="1">
         ...
         </customeraddress>
      </customeraddresses>
   </customer>
</root>
```

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

Example

Not available

Related information

"ApiCustomer XML" on page 75

"ApiCustomerAddress" on page 96

ApiEngagement

The ApiEngagement object allows users to create, retrieve, update and delete contracts within the Changepoint database. Used with the ApiEngagement object, the following objects are also exposed by the Changepoint API:

- "ApiEngBillingRate" on page 145
- "ApiEngBillingRateHistory" on page 156

- "ApiEngFixedFee" on page 163
- "ApiEngFixedFeeItem" on page 175
- "ApiEngFixedFeeItemSplitBillOverride" on page 184
- "ApiEngFixedFeeSplitBillOverride" on page 173
- "ApiEngProduct" on page 186
- "ApiEngProductSplitBillOverride" on page 199
- "ApiEngProjectedResource" on page 200
- "ApiEngRequestProcessingRule" on page 209
- "ApiEngRequestSLA" on page 223
- "ApiEngRevRec" on page 233
- "ApiEngSplitBillingRule" on page 238
- "ApiEngWorkCode" on page 245
- "ApiEngWorkLocation" on page 250

Namespace

Changepoint.ChangepointAPI2.ApiEngagement

Methods

Properties

Property (*=required)	Type	Description
AdditionalContact	Identity	Identifier of the additional contact
AdditionalItemsAmountLimit	Decimal	Billing amount limit for additional items.
AdditionalItemsGLAId	Identity	Billing distribution GL account for additional items.
AllItemsAmountLimit	Decimal	Billing amount limit for all items.

Property (*=required)	Type	Description
AllLocations	Boolean	Read-only. Include all locations
AllowEditToAll	Boolean	Allows everyone to edit the contract
AllowPrjMgrBillCont	Boolean	Allow Project Manager to specify the Billing Contact
AllowProjectOverride	Boolean	Allow Project Manager to override the billable status
AllWorkCodes	Boolean	Read-only. Include all work codes
AlternateName	String	The Contract Alternate Name
AltExpenseApprovalDelegated	Boolean	Alternate expense approval delegated
AltExpenseApprover	Identity	Identifier of the alternate expense approver
AltTimeApprover	Identity	Identifier of the alternate time approver
AltTimeApproverDelegated	Boolean	Flag if Alternate Time Approver has been delegated
AssociatedEngagement	Identity	Identifier of the associated contract
AssociatedWorkGroup	Identity	Identifier of the associated work group
AuditDetail	String	The detail comments of audit
AuditEnabledDate	DateTime	The date Federal Audit option was enabled (if enabled).
AvailableToAll	Boolean	Allows everyone to view the contract.

Property (*=required)	Type	Description
*Billable	Boolean	Flag if the contract can be invoiced.
BillingAddress	Int32	Billing address: • 1 Customer Business Address1 • 2 Customer Business Address2 • 3 Customer Business Address3 • 4 Contact Business Address • 5 Contact Home Address • 6 Contact Other Address
*BillingCurrency	String	Currency code
*BillingOffice	Identity	Identifier of the billing office
*BillingType	String	Billing Type Code: • D – Daily • F – Fixed Fee • H – Hourly • MD – Mixed - Fixed Fee/Daily • MH – Mixed - Fixed Fee/Hourly
*BillToWorkLocation	Identity	Identifier of the work location
*BillToWorkLocationGroup	Identity	Identifier of the work location group

Property (*=required)	Type	Description
BypassMetadataCheck	CPMetadataCheck	Determines the level of MetaData checking in contract and in contract UDFs when adding or updating data. (default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
ContractAmount	Decimal	Contract amount
ContractEndDate	DateTime	Contract end date
ContractNumber	String	Contract number Maximum 256 characters
ContractStartDate	DateTime	Contract start date (effective date)
ContractType	String	Contract type ID
CostCenter	Identity	Identifier of the cost center
CreatedBy	Signature	Contains details about when the contract was created and by who.
*Customer	Identity	Identifier of the customer associated with the contract
DefaultBillingRole	Identity	Identifier of the default billing role
DelegateAltExpenseApprover	Identity	Identifier of the delegate alternate expense approver
DelegateAltTimeApprover	Identity	Identifier of the delegate alternate time approver

Property (*=required)	Type	Description
DelegateExpenseApprover	Identity	Identifier of the delegate expense approver
DelegateTimeApprover	Identity	Identifier of the delegate time approver
Deleted	Boolean	Flag that indicates whether the contract has been deleted.
Description	String	The contract description. Maximum of 2048 characters.
EnableAutoUpdateCost	Boolean	Enable automatic cost update for this contract.
EnableAutoUpdateRate	Boolean	Enable automatic rate update for this contract.
EnableBatchInvoice	Boolean	Enable Batch Invoicing
EnableProjectTimeApproval	Boolean	Allow the Project Manager to specify the time approver
*EngagementId	String	ID of the contract
*EngagementManager	Identity	Identifier of the contract manager
*EngagementStatus	String	Contract status lookup code
EngBillingRates	Generic. List(Of Identity)	List of billing rates
EngBillingRatesHistory	Generic. List(Of Identity)	List of billing rate history
EngFFISplitBillOverrides	Generic. List(Of ApiEngFixedFeeItemSplitBillOverride)	Collection of contract fixedfee item split bill overrides.

Property (*=required)	Type	Description
EngFFSplitBillOverrides	Generic. List(Of ApiEngFixedFeeSplitBillOverride)	Collection of contract fixedfee split bill overrides.
EngFixedFeeItems	Generic. List(Of Identity)	List of fixed fee items
EngFixedFees	Generic. List(Of Identity)	List of contract fixed fees
EngProducts	Generic. List(Of Identity)	List of contract products
EngProductSplitBillOverrides	Generic. List(Of ApiEngProductSplitBillOverride)	Collection of contract product split bill overrides.
EngProjectedResources	Generic. List(Of Identity)	List of projected resources
EngRequestProcessingRules	Generic. List(Of Identity)	List of request processing rules
EngRequestSLAs	Generic. List(Of Identity)	List of Request SLAs
EngRevRec	Object of ApiEngRevRec	Contract revenue recognition
EngSplitBillingRules	Generic. List(Of Identity)	List of split billing rules
EngWorkCode	Object of ApiEngWorkCode	Contract work code
EngWorkLocation	Object of ApiEngWorkLocation	Contract work location
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
ExpenseAmountLimit	Decimal	Billing amount limit for expenses.
ExpenseApprovalDelegated	Boolean	Allow expense approval delegated
ExpenseApprover	Identity	Identifier of the expense approver
*ExpenseBillingPercentage	Decimal	Expense billing percentage
*ExpenseBillingType	String	Expense billing type lookup code

Property (*=required)	Type	Description
*ExpenseGLA	String	Expense GLA account
ExpenseGLAId	Identity	Billing distribution GL account for expenses.
ExpenseMinimum	Decimal	Expense minimum
ExpenseQuantityLimit	Decimal	Billing limit for number of expenses.
FederalAudit	Boolean	Federal audit enabled
FixedFeeAmountLimit	Decimal	Billing amount limit for fixed fees.
ForceFixedFeeSelection	Boolean	Force fixed fee selection
GroupByProjectResourceAnd Rate	Boolean	Whether to group time entries by project and resource expense billing.
InternalContact	Identity	Identifier of internal contact for this contract
InvoiceComments	String	Default invoice comments
InvoiceEmailCc	String	Email 'Cc' recipients for batch invoicing.
InvoiceEmailComment	String	Email message for batch invoicing.
InvoiceEmailSubject	String	Email subject line for batch invoicing.
InvoiceEmailTo	String	Email 'To' recipients for batch invoicing.
*InvoiceFormat	Identity	Identifier of invoice format
InvoiceMaximum	Boolean	Invoice Maximum

Property (*=required)	Type	Description
LineOfBusiness	Identity	Identifier of the line of business associated with the contract
*MainContact	Identity	Identifier of main contact
MaximumAmount	Decimal	Maximum amount of a invoice
mVersion	String	Read-only. The current vresion number for internal version control.
*Name	String	The contract name. Max. 256 characters.
Opportunity	Identity	Read-only. Identifier of the contract's opportunity
OtherBillingInformation	String	Comment. Maximum of 2048 characters.
OverrideTaskRestriction	Boolean	Allow override task restriction
Overtime	Boolean	Overtime
OvertimePercentage	Decimal	Overtime Percentage
PaymentTerms	String	ID of the payment terms
PreventTaxChanges	Boolean	Prevent Tax Changes
ProductAmountLimit	Decimal	Billing amount limit for all items.
ProductQuantityLimit	Decimal	Billing limit for number of expenses.
RejectAdditionalItemsAmount	Boolean	Whether to reject a billing amount that exceeds the limit for additional items.
RejectAllItemsAmount	Boolean	Whether to reject a billing amount that exceeds the limit for all items.

Property (*=required)	Type	Description
RejectExpenseAmount	Boolean	Whether to reject a billing amount that exceeds the limit for expenses.
RejectExpenseQuantity	Boolean	Whether to reject a billing amount that exceeds the limit for the number of expenses.
RejectFixedFeeAmount	Boolean	Whether to reject a billing amount that exceeds the limit for fixed fees.
RejectProductAmount	Boolean	Whether to reject a billing amount that exceeds the limit for products.
RejectProductQuantity	Boolean	Whether to reject a billing amount that exceeds the limit for number of products.
RejectRequestTimeAmount	Boolean	Whether to reject a billing amount that exceeds the limit for number of units of request time.
RejectRequestTimeEffort	Boolean	Whether to reject a billing amount that exceeds the limit for number of units of time.
RejectTimeAmount	Boolean	Whether to reject a billing amount that exceeds the limit for resource time cost.
RejectTimeEffort	Boolean	Whether to reject a billing amount that exceeds the limit for all items.
RemitToAddress	Identity	Identifier of the remit to address
RemoveAllFixedFee	Boolean	Remove all fixed fees Internal use only. Do not set the value.

Property (*=required)	Type	Description
RemoveAllProduct	Boolean	Remove all contract products Internal use only. Do not set the value
Request	Identity	Read-only. Identifier of the contract's request.
RequestProductGLAId	Identity	Billing distribution GL account for products from requests.
RequestTimeAmountLimit	Decimal	Billing amount limit for request resource time.
RequestTimeCapital	Boolean	Request time capital
RequestTimeEffortLimit	Decimal	Billing limit for number of request time units.
RequestTimeGLAId	Identity	Billing distribution GL account for additional items.
ResourcesAllowedEdit	Array of Identity	List of identifiers of the resources who are allowed to edit the contract
ResourcesAllowedView	Array of Identity	List of the identifiers of the resources who are allowed to view the contract
SecondLevelApprover	Identity	Identifier of the second level approver.
SplitBilling	Boolean	Flag determines if split billing can be enabled for a billable contract.
SplitChangeDate	DateTime	Date split billing status was changed
*SupportDesk	Identity	Identifier of the support desk
SupportDeskOverride	Boolean	Allow support desk override

Property (*=required)	Type	Description
sxmlUDF	String	UDF XML string
TaxOnRecordDate	Boolean	If this flag is selected, the contract uses the record date to calculate taxes.
TaxOnSum	Boolean	Whether to calculate tax on total amount of items, or on each item separately.
Template	Boolean	Shows if the contract is a template
TimeAmountLimit	Decimal	Billing amount limit for resource time cost.
TimeApprovalDelegated	Boolean	Allow time approver to be delegated
TimeApprover	Identity	Identifier of the time approver
TimeEffortLimit	Decimal	Billing limit for number of time units.
TimeGLAId	Identity	Billing distribution GL account for additional items.
UpdatedBy	Signature	Details of when the record was updated and by whom.
UseBODailyConversion	Boolean	TRUE if billing office daily conversion is used - daily type contracts only. Set to FALSE for hourly or fixed fee type contracts.
UserDefinedEngagementId	String	Contract number
UseTwoLevelApproval	Boolean	Allow use two level approval
UseTwoLevelOnContractOverage	Boolean	Allow use two level on contract overage

Property (*=required)	Type	Description
UseTwoLevelOnWOffs	Boolean	Allow use two level on write-offs
WarnAdditionalItemsAmount	Boolean	Whether to warn about a billing amount that exceeds the limit for additional items.
WarnAllItemsAmount	Boolean	Whether to warn about a billing amount that exceeds the limit for all items.
WarnExpenseAmount	Boolean	Whether to warn about a billing amount that exceeds the limit for expenses.
WarnExpenseQuantity	Boolean	Whether to warn about a billing amount that exceeds the limit for the number of expenses.
WarnFixedFeeAmount	Boolean	Whether to warn about a billing amount that exceeds the limit for fixed fees.
WarnProductAmount	Boolean	Whether to warn about a billing amount that exceeds the limit for products.
WarnProductQuantity	Boolean	Whether to warn about a billing amount that exceeds the limit for number of products.
WarnRequestTimeAmount	Boolean	Whether to warn about a billing amount that exceeds the limit for number of units of request time.
WarnRequestTimeEffort	Boolean	Whether to warn about a billing amount that exceeds the limit for number of units of time.

Property (*=required)	Type	Description
WarnTimeAmount	Boolean	Whether to warn about a billing amount that exceeds the limit for resource time cost.
WarnTimeEffort	Boolean	Whether to warn about a billing amount that exceeds the limit for all items.
WorkPerformedFor	Identity	Identifier of the customer for which the work was performed

Related information

"ApiEngBillingRate" on page 145

"ApiEngBillingRateHistory" on page 156

"ApiEngFixedFee" on page 163

"ApiEngFixedFeeItem" on page 175

"ApiEngFixedFeeItemSplitBillOverride" on page 184

"ApiEngFixedFeeSplitBillOverride" on page 173

"ApiEngProduct" on page 186

"ApiEngProductSplitBillOverride" on page 199

"ApiEngProjectedResource" on page 200

"ApiEngRequestProcessingRule" on page 209

"ApiEngRequestSLA" on page 223

"ApiEngRevRec" on page 233

"ApiEngSplitBillingRule" on page 238

"ApiEngWorkCode" on page 245

"ApiEngWorkLocation" on page 250

ApiEngagement XML

The ApiEngagement XML is too lengthy to reproduce here. Use the GetXMLStructure method to obtain the XML structure for the ApiEngagement object and its subobjects.

The Action element is required at the main contract and subobjects levels in sXML, the xml string passed for the CreateByXML and UpdateByXML methods. Valid Action values:

1=Create

2=Update

9=Unchanged

Lookup logic for UpdateByXML

Main contract

The following sequence is used to find the contract ID:

1. If the sEngagementId parameter is passed in, the method uses this value for the contract ID.
2. If this fails, the method attempts to extract the contract ID from <contract id> in the XML.
3. If this fails, the contract ID is taken from the object properties.
4. If this fails, an attempt is made to look up the contract ID using <userdefinedengagementid> in the XML. If <userdefinedengagementid> has a value, but the value is invalid or duplicated, then you will get an error and no further attempts are made to look up the contract ID.
5. If <userdefinedengagementid> is empty, an attempt is made to look up the contract ID using <name> in the XML.

Engagement subobjects

The fields / nodes that are passed to look up each Engagement subobject is shown below:

ApiEngFixedFeeSplitBillOverride

If the splitengagementid is not provided in the <engagementsplitbillingrule> node in the XML, the UpdateByXML method uses the following fields to find the SplitEngagement ID:

MainEngagement and Customer - mandatory.

For both nodes, the method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the MainEngagement/Customer ID using <userdefinedid> in the XML; if <userdefinedid> is empty, an attempt is made to look up the MainEngagement/Customer ID using <name> in the XML.

ApiEngFixedFeeItemSplitBillOverride

If the splitengagementid is not provided in the <engagementsplitbillingrule> node in the XML, the UpdateByXML method uses the following fields to find the SplitEngagement ID:

MainEngagement and Customer - mandatory.

For both nodes, the method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the MainEngagement/Customer ID using <userdefinedid> in the XML; if <userdefinedid> is empty, an attempt is made to look up the MainEngagement/Customer ID using <name> in the XML.

ApiEngProductSplitBillOverride

If the splitengagementid is not provided in the <engagementsplitbillingrule> node in the XML, the UpdateByXML method uses the following fields to find the SplitEngagement ID:

MainEngagement and Customer - mandatory.

For both nodes, the method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the MainEngagement/Customer ID using <userdefinedid> in the XML; if <userdefinedid> is empty, an attempt is made to look up the MainEngagement/Customer ID using <name> in the XML.

EngagementBillingRate

If the rateid is not provided in the <engagementbillingrate> node in the XML, the UpdateByXML method uses the following fields/nodes to find the billing rate ID:

Engagement - mandatory, and in all cases is included in the lookup.

Resource - required if the rate is associated with a resource

BillingRole - required if the rate is associated with a billing role

BillingOffice - required if the rate is associated with a billing role

EngagementBillingRateHistory

If the billingrateseffectivedatesid is not provided in the <engagementbillingratehistory> node in the XML, the UpdateByXML method uses the following fields/nodes to find the billingrateseffectivedates ID:

EffectiveDate - mandatory, and in all cases is included in the lookup.

BillingRateId - optional - if not provided in the XML, the method will look up the billingrateid from the engagementbillingrate node that has the same billingrateindex number as the engagementbillingratehistory node.

EngagementFixedFee

If the fixedfeeid is not provided in the <engagementfixedfee> node in the XML, the UpdateByXML method uses the following fields to find the FixedFee ID:

1. Engagement - mandatory, and in all cases is included in the lookup.
2. If UserDefinedFixedFeeId has a value, the method attempts to lookup the Fixed Fee ID using <userdefinedfixedfeeid> in the XML.
3. If UserDefinedFixedFeeId does not have a value, then the lookup is based on Deliverable, BillingAmount and BillingDate, all of which should have values.

EngagementFixedFeeItem

If the fixedfeeid is not provided in the <engagementfixedfeeitem> node in the XML, the UpdateByXML method uses the following fields to find the FixedFeeItem ID:

1. ParentFixedFee - mandatory, and in all cases is included in the lookup.

   The method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the parent Fixed Fee ID using <userdefinedid> and if <userdefinedid> is empty, it uses <name>.
2. If UserDefinedFixedFeeId has a value, the method attempts to lookup the Fixed Fee Item ID using <userdefinedfixedfeeid> in the XML.
3. If UserDefinedFixedFeeId does not have a value, then the lookup is based on Deliverable, BillingAmount and BillingDate, all of which should have values.

EngagementProduct

If the engagementproductid is not provided in the <engagementproduct> node in the XML, the UpdateByXML method uses the following fields to find the EngagementProduct ID:

1. Lookup is based on Engagement, Product, and ProductDate, all of which should have values.

   For Product: the method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the Product ID using <name>.

EngagementProjectedResource

If the projectedresourceid is not provided in the <engagementprojectedresource> node in the XML, the UpdateByXML method uses the following fields to find the ProjectedResource ID:

1. The Engagement, BillingRole, StartDate, and FinishDate - mandatory and in all cases are included in the lookup.

   For BillingRole: the method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the Billing Role ID using <name>.

2. If either or both of Resource and Function have values, then they are used along with the four mandatory fields.

   For Resource: The method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the Resource ID using <userdefinedid>; if <userdefinedid> is empty an attempt is made to look up the Resource ID using <firstname> and <lastname> in the XML.

   For Function: the method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the Function ID using <name>.

EngagementRequestProcessingRule

If the engrequestbillingruleid is not provided in the <engagementrequestprocessingrule> node in the XML, the UpdateByXML method uses the following nodes/fields to find the EngRequestBillingrule ID:

Engagement and RequestType. If RequestType is empty, an attempt is made to look up the RequestType using <requesttypedescription>.

EngagementRequestSLA

If the engagementslaid is not provided in the <engagementrequestsla> node in the XML, the UpdateByXML method uses the following fields to find the EngagementSLA ID:

1. Engagement and Product - mandatory and in all cases are included in the lookup.

   For Product: the method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the Product ID using <name>.

2. If either or both of Priority and RequestType have values, then they are used along with the two mandatory fields.

   For Priority: the method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the Priority ID using <name> in the XML.

   For RequestType: the method extracts the <requesttype> value from the XML; if <requesttype> is empty, an attempt is made to look up the RequestType using <requesttypedescription>.

EngagementSplitBillingRule

If the splitengagementid is not provided in the <engagementsplitbillingrule> node in the XML, the UpdateByXML method uses the following fields to find the SplitEngagement ID:

MainEngagement and Customer - mandatory.

For both nodes, the method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the MainEngagement/Customer ID using <userdefinedid> in the XML; if <userdefinedid> is empty, an attempt is made to look up the MainEngagement/Customer ID using <name> in the XML.

EngagementWorkCode

Lookup is based on Engagement.

The UpdateByXML method can update the DefaultWorkCodeCategory and DefaultWorkCode fields, and change the engworkcode selection list using the <sxmlworkcodes> node.

For DefaultWorkCodeCategory and DefaultWorkCode, the method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the WorkCodeCategory/WorkCode ID using <userdefinedid> in the XML; if <userdefinedid> is

empty, an attempt is made to look up the WorkCodeCategory/WorkCode ID using <name> in the XML.

For each <engworkcode> within the <sxmlworkcodes> node, the method first extracts <workcodecategoryid> and <workcodeid> values from the XML; if <workcodecategoryid> and <workcodeid> are empty, an attempt is made to look up the WorkCodeCategory/WorkCode ID using <userdefinedworkcodecategoryid> and <userdefinedworkcodeid> in the XML; if <userdefinedworkcodecategoryid> and <userdefinedworkcodeid> are empty, an attempt is made to look up the WorkCodeCategory/WorkCode ID using <workcodecategory> and <workcode>in the XML.

EngagementWorkLocation

Lookup is based on Engagement.

The UpdateByXML method can update the DefaultWorkLocationGroup and DefaultWorkLocation fields, and change the engworklocation selection list using the <sxmlworklocations> node.

For DefaultWorkLocationGroup and DefaultWorkLocation, the method extracts the <id> value from the XML; if <id> is empty, an attempt is made to look up the WorkLocationGroup/WorkLocation ID using <userdefinedid> in the XML; if <userdefinedid> is empty, an attempt is made to look up the WorkLocationGroup/WorkLocation ID using <name> in the XML.

For each <engworklocation> within the <sxmlworklocations> node, the method first extracts <worklocationgroupid> and <worklocationid> values from the XML; if <worklocationgroupid> and <worklocationid> are empty, an attempt is made to look up the the WorkLocationGroup/WorkLocation ID using <userdefinedwlgid> and <userdefinedwlid> in the XML; if <userdefinedwlgid> and <userdefinedwlid> are empty, an attempt is made to look up the the WorkLocationGroup/WorkLocation ID using <worklocationgroup> and <worklocation>in the XML.

ApiEngagement: Add

```
Public Overrides Function Add(Optional ByRef sId As String = "") As Int32
```

Purpose

Add a contract and its subobjects to the Changepoint database.

Parameters

Parameter (*=required)	Description
sId	Place holder for the engagementId returned if successful

Returns

0 = Success

Nonzero = Error

Remarks

- If parameter sId is not provided, newly created engagementId is assigned to property engagementId, otherwise assigned to sId
- Ensure all mandatory properties are set
- A unique contract name is required

Example

```
Dim myEng As New ApiEngagement
Dim iRet As Int32
Dim sId As String
'code will be added
myEng.CPConnection = myCon
With myEng
   .Name = "Changepoint Canada"
   .Customer.Id = "{40C01201-37EC-47E1-8226-F505760C42F9}"
   .EngagementManager.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
   .BillingOffice.Id = "{5757A330-3476-11D3-807A-00105A0B7C01}"
   .MainContact.Id = "{CAC61D37-5A9F-420D-8786-9CDDA74A1F3B}"
   .SecondLevelApprover.Id = "{CA59FBCB-017A-4FEF-A07F-17A7A4315C63}"
   .AssociatedWorkGroup.Id = "{7712F1B0-F950-4FD6-B115-BAA2E9FCB468}"
   .InvoiceFormat.Id = "{27DF0C5B-E5DF-11D2-882F-006097B596A6}"
   .SupportDesk.Id = "{548B0D15-0305-4827-A666-E74046AFDD0F}"
   .BillToWorkLocationGroup.Id = "{333A206C-513E-4CAE-841A-EA73A82E8E81}"
   .BillToWorkLocation.Id = "{0409A21B-EFB9-42C4-ADB3-617279F524EF}"
   .TimeApprover.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
   .AltTimeApprover.Id = "{1F0C4CFB-9906-4895-9C9E-43094E0E6F30}"
   .ExpenseApprover.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
   .AltExpenseApprover.Id = "{1F0C4CFB-9906-4895-9C9E-43094E0E6F30}"
   .AvailableToAll = True
   .TaxOnRecordDate = True
   .BypassMetadataCheck = CPMetadataCheck.OnlyMandatory
```

```
    .ContractAmount = 12345687
    .ContractStartDate = CDate("01/01/2008")
    .ContractEndDate = CDate("12/31/2008")
    .Description = "First contract"
    .FederalAudit = False
    .BillingCurrency = "CAD"
    .TimeApprovalDelegated = False
    .AllowProjectOverride = True
    .OvertimePercentage = 100
    .Overtime = True
    .EnableAutoUpdateRate = False
    .EnableAutoUpdateCost = False
    .UseTwoLevelOnWOffs = True
    .UseTwoLevelOnContractOverage = False
    .UseTwoLevelApproval = False
    .ExpenseMinimum = 1000
    .ExpenseApprovalDelegated = False
    .Billable = True
    .PreventTaxChanges = True
    .MaximumAmount = 2000000
    .InvoiceMaximum = True
    .ExpenseBillingPercentage = 100
    .ExpenseBillingType = "A"
    .BillingType = "D"
    .PaymentTerms = "{954F9D11-D72E-11D2-9AE8-006008A4D883}"
    .EngagementStatus = "W"
    .AllowEditToAll = True
    .AllowPrjMgrBillCont = True
    .EnableBatchInvoice = True
    .EnableProjectTimeApproval = True
    .ExpenseGLA = "EM"
    .OverrideTaskRestriction = True
    .SplitBilling = False
    .SupportDeskOverride = False
    .Template = True
    .BillingAddress = 1
    .sxmlUDF = "<root><udf></udf></root>"
    .
    .
End With
iRet = myEng.Add(sId)
```

Related information

"ApiEngagement" on page 98

ApiEngagement: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, Optional ByRef sId As String
= "") As Int32
```

Purpose

Create a contract and its subobjects from an XML string of the Contract object.

Parameters

Parameter (*=required)	Description
*sXML	A new contract XML string which must be passed based on the Engagement xml schema provided by GetXMLStructure. You can pass a partial schema of the main Engagement or subobjects as long as you include the mandatory fields, the XML is wrapped in the <Engagement> </Engagement> nodes, and subobject nodes are valid.
sId	ByRef parameter to pass back the new EngagementId.

Returns

0 = Success

Nonzero = Error

Remarks

The ApiEngagement XML structure can be obtained by the GetXMLStructure or GetByXML method.

BillingRateHistory, FixedFeeItem and SplitBillingRule cannot be added using CreateByXML. Use UpdateByXML instead.

For details of the use of UDF default values, see "UDF default values logic for CreateByXML" on page 746.

Example

Not available

Related information

"ApiEngagement" on page 98

"ApiEngagement XML" on page 113

ApiEngagement: Delete

```
Public Overrides Function Delete(Optional ByVal sId As String = "") As Int32
```

Purpose

Remove the specified contract.

Parameters

Parameter (*=required)	Description
sId	ID of the contract record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property EngagementId is used.

Example

```
Dim myEng As New ApiEngagement
Dim iRet As Int32
myEng.CPConnection = myCon
iRet = myEng.Delete("{C218824A-67C2-4F34-B4F6-8EB48728FCF3}")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: Exists

```
Public Overrides Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Verify whether the contract exists

Parameters

Parameter (*=required)	Description
sId	ID of the contract to verify.

Returns

True if the contract exists, else False.

Remarks

If sId is not provided, property EngagementId is used.

Example

```
Dim myEng As New ApiEngagement
Dim bRet As Boolean
myEng.CPConnection = myCon
bRet = myEng.Exists("{C218824A-67C2-4F34-B4F6-8EB48728FCF3}")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetBillingOffice

```
Public Function GetBillingOffice(ByVal sBillingOfficeId As String) As DataSet
```

Purpose

Retrieve billing office information for the specified billing office.

Parameters

Parameter (*=required)	Description
*sBillingOfficeId	Billing office ID

Returns

Billing office information in a dataset upon success and nothing in a dataset upon error. Returns the following columns:

BillingOfficeId,Description,BaseCurrency,Deleted,CostCenter,RecognizeRevenue, CostRateToUse,FedAudit,AuditEnabledDate,InvoiceFormatSelection,Active, DefaultInvoiceFormat,DefaultPaymentTerms,PreventRateChanges,UseInvoiceApproval, PreventTaxChanges,PreventWriteUps,UseTwoLevelApproval,AllowEngagementOverride, SecondLevelApprover,UseTwoLevelOnContractOverage,UseTwoLevelOnWOffs, EngagementBillable,ForceFixedFeeSelection,RestrictTaskAssignments, AllowOverrideTaskRestriction,AdditionalItem

Remarks

None

Example

```
Dim myEng As New ApiEngagement
Dim dsRet As DataSet
myEng.CPConnection = myCon
dsRet = myEng.GetBillingOffice("{5757A330-3476-11D3-807A-00105A0B7C01}")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetByCustomerId

```
Public Function GetByCustomerId(ByVal sCustomerId As String) As DataSet
```

Purpose

Retrieve a list of billable contracts for a specified customer

Parameters

Parameter (*=required)	Description
*sCustomerId	Customer ID

Returns

Billable contracts including split billing contracts in a dataset upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

EngagementId, Name, Description, CustomerId, BillingOfficeId

Example

```
Dim myEng As New ApiEngagement
Dim dsRet As DataSet
myEng.CPConnection = myCon
dsRet = myEng.GetByCustomerId("{40C01201-37EC-47E1-8226-F505760C42F9}")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetByCustomerName

```
Public Function GetByCustomerName(ByVal sCustomerName As String) As DataSet
```

Purpose

Retrieve a list of contracts for a specified customer

Parameters

Parameter (*=required)	Description
*sCustomerName	Customer name

Returns

All contracts excluding split billing contracts in a dataset upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

EngagementId, Name, AlternateName, UserDefinedEngagementId, CustomerId, CustomerName, BillingOfficeId, BillingOfficeName

Example

```
Dim myEng As New ApiEngagement
Dim dsRet As DataSet
myEng.CPConnection = myCon
dsRet = myEng.GetByCustomerName("Changepoint Corporation")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetById

```
Public Overrides Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Fills the contract object and its subobjects with information of the specified sId passed in the parameter or of the property EngagementId.

Parameters

Parameter (*=required)	Description
sId	Engagement ID

Returns

0 = Success

Nonzero = Error

Remarks

If optional parameter sId is not provided, property EngagementId is used.

Example

```
Dim myEng As New ApiEngagement
Dim iRet As Int32
myEng.CPConnection = myCon
iRet = myEng.GetById("{C218824A-67C2-4F34-B4F6-8EB48728FCF3}")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetByName

```
Public Function GetByName(ByVal sName As String) As DataSet
```

Purpose

Retrieve a contract for a specified name

Parameters

Parameter (*=required)	Description
*sName	Name of contract

Returns

A contract excluding split billing contracts in a dataset upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

EngagementId, Name, AlternateName, UserDefinedEngagementId, CustomerId, CustomerName, BillingOfficeId, BillingOfficeName

Example

```
Dim myEng As New ApiEngagement
Dim dsRet As DataSet
myEng.CPConnection = myCon
dsRet = myEng.GetByName("Installation phase")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetByUserDefinedId

```
Public Function GetByUserDefinedId(ByVal sUserDefinedEngagementId As String)
As Int32
```

Purpose

Fills the object with contract information of the specified user defined contract ID.

Parameters

Parameter (*=required)	Description
*sUserDefinedEngagementId	User defined contract ID

Returns

0 = Success

Nonzero = Error

Remarks

UserDefinedEngagementId is treated as unique identifier, so error number -11 is returned if multiple records found.

Example

```
Dim myEng As New ApiEngagement
Dim iRet As Int32
myEng.CPConnection = myCon
iRet = myEng.GetByUserDefinedId("Eng-2008-000001")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetByXML

```
Public Function GetByXML(Optional ByVal sXML As String = "", Optional ByVal
sEngagementId As String = "") As String
```

Purpose

Takes the XML string passed in the sXML parameter and returns the string filled with data for the contract specified in the sEngagementId parameter and the contract’s subobjects.

Parameters

Parameter (*=required)	Description
sXML	XML string of the contract object, and its subobjects with the fields to be populated specified.
sEngagementId	Contract ID • If no Contract ID is passed in, the XML string (sXML) is examined • If there is no Contract ID in sXML then the object's Contract ID is selected. • If there still is no valid Contract ID, the method returns an error.

Returns

An XML string corresponding to sXML, with data inserted from the identified Contract and its subobjects.

Remarks

If sXML = "" then the XML string provided by GetXMLStructure is used, and the entire Contract and its subobjects are passed back, populated with all available data.

To get only specific data passed back, include in sXML only the corresponding subobjects and elements wrapped in <Engagement> </Engagement>.

Example

Not available

Related information

"ApiEngagement" on page 98

"ApiEngagement XML" on page 113

ApiEngagement: GetCustomerById

```
Public Function GetCustomerById(ByVal sId As String) As DataSet
```

Purpose

Retrieve customer information for a contract by engagementId

Parameters

Parameter (*=required)	Description
*sId	Engagement ID

Returns

Customer information in a dataset upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

CustomerId, CustomerName, EngagementName

Example

```
Dim myEng As New ApiEngagement
Dim dsRet As DataSet
myEng.CPConnection = myCon
dsRet = myEng.GetCustomerById("{C218824A-67C2-4F34-B4F6-8EB48728FCF3}")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetCustomers

```
Public Function GetCustomers() As DataSet
```

Purpose

Retrieve a list of customers for which contracts can be created.

Parameters

None

Returns

Customer information in a dataset upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

CustomerId, Name, AlternateName, UserDefinedCustomerId, Business1Address, Business1AddressLine2, Business1AddressLine3, Business1City, Business1Province, Business1Postal, Business1Country

Example

```
Dim myEng As New ApiEngagement
Dim dsRet As DataSet
myEng.CPConnection = myCon
dsRet = myEng.GetCustomers()
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As String
```

Purpose

Returns the EngagementId based on the UDF Text field and value.

Parameters

Parameter (*=required)	Description
*sUDFField	UDF text field name (for example, Text1)
*sUDFValue	UDF text value

Returns

EngagementId, or an empty string if nothing is found.

Remarks

None

Related information

"ApiEngagement" on page 98

ApiEngagement: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows As Int16 = -1) As
DataSet
```

Purpose

Retrieve a list of contracts.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).

Returns

Engagement records in a dataset upon success and nothing in a dataset upon error.

Returns all rows when method called without parameter.

Remarks

Split billing contracts are not included in the list.

Returns the following columns:

EngagementId, Name, AlternateName, UserDefinedEngagementId, CustomerId, CustomerName, BillingOfficeId, BillingOfficeName

Example

```
Dim myEng As New ApiEngagement
Dim dsRet As DataSet
myEng.CPConnection = myCon
dsRet = myEng.GetList()
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetPreferredResources

```
Public Function GetPreferredResources(ByVal sEngagementId As String, Optional
ByVal sResourceId As String = "") As DataSet
```

Purpose

Retrieve a list of resources that can use this contract

Parameters

Parameter (*=required)	Description
*sEngagementId	Engagement ID
sResourceId	Resource ID

Returns

A dataset of information about the resources that have VE, CE or ME features as well as the contract manager and contract creator if they have the features VE, CE or ME upon success and nothing upon error.

Remarks

Returns the following columns:

ResourceId, ResourceName, AltResourceName

Feature codes:

- CE – Create Engagements
- ME – Edit Engagements
- VE – View Engagements

If sResourceId is not provided, the login userid is used.

Example

```
Dim myEng As New ApiEngagement
Dim dsRet As DataSet
myEng.CPConnection = myCon
dsRet = myEng.GetPreferredResources("{C218824A-67C2-4F34-B4F6-8EB48728FCF3}")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetResourcesWithEngAccess

```
Public Function GetResourcesWithEngAccess() As DataSet
```

Purpose

Retrieve a list of resources that have VE, CE or ME features.

Parameters

None

Returns

A dataset of information about the resources that have VE, CE or ME features upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

ResourceId, ResourceName, AltResourceName

Feature codes:

- CE – Create Engagements
- ME – Edit Engagements
- VE – View Engagements

Example

```
Dim myEng As New ApiEngagement
Dim dsRet As DataSet
myEng.CPConnection = myCon
dsRet = myEng.GetResourcesWithEngAccess()
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetStatusByBillingOfficeId

```
Public Function GetStatusByBillingOfficeId(ByVal sBillingOfficeId As String)
As DataSet
```

Purpose

Retrieve the statuses of the contract workflow for a billing office.

Parameters

Parameter (*=required)	Description
*sBillingOfficeId	Billing office ID

Returns

The statuses of the contract workflow in a dataset upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

Code, Description, Step, EngagementWorkFlowId, Billable

Example

```
Dim myEng As New ApiEngagement
Dim dsRet As DataSet
myEng.CPConnection = myCon
dsRet = myEng.GetStatusByBillingOfficeId("{5757A330-3476-11D3-807A-
00105A0B7C01}")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetUDF

```
Public Function GetUDF(Optional ByVal retOption As CPUDFReturnType =
CPUDFReturnType.OnlyValues) As String
```

Purpose

Retrieve UDF (configurable field) information for engagement

Parameters

Parameter (*=required)	Description
retOption	The CPUDFReturnType Values are: • WithStructure = 0 – returns UDF field data with full field structure • OnlyValues = 1 – returns only UDF fields with data, do not include any field structure or options • OnlyStructure = 2 – returns only UDF XML structure, no field data • OnlyStructureAndOptions = 3 – returns UDF structure and option data without field data • WithStructureAndOptions = 4 – returns UDF field data with full structure and all field options (except entity based and conditional codes)

Returns

An XML string of UDF's

Remarks

The BillingOffice.Id is mandatory.

If EngagementId is not provided, returns default UDF values.

Example

```
Dim myEng As New ApiEngagement
Dim sRet As String
With myEng
   .CPConnection = myCon
   .BillingOffice.Id = "{27AF2827-0E98-430A-B2A0-9E57665E70BB}"
   .EngagementId = "{E90A4835-786F-4AFE-89ED-88A959608277}"
   sRet = .GetUDF(CPUDFReturnType.WithStructureAndOptions)
End With
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal codeName As String, Optional ByVal
searchString As String = "") As String
```

Purpose

Retrieve options for a UDF code

Parameters

Parameter (*=required)	Description
*codeName	The name of the UDF code, for example, "Code3"
SearchString	Returns the options that start with this string.

Returns

An XML string of the option list that exists for a specified code.

Remarks

The BillingOffice.Id is mandatory.

If EngagementId is not provided, returns default UDF values.

Example

```
Dim myEng As New ApiEngagement
Dim sRet As String
With myEng
   .CPConnection = myCon
   .BillingOffice.Id = "{27AF2827-0E98-430A-B2A0-9E57665E70BB}"
   .EngagementId = "{E90A4835-786F-4AFE-89ED-88A959608277}"
   sRet = .GetUDFCodeOptions("Code3", "Changepoint Corp")
End With
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetWorkDefaults

```
Public Function GetWorkDefaults(ByVal sId As String) As DataSet
```

Purpose

Retrieve the default work code and work location information for an Engagement

Parameters

Parameter (*=required)	Description
*sId	Engagement ID

Returns

A record set of information in the EngagementWorkDefault table.

Remarks

Returns the following columns:

EngagementId, WorkLocationGroupId, WorkLocationId, WorkCodeCategoryId, WorkCodeId, AllowRequestTimeLocOverride, AllowRequestTimeWorkOverride, BillToWorklocationGroupId, BillToWorkLocationId

Example

```
Dim myEng As New ApiEngagement
Dim dsRet As DataSet
myEng.CPConnection = myCon
dsRet = myEng.GetWorkDefaults("{C218824A-67C2-4F34-B4F6-8EB48728FCF3}")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Return the XML structure of the ApiEngagement object and its subobjects.

Parameters

None

Returns

An XML string of the ApiEngagement object and its subobjects.

Remarks

This method is used as a template for CreateByXML and UpdateByXML.

Example

Not available

Related information

"ApiEngagement" on page 98

"ApiEngagement XML" on page 113

ApiEngagement: HasP2AInvoice

```
Public Function HasP2AInvoice(ByVal sId As String) As Boolean
```

Purpose

Check if the contract has a pending second level approval invoice.

Parameters

Parameter (*=required)	Description
*sId	Engagement ID

Returns

True = has a pending second level approval invoice

False = no pending second level approval invoice

Remarks

None

Example

```
Dim myEng As New ApiEngagement
Dim bRet As Boolean
myEng.CPConnection = myCon
bRet = myEng.HasP2AInvoice("{C218824A-67C2-4F34-B4F6-8EB48728FCF3}")
```

Related information

"ApiEngagement" on page 98

ApiEngagement: LockEngagement

```
Public Function LockEngagement(ByVal sEngagementId As String, ByVal bLock As
Boolean) As String
```

Purpose

Lock a contract when in edit mode. Releases the contract when edit is finished.

Parameters

Parameter (*=required)	Description
*sEngagementId	Engagement ID
*bLock	True for locking contract. False for releasing contract

Returns

Resource name if contract is locked and an empty string if contract is not locked.

Remarks

It's not necessary to call this method before the Update method is called.

Example

```
Dim myEng As New ApiEngagement
Dim sRet As String
myEng.CPConnection = myCon
sRet = myEng.LockEngagement("{C218824A-67C2-4F34-B4F6-8EB48728FCF3}",True)
```

Related information

"ApiEngagement" on page 98

ApiEngagement: PrintBillingAddress

```
Public Function PrintBillingAddress(ByVal sCustomerId As String, ByVal
sContactID As String, ByVal iAddress As Integer) As String
```

Purpose

Retrieve billing address formatted by country standard for a customer.

Parameters

Parameter (*=required)	Description
*sCustomerId	Customer ID
*sContactId	Main contact ID
*iAddress	Type of address. iAddress values and corresponding returns are as follows: • 1 Customer Business Address1 • 2 Customer Business Address2 • 3 Customer Business Address3 • 4 Contact Business Address • 5 Contact Home Address • 6 Contact Other Address

Returns

Engagement billing address in a string

Remarks

iAddress values and corresponding returns are as follows:

- 1 Customer Business Address1
- 2 Customer Business Address2
- 3 Customer Business Address3
- 4 Contact Business Address
- 5 Contact Home Address
- 6 Contact Other Address

Example

```
Dim myEng As New ApiEngagement
Dim sRet As String
myEng.CPConnection = myCon
```

```
sRet = myEng.PrintBillingAddress("{40C01201-37EC-47E1-8226-F505760C42F9}", "
{CAC61D37-5A9F-420D-8786-9CDDA74A1F3B}", 1)
```

Related information

"ApiEngagement" on page 98

ApiEngagement: SaveUDF

```
Public Function SaveUDF(Optional ByVal sUDF As String = "") As Int32
```

Purpose

Saves (Insert/Update) UDF data for a contract.

Parameters

Parameter (*=required)	Description
sUDF	UDF string in XML format.

Returns

0 = Success

Nonzero = Error

Remarks

The EngagementId and BillingOffice.Id are mandatory fields, so these properties should be populated before calling SaveUDF method.

It's recommended to call the GetUDF method to obtain correct UDF XML format.

For more information, see "Appendix - User Defined Field (UDF) Definitions" on page 310.

Example

```
Dim myEng As New ApiEngagement
Dim iRet As Int32
Dim strXMLUDF as string ="<root></root>"
With myEng
   .CPConnection = myCon
   .BillingOffice.Id = "{27AF2827-0E98-430A-B2A0-9E57665E70BB}"
   .EngagementId = "{E90A4835-786F-4AFE-89ED-88A959608277}"
   iRet = .SaveUDF(strXMLUDF)
```

```
End With
```

Related information

"ApiEngagement" on page 98

"ApiEngagement: GetUDF" on page 135

"UDF XML" on page 742

ApiEngagement: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Update the general information of a contract and its subobjects.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

Before updating a contract, it is recommended that you get the existing contract data by calling the GetById method to populate all the existing subobjects. All values passed to the update method are updated.

Example

```
Dim myEng As New ApiEngagement
Dim iRet As Int32
myEng.CPConnection = myCon
iRet = myEng.GetById("{AF096708-51C6-4528-80A4-086FC0B16CC9}")
With myEng
   .TimeApprover.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
   .AltTimeApprover.Id = "{CA59FBCB-017A-4FEF-A07F-17A7A4315C63}"
   .ExpenseApprover.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
   .AltExpenseApprover.Id = "{CA59FBCB-017A-4FEF-A07F-17A7A4315C63}"
   .AuditDetail = "Change the Approvers."
   ...
```

```
   iRet = .Update()
End With
```

Related information

"ApiEngagement" on page 98

"ApiEngagement: GetById" on page 126

ApiEngagement: UpdateByXML

```
Public Function UpdateByXML(Optional ByVal sXML As String = "", Optional ByVal
sEngagementId As String = "") As Int32
```

Purpose

Update the database using an XML string of the Contract object and its subobjects containing update information.

Parameters

Parameter (*=required)	Description
sXML	An Contract XML string with content to be used to update the fields in the Contract and its subobjects.
sEngagementId	Contract ID of the contract being updated.

Note: The Contract Action must be 2, to update the contract, or 9 to leave the contract unchanged. Subobject Actions can be 1, 2, or 9.
For more information on Actions, see "ApiEngagement XML" on page 113.

Returns

0 = Success

Nonzero = Error

Remarks

BillingRateHistory, FixedFeeItem and SplitBillingRule can be added and updated using UpdateByXML.

BillingRate and BillingRateHistory require the same tag and same index # to be used in <billingrateindex> when updating or adding a new BillingRateHistory record.

Possible values for EngagementWorkcode and EngagementWorklocation selected attribute are 1 (selected) or 0 (not selected).

See .

Example

Not available

Related information

"ApiEngagement" on page 98

"ApiEngagement XML" on page 113

ApiEngBillingRate

The ApiEngBillingRate object represents billing rates set in the contract.

Namespace

Changepoint.ChangepointAPI2.ApiEngBillingRate

Methods

Properties

Property (*=required)	Type	Description
*BillingCurrency	String	Billing currency.
*BillingOffice	Identity	Identifier of the billing office. Mandatory when BillingRole is provided.
BillingOfficeRateId	String	Read-only. Identifier of Billing office rate.
*BillingRole	Identity	Identifier of the billing role. Mandatory when BillingOffice is provided.
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
CostCenter	Identity	Identifier of the cost center. If it requires updating with adding a billing rate, the UpdateEngInfoRequired property should be set to True.
*CostCurrency	String	Cost currency
*CostRate	Decimal	Cost rate
*CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CreatedBy	Signature	The creator of the record and when it was created.

Property (*=required)	Type	Description
Customer	Identity	Read-only. Identifier of the customer
DefaultBillingRole	Identity	Identifier of the default billing role. If it requires updating with adding a billing rate, the UpdateEngInfoRequired property should be set to True.
DiscountPercentage	Decimal	Discount percentage
Deleted	Boolean	Flag that indicates if the object has been deleted.
EnableAutoUpdateCost	Boolean	Auto-update cost rates. If it requires updating with adding a billing rate, the UpdateEngInfoRequired property should be set to True.
EnableAutoUpdateRate	Boolean	Auto-update billing rates. If it requires updating with adding a billing rate, the UpdateEngInfoRequired property should be set to True.
*Engagement	Identity	Identifier of the contract.
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
ExpenseMinimum	Decimal	Expense limit that doesn't require contract expense approver's attention. If requires updating with adding a billing rate, UpdateEngInfoRequired property should be set to True.
mVersion	String	Read-only. The current version number for internal version control.
NegotiatedRate	Decimal	Read-only. Calculated field (StandardRate x DiscountPercentage)

Property (*=required)	Type	Description
*OTCostRate	Decimal	OT cost rate
Overtime	Boolean	Include overtime flag. If it requires updating with adding a billing rate, the UpdateEngInfoRequired property should be set to True.
OvertimePercentage	Decimal	Overtime percentage. If it requires updating with adding a billing rate, the UpdateEngInfoRequired property should be set to True.
RateId	String	Billing rate ID.
RequestOnly	Boolean	Request only.
*Resource	Identity	Identifier of the resource. Mandatory when no BillingRole is provided.
StandardRate	Decimal	Read-only. Standard rate
UpdatedBy	Signature	The updater of the record and when it was updated.
UpdateEngInfoRequired	Boolean	This flag is used only with the Add method for ApiEngBillingRate object. To update contract information when adding a new billing rate, set this flag to True.

Related information

"ApiEngagement" on page 98

ApiEngBillingRate: Add

```
Public Function Add(Optional ByRef sId As String = "") As Int32
```

Purpose

Insert a billing rate for a contract.

Parameters

Parameter (*=required)	Description
sId	Place holder for a newly created RateId.

Returns

0 = Success

Nonzero = Error

Remarks

If parameter sId is not provided newly-created RateId is assigned to property RateId, otherwise assigned to sId.

Ensure all mandatory properties are set.

If there are no value assigned to CostRate and OTCostRate, zero value will be given and saved for these fields. It's recommended to have these properties filled before Add is called.

Both billingOffice and billingrole need to have values assigned in order to be saved. BillingOffice cannot be saved when adding resource rate only.

Both resource rate and billing rate should have already defined in Changepoint.

Contract info associated with billing rate can be saved when UpdateEngInfoRequired is set to True.

Example

```
Dim myBillingRate As New ApiEngBillingRate
Dim iRet As Int32
Dim sId As String

With myBillingRate
   .CPConnection = myCon
   .BillingOffice.Id = "{27AF2827-0E98-430A-B2A0-9E57665E70BB}"
   .Engagement.Id = "{70852075-3A41-4406-955E-B96C777AB431}"
   .BillingRole.Id = "{3FED4827-F58F-4BB1-9BF4-23B06CC0BD09}"
   .Resource.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
   .RequestOnly = True
   .DiscountPercentage = 50
   .CostRate = 750
   .OTCostRate = 100
```

```
   .BillingCurrency = "USD"
   .CostCurrency = "USD"
   ...
   iRet = .Add(sId)
End With
```

Related information

"ApiEngBillingRate" on page 145

ApiEngBillingRate: Delete

```
Public Function Delete(Optional ByVal sId As String = "") As Int32
```

Purpose

Remove a billing rate and its history records.

Parameters

Parameter (*=required)	Description
sId	RateId

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property RateId is used.

Example

```
Dim myBillingRate As New ApiEngBillingRate
Dim iRet As Int32
myBillingRate.CPConnection = myCon
bRet = myBillingRate.Delete("{CC0AA774-969C-43B8-A64E-1A9A91B50F94}")
```

Related information

"ApiEngBillingRate" on page 145

ApiEngBillingRate: Exists

```
Public Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Check whether this billing rate exists or not

Parameters

Parameter (*=required)	Description
sId	RateId

Returns

True if the billing rate exists, else False.

Remarks

If sId is not provided, property RateId is used.

Example

```
Dim myBillingRate As New ApiEngBillingRate
Dim bRet As Boolean
myBillingRate.CPConnection = myCon
bRet = myBillingRate.Exists("{CC0AA774-969C-43B8-A64E-1A9A91B50F94}")
```

Related information

"ApiEngBillingRate" on page 145

ApiEngBillingRate: GetBillingRoles

```
Public Function GetBillingRoles(ByVal sBillingOfficeId As String) As DataSet
```

Purpose

Retrieve billing roles for the billing office.

Parameters

Parameter (*=required)	Description
*sBillingOfficeId	Billing office ID.

Returns

The billing roles for the billing office in a dataset upon success and nothing in a dataset upon error.

Remarks

Returns the following columns: BillingRoleId, Description

Example

```
Dim myBillingRate As New ApiEngBillingRate
Dim dsRet As DataSet
myBillingRate.CPConnection = myCon
dsRet = myBillingRate.GetBillingRoles("{5757A330-3476-11D3-807A-
00105A0B7C01}")
```

Related information

"ApiEngBillingRate" on page 145

ApiEngBillingRate: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As DataSet
```

Purpose

Retrieve all billing rate records for a contract.

Parameters

Parameter (*=required)	Description
*sEngagementId	ID of the contract.

Returns

A dataset of billing rate records upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

BillingOfficeName, BillingRoleDescription, CurrencyDescription, ResourceName,

AlternateResource, BillingRateID, ResourceId, BillingOfficeId, BillingRoleId, StandardRate, DiscountPercentage, BillingRate, Currency, CostRate, CostCurrency, RequestOnly, BillingOfficeRateId, OTCostRate

Example

```
Dim myBillingRate As New ApiEngBillingRate
Dim dsRet As DataSet
myBillingRate.CPConnection = myCon
dsRet = myBillingRate.GetByEngagementId("{C218824A-67C2-4F34-B4F6-
8EB48728FCF3}")
```

Related information

"ApiEngBillingRate" on page 145

ApiEngBillingRate: GetById

```
Public Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Fills the object with billing rate of the specified sId passed in the parameter or of the property RateId.

Parameters

Parameter (*=required)	Description
sId	ID of the billing rate record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property RateId is used.

Example

```
Dim myBillingRate As New ApiEngBillingRate
Dim iRet As Int32
myBillingRate.CPConnection = myCon
iRet = myBillingRate.GetById("{CC0AA774-969C-43B8-A64E-1A9A91B50F94}")
```

Related information

"ApiEngBillingRate" on page 145

ApiEngBillingRate: GetEngagementInfo

```
Public Function GetEngagementInfo(ByVal sEngagementId As String) As DataSet
```

Purpose

Retrieve contract information associated with billing rate.

Parameters

Parameter (*=required)	Description
*sEngagementId	ID of the contract.

Returns

0 = Success

Nonzero = Error

Remarks

Returns the following columns:

BillingCurrency, BillingType, CustomerId, FedAudit, CostCenterId, DefaultBillingRole, EnableAutoUpdateCost, EnableAutoUpdateRate,

ExpenseMinimum, Overtime, OvertimePercentage

Example

```
Dim myBillingRate As New ApiEngBillingRate
Dim dsRet As DataSet
myBillingRate.CPConnection = myCon
dsRet = myBillingRate.GetEngagementInfo("{C218824A-67C2-4F34-B4F6-
8EB48728FCF3}")
```

Related information

"ApiEngBillingRate" on page 145

ApiEngBillingRate: UpdateEngagementInfo

```
Public Function UpdateEngagementInfo() As Int32
```

Purpose

Update contract information associated with billing rate.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

Engagement.Id is required.

Updates the following fields: CostCenter, DefaultBillingRole, EnableAutoUpdateCost, EnableAutoUpdateRate, ExpenseMinimum, Overtime, OvertimePercentage

Example

```
Dim myBillingRate As New ApiEngBillingRate
Dim iRet As Int32
myBillingRate.CPConnection = myCon
With myBillingRate
   .Engagement.Id = "{C218824A-67C2-4F34-B4F6-8EB48728FCF3}"
   .RateId = "{C796C14E-0FD2-4EDE-95E0-FCB31FF7EB93}"
   .ExpenseMinimum = 10000
   .Overtime = True
   .OvertimePercentage = 50
   iRet = .UpdateEngagementInfo()
End With
```

Related information

"ApiEngBillingRate" on page 145

ApiEngBillingRateHistory

The ApiEngBillingRateHistory object allows users to retrieve and update the rate history for contracts.

Namespace

Changepoint.ChangepointAPI2.ApiEngBillingRateHistory

Methods

Properties

Property (*=required)	Type	Description
Active	Boolean	Active
BillingCurrency	String	Billing currency.
BillingOfficeRateId	String	Read-only. Identifier of Billing office rate.
BillingRate	Decimal	Read-only. Calculated field (StandardRate x DiscountPercentage)
*BillingRateId	String	Billing rate ID
*BillingRatesEffectiveDatesId	String	Billing rates effective date ID
BillingRoleId	String	Read-only. Identifier of billing role

Property (*=required)	Type	Description
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
CommentHistory	Object	Read-only. History of comments
Comments	String	Comments. Mandatory when contract audit is on
CostCurrency	String	Cost currency
CostRate	Decimal	Cost rate.
*CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CreatedBy	Signature	The creator of the record and when it was created.
Deleted	Boolean	Flag that indicates if the object has been deleted.
DiscountPercentage	Decimal	Discount percentage
*EffectiveDate	Date	Billing rate effective date
Engagement	Identity	Read-only. Identifier of the contract.

Property (*=required)	Type	Description
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
mVersion	String	Read-only. The current version number for internal version control.
OTCostRate	Decimal	Overtime cost rate.
RequestOnly	Boolean	Request only.
Resource	Identity	Read-only. Identifier of the resource
UpdatedBy	Signature	The updater of the record and when it was updated.

Related information

"ApiEngagement" on page 98

ApiEngBillingRateHistory: Add

```
Public Function Add(Optional ByRef sId As String = "") As Int32
```

Purpose

Insert a new billing rate effective date for a contract.

Parameters

Parameter (*=required)	Description
sId	

Returns

0 = Success

Nonzero = Error

Remarks

- Newly created BillingRatesEffectiveDatesId will be assigned to property BillingRatesEffectiveDatesId.
- RequestOnly fields in all history rates under same billing rate will be updated with the value assigned to current RequestOnly property.
- Ensure all mandatory properties are set.

Example

```
Dim myHistoryRate As New ApiEngBillingRateHistory
Dim iRet As Int32

With myHistoryRate
   .CPConnection = myCon
   .Active = True
   .BillingRateId = "{CC0AA774-969C-43B8-A64E-1A9A91B50F94}"
   .EffectiveDate = CDate("10/15/2008")
   .Comments = "New effective date"
   .DiscountPercentage = 30
   ...
   iRet = .Add()
End With
```

Related information

"ApiEngBillingRateHistory" on page 156

ApiEngBillingRateHistory: Exists

```
Public Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Check whether this billing rate effective date exists or not

Parameters

Parameter (*=required)	Description
sId	BillingRatesEffectiveDatesId

Returns

True if the billing rate effective date exists, else False.

Remarks

If sId is not provided, property BillingRatesEffectiveDatesId is used.

Example

```
Dim myHistoryRate As New ApiEngBillingRateHistory
Dim bRet As Boolean
myHistoryRate.CPConnection = myCon
bRet = myHistoryRate.Exists("{9D19662B-3FEF-4F3B-9E8A-24B40E8ABEEB}")
```

Related information

"ApiEngBillingRateHistory" on page 156

ApiEngBillingRateHistory: GetByBillingRateId

```
Public Function GetByBillingRateId(ByVal sBillingRateId As String) As DataSet
```

Purpose

Retrieve all billing rate effective date (billing rate history) records for a billing rate

Parameters

Parameter (*=required)	Description
*sBillingRateId	ID of the billing rate

Returns

A dataset of billing rate effective date records upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

CustomerId, EngagementId, BillingRateId, BillingRatesEffectiveDatesId, BillingRate, BillingCurrency, BillingOfficeRateId, CostRate, CostCurrency, OTCostRate, DiscountPercentage, EffectiveDate, CommentHistory, ResourceId, ResourceCostRate,

ResourceCostCurrency, ResourceOTCostRate, RoleCostRate, RoleCostCurrency, RoleOTCostRate, Active, RequestOnly

Example

```
Dim myHistoryRate As New ApiEngBillingRateHistory
Dim dsRet As DataSet
myHistoryRate.CPConnection = myCon
dsRet = myHistoryRate.GetByBillingRateId("{CC0AA774-969C-43B8-A64E-
1A9A91B50F94}")
```

Related information

"ApiEngBillingRateHistory" on page 156

ApiEngBillingRateHistory: GetById

```
Public Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Fills the object with billing rate effective date (history rate) record of the specified sId passed in the parameter or of the property BillingRatesEffectiveDatesId.

Parameters

Parameter (*=required)	Description
sId	ID of the billing rates effective date record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property BillingRatesEffectiveDatesId is used.

Example

```
Dim myHistoryRate As New ApiEngBillingRateHistory
Dim iRet As Int32
myHistoryRate.CPConnection = myCon
iRet = myHistoryRate.GetById("{9D19662B-3FEF-4F3B-9E8A-24B40E8ABEEB}")
```

Related information

"ApiEngBillingRateHistory" on page 156

ApiEngBillingRateHistory: Update

```
Public Function Update() As Int32
```

Purpose

Update a billing rate's effective date record.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

Updates the three fields: Active, RequestOnly, Comments

If the value of RequestOnly is changed, then RequestOnly fields in all history rates under same billing rate will be updated as well.

Example

```
Dim myHistoryRate As New ApiEngBillingRateHistory
Dim iRet As Int32
myHistoryRate.CPConnection = myCon
With myHistoryRate
   .BillingRatesEffectiveDatesId = "{2E629DCC-D25B-4A37-9A2D-C1793E624C72}"
   .Active = False
   .Comments = "Not Active"
   .RequestOnly = True
   iRet = .Update()
End With
```

Related information

"ApiEngBillingRateHistory" on page 156

ApiEngFixedFee

The ApiEngFixedFee object allows users to retrieve, add, update and delete fixed fees that are set at the contract level.

Namespace

Changepoint.ChangepointAPI2.ApiEngFixedFee

Methods

Properties

Property (*=required)	Type	Description
Billed	Boolean	Flag that indicates that this fixed fee is invoiced.
*BillingAmount	Decimal	Billing amount
*BillingDate	DateTime	Billing date
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Property (*=required)	Type	Description
*CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CreatedBy	Signature	The creator of the record and when it was created.
CRGL	String	Credit revenue general ledger This property is not used by Add/Update methods, but is populated by GetById method.
CRGLDescription	String	Credit revenue general ledger description This property is not used by Add/Update methods, but is populated by GetById method.
Customer	Identity	Identifier of the customer This property is not used by Add/Update methods, but is populated by GetById method.
Deleted	Boolean	Flag that indicates if the object has been deleted.
*Deliverable	String	Deliverable name
DRGL	String	Debit revenue general ledger This property is not used by Add/Update methods, but is populated by GetById method.

Property (*=required)	Type	Description
DRGLDescription	String	Debit revenue general ledger description This property is not used by Add/Update methods, but is populated by GetById method..
*Engagement	Identity	Identifier of the contract.
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
*FixedFeeId	String	Identifier of the Fixed fee schedule.
InvoiceGLAId	Identity	Invoice distribution GL account Id.
InvoiceWhenTaskCompleted	Boolean	Invoice only when the related task is complete.
mVersion	String	Read-only. The current version number for internal version control.
ParentFixedFeeId	String	Identifies the Parent fixed fee schedule. This property is not used by Add/Update methods, but is populated by GetById method.
PPC	Decimal	Physical percent complete This property doesn't get used by Add/Update methods. Populated by GetById method.
Prepaid	Boolean	Prepaid
RevAdj	Decimal	Revenue adjustment This property is not used by Add/Update methods, but is populated by GetById method.

Property (*=required)	Type	Description
RevMethDescription	String	Revenue method description This property is not used by Add/Update methods, but is populated by GetById method.
RevMethod	Integer	Revenue method This property is not used by Add/Update methods, but is populated by GetById method.
RevRec	Decimal	Revenue recognition This property is not used by Add/Update methods, but is populated by GetById method.
RevTent	Decimal	Tentative revenue recognition This property doesn't get used by Add/Update methods. Populated by GetById method.
RRUpdate	Boolean	Revenue recognition update This property doesn't get used by Add/Update methods. Populated by GetById method.
UserDefinedFixedFeeId	String	User defined fixed fee
WorkCode		Identifier of the work code
WorkCodeCategory	Identity	Identifier of the work code category
WorkLocation	Identity	Identifier of the work location
WorkLocationGroup	Identity	Identifier of the work location group
UpdatedBy	Signature	Updater of the record and when it was updated.

Related information

"ApiEngagement" on page 98

ApiEngFixedFee: Add

```
Public Overrides Function Add(Optional ByRef sId As String = "") As Int32
```

Purpose

Insert a new contract fixed fee schedule record for a contract.

Parameters

Parameter (*=required)	Description
sId	ID of the contract fixed fee record.

Returns

0 = Success

Nonzero = Error

Remarks

- If parameter sId is not provided, the new FixedFeeId is assigned to property FixedFeeId
- If parameter sId is provided, the newly FixedFeeId is assigned to sId.
- Ensure all mandatory properties are set.

Value of the property Billed should be always False. If value is not assigned, it takes a default value False.

Example

```
Dim myEngFixedFee As New ApiEngFixedFee
Dim iRet As Int32
Dim sId As String

myEngFixedFee.CPConnection = myCon
With myEngFixedFee
   .Engagement.Id = "{2E1D37B7-810F-43D6-ABAC-E624AB6F28EC}"
   .Billed = False
   .BillingDate = CDate("10/31/2008")
   .BillingAmount = 50000
   .Deliverable = "Design/Coding"
   .UserDefinedFixedFeeId = "FF000001"
   .WorkCodeCategory.Id = "{AA9C83C1-6E1E-4974-9DEC-7C554CC429D2}"
   .WorkCode.Id = "{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}"
```

```
   .WorkLocationGroup.Id = "{333A206C-513E-4CAE-841A-EA73A82E8E81}"
   .WorkLocation.Id = "{0409A21B-EFB9-42C4-ADB3-617279F524EF}"
   iRet = .Add(sId)
End With
```

Related information

"ApiEngFixedFee" on page 163

ApiEngFixedFee: Delete

```
Public Overrides Function Delete(Optional ByVal sId As String = "") As Int32
```

Purpose

Delete a fixed fee schedule for an contract

Parameters

Parameter (*=required)	Description
sId	ID of the contract fixed fee schedule record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property FixedFeeId is used.

Example

```
Dim myEngFixedFee As New ApiEngFixedFee
Dim iRet As Int32
myEngFixedFee.CPConnection = myCon
iRet = myEngFixedFee.Delete("{8A842477-5C28-4868-A520-846567A75C3A}")
```

Related information

"ApiEngFixedFee" on page 163

ApiEngFixedFee: Exists

```
Public Overrides Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Check whether this fixed fee schedule exists or not

Parameters

Parameter (*=required)	Description
sId	ID of the contract fixed fee schedule record.

Returns

True if the fixed fee schedule exists, else False.

Remarks

If sId is not provided, property FixedFeeId is used.

Example

```
Dim myEngFixedFee As New ApiEngFixedFee
Dim bRet As Boolean
myEngFixedFee.CPConnection = myCon
bRet = myEngFixedFee.Exists("{8A842477-5C28-4868-A520-846567A75C3A}")
```

Related information

"ApiEngFixedFee" on page 163

ApiEngFixedFee: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As DataSet
```

Purpose

Retrieve all fixed fee schedule records for a contract

Parameters

Parameter (*=required)	Description
*sEngagementId	ID of the contract

Returns

A dataset of contract fixed fee schedule records upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

CustomerId, EngagementId, FixedFeeId, BillingDate, BillingAmount, Deliverable, Billed,

CreatedOn, CreatedBy, UpdatedOn, UpdatedBy, RevenueMethod, PPC, RevTent, RevRec,

DrGL, CrGL, RevAdj, RRUpdate, UserDefinedfixedFeeID, Prepaid, ParentFixedFeeId,

DrGLDescription, CrGLDescription,

RevenueMethodDescription, WorkLocationGroupID, WorkLocationGroup, WorkLocationID,

WorkLocation, WorkCodeCategoryId, WorkCodeCategory, WorkCodeId,

WorkCode, ParentDeliverable, ParentBillingDate, HasScheduleItems

Example

```
Dim myEngFixedFee As New ApiEngFixedFee
Dim dsRet As DataSet
myEngFixedFee.CPConnection = myCon
dsRet = myEngFixedFee.GetByEngagementId("{C5CBB921-5FF0-48FA-9A14-
6F4072641FA0}")
```

Related information

"ApiEngFixedFee" on page 163

ApiEngFixedFee: GetById

```
Public Overrides Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Fills the object with the contract fixed fee schedule of the specified sId passed in the parameter or of the property FixedFeeId.

Parameters

Parameter (*=required)	Description
sId	ID of the contract fixed fee schedule record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property FixedFeeId is used.

Example

```
Dim myEngFixedFee As New ApiEngFixedFee
Dim iRet As Int32
myEngFixedFee.CPConnection = myCon
iRet = myEngFixedFee.GetById("{8A842477-5C28-4868-A520-846567A75C3A}")
```

Related information

"ApiEngFixedFee" on page 163

ApiEngFixedFee: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows As Int16 = -1) As
DataSet
```

Purpose

Retrieve contract fixed fee schedule records.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).

Returns

A dataset of contract fixed fee schedule records upon success and nothing in a dataset upon error.

Returns all rows when the method is called without a parameter.

Remarks

Returns the following columns:

EngagementId, EngagementName, CustomerId, CustomerName, FixedFeeId, BillingDate, BillingAmount, Deliverable, Billed

Example

```
Dim myEngFixedFee As New ApiEngFixedFee
Dim dsRet As DataSet
myEngFixedFee.CPConnection = myCon
dsRet = myEngFixedFee.GetList()
```

Related information

"ApiEngFixedFee" on page 163

ApiEngFixedFee: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Update a contract fixed fee schedule record.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

- It is recommended that you use the GetById method to get the existing contract fixed fee before the Update method is called.
- You can add a fixed fee only if the billing amount of the fixed fee is equal to the total billing amount of the fixed fee items.
- You can change the fixed fee billing amount and fixed fee items billing amounts using the Save method of the ApiEngFixedFeeItem object.
- The Value of the property Billed should be always False. If a value is not assigned, it takes a default value False.

Example

```
Dim myEngFixedFee As New ApiEngFixedFee
Dim iRet As Int32
Dim sId As String

myEngFixedFee.CPConnection = myCon
iRet = myEngFixedFee.GetById("{8A842477-5C28-4868-A520-846567A75C3A}")
With myEngFixedFee
   .BillingDate = CDate("10/31/2008")
   .BillingAmount = 50000
   .Deliverable = "Design/Coding"
   .UserDefinedFixedFeeId = "FF000001"
   .WorkCodeCategory.Id = "{AA9C83C1-6E1E-4974-9DEC-7C554CC429D2}"
   .WorkCode.Id = "{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}"
   .WorkLocationGroup.Id = "{333A206C-513E-4CAE-841A-EA73A82E8E81}"
   .WorkLocation.Id = "{0409A21B-EFB9-42C4-ADB3-617279F524EF}"
   ...
   iRet = .Update()
End With
```

Related information

"ApiEngFixedFee" on page 163

ApiEngFixedFeeSplitBillOverride

Purpose

Provides a collection object for split billing override information for contract fixedfees.

Namespace

Changepoint.ChangepointAPI2.ApiEngFixedFeeSplitBillOverride

Methods

None.

Properties

Property (*=required)	Type	Description
Amount if SplitByAmount = true, Amount is required if SplitByAmount = false, Amount is not required	Decimal	Amount of the fixedfee billing that is allocated to this contract. See the SplitByAmount and Percentage properties.
*CreatedBy	Signature	The creator of the record and when it was created.
*Customer	Identity	Customer related to the contract fixedfee.
Deleted	Boolean	Flag that indicates if the contract fixedfee split bill override has been deleted.
*EngagementFixedFee	Identity	Engagement fixedfee.
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
*MainContact	Identity	Contact for the customer related to the split contract fixedfee.
*MainEngagement	Identity	Engagement related to the split contract fixedfee.
mVersion	String	Read-only. The current version number for internal version control.

Property (*=required)	Type	Description
Percentage if SplitByAmount = false, Percentage is required if SplitByAmount = true, Percentage is not required	Decimal	Percentage of total fixedfee billing that is calculated to be allocated to this contract.
*SplitByAmount	Boolean	Set to True to split total fixedfee billing by entering an amount (Amount property) for this contract fixedfee. Set to False to split total fixedfee billing by entering a percentage (Percentage property) for this contract fixedfee.
SplitEngagementId	String	Engagement related to the split contract fixedfee.
UpdatedBy	Signature	The updater of the record and when it was updated.

ApiEngFixedFeeItem

The ApiEngFixedFeeItem object allows users to manually create a fixed fee schedule item.

Namespace

Changepoint.ChangepointAPI2.ApiEngFixedFeeItem

Methods

Properties

Property (*=required)	Type	Description
AllowTaskLink	Boolean	Sets whether or not you can link a task to the fixed fee item.
Billed	Boolean	Flag that indicates that this fixed fee is invoiced. This flag should be always set to False
*BillingAmount	Decimal	Billing amount
*BillingDate	Date	Billing date
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
*CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CreatedBy	Signature	The creator of the record and when it was created.
Deleted	Boolean	Flag that indicates if the contract fixed fee item to be deleted.
*Deliverable	String	Deliverable name
DoNotInvoice	Boolean	Sets whether or not the fixed fee item can be included on an invoice.

Property (*=required)	Type	Description
Engagement	Identity	Identifier of the contract that fixed fee item is associated with. This property doesn't get changed by Save method. Populated by GetById method.
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
*FixedFeeId	String	ID of the fixed fee item
HasUsed	Boolean	Has used. Internal use only
InvoiceGLAId	Indentity	Identifier of the invoice GL account that the fixed fee item is associated with.
InvoiceWhenTaskCompleted	Boolean	Invoice only when the related task is complete.
mVersion	String	Read-only. The current version number for internal version control.
ParentFixedFeeId	String	Id of the fixed fee that this fixed fee item is associated with.
Prepaid	Boolean	Prepaid
RevRec	Decimal	Revenue recognized amount This property doesn't get changed by Save method. Populated by GetById method.
UpdatedBy	Signature	The updater of the record and when it was updated.
UserDefinedFixedFeeId	String	User defined fixed fee item ID
*WorkCode	Identity	Identifier of the work code
*WorkCodeCategory	Identity	Identifier of the work code category

Property (*=required)	Type	Description
*WorkLocation	Identity	Identifier of the work location
*WorkLocationGroup	Identity	Identifier of the work location group
UpdatedBy	Signature	The updater of the record and when it was updated.

Related information

"ApiEngagement" on page 98

"ApiEngFixedFeeItem XML" on page 178

ApiEngFixedFeeItem XML

```
<root>
<engfixedfeeitem>
   <parentfixedfeeid/>
   <parentbillingamount/>
   <item>
      <fixedfeeid/>
      <billingdate/>
      <billingamount/>
      <deliverable/>
      <billed/>
      <userdefinedfixedfeeid/>
      <worklocationgroupid/>
      <worklocationid/>
      <workcodecategoryid/>
      <workcodeid/>
      <prepaid/>
      <invoicewhentaskcompleted/>
      <donotinvoice/>
      <invoiceglaid/>
      <allowtasklink/>
      <deleted/>
   </item>
</engfixedfeeitem>
</root>
```

Comments

Not available

Example

Not available

Related information

"ApiEngFixedFeeItem" on page 175

ApiEngFixedFeeItem: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As DataSet
```

Purpose

Returns a list of fixed fee schedule items for the contract.

Parameters

Parameter (*=required)	Description
sEngagementId	Engagement ID

Returns

Returns the following columns:

FixedFeeId, EngagementId, BillingDate, BillingAmount,

Deliverable, Billed, CreatedOn, CreatedBy, UpdatedOn,

UpdatedBy, RevRec, UserDefinedfixedFeeID, Prepaid, ParentFixedFeeId,

WorkLocationGroupID, WorkLocationGroup, WorkLocationID, WorkLocation,

WorkCodeCategoryId, WorkCodeCategory, WorkCodeId, WorkCode, ParentBillingAmount

Remarks

None.

Example

None.

Related information

"ApiEngFixedFeeItem" on page 175

ApiEngFixedFeeItem: GetById

```
Public Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Fills the object with contract fixed fee schedule item of the specified fixedfeeid passed in the parameter or in the property.

Parameters

Parameter (*=required)	Description
sId	Fixed Fee ID

Returns

0 = Success

Nonzero = Error

Remarks

If optional parameter sId is not provided, FixedFeeId is used.

Example

```
Dim myEngFFItem As New ApiEngFixedFeeItem
Dim iRet As Int32
myEngFFItem.CPConnection = myCon
iRet = myEngFFItem.GetById("{F66A6B2B-0BD4-4B95-A7D1-D901ED3B67AF}")
```

Related information

"ApiEngFixedFeeItem" on page 175

ApiEngFixedFeeItem: GetByParentFixedFeeId

```
Public Function GetByParentFixedFeeId(ByVal sFixedFeeId As String) As String
```

Purpose

Retrieve fixed fee schedule items for the specified fixed fee

Parameters

Parameter (*=required)	Description
*sFixedFeeId	The fixed fee ID

Returns

Engagement fixed fee schedule items in an XML document string.

Remarks

If there are no fixed fee schedule items for the specified fixed fee, an empty string is returned.

Example

```
Dim myEngFFItem As New ApiEngFixedFeeItem
Dim sRet As String = String.Empty
myEngFFItem.CPConnection = myCon
sRet = myEngFFItem.GetByParentFixedFeeId("{269206D7-2487-4F2A-868C-
4FF42C1533AC}")
```

Related information

"ApiEngFixedFeeItem" on page 175

"ApiEngFixedFeeItem XML" on page 178

ApiEngFixedFeeItem: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Retrieve the XML structure of the fixed fee item

Parameters

None

Returns

An XML string of the the contract fixed fee item

Remarks

None

Example

```
Dim myEngFFItem As New ApiEngFixedFeeItem
Dim sXmlStructure As String = String.Empty
myEngFFItem.CPConnection = myCon
sXmlStructure = myEngFFItem.GetXMLStructure()
```

Related information

"ApiEngFixedFeeItem" on page 175

"ApiEngFixedFeeItem XML" on page 178

ApiEngFixedFeeItem: Save

```
Public Function Save(ByVal sXmlFixedFeeItems As String) As Int32
```

Purpose

Update the fixed fee item information of an engagement

Parameters

Parameter (*=required)	Description
*sXmlFixedFeeItems	Fixed fee Items in XML format string.

Returns

0 = Success

Nonzero = Error

Remarks

- It is recommended that you use the GetByParentFixedFeeId or GetXMLStructure methods to get the correct XML format of parameter sXmlFixedFeeItems.

- Includes UPDATE, DELETE, and ADD functionalities.
  - To update a fixedfeeitem, use the GUID of the item to be updated as the value of the <fixedfeeid> tag.
  - To delete a fixedfeeitem, set its <deleted> tag to TRUE.
  - To add a fixedfeeitem, set its <fixedfeeid> tag to empty.
- The <parentfixedfeeid> tag is mandatory and should contain a GUID value which all the fixed fee items are associated with.
  <parentfixedfeeid> has one or more child <item> elements, each of which represents one fixed fee item.
- The parent billing amount is calculated by the system as the sum of the <billingamount> of the passed-in <item> elements, plus the billingamount of any existing fixed fee items in the database. The <parentbillingamount> node is ignored.
- The value of the <billed> tag should be set to 0 or empty

Example

```
Dim myEngFFItem As New ApiEngFixedFeeItem
Dim iRet As Int32
Dim sXmlFixedFeeItems As String

'sXmlFixedFeeItems contains 3 <item>s:
' the first is for an update, the second for delete, the third for add
sXmlFixedFeeItems =
"<root><engfixedfeeitem>
  <parentfixedfeeid>79C54E97-77C1-47E6-8CDD-CA185827743D</parentfixedfeeid>
  <parentbillingamount>40000</parentbillingamount>
  <item>
    <fixedfeeid>{F66A6B2B-0BD4-4B95-A7D1-D901ED3B67AF}</fixedfeeid>
    <billingdate>10/31/2008</billingdate>
    <billingamount>20000</billingamount>
    <deliverable>Coding phase 1</deliverable>
    <billed></billed>
    <userdefinedfixedfeeid></userdefinedfixedfeeid>
    <worklocationgroupid>{333A206C-513E-4CAE-841A-
EA73A82E8E81}</      worklocationgroupid>
    <worklocationid>{0409A21B-EFB9-42C4-ADB3-617279F524EF}</worklocationid>
    <workcodecategoryid>{AA9C83C1-6E1E-4974-9DEC-7C554CC429D2}</workcodecategoryid>
    <workcodeid>{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}</workcodeid>
    <prepaid>0</prepaid>
    <deleted>0</deleted>
  </item>
```

```
  <item><fixedfeeid>{5A54AC3C-E5EE-4FD0-80B7-458B7771829D}</fixedfeeid>
    <billingdate>09/31/2008</billingdate>
    <billingamount>20000</billingamount>
    <deliverable>Design phase</deliverable>
    <billed>0</billed>
    <userdefinedfixedfeeid></userdefinedfixedfeeid>
    <worklocationgroupid>{333A206C-513E-4CAE-841A-EA73A82E8E81}</worklocationgroupid>
    <worklocationid>{0409A21B-EFB9-42C4-ADB3-617279F524EF}</worklocationid>
    <workcodecategoryid>{AA9C83C1-6E1E-4974-9DEC-7C554CC429D2}</workcodecategoryid>
    <workcodeid>{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}</workcodeid>
    <prepaid>0</prepaid>
    <deleted>1</deleted>
  </item>
  <item>
    <fixedfeeid></fixedfeeid>
    <billingdate>11/31/2008</billingdate>
    <billingamount>20000</billingamount>
    <deliverable> Coding phase 2</deliverable>
    <billed></billed>
    <userdefinedfixedfeeid></userdefinedfixedfeeid>
    <worklocationgroupid>333A206C-513E-4CAE-841A-EA73A82E8E81</worklocationgroupid>
    <worklocationid>0409A21B-EFB9-42C4-ADB3-617279F524EF</worklocationid>
    <workcodecategoryid>AA9C83C1-6E1E-4974-9DEC-7C554CC429D2</workcodecategoryid>
    <workcodeid>{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}</workcodeid>
    <prepaid>0</prepaid>
    <deleted>0</deleted>
  </item>
</engfixedfeeitem></root>"

myEngFFItem.CPConnection = myCon
iRet = myEngFFItem.Save(sXmlFixedFeeItems)
```

Related information

"ApiEngFixedFeeItem" on page 175

"ApiEngFixedFeeItem XML" on page 178

"ApiEngFixedFeeItem: GetByParentFixedFeeId" on page 180

"ApiEngFixedFeeItem: GetXMLStructure" on page 181

ApiEngFixedFeeItemSplitBillOverride

Purpose

Provides a collection object for split billing override information for contract fixed fee items.

Namespace

Changepoint.ChangepointAPI2.ApiEngFixedFeeItemSplitBillOverride

Methods

None.

Properties

Property (*=required)	Type	Description
Amount if SplitByAmount = true, Amount is required if SplitByAmount = false, Amount is not required	Decimal	Amount of the fixed fee item billing that is allocated to this contract. See the SplitByAmount and Percentage properties.
CreatedBy	Signature	The creator of the record and when it was created.
*Customer	Identity	Customer related to the contract fixed fee item.
Deleted	Boolean	Flag that indicates if the contract fixed fee item split bill override has been deleted.
*EngagementFixedFeeItem	Identity	Engagement fixed fee item.
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
*MainContact	Identity	Contact for the customer related to the split contract fixed fee item.
*MainEngagement	Identity	Engagement related to the split contract fixed fee item.
mVersion	String	Read-only. The current version number for internal version control.

Property (*=required)	Type	Description
Percentage if SplitByAmount = false, Percentage is required if SplitByAmount = true, Percentage is not required	Decimal	Percentage of total fixed fee item billing that is calculated to be allocated to this contract.
*SplitByAmount	Boolean	Set to True to split total fixed fee item billing by entering an amount (Amount property) for this contract fixed fee item. Set to False to split total fixed fee item billing by entering a percentage (Percentage property) for this contract fixed fee item.
SplitEngagementId	String	Engagement related to the split contract fixed fee item.
UpdatedBy	Signature	The updater of the record and when it was updated.

ApiEngProduct

The ApiEngProduct object allows users to retrieve, update and delete products at the contract level.

Namespace

Changepoint.ChangepointAPI2.ApiEngProduct

Methods

Properties

Property (*=required)	Type	Description
AllowTaskLink	Boolean	Whether the contract product is allowed to be linked to a specific task.
BillableStatus	Identity	Availability for billing.
Billed	Boolean	Flag if it is invoiced.
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
*CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CreatedBy	Signature	The creator of the record and when it was created.
CRGL	String	Credit revenue general ledger This property doesn't get used by Add/Update methods. Populated by GetById method.

Property (*=required)	Type	Description
CRGLDescription	String	Credit revenue general ledger description. This property doesn't get used by Add/Update methods. Populated by GetById method.
Deleted	Boolean	Flag that indicates if the object has been deleted
Description	String	Description of the contract product.
DRGL	String	Debit revenue general ledger This property doesn't get used by Add/Update methods. Populated by GetById method.
DRGLDescription	String	Debit revenue general ledger description. This property doesn't get used by Add/Update methods. Populated by GetById method.
*Engagement	Identity	Identifier for the contract that is associated with the contract product
*EngProductId	String	ID of the contract product
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
FixedCost	Boolean	Fixed cost.
FixedCostOverride	Boolean	Fixed Cost Override
InvoiceGLAId	Identity	Invoice distribution GL account Id
IsLinkedToTask	Boolean	Read-only. Whether the contract product is linked to a specific task.
mVersion	String	Read-only. The current version number for internal version control.

Property (*=required)	Type	Description
*NegotiatedPrice	Decimal	Negotiated Price
PPC	Decimal	Physical % complete This property doesn't get used by Add/Update methods. Populated by GetById method.
*Product	Identity	Identifier for the product
*ProductDate	Date	Product Date
*ProductNegotiatedCost	Decimal	Product Negotiated Cost
Project	Identity	Identifier for the project
*Quantity	Decimal	Quantity
Resource	Identity	Identifier for the resource
RevAdj	Decimal	Revenue adjustment This property doesn't get used by Add/Update methods. Populated by GetById method.
RevMethDescription	String	Revenue Method Description This property doesn't get used by Add/Update methods. Populated by GetById method.
RevMethod	Integer	Revenue method This property doesn't get used by Add/Update methods. Populated by GetById method.
RevRec	Decimal	Revenue recognition This property doesn't get used by Add/Update methods. Populated by GetById method.

Property (*=required)	Type	Description
RevTent	Decimal	Rev Tent This property doesn't get used by Add/Update methods. Populated by GetById method.
RRUpdate	Boolean	RR Update This property doesn't get used by Add/Update methods. Populated by GetById method.
*StandardCost	Decimal	Standard Cost
WorkLocation	Identity	Identifier of the work location.
WorkLocationGroup	Identity	Identifier of the work location group
UpdatedBy	Signature	The updater of the record and when it was updated.

Related information

"ApiEngagement" on page 98

ApiEngProduct: Add

```
Public Overrides Function Add(Optional ByRef sId As String = "") As Int32
```

Purpose

Insert a new contract product record for a contract.

Parameters

Parameter (*=required)	Description
sId	ID of the contract product record.

Returns

0 = Success

Nonzero = Error

Remarks

If parameter sId is not provided, newly created EngProductId is assigned to property EngProductId

If parameter sId is provided newly created EngProductId is assigned to sId.

Value of the property Billed should be always False. If a value is not assigned, it takes a default value False.

Example

```
Dim myEngProduct As New ApiEngProduct
Dim iRet As Int32
Dim sId As String

myEngProduct.CPConnection = myCon
With myEngProduct
   .Engagement.Id = "{2E1D37B7-810F-43D6-ABAC-E624AB6F28EC}"
   .FixedCost = True
   .FixedCostOverride = True
   .Product.Id = "{705E4EBD-C568-496F-832B-4C7168D5D41F}"
   .StandardCost = 16022.0
   .Description = "Added from API"
   .NegotiatedPrice = 9999.0
   .ProductNegotiatedCost = 599.0
   .ProductDate = "12/05/2007"
   .Quantity = 1000
   .Resource.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
   .WorkLocationGroup.Id = "{333A206C-513E-4CAE-841A-EA73A82E8E81}"
   .WorkLocation.Id = "{0409A21B-EFB9-42C4-ADB3-617279F524EF}"
   iRet = .Add(sId)
End With
```

Related information

"ApiEngProduct" on page 186

ApiEngProduct: Delete

```
Public Overrides Function Delete(Optional ByVal sId As String = "") As Int32
```

Purpose

Delete a contract product for a contract

Parameters

Parameter (*=required)	Description
sId	ID of the contract product record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property EngProductId is used.

Example

```
Dim myEngProduct As New ApiEngProduct
Dim iRet As Int32
myEngProduct.CPConnection = myCon
iRet = myEngProduct.Delete("{719FB77B-C7F6-4A1A-A4B1-3A91F82B128E}")
```

Related information

"ApiEngProduct" on page 186

ApiEngProduct: Exists

```
Public Overrides Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Check whether this contract product exists or not

Parameters

Parameter (*=required)	Description
sId	ID of the contract product record.

Returns

True if the contract product exists, else False.

Remarks

If sId is not provided, property EngProductId is used.

Example

```
Dim myEngProduct As New ApiEngProduct
Dim bRet As Boolean
myEngProduct.CPConnection = myCon
bRet = myEngProduct.Exists("{719FB77B-C7F6-4A1A-A4B1-3A91F82B128E}")
```

Related information

"ApiEngProduct" on page 186

ApiEngProduct: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As DataSet
```

Purpose

Retrieve all product records for a contract

Parameters

Parameter (*=required)	Description
*sEngagementId	ID of the engagement

Returns

Engagement product records in a dataset upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

CustomerId, EngagementId, EngagementProductId, ProductId,

Comments, ProductDate, ProductName, ProductUnitCost, ProductCurrency,

Quantity, NegotiatedCost, ProductNegotiatedCost,

ProjectId, ProjectName, WorkLocationGroupId, WorkLocationGroup,

WorkLocationId, WorkLocation, ResourceId, ResourceName,

RevenueMethod, RevenueMethodDescription, ParentProductName, ParentDate,

ParentDescription.

Example

```
Dim myEngProduct As New ApiEngProduct
Dim dsRet As DataSet
myEngProduct.CPConnection = myCon
dsRet = myEngProduct.GetByEngagementId("{2E1D37B7-810F-43D6-ABAC-
E624AB6F28EC}")
```

Related information

"ApiEngProduct" on page 186

ApiEngProduct: GetById

```
Public Overrides Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Fills the object with contract product of the specified sId passed in the parameter or of the property EngProductId.

Parameters

Parameter (*=required)	Description
sId	ID of the contract product record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property EngProductId is used.

Example

```
Dim myEngProduct As New ApiEngProduct
Dim iRet As Int32
myEngProduct.CPConnection = myCon
iRet = myEngProduct.GetById("{719FB77B-C7F6-4A1A-A4B1-3A91F82B128E}")
```

Related information

"ApiEngProduct" on page 186

ApiEngProduct: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows As Int16 = -1) As
DataSet
```

Purpose

Retrieve contract product records.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).

Returns

A dataet of contract product records upon success and nothing in a dataset upon error.

Returns all rows when the method is called without a parameter.

Remarks

Returns the following columns:

CustomerId,EngagementId,EngagementProductId, ProductId, Description, ProductDate,ProductName, ProductUnitCost, ProductCurrency, Quantity, NegotiatedPrice, ProductNegotiatedCost, ProjectId,ProjectName, ShipToWorkLocationGroupId,WorkLocationGroup, ShipToWorkLocationId,WorkLocation, ResourceId, resourcename

Example

```
Dim myEngProduct As New ApiEngProduct
Dim dsRet As DataSet
myEngProduct.CPConnection = myCon
dsRet = myEngProduct.GetList()
```

Related information

"ApiEngProduct" on page 186

ApiEngProduct: GetProductPrice

```
Public Function GetProductPrice(ByVal sProductId As String, ByVal sCurrency As
String) As DataSet
```

Purpose

Retrieve product prices with specific currency for a product

Parameters

Parameter (*=required)	Description
*sProductId	Product ID
*sCurrency	Currency string. This is a three letter currency code which exists in the CurrencyDescription table.

Returns

A dataset of product price records in a dataset upon success and nothing in a dataset upon error.

Remarks

Returns the following columns: Price, ProductFixedUnitCost

Example

```
Dim myEngProduct As New ApiEngProduct
Dim dsRet As DataSet
myEngProduct.CPConnection = myCon
dsRet = myEngProduct.GetProductPrice("{59FB7D7D-6622-4317-A856-D44CBC859237}",
"USD")
```

Related information

"ApiEngProduct" on page 186

ApiEngProduct: GetProducts

```
Public Function GetProducts() As DataSet
```

Purpose

Retrieve contract enabled products

Parameters

None

Returns

A dataset of contract-enabled products upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

ProductCategoryId, ProductId, ProductName, ProductDescription, ProductStatus, ProductNumber, ProductOpenDate, ProductCloseDate, ProductManager, ProductUnitCost, ProductFixedUnitCost, ProductCurrency

Example

```
Dim myEngProduct As New ApiEngProduct
Dim dsRet As DataSet
myEngProduct.CPConnection = myCon
dsRet = myEngProduct.GetProducts()
```

Related information

"ApiEngProduct" on page 186

ApiEngProduct: GetProjects

```
Public Function GetProjects(ByVal sEngagementId As String) As DataSet
```

Purpose

Retrieve projects associated with an Engagement

Parameters

Parameter (*=required)	Description
*sEngagementId	Engagement ID

Returns

A dataset of projects for the contract upon success and nothing in a dataset upon error.

Remarks

Returns the following columns: ProjectId, Name, AlternateName

Example

```
Dim myEngProduct As New ApiEngProduct
Dim dsRet As DataSet
myEngProduct.CPConnection = myCon
dsRet = myEngProduct.GetProjects("{2E1D37B7-810F-43D6-ABAC-E624AB6F28EC}")
```

Related information

"ApiEngProduct" on page 186

ApiEngProduct: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Update a contract product record.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

- Get the contract product record to be updated by using GetById method
- Value of the property Billed should be always False. If value is not assigned, it takes a default value False.

Example

```
Dim myEngProduct As New ApiEngProduct
Dim iRet As Int32
```

```
myEngProduct.CPConnection = myCon
iRet = myEngProduct.GetById("{719FB77B-C7F6-4A1A-A4B1-3A91F82B128E}")

With myEngProduct
   .Description = "Updated from API"
   .NegotiatedPrice = 9999.0
   .ProductNegotiatedCost = 599.0
   .Quantity = 1500
   ...
End With
iRet = myEngProduct.Update()
```

Related information

"ApiEngProduct" on page 186

ApiEngProductSplitBillOverride

Purpose

Provides a collection object for split billing override information for contract products.

Namespace

Changepoint.ChangepointAPI2.ApiEngProductSplitBillOverride

Methods

None.

Properties

Property (*=required)	Type	Description
Amount if SplitByAmount = true, Amount is required if SplitByAmount = false, Amount is not required	Decimal	Amount of the product billing that is allocated to this contract. See the SplitByAmount and Percentage properties.
CreatedBy	Signature	The creator of the record and when it was created.
*Customer	Identity	Customer related to the contract product.

Property (*=required)	Type	Description
Deleted	Boolean	Flag that indicates if the contract product split bill override has been deleted.
*EngagementProduct	Identity	Engagement product.
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
*MainContact	Identity	Contact for the customer related to the contract product.
*MainEngagement	Identity	Engagement related to the contract product.
mVersion	String	Read-only. The current version number for internal version control.
Percentage if SplitByAmount = false, Percentage is required if SplitByAmount = true, Percentage is not required	Decimal	Percentage of total product billing that is calculated to be allocated to this contract.
*SplitByAmount	Boolean	Set to True to split total product billing by entering an amount (Amount property) for this contract product. Set to False to split total product billing by entering a percentage (Percentage property) for this contract product.
SplitEngagementId	String	Engagement related to the split contract product.
UpdatedBy	Signature	The updater of the record and when it was updated.

ApiEngProjectedResource

The ApiEngProjectedResource object allows users to retrieve, update and delete projected resources set at the contract level.

Namespace

Changepoint.ChangepointAPI2.ApiEngProjectedResource

Methods

Properties

Property (*=required)	Type	Description
*BillingOffice	Identity	Identifier of the billing office
*BillingRole	Identity	Identifier of the billing role
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
Comments	String	Comments.
*CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CPFunction	Identity	Identifier of the function

Property (*=required)	Type	Description
CreatedBy	Signature	The creator of the record and when it was created.
CurrentStatus	String	Current status
Deleted	Boolean	Flag that indicates if the contract projected resource has been deleted.
*Engagement	Identity	Identifier of the contract
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
EstimatedHours	Decimal	Estimated hours
*FinishDate	DateTime	Finish date
Initiatorid	String	Initiator ID
mVersion	String	Read-only. The current version number for internal version control.
*ProjectedResourceId	String	Projected Resource ID
RequestedResource	Identity	Identifier of requested resource, editable when resource request is enabled in the contract billing office.
RequestId	String	Request ID
RequestNumber	String	Request Number
Resource	Identity	Identifier of the Assigned resource, editable when resource request is disabled in the contract billing office.
SoftBooked	Boolean	Soft booked
*StartDate	DateTime	Start date
Task	Identity	Identifier of the task

Property (*=required)	Type	Description
TaskCreated	Boolean	Flag that indicates that a task has been assigned to the resource
UpdatedBy	Signature	The updater of the record and when it was updated.

Related information

"ApiEngagement" on page 98

ApiEngProjectedResource: Add

```
Public Overrides Function Add(Optional ByRef sId As String = "") As Int32
```

Purpose

Insert a new projected resource record for a contract.

Parameters

Parameter (*=required)	Description
sId	ID of the contract projected resource record.

Returns

0 = Success

Nonzero = Error

Remarks

- If parameter sId is not provided, newly created ProjectedResourceId is assigned to property ProjectedResourceId
- If parameter sId is provided newly created ProjectedResourceId is assigned to sId.
- Ensure all mandatory properties are set.

Example

```
Dim myEngProjRes As New ApiEngProjectedResource
Dim iRet As Int32
```

```
Dim sId As String

myEngProjRes.CPConnection = myCon
With myEngProjRes
   .BillingOffice.Id = "{DB587BF6-3665-11D4-8E13-0050DA7BA6B1}"
   .BillingRole.Id = "{6DE15305-3726-11D4-8E15-0050DA7BA6B1}"
   .Comments = "Added from COM API"
   .Engagement.Id = "{1EB38A5E-2E62-4C06-AC8D-0A621B1C8ABA}"
   .EstimatedHours = 120
   .FinishDate = CDate("12/31/2008")
   .Resource.Id = "{BFC84F4D-365D-11D4-8E13-0050DA7BA6B1}"
   .StartDate = CDate("10/20/2008")
   .TaskCreated = True
   .Task.Id = "{C595A18C-06B4-432C-A215-6B532D3465DA}"
   .SoftBooked = True
End With
iRet = myEngProjRes.Add(sId)
```

Related information

"ApiEngProjectedResource" on page 200

ApiEngProjectedResource: Delete

```
Public Overrides Function Delete(Optional ByVal sId As String = "") As Int32
```

Purpose

Delete a projected resource for a contract

Parameters

Parameter (*=required)	Description
sId	ID of the contract projected resource record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property ProjectedResourceId is used.

Example

```
Dim myEngProjRes As New ApiEngProjectedResource
Dim iRet As Int32

myEngProjRes.CPConnection = myCon
iRet = myEngProjRes.Delete("{4BAB1168-6D30-4897-9E7E-CFF52DE50544}")
```

Related information

"ApiEngProjectedResource" on page 200

ApiEngProjectedResource: Exists

```
Public Overrides Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Check whether this contract projected resource exists or not

Parameters

Parameter (*=required)	Description
sId	ID of the contract projected resource record.

Returns

True if the contract projected resource exists, else False.

Remarks

If sId is not provided, property ProjectedResourceId is used.

Example

```
Dim myEngProjRes As New ApiEngProjectedResource
Dim bRet As Boolean

myEngProjRes.CPConnection = myCon
myEngProjRes.ProjectedResourceId = "{4BAB1168-6D30-4897-9E7E-CFF52DE50544}"
bRet = myEngProjRes.Exists()
```

Related information

"ApiEngProjectedResource" on page 200

ApiEngProjectedResource: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As DataSet
```

Purpose

Retrieve all projected resource records for a contract

Parameters

Parameter (*=required)	Description
*sEngagementId	ID of the contract

Returns

A dataset of contract projected resource records upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

ProjectedResourceId, CustomerId, EngagementId, BillingOfficeId, BillingRoleId, FunctionId, RequestId, ResourceId, TaskCreated, TaskId, EstimatedHours, StartDate, FinishDate, Comments, SoftBooked, CreatedOn, CreatedBy, UpdatedOn, UpdatedBy, ResourceName, AlternateName, BillingRoleDescription, FunctionDescription, CurrentStatus, RequestNumber, InitiatorId, BillingOfficeName, RequestedResourceId

Example

```
Dim myEngProjRes As New ApiEngProjectedResource
Dim dsRet As DataSet

myEngProjRes.CPConnection = myCon
dsRet = myEngProjRes.GetByEngagementId("{06237CF4-C7D4-4E19-A549-
55039494ED8F}")
```

Related information

"ApiEngProjectedResource" on page 200

ApiEngProjectedResource: GetById

```
Public Overrides Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Fills the object with contract projected resource of the specified sId passed in the parameter or of the property ProjectedResourceId.

Parameters

Parameter (*=required)	Description
sId	ID of the contract projected resource record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property ProjectedResourceId is used.

Example

```
Dim myEngProjRes As New ApiEngProjectedResource
Dim iRet As Int32

myEngProjRes.CPConnection = myCon
iRet = myEngProjRes.GetById("{4BAB1168-6D30-4897-9E7E-CFF52DE50544}")
```

Related information

"ApiEngProjectedResource" on page 200

ApiEngProjectedResource: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows As Int16 = -1) As
DataSet
```

Purpose

Retrieve contract projected resource records.

Parameters

Parameter (*=required)	Description
iRetRows	Number of records to be returned. Default = -1 (return all).

Returns

A dataset of contract projected resource records upon success and nothing in a dataset upon error.

Returns all rows when method called without parameter.

Remarks

Returns the following columns:

EngagementId, CustomerId, ProjectedResourceId, BillingOfficeId, BillingRoleId, ResourceId

Example

```
Dim myEngProjRes As New ApiEngProjectedResource
Dim dsRet As DataSet

myEngProjRes.CPConnection = myCon
dsRet = myEngProjRes.GetList()
```

Related information

"ApiEngProjectedResource" on page 200

ApiEngProjectedResource: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Update a contract projected resource record.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

Get the contract projected resource record to be updated by using GetById method

Example

```
Dim myEngProjRes As New ApiEngProjectedResource
Dim iRet As Int32

myEngProjRes.CPConnection = myCon
iRet = myEngProjRes. GetById("{4BAB1168-6D30-4897-9E7E-CFF52DE50544}")
With myEngProjRes
   .Comments = "Updated from COM API"
   .EstimatedHours = 160
   .FinishDate = CDate("01/31/2009")
   .Resource.Id = "{BFC84F4D-365D-11D4-8E13-0050DA7BA6B1}"
   .StartDate = CDate("11/01/2008")
   ...
End With
iRet = myEngProjRes.Update()
```

Related information

"ApiEngProjectedResource" on page 200

ApiEngRequestProcessingRule

The ApiEngRequestProcessingRule object allows users to retrieve, delete and update request billing types and billing rules for contracts.

Namespace

Changepoint.ChangepointAPI2.ApiEngRequestProcessingRule

Methods

Properties

Property (*=required)	Type	Description
AvailToEnterprise	Boolean	Flag that indicates if it is available to enterprise.
AvailToGuest	Boolean	Flag that indicates if it is available to guest
BillingRole	Identity	Identifier of the billing role. This billing role should have already set as one of the contract billing rates for the specific contract.
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
*CPConnection	ApiConnection	Write only. Connection object that must be assigned before any action to database.

Property (*=required)	Type	Description
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CreatedBy	Signature	The creator of the record and when it was created.
Customer	Identity	Identifier of the customer.
Deleted	Boolean	Flag that indicates the record to be deleted.
*Engagement	String	Identifier of the Contract
*EngRequestBillingRuleId	String	Contract request processing rule ID
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
FixedFee	Identity	Identifier of the fixed fee.
mVersion	String	Read-only. The current version number for internal version control.
*RequestBillingType	Int16	Request billing type Value range: • Billable = 0 • Unbillable = 1 • SLA-based = 2 • Effort and product = 3 • Fixed Fee = 4
RequestBillingTypeDescription	String	Request billing type name
RequestType	String	Request type.
RequestTypeDescription	String	Request type name.
SLARequiredToEnterprise	Boolean	Flag that indicates if SLA is required for Enterprise users to create a request.

Property (*=required)	Type	Description
SLARequiredToGuest	Boolean	Flag that indicates if SLA is required for Client Portal users to create a request.
TrackSLA	Boolean	Flag of track SLA.
UpdatedBy	Signature	The updater of the record and when it was updated.

Related information

"ApiEngagement" on page 98

"ApiEngRequestProcessingRule XML" on page 212

ApiEngRequestProcessingRule XML

```
<root>
<engrequestbillingrule>
   <deleted/>
   <engrequestbillingruleid/>
   <requesttype/>
   <requestbillingtype/>
   <billingroleid/>
   <fixedfeeid/>
   <availtoguest/>
   <slarequiredtoguest/>
   <tracksla/>
   <availtoenterprise/>
   <slarequiredtoenterprise/>
</engrequestbillingrule>
</root>
```

Comments

Not available

Example

Not available

Related information

"ApiEngRequestProcessingRule" on page 209

ApiEngRequestProcessingRule: Add

```
Public Overrides Function Add(Optional ByRef sId As String = "") As Int32
```

Purpose

Insert a new contract request processing rule record for a contract.

Parameters

Parameter (*=required)	Description
sId	ID of the contract request processing rule record.

Returns

0 = Success

Nonzero = Error

Remarks

- If parameter sId is not provided, newly created EngRequestBillingRuleId is assigned to property EngRequestBillingRuleId
- If parameter sId is provided newly created EngRequestBillingRuleId is assigned to sId.
- Ensure all mandatory properties are set.
- When there is no existing request processing rule, error message -10 is written into the log file. This message can be ignored

Example

```
Dim myEngRBRule As New ApiEngRequestProcessingRule
Dim iRet As Int32
Dim sId As String
myEngRBRule.CPConnection = myCon
With myEngRBRule
   .AvailtoGuest = True
   .AvailToEnterprise = True
   .Engagement.Id = "{70852075-3A41-4406-955E-B96C777AB431}"
   .RequestBillingType = 0
   .RequestType = "CR"
   .SLARequiredToEnterprise = True
   .TrackSLA = True
```

```
   iRet =.Add(sId)
End With
```

Related information

"ApiEngRequestProcessingRule" on page 209

ApiEngRequestProcessingRule: Delete

```
Public Overrides Function Delete(Optional ByVal sId As String = "") As Int32
```

Purpose

Delete a contract request processing rule record for a contract

Parameters

Parameter (*=required)	Description
sId	ID of the contract request processing rule record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property EngRequestBillingRuleId is used.

Example

```
Dim myEngRBRule As New ApiEngRequestProcessingRule
Dim iRet As Int32
myEngRBRule.CPConnection = myCon
iRet = myEngRBRule.Delete("{0633FFB7-97E0-42F2-B817-915468319CA9}")
```

Related information

"ApiEngRequestProcessingRule" on page 209

ApiEngRequestProcessingRule: Exists

```
Public Overrides Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Check whether this contract request processing rule exists or not

Parameters

Parameter (*=required)	Description
sId	ID of the contract request processing rule record.

Returns

True if the contract request processing rule exists, else False.

Remarks

If sId is not provided, property EngRequestBillingRuleId is used.

Example

```
Dim myEngRBRule As New ApiEngRequestProcessingRule
Dim bRet As Boolean
With myEngRBRule
   .CPConnection = myCon
   .EngRequestBillingRuleId = "{4BAB1168-6D30-4897-9E7E-CFF52DE50544}"
   bRet = .Exists()
End With
```

Related information

"ApiEngRequestProcessingRule" on page 209

ApiEngRequestProcessingRule: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As DataSet
```

Purpose

Retrieve all request processing rule records for a contract

Parameters

Parameter (*=required)	Description
*sEngagementId	ID of the contract

Returns

A dataset of contract request processing rule records upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

EngRequestBillingRuleId, EngagementId, CustomerId, RequestBillingType, RequestBillingTypeDescription, RequestType, RequestTypeDescription, BillingRoleId, BillingRole.Name, FixedFeeId, FixedFee.Name, AvailToEnterprise, AvailtoGuest, SLARequiredToEnterprise, SLARequiredToGuest, TrackSLA, CreatedBy, CreatedOn, UpdatedBy, UpdatedOn

Example

```
Dim myEngRBRule As New ApiEngRequestProcessingRule
Dim dsRet As DataSet
myEngRBRule.CPConnection = myCon
dsRet = myEngRBRule.GetByEngagementId("{70852075-3A41-4406-955E-
B96C777AB431}")
```

Related information

"ApiEngRequestProcessingRule" on page 209

ApiEngRequestProcessingRule: GetById

```
Public Overrides Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Fills the object with contract request processing rule of the specified sId passed in the parameter or of the property EngRequestBillingRuleId.

Parameters

Parameter (*=required)	Description
sId	ID of the contract request processing rule record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property EngRequestBillingRuleId is used.

Example

```
Dim myEngRBRule As New ApiEngRequestProcessingRule
Dim iRet As Int32
myEngRBRule.CPConnection = myCon
iRet = myEngRBRule.GetById("{4BAB1168-6D30-4897-9E7E-CFF52DE50544}")
```

Related information

"ApiEngRequestProcessingRule" on page 209

ApiEngRequestProcessingRule: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows As Int16 = -1) As
DataSet
```

Purpose

Retrieve contract request processing rule records.

Parameters

Parameter (*=required)	Description
iRetRows	Number of records to be returned. Default = -1 (return all).

Returns

A dataset of contract request processing rule records upon success and nothing in a dataset upon error.

Returns all rows when method called without parameter.

Remarks

Returns the following columns:

EngagementId, EngagementName, CustomerId, EngRequestBillingRuleId, RequestType, RequestTypeDescription, RequestBillingType, RequestBillingTypeDescription

Example

```
Dim myEngRBRule As New ApiEngRequestProcessingRule
Dim dsRet As DataSet
myEngRBRule.CPConnection = myCon
dsRet = myEngRBRule.GetList()
```

Related information

"ApiEngRequestProcessingRule" on page 209

ApiEngRequestProcessingRule: GetRequestTypes

```
Public Function GetRequestTypes(Optional ByVal sRequestTypeCode As String =
"") As DataSet
```

Purpose

Retrieve Request Types

Parameters

Parameter (*=required)	Description
sRequestTypeCode	Request type code

Returns

A dataset of request type records upon success and nothing in a dataset upon error.

Remarks

Returns all request types when parameter sRequestTypeCode is not provided.

Returns the following columns: Code, Description, AvailToGuest

Example

```
Dim myEngRBRule As New ApiEngRequestProcessingRule
Dim dsRet As DataSet
myEngRBRule.CPConnection = myCon
dsRet = myEngRBRule.GetRequestTypes()
```

Related information

"ApiEngRequestProcessingRule" on page 209

ApiEngRequestProcessingRule: GetXmlByEngagementId

```
Public Function GetXmlByEngagementId(ByVal sEngagementId As String) As String
```

Purpose

Retrieve request processing rules for the specified contract

Parameters

Parameter (*=required)	Description
*sEngagementId	Engagement ID

Returns

Engagement request processing rules in an XML document string.

Remarks

If there are no contract request processing rules for the specified contract, an empty string is returned.

Example

```
Dim myEngRBRule As New ApiEngRequestProcessingRule
Dim sRet As String = String.Empty
myEngRBRule.CPConnection = myCon
sRet = myEngRBRule.GetXmlByEngagementId("{70852075-3A41-4406-955E-
B96C777AB431}")
```

Related information

"ApiEngRequestProcessingRule" on page 209

"ApiEngRequestProcessingRule XML" on page 212

"ApiEngRequestProcessingRule: GetXMLStructure" on page 219

ApiEngRequestProcessingRule: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Retrieve the XML structure of the contract request processing rule

Parameters

None

Returns

An XML string of the the contract request processing rules

Remarks

None

Example

```
Dim myEngRBRule As New ApiEngRequestProcessingRule
Dim sXmlStructure As String = String.Empty
myEngRBRule.CPConnection = myCon
sXmlStructure = myEngRBRule.GetXMLStructure()
```

Related information

"ApiEngRequestProcessingRule" on page 209

ApiEngRequestProcessingRule: SaveByXml

```
Public Function SaveByXml(ByVal sEngagementId As String, ByVal sXmlRPRules As
String) As Int32
```

Purpose

Update the request processing rule information of an contract

Parameters

Parameter (*=required)	Description
*sEngagementId	Engagement ID
*sXmlRPRules	Request processing rules in XML string.

Returns

0 = Success

Nonzero = Error

Remarks

- It is recommended to use the GetXmlByEngagementId or GetXMLStructure methods to get correct XML format of parameter sXmlRPRules.
- The <engrequestbillingrule /> tag represents one request processing rule, so multiple entries can be expected. All these entries must belong to the contract specified as parameter sEngagementId.
- Includes ADD, UPDATE and DELETE functionalities. When the value of the deleted tag is true, it indicates DELETE; when the value of the engrequestbillingruleid tag is empty, it indicates ADD, when the value of the engrequestbillingruleid tag is a GUID, it indicates UPDATE.

Example

```
Dim myEngRBRule As New ApiEngRequestProcessingRule
Dim iRet As Int32
Dim sXmlRPRules As String

'sXmlRPRules contains 3 Request Processing rules for update,delete and add
sXmlRPRules =
"<root>
<engrequestbillingrule>
   <deleted>0</deleted>
   <engrequestbillingruleid>{20A95341-AF00-4FFE-BC65-
11E5093B86E1}</engrequestbillingruleid>
   <requesttype></requesttype>
   <requestbillingtype>0</requestbillingtype>
   <billingroleid>{00000000-0000-0000-0000-000000000000}</billingroleid>
   <fixedfeeid>{00000000-0000-0000-0000-000000000000}</fixedfeeid>
   <availtoguest>0</availtoguest>
   <slarequiredtoguest>0</slarequiredtoguest>
   <tracksla>0</tracksla><availtoenterprise>0</availtoenterprise>
   <slarequiredtoenterprise>0</slarequiredtoenterprise>
</engrequestbillingrule>
<engrequestbillingrule>
   <deleted>0</deleted>
   <engrequestbillingruleid></engrequestbillingruleid>
   <requesttype>BR</requesttype>
   <requestbillingtype>0</requestbillingtype>
   <billingroleid></billingroleid><fixedfeeid></fixedfeeid>
   <availtoguest>1</availtoguest>
   <slarequiredtoguest>0</slarequiredtoguest>
   <tracksla>1</tracksla>
   <availtoenterprise>1</availtoenterprise>
   <slarequiredtoenterprise>1</slarequiredtoenterprise>
```

```
</engrequestbillingrule>
<engrequestbillingrule>
   <deleted>0</deleted>
   <engrequestbillingruleid>{F768D80C-4061-414B-AC85-
76B803A8CC01}</engrequestbillingruleid>
   <requesttype>DR</requesttype>
   <requestbillingtype>0</requestbillingtype>
   <billingroleid>{00000000-0000-0000-0000-000000000000}</billingroleid>
   <fixedfeeid>{00000000-0000-0000-0000-000000000000}</fixedfeeid>
   <availtoguest>1</availtoguest>
   <slarequiredtoguest>0</slarequiredtoguest>
   <tracksla>1</tracksla>
   <availtoenterprise>1</availtoenterprise>
   <slarequiredtoenterprise>1</slarequiredtoenterprise>
</engrequestbillingrule>
<engrequestbillingrule>
   <deleted>1</deleted>
   <engrequestbillingruleid>{D127FF27-49B4-47CC-8018-
18554E1F1D0D}</engrequestbillingruleid>
   <requesttype>DR</requesttype>
   <requestbillingtype>2</requestbillingtype>
   <billingroleid>{00000000-0000-0000-0000-000000000000}</billingroleid>
   <fixedfeeid>{00000000-0000-0000-0000-000000000000}</fixedfeeid>
   <availtoguest>1</availtoguest>
   <slarequiredtoguest>0</slarequiredtoguest>
   <tracksla>1</tracksla>
   <availtoenterprise>1</availtoenterprise>
   <slarequiredtoenterprise>1</slarequiredtoenterprise>
</engrequestbillingrule>
</root>"

myEngRBRule.CPConnection = myCon
iRet = myEngRBRule.SaveByXml("{E90A4835-786F-4AFE-89ED-88A959608277}",
sXmlRPRules)
```

Related information

"ApiEngRequestProcessingRule" on page 209

"ApiEngRequestProcessingRule XML" on page 212

"ApiEngRequestProcessingRule: GetXmlByEngagementId" on page 219

"ApiEngRequestProcessingRule: GetXMLStructure" on page 219

ApiEngRequestProcessingRule: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Update a contract request processing rule record.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

It is recommended to get the contract request processing record which is to be updated by using GetById method

Example

```
Dim myEngRBRule As New ApiEngRequestProcessingRule
Dim iRet As Int32
myEngRBRule.CPConnection = myCon
iRet = myEngRBRule.GetById("{20A95341-AF00-4FFE-BC65-11E5093B86E1}")
With myEngRBRule
   .RequestType = "BR"
   .RequestBillingType = 1
   ...
   iRet =.Update()
End With
```

Related information

"ApiEngRequestProcessingRule" on page 209

ApiEngRequestSLA

The ApiEngRequestSLA object represents all request support information created for a contract.

Namespace

Changepoint.ChangepointAPI2.ApiEngRequestSLA

Methods

Properties

Property (*=required)	Type	Description
By24x7	Boolean	Flag that indicates whether the SLA provides support 24 hours, every day.
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
*CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CreatedBy	Signature	The creator of the record and when it was created.
Customer	Identity	Identifier of the customer.
Deleted	Boolean	Flag that indicates if the object has been deleted
EffectiveDate	Date	SLA Effective Date

Property (*=required)	Type	Description
EndDay	Int16	Use the number below for the EndDay value: • 1 - Sunday • 2 - Monday • 3 - Tuesday • 4 - Wednesday • 5 - Thursday • 6 - Friday • 7 - Saturday
EndTime	String	End time, for example, 24:00
*Engagement	Identity	Identifer of the contract
*EngagementSLAId	String	Engagement Request SLA ID Required for GetById and Update methods
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
EscalationHours	Decimal	Escalation hours
ExpiryDate	Date	Expiry date
Invoiced	Boolean	Flag that indicates if contract request SLA has been invoiced
License	String	License
mVersion	String	Read-only. The current version number for internal version control.
Onsite	Boolean	Flag that indicates if it supports onsite
Priority	Identity	Identifier of the priority
*Product	Identity	Identifier of the product
RequestType	String	Request type.

Property (*=required)	Type	Description
RequestTypeName	String	Request type name.
ResolutionHours	Decimal	Resolution hours
ResponseHours	Decimal	Response hours
StartDay	Int16	Use the number below as StartDay value: 1 - Sunday 2 - Monday 3 - Tuesday 4 - Wednesday 5 - Thursday 6 - Friday 7 - Saturday
StartTime	String	Start time, e.g. 09:00
TimeZone	String	Time zone.
TimeZoneName	String	Full name of the time zone
UpdatedBy	Signature	The updater of the record and when it was updated.

Related information

"ApiEngagement" on page 98

ApiEngRequestSLA: Add

```
Public Overrides Function Add(Optional ByRef sId As String = "") As Int32
```

Purpose

Insert a new contract request SLA record for a contract.

Parameters

Parameter (*=required)	Description
sId	ID of the contract SLA record.

Returns

0 = Success

Nonzero = Error

Remarks

- If parameter sId is not provided, newly created engagementSLAId is assigned to property engagementSLAId
- If parameter sId is provided, newly created engagementSLAId is assigned to sId.
- Use numeric representation of the weekday for StartDay and EndDay. See the object properties for detailed information.
- Ensure all mandatory properties are set.

Example

```
Dim myEngReqSLA As New ApiEngRequestSLA
Dim iRet As Int32
Dim sId As String

myEngReqSLA.CPConnection = myCon
With myEngReqSLA
   .By24x7 = False
   .EffectiveDate = CDate("10/27/2008")
   .EndDay = 2
   .EndTime = "18:00"
   .Engagement.Id = "{A617568D-DFBE-444D-84D2-8AA3B94FECC7}"
   .EscalationHours = 4
   .ExpiryDate = CDate("10/27/2009")
   .Onsite = False
   .Priority.Id = "{7D72E3F4-E827-4F0E-8AED-798D31DD6162}"
   .Product.Id = "{EF83EF72-384D-42AC-8317-32116C583082}"
   .RequestType = "BR"
   .ResolutionHours = 24
   .ResponseHours = 8
   .StartDay = 6
   .StartTime = "9:00"
   .TimeZone = "EST"
End With
iRet = myEngReqSLA.Add(sId)
```

Related information

"ApiEngRequestSLA" on page 223

ApiEngRequestSLA: Delete

```
Public Overrides Function Delete(Optional ByVal sId As String = "") As Int32
```

Purpose

Deletes a contract SLA for a contract

Parameters

Parameter (*=required)	Description
sId	ID of the contract SLA record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property EngagementSLAId is used.

Example

```
Dim myEngReqSLA As New ApiEngRequestSLA
Dim iRet As Int32

myEngReqSLA.CPConnection = myCon
iRet = myEngReqSLA.Delete("{C26D51B4-4C22-440A-A21A-4C88FB445BEC}")
```

Related information

"ApiEngRequestSLA" on page 223

ApiEngRequestSLA: Exists

```
Public Overrides Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Verifies whether this contract request SLA exists or not

Parameters

Parameter (*=required)	Description
sId	ID of the contract SLA record to verify

Returns

True if the contract request SLA exists, else False.

Remarks

If sId is not provided, property EngagementSLAId is used.

Example

```
Dim myEngReqSLA As New ApiEngRequestSLA
Dim bRet As Boolean

myEngReqSLA.CPConnection = myCon
myEngReqSLA.EngagementSLAId = "{C26D51B4-4C22-440A-A21A-4C88FB445BEC}"
bRet = myEngReqSLA.Exists ()
```

Related information

"ApiEngRequestSLA" on page 223

ApiEngRequestSLA: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As DataSet
```

Purpose

Retrieve all contract request SLA records for a contract

Parameters

Parameter (*=required)	Description
*sEngagementId	ID of the contract

Returns

A dataset of contract request SLA records upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

CustomerId, EngagementId, EngagementRequestSLAId, ProductId, ProductName,

By24x7, CreatedBy, CreatedOn, EffectiveDate, EndDay, EndTime, EscalationHours,

ExpiryDate, License, Onsite, Priority, PriorityName, RequestType, RTDescription, ResolutionHours, StartDay, StartTime, TimeZone, TimeZoneName, UpdatedBy, UpdatedOn

Example

```
Dim myEngReqSLA As New ApiEngRequestSLA
Dim dsRet As DataSet

myEngReqSLA.CPConnection = myCon
dsRet = myEngReqSLA.GetByEngagementId("{06237CF4-C7D4-4E19-A549-
55039494ED8F}")
```

Related information

"ApiEngRequestSLA" on page 223

ApiEngRequestSLA: GetById

```
Public Overrides Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Fills the object with contract request SLA of the specified sId passed in the parameter or of the property EngagementSLAId.

Parameters

Parameter (*=required)	Description
sId	Engagement SLA ID

Returns

0 = Success

Nonzero = Error

Remarks

If the optional parameter sId is not provided, EngagementSLAId is used.

Example

```
Dim myEngReqSLA As New ApiEngRequestSLA
Dim iRet As Int32

myEngReqSLA.CPConnection = myCon
iRet = myEngReqSLA.GetById("{C26D51B4-4C22-440A-A21A-4C88FB445BEC}")
```

Related information

"ApiEngRequestSLA" on page 223

ApiEngRequestSLA: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows As Int16 = -1) As
DataSet
```

Purpose

Retrieves contract request SLA records.

Parameters

Parameter (*=required)	Description
iRetRows	Number of records to be returned. Default = -1 (return all).

Returns

A dataset of contract request SLA records upon success and nothing in a dataset upon error.

Returns all rows when method called without parameter.

Remarks

Returns the following columns:

CustomerId,EngagementId,EngagementName,EngagementSLA, ProductId,ProductName

Example

```
Dim myEngReqSLA As New ApiEngRequestSLA
Dim dsRet As DataSet
```

```
myEngReqSLA.CPConnection = myCon
dsRet = myEngReqSLA.GetList()
```

Related information

"ApiEngRequestSLA" on page 223

ApiEngRequestSLA: GetProducts

```
Public Function GetProducts() As DataSet
```

Purpose

Retrieves product records

Parameters

None

Returns

A dataset of product records upon success and nothing in a dataset upon error.

Remarks

Returns the following columns:

ProductCategoryId, ProductId, ProductName, ProductNumber, ProductOpenDate, ProductCloseDate, ProductUnitCost, ProductCurrency

Example

```
Dim myEngReqSLA As New ApiEngRequestSLA
Dim dsRet As DataSet

myEngReqSLA.CPConnection = myCon
dsRet = myEngReqSLA.GetProducts()
```

Related information

"ApiEngRequestSLA" on page 223

ApiEngRequestSLA: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Update a contract request SLA record.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

- Get the contract request SLA record to be updated by using GetById method
- Use the numeric representation of the weekdays for StartDay and EndDay. See the object properties for detailed information.

Example

```
Dim myEngReqSLA As New ApiEngRequestSLA
Dim iRet As Int32

myEngReqSLA.CPConnection = myCon
iRet = myEngReqSLA.GetById("{81F1EB3D-18BE-4E61-A5BA-6A279E61ABBC}")
With myEngReqSLA
   .EscalationHours = 24
   .ResolutionHours = 48
   .ResponseHours = 8
   .StartDay = 6
   ...
End With
iRet = myEngReqSLA.Update()
```

Related information

"ApiEngRequestSLA" on page 223

"ApiEngRequestSLA: GetById " on page 230

ApiEngRevRec

The ApiEngRevRec object represents the revenue recognition information for a contract.

Namespace

Changepoint.ChangepointAPI2.ApiEngRevRec

Methods

Properties

Property (*=required)	Type	Description
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
CostCenterType	Boolean	• 1 – Contract Manager • 0 – Billing Office
*CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CreatedBy	Signature	The creator of the record and when it was created.
CRGLTime	String	Revenue Credit Account for Time-Based

Property (*=required)	Type	Description
Deleted	Boolean	Flag that indicates if the object has been deleted
DRGLTime	String	Revenue Debit Account for Time-Based
*EngagementId	String	Contract ID
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
LM	Decimal	Labor Multiplier
mVersion	String	Read-only. The current version number for internal version control.
RevRecMethod	CPRevRecMethod	Revenue Recognition Method Value range: • cpRR_TimeAndExpenseBased = 0 • cpRR_TimeBasedOnly = 1 • cpRR_ExpenseBasedOnly = 2 • cpRR_ DisableRRForTimeAndExpenses = 3
RRAF	Decimal	Revenue Recognition Adjustment Factor
RRAfterDate	DateTime	Transactions after this date
RRDel	Boolean	Flag that indicates if Deliverable/Milestone-based is enabled
RROn	Boolean	Flag that indicates if revenue recognition is enabled
TotRev	Decimal	Total contract amount
UpdatedBy	Signature	The updater of the record and when it was updated.

Related information

"ApiEngagement" on page 98

ApiEngRevRec: GetById

```
Public Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Fills the object with contract revenue recognition information of the specified EngagementId passed in the parameter or in the property.

Parameters

Parameter (*=required)	Description
sId	Optional Contract ID

Returns

0 = Success

Nonzero = Error

Remarks

If optional parameter sId is not provided, Contract.Id is used.

Example

```
Dim myEngRevRec As New ApiEngRevRec
Dim iRet As Int32
myEngRevRec.CPConnection = myCon
iRet = myEngRevRec.GetById("{06237CF4-C7D4-4E19-A549-55039494ED8F}")
```

Related information

"ApiEngRevRec" on page 233

ApiEngRevRec: GetDeliverableAmount

```
Public Function GetDeliverableAmount(ByVal sId As String) As Decimal
```

Purpose

Get a calculated deliverable amount for an Engagement

Parameters

Parameter (*=required)	Description
*sId	Engagement ID

Returns

The total of the deliverable (contract product and fixed fee schedule) amount of the contract.

Remarks

None

Example

```
Dim myEngRevRec As New ApiEngRevRec
Dim dRet As Decimal
Dim myEngRevRec As New ApiEngRevRec.CPConnection = myCon
dRet = myEngRevRec.GetDeliverableAmount("{06237CF4-C7D4-4E19-A549-
55039494ED8F}")
```

Related information

"ApiEngRevRec" on page 233

ApiEngRevRec: Update

```
Public Function Update() As Int32
```

Purpose

Update contract revenue recognition information.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

Get the contract revenue recognition information to be updated by using GetById method

Example

```
Dim myEngRevRec As New ApiEngRevRec
Dim iRet As Int32

myEngRevRec.CPConnection = myCon
iRet = myEngRevRec.GetById("{06237CF4-C7D4-4E19-A549-55039494ED8F}")

With myEngRevRec
   .RROn = True
   .RevRecMethod = CPRevRecMethod.cpRR_DisableRRForTimeAndExpenses
   .CRGLTime = "{4C7FB5EC-184D-4014-BB68-893E256911D8}"
   ...
End With
iRet = myEngRevRec.Update()
```

Related information

"ApiEngRevRec" on page 233

"ApiEngRevRec: GetById" on page 236

ApiEngSplitBillingRule

The ApiEngSplitBillingRule object allows users to retrieve and update contracts with split billing rules.

Namespace

Changepoint.ChangepointAPI2.ApiEngSplitBillingRule

Methods

Properties

Property (*=required)	Type	Description
*CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CreatedBy	Signature	The creator of the record and when it was created.
*Customer	Identity	Identifer of the customer
Deleted	Boolean	Flag that indicates to delete split contract
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
*MainContact	Identity	Identifer of the main contact
*MainEngagement	Identity	Identifer of the main contract
mVersion	String	Read-only. The current version number for internal version control.
*Percentage	Decimal	Split percentage
SplitEngagementId	String	Split contract ID
UpdatedBy	Signature	The updater of the record and when it was updated.

Related information

"ApiEngagement" on page 98

"ApiEngSplitBillingRule XML" on page 239

ApiEngSplitBillingRule XML

```
<root>
<engsplitbillingrule>
   <splitengagementid/>
```

```
    <customerid/>
    <customername/>
    <maincontactid/>
    <maincontactname/>
    <percentage/>
    <deleted/>
</engsplitbillingrule>
</root>
```

Comments

Not available

Example

Not available

Related information

"ApiEngSplitBillingRule" on page 238

ApiEngSplitBillingRule: GetById

```
Public Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Fills the object with contract split billing rule of the specified EngagementId passed in the parameter or in the property.

Parameters

Parameter (*=required)	Description
sId	Contract ID

Returns

0 = Success

Nonzero = Error

Remarks

If the optional parameter sId is not provided, Contract.Id is used.

Example

```
Dim myEngSplitBillingRule As New ApiEngSplitBillingRule
Dim iRet As Int32
myEngSplitBillingRule.CPConnection = myCon
iRet = myEngSplitBillingRule.GetById("{5A390603-6748-454E-8AD3-FC8D65FABA88}")
```

Related information

"ApiEngSplitBillingRule" on page 238

ApiEngSplitBillingRule: GetByMainEngagementId

```
Public Function GetByMainEngagementId(ByVal sEngagementId As String) As String
```

Purpose

Retrieves the contract split billing rule for the specified contract

Parameters

Parameter (*=required)	Description
*sEngagementId	Contract ID

Returns

The Contract Split Billing Rule in an XML document string.

Remarks

If there is no split billing rule for the specified contract, an empty string is returned.

Example

```
Dim myEngSplitBillingRule As New ApiEngSplitBillingRule
Dim sRet As String = String.Empty
myEngSplitBillingRule.CPConnection = myCon
sRet = myEngSplitBillingRule.GetByMainEngagementId("{73501433-32EB-491C-B7E4-
F03E6824659D}")
```

Related information

"ApiEngSplitBillingRule" on page 238

"ApiEngSplitBillingRule XML" on page 239

ApiEngSplitBillingRule: GetSplitEngagementId

```
Public Function getSplitEngagementId(ByVal sEngagementId As String, ByVal
sCustomerId As String) As String
```

Purpose

Retrieves the contract split billing rule for the specified contract

Parameters

Parameter (*=required)	Description
*sEngagementId	Contract ID
*sCustomerId	Customer ID

Returns

The Split Contract Id.

Remarks

None.

Example

None.

Related information

"ApiEngSplitBillingRule" on page 238

"ApiEngSplitBillingRule XML" on page 239

ApiEngSplitBillingRule: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Retrieves the XML structure of the contract split rule

Parameters

None

Returns

XML string of the contract split rule

Remarks

None

Example

```
Dim myEngSplitBillingRule As New ApiEngSplitBillingRule
Dim sXmlStructure As String = String.Empty
myEngSplitBillingRule.CPConnection = myCon
sXmlStructure = myEngSplitBillingRule.GetXMLStructure()
```

Related information

"ApiEngSplitBillingRule" on page 238

"ApiEngSplitBillingRule XML" on page 239

ApiEngSplitBillingRule: Save

```
Public Function Save(ByVal sMainEngagementId As String, ByVal
sXmlEngSplitBillingRules As String) As Int32
```

Purpose

Updates the general information of split billing rule of a contract

Parameters

Parameter (*=required)	Description
*sMainEngagementId	Main contract ID.
*sXmlEngSplitBillingRules	ApiEngSplitBillingRule objects in XML format string.

Returns

0 = Success

Nonzero = Error

Remarks

- Parameter sXmlEngSplitBillingRules must contain the split billing rules that belong to the parameter sMainEngagementId. In other words, sMainEngagementId is the main contract of the split contracts in the sXmlEngSplitBillingRules.
- It is recommended to use the GetByMainEngagement method to get the correct XML format of parameter sXmlEngSplitBillingRules.
- Includes ADD, UPDATE and DELETE functionalities. When the value of the deleted tag is true, it indicates DELETE; when the value of the splitengagementid tag is empty, it indicates ADD, when the value of the splitengagementid tag is a GUID, it indicates UPDATE.

Example

```
Dim myEngSplitBillingRule As New ApiEngSplitBillingRule
Dim sxmlEngSplitBillingRule As String
Dim iRet As Int32

sxmlEngSplitBillingRule =
"<root>
<engsplitbillingrule>
   <splitengagementid>{5A390603-6748-454E-8AD3-
FC8D65FABA88}</splitengagementid>
   <customerid>{E6367714-DBD2-4716-8097-74F8164618DA}</customerid>
   <customername>SHK Trading Secondary Co LTD</customername>
   <maincontactid>{59B84C7E-9033-4771-AD53-80EB3EA8B599}</maincontactid>
   <maincontactname>Jane Smith</maincontactname>
   <percentage>50</percentage>
   <deleted>0</deleted>
</engsplitbillingrule>
<engsplitbillingrule>
   <splitengagementid></splitengagementid>
   <customerid>{40C01201-37EC-47E1-8226-F505760C42F9}</customerid>
   <customername>SHK Trading Co LTD</customername>
   <maincontactid>{BFA422F0-E7EB-4F9B-88D3-E487002E50BB}</maincontactid>
   <maincontactname>Bob Hardy</maincontactname>
   <percentage>50</percentage>
   <deleted>0</deleted>
</engsplitbillingrule>
<engsplitbillingrule>
   <splitengagementid>{2C32B0FB-563C-4CA8-AACC-
F673E2068A60}</   splitengagementid>
```

```
    <customerid>{C2932E66-5B75-46E3-87BE-CDEDA686048B}</customerid>
    <customername>SHK Trading Third Co LTD</customername>
    <maincontactid>{0749DDA0-5A1D-4A02-90A2-00004C65F3AC}</maincontactid>
    <maincontactname>Alice Monroe</maincontactname>
    <percentage>0</percentage>
    <deleted>1</deleted>
</engsplitbillingrule>
</root>"

Set myEngSplitBillingRule.CPConnection = myCon
iRet = myEngSplitBillingRule.Save("{73501433-32EB-491C-B7E4-F03E6824659D}",
sxmlEngSplitBillingRule)
```

Related information

"ApiEngSplitBillingRule" on page 238

"ApiEngSplitBillingRule XML" on page 239

"ApiEngSplitBillingRule: GetByMainEngagementId" on page 241

ApiEngWorkCode

The ApiEngWorkCode object represents the work code information for a contract.

Namespace

Changepoint.ChangepointAPI2.ApiEngWorkCode

Methods

Properties

Property (*=required)	Type	Description
AllWorkCodes	Boolean	Include all work codes
AllowRTCodeOverride	Boolean	Allow request time work code override

Property (*=required)	Type	Description
BypassMetadataCheck	Byte	Determines the level of Metadata checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
*CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CreatedBy	Signature	The creator of the record and when it was created.
DefaultWorkCode	Identity	Identifier of the default work code for a contract
DefaultWorkCodeCategory	Identity	Identifer of the default work code category for a contract
Deleted	Boolean	Flag that indicates whether the record has been deleted.
*Engagement	Identity	Identifer for the contract
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
mVersion	String	Read-only. The current version number for internal version control.

Property (*=required)	Type	Description
sXmlWorkCodes	String	XML document that includes work codes. See Remarks of GetXmlStructure method for the XML structure.
UpdatedBy	Signature	Details of when the record was updated and by whom.

Related information

"ApiEngagement" on page 98

"ApiEngWorkCode XML" on page 247

ApiEngWorkCode XML

```
<root>
<engworkcode>
   <workcodecategoryid/>
   <workcodecategory/>
   <workcodeid/>
   <workcode/>
   <selected/>
</engworkcode>
</root>
```

Comments

Not available

Example

Not available

Related information

"ApiEngWorkCode" on page 245

ApiEngWorkCode: GetById

```
Public Function GetById(Optional ByVal sEngagementId As String = "") As Int32
```

Purpose

Fills the object with work code data related to the specified EngagementId passed in the parameter or in the property.

Parameters

Parameter (*=required)	Description
sEngagementId	Contract ID.

Returns

0 = Success

Nonzero = Error

Remarks

If optional parameter sEngagementId is not provided, Contract.Id is used.

Property sxmlWorkCodes contains all work code category and work Code. If AllWorkCodes is True, then True is selected for all work codes; otherwise, True is selected for the workcodes associated with this contract only.

Example

```
Dim myEngWorkCode As New ApiEngWorkCode
Dim iRet As Int32
myEngWorkCode.CPConnection = myCon
iRet = myEngWorkCode.GetById("{5A390603-6748-454E-8AD3-FC8D65FABA88}")
```

Related information

"ApiEngWorkCode" on page 245

ApiEngWorkCode: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Return the XML structure of the contract work codes

Parameters

None

Returns

An XML string of the the contract work codes

Remarks

None

Example

```
Dim myEngWorkCode As New ApiEngWorkCode
Dim sXmlStructure As String = String.Empty
myEngWorkCode.CPConnection = myCon
sXmlStructure = myEngWorkCode.GetXMLStructure()
```

Related information

"ApiEngWorkCode" on page 245

"ApiEngWorkCode XML" on page 247

ApiEngWorkCode: Save

```
Public Function Save() As Int32
```

Purpose

Update the general information of work codes of a contract

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

It is recommended to use the GetById method or GetXMLStructure to obtain correct XML format of property sxmlWorkCodes.

Example

```
Dim myEngWorkCode As New ApiEngWorkCode
Dim iRet As Int32
Dim sXmlCodes As String
sXmlCodes =
"<root>
<engworkcode>
   <workcodecategoryid>{AA9C83C1-6E1E-4974-9DEC-
7C554CC429D2}</workcodecategoryid>
   <workcodecategory>Default Work Code Category</workcodecategory>
   <workcodeid>{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}</workcodeid>
   <workcode>Default Work Code</workcode><selected>1</selected>
</engworkcode>
<engworkcode>
   ...
</engworkcode>
</root>"

myEngWorkCode.CPConnection = myCon
With myEngWorkCode
   .Engagement.Id = "{5A390603-6748-454E-8AD3-FC8D65FABA88}"
   .AllWorkCodes = False
   .AllowRTCodeOverride = True
   .DefaultWorkCodeCategory.Id = "{B7E06668-7616-42F8-A0C0-97A561F2EFF2}"
   .DefaultWorkCode.Id = "{695D56A1-099F-45D1-8ACB-0ED4D240C8EB}"
   .sxmlWorkCodes = sXmlCodes
End With
iRet = myEngWorkCode.Save()
```

Related information

"ApiEngWorkCode" on page 245

"ApiEngWorkCode XML" on page 247

"ApiEngWorkCode: GetById" on page 247

"ApiEngWorkCode: GetXMLStructure" on page 248

ApiEngWorkLocation

The ApiEngWorkLocation object represents the work locations for contracts.

Namespace

Changepoint.ChangepointAPI2.ApiEngWorkLocation

Methods

Properties

Property (*=required)	Type	Description
AllLocations	Boolean	Include all locations
AllowRTLocOverride	Boolean	Allow request time work location override
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
*CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CreatedBy	Signature	Contains details about when the record was created and by who.
DefaultWorkLocation	Identity	Identifer of the default work location for the contract

Property (*=required)	Type	Description
DefaultWorkLocationGroup	Identity	Identifer of the default work location group for the contract
Deleted	Boolean	Flag that indicates whether the record has been deleted.
*Engagement	Identity	Identifer of the contract
mVersion	String	Read-only. The current version number for internal version control.
sXmlWorkLocations	String	XML document that includes work locations. See Remarks of GetXmlStructure method for the XML structure.
UpdatedBy	Signature	Details of when the record was updated and by whom.

Related information

"ApiEngagement" on page 98

"ApiEngWorkLocation XML" on page 252

ApiEngWorkLocation XML

```
<root>
<engworklocation>
   <worklocationgroupid/>
   <worklocationgroup/>
   <worklocationid/>
   <worklocation/>
   <selected/>
</engworklocation>
</root>
```

Comments

Not available

Example

Not available

Related information

"ApiEngWorkLocation" on page 250

ApiEngWorkLocation: GetById

```
Public Function GetById(Optional ByVal sEngagementId As String = "") As Int32
```

Purpose

Fills the object with work location data related to the specified EngagementId passed in the parameter or in the property.

Parameters

Parameter (*=required)	Description
sEngagementId	Contract ID.

Returns

0 = Success

Nonzero = Error

Remarks

If the optional parameter sEngagementId is not provided, the Contract.Id is used.

Property sxmlWorkLocations contains all GlobalWorkLocations and worklocations. If AllLocations is True, then True is selected for all locations; otherwise True is selected the locations that are associated with this contract only.

Example

```
Dim myEngWorkLoc As New ApiEngWorkLocation
Dim iRet As Int32
myEngWorkLoc.CPConnection = myCon
iRet = myEngWorkLoc.GetById("{5A390603-6748-454E-8AD3-FC8D65FABA88}")
```

Related information

"ApiEngWorkLocation" on page 250

ApiEngWorkLocation: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Return the XML structure of the contract work locations

Parameters

None

Returns

An XML string of the the contract work locations

Remarks

None

Example

```
Dim myEngWorkLoc As New ApiEngWorkLocation
Dim sXmlStructure As String = String.Empty
myEngWorkLoc.CPConnection = myCon
sXmlStructure = myEngWorkLoc.GetXMLStructure()
```

Related information

"ApiEngWorkLocation XML" on page 252

ApiEngWorkLocation: Save

```
Public Function Save() As Int32
```

Purpose

Update the general information of work locations of a contract

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

It is recommended to use the GetById method or GetXMLStructure to obtain the correct XML format of property sXmlWorkLocations.

Example

```
Dim myEngWorkLoc As New ApiEngWorkLocation
Dim iRet As Int32
Dim sXmlLocations As String

sXmlLocations =
"<root>
<engworklocation>
   <worklocationgroupid>{210FD5FB-6493-478D-8A13-
A730F6A581E3}</worklocationgroupid>
   <worklocationgroup>Japan</worklocationgroup>
   <worklocationid>{4EE1226F-A64E-4E93-86D2-CA6A5B89C58C}</worklocationid>
   <worklocation>Tokyu</worklocation><selected>1</selected>
</engworklocation>
<engworklocation>
   ...
</engworklocation>
</root>"

myEngWorkLoc.CPConnection = myCon
With myEngWorkLoc
   .Engagement.Id = "{5A390603-6748-454E-8AD3-FC8D65FABA88}"
   .AllLocations = False
   .AllowRTLocOverride = True
   .DefaultWorkLocationGroup.Id = "{210FD5FB-6493-478D-8A13-A730F6A581E3}"
   .DefaultWorkLocation.Id = "{4EE1226F-A64E-4E93-86D2-CA6A5B89C58C}"
   .sxmlWorkLocations = sXmlLocations
End With
iRet = myEngWorkLoc.Save()
```

Related information

"ApiEngWorkLocation" on page 250

"ApiEngWorkLocation XML" on page 252

"ApiEngWorkLocation: GetById" on page 253

"ApiEngWorkLocation: GetXMLStructure" on page 254

ApiException

The ApiException object catches and tracks the exception information.

Namespace

Changepoint.ChangepointAPI2.ApiException

Methods

None

Properties

Property (*=required)	Type	Description
LogLevel	Byte	Specifies the level of error information returned in COM API log file. The values are as follows: 0 = NoLog 1 = Source 2 = ErrorMessages 3 = InputParameters 4 = Returns 8 = Checkpoint.
LogPath	String	The log file path, default is: C:\Program Files\Changepoint\ Changepoint\ApiLogs\
Logs	CPLogs	Read-only. Logging information which is stored in CPLogs object.
Message	String	Read-only. The error message.
Number	Int32	The error number.
Source	String	The source where the logging started.
StackTrace	String	Read-only. The stack trace if exception occurred.
Warnings	CPLogs	Read-only. Warning information which is stored in CPLogs object.

ApiExchangeRate

The ApiExchangeRate object allows users to add, edit, and retrieve exchange rates from the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiExchangeRate

Methods

Properties

Property (*=required)	Type	Description
*BaseCurr	String	Base Currency. Three-letter ISO code, for example, "CAD"
*CPConnection	ApiConnection	Connection object that must be assigned before any action to database.
CurrentRate	Boolean	Flag if it is current rate.
*EndDate	Date	The End Date
ExchangeRateId	String	Read-only. Changepoint Exchange Rate ID.
*ExchRate	Decimal	Rate
*StartDate	Date	The start date
*ToCurr	String	The currency to be converted to. Three-letter ISO code, for example, "USD".
UpdatedBy	Signature	The updater of the record and when it was updated.

ApiExchangeRate: Add

```
Public Function Add(ByRef oExchangeRate As ApiExchangeRate) As String
```

Purpose

Add a new Exchange Rate into the Changepoint database.

Parameters

Parameter (*=required)	Description
*oExchangeRate	ExchangeRate object.

Returns

ExchangeRateId on success.

Remarks

None

Example

```
Dim myApi as New ApiExchangeRate
myApi.CPConnection = myCon
With myApi
.BaseCurr="CAD"
.ToCurr="USD"
.Rate=0.94
.StartDate="2009-05-06"
.EndDate="2009-08-27"
.CPConnection=myCon
End With
Dim ret as String = myApi.Add(myApi)
```

Related information

"ApiExchangeRate" on page 257

ApiExchangeRate: GetExchangeRateByDate

```
Public Function GetExchangeRateByDate(ByVal BaseCurrencyCode As String, ByVal
ToCurrencyCode As String, ByVal SDate As Date) As String
```

Purpose

Retrieve an Exchange Rate filtered by parameters

Parameters

Parameter (*=required)	Description
*BaseCurrencyCode	Base Currency in Changepoint.
*ToCurrencyCode	To Currency in Changepoint.
*SDate	Exchange Date

Returns

Rate

Remarks

None

Example

Not available

Related information

"ApiExchangeRate" on page 257

ApiExchangeRate: GetExchangeRateId

```
Public Function GetExchangeRateId(ByVal BaseCurrencyCode As String, ByVal
ToCurrencyCode As String, ByVal SDate As Date) As String
```

Purpose

Retrieves an ExchangeRateId filtered by parameters

Parameters

Parameter (*=required)	Description
*BaseCurrencyCode	Base Currency in Changepoint.
*ToCurrencyCode	To Currency in Changepoint.
*SDate	Exchange Date

Returns

ExchangeRateId

Remarks

None

Example

Not available

Related information

"ApiExchangeRate" on page 257

ApiExchangeRate: GetExRate

```
Public Function GetExRate(ByVal sBaseCurrency As String, ByVal sToCurrency As
String, ByVal ActionDate As Date) As ApiExchangeRate
```

Purpose

Retrieves the exchange rate information in the ApiExchangeRate object

Parameters

Parameter (*=required)	Description
*sBaseCurrency	Base Currency in Changepoint.
*sToCurrency	To Currency in Changepoint.
*ActionDate	Exchange Date

Returns

ApiExchangeRate object, which contains exchange rate information. The Fields include: ExchangeRateID, Rate, StartDate, EndDate and CurrentRate.

Remarks

None

Example

Not available

Related information

"ApiExchangeRate" on page 257

ApiExchangeRate: Update

```
Public Function Update(ByRef oExchangeRate As ApiExchangeRate) As Int32
```

Purpose

Updates Exchange Rate information in the Changepoint database.

Parameters

Parameter (*=required)	Description
*oExchangeRate	ApiExchangeRate object.

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

```
Dim myApi as New ApiExchangeRate()
myApi.CPConnection=myCon
Dim myRate as New ApiExchangeRate()
```

```
With myRate
.ExchangeRateId="{}"
.CurrentRate=True
.BaseCurr="CAD"
.ToCurr="USD"
.Rate=0.94
.SartDate="2007-05-06"
.EndDate="2007-08-27"
End With
Dim ret as Int32=myApi.Update(myRate)
```

Related information

"ApiExchangeRate" on page 257

ApiExpense

The ApiExpense object allows users to create, retrieve and update expenses in the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiExpense

Methods

Properties

Property (*=required)	Data Type	Description
AssociatedTask	Identity	Identifier of the associated task
Billable	Boolean	True if billable.
BypassMetadataCheck	CPMetadataCheck	Determines the level of MetaData checking when adding or updating data. The default is to check all fields. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
*Category	Identity	Identifier of the expense category
ChangeReason	String	Reason for changing the expense
CreatedBy	Signature	Contains details about who created the expense and when the expense was created
*Currency	Identity	Identifier of the currency for the expense
*Customer	Identity	Read-only. Identifier of the customer associated with the expense
Description	String	Expense description
Editable	Boolean	Read-only. True if editable.
*Engagement	Identity	Read-only. Identifier of the contract associated with the expense
Entity	CPEntity	Read-only. The Changepoint entity the object represents.
*ExchangeRate	Decimal	Exchange rate for the expense

Property (*=required)	Data Type	Description
ExpenseCode1	Identity	Expense Code1
ExpenseCode2	Identity	Expense Code 2
ExpenseCode3	Identity	Expense Code 3
*ExpenseDate	DateTime	Expense date
ExpenseId	String	Expense ID
ExpenseText1	String	Expense Text 1
ExpenseText2	String	Expense Text 2
ExpenseText3	String	Expense Text 3
*ExpenseType	Identity	Expense type ID
FedAudit	Boolean	Read-only. True if auditing is enabled
FixedFee	Identity	Identifier of the fixed fee
mVersion	String	Read-only. The current version number for internal version control.
*Project	Identity	Identifier of the project associated with the expense
*Quantity	Double	Quantity
RecoverableTax	Identity	Read-only. Identifier of the recoverable tax
ReimbursableExpense	Boolean	Read-only. True if the expense is reimbursable. Default is False.
Resource	Identity	Identifier of the resource associated with the expense
sxmlUDF	String	UDF XML string. Includes all information of UDF code and UDF text

Property (*=required)	Data Type	Description
SubmittedForApproval	Boolean	Read-only. True if submitted for approval
TaxAmount	Decimal	Tax amount
*UnitPrice	Decimal	Unit price
WorkCode	Identity	Identifier of the work code
WorkCodeCategory	Identity	Identifier of the work code category
Workgroup	Identity	Identifier of the workgroup
*WorkLocation	Identity	Identifier of the work location
*WorkLocationGroup	Identity	Identifier of the work location group

Related information

"ApiExpense XML" on page 266

ApiExpense XML

```
<root>
   <expense>
      <bypassmetadatacheck />
      <createdbyid />
      <updatedbyid />
      <expenseid />
      <customer>
         <id />
         <name />
         <alternatename />
         <userdefinedid />
      </customer>
      <engagement>
         <id />
         <name />
         <alternatename />
         <userdefinedid />
      </engagement>
      <project>
         <id />
         <name />
         <alternatename />
```

```
   <userdefinedid />
</project>
<description />
<resource>
   <id />
   <name />
   <alternatename />
   <firstname />
   <lastname />
   <userdefinedid />
</resource>
<category>
   <id />
   <name />
   <alternatename />
</category>
<billable>false</billable>
<expensedate />
<expensetype>
   <id />
   <name />
   <alternatename />
</expensetype>
<quantity />
<unitprice />
<recoverabletax>
   <id />
   <name />
   <alternatename />
</recoverabletax>
<taxamount />
<currency>
   <id />
   <name />
   <alternatename />
</currency>
<exchangerate />
<editable>true</editable>
<submittedforapproval>false</submittedforapproval>
<expensetext1 />
<expensetext2 />
<expensetext3 />
<expensecode1>
   <id />
   <name />
   <alternatename />
</expensecode1>
<expensecode2>
```

```
      <id />
      <name />
      <alternatename />
   </expensecode2>
   <expensecode3>
      <id />
      <name />
      <alternatename />
   </expensecode3>
   <worklocationgroup>
      <id />
      <name />
      <alternatename />
      <userdefinedid />
   </worklocationgroup>
   <worklocation>
      <id />
      <name />
      <alternatename />
      <userdefinedid />
   </worklocation >
   <workcodecategory>
      <id />
      <name />
      <alternatename />
      <userdefinedid />
   </workcodecategory>
   <workcode>
      <id />
      <name />
      <alternatename />
      <userdefinedid />
   </workcode>
   <associatedtask>
      <id />
      <name />
      <alternatename />
   </associatedtask>
   <fixedfee>
      <id />
      <name />
      <alternatename />
   </fixedfee>
   <fedaudit>false</fedaudit>
   <changereason />
   <reimbursableexpense>false</reimbursableexpense>
   <udf />
</expense>
```

```
<root />
```

Comments

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: Add

```
Public Overrides Function Add(Optional ByRef sId As String = "") As Integer
```

Purpose

Adds a new expense to Changepoint

Parameters

Parameter (*=required)	Description
sId	ByRef parameter that will return the new ExpenseId after the expense has been added.

Returns

0 = Success

Nonzero = Error

Remarks

See the ApiExpense properties for mandatory fields.

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, Optional ByRef sId As String
= "") As Int32
```

Purpose

Create an expense using an XML string of the Expense object in Changepoint.

Parameters

Parameter (*=required)	Description
*sXML	A new expense XML string which is passed based on the Expense XML schema
sId	ByRef parameter that returns the new ExpenseId after the expense has been added.

Returns

0 = Success

Nonzero = Error

Remarks

The ApiExpense XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The ByPassMetadataCheck switch will stop metadata validation in the Expense.

Example

Not available

Related information

"ApiExpense" on page 262

"ApiExpense XML" on page 266

"ApiExpense: GetByXML" on page 274

"ApiExpense: GetXMLStructure" on page 287

ApiExpense: Delete

```
Public Overrides Function Delete(Optional ByVal sId as String = "") As Integer
```

Purpose

Deletes the specified expense from Changepoint

Parameters

Parameter (*=required)	Description
sId	Expense ID of the expense to delete

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: Exists

```
Public Overrides Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Verifies whether the expense exists in the database and has not been deleted.

Parameters

Parameter (*=required)	Description
sId	Expense ID of the expense to verify

Returns

True if the expense exists, else False.

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetAssociatedTasks

```
Public Function GetAssociatedTasks(ByVal sResourceId As String, ByVal
sProjectId As String, Optional ByVal sSearchString As String = "") As DataSet
```

Purpose

Returns a list of tasks associated with the expense

Parameters

Parameter (*=required)	Description
*sResourceId	Resource ID
*sProjectId	Project ID
sSearchString	Search string

Returns

A dataset with two columns (TaskId, Name)

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetById

```
Public Overrides Function GetById(Optional ByVal sId as String = "") As
Integer
```

Purpose

Fills the object with data related to the specified ExpenseId passed in the parameter.

Parameters

Parameter (*=required)	Description
sId	Expense ID of the expense whose data is to be retrieved

Returns

0 = Success

Nonzero = Error

Remarks

This method fills the object with current data from the database.

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetByXML

```
Public Function GetByXML(Optional ByVal sXML As String = "", Optional ByVal
sExpenseId As String = "") As String
```

Purpose

Takes the XML string passed in the sXML parameter and returns the string filled with data for the Expense ID specified in the sExpenseId parameter.

Parameters

Parameter (*=required)	Description
sXML	XML string of the Expense object, with the fields specified
sExpenseId	Expense ID • If no Expense ID is passed in, the XML string (sXML) is examined • if there is no Expense ID in sXML then the object's Expense ID is selected. • If there still is no valid ExpenseId, the method returns an error.

Returns

An XML string mirroring sXML with data inserted, or the entire XML of the Expense object including data.

Remarks

If sXML = "" then the XML string provided by GetXMLStructure is used.

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetCategories

```
Public Function GetCategories(Optional ByVal sSearchString as String = "") As
DataSet
```

Purpose

Returns a list of expense categories.

Parameters

Parameter (*=required)	Description
sSearchString	Search string

Returns

A dataset with two columns (ExpenseCategoryId, Name)

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetCurrency

```
Public Function GetCurrency() As DataSet
```

Purpose

Returns a currency list

Parameters

None

Returns

A dataset with two columns (Code, Description)

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetCustomers

```
Public Function GetCustomers(ByVal sResourceId As Guid, Optional ByVal
sSearchString as String = "") As DataSet
```

Purpose

Returns a list of customers.

Parameters

Parameter (*=required)	Description
*sResourceId	Resource ID
sSearchString	Search string

Returns

A dataset with two columns (CustomerId, Name)

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetEngagements

```
Public Function GetEngagements(ByVal sResourceId As String, ByVal sCustomerId
As String, Optional ByVal sSearchString As String = "") As DataSet
```

Purpose

Returns a list of contracts for a specified customer.

Parameters

Parameter (*=required)	Description
*sResourceId	Resource ID
*sCustomerId	Customer ID
sSearchString	Search string

Returns

A dataset with two columns (EngagementId, Name)

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetExpenseList

```
Public Function GetExpenseList(Optional ByVal iRetRows As Short = -1, Optional
ByVal sCustomerId As String = "", Optional ByVal sEngagementId As String = "",
Optional ByVal sProjectId As String = "", Optional ByVal sResourceId As String
= "") As DataSet
```

Purpose

Returns a list of some or all of the expenses that were not submitted to an expense report, based on the specified parameters.

Parameters

Parameter (*=required)	Description
iRetRows	Number of rows to return. Default = -1 (return all).
sCustomerId	Customer ID
sEngagementId	Engagement ID
sProjectId	Project ID
sResourceId	Resource ID

Returns

A dataset with nineteen columns (ExpenseId, CustomerId, EngagementId, ProjectId, ResourceId, CategoryId, ExpenseType, ExpenseDate, Description, Billable, Quantity, UnitPrice, TaxAmount, CurrencyCode, ExchangeRate, Editable, SubmittedForApproval, ExpenseReportID, ReimbursableExpense)

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetFixedFees

```
Public Function GetFixedFees(ByVal sEngagementId As String) As DataSet
```

Purpose

Returns a list of fixed fees for a specifed engagement

Parameters

Parameter (*=required)	Description
*sEngagementId	Engagement ID

Returns

A dataset with two columns (FixedFeeID, FixedFeeDescription)

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As String
```

Purpose

Returns the ExpenseId based on the UDF Text field and value.

Parameters

Parameter (*=required)	Description
*sUDFField	UDF text field name (for example, Text1)
*sUDFValue	UDF text value

Returns

ExpenseId, or an empty string if nothing is found.

Remarks

None

Related information

"ApiExpense" on page 262

ApiExpense: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows as Short = -1) As
DataSet
```

Purpose

Returns a list of some or all expenses that were not submitted to an expense report in Changepoint based on the logged-in user.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).

Returns

A dataset with nineteen columns (ExpenseId, CustomerId, EngagementId, ProjectId, ResourceId, CategoryId, ExpenseType, ExpenseDate, Description, Billable, Quantity, UnitPrice, TaxAmount, CurrencyCode, ExchangeRate, Editable, SubmittedForApproval, ExpenseReportID, ReimbursableExpense) or an empty dataset if nothing is found

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetProjects

```
Public Function GetProjects(ByVal sResourceId As String, ByVal sEngagementId
As String, Optional ByVal sSearchString As String = "") As DataSet
```

Purpose

Returns a list of projects

Parameters

Parameter (*=required)	Description
*sResourceId	Resource ID
*sEngagementId	Engagement ID
sSearchString	Search string

Returns

A dataset with two columns (ProjectId, Name)

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetTypes

```
GetTypes(Optional ByVal sCategoryId As String, Optional ByVal sSearchString As
String = "") As DataSet
```

Purpose

Returns a list of expense types.

Parameters

Parameter (*=required)	Description
sCategoryId	Expense category ID
sSearchString	Search string

Returns

A dataset with columns (ExpenseCategoryId, ExpenseCategoryName, ExpenseTypeId, Description)

Remarks

None

Example

If sCategoryId is not empty, the list contains expense types only for the specified expense category. If sSearchString is not empty, the list contains expense types only for expenses whose descriptions match sSearchString.

Related information

"ApiExpense" on page 262

•

ApiExpense: GetUDF

```
Public Function GetUDF(Optional ByVal retOption As CPUDFReturnType =
CPUDFReturnType.OnlyValues, Optional ByVal actionResourceId As String = "") As
String
```

Purpose

Retrieve UDF (configurable field) information for an expense.

Parameters

Parameter (*=required)	Description
retOption	The CPUDFReturnType Values are: • WithStructure = 0 – returns UDF field data with full field structure • OnlyValues = 1 – returns only UDF fields with data, does not include any field structure or options • OnlyStructure = 2 – returns only UDF XML structure, no field data • OnlyStructureAndOptions = 3 – returns UDF structure and option data (except entity based and conditional codes) without field data • WithStructureAndOptions = 4 – returns UDF field data with full structure and all field options (except entity based and conditional codes)
actionResourceId	The user ID of the user that called the method. Default is the signed-in user id.

Returns

An XML string of UDFs.

Remarks

If ExpenseId is empty, the retOptions that return data – WithStructure (0); OnlyValues (1); and WithStructureAndOptions (4) – return default values for field data.

Example

```
Dim myExp as New ApiExpense()
Dim sRet As String
myExp.CPConnection = myCon
myExp.ExpenseId = "{CA8AC6E5-5F9A-41CA-BD57-9A35D28A7E76}"
sRet = myExp.GetUDF(CPUDFReturnType.WithStructureAndOptions)
```

ApiExpense: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal codeName As String, Optional ByVal
searchString As String = "") As String
```

Purpose

Retrieve UDF code options for the specified expense UDF code.

Parameters

Parameter (*=required)	Description
*codeName	The name of the UDF code, for example, "Code3"
SearchString	Returns the options that start with this string.

Returns

UDF code options in XML format.

Remarks

None.

Example

```
Dim myExp as New ApiExpense()
Dim sRet As String
myExp.CPConnection = myCon
myExp.ExpenseId = "{CA8AC6E5-5F9A-41CA-BD57-9A35D28A7E76}"
sRet = myExp.GetUDFCodeOptions("Code3", "")
```

ApiExpense: GetWorkCodeCategories

```
Public Function GetWorkCodeCategories(ByVal sProjectId As String, Optional
ByVal sSearchString As String = "") As DataSet
```

Purpose

Returns a lists of workcode categories.

Parameters

Parameter (*=required)	Description
*sProjectId	Project ID
	Search string

Returns

A dataset with two columns (WorkCodeCategoryId, Name)

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetWorkCodes

```
Public Function GetWorkCodes(ByVal sProjectId As String, ByVal
sWorkCodeCategoryId As String, Optional ByVal sSearchString As String = "") As
DataSet
```

Purpose

Returns a list of work codes.

Parameters

Parameter (*=required)	Description
*sProjectId	Project ID
*sWorkCodeCategoryId	Work code category ID
sSearchString	Search string

Returns

A dataset with two columns (WorkCodeId, Name)

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetWorkLocationGroups

```
Public Function GetWorkLocationGroups(ByVal sProjectId As String, Optional
ByVal sSearchString As String = "") As DataSet
```

Purpose

Returns a list of work location groups.

Parameters

Parameter (*=required)	Description
*sProjectId	Project ID
sSearchString	Search string

Returns

A dataset with two columns (WorkLocationGroupId, Name)

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetWorkLocations

```
Public Function GetWorkLocations(ByVal sProjectId As String, ByVal
sWorkLocationGroupId As String, Optional ByVal sSearchString As String = "")
```

```
As DataSet
```

Purpose

Returns a list of work locations.

Parameters

Parameter (*=required)	Description
*sProjectId	Project ID
*sWorkLocationGroupId	Work location group ID
sSearchString	Search string

Returns

A dataset with two columns (WorkLocationId, Name)

Remarks

None

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Returns the XML structure of the ApiExpense object.

Parameters

None

Returns

A string containing the XML structure of the ApiExpense object

Remarks

Some fields in the structure will have defaulted data; otherwise, fields are empty.

Example

Not available

Related information

"ApiExpense" on page 262

"ApiExpense XML" on page 266

ApiExpense: SaveUDF

```
Public Function SaveUDF(Optional ByVal sUDF As String = "") As Int32
```

Purpose

Saves (Insert/Update) UDF data for an expense.

Parameters

Parameter (*=required)	Description
sUDF	UDF string in XML format.

Returns

0 = Success

Nonzero = Error

Remarks

The Expense.Id is mandatory, so this property should be populated before calling SaveUDF method. If sUDF is not provided, it takes the value of property sxmlUDF.

To obtain the correct UDF XML format, call the GetUDF method.

Example

```
Dim myExp as New ApiExpense()
Dim iRet As Int32
Dim strXMLUDF as string ="<root></root>"
With myExp
   .CPConnection = myCon
   .ExpenseId = "{CA8AC6E5-5F9A-41CA-BD57-9A35D28A7E76}"
   iRet = .SaveUDF(strXMLUDF)
End With
```

ApiExpense: Update

```
Public Overrides Function Update() As Integer
```

Purpose

Updates expense data in the database with data held in the object.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

Update will write all data held in the Expense object

Example

Not available

Related information

"ApiExpense" on page 262

ApiExpense: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, Optional ByVal sExpenseId As
String = "") As Int32
```

Purpose

Updates a expense using an XML string containing new data

Parameters

Parameter (*=required)	Description
*sXML	The XML string of the Expense object with new data contained in the fields.
sExpenseId	Expense ID of the expense being updated.

Returns

0 = Success

Nonzero = Error

Remarks

Performs the same function as Update except any expense can be updated through this function. The XML sent in the parameter must be of the form in ApiExpense XML. The ApiExpense XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The method uses the following sequence to find the expense ID:

1. If the sExpenseId parameter is passed in, the method uses this value for the expense ID.
2. If this fails, the method attempts to extract the expense ID from <expenseid> in the XML.
3. If this fails, the expense ID is taken from the object properties.

Example

Not available

Related information

"ApiExpense" on page 262

"ApiExpense XML" on page 266

"ApiExpense: GetByXML" on page 274

"ApiExpense: GetXMLStructure" on page 287

ApiExpenseReport

The ApiExpenseReport object allows users to retrieve and mark expense reports as batched in the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiExpenseReport

Methods

Properties

Property (*=required)	Type	Description
Advance	Decimal	Amount of advance applied to this Expense Report
Billable(ByVal Index As Short)	Boolean	Billable flag for each expense on the report. This value is returned by the index from the list of billables flag of expenses.
CategoryId(ByVal Index As Short)	String	ID of Category for each expense on the report. This value is returned by the index from the list of category IDs of expenses.
CategoryName()	Array of String	Name of Category for each expense
Comments	String	Expense report comments
*CPConnection	ApiConnection	Connection object that must be assigned before any action to database.

Property (*=required)	Type	Description
Description(ByVal Index As Short)	String	Description for each expense on the report. This value is returned by the index from the list of exchange rates of expenses.
EngagementId	String	Id of engagement
EngagementName	String	Name of engagement
ExchangeRate(ByVal Index As Short)	Decimal	Exchange rate for each expense on the report. This value is returned by the index from the list of exchange rates of expenses.
ExpenseAmount(ByVal Index As Short)	Decimal	Total expense amount for each expense on the report. This value is returned by the index from the list of amounts of expenses.
ExpenseCount	Integer	Number of expenses on report
ExpenseCurrency(ByVal Index As Short)	String	Currency for each expense. Three-letter ISO code, e.g. "CAD" This value is returned from the list of currencies of expenses.
ExpenseDate(ByVal Index As Short)	Date	Date of each expense on the report. This value is returned by the index from the list of dates of expenses.
ExpenseId(ByVal Index As Short)	String	ID of each expense on the report. This value is returned by the index from the list of IDs of expenses.
ExpenseReportId	String	ID of Expense Report
ExpenseReportNumber	Integer	Expense Report Number
ExpenseTotal	Decimal	Expense Report Total

Property (*=required)	Type	Description
FinanceActionDate	Date	Finance Action Date
FinanceApproverId	String	Finance Approver ID
FinanceApproverName	String	Finance Approver Name
GLAAmount(ByVal Index As Short)	Decimal	Amount for each GLA code. This value is returned by the index from the list of GLA code amounts.
GLABillable(ByVal Index As Short)	Boolean	Billable flag for each GLA code. This value is returned by the index from the list of GLA billable flags.
GLACode(ByVal Index As Short)	String	Code for each GLA code. This value is returned by the index from the list of GLA codes.
GLACount	Integer	Number of GLA codes
GLADescription(ByVal Index As Short)	String	Description of each GLA code. This value is returned by the index from the list of GLA descriptions.
GLAIndex(ByVal Index As Short)	String	Index value for each GLA code. This value is returned by the index from the list of GLA indexes.
GLAReimbursable(ByVal Index As Short)	Boolean	Reimbursable value for each GLA code. This flag is returned by the index from the list of GLA reimbursables.
GlobalWorkgroupId	String	Global Workgroup ID
GlobalWorkGroupName	String	Name of global workgroup
ManagerActionDate	Date	Manager action date
ManagerId	String	Manager's ID

Property (*=required)	Type	Description
ManagerName	String	Manager's name
Price(ByVal Index As Short)	Decimal	Unit price for each expense on the report. This value is returned by the index from the list of prices of expenses.
ProjectId(ByVal Index As Short)	String	Project ID for each expense. This value is returned by the index from the list of project IDs of expenses.
ProjectName(ByVal Index As Short)	String	Name of project for each expense. This value is returned by the index from the list of project names of expenses
Quantity(ByVal Index As Short)	Double	Quantity for each expense on the report. This value is returned by the index from the list of expense quantities of expenses.
Reimbursable(ByVal Index As Short)	Boolean	Reimbursable flag for each expense on the report. This value is rturned by the index from the list of expense reimbursable flags of expenses.
ReportCurrency	String	Currency of expense report
ResourceId	String	Resource ID
ResourceName	String	Resource name
ResourceSubmissionDate	Date	Resource submission date
ReturnCash	Decimal	Return cash
ReturnedAmount	Decimal	Returned amount
ReturnedCurrency	String	Returned currency, it is 3 letter abbreviations, for example, "CAD".

Property (*=required)	Type	Description
Status	String	Status of expense report
TaxAmount(ByVal Index As Short)	Decimal	Recoverable tax amount for each expense. This value is returned by the index from the list of expense recoverable tax amounts of expenses.
TypeId(ByVal Index As Short)	String	ID of expense type for each expense on the report. This value is returend by the index from the list of expense types of expenses.
TypeName(ByVal Index As Short)	String	Name of expense type for each expense on the report. This value is returned by the index from the list of expense type names of expenses.
WorkgroupId	String	Workgroup ID
WorkGroupName	String	Name of Workgroup

ApiExpenseReport: GetExpenseReport

```
Public Function GetExpenseReport(ByVal sExpenseReportid As String) As
ApiExpenseReport
```

Purpose

Retrieve an expense report based upon the passed in ExpenseReportId.

Parameters

Parameter (*=required)	Description
*sExpenseReportId	ID of the expense report.

Returns

ApiExpenseReport object

Remarks

None

Example

```
Dim myApi as New ExpenseReport()
myApi.CPConnection=myCon
Dim ERId as String = "{}"
Dim retER as ExpenseReport = myApi.GetExpenseReport(ERId)
```

Related information

"ApiExpenseReport" on page 291

ApiExpenseReport: GetExpenseReports

```
Public Function GetExpenseReports(ByVal Batched As Boolean) As DataSet
```

Purpose

Retrieves the expense report list from Changepoint database.

Parameters

Parameter (*=required)	Description
*Batched	Boolean value specifying to retrieve invoices that have already been batched.

Returns

A dataset of the expense report list which includes ExpenseReportId and ExpenseReportNumber.

Remarks

None

Example

```
Dim myApi as New ExpenseReport()
myApi.CPConnection=myCon
Dim ret as DataSet = myApi.GetExpenseReports (False)
```

Related information

"ApiExpenseReport" on page 291

ApiExpenseReport: MarkExpenseReportAsBatched

```
Public Function MarkExpenseReportAsBatched(ByVal ExpenseReportId As String) As
Integer
```

Purpose

Mark the specified expense report as batched.

Parameters

Parameter (*=required)	Description
*ExpenseReportId	ID of the Expense Report.

Returns

0 = Success

Nonzero = Error

Remarks

Indicates that the expense report by this ID has been extracted from the Changepoint database. The extract date used is the current date. The expense report can be retrieved again if the batched parameter of the GetApprovedExpenseReports is specified as true.

Example

Not available

Related information

"ApiExpenseReport" on page 291

ApiFunctionDescription

The ApiFunctionDescription object represents function descriptions in the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiFunctionDescription

Methods

Properties

Property (*=required)	Type	Description
*FunctionId	String	Changepoint Function ID.
*Name	String	Name.
Description	String	Description.

ApiFunctionDescription: Add

```
Public Function Add(ByRef oFunctionDescription As ApiFunctionDescription) As
String
```

Purpose

Insert one row into FunctionDescription table.

Parameters

Parameter (*=required)	Description
*oFunctionDescription	ApiFunctionDescription object.

Returns

FunctionId on success.

Remarks

None

Example

```
Dim myFunctionDescription As New ApiFunctionDescription
Dim mFunctionDescription As New ApiFunctionDescription
Dim sRet As String

myFunctionDescription.CPConnection = myCon

With mFunctionDescription
   .Description = "First level management"
   .Name = "Team Lead"
End With

sRet = myFunctionDescription.Add(mFunctionDescription)
```

Related information

"ApiFunctionDescription" on page 297

ApiFunctionDescription: GetFunction

```
Public Function GetFunction(ByRef sFunctionId As String) As
ApiFunctionDescription
```

Purpose

Retrieves a function description record

Parameters

Parameter (*=required)	Description
*sFunctionId	Changepoint Function ID.

Returns

Single object of ApiFunctionDescription.

Remarks

None

Example

Not available

Related information

"ApiFunctionDescription" on page 297

ApiFunctionDescription: GetFunctionByBillingRole

```
Public Function GetFunctionByBillingRole(ByRef sBillingRoleId As String) As
String
```

Purpose

Retrieves the function ID of a billing role

Parameters

Parameter (*=required)	Description
*sBillingRoleId	Billing Role ID in Changepoint. Read-only.

Returns

Function ID for the specific billing role

Remarks

None

Example

Not available

Related information

"ApiFunctionDescription" on page 297

ApiFunctionDescription: GetFunctions

```
Public Function GetFunctions() As Dataset
```

Purpose

Retrieves all function description records

Parameters

None

Returns

A dataset of function descriptions.

Remarks

Returned fields are FunctionId and Name.

Example

Not available

Related information

"ApiFunctionDescription" on page 297

ApiFunctionDescription: Update

```
Public Function Update(ByRef oFunctionDescription As ApiFunctionDescription)
As Int32
```

Purpose

Updates a function description record.

Parameters

Parameter (*=required)	Description
*oFunctionDescription	ApiFunctionDescription object.

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

```
Dim myFunctionDescription As New ApiFunctionDescription
Dim mFunctionDescription As New ApiFunctionDescription
Dim iRet As Int32 = 0

myFunctionDescription.CPConnection = myCon

With mFunctionDescription
   .FunctionId = "{3EA44C0E-0D31-4FB1-A6C3-3703A837B9B2}"
   .Description = "First level management"
   .Name = "Team Leader 1"
End With

iRet = myFunctionDescription.Update(mFunctionDescription)
```

Related information

"ApiFunctionDescription" on page 297

ApiFunctionDescription: UptBillingRoleWithFunction

```
Public Function UptBillingRoleWithFunction(ByRef sBillingRoleId As String,
ByRef sFunctionId As String) As Int32
```

Purpose

Updates the function ID in the billing role record.

Parameters

Parameter (*=required)	Description
*sBillingRoleId	Billing Role ID in Changepoint.
*sFunctionId	Function ID in Changepoint.

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

```
Dim myFunctionDescription As New ApiFunctionDescription
Dim iRet As Int32 = 0

myFunctionDescription.CPConnection = myCon

iRet = myFunctionDescription.UptBillingRoleWithFunction("{7E4F8CF4-363B-11D4-
8AB1-0001023D3221}", "{3EA44C0E-0D31-4FB1-A6C3-3703A837B9B2}")
```

Related information

"ApiFunctionDescription" on page 297

ApiFunctionSkill

The ApiFunctionSkill object represents the skills that are associated to functions within the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiFunctionSkill

Methods

Properties

Property (*=required)	Type	Description
*CPConnection	ApiConnection	Connection to the Changepoint database
*FunctionId	String	Read-only. Changepoint Function ID.

Property (*=required)	Type	Description
*SkillId	String	Skill ID
SkillCompetencyId	String	Skill competency ID

Related information

"ApiFunctionSkill XML" on page 304

ApiFunctionSkill XML

```
<FunctionSkill>
   <FunctionId/>
   <SkillId/>
   <SkillCompetencyId/>
</FunctionSkill>
```

Comments

Not available

Example

Not available

Related information

"ApiFunctionSkill" on page 303

ApiFunctionSkill: GetFunctionSkills

```
Public Function GetFunctionSkills(ByVal sFunctionId As String) As
ApiFunctionSkill()
```

Purpose

Retrieve function skills for a function

Parameters

Parameter (*=required)	Description
*sFunctionId	Changepoint Function ID.

Returns

An array of ApiFunctionSkill objects.

Remarks

None

Example

Not available

Related information

"ApiFunctionSkill" on page 303

ApiFunctionSkill: Save

```
Public Function Save(ByVal sFunctionId As String, ByVal xmlFunctionSkill As
String) As Int32
```

Purpose

Update function skills and skill competencies for a function

Parameters

Parameter (*=required)	Description
*sFunctionId	Changepoint Function ID
*xmlFunctionSkill	An XML string converted from ApiFunctionSkill object by the ToXML method.

Returns

0 = Success

Nonzero = Error

Remarks

Add, update and delete skills and competences for a function:

- The xmlFunctionSkill string should include all function skill and skill competency records for sFunctionId. If any of these records exist in the current table, but are not included with the xmlFunctionSkill, they will be deleted.
- If xmlFunctionSkill is empty, all the skills and skillcompetencies under sFunctionId will be deleted.

Example

```
Dim myFunctionSkill As New ApiFunctionSkill
Dim iRet As Int32
Dim sXmlfunctionskills As String
Dim sFunctionId As String

sXmlfunctionskills =
"<root>" & _
"<FunctionSkill>" & _
   "<FunctionId>{E3389CF7-58E6-44B5-B9C4-008E0FF22B05}</FunctionId>" & _
   "<SkillId>{693CEBFA-039C-11D6-BFAF-0060975AEC0F}</SkillId>" & _
   "<SkillCompetencyId></SkillCompetencyId>
</FunctionSkill>" & _
"<FunctionSkill>" & _
   "<FunctionId>{E3389CF7-58E6-44B5-B9C4-008E0FF22B05}</FunctionId>" & _
   "<SkillId>{693CEBFA-039C-11D6-BFAF-0060975AEC0F}</SkillId>" & _
   "<SkillCompetencyId>{693CEBFB-039C-11D6-BFAF0060975AEC0F}" & _
   "</SkillCompetencyId>" & _
"</FunctionSkill>" & _
"</root>"
sFunctionId = "{E3389CF7-58E6-44B5-B9C4-008E0FF22B05}"
myFunctionSkill.CPConnection = myCon
iRet = myFunctionSkill.Save(sFunctionId, sXmlfunctionskills)
```

Related information

"ApiFunctionSkill" on page 303

"ApiFunctionSkill XML" on page 304

"ApiFunctionSkill: ToXml" on page 306

ApiFunctionSkill: ToXml

```
Public Function ToXml(ByVal oFunctionSkill As ApiFunctionSkill) As Xml.XmlNode
```

Purpose

Convert an ApiFunctionSkill object to XML node.

Parameters

Parameter (*=required)	Description
*oFunctionSkill	ApiFunctionSkill object

Returns

XML (used in Update method)

Remarks

None

Example

Not available

Related information

"ApiFunctionSkill" on page 303

"ApiFunctionSkill XML" on page 304

ApiInvoice

The ApiInvoice object allows users to retrieve and update invoices within the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiInvoice

Methods

Properties

Property (*=required)	Type	Description
AdditionalItemsAdj	Decimal	Total adjustments of additional items
AdditionalItemsToApplyTax1Total	Decimal	Additional items to apply to tax 1
AdditionalItemsToApplyTax2Total	Decimal	Additional items to apply to tax 2
AdditionalItemsTotal	Decimal	Total of additional items
BillingOfficeId	String	Read-only. Identifies the Billing office on the contract.
*CPConnection	ApiConnection	Connection to the Changepoint database
CustomerId	String	Read-only. Identifies the customer.
Deleted	Boolean	Flag that indicates if the object has been deleted. 1 – deleted, 0 – active

Property (*=required)	Type	Description
Description	String	Invoice description
EngagementId	String	Read-only. Identifies the contract.
ExchangeApplied	String	Has the values "DI" for date of invoice or "DO" for date of occurrence. These values determine which exchange rate is used for expenses.
ExpenseTotal	Decimal	Total expense amount summary
ExpenseTotalAdj	Decimal	Expense adjustments
FixedFeeToApplyTax1Total	Decimal	Fixed fees to apply to tax 1 total
FixedFeeToApplyTax2Total	Decimal	Fixed fees to apply to tax 2 total
FixedFeeTotal	Decimal	Fixed fee total
InvoiceCurrency	String	Invoice currency code. Three-letter ISO code, for example, "USD"
InvoiceDate	Date	Date invoice was created
InvoiceId	String	Read-only. Identifies the invoice.
InvoiceNumber	String	Changepoint invoice number.
InvoiceTotal	Decimal	Summary total of invoice
InvoiceTotalPayments	Decimal	Total amount of payments made against invoice
InvoiceTotalAdj	Decimal	Total adjustments on invoice

Property (*=required)	Type	Description
OldTax	Boolean	Determines whether or not to look at tax 1 or 2 information for tax code information.
Paid	Boolean	Payment status of invoice.
ProductToApplyTax1Total	Decimal	Product to apply to tax 1
ProductToApplyTax2Total	Decimal	Product to apply to tax 2
ProductTotal	Decimal	Product total time
ProductTotalAdj	Decimal	Total product adjustments
Status	String	Read-only. Invoice status. One- to three-letter code from InvoiceStatus table. e.g. CR
StatusChangeById	String	Id of the resource who changed the status
StatusChangeDate	Date	Date when status was changed
StatusChangeReason	String	Description of reason for status change
SupportTotal	Decimal	Support time total
SupportTotalAdj	Decimal	Total support adjustments
Tax1Total	Decimal	Total taxes on invoice
Tax1TotalAdj	Decimal	Total tax adjustments
Tax2Total	Decimal	Total tax 2, if any
Tax2TotalAdj	Decimal	Total tax 2 adjustments, if any
TaxOnRecordDate	Boolean	Read-only. If this flag is selected, the contract uses the record date to calculate taxes.

Property (*=required)	Type	Description
TaxCodeAmount(ByVal Index As Short)	Decimal	Amount for a tax code is returned by the index from the list of tax code amounts.
TaxCodeId(ByVal Index As Short)	String	A TaxCodeId is returned from the index from the list of tax code IDs.
TaxCodeName(ByVal Index As Short)	String	A TaxCodeName is returned from the index from the list of tax code names.
TaxCodeRate(ByVal Index as Short)	Decimal	A TaxCodeRate is returned by the index from the list of tax code rates.
TaxCount	Short	Count of number or taxes under new tax structure.
TimeTotal	Decimal	Total time amount summary
TimeTotalAdj	Decimal	Time adjustments
WoExpenseTotal	Decimal	Summary total of expense write offs
WOProductTotal	Decimal	Total product write offs
WOSupportTotal	Decimal	Support write off time
WOTimeTotal	Decimal	Total time written off

Related information

"ApiInvAdditionalItem" on page 328

"ApiInvExpense" on page 329

"ApiInvFixedFee" on page 330

"ApiInvPaymentInfo" on page 331

"ApiInvProduct" on page 332

"ApiInvSupportTime" on page 334

"ApiInvTime" on page 335

"ApiInvWOExpenses" on page 336

"ApiInvWOProduct" on page 338

"ApiInvWOSupport" on page 339

"ApiInvWOTime" on page 340

ApiInvoice: AddPaymentInfo

```
Public Function AddPaymentInfo(ByRef sId As String, ByVal Payment As
ApiInvPaymentInfo) As Int32
```

Purpose

Add a payment information record to an invoice.

Parameters

Parameter (*=required)	Description
*sId	PaymentInfoId
*Payment	See the properties for the ApiInvPaymentInfo object

Returns

0 = Success

Nonzero = Error

Returns Newly created PaymentInfoId in sId upon success.

Remarks

If the net sum of invoice total minus invoice payments is greater than 0 and the status of the invoice was not "PP" (Partially Paid), then the status is set to "PP".

If the net sum is less than or equal to 0 and the status was not "P" (Paid), then the status is set to "P". Note that the API can overpay an invoice (since the application can) and both positive and negative payments are accepted.

The error returns also is written to log file with error message.

Example

```
Dim myInvoice As New ApiInvoice
Dim myPaymentInfo As New ApiInvPaymentInfo
Dim sId As String = ""
Dim iRet As Int32

myInvoice.CPConnection = myCon

With myPaymentInfo
   .InvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
   .PaymentAmount = 5000
   .PaymentCurrencyCode = "USD"
   .PaymentDate = "05/18/2007"
End With

iRet = myInvoice.AddPaymentInfo(sId, myPaymentInfo)
```

Related information

"ApiInvoice" on page 307

"ApiInvPaymentInfo" on page 331

ApiInvoice: CheckPaymentTotal

```
Public Function CheckPaymentTotal(ByVal InvoiceId As String, ByVal
dInvoiceTotal As Decimal) As Int32
```

Purpose

Check whether the invoice total amount is fully paid.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

0 when the invoice amount is fully paid.

Remarks

None

Example

```
Dim myInvoice As New ApiInvoice
Dim iRet As Int32

myInvoice.CPConnection = myCon
'All paid, so returns 0
iRet = myInvoice.CheckPaymentTotal("{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}",
8000)
```

Related information

"ApiInvoice" on page 307

ApiInvoice: GetAdditionalItems

```
Public Function GetAdditionalItems(ByVal InvoiceId As String) As Hashtable
```

Purpose

Retrieve additional item information of an invoice.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

ApiInvAdditionalItem objects in a Hashtable if any exist.

Remarks

None

Example

```
Dim oRet As New Hashtable
Dim myInvoice As new ApiInvoice
Dim sInvoiceId As String

sInvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
```

```
myInvoice.CPConnection = myCon
oRet = myInvoice.GetAdditionalItems(sInvoiceId)
```

Related information

"ApiInvoice" on page 307

"ApiInvAdditionalItem" on page 328

ApiInvoice: GetExpenseWriteOffs

```
Public Function GetExpenseWriteOffs(ByVal InvoiceId As String) As Hashtable
```

Purpose

Retrieve all expenses that were written off on an invoice.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

ApiInvWOExpenses objects in a Hashtable if any exist.

Remarks

None

Example

```
Dim oRet As New Hashtable
Dim myInvoice As new ApiInvoice
Dim sInvoiceId As String

sInvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
myInvoice.CPConnection = myCon
oRet = myInvoice.GetExpenseWriteOffs(sInvoiceId)
```

Related information

"ApiInvoice" on page 307

"ApiInvWOExpenses" on page 336

ApiInvoice: GetInvoice

```
Public Function GetInvoice(ByVal InvoiceNumber As String) As ApiInvoice
```

Purpose

Retrieves the invoice for the specified invoice number.

Parameters

Parameter (*=required)	Description
*InvoiceNumber	Invoice number.

Returns

A single ApiInvoice object.

Remarks

The method determines the invoiceid by looking up the invoice with the field DisplayInvoiceNumber = InvoiceNumber in the invoice table. It then calls the method GetInvoiceById to retrieve the invoice information.

Example

Not available

Related information

"ApiInvoice" on page 307

"ApiInvoice: GetInvoiceByID" on page 316

ApiInvoice: GetInvoiceByID

```
Public Function GetInvoiceByID(ByVal sInvoiceId As String) As ApiInvoice
```

Purpose

Retrieves an invoice record for InvoiceId provided.

Parameters

Parameter (*=required)	Description
*sInvoiceId	ID of the invoice.

Returns

A single ApiInvoice object.

Remarks

This method determines the tax information to be returned based on the value of the OLDTax field in the invoice (returned as the property OldTax). If OldTax is set to 1, then the tax information returned is the value in Tax1Total, etc.

If the value of OldTax is 0 then there could be many taxes. In this case, TaxCount will indicate the number of elements in each of the Tax Arrays: TaxCodeID, TaxCodeAmount, TaxCodeName and TaxCodeRate. These arrays will have no values if it is OldTax.

If it is new tax, the fields Tax1Total, Tax2Total, Tax1TotalAdj and Tax2TotalAdj will have the values 0.

Example

Not available

Related information

"ApiInvoice" on page 307

ApiInvoice: GetInvoicedExpenses

```
Public Function GetInvoicedExpenses(ByVal InvoiceId As String) As Hashtable
```

Purpose

Retrieve all invoiced expenses on an invoice.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

ApiInvExpense objects in a Hashtable if any exist.

Remarks

None

Example

```
Dim oRet As New Hashtable
Dim myInvoice As new ApiInvoice
Dim sInvoiceId As String

sInvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
myInvoice.CPConnection = myCon
oRet = myInvoice.GetInvoicedExpenses(sInvoiceId)
```

Related information

"ApiInvoice" on page 307

"ApiInvExpense" on page 329

ApiInvoice: GetInvoicedFixedFees

```
Public Function GetInvoicedFixedFees(ByVal InvoiceId As String) As Hashtable
```

Purpose

Retrieve all invoiced fixed fees on an invoice.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

ApiInvFixedFee objects in a Hashtable if any exist.

Remarks

None

Example

```
Dim oRet As New Hashtable
Dim myInvoice As new ApiInvoice
Dim sInvoiceId As String

sInvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
myInvoice.CPConnection = myCon
oRet = myInvoice.GetInvoicedFixedFees(sInvoiceId)
```

Related information

"ApiInvoice" on page 307

"ApiInvFixedFee" on page 330

ApiInvoice: GetInvoicedProduct

```
Public Function GetInvoicedProduct(ByVal InvoiceId As String) As Hashtable
```

Purpose

Retrieve all invoiced products on an invoice.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

ApiInvProduct objects in a Hashtable if any exist.

Remarks

None

Example

```
Dim oRet As New Hashtable
Dim myInvoice As new ApiInvoice
Dim sInvoiceId As String

sInvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
myInvoice.CPConnection = myCon
oRet = myInvoice.GetInvoicedProduct(sInvoiceId)
```

Related information

"ApiInvoice" on page 307

"ApiInvProduct" on page 332

ApiInvoice: GetInvoicedSupportTime

```
Public Function GetInvoicedSupportTime(ByVal InvoiceId As String) As Hashtable
```

Purpose

Retrieve all invoiced support or request time on an invoice.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

ApiInvSupportTime objects in a Hashtable if any exist.

Remarks

None

Example

```
Dim oRet As New Hashtable
Dim myInvoice As new ApiInvoice
Dim sInvoiceId As String

sInvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
myInvoice.CPConnection = myCon
oRet = myInvoice.GetInvoicedSupportTime(sInvoiceId)
```

Related information

"ApiInvoice" on page 307

"ApiInvSupportTime" on page 334

ApiInvoice: GetInvoicedTime

```
Public Function GetInvoicedTime(ByVal InvoiceId As String) As Hashtable
```

Purpose

Retrieve all invoiced time on an invoice.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

ApiInvTime objects in a dictionary if any exist.

Remarks

None

Example

```
Dim oRet As New Hashtable
Dim myInvoice As new ApiInvoice
Dim sInvoiceId As String

sInvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
myInvoice.CPConnection = myCon
oRet = myInvoice.GetInvoicedTime(sInvoiceId)
```

Related information

"ApiInvoice" on page 307

"ApiInvTime" on page 335

ApiInvoice: GetInvoices

```
Public Function GetInvoices(ByVal Batched As Boolean, ByVal Status As
CPInvoiceStatus, Optional ByVal CustomerId As String = "", Optional ByVal
EngagementId As String = "") As DataSet
```

Purpose

Retrieve invoice records that are filtered by parameters.

Parameters

Parameter (*=required)	Description
*Batched	Boolean value. True for specifying to retrieve invoices that already have been batched.
*Status	Status of the invoice. See Remarks.
CustomerId	Retrieves invoice records by customer ID.
EngagementId	Retrieves invoice records by Contract ID.

Returns

A dataset of invoice information.

Remarks

Status of the invoice:

- cpInvCommited
- cpInvSent
- cpInvPaid
- cpInvDraft
- cpInvPending_Approval
- cpInvPending_SecondLevel_Approval
- cpInvApproved

- cpInvPartialPaid

The fields returned in the dataset are as follows: InvoiceID, CustomerID, EngagementID, DisplayInvoiceID (used as InvoiceNumber) and Status.

Example

```
Dim myInvoice As New ApiInvoice
Dim oRet As DataSet

myInvoice.CPConnection = myCon
oRet = myInvoice.GetInvoices(False,CPInvoiceStatus.cpInvSent)
```

Related information

"ApiInvoice" on page 307

ApiInvoice: GetPaymentInfo

```
Public Function GetPaymentInfo(ByVal InvoiceId As String) As Hashtable
```

Purpose

Retrieve all payment information on an invoice.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

ApiInvPaymentInfo objects in Hashtable if any exist.

Remarks

None

Example

```
Dim oRet As New Hashtable
Dim myInvoice As new ApiInvoice
Dim sInvoiceId As String

sInvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
myInvoice.CPConnection = myCon
```

```
oRet = myInvoice.GetPaymentInfo(sInvoiceId)
```

Related information

"ApiInvoice" on page 307

"ApiInvPaymentInfo" on page 331

ApiInvoice: GetPaymentTotal

```
Public Function GetPaymentTotal(ByVal InvoiceId As String) As Decimal
```

Purpose

Retrieves the total of all payments made to an invoice and converts each payment to the invoice currency as necessary.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

The total of the payments.

Remarks

None

Example

Not available

Related information

"ApiInvoice" on page 307

ApiInvoice: GetProductWriteOffs

```
Public Function GetProductWriteOffs(ByVal InvoiceId As String) As Hashtable
```

Purpose

Retrieve all products that were written off on an invoice.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

ApiInvWOProduct objects in a Hashtable if any exist.

Remarks

None

Example

```
Dim oRet As New Hashtable
Dim myInvoice As new ApiInvoice
Dim sInvoiceId As String

sInvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
myInvoice.CPConnection = myCon
oRet = myInvoice.GetProductWriteOffs(sInvoiceId)
```

Related information

"ApiInvoice" on page 307

"ApiInvWOProduct" on page 338

ApiInvoice: GetSupportWriteOffs

```
Public Function GetSupportWriteOffs(ByVal InvoiceId As String) As Hashtable
```

Purpose

Retrieves all request time that was written off on an invoice.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

ApiInvWOSupport objects in a Hashtable if any exist.

Remarks

None

Example

```
Dim oRet As New Hashtable
Dim myInvoice As new ApiInvoice
Dim sInvoiceId As String

sInvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
myInvoice.CPConnection = myCon
oRet = myInvoice.GetSupportWriteOffs(sInvoiceId)
```

Related information

"ApiInvoice" on page 307

"ApiInvWOSupport" on page 339

ApiInvoice: GetTimeWriteOffs

```
Public Function GetTimeWriteOffs(ByVal InvoiceId As String) As Hashtable
```

Purpose

Retrieves all time that was written off on an invoice.

Parameters

Parameter (*=required)	Description
*InvoiceId	ID of the invoice.

Returns

ApiInvWOTime objects in a Hashtable if any exist.

Remarks

None

Example

```
Dim oRet As New Hashtable
Dim myInvoice As new ApiInvoice
Dim sInvoiceId As String

sInvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
myInvoice.CPConnection = myCon
oRet = myInvoice.GetTimeWriteOffs(sInvoiceId)
```

Related information

"ApiInvoice" on page 307

"ApiInvWOTime" on page 340

ApiInvoice: MarkInvoiceAsBatched

```
Public Function MarkInvoiceAsBatched(InvoiceId As String) As Int32
```

Purpose

Mark the invoice identified by this ID as batched.

Parameters

Parameter (*=required)	Description
*InvoiceId	Invoice ID.

Returns

Returns 0 when the invoice identified by this ID has been extracted from the Changepoint database and non-zero on error.

Remarks

Error message of the returned error number is documented in the log file.

The extract date is set to the current date.

The invoice can be retrieved again using GetInvoiceById or through GetInvoices and specifying the batched parameter as true.

Example

Not available

Related information

"ApiInvoice" on page 307

"ApiInvoice: GetInvoiceByID" on page 316

"ApiInvoice: GetInvoices" on page 322

ApiInvAdditionalItem

The ApiInvdditionalItem object represents additional item information for an invoice.

Namespace

Changepoint.ChangepointAPI2.ApiInvAdditionalItem

Methods

None

Properties

Property (*=required)	Type	Description
AdditionalitemID	String	Additional Item ID
Amount	Decimal	Amount of an additional item
AppliedTax1	Boolean	Applied Tax 1
AppliedTax2	Boolean	Applied Tax 2
BillingOfficeId	String	Billing office of the engagement
CustomerId	String	Customer ID

Property (*=required)	Type	Description
Description	String	Description of additional item
EngagementId	String	Engagement ID on invoice
EngagementName	String	Engagement name
InvoiceId	String	Invoice ID

Related information

"ApiInvoice" on page 307

ApiInvExpense

The ApiInvExpense object represents expenses invoiced within the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiInvExpense

Methods

None

Properties

Property	Type	Description
AppliedPercentage	Decimal	Applied percentage
BillingOfficeId	String	Billing office of the engagement
CategoryId	String	Id of expense category
CategoryName	String	Name of expense category
CreditNoteID	String	Credit note ID
CustomerId	String	Customer ID
EngagementId	String	Engagement ID on invoice

Property	Type	Description
EngagementName	String	Name of engagement
ExpenseAmount	Decimal	Amount expensed
ExpenseDate	Date	Date expense was incurred
ExpenseId	String	Id of expense that is has been invoiced
ExpenseTypeId	String	Id of Expense Type
ExpenseTypeName	String	Name of expense type
GlobalWorkGroupId	String	GlobalWorkgroupid of resource who incurred expense
GlobalWorkGroupName	String	Name of global workgroup
InvoicedExpenseId	String	Invoiced expense ID
InvoiceId	String	Invoice ID
ProjectId	String	Project against which the expense was booked
Projectname	String	Name of project
RecoverableTaxAmount	Decimal	Recoverable tax amount
ResourceId	String	Id of resource
ResourceName	String	Name of resource
WorkGroupId	String	Id of workgroup of resource
WorkgroupName	String	Name of workgroup

Related information

"ApiInvoice" on page 307

ApiInvFixedFee

The ApiInvFixedFee object represents fixed fees that were invoiced within the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiInvFixedFee

Methods

None

Properties

Property (*=required)	Type	Description
BillingOfficeId	String	Billing office of the engagement
CreditNoteID	String	Credit note ID
CustomerId	String	Customer ID
EngagementId	String	Engagement ID on invoice
EngagementName	String	Name of engagement
FixedFeeAmount	Decimal	Amount of fixed fee
FixedFeeDate	Date	Date of fixed fee
FixedFeeDeliverable	String	Deliverable description
FixedFeeId	String	Id of fixed fee
InvoiceId	String	Invoice ID

Related information

"ApiInvoice" on page 307

ApiInvPaymentInfo

The ApiInvPaymentInfo object represents payment information associated to an invoice.

Populate the ApiInvPaymentInfo Object to add a payment to an Invoice: The ApiLookup object provides methods to retrieve CustomerId, EngagementId, PaymentCurrency and PaymentExchangeRateId information. The PaymentExchangeRate information is returned but does not have to be passed when creating a payment.

Namespace

Changepoint.ChangepointAPI2.ApiInvPaymentInfo

Methods

None

Properties

Property (*=required)	Type	Description
BillingOfficeId	String	Billing office of the engagement
CustomerId	String	Customer ID
EngagementId	String	Engagement ID on invoice
*InvoiceId	String	Invoice ID
*PaymentAmount	Decimal	Payment amount
*PaymentCurrencyCode	String	Currency of payment applied (Three-letter ISO code)
*PaymentDate	Date	Date of payment
PaymentExchangeRate	Double	Exchange rate from PaymentExchangeRateId
PaymentExchangeRateID	String	Exchange rate ID used on the payment.
PaymentInfoId	String	Payment information ID

Related information

"ApiInvoice" on page 307

ApiInvProduct

The ApiInvProduct object represents products invoiced.

Namespace

Changepoint.ChangepointAPI2.ApiInvProduct

Methods

None

Properties

Property (*=required)	Type	Description
AppliedTax1	Boolean	Applied tax 1
AppliedTax2	Boolean	Applied tax 2
BillingOfficeId	String	Billing office of the engagement
CreditNoteId	String	Credit note ID
CustomerId	String	Customer ID
EngagementId	String	Engagement ID on invoice
EngagementName	String	Name of engagement
EngagementProductId	String	Id of EngagementProduct table
InvoicedCost	Decimal	Invoiced cost
InvoicedProductId	String	Invoiced product ID
InvoiceId	String	Invoice ID
ProductCategoryId	String	Id of product category
ProductCategoryName	String	Name of product category
ProductId	String	Id of product invoiced
ProductName	String	Name of product
Quantity	Int32	Quantity of product invoiced
RequestId	String	Id of associated request
RequestNumber	String	Number of associated request
StandardCost	Decimal	Standard cost

Related information

"ApiInvoice" on page 307

ApiInvSupportTime

The ApiInvSupportTime object represents support or request time that has been invoiced.

Namespace

Changepoint.ChangepointAPI2.ApiInvSupportTime

Methods

None

Properties

Property (*=required)	Type	Description
AppliedCostRate	Decimal	Cost rate
AppliedRate	Decimal	Billing rate applied to time.
BillingOfficeId	String	Billing office of the engagement
CreditNoteId	String	Credit note ID
CustomerId	String	Customer ID
EngagementId	String	Engagement ID on invoice
EngagementName	String	Name of engagement
InvoicedRequestId	String	Invoiced request ID
InvoiceId	String	Invoice ID
OverTimeHours	Decimal	Overtime hours booked
RegularHours	Decimal	Regular hours booked
Requestid	String	Id of request against which time was booked

Property (*=required)	Type	Description
RequestNumber	String	Number of request
RequestTimeId	String	Id of request time record
ResourceId	String	Id of resource who booked time
ResourceName	String	Name of resource
StartTime	Date	Date of time booking

Related information

"ApiInvoice" on page 307

ApiInvTime

The ApiInvTime object represents time that was invoiced in Changepoint.

Namespace

Changepoint.ChangepointAPI2.ApiInvTime

Methods

None

Properties

Property (*=required)	Type	Description
AppliedCostRate	Decimal	Cost rate supplied on contract
AppliedRate	Decimal	Billing rate supplied on contract
BillingOfficeId	String	Billing office of the contract
CreditNoteId	String	Credit note ID
CustomerId	String	Customer ID
EngagementId	String	Engagement ID on invoice

Property (*=required)	Type	Description
EngagementName	String	Name of contract pointed to by contract
GlobalWorkGroupId	String	Global work group ID of resource
GlobalWorkGroupName	String	Name of global work group pointed to by global work group ID
InvoicedTimeID	String	Invoiced time ID
InvoiceId	String	Invoice ID
OverTimeHours	Decimal	Number of overtime hours
ProjectId	String	Project against which time was booked
ProjectName	String	Name of the project pointed to by the project ID
RegularHours	Decimal	Number of regular hours
ResourceId	String	Id of resource
ResourceName	String	Name of resource pointed to by resourceId
TaskId	String	Task against which the time was booked
TaskName	String	Name of task pointed to by task ID
TimeDate	Date	Time date
TimeId	String	Time ID
WorkgroupId	String	Workgroup of resource
WorkgroupName	String	Name of workgroup pointed to by workgroup ID

Related information

"ApiInvoice" on page 307

ApiInvWOExpenses

The ApiInvWOExpenses object represents expense that was written off on an invoice.

Namespace

Changepoint.ChangepointAPI2.ApiInvWOExpenses

Methods

None

Properties

Property (*=required)	Type	Description
AppliedPercentage	Decimal	Applied percentage
BillingOfficeId	String	Billing office of the engagement
CategoryId	String	Id of expense category
CategoryName	String	Name of expense category
CreditNoteId	String	Credit note ID
CustomerId	String	Customer ID
EngagementId	String	Engagement ID on invoice
EngagementName	String	Name of engagement
ExpenseAmount	Decimal	Amount expensed
ExpenseDate	Date	Date expense was incurred
ExpenseId	String	Id of expense that has been invoiced
GlobalWorkgroupId	String	Global Workgroup ID of resource who incurred expense
GlobalWorkGroupName	String	Name of global workgroup
InvoiceId	String	Invoice ID
ProjectId	String	Project against which the expense was booked
ProjectName	String	Name of project
RecoverableTaxAmount	Decimal	Recoverable tax amount
ResourceId	String	Id of resource

Property (*=required)	Type	Description
ResourceName	String	Name of resource
WorkgroupId	String	Id of workgroup of resource
WorkgroupName	String	Name of workgroup
WriteOffAmount	Decimal	Amount of product written off
WriteOffExpenseId	String	Id of expense being written off
WriteOffReason	String	Reason for write off
WriteOffReasonId	String	Id of write off description

Related information

"ApiInvoice" on page 307

ApiInvWOProduct

The ApiInvWOProduct object represents product that was written off on an invoice.

Namespace

Changepoint.ChangepointAPI2.ApiInvWOProduct

Methods

None

Properties

Property (*=required)	Type	Description
BillingOfficeId	String	Billing office of the engagement
CreditNoteId	String	Identifies the Credit note.
CustomerId	String	Identifies the Customer.
EngagementId	String	Identifies the contract on the invoice.

Property (*=required)	Type	Description
EngagementName	String	Name of engagement
EngagementProductId	String	Id of EngagementProduct table
	String	Invoice ID
ProductCategoryId	String	Id of product category
ProductCategoryName	String	Name of product category
ProductId	String	Id of product invoiced
ProductName	String	Name of product
RequestId	String	Id of associated request
RequestNumber	String	Number of associated request
WriteOffAmount	Decimal	Amount of product written off
WriteOffReason	String	Reason for write off
WriteOffReasonId	String	Id of write off description
WriteOffProductId	String	Id of write off product record

Related information

"ApiInvoice" on page 307

ApiInvWOSupport

The ApiInvWOSupport object represents all request time that was written off on an invoice.

Namespace

Changepoint.ChangepointAPI2.ApiInvWOSupport

Methods

None

Properties

Property (*=required)	Type	Description
AppliedCostRate	Decimal	Cost rate
AppliedRate	Decimal	Billing rate applied to time
BillingOfficeId	String	Billing office of the contract
CreditNoteId	String	Credit note ID
CustomerId	String	Customer ID
EngagementId	String	Engagement ID on invoice
EngagementName	String	Name of contract pointed to by EngagementId
InvoiceId	String	Invoice ID
RequestId	String	Id of request for which time was written off
RequestNumber	String	Number of request
RequestTimeId	String	Id of time record that was written off
ResourceId	String	Id of resource
ResourceName	String	Name of resource pointed to by resource ID
StartTime	Date	Date of request time booking
WriteOffAmount	Decimal	Amount of write-off in hours
WriteOffReason	String	Text of reason for write off
WriteOffReasonId	String	Id of write off reason
WriteOffRequestTimeId	String	Write off request time ID

Related information

"ApiInvoice" on page 307

ApiInvWOTime

The ApiInvWOTime object represents all time that was written off on an invoice.

Namespace

Changepoint.ChangepointAPI2.ApiInvWOTime

Methods

None

Properties

Property (*=required)	Type	Description
AppliedCostRate	Decimal	Cost rate
AppliedRate	Decimal	Billing rate applied to time
BillingOfficeId	String	Billing office of the contract
CreditNoteId	String	Credit note ID
CustomerId	String	Customer ID
EngagementId	String	Engagement ID on invoice
EngagementName	String	Name of contract pointed to by EngagementId
GlobalWorkGroupId	String	Global workgroup ID of resource
GlobalWorkGroupName	String	Name of global workgroup pointed to by GlobalWorkGroupId
InvoiceId	String	Invoice ID
ProjectId	String	Project against which time was booked
ProjectName	String	Name of project pointed to by ProjectId
ResourceId	String	Id of resource
ResourceName	String	Name of resource pointed to by resourceId
TaskId	String	Task against which the time was booked
TaskName	String	Name of task pointed to by TaskId
TimeDate	Date	Date of time booking

Property (*=required)	Type	Description
TimeId	String	Id of time record that was written off
WorkgroupId	String	Workgroup of resource
WorkgroupName	String	Name of workgroup pointed to by WorkgroupId
WriteOffAmount	Decimal	Amount of write off in hours
WriteOffReason	String	Text of reason for write off
WriteOffReasonId	String	Id of write off reason
WriteOffTimeID	String	Write off time ID

Related information

"ApiInvoice" on page 307

ApiKnowledgeManagement

The ApiKnowledgeManagement object allows users to retrieve, create, edit or delete knowledge items in the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiKnowledgeManagement

Methods

Properties

Property	Type	Description
AllowEdit(ByVal Index As Short)	String	A ResourceId is returned by the index from the list of resource IDs that have edit access.
AllowEditCount	Short	The number of resources that have edit access
AllowEditName(ByVal Index As Short)	String	A resource name is returned by the index from the list of resource names that have edit access.
AllowView(ByVal Index As Short)	String	A ResourceID is returned by the index from the list of resource IDs that have view access.
AllowViewCount	Short	The number of resources with view access.
AllowViewName(ByVal Index As Short)	String	A resource name is returned by the index from the list of resource names that have view access.
attachment	Object	The attachment
AttachmentId	String	The attachment ID
Author	String	Author (created by)
CampaignCategoryId	String	Campaign category ID
*Category	String	The knowledge managment CategoryId

Property	Type	Description
ChangepointLink	String	Name of the Changepoint entity linked to the knowledge item.
CheckedOut	Boolean	Flag that indicates whether the item is checked out or not.
CheckedOutBy	String	Resource ID associated with the user that checked out knowledge item
CheckedOutOn	String	The date the knowledge item was checked out
Comment	String	Comment
Confidential	Boolean	Confidential
*CPConnection	ApiConnection	Connection to the Changepoint database
CreatedBy	String	Who Created the record
CreatedOn	Date	When the record was created
CustomerId	String	Customer related to the knowledge item
Deleted	Boolean	Flag that indicates if the object has been deleted.
Description	String	Description
ExpiryDate	Date	Expiry date of knowledge item
EngagementId	String	Related Engagement
Favorite	Boolean	Marks the knowledge item as a favorite
FileDescription	String	File description
FileExtension	String	Attachment file extension
FileName	String	File name (name only)

Property	Type	Description
ImportedProject	Boolean	Imported project
IsAvailable	Short	Marks knowledge item as being available
IsShared	Boolean	Indicates whether knowledge item is shared
Keywords	String	Keywords used for searching in Changepoint
KMCode1	String	KM Code1
KMCode2	String	KM Code2
KMCode3	String	KM Code3
KMText1	String	KM Text1
KMText2	String	KM Text2
KMText3	String	KM Text3
Password	String	Password
ProductCategoryId	String	Product category ID
ProjectId	String	The related project
ReferenceId	String	Reference ID
ReferenceTable	String	Reference table
ResourceId	String	Related resource
RestrictAttachment	Boolean	Restrict attachment
SortOrder	String	Sort order for knowledge item
*SubCategory	String	Sub category ID
*Title	String	Title

Property	Type	Description
UpdatedBy	String	Resource who last updated the record
UpdatedOn	Date	The date of the last update to the record
Version	Short	Knowledge item version number
WebLink	String	Web link

Related information

"ApiKMVersion" on page 359

ApiKnowledgeManagement: CreateKM

```
Public Function CreateKM(ByVal mKM As ApiKnowledgeManagement) As Int32
```

Purpose

Create a new knowledge management item in Changepoint

Parameters

Parameter (*=required)	Description
*mKM	ApiKnowledgeManagement Object

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

```
Dim myKMItem as New ApiKnowledgeManagement
Dim iRet as Int32 = 0

myKMItem.CPConnection = myCon
With myKMItem
   .AllowViewName(0, "John Smith")
```

```
    .AllowViewName(1, "Jane Doe")
    .Author = "Mel Torn"
    .CheckedOut = False
    .CustomerId = "{f6c77d82-3ada-11d4-81d9-00104b7ad9cd }"
    .Title = "MyTestKMItem"
    .FileName = "MyTestKMItem.Doc"
    .Attachment = myAttachment

End With
iRet = myKMItem.CreateKM(myKMItem)
If iRet <> 0 Then
    'Handle error
End If
```

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: DeleteKMById

```
Public Function DeleteKMById(ByVal sKMId As String) As Int32
```

Purpose

Delete a knowledge management item from Changepoint

Parameters

Parameter (*=required)	Description
*sKMId	ID of the knowledge management item to delete

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

Not available

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: DeleteKMByName

```
Public Function DeleteKMByName(ByVal sKMName As String) As Int32
```

Purpose

Delete a knowledge management item based on the name of the item.

Parameters

Parameter (*=required)	Description
*sKMName	The title of the knowledge management item

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

Not available

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: GetCPLinkList

```
Public Function GetCPLinkList(ByVal sRefType As String) As DataSet
```

Purpose

Return a list of related Changepoint entities based on the ReferenceType

Parameters

Parameter (*=required)	Description
*sRefType	Supported types are: Project, Task, Customer, Contact, Campaign, Competitor, Opportunity, Activity, Contract, Resource, Product, RequestScenario, ProjPortfolioScenario, ResDemandScenario, CandidateScenario

Returns

DataSet

Columns returned vary slightly based on the reference type as follows:

- Project, Customer, Contract, Resource return: (Id, Name, AlternateName)
- All other types return: (Id, Name)

Remarks

None

Example

Not available

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: GetKMById

```
Public Function GetKMById(ByVal sKMId As String) As ApiKnowledgeManagement
```

Purpose

Retrieve a knowledge management item based on the ID of the item

Parameters

Parameter (*=required)	Description
*sKMId	Knowledge management item ID as a GUID string

Returns

ApiKnowledgeManagement object

Remarks

None

Example

Not available

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: GetKMByName

```
Public Function GetKMByName(ByVal sKMName As String) As ApiKnowledgeManagement
```

Purpose

Retrieve a knowledge management item by the title of the item.

Parameters

Parameter (*=required)	Description
*sKMName	Knowledge management item title

Returns

ApiKnowledgeManagement object

Remarks

There is a risk of more than one knowledge management item with the same name. The method returns the first knowledge management item in the list and ignores any others.

Example

Not available

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: GetKMCategories

```
Public Function GetKMCategories() As DataSet
```

Purpose

Return a list of knowledge management categories

Parameters

None

Returns

DataSet (CategoryId, Name)

Remarks

Retrieves categories not subscribed to and those subscribed to by the logged in resource.

Example

Not available

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: GetKMCodeFields

```
Public Function GetKMCodeFields(ByVal sResourceId As String) As Array
```

Purpose

Return the set of usable code fields and their allowable options based on the specified ResourceId

Parameters

Parameter (*=required)	Description
*sResourceId	The logged in resource or related resource to the knowledge management item

Returns

A two dimensional array containing code fields and their options

Remarks

The ResourceId parameter filters the returned values to the code fields which the resource has access to.

Example

Not available

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: GetKMs

```
Public Function GetKMs(Optional ByVal sSearchString As String = "", Optional
ByVal sKMCategory As String = "", Optional ByVal sKMSubCategory As String =
"", Optional ByVal sKMCode1 As String = "", Optional ByVal sKMCode2 As String
= "", Optional ByVal sKMCode3 As String = "") As DataSet
```

Purpose

Retrieve a list of knowledge management items based on specified filter criteria.

Parameters

Parameter (*=required)	Description
sSearchString	Title search string. If the title search string is empty, returns all
sKMCategory	Category ID.
sKMSubCategory	Subcategory ID.
sKMCode1	Code1 ID.
sKMCode2	Code2 ID.
sKMCode3	Code3 ID.

Returns

DataSet (AttachmentId, Title, Category, SubCategory, Rank)

Remarks

Extracts knowledge management items based on the parameters but also limits the returned items to those the logged in resource has access to.

Example

Not avaliable

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: GetKMSubCategories

```
Public Function GetKMSubCategories(ByVal sKMCategory As String) As DataSet
```

Purpose

Return a list of knowledge management subcategories based on the knowledge management category and logged in user

Parameters

Parameter (*=required)	Description
*sKMCategory	Category whose subcategories are being requested.

Returns

DataSet (Code, Description)

Remarks

The returned list is filtered based on KMCategory and includes all sub categories that are not subscribed to by the logged in resource and all those that are subscribed to by the logged in resource.

Example

Not available

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: GetKMTextFields

```
Public Function GetKMTextFields(ByVal sResourceId As String) As Array
```

Purpose

Return a list of code and/or text fields along with their allowable options

Parameters

Parameter (*=required)	Description
*sResourceId	The logged in resource or related resource to the knowledge management item

Returns

A two dimensional array containing code fields and their options

Remarks

None

Example

Not available

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: GetResourcesForKM

```
Public Function GetResourcesForKM(ByVal sResourceId As String) As DataSet
```

Purpose

Return a list of preferred resources based on a specified resource.

Parameters

Parameter (*=required)	Description
*sResourceId	The logged in resource or resource related to the knowledge management item.

Returns

DataSet (Name, ResourceId, AlternateName)

Remarks

The returned list is determined by sResourceId being a member of a Workgroup and having a list of preferred resources defined in Changepoint.

Example

Not available

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: GetWorkgroupForKM

```
Public Function GetWorkgroupForKM(ByVal sResourceId As String) As DataSet
```

Purpose

Return a list of Workgroups based on a specified resource.

Parameters

Parameter (*=required)	Description
*sResourceId	The logged in resource or related resource to the knowledge management item

Returns

DataSet (WorkgroupId, Name)

Remarks

The returned recordset is limited to the Workgroups to which sResourceId is a current member.

Example

Not available

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: UpdateKMById

```
Public Function UpdateKMById(ByVal sKMId As String, ByRef mKM As
ApiKnowledgeManagement) As Int32
```

Purpose

Update Changepoint with new knowledge management data

Parameters

Parameter (*=required)	Description
*sKMId	ID of the knowledge management Item
*mKM	The ApiKnowledgeManagement Object whose data is to be written to the database

Returns

0 = Success

Nonzero = Error

Remarks

The sKMId is used to identify the knowledge management item to update.

Example

```
Dim myKMItem as New ApiKnowledgeManagement
Dim iRet as Int32 = 0
Dim sAttId as String = ""

myKMItem.CPConnection = myCon
myKMItem = myKMItem.GetKMByName("MyTestKMItem")
sAttId = myKMItem.AttachmentId
If sAttId <> "" Then
With myKMItem
   .AllowViewName(2, "Brian Jones")
   .Title = "MyModifiedTestKMItem"
   .Attachment = myUpdatedAttachment
   .ExpiryDate = DateAdd("mmm", 6, Date)

End With
lRet = myKMItem.UpdateKMById (sAttId , myKMItem)
If iRet <> 0 Then
'Handle error
End If
```

Related information

"ApiKnowledgeManagement" on page 342

ApiKnowledgeManagement: UpdateKMByName

```
Public Function UpdateKMByName(ByVal sKMName As String, ByVal mKM As
ApiKnowledgeManagement) As Int32
```

Purpose

Update an existing knowledge management item in Changepoint with new data

Parameters

Parameter (*=required)	Description
*sKMName	Title of the knowledge management item to update
*mKM	ApiKnowledgeManagement Object

Returns

0 = Success

Nonzero = Error

Remarks

KMName is used to identify the knowledge management item to update.

Example

```
Dim myKMItem as New ApiKnowledgeManagement
Dim iRet as Int32 = 0

myKMItem.CPConnection = myCon
myKMItem = myKMItem.GetKMByName("MyTestKMItem")
If myKMItem.AttachmentId <> "" Then
With myKMItem
   .AllowViewName(2, "Brian Jones")
   .Title = "MyModifiedTestKMItem"
   .Attachment = myUpdatedAttachment
   .ExpiryDate = DateAdd("mmm", 6, Date)

End With
iRet = myKMItem.UpdateKMByName ("MyTestKMItem" , myKMItem)
If iRet <> 0 Then
   'Handle error
End If
```

Related information

"ApiKnowledgeManagement" on page 342

ApiKMVersion

The ApiKMVersion object represents version information for knowledge items.

Namespace

Changepoint.ChangepointAPI2.ApiKMVersion

Methods

Properties

Property (*=required)	Type	Description
Attachment	Object	Attachment
*AttachmentId	String	Attachment ID
*Category	String	Knowledge management category
Comment	String	Comment
Confidential	Boolean	Confidential
*CPConnection	ApiConnection	Connection to the Changepoint database
CreatedBy	String	The created resource ID
CreatedOn	Date	Date knowledge item was created.

Property (*=required)	Type	Description
Deleted	Boolean	Flag that indicates if the object has been deleted.
Description	String	Description
ExpiryDate	Date	Expiry date
FileDescription	String	File description
FileExtension	String	Attachment file extension
FileName	String	File Name
IsShared	Boolean	Is Shared
Keywords	String	Keywords
KMCode1	String	KM Code1
KMCode2	String	KM Code2
KMCode3	String	KM Code3
KMText1	String	KM Text1
KMText2	String	KM Text2
KMText3	String	KM Text3
Password	String	Password
ProjectId	String	Project ID
ReferenceId	String	Reference ID
ReferenceTable	String	Reference Table
RestrictAttachment	Boolean	Restrict Attachment
SortOrder	String	Sort order

Property (*=required)	Type	Description
*SubCategory	String	Subcategory
*Title	String	Title
Version	Short	Version number of knowledge item
VersionId	String	Version ID
WebLink	String	Web Link

Related information

"ApiKnowledgeManagement" on page 342

ApiKMVersion: GetVersionControl

```
Public Function GetVersionControl(ByVal AttachmentId As String) As DataSet
```

Purpose

Retrieves version control information of Knowledge Item.

Parameters

Parameter (*=required)	Description
*AttachmentId	The attachment ID

Returns

A dataset fo the knowledge item record

Remarks

Returned columns are version, CheckedoutBy, CheckedoutOn, CheckedFlag, comment, CheckedControl, VersionControl, and versionlimit

Example

Not available

Related information

"ApiKMVersion" on page 359

ApiKMVersion: KMDeleteBatchVersionItems

```
Public Function KMDeleteBatchVersionItems(ByVal sAttachmentId As String, ByVal
sCategory As String, ByVal bDeleteAllFlag As Boolean) As Int32
```

Purpose

Deletes history versions of knowledge items.

Parameters

Parameter (*=required)	Description
*sAttachmentId	The Attachment ID. Can be empty.
*sCategory	The Category ID. Can be empty.
*bDeleteAllFlag	If set to true, deletes all versions.

Returns

0 = Success

Nonzero = Error

Remarks

If bDeleteAllFlag is true, delete all versions of knowledge item by attachment ID or Category ID; otherwise delete specific versions of knowledge item.

Example

Not available

Related information

"ApiKMVersion" on page 359

ApiKMVersion: KMDeleteVersionItem

```
Public Function KMDeleteVersionItem(ByVal sAttachmentId As String, ByVal
sVersionid As String) As Int32
```

Purpose

Deletes a history version of knowledge item.

Parameters

Parameter (*=required)	Description
*sAttachmentId	Attachment ID.
*sVersionid	Version ID.

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

Not available

Related information

"ApiKMVersion" on page 359

ApiKMVersion: KMResetCheckedout

```
Public Function KMResetCheckedout(ByVal sAttachmentId As String) As Int32
```

Purpose

Changes the checkout flag to false if a person who checked out this knowledge item no longer has the edit permission for this knowledge item.

Parameters

Parameter (*=required)	Description
*sAttachmentId	Attachment ID.

Returns

0 = Success

Nonzero = Error

Remarks

Resets checkout to 0 if user has been removed from edit permission list. This function resets the checkout flag when the user's permission is changed to read-only.

In knowledge management version control, the Edit Knowledge feature is required in order to check out a knowledge item. Once a knowledge item is checked out, other users can't edit it.

If the user has the Override Check-Out feature, they can cancel the checked out flag by selecting Check In from the menu in Enterprise.

Example

Not available

Related information

"ApiKMVersion" on page 359

ApiKMVersion: KMUpdateVersion

```
Public Function KMUpdateVersion(ByVal mKM As ApiKMVersion, ByVal AttachUpdate
As Boolean) As Int32
```

Purpose

Adds a new version history of knowledge item by updating current knowledge item.

Parameters

Parameter (*=required)	Description
*mKM	ApiKMVersion. A version object of knowledge item.
*AttachUpdate	The flag if only attachment file has been changed.

Returns

0 = Success

Nonzero = Error

Remarks

AttachUpdate must set to False so that the current version of knowledge item can be updated as well when creating a history version of knowledge item.

If AttachUpdate is set to True, it adds a new version of a knowledge item to KMVersion table and updates only version number, updatedby, updatedon of the current knowledge item in the KMAttachment.

Before using this method, it is recommended to use the GetVersionControl method to check whether version control has been enabled for the knowledge item. In order to update the knowledge item version, version control must be enabled for the knowledge item, and the item must not be checked out.

Example

```
Dim myKMVersion As New ApiKMVersion
Dim iRet As Int32

myKMVersion.CPConnection = myCon

With myKMVersion
   .AttachmentId = "{75606817-003F-491F-AEA9-4B056B0BFEFB}"
   .Title = "ITG Data Relationships"
   .Description = "ITG Data Relationships"
   .ProjectId = "{00000000-0000-0000-0000-000000000000}"
   .Category = "{B11C9334-061E-478E-9378-6E1C79C1E364}"
   .SubCategory = "{4B7A0932-D646-40BE-B902-D7DA8E12C7C9}"
   .Keywords = "ITG; Data Relationships"
   .IsShared = True
   .ExpiryDate = "05/18/2007"
```

```
   ...
End With

iRet = myKMVersion.KMUpdateVersion(myKMVersion, False)
```

Related information

"ApiKMVersion" on page 359

"ApiKMVersion: GetVersionControl" on page 361

ApiKMVersion: LoadKMArticleHist

```
Public Function LoadKMArticleHist(ByVal sVerId As String) As ApiKMVersion
```

Purpose

Retrieve knowledge item by version ID.

Parameters

Parameter (*=required)	Description
*sVerId	Version ID

Returns

ApiKMVersion object. A version object of knowledge item.

Remarks

None

Example

```
Dim myKMVersion As New ApiKMVersion
Dim mKMVersion As New ApiKMVersion

myKMVersion.CPConnection = myCon

mKMVersion = myKMVersion.LoadKMArticleHist("{2EE84D3E-AB3F-45C0-AB9F-
86EB94DB5F09}")
```

Related information

"ApiKMVersion" on page 359

ApiLookup

The ApiLookup object provides common lookup functions.

Namespace

Changepoint.ChangepointAPI2.ApiLookup

Methods

Properties

Property (*=required)	Type	Description
*CPConnection	ApiConnection	Write only. The Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. The ApiException object which holds the exception information.

ApiLookup: GetBillingOffices

```
Public Function GetBillingOffices(Optional ByVal sBillingOfficeId As String =
"") As DataSet
```

Purpose

Retrieve the billing office ID and description for billing offices.

Parameters

Parameter (*=required)	Description
sBillingOfficeId	ID of the billing office to be retrieved

Returns

A dataset of billing office IDs and descriptions.

Remarks

If there is no value for sBillingOfficeId, then returns all billing offices; otherwise, only returns the specified billing office.

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetCampaigns

```
Public Function GetCampaigns () As DataSet
```

Purpose

Returns the CampaignId, Name for all campaigns.

Parameters

None

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetCertificationCycle

```
Public Function GetCertificationCycle() As DataSet
```

Purpose

Returns the ID and description for all certification cycle codes.

Parameters

None

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetCodeByDescription

```
Public Function GetCodeByDescription(ByVal myDescription As String, ByVal
myTableName As String) As String
```

Purpose

Retrieve the Changepoint code of the property described by the variable myDescription.

Parameters

Parameter (*=required)	Description
*myDescription	Property Description.
*myTableName	Table Name to look up.

Remarks

This method can be used for a Changepoint table with Description and Code columns.

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetCodeDetail

```
Public Function GetCodeDetail(ByVal myCodeType As String) As DataSet
```

Purpose

Retrieve code detail and description of the code specified by myCodeType in a dataset.

Parameters

Parameter (*=required)	Description
*myCodeType	Type of the Code.

Remarks

This method can be used for Changepoint table with CodeDetail, Description and CodeType columns.

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetContactsByCustomerId

```
Public Function GetContactsByCustomerId(ByVal sCustomerId As String) As
DataSet
```

Purpose

Returns the contact ID and name of the customer.

Parameters

Parameter (*=required)	Description
*sCustomerId	ID of the customer associated with the contact ID

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetContactsbyCustomerName

```
Public Function GetContactsbyCustomerName(ByVal customerName As String,
Optional ByVal contactFirstName As String = "", Optional ByVal contactLastName
As String = "") As DataSet
```

Purpose

Retrieves the contact ID for the specified ContactFName, ContactLName, and CustomerName.

Parameters

Parameter (*=required)	Description
*CustomerName	The name of the customer to which this contact is associated.
ContactFName	The first name of the contact to look for.
ContactLName	The last name of the contact to look for.

Returns

ContactId, Name in a DataSet

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetCountryCode

```
Public Function GetCountryCode() As DataSet
```

Purpose

Returns Country Code and Description fields.

Parameters

None

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetCurrency

```
Public Overloads Function GetCurrency(Optional ByVal currencyDescription As
String = "") As DataSet
```

Purpose

Returns currency code and description fields.

Parameters

Parameter (*=required)	Description
CurrencyDescription	Currency description, for example, "Canadian".

Remarks

If CurrencyDescription is not specified, returns all enabled currencies defined within Changepoint.

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetEmployeeTypes

```
Public Function GetEmployeeTypes() As DataSet
```

Purpose

Returns the code and description fields from the EmployeeType table.

Parameters

None

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetFeatures

```
Public Function GetFeatures(Optional ByVal roleId As String = "") As DataSet
```

Purpose

Retrieves the feature code and name for the specified role ID.

Parameters

Parameter (*=required)	Description
RoleId	Role ID.

Remarks

If the Role ID string is empty, returns all features.

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetFullName

```
Public Function GetFullName(ByVal firstName As String, ByVal middleName As
String, ByVal lastName As String) As String
```

Purpose

Retrieves a full name with system defined format.

Parameters

Parameter (*=required)	Description
*firstName	First Name
*middleName	Middle Name
*lastName	Last Name

Returns

Full name

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetGlobalWorkgroupIds

```
Public Function GetGlobalWorkgroupIds(Optional ByVal WorkgroupId As String =
"") As DataSet
```

Purpose

Retrieves the global workgroup ID and name fields of the global workgroup.

Parameters

Parameter (*=required)	Description
WorkgroupId	ID of the workgroup.

Returns

A dataset of the GlobalWorkgroupId and Name

Remarks

If a parameter is not provided, the function will return all GlobalWorkgroups.

Example

```
Dim myApi as New ApiLookup ()
myApi.CPConnection=myCon
Dim retId as String="{}"
Dim ret as DataSet = myApi.GetGlobalWorkgroupIds (retId)
```

Related information

"ApiLookup" on page 367

ApiLookup: GetGlobalWorkgroups

```
Public Function GetGlobalWorkgroups(Optional ByVal globalWorkGroupName As
String = "") As DataSet
```

Purpose

Retrieves the global workgroup ID and name fields of the global workgroup.

Parameters

Parameter (*=required)	Description
globalWorkGroupName	The name of global workgroup.

Returns

GlobalWorkgroupId and Name in a DataSet

Remarks

If the globalWorkGroupName string is empty or no parameter is passed, returns all GlobalWorkgroups.

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetHelpDesks

```
Public Function GetHelpDesks() As DataSet
```

Purpose

Returns the ID and description of all help desks.

Parameters

None

Returns

A dataset of help desk IDs and descriptions

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal iEntity As CPEntity, ByVal sUDFField As
String, ByVal sUDFValue As String) As String
```

Purpose

Returns the ID for the specified entity based on the specified UDF field and value.

Parameters

Parameter (*=required)	Description
iEntity	Entity for which the ID is being looked up
sUDFField	UDF text field name (for example, Text1)
sUDFValue	UDF text value

Returns

ID of the specified entity.

Remarks

The method will throw error -10 (The records are not found) when no records are returned based on the search criteria.

The method will throw error -11 (multiple records found) when duplicates exist for the lookup.

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetLookupCodes

```
Public Function GetLookupCodes(ByVal sTableName As String, ByVal
sResourceGWkgpId As String, ByVal sResourceWkgpId As String) As DataSet
```

Purpose

Returns the ID and description for all values for the specified lookup code.

Parameters

Parameter (*=required)	Description
*sTableName	The name of the table containing the lookup code values (for example, "contacttype")
*sResourceGWkgpId	The Global Workgroup ID of the resource for whom the lookup codes are being retrieved.
*sResourceWkgpId	The Workgroup ID of the resource for whom the lookup codes are being retrieved.

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetLookupFields

```
Public Function GetLookupFields(ByVal sTableName As String, ByVal sRetColumn1
As String, Optional ByVal sRetColumn2 As String = "", Optional ByVal
sRetColumn3 As String = "", Optional ByVal sFilter As String = "") As DataSet
```

Purpose

Generate a simple SQL statement to access the database.

Parameters

Parameter (*=required)	Description
*sTableName	Database table name.
*sRetColumn1	The first returned column name for the table.
sRetColumn2	The second returned column name for the table.
sRetColumn3	The third returned column name for the table.
sFilter	Filter criteria string, usually the string after the keyword "WHERE", for example, "Deleted = 0 AND FirstName = 'Richard'"

Returns

A dataset for the records.

Remarks

None

Example

```
Dim myApi as New ApiLookup()
myApi.CPConnection=myCon
Dim ret as DataSet= myApi.GetLookupFields
("Resources","Name","EmployeeType","HireDate"," Deleted = 0 AND Licensed =
1'")
```

Related information

"ApiLookup" on page 367

ApiLookup: GetNameById

```
Public Function GetNameById(ByVal iEntity As CPEntity, ByVal sId As String) As
Identity
```

Purpose

The function returns the name and alternate name for Changepoint entities.

Parameters

Parameter (*=required)	Description
*iEntity	The CPEntity for the lookup.
*sId	Entity ID for lookup.

Returns

The Identity object.

Remarks

The function should only be used to retrieve the name from the Entity table.

Example

```
Dim myApi as New ApiLookup()
myApi.CPConnection=myCon
Dim retId as String="{}"
Dim ret as Identity= myApi.GetNameById (CPEntity.Resource, retId)
```

Related information

"ApiLookup" on page 367

ApiLookup: GetPayrollCycle

```
Public Function GetPayrollCycle() As DataSet
```

Purpose

Retrieve payroll cycles.

Parameters

None

Returns

A dataset with the ID and description of all payroll cycles

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetProvincesByCountry

```
Public Function GetProvincesByCountry(ByVal sCountryCode As String) As DataSet
```

Purpose

Returns the province ID and name fields from the Province table for the specified country code.

Parameters

Parameter (*=required)	Description
*sCountryCode	Country to get the state/province codes for

Returns

A dataset fo province IDs and names for the specified country

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetRoles

```
Public Function GetRoles() As DataSet
```

Purpose

Retrieves all roleId, name and description fields

Parameters

None

Returns

A dataset of all roleId, name and description fields

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetTimeZones

```
Public Function GetTimeZones() As DataSet
```

Purpose

Returns all time zones.

Parameters

None

Returns

A dataset of all time zones

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetUDF

```
Public Function GetUDF(ByVal entity As CPEntity, Optional ByVal retOption As
CPUDFReturnType = CPUDFReturnType.OnlyStructureAndOptions, Optional ByVal
entityId As String = "", Optional ByVal additionalId As String = "", Optional
ByVal actionResourceId As String = "") As String
```

Purpose

Retrieve UDF (configurable fields) for each entity.

Parameters

Parameter (*=required)	Description
*entity	CPEntity, for example, CPEntity.Customer = 2.
retOption	Return options: • WithStructure = 0 – Returns the UDF values and structure • OnlyValues = 1 – Returns the UDF values • OnlyStructure = 2 – Returns the UDF structure • OnlyStructureAndOptions = 3 – Returns the UDF structure and options • WithStructureAndOptions = 4 – Returns the UDF structure, metadata tags, values and all the options for codes (except entity based and conditional codes)
entityId	The entity ID for the UDF, for example, CustomerId for Customer entity.

Parameter (*=required)	Description
additionalId	An additional ID for the UDF is required for the following entities: • Contract – BillingOfficeId • Request – RequestType It is not needed for other entities.
actionResourceId	The user ID of the user that called the method. Default is the login user ID.

Returns

The UDF string in XML format.

Remarks

The function is designed for retrieving UDF information for each entity. It is recommended to use this function to get UDFs.

Example

```
Dim myApi as New ApiLookup ()
myApi.CPConnection=myCon
Dim customerId as String="{}"
Dim ret as String = myApi.GetUDF (CPEntity.Customer,
CPUDFReturnType.OnlyValue, customerId)
```

Related information

"ApiLookup" on page 367

ApiLookup: GetUDFByXML

```
Public Function GetUDFByXML(ByVal sxmlUDF As String, Optional ByVal retOption
As CPUDFReturnType = CPUDFReturnType.OnlyStructure) As String
```

Purpose

Retrieves UDF by provided UDF XML.

Parameters

Parameter (*=required)	Description
*sxmlUDF	XML UDF string
retOption	Return options: • WithStructure = 0 – Returns the UDF values and structure • OnlyValues = 1 – Returns the UDF values • OnlyStructure = 2 – Returns the UDF structure • OnlyStructureAndOptions = 3 – Returns the UDF structure and options • WithStructureAndOptions = 4 – Returns the UDF structure, metadata tags, values and all the options for codes (except entity based and conditional codes)

Returns

The UDF string in XML format.

The following is a return example for the Project entity.


```
<root>
<udf>
<entity>Project</entity>
<entityid>8599cc4c-8642-48d9-baab-9b7b3c620a6f</entityid>
<additionalid actual="" />
<resourceid>b4059754-d32c-11d2-9ae7-006008a4d883</resourceid>
<fields>
   <fielditem index="0" fieldname="ProjectCode1" fieldformat="1">
      <cpcode name="Code1" caption="Code 1" multiselect="0" entitybased="0"
conditional="0" mandatory="0">
      <codeitem text="1" value="24b133bb-758a-42c2-82c6-3894a125b0ad"
selected="0" />
      <codeitem text="1" value="66da0e78-1c34-4bfd-b3be-d5c4fb8f430e"
selected="1" />
      <codeitem text="2" value="61d97a33-96a4-4692-a13d-d1ee9ac2aa70"
selected="0" />
      <codeitem text="3" value="db050f1b-f041-446a-a49c-ea38eb9f888b"
selected="0" />
      </cpcode>
   </fielditem>
   <fielditem index="1" fieldname="ProjectCode2" fieldformat="1">
```

```
      <cpcode name="Code2" caption="Code 2" multiselect="0" entitybased="0"
conditional="0" mandatory="0">
      <codeitem text="a" value="a441fa17-25d1-4c6e-9848-e67f825549d9"
selected="1" />
      <codeitem text="b" value="c5cda39f-dfdb-495b-bca9-d2bda262c151"
selected="0" />
      </cpcode>
   </fielditem>
   <fielditem index="29" fieldname="ProjectText6" fieldformat="3">
      <cptext name="Text6" caption="cal when changed" value="10011.0000"
mandatory="0" maximum="0" minimum="0" noteditable="1"
lockedafterentry="0" duplicatecheck="0" />
   </fielditem>
   <fielditem index="30" fieldname="ProjectText7" fieldformat="3">
      <cptext name="Text7" caption="Test Auto (Numeric)" value=""
mandatory="0" maximum="0" minimum="0" noteditable="0"
lockedafterentry="0" duplicatecheck="0" />
   </fielditem>
</fields>
</udf>
</root>
```

Remarks

The function is specified for getting the UDF structure by passing an existing UDF string.

Example

```
Dim sUDF as String = "<root>
   <udf>
      <entity>Project</entity>
      <entityid>{7BDC32B4-9DC0-41B0-BA02-B4104FF57ACB}</entityid>
      <additionalid actual=""></additionalid>
      <resourceid>{BE19F806-03FD-4DB3-B105-ECFD5E2F9195}</resourceid>
      <fields></fields>
   </udf>
</root>"

Dim udfStructure as String = myApi.GetUDFByXML
(sUDF,CPUDFReturnType.OnlyStructure)
```

Related information

"ApiLookup" on page 367

ApiLookup: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal entity As CPEntity, ByVal codeName As String, Optional ByVal filterFieldsXML As String = "", Optional ByVal searchString As String = "", Optional ByVal udfXML As String = "", Optional ByVal entityId As String = "", Optional ByVal additionalId As String = "", Optional ByVal actionResourceId As String = "") As String
```

Purpose

Retrieves UDF (configurable fields) by provided UDF XML.

Parameters

Parameter (*=required)	Description
*entity	CPEntity, for example, CPEntity.Customer = 2.
*codeName	Code name of UDF, for example, "Code3"
filterFieldsXML	Filter fields XML string.
searchString	The function returns the options that start with this string. If no string is entered, all options are returned.
udfXML	The UDF string in XML format.
entityId	EntityId for the UDF, for example, CustomerId for the Customer entity.
additionalId	An additional ID is required for the following entities: • Contract – BillingOfficeId • Request – RequestType The parameter is not needed for other entities.
actionResourceId	The user ID of the user who called the method. The default is the login user ID.

Returns

The UDF code options for the specified code in string.

The following is a return example for CodeName = Code1:

```
<root>
   <cpcode name="Code1">
      <codeitem text="0" value="24b133bb-758a-42c2-82c6-3894a125b0ad"
selected="0" />
      <codeitem text="1" value="66da0e78-1c34-4bfd-b3be-d5c4fb8f430e"
selected="0" />
      <codeitem text="2" value="61d97a33-96a4-4692-a13d-d1ee9ac2aa70"
selected="0" />
      <codeitem text="3" value="db050f1b-f041-446a-a49c-ea38eb9f888b"
selected="0" />
   </cpcode>
</root>
```

Remarks

The `filterFieldsXML` string should be retrieved by the `GetUDFFilterFields()` function and populated with current values.

Example

```
Dim myApi as New ApiLookup ()
myApi.CPConnection=myCon
Dim PrjId as String = "{}"
'retrieve filterfields temp
Dim sFilter as String = myApi.GetUDFFilterFields(CPEntity.Project, "Code3")
If sFilter <> "<root/>" Then
'this item is conditional item, populate filter fields with current values
…
Else
'this item is NOT conditional item, do not need to populate
End If
Dim sUDF as String = myApi.GetUDF(CPEntity.Project, CPUDFReturnType.OnlyValue,
PrjId)
Dim sCodeOptions as String = myApi.GetUDFCodeOptions (CPEntity.Project,
"Code3", sFilter, "", PrjId)
```

Related information

"ApiLookup" on page 367

ApiLookup: GetUDFFilterFields

```
Public Function GetUDFFilterFields(ByVal entity As CPEntity, ByVal codeName As
String, Optional ByVal additionalId As String = "", Optional ByVal
actionResourceId As String = "") As String
```

Purpose

Retrieves filter fields for the specified code and entity.

Parameters

Parameter (*=required)	Description
*entity	CPEntity, for example, CPEntity.Customer = 2.
*codeName	Code name of UDF, for example, "Code3"
additionalId	An additional ID for the UDFis required for the following entities: • Contract – BillingOfficeId • Request – RequestType It is not needed for other entities.
actionResourceId	The user ID of the user who called the method. Default is the login user ID.

Returns

The filter fields in XML.

The following is a return example for the Project entity:

```
<root>
   <entity value="Project"><id/><name/>
      <field><name>CPProjectType</name><id/><value/></field>
      <field><name>Customer</name><id /><value /></field>
      <field><name>Engagement</name><id /><value /></field>
      <field><name>ProjMgr</name><id /><value /></field>
      <field><name>ProposedPhase</name><id/><value /></field>
      <field><name>Status</name><id /><value /></field>
   </entity>
</root>
```

Remarks

This function is only used to get UDF conditional filter fields which are required for the function `GetUDFCodeOptions()`. `GetUDFCodeOptions()` retrieves the specified UDF code options.

The filterFieldsXML string will be equal to "</root>" if this is not a conditional field. For conditional fields, the function just returns the filter field schema which must be populated with current values before passing it to get the UDF options.

Example

```
Dim myApi as New ApiLookup ()
myApi.CPConnection=myCon
Dim PrjId as String = "{}"
'retrieve filterfields temp
Dim sFilter as String = myApi.GetUDFFilterFields(CPEntity.Project, "Code3")
If sFilter <> "<root/>" Then
'this item is conditional item, populate filter fields with existing values…
Else
'this item is NOT conditional item, do not need to populate
End If
Dim sUDF as String = myApi.GetUDF(CPEntity.Project, CPUDFReturnType.OnlyValue,
PrjId)
Dim sCodeOptions as String = myApi.GetUDFCodeOptions (CPEntity.Project,
"Code3", sFilter, "", PrjId)
```

Related information

"ApiLookup" on page 367

ApiLookup: GetWorkCodeCategories

```
Public Function GetWorkCodeCategories() As DataSet
```

Purpose

Retrieves work code categories

Parameters

None

Returns

A dataset of work code categories

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetWorkCodesByCategory

```
Public Function GetWorkCodesByCategory(Optional ByVal sWorkCodeCategoryId As
String = "") As DataSet
```

Purpose

Retrieves the list of work codes associated with the specified work code category.

Parameters

Parameter (*=required)	Description
sWorkCodeCategoryId	ID of the specified work code category

Returns

A dataset of the WorkCodeId and name

Remarks

If the sWorkCodeCategoryId string is empty or no parameter passed, returns all work codes.

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetWorkgroups

```
Public Function GetWorkgroups(Optional ByVal globalWorkgroupId As String = "",
Optional ByVal workgroupName As String = "") As DataSet
```

Purpose

Retrieves a list of workgroups.

Parameters

Parameter (*=required)	Description
globalWorkgroupId	ID of globalWorkgroup
workgroupName	Name of the Workgroup

Returns

A dataset with globalWorkgroupId, WorkgroupId, and name.

Remarks

If no parameters are passed, returns all workgroups.

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetWorkLocationGroups

```
Public Function GetWorkLocationGroups() As DataSet
```

Purpose

Returns the name and work location groupId fields from the worklocationgroups table.

Parameters

None

Returns

A dataset of name and work location groupID fields

Remarks

None

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: GetWorkLocations

```
Public Overloads Function GetWorkLocations(Optional ByVal sWorkLocationGroupId
As String = "", Optional ByVal sWorklocationGroupName As String = "") As
DataSet
```

Purpose

Retrieves work location information.

Parameters

Parameter (*=required)	Description
sWorklocationGroupName	Name of the work location group (to get the work locations for).
sWorkLocationGroupId	ID of WorkLocationGroup

Returns

Returns one of the following depending on the parameters:

- Name and work location ID fields from the work locations table where the name matches
- Dataset with the WorkLocationGroupId, WorkLocationId, Name and Description
- All work locations, if no parameters are passed

Example

Not available

Related information

"ApiLookup" on page 367

ApiLookup: SaveUDF

```
Public Overloads Function SaveUDF(ByVal entity As CPEntity, ByVal entityId As
String, ByVal sxmlUDF As String, Optional ByVal additionalId As String = "",
Optional ByVal needValidate As Boolean = True, Optional ByVal bypassMetadata
As CPMetadataCheck = CPMetadataCheck.BypassAll) As Int32

Public Overloads Function SaveUDF(ByVal sxmlUDF As String, Optional ByVal
needValidate As Boolean = True, Optional ByVal bypassMetadata As
CPMetadataCheck = CPMetadataCheck.BypassAll) As Int32
```

Purpose

Saves UDF (configurable field) information to the Changepoint database.

Parameters

Parameter (*=required)	Description
*entity	CPEntity, for example, CPEntity.Customer = 2.
*entityId	The entity ID for the UDF, for example, CustomerId for Customer entity.
*sxmlUDF	UDF string in XML format.
additionalId	An additional ID for the UDFis required for the following entities: • Contract – BillingOfficeId • Request – RequestType It is not needed for other entities.
needValidate	The flag for validation required.
bypassMetadata	The purpose of this flag is to disable metadata checking when adding or updating data. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

0 = Success

Nonzero = Error

Remarks

The functions are recommended for saving UDF information for each entity.

There are two overloads functions. In the first function, the values for the entity, entity ID and additional ID will be read-only from each parameter and these values in the sxmlUDF parameter will be ignored. In the second function, these values will be picked up from the sxmlUDF parameter.

> **Note:** The functions are not designed for partial UDF updates. If passing partial UDF field values, the fields that have no values passed will be blank.

Example

```
Dim myApi as New ApiLookup ()
myApi.CPConnection=myCon
Dim PrjId1 as String = "{}"
Dim PrjId2 as String = "{}"
'retrieve UDF from PrjId1
Dim sUDF as String = myApi.GetUDF(CPEntity.Project, CPUDFReturnType.OnlyValue,
PrjId1)
' copy this UDF to PrjId2
Dim ret as Int32 = myApi.SaveUDF(CPEntity.Project, PrjId2, sUDF)
```

Related information

"ApiLookup" on page 367

ApiLookup: ValidateUDF

```
Public Overloads Function ValidateUDF(ByVal entity As Byte, ByVal sxmlUDF As
String, Optional ByVal entityId As String = "", Optional ByVal additionalId As
String = "", Optional ByVal metadataCheck As CPMetadataCheck =
CPMetadataCheck.CheckAll) As Int32

Public Overloads Function ValidateUDF(ByVal sxmlUDF As String, Optional ByVal
metadataCheck As CPMetadataCheck = CPMetadataCheck.CheckAll) As Int32
```

Purpose

Validates UDF information for UDF string.

Parameters

Parameter (*=required)	Description
*entity	CPEntity, for example, CPEntity.Customer = 2.
entityId	EntityId for the UDF, for example, CustomerId for Customer entity.
*sxmlUDF	UDF string in XML format.
additionalId	An additional ID for the UDFis required for the following entities: • Contract – BillingOfficeId • Request – RequestType
metadataCheck	The options for Metadata checking. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

0 = Success

Nonzero = Error

Remarks

There are two overloads functions. In the first function, the values for the entity, entity ID and additional ID will be read-only from each parameter and these values in the sxmlUDF parameter will be ignored. In the second function, these values will be picked up from the sxmlUDF parameter.

Example

```
Dim myApi as New ApiLookup ()
myApi.CPConnection=myCon
Dim PrjId1 as String = "{}"
Dim sUDF as String = "<root><udf><entity>Project</entity></root>"
Dim ret as Int32 = myApi.ValidateUDF(CPEntity.Project, sUDF, PrjId1)
```

Related information

"ApiLookup" on page 367

ApiOpportunity

The ApiOpportunity object allows users to create, retrieve, update and delete opportunities within the Changepoint database, as well as general opportunity information such as: OpportunityFixedFees, OpportunityDetails, OpportunityServices and OpportunityExpenses information.

Namespace

Changepoint.ChangepointAPI2.ApiOpportunity

Methods

Properties

Property (*=required)	Type	Description
ActualClose	String	Actual close date. The string value must be a datetime value.
AlternateName	String	Opportunity alternate name
AvailableToAll	Boolean	True if everyone is allowed to view the opportunity. In the API, this is always set to true.

Property (*=required)	Type	Description
*AwardedToCompetitor	Identity	Identifier of the competitor that the opportunity was awarded to. When adding an opportunity from the API, if AwardedToCompetitor is assigned, it has to be added to the Competitors list before calling the Add method, otherwise it will throw error 980 (message=Awarded competitor id doesn't not exist in competitor list).
BillingOffice	Identity	Identifier of the billing office for the opportunity
BillingType	Identity	Identifier of the billing type for the opportunity. The valid values for the Id property of the Identity data type are D, F, H, MD or MH. The corresponding names are as follows: • D = Daily • F = Fixed Fee • H = Hourly • MD = Mixed - Fixed Fee/Daily • MH = Mixed - Fixed Fee/Hourly
BypassMetadataCheck	CPMetadataCheck	Determines the level of MetaData checking when adding or updating data. The default is to check all fields. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2 For Opportunity, the value set for this flag will be applied to all inner objects on Update and Add.

Property (*=required)	Type	Description
BypassWorkflow	Boolean	True if the steps in the opportunity workflow can be bypassed
Campaign	Identity	Identifier of the campaign for the opportunity
Competitors	Generic.List(Of Identity)	List of identifiers of the competitors selected for this opportunity
CostCenter	Identity	Identifier of the cost center for the opportunity
CreatedBy	Signature	Contains details about who created the opportunity and when the opportunity was created
*Customer	Identity	Identifier of the customer or client to whom the opportunity belongs.
Deleted	Boolean	True if the opportunity has been deleted
Description	String	Description of the opportunity
Entity	CPEntity	Read-only. The Changepoint entity the object represents.
FPBillingOffice	Identity	Identifier of the fiscal period billing office for the opportunity
IncludeInForecast	Boolean	For PSA, this flag is true if the opportunity is included in the forecast rollup. For ITD, this flag is true if the candidate is included in the screening process.
InitialClose	String	Initial close date. The string value must be a datetime value.

Property (*=required)	Type	Description
LockBudget	Boolean	True if the opportunity budget is locked
mVersion	String	Read-only. Current version number for internal version control.
*Name	String	Name of the opportunity
OppExpenses	Generic.List(Of ApiOppExpense)	List of opportunity expenses
OppFixedFees	Generic.List(Of ApiOppFixedFee)	List of opportunity fixed fees
*OpportunityCurrency	Identity	Opportunity currency
OpportunityId	String	Opportunity ID
OpportunityPriority	Identity	Opportunity priority
OpportunityType	Identity	Opportunity type
OppProducts	Generic.List(Of ApiOppProduct)	List of opportunity products
OppServices	Generic.List(Of ApiOppService)	List of opportunity services
OutcomeDescription	String	Description of the outcome
OutcomeLost	Boolean	True if Outcome = Lost
OutcomeNoDecision	Boolean	True if Outcome = NoDecision
OutcomeOther	Boolean	True if Outcome = Other
OutcomeWon	Boolean	True if Outcome = Won
PrimaryContact	Identity	ID of the primary contact for the opportunity

Property (*=required)	Type	Description
Probability	String	Probability. The string value must be in a form that can be converted to a decimal value, for example, “80.0”
Product	Identity	Product ID
Request	Identity	Opportunity request ID
RevenueForecast	Decimal	Revenue forecast for the opportunity
RevisedClose	String	Revised close date. The string value must be a datetime value.
*SalesRep	Identity	Sales representitive for this opportunity
SalesTeam	Generic.List(Of Identity)	List of identifiers of the sales teams selected for this opportunity
*StaffingWorkgroup	Identity	Identifier of the opportunity staffing workgroup
*Status	Identity	Identifier of the opportunity status
sxmlUDF	String	UDF XML string. Includes all UDF (configurable field) code and text information.
UpdatedBy	Signature	The person who last updated the data and the date updated.

Related information

"ApiOpportunity XML" on page 405

"ApiOppFixedFee" on page 440

"ApiOppProduct" on page 443

"ApiOppService" on page 447

"ApiOppExpense" on page 435

ApiOpportunity XML

```
<root>
   <opportunity>
      <bypassmetadatacheck />
      <bypassworkflow />
      <createdbyid />
      <updatedbyid />
      <opportunityid />
      <name />
      <alternatename />
      <customer>
         <id />
         <name />
         <alternatename />
         <userdefinedid />
      </customer>
      <description />
      <salesrep>
         <id />
         <name />
         <alternatename />
         <firstname />
         <lastname />
         <userdefinedid />
      </salesrep>
      <request>
         <id />
         <name />
         <alternatename />
      </request>
      <primarycontact>
         <id />
         <name />
         <alternatename />
         <firstname />
         <lastname />
         <email1 />
      </primarycontact>
      <opportunitytype>
         <id />
         <name />
         <alternatename />
      </opportunitytype>
      <billingoffice>
         <id />
         <name />
         <alternatename />
```

```
</billingoffice>
<billingtype>
   <id />
   <name />
   <alternatename />
</billingtype>
<fpbillingoffice>
   <id />
   <name />
   <alternatename />
</fpbillingoffice>
<staffingworkgroup>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</staffingworkgroup>
<status>
   <id />
   <name />
   <alternatename />
</status>
<probability />
<includeInforecast>false</includeInforecast>
<revenueforecast />
<initialclose />
<revisedclose />
<actualclose />
<availabletoall>true</availabletoall>
<outcomewon>false</outcomewon>
<outcomelost>false</outcomelost>
<outcomenodecision>false</outcomenodecision>
<outcomeother>false</outcomeother>
<awardedtocompetitor>
   <id />
   <name />
   <alternatename />
</awardedtocompetitor>
<outcomedescription />
<opportunitycurrency>
   <id />
   <name />
   <alternatename />
</opportunitycurrency>
<opportunitypriority>
   <id />
   <name />
   <alternatename />
```

```
</opportunitypriority>
<product>
   <id />
   <name />
   <alternatename />
</product>
<campaign>
   <id />
   <name />
   <alternatename />
</campaign>
<costcenter>
   <id />
   <name />
   <alternatename />
</costcenter>
<lockbudget>false</lockbudget>
<deleted>false</deleted>
<salesteams>
   <salesteam>
      <id />
      <name/>
      <alternatename />
      <firstname/>
      <lastname/>
      <userdefinedid />
   </salesteam>
</salesteams>
<competitors>
   <competitor>
      <id />
      <name/>
      <alternatename />
   </competitor>
 </competitors>
<udf />
<opportunityfixedfees>
   ...
</opportunityfixedfees >
<opportunityproducts>
   ...
</opportunityproducts>
<opportunityservices>
   ...
</opportunityservices>
<opportunityexpenses>
   ...
</opportunityexpenses>
```

```
   </opportunity>
</root>
```

Comments

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

The XML for the ApiOpportunity object includes the XML for the ApiOppFixedFee, ApiOppProduct, ApiOppService and ApiOppExpense objects and may contain XML for UDFs (configurable fields). For more information on the XML details for included objects, see the documentation for the included objects.

Example

Not available

Related information

"ApiOpportunity" on page 399

"ApiOppExpense XML" on page 437

"ApiOppFixedFee XML" on page 441

"ApiOppProduct XML" on page 445

"ApiOppService XML" on page 450

"UDF XML" on page 742

ApiOpportunity: Add

```
Public Overrides Function Add(Optional ByRef sId As String = "") As Integer
```

Purpose

Adds a new opportunity to Changepoint

Parameters

Parameter (*=required)	Description
sId	Returns the new OpportunityId after the opportunity has been added.

Returns

0 = Success

Nonzero = Error

Remarks

Adds an opportunity to Changepoint. See the properties of the ApiOpportunity object for mandatory fields. The Opportunity object also contains the list of OppExpense, OppFixedFee, OppProduct and OppService objects.

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, Optional ByRef sId As String
= "") As Int32
```

Purpose

Creates an opportunity using an XML string of the Opportunity object in Changepoint.

Parameters

Parameter (*=required)	Description
*sXML	XML string for the new opportunity, which is passed based on the Opportunity XML schema
sId	ByRef parameter that returns the new OpportunityId after the opportunity has been added.

Returns

0 = Success

Nonzero = Error

Remarks

The ApiOpportunity XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The ByPassMetadataCheck switch will stop any metadata validation in the Opportunity and also in the Opportunity UDFs (configurable fields).

For details of the use of UDF default values, see "UDF default values logic for CreateByXML" on page 746.

Example

Not available

Related information

"ApiOpportunity" on page 399

"ApiOpportunity: GetXMLStructure" on page 429

"ApiOpportunity: GetByXML" on page 414

"ApiOpportunity XML" on page 405

ApiOpportunity: Delete

```
Public Overrides Function Delete(Optional ByVal sId as String = "") As Integer
```

Purpose

Deletes an existing opportunity from Changepoint.

Parameters

Parameter (*=required)	Description
sId	ID of the opportunity to be deleted

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: Exists

```
Public Overrides Function Exists(Optional ByVal sId as String = "") As Boolean
```

Purpose

Verifies whether the oportunity exists in the database and has not been deleted.

Parameters

Parameter (*=required)	Description
sId	Opportunity ID to verify

Returns

True if the opportunity exists

Remarks

If the sId parameter is not specified, then the OpportunityId property of the ApiOpportunity object must be set or you will get an error.

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetBillingOffices

```
Public Function GetBillingOffices(Optional ByVal sSalesRepId As String = "",
Optional ByVal sCurrencyId As String = "", Optional ByVal sSearchString as
String ="") As DataSet
```

Purpose

Returns a list of billing offices that have their billing rate set up for the currency and that the sales rep has access to.

Parameters

Parameter (*=required)	Description
sSalesRepId	Sales Rep ID
sCurrencyId	Currency ID
sSearchString	Search string

Returns

A dataset with two columns (BillingOfficeId, Description)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetBillingTypes

```
Public Function GetBillingTypes(ByVal sBillingOfficeId As String = "") As
DataSet
```

Purpose

Returns a list of billing types available for the specified billing office. When the DailyRateSetup table is populated for the billing office, D and MD billing types are returned, in addition to F, H and MH billing types.

Parameters

Parameter (*=required)	Description
*sBillingOfficeId	ID of the billing office

Returns

A dataset with two columns (Code, Description)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetById

```
Public Overrides Function GetById(Optional ByVal sId as String = "") As
Integer
```

Purpose

Fills the object with data related to the specified OpportunityId passed in the parameter.

Parameters

Parameter (*=required)	Description
sId	

Returns

0 = Success

Nonzero = Error

Remarks

This method fills the object with current data from the database. In the oportunity object, this method also fills the inner expense, fixed fee, product, and service objects.

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetByXML

```
Public Function GetByXML(Optional ByVal sXML as String = "", Optional ByVal
sOpportunityId As String = "") As String
```

Purpose

Takes the XML string passed in the sXML parameter and returns the string filled with data for the opportunity specified in the sOpportunityId parameter.

Parameters

Parameter (*=required)	Description
sXML	XML string of the Opportunity object, with the fields specified
sOpportunityId	ID of the opportunity. • If no OpportunityId is passed in, the XML string (sXML) is examined • If no OpportunityId in sXML then the object's OpportunityId is selected. • If there still is no valid OpportunityId, the method returns an error.

Returns

An XML string mirroring sXML with data inserted or the entire XML of the Opportunity object including data.

Remarks

If sXML = "" then the XML string provided by GetXMLStructure is used.

Example

Not available

Related information

"ApiOpportunity" on page 399

"ApiOpportunity XML" on page 405

ApiOpportunity: GetCompetitors

```
Public Function GetCompetitors() As DataSet
```

Purpose

Returns a list of competitors.

Parameters

None

Returns

A dataset with two columns (CompetitorId, Name)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetContacts

```
Public Function GetContacts (ByVal sCustomerId As String, Optional ByVal
sResourceId As String = "") As DataSet
```

Purpose

Returns a list of contacts.

Parameters

Parameter (*=required)	Description
*sCustomerId	Customer ID
sResourceId	Resource ID. If the resource ID is not passed, the ID of the logged-in user is used.

Returns

A dataset with two columns (ContactId, Name)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetCostCenters

```
Public Function GetCostCenters(Optional ByVal sSearchString As String = "") As
DataSet
```

Purpose

Returns a list of cost centers.

Parameters

Parameter	Description
sSearchString	Search string

Returns

A dataset with two columns (CostCenter, Name)

Remarks

None

Example

Not available

ApiOpportunity: GetCustomers

```
Public Function GetCustomers(Optional ByVal sSearchString as String = "") As
DataSet
```

Purpose

Returns a list of customers.

Parameters

Parameter (*=required)	Description
sSearchString	Search string

Returns

A dataset with two columns (CustomerId, Name)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetFPBillingOffices

```
Public Function GetFPBillingOffices(Optional ByVal sSearchString As String =
"") As DataSet
```

Purpose

Returns a lists of billing offices corresponding to fiscal periods

Parameters

Parameter (*=required)	Description
sSearchString	Search string

Returns

A dataset with two columns (BillingOfficeId, Description)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetIdByUDFText

```
Public Function GetIdByUDFText (ByVal sUDFField As String, ByVal sUDFValue As
String) As String
```

Purpose

Returns the OpportunityId with the UDF Text field populated with the specific value.

Parameters

Parameter (*=required)	Description
*sUDFField	UDF text field name (for example, Text1)
*sUDFValue	UDF text value

Returns

OpportunityId, or an empty string if nothing is found.

Remarks

None

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows as Short = -1) As
DataSet
```

Purpose

Returns a list of some or all Opportunities in Changepoint that are not deleted in the system.

Parameters

Parameter (*=required)	Description
iRetRows	Number of records to be returned. Default = -1 (return all).

Returns

A dataset (OpportunityId, Name) or an empty dataset if nothing is found.

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetOpportunityCurrencies

```
Public Function GetOpportunityCurrencies() As DataSet
```

Purpose

Returns a list of opportunity currencies

Parameters

None

Returns

A dataset with two columns (Code, Description)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetOpportunityStatus

```
Public Function GetOpportunityStatus() As DataSet
```

Purpose

Returns a list of opportunity statuses

Parameters

None

Returns

A dataset with two columns (Code, Description)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetOpportunityTypes

```
Public Function GetOpportunityTypes(ByVal sGlobalWorkgroupId As String, ByVal
sWorkgroupId As String) As DataSet
```

Purpose

Returns a list of Opportunity Types that are available for the specific workgroup and global workgroup.

Parameters

Parameter (*=required)	Description
*sGlobalWorkgroupId	Global Workgroup ID of the resource for whom the codes are being retrieved.
*WorkgroupId	Workgroup ID of the resource for whom the codes are being retrieved.

Returns

A dataset with two columns (Code, Description)

Remarks

This function calls the GetLookUpCodes function in the ApiLookUp object with parameter sTableName="OpportunityType"

Example

Not available

Related information

"ApiOpportunity" on page 399

"ApiLookup: GetLookupCodes" on page 380

ApiOpportunity: GetOppPriorities

```
Public Function GetOppPriorities() As DataSet
```

Purpose

Returns a list of opportunity priorities.

Parameters

None

Returns

A dataset with two columns (Code, Description)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetOppProducts

```
Public Function GetOppProducts (Optional ByVal sSearchString as String = "")
As DataSet
```

Purpose

Returns a list of opportunity products.

Parameters

Parameter (*=required)	Description
sSearchString	Search string

Returns

A dataset with two columns (ProductId, ProductName)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetRequests

```
Public Function GetRequests(Optional ByVal sSearchString as String = "") As
DataSet
```

Purpose

Returns a list of requests

Parameters

Parameter (*=required)	Description
sSearchString	Search string

Returns

A dataset with two columns (RequestId, RequestNum)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetSalesReps

```
Public Function GetSalesReps (Optional ByVal sCustomerId As String = "",
Optional ByVal sBillingOfficeId As String = "", Optional sSearch As String) As
DataSet
```

Purpose

Returns a list of the sales representatives who have access to the customer and billing office using the search string.

Parameters

Parameter (*=required)	Description
sCustomerId	Customer ID
sBillingOfficeId	Billing Office ID
sSearchString	Search string

Returns

A dataset with two columns (ResouceId, Name)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetSalesTeams

```
Public Function Public Function GetSalesTeams(Optional ByVal sResourceId As
String = "") As DataSet
```

Purpose

Returns a list of sales teams or the sales team that the resource can select while creating opportunities.

Parameters

Parameter (*=required)	Description
sResourceId	Resource ID. If the resource ID is not passed, the ID of the logged-in user is used.

Returns

A dataset with two columns (ResourceId, Name)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetSources

```
Public Function GetSources() As DataSet
```

Purpose

Returns a list of opportunity sources that are not deleted in the database.

Parameters

None

Returns

A dataset with two columns (CampaignId, Name)

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetUDF

```
Public Function GetUDF(Optional ByVal UDFOption As CPUDFReturnType =
CPUDFReturnType.OnlyValues, Optional ByVal sOpportunity As String = "",
Optional ByVal actionResourceId As String = "") As String
```

Purpose

Returns, as an XML string, UDF (configurable field) structure, options and/or values that the resource has access to.

Parameters

Parameter (*=required)	Description
UDFOption	Determines how the UDF data is to be returned. • OnlyStructure – returns only UDF XML structure, no field data • OnlyStructureAndOptions - returns UDF structure and option data without field data • OnlyValues - returns only UDF fields with data, do not include any field structure or options • WithStructure – returns UDF field data with full field structure • WithStructureAndOptions – returns UDF field data with full structure and all field options
sOpportunity	ID of the opportunity whose UDF data is required
actionResourceId	The user ID of the user who called the method. The default is the login user ID.

Returns

An XML string of UDFs

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal sCodeName as String, Optional ByVal
sSearchString as String = "") As String
```

Purpose

Returns UDF (configurable field) data for a opportunity as an XML string

Parameters

Parameter (*=required)	Description
*sCodeName	The name of the UDF code. Must be of the form "Code#", for example, "Code10".
sSearchString	Search string. The search string will match all the options in sCodeName and return all options containing sSearchString in the option text.

Returns

An XML string of the option list that exists for a specified code.

Remarks

None

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Returns the XML structure of the ApiOpportunity object and also all the subobjects.

Parameters

None

Returns

Returns the XML structure of the ApiOpportunity object and also all the subobjects.

Remarks

Some fields in the structure will have defaulted data; otherwise, fields are empty.

Example

Not available

Related information

"ApiOpportunity" on page 399

"ApiOpportunity XML" on page 405

ApiOpportunity: SaveOppExpense

```
Public Function SaveOppExpense(ByVal oOppExpense As ApiOppExpense)
```

Purpose

Adds an opportunity expense object to the collection.

Parameters

Parameter (*=required)	Description
*oOppExpense	Opportunity expense object to add

Returns

Not applicable

Remarks

You can create an opportunity expense collection directly and assign it to the opportunity property, in which case, this method is unnecessary.

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: SaveOppFixedFee

```
Public Function SaveOppFixedFee(ByVal oOppFixedFee As ApiOppFixedFee)
```

Purpose

Adds an opportunity fixedfee object to the collection.

Parameters

Parameter (*=required)	Description
*oOppFixedFee	Opportunity fixed fee object to add

Returns

Not applicable

Remarks

You can create an opportunity fixedfee collection directly and assign it to the opportunity property, in which case, this method is unnecessary.

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: SaveOppProduct

```
Public Function SaveOppProduct(ByVal oOppProduct As ApiOppProduct)
```

Purpose

Adds an opportunity product object to the collection.

Parameters

Parameter (*=required)	Description
*oOppProduct	Opportunity product object to add

Returns

Not applicable

Remarks

You can create an opportunity product collection directly and assign it to the opportunity property, in which case, this method is unnecessary.

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: SaveOppService

```
Public Function SaveOppService(ByVal oOppService As ApiOppService)
```

Purpose

Adds an opportunity service object to the collection.

Parameters

Parameter (*=required)	Description
*oOppService	Opportunity service object to add

Returns

Not applicable

Remarks

In order to associated a fixed fee with an opportunity service, the opportunity must be created first and the fixed fee must already be associated at the opportunity level.

You can create an opportunity service collection directly and assign it to the opportunity property, in which case, the SaveOppService method is unnecessary.

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: SaveUDF

```
Public Function SaveUDF(Optional ByVal sUDF As String = "") As Int32
```

Purpose

Saves (Insert/Update) UDF data for an opportunity.

Parameters

Parameter (*=required)	Description
sUDF	UDF string in XML format.

Returns

0 = Success

Nonzero = Error

Remarks

The Opportunity.Id is a mandatory field, so this property should be populated before calling the SaveUDF method. If sUDF is not provided, take the value of property sxmlUDF.

It is recommended to call the GetUDF method to obtain the correct UDF XML format.

Example

```
Dim myOpp as New ApiOpportunity()
Dim iRet As Int32
Dim strXMLUDF as string ="<root></root>"
With myOpp
.CPConnection = myCon
.OpportunityId = "{CA8AC6E5-5F9A-41CA-BD57-9A35D28A7E76}"
iRet = .SaveUDF(strXMLUDF)
End With
```

Related information

"ApiOpportunity" on page 399

"ApiOpportunity: GetUDF" on page 427

"UDF XML" on page 742

ApiOpportunity: Update

```
Public Overrides Function Update() As Integer
```

Purpose

Updates opportunity data in the database with data held in the object

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

The update will write all data held in the Opportunity object including the internal objects for OppExpense, OppFixedFee, OppProduct and OppService objects.

Example

Not available

Related information

"ApiOpportunity" on page 399

ApiOpportunity: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, Optional ByVal
sOpportunityId As String = "") As Int32
```

Purpose

Updates an opportunity using an XML string containing new data.

Parameters

Parameter (*=required)	Description
*sXML	XML string of the Opportunity object, with new data contained in the fields
sOpportunityId	Opportunity ID of the opportunity being updated.

Returns

0 = Success

Nonzero = Error

Remarks

Performs the same function as Update except any Opportunity can be updated through this function. The XML sent in the parameter must be of the form in ApiOpportunity XML. The ApiOpportunity XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The method uses the following sequence to find the opportunity ID:

1. If the sOpportunityId parameter is passed in, the method uses this value for the opportunity ID.
2. If this fails, the method attempts to extract the opportunity ID from <opportunityid> in the XML.
3. If this fails, the opportunity ID is taken from the object properties.
4. If this fails, an attempt is made to look up the opportunity ID using <name> in the XML.

To update the Opportunity Billing Office, include in the XML either:

- BillingOffice.Id

or

- BillingOffice.name, SalesRep, and Currency

Example

Not available

Related information

"ApiOpportunity" on page 399

"ApiOpportunity: GetByXML" on page 414

"ApiOpportunity: GetXMLStructure" on page 429

"ApiOpportunity XML" on page 405

ApiOppExpense

The ApiOppExpense object contains expense data for the opportunity.

Namespace

Changepoint.ChangepointAPI2.ApiOppExpense

Methods

Properties

Property (*=required)	Type	Description
*Action	CPObjectAction	Indicates the action: 1=Create 2=Update 3=Delete 9=Unchange
Billable	Boolean	If true, the opportunity expense is billable
Capital	Boolean	If true, the opportunity expense is a capital expense
*Category	Identity	Identifier of the expense category
Comments	String	Comments about the opportunity expense
DistributeEvenly	Boolean	If true, the opportunity expense is distributed evenly
*ExpenseDate	Date	Date of the opportunity expense
*ExpenseMultiplier	Decimal	Multiplier for the opportunity expense
ExpenseType	Identity	Identifier of the expense type
OpportunityExpenseId	String	ID of the opportunity expense

Property (*=required)	Type	Description
*Quantity	Decimal	Quantity
*UnitCost	Decimal	Unit cost

Related information

"ApiOpportunity" on page 399

"ApiOppExpense XML" on page 437

ApiOppExpense XML

```
<root>
   <opportunity>
      ...
      <opportunityexpenses>
         <opportunityexpense>
            <opportunityexpenseid />
            <category>
               <id />
               <name />
               <alternatename />
            </category>
            <expensetype>
               <id />
               <name />
               <alternatename />
            </expensetype>
            <expensedate />
            <quantity />
            <unitcost />
            <billable>false</billable>
            <expensemultiplier />
            <capital>false</capital>
            <comments />
            <action />
         </opportunityexpense>
      </opportunityexpenses>
   </opportunity>
</root>
```

Comments

The XML for the ApiOppExpenses object is included in the XML for the ApiOpportunity object. The following container elements are mandatory:

```
<root>
   <opportunity>
      <opportunityexpenses>
         <opportunityexpense>
         ...
         </opportunityexpense>
      </opportunityexpenses>
   </opportunity>
</root>
```

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

Example

Not available

Related information

"ApiOpportunity XML" on page 405

"ApiOppExpense" on page 435

ApiOppExpense: GetExpenseCategories

```
Public Function GetExpenseCategories() As DataSet
```

Purpose

Returns a list of opportunity expense categories.

Parameters

None

Returns

A dataset with two columns (Id, Name)

Remarks

None

Example

Not available

Related information

"ApiOppExpense" on page 435

ApiOppExpense: GetExpenseTypesByCategory

```
Public Function GetExpenseTypesByCategory(ByVal sCategoryId As String) As
DataSet
```

Purpose

Returns a list of opportunity expense types for the specifed expense category.

Parameters

Parameter (*=required)	Description
*sCategoryId	Expense category ID

Returns

A dataset with two columns (ExpenseTypeId, Description)

Remarks

None

Example

Not available

Related information

"ApiOppExpense" on page 435

ApiOppExpense: GetList

```
Public Function GetList(ByVal sOpportunityId As String, Optional ByVal
iRetRows As Short = -1) As System.Data.DataSet
```

Purpose

Fills the object with data related to the specified OpportunityId passed in the parameter.

Parameters

Parameter (*=required)	Description
*sOpportunityId	Opportunity ID
iRetRows	Number of rows to return. Default is -1 (return all).

Returns

A dataset with data or an empty dataset if nothing is found.

Remarks

None

Example

Not available

Related information

"ApiOppExpense" on page 435

ApiOppFixedFee

The ApiOppFixedFee object contains fixed fee information for the opportunity.

Namespace

Changepoint.ChangepointAPI2.ApiOppFixedFee

Methods

Properties

Property (*=required)	Type	Description
*Action	CPObjectAction	Indicates the action for opportunity fixed fee: 1=Create 2=Update 3=Delete 9=Unchange
Capital	Boolean	Capital flag
Comments	String	Comments about the opportunity fixed fee
CreatedBy	Signature	Contains details about who created the opportunity and when the opportunity was created.
*Description	String	Description of the opportunity fixed fee
*FixedFeeAmount	Decimal	Amount of the opportunity fixed fee
*FixedFeeDate	DateTime	Date of the opportunity fixed fee
OpportunityFixedFeeID	String	ID of the opportunity fixed fee
UpdatedBy	Signature	Conrtains details about who last updated the opportunity and when the opportunity was last updated.

Related information

"ApiOpportunity" on page 399

"ApiOppFixedFee XML" on page 441

ApiOppFixedFee XML

```
<root>
   <opportunity>
      ...
      <opportunityfixedfees>
         <opportunityfixedfee>
```

```
            <opportunityfixedfeeid />
            <fixedfeedate />
            <description />
            <comments />
            <capital>false</capital>
            <fixedfeeamount />
            <action />
         </<opportunityfixedfee>>
      </opportunityfixedfees>
   </opportunity>
</root>
```

Comments

The XML for the ApiOppFixedFee object is included in the XML for the ApiOpportunity object. The following container elements are mandatory:

```
<root>
   <opportunity>
      <opportunityfixedfees>
         <opportunityfixedfee>
         ...
         </<opportunityfixedfee>
      </opportunityfixedfees>
   </opportunity>
</root>
```

Example

Not available

Related information

"ApiOpportunity XML" on page 405

"ApiOppFixedFee" on page 440

ApiOppFixedFee: GetList

```
GetList(ByVal sOpportunityId As String, Optional ByVal iRetRows As Short = -1)
As DataSet
```

Purpose

Returns a list of fixedfees for the specified opportunity.

Parameters

Parameter (*=required)	Description
*sOpportunityId	Opportunity ID
iRetRows	Number of records to be returned. Default = -1 (return all).

Returns

A dataset with data or an empty dataset if nothing is found

Remarks

None

Example

Not available

Related information

"ApiOppFixedFee" on page 440

ApiOppProduct

The ApiOppProduct object contains product information for the opportunity.

Namespace

Changepoint.ChangepointAPI2.ApiOppProduct

Methods

Properties

Property (*=required)	Type	Description
*Action	CPObjectAction	Indicates the action for the opportunity product: 1=Create 2=Update 3=Delete 9=Unchange
AdjustedPrice	Decimal	Negotiated price of the opportunity product.
CalculationBase	ProductCalculation BaseEnum	Can have one of the following values: • Discount – indicates that the AdjustedPrice field is calculated based on the value provided by the Discount property. • AdjustedPrice– indicates the Discount field is calculated based on the value provided by the AdjustedPrice property. The default is AdjustedPrice.
Capital	Boolean	Capital flag
Comments	String	Comments about the opportunity product.
Cost	Decimal	Cost of the opportunity product.
*DetailDate	DateTime	Date of the opportunity product.
Discount	Decimal	Discount applied to the negotiated price.
OpportunityDetailId	String	ID of opportunity product.
Price	Decimal	Read-only. Standard price of the opportunity product.
*Product	Identity	Identifier of the opportunity product.
*Qty	Double	Quantity

Related information

"ApiOpportunity" on page 399

"ApiOppProduct XML" on page 445

ApiOppProduct XML

```
<root>
   <opportunity>
      ...
      <opportunityproducts>
         <opportunityproduct>
            <opportunitydetailid />
            <product>
               <id />
               <name />
               <alternatename />
            </product>
            <detaildate />
            <qty />
            <price />
            <adjustedprice />
            <cost />
            <discount />
            <capital>false</capital>
            <comments />
            <action />
            <calculationbase>adjustedprice</calculationbase>
         </opportunityproduct>
      </opportunityproducts>
   </opportunity>
</root>
```

Comments

The XML for the ApiOppProduct object is included in the XML for the ApiOpportunity object. The following container elements are mandatory:

```
<root>
   <opportunity>
      <opportunityproducts>
         <opportunityproduct>
            ...
         </opportunityproduct>
      </opportunityproducts>
   </opportunity>
</root>
```

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

Example

Not available

Related information

"ApiOpportunity XML" on page 405

"ApiOppProduct" on page 443

ApiOppProduct: GetList

```
Public Function GetList(ByVal sOpportunityId As String, Optional ByVal
iRetRows As Short = -1) As System.Data.DataSet
```

Purpose

Returns a list of products for the specified opportunity.

Parameters

Parameter (*=required)	Description
*sOpportunityId	Opportunity ID
iRetRows	Number of records to be returned. Default = -1 (return all).

Returns

A dataset with data or an empty dataset if nothing is found

Remarks

None

Example

Not available

Related information

"ApiOppProduct" on page 443

ApiOppProduct: GetProducts

```
Public Function GetProducts (Optional ByVal sSearchString as String = "") As
DataSet
```

Purpose

Returns a list of opportunity products

Parameters

Parameter (*=required)	Description
sSearchString	Search string

Returns

A dataset with a list of opportunity products

Remarks

None

Example

Not available

Related information

"ApiOppProduct" on page 443

ApiOppService

The ApiOppService object contains common service billing information associated with the opportunity.

Namespace

Changepoint.ChangepointAPI2.ApiOppService

Methods

Properties

Property (*=required)		Description
*Action	CPObjectAction	Indicates the action for the opportunity service: 1=Create 2=Update 3=Delete 9=Unchange
BillingOffice	Identity	Identifier of the billing office for the service
*BillingRole	Identity	Identifier of the billing role for the service
BOBillingCurrency	Identity	Identifier of the currency associated with the billing office for the service
BOBillingRate	Decimal	Read-only. Billing rate associated with the billing office for the service
BOResourceCostRate	Decimal	Read-only. Resource cost rate associated with the billing office for the service
BORoleCostRate	Decimal	Read-only. Role cost rate associated with the billing office for the service

Property (*=required)		Description
CalculationBase	Service CalculationBase	Can have one of the following values: • Discount – indicates that the NegotiatedBillingRate field is calculated based on the value provided in the Discount property • NegotiatedRate – indicates the Discount field is calculated based on the value provided by the NegotiatedBillingRate property The default value is NegotiatedRate.
Comments	String	Comments
Discount	Decimal	Discount applied to the service total amount
*EndDate	Date	Service end date
*EstimatedTime	Double	Number of hours needed for service completion
FixedFee	Identity	ID of the fixed fee that is associated with the service.
IsCapital	Boolean	Capital flag
IsFixedFee	Boolean	Read-only. Flag indicating if a fixed fee is associated with this service
*NegotiatedBillingRate	Decimal	Negotiated billing rate for the service
*NegotiatedCostRate	Decimal	Negotiated cost rate for the service
OSFunction	Identity	Identifier of the function associated with selected billing role for the service.
ResourceCostRate	Decimal	Read-only. Resource cost rate for the service

Property (*=required)		Description
Resource	Identity	Resource associated with the service
RoleCostRate	Decimal	Read-only. Role cost rate for the service
ServiceId	String	Service ID
ServiceTotal	Decimal	Read-only. Total billing amount for the service
StandardBillingRate	Decimal	Read-only. Standard billing rate for the service
StandardCostRate	Decimal	Read-only. Standard cost rate for the service
*StartDate	Date	Start date for the service
TotalCost	Decimal	Read-only. Total cost of providing the service. The total cost is calculated when retrieved and is not saved.

Related information

"ApiOpportunity" on page 399

"ApiOppService XML" on page 450

ApiOppService XML

```
<root>
   <opportunity>
      ...
      <opportunityservices>
         <opportunityservice>
            <action />
            <serviceid />
            <billingrole>
               <id />
               <name />
               <alternatename />
            </billingrole>
            <standardbillingrate />
            <negotiatedbillingrate/>
            <standardcostrate/>
```

```
            <negotiatedcostrate/>
            <startdate/>
            <enddate/>
            <estimatedtime/>
            <osfunction>
               <id />
               <name />
               <alternatename />
            </osfunction>
            <resource>
               <id />
               <name />
               <alternatename />
               <firstname />
               <lastname />
               <userdefinedid />
            </resource >
            <cost/>
            <totalcost />
            <servicetotal/>
             <iscapital>false</iscapital>
            <resourcecostrate/>
            <rolecostrate/>
            <billingoffice>
               <id />
               <name />
               <alternatename />
            </billingoffice>
            <bobillingrate/>
            <borolecostrate/>
            <boresourcecostrate/>
            <bobillingcurrency>
               <id />
               <name />
               <alternatename />
            </bobillingcurrency>
            <discount />
            <fixedfee>
               <id />
               <name />
               <alternatename />
            </fixedfee>
            <isfixedfee>false</isfixedfee>
            <comments />
            <calculationbase>negotiatedrate</calculationbase>
         </opportunityservice>
      </opportunityservices>
   </opportunity>
```

```
</root>
```

Comments

The XML for the ApiOppServices object is included in the XML for the ApiOpportunity object. The following container elements are mandatory:

```
<root>
   <opportunity>
      <opportunityservices>
         <opportunityservice>
         ...
         </opportunityservice>
      </opportunityservices>
   </opportunity>
</root>
```

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

Example

Not available

Related information

"ApiOpportunity XML" on page 405

"ApiOppService" on page 447

ApiOppService: Exists

```
Public Function Exists(Optional ByVal sId as String = "") As Boolean
```

Purpose

Verifies whether the oportunity service exists in the database

Parameters

Parameter (*=required)	Description
sId	Opportunity service ID to verify

Returns

True if the opportunity service exists, else False.

Remarks

None

Example

Not available

Related information

"ApiOppService" on page 447

ApiOppService: GetBillingOfficeList

```
Public Function GetBillingOfficeList(ByVal salesRepId As String, Optional
ByVal searchCriteria As String = "") As DataSet
```

Purpose

Returns a list of the billing offices that the Sales Rep has access to and whose names match the search criteria. Supply an empty search string to return all billing offices for that Sales Rep.

Parameters

Parameter (*=required)	Description
salesRepId	Sales rep ID
searchCriteria	Search string

Returns

A dataset with two columns (Id, Name)

Remarks

None

Example

Not available

Related information

"ApiOppService" on page 447

ApiOppService: GetBillingRoleList

```
GetBillingRoleList(Optional ByVal billingOfficeId As String = "") As DataSet
```

Purpose

Returns a list of the billing roles for the billing office

Parameters

Parameter (*=required)	Description
billingOfficeId	Billing office ID

Returns

A dataset with two columns (Code, Description)

Remarks

None

Example

Not available

Related information

"ApiOppService" on page 447

ApiOppService: GetCurrencyByBillingRoleList

```
Public Function GetCurrencyByBillingRoleList(ByVal billingRoleId As String,
ByVal effectiveDate As DateTime) As DataSet
```

Purpose

Returns a list of the currencies based on the specified billing role and effective date.

Parameters

Parameter (*=required)	Description
*billingRoleId	Billing role ID
*effectiveDate	Effective date for the currency

Returns

A dataset with 2 columns (Code, Description).

Remarks

None

Example

Not available

Related information

"ApiOppService" on page 447

ApiOppService: GetFixedFeeList

```
Public Function GetFixedFeeList(Optional ByVal sOpportunityId As String = "")
As DataSet
```

Purpose

Returns a list of the opportunity fixed fees.

Parameters

Parameter (*=required)	Description
sOpportunityId	Opportunity ID

Returns

A dataset with the four columns (OpportunityFixedFeeId, Deliverable, FixedFeeAmount, FixedFeeDate).

Remarks

None

Example

Not available

Related information

"ApiOppService" on page 447

ApiOppService: GetList

```
Public Function GetList(ByVal sOpportunityId As String, Optional ByVal
iRetRows As Short = -1) As System.Data.DataSet
```

Purpose

Fills the object with data related to the specified OpportunityId passed in the parameter.

Parameters

Parameter (*=required)	Description
*sOpportunityId	Opportunity ID
iRetRows	Number of rows to return. Default is -1 (returns all rows).

Returns

A dataset with 15 columns (ServicesId, OpportunityId, CustomerId, CustomerName, AlternateName, CustomerAlternateName, BillingRoleId, BillingRoleDescription, FunctionId, FunctionDescription, ResourceId, ResourceName, ResourceAlternateName, EstimatedTime, ServiceTotal) or an empty dataset if nothing is found.

Remarks

None

Example

Not available

Related information

"ApiOppService" on page 447

ApiOppService: GetResourceList

```
Public Function GetResourceList(Optional ByVal searchCriteria As String = "")
As DataSet
```

Purpose

Returns a list of the available resources who have the TTP (Track Project Related Time) feature and who meet the search criteria.

Parameters

Parameter (*=required)	Description
searchCriteria	Search string. The search criteria are applied to the name and alternate name.

Returns

A dataset with two columns (Id, Name)

Remarks

None

Example

Not available

Related information

"ApiOppService" on page 447

ApiProduct

The ApiProduct object allows users to create, retrieve, edit and delete products in the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiProduct

Methods

Properties

Property (*=required)	Type	Description
AvailToGuest	Boolean	Check if available for Client Portal users.
ConvertOnCurrChg	Boolean	Convert when currency is changed.
*CPConnection	ApiConnection	The Connection object must be assigned before any action to the database.
CR_RevRec_GL	String	CR_RevRec_GL
Deleted	Boolean	Flag that indicates if the object has been deleted.
DicPrice	Array of ApiProductCurrency objects	Product prices
DicTeam	Array of ApiProductTeamMember objects	Product team members
DR_RevRec_GL	String	DR_RevRec_GL
*ProductCategoryId	String	Product Category ID
ProductCloseDate	DateTime	Product Close Date
ProductCurrency	String	Product Currency
ProductDescription	String	Product Description
ProductDetailDescription	String	Product Detail Description
ProductDevelopers	String	Product Developer
ProductFixedUnitCost	Boolean	Fixed Unit Cost
*ProductId	String	Product ID

Property (*=required)	Type	Description
ProductManager	String	Product Manager
ProductName	String	Product Name
ProductNumber	String	Product Number
ProductOpenDate	DateTime	Product Open Date
ProductStatus	String	Product Status
ProductUnitCost	Double	Product Unit Cost
ShowOnEng	Boolean	Show on Engagement
ShowOnOpp	Boolean	Show on Opportunity
ShowOnReq	Boolean	Show on Request
sxmlUDF	String	UDF XML string. Includes all information of UDF code and UDF text.
WorkgroupId	String	Workgroup ID.

Related information

"ApiProductCurrency" on page 479

"ApiProductTeamMember" on page 479

ApiProduct: AddProdPrice

```
Public Function AddProdPrice(ByRef oTmpProd As ApiProduct, ByVal sProdCurId As
String, ByVal sCurCode As String, ByVal curPrice As Decimal) As Boolean
```

Purpose

Add a new price to the ApiProduct object oTmpProd

Parameters

Parameter (*=required)	Description
*oTmpProd	ApiProduct object
*sProdCurId	Product currency ID. The unique identifier for the Product Price record. For a new Product Price, pass empty string for sProdCurId, for example, sProdCurId = ""
*sCurCode	Three-letter ISO currency code, for example, "CAD" or "USD"
*curPrice	Product price

Returns

True for a successful addition of a price to the product, else false

Remarks

None

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: AddTeamMember

```
Public Function AddTeamMember(ByRef oTmpProd As ApiProduct, ByVal sResourceId
As String) As Boolean
```

Purpose

Add Team Member into the ApiProduct Team in oTmpProd

Parameters

Parameter (*=required)	Description
*oTmpProd	The Product object to which a new team member will be added.
*sResourceId	The Resource to add to the team in oTmpProd

Returns

True on Success else False

Remarks

None

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: CreateProduct

```
Public Function CreateProduct(ByVal oTmpProd As ApiProduct, Optional ByRef
vProductId As String = "") As Int32
```

Purpose

Add a new product to Changepoint based on the data held in oTmpProd

Parameters

Parameter (*=required)	Description
*oTmpProd	Populated ApiProduct object
vProductId	Returns the ProductId in string if successful.

Returns

0 = Success

Nonzero = Error

Remarks

This process adds a new product but also adds pricing, Product Team and UDF data to Changepoint.

There are two fields that must have a value. They are: ProductId, ProductCategoryId

Example

```
Dim myProduct As New ApiProduct
Dim myPrice As New ApiProductCurrency
Dim dsProdCat As DataSet
Dim sProdCatId As String = ""

myProduct.CpConnection = myCon
'Get a list of product categories and extract the Id
dsProdCat = myProduct.GetProductCategories()
If ds.tables.Count > 0 AndAlso ds.Tables(0)
   For Each mRow As DataRow in ds.tables(0).Rows
      If mRow.Item("ProductCategoryName").ToString = "MyProductCategory" Then
         sProdCatId = mRow.Item ("ProductCategoryId").ToString
         Exit For
      End If
   Next mRow
Else

End If

If sProdCatId = "" Then
   'Handle error
End If
With myProduct
   .ProductName = "My Test Product 1"
   .ProductCategoryId = sProdCatId
   .ProductStatus =
   .ProductNumber = "P001"
   .ProductDescription = "Test Description"

End With
'Add a new price to the object
'Note: Use the GetProductCurrency("") method to extract a list of currency
codes with 'descriptions if the code is unknown and only a description is
known i.e. "American 'Dollar" the returned list can be searched for the
matching description.

If myProduct.AddProdPrice(myProduct, "", "CAD", 2000.00) <> 0 Then
```

```
   'Handle error
End If
'Add a team member named John Smith, using the preferred resources
'of the logged in user.

Dim dsPrefRes as New DataSet
dsPrefRes = myProduct.GetPrefRes()
If dsPrefRes.Tables.Count > 0 AndAlso dsPrefRes.Tables(0).Rows.Count > 0 Then
   For each mDataRow As DataRow In dsPrefRes.Tables(0).Rows
      If dsPrefRes.Item("ResourceName").ToString = "John Smith" Then
         If myProduct.AddTeamMember(myProduct, dsPrefRes.Item
("ResourceId").ToString) = 0 Then
            Exit For
         Else
            'Handle error
         End If
      End If
   Next
End If
If myProduct.CreateProduct(myProduct) <> 0
   'Handle error
End If
```

Related information

"ApiProduct" on page 457

ApiProduct: DeleteProduct

```
Public Function DeleteProduct(ByVal sProductId As String, ByVal sProdCatId As
String) As Int32
```

Purpose

Delete an existing product from Changepoint

Parameters

Parameter (*=required)	Description
*sProductId	The ID of the product to be deleted
*sProdCatId	Product Category to which the product belongs.

Returns

0 = Success

Nonzero = Error

Remarks

UDF (configurable field) data related to the product is also removed.

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: DelProdPrice

```
Public Function DelProdPrice(ByVal sProductId As String, Optional ByVal
sProdCurId As String = "") As Boolean
```

Purpose

Delete product prices

Parameters

Parameter (*=required)	Description
*sProductId	Used to delete all Product Price(s) associated with the Product if no sProdCurId is passed in.
sProdCurId	ID of the Product price record to be deleted

Returns

True on success else False

Remarks

The method deletes product price records from the database.

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: DelTeamMember

```
Public Function DelTeamMember(ByVal sProdCatId As String, ByVal sProductId As
String, Optional ByVal sResourceId As String = "") As Boolean
```

Purpose

Delete team members

Parameters

Parameter (*=required)	Description
*sProdCatId	Product CategoryId of the Product the team member(s) are associated with
*sProductId	ProductId of the Product the team member(s) are associated with
sResourceId	If provided then the single team member associated to the ResourceId will be deleted. If not provided then all team members associated to the ProductId will be deleted.

Returns

True on success else False

Remarks

The method deletes ProductTeam members from the database.

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: GetPrefRes

```
Public Function GetPrefRes() As Dataset
```

Purpose

Retrieve a list of preferred resources

Parameters

None

Returns

Dataset(ResourceId, ResourceName, WorkgroupId, WorkgroupName, GlobalWorkgroupId, GlobalWorkgroupName, Language, BaseCurrency, ResourceType, AltResourceName, WAltName)

WAltName is the Workgroup alternate name.

Remarks

The list is generated based on the logged in user and their preferred resources list as defined in Changepoint.

Can be used for selection of new Team members.

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: GetProductById

```
Public Function GetProductById (ByVal sProductId As String) As ApiProduct
```

Purpose

Retrieves product data for a specified product ID

Parameters

Parameter (*=required)	Description
*sProductId	ID of the product whose data is required

Returns

An ApiProduct object.

Remarks

Includes related Product Team and Price data

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: GetProductByName

```
Public Function GetProductByName (ByVal sProductName As String) As ApiProduct
```

Purpose

Retrieves current product data based on the product's name

Parameters

Parameter (*=required)	Description
*sProductName	Name of the product being retrieved

Returns

An Api product object.

Remarks

Includes related Product Team and Price data

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: GetProductCategories

```
Public Function GetProductCategories() As Dataset
```

Purpose

Returns a list of all product categories

Parameters

None

Returns

Dataset (ProductCategoryID, ProductCategoryName)

Remarks

None

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: GetProductCurrency

```
Public Function GetProductCurrency(ByRef sLang As String) As Dataset
```

Purpose

Retrieve product currencies

Parameters

Parameter (*=required)	Description
*sLang	Language, for example, "E"

Returns

Dataset (Code, Description)

Remarks

The parameter sLang can be an empty string as the language is taken from the base language set in Changepoint.

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: GetProductIdByName

```
Public Function GetProductIdByName(ByVal sName As String) As String
```

Purpose

Get the ProductId based on the name of the product

Parameters

Parameter (*=required)	Description
*sName	Name of the product to retrieve

Returns

ProductId as a string

Remarks

None

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: GetProductPrice

```
Public Function GetProductPrice(ByVal sProductId As String, ByVal sProdCatId
As String, ByVal sLang As String) As Dataset
```

Purpose

Return product pricing data based on the product name, category and language

Parameters

Parameter (*=required)	Description
*sProductId	ID of the product whose pricing data is required.
*sProdCatId	Product category of sProductId
*sLang	Language for pricing

Returns

Dataset (ProductCurrencyId, Code, Price, Description)

Remarks

None

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: GetProducts

```
Public Function GetProducts (Optional ByVal sCatId As String = "", Optional
ByVal sStatus As String = "") As DataSet
```

Purpose

Retrieve all products by the category and/or status of the product

Parameters

Parameter (*=required)	Description
sCatId	ID of the Product Category.
sStatus	Status of the product.

Returns

A Dataset (ProductId, ProductName) ordered by product name

Remarks

None

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: GetProductStatus

```
Public Function GetProductStatus() As Dataset
```

Purpose

Retrieve all available product status values based on the current logged-in Changepoint user.

Parameters

None

Returns

Dataset (Code, Description)

Remarks

The returned status data is filtered based on the Workgroup or GlobalWorkgroup of the logged-in Changepoint user.

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: GetProductTeam

```
Public Function GetProductTeam(ByVal sProductId As String, ByVal sProdCatId As
String) As Dataset
```

Purpose

Retrieve Product team resources based on a specified Product and its category

Parameters

Parameter (*=required)	Description
*sProductId	The ID of the product whose team is required
*sProdCatId	The category of the Product sProductId.

Returns

A Dataset (ResourceId, Name, AlternateName)

Remarks

None

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: GetResource

```
Public Function GetResource() As Dataset
```

Purpose

Retrieve a list of all resources currently assigned to a workgroup

Parameters

None

Returns

Dataset (ResourceId, Name, AlternateName, BaseCurrency)

Remarks

None

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: getUDF

```
Public Function getUDF (ByVal sProductId As String, Optional ByVal retOption
As Byte = 1, Optional ByVal actionResourceId As String = "") As String
```

Purpose

Extract UDF (configurable field) data as an XML string based on the logged in resource and/or product

Parameters

Parameter (*=required)	Description
*sProductId	ID of the product whose UDF data is required.
retOption	Return options: • WithStructure = 0 – Returns the UDF values and structure • OnlyValues = 1 – Returns the UDF values • OnlyStructure = 2 – Returns the UDF structure • OnlyStructureAndOptions = 3 – Returns the UDF structure and options • WithStructureAndOptions = 4 – Returns the UDF structure, metadata tags, values and all the options for codes (except entity based and conditional codes)
actionResourceId	The logged in user or user activating the function.

Returns

UDF string in XML format.

Remarks

Use an empty value for sProductId to have default UDF values included in the returned string.

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: getUDFCodeOptions

```
Public Function getUDFCodeOptions (ByVal codeName As String, Optional ByVal
SearchString As String = "", Optional ByRef lErrCode As Int32 = 0) As String
```

Purpose

Retrieve UDF Code options as an XML string based on the code name and/or the action resource, a search string

Parameters

Parameter (*=required)	Description
*codeName	The name of the code whose options are needed.
SearchString	Search string for filter.
lErrCode	Returns the error number if an error occurred.

Returns

UDF code options in XML.

Remarks

The search string will be used to search the options text for all occurrences of sSearchString.

The code name is of the form "Code#" where "#" is the code number, for example, "Code10".

Example

Not available

Related information

"ApiProduct" on page 457

ApiProduct: UpdateProduct

```
Public Function UpdateProduct(ByRef oTmpProd As ApiProduct) As Int32
```

Purpose

Update Changepoint with updates to an existing product.

Parameters

Parameter (*=required)	Description
*oTmpProd	Populated Product object

Returns

0 = Success

Nonzero = Error

Remarks

Also updates Product Team data as well as pricing and UDF data.

Ensure that ProductId and ProductCategoryId are populated as the update process will fail without these two values.

Example

```
Dim myProduct As ApiProduct
Dim myProdCurrency as New ApiProductCurrency
Dim myPriceDic As New ApiProductCurrency()
Dim sErrorString As String = ""

myProduct.CpConnection = myCon
myProduct = myProduct.GetProductByName("MyTestProduct")
With myProduct
   .ProductName = "My Updated Test Product 1"
   .ProductDescription = "Updated Test Description"
   ...
End With
'Want to update the CAD price. Get an array of ApiProductCurrency objects from
the 'returned an array of objects
Dim sProdCurrId as String = ""
myPriceDic = myProduct.DicPrice
Dim iLen as Integer = 0
Do While iLen < myPriceDic.Length
   myProdCurrency = myPriceDic(iLen)
   If myProdCurrency.Code = "CAD" Then
      myProdCurrency.Price = 5000.00
      Exit Do
   Else
      myProdCurrency = Nothing
   End If
```

```
Loop
If myProduct.UpdateProduct(myProduct) <> 0
   'Handle update error
End If
```

Related information

"ApiProduct" on page 457

ApiProduct: UpdProdPrice

```
Public Function UpdProdPrice (ByVal sProdCurId As String, ByVal sCurCode As
String, ByVal curPrice As Decimal) As Boolean
```

Purpose

Updates the product price in the object oTmpProd

Parameters

Parameter (*=required)	Description
*sProdCurId	Product currency ID.
*sCurCode	Product currency code.
*curPrice	Updated Product Price.

Returns

Returns true upon success else false.

Remarks

None

Example

Not available

Related information

"ApiProduct" on page 457

ApiProductCurrency

The ApiProductCurrency object represents the currency information associated to a product.

Namespace

Changepoint.ChangepointAPI2.ApiProductCurrency

Methods

None

Properties

Property (*=required)	Type	Description
ProdCurId	String	Product Currency ID
CurCode	String	Code Currency
Price	Decimal	Price
Deleted	Boolean	Flag that indicates if the object has been deleted. Deleted = 0 Record is active = 1.

Related information

"ApiProduct" on page 457

"ApiProductTeamMember" on page 479

ApiProductTeamMember

The ApiProductTeamMember object represents all resources that have access to a product.

Namespace

Changepoint.ChangepointAPI2.ApiProductTeamMember

Methods

None

Properties

Property (*=required)	Type	Description
ResourceId	String	Resource ID
Deleted	Boolean	Flag that indicates if the object has been deleted.

Related information

"ApiProduct" on page 457

"ApiProductCurrency" on page 479

ApiProject

The ApiProject object allows users to add, retrieve, update and delete project information within the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiProject

Methods

Properties

Property (*=required)	Type	Description
AllLocations	Boolean	The flag of all locations. Default: true.
AllowCreateSubProject	Boolean	Flag determines if resources can create subprojects under the project.
AllowExpenseLocOverride	Boolean	Flag determines if project resources can override the default expense location when entering project expenses.

Property (*=required)	Type	Description
AllowExpenseOveride	Boolean	Flag determines if expense overrides are allowed.
AllowNonProjectLocations	Boolean	Flag determines if project resources are allowed to choose a location on an expense that is not in the project's work locations list.
AllowTransfer	Boolean	Flag determines if the project can be transfered to other project managers.
AllWorkCodes	Boolean	The flag of all work codes.
AlternateName	String	Project alternate name.
AltSecondLevelApprover	Identity	Identifer of the alternate second level approver.
AvailToGuest	Boolean	Flag determines if the project is available for viewing to guests.
BaselineFinish	DateTime	Baseline finish of project.
BaseLineHours	Double	Baseline hours of project.
BaselineStart	DateTime	Baseline start of project.
Billable	Boolean	Flag determines if the project is billable.
BillingContact	Identity	Identifier of the billing contact assigned to the project. This person receives the invoices for billable work.

Property (*=required)	Type	Description
*BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: CPMetadataCheck.CheckAll = 0 CPMetadataCheck.BypassAll = 1 CPMetadataCheck.OnlyMandatory = 2 For resources, the value set for this flag will be applied to all inner objects on Update and Add.
BypassWorkflow	Boolean	The flag determines if steps in the project workflow can be bypassed. Values: Bypass workflow = 1 Do not bypass workflow = 0
CoMgrAddNewTask	Boolean	When set to true, enables project co-managers to add new tasks.
CoMgrAssignSumTask	Boolean	When set to true, enables project co-managers to assign summary tasks.
CoMgrDeleteTask	Boolean	Flag determines if project co-managers are allowed to delete tasks.
CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object that handles and traces the exception information.
CPProjectType	Identity	Identifier of the Changepoint Project type

Property (*=required)	Type	Description
*Currency	String	Currency string.
*Customer	Identity	Identifier of the customer associated with the project.
CreatedBy	Signature	Resource who created the record.
Deleted	Boolean	Flag that indicates if the record was deleted.
Description	String	Description of the project.
DisplaySubProject	Boolean	When set to true, enables subprojects to be displayed.
*Engagement	Identity	Identifer of the contract associated with the project.
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
ExpenseBudget	Decimal	Amount of expense budget.
*FirstStatus	DateTime	Date resources are required to start entering status updates for their assigned tasks.
IncludePlannedInRollup	Boolean	When set to true, includes the project in planned in rollup.
InheritProjectManager	Boolean	When set to true, inherits project manager.
LabourBudget	Decimal	Amount of labour budget.
MSP_ProjectId	Int32	MSP project ID.
mVersion	String	Read-only. The current version number for internal version control.

Property (*=required)	Type	Description
*Name	String	Project name.
OtherExpenseBudget	Decimal	Amount of other expense budget.
*PlannedFinish	DateTime	Planned finish date.
*PlannedHours	Double	Planned hours.
*PlannedStart	DateTime	Planned start date.
ProjectBillingOffice	Identity	Identifier of the billing office for the project.
ProjectContact	Identity	Identifier of the project contact.
*ProjectId	String	Project ID.
*ProjectManager	Identity	Identifier of the project manager.
ProjectPriority	Double	Project priority number.
*ProjectStatus	String	Project status.
*ProjectType	String	Project type.
ProjectValue	Decimal	Amount of project value.
ProposedPhase	String	Proposed phase.
SecondLevelApproval	Boolean	Flag determines if second level approval is required.
SecondLevelApprover	Identity	Identifier of the second level approver.
StatusFrequency	Int16	Identifies how often resources are required to provide status updates for their assigned tasks.
SubProjectOrder	Int32	Sub-project order

Property (*=required)	Type	Description
SumMgrAddNewTask	Boolean	Flag determines if summary managers are allowed to add new tasks to the project.
SumMgrDeleteTask	Boolean	Flag determines if summary managers are allowed to delete tasks from the project.
SumMgrEditTask	Boolean	Flag determines if summary managers are allowed to edit tasks for the project.
sxmlUDF	String	The UDF string in XML format.
UserDefinedProjectId	String	User defined project ID.
UpdatedBy	Signature	Details of when the record was updated and by whom.
*WorkPerformedFor	Identity	Identifier of the customer for which the work was performed.
*WorkCode	Identity	Identifier of the work code.
*WorkCodeCategory	Identity	Identifier of the work code category.
*WorkLocation	Identity	Identifier of the work location.
*WorkLocationGroup	Identity	Identifier of the work location group.

Related information

"ApiProject XML" on page 486

"ApiTask" on page 672

"ApiTaskAssignment" on page 691

ApiProject XML

```
<root>
   <project>
      <bypassmetadatacheck>checkall</bypassmetadatacheck>
      <createdbyid />
```

```
<updatedbyid />
<projectid />
<name />
<alternatename />
<alllocations>false</alllocations>
<allowcreatesubproject>false</allowcreatesubproject>
<allowexpenselocoverride>false</allowexpenselocoverride>
<allowexpenseoveride>false</allowexpenseoveride>
<allownonprojectlocations>false</allownonprojectlocations>
<allowtransfer>false</allowtransfer>
<allworkcodes>false</allworkcodes>
<altsecondlevelapprover>
   <id />
   <name />
   <alternatename />
   <firstname />
   <lastname />
   <userdefinedid />
</altsecondlevelapprover>
<availtoguest />
<baselinefinish />
<baselinehours />
<baselinestart />
<billable>false</billable>
<billingcontact>
   <id />
   <name />
   <alternatename />
   <firstname />
   <lastname />
</billingcontact>
<bypassworkflow>false</bypassworkflow>
<comgraddnewtask>false</comgraddnewtask>
<comgrassignsumtask>false</comgrassignsumtask>
<comgrdeletetask>false</comgrdeletetask>
<cpprojecttype />
<currency />
<customer>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</customer>
<description />
<displaysubproject>false</displaysubproject>
<engagement>
   <id />
   <name />
```

```
   <alternatename />
   <userdefinedid />
</engagement>
<entity />
<expensebudget />
<firststatus />
<includeplannedinrollup>false</includeplannedinrollup>
<inheritprojectmanager>false</inheritprojectmanager>
<labourbudget />
<otherexpensebudget />
<plannedfinish />
<plannedhours />
<plannedstart />
<projectbillingoffice>
   <id />
   <name />
   <alternatename />
</projectbillingoffice>
<projectcontact>
   <id />
   <name />
   <alternatename />
   <firstname />
   <lastname />
</projectcontact>
<projectmanager>
   <id/>
   <name/>
   <alternatename/>
   <firstname />
   <lastname />
   <userdefinedid />
</projectmanager>
<projectpriority />
<projectstatus />
<projecttype />
<projectvalue />
<proposedphase />
<secondlevelapproval>false<secondlevelapproval>
<secondlevelapprover>
   <id/>
   <name/>
   <alternatename/>
   <firstname />
   <lastname />
   <userdefinedid />
</secondlevelapprover>
<statusfrequency />
```

```
      <summgraddnewtask>false</summgraddnewtask>
      <summgrdeletetask>false</summgrdeletetask>
      <summgredittask>false</summgredittask>
      <sxmludf />
      <userdefinedprojectid />
      <workperformedfor>
         <id />
         <name />
         <alternatename />
      </workperformedfor>
      <workcode>
         <id />
         <name />
         <alternatename />
         </workcode>
      <workcodecategory>
         <id />
         <name />
         <alternatename />
      </workcodecategory>
      <worklocation>
         <id />
         <name />
         <alternatename />
         <userdefinedid />
      </worklocation>
      <worklocationgroup>
         <id />
         <name />
         <alternatename />
         <userdefinedid />
      </worklocationgroup>
      <iUpdate />
      <Tasks>
      ...
      </Tasks>
   </Project>
</root>
```

Comments

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

The XML for the ApiProject object includes the XML for the ApiTask and ApiTaskAssignment objects and may contain XML for UDFs (configurable fields). For more

information on the XML details for included objects, see the documentation for the included objects.

For the use of the `<iUpdate>` flag in ApiProject XML, ApiTask XML, and APiTaskAssignment XML, see "ApiProject: CreateByXML" on page 492.

Example

```
<root>
   <Project>
      <Customer><id>3367544D-2DA3-47B0-894C-
F10E028FAC0A</id><name/><alternatename/></Customer>
      <Engagement><id>C1802B4A-FE60-4622-A2AC-
F37F6664F81C</id><name/><alternatename/></Engagement>
      <ProjectId></ProjectId>
      <ProjectName>Richard_Test803</ProjectName>
      <Description>new from Com</Description>
      <Billable>1</Billable>
      <AllowNonProjectLocations>1</AllowNonProjectLocations>
      <AllLocations>1</AllLocations>
      <AllWorkCodes>1</AllWorkCodes>
      <ProjectType>SR</ProjectType>
      <CPProjectType><id/><name/><alternatename/></CPProjectType>
      <ProjectStatus>A</ProjectStatus>
      <Currency>CAD</Currency>
      <PlannedStart>2005/12/20</PlannedStart>
      <PlannedFinish>2005/12/31</PlannedFinish>
      <PlannedHours>400</PlannedHours>
      <FirstStatus>2005/9/14</FirstStatus>
      <StatusFrequency>1</StatusFrequency>
      <ProjectValue>100</ProjectValue><WorkLocationGroup><id>333A206C-513E-
4CAE-841A-EA73A82E8E81</id><name/><alternatename/></WorkLocationGroup>
      <WorkLocation><id>0409A21B-EFB9-42C4-ADB3-
617279F524EF<id/><name/><alternatename/></WorkLocation>
      <WorkCodeCategory><id>AA9C83C1-6E1E-4974-9DEC-
7C554CC429D2<id/><name/><alternatename/></WorkCodeCategory>
      <WorkCode><id>8A4BEED2-A035-4407-948D-
248681472747</id><name/><alternatename/></WorkCode>
      <udf>
      ...
      </udf>
      <iUpdate>1</iUpdate>
      <Tasks>
      ...
      </Tasks>
   </Project>
</root>
```

Related information

"ApiProject" on page 480

"ApiTask XML" on page 676

"ApiTaskAssignment XML" on page 694

"UDF XML" on page 742

ApiProject: Add

```
Public Overrides Function Public Overrides Function Add(Optional ByRef sId As
String = "") As Int32
```

Purpose

Adds a new project to Changepoint database by assigned project ID.

Parameters

Parameter (*=required)	Description
sId	Project ID for new Project.

Returns

0 = Success

Nonzero = Error

Remarks

Ensure that the Project object properties contain the mandatory data.

Example

```
Dim myPrj as New ApiProject()
With myPrj
.Name="Richard Project2007"
.Customer=New Identity("{}")
.Engagement=New Identity("{}")

.CPConnection=myCon
End With
Dim retId as String=""
Dim ret as Int32=myPrj.Add(retId)
```

Related information

"ApiProject" on page 480

ApiProject: CreateByXML

```
Public Function CreateByXML(ByVal sManagerId As String, ByVal sXML As String,
Optional ByVal bIgnoreWarning As Boolean = False, Optional ByRef sId As String
= "") As Int32
```

Purpose

Creates and updates a project, tasks and task assignments from passed-in XML data.

Parameters

Parameter (*=required)	Description
*sManagerId	ID for the project manager
*sXML	Project (Tasks/TaskAssignments) in XML format.
bIgnoreWarning	Flag to ignore the resource leveling warning.
sId	ID for the new project.

Returns

0 = Success

Nonzero = Error

Remarks

The XML template can be retrieved by the `CreateByXMLTemplate` method.

The mandatory `<iUpdate>` element as a child of `<Project>` is an action flag with available values:

1 - Add

2 - Update

9 - No change

In addition, there are mandatory `<iUpdate>` flags in the Task XML and in the TaskAssignment XML that are child elements of `<Task>` and `<Assignment>` respectively, and that determine the action for the Task and TaskAssignment respectively:

1 - Add

2 - Update

3 - Delete

9 - No change

For example, to create a task and a task assignment for an existing project, set iUpdate (project level) to '2', iUpdate (task level) to '1', and iUpdate (task assignment level) to '1'.

For details of the use of UDF default values, see "UDF default values logic for CreateByXML" on page 746.

Example

```
Dim myPrj as New ApiProject()
myPrj.CPConnection=myCon
Dim ManagerId as String="{}"
Dim strXML as string ="<root></root>"
Dim retId as string = ""
Dim ret as Int32=myPrj.CreateByXML (ManagerId, strXML, False, retId)
```

Related information

"ApiProject" on page 480

"ApiProject: CreateByXML" on page 492

"ApiProject XML" on page 486

ApiProject: CreateByXMLTemplate

```
Public Function CreateByXMLTemplate() As String
```

Purpose

Return the XML structure of the project, task and task assignment.

Parameters

None

Returns

XML structure in a string.

Remarks

None

Example

```
Dim myPrj as New ApiProject()
myPrj.CPConnection=myCon
Dim sRet as string = ""
Dim sRet= myPrj.CreateByXMLTemplate()
```

Related information

"ApiProject" on page 480

"ApiProject XML" on page 486

ApiProject: Delete

```
Public Overrides Function Delete(Optional ByVal sId As String = "") As Int32
```

Purpose

Deletes existing project from Changepoint database.

Parameters

Parameter (*=required)	Description
sId	ID of project that will be deleted.

Returns

0 = Success

Nonzero = Error

Remarks

If a parameter is not provided, the function will return error number -7.

Example

```
Dim myPrj as New ApiProject()
myPrj.CPConnection=myCon
Dim retId as String="{}"
Dim ret as Int32=myPrj.Delete(retId)
```

Related information

"ApiProject" on page 480

ApiProject: Exists

```
Public Overrides Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Checks if the project exists.

Parameters

Parameter (*=required)	Description
sId	ID of project that will be checked.

Returns

True if project exists.

Remarks

If a parameter is not provided the function will return error number -7.

Example

```
Dim myPrj as New ApiProject()
myPrj.CPConnection=myCon
Dim retId as String="{}"
Dim ret as Boolean=myPrj.Exists(retId)
```

Related information

"ApiProject" on page 480

ApiProject: GetBaselineHistory

```
Public Function GetBaselineHistory(ByVal sProjectId As String) As DataSet
```

Purpose

Retrieves the baseline history information for a project

Parameters

Parameter (*=required)	Description
*sProjectId	The ID of the Project

Returns

A dataset containing all the fields in the DS_ProjectBaseline view in the database.

Remarks

Contains the current baseline and all historical baselines for the project in descending order by createdon date. If the parameter is not provided or is incorrect, the function will throw error number -7.

Example

```
Dim myPrj as New ApiProject( )
myPrj.CPConnection = myCon
Dim ds as New Dataset
ds = prj.GetBaselineHistory(sProjectId)
```

Related information

"ApiProject" on page 480

ApiProject: GetById

```
Public Overrides Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Retrieves a project from the Changepoint database.

Parameters

Parameter (*=required)	Description
sId	ID of project that will be retrieved.

Returns

0 = Success

Nonzero = Error

Remarks

If parameter is not provided the function will return error number -7.

Example

```
Dim myPrj as New ApiProject()
myPrj.CPConnection=myCon
Dim prjId as String="{}"
Dim ret as Int32=myPrj.GetById(prjId)
```

Related information

"ApiProject" on page 480

ApiProject: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As String
```

Purpose

Returns the ProjectId based on the UDF Text field and value.

Parameters

Parameter (*=required)	Description
*sUDFField	UDF text field name (for example, Text1)
*sUDFValue	UDF text value

Returns

ProjectId, or an empty string if nothing is found.

Remarks

None

Related information

"ApiProject" on page 480

ApiProject: GetList

```
Public Overloads Overrides Function GetList(Optional ByVal iRetRows As Int16 =
-1) As DataSet

Public Overloads Function GetList(ByVal sCustomerId As String, ByVal
sEngagementId As String, Optional ByVal iRetRows As Int16 = -1) As DataSet
```

Purpose

Retrieves the Project list from Changepoint database.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).
*sCustomerId	Customer ID as a filter.
*sEngagementId	Engagement ID as a filter.

Returns

A dataset of Project list.

Remarks

The dataset will include `CustomerId`, `EngagementId`, `ProjectId`, `ProjectName`, and `UserDefinedProjectId`.

Example

```
Dim myPrj as New ApiProject()
myPrj.CPConnection=myCon
Dim ret as Dataset=myPrj.GetList(-1)
```

Related information

"ApiProject" on page 480

ApiProject: GetManager

```
Public Function GetManager(Optional ByVal sProjectId As String = "") As
DataSet
```

Purpose

Retrieve manager of the project.

Parameters

Parameter (*=required)	Description
sProjectId	ID of the project.

Returns

A dataset for the ResourceId and the Name of the project manager.

Remarks

None

Example

```
Dim myPrj as New ApiProject()
myPrj.CPConnection=myCon
Dim ret as Dataset=myPrj.GetManager("{}")
```

Related information

"ApiProject" on page 480

ApiProject: GetUDF

```
Public Function GetUDF(Optional ByVal retOption As CPUDFReturnType =
CPUDFReturnType.OnlyValues) As String
```

Purpose

Retrieves UDF (configurable field) information for a Project object.

Parameters

Parameter (*=required)	Description
retOption	Return options: retOption = 0 – Returns the UDF values and structure retOption = 1 – Returns the UDF values retOption = 2 – Returns the UDF structure retOption = 3 – Returns the UDF structure and options retOption = 4 – Returns the UDF structure, metadata tags, values and all the options for codes (except entity based and conditional codes)

Returns

UDF string in XML format.

Remarks

Always returns the UDF for the current Project object.

Example

```
Dim myPrj as New ApiProject()
myPrj.CPConnection=myCon
Dim ret as String=myPrj.GetUDF()
```

Related information

"ApiProject" on page 480

ApiProject: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal codeName As String, Optional ByVal
searchString As String = "") As String
```

Purpose

Retrieves UDF (configurable field) code options.

Parameters

Parameter (*=required)	Description
*codeName	Code name for retrieving options.
searchString	An optional search string. If specified, the function will return the options starting with this string.

Returns

Code options string in XML format.

Remarks

This function requires all of the Project object properties to be filled in.

Example

```
Dim myPrj as New ApiProject()
myPrj.CPConnection=myCon
Dim prjId as String="{}"
Dim ret as Int32=myPrj.GetById(prjId)
If ret=0 then
Dim retOptions as String=myPrj.GetUDFCodeOptions("Code1")
End if
```

Related information

"ApiProject" on page 480

ApiProject: SaveBaseline

```
Public Function SaveBaseline(ByVal sProjectId As String) As Int32
```

Purpose

Saves current PlannedStart, PlannedFinish, PlannedHours information for a project into the baseline fields

Parameters

Parameter (*=required)	Description
*sProjectId	The ID of the Project

Returns

0 = Success

Nonzero = Error

Check the CPException.haserror for any errors.

Remarks

Current Planned information is saved to the Baseline fields in the project and a new record is created in the Baselines table.

Example

```
Dim myPrj as New ApiProject( )
myPrj.CPConnection = myCon
Dim iRet as Int32 = -1
iRet = prj.SaveBaseline (sProjectId)
If Not prj.CPException.haserror Then

Else

End if
```

Related information

"ApiProject" on page 480

ApiProject: SaveUDF

```
Public Function SaveUDF(Optional ByVal sUDF As String = "") As Int32
```

Purpose

Saves (Insert/Update) UDF information for a Project object.

Parameters

Parameter (*=required)	Description
sUDF	UDF string in XML format.

Returns

0 = Success

Nonzero = Error

Remarks

Always saves the UDF for the current Project object.

Example

```
Dim myPrj as New ApiProject()
myPrj.CPConnection=myCon
Dim strXML as string ="<root></root>"
Dim ret as Int32=myPrj.SaveUDF(strXML)
```

Related information

"ApiProject" on page 480

ApiProject: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Updates a project in the Changepoint database

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

The Project object is updated with all the properties assigned. The Baseline fields are no longer saved as part of the Update method. For more information, see "ApiProject: SaveBaseline" on page 501.

Example

```
Dim myPrj as New ApiProject()
With myPrj
.ProjectId="{}"
.Name="Richard Project2007"
.Customer=New Identity("{}")
.Engagement=New Identity("{}")

.CPConnection=myCon
End With
Dim ret as Int32=myPrj.Update()
```

Related information

"ApiProject" on page 480

ApiRequest

The ApiRequest object allows users to create, retrieve, update and delete Changepoint requests. With the migration to .NET in Changepoint version 12.0, the interface IRequestBase inherits IBusinessPropertyBase and the class ApiRequest implements IRequestBase.

Namespace

Changepoint.ChangepointAPI2.ApiRequest

Methods

Properties

Property (*=required)	Type	Description
Asset	Identity	Identifier of the related asset.
Assignment	Identity	Identifer of the resource assigned to the request.
*BypassMetadataCheck	CPMetaDataCheck	Determines the level of MetaData checking when adding or updating data. (default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2 The UDF metadatachecking is also bypassed when this switch is set.
*ByPassWorkflow	CPMetaDataCheck	Switch to turn off Workflow for the request. Note: Bypassing workflow is not recommended unless the consequences are understood.
Category	Identity	Identifier of the request category
Cause	String	Description of the cause for the request

Property (*=required)	Type	Description
CreatedBy	Signature	Stores the name of the resource that created the request record and the time it was created.
CreatedTimeZone	String	Time zone where the request was created.
*Customer	Identity	Identifier of the customer or client to whom the request belongs
*CPConnection	ApiConnection	The connection object that must be assigned befoe any action to the database.
DateRequired	Date	Date on which the request is to be completed
Deleted	Boolean	Always returns false
*Description	String	Description of the request
Details	String	Details of the request
*Engagement	Identity	Identifier of the contract related to the request.
Entity	CPEntity	Read-only. The Changepoint entity the object represents.
EstimatedHours	Double	Estimated hours for completion
ExpenseCost	Double	Cost of any expenses
HelpDesk	Identity	Identifier of the related help/support desk
History	String	History of changes to the request.
HistoryUpdate	String	Current changes to be added to the history

Property (*=required)	Type	Description
*Initiator	Identity	Identifier of the resource who initiated the request
mVersion	String	Read-only. The current version number for internal version control.
ParentRequest	Identity	Identifier of the parent request
*Priority	Identity	Identifier of the priority of the request
Product	Identity	Identifier of the product related to the request
ProductCost	Double	The cost of the product
Project	Identity	Identifier of the related project
PublicComment	Boolean	Switch indicating that comments can be placed in the request when viewed from a portal.
Quantity	Integer?	Quantity of product
ReferenceNumber	String(100)	A reference number for the request
RequestDemand	String	XML string of demand estimated for the request.
RequestDemand Necessary	Boolean	Switch that indicates any new requests must also have accompanying demand data
*RequestId	String	The ID of the request. The request object generates the requestid if a create function (Add or CreateByXml) is called. Mandatory only if an update function (Update or UpdateByXml) is called.

Property (*=required)	Type	Description
RequestNumber	String	Depending on system settings, this can be autogenerated or will require manual input.
*RequestType	Identity	Identifer of the type of request. Note that the value of the Id property of the Identity data type is not a GUID but a string
ResourceOrQueue	Boolean	Switch indicating whether the request is assigned to a resource or will be placed on a queue
Responsible	Identity	Identifier of the resource responsible for the request
ServicesCost	Double	Cost of services applied to the request.
Solution	String	Description of the solution
*Status	String(3)	The status of the request. This setting is modified by Workflow unless workflow is bypassed.
SubCategory	Identity	Identifier of the subcategory of the request.
Suspend	Boolean	Switch that suspends the request.
sxmlUDF	String	The UDF string in XML format
UpdatedBy	Signature	Who last updated the record and when

Related information

"ApiRequest XML" on page 509

"ApiRequestDemand" on page 553

ApiRequest XML

```
<root>
   <request>
   <bypassmetadatacheck>checkall</bypassmetadatacheck>
```

```
<createdbyid />
<createdon />
<updatedbyid />
<updatedon />
<asset>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</asset>
<assignment>
   <id />
   <firstname />
   <lastname />
   <alternatename />
   <userdefinedid />
   <name />
</assignment>
<category>
   <id />
   <name />
   <alternatename />
</category>
<cause />
<customer>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</customer>
<daterequired />
<description />
<details />
<engagement>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</engagement>
<estimatedhours />
<expensecost />
<helpdesk>
   <id />
   <name />
   <alternatename />
</helpdesk>
<history />
<historyupdate />
```

```
<initiator>
   <id />
   <firstname />
   <lastname />
   <alternatename />
   <userdefinedid />
   <name />
</initiator>
<parentrequest>
   <id />
   <name />
   <alternatename />
</parentrequest>
<priority>
   <id />
   <name />
   <alternatename />
</priority>
<product>
   <id />
   <name />
   <alternatename />
</product>
<project>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</project>
<publiccomment />
<quantity />
<referencenumber />
<requestid />
<requestnumber />
<requesttype>
   <id />
   <name />
   <alternatename />
</requesttype>
<resourceorqueue />
<responsible>
   <id />
   <firstname />
   <lastname />
   <alternatename />
   <userdefinedid />
   <name />
</responsible>
```

```
   <servicescost />
      <productcost />
   <solution />
   <status />
   <subcategory>
      <id />
      <name />
      <alternatename />
   </subcategory>
   <suspend />
   <createdtimezone />
   <reqdemandnecessary />
   <udf />
   <requestdemand />
   <bypassworkflow />
</request>
</root>
```

Comments

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34. The following special considerations apply to request node lookups:

- For the assignment node, the name needs to be provided if queueId is required; otherwise, the userdefinedid or firstname and lastname are required to find the assigned resourceid for the request.
- For the asset node, the assetnumber as userdefinedid or assettype and description as name are required to look up the id.
- For the parentrequest node, the requestnum as name is required to look up the id.

The XML for the ApiRequest object includes the XML for the ApiRequestDemand object and may contain XML for UDFs (configurable fields). For more information on the XML details for included objects, see the documentation for the included objects.

Example

```
<root>
   <request>
   <bypassmetadatacheck>0</bypassmetadatacheck>
   <createdbyid />
   <createdon />
   <updatedbyid />
   <updatedon />
   <asset>
```

```
   <id />
   <name>11. TZUWYQEJJA. AT</name>
   <alternatename />
   <userdefinedid />
</asset>
<assignment>
   <id />
   <firstname />
   <lastname />
   <alternatename />
   <userdefinedid />
   <name>Default Bug Queue</name>
</assignment>
<category>
   <id />
   <name>Access</name>
   <alternatename />
</category>
<cause />
<customer>
   <id />
   <name>PB Client</name>
   <alternatename />
   <userdefinedid />
</customer>
<daterequired>11/30/2009</daterequired>
<description>PR with contact - API test</description>
<details />
<engagement>
   <id />
   <name>PB Planning</name>
   <alternatename />
   <userdefinedid />
</engagement>
<estimatedhours />
<expensecost />
<helpdesk>
   <id />
   <name>Default Support Desk</name>
   <alternatename />
</helpdesk>
<history />
<historyupdate />
<initiator>
   <id />
   <firstname>John</firstname>
   <lastname>Smith</lastname>
   <alternatename />
```

```
   <userdefinedid />
   <name />
</initiator>
<parentrequest>
   <id />
   <name>HD-2009-10039</name>
   <alternatename />
</parentrequest>
<priority>
   <id />
   <name>4. POBJG RP</name>
   <alternatename />
</priority>
<product>
   <id />
   <name>DL Application 01</name>
   <alternatename />
</product>
<project>
   <id />
   <name />
   <alternatename />
   <userdefinedid>PROJ2008344</userdefinedid>
</project>
<publiccomment />
<quantity />
<referencenumber />
<requestid />
<requestnumber />
<requesttype>
   <id />
   <name>Planning Request</name>
   <alternatename />
</requesttype>
<resourceorqueue />
<responsible>
   <id />
   <firstname />
   <lastname />
   <alternatename />
   <userdefinedid>00012345</userdefinedid>
   <name />
</responsible>
<servicescost />
<productcost />
<solution />
<status>APP</status>
<subcategory>
```

```
      <id>{4CB835EF-3666-11D4-B417-0050DA7707C1}</id>
      <name />
      <alternatename />
   </subcategory>
   <suspend />
   <createdtimezone />
   <reqdemandnecessary />
   <udf>
   ...
   </udf>
   <requestdemand>
   ...
   </requestdemand>
   <bypassworkflow>0</bypassworkflow>
   </request>
</root>
```

Related information

"ApiRequest" on page 504

"ApiRequestDemand XML" on page 554

"UDF XML" on page 742

ApiRequest: Add

```
Public Overrides Function Add(Optional ByRef sId as String = "") As Int32
```

Purpose

Create a new request in Changepoint

Parameters

Parameter (*=required)	Description
sId	The new ID for the new Request.

Returns

0 = Success

Nonzero = Error

Remarks

Creates a new request and returns the new ID value through the reference parameter.

Example

```
Dim myRequest as New ApiRequest
Dim iRet as Int32 = 0
Dim sNewId as String = ""

myRequest.CPConnection = myCon

With myRequest
   'Set property values as necessary, or use an XML string and avoid setting
'properties individually.
   ....
End With
iRet = myRequest.Add(sNewId)
If iRet = 0 Then Return sNewId
```

Related information

"ApiRequest" on page 504

ApiRequest: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, Optional ByRef sId As String
= "") As Int32
```

Purpose

Create a request using an XML string of the Request object in Changepoint.

Parameters

Parameter (*=required)	Description
*sXML	A new request XML string.
sId	ByRef parameter that returns the new RequestId after the request has been added.

Returns

0 = Success

Nonzero = Error

Remarks

The ApiRequest XML structure can be obtained by the GetXMLStructure or the GetByXML methods.

The ByPassMetadataCheck switch will stop any meta data validation in Request and also in Request UDFs.

It is not recommended to set the ByPassWorkflow switch for a request using workflow. Only set this switch when workflow is not expected to be active for the Request.

For details of the use of UDF default values, see "UDF default values logic for CreateByXML" on page 746.

Example

```
Dim myRequest As New ApiRequest
Dim iRet As Int32 = 0
'Set the connection to the database
myRequest.CPConnection = myCon
'Get Request XML structure
sMyXML = myRequest.GetXMLStructure()
'populate sMyXML with data and then pass it to CreateByXML
'
iRet = myRequest.CreateByXML(sMyXML)
```

Related information

"ApiRequest" on page 504

"ApiRequest XML" on page 509

ApiRequest: Delete

```
Public Overrides Function Delete(Optional ByVal sId as String = "") As Int32
```

Purpose

Delete the specified request

Parameters

Parameter (*=required)	Description
sId	The ID of the request to delete.

Returns

0 = Success Nonzero = Error

Remarks

If sId is an empty string, the object's RequestId is used; otherwise, an error is returned.

Because the method falls back, always ensure that the RequestId is passed unless the Request to which the object refers is the request to be deleted.

Example

```
Dim myRequest as New ApiRequest
Dim iRet as Int32 = 0

myRequest.CPConnection = myCon
iRet = myRequest.Delete("{f012628c-3717-11d4-8e11-00105a9e2ddf}")
If iRet = 0 Then
   'Continue processing
Else
   'Handle the error
End If
```

Related information

"ApiRequest" on page 504

ApiRequest: DeleteAttachmentById

```
Public Function DeleteAttachmentById(ByVal sAttachmentId as String) As Int32
```

Purpose

Delete the specified attachment

Parameters

Parameter (*=required)	Description
*sAttachmentId	The ID of the attachment to delete.

Returns

0 = Success

Nonzero = Error

Remarks

If sAttachmentId is an empty string an error is generated.

Example

```
Dim myRequest as New ApiRequest
Dim iRet as Int32 = 0
Dim ds As New DataSet
Dim sAttachmtName As String = "myAttachment.Doc"
Dim sAttachmtId As String = ""

myRequest.CPConnection = myCon
'Get a list of all attachments
ds = myRequest.GetAttachments("{f012628c-3717-11d4-8e11-00105a9e2ddf}")
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.count > 0 Then
   For Each mRow As DataRow in ds.Tables(0).Rows
      If mRow.Item("Name").ToString = sAttachmtName Then
         sAttachmtId = mRow.Item("AttachmentId").ToString
         Exit For
      End If
   Next mRow
End If
iRet = mRequest.DeleteAttachmentById(sAttachmtId)
'Continue processing
```

Related information

"ApiRequest" on page 504

ApiRequest: DeleteByNumber

```
Public Function DeleteByNumber(Optional ByVal sReqNumber as String = "") As
Int32
```

Purpose

Delete the request with the specified request number.

Parameters

Parameter (*=required)	Description
sReqNumber	The number of the request.

Returns

0 = Success

Nonzero = Error

Remarks

If sReqNumber is an empty string, an error is generated.

Example

```
Dim myRequest as New ApiRequest
Dim iRet as Int32 = 0

myRequest.CPConnection = myCon
iRet = myRequest.DeleteByNumber("REQ-008")
If iRet <> 0 Then
   'Handle error
End If
'Continue processing
```

Related information

"ApiRequest" on page 504

ApiRequest: DemandNecessary

```
Public Function DemandNecessary(ByVal sType as String) As Boolean
```

Purpose

To check if request demand data is required for the stated request type

Parameters

Parameter (*=required)	Description
*sType	The request type

Returns

True if request demand data is required

Remarks

None

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: Exists

```
Public Overrides Function Exists(Optional ByVal sId as String = "") As Boolean
```

Purpose

Check whether the specified RequestId exists in Changepoint.

Parameters

Parameter (*=required)	Description
sId	The RequestId

Returns

True if the specified request exists

Remarks

The method does not fall back and use the object's RequestId if the parameter is an empty string. The method will return false.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetAssets

```
Public Function GetAssets(Optional ByVal sCompanyId as String = _NULLGUID,
Optional ByVal sInitiatorId as String = _NULLGUID) As DataSet
```

Purpose

Retrieve all non deleted Assets where the Asset Assignee is sInitiatorId or sCompanyId

Parameters

Parameter (*=required)	Description
sCompanyId	ID of the related customer
sInitiatorId	The ID of the initiator of the request. May be a resource or contact.

Returns

A dataset (AssetId, AssetNumber, AssetName, type)

Remarks

None

Example

```
Dim myRequest as New ApiRequest
Dim iRet as Int32 = 0
Dim mLookup as New ApiLookup
```

```
Dim ds As New DataSet
Dim sCompanyName As String = "SomeCompany Inc"
Dim sCompanyId As String = ""

'Set the Connection
myRequest.CPConnection = myCon
mLookup.CPConnection = myCon

'Make use of the Lookup object to retrieve the Customer/Company Id
ds = mLookup.GetLookupFields("Customer", "CustomerId", , , "Name = " &
   sCompanyName & " And Deleted = 0")
'Extract the Id from the dataset
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count = 1 Then
   sCompanyId = ds.Tables(0).Rows(0).Item("CustomerId").ToString
End If
're-use the dataset
ds = New DataSet
ds = myRequest.GetAssets(sCompanyId, )
'process the returned dataset
```

Related information

"ApiRequest" on page 504

ApiRequest: GetAssignees

```
Public Function GetAssignees(ByVal sSearchString as String) As DataSet
```

Purpose

Retrieve a list of resources that can be assigned to a request

Parameters

Parameter (*=required)	Description
*sSearchString	The string may be a resource name or resource alternate name. It may also be the name of a queue or just an empty string. The pipe character (|) is used to escape special characters

Returns

A DataSet (Id, Name, AlternateName, Restype where ResType = 1 for Resource data or 2 for Queue data)

Remarks

It is not necessary to have any data in the object itself.

Example

```
Dim myRequest as New ApiRequest
Dim ds As New DataSet

'Set the Connection
myRequest.CPConnection = myCon
'pass in an empty string to get all resources
ds = myRequest .GetAssignees("")
If ds.Tables.count > 0 AndAlso ds.Tables(0).Rows.Count > 0 Then
   'iterate and extract data
End If
```

Related information

"ApiRequest" on page 504

ApiRequest: GetAssigneesBySupportDeskId

```
Public Function GetAssigneesBySupportDeskId(ByVal sId As String)) As DataSet
```

Purpose

Retrieve a list of resources and queue by a support desk that can be assigned to a request.

Parameters

Parameter (*=required)	Description
*sId	ID of the support desk

Returns

A DataSet (Id, Name, AlternateName, Restype where ResType = 1 for Resource data or 2 for Queue data)

Remarks

It is not necessary to have any data in the object itself.

Example

```
Dim myRequest as New ApiRequest
Dim ds As New DataSet
myRequest.CPConnection = myCon
ds = myRequest.GetAssigneesBySupportDeskId("{0FD91759-8C51-11D3-8AA6-
0060975AEC0F}")
If ds.Tables.count > 0 AndAlso ds.Tables(0).Rows.Count > 0 Then
   'iterate and extract data
End If
```

Related information

"ApiRequest" on page 504

ApiRequest: GetAttachmentById

```
Public Function GetAttachmentById(ByVal sAttachmentId as String) As Byte()
```

Purpose

Retrieves an attachment from Changepoint

Parameters

Parameter (*=required)	Description
*sAttachmentId	The ID of the request's attachment.

Returns

A byte array containing the requested attachment

Remarks

None

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetAttachments

```
Public Function GetAttachments(ByVal sRequestId as String) As DataSet
```

Purpose

Retrieve information on all attachments related to the specified request.

Parameters

Parameter (*=required)	Description
*sRequestId	The ID of the request whose attachment data is being retrieved.

Returns

A dataset (AttachmentId, Name, Description, Shared, FileName)

Remarks

None

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetById

```
Public Overrides Function GetById(Optional ByVal sId as String = "") As Int32
```

Purpose

Retrieve current request data into the object

Parameters

Parameter (*=required)	Description
sId	The ID of the request whose data is being retrieved.

Returns

0 = Success

Nonzero = Error

Remarks

Fills the object with current data

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetByNumber

```
Public Function GetByNumber(Optional ByVal sNumber as String = "") As DataSet
```

Purpose

Retrieve current request data based on the request number passed.

Parameters

Parameter (*=required)	Description
sNumber	The request number of the request required.

Returns

Dataset

Remarks

Returns the same fields as GetbyId

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetByXML

```
Public Function GetByXML(Optional ByVal sXML as String = "",Optional ByVal
sRequestId as String = "") As String
```

Purpose

Retrieve current request data based on the request number passed and the fields in the XML string.

Parameters

Parameter (*=required)	Description
sXML	The XML string containing fields where data is required.
sRequestId	The request whose data is being retrieved.

Returns

An XML string.

Remarks

The XML string sXML can be varied in the number of fields. If only a small subset of data is required, any unwanted fields can be removed from sXML. The returned XML string will mirror sXML in the fields contained in the XML.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetCategories

```
Public Function GetCategories(ByVal sRequestTypeId as String) As DataSet
```

Purpose

Retrieve all request categories for a specified request type.

Parameters

Parameter (*=required)	Description
*sRequestTypeId	The request type

Returns

A dataset (Code, Description), where Code is the unique identifier for the category.

Remarks

sRequestType is always a string with a maximum of three characters. Anything else will create an error.

Example

Not available

ApiRequest: GetCompanies

```
Public Function GetCompanies(Optional ByVal sSearchString as String = "",
Optional ByVal sInitiatorId as String = "") As DataSet
```

Purpose

Retrieve all customers of the request based on sSearchString and sInitiatorId

Parameters

Parameter (*=required)	Description
sSearchString	The search string can be a customer Name, AlternateName or UserDefinedCustomerId. The pipe character(|) can be used to escape a special character.
sInitiatorId	The request initiator

Returns

A dataset (CustomerId, Name, AlternateName, UserDefinedId).

Remarks

The customers returned are based on the search string and on whether the initiator created the Customer record or has access to the customer.

Similarly, customer records will also be returned if the initiator has access to contracts under the specified customer or created any contracts under the specified customer.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetEngagements

```
Public Function GetEngagements(ByVal sCompanyId as String) As DataSet
```

Purpose

Retrieve all Engagements under a specified customer.

Parameters

Parameter (*=required)	Description
*sCompanyId	The customer whose contracts are required

Returns

A dataset (EngagementId, Name, AlternateName).

Remarks

Returns all contracts under the specified customer where the logged in user has access to the contract.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As String
```

Purpose

Returns the RequestId based on the UDF Text field and value.

Parameters

Parameter (*=required)	Description
*sUDFField	UDF text field name (for example, Text1)
*sUDFValue	UDF text value

Returns

RequestId, or an empty string if nothing is found.

Remarks

None

Related information

"ApiRequest" on page 504

ApiRequest: GetInitiatorName

```
Public Function GetInitiatorName(ByVal sInitiatorId as String) As DataSet
```

Purpose

Retrieve the name of the resource who created the request.

Parameters

Parameter (*=required)	Description
*sInitiatorId	The ID of the resource.

Returns

A dataset (Name, AlternateName).

Remarks

Returns the initiator name whether the initiator is a resource or contact.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetInitiators

```
Public Function GetInitiators(Optional ByVal sSearchString as String = "") As
DataSet
```

Purpose

Retrieve a list of initiators based on the specified search string.

Parameters

Parameter (*=required)	Description
sSearchString	The search string. May contain a resource or contact name or alternate name. The string can use the pipe character ((|)) to escape special characters.

Returns

A dataset (Name, AlternateName, Id, ResType) where ResType = 0 means the ID is a Changepoint resource and resType = 1 means a Changepoint Contact.

Remarks

Returns the initiator name whether the initiator is a resource or contact.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows as Int16 = -1) As
DataSet
```

Purpose

Retrieve a list of requests based on the parameter iRetRows.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).

Returns

A dataset (RequestId, EngagementId, RequestNumber, RequestType, RequestStatus)

Remarks

The records are ordered based on the RequestNumber

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetPriorities

```
Public Function GetPriorities(Optional ByVal sEngagementId as String = _
NULLGUID, Optional ByVal sProductId as String = _NULLGUID) As DataSet
```

Purpose

Retrieve a list of priorities based on the parameters for EngagementId and ProductId

Parameters

Parameter (*=required)	Description
sEngagementId	The contract ID
sProductId	The product ID

Returns

A dataset (Code, Description, NoSLA)

Remarks

In this version, the method makes use of the request type and takes its value from the object itself. If this call is being made from an empty object, set the Request type property for greater filtering. If there is no request type specified then all request types will be examined.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetProductNameById

```
Public Function GetProductNameById(ByVal sProductId as String) As String
```

Purpose

Retrieve the name of the product identified by the ProductId

Parameters

Parameter (*=required)	Description
*sProductId	The product whose name is required.

Returns

The product name as a string

Remarks

No name will be returned if the product has been deleted.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetProducts

```
Public Function GetProducts(ByVal sProductId as String) As DataSet
```

Purpose

Retrieve the name of the product identified by the ProductId

Parameters

Parameter (*=required)	Description
*sProductId	The product whose name is required.

Returns

A dataset (ProductId, Name, Type).

Remarks

Type = 1 ignores any product open or close dates.

Type = 2 indicates a product that has open and close dates and the date of the query places the product within the open date time frame.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetProjects

```
Public Function GetProjects(Optional ByVal sEngagementId as String = "",
Optional ByVal sSearchString as String = "") As DataSet
```

Purpose

Retrieve all projects under sEngagementId and meeting the criteria in sSearchString

Parameters

Parameter (*=required)	Description
sEngagementId	The contract whose projects are being searched.
sSearchString	Can be any project name or alternate name. The pipe character is used to escape special characters.

Returns

A dataset (ProjectId, Name, AlternateName).

Remarks

The logged-in Changepoint user also acts to filter returned projects. The logged-in user must be a manager or co-manager on the project, or a resource who is assigned to a task within the project, or a resource who has view access to the project.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetRDSetInfo

```
Public Function GetRDSetInfo(ByVal sReqType as String) As DataSet
```

Purpose

Retrieve the two switches for resource demand that control whether demand is necessary and whether demand data will be viewable from Request in Enterprise.

Parameters

Parameter (*=required)	Description
*sReqType	The request type whose settings are being extracted.

Returns

A dataset (EnableResDemandInfo, ResDemandInfoMandatory).

Remarks

EnableResDemandInfo indicates whether resource demand information is visible in the Request interface in Changepoint Enterprise, while ResDemandInfoMandatory indicates that Resource Demand must be included for any new requests.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetRequests

```
Public Function GetRequests(Optional ByVal sCustomerName as String = "",
Optional ByVal sCustomerId as String = "", Optional ByVal sEngagementName as
String = "", Optional ByVal sEngagementId as String = "", Optional ByVal
sProjectName as String = "", Optional ByVal sProjectId as String = "",
Optional ByVal sRequestType as String = "", Optional ByVal sPriorityId as
String = "", Optional ByVal sCategoryId as String = "", Optional ByVal
sSubCategoryId as String = "") As DataSet
```

Purpose

Retrieve all requests based on the values in the parameters.

Parameters

Parameter (*=required)	Description
sCustomerName	Name of the customer. If multiple customer names are passed in, each customer name must be separated by a pipe character (|).
sCustomerId	ID of the customer. If multiple customer IDs are passed in, each ID must be separated by a pipe character (|).

Parameter (*=required)	Description
sEngagementName	Name of the Contract. If multiple enagagement names are passed in, each name must be separated by a pipe character (|).
sEngagementId	ID of the Contract. If multiple contract IDs are passed in, each ID must be separated by a pipe character (|).
sProjectName	Name of the project. If multiple project names are passed in, each name must be separated by a pipe character (|).
sProjectId	Id of the project. If multiple project IDs are passed in, each ID must be separated by a pipe character (|).
sRequestType	The request type. If multiple request types are passed in, each requst type must be separated by a pipe character (|).
sPriorityId	The request priority. If multiple request prioirities are passed in, each priority must be separated by a pipe character (|).
sCategoryId	The request category. If multiple request categories are passed in, each category must be separated by a pipe character (|).
sSubCategoryId	The request sub category. If multiple request sub categories are passed in, each sub category must be separated by a pipe character (|).

Returns

A dataset (RequestId, RequestNum).

Remarks

Criteria for the search can be any combination of parameter values. Keep in mind the relationship between parameter fields and their effect on requests. The greater the number of parameters the narrower the search which raises the chances of no data being returned due to conflicting parameters.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetResponsibles

```
Public Function GetResponsibles(Optional ByVal sSearchString as String = "")
As DataSet
```

Purpose

Retrieve all resources that can be assigned to a request.

Parameters

Parameter (*=required)	Description
sSearchString	The search string can contain a resource name or alternate name. Any special characters can be escaped using the pipe character in front of the special character.

Returns

A dataset (ResourceId, Name, AlternateName).

Remarks

Resources must belong to a Workgroup in order to be assigned to requests or tasks.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetStatuses

```
Public Function GetStatuses(ByVal sRequestId As String, ByVal sActionResource
As String) As DataSet
```

Purpose

Retrieve all possible status values based on the request and the action resource as determined by Workflow.

Parameters

Parameter (*=required)	Description
*sRequestId	The request ID.
*sActionResource	The ID of the resource querying for information.

Returns

A dataset (Code, Description).

Remarks

Returned status values are based on sActionResource and their access to different statuses as determined by Workflow, the Request type and the current state of sRequestId in the Workflow.

Requests that are not attached to any Workflow will cause this method to return the entire status list.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetSubCategories

```
Public Function GetSubCategories(Optional ByVal sRequestCategoryId As String =
_NULLGUID) As DataSet
```

Purpose

Retrieve all sub categories for the specified request category that are not deleted from the system.

Parameters

Parameter (*=required)	Description
sRequestCategoryId	The request category

Returns

A dataset (Code, Description).

Remarks

None

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetSupportDesks

```
Public Function GetSupportDesks(ByVal sEngagementId As String, ByRef
sDefaultId As String, ByRef iAllowOverride As Int32) As DataSet
```

Purpose

Retrieve a list of all available Support Desks. Also return the specified default Support Desk and the value for the Support Desk Override switch.

Parameters

Parameter (*=required)	Description
*sEngagementId	The Contract whose available Support Desks are to be viewed.
*sDefaultId	Return parameter. The Default Support Desk
*iAllowOverride	Return parameter. The Support Desk Override switch.

Returns

A dataset (HelpDeskId, AllowHDOverride).

Remarks

None

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetTypes

```
Public Function GetTypes(Optional ByVal sEngagementId As String = "") As
DataSet
```

Purpose

Retrieve a list of request types in Changepoint.

Parameters

Parameter (*=required)	Description
sEngagementId	Engagement ID

Returns

If sEngagementId is provided, then returns request types available to this contract and login user, otherwise returns all request types.

A dataset (Code, Description, AvailToGuest, AvailToBudgetContingency, AvailToBudgetItems, EnableResDemandInfo, ResDemandInfoMandatory).

Remarks

The returned dataset also includes several switch settings that set access to the type and whether resource demand data is necessary for new requests of that type.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetUDF

```
Public Function GetUDF(Optional ByVal UDFOption As CPUDFReturnType =
CPUDFReturnType.OnlyValues) As String
```

Purpose

Retrieve the UDF (configurable field) XML string for the request.

Parameters

Parameter (*=required)	Description
UDFOption	The CPUDFReturnType Values are: • OnlyStructure - return only UDF XML structure, no field data • OnlyStructureAndOptions - return UDF structure and option data without field data • OnlyValues - return only UDF fields with data, do not include any field structure or options • WithStructure – return UDF field data with full field structure • WithStructureAndOptions – return UDF field data with full structure and all field options

Returns

A UDF XML string that varies in detail based on the value of the parameter UDFOption.

Remarks

The method uses UDFOption, RequestId and the Request Type to determine the contents of the UDF XML string returned.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal sCodeName As String, Optional ByVal
sSearchString As String = "") As String
```

Purpose

Retrieve the UDf code options as a XML string based on sCodeName or sSearchString.

Parameters

Parameter (*=required)	Description
*sCodeName	The name of the UDF code whose options are required. i.e "Code1", Text6
sSearchString	A search string used to extract one or more UDf Codes.

Returns

A UDF XML string that varies in detail based on the value of the parameter UDFOption.

Remarks

None

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetUDFXMLStructure

```
Public Function GetUDFXMLStructure(Optional ByVal sReqTypeId as String = "")
As String
```

Purpose

Retrieve the UDF XML structure for the specified type sReqTypeId.

Parameters

Parameter (*=required)	Description
sReqTypeId	The request type whose UDF XML structure is required.

Returns

A UDF XML string of the UDF structure for sReqTypeId.

Remarks

None

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: GetXMLStructure

```
Public Function GetXMLStructure()As String
```

Purpose

Retrieve the XML structure for the request object.

Parameters

None

Returns

An XML string of the request.

Remarks

None

Example

Not available

Related information

"ApiRequest" on page 504

"ApiRequest XML" on page 509

ApiRequest: SaveUDF

```
Public Overrides Function SaveUDF(Optional ByVal sXML as String = "", ByVal
bypassMetadata as CPMetadataCheck) As Int32
```

Purpose

Update the request UDF data based on the parameter sXML.

Parameters

Parameter (*=required)	Description
sXML	UDF XML string with data.
bypassMetadata	Determines the level of metadata checking for sXMLUDF. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

0 = Success

Nonzero = Error

Remarks

The RequestId and Request Type are first taken from the object. If there are no values then the sXML is accessed for these values. An error is thrown if the RequestId and Request Type still are not present.

The RequestId and the Request Type are required values for the save process.

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequest: SetPropertiesByXML

```
Public Function SetPropertiesByXML(Optional ByVal sXML as String = "",
Optional ByVal bNotInitialize as Boolean = True) As Int32
```

Purpose

Set the properties of the object with the values in the fields of sXML.

Parameters

Parameter (*=required)	Description
sXML	Request XML string with data.
bNotInitialize	If False the object is re-initialized. It is recommended to always set this switch to True

Returns

0 = Success

Nonzero = Error

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods.

The minimal XML that can be sent in to the method is:

```
<root>
   <request></request>
</root>
```

In the case of an empty Request object and sXML has a value for RequestId, the object will first retrieve existing data for the RequestId in sXML and then replace existing field values with new ones from sXML.

If sXML contains fields with no values, the method will overwrite the existing field value and replace it with the "empty" value from sXML, in effect erasing the field.

When the original field value is to be retained, the field in sXML must be removed or its value must be the same as the current data held in the object.

Example

Not available

Related information

"ApiRequest" on page 504

"ApiRequest XML" on page 509

"ApiRequest: CreateByXML" on page 516

"ApiRequest: UpdateByXML" on page 550

ApiRequest: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Update the database with data held in the objects properties.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

The update process will update all fields in Request.

The ByPassMetadataCheck switch will stop any metadata validation in Request and also in Request UDF's

It is not recommended to set the ByPassWorkflow switch for a request using Workflow. Only set this switch where Workflow is not expected to be active for the Request.

Example

```
Dim myRequest as New ApiRequest
Dim iRet as Int32 = 0
Dim sNewId as String = ""

myRequest.CPConnection = myCon
'Get latest data into the object
iRet = myRequest.GetById("{f012868e-3717-11d4-8e11-00105a9e2ddf }")
'Make changes
With myRequest
   'Set property values as necessary, or use an XML string and avoid the With
'statement.
   .Responsible.Id = "{bff37287-abbf-483f-8f0b-da17a8f24ce5 }"
   .ProductCost = 1200.00
   .Product.Id = "{ddbd4f63-363b-48f4-9ee8-05a852ebfd5a }"
   .Quantity = 5
   .Assignment.Id = "{4fb06751-9a3b-42f5-801a-73032b97edff }"
   .EstimatedHours = 100
   .
End With
iRet = myRequest.Update()
```

Related information

"ApiRequest" on page 504

ApiRequest: UpdateByXML

```
Public Function UpdateByXML(Optional ByVal sXML As String = "", Optional ByVal
sRequestId As String = "") As Int32
```

Purpose

Update the database using an XML string of the Request object containing update information.

Parameters

Parameter (*=required)	Description
sXML	A request XML string with updated fields.
sRequestId	Request ID of the request being updated.

Returns

0 = Success

Nonzero = Error

Remarks

The ApiRequest XML structure can be obtained by GetXMLStructure or GetByXML methods.

The update process will update all fields in Request.

The ByPassMetadataCheck switch will stop any meta data validation in Request and also in Request UDF's.

It is not recommended to set the ByPassWorkflow switch for a request using workflow. Only set this switch when workflow is not expected to be active for the Request.

It is recommended to remove the XML tags if you want to retain the original data in the database.

The method uses the following sequence to find the request ID:

1. If the sRequestId parameter is passed in, the method uses this value for the request ID.
2. If this fails, the method attempts to extract the request ID from <requestid> in the XML.
3. If this fails, the request ID is taken from the object properties.
4. If this fails, an attempt is made to look up the request ID using <requestnumber> in the XML.

Example

```
Dim myRequest As New ApiRequest
Dim sMyXML As String = ""
Dim iRet As Int32 = 0
'Set the connection to the database
myRequest.CPConnection = myCon
```

```
'Get Request XML structure with existing data if it's necessary.
sMyXML = myRequest.GetByXML("{5B401107-53D3-4328-8369-3F6F2BDAA86C}")
'modify sMyXML and then pass it to UpdateByXML
'
iRet = myRequest.UpdateByXML(sMyXML)
```

Related information

"ApiRequest" on page 504

"ApiRequest XML" on page 509

ApiRequest: UploadAttachment

```
Public Function UploadAttachment(ByVal sRequestId As String, ByVal sName As
String, ByVal sDescription As String, ByVal sFileName As String, ByVal iShared
As Int32, ByRef ArrAttachment As Byte()) As String
```

Purpose

Upload an attachment to Changepoint and link the attachment to a specified request.

Parameters

Parameter (*=required)	Description
*sRequestId	The Request to which the attachment belongs.
*sName	The name of the attachment
*sDescription	A description of the attachment
*sFileName	The filename of the attachment. The path to the file is part of sFileName
*iShared	Switch indicating whether the attachment can be shared.
*arrAttachment	The attachment as a byte array.

Returns

String (The new AttachmentId)

Remarks

None

Example

Not available

Related information

"ApiRequest" on page 504

ApiRequestDemand

The ApiRequestDemand object is a subobject of the Request object and reflects a request as a demand element.

Namespace

Changepoint.ChangepointAPI2.ApiRequestDemand

Methods

The SaveRequestDemand has been removed as a separate method. The functionality is now part of Request.Add and Request.Update. Fill in the Request.RequestDemand property as appropriate.

Properties

Property (*=required)	Type	Description
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
*CPConnection	ApiConnection	Write-only. The connection object that must be assigned before any action to the database.

Related information

"ApiRequest" on page 504

"ApiRequestDemand XML" on page 554

ApiRequestDemand XML

```
<root>
   <requestdemand>
      <r status='' >
         <RequestDemandId></RequestDemandId>
         <rid>{615FC0FC-1B7A-4519-A6BC-BCD74584339D}</rid>
         <wid>{6C9EF9D1-DAEF-11D2-882B-006097B596A6}</wid>
         <fid></fid>
         <hrs>8</hrs>
         <sdate>05/18/2007</sdate>
         <edate>05/19/2007</edate>
         <softbooked>1</softbooked>
         <updateavailability>0</updateavailability>
         <showontimesheet>0</showontimesheet>
      </r>
      <checkall>
         <softbooked/>
         <updateavailability/>
         <showontimesheet/>
      </checkall>
   </requestdemand>
```

```
</root>
```

Comments

Element	Comments
<r status=''>	Possible values for status are: • '' which means the record will be ignored • 'N' new record to be inserted • 'E' edit existing record • 'D' delete existing record.
<RequestDemandId>	RequestDemandId
<rid>	ResourceId
<wid>	WorkgroupId
<fid>	FunctionId
<durationchange>	• 0 If duration has not changed • 1 if duration has changed When durationchange is set to 1, the records in RequestDemandDailyAllocation table will be deleted. When any one of the tags <sdate>, <edate> or <hrs> is changed, you must set <durationchange> to 1 to ensure that the records in RequestDemandDailyAllocation table can be updated accordingly.
<hrs>	Required hours
<sdate>	Start Date
<edate>	End Date
<softbooked>	Soft-booked
<updateavailability>	Add to calendar
<showontimesheet>	Show on time sheet If the softbooked, updateavailability or showontimesheet is set to 1 in the <checkall> section, the corresponding tag will be set to 1 for each RequestDemand record included in the XML string.

Example

Not available

Related information

"ApiRequestDemand" on page 553

"ApiRequest XML" on page 509

ApiRequestDemand: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows As Int16 = -1) As
DataSet
```

Purpose

Retrieve request demand records.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).

Returns

A dataset of RequestDemand records upon success and nothing in a dataset upon error.

Returns all rows when the method is called with parameter -1.

Remarks

Check log file upon error

Example

```
Dim oReqDemand As New ApiRequestDemand
Dim dsRet As DataSet

oReqDemand.CPConnection = myCon
dsRet = oReqDemand.GetList ()
```

Related information

"ApiRequestDemand" on page 553

ApiRequestDemand: GetByRequestId

```
Public Function GetByRequestId(ByVal Optional sRequestId As String = "") As
String
```

Purpose

Retrieve request demand records for a request

Parameters

Parameter (*=required)	Description
sRequestId	Optional. The ID of the Request

Returns

Request demand records for a request in an XML string

Request demand XML schema only when the method is called with an empty string or when the passed request has no request demand records.

Remarks

Ensure that you input the correct ResourceId and WorkgoupId. The resource must belong to the workgoup in Changepoint.

Check log file upon error

Example

```
Dim oReqDemand As New ApiRequestDemand
Dim sRet As String

oReqDemand.CPConnection = myCon
sRet = oReqDemand.GetByRequestID ("{FF636BB5-B108-4742-BA1D-E7D7BA9CDF3D}")
```

Related information

"ApiRequestDemand" on page 553

"ApiRequestDemand XML" on page 554

ApiRequestTime

The ApiRequestTime object allows users to add, retrieve, update, approve or reject request time information and submit request time, time and standard time information for resources within the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiRequestTime

Methods

Properties

Property (*=required)	Type	Description
AdjustedOvertimeHours	Double	The adjusted overtime hours.
AdjustedRegularHours	Double	The adjusted regular hours.
AdjustmentStatus	Boolean	The Adjustment Status.
AdjustmentTimeParent	String	The AdjustmentTimeParent ID.
AdjustmentTimeStatus	String	The Adjustment Time Status.
BookedByResource	Identity	Read-only. Identifier of the resource that booked time.
*BypassMetadataCheck	CPMetaDataCheck	Determines the level of MetaData checking when adding or updating data. (default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
*CPConnection	ApiConnection	Write-only. The connection object that must be assigned before any action to the database.
CreatedBy	Signature	The creator of the record and when it was created.
CreatedOn	Date	Date and time that the record was created.
Customer	Identity	Read-only. Identifier of the customer.
Deleted	Boolean	Flag that indicates if the object has been deleted.
Description	String	Information about how the time was spent.

Property (*=required)	Type	Description
Editable	Boolean	Read-only. When set to true, the request time is editable.
EndTime	Date	The end date and time of the hours worked. This value is calculated from regular and overtime hours with the start time internally.
Engagement	Identity	Read-only. Identifier of the contract.
Entity	CPEntity	Read-only. The Changepoint entity that the object represents.
FedAudit	Boolean	Read-only. When set to true, auditing is enabled.
HistoryComments	String	Request time history commentsRead-only.
mVersion	String	The current version number for internal version control.
OvertimeHours	Double	The number of overtime hours spent working on the request.
*RegularHours	Double	The number of hours spent working on the request.
*Request	Identity	Identifier of the request
*RequestTimeId	String	The ID of the Changepoint Request Time. Optional when adding the requesttime record.
*Resource	Identity	Identifier of the resource.
StartTime	Date	The start date and start time of the hours worked.

Property (*=required)	Type	Description
TimeCode1	Identity	Custom time code field.
TimeCode2	Identity	Custom time code field.
TimeCode3	Identity	Custom time code field.
*TimeZone	Identity	The time zone in which the hours were worked.
UpdatedBy	Signature	The updater of the record and when it was updated.
UpdatedOn	Date	Date that the record was most recently updated.
*WorkCode	Identity	Identifier of the work code that describes the type of work requested.
*WorkCodeCategory	Identity	Identifier of the work code category for the request.
*WorkLocation	Identity	Identifier of the location where the requested work was performed.
*WorkLocationGroup	Identity	Identifier of the work location group.

ApiRequestTime: Add

```
Public Overloads Overrides Function Add(Optional ByRef sId As String = "") As
Int32
```

Purpose

Insert a new request time record.

Parameters

Parameter (*=required)	Description
sId	The ID of the request time record.

Returns

0 = Success

Nonzero = Error

Remarks

If parameter sId is not provided, newly created requesttimeid is assigned to property RequestTimeId

If parameter sId is provided, newly created requesttimeid is assigned to sId.

Check log file upon error.

Example

```
Dim oReqTime As New ApiRequestTime
Dim iRet As Int32
Dim sId As String

oReqTime.CPConnection = myCon

With oReqTime
   .BookedByResource.Id = "{615FC0FC-1B7A-4519-A6BC-BCD74584339D}"
   .CreatedBy.ActionPerson.Id = "{615FC0FC-1B7A-4519-A6BC-BCD74584339D}"
   .Description = "Bug verification"
   .RegularHours = 8
      .OvertimeHours = 2
.Request.Id = "{505BDF7D-064C-4D07-A69E-112C0A14838D}"
.Resource.Id = "{BAA411CF-F648-47CD-A4F1-12C259EE25BF}"
   .StartTime = "05/14/2006 9:00:00 AM"
   .TimeCode1 = "{44B890C0-6AB9-47B7-A9F1-AF5D251DACF5}"
   .TimeCode2 = "{098D4E7E-AD20-47B7-AE0C-5B1E5D9A48EE}"
   .TimeCode3 = "{E33B0D35-904C-492A-879B-67CBDFCC9573}"
   .TimeZone = "EST"
   .WorkLocationGroup.Id = "{333A206C-513E-4CAE-841A-EA73A82E8E81}"
   .WorkLocation.Id = "{4F080CE6-C0EA-43D7-B4AC-5804144386CA}"
   .WorkCodeCategory.Id = "{AA9C83C1-6E1E-4974-9DEC-7C554CC429D2}"
   .WorkCode.Id = "{8A4BEED2-A035-4407-948D-248681472747}"
End With

iRet = oReqTime.Add(sId)
```

Related information

"ApiRequestTime" on page 558

ApiRequestTime: ApproveOrReject

```
Public Function ApproveOrReject(ByVal sResourceId As String, ByVal sRequestId
As String, ByVal daStartDate As Date, ByVal daEndDate As Date, ByVal bAction
As Boolean, Optional ByVal sReason As String = "") As Int32
```

Purpose

Approve or reject all request time records which StartTime falls between the dates specified in the daStartDate and daEndDate parameters.

Parameters

Parameter (*=required)	Description
*sResourceId	The ID of the resource.
*sRequestId	The ID of the request.
*daStartDate	Start date of the period.
*daEndDate	End date of the period.
*bAction	• 0 for rejection. • 1 for approval.
sReason	Rejection reason. • Required when bAction = 0 (False) • Optional when bAction = 1 (True)

Returns

0 = Success

Nonzero = Error

Remarks

Either sResourceId or sRequestId can be an empty string:

- Approve or reject request time for a specified resource in sResourceId when sRequestId is an empty string.

- Approve or reject request time for a specified request in sRequestId when sResourceId is an empty string.

Check log file upon error.

Example

```
Dim oReqTime As New ApiRequestTime
Dim iRet As Int32

oReqTime.CPConnection = myCon
iRet = oReqTime. ApproveOrReject ("{BAA411CF-F648-47CD-A4F1-12C259EE25BF}", "
{FF72598D-718D-4BCA-AAEC-152564EBD446}","5/14/2007","5/18/2007 ", 0, "Please
use other request. ")
```

Related information

"ApiRequestTime" on page 558

ApiRequestTime: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, Optional ByRef sId As String
="")As Int32
```

Purpose

Create a request time using an XML string of the request time object in Changepoint.

Parameters

Parameter (*=required)	Description
*sXML	A new request time XML string.
sId	ByRef parameter that returns the new RequestTimeId after the request time has been added.

Returns

0 = Success

Nonzero = Error

Remarks

The ApiRequestTime XML structure can be obtained by the GetXMLStructure or the GetByXML methods.

The ByPassMetadataCheck switch will stop any meta data validation in RequestTime.

Example

Not available

Related information

"ApiRequestTime" on page 558

"ApiRequestTime XML" on page 576

ApiRequestTime: Delete

```
Public Overrides Function Delete(Optional ByVal sId As String = "") As Int32
```

Purpose

Delete a request time record.

Parameters

Parameter (*=required)	Description
sId	The ID of the request time record.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, property RequestTimeId is used.

Check log file upon error.

Example

```
Dim oReqTime As New ApiRequestTime
```

```
Dim iRet As Int32

oReqTime.CPConnection = myCon
oReqTime.RequestTimeId = "{FF636BB5-B108-4742-BA1D-E7D7BA9CDF3D}"
iRet = oReqTime.Delete()
```

Related information

"ApiRequestTime" on page 558

ApiRequestTime: Exists

```
Public Overrides Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Check whether this request time exists or not

Parameters

Parameter (*=required)	Description
sId	The ID of the request time record.

Returns

True if the request time exists, else False.

Remarks

If sId is not provided, property RequestTimeId is used.

Check log file upon error

Example

```
Dim oReqTime As New ApiRequestTime
Dim bRet As Boolean

oReqTime.CPConnection = myCon
oReqTime.RequestTimeId = "{FF636BB5-B108-4742-BA1D-E7D7BA9CDF3D}"
bRet = oReqTime.Exists()
```

Related information

"ApiRequestTime" on page 558

ApiRequestTime: GetById

```
Public Overrides Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Retrieve a request time record.

Parameters

Parameter (*=required)	Description
sId	The ID of the request time record.

Returns

0 = Success

Nonzero = Error

Remarks

Populate the properties of the object

If sId is not provided, property RequestTimeId is used.

Check log file upon error

Example

```
Dim oReqTime As New ApiRequestTime
Dim iRet As Int32

oReqTime.CPConnection = myCon
oReqTime.RequestTimeId = "{FF636BB5-B108-4742-BA1D-E7D7BA9CDF3D}"
iRet = oReqTime. GetById ()
```

Related information

"ApiRequestTime" on page 558

ApiRequestTime: GetByRequest

```
Public Function GetByRequest(ByVal sRequestId As String, ByVal daStartDate As
Date, ByVal daEndDate As Date) As DataSet
```

Purpose

Retrieve all request time records which StartTime falls between the dates specified in the daStartDate and daEndDate parameters for the target request.

Parameters

Parameter (*=required)	Description
*sRequestId	The ID of the target request
*daStartDate	Start date of the period
*daEndDate	End date of the period

Returns

A dataset of RequestTime records upon success and nothing in a dataset upon error.

Remarks

Check log file upon error

Example

```
Dim oReqTime As New ApiRequestTime
Dim dsRet As DataSet

oReqTime.CPConnection = myCon
dsRet = oReqTime.GetByRequest ("{FF72598D-718D-4BCA-AAEC-152564EBD446}",
"5/14/2007 9:00:00 AM","5/18/2007 5:00:00 PM")
```

Related information

"ApiRequestTime" on page 558

ApiRequestTime: GetByRes

```
Public Function GetByRes(ByVal sResourceId As String, ByVal daStartDate As
Date, ByVal daEndDate As Date) As DataSet
```

Purpose

Retrieve all the request time records, where the start time falls between the dates specified in the daStartDate and daEndDate parameters for the target resource.

Parameters

Parameter (*=required)	Description
*sResourceId	ID of the target resource.
*daStartDate	Start date of the period.
*daEndDate	End date of the period

Returns

A dataset of RequestTime records upon success and nothing in a dataset upon error.

Remarks

Check log file upon error

Example

```
Dim oReqTime As New ApiRequestTime
Dim dsRet As DataSet

oReqTime.CPConnection = myCon
dsRet = oReqTime.GetByRes ("{BAA411CF-F648-47CD-A4F1-12C259EE25BF}",
"5/14/2007 9:00:00 AM","5/18/2007 5:00:00 PM")
```

Related information

"ApiRequestTime" on page 558

ApiRequestTime: GetByXML

```
Public Function GetByXML(Optional ByVal sXML As String = "", Optional ByVal
sRequestTimeId As String = "") As String
```

Purpose

Takes the XML string passed in the sXML parameter and returns the string filled with data for the request time specified in the sRequestTimeId parameter

Parameters

Parameter (*=required)	Description
sXML	The XML string containing fields where data is required.
sRequestTimeId	The request time whose data is being retrieved.

Returns

An XML string.

Remarks

The XML string sXML can be varied in the number of fields. If only a small subset of data is required, any unwanted fields can be removed from sXML. The returned XML string will mirror sXML in the fields contained in the XML.

Example

Not available

Related information

"ApiRequestTime" on page 558

ApiRequestTime: GetIdsWithinPeriod

```
Public Function GetIdsWithinPeriod(ByVal daStartDate As Date, ByVal daEndDate
As Date) As DataSet
```

Purpose

Retrieve all the request time IDs from request time records, where the start time falls between the dates specified in the daStartDate and daEndDate parameters.

Parameters

Parameter (*=required)	Description
*daStartDate	Start date of the period.
*daEndDate	End date of the period.

Returns

A dataset of RequestTime IDs upon success and nothing in a dataset upon error.

Remarks

Check log file upon error

Example

```
Dim oReqTime As New ApiRequestTime
Dim dsRet As DataSet

oReqTime.CPConnection = myCon
dsRet = oReqTime.GetIdsWithinPeriod ("5/14/2007 9:00:00 AM","5/18/2007 5:00:00
PM")
```

Related information

"ApiRequestTime" on page 558

ApiRequestTime: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows As Int16 = -1) As
DataSet
```

Purpose

Retrieve request time records.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).

Returns

A dataset of RequestTime records upon success and nothing in a dataset upon error.

Returns all rows whenthe method is called without a parameter.

Columns returned are as follows:

RequestId, RequestNum, CustomerId, EngagementId, ResourceId, RName, RequestTimeId

Remarks

Check log file upon error

Example

```
Dim oReqTime As New ApiRequestTime
Dim dsRet As DataSet

oReqTime.CPConnection = myCon
dsRet = oReqTime.GetList ()
```

Related information

"ApiRequestTime" on page 558

ApiRequestTime: GetXMLStructure

```
Public Function GetXMLStructure()As String
```

Purpose

Retrieve the XML structure for the APIRequestTime object.

Parameters

None

Returns

An XML string of the request time.

Remarks

None

Example

Not available

Related information

"ApiRequestTime" on page 558

"ApiRequestTime XML" on page 576

ApiRequestTime: Submit

```
Public Function Submit(ByVal sResourceId As String, ByVal daStartDate As
DateTime, ByVal daEndDate As DateTime) As Int32
```

Purpose

Submit time, request time and standard time where the start time falls between the dates specified in the daStartDate and daEndDate parameters for a target resource.

Parameters

Parameter (*=required)	Description
*sResourceId	The ID of the target resource.
*daStartDate	Start date of the period
*daEndDate	End date of the period

Returns

0 = Success

Nonzero = Error

Remarks

Check log file upon error

Example

```
Dim oReqTime As New ApiRequestTime
Dim iRet As Int32

oReqTime.CPConnection = myCon
iRet = oReqTime.Submit ("{BAA411CF-F648-47CD-A4F1-12C259EE25BF}",
"5/14/2007","5/18/2007 ")
```

Related information

"ApiRequestTime" on page 558

ApiRequestTime: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Update a request time record.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

Ensure all mandatory properties are set.

Get the requesttime record which is intended to update by using GetById method.

Set HistoryComments property when FedAudit flag of the Contract is enabled.

Check log file upon error.

Example

```
Dim oReqTime As New ApiRequestTime
Dim iRet As Int32

oReqTime.CPConnection = myCon
With oReqTime
   .BookedByResource.Id = "{615FC0FC-1B7A-4519-A6BC-BCD74584339D}"
   .Description = "Bug verification - update"
   .HistoryComments = "Made the changes to RequestTime record."
   .RegularHours = 8
   .OvertimeHours = 2
   .Request.Id = "{505BDF7D-064C-4D07-A69E-112C0A14838D}"
   .RequestTimeId = "{C6FA1807-F81A-41CB-9567-474A5CC9044B}"
   .Resource.Id = "{BAA411CF-F648-47CD-A4F1-12C259EE25BF}"
   .StartTime = "05/14/2006 9:00:00 AM"
   .TimeCode1 = "{44B890C0-6AB9-47B7-A9F1-AF5D251DACF5}"
   .TimeCode2 = "{098D4E7E-AD20-47B7-AE0C-5B1E5D9A48EE}"
   .TimeCode3 = "{E33B0D35-904C-492A-879B-67CBDFCC9573}"
   .TimeZone = "EST"
   .UpdatedBy.ActionPerson.Id = "{615FC0FC-1B7A-4519-A6BC-BCD74584339D}"
   .WorkLocationGroup.Id = "{333A206C-513E-4CAE-841A-EA73A82E8E81}"
   .WorkLocation.Id = "{4F080CE6-C0EA-43D7-B4AC-5804144386CA}"
   .WorkCodeCategory.Id = "{AA9C83C1-6E1E-4974-9DEC-7C554CC429D2}"
```

```
   .WorkCode.Id = "{8A4BEED2-A035-4407-948D-248681472747}"
End With

iRet = oReqTime.Update()
```

Related information

"ApiRequestTime" on page 558

ApiRequestTime: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, Optional ByVal sId As String
= "") As Int32
```

Purpose

Update request time using an XML string of the request time object in Changepoint.

Parameters

Parameter (*=required)	Description
sXML	A request time XML string with updated fields.
sId	ID of the request time being updated.

Returns

0 = Success

Nonzero = Error

Remarks

The ApiRequestTime XML structure can be obtained by GetXMLStructure or GetByXML methods.

The update process will update all fields in Request time.

The ByPassMetadataCheck switch will stop any meta data validation in RequestTime.

It is recommended to remove the XML tags if you want to retain the original data in the database.

The method uses the following sequence to find the request time ID:

1. If the sRequestTimeId parameter is passed in, the method uses this value for the request time ID.
2. If this fails, the method attempts to extract the request time ID from <requesttimeid> in the XML.
3. If this fails, the request time ID is taken from the object properties.

Example

Not available

Related information

"ApiRequestTime" on page 558

"ApiRequestTime XML" on page 576

ApiRequestTime XML

```
<root>
  <requesttime>
    <bypassmetadatacheck />
    <createdby>
      <id />
      <name />
      <alternatename />
      <firstname />
      <lastname />
      <userdefinedid />
    </createdby>
    <createdon />
    <updatedby>
      <id />
      <name />
      <alternatename />
      <firstname />
      <lastname />
      <userdefinedid />
    </updatedby>
    <updatedon />
    <requesttimeid />
    <adjustedovertimehours />
    <adjustedregularhours />
    <adjustmentstatus>false</adjustmentstatus>
    <adjustmenttimeparent />
    <adjustmenttimestatus />
    <bookedbyresource>
      <id />
      <name />
      <alternatename />
      <firstname />
```

```
  <lastname />
  <userdefinedid />
</bookedbyresource>
<customer>
  <id />
  <name />
  <alternatename />
  <userdefinedid />
</customer>
<description />
<engagement>
  <id />
  <name />
  <alternatename />
  <userdefinedid />
</engagement>
<request>
  <id />
  <name />
  <alternatename />
  <userdefinedid />
</request>
<timezone>
  <id />
  <name />
  <alternatename />
  <userdefinedid />
</timezone>
<fedaudit>false</fedaudit>
<historycomments />
<overtimehours />
<regularhours />
<resource>
  <id />
  <name />
  <alternatename />
  <firstname />
  <lastname />
  <userdefinedid />
</resource>
<starttime />
<endtime />
<timecode1>
  <id />
  <name />
  <alternatename />
</timecode1>
<timecode2>
  <id />
  <name />
  <alternatename />
</timecode2>
<timecode3>
  <id />
  <name />
  <alternatename />
```

```
    </timecode3>
    <editable />
    <workcode>
      <id />
      <name />
      <alternatename />
      <userdefinedid />
    </workcode>
    <workcodecategory>
      <id />
      <name />
      <alternatename />
      <userdefinedid />
    </workcodecategory>
    <worklocation>
      <id />
      <name />
      <alternatename />
      <userdefinedid />
    </worklocation>
    <worklocationgroup>
      <id />
      <name />
      <alternatename />
      <userdefinedid />
    </worklocationgroup>
  </requesttime>
</root>
```

Comments

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34. The following special considerations apply to request node lookups:

- For the assignment node, the name needs to be provided if queueId is required; otherwise, the userdefinedid or firstname and lastname are required to find the assigned resourceid for the request.
- For the asset node, the assetnumber as userdefinedid or assettype and description as name are required to look up the id.
- For the parentrequest node, the requestnum as name is required to look up the id.

The XML for the ApiRequest object includes the XML for the ApiRequestDemand object and may contain XML for UDFs (configurable fields). For more information on the XML details for included objects, see the documentation for the included objects.

Example

Not available

Related information

"ApiRequestTime" on page 558

"UDF XML" on page 742

ApiResource

The ApiResource object allows users to create, retrieve, update and delete Changepoint users as well as general resource information such as, addresses, payroll information and billing rates. With the migration to .NET in version 12.0, the interface IResourceBase inherits IBusinessPropertyBase and the class ResourceClass implements IResourceBase.

The Resource object is composed of several objects: ApiResourceAddress, ApiResourcePayroll and a collection of ApiResourceRate objects.

Each of the subobjects can update data and retrieve current data into their properties. In some cases, such as ApiResourceRate, new rates can also be added. It is recommended that you use smaller subobjects when a specific change is only in a single area under the smaller object's control.

Namespace

Changepoint.ChangepointAPI2.ApiResource

Methods

Properties

Property (*=required)	Type	Description
Address	ApiResourceAddress	The ApiAddress object that contains address data for the resource.
AllowGaps	Boolean	A flag that allows resources that submit time daily to submit time in any order (for example, Monday, Wednesday, Tuesday, Thursday, Friday) and to skip days altogether.
AlternateName	String	Alternate name (for what may be a name in a foreign language).
ApiResourceRates	Array of ResourceRate objects	Returns all the rates held in the Resource object.
*BaseCurrency	String(3)	Three-letter ISO currency code for the base currency of the resource.

Property (*=required)	Type	Description
BypassMetadataCheck	CPMetadataCheck	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). For resources, the value set for this flag will be applied to all inner objects on Update and Add. Value range: • CPMetadataCheck .CheckAll = 0 • CPMetadataCheck .BypassAll = 1 • CPMetadataCheck .OnlyMandatory = 2
CostCenter	Identity	Identifier of the cost center associated with the resource.
CPConnection	ApiConnection	Write-only. Sets the database connection for the object.
CreatedBy	Signature	Contains details about who created the resource and when the resource was created
DelegateExpenseApprover	Identity	Identifier of the resource that the current resource delegated their expense approval rights to
DelegateTimeApprover	Identity	Identifier of the resource that the current resource delegated their time approval rights to
Deleted	Boolean	Indicates whether the Resource has been deleted
Description	String(variable length)	Description

Property (*=required)	Type	Description
EmployeeType	String(3)	FT (full time), PT (part time), CO (contractor), TE (temporary)
Entity	CPEntity	Read-only. The Changepoint entity the object represents.
ExemptFromResRequest	Boolean	True if this resource is exempt from resource requests
ExpenseApprovalDelegated	Boolean	True if resource has delegated their expense approval rights
ExpenseApprover	Identity	Identifier of the expense approver
*FirstName	String(50)	First name. When a value is stored, the Name property is also updated.
GlobalWorkgroup	Identity	Identifier of the global workgroup to which the resource belongs
HireDate	Date	The date the resource was hired.
IndustryExperience	Long	Number of years in an industry
InOut	Boolean	When set to true, indicates that the resource is in the office.
InOutMsg	String(255)	Message entered by resource to display when they are out of the office.
Language	String	Primary language of the resource

Property (*=required)	Type	Description
*LastName	String(50)	Last name. When a value is stored the Name property is also updated.
License	Long	Indicates whether the resource is licensed or not. 1 – Means licensed. 0 – Means no license
Licensed	Boolean	Licensed Enterprise user or not
MailURL	String(255)	Primary mail URL of the resource.
MiddleName	String(50)	The resource's middle name
mVersion	String	Read-only. The current version number for internal version control.
Name	String(50)	Resource full name
NTUserId	String(255)	NT domain ID
OrganizationalInfoEffectiveDate	Date	Date on which the manager to whom the resource will report comes into effect.
Payroll	ApiResourcePayroll	ApiResourcePayroll object.
PrimaryFuncEffectiveDate	Date	The date (without the time portion) when the primary function comes into effect for the Resource.
PrimaryFunction	Identity	Identifier of the resource's primary function.

Property (*=required)	Type	Description
ReportsTo	Identity	Identifier of the manager to which the resource reports
ReportsToEffectiveDate		Renamed in Changepoint 2014: OrganizationalInfoEffectiveDate
ResourceId	String	Changepoint ResourceId. This is the primary identifier for the resource. The resource object generates the resourceid if a create function (Add or CreateByXml) is called. Mandatory only if an update function (Update or UpdateByXml) is called.
ResourceRoles	Generic.List(Of String)	To set the resources roles to be saved by Add()/Update() functions
ResourceFeatures	Generic.List(Of String)	To set the resources features to be saved by Add()/Update() functions
ResourceType	String(3)	Enterprise user (E), Undefined resource(N)

Property (*=required)	Type	Description
Signature	String	Signature **Note:** Changepoint business objects do not allow for impersonation. They automatically assign the currently logged-in user as the CreatedBy/UpdatedBy when creating/updating the object. This behavior is consistent across all Changepoint business objects.
SQLUserId	String	SQL login ID
sxmlUDF	String	The UDF string in XML format
TerminationDate	Date	The date on which the resource is terminated. The API does not handle errors from the GUI, for example if resource is a time approver and can not be terminated. **Note:** The Add method does not set the TerminationDate when creating a resource. To set the TerminationDate when you create a resource, call the Update method immediately after the Add method.
TimeApprover	Identity	ID of the resource who is the time approver for the current resource
TimeApprovalDelegated	Boolean	Indicates whether or not the resource has delegated their time approval rights

Property (*=required)	Type	Description
TimeZone	Identity	The resource's time zone. Required for Changepoint Outlook Calendar synchronization, which supports two-way synchronization between a resource's Changepoint Calendar and Outlook Calendar.
Title	String	Title
TitleEffectiveDate	Date	The date a title change will come into effect.
UpdatedBy	Signature	The person who last updated the data. The date updated is also available from the Signature object.
UserDefinedResourceId	String	Id defined by the creator of the resource.
WebPassword	String	Web Password for the resource. This property is used to change passwords by setting the property. If someone attempts to retrieve the value, an empty string is returned.
Workgroup	Identity	Identifier of the workgroup to which the resource belongs.

Related information

"ApiResource XML" on page 588

"ApiResourceAddress" on page 631

"ApiResourcePayroll" on page 639

"ApiResourceRate" on page 648

ApiResource XML


```
<root>
<resource>
   <bypassmetadatacheck />
   <createdbyid />
   <createdon />
   <updatedbyid />
   <updatedon />
   <resourceid />
   <resourcetype />
   <firstname />
   <middlename />
   <lastname />
   <description />
   <title />
   <titleeffectivedate />
   <userdefinedresourceid />
   <alternatename />
   <sqluserid />
   <ntuserid />
   <licensed>false</licensed>
   <license />
   <employeetype>ft</employeetype>
   <signature />
   <webpassword />
   <allowgaps>true</allowgaps>
   <mailurl />
   <primaryfunction>
      <id />
      <name />
      <alternatename />
   </primaryfunction>
   <primaryfunceffectivedate />
   <globalworkgroup>
      <id />
      <name />
      <alternatename />
   </globalworkgroup>
   <workgroup>
      <id />
      <name />
      <alternatename />
   </workgroup>
   <language />
   <basecurrency>CAD</basecurrency>
   <industryexperience>0</industryexperience>
   <hiredate />
```

```
<terminationdate />
<costcenter>
   <id />
   <name />
   <alternatename />
</costcenter>
<timeapprover>
   <id />
   <firstname />
   <lastname />
   <alternatename />
   <userdefinedid />
</timeapprover>
<delegatetimeapprover>
   <id />
   <firstname />
   <lastname />
   <alternatename />
   <userdefinedid />
</delegatetimeapprover>
<timeapprovaldelegated>false</timeapprovaldelegated>
<expenseapprover>
   <id />
   <firstname />
   <lastname />
   <alternatename />
   <userdefinedid />
</expenseapprover>
<delegateexpenseapprover>
   <id />
   <firstname />
   <lastname />
   <alternatename />
   <userdefinedid />
</delegateexpenseapprover>
<expenseapprovaldelegated>false</expenseapprovaldelegated>
<reportsto>
   <id />
   <firstname />
   <lastname />
   <alternatename />
   <userdefinedid />
</reportsto>
<organizationalinfoeffectivedate />
<address>
...
</address>
<payroll>
```

```
   ...
   </payroll>
   <resourcerates>
   ...
   </resourcerates>
   <udf />
   <inout>false</inout>
   <inoutmsg />
   <timezone>
      <id />
      <name />
      <alternatename />
   </timezone>
   <exemptfromresrequest>false</exemptfromresrequest>
   <roles>
     <role>
       <id/>
       <name/>
       <alternatename/>
     </role>
   </roles>
   <features>
     <feature>
       <id/>
       <name/>
       <alternatename/>
     </feature>
   </features>
</resource>
</root>
```

Comments

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

The XML for the ApiResource object includes the XML for the ApiResourceAddress, ApiResourcePayroll, and ApiResourceRate objects and may contain XML for UDFs (configurable fields). For more information on the XML details for included objects, see the documentation for the included objects.

Example

```
<root>
<resource>
   <bypassmetadatacheck>0</bypassmetadatacheck>
   <createdbyid />
```

```
<createdon />
<updatedbyid />
<updatedon />
<resourceid />
<resourcetype>E</resourcetype>
<firstname>John</firstname>
<middlename />
<lastname>Smith</lastname>
<description />
<title />
<titleeffectivedate />
<userdefinedresourceid>RES000003</userdefinedresourceid>
<alternatename />
<sqluserid />
<ntuserid />
<licensed>false</licensed>
<license />
<employeetype>FT</employeetype>
<signature />
<webpassword>js1234567</webpassword>
<allowgaps>true</allowgaps>
<mailurl />
<primaryfunction>
   <id />
   <name>Management</name>
   <alternatename />
</primaryfunction>
<primaryfunceffectivedate />
<globalworkgroup>
   <id />
   <name>Product Development</name>
   <alternatename />
</globalworkgroup>
<workgroup>
   <id />
   <name>Enterprise Development</name>
   <alternatename />
</workgroup>
<language>E</language>
<basecurrency>CAD</basecurrency>
<industryexperience>0</industryexperience>
<hiredate>11/01/2009</hiredate>
<terminationdate />
<costcenter>
   <id />
   <name>Default Cost Center Test</name>
   <alternatename />
</costcenter>
```

```
<timeapprover>
   <id />
   <firstname>Susan</firstname>
   <lastname>Harding</lastname>
   <alternatename />
   <userdefinedid />
</timeapprover>
<delegatetimeapprover>
   <id />
   <firstname>Michael</firstname>
   <lastname>Park</lastname>
   <alternatename />
   <userdefinedid />
</delegatetimeapprover>
<timeapprovaldelegated>false</timeapprovaldelegated>
<expenseapprover>
   <id />
   <firstname>Susan</firstname>
   <lastname>Harding</lastname>
   <alternatename />
   <userdefinedid />
</expenseapprover>
<delegateexpenseapprover>
   <id />
   <firstname>Michael</firstname>
   <lastname>Park</lastname>
   <alternatename />
   <userdefinedid />
</delegateexpenseapprover>
<expenseapprovaldelegated>false</expenseapprovaldelegated>
<reportsto>
   <id>{7E4F8D03-363B-11D4-8AB1-0001023D3221}</id>
   <firstname />
   <lastname />
   <alternatename />
   <userdefinedid />
</reportsto>
<reportstoeffectivedate />
<address>
...
</address>
<payroll>
...
</payroll>
<resourcerates>
...
</resourcerates>
<udf>
```

```
 <entity>Resource</entity>
<entityid />
<additionalid actual="" />
<resourceid>{6EE89511-89D3-4832-AB48-1F7C82C0477E}</resourceid>
<fields />
</udf>
<inout>false</inout>
<inoutmsg />
<timezone>
   <id>EST</id>
   <name />
   <alternatename />
</timezone>
<exemptfromresrequest>false</exemptfromresrequest>
<roles>
  <role>
    <id/>
    <name/>
    <alternatename/>
  </role>
</roles>
<features>
  <feature>
    <id/>
    <name/>
    <alternatename/>
  </feature>
</features>
<roles>
  <role>
    <id>3F2504E0-4F89-11D3-9A0C-0305E82C3301</id>
    <name>QA</name>
    <alternatename>Senior QA Analyst</alternatename>
  </role>
</roles>
<features>
  <feature>
    <id>EPJ</id>
    <name>Edit Projects</name>
    <alternatename></alternatename>
  </feature>
</features>
</resource>
</root>
```

Related information

"ApiResource" on page 579

"ApiResourceAddress" on page 631

"ApiResourcePayroll" on page 639

"ApiResourceRate" on page 648

ApiResource: Add

```
Public Overrides Function Add(Optional ByRef sId As String = "") As Integer
```

Purpose

Adds a new resource to Changepoint.

Parameters

Parameter (*=required)	Description
sId	ByRef parameter that will return the new ResourceId after the resource has been added.

Returns

0 = Success

Nonzero = Error

Remarks

Adds a Resource to Changepoint. Mandatory fields in order to add a new resource are the following fields:

- First Name
- Last Name
- Name (updated automatically when First, Middle and Last Name are set)
- BaseCurrency

Note: The Add method does not set the TerminationDate when creating a resource. To set the TerminationDate when you create a resource, call the Update method immediately after the Add method.

Error numbers are documented in the "Error Messages.txt" file.

The Resource object also contains Address and Payroll objects that can be accessed through Resource.Address and Resource.Payroll.

New rates can be added to the object using one of two SetRates methods.

Note: Assigning a new resource to a GlobalWorkgroup and Workgroup causes the resource to become active in Changepoint; otherwise, they are added with an unassigned state.

Example

```
Dim myResource as New ApiResource
Dim iRet as Int32 = 0
Dim sNewId as String = ""
Dim myNewRate as ApiResourceRate

myResource.CPConnection = myCon
With myResource
   .FirstName = "John"
   .LastName = "Smith"
   .BaseCurrency = "CAD"
   ...
   .Address.AddressLine = "131 Winding Lane"
   .Address.City = "Toronto"
   .Address.StateProvince = New Identity("{06810a99-77ba-41ce-91be-
2f8747ccc65f}")
   ...
   .Payroll.HoursPerDay = 8.0
   .Payroll.NonWorkingDay = "1000001"
   .Payroll.AnnualVacationHours = 240

End With
myNewRate = New ApiResourceRate
myNewRate = myCon
With myNewRate
   .Currency = "CAD"
   .EffectiveDate = Now.Date
   .HourlyBilling = 80.00
   .DailyBilling = 500.00
End With
'Place the new rate into the resource object
```

```
iRet = myResource.SetRates(myNewRate)
If iRet <> 0 then
   'Handle error
End If
'Save the resource
iRet = myResource.Add(sNewId)
Return iRet
```

Related information

"ApiResource" on page 579

ApiResource: CanUnassign

```
Public Function CanUnassign(ByVal sResourceId as String, ByVal dEffDate as
Date) As Int32
```

Purpose

Checks whether the specified resource can be unassigned successfully.

Parameters

Parameter (*=required)	Description
*sResourceId	The ResourceId of the resource to be unassigned.
*dEffDate	The effective date the unassignment will take place

Returns

0 = Success

Nonzero = Cannot unassign; check the list of errors for the specific reason.

Remarks

The deletion process first unassigns the resource before deleting. Note that deletions are soft deletions, where the deleted flag is 1.

The process of unassigning a resource can result in errors if the resource has already been deleted.

Example

```
Dim myResource as New ApiResource
```

```
Dim iRet as Int32 = 0
Dim sResId as String = ""

sResId = "{5B401107-53D3-4328-8369-3F6F2BDAA86C}"
'Set the connection to the database
myResource.CPConnection = myCon
'Check if the resource can be unassigned
iRet = myResource.CanUnassign (sResId, "05/25/2007")
'Unassign the resource
If iRet = 0 then
   iRet = myResource.UnassignResource(sResId, "05/25/2007", False)
End If
```

Related information

"ApiResource" on page 579

ApiResource: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, Optional ByRef sId As String
= "") As Int32
```

Purpose

Create a resource using an XML string of the Resource object in Changepoint.

Parameters

Parameter (*=required)	Description
*sXML	A new resource XML string.
sId	ByRef parameter that returns the new ResourceId after the resource has been added.

Returns

0 = Success

Nonzero = Error

Remarks

The ApiResource XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The ByPassMetadataCheck switch will stop any meta data validation in Resource and also in Resource UDFs.

For details of the use of UDF default values, see "UDF default values logic for CreateByXML" on page 746.

Example

```
Dim myResource As New ApiResource
Dim sMyXML As String = ""
Dim iRet As Int32 = 0
'Set the connection to the database
myResource.CPConnection = myCon
'Get Resource XML structure
sMyXML = myResource.GetXMLStructure()
'populate sMyXML with data and then pass it to CreateByXML
'
iRet = myResource.CreateByXML(sMyXML)
```

Related information

"ApiResource" on page 579

"ApiResource XML" on page 588

ApiResource: Delete

```
Public Overrides Function Delete(Optional ByVal sId as String = "") As Integer
```

Purpose

Deletes an existing resource from Changepoint.

Parameters

Parameter (*=required)	Description
sId	The ResourceId of the resource to be deleted.

Returns

0 = Success

Nonzero = Error

Remarks

The deletion process first unassigns the resource before deleting. Deletions are "soft" deletes, where the deleted flag is 1.

Example

```
Dim myResource as New ApiResource
Dim iRet as Int32 = 0
Dim sResId as String = ""

sResId = "{5B401107-53D3-4328-8369-3F6F2BDAA86C}"
'Set the connection to the database
myResource.CPConnection = myCon
iRet = myResource.Delete(sResId)
Return iRet
```

Related information

"ApiResource" on page 579

ApiResource: Exists

```
Public Overrides Function Exists(Optional ByVal sId as String = "") As Boolean
```

Purpose

Checks whether the resource exists in the database and has not been deleted.

Parameters

Parameter (*=required)	Description
sId	Resource needing confirmation whether they still exist in Changepoint.

Returns

True if the resource exists, else False.

Remarks

This method checks against the deleted flag of the resource, as resource records are soft deleted in Changepoint.

Example

```
Dim myResource as New ApiResource
Dim bRet as Boolean

'Set the connection to the database
myResource.CPConnection = myCon
'Check if the resource exists in Changepoint
bRet = myResource.Exists("{5B401107-53D3-4328-8369-3F6F2BDAA86C}")
```

Related information

"ApiResource" on page 579

ApiResource: GetAllByWorkgroup

```
Public Function GetAllByWorkgroup(Optional ByVal sWorkgroupId As String = "",
Optional Byval sWorkgroupName as String = "") As DataSet
```

Purpose

Retrieves all resources who belong to the specified sWorkgroupId or Workgroup identified by sWorkgroupName

Parameters

Parameter (*=required)	Description
sWorkgroupId	Workgroup ID.
sWorkgroupName	Name of the workgroup.

Returns

A dataset of resources. If sWorkgroupName is used, the WorkgroupId is returned as part of the dataset. Since there is a possibility of duplicate Workgroup names, the dataset includes (WorkgroupId, ResourceId, ResourceName).

When sWorkgroupId is passed, the returned dataset consists of (ResourceId, ResourceName).

Remarks

This method relies on the parameters passed in for Workgroup information; if the workgroup information is missing, the method falls back to the Workgroup of the resource object. If there is none it will return an error.

Example

```
Dim myResource as New ApiResource
Dim ds as New DataSet
'Set the connection to the database
myResource.CPConnection = myCon
'Get all the resources in the Workgroup MyWorkgroupName
ds = myResource.GetAllByWorkgroup("", "MyWorkgroupName")
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count > 0 then
   ...
      Else
      ...End If
```

Related information

"ApiResource" on page 579

ApiResource: GetAllNames

```
Public Function GetAllNames() As DataSet
```

Purpose

Retrieves the names of all resources not deleted from Changepoint.

Parameters

None

Returns

A dataset containing two columns (ResourceId, Name)

Remarks

None

Example

```
Dim myResource as New ApiResource
Dim ds as DataSet
```

```
'Set the connection to the database
myResource.CPConnection = myCon

ds = myResource.GetAllNames()
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count > 0 Then
'Iterate through the dataset as necessary
End If
```

Related information

"ApiResource" on page 579

ApiResource: GetById

```
Public Overrides Function GetById(Optional ByVal sId as String = "") As
Integer
```

Purpose

Retrieves resource, resource address, resource payroll, rate data into the object specified by ResourceId.

Parameters

Parameter (*=required)	Description
sId	ResourceId of the resource whose data is to be retrieved.

Returns

0 = Success

Nonzero = Error

Remarks

This method fills the object with current data from the database. In the Resource object this method also fills the inner Address and Payroll objects along with existing Rate data.

The WebPassword cannot be retrieved, it can only set during an update.

Example

```
Dim myResource as New ApiResource
Dim iRet as Integer
Dim sResourceId as String = ""
```

```
Dim sFirstName as String = ""
Dim sLastName as String = ""
Dim sName as String = ""

'Set the connection to the database
myResource.CPConnection = myCon
'Retrieve current resource data into the object
iRet = myResource.GetById("{5B401107-53D3-4328-8369-3F6F2BDAA86C}")
'Extract resource data
If iRet = 0 then
   With myResource
      sResourceId = .ResourceId
      sFirstName = .FirstName
      sLastName = .LastName
      sName = .Name
      ...
   End With
Else
   Return iRet
End If
```

Related information

"ApiResource" on page 579

"ApiResource" on page 579

ApiResource: GetByXML

```
Public Function GetByXML(ByVal sXML As String, Optional ByVal sResourceId As
String = "") As String
```

Purpose

Takes the XML string passed in (sXML) and returns the string filled with data for the specified ResourceId (sResourceId)

Parameters

Parameter (*=required)	Description
*sXML	XML string of the Resource object, with any fields of no interest removed.
sResourceId	The Login of the resource in question. If no ResourceId is passed in, the XML string (sXML) is examined if no ResourceId in sXML then the object's ResourceId is selected. If there still is no valid ResourceId, the method returns an error.

Returns

An XML string mirroring sXML with data inserted or the entire XML of the Resource object including data.

Remarks

If sXML = "" then all data for the resource will be returned in the XML string.

If no ResourceId is specified (sResourceId), the Resource object is examined and if there is a ResourceId then data will be returned for that ResourceId value.

The current implementation of this method does not support extracting specific rates, if the resourcerates tag is included, all rates will be included and if the resourcerates tag is excluded, all rates will be excluded. The same concept applies to UDF data, either all UDFs for the current resource are returned or none are returned.

Example

```
Dim myResource as New ApiResource
Dim sMyXML as String = ""
'For the variable sMyXML values see ApiResource XML
Dim sRetXML as String = ""
Dim iRet as Integer = 0

'Set the connection to the database
myResource.CPConnection = myCon

'Get the Resource data as an XML string
sRetXML = myResource.GetByXML (sMyXML, "{7C8907BA-EDD5-401A-919E-
FACF4261ABD3}")

Return sRetXML
```

Related information

"ApiResource" on page 579

ApiResource: GetDefaultEffDate

```
Public Function GetDefaultEffDate(ByVal sResourceId As String) As Date
```

Purpose

Returns the default effective date that is used by the system for setting effective date values.

Parameters

Parameter (*=required)	Description
*sResourceId	The resource whose default effective date is to be retrieved.

Returns

The default effective date.

Remarks

None

Example

```
Dim myResource as New ApiResource
Dim myDefaultDate as Date

'Set the connection to the database
myResource.CPConnection = myCon
myDefaultDate = myResource.GetDefaultEffDate("{7C8907BA-EDD5-401A-919E-
FACF4261ABD3}")
```

Related information

"ApiResource" on page 579

ApiResource: GetFullName

```
Public Function GetFullName(ByVal sFirstName As String, ByVal sMiddleName As
String, ByVal sLastName As String) As String
```

Purpose

Returns a formatted resource name based on the current Changepoint system name format settings. If the settings are f m l, the return name will follow FirstName MiddleName LastName format.

Parameters

Parameter (*=required)	Description
*sFirstName	The first name of the resource
*sMiddleName	The middle name of the resource
*sLastName	The last name of the resource.

Returns

The resource name as a string

Remarks

None

Example

```
Dim myResource as New ApiResource
Dim myResourceName as String = ""

'Set the connection to the database
myResource.CPConnection = myCon
myResourceName = myResource.GetFullName ("John", "D","Smith")
```

Related information

"ApiResource" on page 579

ApiResource: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As String
```

Purpose

Returns the ResourceId based on the UDF Text field and value.

Parameters

Parameter (*=required)	Description
*sUDFField	UDF text field name (for example, Text1)
*sUDFValue	UDF text value

Returns

ResourceId, or an empty string if nothing is found.

Remarks

None

Related information

"ApiResource" on page 579

ApiResource: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows as Short = -1) As
DataSet
```

Purpose

Returns a list of some or all resources in Changepoint who are not deleted in the system.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).

Returns

A dataset with data or an empty dataset if nothing is found

Remarks

The dataset is made up of two fields: ResourceId and Name

Example

```
Dim myResource as New ApiResource
Dim ds as DataSet
Dim sResourceId as String = ""

'Set the connection to the database
myResource.CPConnection = myCon
'Get all resources
ds = myResource.GetList(-1)
'Iterate through the dataset
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count > 0 then
   For each mRow As DataRow in ds.Tables(0).Rows
      If mRow.Item("Name").ToString = "John Smith" Then
         sResourceId = mRow.Item("ResourceId")
         Exit For
      End If
   Next mRow
Else
   Return sResourceId
End If
```

Related information

"ApiResource" on page 579

ApiResource: GetRate

```
Public Function GetRate(Optional ByVal sResourceRateId As String = "",
Optional ByVal sResourceId As String = "") As String
```

Purpose

Returns a rate as an XML string based on sResourceRateId and sResourceId.

Parameters

Parameter (*=required)	Description
sResourceRateId	The ID of the resource rate being extracted.
sResourceId	The ResourceId of the resource whose rate is being extracted.

Returns

An XML string of the resource rate. If no rate is found, an XML string is returned with no values in the fields.

Remarks

If no ResourceId is specified (sResourceId) then the object's ResourceId value is used. If there is none, data is extracted based on sResourceRateId alone.

When there is a ResourceId specified in sResourceId the provided rate is checked for validity against the rates for the Resource in sResourceId.

If only the ResourceRateId is used, it will return the value of the related rate for the resource.

Example

```
Dim myResource as New ApiResource
Dim sResourceName as String = "John Q Smith"
Dim sResId as String = ""
Dim sResRateId as String = "{7143D807-D450-490F-AD20-629C43D54266}"
Dim sMyRateXml as String = ""
Dim ds as New DataSet

'Set the connection to the database
myResource.CPConnection = myCon
'Get the ResourceId, as we have none to link to the
ds = myResource.GetResourceIdsByName(sResourceName, "111")
'Note, for this example there is the assumption the name is unique therefore
only a 'single record is returned.
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count = 1 Then
   sResId = ds.Tables(0).Rows(0).Item("ResourceId").ToString
End If
if sResId <> "" Then
   sMyRateXml = myResource.GetRate (sResRateId, sResId)
End If
```

Related information

"ApiResource" on page 579

ApiResource: GetResourceIdsByName

```
Public Function GetResourceIdsByName(ByVal ResourceName As String, Optional
Byval sNameFormat as String = "111") As DataSet
```

Purpose

Return all resources that match the criteria passed in and have not been deleted in Changepoint.

Parameters

Parameter (*=required)	Description
*ResourceName	The name of a resource
sNameFormat	Denotes the format of ResourceName. The default is "111" meaning the first, middle and last names are present in the parameter ResourceName. The format for first and last name only is "101", which indicates that the name string consists of a first name and a last name with no middle name.

Returns

A dataset with a single column (ResourceId)

Remarks

Note that duplicate records are possible, as names are not unique.

Example

```
Dim myResource as New ApiResource
Dim ds as New DataSet

'Set the connection to the database
myResource.CPConnection = myCon
ds = myResource.GetResourceIdsByName("John Q Smith", "111")
```

Related information

"ApiResource" on page 579

ApiResource: GetResourcesByUserDefinedId

```
Public Function GetResourcesByUserDefinedId(ByVal sUserDefinedId As String) As
DataSet
```

Purpose

Returns a dataset of all resources that match the UserDefinedId as specified in the parameter sUserDefinedId and have not been deleted in Changepoint.

Parameters

Parameter (*=required)	Description
*sUserDefinedId	The UserDefinedId of the resource.

Returns

A dataset with two columns (ResourceId, Name)

Remarks

Note that multiple records are possible as the UserDefinedId is not guaranteed to be unique in Changepoint.

Example

```
Dim myResource as New ApiResource
Dim ds as New DataSet

'Set the connection to the database
myResource.CPConnection = myCon
ds = myResource.GetResourcesByUserDefinedId("Resource99")
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count > 0 Then
' Iterate through the dataset, until desired data is found
...
End If
```

Related information

"ApiResource" on page 579

ApiResource: GetResTypes

```
Public Function GetResTypes() As DataSet
```

Purpose

Returns all Resource types in Changepoint

Parameters

None

Returns

A dataset with two columns (Code, Description)

Remarks

In the returned dataset, the "Code" field will be a three character code and is the unique identifier for the Resource type

Example

```
Dim myResource as New ApiResource
Dim ds as New DataSet

'Set the connection to the database
myResource.CPConnection = myCon
ds = myResource.GetResourcesByUserDefinedId("Resource99")
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count > 0 Then
' Iterate through the dataset, until desired data is found
...
End If
```

Related information

"ApiResource" on page 579

ApiResource: GetUDF

```
Public Function GetUDF(ByVal sWorkgroupId as String, ByVal sActionResourceId
as String, Optional ByVal sResourceId as String = "", Optional ByVal UDFOption
As CPUDFReturnType = CPUDFReturnType.OnlyValues) As String
```

Purpose

Returns UDF (configurable field) data for a specified resource as an XML string.

Parameters

Parameter (*=required)	Description
*sWorkgroupId	The workgroup of the resource
*sActionResourceId	The resource querying for information
sResourceId	The resource whose UDF data is required
UDFOption	Determines how the UDF data is to be returned.

Returns

An XML string of UDFs

Remarks

CPUDFReturnType Value	Description
OnlyStructure	Returns only UDF XML structure, no field data
OnlyStructureAndOptions	Returns UDF structure and option data without field data
OnlyValues	Returns only UDF fields with data, do not include any field structure or options
WithStructure	Returns UDF field data with full field structure
WithStructureAndOptions	Returns UDF field data with full structure and all field options

Example

```
Dim myResource as New ApiResource
Dim sRetUDF as String
Dim sWorkgroup as String = ""
Dim sResId as String = ""

sWorkgroup = "{79d28dde-d4f7-42c9-800b-cbfaf527259c}"
sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
'Set the connection to the database
myResource.CPConnection = myCon
'Get the UDF XML string
sRetUDF = myResource.GetUDF(sWorkgroup, myCon.ChangepointUserId, sResId,
CPUDFReturnType.OnlyValues)
```

```
Return sRetUDF
```

Related information

"ApiResource" on page 579

ApiResource: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal sResourceId as String, ByVal sCodeName
as String, Optional ByVal sSearchString as String = "") As String
```

Purpose

Returns UDF (configurable field) data for a resource as an XML string.

Parameters

Parameter (*=required)	Description
*sCodeName	The name of the UDF code. Must be of the form "Code#", for example, "Code10".
sSearchString	Search string. The search string will match all the options in sCodeName and return all options containing sSearchString in the option text.

Returns

An XML string of the option list that exists for a specified code.

Remarks

The search string matches all the options in sCodeName and returns all options containing sSearchString in the option text.

This method uses the login User ResourceId to extract the UDFCode options. Although GetById must be called to retrieve current data into the object, the ResourceId passed into GetById is not used to retrieve UDF Code options.

Example

```
Dim myResource as New ApiResource
Dim sRetUDFOptions as String
Dim sCodeName as String = ""
Dim sSearchStr as String = ""
```

```
sCodeName = "Code1"
'Set the connection to the database
myResource.CPConnection = myCon
'Get the UDF Code options as an XML string
sRetUDFOptions = myResource.GetUDFCodeOptions(sResourceId, sCodeName,
sSearchStr)
Return sRetUDFOptions
```

Related information

"ApiResource" on page 579

ApiResource: GetWorkgroupInfo

```
Public Function GetWorkgroupInfo(ByVal sId as String) As DataSet
```

Purpose

Retrieves the current Globalworkgroup and Workgroup that the specified resource (sId) is related to

Parameters

Parameter (*=required)	Description
*sId	The Resource whose workgroup information is required. The method will throw an error if a valid GUID string is not passed in the parameter.

Returns

A dataset with GlobalWorkgroup and Workgroup information. The dataset contains the columns GlobalWorkgroupId, GlobalWorkgroup, WorkgroupId, Workgroup

Remarks

This method relies solely on the value in sId for the ResourceId, if an empty string or invalid GUID string is passed, an error is raised.

Example

```
Dim myResource as New ApiResource
Dim ds as DataSet
Dim sResId as String = ""
```

```
sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
'Set the connection to the database
myResource.CPConnection = myCon
ds = myResource.GetWorkgroupInfo (sResId)
Return ds
```

Related information

"ApiResource" on page 579

ApiResource: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Return the XML structure of the ApiResource object

Parameters

None

Returns

An XML string of the ApiResource object.

Remarks

Some fields in the structure will have defaulted data, otherwise fields are empty.

Example

```
Dim myResource as New ApiResource
Dim sStructure as String = ""

sStructure = myResource.GetXMLStructure
```

Related information

"ApiResource" on page 579

"ApiResource XML" on page 588

ApiResource: SaveUDF

```
Public Overrides Function SaveUDF(Optional ByVal sXML as String = "", ByVal
bypassMetadata as CPMetadataCheck) As Int32
```

Purpose

Updates the resource UDF data based on the parameter sXML.

Parameters

Parameter (*=required)	Description
sXML	UDF string in XML format.
bypassMetadata	Determines the level of metadata checking for sXMLUDF. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

0 = Success

Nonzero = Error

Remarks

The ResourceId is first taken from the object. If there are no values, then the sXML is accessed for these values. An error is thrown if the RrsourceId still is not present. The ResourceId is a required value for the save process.

If sUDF is not provided, take the value of property sxmlUDF.

It is recommended to call the GetUDF method to obtain the correct UDF XML format.

Example

Not available

Related information

"ApiResource" on page 579

"ApiResource: GetUDF" on page 612

"UDF XML" on page 742

ApiResource: SetPropertiesByXML

```
Public Function SetPropertiesByXML(ByVal sXML as String, Optional ByVal
bNotInitialize as Boolean = True) As Int32
```

Purpose

Sets the properties of the object via an XML string

Parameters

Parameter (*=required)	Description
*sXML	XML string of the Resource object with data to load into the object
bNotInitialize	Switch that initializes the object. Normal use is to set as "True"

Returns

0 = Success

Nonzero = Error

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods.

The XML string can be the entire resource object with all fields filled or only partially.

Empty fields in the XML string will result in the object's matching field being overwritten with an empty value (that is, the field will be cleared). Therefore if existing data in the object is to be retained, remove the field from the XML string.

At a minimum the ResourceId and ByPassMetaDataCheck fields should be present in the XML string

When called from an empty object, the method will first retrieve current data into the object before applying the new field values as held in sXML.

In the case where properties in the object have been altered prior to SetPropertiesByXML being called, the method will try to preserve those values when retrieving base data.

It is best to load an empty object with data first before applying any changes.

Example

```
Dim myResource as New ApiResource
Dim iRet as Int32 = 0
Dim sResId as String = ""

sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
'Set the connection to the database
myResource.CPConnection = myCon
'Get current data for the Resource, or avoid and let SetPropertiesByXML load
the data
iRet = myResource.GetById(sResId)
'Load the object with any changes, refer to ApiResource XML for XML that can
be 'used in sUpdateXML
iRet = myResource.SetPropertiesByXML(sUpdateXML, True)
If iRet = 0 Then
   iRet = myResource.Update
End If
```

Related information

"ApiResource" on page 579

"ApiResource XML" on page 588

"ApiResource: CreateByXML" on page 597

"ApiResource: UpdateByXML" on page 623

ApiResource: SetRates(oRate)

```
Public Function SetRates(ByVal oRate as ApiResourceRate) As Int32
```

Purpose

Loads the resource object with a new rate or replace an existing rate within the object

Parameters

Parameter (*=required)	Description
*oRate	An ApiResourceRate object

Returns

0 = Success

Nonzero = Error

Remarks

For new rates, always set the Resource property of the rate to the resource object's ResourceId to link the rate to the resource. There should be no value in the ResourceRateId field as that will be created when the data is updated.

In the case of an existing rate that is being updated, ensure the ResourceRateId is correct as the method checks against current data whether the rate belongs to the specified resource.

If the only change to the resource is to a rate it may be easier to do the Add or Update from the ApiResourceRate object as opposed to the ApiResource object.

Example

```
Dim myRate as New ApiResourceRate
Dim myResource as New ApiResource
Dim iRet as Int32 = 0
Dim sResId as String = ""

sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
'Set the connection to the database
myResource.CPConnection = myCon
'Set the Rate object with data
With myRate
   'Set the connection to the database for the rate object
   .CPConnection = myCon
   .Resource = New Identity(sResId)
   .Currency = "CAD"
   .DailyBilling = 500.00
   .HourlyBilling = 60.00
   .EffectiveDate = "07/15/2007"
End With
'Get existing data for the resource, including any rates
iRet = myResource.GetById(sResId)
'Move the new rate into the object
If iRet = 0 Then
   iRet = myResource.SetRates_2(myRate)
End If
'Update the resource data
If iRet = 0 Then
   iRet = myResource.Update
End if
```

Related information

"ApiResource" on page 579

ApiResource: SetRates(sRatesXML)

```
Public Function SetRates(ByVal sRatesXML As String) As Int32
```

Purpose

Loads the resource object with one or more rates, or updates a rate by replacing it with a rate with updated data.

Parameters

Parameter (*=required)	Description
*sRatesXML	XML string containing one or more ResourceRate object

Returns

0 = Success

Nonzero = Error

Remarks

Since XML is used, more than one rate can be added to the Resource object at one time.

New rates should not have any value in the <resourcerateId> field of the XML, but a value should be in the <resourceId> field. If the <resourcerateId> field does have a GUID string it will be matched to the current list of rates and if there's a match the old rate will be replaced with the new one.

If the ResourceRateId does not match it is added to the object but will not be written to the database as it likely does not belong to the resource.

The XML structure for the parameter is the same as in the ApiResource XML, namely `<resourcerates><rate></rate><rate></rate></resourcerates>`. For more information, see the "ApiResource XML" section on page 588 and "ApiResourceRate XML" on page 650.

Example

```
Dim myResource as New ApiResource
Dim sResId as String = ""
```

```
Dim iRet as Int32 = 0

sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
'Set the connection to the database
myResource.CPConnection = myCon
'Retrieve current data into the object
iRet = myResource.GetById(sResId)
If iRet = 0 Then
   'Load rate data into the object
   iRet = myResource.SetRates(sNewRatesXML)
End If

If iRet = 0 Then
   'Update the resource if necessary or continue editing
End If
```

Related information

"ApiResource" on page 579

ApiResource: UnassignResource

```
Public Function UnassignResource(ByVal sResourceId As String, Optional ByVal
dEffectiveDate As Date = _BASEDATE, Optional ByVal bCheckIfCanUnassign As
Boolean = True) As Int32
```

Purpose

Unassigns a specified resource in Changepoint

Parameters

Parameter (*=required)	Description
*sResourceId	The ID of the resource to be unassigned.
dEffectiveDate	The date on which the unassignment is to occur. *Default = 1/1/1753.
bCheckIfCanUnassign	Indicates whether to check if the resource can be unassigned.

Returns

0 = Success

Nonzero = Error

Remarks

Unassigning a resource is similar to deleting. This process will be halted if:

- the resource is actively involved in projects or tasks where time has been booked and/or billings have been booked to contracts that are active
- the resource is a project manager and there are active projects under their control
- there is a future change to the resource set to occur (such as a transfer or even a future unassignment)

Example

```
Dim myResource As New ApiResource
Dim sResId As String = ""
Dim iRet As Int32 = 0

sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
'Set the connection to the database
myResource.CPConnection = myCon
'Unassign the resource today, and also check if the resource can be unassigned
'before proceeding
iRet = myResource.UnassignResource(sResId, Now, True)
```

Related information

"ApiResource" on page 579

ApiResource: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, Optional ByVal sResourceId
As String = "") As Int32
```

Purpose

Updates a resource using an XML string containing new data

Parameters

Parameter (*=required)	Description
*sXML	The XML string of the resource object with new data contained in the fields.
sResourceId	Resource ID of the resource being updated.

Returns

0 = Success

Nonzero = Error

Remarks

Performs the same function as Update except any resource can be updated through this function. The XML sent in the parameter should be of the form in ApiResource XML. The ApiResource XML structure can be obtained by the GetXMLStructure or GetByXML methods.

Ensure that any fields that are not updated are removed from the XML string.

The update process will first get any current data before applying the updated data in the XML string.

It is also possible to add new rates as well as update existing ones. New rates should have empty <ResourceRateId> tags in the XML string

The method uses the following sequence to find the resource ID:

1. If the sResourceId parameter is passed in, the method uses this value for the resource ID.
2. If this fails, the method attempts to extract the resource ID from <resourceid> in the XML.
3. If this fails, the resource ID is taken from the object properties.
4. If this fails, an attempt is made to look up the resource ID using <userdefinedresourceid> in the XML. If <userdefinedresourceid> has a value, but the value is invalid or duplicated, then you will get an error and no further attempts are made to look up the resource ID.

5. If <userdefinedresourceid> is empty, an attempt is made to look up the resource ID using <firstname> and <lastname> in the XML.

Example

```
Dim myResource As New ApiResource
Dim sResId As String = String.Empty
Dim iRet As Int32 = 0

sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
'Set the connection to the database
myResource.CPConnection = myCon
'Get Resource XML structure with existing data if it's necessary.
sMyXML = myResource.GetByXML(sResId)
'modify sMyXML and then pass it to UpdateByXML
'
iRet = myResource.UpdateByXML(sMyXML)
```

Related information

"ApiResource" on page 579

Related information

"ApiResource XML" on page 588

ApiResource: UpdateLoginInfo

```
Public Function UpdateLoginInfo(ByVal sId As String, Optional ByVal sNtLogin
As String = "") As Int32
```

Purpose

Updates the basic login information for a resource

Parameters

Parameter (*=required)	Description
*sId	The ResourceId of the resource whose login information is being updated
sNtLogin	New NT Login as a string

Returns

0 = Success

Nonzero = Error

Remarks

Before changing the login data, the method does a search for other resources that have the same logins. If there are others, an error is raised.

Example

```
Dim myResource as New ApiResource
Dim sResId As String = ""

'Set the connection to the database
myResource.CPConnection = myCon
sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
iRet = myResource.UpdateLoginInfo (sResId, "JohnsLogin","JohnsNtL0g1n")
```

Related information

"ApiResource" on page 579

ApiResource: UpdateRates

```
Public Function UpdateRates(ByVal sRateXML as String) As Int32
```

Purpose

Updates one or more rates directly from an XML string

Parameters

Parameter (*=required)	Description
*sRateXML	XML string containing one or more ResourceRate object

Returns

0 = Success

Nonzero = Error

Remarks

The XML string passed in the parameter is of the form

<resourcerates><rate></rate><rate></rate></resourcerates>

which is the same form used in the XML for the ApiResource object.

This method will extract the rate objects and update the data for each. Ensure the <ResourceId> and <ResourceRateId> tags are filled with the correct values.

You cannot create a new rate using this method.

If the XML string is empty, the method will attempt to save the rates held within the object itself.

Example

```
Dim myResource as New ApiResource
Dim iRet as Int32 = 0
'Set the connection to the database
myResource.CPConnection = myCon

iRet = myResource.UpdateRates(sMyUpdateRateXML)
```

Related information

"ApiResource" on page 579

"ApiResource XML" on page 588

ApiResource: UpdateRolesAndFeatures

```
Public Function UpdateRolesAndFeatures(ByVal sResourceId As String, ByVal
sRoles As String, ByVal sFeatures As String) As Int32
```

Purpose

Replace the resource's roles and features with passed in data.

Parameters

Parameter (*=required)	Description
*sResourceId	Resource whose roles and features are being modified.
*sRoles	A string of Role IDs separated by the pipe character (|)
*sFeature	A string of Feature IDs separated by the pipe character (|)

Returns

0 = Success

Nonzero = Error

Remarks

In both sRoles and sFeatures, the update process removes any existing roles or features from the resource. If you pass in an empty string "" for sRoles, all roles assigned to the resource are removed. If you pass in an empty string "" for sFeatures, all features assigned to the resource are removed.

Example

Not available

Related information

"ApiResource" on page 579

ApiResource: UpdateWebPassword

```
Public Function UpdateWebPassword(Optional ByVal sResourceId As String = "",
Optional ByVal sPassword As String = "") As Int32
```

Purpose

Allows changes to the web password for a specified resource

Parameters

Parameter (*=required)	Description
sResourceId	ResourceId of resource whose password is being changed.
sPassword	New password

Returns

0 = Success

Nonzero = Error

Remarks

The password will be encrypted and saved to the database; therefore, it is not necessary to pass in an encrypted form of the new password.

Example

```
Dim myResource as New ApiResource
Dim sNewPassword as String = "J0hns563*7"
Dim iRet as Int32 = 0
Dim sResId As String = ""
'Set the connection to the database
myResource.CPConnection = myCon
sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"

iRet = myResource.UpdateWebPassword(sResId, sNewPassword)
```

Related information

"ApiResource" on page 579

ApiResource: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Updates Resource data in the database with data held in the object

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

Update will write all data held in the Resource object including the internal objects for Address, Payroll and any Rate objects.

The API does not verify if data in the resource objects or subobjects have been modified. When an Update() method is called, the entire object and subobject collections are updated regardless whether modifications have been done or not in the object or subobject data. The UpdatedOn and UpdatedBy fields in all related tables such as ResourceRate are overwritten, each time the Update() method is used.

This process carries a higher overhead than calling update from the smaller objects such as Address. Therefore it is a good idea to use Update when there are large changes across the entire Resource. If changes are centered on an area such as Address or Payroll, it is quicker to instantiate an instance of those objects and run an update from the Address or Payroll objects.

Depending on requirements, metadata checking can be turned on or off by using the ByPassMetaDataCheck switch. Setting this switch will halt any metadata checking in the object including UDFs.

Example

```
Dim myResource as New ApiResource
Dim iRet as Int32 = 0
Dim sResId As String = ""
Dim myNewRate as ApiResourceRate
'Set the connection to the database
myResource.CPConnection = myCon
sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
'Get current data
iRet = myResource.GetById(sResId)
If iRet <> 0 Then
'Handle the error
End If
'edit some data
With myResource
   .Title = "New Team Lead"
   .TitleEffectiveDate = "08/15/2007"
   .BaseCurrency = "USD"
   ...
```

```
   .Address.AddressLine = "131 Winding Lane"
   .Address.City = "Toronto"
   .Address.StateProvince = New Identity("{06810a99-77ba-41ce-91be-
2f8747ccc65f}")
   ...
   .Payroll.HoursPerDay = 8.0
   .Payroll.NonWorkingDay = "1000001"
   .Payroll.AnnualVacationHours = 300
   .Payroll.LieuCarryOver = 80
   ...
End With
myNewRate = New ApiResourceRate
myNewRate = myCon
With myNewRate
   .Currency = "USD"
   .EffectiveDate = "08/15/2007"
   .HourlyBilling = 90.00
   .DailyBilling = 900.00
End With
'Place the new rate into the resource object
iRet = myResource.SetRates(myNewRate)
If iRet <> 0 then
   'Handle error
End If
'Update the resource
iRet = myResource.Update
Return iRet
```

Related information

"ApiResource" on page 579

ApiResourceAddress

The ApiResourceAddress object contains common address information. IResourceAddress inherits properties from the IAddressBase interface.

Namespace

Changepoint.ChangepointAPI2.ApiResourceAddress

Methods

There is no Add method as a new address record is automatically created in the database when a resource is created in Changepoint, whether through the API or Enterprise.

Properties

Property (*=required)	Type	Description
AddressLine	String(255)	The business address for the resource.
ByPassMetaDataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: CPMetadataCheck.CheckAll = 0 CPMetadataCheck.BypassAll = 1 CPMetadataCheck.OnlyMandatory = 2
CellPhone	String(50)	The resource's cell phone number.
City	String(255)	The city or town for the business address.
Country	Identity	Identifier of the country for the business address.
*CPConnection	ApiConnection	Write-only. The connection object that must be assigned before any action to the database.
Deleted	Boolean	Flag to indicate whether record is deleted.
Email	String(100)	Email address of the resource
Entity	CPEntity	Read-only. The Changepoint entity the object represents.
Fax	String(50)	The fax phone number.
HomePhone	String(50)	The resource's primary phone number.

Property (*=required)	Type	Description
Location	Identity	Identifier of the resource business location, for example, the city where the resource works, or the sales region the resource covers.
mVersion	String	Read-only. The current version number for internal version control.
OfficePhone	String(50)	The resource's office phone number.
OfficePhoneExt	String(50)	The extension for the office phone number.
Pager	String(50)	The resource's pager number.
PostalCode	String(20)	The postal or zip code for the business address.
*ResourceId	String	The resource linked to the address data.
ShowCell	Boolean	Switch used to show or hide the cell phone number.
ShowHome	Boolean	Switch used to show or hide the primary phone number.
ShowPager	Boolean	Switch used to display or hide a resources pager number for public access.
StateProvince	Identity	Identifier of the state, province, or designated region for the business address.
UpdatedBy	Signature	Shows who last updated the record and when.

Related information

"ApiResource" on page 579

"ApiResourceAddress XML" on page 634

ApiResourceAddress XML

```
<root>
   <resource>
      ...
      <address>
         <updatedbyid />
         <updatedon />
         <deleted>false</deleted>
         <addressline />
         <city />
         <stateprovince>
            <id />
            <name />
            <alternatename />
         </stateprovince>
         <country>
            <id />
            <name />
            <alternatename />
         </country>
         <postalcode />
         <resourceid />
         <location >
             <id />
             <name />
             <alternatename />
         </location >
         <officephone />
         <officephoneext />
         <homephone />
         <cellphone />
         <pager />
         <fax />
         <email />
         <showcell>false</showcell>
         <showhome>false</showhome>
         <showpager>false</showpager>
         <bypassmetadatacheck />
      </address>
      ...
   </resource>
</root>
```

Comments

The XML for the ApiResourceAddress object is included in the XML for the ApiResource object. The following container elements are mandatory:

```
<root>
   <resource>
      <address>
         ...
      </address>
   </resource>
</root>
```

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

Example

```
<root>
   <resource>
      ...
      <address>
         <updatedbyid />
         <updatedon />
         <deleted>false</deleted>
         <addressline>123 Main DR</addressline>
         <city>Richmond Hill</city>
         <stateprovince>
            <id />
            <name>Ontario</name>
            <alternatename />
         </stateprovince>
         <country>
            <id />
            <name>Canada</name>
            <alternatename />
         </country>
         <postalcode>M2J4R8</postalcode>
         <resourceid />
         <location >
             <id />
             <name>Central Ontario<name />
             <alternatename />
         </location >
         <officephone />
         <officephoneext />
         <homephone />
         <cellphone />
```

```
            <pager />
            <fax />
            <email>John.Smith@changepoint.com</email>
            <showcell>false</showcell>
            <showhome>false</showhome>
            <showpager>false</showpager>
            <bypassmetadatacheck>0</bypassmetadatacheck>
         </address>
         ...
      </resource>
   </root>
```

Related information

"ApiResourceAddress" on page 631

"ApiResource XML" on page 588

ApiResourceAddress: GetById

```
Public Overrides Function GetById(Optional ByVal sId as String = "") As Int32
```

Purpose

Retrieves current resource address data

Parameters

Parameter (*=required)	Description
sId	The ID of the entity (resource).

Returns

0 = Success

Nonzero = Error

Remarks

Retrieves current data into the object. Any existing data is replaced with fresh data from the database.

Example

Not available

Related information

"ApiResourceAddress" on page 631

ApiResourceAddress: New

```
Public Overrides Function New()
```

Purpose

Instantiates a new object of ApiResourceAddress Type

Parameters

None

Returns

Not applicable

Remarks

Instantiates a new ApiResourceAddress object.

Example

Not available

Related information

"ApiResourceAddress" on page 631

ApiResourceAddress: SetPropertiesByXML

```
Public Function SetPropertiesByXML(ByVal sXML as String, Optional ByVal
bNotInitialize as Boolean = True) As Int32
```

Purpose

Set the properties of the object using an XML string

Parameters

Parameter (*=required)	Description
*sXML	XML string of the Address object with data in the fields.
bNotInitialize	Switch that initializes the object. Normal use is to set as "True"

Returns

0 = Success

Nonzero = Error

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods of ApiResource.

The minimal XML that can be passed to this method is:

```
<root>
   <address>
      <resourceid></resourceid>
   </address >
</root>
```

If this method is called from an empty object, the object will fill itself first with the most recent data before applying any changes from sXML.

Note that empty fields in sXML will clear fields in the object. Therefore to preserve values, either remove the tags from the XML string or ensure the tag values are the same as currently in the database.

Example

Not available

Related information

"ApiResourceAddress" on page 631

"ApiResourceAddress XML" on page 634

"ApiResource: CreateByXML" on page 597

"ApiResource: UpdateByXML" on page 623

ApiResourceAddress: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Updates the database with the data held in the object properties.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

```
Dim myAddress as New ApiResourceAddress
Dim iRet as Int32 = 0
myAddress.CPConnection = myCon
With myAddress
   .ResourceId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
   .AddressLine = "131 Winding Lane"
   .City = "Toronto"
   .StateProvince = New Identity("{06810a99-77ba-41ce-91be-2f8747ccc65f}")
   ...
End With
iRet = myAddress.Update
```

Related information

"ApiResourceAddress" on page 631

ApiResourcePayroll

The ApiResourcePayroll object contains resource payroll information.

Namespace

Changepoint.ChangepointAPI2.ApiResourcePayroll

Methods

Properties

Property (*=required)	Type	Description
AnnualTargetHours	Double	The number of hours the organization expects the resource to work each year.
AnnualVacationHours	Double	The number of vacation hours a resource can accrue each year.
BypassMetadataCheck	Byte	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
*CertificationCycle	Identity	Identifier of the frequency of time submission. The valid values of the Id property of the Identity data type are: • D = Daily • W = Weekly • N = Not Applicable.
CertificationHours	Double	Hours required by resource before they can certify.

Property (*=required)	Type	Description
ConversionToDay	Double	Daily conversion factor for the resource
EarliestTimeEntry	Date	The date of the very first time entry.
Entity	CPEntity	Read-only. The Changepoint entity the object represents.
*FirstPayroll	Date	The date that starts the resource's payroll cycle.
HoursPerDay	Double	A resource's required working hours per day. On create, if no value is passed, HoursPerDay is Null. On update, to change from any value to null, pass in 'null'.
LieuCarryOver	Double	Lieu time carried over from the previous year.
mVersion	String	The current version number for internal version control.
NonWorkingDay	String	A sequence of seven numbers representing days of the week in which 1 represents a nonworking day and 0 represents a work day. The sequence starts with Sunday and ends on Saturday. For example, if Saturday and Sunday are nonworking days, the sequence is 1000001.
*PayrollCycle	Identity	Changepoint lookup code. Defines the pay period (weekly, bi-weekly, monthly, semi-monthly).
PayrollGroup	String	The name of the payroll group to which the resource belongs.
PayrollLagDays	Integer	The maximum number of days after a payroll period end date that time can be entered for that payroll period.

Property (*=required)	Type	Description
ResourceId	String	ID of the resource.
Updatedby	Signature	Details about when the object was updated and by whom.
VacationCarryOver	Double	Vacation time carried over from the previous year.

Related information

"ApiResource" on page 579

"ApiResourcePayroll XML" on page 642

ApiResourcePayroll XML

```
<root>
   <resource>
      ...
      <payroll>
         <updatedbyid />
         <updatedon />
         <annualtargethours>0</annualtargethours>
         <annualvacationhours />
         <certificationcycle>
            <id />
            <name />
            <alternatename />
         </certificationcycle>
         <certificationhours />
         <earliesttimeentry />
         <firstpayroll />
         <hoursperday />
         <lieucarryover>0</lieucarryover>
         <payrollcycle>
            <id />
            <name />
            <alternatename />
         </payrollcycle>
         <payrollgroup />
         <payrolllagdays>0</payrolllagdays>
         <resourceid />
         <vacationcarryover />
         <nonworkingday>1000001</nonworkingday>
         <conversiontoday>8</conversiontoday>
```

```
         <bypassmetadatacheck />
      </payroll>
      ...
   </resource>
</root>
```

Comments

The XML for the ApiResourcePayroll object is included in the XML for the ApiResource object. The following container elements are mandatory:

```
<root>
   <resource>
      <payroll>
         ...
      </payroll>
   </resource>
</root>
```

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

Example

```
<root>
   <resource>
      ...
      <payroll>
         <updatedbyid />
         <updatedon />
         <annualtargethours>0</annualtargethours>
         <annualvacationhours />
         <certificationcycle>
            <id />
            <name>Weekly</name>
            <alternatename />
         </certificationcycle>
         <certificationhours>40</certificationhours>
         <earliesttimeentry />
         <firstpayroll>12/02/2009</firstpayroll>
         <hoursperday>8</hoursperday>
         <lieucarryover>0</lieucarryover>
         <payrollcycle>
            <id />
            <name>Bi-weekly</name>
            <alternatename />
         </payrollcycle>
         <payrollgroup />
```

```
            <payrolllagdays>7</payrolllagdays>
            <resourceid />
            <vacationcarryover>0</vacationcarryover>
            <nonworkingday>1000001</nonworkingday>
            <conversiontoday>8</conversiontoday>
            <bypassmetadatacheck />
        </payroll>
        ...
    </resource>
</root>
```

Related information

"ApiResourcePayroll" on page 639

"ApiResource XML" on page 588

ApiResourcePayroll: GetById

```
Public Overrides Function GetById(Optional ByVal sId as String = "") As Int32
```

Purpose

Retrieves current resource payroll data

Parameters

Parameter (*=required)	Description
sId	The ID of the entity (resource).

Returns

0 = Success

Nonzero = Error

Remarks

Retrieves current data into the object. Any existing data is replaced with fresh data from the database.

Example

Not available

Related information

"ApiResourcePayroll" on page 639

ApiResourcePayroll: New

```
Public Function New()
```

Purpose

Instantiates a new object of ApiResourcePayroll Type

Parameters

None

Returns

Not applicable

Remarks

Instantiates a new ApiResourcePayroll object.

The ByPassMetadataCheck flag is set to False by default.

Example

Not available

Related information

"ApiResourcePayroll" on page 639

ApiResourcePayroll: SetPropertiesByXML

```
Public Function SetPropertiesByXML(ByVal sXML as String, Optional ByVal
bNotInitialize As Boolean = True) As Int32
```

Purpose

Sets the properties of the object using an XML string

Parameters

Parameter (*=required)	Description
*sXML	XML string of the ApiResourcePayroll object with data in the fields.
bNotInitialize	Switch that initializes the object. Normal use is to set as "True"

Returns

0 = Success

Nonzero = Error

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods of ApiResource.

The minimal XML that can be passed to this method is:

```
<root>
   <payroll>
      <resourceid></resourceid>
   </payroll>
</root>
```

If this method is called from an empty object, the object will fill itself first with the most recent data first before applying any changes from sXML.

Note that empty fields in sXML will clear fields in the object. Therefore to preserve values, either remove the tags from the XML string or ensure the tag values are the same as currently in the database.

Example

Not available

Related information

"ApiResourcePayroll" on page 639

"ApiResourcePayroll XML" on page 642

"ApiResource: CreateByXML" on page 597

"ApiResource: UpdateByXML" on page 623

ApiResourcePayroll: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Updates the database with the data held in the object properties.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

```
Dim myPayroll as New ApiResourcePayroll
Dim iRet as Int32 = 0

myPayroll .CPConnection = myCon

With myPayroll
   .ResourceId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
   .HoursPerDay = 8.0
   .NonWorkingDay = "1000001"
   .AnnualVacationHours = 240
   ...
End With
iRet = myPayroll.Update
```

Related information

"ApiResourcePayroll" on page 639

ApiResourceRate

The ApiResourceRate object allows users to get, update and add resource rate information.

Namespace

Changepoint.ChangepointAPI2.ApiResourceRate

Methods

Properties

Property (*=required)	Type	Description
Active	Boolean	Indicates whether the rate is active and will take effect on the effective date.
ByPassMetaDataCheck	Boolean	Determines the level of MetaData checking when adding or updating data. (The default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
Comments	String	Comment.
CommentsHistory	String	Read-only. Cumulative (historical) comments to date regarding changes made to the resource's rates. Get only.

Property (*=required)	Type	Description
Currency	String	The currency type for the billing rate.
CPConnection	ApiConnection	The connection object that must be assigned before any action to the database.
DailyBilling	Currency	The amount of money the organization charges for each full day the resource works for a customer.
DailyCost	Currency	The amount of money it costs the organization for each full day the resource works.
EffectiveDate	Date	The date the rate takes effect.
Entity	CPEntity	Read-only. The Changepoint entity the object represents.
HourlyBilling	Currency	The amount of money the organization charges for each hour the resource works for a customer.
HourlyCost	Currency	The amount of money it costs the organization for each hour the resource works.
mVersion	String	The current version number for internal version control.
OverTimeBilling	Currency	The amount of money the organization charges for each overtime hour (hours worked over the value in the Full Day Setting field) the resource works for a customer.
OverTimeCost	Currency	The amount of money it costs the organization for each overtime hour the resource works.

Property (*=required)	Type	Description
Rate1	Currency	Used to store additional information used for custom reporting purposes only.
Rate2	Currency	Used to store additional information used for custom reporting purposes only.
Rate3	Currency	Used to store additional information used for custom reporting purposes only.
Resource	Identity	Identifier of the resource to whom the rate applies.
ResourceRateId	String	The row ID of the resource rate record. The ID is set to "" if it is a new rate.
UpdatedBy	Signature	Who last updated the record and when.

Related information

"ApiResource" on page 579

"ApiResourceRate XML" on page 650

ApiResourceRate XML

```
<root>
   <resource>
      ...
      <resourcerates>
         <rate>
            <updatedby />
            <updatedon />
            <active>true</active>
            <comments />
            <commentshistory />
            <currency />
            <dailybilling>0</dailybilling>
            <dailycost>0</dailycost>
            <effectivedate />
            <hourlybilling>0</hourlybilling>
            <hourlycost>0</hourlycost>
            <overtimebilling>0</overtimebilling>
            <overtimecost>0</overtimecost>
            <rate1>0</rate1>
```

```
            <rate2>0</rate2>
            <rate3>0</rate3>
            <resourcerateid />
            <resource>
               <id />
               <firstname />
               <lastname />
               <alternatename />
               <userdefinedid />
            </resource>
            <bypassmetadatacheck />
         </rate>
      </resourcerates>
      ...
   </resource>
</root>
```

Comments

The XML for the ApiResourceRate object is included in the XML for the ApiResource object. The following container elements are mandatory:

```
<root>
   <resource>
      <resourcerates>
         <rate>
            ...
         </rate>
      </resourcerates>
   </resource>
</root>
```

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

Example

```
<root>
   <resource>
      ...
      <resourcerates>
         <rate>
            <updatedby />
            <updatedon />
            <active>true</active>
            <comments />
            <commentshistory />
            <currency>CAD</currency>
```

```
            <dailybilling>700.00</dailybilling>
            <dailycost>350.00</dailycost>
            <effectivedate>11/30/2009 </effectivedate>
            <hourlybilling>100.00</hourlybilling>
            <hourlycost>50.00</hourlycost>
            <overtimebilling>0</overtimebilling>
            <overtimecost>0</overtimecost>
            <rate1>0</rate1>
            <rate2>0</rate2>
            <rate3>0</rate3>
            <resourcerateid />
            <resource>
               <id />
               <firstname />
               <lastname />
               <alternatename />
               <userdefinedid />
            </resource>
            <bypassmetadatacheck>0</bypassmetadatacheck>
         </rate>
      <resourcerates>
      ...
   </resource>
</root>
```

Related information

"ApiResourceRate" on page 648

"ApiResource XML" on page 588

ApiResourceRate: Add

```
Public Overrides Function Add(Optional ByVal sResId As String = "{00000000-
0000-0000-0000-000000000000}", Optional ByRef sId As String = "")As Int32
```

Purpose

Adds a new rate to Changepoint

Parameters

Parameter (*=required)	Description
sResId	The ID of the entity (resource) related to the new rate being added.
sId	The new ResourceRateId.

Returns

0 = Success

Nonzero = Error

Remarks

If sResId contains a value that does not exist in Changepoint as a current resource, then the ResourceId of the object is used as the target resource to which the new rate will apply.

Example

```
Dim myRate As New ApiResourceRate
Dim iRet As New Int32 = 0
Dim sNewId As String = ""

myRate.CPConnection = myCon

With myRate
.ResourceId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
.Currency = "CAD"
.EffectiveDate = Now.Date
.HourlyBilling = 80.00
.DailyBilling = 500.00
End With
iRet = myRate.Add("{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}", sNewId)
If iRet <> 0 or sNewId.Length = 0 Then
'Handle error
End If
'Continue processing
```

Related information

"ApiResourceRate" on page 648

ApiResourceRate: GetById

```
Public Overrides Function GetById(Optional ByVal sId as String = "", Optional
ByVal sResourceId as String ) As Int32
```

Purpose

Retrieves current resource rate data into the object.

Parameters

Parameter (*=required)	Description
sId	ResourceRateId.
sResourceId	ResourceId

Returns

0 = Success

Nonzero = Error

Remarks

Retrieves current data into the object. Any existing data is replaced with fresh data from the database.

Example

Not available

Related information

"ApiResourceRate" on page 648

ApiResourceRate: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Returns the XML structure of the ApiResourceRate object.

Parameters

None

Returns

An XML string of the ApiResourceRate object.

Remarks

Some fields in the structure will have defaulted data, othewise fields are empty.

Example

```
Dim myRate As New ApiResourceRate
Dim sStructure as String = ""
sStructure = myRate.GetXMLStructure()
```

Related information

"ApiResourceRate" on page 648

"ApiResourceRate XML" on page 650

ApiResourceRate: New

```
Public Function New()
```

Purpose

Instantiates a new object of ApiResourceRate Type

Parameters

None

Returns

Not applicable

Remarks

Instantiates a new ApiResourceRate object.

The ByPassMetadataCheck flag is set to False by default.

Example

Not available

Related information

"ApiResourceRate" on page 648

ApiResourceRate: SetPropertiesByXML

```
Public Function SetPropertiesByXML(Optional ByRef sXML as String = "",
Optional ByVal bNotInitialize As Boolean = True ) As Int32
```

Purpose

Set the properties of the object using an XML string

Parameters

Parameter (*=required)	Description
sXML	XML string of the ApiResourceRate object with data in the fields.
bNotInitialize	Switch that initializes the object. Normal use is to set as "True."

Returns

This legacy method has been superseded by the CreateByXML and UpdateByXML methods of ApiResource

The minimal XML that can be passed to this method is:

```
<root>
   <rate>
      <resourceid></resourceid>
   </rate>
</root>
```

If this method is called from an empty object, the object will fill itself first with the most recent data first before applying any changes from sXML.

Note that empty fields in sXML clear fields in the object. Therefore to preserve values, either remove the tags from the XML string or ensure the tag values are the same as currently in the database.

Remarks

None

Example

Not available

Related information

"ApiResourceRate" on page 648

"ApiResourceRate XML" on page 650

"ApiResource: CreateByXML" on page 597

"ApiResource: UpdateByXML" on page 623

ApiResourceRate: Update

```
Public Overrides Function Update() As Integer
```

Purpose

Updates the database with the data held in the object properties.

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

If the <ResourcerateId> tag is empty, the update process will consider the rate data as new and will add a new rate. If the rate is new, the object's ResourceRateId property will contain the value of the new GUID of the record in the database.

Example

```
Dim myRate as New ApiResourceRate
Dim iRet as Int32 = 0
Dim sNewRateId as String = ""
```

```
myRate .CPConnection = myCon

With myRate
   .ResourceId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
   .Currency = "CAD"
   .EffectiveDate = Now.Date
   .HourlyBilling = 80.00
   .DailyBilling = 500.00
   ...
End With
iRet = myRate.Update
If iRet = 0 Then
   sNewRateId = myRate.ResourceRateId
End If
Return sNewRateId
```

Related information

"ApiResourceRate" on page 648

ApiSkill

The ApiSkill object allows users to view, create and edit skills for resources.

Namespace

Changepoint.ChangepointAPI2.ApiSkill

Methods

Properties

Property (*=required)	Type	Description
*CPConnection	ApiConnection	Connection to the database
Description	String	Description of the skill

Property (*=required)	Type	Description
*Name	String	The name of the skill
*SkillCategoryId	String	The Changepoint skill category ID
SkillId	String	Changepoint skill code.

Related information

"ApiSkillCategory" on page 663

"ApiSkillCompetency" on page 668

ApiSkill: Add

```
Public Overrides Function Add(ByRef oSkill As ApiSkill) As String
```

Purpose

Add a new skill to Changepoint.

Parameters

Parameter (*=required)	Description
*oSkill	ApiSkill object.

Returns

A new SkillId on success else an empty string

Remarks

The SkillCategoryId field must contain a valid GUID value.

Example

```
Dim mySkill as New ApiSkill
Dim sNewId as String = ""
Dim ds as New DataSet
Dim mySkillCat as New ApiSkillCategory

mySkill.CPConnection = MyCon
mySkillCat.CPConnection = MyCon
```

```
ds = mySkillCat.GetSkillCategories()
If ds.tables.Count > 0 AndAlso ds.Tables(0)
   For Each mRow As DataRow in ds.tables(0).Rows
      If mRow.Item("Name").ToString = "SkillCategory1" Then
         mySkill.SkillCategoryId = mRow.Item ("Code").ToString
         Exit For
      End If
   Next mRow
   Else
   ...
End If

'If the Id field is empty then the skill category was not found and the
process stops
If mySkill.SkillCategoryId.Length = 0 Then
   'Handle error
Else
   With mySkill
      .Description = "A test skill description"
      .Name = "MyTestSkill-1"
   End With
   'Add the new skill to Changepoint
   sNewId = mySkill.Add(mySkill)
End If
'Continue processing
```

Related information

"ApiSkill" on page 658

ApiSkill: GetSkill

```
Public Function GetSkill(ByRef sSkillId As String) As ApiSkill
```

Purpose

Return a skill based on the specified SkillId

Parameters

Parameter (*=required)	Description
*sSkillId	Changepoint Skill ID.

Returns

An ApiSkill object

Remarks

None

Example

Not available

Related information

"ApiSkill" on page 658

ApiSkill: GetSkills

```
Public Function GetSkills(ByRef sSkillCategoryId As String) As DataSet
```

Purpose

Return a list of skills based on a specified Skill category.

Parameters

Parameter (*=required)	Description
*sSkillCategoryId	Skill Category ID.

Returns

DataSet (SkillCode, Name) ordered by name

Remarks

None

Example

Not available

Related information

"ApiSkill" on page 658

ApiSkill: Update

```
Public Overrides Function Update(ByRef oSkill As ApiSkill) As Int32
```

Purpose

Update a skill

Parameters

Parameter (*=required)	Description
*oSkill	ApiSkill object.

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

```
Dim mySkill as New ApiSkill
Dim sId as String = ""
Dim ds as New DataSet
Dim mySkillCat as New ApiSkillCategory
Dim iErr as Int32 = 0

mySkill.CPConnection = MyCon
mySkillCat.CPConnection = MyCon
'get a list of skill categories and extract necessary code
ds = mySkillCat.GetSkillCategories(sErr)
If ds.tables.Count > 0 AndAlso ds.Tables(0)
   For Each mRow As DataRow in ds.tables(0).Rows
   If mRow.Item("Name").ToString = "SkillCategory1" Then
      mySkill.SkillCategoryId = mRow.Item ("Code").ToString
      Exit For
   End If
   Next mRow
   Else
   ...
End If
'If the Id field is empty then the skill category was not found
```

```
'and the process stops
If mySkill.SkillCategoryId.Length = 0 Then
   'Handle error
Else
ds = New DataSet
'Retrieve all skills in the category
ds = mySkill.GetSkills(mySkill.SkillCategoryId)
'Iterate and extract the required skill code
If ds.tables.Count > 0 AndAlso ds.Tables(0).Rows.Count > 0 Then
   For Each mRow As DataRow in ds.tables(0).Rows
      If mRow.Item("Name").ToString = "MyTestSkill-1" Then
         sId = mRow.Item("SkillCode").ToString
         Exit For
      End If

   mySkill = mySkill.GetSkill(sId)
      With mySkill
         .Description = "An updated test skill description"
         .Name = "MyUpdatedTestSkill-1"
      End With
   'Add the updated skill back into Changepoint
   iErr = mySkill.Update(mySkill)
End If
If lErr <> 0 Then
   'Handle error
End If
'Continue processing
```

Related information

"ApiSkill" on page 658

ApiSkillCategory

The ApiSkillCategory object represents skill categories within the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiSkillCategory

Methods

Properties

Property (*=required)	Type	Description
*CPConnection	ApiConnection	Connection to the database.
Description	String	Description of skill category.
*Name	String	The name of skill category.
SkillCategoryId	String	Read-only. Changepoint skill category code.

Related information

"ApiSkill" on page 658

"ApiSkillCompetency" on page 668

ApiSkillCategory: Add

```
Public Overrides Function Add(ByRef oSkillCategory As ApiSkillCategory) As
String
```

Purpose

Create a new skill category in Changepoint

Parameters

Parameter (*=required)	Description
*oSkillCategory	ApiSkillCategory object.

Returns

A new SkillCategoryId on success else an empty string.

Remarks

None

Example

```
Dim mySkillCategory as New ApiSkillCategory
Dim sNewId as String = ""

mySkillCategory.CPConnection = myCon
mySkillCategory.Name = "SkillCategory1"
mySkillCategory.Description = "Test Category"
sNewId = mySkillCategory.Add(mySkillCategory)
if sNewId.Length = 0 Then
   'Handle error
End if
```

Related information

"ApiSkillCategory" on page 663

ApiSkillCategory: GetSkillCategories

```
Public Function GetSkillCategories(Optional ByRef ErrorValue As Long) As
ADODB.Recordset
```

Purpose

Get a list of all current skill categories in Changepoint.

Parameters

Parameter (*=required)	Description
ErrorValue	Error value returned.

Returns

ADODB.Recordset (Code, Name)

Remarks

None

Example

Not available

Related information

"ApiSkillCategory" on page 663

ApiSkillCategory: GetSkillCategory

```
Public Function GetSkillCategory(ByRef sSkillCategoryId As String) As
ApiSkillCategory
```

Purpose

Get a skill category based on a specified skill category ID

Parameters

Parameter (*=required)	Description
*sSkillCategoryId	Changepoint Skill Category ID.

Returns

A single ApiSkillCategory object.

Remarks

None

Example

Not available

Related information

"ApiSkillCategory" on page 663

ApiSkillCategory: Update

```
Public Overrides Function Update(ByRef oSkillCategory As ApiSkillCategory) As
Int32
```

Purpose

Update an existing skill category in Changepoint.

Parameters

Parameter (*=required)	Description
*oSkillCategory	ApiSkillCategory object.

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

```
Dim mySkillCategory as New ApiSkillCategory
Dim sNewId As String = ""
Dim sSkillCatId As String = ""
Dim iErr As Int32
Dim ds As DataSet

mySkillCategory.CPConnection = myCon

'Get a list of categories, and extract the correct code value.
'If you already have the code this part can be skipped
ds = mySkillCategory.GetSkillCategories()
If ds.tables.Count > 0 AndAlso ds.Tables(0)
   For Each mRow As DataRow in ds.tables(0).Rows
      If mRow.Item("Name").ToString = "SkillCategory1" Then
         sSkillCatId = mRow.Item ("Code").ToString
         Exit For
      End If
   Next mRow
Else
   ...
End If

'Retrieve the required skill category
If sSkillCatId.Length > 0 Then
   mySkillCategory = mySkillCategory.GetSkillCategory(sSkillCatId)
End If

'Update the data
mySkillCategory.Name = "SkillUpdatedCategory1"
```

```
mySkillCategory.Description = "Update Test Category"

'Save the data
iErr = mySkillCategory.Update(mySkillCategory)
If lErr <> 0 Then
   'Handle error
End if
```

Related information

"ApiSkillCategory" on page 663

ApiSkillCompetency

The ApiSkillCompetency object allows users to create, retrieve and update competencies for skills within the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiSkillCompetency

Methods

Properties

Property (*=required)	Type	Description
SkillCompetencyId	String	Changepoint skill competency code
*SkillCategoryId	String	Changepoint skill category code
*SkillId	String	Changepoint skill code
*Name	String	The name of skill category
Description	String	Description of skill category

Related information

"ApiSkill" on page 658

"ApiSkillCategory" on page 663

ApiSkillCompetency: Add

```
Public Overrides Function Add(ByRef oSkillCompetency As ApiSkillCompetency) As
String
```

Purpose

Add a new skill competency to Changepoint

Parameters

Parameter (*=required)	Description
*oSkillCompetency	ApiSkillCompetency object

Returns

String (a new SkillCompencyId)

Remarks

Both SkillId and SkillCategoryId fields must contain a valid GUID value.

Example

```
Dim mySkillCompetency As New ApiSkillCompetency
Dim sNewId As String = ""
mySkillCompetency.CPConnection = myCon
mySkillCompetency.SkillId = "{0609965a-3344-4126-89dd-0aece80ec474}"
mySkillCompetency.SkillCategoryId = "{2435a154-a3d9-4a3d-a317-26ba9080b03e}"
mySkillCompetency.Name = "Test Skill Competency"
mySkillCompetency.Description = "Test description"
sNewId = mySkillCompetency.Add(mySkillCompetency)
   'Continue processing
```

Related information

"ApiSkillCompetency" on page 668

ApiSkillCompetency: GetSkillCompetencies

```
Public Function GetSkillCompetencies(ByRef sSkillId As String) As DataSet
```

Purpose

Get a list of all current Skill Competencies in Changepoint

Parameters

Parameter (*=required)	Description
*sSkillId	Changepoint Skill ID.

Returns

DataSet (Code, Name)

Remarks

None

Example

Not available

Related information

"ApiSkillCompetency" on page 668

ApiSkillCompetency: GetSkillCompetency

```
Public Function GetSkillCompetency(ByRef sSkillCompetencyId As String) As
ApiSkillCompetency
```

Purpose

Retrieve data on a single skill competency based on a specified SkillCompetencyId

Parameters

Parameter (*=required)	Description
*sSkillCompetencyId	Changepoint Skill Competency ID.

Returns

A single ApiSkillCompetency object.

Remarks

None

Example

Not available

Related information

"ApiSkillCompetency" on page 668

ApiSkillCompetency: Update

```
Public Overrides Function Update(ByRef oSkillCompetency As ApiSkillCompetency)
As Int32
```

Purpose

Update a skill competency in Changepoint

Parameters

Parameter (*=required)	Description
*oSkillCompetency	ApiSkillCompetency object.

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

```
Dim mySkillCompetency As New ApiSkillCompetency
Dim sNewId As String = ""
Dim iErr As Int32
```

```
mySkillCompetency.CPConnection = myCon
   'If the SkillCompetencyId is not known, only the name then use
GetSkillCompetencies and
   'locate the required SkillCompetency and extract the Id value
mySkillCompetency = mySkillCompetency.GetSkillCompetency("{8c17755c-582b-4579-
8774-72643a9b1cd3 }")
mySkillCompetency.Name = "Test Skill Competency updated"
mySkillCompetency.Description = "Update test description"
iErr = mySkillCompetency.Update(mySkillCompetency)
If lErr <> 0 Then
   'Handle error
End If
'Continue processing
```

Related information

"ApiSkillCompetency" on page 668

ApiTask

The ApiTask object allows users to add, retrieve, update or delete task information within the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiTask

Methods

Properties

Property (*=required)	Type	Description
ActualFinish	DateTime	Read-only. Actual finish date
ActualHours	Double	Read-only. Actual hours
ActualStart	DateTime	Read-only. Actual start date
AffectUtilization	Boolean	Affect utilization
AllowTALocOverride	Boolean	When set to true, allows the task location to be overridden
AllowTAWorkCodeOverride	Boolean	When set to true, allows the task work code to be overridden
BaselineFinish	DateTime	Baseline finish of Task
BaseLineHours	Double	Baseline hours of Task
BaselineStart	DateTime	Baseline start of Task
Billable	Boolean	When set to true, the task is billable
BillingRole	Identity	Identifier of the billing role
BypassMetadataCheck	CPMetadataCheck	Determines the level of MetaData checking when adding or updating data (default is to Check All fields) Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Property (*=required)	Type	Description
Capital	Boolean	When set to true, the task is a capital task
*Completed	Boolean	When set to true, the task is completed
*CPConnection	ApiConnection	Write only. Connection object that must be assigned before any action to database
CPException	ApiException	Read-only. Exception object for handling and tracing exception information
Createdby	Signature	Resource who created the task
Customer	Identity	Read-only. Identifier of the customer
Description	String	Description string
Engagement	Identity	Read-only. Identifier of the contract.
EngagementProduct	Generic.List(Of Identity)	List of Contract Products.
Entity	CPEntity	Read-only. The Changepoint entity that the object represents
FixedFee	Boolean	When set to true, the task is a fixed fee task
FixedFeeId	String	Fixed fee ID
Functions	Identity	Identifier of the function
IgnoreWarning	Boolean	Ignore the Resource leveling warning
Locked	Boolean	When set to true, the task is locked

Property (*=required)	Type	Description
Milestone	Boolean	When set to true, the task is a milestone
MSP_TaskUniqueId	Int32	MSP Task ID
mVersion	String	Read-only. The current vresion number for internal version control
*Name	String	Task name
OpenTask	Boolean	When set to true, the task is open
ParentTaskId	String	Parent Task ID If provided as part of an "Add" method, it will not only create the task but also set the parent of the task to the ParentTaskId provided.
Phase	String	Phase
*PlannedFinish	DateTime	Planned finish date
*PlannedHours	Double	Planned hours
*PlannedStart	DateTime	Planned start date
ProductReadyToInvoice	Boolean	Whether the product is ready to be invoiced.
*Project	Identity	Identifier of the project
Request	Identity	Request associated to the task
*StatusNotRequired	Boolean	Status not required
sxmlUDF	String	The configurable field string in XML format
*TaskId	String	Task ID. Mandatory for update and delete.

Property (*=required)	Type	Description
UpdateAvailability	Boolean	The flag of update availability
Updatedby	Signature	Resource who updated the task
WBS	String	WBS string
WBSOrder	Int32	WBS order
*WorkCode	Identity	Identifier of the work code
*WorkCodeCategory	Identity	Identifier of the work code category
*WorkLocation	Identity	Identifier of the work location
*WorkLocationGroup	Identity	Identifier of the work location group

Related information

"ApiTask XML" on page 676

"ApiProject" on page 480

"ApiTaskAssignment" on page 691

ApiTask XML

```
<root>
   <project>
      <tasks>
         <task>
            <bypassmetadatacheck>checkall</bypassmetadatacheck>
            <createdbyid />
            <updatedbyid />
            <taskid />
            <name />
            <actualfinish/>
            <actualhours/>
            <actualstart/>
            <affectutilization>false</affectutilization>
            <allowtalocoverride>false</allowtalocoverride>
            <allowtaworkcodeoverride>false</allowtaworkcodeoverride>
            <baselinefinish />
            <baselinehours />
            <baselinestart />
            <billable>false</billable>
```

```
<billingrole>
   <id />
   <name />
   <alternatename />
</billingrole>
<capital>false</capital>
<completed>false</completed>
<customer>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</customer>
<description />
<engagement>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</engagement>
<entity />
<fixedfee>false</fixedfee>
<fixedfeeid />
<functions>
   <id />
   <name />
   <alternatename />
</functions>
<ignorewarning>false<ignorewarning>
<locked>false<locked>
<milestone>false</milestone>
<opentask>false</opentask>
<phase />
<plannedstart />
<plannedfinish />
<plannedhours />
<project>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</project>
<request>
   <id />
   <name />
   <alternatename />
</request>
<statusnotrequired>false</statusnotrequired>
```

```
            <sxmludf />
            <updateavailability>false</updateavailability>
            <workcode>
               <id />
               <name />
               <alternatename />
               <userdefinedid />
            </workcode>
            <workcodecategory>
               <id />
               <name />
               <alternatename />
               <userdefinedid />
            </workcodecategory>
            <worklocation>
               <id />
               <name />
               <alternatename />
               <userdefinedid />
            </worklocation>
            <worklocationgroup>
               <id />
               <name />
               <alternatename />
               <userdefinedid />
            </worklocationgroup>
            <iUpdate>1</iUpdate>
            <ProductReadyToInvoice />
            <EngagementProducts>
              <EngagementProduct>
               <id />
               <name />
               <alternatename />
              </EngagementProduct>
            </EngagementProducts>
            </ProductReadyToInvoice>
            <taskassignments>
            ...
            </taskassignments>
         </task>
      </tasks>
   </project>
</root>
```

Comments

The XML for the ApiTask object is included in the XML for the ApiProject object. The following container elements are mandatory:

```
<root>
   <project>
      <tasks>
         <task>
            ...
         </task>
      </tasks>
   </project>
</root>
```

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

For the use of the `<iUpdate>` flag in ApiProject XML, ApiTask XML, and APiTaskAssignment XML, see "ApiProject: CreateByXML" on page 492.

Example

```
<root>
   <Project>
      <Tasks>
         <Task>
            <TaskId></TaskId>
            <TaskName>Taskt1</TaskName>
            <Description>API test</Description>
            <PlannedStart>2005/9/14</PlannedStart>
            <PlannedFinish>2005/9/30</PlannedFinish>
            <PlannedHours>50</PlannedHours>
            <WBS>1</WBS>
            <WBSOrder>1</WBSOrder>
            <Billable>0</Billable>
            <FixedFee>0</FixedFee>
            <Functions><id>0703CD9C-8D68-4FAC-B487-
31705D989BA2</id><name/><alternatename/></FunctionId>
            <Locked>0</Locked>
            <AffectUtilization>0</AffectUtilization>
            <StatusNotRequired>0</ StatusNotRequired>
            <WorkLocationGroup><id>333A206C-513E-4CAE-841A-
EA73A82E8E81<id/><name/><alternatename/></WorkLocationGroup>
            <WorkLocation><id>0409A21B-EFB9-42C4-ADB3-
617279F524EF</id><name/><alternatename/></WorkLocation>
            <WorkCodeCategory><id>AA9C83C1-6E1E-4974-9DEC-
7C554CC429D2</id><name/><alternatename/></WorkCodeCategory>
```

```
            <WorkCode><id>8A4BEED2-A035-4407-948D-
248681472747</id><name/><alternatename/></WorkCode>
            <udf>
            ...
            </udf>
            <iUpdate>1</iUpdate>
            <ProductReadyToInvoice>true</ProductReadyToInvoice>
            <EngagementProducts>
             <EngagementProduct>
               <id>2C016CB9-269E-E921-EC37-900A765B1801</id>
               <name />
               <alternatename />
             </EngagementProduct>
             <EngagementProduct>
               <id>71A885B5-96A5-82E4-4D35-977BB2A0B2AA</id>
               <name />
               <alternatename />
             </EngagementProduct>
            </EngagementProducts>
            <TaskAssignments>
            ...
            </TaskAssignments>
         </Task>
      </Tasks>
   </Project>
</root>
```

Related information

"ApiTask" on page 672

"ApiProject XML" on page 486

"ApiTaskAssignment XML" on page 694

"UDF XML" on page 742

ApiTask: Add

```
Public Overrides Function Add(Optional ByRef sId As String = "") As Int32
```

Purpose

Add a new Task to Changepoint database by assigned Task id.

Parameters

Parameter (*=required)	Description
sId	Return parameter: Task id for the new Task.

Returns

0 = Success

Nonzero = Error

Remarks

The Task object properties must be filled with data.

Example

```
Dim myApi as New ApiTask()
With myApi
.Name="Richard Task2007"
.Project=New Identity("{}")
.WorkCode=New Identity("{}")

.CPConnection=myCon
End With
Dim retId as String=""
Dim ret as Int32= myApi.Add(retId)
```

Related information

"ApiTask" on page 672

ApiTask: Delete

```
Public Overrides Function Delete(Optional ByVal sId As String = "") As Int32
```

Purpose

Delete existing Task from Changepoint database.

Parameters

Parameter (*=required)	Description
sId	Return parameter: The ID of Task that will be deleted.

Returns

0 = Success

Nonzero = Error

Remarks

If a parameter is not provided, the function will delete the object itself and the function will return 0..

Example

```
Dim myApi as New ApiTask()
myPrj.CPConnection=myCon
Dim retId as String="{}"
Dim ret as Int32= myApi.Delete(retId)
```

Related information

"ApiTask" on page 672

ApiTask: Exists

```
Public Overrides Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Check if the Task exists.

Parameters

Parameter (*=required)	Description
sId	Id of Task that will be checked.

Returns

True if the task exists

Remarks

None

Example

```
Dim myApi as New ApiTask()
myApi.CPConnection=myCon
Dim retId as String="{}"
Dim ret as Boolean= myApi.Exists(retId)
```

Related information

"ApiTask" on page 672

ApiTask: GetBaselineHistory

```
Public Function GetBaselineHistory(ByVal sTaskId As String) As DataSet
```

Purpose

Retrieves Baseline History information for a task

Parameters

Parameter (*=required)	Description
*sTaskId	ID of the Task

Returns

A dataset containing all the fields in the DS_TaskBaseline view in the database.

Remarks

Contains the current baseline and all historical baselines for the task in descending order by createdon date.

Example

```
Dim myApi as New ApiTask( )
myApi.CPConnection = myCon
Dim ds as New Dataset
```

```
ds = myApi.GetBaselineHistory(sTaskId)
```

Related information

"ApiTask" on page 672

ApiTask: GetById

```
Public Overrides Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Retrieve Task object from Changepoint database.

Parameters

Parameter (*=required)	Description
sId	Id of Task that will be retrieved.

Returns

0 = Success

Nonzero = Error

Remarks

If a parameter is not provided the function will return an error message.

Example

```
Dim myApi as New ApiTask()
myApi.CPConnection=myCon
Dim tskId as String="{}"
Dim ret as Int32= myApi.GetById(tskId)
```

Related information

"ApiTask" on page 672

ApiTask: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As String
```

Purpose

Returns the TaskId based on the UDF Text field and value.

Parameters

Parameter (*=required)	Description
*sUDFField	UDF text field name (for example, Text1)
*sUDFValue	UDF text value

Returns

TaskId, or an empty string if nothing is found.

Remarks

None

Related information

"ApiTask" on page 672

ApiTask: GetList

```
Public Overloads Overrides Function GetList(Optional ByVal iRetRows As Int16 =
-1) As DataSet
```

Purpose

Retrieve the Task list from Changepoint database.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).

Returns

Return dataset of Task list.

Remarks

The dataset will include `CustomerId`, `EngagementId`, `Project ID`, `TaskId` and `TaskName`.

Example

```
Dim myApi as New ApiTask()
myApi.CPConnection=myCon
Dim ret as Dataset= myApi.GetList(-1)
```

Related information

"ApiTask" on page 672

ApiTask: GetUDF

```
Public Function GetUDF(Optional ByVal retOption As CPUDFReturnType =
CPUDFReturnType.OnlyValues) As String
```

Purpose

Retrieve UDF (configurable field) information for Task object.

Parameters

Parameter (*=required)	Description
retOption	Return options: • WithStructure = 0 – Returns the UDF values and structure • OnlyValues = 1 – Returns the UDF values • OnlyStructure = 2 – Returns the UDF structure • OnlyStructureAndOptions = 3 – Returns the UDF structure and options • WithStructureAndOptions = 4 – Returns the UDF structure, metadata tags, values and all the options for codes (except entity based and conditional codes)

Returns

UDF string in XML format.

Remarks

Always return UDF for current Task object.

Example

```
Dim myApi as New ApiTask()
myApi.CPConnection=myCon
Dim ret as String= myApi.GetUDF()
```

Related information

"ApiTask" on page 672

ApiTask: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal codeName As String, Optional ByVal
searchString As String = "") As String
```

Purpose

Retrieve UDF code option.

Parameters

Parameter (*=required)	Description
*codeName	Code name for retrieving options.
searchString	Returns the options that start with this string.

Returns

Code options string in XML format.

Remarks

All of the Task object properties must be filled.

Example

```
Dim myApi as New ApiTask()
myApi.CPConnection=myCon
Dim tskId as String="{}"
Dim ret as Int32= myApi.GetById(tskId)
If ret=0 then
  Dim retOptions as String= myApi.GetUDFCodeOptions("Code1")
End if
```

Related information

"ApiTask" on page 672

ApiTask: HasAssignments

```
Public Function HasAssignments(Optional ByVal sId As String = "") As Boolean
```

Purpose

Check if there has assignment under the Task.

Parameters

Parameter (*=required)	Description
sId	ID of the Task to check.

Returns

True if task has an assignment.

Remarks

None

Example

```
Dim myPrj as New ApiTask()
myPrj.CPConnection=myCon
Dim ret as Boolean=myPrj.HasAssignments ("{}")
```

Related information

"ApiTask" on page 672

ApiTask: SaveBaseline

```
Public Function SaveBaseline(ByVal sTaskId As String) As Int32
```

Purpose

Saves current PlannedStart, PlannedFinish, PlannedHours information for a task into the baseline fields

Parameters

Parameter (*=required)	Description
*sTaskId	ID of the Task

Returns

0 = Success

Nonzero = Error

Check the CPException.haserror for any errors.

Remarks

Current Planned information is saved to the Baseline fields in the task and a new record is created in the Baselines table.

Example

```
Dim myApi as New ApiTask( )
myApi.CPConnection = myCon
Dim iRet as Int32 = -1
iRet = myApi.SaveBaseline (sTaskId)
If Not myApi.CPException.haserror Then
   ...
Else
   ...
End if
```

Related information

"ApiTask" on page 672

ApiTask: SaveUDF

```
Public Function SaveUDF(Optional ByVal sUDF As String = "") As Int32
```

Purpose

Save (Insert/Update) UDF information for Task object.

Parameters

Parameter (*=required)	Description
sUDF	UDF string in XML format.

Returns

0 = Success

Nonzero = Error

Remarks

Always save UDF for current Task object.

Example

```
Dim myApi as New ApiTask()
myApi.CPConnection=myCon
Dim strXML as string ="<root></root>"
Dim ret as Int32= myApi.SaveUDF(strXML)
```

Related information

"ApiTask" on page 672

Related information

"ApiTask" on page 672

ApiTask: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Update Task object to Changepoint database

Parameters

None

Returns

0 = Success

Nonzero = Error

Remarks

The Task object will be updated with all the properties assigned. Please note the Baseline fields are no longer saved as part of the Update method. Please refer to the new SaveBaselines( ) method.

Example

```
Dim myApi as New ApiTask()
With myApi
.TaskId="{}"
.Name="Richard Task2007"
.Project=New Identity("{}")
.WorkCode=New Identity("{}")
   ...
.CPConnection=myCon
End With
Dim ret as Int32= myApi.Update()
```

Related information

"ApiTask" on page 672

ApiTaskAssignment

The ApiTaskAssignment object allows users to add, retrieve, update or delete TaskAssignment information within the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiTaskAssignment

Methods

Properties

Property (*=required)	Type	Description
AllowTALocOverride	Boolean	When set to true, allows the TaskAssignment location to be overriden.
AllowTAWorkCodeOverride	Boolean	When set to true, allows the TaskAssignment work code to be overriden.
BaselineStart	DateTime	Baseline start of TaskAssignment.
BaselineFinish	DateTime	Baseline finish of TaskAssignment.
BaselineHours	Double	Baseline hours of TaskAssignment.
BaselineResourceAssigned	Identity	Identifier of the assigned resource
BillingRole	Identity	Identifier of the billing role. This billing role should already have been defined as billing rate for the contract. If billing role is required, make sure that this billing role is associated with the resource.

Property (*=required)	Type	Description
*BypassMetadataCheck	CPMetadataCheck	Determines the level of MetaData checking when adding or updating data. (default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
*CPConnection	ApiConnection	Write-only. Connection object that must be assigned before any action to database.
CPException	ApiException	Read-only. Exception object handles and traces the exception information.
CreatedBy	Signature	Resource who created the record and when it was created.
Deleted	Boolean	Flag that indicates if the object has been deleted
*Entity	CPEntity	Read-only. Changepoint entity that the object represents.
Functions	Identity	Identifier of the function.
Locked	Boolean	When set to true, the task assignment is locked.
ManagerComment	String	Comments from manager.
mVersion	String	Read-only. Current version number for internal version control.
*PlannedFinish	DateTime	Planned finish date.
*PlannedHours	Double	Planned hours.

Property (*=required)	Type	Description
*PlannedStart	DateTime	Planned start date.
*Resource	Identity	Identifier of the resource.
*SoftBooked	Boolean	When set to true, the task assignment is soft booked.
*StatusNotRequired	Boolean	Status not required.
sxmlUDF	String	UDF string in XML format.
*Task	Identity	Identifier of the task.
*TaskAssignmentId	String	TaskAssignment ID. Mandatory in update and delete.
UpdatedBy	Signature	Resource who updated the record and when it was updated.
*WorkCode	Identity	Identifier of the work code.
*WorkCodeCategory	Identity	Identifier of the work code category.
*WorkLocation	Identity	Identifier of the work location.
*WorkLocationGroup	Identity	Identifier of the work location group.

Related information

"ApiTaskAssignment XML" on page 694

"ApiTask" on page 672

"ApiProject" on page 480

ApiTaskAssignment XML

```
<root>
   <project>
   ...
      <tasks>
         <task>
         ...
            <taskassignments>
               <assignment>
```

```
<bypassmetadatacheck>checkall</bypassmetadatacheck>
<createdbyid />
<updatedbyid />
<taskassignmentid/>
<allowtalocoverride>false</allowtalocoverride>
<allowtaworkcodeoverride>false</allowtaworkcodeoverride>
<billingrole>
   <id />
   <name />
   <alternatename />
</billingrole>
<entity />
<functions>
   <id />
   <name />
   <alternatename />
</functions>
<locked>false</locked>
<managercomment />
<plannedfinish />
<plannedhours />
<plannedstart />
<resource>
   <id />
   <name />
   <alternatename />
   <firstname />
   <lastname />
   <userdefinedid />
</resource>
<softbooked>false</softbooked>
<statusnotrequired>false</statusnotrequired>
<sxmludf />
<task>
   <id />
   <name />
   <alternatename />
</task>
<workcode>
   <id />
   <name />
   <alternatename />
   <userdefinedid />
</workcode>
<workcodecategory>
   <id />
   <name />
   <alternatename />
```

```
                    <userdefinedid />
                 </workcodecategory>
                 <worklocation>
                    <id />
                    <name />
                    <alternatename />
                    <userdefinedid />
                 </worklocation>
                 <worklocationgroup>
                    <id />
                    <name />
                    <alternatename />
                    <userdefinedid />
                 </worklocationgroup>
                 <iUpdate>1</iUpdate>
                 <ignorewarning>false<ignorewarning>
              </assignment>
           </taskassignments>
        </task>
     </tasks>
  </project>
</root>
```

Comments

The XML for the ApiTaskAssignment object is included in the XML for the ApiTask object, which is included in the XML for the ApiProject object. The following container elements are mandatory:

```
<root>
   <project>
      <tasks>
         <task>
            <taskassignments>
               <assignment>
                  ...
               </assignment>
            </taskassignments>
         </task>
      </tasks>
   </project>
</root>
```

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

For the use of the `<iUpdate>` flag in ApiProject XML, ApiTask XML, and APiTaskAssignment XML, see "ApiProject: CreateByXML" on page 492.

Example

```
<root>
   <Project>
   ...
      <Tasks>
         <Task>
         ...
            <TaskAssignments>
               <Assignment>
                  <TaskAssignmentId></TaskAssignmentId>
                  <Resource><id>BDF1C72F-ADED-49EE-A708-
5D44C7AD9A33</id><name/><alternatename/></Resource>
                  <BillingRole><id>00000000-0000-0000-0000-
000000000000</id><name/><alternatename/></BillingRole>
                  <Locked>0</Locked>
                  <SoftBooked>1<SoftBooked/>
                  <PlannedStart>2005/9/14</PlannedStart>
                  <PlannedFinish>2005/9/30</PlannedFinish>
                  <PlannedHours>51</PlannedHours>
                  <ManagerComment/>
                  <StatusNotRequired>0</ StatusNotRequired>
                  <WorkLocationGroup><id>333A206C-513E-4CAE-841A-
EA73A82E8E81</id><name/><alternatename/></WorkLocationGroup>
                  <WorkLocation><id>0409A21B-EFB9-42C4-ADB3-
617279F524EF</id><name/><alternatename/></WorkLocation>
                  <WorkCodeCategory><id>AA9C83C1-6E1E-4974-9DEC-
7C554CC429D2</id><name/><alternatename/></WorkCodeCategoryId>
                  <WorkCode><id>8A4BEED2-A035-4407-948D-
248681472747</id><name/><alternatename/></WorkCode>
                  <iUpdate>1</iUpdate>
               </Assignment>
            </TaskAssignments>
         </Task>
      </Tasks>
   </Project>
</root>
```

Related information

"ApiProject XML" on page 486

"ApiTask XML" on page 676

"ApiTaskAssignment" on page 691

ApiTaskAssignment: Add

```
Public Overrides Function Add(Optional ByRef sId As String = "") As Int32
```

Purpose

Add a new TaskAssignment to the Changepoint database.

Parameters

Parameter (*=required)	Description
sId	Return parameter: TaskAssignment id for new TaskAssignment.

Returns

0 = Success

-15 = Success with warning

Nonzero = Error

Remarks

The TaskAssignment object properties must be filled with at least the mandatory data.

When "-15" is returned, it indicates that task planned hours is updated with the total planned hours of task assignments. This is necessary step for the API to update the task planned hours, so this should be treated as notification.

Example

```
Dim myApi as New ApiTaskAssignment()
With myApi
.Resource=New Identity("{}","Richard")
.Project=New Identity("{}")
.Task=New Identity("{}")
.WorkCode=New Identity("{}")
...
.CPConnection=myCon
End With
Dim retId as String=""
Dim ret as Int32= myApi.Add(retId)
```

Related information

"ApiTaskAssignment" on page 691

ApiTaskAssignment: Delete

```
Public Overrides Function Delete(Optional ByVal sId As String = "") As Int32
```

Purpose

Delete existing TaskAssignment from Changepoint database.

Parameters

Parameter (*=required)	Description
sId	Id of TaskAssignment that will be deleted.

Returns

0 = Success

Nonzero = Error

Remarks

If a parameter is not provided the function will delete the object itself and return 0.

Example

```
Dim myApi as New ApiTaskAssignment()
myPrj.CPConnection=myCon
Dim retId as String="{}"
Dim ret as Int32= myApi.Delete(retId)
```

Related information

"ApiTaskAssignment" on page 691

ApiTaskAssignment: Exists

```
Public Overrides Function Exists(Optional ByVal sId As String = "") As Boolean
```

Purpose

Check if the TaskAssignment exists.

Parameters

Parameter (*=required)	Description
sId	ID of TaskAssignment that will be checked.

Returns

True if the task assignment exists.

Remarks

If a parameter is not provided the function will return an error message.

Example

```
Dim myApi as New ApiTaskAssignment()
myApi.CPConnection=myCon
Dim retId as String="{}"
Dim ret as Boolean= myApi.Exists(retId)
```

Related information

"ApiTaskAssignment" on page 691

ApiTaskAssignment: GetBaselineHistory

```
Public Function GetBaselineHistory(ByVal sTaskAssignmentId As String) As
DataSet
```

Purpose

Retrieves Baseline History information for a TaskAssignment

Parameters

Parameter (*=required)	Description
*sTaskAssignmentId	ID of the TaskAssignment

Returns

A dataset containing all the fields in the DS_TaskAssignmentBaseline view in the database.

Remarks

Contains the current baseline and all historical baselines for the task in descending order by createdon date.

Example

```
Dim myApi as New ApiTaskAssignment ( )
myApi.CPConnection = myCon
Dim ds as New Dataset
ds = myApi.GetBaselineHistory(sTaskAssignmentId)
```

Related information

"ApiTaskAssignment" on page 691

ApiTaskAssignment: GetById

```
Public Overrides Function GetById(Optional ByVal sId As String = "") As Int32
```

Purpose

Retrieve TaskAssignment object from Changepoint database.

Parameters

Parameter (*=required)	Description
sId	ID of TaskAssignment that will be retrieved.

Returns

0 = Success

Nonzero = Error

Remarks

If a parameter is not provided the function will return an error message.

Example

```
Dim myApi as New ApiTaskAssignment()
myApi.CPConnection=myCon
Dim tskId as String="{}"
Dim ret as Int32= myApi.GetById(tskId)
```

Related information

"ApiTaskAssignment" on page 691

ApiTaskAssignment: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As String
```

Purpose

Returns the TaskAssignmentId based on the UDF Text field and value.

Parameters

Parameter (*=required)	Description
*sUDFField	UDF text field name (for example, Text1)
*sUDFValue	UDF text value

Returns

TaskAssignmentId, or an empty string if nothing is found.

Remarks

None

Related information

"ApiTaskAssignment" on page 691

ApiTaskAssignment: GetList

```
Public Overloads Overrides Function GetList(Optional ByVal iRetRows As Int16 =
-1) As DataSet
```

Purpose

Retrieve the TaskAssignment list from Changepoint database.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).

Returns

A dataset of the TaskAssignment list.

Remarks

The dataset will include `TaskId`, `TaskName`, `ResourceId`, `ResourceName` and `TaskAssignmentId`.

Example

```
Dim myApi as New ApiTaskAssignment()
myApi.CPConnection=myCon
Dim ret as Dataset= myApi.GetList(-1)
```

Related information

"ApiTaskAssignment" on page 691

ApiTaskAssignment: GetUDF

```
Public Function GetUDF(Optional ByVal retOption As CPUDFReturnType =
CPUDFReturnType.OnlyValues) As String
```

Purpose

Retrieve UDF (configurable field) information for TaskAssignment object.

Parameters

Parameter (*=required)	Description
retOption	Return options: • WithStructure = 0 – Returns the UDF values and structure • OnlyValues = 1 – Returns the UDF values • OnlyStructure = 2 – Returns the UDF structure • OnlyStructureAndOptions = 3 – Returns the UDF structure and options • WithStructureAndOptions = 4 – Returns the UDF structure, metadata tags, values and all the options for codes (except entity based and conditional codes)

Returns

UDF string in XML format.

Remarks

Always return UDF for current TaskAssignment object.

Example

```
Dim myApi as New ApiTaskAssignment()
myApi.CPConnection=myCon
Dim ret as String= myApi.GetUDF()
```

Related information

"ApiTaskAssignment" on page 691

ApiTaskAssignment: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal codeName As String, Optional ByVal
searchString As String = "") As String
```

Purpose

Retrieve UDF code option.

Parameters

Parameter (*=required)	Description
*codeName	Code name for retrieving options.
searchString	Returns the options that start with this string.

Returns

Code options string in XML format.

Remarks

All of the TaskAssignment object properties must be filled.

Example

```
Dim myApi as New ApiTaskAssignment()
myApi.CPConnection=myCon
Dim tskId as String="{}"
Dim ret as Int32= myApi.GetById(tskId)
If ret=0 then
  Dim retOptions as String= myApi.GetUDFCodeOptions("Code1")
End if
```

Related information

"ApiTaskAssignment" on page 691

ApiTaskAssignment: SaveUDF

```
Public Function SaveUDF(Optional ByVal sUDF As String = "") As Int32
```

Purpose

Save (Insert/Update) UDF information for TaskAssignment object.

Parameters

Parameter (*=required)	Description
sUDF	UDF string in XML format.

Returns

0 = Success

Nonzero = Error

Remarks

Always save UDF for current TaskAssignment object.

Example

```
Dim myApi as New ApiTaskAssignment()
myApi.CPConnection=myCon
Dim strXML as string ="<root></root>"
Dim ret as Int32= myApi.SaveUDF(strXML)
```

Related information

"ApiTaskAssignment" on page 691

ApiTaskAssignment: StatusTask

```
Public Function StatusTask(ByVal sTaskAssignmentId As String, ByVal
dtForecastStart As DateTime, ByVal dtForecastFinish As DateTime, Optional
ByVal dRemainingHours As Double = -1.0, Optional ByVal dPercentComplete as
Double = -1.0, ByVal sResComments As String, ByVal sMgrComments As String) As
Boolean
```

Purpose

Provides ability to status a task.

Parameters

Parameter (*=required)	Description
*sTaskAssignmentId	Task assignment ID for statusing.
dRemainingHours	Remaining hours to be set for the task.
*dtForecastStart	Forecast start date to be set for the task.
*dtForecastFinish	Forecast finish date to be set for the task.

Parameter (*=required)	Description
dPercentComplete	Percent complete value to be set for the task.
*sResComments	Resource comments
*sMgrComments	Manager comments

Returns

True = Task was statused

False = Task was not statused

Remarks

The parameters dRemainingHours and dPercentComplete are optional and default to -1, but one or the other is calculated depending on what value is passed in. When -1 is passed in for dRemainingHours then the Percent complete will be calculated. If -1 is passed in for dPercentComplete then Remaining hours will be calculated.

If both Remaining hours and Percent complete are provided, the Remaining hours will be honored.

The task can only be statused by the resource assigned to the task or the manager or co-manager of the project.

Manager comments can only be set by the project manager.

Example

```
Dim oTA as ApiTaskAssignment = New ApiTaskAssignment
Dim sId as String = "{}"
Dim dRHours as Double = 40.0
Dim dRHours as DateTime = "5/18/2007"
Dim dtFFinish as DateTime = "6/19/2007"
Dim dPComplete as Double = 25.0
Dim sRComments as String = "Resource comments"
Dim sMComments as String = "Manager comments"
Dim bRet as Boolean

bRet = oTA.StatusTask(sId, dRHours, dFStart, dFFinish, dPComplete,
sRComments, sMComments )

If Not oTA.CPException.HasError Then
'User Code
```

```
Else
'Handle Error
End If
```

Related information

"ApiTaskAssignment" on page 691

ApiTaskAssignment: Update

```
Public Overrides Function Update() As Int32
```

Purpose

Update TaskAssignment object to Changepoint database

Parameters

None

Returns

0 = Success

-15 = Success with warning

Nonzero = Error

Remarks

When "-15" is returned, it indicates that task planned hours is updated with the total planned hours of task assignments. This is a necessary step for the API to update the task planned hours, so this should be treated as notification

The TaskAssignment object will be updated with all the properties assigned.

Example

```
Dim myApi as New ApiTaskAssignment()
With myApi
.TaskAssignmentId="{}"
.Resource=New Identity("{}","David")
.WorkCode=New Identity("{}")

.CPConnection=myCon
End With
Dim ret as Int32= myApi.Update()
```

Related information

"ApiTaskAssignment" on page 691

ApiTime

The ApiTime object allows you to submit, approve, and reject time information of all types, and to add, retrieve, and update project time for resources within the Changepoint database.

Namespace

Changepoint.ChangepointAPI2.ApiTime

Methods

Properties

Property (*=required)	Type	Description
AdjustedOvertimeHours	Decimal	Adjusted overtime hours
AdjustedRegularHours	Decimal	Adjusted regular hours

Property (*=required)	Type	Description
AdjustmentStatus	Boolean	Adjustment status
AdjustmentTimeParent	String	Read-only. Adjustment time parent ID.
AdjustmentTimeStatus	String	Adjustment time status
*BookedByResourceId	String	Read-only. The resource ID who booked time.
ByPassLagDayCheck	Boolean	When set to true, PayrollLag is not validated.
*BypassMetadataCheck	CPMetaDataCheck	Determines the level of MetaData checking when adding or updating data. (default is to Check All fields). Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2
Comments	String	Used by update only when FedAudit is set to true.
*CPConnection	ApiConnection	Connection to the Changepoint database
CreatedBy	Signature	Creator of the record and date when it was created.
CreatedOn	DateTime	Date and time that the record was created.
Customer	Identity	Read-only. Identifier of the customer
Deleted	Boolean	When set to true, the time record has been deleted.
Description	String	Description

Property (*=required)	Type	Description
Editable	Boolean	Read-only. When set to true, the time record can be edited.
Engagement	Identity	Read-only. Identifier of the contract.
Entity	CPEntity	Read-only. Changepoint entity that the object represents.
FedAudit	Boolean	Read-only. When set to true, auditing is enabled.
HistoryComments	String	Required field if federal audit is enabled.
mVersion	String	Read-only. Current version number for internal version control.
OvertimeHours	Decimal	Overtime hours
Project	Identity	Read-only. Identifier of the project.
*RegularHours	Decimal	Regular hours
*Resource	Identity	Read-only. Changepoint resource ID.
*StartTime	Date	The start time
*Task	Identity	Read-only. Changepoint task ID.
TimeCode1	Identity	Time Code1
TimeCode2	Identity	Time Code2
TimeCode3	Identity	Time Code3
*TimeDate	DateTime	Date and time
*TimeId	String	Read-only. Changepoint time ID.
UpdatedBy	Signature	Updater of the record and date when the record was updated.

Property (*=required)	Type	Description
UpdatedOn	DateTime	Date that the record was last updated.
*WorkCodeCategory	Identity	Work code category ID
*WorkCode	Identity	Work code ID
*WorkLocationGroup	Identity	Work location group ID
*WorkLocation	Identity	Work location ID

ApiTime XML

```
<root>
  <time>
    <bypassmetadatacheck />
    <comments />
    <createdbyid />
    <createdon />
    <updatedbyid />
    <updatedon />
    <timeid />
    <adjustedovertimehours />
    <adjustedregularhours />
    <adjustmentstatus />
    <adjustmenttimeparent />
    <adjustmenttimestatus />
    <billable />
    <bookedbyresource>
      <id />
      <name />
      <alternatename />
      <firstname />
      <lastname />
      <userdefinedid />
    </bookedbyresource>
    <customer>
      <id />
      <name />
      <alternatename />
      <userdefinedid />
    </customer>
    <description />
    <engagement>
      <id />
      <name />
      <alternatename />
      <userdefinedid />
    </engagement>
    <fedaudit>false</fedaudit>
    <historycomments />
    <overtimehours />
```

```
<project>
  <id />
  <name />
  <alternatename />
  <userdefinedid />
</project>
<regularhours />
<resource>
  <id />
  <name />
  <alternatename />
  <firstname />
  <lastname />
  <userdefinedid />
</resource>
<starttime />
<task>
  <id />
  <name />
  <alternatename />
</task>
<timecode1>
  <id />
  <name />
  <alternatename />
</timecode1>
<timecode2>
  <id />
  <name />
  <alternatename />
</timecode2>
<timecode3>
  <id />
  <name />
  <alternatename />
</timecode3>
<timedate />
<editable />
<workcode>
  <id />
  <name />
  <alternatename />
  <userdefinedid />
</workcode>
<workcodecategory>
  <id />
  <name />
  <alternatename />
  <userdefinedid />
</workcodecategory>
<worklocation>
  <id />
  <name />
  <alternatename />
  <userdefinedid />
</worklocation>
```

```
    <worklocationgroup>
      <id />
      <name />
      <alternatename />
      <userdefinedid />
    </worklocationgroup>
  </time>
</root>
```

Comments

Some of the XML elements have child nodes (id, name, alternatename, etc.). For more information, see "About APIs and XML" on page 34.

Related information

"ApiTime" on page 709

ApiTime: Add

```
Public Function Add(ByVal oTime As ApiTime) As String
```

Purpose

Adds a new time record into Changepoint.

Parameters

Parameter (*=required)	Description
*oTime	ApiTime object, which contains the time information that will be added.

Returns

TimeId, which was added if successful.

Remarks

Return empty when error occurred. If returned empty, oTime is null, no time is to be added. The error message is written to the log file.

Example

```
Dim myTime As New ApiTime
Dim mTime As New ApiTime
```

```
Dim sRet As String
myTime.CPConnection = myCon

With mTime
   .AdjustedOvertimeHours = 0
   .AdjustedRegularHours = 0
   .AdjustmentStatus = False
   .BookedByResourceId = "{7E4F8CE9-363B-11D4-8AB1-0001023D3221}"
   .Description = "Case 1"
   .OvertimeHours = 2
   .ResourceId = "{7E4F8CE9-363B-11D4-8AB1-0001023D3221}"
   .RegularHours = 8
   .StartTime = "5/18/2007"
   .TaskId = "{61396453-4DBF-4FD5-9F46-299BD2374A35}"
   .TimeDate = "5/18/2007"
   .WorkCodeCategoryId = "{AA9C83C1-6E1E-4974-9DEC-7C554CC429D2}"
   .WorkCodeId = "{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}"
   .WorkLocationGroupID = "{6AAF1A2D-2535-40E3-8D03-07801AE85E41}"
   .WorkLocationID = "{0175FA25-7CAA-4DC3-BB95-62A37A36134F}"
End With

sRet = myTime.Add(mTime)
```

Related information

"ApiTime" on page 709

ApiTime: ApproveRejectTimes

```
Public Function ApproveRejectTimes(ByVal sResourceId As String, ByVal sTaskId
As String, ByVal daStartDate As Date, ByVal daEndDate As Date, ByVal bAction
As Boolean, Optional ByVal sReason As String) As Int32
```

Purpose

Approve or reject time records which StartTime falls between the dates specified in the daStartDate and daEndDate parameters.

Parameters

Parameter (*=required)	Description
*sResourceId	Changepoint assigned Resource ID. Cannot be empty.
*sTaskId	Task ID in Changepoint. Cannot be empty.
*daStartDate	Start date. Cannot be empty.
*daEndDate	End date. Cannot be empty.
*bAction	• 0 for rejection • 1 for approval.
sReason	Reason for action. sReason cannot be empty if bAction equals zero.

Returns

0 = Success

Nonzero = Error

Remarks

Error message for the returned error number is documented in the log file.

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, Optional ByRef sId As String
= "") As Int32
```

Purpose

Create a time record by using an XML string of the Time object in Changepoint.

Parameters

Parameter (*=required)	Description
*sXML	A new time XML string which can be passed based on the Time XML schema
sId	ByRef parameter that will return the new TimeId after the time has been added.

Returns

0 = Success

Nonzero = Error

Remarks

The ApiTime XML structure can be obtained by the GetXMLStructure method.

The ByPassMetadataCheck switch will stop any metadata validation in Time.

Example

Not available

Related information

"ApiTime" on page 709

"ApiTime XML" on page 713

the "ApiTime: GetXMLStructure" section on page 730

ApiTime: Delete

```
Public Overrides Function Delete(Optional ByVal sId As String = "") As Integer
```

Purpose

Deletes the time by the time ID.

Parameters

Parameter (*=required)	Description
sId	ID of the time to be deleted.

Returns

0 = Success

Nonzero = Error

Remarks

If sId is not provided, the value of the property TimeId is used.

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: Exists

```
Public Overrides Function Exists(Optional sId As String = "") As Boolean
```

Purpose

Verifies whether the time exists in the Changepoint database.

Parameters

Parameter (*=required)	Description
sId	Time ID of the time to verify

Returns

True if the time exists, else False.

Remarks

None

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: GetById

```
Public Overrides Function GetById(Optional ByVal sId As String = "") As
Integer
```

Purpose

Fills the object with data related to the specified TimeId passed in the parameter.

Parameters

Parameter (*=required)	Description
sId	Id of time record that will be retrieved.

Returns

0 = Success

Nonzero = Error

Remarks

If optional parameter sId is not provided, the property TimeId is used.

Example

Not available

Related information

"ApiTime XML" on page 713

ApiTime: GetByXML

```
Public Function GetByXML(Optional ByVal sXML As String = "", Optional ByVal
sTimeId As String = "") As String
```

Purpose

Takes the XML string passed in the sXML parameter and returns the string filled with data for the time specified in the sTimeId parameter.

Parameters

Parameter (*=required)	Description
sXML	XML string of the Time object, with the fields specified
sTimeId	Time ID • If no Time ID is passed in, the XML string (sXML) is examined • if there is no Time ID in sXML then the object'sTime ID is selected. • If there still is no valid TimeId, the method returns an error.

Returns

An XML string mirroring sXML with data inserted, or the entire XML of the Time object including data.

Remarks

If sXML = "" then the XML string provided by GetXMLStructure is used.

Example

Not available

Related information

"ApiTime" on page 709

"ApiTime XML" on page 713

ApiTime: GetList

```
Public Overrides Function GetList(Optional ByVal iRetRows As Short = -1) As
System.Data.DataSet
```

Purpose

Returns a lits of all time by login user ID.

Parameters

Parameter (*=required)	Description
iRetRows	The number of records to be returned. Default = -1 (return all).

Returns

Time records in a dataset upon success and nothing in a dataset upon error.

Returns all rows with no parameter.

Remarks

Returns the columns in the time table.

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: GetTasksWithinPeriod

```
Public Function GetTasksWithinPeriod(ByVal daStartDate As Date, ByVal
daEndDate As Date) As Dataset
```

Purpose

Retrieve task information in time records, where the start time falls between the dates specified in the daStartDate and daEndDate parameters.

Parameters

Parameter (*=required)	Description
*daStartDate	Start date of the period
*daEndDate	End date of the period

Returns

A dataset of task information in time records.

Remarks

The fields include: TaskId, TaskName

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: GetTimeByRes

```
Public Function GetTimeByRes(ByVal sResourceId As String, ByVal daStartDate As
Date, ByVal daEndDate As Date) As Dataset
```

Purpose

Retrieve time records, where the start time falls between the dates specified in the daStartDate and daEndDate parameters for the target resource.

Parameters

Parameter (*=required)	Description
*sResourceId	Changepoint assigned Resource ID
*daStartDate	Start date.
*daEndDate	End date.

Returns

A dataset of time information.

Remarks

The fields include: TimeId, TaskId, ResourceId, Description, etc.

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: GetTimeByTask

```
Public Function GetTimeByTask(ByVal sTaskId As String, ByVal daStartDate As
Date, ByVal daEndDate As Date) As Dataset
```

Purpose

Retrieve time records, where the start time falls between the dates specified in the daStartDate and daEndDate parameters for the target task.

Parameters

Parameter (*=required)	Description
*sTaskId	Changepoint created task ID.
*daStartDate	Start date.
*daEndDate	End date.

Returns

A dataset of time information.

Remarks

The fields include: TimeId, TaskId, ResourceId, Description, etc.

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: GetTimeCodes

```
Public Function GetTimeCodes(ByVal sGlobalWorkgroupId As String, ByVal
sWorkgroupId As String, ByVal sCodeType As String) As Dataset
```

Purpose

Retrieve time code information for a workgroup.

Parameters

Parameter (*=required)	Description
*sGlobalWorkgroupId	GlobalWorkgroup ID.
*sWorkgroupId	Workgroup ID.
*sCodeType	Required. Code Type: • CODE1 • CODE2 • CODE3

Returns

A dataset of time code information.

Remarks

None

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: GetTimeIdsWithinPeriod

```
Public Function GetTimeIdsWithinPeriod(ByVal daStartDate As Date, ByVal
daEndDate As Date) As Dataset
```

Purpose

Retrieve time records, where the start time falls between the dates specified in the daStartDate and daEndDate parameters.

Parameters

Parameter (*=required)	Description
*daStartDate	Start date
*daEndDate	End date

Returns

A dataset of TimeId.

Remarks

None

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: GetWorkCodeCategories

```
Public Function GetWorkCodeCategories(ByVal sResourceId As String, ByVal
sTaskId As String, Optional ByVal sSearchString As String = "") As DataSet
```

Purpose

Returns a lists of work code categories based on the project or task level settings.

Parameters

Parameter (*=required)	Description
sResourceId	Resource ID
sTaskId	Task ID
sSearchString	Search string

Returns

A dataset with two columns (WorkCodeCategoryId, Name)

Remarks

None

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: GetWorkCodes

```
Public Function GetWorkCodes(ByVal sResourceId As String, ByVal sTaskId As
String, ByVal sWorkCodeCategoryId As String, Optional ByVal sSearchString As
String = "") As DataSet
```

Purpose

Returns a list of the work codes based on project or task level settings.

Parameters

Parameter (*=required)	Description
*sResourceID	Resource ID
*sTaskId	Task ID
*sWorkCodeCategoryId	Work code category ID
sSearchString	Search string

Returns

A dataset with two columns (WorkCodeId, Name)

Remarks

None

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: GetWorkLocationGroups

```
Public Function GetWorkLocationGroups(ByVal sResourceId As String, ByVal
sTaskId As String, Optional ByVal sSearchString As String = "") As DataSet
```

Purpose

Returns a list of work location groups based on the project or task level settings.

Parameters

Parameter (*=required)	Description
*sResourceId	Resource ID
*sTaskId	Task ID
sSearchString	Search string

Returns

A dataset with two columns (WorkLocationGroupId, Name)

Remarks

None

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: GetWorkLocations

```
Public Function GetWorkLocations(ByVal sResourceId As String, ByVal sTaskId As
String, ByVal sWorkLocationGroupId As String, Optional ByVal sSearchString As
String = "") As DataSet
```

Purpose

Returns a list of work locations based on the project level settings.

Parameters

Parameter (*=required)	Description
*sResourceId	Resource ID
sTaskId	Task ID
*sWorkLocationGroupId	Work location group ID
sSearchString	Search string

Returns

A dataset with two columns (WorkLocationId, Name)

Remarks

None

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: GetXMLStructure

```
Public Function GetXMLStructure() As String
```

Purpose

Return the XML structure of the ApiTime object

Parameters

None

Returns

An XML string of the ApiTime object

Remarks

None

Example

Not available

Related information

"ApiTask" on page 672

"ApiTime XML" on page 713

ApiTime: SubmitTimes

```
Public Function SubmitTimes(ByVal sResourceId As String, ByVal sTaskId As
String, ByVal daStartDate As Date, ByVal daEndDate As Date) As Int32
```

Purpose

Submit time, request time and standard time where the start time falls between the dates specified in the daStartDate and daEndDate parameters for a target resource.

Parameters

Parameter (*=required)	Description
*sResourceId	Changepoint assigned Resource ID. Cannot be empty.
*sTaskId	Task ID in Changepoint.This is an unused parameter and should be defaulted to a null ID, e.g. 00000000-0000-0000-0000-000000000000.
*daStartDate	Start date.
*daEndDate	End date.

Returns

0 = Success

Nonzero = Error

Remarks

None

Example

Not available

Related information

"ApiTime" on page 709

ApiTime: Update

```
Public Function Update(ByVal oTime As ApiTime) As Int32
```

Purpose

Update the general information of a time record.

Parameters

Parameter (*=required)	Description
*oTime	ApiTime object, which contains time information that will be updated.

Returns

0 = Success

Nonzero = Error

Remarks

Error message of the returned error number is documented in the log file.

Example

```
Dim myTime As New ApiTime
Dim mTime As New ApiTime
Dim iRet As Int32

myTime.CPConnection = myCon
mTime = myTime.GetTimeById("{A0A6F42C-B38D-4C45-AC6F-72EB638ADB62}")
```

```
With mTime
   .OvertimeHours = 4
End With

myTime.CPConnection =myCon
iRet = myTime.Update(mTime)
```

Related information

"ApiTime" on page 709

ApiTime: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, Optional ByVal sId As String
= "") As Int32
```

Purpose

Update a time record using an XML string of the time object in Changepoint.

Parameters

Parameter (*=required)	Description
*sXML	A time XML string with updated fields.
sTimeId	ID of the time record being updated.

Returns

0 = Success

Nonzero = Error

Remarks

The method uses the following sequence to find the Time ID:

1. If the sTimeId parameter is passed in, the method uses this value for the Time ID.
2. If this fails, the method attempts to extract the Time ID from <timeid> in the XML.
3. If this fails, the Time ID is taken from the object properties.

Example

Not available

Related information

"ApiTime" on page 709

"ApiTime XML" on page 713

ApiUser

The ApiUser object is for holding login user information.

Namespace

Changepoint.ChangepointAPI2.ApiUser

Methods

None

Properties

Property (*=required)	Type	Description
AccessToken	String	The token string that is used by web service login.
GlobalWorkgroupId	String	GlobalWorkgroupId of login user in Changepoint.
LoginAttempts	Byte	Read-only. Number of failed API login attempts.
LoginId	String	The Changepoint LoginId (email address of resource).
LoginPassword	String	Write-only. The password associated with LoginId.
LoginStatus	Boolean	Read-only. Indicates whether the user is signed in.

Property (*=required)	Type	Description
LoginTime	DateTime	Read-only. The date and time that the user signed in.
LoginType	CPLoginType	The Login Type: CPLoginType.SQL=0
Name	String	Login user name in Changepoint.
TokenExpiry	Int32	Token expiry time span in seconds.
TokenExpiryTime	DateTime	Token expiry time, it is an absolute time.
UserId	String	Login user ResourceId in Changepoint database
WorkgroupId	String	WorkgroupId of login user in Changepoint.

Identity

The Identity object is designed for integrating Changepoint Entity ID and Name information.

Namespace

Changepoint.BusinessBase.Identity

Methods

Properties

Property (*=required)	Type	Description
AlternateName	String	Alternate Name of Identity object.
Id	String	Identifies the Identity object.
Name	String	Name of Identity object.

Identity: New

```
Public Function New ()
```

```
Public Function New(ByVal Id As String)

Public Function New(ByVal Id As String, ByVal name As String)

Public Function New(ByVal Id As String, ByVal name As String, ByVal
alternateName As String)
```

Purpose

Constructors of Identity object.

Parameters

Parameter (*=required)	Description
*Id	ID of the Identity object.
*name	Name of the Identity object.
*alternateName	Alternate Name of the Identity object.

Remarks

None

Example

```
Dim PrjIdentity as New Identity("{}", "ProjectA", "P1507")
```

Related information

"Identity" on page 735

Signature

The Signature object is typically designed for integrating Changepoint Identity with time stamp, like CreatedBy or UpdatedBy.

Namespace

Changepoint.BusinessBase.Signature

Methods

Properties

Property (*=required)	Type	Description
ActionDate	DateTime	Time stamp.
ActionPerson	Identity	Identifier of the resource

Signature: New

```
Public Function New()

Public Function New(ByVal id As String)

Public Function New(ByVal id As String, ByVal actionDate As DateTime)

Public Function New(ByVal id As String, ByVal name As String)

Public Function New(ByVal id As String, ByVal name As String, ByVal actionDate
As DateTime)

Public Function New(ByVal id As String, ByVal name As String, ByVal
alternateName As String)

Public Function New(ByVal id As String, ByVal name As String, ByVal
alternateName As String, ByVal actionDate As DateTime)
```

Purpose

Constructors of Signature object.

Parameters

Parameter (*=required)	Description
*Id	ID of the Identity object.
*name	Name of the Identity object.
*alternateName	Alternate Name of the Identity object.
*actionDate	Time stamp.

Remarks

None

Example

Dim createdBy as New Signature("{}", "ApiUser", "2007-5-6")

Related information

"Signature" on page 736

Enumeration

Contains predefined constant bytes. For more information, see the following:

CPDuplicateType Enumeration

Contains predefined constant bytes that indicate the metadata duplicate checking status for the text field.

Namespace

Changepoint.UIBase.CPDuplicateType

Enumeration

```
NotCheck = 0
WarnIfDuplicate = 1
RejectIfDuplicate = 2
WarnIfSimilar = 3
```

Related information

"Enumeration" on page 738

CPEntity Enumeration

Contains predefined constant bytes that represent each Changepoint entity.

Namespace

Changepoint.BusinessBase.CPEntity

Enumeration

```
UnKnown = 0
Resource = 1
Customer = 2
Engagement = 3
Project = 4
Task = 5
TaskAssignment = 6
Time = 7
Expense = 8
Opportunity = 9
Product = 10
SupportDesk = 11
Request = 12
Contact = 13
Budget = 14
Invoice = 15
KnowledgeManagement = 16
Portfolio = 17
RequestTime = 18
RequestDemand = 19
ResourceAddress = 20
ResourceRate = 21
ResourcePayroll = 22
EngRevRec = 23
EngProduct = 24
EngProjectedResource = 25
EngRequestSLA = 26
EngBillingRate = 27
EngBillingRateHistory = 28
EngFixedFee = 29
EngFixedFeeItem = 30
EngRequestBillingRule = 31
EngSplitBillingRule = 32
EngWorkCode = 33
```

```
EngWorkLocation = 34
OppFixedFee = 35
OppService = 36
OppExpense = 37
OppProduct = 38
```

Related information

"Enumeration" on page 738

CPLogLevelType Enumeration

Contains predefined constant bytes to represent log levels.

Namespace

Changepoint.UIBase.CPLogLevelType

Enumeration

```
NoLog = 0
Source = 1
ErrorMessage = 2
InputParameters = 3
Returns = 4
Warning = 5
Checkpoint = 8
```

Related information

"Enumeration" on page 738

CPMetadataCheck Enumeration

Contains predefined constant bytes that indicate whether to check Changepoint metadata on Add/Create and Update events.

Namespace

Changepoint.UIBase.CPMetadataCheck

Enumeration

```
CheckAll = 0
BypassAll = 1
OnlyMandatory = 2
```

Related information

"Enumeration" on page 738

CPObjectAction Enumeration

Contains predefined constant bytes that represent object actioms.

Namespace

Changepoint.BusinessBase.CPObjectAction

Enumeration

```
GetValues = 0
Create = 1
Update = 2
Delete = 3
GetStatus = 6
UnChange = 9
```

CPProfileType Enumeration

Contains predefined constant bytes to indicate the display mode for Changepoint.

Namespace

Changepoint.UIBase.CPProfile

Enumeration

```
PSO = 0
ITD = 1
```

Related information

"Enumeration" on page 738

CPRevRecMethod Enumeration

Contains predefined constant bytes that represent revenue recognition methods.

Namespace

Changepoint.BusinessBase.CPRevRecMethod

Enumeration

```
cpRR_TimeAndExpenseBased = 0
```

```
cpRR_TimeBasedOnly = 1
cpRR_ExpenseBasedOnly = 2
cpRR_DisableRRForTimeAndExpenses = 3
```

Related information

"Enumeration" on page 738

CPUDFReturnType Enumeration

Contains predefined constant bytes that indicate how the return data will be constructed on the event of get user defined data.

Namespace

Changepoint.UIBase.CPUDFReturnType

Enumeration

```
WithStructure = 0
OnlyValues = 1
OnlyStructure = 2
OnlyStructureAndOptions = 3
WithStructureAndOptions = 4
```

Related information

"Enumeration" on page 738

UDF XML

```
<root>
   <udf>
      <entity/>
      <entityid/>
      <additionalid actual=''/>
      <resourceid/>
      <fields>
         <fielditem index='' fieldname='' fieldformat='1'>
         <cpcode name='' caption='' multiselect='0' entitybased="0"
            conditional="0" mandatory="0" >
           <codeitem text='' value='' selected='0'/>
         </cpcode>
         </fielditem>
         <fielditem index='' fieldname='' fieldformat='2'>
         <cptext name='' caption='' value='' mandatory="0" maximum="0"
             minimum="0" noteditable="0" lockedafterentry="0"
             duplicatecheck="0" />
```

```
        </fielditem>
      </fields>
   </udf>
</root>
```

Comments

There are a number of customizable code and text fields (UDFs) for some entities in Changepoint. In the API, these fields have been serialized as XML string, which are named sxmlUDF for entities.

The following table provides additional information on the UDF XML elements. Mandatory elements are marked with an asterisk (*).

Element	Comments
*root	Root node of XML
*udf	Root node of UDF. Child node of "root".
additionalid	An additional ID, which depends on the entity: • If Entity="Engagement", AdditionalId is BillingOfficeId (mandatory); • If Entity="Request", AdditionalId is RequestType (mandatory); • Otherwise should be empty. The “actual” attribute is the actual field name of additionalId.
codeitem	Code option for a code field. Codeitem has the following attributes: • text – code field text that is displayed in the drop-down list. • value – value associated with the text. Cannot be empty. • selected – selection flag for the code option. 1 = selected For code fields for which the multiselect flag is not enabled, there is only one codeitem allowed and the selected flag is set to 1.

Element	Comments
cpcode	UDF code field. The code field has the following attributes: • name – field name (mandatory). • caption – field caption (mandatory) • multiselect – multiple select flag (mandatory). A value of 1 indicates that multiple selection is enabled, and a value of 0 indicates that multiple selection is disabled. • entitybased – entity-based code flag (read-only). Indicates that the code field selection is based on the entity values: 1 is entity based code; 0 is not entity based code. • conditional – conditional flag (read-only). Indicates whether the code field selections have conditional relations to the other fields. • mandatory – flag that indicates if the field is mandatory.
cptext	UDF text field. The text field has the following attributes: • name – field name (mandatory). • caption – field caption (mandatory) • value – text value (mandatory). An empty string is allowed. • mandatory – flag that indicates if the field is mandatory. • maximum – metadata flag for the maximum requirement (read-only). Only applied for text and numeric fields, it is the limit for the maximum string length in text field value and maximum number in numeric field value. Zero means no limit. • minimum – metadata flag for minimum requirement (read-only). Only applied for text and numeric fields, it is the limit for minimum string length in text field value and minimum number in numeric field value. Zero means no limit. • noteditable – metadata flag for fields that cannot be edited (read-only). • lockedafterentry – metadata flag for locked after entry fields (read-only). • duplicatecheck – metadata flag for duplicate checking (read-only).
*entity	Changepoint entity name.
entityid	ID of the entity. Omitted for a new object.

Element	Comments
fielditem	Root node of each field. Each fielditem must only have only one cpcode or cptext child node. Fielditem has the following attributes: • index – index number for each field. Do not fill. It is reserved by the system. • fieldname – full name of the field (mandatory). It is composed of the entity name and field name • fieldformat – field type (mandatory) 1 = Code 2 = Text 3 = Numeric 4 = DateTime
fields	Root node of all fields.
resourceid	The resource that performs the action. If omitted, system will use login user.

Example

```
<root>
 <udf>
  <entity>Project</entity>
  <entityid>8599cc4c-8642-48d9-baab-9b7b3c620a6f</entityid>
  <additionalid actual="" />
  <resourceid>b4059754-d32c-11d2-9ae7-006008a4d883</resourceid>
  <fields>
    <fielditem index="0" fieldname="ProjectCode1" fieldformat="1">
      <cpcode name="Code1" caption="Code 1" multiselect="0" entitybased="0"
conditional="0"
         mandatory="0">
       <codeitem text="1" value="24b133bb-758a-42c2-82c6-3894a125b0ad" selected="0" />
       <codeitem text="1" value="66da0e78-1c34-4bfd-b3be-d5c4fb8f430e" selected="1" />
       <codeitem text="2" value="61d97a33-96a4-4692-a13d-d1ee9ac2aa70" selected="0" />
       <codeitem text="3" value="db050f1b-f041-446a-a49c-ea38eb9f888b" selected="0" />
      </cpcode>
    </fielditem>
    <fielditem index="1" fieldname="ProjectCode2" fieldformat="1">
      <cpcode name="Code2" caption="Code 2" multiselect="0" entitybased="0"
conditional="0"
        mandatory="0">
       <codeitem text="a" value="a441fa17-25d1-4c6e-9848-e67f825549d9" selected="1" />
       <codeitem text="b" value="c5cda39f-dfdb-495b-bca9-d2bda262c151" selected="0" />
      </cpcode>
```

```
    </fielditem>
    <fielditem index="29" fieldname="ProjectText6" fieldformat="3">
      <cptext name="Text6" caption="cal when changed" value="10011.0000" mandatory="0"
        maximum="0" minimum="0" noteditable="1" lockedafterentry="0" duplicatecheck="0"
/>
    </fielditem>
    <fielditem index="30" fieldname="ProjectText7" fieldformat="3">
      <cptext name="Text7" caption="Test Auto (Numeric)" value="" mandatory="0"
maximum="0"
        minimum="0" noteditable="0" lockedafterentry="0" duplicatecheck="0" />
    </fielditem>
  </fields>
 </udf>
</root>
```

UDF default values logic for CreateByXML

1. If you pass in the udf node in the xml file with a valid value, the value is saved.
2. If you pass in an empty udf node (that is, a <cpcode> with no <codeitem>s or <cptext> with value=""), there are two possibilities:
   a. If there is a udf default value, the node is populated with the default value.
   b. If there is no udf default value, the udf value remains empty.

   See `<cpcode name= "Code3" ...>` and `<cptext name= "Text1" ...>` in the example below.
3. If you do not pass in the udf node, the udf is not created.
4. If you pass in the udf node with an incorrect value, a warning for error code -10 is displayed in the log.

Examples of an empty udf node:

```
<cpcode name= "Code1" caption="Code 1" multiselect="0" entitybased="0" conditional="0">
mandatory="0">
<codeitem text="1" value="24b133bb-758a-42c2-82c6-3894a125b0ad" selected="0" />
<codeitem text="2" value="61d97a33-96a4-4692-a13d-d1ee9ac2aa70" selected="0" />
</cpcode>
<cpcode name= "Code3" caption="Code 3" multiselect="0" entitybased="0" conditional="0">
</cpcode>
<cptext name="Text1" caption="Text 1" value="" />
```

Error messages in the COM API

The following is a list of error numbers and error messages in the COM API.

```
-1: Operation failed (system error).
-7: Parameters are missing or not correct.
-8: Connection String is missing or not correct.
-9: Unknown error.
-10: The records are not found.
-11: Multiple records found.
-12: Error found in field:
-13: Invalid Id:
-14: General validation error.
-15: Operation complete with warnings.
-16: GUID format string is required.

(Login)
-20: Log Path does not exist.
-21: Cannot load error message file.
-22: Validate Changepoint user failed.
-23: User name or password is not correct.
-24: Login failed.
-25: Connection is invalid, please login first.
-26: ADO connection string is invalid.
-27: Password update error: The number of characters in the password cannot be 0.
-28: The number of characters in the password exceeds the maximum number of characters
allowed.
-29: The password cannot exceed 50 characters.
-30: The password does not have enough special characters.
-31: The password does not have enough uppercase characters.
-32: The password does not have enough numeric characters.
-33: The password does not have enough lowercase characters.
-34: The password cannot contain the user's first name or last name.
-35: The first character in the password cannot be a numeric character.
-36: The last character in the password cannot be a numeric character.
-37: The password contains a repeated character that exceeds the number of times
characters can be used consecutively.
-38: The password was used previously, less than the required number of password changes
ago.
-39: The new password uses too many characters from the previous password.
-40: The password is too short.
-41: The password cannot contain spaces.
-42: The password cannot contain pipe characters (|).
-43: You are required to change your password.


(UDF)
-50: XML format error.
-51: UDF Entity is not valid.
-52: UDF ResourceId is not valid.
-53: UDF AdditionalId is not valid.
```

```
-54: UDF EntityId is not valid.
-55: WorkgroupId/GlobalWorkgroupId is not valid.
-56: UDF field is not valid.
-57: UDF field format is not valid.

-60: The customer is required.
-61: The engagement is required.
-62: The Resource is required.
-63: The Project is required.
-64: The Task is required.

-70: The work location group is required.
-71: The work location is required.
-72: The work code category is required.
-73: The work code is required.
-74: The UDF Code Item Name is required.
-75: The UDF Text Item Name is required.
-76: The UDF Text format is required.
-77: Loading UDF error.
-78: Saving UDF error.
-79: UDF Validation error.

(Metadata)
-80: System metadata Mandatory check failed.
-81: System metadata Hidden/UnUsed check failed.
-82: System metadata Locked After Entry check failed.
-83: System metadata Minimum length/value check failed.
-84: System metadata Maximum length/value check failed.
-85: System metadata Duplicate check failed.
-86: System metadata Duplicate check error.
-87: System metadata Not Editable check failed.
-89: System metadata checking failed.

(Work flow)
-90: No workflow state definition is matched.
-91: No transition is allowed to specified status (state).
-99: No license exists for Feature
-100: Bad Payroll Cycle Value
-101: Bad Certification Cycle Value
-102: Bad Parameters passed to workgroup member history update (API internal error)
-103: Bad connection
-104: No resource id
-105: Problem removing license or updating signature during delete
-106: First Name is required
-107: First Name cannot contain the character '[' or ']'
-108: Last Name is required
-109: Last Name cannot contain the character '[' or ']'
-110: Middle Name cannot contain the character '[' or ']'
```

-111: A resource with the same name already exists
-112: A resource with the same SQL Login already exists
-113: A resource with the same NT Login already exists
-114: E-mail address has been chosen by another resource
-115: Unknown Validation error, API code probably needs updating
-116: Cannot delete or un-assign Resource is an active Project Manager
-117: Cannot delete or un-assign Resource is an active Project Time Approver
-118: Cannot delete or un-assign Resource is an active Project Alternate Time Approver
-119: At least one effective date must be set active on add or update of resource rates
-120: Error with First Payroll Date
-121: Base Currency is required
-122: Error trying to activate an already activated resource
-123: Tried to transfer a non-active or deleted resource
-124: Tried to un-assign or delete where there are acted upon transfers or un-assign in the future
-125: Tried to transfer a non-active or deleted resource
-126: Tried to un-assign a non-active or deleted resource
-127: Tried to insert a delete in the past when there are records between that date and today
-128: Adding resource did not return a resource id
-129: No resource exists during delete
-130: Cannot delete or un-assign Resource as they are an active Engagement Manager
-131: Cannot delete or un-assign Resource
-132: Cannot delete or un-assign Resource as they are the second level approver and there are invoices pending approval
-133: Cannot delete or un-assign Resource as they are a second level approver on a billing office
-134: Cannot delete or un-assign Resource as they are an active Engagement Time Approver
-135: Cannot delete or un-assign Resource as they are an active Engagement Expense Approver
-136: Cannot delete or un-assign Resource as they are an active Engagement Alternate Time Approver
-137: Cannot delete or un-assign Resource as they are an active Engagement Alternate Expense Approver
-138: Cannot delete or un-assign Resource as they are an active Opportunity Sales Rep
-139: Cannot delete or un-assign Resource as they are an active Standard Task Time Approver
-140: Cannot delete or un-assign Resource as they are an active Standard Task Alternate Time Approver
-141: Cannot delete or un-assign Resource as they are an active Time Approver
-142: Cannot delete or un-assign Resource as they are an active Delegated Time Approver
-143: Cannot delete or un-assign Resource as they are an active Expense Approver
-144: Cannot delete or un-assign Resource as they are an active Delegated Expense Approver
-145: Cannot delete or un-assign Resource as they are an active Threshold Expense Approver
-146: Cannot delete or un-assign Resource as they are an active Support Desk Manager

-147: Cannot delete or un-assign Resource as they are both an active Expense Approver and an active Time Approver
-148: Bad ResourceId passed to Update Manager Changes
-149: Cannot add or update a rate for a resource under federal audit without comments
-150: Tried to add a resource rate without a valid effective date
-151: Tried to update a rate for a resource under federal audit without comments
-152: Unable to save a new billing rate with the same active effective date
-153: Effective dates for rates must be between '01/01/1984' and '12/31/2049'
-154: There already exists an active rate for this date on update

(Customer)
-155: No CustomerId passed to Delete Customer
-156: Attempted to delete a customer with an active engagement
-157: Problem removing Guest Licenses from customer to be deleted
-158: Internal problem with DeleteCustomer, stored proc did not return numeric parameters
-159: The Billing Office Id is invalid/duplicate.
-160: The Customer name is required.
-161: The Customer Account Type is required.
-162: The Customer Sales status is required.
-163: The Customer status is required.
-170: Customer information does not pass system metadata Mandatory check
-171: Customer information does not pass system metadata Minimum length/value check
-172: Customer information does not pass system metadata Maximum length/value check
-173: Customer information does not pass system metadata Not Editable check
-174: Customer information does not pass system metadata Locked After Entry check
-175: Customer information does not pass system metadata Duplicate check
-176: Add Customer Failed, Reason unknown
-177: No customer Object passed to UpdateCustomer
-178: Error During update of Customer information
-179: Resource Postal Code not a valid format
-180: Customer Business 1 Postal Code not a valid format
-181: Customer Business 2 Postal Code not a valid format
-182: Customer Business 3 Postal Code not a valid format

-183: No Customer Id passed to Add Contact
-184: First Name is Mandatory for Contact
-185: Last Name is Mandatory for Contact
-186: No Contact object passed to Add Contact
-187: No ContactId returned from Add, reason unknown
-188: Contact information does not pass system metadata Mandatory check
-189: Contact information does not pass system metadata Minimum length/value check
-190: Contact information does not pass system metadata Maximum length/value check
-191: Contact information does not pass system metadata Not Editable check
-192: Contact information does not pass system metadata Locked After Entry check
-193: Contact information does not pass system metadata Duplicate check
-194: Contact Business Postal Code not a valid format

-195: Contact Home Postal Code not a valid format
-196: Contact Other Postal Code not a valid format
-197: No Contact ID passed to Delete Contact
-198: Internal problem with deletes
-199: Problem removing Guest License from contact to be deleted
-200: Internal problem with DeleteContact, stored proc did not return numeric parameters
-201: No contact object passed to Update Contact
-202: Bad Parameters passed to Update Contact
-203: No Fields to Update passed to Update Contact
-204: No InvoiceId passed
-205: InvoiceId is not valid
-206: No Invoice Number Passed to GetInvoice
-207: Invalid Invoice number
-208: GetPayment does not return a record.
-209: Payment Amount must be non 0
-210: Payment Date must be valid
-211: Payment Currency must be a valid currency code
-212: There is no valid exchange rate for the currency, date combination passed
-213: AddPayment failed to insert row in PaymentInfo Table
-214: Cannot add a payment to an invoice that is not in the status Paid, Partial Paid, Committed or sent
-215: GetPaymentTotals returned no record set
-216: No BillingOfficeId passed to GetBillingOfficeById
-217: BillingOfficeId is invalid
-218: Error trying to retrieve new tax information for invoice
-219: Time Records Missing but Pointed to By InvoicedTime
-220: Expense Records Missing but Pointed to By InvoicedExpense
-221: FixedFee Records Missing but pointed to by InvoicedFixedFees
-222: SupportTime records Missing but pointed to by InvoicedRequestTime
-223: Product records missing but pointed to by InvoicedProduct
-224: WriteOffTime records missing but pointed to by WriteOffTime
-225: WriteOffExpense records missing but pointed to by WriteOffExpense
-226: WriteOffRequestTime records missing but pointed to by WriteOffRequestTime
-227: WriteOffProduct records missing but pointed to by WriteOffProduct
-228: Status not passed to GetCreditNotes
-229: Cannot get a recordset from CreditNotes Table lookup
-230: Missing CreditNoteNumber
-231: Invalid CreditNoteNumber
-232: Missing CreditNoteId
-233: Invalid CreditNoteId
-234: Missing ExpenseReportId
-235: Invalid ExpenseReportId
-236: No record set returned trying to get values for names from expense report
-237: No record set returned trying to get individual expense records for expense report
-238: No record set returned trying to get records from ExpenseCategoryTotals
-239: The EngagementId is required.
-240: The Engagement Name is required.

-241: Federal audit enabled. Updates field is required.
-245: This engagement is locked by other user. You can only view.
-246: The effective date is set to today.
-247: Please enter a Billing Role or a resource
-248: Negotiated Rate cannot exceed 99999999.99.
-249: The Cost Rate cannot exceed 99999999.99.
-250: The discount percentage must be a number between -99999999.99 and 100.
-251: The billing rate cannot exceed 99999999.99.
-252: If the engagement uses two level approval on contract overage, contract amount is required.
-253: The Federal Audit is on. The Invoice Comments are required.
-254: The expense amount cannot exceed 99999999999999999.99
-255: The overtime percentage must be between 0 and 999.99.
-256: The Maximum Recognizable Revenue must be a number between 0 and 9999999999999999999999.99
-257: The labor multiplier must be a number between 0 and 1000.
-258: The adjustment factor for revenue recognition must be a number between 1 and 100.
-259: Transactions after this date is invalid
-260: If there is pending time under this engagement, the time approver is required.
-261: If there is pending expense under this engagement, the expense approver is required.
-262: The Associated engagement cannot be itself.
-263: The field contract number cannot contain the character |
-264: The fields Work Performed for and customer cannot be the same.
-265: Effective date or end date have to be date
-266: No Invoice Number Passed to GetInvoice
-267: The Contract End Date must be later than the Contract Start Date.
-268: The Main contact is required because engagement is billable.
-269: The Contract amount must be between 0 and 999999999999999999999.99
-270: The Maximum per invoice must be between 0 and 999999999999999999999.99
-271: The Expense percentage must be between 0 and 999.99
-272: If the engagement has a time approver, it must also have an alternate time approver.
-273: If the engagement does not have a time approver, it cannot have an alternate time approver.
-274: Time approver and alternate time approver cannot be the same.
-275: If the engagement has expense approver, it must have alternate expense approver.
-276: If the engagement does not have expense approver, it cannot have alternate expense approver.
-277: Expense approver and alternate expense approver cannot be the same.
-278: The second level approver is required.
-279: The second level approver cannot be the same as engagement manager.
-280: The field name/alternate name cannot contain the character |
-281: The customer Id does not exist in Changepoint.
-282: The percentage must be between 0 and 100.
-283: Valid main contact Id is required.
-284: The billing amount cannot exceed 99999999999999.99

-286: Cannot delete the fixed fee as it is invoiced
-287: Cannot delete the fixed fee as it is revenue recognized
-288: Cannot delete the fixed fee as it is linked with a task
-289: Cannot delete product as it is revenue recognized
-290: Cannot delete product as it is invoiced
-291: The quantity cannot exceed 9999999.99
-292: The negotiated price cannot exceed 999999999999.9999
-293: The negotiated cost cannot exceed 999999999999.9999
-294: Both the quantity and the negotiated price cannot have negative values
-295: Both the quantity and the negotiated cost cannot have negative values
-296: This engagementid does not exist in Changepoint
-297: The product is required
-298: Engagement information does not pass system metadata Locked After Entry check
-299: Engagement information does not pass system metadata Duplicate check
-301: The estimated hours must be between 0 and 99999.99
-302: The start date is required.
-303: The finish date is required.
-304: Some items have not been fully invoiced and committed. The billing office cannot be changed.
-305: Cannot change the billing office. It will delete all billing roles and resource rates in the Billing Rates dialog box.
-306: Cannot change the billing office. All the services, taxes, approvals and other default information will be deleted. Any tasks created with previous billing roles must be edited and the billing roles from the new billing office selected. Otherwise, resource rates will be used for billing.
-307: Some items have not been fully invoiced and committed. The Billable status cannot be changed.
-308: All the split billing rules have to be deleted before the engagement is changed to UnBillable
-309: Changing the billable status will not change the billable status of projects, tasks and expenses under this engagement. Once the billable status is changed, this engagement will be unavailable for invoicing. Outstanding time/expenses requiring invoicing should be invoiced prior to changing the billable status.
-310: Revenue has already been recognized for this engagement. The currency cannot be changed.
-311: Cannot change the Engagement Status. There are projects under this engagement with either a Draft or Active status. It is advised to mark these projects as completed or inactive to ensure that no more time and/or expenses are booked against them.
-312: Fixed Fee schedule records are linked to a task. The billing type cannot be changed.
-313: Fixed fees have been billed. The billing type cannot be changed.
-314: Revenue has been recognized. The billing type cannot be changed.
-315: Tasks are associated with this engagement. The billing type cannot be changed to fixed fee only.
-316: You have assigned task(s), which are not linked to a fixed fee schedule. The billing type cannot be changed to fixed fee only.

-317: Cannot change the billing type. It will reset all billing roles and resource rates in the Billing Rates dialog box.
-318: Some items have not been fully invoiced and committed. The billing type cannot be changed.
-319: Fixed Fee schedule records are linked to a task. The billing Office cannot be changed.
-320: Fixed fees have been billed. The billing Office cannot be changed.
-321: Revenue has been recognized. The billing office cannot be changed.
-322: Cannot Delete Engagement because there is any active invoice or project attached to this engagement
-323: Cannot Delete Engagement because there is any billed fixed fee attached to this engagement
-324: Cannot Delete Engagement because there is any billed engagement product attached to this engagement
-325: Cannot Delete Engagement because it is a split engagement
-326: Cannot Delete Engagement because it is recognized
-327: Cannot Delete Engagement because there is any recognized fixed fee attached to this engagement
-328: Cannot Delete Engagement because there is any recognized engagement product attached to this engagement
-329: Cannot Delete Engagement because it is not closed
-330: End time is required
-331: The start day must be between 0 and 6.
-332: The end day must be between 0 and 6.
-333: The effective date must be before the expiry date.
-334: The response hours cannot exceed 99999.99 hours.
-335: The resolution hours cannot exceed 99999.99 hours.
-336: The escalation hours cannot exceed 99999.99 hours.
-337: The start day is required
-338: The end day is required
-339: The start time is required
-340: The default work location is missing from the parameter xmlWorkLocation or its value of the Selected tag is not set to 1.
-341: The default work code is missing from the parameter xmlWorkCode or its value of the Selected tag is not set to 1.
-342: The billing rate cannot be deleted because the engagement is federal audit enabled
-343: The end time must be after the start time.
-344: The end time must be between 00:00 and 24:00
-345: The product Id is required.
-346: The start time must be between 00:00 and 23:59
-347: The engagement product is invoiced. It cannot be modified.
-348: The total billing amount cannot exceed 999999999999.99
-351: The fixed fee schedule is billed. It cannot be modified.
-353: The BillingRatesEffectiveDatesId is required.
-354: The total of split billing percentage has to be 0 or 100.
-355: This billing rate is already defined for the Engagement.
-356: The billing rate Id is required.

-357: A rate with this effective date was already specified
-358: Resource has been assigned viewing access to this engagement but does not have the View Engagement feature.
-359: Resource has been assigned editing access to this engagement but does not have the Edit Engagement feature.
-360: Resource has been granted access to this engagement but does not have the Create Engagement feature.
-361: Resource is unassigned or deleted. Please remove this Resource from the access list.
-362: Engagement manager is an unassigned or deleted Resource
-363: Invalid text format in Engagement user defined fields
-364: Duplicate check error in fields
-365: The split company cannot be duplicated
-366: The Maximum amount is required.
-368: No earlier record was found. The resource was created later then the active date.
-369: Unable to activate the resource on that date. Please select another effective date.
-370: Cannot un-assign a resource with an effective date which is earlier than init date
-371: No ProjectId passed to GetProject
-372: Bad ProjectId passed to GetProject
-373: Attempt to mark an expense report as batched when it's already marked
-374: Attempt to mark an invoice as batched when it's already marked
-375: The contact Birthday is not valid date format
-376: The contact Anniversary is not valid date format
-381: The request has been invoiced or time was booked against it. It cannot be deleted.
-382: Cannot find request by number
-383: Request Number is not unique
-384: Request Id is required
-385: Request Object is empty
-386: Request has been locked
-387: Cannot update request
-388: Cannot save UDF
-389: Cannot create request
-390: Field Data Require must be a date
-391: Field Estimated Hours must be a number
-392: Field Services Cost must be a number
-393: Field Expense Cost must be a number
-394: Field Product Cost must be a number
-395: Request Number is empty
-396: Parent request cannot be the same as request itself
-397: The field exceeds the limit
-398: The field contains invalid char |
-399: Validation error
-406: No earlier record was found. The resource was created later then the active date
-407: Unable to active the resource on that date. Please select another effective date
-408: Tried to un-assign a resource with an effective date that is earlier than initial date

-409: Licenses used out.
-410: License Usage has been tampered.
-411: Request type is invalid.
-412: The knowledge item is checked out, cannot be updated.
-413: The WorkGroupId is required.
-414: The StartDate or EndDate is invalid.
-415: The RequestDemandId is required.
-416: The Request Demand information is required.
-417: The field Effort is supposed to be a numeric value which is greater than 0 and less than 999999999.99.
-418: Cannot set resource workgroup to empty while updating.
-419: There is already a request processing rule with this request type.

(Project)
-420: Can not find project in XML.
-421: The ProjectId is required.
-422: The project name is required.
-423: The planned finish date is required.
-424: The number of planned hours cannot exceed 9999999.99.
-425: The planned start date is required.
-426: The Project Manager is required.
-427: The Project ProposedPhase is required.
-428: The Project currency is required.
-429: The Project first status is required.
-430: The Billable status is required.
-432: The Project Type is required.
-433: The Project Status is required.
-434: The Status Frequency is required.
-435: The project already exists, insert failed.
-436: Engagement status is closed or inactive or deleted.
-437: Customer is deleted.
-438: Can not assign ProposedPhase on new project while Workflow validation enabled.
-439: The project name cannot contain any following character: ',",~,:,?,#,|,&,/,\,%,*,<,>.

(Tasks)
-440: The TaskId is required.
-441: The Completed flag is required.
-448: The WBS is required.
-449: The WBS Order is required.
-450: The Task Name is required.
-451: The Update flag is required.
-452: The Task already exists, insert failed.
-453: Time has been booked against this task, it cannot be deleted.
-454: This task has subprojects, it cannot be deleted.
-455: This task has been locked, it cannot be deleted.

-456: There has assignment which booked time or expense under this task or it is a MSP project, the Task cannot be deleted.
-457: The planned hours cannot be updated because task planned hours and total planned hours of the task assignments do not match. Task planned hours can be updated by adding or updating a task assignment in ApiAssignment class.
-458: The number of planned hours cannot exceed 99999.99.
-459: The task cannot be created under the parent task because a Resource has been assigned and time has been booked against the parent task.
-479: The task cannot be deleted. Resource requests are enabled for the task, and there is a task assignment for the task or one of its subtasks.
-480: Expense has been booked and not invoiced yet, it cannot be delete
-490: The task cannot be deleted. The task has subtasks or task assignments with planned finish dates in a past fiscal period.

(TaskAssignment)
-460: The TaskAssignmentId is required.
-461: There has booked time or expense against, or it is a MSP project, or it has subproject or the planned finish date is in the past fiscal period with planned hours > 0, the TaskAssignment and it's associated Task cannot be deleted.
-462: Can not assign the same resource again in same task.
-465: The ResourceId is required.
-466: The TaskId is required.
-467: The Locked status is required.
-468: The TaskAssignment already exists, insert failed.
-469: The SoftBooked is required.
-470: The StatusNotRequired is set to true because the project type is Status Not Required.
-471: Task planned hours updated with the total planned hours of task assigments.
-481: Remaining Hours must be between 0 and 99999.99.
-482: Forecast start Date should be before Forecast Finish date.
-483: Actual Start Date should be before Forecast Finish date.
-484: You cannot re-open a completed task.
-485: The resource is neither assigned to this task nor is a manager/co-manager for this project.
-491: The parent task cannot be an open task.
-492: Cannot update a summary task to an open task.

(Time)
-500: The TimeId is required.
-501: The Time Date is missing or not correct.
-502: The regular hours are not valid.
-503: The over time hours are not valid.
-504: The adjusted regular hours are not valid.
-505: The adjusted over time hours are not valid.
-506: Time code1 is Mandatory field.
-507: Time code2 is Mandatory field.

-508: Time code3 is Mandatory field.
-509: Time code1 is unused field.
-510: Time code2 is unused field.
-511: Time code3 is unused field.
-512: The comment is required for the federal audit tasks.
-513: Only the un-submitted time can be updated.
-514: Two many time records selected, please narrow the period.
-520: All time bookings have been submitted within period.
-521: There are time bookings which have not been submitted before the period.
-522: Resource has not booked time for more than 1 week since last time booking before this period.
-523: The total hours do not meet required certification hours.
-524: Not pass daily time submission check.
-525: Some un-submitted time entries are beyond the payroll lag and can no longer be submitted.
-526: There is no un-submitted time in the period.
-527: The reject reason is required.
-528: Can not find the time for approval or rejection
-529: The Start Time is not valid.
-530: Time cannot be submitted because some time entries are in a close period.
-531: Unable to create a Time record. Please check your data such as task assignment etc...

(Request Time)
-550: The RequestTimeId is not required. The RequestTimeId is created by Changepoint API RequestTime class.
-551: The regular hours should be greater than 0.
-552: The RequestTimeId is required.
-553: The RequestTimeId is invalid.
-554: The RequestTimeId does not exist.
-555: Only the un-submitted and non adjusted requested time can be updated.
-556: Start Date cannot be later than End Date
-557: Either ResourceId or RequestId is required.
-558: The TimeZone is invalid.
-559: This time entry cannot be booked because the date is in a close period.
-560: This time entry cannot be updated because the date is in a close period.
-561: This time entry cannot be approved because the date is in a close period.
-562: The TimeZone is required.
-563: This time entry cannot be booked because restrict time booking option is enabled.
-564: The time entry is beyond the payroll lag and cannot be added/updated.

(Request Demand)
-570: The value of this field should be GUID.
-571: The value of this field should be 0 or 1.
-572: The StartDate cannot be later than EndDate.
-573: This xml tag is required.
-574: This RequestDemandId does not exist.

-575: This request has been locked by other user. Request demand information cannot be updated.
-576: This Request demand has been locked by other user. Request demand information cannot be updated.

(Qualification)
-601: Category is required
-602: Qualification is required
-603: Competency is required
-604: Name is required
-605: Name cannot be longer than 255 characters
-606: Description cannot be longer than 255 characters
-607: Function is required
-608: Billing role is required
-609: FunctionId in the xmlFunctionSkill does not match with the value of sFunctionId or skillid is missing.
-610: There is no function for this billing role.

(ExchangeRate)
-701: Base currency is invalid
-702: To currency is invalid
-703: Action date is invalid
-704: Start date is invalid
-705: End date is invalid
-706: Rate is invalid
-707: The start date must be before the end date
-708: Base Currency cannot be the same as To Currency
-709: The exchange rate periods must be contiguous for exchange rates of the same type.
-710: This is not an existing exchange rate. It cannot be updated.
-711: Base Currency cannot be changed when updating
-712: To Currency cannot be changed when updating

(Engagement)
-800: The fixed fee schedule has parent fixed fee, it cannot add fixed fee item.
-801: The billing date for the schedule item cannot be earlier than the billing date for the parent fixed fee
-802: The Deliverable is required.
-803: The parent fixed fee is required.
-804: The billing office is required for the billing role.
-805: The AuditDetail is required.
-806: The Engagement Status is required.
-807: The Engagement Manager is required.
-808: The Billing Office is required.
-809: The Support Desk is required.
-810: The Associated Workgroup is required.
-811: The Billing Type is required.
-812: The Invoice Format is required.

-813: The Expense Billing Type is required.
-814: The Expense GLA is required.
-815: The Work Location Group is required.
-816: The Work Location is required.
-817: The Billing Date is required.
-818: The billing office is provided, so billing role is required.
-819: The fixed fee Id is required.
-820: The projected resource Id is required.
-821: The request billing rule Id is required.
-822: This SLA has already been invoiced. It cannot be updated.
-823: This SLA has already been invoiced. It cannot be deleted.
-824: An active invoice is associated with this engagement. No SLA can be added.
-825: The Engagement SLA Id is required.
-826: This product has already been defined with the same priority and request type in the SLA.
-827: The split billing rule is not enabled for this engagement.
-828: Some items have not been fully invoiced and committed. The split billing rule cannot be changed.
-829: The request type is not a valid code.
-830: The request billing type is not valid.
-831: The billing office is provided. Either remove the value of the billing office or privide the billing role.
-832: This engagement has been locked by other user. It cannot be updated.
-833: This engagement projected resource has been locked by other user. It cannot be updated or deleted.
-834: This engagement has been locked by other user. It cannot be updated or deleted.
-835: CRGLTime cannot be updated because RevRecMethod is [Expense Base Only] or [Disable Recognition for Time & Expenses].
-836: DRGLTime cannot be updated because RevRecMethod is [Expense Base Only] or [Disable Recognition for Time & Expenses].
-837: Revenue Recognition cannot be updated because it is not enabled.
-838: Can't find a billing rate for this resource.
-839: Can't find a billing rate for this billing role.
-840: The value of [Active] cannot be False. The effective date of at least one record must be no later than today.
-841: Engagement federal audit is enabled. The comments is required.
-842: The action is not valid for engagement.
-843: Load list by xml failed for
-844: The action has to be set in the XML for
-845: The action is not valid for
-846: The sub-object engagement ids should match.
-847: The Sub object id is ignored for creating by xml.
-848: The engagement id should be same as the engagement id in the main level.
-849: The adjusted original split billing percentage must be a number greater than 0 and cannot exceed 100.
-850: The <action> value should be set to '1' or 'create' for CreateByXML.
-851: The fixed fee items with no parent fixed fee specified are ignored.

-852: The Billing Currency is required.
-999: The parent's schedule's billing amount will be overridden by the total of the billing amount of all the fixed fee items.

(Opportunity)
-900: Add Opportunity failed.
-901: Error updating UDF data for opportunity.
-902: The OpportunityId is required.
-903: The Opportunity name is required.
-904: The Opportunity name and altrenate name cannot contain any of the following character(s): |.
-905: The quantity must be a number between -9999999.99 and 9999999.99.
-906: The Customer id is required.
-907: The Customer id is invalid.
-908: The sales represenative is required.
-909: The sales represenative id is invalid.
-910: The currency is required.
-911: The currency is invalid.
-912: The Staffing Workgroup id is required.
-913: The Staffing Workgroup id is invalid.
-914: The CustomerId cannot be changed.
-915: The PrimaryContact id is invalid.
-916: The fields Description and Outcome Details cannot contain any of the following character(s): |.
-917: The fields Description and Outcome Details cannot exceed 2048 characters.
-918: The unit cost must be a number between -99999999999.9999 and 99999999999.9999.
-919: The quantity and unit cost cannot both be negative.
-920: The revenue forecast must be a number greater than or equal to 0 and less than 100000000000000.
-921: The selected expense type does not belong to selected expense category.
-922: The date of expense is outside the fiscal period defined for the billing office. The expense cannot be saved.
-923: The Initial close field is not a valid date.
-924: The Revised close field is not a valid date.
-925: The outcome date is required.
-926: Delete Opportunity failed.
-927: The OpportunityFixedFeeId is required.
-928: The Deliverable description is required.
-929: The Deliverable description cannot contain any of the following character(s): |.
-930: The Deliverable description cannot exceed 255 characters.
-931: The item comments cannot contain any of the following character(s): |.
-932: The item Date is required.
-933: The item Date is not a valid date.
-934: Validation for opportunity Product is failed.
-935: Retrieve existing opportunity fixed fee failed.
-936: The amount of deliverable must be a number greater than or equal to 0.
-937: The amount of deliverable must be a number less than 100000000000000.

-938: The date of this deliverable is outside the fiscal period defined for the billing office. The deliverable cannot be saved.
-939: The record is associated with a service record. You must first remove the association before proceeding with deleting this record.
-940: Validation for opportunity FixedFee is failed.
-941: The action has to be set for Opportunity sub-item in XML.
-942: Load fixed fee collection by opportunity id failed.
-943: Load fixed fee collection by xml failed.
-944: Retrieve existing opportunity expense failed.
-945: Validation for opportunity expense is failed.
-946: Load expense collection by opportunity id failed.
-947: Load expense collection by xml failed.
-948: The OpportunityExpenseId is required.
-949: Loading expense categories failed.
-950: The ServiceId is required.
-951: The start date must be before the end date.
-952: Error while trying to determine start and end dates for the Opportunity Billing Office.
-953: The start and end date of the service must fall within the fiscal period defined for the associated cost structure.
-954: Error while trying to determine start and end dates for Opportunity Fiscal Period.
-955: The start date of the service cannot be before the start date of the current fiscal period.
-956: The Estimated Time must be greater than 0 and less or equal to 99999.99 day(s).
-957: Select the cost structure currency.
-958: The Discount must be a number between -99999999.99 and 100.
-959: The Negotiaded Cost Rate must be a number between 0 and 99999999.99.
-960: The Negotiated Billing Rate must be a number between 0 and 99999999.99.
-961: The opportunity uses the fixed fee billing type. The selection of a fixed fee is required for the service.
-962: The OpportunityDetailId is required.
-963: The Product is required.
-964: The Product is invalid.
-965: The expense category id is not valid.
-966: Loading expense types failed.
-967: Only one decision outcome can be set.
-968: The Product quantity cannot be less than -9999999.99.
-969: The Product quantity cannot exceed 9999999.99.
-970: The negotiated price must be a number between -999999999999.9999 and 999999999999.9999.
-971: The cost must be a number between -999999999999.9999 and 999999999999.9999.
-972: The product discount must be a number between -99999999.9999 and 100.
-973: The standard price with applied discount cannot exceed 999999999999.9999.
-974: At least one of the values for the quantity and the cost must be positive. They cannot both be negative.
-975: At least one of the values for the quantity and the negotiated price must be positive. They cannot both be negative.

-976: The date of this opportunity product is outside the fiscal period defined for the billing office. The opportunity product cannot be saved.
-977: The Billing Role is not in the allowed range for the selected Billing Office.
-978: The Currency is not in the allowed range for the selected Billing Role.
-979: The service element has to be set as the root of each Opportunity Service in XML.
-980: The awarded competitor id does not exists in competitor list.
-981: Retrieve existing opportunity product failed.
-982: The Opportunity Service associated Fixed Fee is invalid.
-983: Load product collection by opportunity id failed.
-984: Load product collection by xml failed.
-985: The request id is invalid.
-987: The RevRec Method is not valid.
-988: Validation for opportunity Services has failed.
-989: Retrieve existing opportunity service failed.
-990: The sales represenative id and staffing workgroup id are invalid.
-991: The billing office id is invalid.
-992: The billing type is invalid.
-993: The fiscal period billing office id is invalid.
-994: The Opportunity is locked and cannot be saved.
-995: Update Opportunity failed.
-996: Error saving competitors.
-997: Error saving sales team.
-998: The expense multiplier must be a positive number and cannot exceed 999.99.
-1000: Loading expense type default values failed.

(Knowledge management)
-1001: Title is missing in knowledge management.
-1002: Missing Category in knowledge management.
-1003: Missing SubCategory in knowledge management.
-1004: Missing KMCode1 in knowledge management.
-1005: Missing KMCode2 in knowledge management.
-1006: Missing KMCode 3 in knowledge management
-1007: Missing KMText1 in knowledge management.
-1008: Missing KMText2 in knowledge management.
-1009: Missing KMText3 in knowledge management.
-1010: ProjectId is not in a valid format.
-1011: ReferenceTable is not valid.

(WorkCode/Location)
-1100: The work code category is required.
-1101: The work code is required.
-1102: The work location group is required.
-1103: The work location is required.
-1104: The EngagementId is required.
-1105: The default Work code category (DefaultWCCId) is required.
-1106: The default work code (DefaultWCId) is required.
-1107: The Allow work code override for request time (AllowRTCodeOverride) is required.

-1108: The AllWorkCodes is required.
-1109: The default Work Location Group (DefaultWLGId) is required.
-1110: The default work Location (DefaultWLId) is required.
-1111: The Allow work location override for request time (AllowRTLocOverride) is required.
-1112: The AllLocations is required.

(Cont. from Request and Resource)
-1200: No data for specified Id.
-1201: Cannot delete or un-assign Resource as they are an active Resource Time Approver.
-1202: Cannot delete or un-assign Resource as they are an active Resource Expense Approver
-1203: Cannot delete or un-assign Resource as they are an active non Project Time Approver.
-1204: Cannot delete or un-assign Resource as they are an active non Project Alternate Time Approver.
-1205: Cannot delete or un-assign Resource as they have an IPI metric assigned.
-1206: Error removing Roles and Features during unassignment.
-1207: Resource does not belong to a workgroup.
-1208: Error adding new Resource.
-1209: Failed to update ManagerChanges.
-1210: Failed to update Resource Functions.
-1211: Failed to add new Rate to the Resource object.
-1212: Resource belongs to more than one workgroup.
-1213: Failed to update Address info.
-1214: Failed to update Payroll info.
-1215: Failed to update WorkgroupHistoryMember.
-1216: Resource does not exist.
-1217: Invalid AttachmentId.
-1218: Failed to add or update rate info.
-1219: Cannot delete or un-assign Resource as they are inactive or deleted.
-1220: Cannot delete or un-assign Resource as there are acted upon transfers or un-assigns in the future.
-1221: Cannot Delete Request as it is referenced in one or more Budgets.
-1222: An action has already been scheduled on the same date, please select another date.
-1223: No ResourceRateId returned from ResourceRate.Add method.
-1224: Invalid Request Status based on allowable status values.
-1225: The Support Desk is required.
-1226: The Initiator is required.
-1227: The Request priority is required.
-1228: The Product is required.
-1229: The Description is required.
-1230: Cannot delete or un-assign Resource as they have other resources reporting to them.
-1231: Currency is required for Resource Rate.

-1232: The Assignment, resource or queue, cannot be used because it does not exist or access restriction is enabled.
-1233: The Project is not associated with the engagement for the request.
-1234: The Resource Type is required.
-1235: The Resource Type is invalid.
-1236: The Resource Global Workgroup or Workgroup is not valid.

(continued from Contact)
-1300: Can not delete contact because it is the main contact on a contract

(continued from Opportunity)
-1400: The Competitor in the Outcome cannot be set when opportunity is not lost.
-1401: Loading competitors failed.
-1402: Loading sales team failed.
-1403: The expense category is required.
-1404: The expense category is invalid.
-1405: Can not assign Status on new opportunity while Workflow validation enabled.
-1406: The Opportunity Status is required.
-1407: The awarded competitor is required.
-1408: The Actual close field is not a valid date.
-1409: The Billing Role is required.
-1410: Load service collection by xml failed.
-1411: Load list of identity collection by xml failed.

(Expense)
-1500: Add Expense failed.
-1501: Add expense history failed.
-1502: The ExpenseId is required.
-1503: The expense is not editable and it cannot be modified.
-1504: The Project is invalid.
-1505: The Project cannot be changed and it was reset.
-1506: The Category is required.
-1507: The ExpenseType is required.
-1508: The WorkLocationGroup is required.
-1509: The WorkLocation is required.
-1510: The Currency is required.
-1511: The ExpenseDate is required.
-1512: The ExpenseDate is not a valid date.
-1513: The Quantity must be a number between -9999999.99 and 9999999.99, different than 0.
-1514: The UnitPrice must be a number between -99999999999.9999 and 99999999999.9999, different than 0.
-1515: The exchange rate could not be found.
-1516: The ExchangeRate must be a number greater than 0 and cannot exceed 99999999.99999999999.
-1517: The Description is required since federal audit is enabled.
-1518: The ChangeReason is required.

```
-1519: Error loading expense taxes.
-1520: The ExpenseCode1 field is mandatory.
-1521: The ExpenseCode2 field is mandatory.
-1522: The ExpenseCode3 field is mandatory.
-1523: Error updating expense
-1524: The project does not allow to change WorkLocationGroup. Expense WorkLocationGroup
was changed to project's default value.
-1525: The project does not allow to change WorkLocation. Expense WorkLocation was
changed to project's default value.
-1526: The Exchange rate is disabled and it was replaced with default value.
-1527: The project does not allow to change expense Billable flag. Expense Billable flag
was changed to project's default value.
-1528: The RecoverableTax is not editable and it cannot be changed.
-1529: The total of the expense exceeds the limit.
-1530: The expense does not have recoverable taxes. The TaxAmount was set to 0.
-1531: The TaxAmount is not editable and it cannot be changed.
-1532: Invalid tax amount.
-1533: This field in the expense type is set to be fixed and it was replaced with
default value.
-1534: The ExpenseCode1 value was not valid and it was reset.
-1535: The ExpenseCode2 value was not valid and it was reset.
-1536: The ExpenseCode3 value was not valid and it was reset.
-1537: This field in the expense type is disabled and it was replaced with default
value.
-1538: This expense cannot be deleted. The ExpenseId is invalid or this expense has been
submitted to an expense report.
-1539: The Expense Resource cannot be changed and it was reset.
-1540: The resource is not allowed to create the expense for the project.
-1541: The Task is not valid and it was reset.
-1542: The FixedFee is not valid and it was reset.
-1602: Time and/or expenses entered for this project have not been fully invoiced and
archived. It cannot be deleted.


(Web Services)
-2000: This is not a SOAP request.
-2001: Login token is not valid, please login.
-2002: Login token expired, please login again.
-2003: Login user password is not provided by AccessToken.
```

Troubleshooting the COM API

Use the COM API troubleshooting section to identify the possible cause of and solution to issues you encounter while using the COM API.

Validation error: 22

Example:

```
Error -22: Validate Changepoint user failed.
ErrNumber=-2147217843 ErrDes="Login failed for user 'CPAdmin'."
```

Possible causes:

- CPAdmin doesn't have CPACCESS role.
- SQLLogin table does not have SQL account mapped to a Changepoint resource.

Solution:

Update the SQLLogin table manually or via the Resource Management page in Enterprise or System Manager.

Loading error: 48

Example:

```
Error 48: Error Loading DLL
Error Message(s)
Failed on connect to Database.
```

Possible causes:

- MDAC is not installed or registered properly.
- XML Parser 4.0 is not installed.

Solution:

- Install and register MDAC.
- Download XML Parser 4.0 from the Microsoft site.

Using COM API on a 64 bit server

Example:

```
Could not load file or assembly 'Changepoint.LegacyInterface... attempt was
made to load a program with an incorrect format...
```

Possible causes:

Windows 64-bit processes cannot load 32-bit components. The COM API calls several components and one of them (Changepoint.LegacyInterface.dll), is a 32-bit component. As a

result, the client program might encounter the following error message, "Could not load file or assembly 'Changepoint.LegacyInterface... attempt was made to load a program with an incorrect format..." when calling some methods in the COM API.

Solution:

In order to avoid this issue, it is recommended that you compile the client program as a 32-bit application by selecting "x86" from the Target CPU dropdown in the Advanced Compile Options dialog box of Compile tab in the project property.

Incomplete processing

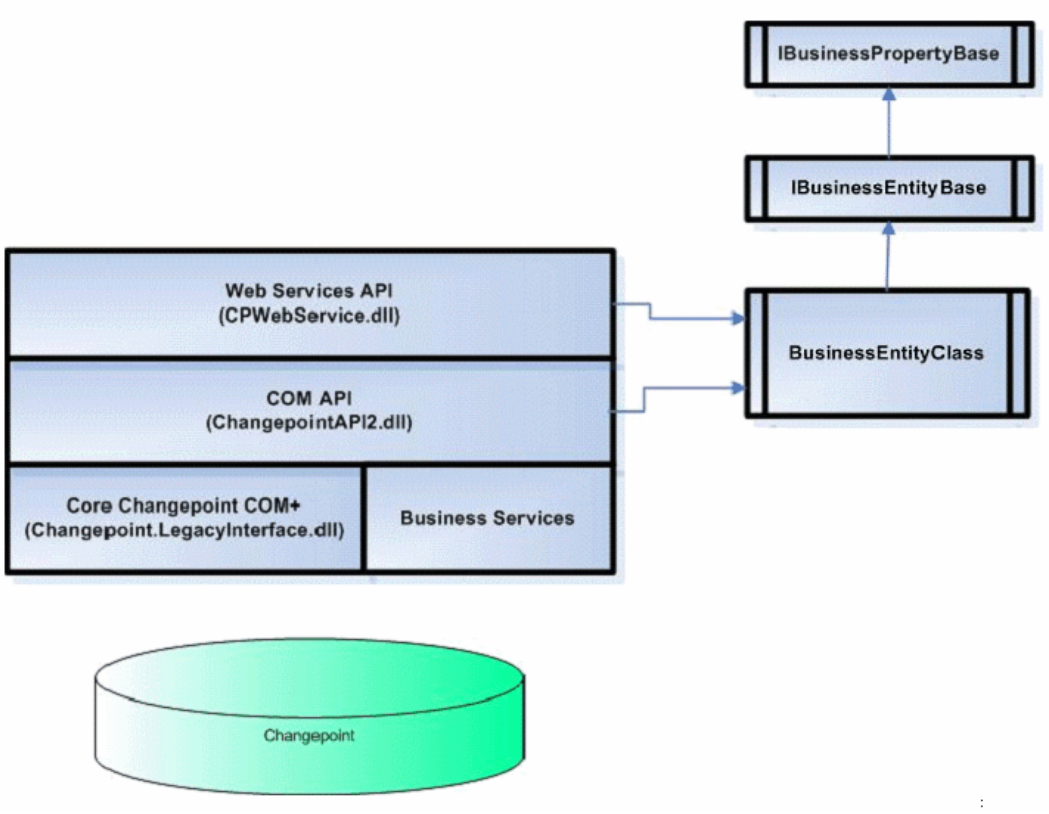
ApiCustomer, ApiEngagement, ApiResource and ApiRequest objects process multiple table updates in the Add and Update methods. During the processing Saving functionality, it is possible that one of the processes can fail even though some information has already been saved. If an error occurs, Delete method can be used to clean up a partial save.

2. Web Services API Objects and Methods

Web Services API Overview

The Changepoint Web Service interface is essentially a Web Service "wrapper" for the Changepoint COM API. When the Web Service interface receives a SOAP request that has been validated, the SOAP is parsed to determine the content of the request, and then the Web Service interface calls the corresponding method from the COM API. The result returned from the COM API method is translated into a SOAP response, which is then returned to the original requestor.

The COM API and Web Services API share the business entity object definition. Although they share some common properties, they have their own instances and their own functional interfaces.



The Changepoint API is not designed to check for and enforce business relations as is done in the user interface. Note the following limitations:

- If a GUID is passed as a parameter to a method, the API attempts to save it as is (for example, .Project.Id or .Product.Id when creating a Request object)

- You must pass the correct GUID to the API methods.

For information on requirements and installation, refer to the *API Installation Guide*, which is available though the Client Portal in the 2012 Release Notes and Patches team folder.

Note: New entity objects will not retain backward compatibility with previous versions.

Parameters

All parameters are mandatory in Web Service API methods.

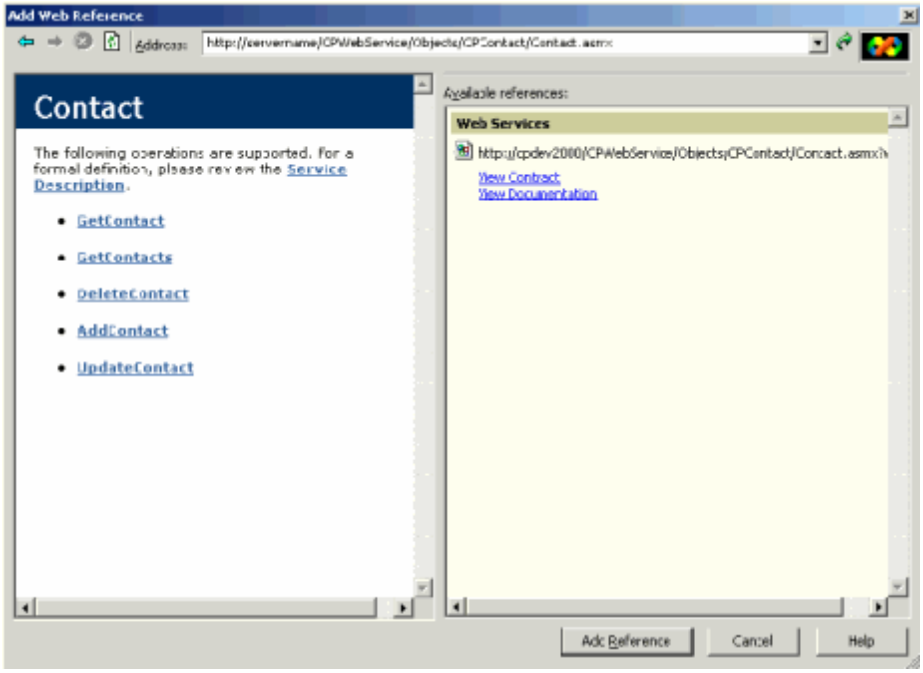
About the Changepoint Web Services API interface

For information on using the Changepoint Web Services Interface, see the following:

- "Adding a Web Services API reference to your .NET project" on page 770
- "Logging in to the Web Services API interface" on page 771
- "SOAP example" on page 772
- "Web Services API method example" on page 774

Adding a Web Services API reference to your .NET project

1. In your .NET project, select **Project** > **Add Web Reference** from the Menu bar.
2. In the Web Reference dialog box, enter the URL where the web service is located:



The format of the URL is http:*//servername/VirtualDirectory/Service.asmx*, where

- *servername* is the name (and port number) of the IIS server where the web services are installed, for example, localhost.
- *VirtualDirectory* is the name of the virtual directory where the web services are installed, for example, CPWebServices.
- *Service.asmx* is the path to .asmx file of the web service reference that you are adding. For example:

```
http://servername/VirtualDirectory/Objects/CPContact/Contact.asmx
http://servername/VirtualDirectory/Objects/CPCreditNote/CreditNote.asmx
http://servername/VirtualDirectory/Objects/CPCustomer/Customer.asmx
http://servername/VirtualDirectory/Objects/CPEngagement/Engagement.asmx
http://servername/VirtualDirectory/Objects/CPExpense/ExpenseReport.asmx
http://servername/VirtualDirectory/Objects/CPInvoice/Invoice.asmx
http://servername/VirtualDirectory/Objects/CPLookupids/LookupIds.asmx
http://servername/VirtualDirectory/Objects/CPProject/Project.asmx
http://servername/VirtualDirectory/Objects/CPResource/Resource.asmx
```

Logging in to the Web Services API interface

Before you can access any of the other Changepoint Web Services API methods, you must first log in to the Web Services API interface and generate a security token. This level of security ensures that only trusted users can access the Web Services API interface.

Once this token has been generated, it is passed to any additional Web Services API methods that are invoked to establish if the user should have access to the requested method.

1. Define a proxy class reference to the Web Services API class:

```
Dim proxy As New webWSLogin.WSLoginWse
```

2. Implement an instance of the UsernameToken class from the Web Services Enhancements (WSE 2.0) as shown in the following example. The user logon name and password arguments are the same user name (e-mail address) and password that the user would use to log into Changepoint)

```
Dim LoginToken As UsernameToken
LoginToken = New UsernameToken(UserName, Password, PasswordOption.SendHashed)
LoginToken.Id = "LOGIN"
```

3. Add this token to the proxy class.

```
proxy.RequestSoapContext.Security.Tokens.Add(LoginToken)
```

```
proxy.RequestSoapContext.Security.Elements.Add(New MessageSignature
(LoginToken))
```

4. Invoke the SOAP request.

```
Dim loginUser As webLogin.WSUser
loginUser=proxy.Login(UserName)
```

5. If the login attempt is successful, the Web Services API returns the user login information with the APIUser object.

Related information

"SOAP example" on page 772

"Web Services API method example" on page 774

"ApiUser class" on page 1351

SOAP example

The following code is a SOAP example that was sent to the server. The user name is "alisa" and the password is encrypted.

```
<?XML version=1.0 encoding=utf-8 ?>
<soap:Envelope XMLns:soap=http://schemas.XMLsoap.org/soap/envelope/
XMLns:xsi=http://www.w3.org/2001/XMLSchema-instance
XMLns:xsd=http://www.w3.org/2001/XMLSchema
XMLns:wsa=http://schemas.XMLsoap.org/ws/2004/03/addressing
XMLns:wsse=http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-
secext-1.0.xsd XMLns:wsu=http://docs.oasis-open.org/wss/2004/01/oasis-200401-
wss-wssecurity-utility-1.0.xsd>
- <soap:Header>

<wsa:Action>http://changepoint.com/changepoint/CPWebService/Login/Login</wsa:A
ction>
  <wsa:MessageID>uuid:cc464495-9e41-4478-aa2a-32de0adb25de</wsa:MessageID>
- <wsa:ReplyTo>

<wsa:Address>http://schemas.XMLsoap.org/ws/2004/03/addressing/role/anonymous</
wsa:Address>
  </wsa:ReplyTo>
  <wsa:To>http://localhost:8080/CPWebService/WSLogin.asmx</wsa:To>
- <wsse:Security soap:mustUnderstand=1>
- <wsu:Timestamp wsu:Id=Timestamp-0a5cfb4a-68a1-4152-be76-88bc82654347>
  <wsu:Created>2007-04-26T16:12:17Z</wsu:Created>
  <wsu:Expires>2007-04-26T16:17:17Z</wsu:Expires>
  </wsu:Timestamp>
```

```
- <wsse:UsernameToken XMLns:wsu=http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-utility-1.0.xsd wsu:Id=LOGIN>
  <wsse:Username>api</wsse:Username>
  <wsse:Password Type=http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-
username-token-profile-
1.0#PasswordDigest>SYq2mj0w+HUSSFNL2x5kHko1SJ4=</wsse:Password>
  <wsse:Nonce>BCg8B1PEhgkJdZesp6ODxw==</wsse:Nonce>
  <wsu:Created>2007-04-26T16:12:17Z</wsu:Created>
  </wsse:UsernameToken>
  </wsse:Security>
  </soap:Header>
- <soap:Body>
- <Login XMLns=http://changepoint.com/changepoint/CPWebService/Login>
  <userLoginId>api</userLoginId>
  </Login>
  </soap:Body>
  </soap:Envelope>
```

SOAP header example

```
Public Class UserToken
   Friend Shared Function SetToken(ByRef proxy As WebServicesClientProtocol,
Optional ByVal UserName As String = "", Optional ByVal Password As String =
"") As Long
   Dim TextToken As UsernameToken

   If Len(UserName) = 0 Then
      UserName = MainForm.mUserName
      Password = MainForm.mPassword
   End If

   If Len(UserName) = 0 Or Len(Password) = 0 Then
      Throw New ApplicationException("User name or password can not be
empty.")
   End If

   RedirectServer(proxy)

   TextToken = New UsernameToken(UserName, Password,
PasswordOption.SendPlainText)
      proxy.RequestSoapContext.Security.Tokens.Add(TextToken)
      proxy.RequestSoapContext.Security.Elements.Add(New MessageSignature
(TextToken))

   End Function
```

```
   Friend Shared Function SetLoginToken(ByRef proxy As
WebServicesClientProtocol, ByVal UserName As String, ByVal Password As String)
As Long
      Dim LoginToken As UsernameToken

      RedirectServer(proxy)

      LoginToken = New UsernameToken(UserName, Password,
PasswordOption.SendHashed)
      LoginToken.Id = "LOGIN"
      proxy.RequestSoapContext.Security.Tokens.Add(LoginToken)

   End Function

   Private Shared Sub RedirectServer(ByRef proxy As WebServicesClientProtocol)
      ' Reset server port
      If Not MainForm.mDefaultPort Then
         If Len(MainForm.mUsePort) > 0 Then
            proxy.Url = proxy.Url.Replace("localhost", "localhost:" &
MainForm.mUsePort)
         End If
      End If

      ' Re-direct web server
         If Len(MainForm.mServerName) > 0 Then
            proxy.Url = proxy.Url.Replace("localhost", MainForm.mServerName)
         End If

         'using SSL
         If MainForm.mUsingSSL Then
            proxy.Url = proxy.Url.Replace("http://", "https://")
      End If
   End Sub
End Class
```

Web Services API method example

The following example shows how to retrieve project information by using the method GetById().

1. Define a proxy class reference to the Web Services Class.

```
Dim mProxy As New webProject.APIProject
```

2. Implement an instance of the UsernameToken class, set the user login name as UserId and the password as AccessToken (both returned by the Login method documented above), set the password option as SendPlainText.

When adding UserToken to the SOAP message after logging in, the password must be sent in plain text. An example of the SetToken method of the UserToken.vb file in the Web Services Client Demo application is shown below:

```
Dim TextToken As UsernameToken
TextToken = New UsernameToken(UserName, AccessToken,
PasswordOption.SendPlainText)
```

3. Add this token to the proxy class.

```
mProxy.RequestSoapContext.Security.Tokens.Add(TextToken)
```

4. Invoke the Web Services API method.

```
Dim oRet As webProject.WSProject = mproxy.GetById("{5077E6E3-3ACC-444A-BF98-
088CAE8A952F}")
```

5. The SOAP is sent to the server. The username is the resource ID which was returned from the Login method, password is the access token returned by the Login method.

WCF for Changepoint Web Services APIs

This section explains how to access web services APIs for WCF, and describes services for certain methods for uploading and downloading attachments.

The WCF-specific APIs, all of which relate to Knowledge Management, KMVersion, or Request, are documented immediately after the corresponding WSE APIs.

Accessing WCF Web Services APIs

For your code to access the web services API, it must log in using WSLogin.svc to reinitialize the cached APIUser object.

Example:

```
Var client = new ServiceReferenceWSLogin.WSLoginClient();
client.ClientCredentials.UserName.UserName = "cpadmin";
client.ClientCredentials.UserName.Password = "cpadmin";
client.ClientCredentials.ServiceCertificate.Authentication.CertificateValidati
onMode = X509CertificateValidationMode.None;
var r1 = client.Login("cpadmin");
… …
```

Before using API methods, pass in the same ClientCredentials information.

Example:

```
Var client = new CustomerClient();
client.ClientCredentials.UserName.UserName = "cpadmin";
```

```
client.ClientCredentials.UserName.Password = "cpadmin";
client.ClientCredentials.ServiceCertificate.Authentication.CertificateValidati
onMode = X509CertificateValidationMode.None;
var r1 = client.GetXMLStructure();
… …
```

New services for uploading and downloading attachments

To support the WCF API, the following service methods are located in their own service files for uploading and downloading attachments. WCF Streaming and Caching are required for these services. You must set the usecache attribute in the web.config to true in order to use these methods.

KnowledgeManagementUploadDownload.svc for KnowledgeManagement object:

The following five methods are not available in KnowledgeManagement.svc. They are provided in KnowledgeManagementUploadDownload.svc

- GetKMByName
- GetKMById
- CreateKM
- UpdateKMByName
- UpdateKMById

KMVersionUploadDownload.svc for KMVersion object:

The following two methods are not available in KMVersion.svc. They are provided in KMVersionUploadDownload.svc

- LoadKMArticleHist
- KMUpdateVersion

RequestUploadDownload.svc for Request object:

The following two methods are not available in Request.svc. They are provided in RequestUploadDownload.svc

- GetAttachmentById
- UploadAttachment

For the above three services, manually adjust the binding information on the attributes maxReceivedMessageSize and transferMode in the app.config file generated on the client application.

For example, for the RequestUploadDownload.svc service, set the binding information as below:

```
<basicHttpBinding>
  <binding name="BasicHttpBinding_IRequestUploadDownload" closeTimeout="00:01:00"
openTimeout="00:01:00" receiveTimeout="00:10:00" sendTimeout="00:10:00"
allowCookies="false" bypassProxyOnLocal="false"
hostNameComparisonMode="StrongWildcard"
maxBufferSize="65536" maxBufferPoolSize="524288"
maxReceivedMessageSize="2147483647"
messageEncoding="Text" textEncoding="utf-8"
transferMode="Streamed"
useDefaultWebProxy="true">
<readerQuotas maxDepth="32" maxStringContentLength="8192" maxArrayLength="16384"
maxBytesPerRead="4096" maxNameTableCharCount="16384" />
<security mode="None">
<transport clientCredentialType="None" proxyCredentialType="None" realm="" />
<message clientCredentialType="UserName" algorithmSuite="Default" />
</security>
  </binding>
```

Contact

The Contact object allows you to create, update or delete contacts in the Changepoint database.

Namespace

http://changepoint.com/changepoint/CPWebService/Contact

URL

http://YourWebServer/changepoint/CPWebService/Contact.asmx

Methods

Properties

For more information, see the "ApiContact" section on page 43.

Contact: AddContact

```
Public Function AddContact(ByVal oContact As ApiContact) As WSInt32
```

Purpose

Adds a new contact to a customer in Changepoint.

Parameters

Parameter	Description
oContact	The populated ApiContact object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Use the GetByUserDefinedId method of the Customer object to find the customer ID to which to attach the contact.

Example

```
Dim proxy As New WebContact.ContactWse
Dim mContact As New webContact.ApiContact
Dim iRet As Int32

UserToken.SetToken(proxy, mUserName, mPassword)


With mContact
   .Reference = False
   .History = "History"
   .Customer.Id = "{D0674B9C-BBF7-4BB9-8D38-689EB1DC2BDC}"
   .prefix = "Mr."
   .FirstName = "Jane"
   .LastName = "Smith"
   .MiddleName = "Rose"
   .ContactType.Id = "{9146E6A3-7A3D-11D2-80ED-0060975AEC0F}"
   .Description = "Main contact"
   .Title = "Title"
   .Department = "0"
   .Office = "Office"
   .Profession = "Profession"
   .ManagerName = "ManagerName__random__"
   .AssistantName = "Assistant Name"
   .SpouseName = "Spouse Name"
   .Birthday = "1980/10/27"
   .Anniversary = "2006/1/1"
   .BusinessAddress = "Business Address"
   .BusinessAddressLine2 = "Business Address Line2"
   .BusinessAddressLine3 = "Business Address Line3"
   .BusinessCity = "Business City"
   .BusinessProvince.Id = "{04B6E8AE-79C6-4402-87B2-1BBC5D8914CB}"
```

```
    .BusinessPostal = "Business Postal"
    .HomeAddress = "Home Address"
    .HomeAddressLine2 = "Home Address Line2"
    .HomeAddressLine3 = "Home Address Line3"
    .HomeCity = "Home City"
    .HomeProvince.Id = "{00000000-0000-0000-0000-000000000000}"
    .HomePostal = "Home Postal"
    .IgnoreSomeMetaDataChecks = False
    .OtherAddress = "OtherAddress"
    .OtherAddressLine2 = "Other Address Line2"
    .OtherAddressLine3 = "Other Address Line3"
    .OtherCity = "Other City"
    .OtherProvince.Id = "{00000000-0000-0000-0000-000000000000}"
    .OtherCountry.Id = "CAN"
    .OtherPostal = "Other Postal"
    .AssistantPhone = "Assistant Phone"
    .BusinessPhone = "Business Phone"
    .BusinessPhone2 = "Business Phone2"
    .BusinessFax = "Business Fax"
    .CarPhone = "Car Phone"
    .HomePhone = "Home Phone"
    .HomePhone2 = "Home Phone2"
    .HomeFax = "Home Fax"
    .ManagerPhone = "Manager Phone"
    .MobilePhone = "Mobile Phone"
    .OtherPhone = "Other Phone"
    .OtherFax = "Other Fax"
    .Pager = "123"
    .PrimaryPhone = "321123123"
    .Email1 = "Email1"
    .Email2 = "Email2"
    .Email3 = "Email3"
    .WebAddress = "Web Address"
    .HomeWorkLocationGroup.Id = "{00000000-0000-0000-0000-000000000000}"
    .HomeWorkLocation.Name = "Home Work Location Name"
    .HomeWorkLocation.Id = "{00000000-0000-0000-0000-000000000000}"
    .BusinessWorkLocationGroup.Id = "{6AAF1A2D-2535-40E3-8D03-07801AE85E41}"
    .BusinessWorkLocation.Id = "{0175FA25-7CAA-4DC3-BB95-62A37A36134F}"
    .OtherWorkLocationGroup.Id = "{00000000-0000-0000-0000-000000000000
.OtherWorkLocation.Id = "{00000000-0000-0000-0000-000000000000}"
    .NickName = "Nick Name"
    .AvailableToAll = True
    .ContactExists = False
    .ContactCode1.Id = "{AF4F4AEF-36F4-11D4-B418-0050DA7707C1}"
    .ContactCode2.Id = "{AF4F4AF1-36F4-11D4-B418-0050DA7707C1}"
    .ContactCode3.Id = "{AF4F4AF7-36F4-11D4-B418-0050DA7707C1}"
    .ContactText1 = "Contact Text1"
```

```
   .ContactText2 = "Contact Text2"
   .ContactText3 = "Contact Text3"
End With

iRet = proxy.AddContact(mContact).Value
...'Continue processing
```

Related information

"Contact" on page 777

"Customer: GetByUserDefinedId" on page 802

Contact: CheckCustomerReference

```
Public Function CheckCustomerReference(ByVal sCustomerId As String) As
WSBoolean
```

Purpose

Check whether this customer can be referenced or not.

Parameters

Parameter	Description
sCustomerId	Contact ID to retrieve.

Returns

Type	Description
WSBoolean	True if the customer is willing to serve as a reference, else false. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

None

Example

Not available

Related information

"Contact" on page 777

Contact: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, ByRef sId As String) As
WSInt32
```

Purpose

Creates a contact using an XML string of the Contact object in Changepoint.

Parameters

Parameter	Description
sXML	A new contact XML string which is passed based on the contact XML schema
sId	ByRef parameter that will return the new ContactId after the contact has been added.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The ApiContact XML structure can be obtained by the GetXMLStructure method. The ByPassMetadataCheck switch will stop any metadata validation in Contact.

Example

Not available

Related information

"Contact" on page 777

"ApiContact XML" on page 50

"Contact: GetXMLStructure" on page 788

Contact: DeleteContact

```
Public Function DeleteContact(ByVal sContactId As String) As WSInt32
```

Purpose

Deletes a contact in Changepoint.

Parameters

Parameter	Description
sContactId	Contact ID

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

Not available

Related information

"Contact" on page 777

Contact: Exists

```
Public Function Exists(ByVal sId As String) As WSBoolean
```

Purpose

Check if the contact exists.

Parameters

Parameter	Description
sId	ContactId

Returns

Type	Description
WSBoolean	True if the contact exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

Not available

Related information

"Contact" on page 777

Contact: GetByXML

```
Public Function GetByXML(ByVal sXML As String, ByVal sContactId As String) As
WSString
```

Purpose

Takes the XML string passed in (sXML) and returns the string filled with data for the specified ContactId (sContactId).

Parameters

Parameter	Description
sXML	XML string of the Contact object with the fields specified.
sContactId	Contact ID. • If no ContactID is passed in, the XML string (sXML) is examined • if there is no Contact ID in sXML then the object's Contact ID is selected. • If there still is no valid ContactId, the method returns an error.

Returns

Type	Description
WSString	An XML string mirroring sXML with data inserted or the entire XML of the Contact object including data. Access the returned string through the .Value property of WSString

Remarks

If sXML = "" then the XML string provided by GetXMLStructure is used.

Example

Not available

Related information

"Contact" on page 777

"ApiCustomer XML" on page 75

Contact: GetContact

```
Public Function GetContact(ByVal sContactID As String) As WSContact
```

Purpose

Retrieve a contact by contact ID

Parameters

Parameter	Description
sContactId	Contact ID to retrieve.

Returns

Type	Description
WSContact	Returns an ApiContact object. Access the returned ApiContact object through the .value property of WSContact.

Remarks

None

Example

Not available

Related information

"Contact" on page 777

Contact: GetContacts

```
Public Function GetContacts(ByVal sCustomerId As String) As WSDataSet
```

Purpose

Retrieve contact records for a customer.

Parameters

Parameter	Description
sCustomerId	Customer ID to retrieve the contacts for.

Returns

Type	Description
WSDataSet	WSDataset (FirstName, MiddleName, LastName, Name, and ContactId) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Contact" on page 777

Contact: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As WSString
```

Purpose

Returns the ContactId based on the UDF Text field and value.

Parameters

Parameter	Description
sUDFField	UDF text field name (for example, Text1)
sUDFValue	UDF text field value

Returns

Type	Description
WSString	String with the ContactId or an empty string. Access the returned string through the .Value property of WSString

Remarks

None

Example

Not available

Related information

"Contact" on page 777

Contact: GetXMLStructure

```
Public Function GetXMLStructure() As WSString
```

Purpose

Returns the XML structure of the ApiContact object

Parameters

None

Returns

Type	Description
WSString	String with the XML structure of the ApiContact object. Access the returned string through the .Value property of WSString

Remarks

Some fields in the structure will have defaulted data, otherwise fields are empty.

See ApiContact XML for the returned structure.

Example

Not available

Related information

"Contact" on page 777

"ApiContact XML" on page 50

Contact: SetPropertiesByXml

```
SetPropertiesByXml(ByVal sXML As String) As WSContact
```

Purpose

Sets the properties of the object via an XML string.

Parameters

Parameter	Description
sXML	XML string of the contact object with data to load into the object.

Returns

Type	Description
WSContact	Returns an ApiContact object. Access the object through the .Value property of WSContact.

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods. If you use the SetPropertiesByXML method, you must pass in the entire XML string.

The XML string can be the entire contact object with all fields filled or only partially.

Empty fields in the XML string will result in the object's matching field being overwritten with an empty value (that is, the field will be cleared). Therefore if existing data in the object is to be retained, remove the field from the XML string.

When called from an empty object, the method will first retrieve current data into the object before applying the new field values as held in sXML.

In the case where properties in the object have been altered prior to SetPropertiesByXML being called, any nodes passed in the sXML will overwrite the object properties.

It is best to load an empty object with data first before applying any changes.

Example

Not available

Related information

"Contact" on page 777

"ApiContact XML" on page 50

"Contact: CreateByXML" on page 782

"Contact: UpdateByXML" on page 790

Contact: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, ByVal sContactId As String)
As WSInt32
```

Purpose

Update the database using an XML string of the Contact object containing update information.

Parameters

Parameter	Description
sXML	A contact XML string with updated fields.
sContactId	Contact being updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The method uses the following sequence to find the contact ID:

1. If the sContactId parameter is passed in, the method uses this value for the contact ID.

2. If this fails, the method attempts to extract the contact ID from <contactid> in the XML.
3. If this fails, the contact ID is taken from the object properties.
4. If this fails, an attempt is made to look up the contact ID using <email1> in the XML. If <email1> has a value, but the value is invalid or duplicated, then you will get an error and no further attempts are made to look up the contact ID.
5. If <email1> is empty, an attempt is made to look up the contact ID using <name> in the XML.

Example

Not available

Related information

"Contact" on page 777

"ApiContact XML" on page 50

Contact: UpdateContact

```
Public Function UpdateContact(ByVal oContact As ApiContact) As WSInt32
```

Purpose

Updates a contact in Changepoint.

Parameters

Parameter	Description
oContact	The populated ApiContact object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

```
Dim proxy As New WebContact.ContactWse
Dim mContact As New webContact.ApiContact
Dim iRet As Int32

UserToken.SetToken(proxy, mUserName, mPassword)

mContact = proxy.GetContact("{7C2EC28B-4066-11D4-A17D-
00010228881E}").myContact

With mContact
   .Title = "Account Manager"
   .ManagerName = "John Smith"
   .BusinessPhone = "416-123-4567"
End With

iRet = proxy.UpdateContact(mContact).Value
```

Related information

"Contact" on page 777

CreditNote

The CreditNote object allows users to retrieve credit notes in the Changepoint database.

Namespace

`http://changepoint.com/changepoint/CPWebService/CreditNote`

URL

`http://webserver/CPWebService/Objects/CPCreditNote/CreditNote.asmx`

Methods

Properties

For more information, see the "ApiCreditNote" section on page 66.

CreditNote: GetCreditNote

```
Public Function GetCreditNote(ByVal sCreditNoteNumber As String) As
WSCreditNote
```

Purpose

Retrieve a credit note record by credit note number.

Parameters

Parameter	Description
sCreditNoteNumber	Credit note number.

Returns

Type	Description
WSCreditNote	Returns an ApiCreditNote object. Access the object through the .Value property of WSCreditNote.

Remarks

None

Example

Not available

Related information

"CreditNote" on page 792

CreditNote: GetCreditNoteById

```
Public Function GetCreditNoteById(ByVal sCreditNoteId As String) As
WSCreditNote
```

Purpose

Retrieve a credit note record by Credit Note ID

Parameters

Parameter	Description
sCreditNoteId	CreditNoteId.

Returns

Type	Description
WSCreditNote	Returns an ApiCreditNote object. Access the object through the .Value property of WSCreditNote.

Remarks

None

Example

Not available

Related information

"CreditNote" on page 792

CreditNote: GetCreditNotes

```
Public Function GetCreditNotes(ByVal status As ApiInvoice.CPInvoiceStatus) As
WSDataSet
```

Purpose

Retrieve credit note records by invoice status

Parameters

Parameter	Description
Status	Status of the invoice

Returns

Type	Description
WSDataSet	WSDataset (CreditNoteId, InvoiceId, InvoiceNumber, CreditNote number). Access the dataset through the .Value property of WSDataSet.

Remarks

The following statuses apply to the credit note:

- cpInvDraft
- cpInvPending_Approval
- cpInvPending_SecondLevel_Approval
- cpInvApproved
- cpInvCommited
- cpInvSent
- cpInvPartialPaid
- cpInvPaid

The returned InvoiceNumber in the dataset is from DisplayInvoiceId, and the returned CreditNote number is from DisplayInvoiceId and 'CN' (for example, "00005CN").

Example

```
Dim myProxy As New webCreditNote.CreditNoteWse
Dim oRet As webCreditNote.WSDataset
UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetCreditNotes(webCreditNote.CPInvoiceStatus.cpInvDraft)
```

Related information

"CreditNote" on page 792

Customer

The Customer object allows users to create, retrieve and update customers in the Changepoint database. Used with the ApiCustomer object, the ApiCustomerAddress object is also exposed by the ChangepointAPI2

Namespace

`http://changepoint.com/changepoint/CPWebService/Customer`

URL

`http://webserver/CPWebService/Objects/CPCustomer/Customer.asmx`

Methods

Properties

For more information, see the "ApiCustomer" section on page 70 and the "ApiCustomerAddress" section on page 96.

Customer: Add

```
Public Function Add(ByRef sId As String, ByVal oCustomer As ApiCustomer) As
WSInt32
```

Purpose

Add a new customer to Changepoint database.

Parameters

Parameter	Description
sId	CustomerId
oCustomer	A populated ApiCustomer object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myCus As New webCustomer.ApiCustomer
Dim myProxy As New webCustomer.CustomerWse
Dim sId As String = String.Empty
Dim bo() As webCustomer.Identity
ReDim bo(0)
bo(0) = New webCustomer.Identity
bo(0).Id = "{5757A330-3476-11D3-807A-00105A0B7C01}"
With myCus
.Name = "Changepoint Canada"
.AccountManager = New webCustomer.Identity()
.AccountManager.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
.AccountType = New webCustomer.Identity()
```

```
.AccountType.Id = "{9146E6A9-7A3D-11D2-80ED-0060975AEC0F}"
.AllowEngagement = True
.CustomerStatus = New webCustomer.Identity()
.CustomerStatus.Id = "{9146E69E-7A3D-11D2-80ED-0060975AEC0F}"
.SalesStatus = New webCustomer.Identity()
.SalesStatus.Id = "C"
.SalesRep = New webCustomer.Identity()
.SalesRep.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
.BusinessAddress1 = New webCustomer.ApiCustomerAddress
.BusinessAddress1.AddressLine = "30 Leek Crescent, Suite 400"
.BusinessAddress1.City = "Richmond Hill"
.BusinessAddress1.StateProvince = New webCustomer.Identity
.BusinessAddress1.StateProvince.Id = "{27044F87-175F-4F45-A8AD-12750600EE49}"
.BusinessAddress1.Country = New webCustomer.Identity
.BusinessAddress1.Country.Id = "CAN"
.BusinessAddress1.PostalCode = "L4B 4N4"
.BillingOffices = bo
.
.
End With
'set the SOAP header with UsernameToken which includes
'login user and access token
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSInt32 = myProxy.Add(sId, myCus)
```

Related information

"Customer" on page 795

Customer: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, ByRef sId As String) As
WSInt32
```

Purpose

Create a customer using an XML string of the Customer object in Changepoint.

Parameters

Parameter	Description
sXML	A new customer XML string which can be passed based on the Customer xml schema
sId	ByRef parameter that will return the new CustomerId after the customer has been added.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The ApiCustomer XML structure can be obtained by the GetXMLStructure method.

The ByPassMetadataCheck switch will stop any metadata validation in Customer and also in Customer UDFs.

For details of the use of UDF default values, see "UDF default values logic for CreateByXML" on page 746.

Example

Not available

Related information

"Customer" on page 795

"ApiCustomer XML" on page 75

Customer: Delete

```
Public Function Delete(ByVal sId As String) As WSInt32
```

Purpose

Deletes the Customer from Changepoint database.

Parameters

Parameter	Description
sId	ID of the customer to be deleted.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Customers with active contracts will not be deleted.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webCustomer.CustomerWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSInt32 = myProxy.Delete("{CA8AC6E5-5F9A-41CA-BD57-
9A35D28A7E76}")
```

Related information

"Customer" on page 795

Customer: Exists

```
Public Function Exists(ByVal sId As String) As WSBoolean
```

Purpose

Check if the customer exists.

Parameters

Parameter	Description
sId	CustomerId

Returns

Type	Description
WSBoolean	True if the customer exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webCustomer.CustomerWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSBoolean = myProxy.Exists("{CA8AC6E5-5F9A-41CA-BD57-
9A35D28A7E76}")
```

Related information

"Customer" on page 795

Customer: GetById

```
Public Function GetById(ByVal sId As String) As WSCustomer
```

Purpose

Fills the ApiCustomer object with customer information of the specified sId passed in the parameter.

Parameters

Parameter	Description
sId	ID of customer that will be retrieved.

Returns

Type	Description
WSCustomer	Returns an ApiCustomer object. Access the object through the .Value property of WSCustomer.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webCustomer.CustomerWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSCustomer = myProxy.GetById("{CA8AC6E5-5F9A-41CA-
BD57-9A35D28A7E76}")
```

Related information

"Customer" on page 795

Customer: GetByUserDefinedId

```
Public Function GetByUserDefinedId(ByVal userDefinedCusomertId As String,
ByRef retDataSet As DataSet) As WSCustomer
```

Purpose

Fills the object with customer information of the specified user defined customer ID.

Parameters

Parameter	Description
userDefinedCustumerId	User Defined Customer ID.
retDataSet	A list of customers retrieved in a dataset by the userDefinedCustomerId.

Returns

Type	Description
WSCustomer	Returns an ApiCustomer object. Access the object through the .Value property of WSCustomer.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

WSCustomer.value (ApiCustomer) is filled with the first customer from the retDataSet list.

Returns the following columns: CustomerId, Name.

Example

```
Dim myProxy As New webCustomer.CustomerWse
Dim dsRet As DataSet
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSCustomer = myProxy.GetByUserDefinedId("Cust-2008-
0001", dsRet)
```

Related information

"Customer" on page 795

Customer: GetByXML

```
Public Function GetByXML(ByVal sXML As String, ByVal sCustomerId As String) As
WSString
```

Purpose

Takes the XML string passed in (sXML) and returns the string filled with data for the specified CustomerId (sCustomerId).

Parameters

Parameter	Description
sXML	XML string of the Customer object with the fields specified.
sCustomerId	Customer ID. • If no Customer ID is passed in, the XML string (sXML) is examined • if there is no Customer ID in sXML then the object's Customer ID is selected. • If there still is no valid CustomerId, the method returns an error.

Returns

Type	Description
WSString	An XML string mirroring sXML with data inserted or the entire XML of the Customer object including data. Access the returned string through the .Value property of WSString.

Remarks

If sXML = "" then the xml string provided by GetXMLStructure is used.

Example

Not available

Related information

"Customer" on page 795

Customer: GetCampaigns

```
Public Function GetCampaigns() As WSDataset
```

Purpose

Retrieve a list of campaigns in a dataset.

Parameters

None

Returns

Type	Description
WSDataSet	WSDataset (CampaignId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webCustomer.CustomerWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSDataset = myProxy.GetCampaigns()
```

Related information

"Customer" on page 795

Customer: GetContactsByCustomerId

```
Public Function GetContactsByCustomerId(ByVal sId As String) As WSDataSet
```

Purpose

Retrieve a list of contacts in a dataset by CustomerId.

Parameters

Parameter	Description
sId	ID of Customer

Returns

Type	Description
WSDataSet	WSDataset (ContactId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webCustomer.CustomerWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSDataset = myProxy.GetContactsByCustomerId("
{CA8AC6E5-5F9A-41CA-BD57-9A35D28A7E76}")
```

Related information

"Customer" on page 795

Customer: GetContactsbyCustomerName

```
Public Function GetContactsbyCustomerName(ByVal customerName As String, ByVal
contactFirstName As String, ByVal contactLastName As String) As WSDataSet
```

Purpose

Retrieve a list of contacts in a dataset by filters provided by parameters

Parameters

Parameter	Description
customerName	Name of Customer
contactFirstName	First name of contact. An An empty string is allowed.
contactLastName	Last name of contact. An An empty string is allowed.

Returns

Type	Description
WSDataSet	WSDataset (ContactId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webCustomer.CustomerWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSDataset = myProxy. GetContactsbyCustomerName
("Changepoint Corporation","","")
```

Related information

"Customer" on page 795

Customer: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As WSString
```

Purpose

Returns the CustomerId based on the UDF Text field and value.

Parameters

Parameter	Description
sUDFField	UDF text field name (for example, Text1)
sUDFValue	UDF text field value

Returns

Type	Description
WSString	String with the CustomerId or an empty string. Access the returned string through the .Value property of WSString

Remarks

None

Example

Not available

Related information

"Customer" on page 795

Customer: GetList

```
Public Function GetList(ByVal iRetRows As Int16) As WSDataset
```

Purpose

Retrieve a list of customers in a dataset.

Parameters

Parameter	Description
iRetRows	The number of records to be returned. "-1" = return all.

Returns

Type	Description
WSDataSet	WSDataset (CustomerId, CustomerName, UserDefinedCustomerId) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webCustomer.CustomerWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSDataset = myProxy.GetList(-1)
```

Related information

"Customer" on page 795

Customer: GetResourcesByUserDefinedId

```
Public Function GetResourcesByUserDefinedId(ByVal sUserDefinedId As String) As
WSDataSet
```

Purpose

Retrieve a list of customer resources for a specified user-defined ID.

Parameters

Parameter	Description
sUserDefinedId	User-defined ID for which to look up resources

Returns

Type	Description
WSDataSet	WSDatase (ResourceId, Name). Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Customer" on page 795

Customer: GetUDF

```
Public Function GetUDF(ByVal retOption As CPUDFReturnType, ByVal entityId As
String, ByVal actionResourceId As String) As WSString
```

Purpose

Retrieve UDF information for customer in an XML string.

Parameters

Parameter	Description
retOption	The CPUDFReturnType Values are: • WithStructure = 0 – returns UDF field data with full field structure • OnlyValues = 1 – • returns only UDF fields with data, does not include any field structure or options • OnlyStructure = 2 – • returns only UDF XML structure, no field data • OnlyStructureAndOptions = 3 – • returns UDF structure and option data without field data • WithStructureAndOptions = 4 – returns UDF field data with full structure and all field options (except entity-based and conditional codes)
entityId	CustomerId. An empty string is allowed.
actionResourceId	The ResourceId of the user that called the method. An An empty string is allowed.

Returns

Type	Description
WSString	UDF information for customer in an XML string Access the returned string through the .Value property of WSString

Remarks

If entityId is empty, the retOption are only available on OnlyStructure (2) and OnlyStructureAndOptions (3).

If actionResourceId is empty, login user ResourceId is used.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webCustomer.CustomerWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSString = myProxy.GetUDF
(CPUDFReturnType.WithStructureAndOptions,"{CA8AC6E5-5F9A-41CA-BD57-
9A35D28A7E76}","")
```

Related information

"Customer" on page 795

Customer: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal sCustomerId As String, ByVal codeName
As String, ByVal searchString As String) As WSString
```

Purpose

Retrieve UDF code options in XML format for specified customer UDF code.

Parameters

Parameter	Description
sCustomerId	CustomerId. An An empty string is allowed.
codeName	Name of the UDF code, for example, "Code3"
searchString	Returns the options that start with this string. An empty string is allowed.

Returns

Type	Description

WSString	`WSString{`
	`WSException WSException;`
	`String value;`
	`}`

Remarks

- If the sCustomerId is not provided, returns default UDF values.
- Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webCustomer.CustomerWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSString = myProxy.GetUDFCodeOptions("{CA8AC6E5-5F9A-
41CA-BD57-9A35D28A7E76}","Code2","")
```

Related information

"Customer" on page 795

Customer: GetXMLStructure

```
Public Function GetXMLStructure() As WSString
```

Purpose

Returns the XML structure of the ApiCustomer object

Parameters

None

Returns

Type	Description
WSString	String with the XML structure of the ApiCustomer object. Access the returned string through the .Value property of WSString

Remarks

None

Example

```
Dim myProxy As New webCustomer.CustomerWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSString = myProxy.GetXMLStructure()
```

Related information

"Customer" on page 795

"ApiCustomer XML" on page 75

Customer: SaveUDF

```
Public Function SaveUDF(ByVal sXMLUDF As String, ByVal needValidate As
Boolean, ByVal bypassMetadata As CPMetadataCheck) As WSInt32
```

Purpose

Saves (Insert/Update) UDF information for a customer.

Parameters

Parameter	Description
sXMLUDF	UDF string in XML format to be saved.
needValidate	Flag that determines whether validation is required for sXMLUDF.
bypassMetadata	Determines the level of metadata checking for sXMLUDF. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2 If needValidate is False, then bypassMetadata must be set to CPMetadataCheck.BypassAll = 1.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

To obtain the correct UDF XML format, call the GetUDF method.

Example

```
Dim myProxy As New webCustomer.CustomerWse
Dim strXMLUDF as string ="<root>…</root>"
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSInt32 = myProxy.SaveUDF(strXMLUDF, True, 0)
```

Related information

"Customer" on page 795

"Customer: GetUDF" on page 810

Customer: SetPropertiesByXml

```
Public Function SetPropertiesByXml(ByVal sXML As String) As WSCustomer
```

Purpose

Sets the properties of the object via an XML string.

Parameters

Parameter	Description
sXML	XML string of the customer object with data to load into the object.

Returns

Type	Description
WSCustomer	Returns an ApiCustomer object. Access the object through the .Value property of WSCustomer.

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods. If you use the SetPropertiesByXML method, you must pass in the entire XML string.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

To obtain the correct XML format, call the GetXMLStructure method.

Example

```
Dim myProxy As New webCustomer.CustomerWse
Dim sXML as string ="<root>…</root>"
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSCustomer = myProxy.SetPropertiesByXML(sXML)
```

Related information

"Customer" on page 795

"Customer: GetXMLStructure" on page 812

"Customer: CreateByXML" on page 798

"Customer: UpdateByXML" on page 817

Customer: Update

```
Public Function Update(ByVal oCustomer As ApiCustomer) As WSInt32
```

Purpose

Update a customer record.

Parameters

Parameter	Description
oCustomer	ApiCustomer object to be updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

It is recommended to use GetById to populate ApiCustomer with the existing data before the Update method is called.

Example

```
Dim myProxy As New webCustomer.CustomerWse
Dim myCustomer As New webCustomer.ApiCustomer
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webCustomer.WSCustomer = myProxy.GetById("{CA8AC6E5-5F9A-41CA-
BD57-9A35D28A7E76}")
If Not oRet.WSException.HaveErrors Then
   myCustomer = oRet.value
```

```
End If
With myCustomer
   .SalesStatus.Id = "C"
   .BusinessAddress1.AddressLine = "30 Leek Crescent, Suite 400"
   .BusinessAddress1.City = "Richmond Hill"
   .BusinessAddress1.StateProvince.Id = "{27044F87-175F-4F45-A8AD-
12750600EE49}"
   .BusinessAddress1.Country.Id = "CAN"
   .BusinessAddress1.PostalCode = "L4B 4N4"
   .
   .
End With
Dim oRetInt32 As webCustomer.WSInt32 = myProxy.Update(myCustomer)
```

Related information

"Customer" on page 795

"Customer: GetById" on page 801

Customer: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, ByVal sCustomerId As String)
As WSInt32
```

Purpose

Update the database using an XML string of the Customer object containing update information.

Parameters

Parameter	Description
sXML	A customer XML string with updated fields.
sCustomerId	Customer ID of the customer being updated.

Returns

Not applicable

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The method uses the following sequence to find the customer ID:

1. If the sCustomerId parameter is passed in, the method uses this value for the customer ID.
2. If this fails, the method attempts to extract the customer ID from <customerid> in the XML.
3. If this fails, the customer ID is taken from the object properties.
4. If this fails, an attempt is made to look up the customer ID using <userdefinedcustomerid> in the XML. If <userdefinedcustomerid> has a value, but the value is invalid or duplicated, then you will get an error and no further attempts are made to look up the customer ID.
5. If <userdefinedcustomerid> is empty, an attempt is made to look up the customer ID using <name> in the XML.

Example

Not available

Related information

"Customer" on page 795

"ApiCustomer XML" on page 75

Engagement

The Engagement object allows users to create, retrieve, update and delete contracts within the Changepoint database. Used with the Engagement object, the following objects are also exposed by the Changepoint API:

Namespace

```
http://changepoint.com/changepoint/CPWebService/Engagement
```

URL

http://webserver/CPWebService/Objects/CPEngagement/Engagement.asmx

Methods

Properties

For more information, see the "ApiEngagement" section on page 98.

Related information

"EngBillingRate" on page 851

"EngBillingRateHistory" on page 861

"EngFixedFee" on page 867

"EngFixedFeeItem" on page 876

"EngFixedFeeItemSplitBillOverride" on page 882

"EngFixedFeeSplitBillOverride" on page 883

"EngProduct" on page 883

"EngProductSplitBillOverride" on page 895

"EngProjectedResource" on page 895

"EngRequestProcessingRule" on page 904

"EngRequestSLA" on page 917

"EngRevRec" on page 927

"EngSplitBillingRule" on page 931

"EngWorkCodeLocation" on page 937

Engagement: Add

```
Public Function Add(ByRef sId As String, ByVal oEngagement As ApiEngagement)
As WSInt32
```

Purpose

Add a contract and its subobjects to the Changepoint database.

Parameters

Parameter	Description
sId	Place holder for the engagementId returned if successful
oEngagement	Populated ApiEngagement object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Ensure all mandatory properties are set.

The contract name must be unique.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
Dim myEng As New webEngagement.ApiEngagement
Dim sId As String = String.Empty
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
With myEng
   .Name = "Changepoint Canada"
   .Customer = New webEngagement.Identity()
   .Customer.Id = "{40C01201-37EC-47E1-8226-F505760C42F9}"
   .EngagementManager = New webEngagement.Identity()
   .EngagementManager.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
   .BillingOffice = New webEngagement.Identity()
   .BillingOffice.Id = "{5757A330-3476-11D3-807A-00105A0B7C01}"
   .MainContact = New webEngagement.Identity()
   .MainContact.Id = "{CAC61D37-5A9F-420D-8786-9CDDA74A1F3B}"
   .SecondLevelApprover = New webEngagement.Identity()
   .SecondLevelApprover.Id = "{CA59FBCB-017A-4FEF-A07F-17A7A4315C63}"
   .AssociatedWorkgroup = New webEngagement.Identity()
```

```
.AssociatedWorkGroup.Id = "{7712F1B0-F950-4FD6-B115-BAA2E9FCB468}"
.InvoiceFormat = New webEngagement.Identity()
.InvoiceFormat.Id = "{27DF0C5B-E5DF-11D2-882F-006097B596A6}"
.SupportDesk = New webEngagement.Identity()
.SupportDesk.Id = "{548B0D15-0305-4827-A666-E74046AFDD0F}"
.BillToWorkLocationGroup = New webEngagement.Identity()
.BillToWorkLocationGroup.Id = "{333A206C-513E-4CAE-841A-EA73A82E8E81}"
.BillToWorkLocation = New webEngagement.Identity()
.BillToWorkLocation.Id = "{0409A21B-EFB9-42C4-ADB3-617279F524EF}"
.TimeApprover = New webEngagement.Identity()
.TimeApprover.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
.AltTimeApprover = New webEngagement.Identity()
.AltTimeApprover.Id = "{1F0C4CFB-9906-4895-9C9E-43094E0E6F30}"
.ExpenseApprover = New webEngagement.Identity()
.ExpenseApprover.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
.AltExpenseApprover = New webEngagement.Identity()
.AltExpenseApprover.Id = "{1F0C4CFB-9906-4895-9C9E-43094E0E6F30}"
.AvailableToAll = True
.TaxOnRecordDate = True
.BypassMetadataCheck = webEngagement.CPMetadataCheck.OnlyMandatory
.ContractAmount = 12345687
.ContractStartDate = CDate("01/01/2008")
.ContractEndDate = CDate("12/31/2008")
.Description = "First contract"
.FederalAudit = False
.BillingCurrency = "CAD"
.TimeApprovalDelegated = False
.AllowProjectOverride = True
.OvertimePercentage = 100
.Overtime = True
.EnableAutoUpdateRate = False
.EnableAutoUpdateCost = False
.UseTwoLevelOnWOffs = True
.UseTwoLevelOnContractOverage = False
.UseTwoLevelApproval = False
.ExpenseMinimum = 1000
.ExpenseApprovalDelegated = False
.Billable = True
.PreventTaxChanges = True
.MaximumAmount = 2000000
.InvoiceMaximum = True
.ExpenseBillingPercentage = 100
.ExpenseBillingType = "A"
.BillingType = "D"
.PaymentTerms = "{954F9D11-D72E-11D2-9AE8-006008A4D883}"
.EngagementStatus = "W"
.AllowEditToAll = True
.AllowPrjMgrBillCont = True
```

```
    .EnableBatchInvoice = True
    .EnableProjectTimeApproval = True
    .ExpenseGLA = "EM"
    .OverrideTaskRestriction = True
    .SplitBilling = False
    .SupportDeskOverride = False
    .Template = True
    .BillingAddress = 1
    .sXMLUDF = "<root><udf>…</udf></root>"
    .
    .
End With
Dim oRet As webEngagement.WSInt32 = myProxy.Add(sId, myEng)
```

Related information

"Engagement" on page 818

Engagement: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, ByRef sId As String) As
WSInt32
```

Purpose

Creates a contract and its subobjects using an XML string of the Engagement object.

Parameters

Parameter	Description
sXML	A new contract XML string which must be passed based on the Engagement xml schema provided by GetXMLStructure. You can pass a partial schema of the main Engagement or subobjects as long as you include the mandatory fields, the XML is wrapped in the <Engagement> </Engagement> nodes, and subobject nodes are valid.
sId	ByRef parameter to pass back the new EngagementId.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 using WSInt32.Value .

Remarks

The ApiEngagement XML structure can be obtained by the GetXMLStructure or GetByXML method.

BillingRateHistory, FixedFeeItem and SplitBillingRule cannot be added using CreateByXML. Use UpdateByXML instead.

For details of the use of UDF default values, see "UDF default values logic for CreateByXML" on page 746.

Example

Not available

Related information

"Engagement" on page 818.

Engagement: Delete

```
Public Function Delete(ByVal sId As String) As WSInt32
```

Purpose

Remove the specified contract.

Parameters

Parameter	Description
sId	ID of the contract record.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSInt32 = myProxy.Delete("{C218824A-67C2-4F34-B4F6-
8EB48728FCF3}")
```

Related information

"Engagement" on page 818

Engagement: Exists

```
Public Function Exists(ByVal sId As String) As WSBoolean
```

Purpose

Verify whether the contract exists

Parameters

Parameter	Description
sId	ID of the contract to verify.

Returns

Type	Description
WSBoolean	True if the contract exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSBoolean = myProxy.Exists("{C218824A-67C2-4F34-
B4F6-8EB48728FCF3}")
```

Related information

"Engagement" on page 818

Engagement: GetBillingOffice

```
Public Function GetBillingOffice(ByVal sBillingOfficeId As String) As
WSDataSet
```

Purpose

Retrieve billing office information for the specified billing office.

Parameters

Parameter	Description
sBillingOfficeId	Billing office ID

Returns

Type	Description
WSDataSet	WSDataset (BillingOfficeId, Description, BaseCurrency, Deleted, CostCenter, RecognizeRevenue, CostRateToUse, FedAudit, AuditEnabledDate, InvoiceFormatSelection, Active, DefaultInvoiceFormat, DefaultPaymentTerms, PreventRateChanges, UseInvoiceApproval, PreventTaxChanges, PreventWriteUps, UseTwoLevelApproval, AllowEngagementOverride, SecondLevelApprover, UseTwoLevelOnContractOverage, UseTwoLevelOnWOffs, EngagementBillable, ForceFixedFeeSelection, RestrictTaskAssignments, AllowOverrideTaskRestriction, AdditionalItem) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSDataSet = myProxy.GetBillingOffice("{5757A330-
3476-11D3-807A-00105A0B7C01}")
```

Related information

"Engagement" on page 818

Engagement: GetByCustomerId

```
Public Function GetByCustomerId(ByVal sCustomerId As String) As WSDataSet
```

Purpose

Retrieve a list of billable contracts for a specified customer

Parameters

Parameter	Description
sCustomerId	Customer ID

Returns

Type	Description
WSDataSet	WSDataset (EngagementId, Name, Description, CustomerId, BillingOfficeId) `Access the dataset through the .Value property of WSDataSet.`

Remarks

Split billing contracts are also included.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSDataSet = myProxy.GetByCustomerId("{40C01201-37EC-
47E1-8226-F505760C42F9}")
```

Related information

"Engagement" on page 818

Engagement: GetByCustomerName

```
Public Function GetByCustomerName(ByVal sCustomerName As String) As WSDataSet
```

Purpose

Retrieve a list of contracts for a specified customer

Parameters

Parameter	Description
sCustomerName	Customer name

Returns

Type	Description
WSDataSet	WSDataset (EngagementId, Name, AlternateName, UserDefinedEngagementId, CustomerId, CustomerName, BillingOfficeId, BillingOfficeName) Access the dataset through the .Value property of WSDataSet.

Remarks

Returns all contracts except split billing contracts.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSDataSet = myProxy.GetByCustomerName("Changepoint
Corporation")
```

Related information

"Engagement" on page 818

Engagement: GetById

```
Public Function GetById(ByVal sId As String) As WSEngagement
```

Purpose

Fills the ApiEngagement object and its subobjects with contract information of the specified sId passed in the parameter.

Parameters

Parameter	Description
sId	Engagement ID

Returns

Type	Description
WSEngagement	Returns an ApiEngagement object. Access the object through the .Value property of WSEngagement.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSEngagement = myProxy.GetById("{C218824A-67C2-4F34-
B4F6-8EB48728FCF3}")
```

Related information

"Engagement" on page 818

Engagement: GetByName

```
Public Function GetByName(ByVal sName As String) As WSDataSet
```

Purpose

Retrieve a contract for a specified name

Parameters

Parameter	Description
sName	Name of engagement

Returns

Type	Description
WSDataSet	WSDataset (EngagementId, Name, AlternateName, UserDefinedEngagementId, CustomerId, CustomerName, BillingOfficeId, BillingOfficeName) Access the dataset through the .Value property of WSDataSet.

Remarks

Split billing contracts won't be returned.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSDataSet = myProxy.GetByName("Installation phase")
```

Related information

"Engagement" on page 818

Engagement: GetByUserDefinedId

```
Public Function GetByUserDefinedId(ByVal sUserDefinedEngagementId As String)
As WSEngagement
```

Purpose

Fills the ApiEngagement object with contract information of the specified user defined contract ID.

Parameters

Parameter	Description
sUserDefinedEngagementId	User defined contract ID

Returns

Type	Description
WSEngagement	Returns an ApiEngagement object. `Access the object through the .Value property of WSEngagement.`

Remarks

UserDefinedEngagementId is treated as a unique identifier, so error number -11 is returned if multiple records found.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSEngagement = myProxy.GetByUserDefinedId("Eng-2008-
000001")
```

Related information

"Engagement" on page 818

Engagement: GetByXML

```
Public Function GetByXML(ByVal sXML As String, ByVal sEngagementId As String)
As WSString
```

Purpose

Takes the XML string passed in (sXML) and returns the string filled with data for the Engagement specified by sEngagementId, and the Engagement’s subobjects.

Parameters

Parameter	Description
sXML	XML string of the Engagement object and its subobjects with the fields to be populated specified.
sEngagementId	Engagement ID. • If no Engagement ID is passed in, the XML string (sXML) is examined • if there is no Engagement ID in sXML then the object's Engagement ID is selected. • If there still is no valid EngagementId, the method returns an error.

Returns

Type	Description
WSString	An XML string mirroring sXML with data inserted or the entire XML of the Engagement object including data. Access the returned string using WSString.Value.

Remarks

If sXML = "" then the XML string provided by GetXMLStructure is used, and the entire Engagement and its subobjects are passed back, populated with all available data.

To get only specific data passed back, include in sXML only the corresponding subobjects and elements wrapped in <Engagement> </Engagement>.

Example

Not available

Related information

"Engagement" on page 818.

Engagement: GetCustomerById

```
Public Function GetCustomerById(ByVal sId As String) As WSDataSet
```

Purpose

Retrieve customer information for a contract by engagementId

Parameters

Parameter	Description
sId	Engagement ID

Returns

Type	Description
WSDataSet	WSDataset (CustomerId, CustomerName, EngagementName) `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSDataSet = myProxy.GetCustomerById("{C218824A-67C2-
4F34-B4F6-8EB48728FCF3}")
```

Related information

"Engagement" on page 818

Engagement: GetCustomers

```
Public Function GetCustomers() As WSDataSet
```

Purpose

Retrieve a list of customers for which contracts can be created.

Parameters

None

Returns

Type	Description
WSDataSet	WSDataset (CustomerId, Name, AlternateName, UserDefinedCustomerId, Business1Address, Business1AddressLine2, Business1AddressLine3, Business1City, Business1Province, Business1Postal, Business1Country) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSDataSet = myProxy.GetCustomers()
```

Related information

"Engagement" on page 818

Engagement: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As WSString
```

Purpose

Returns the EngagementId based on the UDF Text field and value.

Parameters

Parameter	Description
sUDFField	UDF text field name (for example, Text1)
sUDFValue	UDF text field value

Returns

Type	Description
WSString	String with the EngagementId or an empty string. Access the returned string through the .Value property of WSString

Remarks

None

Example

Not available

Related information

"Engagement" on page 818

Engagement: GetList

```
Public Function GetList(ByVal iRetRows As Int16) As WSDataSet
```

Purpose

Retrieve a list of contracts.

Parameters

Parameter	Description
iRetRows	The number of records to be returned. "-1" = return all.

Returns

Type	Description
WSDataSet	WSDataset (EngagementId, Name, AlternateName, UserDefinedEngagementId, CustomerId, CustomerName, BillingOfficeId, BillingOfficeName) `Access the dataset through the .Value property of WSDataSet.`

Remarks

Split billing contracts are not included in the list.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSDataSet = myProxy.GetList(-1)
```

Related information

"Engagement" on page 818

Engagement: GetPreferredResources

```
Public Function GetPreferredResources(ByVal sEngagementId As String, ByVal
sResourceId As String) As WSDataSet
```

Purpose

Retrieve a list of resources that can use this contract

Parameters

Parameter	Description
sEngagementId	Engagement ID
sResourceId	Resource ID

Returns

Type	Description
WSDataSet	WSDataset (ResourceId, ResourceName, AltResourceName) Access the dataset through the .Value property of WSDataSet.

Remarks

Upons success, returns a list of resources that have the View Engagements, Create Engagements, or Edit Engagements security features. The list includes the contract manager and contract creator if they have these security features as well.

If sResourceId = "", then login user resourceId is used.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSDataSet = myProxy.GetPreferredResources("
{C218824A-67C2-4F34-B4F6-8EB48728FCF3}","")
```

Related information

"Engagement" on page 818

Engagement: GetResourcesWithEngAccess

```
Public Function GetResourcesWithEngAccess() As WSDataSet
```

Purpose

Retrieve a list of resources that have the View Engagements, Create Engagements, or Edit Engagements security features.

Parameters

None

Returns

Type	Description
WSDataSet	WSDataset (ResourceId, ResourceName, AltResourceName) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSDataSet = myProxy.GetResourcesWithEngAccess()
```

Related information

"Engagement" on page 818

Engagement: GetStatusByBillingOfficeId

```
Public Function GetStatusByBillingOfficeId(ByVal sBillingOfficeId As String)
As WSDataSet
```

Purpose

Retrieve the statuses of the contract workflow for a billing office.

Parameters

Parameter	Description
sBillingOfficeId	Billing office ID

Returns

Type	Description
WSDataSet	WSDataset (Code, Description, Step, EngagementWorkFlowId, Billable) `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSDataSet = myProxy.GetStatusByBillingOfficeId("
{5757A330-3476-11D3-807A-00105A0B7C01}")
```

Related information

"Engagement" on page 818

Engagement: GetUDF

```
Public Function GetUDF(ByVal sEngagementId As String, ByVal sBillingOfficeId
As String, ByVal retOption As CPUDFReturnType) As WSString
```

Purpose

Retrieve UDF (configurable field) information for a contract in an XML string.

Parameters

Parameter	Description
sEngagementId	EngagementId. An empty string is allowed.
sBillingOfficeId	BillingOfficeId

Parameter	Description
retOption	The CPUDFReturnType Values are: • WithStructure = 0 – returns UDF field data with full field structure • OnlyValues = 1 – returns only UDF fields with data, does not include any field structure or options • OnlyStructure = 2 – returns only UDF XML structure, no field data • OnlyStructureAndOptions = 3 – returns UDF structure and option data without field data • WithStructureAndOptions = 4 – returns UDF field data with full structure and all field options (except entity-based and conditional codes)

Returns

Type	Description
WSString	XML string with UDF information Access the returned string through the .Value property of WSString

Remarks

If sEngagementId = "", then default UDF values will be returned.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSString = myProxy.GetUDF("{E90A4835-786F-4AFE-89ED-
88A959608277}","{27AF2827-0E98-430A-B2A0-
9E57665E70BB}",CPUDFReturnType.WithStructureAndOptions)
```

Related information

"Engagement" on page 818

Engagement: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal sEngagementId As String, ByVal
sBillingOfficeId As String, ByVal codeName As String, ByVal searchString As
String) As WSString
```

Purpose

Retrieve UDF code options in XML format for specified contract UDF code.

Parameters

Parameter	Description
sEngagementId	EngagementId. An empty string is allowed.
sBillingOfficeId	BillingOfficeId.
codeName	The name of the UDF code, for example, "Code3"
SearchString	The text that the code options must start with. An empty string is allowed.

Returns

Type	Description
WSString	XML string of the option list that exists for a specified code. Access the returned string through the .Value property of WSStringv

Remarks

If sEngagementId = "", then default UDF values will be returned.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
```

```
Dim oRet As webEngagement.WSString = myProxy.GetUDFCodeOptions("{E90A4835-
786F-4AFE-89ED-88A959608277}","{27AF2827-0E98-430A-B2A0-
9E57665E70BB}","Code3", "Changepoint Corp")
```

Related information

"Engagement" on page 818

Engagement: GetWorkDefaults

```
Public Function GetWorkDefaults(ByVal sId As String) As WSDataSet
```

Purpose

Retrieve the default work code and work location information for an Engagement

Parameters

Parameter	Description
sId	Engagement ID

Returns

Type	Description
WSDataSet	WSDataset (EngagementId, WorkLocationGroupId, WorkLocationId, WorkCodeCategoryId, WorkCodeId, AllowRequestTimeLocOverride, AllowRequestTimeWorkOverride, BillToWorklocationGroupId, BillToWorkLocationId) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSDataSet = myProxy.GetWorkDefaults("{C218824A-67C2-
4F34-B4F6-8EB48728FCF3}")
```

Related information

"Engagement" on page 818

Engagement: GetXMLStructure

```
Public Function GetXMLStructure() As WSString
```

Purpose

Returns the XML structure of the Engagement object including subobjects

Parameters

None

Returns

Type	Description
WSString	String with the XML structure of the ApiEngagement object and subobjects. Access the returned string using WSString.Value

Remarks

```
This method is used as a template for CreateByXML and UpdateByXML.
```

Example

Not available

Related information

"Engagement" on page 818

Engagement: HasP2AInvoice

```
Public Function HasP2AInvoice(ByVal sId As String) As WSBoolean
```

Purpose

Check if the contract has a pending second level approval invoice.

Parameters

Parameter	Description
sId	Engagement ID

Returns

Type	Description
WSBoolean	True if there is a pending second level approval invoice for this contract, else False Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSBoolean = myProxy.HasP2AInvoice("{C218824A-67C2-
4F34-B4F6-8EB48728FCF3}")
```

Related information

"Engagement" on page 818

Engagement: LockEngagement

```
Public Function LockEngagement(ByVal sEngagementId As String, ByVal bLock As
Boolean) As WSString
```

Purpose

Lock a contract when in edit mode. Releases the contract when edit is finished.

Parameters

Parameter	Description
sEngagementId	Engagement ID
bLock	True for locking contract. False for releasing contract

Returns

Type	Description
WSString	Returns resource name if contract is locked and an empty string if contract is not locked. Access the returned string through the .Value property of WSString

Remarks

It's not necessary to call this method before the Update method is called.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSString = myProxy.LockEngagement("{C218824A-67C2-
4F34-B4F6-8EB48728FCF3}",True)
```

Related information

"Engagement" on page 818

Engagement: PrintBillingAddress

```
Public Function PrintBillingAddress(ByVal sCustomerId As String, ByVal
sContactID As String, ByVal iAddress As Integer) As WSString
```

Purpose

Retrieve billing address formatted by country standard for a customer.

Parameters

Parameter	Description
sCustomerId	Customer ID
sContactId	Main contact ID
iAddress	iAddress values and corresponding returns are as follows: • 1 Customer Business Address1 • 2 Customer Business Address2 • 3 Customer Business Address3 • 4 Contact Business Address • 5 Contact Home Address • 6 Contact Other Address

Returns

Type	Description
WSString	String with the formatted billing address. Access the returned string through the .Value property of WSString

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSString = myProxy.PrintBillingAddress("{40C01201-
37EC-47E1-8226-F505760C42F9}", "{CAC61D37-5A9F-420D-8786-9CDDA74A1F3B}", 1)
```

Related information

"Engagement" on page 818

Engagement: SaveUDF

```
Public Function SaveUDF(ByVal sXMLUDF As String, ByVal bypassMetadata As
CPMetadataCheck) As WSInt32
```

Purpose

Saves (Insert/Update) UDF information for a contract.

Parameters

Parameter	Description
sXMLUDF	UDF string in XML format to be saved.
bypassMetadata	Determines the level of metadata checking for sXMLUDF. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Call the GetUDF method to obtain the correct UDF XML format.

Example

```
Dim myProxy As New webEngagement.EngagementWse
Dim strXMLUDF as string ="<root>…</root>"
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSInt32 = myProxy.SaveUDF(strXMLUDF, 2)
```

Related information

"Engagement" on page 818

"UDF XML" on page 742

Engagement: Update

```
Public Function Update(ByVal oEngagement As ApiEngagement) As WSInt32
```

Purpose

Update the general information of a contract and its subobjects.

Parameters

Parameter	Description
oEngagement	ApiEngagement object to be updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Use GetById to populate ApiEngagement with the existing data before the Update method is called.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngagement.EngagementWse
Dim myEng As New webEngagement.ApiEngagement
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngagement.WSEngagement = myProxy.GetById("{C218824A-67C2-4F34-
B4F6-8EB48728FCF3}")
If Not oRet.WSException.HaveErrors Then
myEng = oRet.value
End If
With myEng
   .TimeApprover = New webEngagement.Identity()
```

```
    .TimeApprover.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
    .AltTimeApprover = New webEngagement.Identity()
    .AltTimeApprover.Id = "{CA59FBCB-017A-4FEF-A07F-17A7A4315C63}"
    .ExpenseApprover = New webEngagement.Identity()
    .ExpenseApprover.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
    .AltExpenseApprover = New webEngagement.Identity()
    .AltExpenseApprover.Id = "{CA59FBCB-017A-4FEF-A07F-17A7A4315C63}"
    .AuditDetail = "Change the Approvers."
    .
    .
End With
Dim oRetInt32 As webEngagement.WSInt32 = myProxy.Update(myEng)
```

Related information

"Engagement" on page 818

"Engagement: GetById" on page 829

Engagement: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, ByVal sEngagementId As
String) As WSInt32
```

Purpose

Updates the database using an XML string of the Engagement object and its subobjects containing update information.

Parameters

Parameter	Description
sXML	A Engagement XML string with content to be used to update the fields in the Engagement and its subobjects.
sEngagementId	Engagement ID of the Engagement being updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 using WSInt32.Value.

Note: The Engagement Action must be 2, to update the Engagement, or 9 to leave the Engagement unchanged. Subobject Actions can be 1, 2, or 9.
For more information on Actions, see "ApiEngagement XML" on page 113.

Remarks

BillingRateHistory, FixedFeeItem and SplitBillingRule can be added and updated using UpdateByXML.

BillingRate and BillingRateHistory require the same tag and same index # to be used in <billingrateindex> when updating or adding a new BillingRateHistory record.

Possible values for EngagementWorkcode and EngagementWorklocation selected attribute are 1 (selected) or 0 (not selected).

See .

Example

Not available

Related information

"Engagement" on page 818.

EngBillingRate

The EngBillingRate object represents billing rates set in the contract.

Namespace

`http://changepoint.com/changepoint/CPWebService/EngBillingRate`

URL

`http://webserver/CPWebService/Objects/CPEngagement/EngBillingRate.asmx`

Methods

Properties

For more information, see the "ApiEngBillingRate" section on page 145.

Related information

"Engagement" on page 818

EngBillingRate: Add

```
Public Function Add(ByRef sId As String, ByVal oEngBillingRate As
ApiEngBillingRate) As WSInt32
```

Purpose

Insert a billing rate for a contract.

Parameters

Parameter	Description
sId	Placeholder of the returned value of new RateId
oEngBillingRate	Populated ApiEngBillingRate object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Ensure all mandatory properties are set.

If there are no values assigned to CostRate and OTCostRate, zero value will be given and saved for these fields. It's recommended to have these properties filled before Add is called.

Both BillingOffice and BillingRole must have values assigned in order to be saved. BillingOffice cannot be saved when adding a resource rate only.

Both resource rate and billing rate should be already defined in Changepoint.

Contract info associated with the billing rate can be saved when UpdateEngInfoRequired is set to True.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngBillingRate.EngBillingRateWse
Dim myBillingRate As New webEngBillingRate.ApiEngBillingRate
Dim sId As String = String.Empty
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
With myBillingRate
   .CPConnection = myCon
   .BillingOffice = New webEngBillingRate.Identity()
   .BillingOffice.Id = "{27AF2827-0E98-430A-B2A0-9E57665E70BB}"
   .Engagement = New webEngBillingRate.Identity()
   .Engagement.Id = "{70852075-3A41-4406-955E-B96C777AB431}"
   .BillingRole = New webEngBillingRate.Identity()
   .BillingRole.Id = "{3FED4827-F58F-4BB1-9BF4-23B06CC0BD09}"
   .Resource = New webEngBillingRate.Identity()
   .Resource.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
   .RequestOnly = True
   .DiscountPercentage = 50
```

```
    .CostRate = 750
    .OTCostRate = 100
    .BillingCurrency = "USD"
    .CostCurrency = "USD"
    .
    .
End With
Dim oRet As webEngBillingRate.WSInt32 = myProxy.Add(sId, myBillingRate)
```

Related information

"EngBillingRate" on page 851

EngBillingRate: Delete

```
Public Function Delete(ByVal sId As String) As WSInt32
```

Purpose

Remove a billing rate and its history records.

Parameters

Parameter	Description
sId	RateId

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngBillingRate.EngBillingRateWse
```

```
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngBillingRate.WSInt32 = myProxy.Delete("{CC0AA774-969C-43B8-
A64E-1A9A91B50F94}")
```

Related information

"EngBillingRate" on page 851

EngBillingRate: Exists

```
Public Function Exists(ByVal sId As String) As WSBoolean
```

Purpose

Check whether this billing rate exists or not

Parameters

Parameter	Description
sId	RateId

Returns

Type	Description
WSBoolean	True if the billing rate exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngBillingRate.EngBillingRateWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngBillingRate.WSBoolean = myProxy.Exists("{CC0AA774-969C-43B8-
A64E-1A9A91B50F94}")
```

Related information

"EngBillingRate" on page 851

EngBillingRate: GetBillingRoles

```
Public Function GetBillingRoles(ByVal sBillingOfficeId As String) As WSDataSet
```

Purpose

Retrieve billing roles for the billing office.

Parameters

Parameter	Description
sBillingOfficeId	Billing office ID.

Returns

Type	Description
WSDataSet	WSDataset (BillingRoleId, Description) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngBillingRate.EngBillingRateWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngBillingRate.WSDataSet = myProxy.GetBillingRoles("{5757A330-
3476-11D3-807A-00105A0B7C01}")
```

Related information

"EngBillingRate" on page 851

EngBillingRate: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As WSDataSet
```

Purpose

Retrieve all billing rate records for a contract.

Parameters

Parameter	Description
sEngagementId	ID of the contract.

Returns

Type	Description
WSDataSet	WSDataset (BillingOfficeName, BillingRoleDescription, CurrencyDescription, ResourceName, AlternateResource, BillingRateID, ResourceId, BillingOfficeId, BillingRoleId, StandardRate, DiscountPercentage, BillingRate, Currency, CostRate, CostCurrency, RequestOnly, BillingOfficeRateId, OTCostRate.) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngBillingRate.EngBillingRateWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngBillingRate.WSDataSet = myProxy.GetByEngagementId("
{C218824A-67C2-4F34-B4F6-8EB48728FCF3}")
```

Related information

"EngBillingRate" on page 851

EngBillingRate: GetById

```
Public Function GetById(ByVal sId As String) As WSEngBillingRate
```

Purpose

Fills the ApiEngBillingRate object with billing rate of the specified sId passed in the parameter.

Parameters

Parameter	Description
sId	ID of the billing rate record.

Returns

Type	Description
WSEngBillingRate	Returns an ApiEngBillingRate object. `Access the object through the .Value property of WSEngBillingRate.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngBillingRate.EngBillingRateWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngBillingRate.WSEngBillingRate = myProxy.GetById("{CC0AA774-
969C-43B8-A64E-1A9A91B50F94}")
```

Related information

"EngBillingRate" on page 851

EngBillingRate: GetEngagementInfo

```
Public Function GetEngagementInfo(ByVal sEngagementId As String) As WSDataSet
```

Purpose

Retrieve contract information associated with billing rate.

Parameters

Parameter	Description
sEngagementId	The ID of the contract.

Returns

Type	Description
WSDataSet	WSDataset (BillingCurrency, BillingType, CustomerId, FedAudit, CostCenterId, DefaultBillingRole, EnableAutoUpdateCost, EnableAutoUpdateRate, ExpenseMinimum, Overtime, OvertimePercentage) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

```
Dim myProxy As New webEngBillingRate.EngBillingRateWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngBillingRate.WSDataSet = myProxy.GetEngagementInfo("
{C218824A-67C2-4F34-B4F6-8EB48728FCF3}")
```

Related information

"EngBillingRate" on page 851

EngBillingRate: UpdateEngagementInfo

```
Public Function UpdateEngagementInfo(ByVal oEngBillingRate As
ApiEngBillingRate) As WSInt32
```

Purpose

Update contract information associated with billing rate.

Parameters

Parameter	Description
oEngBillingRate	Populated ApiEngBillingRate object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Engagement.Id and RateId are required.

Updates the following fields: CostCenter, DefaultBillingRole, EnableAutoUpdateCost, EnableAutoUpdateRate, ExpenseMinimum, Overtime, OvertimePercentage

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngBillingRate.EngBillingRateWse
Dim myBillingRate As New webEngBillingRate.ApiEngBillingRate
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
With myBillingRate
   .Engagement = New webEngBillingRate.Identity()
   .Engagement.Id = "{C218824A-67C2-4F34-B4F6-8EB48728FCF3}"
   .RateId = "{C796C14E-0FD2-4EDE-95E0-FCB31FF7EB93}"
   .ExpenseMinimum = 10000
   .Overtime = True
   .OvertimePercentage = 50
End With
Dim oRet As webEngBillingRate.WSInt32
oRet = myProxy.UpdateEngagementInfo(myBillingRate)
```

Related information

"EngBillingRate" on page 851

EngBillingRateHistory

The EngBillingRateHistory object allows users to retrieve and update the rate history for contracts.

Namespace

`http://changepoint.com/changepoint/CPWebService/EngBillingRateHistory`

URL

http://webserver/CPWebService/Objects/CPEngagement/EngBillingRateHistory.asmx

Methods

Properties

For more information, see the "ApiEngBillingRateHistory" section on page 156.

Related information

"Engagement" on page 818

EngBillingRateHistory: Add

```
Public Function Add(ByRef sId As String, ByVal oEngBillingRateHistory As
ApiEngBillingRateHistory) As WSInt32
```

Purpose

Insert a new billing rate effective date for a contract.

Parameters

Parameter	Description
sId	Placeholder of the returned value of new billing rates effective date ID.
oEngBillingRateHistory	Populated ApiEngBillingRateHistory object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Newly created BillingRatesEffectiveDatesId will be assigned to property BillingRatesEffectiveDatesId.

RequestOnly fields in all history rates under same billing rate will be updated with the value assigned to current RequestOnly property.

Ensure all mandatory properties are set.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngBillingRateHistory.EngBillingRateHistoryWse
Dim myHistoryRate As New webEngBillingRateHistory.ApiEngBillingRateHistory
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
With myHistoryRate
   .CPConnection = myCon
   .Active = True
   .BillingRateId = "{CC0AA774-969C-43B8-A64E-1A9A91B50F94}"
   .EffectiveDate = CDate("10/15/2008")
   .Comments = "New effective date"
   .DiscountPercentage = 30
```

```
    .
    .
End With
Dim oRet As webEngBillingRateHistory.WSInt32 = myProxy.Add("", myHistoryRate)
```

Related information

"EngBillingRateHistory" on page 861

EngBillingRateHistory: Exists

```
Public Function Exists(ByVal sId As String) As WSBoolean
```

Purpose

Check whether this billing rate effective date exists or not

Parameters

Parameter	Description
sId	The ID of the billing rates effective date (BillingRatesEffectiveDatesId).

Returns

Type	Description
WSBoolean	True if the billing rate effective date exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngBillingRateHistory.EngBillingRateHistoryWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngBillingRateHistory.WSBoolean = myProxy.Exists("{9D19662B-
3FEF-4F3B-9E8A-24B40E8ABEEB}")
```

Related information

"EngBillingRateHistory" on page 861

EngBillingRateHistory: GetByBillingRateId

```
Public Function GetByBillingRateId(ByVal sBillingRateId As String) As
WSDataSet
```

Purpose

Retrieve all billing rate effective date (billing rate history) records for a billing rate

Parameters

Parameter	Description
sBillingRateId	The ID of the billing rate

Returns

Type	Description
WSDataSet	WSDataset (CustomerId, EngagementId, BillingRateId, BillingRatesEffectiveDatesId, BillingRate, BillingCurrency, BillingOfficeRateId, CostRate, CostCurrency, OTCostRate, DiscountPercentage, EffectiveDate, CommentHistory, ResourceId, ResourceCostRate, ResourceCostCurrency, ResourceOTCostRate, RoleCostRate, RoleCostCurrency, RoleOTCostRate, Active, RequestOnly) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngBillingRateHistory.EngBillingRateHistoryWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
```

```
Dim oRet As webEngBillingRateHistory.WSDataSet = myProxy. GetByBillingRateId("
{CC0AA774-969C-43B8-A64E-1A9A91B50F94}")
```

Related information

"EngBillingRateHistory" on page 861

EngBillingRateHistory: GetById

```
Public Function GetById(ByVal sId As String) As WSEngBillingRateHistory
```

Purpose

Fills the ApiEngBillingRateHistory object with billing rate effective date (history rate) record of the specified sId passed in the parameter.

Parameters

Parameter	Description
sId	The ID of the billing rates effective date (BillingRatesEffectiveDatesId).

Returns

Type	Description
WSEngBillingRateHistory	Returns an ApiEngBillingRateHistory object. `Access the object through the .Value property of WSEngBillingRateHistory.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngBillingRateHistory.EngBillingRateHistoryWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngBillingRateHistory.WSEngBillingRateHistory = myProxy.GetById
("{9D19662B-3FEF-4F3B-9E8A-24B40E8ABEEB}")
```

Related information

"EngBillingRateHistory" on page 861

EngBillingRateHistory: Update

```
Public Function Update(ByVal oEngBillingRateHistory As
ApiEngBillingRateHistory) As WSInt32
```

Purpose

Update a billing rate's effective date record.

Parameters

Parameter	Description
oEngBillingRateHistory	Populated ApiEngBillingRateHistory object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Updates the three fields: Active, RequestOnly, Comments

If the value of RequestOnly is changed, then RequestOnly fields in all history rates under same billing rate will be updated as well.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngBillingRateHistory.EngBillingRateHistoryWse
Dim myHistoryRate As New webEngBillingRateHistory.ApiEngBillingRateHistory
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
```

```
With myHistoryRate
   .BillingRatesEffectiveDatesId = "{2E629DCC-D25B-4A37-9A2D-C1793E624C72}"
   .Active = False
   .Comments = "Not Active"
   .RequestOnly = True
End With
Dim oRet As webEngBillingRateHistory.WSInt32 = myProxy.Update(myHistoryRate)
```

Related information

"EngBillingRateHistory" on page 861

EngFixedFee

The EngFixedFee object allows users to retrieve, add, update and delete fixed fee set at the contract level.

Namespace

`http://changepoint.com/changepoint/CPWebService/EngFixedFee`

URL

http://webserver/CPWebService/Objects/CPEngagement/EngFixedFee.asmx

Methods

Properties

For more information, see the "ApiEngFixedFee" section on page 163.

Related information

"Engagement" on page 818

EngFixedFee: Add

```
Public Function Add(ByRef sId As String, ByVal oEngFixedFee As ApiEngFixedFee)
As WSInt32
```

Purpose

Insert a new contract fixed fee schedule record for a contract.

Parameters

Parameter	Description
sId	Placeholder of the returned value of new FixedFeeId.
oEngFixedFee	Populated ApiEngFixedFee object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Ensure all mandatory properties are set.

The value of the property Billed should be always False. If a value is not assigned, it takes a default value False.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngFixedFee.EngFixedFeeWse
Dim myEngFixedFee As New webEngFixedFee.ApiEngFixedFee
Dim sId As String = String.Empty
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
With myEngFixedFee
```

```
    .Engagement = New webEngFixedFee.Identity()
    .Engagement.Id = "{2E1D37B7-810F-43D6-ABAC-E624AB6F28EC}"
    .Billed = False
    .BillingDate = CDate("10/31/2008")
    .BillingAmount = 50000
    .Deliverable = "Design/Coding"
    .UserDefinedFixedFeeId = "FF000001"
    .WorkCodeCategory = New webEngFixedFee.Identity()
    .WorkCodeCategory.Id = "{AA9C83C1-6E1E-4974-9DEC-7C554CC429D2}"
    .WorkCode = New webEngFixedFee.Identity()
    .WorkCode.Id = "{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}"
    .WorkLocationGroup = New webEngFixedFee.Identity()
    .WorkLocationGroup.Id = "{333A206C-513E-4CAE-841A-EA73A82E8E81}"
    .WorkLocation = New webEngFixedFee.Identity()
    .WorkLocation.Id = "{0409A21B-EFB9-42C4-ADB3-617279F524EF}"
End With
Dim oRet As webEngFixedFee.WSInt32 = myProxy.Add(sId, myEngFixedFee)
```

Related information

"EngFixedFee" on page 867

EngFixedFee: Delete

```
Public Function Delete(ByVal sId As String) As WSInt32
```

Purpose

Delete a fixed fee schedule for an contract

Parameters

Parameter	Description
sId	ID of the contract fixed fee schedule record.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngFixedFee.EngFixedFeeWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngFixedFee.WSInt32 = myProxy.Delete("{8A842477-5C28-4868-A520-
846567A75C3A}")
```

Related information

"EngFixedFee" on page 867

EngFixedFee: Exists

```
Public Function Exists(ByVal sId As String) As WSBoolean
```

Purpose

Check whether this fixed fee schedule exists or not

Parameters

Parameter	Description
sId	ID of the contract fixed fee schedule record.

Returns

Type	Description
WSBoolean	True if the fixed fee schedule exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngFixedFee.EngFixedFeeWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngFixedFee.WSBoolean = myProxy.Exists("{8A842477-5C28-4868-
A520-846567A75C3A}")
```

Related information

"EngFixedFee" on page 867

EngFixedFee: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As WSDataSet
```

Purpose

Retrieve all fixed fee schedule records for a contract

Parameters

Parameter	Description
sEngagementId	ID of the engagement

Returns

Type	Description
WSDataSet	WSDataset (CustomerId, EngagementId, FixedFeeId, BillingDate, BillingAmount, Deliverable, Billed, CreatedOn, CreatedBy, UpdatedOn, UpdatedBy, RevenueMethod, PPC, RevTent, RevRec, DrGL, CrGL, RevAdj, RRUpdate, UserDefinedfixedFeeID, Prepaid, ParentFixedFeeId, DrGLDescription, CrGLDescription, RevenueMethodDescription, WorkLocationGroupID, WorkLocationGroup, WorkLocationID, WorkLocation, WorkCodeCategoryId, WorkCodeCategory, WorkCodeId, WorkCode, ParentDeliverable, ParentBillingDate, HasScheduleItems) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngFixedFee.EngFixedFeeWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngFixedFee.WSDataSet = myProxy.GetByEngagementId("{C5CBB921-
5FF0-48FA-9A14-6F4072641FA0}")
```

Related information

"EngFixedFee" on page 867

EngFixedFee: GetById

```
Public Function GetById(ByVal sId As String) As WSEngFixedFee
```

Purpose

Fills the ApiEngFixedFee object with the contract fixed fee schedule of the specified sId passed in the parameter.

Parameters

Parameter	Description
sId	ID of the contract fixed fee schedule record.

Returns

Type	Description
WSEngFixedFee	Returns an ApiEngFixedFee object. `Access the object through the .Value property of WSEngFixedFee.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngFixedFee.EngFixedFeeWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngFixedFee.WSEngFixedFee = myProxy.GetById("{8A842477-5C28-
4868-A520-846567A75C3A}")
```

Related information

"EngFixedFee" on page 867

EngFixedFee: GetList

```
Public Function GetList(ByVal iRetRows As Int16) As WSDataSet
```

Purpose

Retrieve contract fixed fee schedule records.

Parameters

Parameter	Description
iRetRows	The number of records to be returned. "-1" = return all.

Returns

Type	Description
WSDataSet	WSDataset (EngagementId, EngagementName, CustomerId, CustomerName, FixedFeeId, BillingDate, BillingAmount, Deliverable, Billed) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngFixedFee.EngFixedFeeWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngFixedFee.WSDataSet = myProxy.GetList(-1)
```

Related information

"EngFixedFee" on page 867

EngFixedFee: Update

```
Public Function Update(ByVal oEngFixedFee As ApiEngFixedFee) As WSInt32
```

Purpose

Update a contract fixed fee schedule record.

Parameters

Parameter	Description
oEngFixedFee	Populated ApiEngFixedFee object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Use GetById to populate ApiEngFixedFee with the existing data before the Update method is called.

You can add a fixed fee only if the billing amount of the fixed fee is equal to the total billing amount of the fixed fee items.

You can change the fixed fee billing amount and fixed fee items billing amounts using the Save method of the ApiEngFixedFeeItem object.

The value of the property Billed should be always False. If a value is not assigned, it takes a default value False.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngFixedFee.EngFixedFeeWse
Dim myEngFixedFee As New webEngFixedFee.ApiEngFixedFee
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngFixedFee.WSEngFixedFee = myProxy.GetById("{8A842477-5C28-
4868-A520-846567A75C3A}")
If Not oRet.WSException.HaveErrors Then
   myEngFixedFee = oRet.value
End If
With myEngFixedFee
   .BillingDate = CDate("10/31/2008")
   .BillingAmount = 50000
   .Deliverable = "Design/Coding"
   .UserDefinedFixedFeeId = "FF000001"
   .WorkCodeCategory = New webEngFixedFee.Identity()
   .WorkCodeCategory.Id = "{AA9C83C1-6E1E-4974-9DEC-7C554CC429D2}"
   .WorkCode = New webEngFixedFee.Identity()
   .WorkCode.Id = "{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}"
   .WorkLocationGroup = New webEngFixedFee.Identity()
   .WorkLocationGroup.Id = "{333A206C-513E-4CAE-841A-EA73A82E8E81}"
   .WorkLocation = New webEngFixedFee.Identity()
   .WorkLocation.Id = "{0409A21B-EFB9-42C4-ADB3-617279F524EF}"
   .
   .
End With
Dim oRet As webEngFixedFee.WSInt32 = myProxy.Update(myEngFixedFee)
```

Related information

"EngFixedFee" on page 867

EngFixedFeeItem

The EngFixedFeeItem object allows users to manually create a fixed fee schedule item.

Namespace

`http://changepoint.com/changepoint/CPWebService/EngFixedFeeItem`

URL

http://webserver/CPWebService/Objects/CPEngagement/EngFixedFeeItem.asmx

Methods

Properties

For more information, see the "ApiEngFixedFeeItem" section on page 175.

Related information

"Engagement" on page 818

EngFixedFeeItem: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As WSDataSet
```

Purpose

Returns a list of fixed fee schedule items for the contract.

Parameters

Parameter	Description
sEngagementId	Engagement ID

Returns

Type	Description
WSDataSet	WSDataset (FixedFeeId, EngagementId, BillingDate, BillingAmount, Deliverable, Billed, CreatedOn, CreatedBy, UpdatedOn, UpdatedBy, iRevRec, UserDefinedfixedFeeID, Prepaid, ParentFixedFeeId, WorkLocationGroupID, WorkLocationGroup, WorkLocationID, WorkLocation, WorkCodeCategoryId, WorkCodeCategory, WorkCodeId, WorkCode, ParentBillingAmount) Access the dataset through the .Value property of WSDataSet.

Remarks

None.

Example

None.

Related information

"EngFixedFeeItem" on page 876

EngFixedFeeItem: GetById

```
Public Function GetById(ByVal sId As String) As WSEngFixedFeeItem
```

Purpose

Fills the ApiEngFixedFeeItem object with fixed fee schedule item information of the specified fixedfeeid passed in the parameter sId.

Parameters

Parameter	Description
sId	Fixed Fee ID

Returns

Type	Description
WSEngFixedFeeItem	Returns an ApiEngFixedFeeItem object. `Access the object through the .Value property of WSEngFixedFeeItem.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngFixedFeeItem.EngFixedFeeItemWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngFixedFeeItem.WSEngFixedFeeItem = myProxy.GetById("{F66A6B2B-
0BD4-4B95-A7D1-D901ED3B67AF}")
```

Related information

"EngFixedFeeItem" on page 876

EngFixedFeeItem: GetByParentFixedFeeId

```
Public Function GetByParentFixedFeeId(ByVal sFixedFeeId As String) As WSString
```

Purpose

Retrieve fixed fee schedule items for the specified fixed fee

Parameters

Parameter	Description
sFixedFeeId	Fixed fee ID

Returns

Type	Description
WSString	XML string of contract fixed fee schedule items, or empty string if there are no fixed fee schedule items for the specified fixed fee. Access the returned string through the .Value property of WSString

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngFixedFeeItem.EngFixedFeeItemWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngFixedFeeItem.WSString = myProxy.GetByParentFixedFeeId("
{269206D7-2487-4F2A-868C-4FF42C1533AC}")
```

Related information

"EngFixedFeeItem: GetXMLStructure" on page 879

EngFixedFeeItem: GetXMLStructure

```
Public Function GetXMLStructure() As WSString
```

Purpose

Retrieve the XML structure of the fixed fee item

Parameters

None

Returns

Type	Description
WSString	XML structure of the fixed fee item. Access the returned string through the .Value property of WSString

Remarks

None

Example

```
Dim myProxy As New webEngFixedFeeItem.EngFixedFeeItemWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngFixedFeeItem.WSString = myProxy.GetXMLStructure()
```

Related information

"EngFixedFeeItem" on page 876

"ApiEngFixedFeeItem XML" on page 178

EngFixedFeeItem: Save

```
Public Function Save(ByVal sXmlFixedFeeItems As String) As WSInt32
```

Purpose

Update the fixed fee item information of an engagement

Parameters

Parameter	Description
sXmlFixedFeeItems	Fixed fee Items in XML format string.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

- It is recommended to use the GetByParentFixedFeeId or GetXMLStructure methods to get the correct XML format of parameter sXmlFixedFeeItems.

- Includes UPDATE, DELETE, and ADD functionalities.
  - To update a fixedfeeitem, use the GUID of the item to be updated as the value of the <fixedfeeid> tag.
  - To delete a fixedfeeitem, set its <deleted> tag to TRUE.
  - To add a fixedfeeitem, set its <fixedfeeid> tag to empty.
- The <parentfixedfeeid> tag is mandatory and should contain a GUID value which all the fixed fee items are associated with. <parentfixedfeeid> has one or more child <item> elements, each of which represents one fixed fee item.
- The parent billing amount is calculated by the system as the sum of the <billingamount> of the passed-in <item> elements, plus the billingamount of any existing fixed fee items in the database. The <parentbillingamount> node is ignored.
- The value of the <billed> tag should be set to 0 or empty
- Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngFixedFeeItem.EngFixedFeeItemWse
Dim sXmlFixedFeeItems As String
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)

'sXmlFixedFeeItems contains 3 items for update,delete and add
sXmlFixedFeeItems =
"<root><engfixedfeeitem>
<parentfixedfeeid>79C54E97-77C1-47E6-8CDD-CA185827743D</parentfixedfeeid>
<parentbillingamount>40000</parentbillingamount>
<item>
<fixedfeeid>{F66A6B2B-0BD4-4B95-A7D1-D901ED3B67AF}</fixedfeeid>
<billingdate>10/31/2008</billingdate>
<billingamount>20000</billingamount>
<deliverable>Coding phase 1</deliverable>
<billed></billed>
<userdefinedfixedfeeid></userdefinedfixedfeeid>
<worklocationgroupid>{333A206C-513E-4CAE-841A-
EA73A82E8E81}</worklocationgroupid>
<worklocationid>{0409A21B-EFB9-42C4-ADB3-617279F524EF}</worklocationid>
<workcodecategoryid>{AA9C83C1-6E1E-4974-9DEC-
7C554CC429D2}</workcodecategoryid>
<workcodeid>{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}</workcodeid>
```

```
<prepaid>0</prepaid>
<deleted>0</deleted>
</item>
<item><fixedfeeid>{5A54AC3C-E5EE-4FD0-80B7-458B7771829D}</fixedfeeid>
<billingdate>09/31/2008</billingdate>
<billingamount>20000</billingamount>
<deliverable>Design phase</deliverable>
<billed>0</billed>
<userdefinedfixedfeeid></userdefinedfixedfeeid>
<worklocationgroupid>{333A206C-513E-4CAE-841A-
EA73A82E8E81}</worklocationgroupid>
<worklocationid>{0409A21B-EFB9-42C4-ADB3-617279F524EF}</worklocationid>
<workcodecategoryid>{AA9C83C1-6E1E-4974-9DEC-
7C554CC429D2}</workcodecategoryid>
<workcodeid>{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}</workcodeid>
<prepaid>0</prepaid>
<deleted>1</deleted>
</item>
<item>
<fixedfeeid></fixedfeeid>
<billingdate>11/31/2008</billingdate>
<billingamount>20000</billingamount>
<deliverable> Coding phase 2</deliverable>
<billed></billed>
<userdefinedfixedfeeid></userdefinedfixedfeeid>
<worklocationgroupid>333A206C-513E-4CAE-841A-
EA73A82E8E81</worklocationgroupid>
<worklocationid>0409A21B-EFB9-42C4-ADB3-617279F524EF</worklocationid>
<workcodecategoryid>AA9C83C1-6E1E-4974-9DEC-7C554CC429D2</workcodecategoryid>
<workcodeid>{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}</workcodeid>
<prepaid>0</prepaid>
<deleted>0</deleted>
</item>
</engfixedfeeitem></root>"

Dim oRet As webEngFixedFeeItem.WSInt32 = myProxy.Save(sXmlFixedFeeItems)
```

Related information

"EngFixedFeeItem" on page 876

"EngFixedFeeItem: GetXMLStructure" on page 879

EngFixedFeeItemSplitBillOverride

Provides a collection object for split billing override information for contract fixed fee items.

Namespace

`http://changepoint.com/changepoint/CPWebService/EngFixedFeeItemSplitBillOverride`

URL

http://webserver/CPWebService/Objects/CPEngagement/EngFixedFeeItemSplitBillOverride.asmx

Methods

None.

Properties

For more information, see the "ApiEngFixedFeeItemSplitBillOverride" section on page 184.

EngFixedFeeSplitBillOverride

Provides a collection object for split billing override information for contract fixed fees.

Namespace

`http://changepoint.com/changepoint/CPWebService/EngFixedFeeSplitBillOverride`

URL

http://webserver/CPWebService/Objects/CPEngagement/EngFixedFeeSplitBillOverride.asmx

Methods

None.

Properties

For more information, see the "ApiEngFixedFeeSplitBillOverride" section on page 173.

EngProduct

The EngProduct object allows users to retrieve, update and delete product set at the contract level.

Namespace

`http://changepoint.com/changepoint/CPWebService/EngProduct`

URL

http://webserver/CPWebService/Objects/CPEngagement/EngProduct.asmx

Methods

Properties

For more information, see the "ApiEngProduct" section on page 186.

Related information

"Engagement" on page 818

EngProduct: Add

```
Public Function Add(ByRef sId As String, ByVal oEngProduct As ApiEngProduct)
As WSInt32
```

Purpose

Insert a new contract product record for a contract.

Parameters

Parameter	Description
sId	Placeholder of the returned value of new EngProductId.
oEngProduct	Populated ApiEngProduct object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

The value of the property Billed should be always False. If a value is not assigned, it takes the default value False.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProduct.EngProductWse
Dim myEngProduct As New webEngProduct.ApiEngProduct
Dim sId As String = String.Empty
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
With myEngProduct
   .Engagement = New webEngProduct.Identity()
   .Engagement.Id = "{2E1D37B7-810F-43D6-ABAC-E624AB6F28EC}"
   .FixedCost = True
   .FixedCostOverride = True
   .Product = New webEngProduct.Identity()
   .Product.Id = "{705E4EBD-C568-496F-832B-4C7168D5D41F}"
   .StandardCost = 16022.0
   .Description = "Added from API"
   .NegotiatedPrice = 9999.0
   .ProductNegotiatedCost = 599.0
   .ProductDate = "12/05/2007"
   .Quantity = 1000
   .Resource = New webEngProduct.Identity()
   .Resource.Id = "{6EE89511-89D3-4832-AB48-1F7C82C0477E}"
   .WorkLocationGroup = New webEngProduct.Identity()
   .WorkLocationGroup.Id = "{333A206C-513E-4CAE-841A-EA73A82E8E81}"
   .WorkLocation = New webEngProduct.Identity()
   .WorkLocation.Id = "{0409A21B-EFB9-42C4-ADB3-617279F524EF}"
End With
Dim oRet As webEngProduct.WSInt32 = myProxy.Add(sId, myEngProduct)
```

Related information

"EngProduct" on page 883

EngProduct: Delete

```
Public Function Delete(ByVal sId As String) As WSInt32
```

Purpose

Delete a contract product for a contract

Parameters

Parameter	Description
sId	ID of the contract product record.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProduct.EngProductWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProduct.WSInt32 = myProxy.Delete("{719FB77B-C7F6-4A1A-A4B1-
3A91F82B128E}")
```

Related information

"EngProduct" on page 883

EngProduct: Exists

```
Public Function Exists(ByVal sId As String) As WSBoolean
```

Purpose

Check whether this contract product exists or not.

Parameters

Parameter	Description
sId	ID of the contract product record.

Returns

Type	Description
WSBoolean	True if the contract product exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProduct.EngProductWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProduct.WSBoolean = myProxy.Exists("{719FB77B-C7F6-4A1A-
A4B1-3A91F82B128E}")
```

Related information

"EngProduct" on page 883

EngProduct: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As WSDataSet
```

Purpose

Retrieve all product records for a contract

Parameters

Parameter	Description
sEngagementId	ID of the engagement

Returns

Type	Description
WSDataSet	WSDataset (CustomerId, EngagementId, EngagementProductId, ProductId, Comments, ProductDate, ProductName, ProductUnitCost, ProductCurrency, Quantity, NegotiatedCost, ProductNegotiatedCost, ProjectId, ProjectName, WorkLocationGroupId, WorkLocationGroup, WorkLocationId, WorkLocation, ResourceId, .ResourceName, RevenueMethod, RevenueMethodDescription, ParentProductName, ParentDate, ParentDescription) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProduct.EngProductWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProduct.WSDataSet = myProxy.GetByEngagementId("{2E1D37B7-
810F-43D6-ABAC-E624AB6F28EC}")
```

Related information

"EngProduct" on page 883

EngProduct: GetById

```
Public Function GetById(ByVal sId As String) As WSEngProduct
```

Purpose

Fills the ApiEngProduct object with contract product information of the specified sId passed in the parameter.

Parameters

Parameter	Description
sId	The ID of the contract product record.

Returns

Type	Description
WSEngProduct	Returns an ApiEngProduct object. `Access the object through the .Value property of WSEngProduct.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProduct.EngProductWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProduct.WSEngProduct = myProxy.GetById("{719FB77B-C7F6-4A1A-
A4B1-3A91F82B128E}"
```

Related information

"EngProduct" on page 883

EngProduct: GetList

```
Public Function GetList(ByVal iRetRows As Int16) As WSDataSet
```

Purpose

Retrieve contract product records.

Parameters

Parameter	Description
iRetRows	The number of records to be returned. "-1" = return all.

Returns

Type	Description
WSDataSet	WSDataset (CustomerId, EngagementId, EngagementProductId, ProductId, Description, ProductDate, ProductName, ProductUnitCost, ProductCurrency, Quantity, NegotiatedPrice, ProductNegotiatedCost, ProjectId, ProjectName, ShipToWorkLocationGroupId, WorkLocationGroup, ShipToWorkLocationId, WorkLocation, ResourceId, resourcename) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProduct.EngProductWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProduct.WSDataSet = myProxy.GetList(-1)
```

Related information

"EngProduct" on page 883

EngProduct: GetProductPrice

```
Public Function GetProductPrice(ByVal sProductId As String, ByVal sCurrency As
String) As WSDataSet
```

Purpose

Retrieve product prices with specific currency for a product

Parameters

Parameter	Description
sProductId	Product ID
sCurrency	Currency string. This is a three letter currency code which exists in the CurrencyDescription table.

Returns

Type	Description
WSDataSet	WSDataset (Price, ProductFixedUnitCost) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProduct.EngProductWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProduct.WSDataSet = myProxy.GetProductPrice("{59FB7D7D-6622-
4317-A856-D44CBC859237}", "USD")
```

Related information

"EngProduct" on page 883

EngProduct: GetProducts

```
Public Function GetProducts() As WSDataSet
```

Purpose

Retrieve contract enabled products

Parameters

None

Returns

Type	Description
WSDataSet	WSDataset (ProductCategoryId, ProductId, ProductName, ProductDescription, ProductStatus, ProductNumber, ProductOpenDate, ProductCloseDate, ProductManager, ProductUnitCost, ProductFixedUnitCost, ProductCurrency) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProduct.EngProductWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProduct.WSDataSet = myProxy.GetProducts()
```

Related information

"EngProduct" on page 883

EngProduct: GetProjects

```
Public Function GetProjects(ByVal sEngagementId As String) As WSDataSet
```

Purpose

Retrieve projects associated with an Engagement

Parameters

Parameter	Description
sEngagementId	Engagement ID

Returns

Type	Description
WSDataSet	WSDataset (ProjectId, Name, AlternateName) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProduct.EngProductWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProduct.WSDataSet = myProxy.GetProjects("{2E1D37B7-810F-
43D6-ABAC-E624AB6F28EC}")
```

Related information

"EngProduct" on page 883

EngProduct: Update

```
Public Function Update(ByVal oEngProduct As ApiEngProduct) As WSInt32
```

Purpose

Update a contract product record.

Parameters

Parameter	Description
oEngProduct	Populated ApiEngProduct object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Use GetById to populate ApiEngProduct with the existing data before the Update method is called.

The value of the Billed property should be always False. The default value is False.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and the logs.

Example

```
Dim myProxy As New webEngProduct.EngProductWse
Dim myEngProduct As New webEngProduct.ApiEngProduct
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProduct.WSEngProduct = myProxy.GetById("{719FB77B-C7F6-4A1A-
A4B1-3A91F82B128E}")
If Not oRet.WSException.HaveErrors Then
myEngProduct = oRet.value
End If
With myEngProduct
   .Description = "Updated from API"
   .NegotiatedPrice = 9999.0
   .ProductNegotiatedCost = 599.0
   .Quantity = 1500
   .
   .
End With
Dim oRet As webEngProduct.WSInt32 = myProxy.Update(myEngProduct)
```

Related information

"EngProduct" on page 883

EngProductSplitBillOverride

Provides a collection object for split billing override information for contract products.

Namespace

`http://changepoint.com/changepoint/CPWebService/EngProductSplitBillOverride`

URL

http://webserver/CPWebService/Objects/CPEngagement/EngProductSplitBillOverride.asmx

Methods

None.

Properties

For more information, see the "ApiEngProductSplitBillOverride" section on page 199.

EngProjectedResource

The EngProjectedResource object allows users to retrieve, update and delete projected resources set at the contract level.

Namespace

`http://changepoint.com/changepoint/CPWebService/EngProjectedResource`

URL

http://webserver/CPWebService/Objects/CPEngagement/EngProjectedResource.asmx

Methods

Properties

For more information, see the "ApiEngProjectedResource" section on page 200.

Related information

"Engagement" on page 818

EngProjectedResource: Add

```
Public Function Add(ByRef sId As String, ByVal oEngProjectedResource As
ApiEngProjectedResource) As WSInt32
```

Purpose

Insert a new projected resource record for a contract.

Parameters

Parameter	Description
sId	Placeholder of the returned value of new ProjectedResourceId.
oEngProjectedResource	Populated ApiEngProjectedResource object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Ensure all mandatory properties are set.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProjectedResource.EngProjectedResourceWse
Dim myEngProjRes As New webEngProjectedResource.ApiEngProjectedResource
Dim sId As String = String.Empty
'set the SOAP header with UsernameToken
```

```
UserToken.SetToken(myProxy, mUserName, mPassword)
With myEngProjRes
   .BillingOffice = New webEngProjectedResource.Identity()
   .BillingOffice.Id = "{DB587BF6-3665-11D4-8E13-0050DA7BA6B1}"
   .BillingRole = New webEngProjectedResource.Identity()
   .BillingRole.Id = "{6DE15305-3726-11D4-8E15-0050DA7BA6B1}"
   .Comments = "Added from COM API"
   .Engagement = New webEngProjectedResource.Identity()
   .Engagement.Id = "{1EB38A5E-2E62-4C06-AC8D-0A621B1C8ABA}"
   .EstimatedHours = 120
   .FinishDate = CDate("12/31/2008")
   .Resource = New webEngProjectedResource.Identity()
   .Resource.Id = "{BFC84F4D-365D-11D4-8E13-0050DA7BA6B1}"
   .StartDate = CDate("10/20/2008")
   .TaskCreated = True
   .Task = New webEngProjectedResource.Identity()
   .Task.Id = "{C595A18C-06B4-432C-A215-6B532D3465DA}"
   .SoftBooked = True
End With
Dim oRet As webEngProjectedResource.WSInt32 = myProxy.Add(sId, myEngProjRes)
```

Related information

"EngProjectedResource" on page 895

EngProjectedResource: Delete

```
Public Function Delete(ByVal sId As String) As WSInt32
```

Purpose

Delete a projected resource for a contract

Parameters

Parameter	Description
sId	The ID of the contract projected resource record.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProjectedResource.EngProjectedResourceWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProjectedResource.WSInt32 = myProxy.Delete("{4BAB1168-6D30-
4897-9E7E-CFF52DE50544}")
```

Related information

"EngProjectedResource" on page 895

EngProjectedResource: Exists

```
Public Function Exists(ByVal sId As String) As WSBoolean
```

Purpose

Check whether this contract projected resource exists or not

Parameters

Parameter	Description
sId	ID of the contract projected resource record.

Returns

Type	Description
WSBoolean	True if the contract projected resource exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProjectedResource.EngProjectedResourceWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProjectedResource.WSBoolean = myProxy.Exists("{4BAB1168-
6D30-4897-9E7E-CFF52DE50544}")
```

Related information

"EngProjectedResource" on page 895

EngProjectedResource: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As WSDataSet
```

Purpose

Retrieve all projected resource records for a contract

Parameters

Parameter	Description
sEngagementId	ID of the engagement

Returns

Type	Description
WSDataSet	WSDataset (ProjectedResourceId, CustomerId, EngagementId, BillingOfficeId, BillingRoleId, FunctionId, RequestId, ResourceId, TaskCreated, TaskId, EstimatedHours, StartDate, FinishDate, Comments, SoftBooked, CreatedOn, CreatedBy, UpdatedOn, UpdatedBy, ResourceName, AlternateName, BillingRoleDescription, FunctionDescription, CurrentStatus, RequestNumber, InitiatorId, BillingOfficeName) `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProjectedResource.EngProjectedResourceWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProjectedResource.WSDataSet = myProxy.GetByEngagementId("
{06237CF4-C7D4-4E19-A549-55039494ED8F}")
```

Related information

"EngProjectedResource" on page 895

EngProjectedResource: GetById

```
Public Function GetById(ByVal sId As String) As WSEngProjectedResource
```

Purpose

Fills the ApiEngProjectedResource object with contract projected resource information of the specified sId passed in the parameter.

Parameters

Parameter	Description
sId	The ID of the contract projected resource record.

Returns

Type	Description
WSEngRequestProcessingRule	Returns an ApiEngRequestProcessingRule object. `Access the object through the .Value property of WSEngRequestProcessingRule.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProjectedResource.EngProjectedResourceWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProjectedResource.WSEngProjectedResource = myProxy.GetById("
{4BAB1168-6D30-4897-9E7E-CFF52DE50544}")
```

Related information

"EngProjectedResource" on page 895

EngProjectedResource: GetList

```
Public Function GetList(ByVal iRetRows As Int16) As WSDataSet
```

Purpose

Retrieve contract projected resource records.

Parameters

Parameter	Description
iRetRows	The number of records to be returned. "-1" = return all.

Returns

Type	Description
WSDataSet	WSDataset (EngagementId, CustomerId, ProjectedResourceId, BillingOfficeId, BillingRoleId, ResourceId) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProjectedResource.EngProjectedResourceWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProjectedResource.WSDataSet = myProxy.GetList(-1)
```

Related information

"EngProjectedResource" on page 895

EngProjectedResource: Update

```
Public Function Update(ByVal oEngProjectedResource As ApiEngProjectedResource)
As WSInt32
```

Purpose

Update a contract projected resource record.

Parameters

Parameter	Description
oEngProjectedResource	Populated ApiEngProjectedResource object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

It's recommended to use GetById to populate ApiEngProjectedResource with the existing data before Update method is called.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngProjectedResource.EngProjectedResourceWse
Dim myEngProjRes As New webEngProjectedResource.ApiEngProjectedResource
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngProjectedResource.WSEngProjectedResource = myProxy.GetById("
{4BAB1168-6D30-4897-9E7E-CFF52DE50544}")
If Not oRet.WSException.HaveErrors Then
   myEngProjRes = oRet.value
End If
With myEngProjRes
   .Comments = "Updated from COM API"
   .EstimatedHours = 160
   .FinishDate = CDate("01/31/2009")
   .Resource = New webEngProjectedResource.Identity()
   .Resource.Id = "{BFC84F4D-365D-11D4-8E13-0050DA7BA6B1}"
   .StartDate = CDate("11/01/2008")
   .
   .
End With
```

```
Dim oRet As webEngProjectedResource.WSInt32 = myProxy.Update(myEngProjRes)
```

Related information

"EngProjectedResource" on page 895

EngRequestProcessingRule

The EngRequestProcessingRule object allows users to retrieve, delete and update request billing types and billing rules for contracts.

Namespace

```
http://changepoint.com/changepoint/CPWebService/EngRequestProcessingRule
```

URL

```
http://webserver/CPWebService/Objects/CPEngagement/EngRequestProcessingRule.as
mx
```

Methods

Properties

For more information, see the "ApiEngRequestProcessingRule" section on page 209.

Related information

"Engagement" on page 818

EngRequestProcessingRule: Add

```
Public Function Add(ByRef sId As String, ByVal oEngRPRule As
ApiEngRequestProcessingRule) As WSInt32
```

Purpose

Insert a new contract request processing rule record for a contract.

Parameters

Parameter	Description
sId	Placeholder of the returned value of new EngRequestBillingRuleId.
oEngRPRule	Populated ApiEngRequestProcessingRule object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Ensure all mandatory properties are set.

When there is no existing request processing rule, error message -10 is written into the log file. This message can be ignored

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestProcessingRule.EngRequestProcessingRuleWse
Dim myEngRBRule As New webEngRequestProcessingRule.ApiEngRequestProcessingRule
Dim sId As String = String.Empty
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
With myEngRBRule
```

```
    .AvailtoGuest = True
    .AvailToEnterprise = True
    .Engagement = New webEngRequestProcessingRule.Identity()
    .Engagement.Id = "{70852075-3A41-4406-955E-B96C777AB431}"
    .RequestBillingType = 0
    .RequestType = "CR"
    .SLARequiredToEnterprise = True
    .TrackSLA = True
End With
Dim oRet As webEngRequestProcessingRule.WSInt32 = myProxy.Add(sId,
myEngRBRule)
```

Related information

"EngRequestProcessingRule" on page 904

EngRequestProcessingRule: Delete

```
Public Function Delete(ByVal sId As String) As WSInt32
```

Purpose

Delete a contract request processing rule record for a contract

Parameters

Parameter	Description
sId	ID of the contract request processing rule record.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestProcessingRule.EngRequestProcessingRuleWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestProcessingRule.WSInt32 = myProxy.Delete("{0633FFB7-
97E0-42F2-B817-915468319CA9}")
```

Related information

"EngRequestProcessingRule" on page 904

EngRequestProcessingRule: Exists

```
Public Function Exists(ByVal sId As String) As WSBoolean
```

Purpose

Check whether this contract request processing rule exists or not

Parameters

Parameter	Description
sId	ID of the contract request processing rule record.

Returns

Type	Description
WSBoolean	True if the contract request processing rule exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestProcessingRule.EngRequestProcessingRuleWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
```

```
Dim oRet As webEngRequestProcessingRule.WSBoolean = myProxy.Exists("{4BAB1168-
6D30-4897-9E7E-CFF52DE50544}")
```

Related information

"EngRequestProcessingRule" on page 904

EngRequestProcessingRule: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As WSDataSet
```

Purpose

Retrieve all request processing rule records for a contract

Parameters

Parameter	Description
sEngagementId	ID of the engagement

Returns

Type	Description
WSDataSet	WSDataset (EngRequestBillingRuleId, EngagementId, CustomerId, RequestBillingType, RequestBillingTypeDescription, RequestType, RequestTypeDescription, BillingRoleId, BillingRole.Name, FixedFeeId, FixedFee.Name, AvailToEnterprise, AvailtoGuest, SLARequiredToEnterprise, SLARequiredToGuest, TrackSLA, CreatedBy, CreatedOn, UpdatedBy, UpdatedOn) Access the dataset through the .Value property of WSDataSet.

Remarks

- RBTDescription is the RequestBillingTypeDescription.
- Code is the RequestType
- RTDescription is the RequestTypeDescription
- BillingRole is BillingRole.Name

- FixedFee is FixedFee.Name.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestProcessingRule.EngRequestProcessingRuleWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestProcessingRule.WSDataSet = myProxy. GetByEngagementId
("{70852075-3A41-4406-955E-B96C777AB431}")
```

Related information

"EngRequestProcessingRule" on page 904

EngRequestProcessingRule: GetById

```
Public Function GetById(ByVal sId As String) As WSEngRequestProcessingRule
```

Purpose

Fills the ApiEngRequestProcessingRule object with contract request processing rule information of the specified sId passed in the parameter.

Parameters

Parameter	Description
sId	The ID of the contract request processing rule record.

Returns

Type	Description
WSEngRequestProcessingRule	Returns an ApiEngRequestProcessingRule object. `Access the object through the .Value property of WSEngRequestProcessingRule.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is true, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestProcessingRule.EngRequestProcessingRuleWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestProcessingRule.WSEngRequestProcessingRule =
myProxy.GetById("{4BAB1168-6D30-4897-9E7E-CFF52DE50544}")
```

Related information

"EngRequestProcessingRule" on page 904

EngRequestProcessingRule: GetList

```
Public Function GetList(ByVal iRetRows As Int16) As WSDataSet
```

Purpose

Retrieve contract request processing rule records.

Parameters

Parameter	Description
iRetRows	The number of records to be returned. "-1" = return all.

Returns

Type	Description
WSDataSet	WSDataset (EngagementId, EngagementName, CustomerId, EngRequestBillingRuleId, RequestType, RequestTypeDescription, RequestBillingType, RequestBillingTypeDescription) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestProcessingRule.EngRequestProcessingRuleWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestProcessingRule.WSDataSet = myProxy.GetList(-1)
```

Related information

"EngRequestProcessingRule" on page 904

EngRequestProcessingRule: GetRequestTypes

```
Public Function GetRequestTypes(ByVal sRequestTypeCode As String) As WSDataSet
```

Purpose

Retrieve Request Types

Parameters

Parameter	Description
sRequestTypeCode	Request type code

Returns

Type	Description
WSDataSet	WSDataset (Code, Description, AvailToGuest) Access the dataset through the .Value property of WSDataSet.

Remarks

Returns all request types when sRequestTypeCode = "" .

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestProcessingRule.EngRequestProcessingRuleWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestProcessingRule.WSDataSet = myProxy.GetRequestTypes
("")
```

Related information

"EngRequestProcessingRule" on page 904

EngRequestProcessingRule: GetXmlByEngagementId

```
Public Function GetXmlByEngagementId(ByVal sEngagementId As String) As
WSString
```

Purpose

Retrieve request processing rules in XML format for the specified contract

Parameters

Parameter	Description
sEngagementId	Engagement ID

Returns

Type	Description
WSString	XML string of the request processing rules for the contract, or and empty string is there are no rules. Access the returned string through the .Value property of WSString

Remarks

If there are no contract request processing rules for the specified contract, empty string is returned.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestProcessingRule.EngRequestProcessingRuleWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestProcessingRule.WSString =
myProxy.GetXmlByEngagementId("{70852075-3A41-4406-955E-B96C777AB431}")
```

Related information

"EngRequestProcessingRule" on page 904

"EngRequestProcessingRule: GetXMLStructure" on page 913

EngRequestProcessingRule: GetXMLStructure

```
Public Function GetXMLStructure() As WSString
```

Purpose

Retrieve the XML structure of the contract request processing rule

Parameters

None

Returns

Type	Description
WSString	XML structure of the contract request processing rule Access the returned string through the .Value property of WSString

Remarks

For more information, see the "ApiEngRequestProcessingRule XML" section on page 212.

Example

```
Dim myProxy As New webEngRequestProcessingRule.EngRequestProcessingRuleWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestProcessingRule.WSString = myProxy.GetXMLStructure()
```

Related information

"EngRequestProcessingRule" on page 904

EngRequestProcessingRule: SaveByXml

```
Public Function SaveByXml(ByVal sEngagementId As String, ByVal sXmlRPRules As
String) As WSInt32
```

Purpose

Update the request processing rule information of an contract

Parameters

Parameter	Description
sEngagementId	Engagement ID
sXmlRPRules	Request processing rules in XML string.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Use the GetXmlByEngagementId or GetXMLStructure methods to get the correct XML format of parameter sXmlRPRules.

The tag <engrequestbillingrule /> represents one request processing rule; therefore, multiple entries can be expected. All these entries should belong to the contract specified as parameter sEngagementId.

Includes ADD, UPDATE and DELETE functionalities. When the value of the deleted tag is true, it indicates DELETE; when the value of the engrequestbillingruleid tag is empty, it indicates ADD, when the value of the engrequestbillingruleid tag is a GUID, it indicates UPDATE.

Example

```
Dim myProxy As New webEngRequestProcessingRule.EngRequestProcessingRuleWse
```

```
Dim sXmlRPRules As String
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)

'sXmlRPRules contains 3 Request Processing rules for update,delete and add
sXmlRPRules =
"<root>
<engrequestbillingrule>
<deleted>0</deleted>
<engrequestbillingruleid>{20A95341-AF00-4FFE-BC65-
11E5093B86E1}</engrequestbillingruleid>
<requesttype></requesttype>
<requestbillingtype>0</requestbillingtype>
<billingroleid>{00000000-0000-0000-0000-000000000000}</billingroleid>
<fixedfeeid>{00000000-0000-0000-0000-000000000000}</fixedfeeid>
<availtoguest>0</availtoguest>
<slarequiredtoguest>0</slarequiredtoguest>
<tracksla>0</tracksla><availtoenterprise>0</availtoenterprise>
<slarequiredtoenterprise>0</slarequiredtoenterprise>
</engrequestbillingrule>
<engrequestbillingrule>
<deleted>0</deleted>
<engrequestbillingruleid></engrequestbillingruleid>
<requesttype>BR</requesttype>
<requestbillingtype>0</requestbillingtype>
<billingroleid></billingroleid><fixedfeeid></fixedfeeid>
<availtoguest>1</availtoguest>
<slarequiredtoguest>0</slarequiredtoguest>
<tracksla>1</tracksla>
<availtoenterprise>1</availtoenterprise>
<slarequiredtoenterprise>1</slarequiredtoenterprise>
</engrequestbillingrule>
<engrequestbillingrule>
<deleted>0</deleted>
<engrequestbillingruleid>{F768D80C-4061-414B-AC85-
76B803A8CC01}</engrequestbillingruleid>
<requesttype>DR</requesttype>
<requestbillingtype>0</requestbillingtype>
<billingroleid>{00000000-0000-0000-0000-000000000000}</billingroleid>
<fixedfeeid>{00000000-0000-0000-0000-000000000000}</fixedfeeid>
<availtoguest>1</availtoguest>
<slarequiredtoguest>0</slarequiredtoguest>
<tracksla>1</tracksla>
<availtoenterprise>1</availtoenterprise>
<slarequiredtoenterprise>1</slarequiredtoenterprise>
</engrequestbillingrule>
<engrequestbillingrule>
```

```
<deleted>1</deleted>
<engrequestbillingruleid>{D127FF27-49B4-47CC-8018-
18554E1F1D0D}</engrequestbillingruleid>
<requesttype>DR</requesttype>
<requestbillingtype>2</requestbillingtype>
<billingroleid>{00000000-0000-0000-0000-000000000000}</billingroleid>
<fixedfeeid>{00000000-0000-0000-0000-000000000000}</fixedfeeid>
<availtoguest>1</availtoguest>
<slarequiredtoguest>0</slarequiredtoguest>
<tracksla>1</tracksla>
<availtoenterprise>1</availtoenterprise>
<slarequiredtoenterprise>1</slarequiredtoenterprise>
</engrequestbillingrule>
</root>"

Dim oRet As webEngRequestProcessingRule.WSInt32 = myProxy.SaveByXml("
{E90A4835-786F-4AFE-89ED-88A959608277}", sXmlRPRules)
```

Related information

"EngRequestProcessingRule" on page 904

"EngRequestProcessingRule: GetXmlByEngagementId" on page 912

"EngRequestProcessingRule: GetXMLStructure" on page 913

EngRequestProcessingRule: Update

```
Public Function Update(ByVal oEngRPRule As ApiEngRequestProcessingRule) As
WSInt32
```

Purpose

Update a contract request processing rule record.

Parameters

Parameter	Description
oEngRPRule	Populated ApiEngRequestProcessingRule object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Use GetById to populate ApiEngRequestProcessingRule with the existing data before Update method is called.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestProcessingRule.EngRequestProcessingRuleWse
Dim myEngRBRule As New webEngRequestProcessingRule.ApiEngRequestProcessingRule
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestProcessingRule.WSEngRequestProcessingRule =
myProxy.GetById("{4BAB1168-6D30-4897-9E7E-CFF52DE50544}")
If Not oRet.WSException.HaveErrors Then
myEngRBRule = oRet.value
End If
With myEngRBRule
.RequestType = "BR"
.RequestBillingType = 1
.
.
End With
Dim oRet As webEngRequestProcessingRule.WSInt32 = myProxy.Update(myEngRBRule)
```

Related information

"EngRequestProcessingRule" on page 904

"EngRequestProcessingRule: GetById" on page 909

EngRequestSLA

The EngRequestSLA object represents all request support information created for a contract.

Namespace

```
http://changepoint.com/changepoint/CPWebService/EngRequestSLA
```

URL

```
http://webserver/CPWebService/Objects/CPEngagement/EngRequestSLA.asmx
```

Methods

Properties

For more information, see the "ApiEngRequestSLA" section on page 223.

Related information

"Engagement" on page 818

EngRequestSLA: Add

```
Public Function Add(ByRef sId As String, ByVal oEngRequestSLA As
ApiEngRequestSLA) As WSInt32
```

Purpose

Insert a new contract request SLA record for a contract.

Parameters

Parameter	Description
sId	Placeholder of the returned ID of the contract SLA record.
oEngRequestSLA	Populated ApiEngRequestSLA object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Use numeric representation of the weekday for StartDay and EndDay. See Properties for detailed information.

Ensure all mandatory properties are set.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestSLA.EngRequestSLAWse
Dim myEngReqSLA As New webEngRequestSLA.ApiEngRequestSLA
Dim sId As String = String.Empty
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
With myEngReqSLA
   .By24x7 = False
   .EffectiveDate = CDate("10/27/2008")
   .EndDay = 2
   .EndTime = "18:00"
   .Engagement = New webEngRequestSLA.Identity()
   .Engagement.Id = "{A617568D-DFBE-444D-84D2-8AA3B94FECC7}"
   .EscalationHours = 4
   .ExpiryDate = CDate("10/27/2009")
   .Onsite = False
   .Priority = New webEngRequestSLA.Identity()
   .Priority.Id = "{7D72E3F4-E827-4F0E-8AED-798D31DD6162}"
   .Product = New webEngRequestSLA.Identity()
   .Product.Id = "{EF83EF72-384D-42AC-8317-32116C583082}"
   .RequestType = "BR"
   .ResolutionHours = 24
   .ResponseHours = 8
   .StartDay = 6
   .StartTime = "9:00"
   .TimeZone = "EST"
```

```
End With
Dim oRet As webEngRequestSLA.WSInt32 = myProxy.Add(sId, myEngReqSLA)
```

Related information

"EngRequestSLA" on page 917

EngRequestSLA: Delete

```
Public Function Delete(ByVal sId As String) As WSInt32
```

Purpose

Deletes a contract SLA for a contract

Parameters

Parameter	Description
sId	ID of the contract SLA record.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestSLA.EngRequestSLAWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestSLA.WSInt32 = myProxy.Delete("{C26D51B4-4C22-440A-
A21A-4C88FB445BEC}")
```

Related information

"EngRequestSLA" on page 917

EngRequestSLA: Exists

```
Public Function Exists(ByVal sId As String) As WSBoolean
```

Purpose

Check whether this contract request SLA exists or not

Parameters

Parameter	Description
sId	ID of the contract SLA record.

Returns

Type	Description
WSBoolean	True if the contract request SLA exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestSLA.EngRequestSLAWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestSLA.WSBoolean = myProxy.Exists("{C26D51B4-4C22-440A-
A21A-4C88FB445BEC}")
```

Related information

"EngRequestSLA" on page 917

EngRequestSLA: GetByEngagementId

```
Public Function GetByEngagementId(ByVal sEngagementId As String) As WSDataSet
```

Purpose

Retrieve all contract request SLA records for a contract

Parameters

Parameter	Description
sEngagementId	ID of the contract

Returns

Type	Description
WSDataSet	WSDataset (CustomerId, EngagementId, EngagementRequestSLAId, ProductId, ProductName, By24x7, CreatedBy, CreatedOn, EffectiveDate, EndDay, EndTime, EscalationHours, ExpiryDate, License, Onsite, Priority, PriorityName, RequestType, RTDescription, ResolutionHours, StartDay, StartTime, TimeZone, TimeZoneName, UpdatedBy, UpdatedOn) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestSLA.EngRequestSLAWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestSLA.WSDataSet = myProxy. GetByEngagementId("
{06237CF4-C7D4-4E19-A549-55039494ED8F}")
```

Related information

"EngRequestSLA" on page 917

EngRequestSLA: GetById

```
Public Function GetById(ByVal sId As String) As WSEngRequestSLA
```

Purpose

Fills the ApiEngRequestSLA object with contract request SLA information of the specified sId passed in the parameter.

Parameters

Parameter	Description
sId	Engagement SLA ID

Returns

Type	Description
WSEngRequestSLA	Returns an ApiEngRequestSLA object. `Access the object through the .Value property of WSEngRequestSLA.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestSLA.EngRequestSLAWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestSLA.WSEngRequestSLA = myProxy.GetById("{C26D51B4-
4C22-440A-A21A-4C88FB445BEC}")
```

Related information

"EngRequestSLA" on page 917

EngRequestSLA: GetList

```
Public Function GetList(ByVal iRetRows As Int16) As WSDataSet
```

Purpose

Retrieves contract request SLA records.

Parameters

Parameter	Description
iRetRows	The number of records to be returned. "-1" = return all.

Returns

Type	Description
WSDataSet	WSDataset (CustomerId, EngagementId, EngagementName, EngagementSLA, ProductId, ProductName) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestSLA.EngRequestSLAWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestSLA.WSDataSet = myProxy.GetList(-1)
```

Related information

"EngRequestSLA" on page 917

EngRequestSLA: GetProducts

```
Public Function GetProducts() As WSDataSet
```

Purpose

Retrieves product records

Parameters

None

Returns

Type	Description
WSDataSet	WSDataset (ProductCategoryId, ProductId, ProductName, ProductNumber, ProductOpenDate, ProductCloseDate, ProductUnitCost, ProductCurrency) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestSLA.EngRequestSLAWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestSLA.WSDataSet = myProxy.GetProducts()
```

Related information

"EngRequestSLA" on page 917

EngRequestSLA: Update

```
Public Function Update(ByVal oEngRequestSLA As ApiEngRequestSLA) As WSInt32
```

Purpose

Update a contract request SLA record.

Parameters

Parameter	Description
oEngRequestSLA	Populated ApiEngRequestSLA object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Use GetById to populate ApiEngRequestSLA with the existing data before Update method is called.

Use the numeric representation of the weekdays for StartDay and EndDay. See Properties for detailed information.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRequestSLA.EngRequestSLAWse
Dim myEngRequestSLA As New webEngRequestSLA.ApiEngRequestSLA
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRequestSLA.WSEngRequestSLA = myProxy.GetById("{C26D51B4-
4C22-440A-A21A-4C88FB445BEC}")
If Not oRet.WSException.HaveErrors Then
myEngRequestSLA = oRet.value
End If
With myEngReqSLA
   .EscalationHours = 24
   .ResolutionHours = 48
   .ResponseHours = 8
   .StartDay = 6
   .
   .
End With
Dim oRetInt32 As webEngRequestSLA.WSInt32 = myProxy.Update(myEngReqSLA)
```

Related information

"EngRequestSLA" on page 917

"EngRequestSLA: GetById" on page 923

EngRevRec

The EngRevRec object represents the revenue recognition information for a contract.

Namespace

`http://changepoint.com/changepoint/CPWebService/EngRevRec`

URL

`http://webserver/CPWebService/Objects/CPEngagement/EngRevRec.asmx`

Methods

Properties

For more information, see the "ApiEngRevRec" section on page 233.

Related information

"Engagement" on page 818

EngRevRec: GetById

```
Public Function GetById(ByVal sId As String) As WSEngRevRec
```

Purpose

Fills the ApiEngRevRec object with contract revenue recognition information of the specified EngagementId passed in the parameter sId.

Parameters

Parameter	Description
sId	Engagement ID

Returns

Type	Description
WSEngRevRec	Returns an ApiEngRevRec object. `Access the object through the .Value property of WSEngRevRec.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRevRec.EngRevRecWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRevRec.WSEngRevRec = myProxy.GetById("{06237CF4-C7D4-4E19-
A549-55039494ED8F}")
```

Related information

"EngRevRec" on page 927

EngRevRec: GetDeliverableAmount

```
Public Function GetDeliverableAmount(ByVal sEngagementId As String) As
WSDecimal
```

Purpose

Get a calculated deliverable amount for an Engagement

Parameters

Parameter	Description
sEngagementId	Engagement ID

Returns

Type	Description
WSDecimal	Returns the total of the deliverable (contract product and fixed fee schedule) amount of the contract. Access the amount through the .Value property of WSDecimal.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngRevRec.EngRevRecWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRevRec.WSDecimal = myProxy.GetDeliverableAmount("{06237CF4-
C7D4-4E19-A549-55039494ED8F}")
```

Related information

"EngRevRec" on page 927

EngRevRec: Update

```
Public Function Update(ByVal oEngRevRec As ApiEngRevRec) As WSInt32
```

Purpose

Update contract revenue recognition information.

Parameters

Parameter	Description
oEngRevRec	Populated ApiEngRevRec object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

It's recommended to use GetById to populate ApiRevRec with the existing data before Update method is called.

Example

```
Dim myProxy As New webEngRevRec.EngRevRecWse
Dim myEngRevRec As New webEngRevRec.ApiEngRevRec
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngRevRec.WSEngRevRec = myProxy.GetById("{06237CF4-C7D4-4E19-
A549-55039494ED8F}")

If Not oRet.WSException.HaveErrors Then
   myEngRevRec = oRet.value
End If
With myEngRevRec
   .RROn = True
   .RevRecMethod = 3
   .CRGLTime = "{4C7FB5EC-184D-4014-BB68-893E256911D8}"
   .
   .
End With
Dim oRetInt32 As webEngRevRec.WSInt32 = myProxy.Update(myEngRevRec)
```

Related information

"EngRevRec" on page 927

EngSplitBillingRule

The EngSplitBillingRule object allows users to retrieve and update contracts with split billing rules.

Namespace

`http://changepoint.com/changepoint/CPWebService/EngSplitBillingRule`

URL

`http://webserver/CPWebService/Objects/CPEngagement/EngSplitBillingRule.asmx`

Methods

Properties

For more information, see the "ApiEngSplitBillingRule" section on page 238.

Related information

"Engagement" on page 818

EngSplitBillingRule: GetById

```
Public Function GetById(ByVal sId As String) As WSEngSplitBillingRule
```

Purpose

Fills the ApiEngSplitBillingRule object with the split billing rule information of the specified EngagementId passed in the parameter sId.

Parameters

Parameter	Description
sId	ID of contract.

Returns

Type	Description
WSEngSplitBillingRule	Returns an ApiEngSplitBillingRule object. `Access the object through the .Value property of WSEngSplitBillingRule.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngSplitBillingRule.EngSplitBillingRuleWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngSplitBillingRule.WSEngSplitBillingRule = myProxy.GetById("
{5A390603-6748-454E-8AD3-FC8D65FABA88}")
```

Related information

"EngSplitBillingRule" on page 931

EngSplitBillingRule: GetByMainEngagementId

```
Public Function GetByMainEngagementId(ByVal sEngagementId As String) As
WSString
```

Purpose

Retrieve the contract split billing rules for the specified contract

Parameters

Parameter	Description
sEngagementId	Engagement ID

Returns

Type	Description
WSString	String with the contract split billing rules for the specified contract, or an empty string if there are no rules. Access the returned string through the .Value property of WSString

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngSplitBillingRule.EngSplitBillingRuleWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngSplitBillingRule.WSString = myProxy. GetByMainEngagementId("
{73501433-32EB-491C-B7E4-F03E6824659D}")
```

Related information

"EngSplitBillingRule" on page 931

EngSplitBillingRule: GetSplitEngagementId

```
Public Function getSplitEngagementId(ByVal sEngagementId As String, ByVal
sCustomerId As String) As WSString
```

Purpose

Returns the split contract ID.

Parameters

Parameter	Description
sEngagementId	Engagement ID
sCustomerId	Customer ID

Returns

Type	Description
WSString	String with the split contractID. Access the returned string through the .Value property of WSString

Remarks

None.

Example

None.

Related information

"EngSplitBillingRule" on page 931

EngSplitBillingRule: GetXMLStructure

```
Public Function GetXMLStructure() As WSString
```

Purpose

Retrieve the XML structure of the contract split rule

Parameters

None

Returns

Type	Description
WSString	XML structure of the contract split rule. Access the returned string through the .Value property of WSString

Remarks

For more information, see the "ApiEngSplitBillingRule XML" section on page 239.

Example

```
Dim myProxy As New webEngSplitBillingRule.EngSplitBillingRuleWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngSplitBillingRule.WSString = myProxy.GetXMLStructure()
```

Related information

"EngSplitBillingRule" on page 931

EngSplitBillingRule: Save

```
Public Function Save(ByVal sMainEngagementId As String, ByVal
sXmlEngSplitBillingRules As String) As WSInt32
```

Purpose

Updates the general information of split billing rule of a contract

Parameters

Parameter	Description
sMainEngagementId	The main contract ID.
sXmlEngSplitBillingRules	ApiEngSplitBillingRule objects in XML format string.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

The parameter sXmlEngSplitBillingRules must contain split billing rules that belong to the parameter sMainEngagementId. In other words, sMainEngagementId is a main contract of the split contracts in the sXmlEngSplitBillingRules.

Use the GetByMainEngagement method to get the correct XML format of parameter sXmlEngSplitBillingRules.

Includes ADD, UPDATE and DELETE functionalities. When the value of the deleted tag is true, it indicates DELETE; when the value of the splitengagementid tag is empty, it indicates ADD, when the value of the splitengagementid tag is a GUID, it indicates UPDATE.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngSplitBillingRule.EngSplitBillingRuleWse
Dim sXMLEngSplitBillingRule As String
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)

sXMLEngSplitBillingRule =
"<root>
<engsplitbillingrule>
<splitengagementid>{5A390603-6748-454E-8AD3-FC8D65FABA88}</splitengagementid>
<customerid>{E6367714-DBD2-4716-8097-74F8164618DA}</customerid>
<customername>SHK Trading Secondary Co LTD</customername>
<maincontactid>{59B84C7E-9033-4771-AD53-80EB3EA8B599}</maincontactid>
<maincontactname>Jane Smith</maincontactname>
<percentage>50</percentage>
<deleted>0</deleted>
</engsplitbillingrule>
<engsplitbillingrule>
<splitengagementid></splitengagementid>
<customerid>{40C01201-37EC-47E1-8226-F505760C42F9}</customerid>
<customername>SHK Trading Co LTD</customername>
<maincontactid>{BFA422F0-E7EB-4F9B-88D3-E487002E50BB}</maincontactid>
<maincontactname>Bob Hardy</maincontactname>
<percentage>50</percentage>
<deleted>0</deleted>
</engsplitbillingrule>
<engsplitbillingrule>
<splitengagementid>{2C32B0FB-563C-4CA8-AACC-F673E2068A60}</splitengagementid>
<customerid>{C2932E66-5B75-46E3-87BE-CDEDA686048B}</customerid>
<customername>SHK Trading Third Co LTD</customername>
<maincontactid>{0749DDA0-5A1D-4A02-90A2-00004C65F3AC}</maincontactid>
<maincontactname>Alice Monroe</maincontactname>
<percentage>0</percentage>
<deleted>1</deleted>
</engsplitbillingrule>
</root>"
```

```
Dim oRet As webEngSplitBillingRule.WSInt32 = myProxy.Save("{73501433-32EB-
491C-B7E4-F03E6824659D}", sXMLEngSplitBillingRule)
```

Related information

"EngSplitBillingRule" on page 931

"EngSplitBillingRule: GetByMainEngagementId" on page 932

"EngSplitBillingRule: GetXMLStructure" on page 934

EngWorkCodeLocation

The EngWorkCodeLocation object represents the work code and work location information for a contract.

Namespace

```
http://changepoint.com/changepoint/CPWebService/EngWorkCodeLocation
```

URL

```
http://webserver/CPWebService/Objects/CPEngagement/EngWorkCodeLocation.asmx
```

Methods

Properties

For more information, see the "ApiEngWorkCode" section on page 245.

Related information

"Engagement" on page 818

EngWorkCodeLocation: GetWorkCode

```
Public Function GetWorkCode(ByVal sEngagementId As String) As WSEngWorkCode
```

Purpose

Fills the ApiEngWorkCode object with work code information of the specified EngagementId passed in the parameter.

Parameters

Parameter	Description
sEngagementId	ID of contract that will be retrieved.

Returns

Type	Description
WSEngWorkCode	Returns an ApiEngWorkCode object. Access the object through the .Value property of WSEngWorkCode.

Remarks

Property sXMLWorkCodes contains all work code category and work Code. If AllWorkCodes is True, then True is selected for all work codes; otherwise, True is selected for the workcodes associated with this contract only.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngWorkCodeLocation.EngWorkCodeLocationWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngWorkCodeLocation.WSEngWorkCode = myProxy.GetWorkCode("
{5A390603-6748-454E-8AD3-FC8D65FABA88}")
```

Related information

"EngWorkCodeLocation" on page 937

EngWorkCodeLocation: GetWorkCodeXmlStructure

```
Public Function GetWorkCodeXmlStructure() As WSString
```

Purpose

Returns the XML structure of the contract work codes

Parameters

None

Returns

Type	Description
WSString	XML structure of the contract work codes Access the returned string through the .Value property of WSString

Remarks

For more information, see the "ApiEngWorkCode XML" section on page 247.

Example

```
Dim myProxy As New webEngWorkCodeLocation.EngWorkCodeLocationWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngWorkCodeLocation.WSString=myProxy.GetWorkCodeXmlStructure()
```

Related information

"EngWorkCodeLocation" on page 937

EngWorkCodeLocation: SaveWorkCode

```
Public Function SaveWorkCode(ByVal oApiEngWorkCode As ApiEngWorkCode) As
WSInt32
```

Purpose

Update the general information of work codes of a contract

Parameters

Parameter	Description
oApiEngWorkCode	Populated ApiEngWorkCode object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Use the GetWorkCode method or GetWorkCodeXMLStructure to obtain the correct XML format of property sXMLWorkCodes of ApiEngWorkCode object.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngWorkCodeLocation.EngWorkCodeLocationWse
Dim myEngWorkCode As New webEngWorkCodeLocation.ApiEngWorkCode
Dim sXmlCodes As String =
"<root>
<engworkcode>
<workcodecategoryid>{AA9C83C1-6E1E-4974-9DEC-
7C554CC429D2}</workcodecategoryid>
<workcodecategory>Default Work Code Category</workcodecategory>
<workcodeid>{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}</workcodeid>
<workcode>Default Work Code</workcode><selected>1</selected>
</engworkcode>
<engworkcode>
   .
   .
</engworkcode>
</root>"

'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngWorkCodeLocation.WSEngWorkCode = myProxy.GetWorkCode("
{5A390603-6748-454E-8AD3-FC8D65FABA88}")
If Not oRet.WSException.HaveErrors Then
myEngWorkCode = oRet.value
End If
With myEngWorkCode
   .AllWorkCodes = False
   .AllowRTCodeOverride = True
```

```
    .DefaultWorkCodeCategory = New webEngWorkCodeLocation.Identity()
    .DefaultWorkCodeCategory.Id = "{B7E06668-7616-42F8-A0C0-97A561F2EFF2}"
    .DefaultWorkCode = New webEngWorkCodeLocation.Identity()
    .DefaultWorkCode.Id = "{695D56A1-099F-45D1-8ACB-0ED4D240C8EB}"
    .sXMLWorkCodes = sXmlCodes
    .
    .
End With
Dim oRetInt32 As webEngWorkCodeLocation.WSInt32 = myProxy.SaveWorkCode
(myEngWorkCode)
```

Related information

"EngWorkCodeLocation" on page 937

"EngWorkCodeLocation: GetWorkCode" on page 937

"EngWorkCodeLocation: GetWorkCodeXmlStructure" on page 938

EngWorkCodeLocation: GetWorkLocation

```
Public Function GetWorkLocation(ByVal sEngagementId As String) As
WSEngWorkLocation
```

Purpose

Fills the ApiEngWorkLocation object with work location information of the specified EngagementId passed in the parameter.

Parameters

Parameter	Description
sEngagementId	ID of contract that will be retrieved.

Returns

Type	Description
WSEngWorkLocation	Returns an ApiEngWorkLocation object. `Access the object through the .Value property of WSEngWorkLocation.`

Remarks

Property sXMLWorkLocations contains all GlobalWorkLocations and worklocations. If AllLocations is True, then True is selected for all locations; otherwise True is selected the locations that are associated with this contract only.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngWorkCodeLocation.EngWorkCodeLocationWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngWorkCodeLocation.WSEngWorkLocation = myProxy.GetWorkLocation
("{5A390603-6748-454E-8AD3-FC8D65FABA88}")
```

Related information

"EngWorkCodeLocation" on page 937

EngWorkCodeLocation: GetWorkLocationXmlStructure

```
Public Function GetWorkLocationXmlStructure() As WSString
```

Purpose

Returns the XML structure of the contract work locations

Parameters

None

Returns

Type	Description
WSString	XML structure of the contract work locations. Access the returned string through the .Value property of WSString

Remarks

For more information, see the "ApiEngWorkLocation XML" section on page 252.

Example

```
Dim myProxy As New webEngWorkCodeLocation.EngWorkCodeLocationWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As
webEngWorkCodeLocation.WSString=myProxy.GetWorkLocationXmlStructure()
```

Related information

"EngWorkCodeLocation" on page 937

EngWorkCodeLocation: SaveWorkLocation

```
Public Function SaveWorkLocation(ByVal oApiEngWorkLocation As
ApiEngWorkLocation) As WSInt32
```

Purpose

Update the general information of work locations of an engagement

Parameters

Parameter	Description
oApiEngWorkLocation	Populated ApiEngWorkLocation object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

It is recommended to use the GetWorkLocation method or GetWorkLocationXMLStructure to obtain the correct XML format of property sXmlWorkLocations of ApiEngWorkLocation object.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webEngWorkCodeLocation.EngWorkCodeLocationWse
Dim myEngWorkLocation As New webEngWorkCodeLocation.ApiEngWorkLocation
Dim sXmlLocations As String =
"<root>
<engworklocation>
<worklocationgroupid>{210FD5FB-6493-478D-8A13-
A730F6A581E3}</worklocationgroupid>
<worklocationgroup>Japan</worklocationgroup>
<worklocationid>{4EE1226F-A64E-4E93-86D2-CA6A5B89C58C}</worklocationid>
<worklocation>Tokyu</worklocation><selected>1</selected>
</engworklocation>
<engworklocation>
   .
   .
</engworklocation>
</root>"

'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webEngWorkCodeLocation.WSEngWorkLocation = myProxy.GetWorkLocation
("{5A390603-6748-454E-8AD3-FC8D65FABA88}")
If Not oRet.WSException.HaveErrors Then
   myEngWorkLocation = oRet.value
End If
With myEngWorkLocation
   .AllLocations = False
   .AllowRTLocOverride = True
   .DefaultWorkLocationGroup = New webEngWorkCodeLocation.Identity()
   .DefaultWorkLocationGroup.Id = "{210FD5FB-6493-478D-8A13-A730F6A581E3}"
   .DefaultWorkLocation = New webEngWorkCodeLocation.Identity()
   .DefaultWorkLocation.Id = "{4EE1226F-A64E-4E93-86D2-CA6A5B89C58C}"
   .sXMLWorkLocations = sXmlLocations
End With
Dim oRetInt32 As webEngWorkCodeLocation.WSInt32 = myProxy.SaveWorkLocation
(myEngWorkLocation)
```

Related information

"EngWorkCodeLocation" on page 937

"EngWorkCodeLocation: GetWorkLocation" on page 941

"EngWorkCodeLocation: GetWorkLocationXmlStructure" on page 942

ExchangeRate

The ExchangeRate object allows users to add, edit, and retrieve exchange rates from the Changepoint database.

Namespace

`http://changepoint.com/changepoint/CPWebService/ExchangeRate`

URL

`http://webserver/CPWebService/Objects/CPExchangeRate/ExchangeRate.asmx`

Namespace

ChangepointAPI.ExchangeRate

Methods

Properties

For more information, see the "ApiExchangeRate" section on page 257.

ExchangeRate: Add

```
Public Function Add(ByVal oExchangeRate As ApiExchangeRate) As WSString
```

Purpose

Adds a new Exchange Rate into the Changepoint database.

Parameters

Parameter	Description
oExchangeRate	ApiExchangeRate object.

Returns

Type	Description
WSString	ExchangeRateId returned on success. Access the returned string through the .Value property of WSString

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim oExRate As New webExchangeRate.ApiExchangeRate
With oExRate
.BaseCurr = "CAD"
.ToCurr = "USD"
.StartDate = "2007-06-19"
.EndDate = "2007-10-31"
.Rate = "0.925"
.CurrentRate = True
End With
Dim proxy As New webExchangeRate.ExchangeRateWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As webExchangeRate.WSString = Proxy.Add(oExRate)
```

Related information

"ExchangeRate" on page 945

ExchangeRate: GetExchangeRateByDate

```
Public Function GetExchangeRateByDate(ByVal BaseCurrencyCode As String, ByVal
ToCurrencyCode As String, ByVal SDate As Date) As WSString
```

Purpose

Retrieve an Exchange Rate filtered by Parameters

Parameters

Parameter	Description
BaseCurrencyCode	Base currency three-character ISO code in Changepoint.
ToCurrencyCode	To Currency three-character ISO code in Changepoint.
SDate	Exchange Date

Returns

Type	Description
WSString	ExchangeRate returned on success. Access the returned string through the .Value property of WSString

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

Not available

Related information

"ExchangeRate" on page 945

ExchangeRate: GetExchangeRateId

```
Public Function GetExchangeRateId(ByVal BaseCurrencyCode As String, ByVal
ToCurrencyCode As String, ByVal SDate As Date) As WSString
```

Purpose

Retrieve an ExchangeRateId filtered by Parameters

Parameters

Parameter	Description
BaseCurrencyCode	Base currency three-character ISO code in Changepoint.
ToCurrencyCode	To Currency three-character ISO code in Changepoint.
SDate	Exchange Date

Returns

Type	Description
WSString	ExchangeRateId returned on success. Access the returned string through the .Value property of WSString

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

Not available

Related information

"ExchangeRate" on page 945

ExchangeRate: GetExRate

```
Public Function GetExRate(ByVal sBaseCurr As String, ByVal sToCurr As String,
ByVal ActionDate As Date) As WSExchangeRate
```

Purpose

Retrieve Exchange Rate info in ApiExchangeRate object

Parameters

Parameter	Description
sBaseCurrency	Base currency three-character ISO code in Changepoint.
sToCurrency	To Currency three-character ISO code in Changepoint.
ActionDate	Exchange Date

Returns

Type	Description
WSExchangeRate	Returns an ApiExchangeRate object. `Access the object through the .Value property of WSExchangeRate.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webExchangeRate.ExchangeRateWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As webExchangeRate.ApiWSExchangeRate = Proxy.GetExRate("CAD", "USD",
"2007-04-12")
```

Related information

"ExchangeRate" on page 945

ExchangeRate: Update

```
Public Function Update(ByVal oExchangeRate As ApiExchangeRate) As WSInt32
```

Purpose

Update Exchange Rate info to Changepoint database

Parameters

Parameter	Description
oExchangeRate	ApiExchangeRate object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webExchangeRate.ExchangeRateWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As webExchangeRate.WSExchangeRate = Proxy.GetExRate("CAD", "USD",
"2007-04-12")
If Not oRet.WSException.HaveErrors then
Dim oExRate As webExchangeRate.ApiExchangeRate = oRet.value
With oExRate
.BaseCurr = "CAD"
.ToCurr = "USD"
.StartDate = "2007-08-27"
.EndDate = "2007-10-10"
.Rate = "0.925"
.CurrentRate = True
End With
Dim iRet As webExchangeRate.WSInt32 = Proxy.Update(oExRate)
End If
```

Related information

"ExchangeRate" on page 945

Expense

The Expense object allows users to create, retrieve and update expenses in the Changepoint database.

Namespace

`http://changepoint.com/changepoint/CPWebService/Expense`

URL

`http://webserver/CPWebService/Objects/CPExpense/Expense.asmx`

Methods

Properties

For more information, see "ApiExpense" on page 262.

Related information

"ApiExpense XML" on page 266

Expense: Add

```
Public Function Add(ByRef sId As String, ByVal oExpense As ApiExpense) As
WSInt32
```

Purpose

Adds a new expense to Changepoint

Parameters

Parameter	Description
sId	ByRef parameter that will return the new ExpenseId after the expense has been added.
oExpense	Expense object that will be added.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

See the ApiExpense properties for mandatory fields.

Example

Not available

Related information

"Expense" on page 951

"ApiExpense" on page 262

Expense: CreateByXML

```
CreateByXML(ByVal sXML As String, ByRef sId As String) As WSInt32
```

Purpose

Create an expense using an XML string of the Expense object in Changepoint.

Parameters

Parameter	Description
sXML	A new expense XML string which is passed based on the Expense XML schema
sId	ByRef parameter that returns the ExpenseId after the expense has been added.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The ApiExpense XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The ByPassMetadataCheck switch will stop metadata validation in the Expense.

Example

Not available

Related information

"Expense" on page 951

"Expense: GetByXML" on page 957

"Expense: GetXMLStructure" on page 973

"ApiExpense XML" on page 266

Expense: Delete

```
Public Function Delete(ByVal sId as String = "") As WSInt32
```

Purpose

Deletes the specified expense from Changepoint

Parameters

Parameter	Description
sId	Expense ID of the expense to delete

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: Exists

```
Public Function Exists(ByVal sId As String) As WSBoolean
```

Purpose

Verifies whether the expense exists in the database and has not been deleted.

Parameters

Parameter	Description
sId	Expense ID of the expense to verify

Returns

Type	Description
WSBoolean	True if the expense exists, else false. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetAssociatedTasks

```
Public Function GetAssociatedTasks(ByVal sResourceId As String, ByVal
sProjectId As String, ByVal sSearchString As String) As WSDataSet
```

Purpose

Returns a list of tasks associated with the expense

Parameters

Parameter	Description
sResourceId	Resource ID
sProjectId	Project ID
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (TaskId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetById

```
Public Function GetById(ByVal sId As String) As WSExpense
```

Purpose

Fills the object with data related to the specified ExpenseId passed in the parameter.

Parameters

Parameter	Description
sId	Expense ID of the expense whose data is to be retrieved

Returns

Type	Description
WSExpense	Access the object through the .Value property of WSExpense.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetByXML

```
Public Function GetByXML(ByVal sXML As String, ByVal sExpenseId As String) As
WSString
```

Purpose

Takes the XML string passed in the sXML parameter and returns the string filled with data for the Expense ID specified in the sExpenseId parameter.

Parameters

Parameter	Description
sXML	XML string of the Expense object, with the fields specified
sExpenseId	Expense ID • If no Expense ID is passed in, the XML string (sXML) is examined • if there is no Expense ID in sXML then the object's Expense ID is selected. • If there still is no valid ExpenseId, the method returns an error.

Returns

Type	Description
WSString	An XML string mirroring sXML with data inserted or the entire XML of the Expense object including data. Access the returned string through the .Value property of WSString

Remarks

If sXML = "" then the XML string provided by GetXMLStructure is used.

Example

Not available

Related information

"Expense" on page 951

Expense: GetCategories

```
Public Function GetCategories(ByVal sSearchString As String) As WSDataSet
```

Purpose

Returns a list of expense categories.

Parameters

Parameter	Description
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (ExpenseCategoryId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetCurrency

```
Public Function GetCurrency() As WSDataSet
```

Purpose

Returns a currency list

Parameters

None

Returns

Not applicable

Type	Description
WSDataset	WSDataset (Code, Description) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetCustomers

```
Public Function GetCustomers(ByVal sResourceId As Guid, ByVal sSearchString as
String = "") As WSDataSet
```

Purpose

Returns a list of customers.

Parameters

Parameter	Description
sResourceId	Resource ID
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (CustomerId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetEngagements

```
Public Function GetEngagements(ByVal sResourceId As String, ByVal sCustomerId
As String, ByVal sSearchString As String) As WSDataSet
```

Purpose

Returns a list of contracts for a specified customer.

Parameters

Parameter	Description
sResourceId	Resource ID
sCustomerId	Customer ID
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (EngagementId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetExpenseList

```
Public Function GetExpenseList(ByVal iRetRows As Short, ByVal sCustomerId As
String, ByVal sEngagementId As String, ByVal sProjectId As String, ByVal
sResourceId As String) As WSDataSet
```

Purpose

Returns a list of some or all of the expenses based on the specified parameters.

Parameters

Parameter	Description
iRetRows	Number of rows to return. Default = -1 (return all).
sCustomerId	Customer ID
sEngagementId	Engagement ID
sProjectId	Project ID
sResourceId	Resource ID

Returns

Type	Description
WSDataset	WSDataset (ExpenseId, CustomerId, EngagementId, ProjectId, ResourceId, CategoryId, ExpenseType, ExpenseDate, Description, Billable, Quantity, UnitPrice, TaxAmount, CurrencyCode, ExchangeRate, Editable, SubmittedForApproval, ExpenseReportID, ReimbursableExpense) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetFixedFees

```
Public Function GetFixedFees(ByVal sEngagementId As String) As WSDataSet
```

Purpose

Returns a list of fixed fees for a specifed contract.

Parameters

Parameter	Description
sEngagementId	Engagement ID

Returns

Type	Description
WSDataset	WSDataset (FixedFeeID, FixedFeeDescription) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As WSString
```

Purpose

Returns the ExpenseId based on the UDF Text field and value.

Parameters

Parameter	Description
sUDFField	UDF text field name (for example, Text1)
sUDFValue	UDF text field value

Returns

Type	Description
WSString	String with the ExpenseId or an empty string. Access the returned string through the .Value property of WSString

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetList

Public Function GetList(ByVal iRetRows As Int16) As WSDataSet

Purpose

Returns a list of some or all expenses that were not submitted to an expense report in Changepoint.

Parameters

Parameter	Description
iRetRows	The number of rows to be returned. To return all rows, you must pass -1.

Returns

Type	Description
WSDataset	WSDataset (ExpenseId, CustomerId, EngagementId, ProjectId, ResourceId, CategoryId, ExpenseType, ExpenseDate, Description, Billable, Quantity, UnitPrice, TaxAmount, CurrencyCode, ExchangeRate, Editable, SubmittedForApproval, ExpenseReportID, ReimbursableExpense) or an empty dataset if nothing is found. Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetProjects

```
Public Function GetProjects(ByVal sResourceId As String, ByVal sEngagementId
As String, ByVal sSearchString As String) As WSDataSet
```

Purpose

Returns a list of projects

Parameters

Parameter	Description
sResourceId	Resource ID
sEngagementId	Engagement ID
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (ProjectId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetTypes

```
Public Function GetTypes(ByVal sCategoryId As String, ByVal sSearchString As
String) As WSDataSet
```

Purpose

Returns a list of expense types.

Parameters

Parameter	Description
sCategoryId	Expense category ID
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (ExpenseCategoryId, ExpenseCategoryName, ExpenseTypeId, Description) Access the dataset through the .Value property of WSDataSet.

Remarks

If sCategoryId is not empty, the list contains expense types only for the specifed expense category. If sSearchString is not empty, the list contains expense types only for expenses whose descriptions match sSearchString.

Example

Not available

Related information

"Expense" on page 951

Expense: GetUDF

```
Public Function GetUDF(ByVal retOption As CPUDFReturnType, ByVal entityId As
String, ByVal actionResourceId As String) As WSString
```

Purpose

Retrieve UDF information for an expense in an XML string.

Parameters

Parameter	Description
retOption	The CPUDFReturnType Values are: • WithStructure = 0 – returns UDF field data with full field structure • OnlyValues = 1 – returns only UDF fields with data, does not include any field structure or options • OnlyStructure = 2 – returns only UDF XML structure, no field data • OnlyStructureAndOptions = 3 – returns UDF structure, and option data (except entity-based and conditional codes), without field data • WithStructureAndOptions = 4 – returns UDF field data with full structure and all field options (except entity-based and conditional codes)
entityId	ExpenseId. An empty string is allowed.
actionResourceId	The ResourceId of the user that called the method. An empty string is allowed.

Returns

Type	Description
WSString	UDF information for the expense in an XML string Access the returned string through the .Value property of WSString

Remarks

If ExpenseId is empty, the retOptions that return data – WithStructure (0); OnlyValues (1); and WithStructureAndOptions (4) – return default values for field data.

If actionResourceId is empty, the signed-in user ResourceId is used.

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webExpense.ExpenseWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
```

```
Dim oRet As webExpense.WSString = myProxy.GetUDF
(CPUDFReturnType.WithStructureAndOptions,"{CA8AC6E5-5F9A-41CA-BD57-
9A35D28A7E76}","")
```

Expense: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal codeName As String, ByVal searchString
As String) As WSString
```

Purpose

Retrieve UDF code options in XML format for the specified expense UDF code.

Parameters

Parameter	Description
codeName	Name of the UDF code, for example, "Code3"
searchString	Returns the options that start with this string. An empty string is allowed.

Returns

Type	Description

WSString	`WSString{`
	`WSException WSException;`
	`String value;`
	`}`

Remarks

- Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

Example

```
Dim myProxy As New webExpense.ExpenseWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webExpense.WSString = myProxy.GetUDFCodeOptions("Code2","")
```

Expense: GetWorkCodeCategories

```
Public Function GetWorkCodeCategories(ByVal sProjectId As String, ByVal
sSearchString As String) As WSDataSet
```

Purpose

Returns a lists of workcode categories.

Parameters

.

Parameter	Description
sProjectId	Project ID
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (WorkCodeCategoryId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetWorkCodes

```
Public Function GetWorkCodes(ByVal sProjectId As String, ByVal
sWorkCodeCategoryId As String, ByVal sSearchString As String) As WSDataSet
```

Purpose

Returns a list of work codes

Parameters

Parameter	Description
sProjectId	Project ID
sWorkCodeCategoryId	Work code category ID
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (WorkCodeId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetWorkLocationGroups

```
Public Function GetWorkLocationGroups(ByVal sProjectId As String, ByVal
sSearchString As String) As WSDataSet
```

Purpose

Returns a list of work location groups.

Parameters

Parameter	Description
sProjectId	Project ID
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (WorkLocationGroupId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetWorkLocations

```
Public Function GetWorkLocations(ByVal sProjectId As String, ByVal
sWorkLocationGroupId As String, ByVal sSearchString As String) As WSDataSet
```

Purpose

Returns a list of work locations.

Parameters

Parameter	Description
sProjectId	Project ID
sWorkLocationGroupId	Work location group ID
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (WorkLocationId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: GetXMLStructure

```
Public Function GetXMLStructure() As WSString
```

Purpose

Returns the XML structure of the Expense object.

Parameters

None

Returns

Type	Description
WSString	A string containing the XML structure of the Expense object. Access the returned string through the .Value property of WSString

Remarks

Some fields in the structure will have defaulted data; otherwise, fields are empty.

Example

Not available

Related information

"Expense" on page 951

"ApiExpense XML" on page 266

Expense: SaveUDF

```
Public Function SaveUDF(ByVal sXMLUDF As String, ByVal needValidate As
Boolean, ByVal bypassMetadata As CPMetadataCheck) As WSInt32
```

Purpose

Saves (Insert/Update) UDF information for an expense.

Parameters

Parameter	Description
sXMLUDF	UDF string in XML format to be saved.
needValidate	Flag that determines whether validation is required for sXMLUDF.

Parameter	Description
bypassMetadata	Determines the level of metadata checking for sXMLUDF. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2 If needValidate is False, then bypassMetadata value is ignored.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

To obtain the correct UDF XML format, call the GetUDF method.

Example

```
Dim myProxy As New webExpense.ExpenseWse
Dim strXMLUDF as string ="<root>…</root>"
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
```

Dim oRet As webExpense.WSInt32 = myProxy.SaveUDF(strXMLUDF, True, 0)

Expense: Update

```
Public Function Update(ByVal oExpense As ApiExpense) As WSInt32
```

Purpose

Updates expense data in the database with data held in the object.

Parameters

None

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

Not available

Related information

"Expense" on page 951

Expense: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, ByVal sExpenseId As String)
As WSInt32
```

Purpose

Updates a expense using an XML string containing new data

Parameters

Parameter	Description
sXML	The XML string of the Expense object with new data contained in the fields.
sExpenseId	Expense ID

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Performs the same function as Update except any expense can be updated through this function. The XML sent in the parameter must be of the form in ApiExpense XML. The ApiExpense XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The method uses the following sequence to find the expense ID:

1. If the sExpenseId parameter is passed in, the method uses this value for the expense ID.
2. If this fails, the method attempts to extract the expense ID from <expenseid> in the XML.
3. If this fails, the expense ID is taken from the object properties.

Example

Not available

Related information

"Expense" on page 951

"Expense: GetByXML" on page 957

"Expense: GetXMLStructure" on page 973

"ApiExpense XML" on page 266

ExpenseReport

The ExpenseReport object allows users to retrieve and mark expense reports as batched in the Changepoint database.

Namespace

`http://changepoint.com/changepoint/CPWebService/ExpenseReport`

URL

```
http://webserver/CPWebService/Objects/CPExpense/ExpenseReport.asmx
```

Methods

Properties

For more information, see the "ApiExpenseReport" section on page 291.

ExpenseReport: GetExpenseReport

```
Public Function GetExpenseReport(ByVal ExpenseReportId As String) As
WSExpenseReport
```

Purpose

Retrieves an expense report of specified ExpenseReportId.

Parameters

Parameter	Description
sExpenseReportId	Expense Report Id.

Returns

Type	Description
WSExpenseReport	Returns an ApiExpenseReport object. `Access the object through the .Value property of WSExpenseReport.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webExpenseReport.ExpenseReportWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId As String = "{…}"
Dim oRet As webExpenseReport.WSExpenseReport = Proxy.GetExpenseReport(sId)
```

Related information

"ExpenseReport" on page 977

ExpenseReport: GetExpenseReports

```
Public Function GetExpenseReports(ByVal bBatched As Boolean) As WSDataSet
```

Purpose

Retrieves the expense report list from Chagnepoint database.

Parameters

Parameter	Description
Batched	Boolean value specified to retrieve invoices that have already been batched.

Returns

Type	Description
WSDataSet	WSDataset (ExpenseReportId, ExpenseReportNumber) Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webExpenseReport.ExpenseReportWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As webExpenseReport.WSDataSet = Proxy.GetExpenseReports(False)
```

Related information

"ExpenseReport" on page 977

ExpenseReport: MarkExpenseReportAsBatched

```
Public Function MarkExpenseReportAsBatched(ByVal ExpenseReportId As String) As
WSInt32
```

Purpose

Mark the specified expense report as batched.

Parameters

Parameter	Description
ExpenseReportId	ID of the Expense Report.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Indicates that the expense report by this ID has been extracted from the Changepoint database. The extract date used is the current date. The expense report can be retrieved again if the batched parameter of the GetApprovedExpenseReports is specified as true.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

Not available

Related information

"ExpenseReport" on page 977

Functions

The Functions object allows users to retrieve and update resource function within the Changepoint database.

Namespace

`http://changepoint.com/changepoint/CPWebService/Functions`

URL

`http://webserver/CPWebService/Objects/CPSkill/Functions.asmx`

Methods

Properties

For more information, see the "ApiFunctionDescription" section on page 297.

Functions: Add

```
Public Function Add(ByVal myFunctionDescription As ApiFunctionDescription) As
WSString
```

Purpose

Insert one row into FunctionDescription table.

Parameters

Parameter	Description
myFunctionDescription	ApiFunctionDescription object.

Returns

Type	Description
WSString	Returns a FunctionId. Access the returned string through the .Value property of WSString.

Remarks

None

Example

```
Dim myProxy As New webFunctions.FunctionsWse
Dim myFunction As New webFunctions. ApiFunctionDescription
Dim oRet As webFunctions.WSString
UserToken.SetToken(myProxy, mUserName, mPassword)
With myFunction
   .Deleted = False
   .Description = "Managing projects"
   .Name = "Project Manager"
End With
oRet = myProxy.Add(myFunction)
```

Related information

"Functions" on page 981

Functions: GetFunction

```
Public Function GetFunction(ByVal sFunctionId As String) As WSFunction
```

Purpose

Retrieve a function description record

Parameters

Parameter	Description
sFunctionId	Changepoint Function Id.

Returns

Type	Description
WSFunction	Returns an ApiFunction object. Access the object through the .Value property of WSFunction.

Remarks

None

Example

Not available

Related information

"Functions" on page 981

Functions: GetFunctionByBillingRole

```
Public Function GetFunctionByBillingRole(ByVal sBillingRoleId As String) As
WSString
```

Purpose

Retrieve a function ID of a billing role

Parameters

Parameter	Description
sBillingRoleId	Read-only. Billing Role ID in Changepoint.

Returns

Type	Description
WSString	Returns Function ID for the specific billing role. Access the returned string through the .Value property of WSString

Remarks

None

Example

Not available

Related information

"Functions" on page 981

Functions: GetFunctions

```
Public Function GetFunctions() As WSDataSet
```

Purpose

Retrieve all function description records

Parameters

None

Returns

Type	Description
WSDataSet	WSDataset (FunctionId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None.

Example

Not available

Related information

"Functions" on page 981

Functions: Update

```
Public Function Update(ByVal myFunctionDescription As ApiFunctionDescription)
As WSInt32
```

Purpose

Update a function description record.

Parameters

Parameter	Description
myFunctionDescription	ApiFunctionDescription object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

```
Dim myProxy As New webFunctions.FunctionsWse
Dim myFunction As New webFunctions.ApiFunctionDescription
Dim oRet As webFunctions.WSInt32

UserToken.SetToken(myProxy, mUserName, mPassword)

With myFunction
   .FunctionId = "{3EA44C0E-0D31-4FB1-A6C3-3703A837B9B2}"
   .Description = "First level management"
   .Name = "Team Leader 1"
End With
oRet = myProxy.Update(myFunction)
```

Related information

"Functions" on page 981

Functions: UptBillingRoleWithFunction

```
Public Function UptBillingRoleWithFunction(ByVal sBillingRoleId As String,
ByVal sFunctionId As String) As WSInt32
```

Purpose

Updates the function ID in the billing role record.

Parameters

Parameter	Description
sBillingRoleId	Billing Role ID in Changepoint.
sFunctionId	Function ID in Changepoint.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

```
Dim myProxy As New webFunctions.FunctionsWse
Dim oRet As webFunctions.WSInt32

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.UptBillingRoleWithFunction("{7E4F8CF4-363B-11D4-8AB1-
0001023D3221}", "{3EA44C0E-0D31-4FB1-A6C3-3703A837B9B2}")
```

Related information

"Functions" on page 981

FunctionSkill

The FunctionSkill object represents the skills that are associated to functions within the Changepoint database.

Namespace

`http://changepoint.com/changepoint/CPWebService/FunctionSkill`

URL

`http://webserver/CPWebService/Objects/CPSkill/FunctionSkill.asmx`

Methods

Properties

For more information, see the "ApiFunctionSkill" section on page 303.

FunctionSkill: GetFunctionSkills

```
Public Function GetFunctionSkills(ByVal sFunctionId As String) As
WSFunctionSkillArray
```

Purpose

Retrieve function skills for a function

Parameters

Parameter	Description
sFunctionId	Changepoint Function ID.

Returns

Type	Description
WSFunctionSkillArray	Returns an array of ApiFunctionSkill objects. Access the array through the .Value property of WSFunctionSkillArray.

Remarks

None

Example

Not available

Related information

"FunctionSkill" on page 987

FunctionSkill: Save

```
Public Function Save(ByVal sFunctionId As String, ByVal myFunctionSkills() As
ApiFunctionSkill) As WSInt32
```

Purpose

Update function skills and skill competencies for a function.

Parameters

Parameter	Description
sFunctionId	Changepoint Function ID
myFunctionSkills()	Array of the ApiFunctionSkill object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Add, update and delete skills and competences for a function.

The myFunctionSkills array should include all function skill and skill competency records for sFunctionId. If any of these records exist in the current table, but are not included with the myFunctionSkills, they will be deleted.

Example

```
Dim myProxy As New webFunctionSkill.FunctionSkillWse
Dim myFunction As New webFunctionSkill.ApiFunctionSkill
Dim myFunctionSkills() As webFunctionSkill.ApiFunctionSkill
Dim sFunctionId As String
Dim oRet As webFunctionSkill.WSInt32

UserToken.SetToken(myProxy, MainForm.mUserName, MainForm.mPassword)
With myFunction
   .FunctionId = "{E3389CF7-58E6-44B5-B9C4-008E0FF22B05}"
   .SkillId = "{693CEBFA-039C-11D6-BFAF-0060975AEC0F}"
   .SkillCompetencyId = "{693CEBFB-039C-11D6-BFAF0060975AEC0F}"
End With

ReDim myFunctionSkills(0)
myFunctionSkills(0) = myFunction

sFunctionId = "{E3389CF7-58E6-44B5-B9C4-008E0FF22B05}"
oRet = myProxy.Update(sFunctionId, myFunctionSkills)
```

Related information

"FunctionSkill" on page 987

FunctionSkill: ToXml

```
Public Function ToXml(ByVal oFunctionSkill As ApiFunctionSkill) As WSXmlNode
```

Purpose

Convert an ApiFunctionSkill object to an XML node.

Parameters

Parameter	Description
oFunctionSkill	ApiFunctionSkill object

Returns

Type	Description
WSXmlNode	XML node in WSXmlNode object. Access the node through the .Value property of WSXmlNode.

Remarks

For more information, see the "ApiFunctionSkill XML" section on page 304.

Example

Not available

Related information

"FunctionSkill" on page 987

Invoice

The Invoice object allows users to retrieve and update invoices within the Changepoint database.

Namespace

`http://changepoint.com/changepoint/CPWebService/Invoice`

Url

`http://webserver/CPWebService/Objects/CPInvoice/Invoice.asmx`

Methods

Properties

For more information, see the "ApiInvoice" section on page 307.

Related information

"ApiInvAdditionalItem" on page 328

"ApiInvExpense" on page 329

"ApiInvFixedFee" on page 330

"ApiInvPaymentInfo" on page 331

"ApiInvProduct" on page 332

"ApiInvSupportTime" on page 334

"ApiInvTime" on page 335

"ApiInvWOExpenses" on page 336

"ApiInvWOProduct" on page 338

"ApiInvWOSupport" on page 339

"ApiInvWOTime" on page 340

Invoice: AddPaymentInfo

```
Public Function AddPaymentInfo(ByRef sId As String, ByVal sInvoiceId As
String, ByVal sCustomerId As String, ByVal sBillingOfficeId As String, ByVal
sEngagementId As String, ByVal sPaymentCurrencyCode As String, ByVal
dPaymentAmount As Decimal, ByVal dtaPaymentDate As Date, ByVal sPaymentExRate
As String, ByVal dPaymentExRate As Decimal) As WSInt32
```

Purpose

Adds a payment information record to an invoice.

Parameters

Parameter	Description
sId	PaymentInfoId
sInvoiceId	Invoice Id
sCustomerId	Customer Id
sBillingOfficeId	Billing office Id
sEngagementId	Engagement Id
sPaymentCurrencyCode	Payment Currency Code
dPaymentAmount	Payment Amount
dtaPaymentDate	Payment Date
sPaymentExRate	Payment Exchange Rate Id
dPaymentExRate	Payment Exchange Rate

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Populate the ApiInvPaymentInfo object to add a payment to an Invoice: the LookupIds object provides methods to retrieve CustomerId, EngagementId, PaymentCurrency and PaymentExchangeRateId information. The PaymentExchangeRate information is returned but does not have to be passed when creating a payment.

If the net of the invoice total minus the invoice payments is greater than 0, and the status of the invoice was not "PP" (Partially Paid), then the status is set to "PP".

If the net of the invoice minus the invoice payments is less than or equal to 0, and the status was not "P" (Paid), then the status is set to "P". Note that the API can overpay an invoice (since the application can), and both positive and negative payments are accepted.

The error returns are written to log file.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSInt32
Dim sId As String = ""
Dim sInvoiceId As String = "{0385FABE-37C9-11D4-8E19-0050DA7BA6B1}"
Dim sCustomerId As String = "{7A2F9C0E-370A-11D4-8E15-0050DA7BA6B1}"
Dim sBillingOfficeId As String = "{DB587BF6-3665-11D4-8E13-0050DA7BA6B1}"
Dim sEngagementId As String = "{6D4CD6E2-3724-11D4-8E15-0050DA7BA6B1}"
Dim sPaymentCurrencyCode As String = "USD"
Dim dPaymentAmount As Decimal = 3000
Dim dtaPaymentDate As Date = "05/18/2007"
Dim sPaymentExRate As String = "{6FFCABE0-3473-11D3-807A-00105A0B7C01}"
Dim dPaymentExRate As Decimal = 1.5

UserToken.SetToken(myProxy, mUserName, mPassword)

oRet = myProxy.AddPaymentInfo(sId, sInvoiceId, sCustomerId, sBillingOfficeId,
sEngagementId, sPaymentCurrencyCode, dPaymentAmount, dtaPaymentDate,
sPaymentExRate, dPaymentExRate)
```

Related information

"Invoice" on page 990

Invoice: AddPaymentInfoByClass

```
Public Function AddPaymentInfoByClass(ByRef sId As String, ByVal oPayment As
ApiInvPaymentInfo) As WSInt32
```

Purpose

Adds a payment information record to an invoice.

Parameters

Parameter	Description
sId	PaymentInfoId. Newly created PaymentInfoId is returned in sId upon success.
oPayment	Populated ApiInvPaymentInfo. For more information, see the "ApiInvPaymentInfo" section on page 331.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

If the net of the invoice minus the invoice payments is less than or equal to 0, and the status was not "P" (Paid), then the status is set to "P". Note that the API can overpay an invoice (since the application can), and both positive and negative payments are accepted.

The error returns are written to the log file with error messages.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim myApiInvPaymentInfo As New webInvoice.ApiInvPaymentInfo
Dim oRet As webInvoice.WSInt32
```

```
Dim sId As String = ""

UserToken.SetToken(myProxy, MainForm.mUserName, MainForm.mPassword)

With myApiInvPaymentInfo
   .sInvoiceId = "{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}"
   .dPaymentAmount = 5000
   .sPaymentCurrencyCode = "USD"
   .dtaPaymentDate = "05/18/2007"
End With

oRet = myProxy.AddPaymentInfoByClass(sId, myApiInvPaymentInfo)
```

Related information

"Invoice" on page 990

Invoice: CheckPaymentTotal

```
Public Function CheckPaymentTotal(ByVal sInvoiceId As String, ByVal
dInvoiceTotal As Decimal) As WSInt32
```

Purpose

Check whether the invoice total amount is fully paid.

Parameters

Parameter	Description
InvoiceId	ID of the invoice.
dInvoiceTotal	Invoice total.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSInt32

UserToken.SetToken(myProxy, mUserName, mPassword)
'All paid, so returns 0
oRet = myProxy.CheckPaymentTotal("{5B60D14D-8C29-491E-86A6-867DEA4CFB7B}",
8000)
```

Related information

"Invoice" on page 990

Invoice: GetAdditionalItems

```
Public Function GetAdditionalItems(ByVal sInvoiceId As String) As
WSAdditionalItemArray
```

Purpose

Retrieves additional item information of an invoice.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice.

Returns

Type	Description
WSAdditionalItemArray	Returns an array of ApiInvAdditionalItem objects. Access the array through the .Value property of WSAdditionalItemArray.

Remarks

For more information, see the "ApiInvAdditionalItem" section on page 328.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSAdditionalItemArray

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetAdditionalItems("{30CA19EB-37D5-11D4-8AB1-0001023D3221}")
```

Related information

"Invoice" on page 990

Invoice: GetBillingOfficeById

```
Public Function GetBillingOfficeById(ByVal sBillingOfficeId As String)
WSDataSet
```

Purpose

Retrieves billing office ID and description.

Parameters

Parameter	Description
sBillingOfficeId	ID of the Billing Office.

Returns

Type	Description
WSDataSet	WSDataset (billingofficeId, name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Invoice" on page 990

Invoice: GetExpenseWriteOffs

```
Public Function GetExpenseWriteOffs(ByVal sInvoiceId As String) As
WSExpenseWriteOffArray
```

Purpose

Retrieves all expenses that were written off on an invoice.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice.

Returns

Type	Description
WSExpenseWriteOffArray	Returns an array of ApiInvWOExpenses objects. Access the array through the .Value property of WSExpenseWriteOffArray.

Remarks

For more information, see the "ApiInvWOExpenses" section on page 336.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSExpenseWriteOffArray
UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetExpenseWriteOffs("{30CA19EB-37D5-11D4-8AB1-0001023D3221}")
```

Related information

"Invoice" on page 990

Invoice: GetInvoice

```
Public Function GetInvoice(ByVal sInvoiceNumber As String) As WSInvoice
```

Purpose

Retrieves the invoice for the specified invoice number.

Parameters

Parameter	Description
sInvoiceNumber	Invoice Number (DisplayInvoiceNumber column)

Returns

Type	Description
WSInvoice	Returns an ApiInvoice object. Access the object through the .Value property of WSInvoice.

Remarks

None

Example

Not available

Related information

"Invoice" on page 990

Invoice: GetInvoiceById

```
Public Function GetInvoiceById(ByVal sInvoiceId As String) As WSInvoice
```

Purpose

Retrieves an invoice record for InvoiceId provided.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice

Returns

Type	Description
WSInvoice	Returns an ApiInvoice object. Access the object through the .Value property of WSInvoice.

Remarks

Note this method determines the tax information to be returned based upon the value of the OLDTax field in the invoice (returned as the property OldTax). If OldTax is set to 1, then the tax information returned is the values in Tax1Total, etc.

If the value of OldTax is 0, then there could be many taxes. In this case, TaxCount will indicate the number of elements in each of the Tax Arrays: TaxCodeID, TaxCodeAmount, TaxCodeName and TaxCodeRate. These arrays will have no values if it is OldTax. If it is new tax, the fields Tax1Total, Tax2Total, Tax1TotalAdj and Tax2TotalAdj will have the values 0.

Example

Not available

Related information

"Invoice" on page 990

Invoice: GetInvoicedExpenses

```
Public Function GetInvoicedExpenses(ByVal sInvoiceId As String) As
WSInvoicedExpenseArray
```

Purpose

Retrieves all invoiced expenses on an invoice.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice.

Returns

Type	Description
WSInvoicedExpenseArray	Returns an array of ApiInvExpense objects. Access the array through the .Value property of WSInvoicedExpenseArray.

Remarks

For more information, see the "ApiInvExpense" section on page 329.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSInvoicedExpenseArray

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetInvoicedExpenses("{30CA19EB-37D5-11D4-8AB1-0001023D3221}")
```

Related information

"Invoice" on page 990

Invoice: GetInvoicedFixedFees

```
Public Function GetInvoicedFixedFees(ByVal sInvoiceId As String) As
WSInvoicedFixedFeeArray
```

Purpose

Retrieves all invoiced fixed fees on an invoice.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice.

Returns

Type	Description
WSInvoicedFixedFeeArray	Returns an array of ApiInvFixedFee objects. Access the array through the .Value property of WSInvoicedFixedFeeArray.

Remarks

For more information, see the "ApiInvFixedFee" section on page 330.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSInvoicedFixedFeeArray

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetInvoicedFixedFees("{30CA19EB-37D5-11D4-8AB1-0001023D3221}")
```

Related information

"Invoice" on page 990

Invoice: GetInvoicedProduct

```
Public Function GetInvoicedProduct(ByVal sInvoiceId As String) As
WSInvoicedProductArray
```

Purpose

Retrieves all invoiced products on an invoice.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice.

Returns

Type	Description
WSInvoicedProductArray	Returns an array of ApiInvProduct objects. Access the array through the .Value property of WSInvoicedProductArray.

Remarks

For more information, see the "ApiInvProduct" section on page 332.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSInvoicedProductArray

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetInvoicedProduct("{30CA19EB-37D5-11D4-8AB1-0001023D3221}")
```

Related information

"Invoice" on page 990

Invoice: GetInvoicedSupportTime

```
Public Function GetInvoicedSupportTime(ByVal sInvoiceId As String) As
WSInvoicedSupportTimeArray
```

Purpose

Retrieves all invoiced support or request time on an invoice.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice.

Returns

Type	Description
WSInvoicedSupportTimeArray	Returns an array of ApiInvSupportTime objects. Access the array through the .Value property of WSInvoicedSupportTimeArray.

Remarks

For more information, see the "ApiInvSupportTime" section on page 334.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSInvoicedSupportTimeArray

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetInvoicedSupportTime("{30CA19EB-37D5-11D4-8AB1-
0001023D3221}")
```

Related information

"Invoice" on page 990

Invoice: GetInvoicedTime

```
Public Function GetInvoicedTime(ByVal sInvoiceId As String) As
WSInvoicedTimeArray
```

Purpose

Retrieves all invoiced time on an invoice.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice.

Returns

Type	Description
WSInvoicedTimeArray	Returns an array of ApiInvTime objects. Access the array through the .Value property of WSInvoicedTimeArray.

Remarks

For more information, see the "ApiInvTime" section on page 335.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSInvoicedTimeArray

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetInvoicedTime("{30CA19EB-37D5-11D4-8AB1-0001023D3221}")
```

Related information

"Invoice" on page 990

Invoice: GetInvoices

```
Public Function GetInvoices(ByVal bBatched As Boolean, ByVal iStatus As
Integer, ByVal sCustomerId As String, ByVal engagementId As String) As
WSDataset
```

Purpose

Retrieves invoice records that are filtered by Parameters

Parameters

Parameter	Description
bBatched	True for specifying to retrieve invoices that already have been batched.
iStatus	Status of the invoice. See Remarks.
sCustomerId	Retrieves invoice objects by customer Id. Can be empty string.
engagementId	Retrieves invoice objects by Engagement Id. Can be empty string.

Returns

Type	Description
WSDataSet	WSDataset (InvoiceId, CustomerId, EngagementId, DisplayInvoiceId, Status) Access the dataset through the .Value property of WSDataSet.

Remarks

Use the numeric value as a parameter.

- cpInvDraft = 0
- cpInvPending_Approval = 1
- cpInvPending_SecondLevel_Approval = 2
- cpInvApproved = 3
- cpInvCommited = 4
- cpInvSent = 5
- cpInvPartialPaid = 6
- cpInvPaid = 7
- cpInvOverPayment = -1000000

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As New webInvoice.WSDataset
UserToken.SetToken(myProxy, MainForm.mUserName, MainForm.mPassword)
'Returns all draft invoices
oRet = myProxy.GetInvoices(False, 0, "", "")
```

Related information

"Invoice" on page 990

Invoice: GetPaymentInfo

```
Public Function GetPaymentInfo(ByVal sInvoiceId As String) As
WSPaymentsInfoArray
```

Purpose

Retrieves all payment information on an invoice.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice.

Returns

Type	Description
WSPaymentsInfoArray	Returns an array of ApiInvPaymentInfo objects. Access the array through the .Value property of WSPaymentsInfoArray.

Remarks

For more information, see the "ApiInvPaymentInfo" section on page 331.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSPaymentsInfoArray
UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetPaymentInfo("{30CA19EB-37D5-11D4-8AB1-0001023D3221}")
```

Related information

"Invoice" on page 990

Invoice: GetPaymentTotals

```
Public Function GetPaymentTotals(ByVal sInvoiceId As String) As WSDecimal
```

Purpose

Retrieves the total of all payments made to an invoice converting each payment to the invoice currency as necessary.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice.

Returns

Type	Description
WSDecimal	Returns total payment of an invoice. Access the amount through the .Value property of WSWSDecimal.

Remarks

None

Example

Not available

Related information

"Invoice" on page 990

Invoice: GetProductWriteOffs

```
Public Function GetProductWriteOffs(ByVal sInvoiceId As String) As
WSProductWriteOffArray
```

Purpose

Retrieves all products that were written off on an invoice.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice.

Returns

Type	Description
WSProductWriteOffArray	Returns an array of ApiInvWOProduct objects. Access the array through the .Value property of WSProductWriteOffArray.

Remarks

For more information, see the "ApiInvWOProduct" section on page 338.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSProductWriteOffArray

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetProductWriteOffs("{30CA19EB-37D5-11D4-8AB1-0001023D3221}")
```

Related information

"Invoice" on page 990

Invoice: GetSupportWriteOffs

```
Public Function GetSupportWriteOffs(ByVal sInvoiceId As String) As
WSSupportTimeWriteOffArray
```

Purpose

Retrieves all request time that were written off on an invoice.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice.

Returns

Type	Description
WSSupportTimeWriteOffArray	Returns an array of ApiInvWOSupport objects. Access the array through the .Value property of WSSupportTimeWriteOffArray.

Remarks

For more information, see the "ApiInvWOSupport" section on page 339.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSSupportTimeWriteOffArray

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetSupportWriteOffs("{30CA19EB-37D5-11D4-8AB1-0001023D3221}")
```

Related information

"Invoice" on page 990

Invoice: GetTimeWriteOffs

```
Public Function GetTimeWriteOffs(ByVal sInvoiceId As String) As
WSTimeWriteOffArray
```

Purpose

Retrieves all time that were written off on an invoice.

Parameters

Parameter	Description
sInvoiceId	ID of the invoice.

Returns

Type	Description
WSTimeWriteOffArray	Returns an array of ApiInvWOTime objects. Access the array through the .Value property of WSTimeWriteOffArray.

Remarks

For more information, see the "ApiInvWOTime" section on page 340.

Example

```
Dim myProxy As New webInvoice.InvoiceWse
Dim oRet As webInvoice.WSTimeWriteOffArray

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetTimeWriteOffs("{30CA19EB-37D5-11D4-8AB1-0001023D3221}")
```

Related information

"Invoice" on page 990

Invoice: MarkInvoiceAsBatched

```
Public Function MarkInvoiceAsBatched(ByVal sInvoiceId As String) As WSInt32
```

Purpose

Mark the invoice identified by this ID as batched.

Parameters

Parameter	Description
sInvoiceId	Invoice ID

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The error returns are written to the log file with messages.

The extract date is set to the current date.

The invoice can be retrieved again using GetInvoiceById or through GetInvoices and specifying the batched parameter as true.

Example

Not available

Related information

"Invoice" on page 990

"Invoice: GetInvoiceById" on page 999

"Invoice: GetInvoices" on page 1005

KnowledgeManagement

The KnowledgeManagement object allows users to retrieve, create, edit or delete knowledge items in the Changepoint database. Related to the KnowledgeManagement object is the following object:

Namespace

`http://changepoint.com/changepoint/CPWebService/ KnowledgeManagement`

URL

`http://webserver/CPWebService/Objects/CPKnowledgeManagement /KnowledgeManagement.asmx`

Methods

Properties

For more information, see the "ApiKnowledgeManagement" section on page 342.

Related information

"KMVersion" on page 1033

KnowledgeManagement: CreateKM - for WSE only

```
Public Function CreateKM(ByVal KM As ApiKnowledgeManagement) As WSInt32
```

Purpose

Create a new knowledge item in Changepoint

Parameters

Parameter	Description
KM	ApiKnowledgeManagement Object

Returns

Type	Description
WSInt32.	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The attachment is uploaded by DIME.

Example

```
Dim myKMItem as New WebKnowledgeManagement.ApiKnowledgeManagement
Dim proxy As New WebKnowledgeManagement.KnowledgeManagementWse
Dim iRet as Int32 = 0
Dim Id as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
Id = proxy.GetUniqueId
With myKMItem
   .AllowViewName.SetValue( "John Smith", 0)
   .AllowViewName.SetValue("Jane Doe", 1)
   .AttachmentId = Id
   .Author = "Mel Torn"
   .CheckedOut = False
   .CustomerId = "{f6c77d82-3ada-11d4-81d9-00104b7ad9cd }"
   .Title = "MyTestKMItem"
   .FileName = "MyTestKMItem.Doc"
   ...
```

```
End With
iRet = proxy.CreateKM(myKMItem).Value
If iRet <> 0 Then
...'Handle error
End If
```

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagementUploadDownload: CreateKM - for WCF only

```
Public Function CreateKM(Username As String, km As ApiKnowledgeManagement,
Attachment As Stream) As WSInt32
```

Purpose

Create a new knowledge item in Changepoint.

Parameters

Parameter	Description
Username	Username of logged-in user
KM	ApiKnowledgeManagement Object - filled in
Attachment	Item to attach as a Stream type. **Note:** The .Attachment property of the ApiKnowledgeManagement object must be empty.

Returns

Type	Description
WSInt32.	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

KnowledgeManagement: DeleteKMById

```
Public Function DeleteKMById(ByVal sKMId As String) As WSInt32
```

Purpose

Delete a knowledge item from Changepoint

Parameters

Parameter	Description
sKMId	KM ID of the item to delete

Returns

Type	Description
WSInt32.	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

Not available

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagement: DeleteKMByName

```
Public Function DeleteKMByName(ByVal KMName As String) As WSInt32
```

Purpose

Delete a knowledge item based on the name of the item.

Parameters

Parameter	Description
KMName	Title of the knowledge item

Returns

Type	Description
WSInt32.	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

Not available

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagement: GetCPLinkList

```
Public Function GetCPLinkList(ByVal sRefType As String) As WSDataSet
```

Purpose

Returns a list of related Changepoint entities based on the ReferenceType

Parameters

Parameter	Description
sRefType	Supported types are: Project, Task, Customer, Contact, Campaign, Competitor, Opportunity, Activity, Engagement, Resource, Product, RequestScenario, ProjPortfolioScenario, ResDemandScenario, CandidateScenario

Returns

Type	Description
WSDataSet	WSDataset (Id, Name, AlternateName) or WSDataset (Id, Name) Access the returned dataset through the .Value property of WSDataSet.

Remarks

Columns returned in the dataset depend on the reference type as follows:

- Project, Customer, Engagement, Resource return: (Id, Name, AlternateName)
- All other types return: (Id, Name)

Example

Not available

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagement: GetKMById - for WSE only

```
Public Function GetKMById(ByVal KMId As String) As WSKnowledgeManagement
```

Purpose

Retrieve a knowledge item based on the ID of the item

Parameters

Parameter	Description
KMId	Knowledge item ID as a GUID string

Returns

Type	Description
WSKnowledgeManagement	Returns an ApiKnowledgeManagement object. Access the object through the .Value property of WSKnowledgeManagement.

Remarks

None

Example

Not available

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagementUploadDownload: GetKMById - for WCF only

```
Public Function GetKMById(KMId As String, Username As String, out Data As
Stream) As KnowledgeManagementWSKnowledgeManagement
```

Purpose

Retrieve a knowledge item based on the ID of the item

Parameters

Parameter	Description
KMId	Knowledge item ID
Username	Username of logged-in User
Data	Attachment as an output Stream. **Note:** the .Attachment property of the ApiKnowledgeManagement object is set to null.

Returns

Type	Description
KnowledgeManagementWSKnowledgeManagement	Returns an ApiKnowledgeManagement object. Access the object through the .Value property of KnowledgeManagementWSKnowledgeManagement.

KnowledgeManagement: GetKMByName - for WSE only

```
Public Function GetKMByName(ByVal KMName As String) As WSKnowledgeManagement
```

Purpose

Retrieve knowledge item by the Title of the item.

Parameters

Parameter	Description
KMName	KM Title

Returns

Type	Description
WSKnowledgeManagement	Returns an ApiKnowledgeManagement object. Access the object through the .Value property of WSKnowledgeManagement.

Remarks

It is possible that there may be more than one Knowledge Management item with the same name. The method returns the first knowledge item in the list and ignores any others.

Example

Not available

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagementUploadDownload: GetKMByName - for WCF only

```
Public Function GetKMByName(KMName As String, Username As String, out Data As
Stream) As KnowledgeManagementWSKnowledgeManagement
```

Purpose

Retrieve knowledge item by the Title of the item.

Parameters

Parameter	Description
KMName	KM Title
Username	Username of logged-in User
Data	Attachment as an output Stream. **Note:** the .Attachment property of the ApiKnowledgeManagement object is set to null.

Returns

Type	Description
KnowledgeManagementWSKnowledgeManagement	Returns an ApiKnowledgeManagement object. Access the object through the .Value property of KnowledgeManagementWSKnowledgeManagement.

KnowledgeManagement: GetKMCategories

```
Public Function GetKMCategories() As WSDataSet
```

Purpose

Returns a list of KM categories

Parameters

None

Returns

Type	Description
WSDataSet	WSDataSet (CategoryId, Name) Access the DataSet through the .Value property of WSDataSet.

Remarks

Retrieves all categories, whether or not they are subscribed to by the current logged in resource.

Example

Not available

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagement: GetKMCodeFields - for WSE only

```
Public Function GetKMCodeFields(ByVal ResourceId As String) As WSCodes
```

Purpose

Returns the set of usable code fields and their allowable options based on the specified ResourceId

Parameters

Parameter	Description
ResourceId	The logged in resource or related resource to the knowledge item

Returns

Type	Description
WSCodes	A two dimensional array containing code fields and their options. Access the returned CPCodes array through the .Value property of WSCodes.

Remarks

The ResourceId parameter filters the returned values to the code fields which the resource has access to.

Example

Not available

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagement: GetKMCodeFields - for WCF only

```
Public Function GetKMCodeFields(ByVal ResourceId As String) As WSCodes2
```

Purpose

Returns the set of usable code fields and their allowable options based on the specified ResourceId

Parameters

Parameter	Description
ResourceId	The logged in resource or related resource to the knowledge item

Returns

Type	Description
WSCodes2	A two dimensional array containing code fields and their options. Access the returned CPCodes array through the .Value property of WSCodes2.

Remarks

WSCodes2 is a new return type to replace the WSCodes for WCF API.

KnowledgeManagement: GetKMs

```
Public Function GetKMs(ByVal SearchString As String, ByVal KMCategory As
String, ByVal KMSubCategory As String, ByVal KMCode1 As String, ByVal KMCode2
As String, ByVal KMCode3 As String) As WSDataSet
```

Purpose

Retrieve a list of knowledge items based on specified filter criteria.

Parameters

Parameter	Description
sSearchString	The title search string. If the title search string is empty, returns all
KMCategory	Category Id. An empty string is allowed.
KMSubCategory	Subcategory Id. An empty string is allowed.
KMCode1	Code1 Id. An empty string is allowed.
KMCode2	Code2 Id. An empty string is allowed.
KMCode3	Code3 Id. An empty string is allowed.

Returns

Type	Description
WSDataSet	WSDataSet (AttachmentId, Title, Category, SubCategory, Rank) Access the DataSet through the .Value property of WSDataSet.

Remarks

Extracts knowledge items based on the Parameters but also limits the returned items to those that the logged in resource has access to.

Example

Not available

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagement: GetKMSubCategories

```
Public Function GetKMSubCategories(ByVal KMCategory As String) As WSDataSet
```

Purpose

Returns a list of KM subcategories based on the KM Category and logged in user

Parameters

Parameter	Description
KMCategory	Category whose subcategories are being requested.

Returns

Type	Description
WSDataSet	WSDataSet (Code, Description) Access the DataSet through the .Value property of WSDataSet.

Remarks

The returned list is filtered based on KMCategory and includes all subcategories that are not subscribed to by the logged in resource and all those that are subscribed to by the logged in resource.

Example

Not available

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagement: GetTextFields - for WSE only

```
Public Function GetTextFields(ByVal ResourceId As String) As WSTexts
```

Purpose

Returns a list of code and/or text fields along with their allowable options

Parameters

Parameter	Description
ResourceId	The logged in resource or related resource to the knowledge item

Returns

Type	Description
WSTexts	A collection of CPTexts objects containing text fields. Access the collection through the .Value property of WSTexts.

Remarks

None

Example

Not available

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagement: GetTextFields - for WCF only

```
Public Function GetTextFields(ByVal ResourceId As String) As WSTexts2
```

Purpose

Returns a list of code and/or text fields along with their allowable options

Parameters

Parameter	Description
ResourceId	The logged in resource or related resource to the knowledge item

Returns

Type	Description
WSTexts2	A collection of CPTexts objects containing text fields. Access the collection through the .Value property of WSTexts2.

Remarks

WSTexts2 is a new return type to replace WSTexts for the WCF API.

KnowledgeManagement: GetResourcesForKM

```
Public Function GetResourcesForKM(ByVal ResourceId As String) As WSDataSet
```

Purpose

Returns a list of preferred resources based on a specified resource.

Parameters

Parameter	Description
ResourceId	The logged in resource or resource related to the knowledge item.

Returns

Type	Description
WSDataSet	WSDataSet (Name, ResourceId, AlternateName) Access the DataSet through the .Value property of WSDataSet.

Remarks

The returned list is determined by ResourceId being a member of a Workgroup and having a list of preferred resources defined in Changepoint.

Example

Not available

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagement: GetWorkgroupForKM

```
Public Function GetWorkgroupForKM(ByVal ResourceId As String) As WSDataSet
```

Purpose

Returns a list of Workgroups based on a specified resource.

Parameters

Parameter	Description
ResourceId	The logged in resource or related resource to the knowledge item

Returns

Type	Description
WSDataSet	WSDataSet (WorkgroupId, Name) Access the DataSet through the .Value property of WSDataSet.

Remarks

The returned recordset is limited to the Workgroups to which ResourceId is a current member.

Example

Not available

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagement: UpdateKMById - for WSE only

```
Public Function UpdateKMById(ByVal KMId As String, ByVal KM As
ApiKnowledgeManagement) As WSInt32
```

Purpose

Update Changepoint with existing KM data

Parameters

Parameter	Description
KMId	ID of the KM Item
KM	The ApiKnowledgeManagement Object whose data is to be updated in the database

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The KMId is used to identify the knowledge item to update.

Example

```
Dim myKMItem as New WebKnowledgeManagement.ApiKnowledgeManagement
Dim proxy As New WebKnowledgeManagement.KnowledgeManagementWse
Dim iRet as Int32 = 0
Dim sAttId as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
myKMItem = proxy.GetKMById(KMId).value
sAttId = myKMItem.AttachmentId
If sAttId <> "" Then
With myKMItem
   .AllowViewName( "Brian Jones", 2)
   .Title = "MyModifiedTestKMItem"
   .Attachment = myUpdatedAttachment
   .ExpiryDate = DateAdd("mmm", 6, Date)
   ...
End With
iRet = proxy.UpdateKMById(sAttId , myKMItem).Value
If iRet <> 0 Then
   ...'Handle error
End If
```

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagementUploadDownload: UpdateKMById - for WCF only

```
Public Function UpdateKMById(Username As String, km As ApiKnowledgeManagement,
KMId As String, Attachment As Stream) As WSInt32
```

Purpose

Update Changepoint with existing KM data

Parameters

Parameter	Description
Username	Username of logged-in User
KM	The ApiKnowledgeManagement Object whose data is to be updated in the database

Parameter	Description
KMId	ID of the KM Item
Attachment	Item to attach as a Stream type. **Note:** the .Attachment property of the ApiKnowledgeManagement object must be empty.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

KnowledgeManagement: UpdateKMByName - for WSE only

```
Public Function UpdateKMByName(ByVal KMName As String, ByVal KM As
ApiKnowledgeManagement) As WSInt32
```

Purpose

Update an existing knowledge item in Changepoint

Parameters

Parameter	Description
KMName	Title of the knowledge item to update
KM	ApiKnowledgeManagement Object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

KMName is used to identify the knowledge item to update.

Example

```
Dim myKMItem as New WebKnowledgeManagement.ApiKnowledgeManagement
Dim proxy As New WebKnowledgeManagement.KnowledgeManagementWse
Dim iRet as Int32 = 0
Dim sAttId as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
myKMItem = proxy.GetKMByName("MyTestKMItem").value
sAttId = myKMItem.AttachmentId
If sAttId <> "" Then
With myKMItem
   .AllowViewName( "Brian Jones", 2)
   .Title = "MyModifiedTestKMItem"
   .Attachment = myUpdatedAttachment
   .ExpiryDate = DateAdd("mmm", 6, Date)
   ...
End With
iRet = proxy.UpdateKMByName("MyTestKMItem" , myKMItem).Value
If iRet <> 0 Then
   ...'Handle error
End If
```

Related information

"KnowledgeManagement" on page 1012

KnowledgeManagementUploadDownload: UpdateKMByName - for WCF only

```
Public Function UpdateKMByName(Username As String, km As
ApiKnowledgeManagement, KMName As String, Attachment As Stream) As WSInt32
```

Purpose

Update an existing knowledge item in Changepoint

Parameters

Parameter	Description
Username	Username of logged-in User
KM	The ApiKnowledgeManagement object whose data is to be updated in the database
KMName	Title of the KM Item
Attachment	Item to attach as a Stream type. **Note:** the .Attachment property of the ApiKnowledgeManagement object must be empty.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

KMVersion

The KMVersion object represents version information for knowledge items.

Namespace

`http://changepoint.com/changepoint/CPKnowledgeManagement`

URL

`http://webserver/CPWebService/Objects/CPKnowledgeManagement/KMVersion.asmx`

Methods

Properties

For more information, see the "ApiKMVersion" section on page 359.

Related information

"KnowledgeManagement" on page 1012

KMVersion: GetVersionControl

```
Public Function GetVersionControl(ByVal sAttachmentId As String) As WSDataSet
```

Purpose

Retrieve version control information of Knowledge Item.

Parameters

Parameter	Description
sAttachmentId	Attachment ID

Returns

Type	Description
WSDataSet	DataSet (version, CheckedoutBy, CheckedoutOn, CheckedFlag, comment, CheckedControl, VersionControl, versionlimit) Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"KMVersion" on page 1033

KMVersion: KMDeleteBatchVersionItems

```
Public Function KMDeleteBatchVersionItems(ByVal sAttachmentId As String, ByVal
sCategory As String, ByVal bDeleteAllFlag As Boolean) As WSInt32
```

Purpose

Delete history versions of knowledge items.

Parameters

Parameter	Description
sAttachmentId	The Attachment Id. Can be empty.
sCategory	The Category Id. Can be empty.
bDeleteAllFlag	The flag if delete all version.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

If bDeleteAllFlag is true, delete all versions of knowledge item by attachment ID or Category Id; otherwise delete specific versions of knowledge item.

Example

Not available

Related information

"KMVersion" on page 1033

KMVersion: KMDeleteVersionItem

```
Public Function KMDeleteVersionItem(ByVal sAttachmentId As String, ByVal
sVersionid As String) As WSInt32
```

Purpose

Delete a history version of knowledge item.

Parameters

Parameter	Description
sAttachmentId	Attachment ID.
sVersionid	Version ID.

Returns

Type	Description
WSInt32.	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

Not available

Related information

"KMVersion" on page 1033

KMVersion: KMResetCheckedout

```
Public Function KMResetCheckedout(ByVal sAttachmentId As String) As WSInt32
```

Purpose

Changes checkout flag to false if a person who checked out this knowledge item no longer has the edit permission for this knowledge item.

Parameters

Parameter	Description
sAttachmentId	Attachment ID.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Reset checkout to 0 if user has been removed from edit permission list. This function will make sure checkout flag is reset when the user's permission is changed to read-only.

In KM version control, Edit Knowledge feature is required in order to check out a knowledge item. Once a knowledge item is checked out, other users can't edit it.

Users with the Override Check-Out feature can cancel the check out by selecting Check In from the menu in the Enterprise.

Example

Not available

Related information

"KMVersion" on page 1033

KMVersion: KMUpdateVersion - for WSE only

```
Public Function KMUpdateVersion(ByVal mKM As ApiKMVersion, ByVal AttachUpdate
As Boolean) As WSInt32
```

Purpose

Adds a new version history of knowledge item by updating current knowledge item.

Parameters

Parameter	Description
mKM	ApiKMVersion object.
AttachUpdate	The flag if only attachment file has been changed.

Returns

Type	Description
WSInt32.	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

AttachUpdate must set to False so that the current version of knowledge item can be updated as well when creating a history version of knowledge item.

If AttachUpdate is set to True, it adds a new version of a knowledge item to KMVersion table and updates only version number, updatedby, updatedon of the current knowledge item in the KMAttachment.

It is recommended to use the GetVersionControl method to check if this knowledge item can have version control ability before using this method. In order to add knowledge item version, version control must be enabled for the knowledge item and the knowledge item must not be checked out.

Example

```
Dim myKMVersion As New WebKMVersion.ApiKMVersion
Dim proxy As New WebKMVersion.KMVersionWse
Dim iRet As Int32

UserToken.SetToken(proxy, mUserName, mPassword)

With myKMVersion
```

```
    .AttachmentId = "{75606817-003F-491F-AEA9-4B056B0BFEFB}"
    .Title = "ITG Data Relationships"
    .Description = "ITG Data Relationships"
    .ProjectId = "{00000000-0000-0000-0000-000000000000}"
    .Category = "{B11C9334-061E-478E-9378-6E1C79C1E364}"
    .SubCategory = "{4B7A0932-D646-40BE-B902-D7DA8E12C7C9}"
    .Keywords = "ITG; Data Relationships"
    .IsShared = True
    .ExpiryDate = "05/18/2007"
    ...
End With

iRet = proxy.KMUpdateVersion(myKMVersion, False).Value
...
```

Related information

"KMVersion" on page 1033

KMVersionUploadDownload: KMUpdateVersion - for WCF only

```
Public Function KMUpdateVersion(AttachUpdate As Boolean, UserName As String,
mKM As ApiKMVersion) As WSInt32
```

Purpose

Adds a new version history of knowledge item by updating current knowledge item.

Parameters

Parameter	Description
AttachUpdate	The flag if only attachment file has been changed.
Username	Username of logged-in User
mKM	ApiKMVersion object.

Returns

Type	Description
WSInt32.	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

KMVersion: LoadKMArticleHist - for WSE only

```
Public Function LoadKMArticleHist(ByVal sVerId As String) As WSKMVersion
```

Purpose

Retrieve knowledge item by version ID.

Parameters

Parameter	Description
sVerId	Version ID

Returns

Type	Description
WSKMVersion	Returns a ApiKMVersion object. Access the object through the .Value property of WSKMVersion.

Remarks

None

Example

```
Dim mKMVersion As New CPKMVersion
Dim proxy As New WebKMVersion.KMVersionWse

UserToken.SetToken(proxy, mUserName, mPassword)

mKMVersion = proxy.LoadKMArticleHist("{2EE84D3E-AB3F-45C0-AB9F-
86EB94DB5F09}").MyKMVersion
...'process data
```

Related information

"KMVersion" on page 1033

KMVersionUploadDownload: LoadKMArticleHist - for WCF only

```
Public Function LoadKMArticleHist(UserName As String, sVerId As String, out
Data As Stream) As KMVersionWSKMVersion
```

Purpose

Retrieve knowledge item by version ID.

Parameters

Parameter	Description
Username	Username of logged-in User
Data	Stream output from the attachment for the KMVersion object.
sVerId	Version ID

Returns

Type	Description
KMVersionWSKMVersion	Returns a ApiKMVersion object. Access the object through the .Value property of KMVersionWSKMVersion.

Remarks

The Attachment property is set to null. The attachment is loaded to the Data parameter as a stream in the method call.

LookupIds

LookupIds provides some common functions to retrieve information from the Changepoint database.

Namespace

```
http://changepoint.com/changepoint/CPWebService/LookupIds
```

URL

`http://webserver/CPWebService/Objects/CPLookupIds/LookupIds.asmx`

Methods

Properties

Property	Type	Description
CPConnection	ApiConnection	Write-only. The Connection object which must be assigned before any action to database.
CPException	ApiException	Read-only. The ApiException object which holds the exception information.

LookupIds: DeleteUDF

```
Public Function DeleteUDF(ByVal entity As CPEntity, ByVal entityId As String,
ByVal actionResourceId As String) As WSInt32
```

Purpose

Delete all UDFs (configurable fields) associated with the entity ID.

Parameters

Parameter	Description
entity	CPEntity, for example, CPEntity.Customer = 2.
entityId	EntityId for the UDF, for example, CustomerId for the Customer entity.
actionResourceId	Resource ID of the caller. The default is the login user ID.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

```
Dim proxy As New webLookupIds.LookupIdsWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webLookupIds.WSInt32 = proxy.DeleteUDF(CPEntity.Resource, sId, "")
```

Related information

"LookupIds" on page 1041

LookupIds: GetBillingOffices

```
Public Function GetBillingOffices(ByVal sBillingOfficeId As String) As
WSDataSet
```

Purpose

Retrieve the billing office ID and description for billing offices.

Parameters

Parameter	Description
sBillingOfficeId	ID of the billing office to be retrieved

Returns

Type	Description
WSDataset	WSDataset (BillingOfficeId, Description) `Access the dataset through the .Value property of WSDataSet.`

Remarks

If there is empty value for sBillingOfficeId, the function returns all billing offices; otherwise, only returns the specified billing office.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

```
Dim proxy As New webLookupIds.LookupIdsWse()
   'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As webLookupIds.WSDataset = proxy.GetBillingOfficeId("")
```

Related information

"LookupIds" on page 1041

LookupIds: GetCertificationCycle

```
Public Function GetCertificationCycle() As WSDataSet
```

Purpose

Retrieve the certification cycle.

Parameters

None

Returns

Type	Description
WSDataset	WSDataset `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

```
Dim proxy As New webLookupIds.LookupIdsWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As webLookupIds.WSDataset = proxy.GetCertificationCycle()
```

Related information

"LookupIds" on page 1041

LookupIds: GetCodeByDescription

```
Public Function GetCodeByDescription(ByVal Description As String, ByVal
TableName As String) As WSString
```

Purpose

Retrieve the Changepoint code of the property described by the description.

Parameters

Parameter	Description
Description	Code description.
TableName	Table name to lookup.

Returns

Type	Description
WSString	Changepoint code that corresponds to the specified description. Access the returned string through the .Value property of WSString

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetCodeDetail

```
Public Function GetCodeDetail(ByVal CodeType As String) As WSDataSet
```

Purpose

Retrieve Changepoint code details and description for the specified code type from the CodeDetail table.

Parameters

Parameter	Description
CodeType	Code type.

Returns

Type	Description
WSDataset	WSDataset of Changepoint code details and descriptions for the specified code type from the CodeDetail table. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetCountryCode

```
Public Function GetCountryCode() As WSDataSet
```

Purpose

Retrieve Country Code and Description.

Parameters

None

Returns

Type	Description
WSDataset	WSDataset of country codes and descriptions. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetCurrencyCodes

```
Public Function GetCurrencyCodes(ByVal CurrencyDescription As String) As
WSDataSet
```

Purpose

Retrieve currency codes.

Parameters

Parameter	Description
CurrencyDescription	Currency description, for example, "Canadian dollar"

Returns

Type	Description
WSDataset	WSDataset of currency codes and descriptions. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetFeatures

```
Public Function GetFeatures(ByVal RoleId As String) As WSDataset
```

Purpose

Retrieve the feature code and feature name for the specified role.

Parameters

Parameter	Description
RoleId	Role ID

Returns

Type	Description
WSDataset	WSDataset of the feature codes and feature names for the specified role. Access the dataset through the .Value property of WSDataSet.

Remarks

If RoleId string is empty, the function returns all features.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetGlobalWorkgroupIds

```
Public Function GetGlobalWorkgroupIds(ByVal WorkgroupId As String) As
WSDataset
```

Purpose

Retrieve the global workgroup ID and name.

Parameters

Parameter	Description
WorkgroupId	Workgroup ID

Returns

Type	Description
WSDataset	WSDataset of global workgroup IDs and names. `Access the dataset through the .Value property of WSDataSet.`

Remarks

If WorkgroupId string is empty, returns all GlobalWorkgroups.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetHelpDesks

```
Public Function GetHelpDesks() As WSDataSet
```

Purpose

Retrieve all help desk ID and descriptions.

Parameters

None

Returns

Type	Description
WSDataset	WSDataset of help desk IDs and descriptions. Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal iEntity As CPEntity, ByVal sUDFField As
String, ByVal sUDFValue As String) As WSString
```

Purpose

Returns the ID for the specified entity based on the specified UDF field and value.

Parameters

Parameter	Description
iEntity	Entity for which the ID is being looked up
sUDFField	UDF text field name (for example, Text1)
sUDFValue	UDF text value

Returns

Type	Description
WSString	String with the ID of the entity. `Access the returned string through the .Value property of WSString.`

Remarks

The method will throw error -10 (The records are not found) when no records are returned based on the search criteria.

The method will throw error -11 (multiple records found) when duplicates exist for the lookup.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetLookupCodes

```
Public Function GetLookupCodes(ByVal sTableName As String, ByVal
sResourceGWkgpId As String, ByVal sResourceWkgpId As String) As WSDataSet
```

Purpose

Returns the ID and description for all the specified lookup code values.

Parameters

Parameter	Description
sTableName	Name of the table containing the lookup code values (for example, "contacttype")
sResourceGWkgpId	Global Workgroup ID of the caller.
sResourceWkgpId	Workgroup ID of the caller.

Returns

Type	Description
WSDataset	WSDataset of IDs and descriptions for the specified lookup code values. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetLookupFields

```
Public Function GetLookupFields(ByVal sTableName As String, ByVal sRetColumn1
As String, ByVal sRetColumn2 As String, ByVal sRetColumn3 As String, ByVal
sFilter As String) As WSDataset
```

Purpose

Generate a simple SQL statement to access the database.

Parameters

Parameter	Description
sTableName	Database table name.
sRetColumn1	First returned column name for the table.
sRetColumn2	Second returned column name for the table
sRetColumn3	Third returned column name for the table
sFilter	Filter criteria string, usually the string after the key word "WHERE", for example, "Deleted = 0 AND FirstName = 'Richard'"

Returns

Type	Description
WSDataset	WSDataset of the records. Access the dataset through the .Value property of WSDataSet.

Remarks

The first two parameters cannot be empty.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetNameById

```
Public Function GetNameById(ByVal iEntity As CPEntity, ByVal sId As String) As
WSIdentity
```

Purpose

Retrieves the entity name and alternate name from the Entity table for the specified Changepoint entity.

Parameters

Parameter	Description
iEntity	CPEntity, for example, CPEntity.Customer = 2.
sId	Entity ID for lookup.

Returns

Type	Description
WSIdentity	Identity information for the specified Changepoint entity. Access the identity information through the .Value property of WSIdentity.

Remarks

The function should only be used to retrieve the name from the Entity table.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetPayrollCycle

```
Public Function GetPayrollCycle() As WSDataSet
```

Purpose

Retrieve the payroll cycle ID and description values.

Parameters

None

Returns

Type	Description
WSDataset	WSDataset of payroll cycle IDs and descriptions. Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetProvincesByCountry

```
Public Function GetProvincesByCountry(ByVal CountryCode As String) As
WSDataSet
```

Purpose

Returns the province ID and name fields from the Province table for the specified country code.

Parameters

Parameter	Description
CountryCode	Country code, for example, "CAN"

Returns

Type	Description
WSDataset	WSDataset of province IDs and names. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetRoles

```
Public Function GetRoles() As WSDataset
```

Purpose

Retrieve all the role IDs and names.

Parameters

None

Returns

Type	Description
WSDataset	WSDataset of role IDs and names. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetTimeZones

```
Public Function GetTimeZones() As WSDataSet
```

Purpose

Retrieve all time zones.

Parameters

None

Returns

Type	Description
WSDataset	WSDataset of time zones. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetUDF

```
Public Function GetUDF(ByVal entity As CPEntity, ByVal retOption As
CPUDFReturnType, ByVal entityId As String, ByVal additionalId As String, ByVal
actionResourceId As String) As WSString
```

Purpose

Retrieve UDFs (configurable fields) for each Changepoint entity.

Parameters

Parameter	Description
entity	CPEntity, for example, CPEntity.Customer = 2.
retOption	The CPUDFReturnType Values are: • WithStructure = 0 – returns UDF field data with full field structure • OnlyValues = 1 – returns only UDF fields with data, does not include any field structure or options • OnlyStructure = 2 – returns only UDF XML structure, no field data • OnlyStructureAndOptions = 3 – returns UDF structure and option data without field data • WithStructureAndOptions = 4 – returns UDF field data with full structure and all field options (except entity-based and conditional codes)
entityId	Entity ID for the UDF, for example, CustomerId for Customer entity.
additionalId	AdditionalId for the UDF. • For Engagement: it is BillingOfficeId; • For Request: it is RequestType. • It is not needed for other Entities.
actionResourceId	The Resource ID of the user that called the method. Default is the login user ID.

Returns

Type	Description
WSString	WSDataset of UDF information. `Access the dataset through the .Value property of WSDataSet.`

Remarks

The function is designed for retrieving UDF information for each Changepoint entity. It is recommended that you use this function to get UDFs.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

Example

```
Dim proxy As New webLookupIds.LookupIdsWse()
'set the SOAP header with UsernameToken
Dim customerId as String="{…}"
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As webLookupIds.WSDataset = proxy.GetUDF(2, 1, customerId)
```

Related information

"LookupIds" on page 1041

LookupIds: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal entity As CPEntity, ByVal codeName As
String, ByVal filterFieldsXML As String, ByVal searchString As String, ByVal
entityId As String, ByVal additionalId As String, ByVal actionResourceId As
String) As WSString
```

Purpose

Retrieve UDF by provided UDF XML.

Parameters

Parameter	Description
entity	CPEntity, for example, CPEntity.Customer = 2.
codeName	Code name of UDF, for example, "Code3"
filterFieldsXML	Filter fields XML string.
searchString	The function will return the options that start with this string (unless no string is entered).
entityId	EntityId for the UDF, for example, CustomerId for the Customer entity.
additionalId	AdditionalId for the UDF. • For Engagement: it is BillingOfficeId • For Request: it is RequestType • It is not needed for other Entities
actionResourceId	User ID of the user that called the method. The default is the login user ID.

Returns

Type	Description
WSString	XML string of UDF code options. Access the returned string through the .Value property of WSString.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Get the filterFieldsXML string by using the GetUDFFilterFields() function and populate it with current values.

Example

Example for options string: (CodeName = Code1)

```
<root>
<cpcode name="Code1">
<codeitem text="0" value="24b133bb-758a-42c2-82c6-3894a125b0ad" selected="0"
/>
<codeitem text="1" value="66da0e78-1c34-4bfd-b3be-d5c4fb8f430e" selected="0"
/>
<codeitem text="2" value="61d97a33-96a4-4692-a13d-d1ee9ac2aa70" selected="0"
/>
<codeitem text="3" value="db050f1b-f041-446a-a49c-ea38eb9f888b" selected="0"
/>
</cpcode>
</root>
Dim proxy As New webLookupIds.LookupIdsWse()
'set the SOAP header with UsernameToken
Dim customerId as String="{…}"
UserToken.SetToken(proxy, mUserName, mPassword)
'retrieve filter fields template
Dim oRet As webLookupIds.WSString = proxy.GetUDFFilterFields(2, "Code3", "",
"")
If Not oRet.WSException.HaveErrors then
   Dim sFilter As String = oRet.Value
   If sFilter <> "<root/>" then
'this item is conditional item, populate filter fields with current values
...
Else
'this item is NOT conditional item, do not need to populate
End If
   oRet=proxy.GetUDFCodeOptions(2, "Code3", sFilter, "", "", "", "")
End If
```

Related information

"LookupIds" on page 1041

"LookupIds: GetUDFFilterFields" on page 1063

LookupIds: GetUDFFilterFields

```
Public Function GetUDFFilterFields(ByVal entity As CPEntity, ByVal codeName As
String, ByVal additionalId As String, ByVal actionResourceId As String) As
WSString
```

Purpose

Retrieve filter fields for the specified code and entity.

Parameters

Parameter	Description
entity	CPEntity, for example, CPEntity.Customer = 2.
codeName	Code name of UDF, for example, "Code3"
additionalId	AdditionalId for the UDF. • For Engagement: it is BillingOfficeId • For Request: it is RequestType • It is not needed for other Entities
actionResourceId	The user ID of the user that called the method. Default is the login user ID.

Returns

Type	Description
WSString	XML string of the UDF conditional filter fields. Access the returned string through the .Value property of WSString.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

This function is only used to get UDF conditional filter fields, which are required for the function `GetUDFCodeOptions()` that retrieves specified UDF code options.

The filterFieldsXML string will be equal to "</root>" if this is not a conditional field. For a conditional field, the function just returns the filter fields schema. The function must be populated with current values before passing it to get UDF options.

Example

Returns example: (Entity = Project)

```
<root>
<entity value="Project"><id/><name/>
<field><name>CPProjectType</name><id/><value/></field>
<field><name>Customer</name><id /><value /></field>
```

```
<field><name>Engagement</name><id /><value /></field>
<field><name>ProjMgr</name><id /><value /></field>
<field><name>ProposedPhase</name><id/><value /></field>
<field><name>Status</name><id /><value /></field>
</entity>
</root>
Dim proxy As New webLookupIds.LookupIdsWse()
'set the SOAP header with UsernameToken
Dim customerId as String="{...}"
UserToken.SetToken(proxy, mUserName, mPassword)
'retrieve filter fields template
Dim oRet As webLookupIds.WSString = proxy.GetUDFFilterFields(2, "Code3", "",
"")
If Not oRet.WSException.HaveErrors then
   Dim sFilter As String = oRet.Value
   If sFilter <> "<root/>" then
'this item is conditional item, populate filter fields with current values
...
Else
'this item is NOT conditional item, do not need to populate
End If
   oRet=proxy.GetUDFCodeOptions(2, "Code3", sFilter, "", "", "", "")
End If
```

Related information

"LookupIds" on page 1041

"LookupIds: GetUDFCodeOptions" on page 1061

LookupIds: GetWorkCodeCategories

```
Public Function GetWorkCodeCategories() As WSDataSet
```

Purpose

Retrieve the work code category list.

Parameters

None

Returns

Type	Description
WSDataset	WSDataset of work code categories. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetWorkCodesByCategory

```
Public Function GetWorkCodesByCategory(ByVal sWorkCodeCategoryId As String) As
WSDataSet
```

Purpose

Retrieve the list of work codes associated with the specified work code category.

Parameters

Parameter	Description
sWorkCodeCategoryId	ID of the WorkCode category

Returns

Type	Description
WSDataset	WSDataset of work codes. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetWorkgroupIdByName

```
Public Function GetWorkgroupIdByName(ByVal WorkgroupName As String) As
WSDataset
```

Purpose

Retrieve the workgroup list associated with the specified workgroup name.

Parameters

Parameter	Description
WorkgroupName	Workgroup name.

Returns

Type	Description
WSDataset	WSDataset of workgroups. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetWorkLocationGroups

```
Public Function GetWorkLocationGroups() As WSDataSet
```

Purpose

Retrieve the work location group list.

Parameters

None

Returns

Type	Description
WSDataset	WSDataset of work location groups. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetWorkLocationsByGroupId

```
Public Function GetWorkLocationsByGroupId(ByVal sWorkLocationGroupId As
String) As WSDataSet
```

Purpose

Retrieve the work location list associated with the specified work location ID.

Parameters

Parameter	Description
sWorkLocationGroupId	WorkLocation Group ID

Returns

Type	Description
WSDataset	WSDataset of work locations. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: GetWorkLocationsByGroupName

```
Public Function GetWorkLocationsByGroupName(ByVal WorkLocationGroupName As
String) As WSDataSet
```

Purpose

Retrieve the work location group list associated with the specified work location group name.

Parameters

Parameter	Description
WorkLocationGroupName	WorkLocation Group name.

Returns

Type	Description
WSDataset	WSDataset of work location groups. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

Not available

Related information

"LookupIds" on page 1041

LookupIds: SaveUDF

```
Public Function SaveUDF(ByVal sXMLUDF As String, ByVal needValidate As
Boolean, ByVal bypassMetadata As CPMetadataCheck) As WSInt32
```

Purpose

Save UDF information to the Changepoint database.

Parameters

Parameter	Description
sXMLUDF	UDF string in XML format.
needValidate	The flag for validation required.
bypassMetadata	The options for Metadata checking. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Use this function to save UDF info for each entity.

This function is designed for updating all UDF fields at once. If you pass values for some UDF fields only, the values for UDF fields for which no values are passed will be cleared.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webLookupIds.LookupIdsWse()
'set the SOAP header with UsernameToken
Dim sXML as String="<root><udf>…</root>"
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As webLookupIds.WSInt32 = proxy.SaveUDF(sXML, True, True)
```

Related information

"LookupIds" on page 1041

LookupIds: ValidateUDF

```
Public Function ValidateUDF(ByVal entity As CPEntity, ByVal sXMLUDF As String,
ByVal entityId As String, ByVal additionalId As String, ByVal metadataCheck As
CPMetadataCheck) As WSInt32
```

Purpose

Validate the UDF information for UDF string.

Parameters

Parameter	Description
entity	CPEntity, for example, CPEntity.Customer = 2.
sXMLUDF	UDF string in XML format.
entityId	EntityId for the UDF, for example, CustomerId for Customer entity.
additionalId	AdditionalId for the UDF. • For Engagement: it is BillingOfficeId; • For Request: it is RequestType. • It is not needed for other Entities.
metadataCheck	The options for Metadata checking. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webLookupIds.LookupIdsWse()
'set the SOAP header with UsernameToken
Dim sXML as String="<root><udf>…</root>"
Dim CustomerId as String="{…}"
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As webLookupIds.WSInt32 = proxy.ValidateUDF(2, sXML, CustomerId, "",
0)
```

Related information

"LookupIds" on page 1041

Opportunity

The Opportunity object allows users to create, retrieve, update and delete Changepoint opportunities as well as information related to the opportunity such as fixed fees, product, services and expenses information.

Namespace

`http://changepoint.com/changepoint/CPWebService/Opportunity`

URL

`http://webserver/CPWebService/Objects/CPOpportunity/Opportunity.asmx`

Methods

Properties

For more information, see the "ApiOpportunity" section on page 399.

Related information

"ApiOpportunity XML" on page 405

"OppExpense" on page 1105

"OppFixedFee" on page 1109

"OppProduct" on page 1110

"OppService" on page 1112

Opportunity: Add

```
Public Function Add(ByRef sId As String, ByVal oOpportunity As ApiOpportunity)
As WSInt32
```

Purpose

Adds a new opportunity to Changepoint.

Parameters

Parameter	Description
sId	Returns the new OpportunityId after the opportunity has been added.
oOpportunity	ApiOpportunity object that contains all the data for the new opportunity being added to Changepoint.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Adds an opportunity to Changepoint. See the properties of the ApiOpportunity object for mandatory fields.The Opportunity object also contains the list of OppExpense, OppFixedFee, OppProduct and OppService objects.

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, ByRef sId As String) As
WSInt32
```

Purpose

Creates an opportunity using an XML string of the Opportunity object in Changepoint.

Parameters

Parameter	Description
sXML	XML string for the new opportunity, which is passed based on the Opportunity XML schema
sId	Returns the new OpportunityId after the opportunity has been added.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The Opportunity XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The ByPassMetadataCheck switch will stop any metadata validation in the Opportunity and also in the Opportunity UDFs (configurable fields).

For details of the use of UDF default values, see "UDF default values logic for CreateByXML" on page 746.

Example

Not available

Related information

"Opportunity" on page 1073

"Opportunity: GetXMLStructure" on page 1098

"Opportunity: GetByXML" on page 1082

"ApiOpportunity XML" on page 405

Opportunity: Delete

```
Public Function Delete(ByVal sId as String = "") As WSInt32
```

Purpose

Deletes an existing opportunity from Changepoint.

Parameters

Parameter	Description
sId	ID of the opportunity to be deleted

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: Exists

```
Public Function Exists(ByVal sOpportunityId As String) As WSBoolean
```

Purpose

Verifies whether the oportunity exists in the database and has not been deleted.

Parameters

Parameter	Description
sOpportunityId	Opportunity ID to verify

Returns

Type	Description
WSBoolean	True if the opportunity exists. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetBillingOffices

```
Public Function GetBillingOffices(ByVal sSalesRepId As String, ByVal
sCurrencyId As String, ByVal sSearchString As String) As WSDataSet
```

Purpose

Returns a list of billing offices based on currency.

Parameters

Parameter	Description
sSalesRepId	Sales Rep ID
sCurrencyId	Currency ID
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (BillingOfficeId, Description). Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetBillingTypes

```
Public Function GetBillingTypes(ByVal sBillingOfficeId As String) As WSDataSet
```

Purpose

Returns a list of billing types for the specified billing office.

Parameters

Parameter	Description
sBillingOfficeId	ID of the billing office

Returns

Type	Description
WSDataset	WSDataset (Code, Description) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetById

```
Public Function GetById(ByVal sOpportunityId As String) As WSOpportunity
```

Purpose

Fills the object with data related to the specified OpportunityId passed in the parameter.

Parameters

Parameter	Description
sOpportunityId	

Returns

Type	Description
WSOpportunity	Returns an ApiOpportunity object. Access the object through the .Value property of WSOpportunity.

Remarks

This method fills the object with current data from the database. In the oportunity object, this method also fills the inner expense, fixed fee, product, and service objects.

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetByXML

```
Public Function GetByXML(ByVal sXML As String, ByVal sOpportunityId As String)
As WSString
```

Purpose

Takes the XML string passed in the sXML parameter and returns the string filled with data for the opportunity specified in the sOpportunityId parameter.

Parameters

Parameter	Description
sXML	XML string of the Opportunity object, with the fields specified
sOpportunityId	ID of the opportunity. • If no OpportunityId is passed in, the XML string (sXML) is examined • If no OpportunityId in sXML then the object's OpportunityId is selected. • If there still is no valid OpportunityId, the method returns an error.

Returns

Type	Description
WSString	An XML string mirroring sXML with data inserted or the entire XML of the Opportunity object including data. Access the returned string through the .Value property of WSString

Remarks

If sXML = "" then the xml string provided by GetXMLStructure is used.

Example

Not available

Related information

"Opportunity" on page 1073

"ApiOpportunity XML" on page 405

Opportunity: GetCompetitors

```
Public Function GetCompetitors() As WSDataSet
```

Purpose

Returns a list of competitors or the competitors associated (if any) with the opportunity.

Parameters

None

Returns

Type	Description
WSDataset	WSDataset (CompetitorId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetContacts

```
Public Function GetContacts(ByVal sCustomerId As String, ByVal sResourceId As
String) As WSDataSet
```

Purpose

Returns a list of contacts

Parameters

Parameter	Description
sCustomerId	Selected customer ID
sResourceId	Resource ID. If the resource ID is not passed, the ID of the logged-in user is used.

Returns

Type	Description
WSDataset	WSDataset (ContactId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetCostCenters

```
Public Function GetCostCenters(ByVal sSearchString As String) As WSDataSet
```

Purpose

Returns a list of cost centers.

Parameters

Parameter	Description
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (CostCenter, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetCustomers

```
Public Function GetCustomers(ByVal sSearchString as String) As WSDataSet
```

Purpose

Returns a list of customers

Parameters

Parameter	Description
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (CustomerId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetFPBillingOffices

```
Public Function GetFPBillingOffices(ByVal sSearchString As String) As
WSDataSet
```

Purpose

Returns a lists of billing offices corresponding to fiscal periods

Parameters

Parameter	Description
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (BillingOfficeId, Description) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As WSString
```

Purpose

Returns the OpportunityId based on the UDF Text field and value.

Parameters

Parameter	Description
sUDFField	UDF text field name (for example, Text1)
sUDFValue	UDF text field value

Returns

Type	Description
WSString	String with the OpportunityId or an empty string

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetList

```
Public Function GetList(ByVal iRetRows As Int16) As WSDataSet
```

Purpose

Returns a list of some or all Opportunities in Changepoint that are not deleted in the system.

Parameters

Parameter	Description
iRetRows	Number of records to be returned. To return all, you must pass in -1.

Returns

Type	Description
WSDataset	WSDataset (OpportunityId, Name) or an empty dataset if nothing is found. Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetOpportunityCurrencies

```
Public Function GetOpportunityCurrencies() As WSDataSet
```

Purpose

Returns a list of currencies

Parameters

None

Returns

Type	Description
WSDataset	WSDataset (Code, Description) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetOpportunityStatus

```
Public Function GetOpportunityStatus() As WSDataSet
```

Purpose

Returns a list of opportunity statuses

Parameters

None

Returns

Type	Description
WSDataset	WSDataset (Code, Description) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetOpportunityTypes

```
Public Function GetOpportunityTypes(ByVal sGlobalWorkgroupId As String, ByVal
sWorkgroupId As String) As WSDataSet
```

Purpose

Returns a list of the Code and Description fields from the OpportunityType table.

Parameters

Parameter	Description
sGlobalWorkgroupId	Global Workgroup ID of the resource for whom the codes are being retrieved.
sWorkgroupId	Workgroup ID of the resource for whom the codes are being retrieved.

Returns

Type	Description
WSDataset	WSDataset (Code, Description) Access the dataset through the .Value property of WSDataSet.

Remarks

This function calls the GetLookUpCodes function in the LookupIds object with parameter sTableName="OpportunityType"

Example

Not available

Related information

"Opportunity" on page 1073

"LookupIds: GetLookupCodes" on page 1053

Opportunity: GetOppPriorities

```
Public Function GetOppPriorities() As WSDataSet
```

Purpose

Returns a list of opportunity priorities

Parameters

None

Returns

Type	Description
WSDataset	WSDataset (Code, Description) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetOppProducts

```
Public Function GetOppProducts (ByVal sSearchString as String) As WSDataSet
```

Purpose

Returns a list of opportunity products.

Parameters

Parameter	Description
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (ProductId, ProductName) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetRequests

```
Public Function GetRequests(ByVal sSearchString as String) As WSDataSet
```

Purpose

Returns a list of requests

Parameters

Parameter	Description
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (RequestId, RequestNum) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetSalesReps

```
Public Function GetSalesReps(ByVal sCustomerId As String, ByVal
sBillingOfficeId As String, ByVal sSearchString As String) As WSDataSet
```

Purpose

Returns a list of the sales representatives based on customer.

Parameters

Parameter	Description
sCustomerId	Customer ID
sBillingOfficeId	Billing Office ID
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (ResouceId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetSalesTeams

```
Public Function GetSalesTeams(ByVal sResourceId As String) As WSDataSet
```

Purpose

Returns a list of sales teams.

Parameters

Parameter	Description
sResourceId	Resource ID. If the resource ID is not passed, the ID of the logged-in user is used.

Returns

Type	Description
WSDataset	WSDataset (ResourceId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetSources

```
Public Function GetSources() As WSDataSet
```

Purpose

Returns a list of sources.

Parameters

None

Returns

Type	Description
WSDataset	WSDataset (CampaignId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetUDF

```
Public Function GetUDF(ByVal retOption As CPUDFReturnType, ByVal entityId As
String, ByVal actionResourceId As String) As WSString
```

Purpose

Returns UDF (configurable field) data for a specified opportunity as an XML string.

Parameters

Parameter	Description
entityId	ID of the opportunity whose UDF data is required
retOption	Determines how the UDF data is to be returned. • OnlyStructure – returns only UDF XML structure, no field data • OnlyStructureAndOptions - returns UDF structure and option data without field data • OnlyValues - returns only UDF fields with data, do not include any field structure or options • WithStructure – returns UDF field data with full field structure • WithStructureAndOptions – returns UDF field data with full structure and all field options
actionResourceId	The Resource ID of the user that called the method. Default is the login user ID.

Returns

Type	Description
WSString	An XML string of UDFs. Access the returned string through the .Value property of WSString

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal sOpportunityId As String, ByVal
sCodeName As String, ByVal sSearchString As String) As WSString
```

Purpose

Returns UDF (configurable field) data for a opportunity as an XML string

Parameters

Parameter	Description
sOpportunityId	Opportunity ID
sCodeName	The name of the UDF code. Must be of the form "Code#", for example, "Code10".
sSearchString	Search string. The search string will match all the options in sCodeName and return all options containing sSearchString in the option text.

Returns

Type	Description
WSString	An XML string of the option list that exists for a specified code. Access the returned string through the .Value property of WSString.

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: GetXMLStructure

```
Public Function GetXMLStructure() As WSString
```

Purpose

Returns the XML structure of the ApiOpportunity object.

Parameters

None

Returns

Type	Description
WSString	Returns the XML structure of the ApiOpportunity object. Access the returned string through the .Value property of WSString.

Remarks

Some fields in the structure will have defaulted data; otherwise, fields are empty.

Example

Not available

Related information

"Opportunity" on page 1073

"ApiOpportunity XML" on page 405

Opportunity: SaveUDF

```
Public Function SaveUDF(ByVal sxmlUDF As String, ByVal needValidate As
Boolean, ByVal bypassMetadata As CPMetadataCheck) As WSInt32
```

Purpose

Saves (Insert/Update) UDF information for an opportunity.

Parameters

Parameter	Description
sXMLUDF	UDF string in XML format to be saved.
needValidate	Flag that determines whether validation is required for sXMLUDF.
bypassMetadata	Determines the level of metadata checking for sXMLUDF. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2 If needValidate is False, then bypassMetadata must be set to CPMetadataCheck.BypassAll = 1.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

To obtain the correct UDF XML format, call the GetUDF method.

Example

Not available

Related information

"Opportunity" on page 1073

"Opportunity: GetUDF" on page 1096

Opportunity: SaveOppExpense

```
Public Sub SaveOppExpense(ByVal oOppExpense As ApiOppExpense, ByVal
oOpportunity As ApiOpportunity)
```

Purpose

Adds an opportunity expense object to the collection.

Parameters

Parameter	Description
oOppExpense	Opportunity expense object to add
oOpportunity	Opportunity object

Returns

Not applicable

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: SaveOppFixedFee

```
Public Sub SaveOppFixedFee(ByVal oOppFixedFee As ApiOppFixedFee, ByVal
oOpportunity As ApiOpportunity)
```

Purpose

Adds an opportunity fixedfee object to the collection.

Parameters

Parameter	Description
oOppFixedFee	Opportunity fixed fee object to add
oOpportunity	Opportunity object

Returns

Not applicable

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: SaveOppProduct

```
Public Sub SaveOppProduct(ByVal oOppProduct As ApiOppProduct, ByVal
oOpportunity As ApiOpportunity)
```

Purpose

Adds an opportunity product object to the collection.

Parameters

Parameter	Description
oOppProduct	Opportunity product object to add
oOpportunity	Opportunity object

Returns

Not applicable

Remarks

None

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: SaveOppService

```
Public Sub SaveOppService(ByVal oOppService As ApiOppService, ByVal
oOpportunity As ApiOpportunity)
```

Purpose

Adds an opportunity service object to the collection.

Parameters

Parameter	Description
oOppService	Opportunity service object to add
oOpportunity	Opportunity object

Returns

Not applicable

Remarks

In order to associated a fixed fee with an opportunity service, the opportunity must be created first and the fixed fee must already be associated at the opportunity level.

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: Update

```
Public Function Update() As WSInt32
```

Purpose

Updates opportunity data in the database with data held in the object

Parameters

None

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The update will write all data held in the Opportunity object including the internal objects for Expense, FixedFee, Product and Service objects.

Example

Not available

Related information

"Opportunity" on page 1073

Opportunity: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, ByVal sOpportunityId As
String) As WSInt32
```

Purpose

Updates an opportunity using an XML string containing new data.

Parameters

Parameter	Description
sXML	XML string of the Opportunity object, with new data contained in the fields
sOpportunityId	ID of the opportunity. If OpportunityId is in the sXML parameter, then you can pass in "" for sOpportunityId parameter.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Performs the same function as Update except any Opportunity can be updated through this function. The XML sent in the parameter must be of the form in ApiOpportunity XML. The ApiOpportunity XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The method uses the following sequence to find the opportunity ID:

1. If the sOpportunityId parameter is passed in, the method uses this value for the opportunity ID.
2. If this fails, the method attempts to extract the opportunity ID from <opportunityid> in the XML.
3. If this fails, the opportunity ID is taken from the object properties.
4. If this fails, an attempt is made to look up the opportunity ID using <name> in the XML.

Example

Not available

Related information

"Opportunity" on page 1073

"Opportunity: GetByXML" on page 1082

"Opportunity: GetXMLStructure" on page 1098

"ApiOpportunity XML" on page 405

OppExpense

The OppExpense object contains expense data for the opportunity.

Namespace

`http://changepoint.com/changepoint/CPWebService/OppExpense`

URL

`http://webserver/CPWebService/Objects/CPOpportunity/ OppExpense.asmx`

Methods

Properties

For more information, see the "ApiOppExpense" section on page 435

Related information

"Opportunity" on page 1073

"ApiOppExpense XML" on page 437

OppExpense: GetExpenseCategories

```
Public Function GetExpenseCategories() As WSDataSet
```

Purpose

Returns a list of opportunity expense categories

Parameters

None

Returns

Type	Description
WSDataset	WSDataset (Id, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"OppExpense" on page 1105

OppExpense: GetExpenseTypesByCategory

```
Public Function GetExpenseTypesByCategory(ByVal sCategoryId As String) As
WSDataSet
```

Purpose

Returns a list of opportunity expense types.

Parameters

Parameter	Description
sCategoryId	Expense category ID

Returns

Type	Description
WSDataset	WSDataset (ExpenseTypeId, Description) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"OppExpense" on page 1105

OppExpense: GetList

```
Public Function GetList(ByVal sOpportunityId As String, ByVal iRetRows As
Int16) As WSDataSet
```

Purpose

Fills the object with data related to the specified OpportunityId passed in the parameter.

Parameters

Parameter	Description
sOpportunityId	Opportunity ID
iRetRows	Number of rows to return. To return all rows, you must pass in -1.

Returns

Type	Description
WSDataset	WSDataset with opportunity expense data, or an empty dataset if nothing is found. Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"OppExpense" on page 1105

OppExpense: New

```
Public Function New()
```

Purpose

Instantiates a new OppExpense object.

Parameters

None

Returns

Not applicable

Remarks

In the new OppExpense object, the ByPassMetadataCheck flag is set to False by default.

Example

Not available

Related information

"OppExpense" on page 1105

OppFixedFee

The OppFixedFee object contains fixed fee data for the opportunity.

Namespace

`http://changepoint.com/changepoint/CPWebService/OppFixedFee`

URL

`http://webserver/CPWebService/Objects/CPOpportunity/ OppFixedFee.asmx`

Methods

Properties

For more information, see the "ApiOppFixedFee" section on page 440

Related information

"Opportunity" on page 1073

"ApiOppFixedFee XML" on page 441

OppFixedFee: GetList

```
Public Function GetList(ByVal sOpportunityId As String, ByVal iRetRows As
Int16) As WSDataSet
```

Purpose

Returns a list of fixed fees for the specified opportunity.

Parameters

Parameter	Description
sOpportunityId	Opportunity ID
iRetRows	Number of rows to return. To return all rows, you must pass in -1.

Returns

Type	Description
WSDataset	WSDataset with opportunity data, or an empty dataset if nothing is found. Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"OppFixedFee" on page 1109

OppProduct

The OppProduct object contains product data for the opportunity.

Namespace

`http://changepoint.com/changepoint/CPWebService/OppProduct`

URL

`http://webserver/CPWebService/Objects/CPOpportunity/OppProduct.asmx`

Methods

Properties

For more information, see the "ApiOppProduct" section on page 443

Related information

"Opportunity" on page 1073

"ApiOppProduct XML" on page 445

OppProduct: GetList

```
Public Function GetList(ByVal sOpportunityId As String, ByVal iRetRows As
Int16) As WSDataSet
```

Purpose

Returns a list of opportunity products for the specified opportunity.

Parameters

Parameter	Description
sOpportunityId	Opportunity ID
iRetRows	Number of rows to return. To return all rows, you must pass in -1.

Returns

Type	Description
WSDataset	WSDataset with opportunity product data, or an empty dataset if nothing is found. Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"OppProduct" on page 1110

OppProduct: GetProducts

```
Public Function GetProducts(ByVal sSearchString As String) As WSDataSet
```

Purpose

Returns a list of opportunity products

Parameters

Parameter	Description
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset containing a list of opportunity products. Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"OppProduct" on page 1110

OppService

The OppService object contains service data for the opportunity.

Namespace

`http://changepoint.com/changepoint/CPWebService/OppService`

URL

`http://webserver/CPWebService/Objects/CPOpportunity/ OppService.asmx`

Methods

Properties

For more information, see the "ApiOppService" section on page 447.

Related information

"Opportunity" on page 1073

"ApiOppService XML" on page 450

OppService: GetBillingOfficeList

```
GetBillingOfficeList(ByVal salesRepId As String, Optional ByVal searchCriteria
As String) As WSDataSet
```

Purpose

Returns a list of the available billing offices.

Parameters

Parameter	Description
salesRepId	Sales rep ID
searchCriteria	Search string

Returns

Type	Description
WSDataset	WSDataset (Id, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"OppService" on page 1112

OppService: GetBillingRoleList

```
Public Function GetBillingRoleList(ByVal billingOfficeId As String) As
WSDataSet
```

Purpose

Returns a list of the billing roles for the billing office

Parameters

Parameter	Description
sBillingOfficeId	Billing office ID. If an empty string is passed in, then the value of the BillingOffice property is used

Returns

Type	Description
WSDataset	WSDataset (Code, Description) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"OppService" on page 1112

OppService: GetCurrencyByBillingRoleList

```
GetCurrencyByBillingRoleList(ByVal billingRoleId As String, ByVal
effectiveDate As DateTime) As WSDataSet
```

Purpose

Returns a list of the currencies based on the specified billing role and effective date.

Parameters

Parameter	Description
billingRoleId	Billing role ID
effectiveDate	Effective date for the currency

Returns

Type	Description
WSDataset	WSDataset (Code, Description) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"OppService" on page 1112

OppService: GetFixedFeeList

```
Public Function GetFixedFeeList(ByVal sOpportunityId As String) As WSDataSet
```

Purpose

Returns a list of the opportunity fixed fees for the opportunity service.

Parameters

Parameter	Description
sOpportunityId	Opportunity ID

Returns

Type	Description
WSDataset	WSDataset (OpportunityFixedFeeId, Deliverable, FixedFeeAmount, FixedFeeDate) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"OppService" on page 1112

OppService: GetList

```
GetList(ByVal sOpportunityId As String, ByVal iRetRows As Int16) As WSDataSet
```

Purpose

Fills the object with data related to the specified OpportunityId passed in the parameter.

Parameters

Parameter	Description
sOpportunityId	Opportunity ID
iRetRows	Number of rows to return. To return all rows, you must pass in -1.

Returns

Type	Description
WSDataset	WSDataset (ServicesId, OpportunityId, CustomerId, CustomerName, AlternateName, CustomerAlternateName, BillingRoleId, BillingRoleDescription, FunctionId, FunctionDescription, ResourceId, ResourceName, ResourceAlternateName, EstimatedTime, ServiceTotal), or an empty dataset if nothing is found. Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"OppService" on page 1112

OppService: GetResourceList

```
GetResourceList(ByVal searchCriteria As String) As WSDataSet
```

Purpose

Returns a list of the available resources who have the TTP (Track Project Related Time) feature and who meet the search criteria.

Parameters

Parameter	Description
sSearchString	Search string

Returns

Type	Description
WSDataset	WSDataset (Id, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"OppService" on page 1112

Product

The Product object allows users to create, retrieve, edit and delete products in the Changepoint database. Related to the Product object are the following objects:

- "ApiProductCurrency" on page 479.
- "ApiProductTeamMember" on page 479.

Namespace

`http://changepoint.com/changepoint/CPWebService/Product`

URL

`http://webserver/CPWebService/Objects/CPProduct/Product.asmx`

Methods

Properties

For more information, see the "ApiProduct" section on page 457.

Product: AddProdPrice

```
Public Function AddProdPrice(ByVal oCPProd As ApiProduct, ByVal sProdCurId As
String, ByVal sCurCode As String, ByVal curPrice As Decimal) As WSBoolean
```

Purpose

Add a new price to the ApiProduct object oCPProd

Parameters

Parameter	Description
oCPProd	ApiProduct object
sProdCurId	Product currency ID. The unique identifier for the Product Price record. For a new Product Price, pass an empty string for sProdCurId.
sCurCode	Three letter Currency code, such as "CAD" or "USD"
curPrice	Product price

Returns

Type	Description
WSBoolean.	True for a successful addition else false. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

None

Example

Not available

Related information

"Product" on page 1118

Product: AddTeamMember

```
Public Function AddTeamMember(ByRef oCPProd As ApiProduct, ByVal sResourceId
As String) As WSBoolean
```

Purpose

Add Team Member into the ApiProductTeam in oCPProd

Parameters

Parameter	Description
oCPProd	ApiProduct object to which a new team member will be added.
sResourceId	Resource to add to the team in oCPProd

Returns

Type	Description
WSBoolean.	True for a successful addition else false. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

None

Example

Not available

Related information

"Product" on page 1118

Product: CreateProduct

```
Public Function CreateProduct(ByVal oCPProd As ApiProduct, ByRef vProductId As
String) As WSInt32
```

Purpose

Add a new product to Changepoint based on the data held in oCPProd

Parameters

Parameter	Description
oCPProd	Populated ApiProduct object
vProductId	ProductId in string. An empty string is allowed.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

This process adds a new product but also adds pricing, Product Team and UDF data to Changepoint.

There are two fields that must have a value: ProductId, ProductCategoryId

Example

```
Dim proxy As New WebProduct.ProductWse
Dim myProduct As New WebProduct.ApiProduct
Dim sErrorString As String = ""
Dim dsProdCat As New DataSet
Dim sProdCatId As String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
'Get a list of product categories and extract the Id
dsProdCat = proxy.GetProductCategories().Value
If dsProdCat.Tables.Count > 0 AndAlso dsProdCat.Tables(0).Rows.Count > 0 Then
   For Each mRow As DataRow In dsProdCat.Tables(0).Rows
      If mRow.Item("ProductCategoryName").ToString = "MyProductCategory" Then
         sProdCatId = mRow.Item ("ProductCategoryId").ToString
         Exit For
      End If
   Next mRow
End If
If sProdCatId = "" Then
   ...'Handle error
End If
```

```
With myProduct
   .ProductName = "My Test Product 1"
   .ProductCategoryId = sProdCatId
   .ProductNumber = "P001"
   .ProductDescription = "Test Description"
...
End With
'Add a new price to the object
'Note: Use the GetProdCurrency("") method to extract a list of currency codes
with 'descriptions if the code is unknown and only a description is known i.e.
"American 'Dollar" the returned list can be searched for the matching
description.
If proxy.AddProdPrice( myProduct,"", "CAD", 2000.00) <> 0 Then
   ...'Handle error
End If
'Add a team member named John Smith, using the preferred resources
'of the logged in user.
Dim dsPrefRes as New DataSet
dsPrefRes = proxy.GetPrefRes().Value
If dsPrefRes.Tables.Count > 0 AndAlso dsPrefRes.Tables(0).Rows.Count > 0 Then
   For each mDataRow As DataRow In dsPrefRes.Tables(0).Rows
      If dsPrefRes.Item("ResourceName").ToString = "John Smith" Then
         If proxy.AddTeamMember(myProduct, dsPrefRes.Item
("ResourceId").ToString) = 0 Then
         Exit For
      Else
      ...'Handle error
      End If
   End If
   Next
End If

If proxy.CreateProduct(myProduct).Value <> 0
   ...'Handle error
End If
```

Related information

"Product" on page 1118

Product: DeleteProduct

```
Public Function DeleteProduct(ByVal sProductId As String, ByVal sProdCatId As
String) As WSInt32
```

Purpose

Delete an existing product from Changepoint

Parameters

Parameter	Description
sProductId	ID of the product to be deleted
sProdCatId	Product Category to which the product belongs.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

UDF (configurable field) data related to the product is also removed.

Example

Not available

Related information

"Product" on page 1118

Product: DelProdPrice

```
Public Function DelProdPrice(ByVal sProductId As String, ByVal sProdCurId As
String) As WSBoolean
```

Purpose

Delete Product Price(s) as specified by sProdCurId

Parameters

Parameter	Description
sProductId	Used to delete all Product Price(s) associated with the Product if no sProdCurId is passed in.
sProdCurId	ID of the Product price record to be deleted

Returns

Type	Description
WSBoolean	True on success, else False Access the returned Boolean through the .Value property of WSBoolean.

Remarks

The method deletes ProductPrice from the database.

Example

Not available

Related information

"Product" on page 1118

Product: DelTeamMember

```
Public Function DelTeamMember(ByVal sProdCatId As String, ByVal sProductId As
String, ByVal sResourceId As String) As WSBoolean
```

Purpose

Delete Team Member(s)

Parameters

Parameter	Description
sProdCatId	Product CategoryId of the Product the team member(s) are associated with
sProductId	ProductId of the Product the team member(s) are associated with
sResourceId	If a value is provided, then the single team member associated to the ResourceId will be deleted. If an empty string is provided, then all team members associated to the ProductId will be deleted.

Returns

Type	Description
WSBoolean	True on success else False Access the returned Boolean through the .Value property of WSBoolean.

Remarks

The method deletes ProductTeam members from the database.

Example

Not available

Related information

"Product" on page 1118

Product: GetPrefRes

```
Public Function GetPrefRes() As WSDataset
```

Purpose

Retrieve a list of preferred resources

Parameters

None

Returns

Type	Description
WSDataset	WSDataset(ResourceId, ResourceName, WorkgroupId, WorkgroupName, GlobalWorkgroupId, GlobalWorkgroupName, Language, BaseCurrency, ResourceType, AltResourceName, WAltName) WAltName is the Workgroup alternate name. Access the returned dataset through the .Value property of WSDataSet.

Remarks

The list is generated based on the logged in user and their preferred resources list as defined in Changepoint.

Can be used for selection of new Team members.

Example

Not available

Related information

"Product" on page 1118

Product: GetProdCur

```
Public Function GetProdCur(ByVal sLang As String) As WSDataset
```

Purpose

Retrieve product currencies

Parameters

Parameter	Description
sLang	Language, for example, "E"

Returns

Type	Description
WSDataset	WSDataset (Code, Description) Access the DataSet through the .Value property of WSDataSet.

Remarks

The parameter sLang can be an empty string, since the language is taken from the base language set in Changepoint.

Example

Not available

Related information

"Product" on page 1118

Product: GetProdTeam

```
Public Function GetProdTeam (ByVal oCPProd As ApiProduct) As WSDataset
```

Purpose

Retrieve Product team resources based on a specified Product and its category

Parameters

Parameter	Description
oCPProd	oCPProd

Returns

Type	Description
WSDataset	WSDataset (ResourceId, Name, AlternateName) Access the dataset through the .Value property of WSDataSet.

Remarks

Property .ProductCategoryId and .ProductId are required.

Example

Not available

Related information

"Product" on page 1118

Product: GetProductById

```
Public Function GetProductById (ByVal sProductId As String) As WSProduct
```

Purpose

Retrieve product data for a specified product Id

Parameters

Parameter	Description
sProductId	ID of the product whose data is required

Returns

Type	Description
WSProduct	Returns an ApiProduct object. Access the object through the .Value property of WSProduct.

Remarks

Includes related Product Team and Price data

Example

Not available

Related information

"Product" on page 1118

Product: GetProductByName

```
Public Function GetProductByName (ByVal sProductName As String) As WSProduct
```

Purpose

Retrieve current product data based on the product's name

Parameters

Parameter	Description
sProductName	Name of the product being retrieved

Returns

Type	Description
WSProduct	Returns an ApiProduct object. Access the object through the .Value property of WSProduct.

Remarks

Includes related Product Team and Price data

Example

Not available

Related information

"Product" on page 1118

Product: GetProductCategories

```
Public Function GetProductCategories() As WSDataset
```

Purpose

Returns a list of all product categories

Parameters

None

Returns

Type	Description
WSDataSet	WSDataSet (ProductCategoryID, ProductCategoryName) Access the DataSet through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Product" on page 1118

Product: GetProductCurrency

```
Public Function GetProductCurrency (ByVal sLang As String) As WSDataset
```

Purpose

Retrieve product currencies

Parameters

Parameter	Description
sLang	Language, for example, "E"

Returns

Type	Description
WSDataset	WSDataset (Code, Description) Access the DataSet through the .Value property of WSDataSet.

Remarks

The parameter sLang can be an empty string, since the language is taken from the base language set in Changepoint.

Example

Not available

Related information

"Product" on page 1118

Product: GetProductIdByName

```
Public Function GetProductIdByName(ByVal sName As String) As WSString
```

Purpose

Get the ProductId based on the name of the product.

Parameters

Parameter	Description
sName	Name of the product to retrieve.

Returns

Type	Description
WSString	WSString with the ProductId. Access the returned string through the .Value property of WSString.

Remarks

None

Example

Not available

Related information

"Product" on page 1118

Product: GetProductPrice

```
Public Function GetProductPrice(ByVal sProductId As String, ByVal sProdCatId
As String, ByVal sLang As String) As WSDataset
```

Purpose

Returns product pricing data based on the Product name, Category and language

Parameters

Parameter	Description
sProductId	ID of the product whose pricing data is required.
sProdCatId	The Product Category of sProductId
sLang	Language for pricing

Returns

Type	Description
WSDataset	WSDataset (ProductCurrencyId, Code, Price, Description) Access the DataSet through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Product" on page 1118

Product: GetProducts

```
Public Function GetProducts(ByVal sCatId As String, ByVal sStatus As String)
As WSDataset
```

Purpose

Retrieve all products by the category and/or status of the product

Parameters

Parameter	Description
sCatId	ID of the Product Category.
sStatus	The status of the product.

Returns

Type	Description
WSDataset	WSDataset (ProductId, ProductName) ordered by product name Access the DataSet through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Product" on page 1118

Product: GetProductStatus

```
Public Function GetProductStatus() As WSDataset
```

Purpose

Retrieve all available product status values based on the current logged-in user.

Parameters

None

Returns

Type	Description
WSDataset	WSDataset (Code, Description) Access the DataSet through the .Value property of WSDataSet.

Remarks

The returned status data is filtered based on the Workgroup or GlobalWorkgroup of the logged-in user.

Example

Not available

Related information

"Product" on page 1118

Product: GetProductTeam

```
Public Function GetProductTeam (ByVal sProductId As String, ByVal sProdCatId
As String) As WSDataset
```

Purpose

Retrieve Product team resources based on a specified Product and its category

Parameters

Parameter	Description
sProductId	ID of the product whose team is required
sProdCatId	Category of the Product sProductId.

Returns

Type	Description
WSDataset	WSDataset (ResourceId, Name, AlternateName) Access the DataSet through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Product" on page 1118

Product: GetResource

```
Public Function GetResource() As WSDataset
```

Purpose

Retrieve a list of all resources currently assigned to a workgroup

Parameters

None

Returns

Type	Description
WSDataset	WSDataset (ResourceId, Name, AlternateName, BaseCurrency) Access the DataSet through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Product" on page 1118

Product: GetUDF

```
Public Function GetUDF(ByVal entity As CPEntity, ByVal retOption As
CPUDFReturnType, ByVal entityId As String, ByVal actionResourceId As String)
As WSString
```

Purpose

Extract UDF data as an XML string based on the logged in resource and/or product

Parameters

Parameter	Description
entity	The entity whose UDF data is being sought. In this case the entity is 'Product'
retOption	The CPUDFReturnType Values are: • WithStructure = 0 – returns UDF field data with full field structure • OnlyValues = 1 – returns only UDF fields with data, does not include any field structure or options • OnlyStructure = 2 – returns only UDF XML structure, no field data • OnlyStructureAndOptions = 3 – returns UDF structure and option data without field data • WithStructureAndOptions = 4 – returns UDF field data with full structure and all field options (except entity-based and conditional codes)
entityId	ID of the product whose UDF data is required.
actionResourceId	The logged in user or user activating the function.

Returns

Type	Description
WSString.	The UDF string in XML format. Access the returned string through the .Value property of WSString.

Remarks

Use an empty value for sProductId to have default UDF values included in the returned string.

Example

Not available

Related information

"Product" on page 1118

Product: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal sProductId As String, ByVal codeName
As String, ByVal searchString As String) As WSString
```

Purpose

Retrieve UDF Code options as an XML string based on the Code name and/or the action resource, a search string

Parameters

Parameter	Description
sProductId	Product ID
codeName	Name of the code whose options are needed
SearchString	Search string for filter

Returns

Type	Description
WSString	The UDF code options string in XML format. Access the returned string through the .Value property of WSString.

Remarks

The search string will be used to search the options text for all occurrences of searchString.

Use an empty value for searchString to have all options for the code included in the returned string.

The Code name is of the form "Code#" where "#" is the code number, for example, "Code10"

Example

Not available

Related information

"Product" on page 1118

Product: SaveUDF

```
Public Function SaveUDF(ByVal sXMLUDF As String, ByVal needValidate As
Boolean, ByVal bypassMetadata As CPMetadataCheck) As WSInt32
```

Purpose

Save UDF (configurable field) information to the Changepoint database.

Parameters

Parameter	Description
sXMLUDF	UDF string in XML format.
needValidate	The flag for validation required.
bypassMetadata	The options for Metadata checking. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

It is recommended that you use this function to save UDF info for each entity.

This function is designed for updating all UDF fields at once. If you pass values for some UDF fields only, the values for UDF fields for which no values are passed will be cleared.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webProduct.ProductWse()
'set the SOAP header with UsernameToken
Dim sXML as String="<root><udf>…</root>"
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As webProduct.WSInt32 = proxy.SaveUDF(sXML, True, True)
```

Related information

"Product" on page 1118

Product: UpdateProduct

```
Public Function UpdateProduct (ByRef oCPProd As ApiProduct) As WSInt32
```

Purpose

Update Changepoint with updates to an existing product.

Parameters

Parameter	Description
oCPProd	Populated ApiProduct object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned value through the .Value property of WSInt32

Remarks

Also updates Product Team data as well as pricing and UDF data.

Ensure that ProductId and ProductCategoryId are populated as the update process will fail without these two values.

Example

```
Dim proxy As New WebProduct.ProductWse
Dim myProduct As New WebProduct.ApiProduct
```

```
Dim myProdCurrency as New WebProduct.ApiProductCurrency
Dim myPriceLst As WebProduct.ApiProductCurrency()

UserToken.SetToken(proxy, mUserName, mPassword)
myProduct = proxy.GetProductByName("MyTestProduct").value
With myProduct
   .ProductName = "My Updated Test Product 1"
   .ProductDescription = "Updated Test Description"
   ...
End With
'Want to update the CAD price. Get an array of ProductCurrency objects from
the 'returned array of ApiProductCurrency objects and find the needed object
Dim sProdCurrId as String = ""
myPriceLst = myProduct.DicPrice
Dim iCnt as Integer = 0

Do While iCnt < myPriceLst.Length
   myProdCurrency = myPriceLst(iCnt)
   If myProdCurrency.Code = "CAD" Then
      sProdCurrId = myProdCurrency.ProdCurId
      Exit Do
   End If
   iCnt = iCnt + 1
Loop
'Update the price
If Not proxy.UpdProdPrice(sProdCurrId, "CAD", 5000.00).Value Then
   ...'Handle error
End If
'Update Changepoint
If proxy.UpdateProduct(myProduct).Value <> 0
   ...'Handle update error
End If
```

Related information

"Product" on page 1118

Product: UpdProdPrice

```
Public Function UpdProdPrice (ByVal sProdCurId As String, ByVal sCurCode As
String, ByVal curPrice As Decimal) As WSBoolean
```

Purpose

Updates the product price in the object oTmpProd

Parameters

Parameter	Description
sProdCurId	Product currency ID.
sCurCode	Product currency code.
curPrice	Updated price Product Price.

Returns

Type	Description
WSBoolean	Returns true upon success else false. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

None

Example

Not available

Related information

"Product" on page 1118

Project

The Project object allows users to add, retrieve, update and delete project information within the Changepoint database.

Namespace

`http://changepoint.com/changepoint/CPWebService/Project`

URL

`http://webserver/CPWebService/Objects/CPProject/Project.asmx`

Methods

Properties

For more information, see the "ApiProject" section on page 480.

Related information

"ApiProject XML" on page 486.

"Task" on page 1295

"TaskAssignment" on page 1309

Project: Add

```
Public Function Add(ByRef sId As String, ByVal oProject As ApiProject) As
WSInt32
```

Purpose

Add a new project to Changepoint database by assigned project Id.

Parameters

Parameter	Description
sId	Project ID for new Project, if it is not assigned value before passing, the system will create a new GUID and return it.
oProject	ApiProject object which will be added.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim objApi As New webProject.APIProject
With ojbApi
.Name="Richard Project36"
.Customer=New webProject.Identity()
.Customer.Id= "{…}"
.Engagement=New webProject.Identity()
.Engagement.Id = "{…}"
.Currency="CAD"
...
End With
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy)
Dim sId as String = ""
Dim oRet As webTask.WSInt32 = proxy.Add(sId, objApi)
```

Related information

"Project" on page 1143

Project: CreateByXML

```
Public Function CreateByXML(ByVal sManagerId As String, ByVal sXML As String,
ByVal bIgnoreWarning As Boolean, ByRef sId As String) As WSInt32
```

Purpose

Creates and updates a project, tasks and task assignments from passed-in XML data.

Parameters

Parameter	Description
sManagerId	Manager for new Project.
sXML	Project (Tasks/TaskAssignments) in XML format.
bIgnoreWarning	Flag for ignore the Resource Leveling warning.
sId	Project ID for new Project. Project ID for new Project, if it is not assigned value before passing, the system will create a new GUID and return it.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

The XML template can be retrieved by the CreateByXMLTemplate method.

The mandatory `<iUpdate>` element as a child of `<Project>` is an action flag with available values:

1 - Add

2 - Update

9 - No change

In addition, there are mandatory `<iUpdate>` flags in the Task XML and in the TaskAssignment XML that are child elements of `<Task>` and `<Assignment>` respectively, and that determine the action for the Task and TaskAssignment respectively:

1 - Add

2 - Update

3 - Delete

9 - No change

For example, to create a task and a task assignment for an existing project, set iUpdate (project level) to '2', iUpdate (task level) to '1', and iUpdate (task assignment level) to '1'.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

For details of the use of UDF default values, see "UDF default values logic for CreateByXML" on page 746.

Example

```
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim ManagerId As String="{…}"
Dim sXML As String = "<root>…</root>"
Dim sId as String = ""
Dim oRet As webProject.WSInt32 = proxy.CreateByXML(ManagerId, sXML, True, sId)
```

Related information

"Project" on page 1143

"Project: CreateByXMLTemplate" on page 1147

"ApiProject XML" on page 486

"ApiTask XML" on page 676

"ApiTaskAssignment XML" on page 694

Project: CreateByXMLTemplate

```
Public Function CreateByXMLTemplate() As WSString
```

Purpose

Returns the XML structure of the project, task and task assignment.

Parameters

None

Returns

Type	Description
WSString	XML structure of the project, task and task assignment Access the returned string through the .Value property of WSString.

Remarks

None

Example

```
Dim myProxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webProject.WSString = myProxy.CreateByXMLTemplate()
```

Related information

"Project" on page 1143

"ApiProject XML" on page 486

Project: Delete

```
Public Function Delete(ByVal sId As String) As WSInt32
```

Purpose

Delete existing project from Changepoint database.

Parameters

Parameter	Description
sId	Project ID for deleting.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webProject.WSInt32 = proxy.Delete(sId)
```

Related information

"Project" on page 1143

Project: Exists

```
Public Function Exists(ByVal sId As String) As wsBoolean
```

Purpose

Check if the project exists.

Parameters

Parameter	Description
sId	Project ID for checking.

Returns

Type	Description
WSBoolean	True if the project exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webProject.wsBoolean = proxy.Exists(sId)
```

Related information

"Project" on page 1143

Project: GetBaselineHistory

```
Public Function GetBaselineHistory(ByVal sProjectId As String) As WSDataSet
```

Purpose

Retrieves Baseline History information for a project

Parameters

Parameter	Description
sProjectId	ID of the project

Returns

Type	Description
WSDataset	WSDataset containing all the fields in the DS_ProjectBaseline view in the database. Access the dataset through the .Value property of WSDataSet.

Remarks

Contains the current baseline and all historical baselines for the project in descending order by createdon date. Check return object WSException.HaveErrors before reading the value. If the parameter is not provided or is incorrect, the function will throw error number -7.

Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken which include login user and access
token
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webProject.WSDataset = proxy.GetBaselineHistory(sProjectId)
...
```

Project: GetById

```
Public Function GetById(ByVal sId As String) As wsProject
```

Related information

"Project" on page 1143

Purpose

Retrieve Project object from Changepoint database.

Parameters

Parameter	Description
sId	ID of project which will be retrieved.

Returns

Type	Description
WSProject	Returns an ApiProject object. `Access the object through the .Value property of WSProject.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webProject.wsProject = proxy.GetById(sId)
```

Related information

"Project" on page 1143

Project: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As WSString
```

Purpose

Returns the ProjectId based on the UDF Text field and value.

Parameters

Parameter	Description
sUDFField	UDF text field name (for example, Text1)
sUDFValue	UDF text field value

Returns

Type	Description
WSString	String with the ProjectId or an empty string. Access the returned string through the .Value property of WSString

Remarks

None

Example

Not available

Related information

"Project" on page 1143

Project: GetList

```
Public Function GetList(ByVal sCustomerId As String, ByVal sEngagementId As
String, ByVal iRetRows As Int16) As WSDataset
```

Purpose

Retrieve Project list from Changepoint database.

Parameters

Parameter	Description
sCustomerId	CustomerId filter. An empty string is allowed.
sEngagementId	EngagementId filter. An empty string is allowed.
iRetRows	The number of records to be returned. To return all, you must pass in -1

Returns

Type	Description
WSDataset	WSDataset (CustomerId, EngagementId, ProjectId, ProjectName, UserDefinedProjectId) Access the dataset through the .Value property of WSDataSet.

Remarks

Returns all rows when sCustomerId = "", sEngagementId= "" and iRetRows = -1

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim customerId as String = "{…}"
Dim oRet As webProject.WSDataset = proxy.GetList(customerId, "", 100)
```

Related information

"Project" on page 1143

Project: GetManager

```
Public Function GetManager(ByVal sProjectId As String) As WSDataSet
```

Purpose

Retrieve manager of the project.

Parameters

Parameter	Description
sProjectId	ID of the project.

Returns

Type	Description
WSDataset	WSDataset (ResourceId, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None.

Example

```
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webProject.WSDataset = proxy.GetManager ("{…}")
```

Related information

"Project" on page 1143

Project: GetUDF

```
Public Function GetUDF(ByVal retOption As CPUDFReturnType, ByVal entityId As
String, ByVal actionResourceId As String) As WSString
```

Purpose

Retrieve Project UDF (configurable field) information.

Parameters

Parameter	Description
retOption	The CPUDFReturnType Values are: • WithStructure = 0 – returns UDF field data with full field structure • OnlyValues = 1 – returns only UDF fields with data, does not include any field structure or options • OnlyStructure = 2 – returns only UDF XML structure, no field data • OnlyStructureAndOptions = 3 – returns UDF structure and option data without field data • WithStructureAndOptions = 4 – returns UDF field data with full structure and all field options (except entity-based and conditional codes)
entityId	Project ID for the UDF.
actionResourceId	ResourceId for calling the function, default is login userid.

Returns

Type	Description
WSString	XML string of project UDF information. Access the returned string through the .Value property of WSString.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webProject.wsString = proxy.GetUDF(1, sId, "")
```

Related information

"Project" on page 1143

Project: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal sProjectId As String, ByVal codeName
As String, ByVal searchString As String) As WSString
```

Purpose

Retrieves UDF code options.

Parameters

Parameter	Description
sProjectId	ID of the project.
codeName	Code name for retrieving options.
searchString	If not empty, the function will return the options starting with this string. An empty string is allowed.

Returns

Type	Description
WSString	XML string of UDF code options. Access the returned string through the .Value property of WSString.

Remarks

Returns code options string in XML format.

Example

```
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webProject.WSString = proxy.GetUDFCodeOptions("{…}","Code1", "")
```

Related information

"Project" on page 1143

Project: SaveBaseline

```
Public Function SaveBaseline(ByVal sProjectId As String) As WSInt32
```

Purpose

Saves current PlannedStart, PlannedFinish, PlannedHours information for a project into the baseline fields.

Parameters

Parameter	Description
sProjectId	The ID of the project

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Current Planned information is saved to the Baseline fields in the project and a new record is created in the Baselines table. Check return object WSException.HaveErrors before reading the value.

Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken which include login user and access
token
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webProject.WSInt32 = proxy.SaveBaseline(sProjectId)
...
```

Related information

"Project" on page 1143

Project: SaveUDF

```
Public Function SaveUDF(ByVal sXMLUDF As String, ByVal needValidate As
Boolean, ByVal bypassMetadata As CPMetadataCheck) As WSInt32
```

Purpose

Save (Insert/Update) UDF information for Project object.

Parameters

Parameter	Description
sXMLUDF	The UDF fields in XML string.
needValidate	The flag for validation required.
bypassMetadata	The options for Metadata checking. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim sXML as String = "<root><udf>…</root>"
Dim oRet As webProject.wsInt32 = proxy.SaveUDF(sXML, True, 0)
```

Related information

"Project" on page 1143

Project: Update

```
Public Function Update(ByVal oProject As ApiProject) As WSInt32
```

Purpose

Update Project object in Changepoint database.

Parameters

Parameter	Description
oProject	ApiProject object which will be updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error `Access the returned Int32 through the .Value property of WSInt32.`

Remarks

The Project object will be updated with all the properties assigned. Note that the Baseline fields are no longer saved as part of the Update method. See the SaveBaseline method.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webProject.wsProject = proxy.GetById(sId)
If Not oRet.WSException.HaveErrors then
Dim objApi As webProject.APIProject = oRet.value
```

```
With ojbApi
'update with new data
.Status = "A"
.Currency="CAD"
…
End With
Dim iRet As webProject.WSInt32 = proxy.Update(objApi)
End If
```

Related information

"Project" on page 1143

"Project: SaveBaseline" on page 1157

Request

The ApiRequest object allows users to create, retrieve, update and delete Changepoint requests. With the migration to .NET in version 12.0, the interface IRequestBase inherits IBusinessPropertyBase and the class ApiRequest implements IRequestBase.

Namespace

`http://changepoint.com/changepoint/CPWebService/Request`

URL

`http://webservice/CPWebService/Objects/CPRequest/Request.asmx`

Methods

Properties

For more information, see the "ApiRequest" section on page 504.

Related information

"ApiRequest XML" on page 509

"RequestDemand" on page 1207

Request: Add

```
Public Function Add(ByRef sId as String, ByVal oRequest As ApiRequest) As
WSInt32
```

Purpose

Create a new request in Changepoint

Parameters

Parameter	Description
sId	New ID of the entity (Request).
oRequest	ApiRequest object containing all data for the new request.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Creates a new request and returns the new ID value through the reference parameter.

Example

```
Dim proxy As New WebRequest.RequestWse
Dim myRequest as New WebRequest.ApiRequest
Dim iRet as Int32 = 0
Dim sNewId as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
With myRequest
   'Set property values as necessary, or use an XML string and avoid setting
   'properties individually.
   ...
End With
iRet = proxy.Add(sNewId, myRequest).Value
If iRet = 0 Then Returns sNewId
```

Related information

"Request" on page 1161

Request: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, ByRef sId As String) As
WSInt32
```

Purpose

Create a request using an XML string of the Request object in Changepoint.

Parameters

Parameter	Description
sXML	A new request XML string.
sId	ByRef parameter that returns the new RequestId after the request has been added.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

ApiRequest XML structure can be obtained by GetXMLStructure or GetByXML methods.

The ByPassMetadataCheck switch will stop any meta data validation in Request and also in Request UDF's.

It is not recommended to set the ByPassWorkflow switch for a request using workflow. Only set this switch when workflow is not expected to be active for the Request.

For details of the use of UDF default values, see "UDF default values logic for CreateByXML" on page 746.

Example

```
Dim proxy As New WebRequest.RequestWse
Dim wsRet As WebRequest.WSInt32
Dim sMyXML As String = ""
Dim iRet As Int32 = 0
UserToken.SetToken(proxy, mUserName, mPassword)
'Get Request XML structure
sMyXML = proxy.GetXMLStructure().Value
'populate sMyXML with data and then pass it to CreateByXML
'
wsRet = proxy.CreateByXML(sMyXML)
```

Related information

"Request" on page 1161

"Request: GetByXML" on page 1177

"Request: GetXMLStructure" on page 1197

"ApiRequest XML" on page 509

Request: Delete

```
Public Function Delete(ByVal sId as String) As WSInt32
```

Purpose

Delete the specified request

Parameters

Parameter	Description
sId	The ID of the request to delete.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

If sId is an empty string, the object's RequestId is used; otherwise, an error is returned.

Because the method "falls back" always ensure the RequestId is passed unless the request (that the object refers to) is the request to be deleted.

Example

```
Dim proxy As New WebRequest.RequestWse
Dim iRet as Int32 = 0

UserToken.SetToken(proxy, mUserName, mPassword)
iRet = proxy.Delete("{f012628c-3717-11d4-8e11-00105a9e2ddf}")
If iRet = 0 Then
   ...'Continue processing
Else
   ...'Handle the error
End If
```

Related information

"Request" on page 1161

Request: DeleteAttachmentById

```
Public Function DeleteAttachmentById(ByVal sAttachmentId as String) As WSInt32
```

Purpose

Delete the specified attachment

Parameters

Parameter	Description
sAttachmentId	The ID of the attachment to delete.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

If sAttachmentId is an empty string, an error is generated.

Example

```
Dim proxy As New WebRequest.RequestWse
Dim iRet as Int32 = 0
Dim ds As New DataSet
Dim sAttachmtName As String = "myAttachment.Doc"
Dim sAttachmtId As String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
iRet = proxy.DeleteAttachmentById(sAttachmtId).Value
...'Continue processing
```

Related information

"Request" on page 1161

Request: DeleteByNumber

```
Public Function DeleteByNumber(ByVal sReqNumber as String) As WSInt32
```

Purpose

Delete the request with the specified RequestNumber

Parameters

Parameter	Description
sReqNumber	Number of the request.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

If sAttachmentId is an empty string an error is generated.

Example

```
Dim proxy As New WebRequest.RequestWse
Dim iRet as Int32 = 0

UserToken.SetToken(proxy, mUserName, mPassword)
iRet = proxy.DeleteByNumber("REQ-008").Value
If iRet <> 0 Then
   ...'Handle error
End If
   ...'Continue processing
```

Related information

"Request" on page 1161

Request: DemandNecessary

```
Public Function DemandNecessary(ByVal sType as String) As WSBoolean
```

Purpose

Is Request demand data required for the stated request type

Parameters

Parameter	Description
sType	Request type

Returns

Type	Description
WSBoolean.	True if request demand data is required. Access the returned Boolean through the .Value property of WSBoolean.

Remarks

None

Example

Not available

Related information

"Request" on page 1161

Request: Exists

```
Public Function Exists(ByVal sId as String) As WSBoolean
```

Purpose

Checks whether the specified RequestId exists in Changepoint.

Parameters

Parameter	Description
sId	RequestId

Returns

Type	Description
WSBoolean	True if the specified request exists. Access the returned Boolean through the .Value property of WSBoolean.

Remarks

The method does not "fall back" and use the object's RequestId if the parameter is an empty string. The method will return false.

Example

Not available

Related information

"Request" on page 1161

Request: GetAssets

```
Public Function GetAssets(ByVal sCompanyId as String, ByVal sInitiatorId as
String) As WSDataSet
```

Purpose

Retrieve all non-deleted Assets where the Asset Assignee is sInitiatorId or sCompanyId

Parameters

Parameter	Description
sCompanyId	ID of the related customer
sInitiatorId	The ID of the Initiator of the request. May be a resource or contact.

Returns

Type	Description
WSDataSet	WSDataSet. (AssetId, AssetNumber, AssetName, type) Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Request" on page 1161

Request: GetAssignees

```
Public Function GetAssignees(ByVal sSearchString as String) As WSDataSet
```

Purpose

Retrieve a list of resources that can be assigned to a request

Parameters

Parameter	Description
sSearchString	The string may be a resource name or resource alternate name. It may also be the name of a queue or just an empty string. The pipe character (|) is used to escape special characters

Returns

Type	Description
WSDataSet	WSDataSet (Id, Name, AlternateName, ResType where ResType = 1 for Resource data or 2 for Queue data) Access the returned dataset through the .Value property of WSDataSet.

Remarks

It is not necessary to have any data in the object itself.

Example

```
Dim proxy As New WebRequest.RequestWse
Dim ds As New DataSet

UserToken.SetToken(proxy, mUserName, mPassword)
'pass in an empty string to get all resources
ds = proxy .GetAssignees("").Value
If ds.Tables.count > 0 AndAlso ds.Tables(0).Rows.Count > 0 Then
   ...'iterate and extract data
End If
```

Related information

"Request" on page 1161

Request: GetAssigneesBySupportDeskId

```
Public Function GetAssigneesBySupportDeskId(ByVal sId As String) As WSDataSet
```

Purpose

Retrieve a list of resources and queue by a support desk that can be assigned to a request

Parameters

Parameter	Description
sId	ID of the support desk.

Returns

Type	Description
WSDataSet	WSDataSet. (Id, Name, AlternateName, Restype where ResType = 1 for Resource data or 2 for Queue data) Access the returned dataset through the .Value property of WSDataSet.

Remarks

It is not necessary to have any data in the object itself.

Example

```
Dim proxy As New WebRequest.RequestWse
Dim ds As New DataSet

UserToken.SetToken(proxy, mUserName, mPassword)
'pass in an empty string to get all resources
ds = proxy. GetAssigneesBySupportDeskId("{0FD91759-8C51-11D3-8AA6-
0060975AEC0F}").Value
If ds.Tables.count > 0 AndAlso ds.Tables(0).Rows.Count > 0 Then
   ...'iterate and extract data
End If
```

Related information

"Request" on page 1161

Request: GetAttachmentById - for WSE only

```
Public Function GetAttachmentById(ByVal sAttachmentId as String) As WSInt32
```

Purpose

Retrieve an attachment from Changepoint

Parameters

Parameter	Description
sAttachmentId	The ID of the attachment required.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The attachment is placed in the ResponseSoapContext.Attachments collection

Example

```
Dim proxy As New WebRequest.RequestWse
Dim iret As Int32 = 0
Dim sret As Byte()

UserToken.SetToken(proxy, mUserName, mPassword)

'Get the attachment
iret = proxy.GetAttachmentById("{B94491E5-F704-4D6B-80D2-18E9E8E12A5F}").value

If proxy.ResponseSoapContext.Attachments.Count = 1 AndAlso iret = 0 Then
   ReDim sret(proxy.ResponseSoapContext.Attachments.Item(0).Stream.Length)
   iret = proxy.ResponseSoapContext.Attachments.Item(0).Stream.Read(sret, 0,
proxy.ResponseSoapContext.Attachments.Item(0).Stream.Length)
End If
...'process sret as required
```

Related information

"Request" on page 1161

RequestUploadDownload: GetAttachmentById - for WCF only

```
Public Function GetAttachmentById(UserName As String, sAttachmentId As String,
out Data As Stream) As WSInt32
```

Purpose

Retrieve an attachment from Changepoint

Parameters

Parameter	Description
Username	Username of logged-in User
sAttachmentId	The ID of the attachment required.
Data	Stream output from the attachment for the Request object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The attachment is placed in the Data parameter as a stream in the method call.

Request: GetAttachments

```
Public Function GetAttachments(ByVal sRequestId as String) As WSDataSet
```

Purpose

Retrieve info on all attachments related to the specified request.

Parameters

Parameter	Description
sRequestId	ID of the request whose attachment data is being retrieved.

Returns

Type	Description
WSDataSet	WSDataSet. (AttachmentId, Name, Description, Shared, FileName) Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Request" on page 1161

Request: GetById

```
Public Function GetById(ByVal sId as String) As WSRequest
```

Purpose

Retrieve current request data into the object

Parameters

Parameter	Description
sId	The ID of the request whose data is being retrieved.

Returns

Type	Description
WSRequest	Returns an ApiRequest object. Access the object through the .Value property of WSRequest.

Remarks

None

Example

Not available

Related information

"Request" on page 1161

Request: GetByNumber

```
Public Function GetByNumber(ByVal sNumber as String) As WSDataSet
```

Purpose

Retrieve current request data based on the request number passed.

Parameters

Parameter	Description
sNumber	Request number of the request required.

Returns

Type	Description
WSDataSet	WSDataSet. The returned dataset contains the same fields as those returned in GetById. Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Request" on page 1161

Request: GetByXML

```
Public Function GetByXML(ByVal sXML as String, ByVal sRequestId as String) As
WSString
```

Purpose

Retrieve current request data based on the request number passed and the fields in the XML string.

Parameters

Parameter	Description
sXML	The XML string containing fields where data is required.
sRequestId	The request whose data is being retrieved.

Returns

Type	Description
WSString	XML string of request data. Access the returned string through the .Value property of WSString.

Remarks

The XML string sXML can be varied in the number of fields. If only a small subset of data is required, any unwanted fields can be removed from sXML. The returned XML string will mirror sXML in the fields contained in the XML.

Example

Not available

Related information

"Request" on page 1161

Request: GetCategories

```
Public Function GetCategories(ByVal sRequestTypeId as String) As WSDataSet
```

Purpose

Retrieve all request categories for a specified request type.

Parameters

Parameter	Description
sRequestTypeId	Request type

Returns

Type	Description
WSDataSet	The returned dataset contains the fields (Code, Description), where Code is the unique Identifier for the category. Access the returned dataset through the .Value property of WSDataSet.

Remarks

sRequestType is always a string of 3 characters. Anything else will create an error.

Example

Not available

Related information

"Request" on page 1161

Request: GetCompanies

```
Public Function GetCompanies(ByVal sSearchString as String, ByVal sInitiatorId
as String) As WSDataSet
```

Purpose

Retrieve all customers based on sSearchString and sInitiatorId

Parameters

Parameter	Description
sSearchString	The search string can be a customer Name, AlternateName or UserDefinedCustomerId the pipe character(|) can be used to escape a special character.
sInitiatorId	Request initiator

Returns

Type	Description
WSDataSet	WSDataSet (CustomerId, Name, AlternateName, UserDefinedId). Access the returned dataset through the .Value property of WSDataSet.

Remarks

The customers returned are based on the search string but also on whether the initiator created the Customer record or has access to the customer.

Similarly customer records will also be returned if the initiator has access to contracts under the specified customer or is the creator of any contracts under the specified customer.

Example

Not available

Related information

"Request" on page 1161

Request: GetEngagements

```
Public Function GetEngagements(ByVal sCompanyId as String) As WSDataSet
```

Purpose

Retrieve all Engagements under a specified customer.

Parameters

Parameter	Description
sCompanyId	The customer whose contracts are required

Returns

Type	Description
WSDataSet	WSDataSet. (EngagementId, Name, AlternateName). Access the returned dataset through the .Value property of WSDataSet.

Remarks

Returns all contracts under the specified customer where the logged in user has access to the contract.

Example

Not available

Related information

"Request" on page 1161

Request: GetInitiatorName

```
Public Function GetInitiatorName(ByVal sInitiatorId as String) As WSDataSet
```

Purpose

Retrieve the name of the specified initiator.

Parameters

Parameter	Description
sInitiatorId	Resource ID or contact ID whose name is required.

Returns

Type	Description
WSDataSet	WSDataSet. (Name, AlternateName). Access the returned dataset through the .Value property of WSDataSet.

Remarks

Returns the initiator name whether the initiator is a resource or contact.

Example

Not available

Related information

"Request" on page 1161

Request: GetInitiators

```
Public Function GetInitiators(ByVal sSearchString as String) As WSDataSet
```

Purpose

Retrieve a list of initiators based on the specified search string.

Parameters

Parameter	Description
sSearchString	Search string. May contain a resource or contact name or alternate name. The string can use the pipe character (|) to escape special characters.

Returns

Type	Description
WSDataSet	WSDataSet (Name, AlternateName, Id, ResType) where ResType = 0 means the ID is a Changepoint resource and resType = 1 means a Changepoint Contact. Access the returned dataset through the .Value property of WSDataSet.

Remarks

Returns the initiator name whether the initiator is a resource or contact.

Example

Not available

Related information

"Request" on page 1161

Request: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As WSString
```

Purpose

Returns the RequestId based on the UDF Text field and value.

Parameters

Parameter	Description
sUDFField	UDF text field name (for example, Text1)
sUDFValue	UDF text field value

Returns

Type	Description
WSString	String with the RequestId or an empty string. Access the returned string through the .Value property of WSString

Remarks

None

Example

Not available

Related information

"Request" on page 1161

Request: GetList

```
Public Function GetList(ByVal iRetRows as Int16) As WSDataSet
```

Purpose

Retrieve a list of requests based on the parameter iRetRows.

Parameters

Parameter	Description
iRetRows	The number of records to be returned. To return all, you must pass in -1.

Returns

Type	Description
WSDataSet	WSDataSet (RequestId, EngagementId, RequestNumber, RequestType, RequestStatus) Access the returned dataset through the .Value property of WSDataSet.

Remarks

The records are ordered based on the RequestNumber

Example

Not available

Related information

"Request" on page 1161

Request: GetPriorities

```
Public Function GetPriorities(ByVal sEngagementId as String, ByVal sProductId
as String) As WSDataSet
```

Purpose

Retrieve a list of priorities based on the parameters for EngagementId and ProductId

Parameters

Parameter	Description
sEngagementId	The engagement
sProductId	The Product

Returns

Type	Description
WSDataSet	WSDataSet (Code, Description, NoSLA) Access the returned dataset through the .Value property of WSDataSet.

Remarks

The method does make use of the request type, and it takes its value from the object itself. Therefore, if this call is being made from an empty object, set the Request type property.

Example

Not available

Related information

"Request" on page 1161

Request: GetProductNameById

```
Public Function GetProductNameById(ByVal sProductId as String) As WSString
```

Purpose

Retrieve the name of the product Identified by the ProductId

Parameters

Parameter	Description
sProductId	The product whose name is required.

Returns

Type	Description
WSString	Product name as a string. Access the returned string through the .Value property of WSString.

Remarks

No name will be returned if the product has been deleted.

Example

Not available

Related information

"Request" on page 1161

Request: GetProducts

```
Public Function GetProducts(ByVal sProductId as String) As WSDataSet
```

Purpose

Retrieve the name of the product Identified by the ProductId

Parameters

Parameter	Description
sProductId	The product whose name is required.

Returns

Type	Description
WSDataSet	WSDataSet (ProductId, Name, Type). Access the returned dataset through the .Value property of WSDataSet.

Remarks

Type = 1 ignores any product open or close dates. Type = 2 indicates a product that has open and close dates and the date of the query places the product within the open date time frame.

Example

Not available

Related information

"Request" on page 1161

Request: GetProjects

```
Public Function GetProjects(ByVal sEngagementId as String, ByVal sSearchString
as String) As WSDataSet
```

Purpose

Retrieve all projects under sEngagementId and meeting the criteria in sSearchString

Parameters

Parameter	Description
sEngagementId	The contract whose projects are being searched.
sSearchString	Can be any project name or alternate name. The pipe character (|) is used to escape special characters.

Returns

Type	Description
WSDataSet	WSDataSet. (ProjectId, Name, AlternateName). Access the returned dataset through the .Value property of WSDataSet.

Remarks

The logged-in Changepoint user also acts to filter returned projects. The logged-in user must be a manager or co-manager on the project, or a resource who is assigned to a task within the

project, or a resource who has view access to the project.

Example

Not available

Related information

"Request" on page 1161

Request: GetRDSetInfo

```
Public Function GetRDSetInfo(ByVal sReqType as String) As WSDataSet
```

Purpose

Retrieve the two switches for resource demand that control whether demand is necessary and whether demand data will be viewable from Request in Enterprise.

Parameters

Parameter	Description
sReqType	Request type whose settings are being extracted.

Returns

Type	Description
WSDataSet	WSDataSet (EnableResDemandInfo, ResDemandInfoMandatory) . Access the returned dataset through the .Value property of WSDataSet.

Remarks

EnableResDemandInfo indicates whether resource demand info is visible in the Request interface in Changepoint Enterprise, while ResDemandInfoMandatory indicates that Resource Demand must be included for any new requests.

Example

Not available

Related information

"Request" on page 1161

Request: GetRequests

```
Public Function GetRequests(ByVal sCustomerName as String,
 ByVal sCustomerId as String, ByVal sEngagementName as String,
 ByVal sEngagementId as String, ByVal sProjectName as String,
 ByVal sProjectId as String, ByVal sRequestType as String,
 ByVal sPriorityId as String, ByVal sCategoryId as String,
 ByVal sSubCategoryId as String) As WSDataSet
```

Purpose

Retrieve all requests based on the values in the parameters.

Parameters

Parameter	Description
sCustomerName	Name of the customer. If multiple customer names are passed in, each name must be separated by a pipe character (|).
sCustomerId	ID of the customer. If multiple customer IDs are passed in, each ID must be separated by a pipe character (|).
sEngagementName	Name of the contract. If multiple contract names are passed in, each name must be separated by a pipe character (|).
sEngagementId	ID of the contract. If multiple contract Ids are passed in, each ID must be separated by a pipe character (|).
sProjectName	Name of the project. If multiple project names are passed in, each name must be separated by a pipe character (|).
sProjectId	ID of the project. If multiple project Ids are passed in, each ID must be separated by a pipe character (|).
sRequestType	The request type. If multiple request types are passed in, each type must be separated by a pipe character (|).
sPriorityId	The request priority. If multiple priority Ids are passed in, each customer name must be separated by a pipe character (|).

Parameter	Description
sCategoryId	The request category. If multiple category IDs are passed in, each ID must be separated by a pipe character (|).
sSubCategoryId	The request subcategory. If multiple subcategory Ids are passed in, each ID must be separated by a pipe character (|).

Returns

Type	Description
WSDataSet	WSDataSet. (RequestId, RequestNum). Access the returned dataset through the .Value property of WSDataSet.

Remarks

Criteria for the search can be any combination of parameter values. Always keep in mind the relationship between parameter fields and their effect on requests.

Example

```
Not available
```

Related information

"Request" on page 1161

Request: GetResponsibles

```
Public Function GetResponsibles(ByVal sSearchString as String) As WSDataSet
```

Purpose

Retrieve all resources that can be assigned to a request.

Parameters

Parameter	Description
sSearchString	The search string can contain a resource name or alternate name. Any special characters can be escaped using the pipe character (|) in front of the special character.

Returns

Type	Description
WSDataSet	WSDataSet (ResourceId, Name, AlternateName). Access the returned dataset through the .Value property of WSDataSet.

Remarks

Resources must belong to a Workgroup before they can be assigned to requests or tasks.

Example

Not available

Related information

"Request" on page 1161

Request: GetStatuses

```
Public Function GetStatuses(ByVal sRequestId As String, ByVal sActionResource
As String) As WSDataSet
```

Purpose

Retrieve all possible status values based on the request and the action resource, as determined by workflow.

Parameters

Parameter	Description
sSearchString	The request type whose workflow statuses are being extracted.

Returns

Type	Description
WSDataSet	WSDataSet. (Code, Description). Access the returned dataset through the .Value property of WSDataSet.

Remarks

Returned status values are based on sActionResource and their access to different statuses as determined by Workflow, the Request type and the current state of sRequestId in the Workflow.

Requests that are not attached to any Workflow will cause this method to return the entire status list.

Example

Not available

Related information

"Request" on page 1161

Request: GetSubCategories

```
Public Function GetSubCategories(ByVal sRequestCategoryId As String) As
WSDataSet
```

Purpose

Retrieve all subcategories for the specified request category that are not deleted from the system.

Parameters

Parameter	Description
sRequestCategoryId	The request category

Returns

Type	Description
WSDataSet	WSDataSet (Code, Description). Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Request" on page 1161

Request: GetSupportDesks

```
Public Function GetSupportDesks(ByVal sEngagementId As String, ByRef
sDefaultId As String, ByRef iAllowOverride As Int32) As WSDataSet
```

Purpose

Retrieve a list of all available Support Desks. Also return the specified default Support Desk and the value for the Support Desk Override switch.

Parameters

Parameter	Description
sEngagementId	The Engagement whose available Support Desks are to be viewed.
sDefaultId	Returns parameter. The Default Support Desk
iAllowOverride	Returns parameter. The Support Desk Override switch.

Returns

Type	Description
WSDataSet	WSDataSet (HelpDeskId, AllowHDOverride). Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Request" on page 1161

Request: GetTypes

```
Public Function GetTypes (ByVal sEngagementId As String) As WSDataSet
```

Purpose

Retrieve a list of all request types in Changepoint.

Parameters

Parameter	Description
sEngagementId	Engagement ID. An empty string is allowed.

Returns

Type	Description
WSDataSet	WSDataSet (Code, Description, AvailToGuest, AvailToBudgetContingency, AvailToBudgetItems, EnableResDemandInfo, ResDemandInfoMandatory). Access the returned dataset through the .Value property of WSDataSet.

Remarks

If sEngagementId = "", then returns all request types, otherwise returns request types available to this contract and login user.

The returned dataset also includes several switch settings that set access to the type and whether resource demand data is necessary for new requests of that type.

Example

Not available

Related information

"Request" on page 1161

Request: GetUDF

```
Public Function GetUDF (ByVal sRequestId As String, ByVal sReqType As String,
ByVal UDFOption As CPUDFReturnType) As WSString
```

Purpose

Retrieve the UDF XML string for the request.

Parameters

Parameter	Description
sRequestId	The request ID
sReqType	The request type
UDFOption	The CPUDFReturnType Values are: • OnlyStructure - return only UDF XML structure, no field data • OnlyStructureAndOptions - return UDF structure and option data without field data • OnlyValues - return only UDF fields with data, do not include any field structure or options • WithStructure – return UDF field data with full field structure • WithStructureAndOptions – return UDF field data with full structure and all field options

Returns

Type	Description
WSString	A UDF XML string that varies in detail based on the value of the parameter UDFOption. Access the returned string through the .Value property of WSString.

Remarks

The method uses UDFOption, RequestId and the Request Type to determine the contents of the UDF XML string returned.

Example

Not available

Related information

"Request" on page 1161

Request: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions (ByVal sReqType As String, ByVal sCodeName
As String, ByVal sSearchString As String) As WSString
```

Purpose

Retrieve the UDF code options as a XML string based on sCodeName or sSearchString.

Parameters

Parameter	Description
sReqType	The request type
sCodeName	The name of the UDF code whose options are required, for example, "Code1", Text6
sSearchString	A search string used to extract one or more UDF Codes.

Returns

Type	Description
WSString	A UDF XML string that varies in detail based on the value of the parameter UDFOption. Access the returned string through the .Value property of WSString.

Remarks

None

Example

Not available

Related information

"Request" on page 1161

Request: GetUDFXMLStructure

```
Public Function GetUDFXMLStructure(ByVal sReqTypeId as String) As WSString
```

Purpose

Retrieve the UDF XML structure for the specified type sReqTypeId.

Parameters

Parameter	Description
sReqTypeId	The request type whose UDF XML structure is required.

Returns

Type	Description
WSString	A UDF XML string of the UDF structure for sReqTypeId. Access the returned string through the .Value property of WSString.

Remarks

None

Example

Not available

Related information

"Request" on page 1161

Request: GetXMLStructure

```
Public Function GetXMLStructure()As WSString
```

Purpose

Retrieve the XML structure for the request object.

Parameters

None

Returns

Type	Description
WSString	An XML string of the Request object. Access the returned string through the .Value property of WSString.

Remarks

None

Example

Not available

Related information

"Request" on page 1161

"ApiRequest XML" on page 509

Request: New

```
Public Function New()
```

Purpose

Initialize a new request object.

Parameters

None

Returns

Not applicable

Remarks

None

Example

Not available

Related information

"Request" on page 1161

Request: SaveUDF

```
Public Function SaveUDF(ByVal sXML as String, ByVal bypassMetadata as
CPMetadataCheck) As WSInt32
```

Purpose

Update the request UDF (configurable field) data based on the parameter sXML.

Parameters

Parameter	Description
sXML	UDF XML string with data.
bypassMetadata	Determines the level of metadata checking for sXMLUDF. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The RequestId and Request Type are first taken from the object. If there are no values then the sXML is accessed for these values. An error is thrown if the RequestId and Request Type still are not present.

The RequestId and the Request Type are required values for the save process.

Example

Not available

Related information

"Request" on page 1161

Request: SetPropertiesByXML

```
Public Function SetPropertiesByXML(ByVal sXML as String, ByVal bNotInitialize
as Boolean) As WSInt32
```

Purpose

Set the properties of the object with the values in the fields of sXML.

Parameters

Parameter	Description
sXML	Request XML string with data to load into the object.
bNotInitialize	If False the object is re-initialized. It is recommended to always set this switch to True.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods. If you use the SetPropertiesByXML method, you must pass in the entire XML string.

The minimal XML that can be sent in to the method is:

```
<root>
   <request></request>
</root>
```

In the case of an empty Request object and sXML has a value for RequestId, the object will first retrieve existing data for the RequestId in sXML and then replace existing field values with new ones from sXML.

If sXML contains fields with no values, the method will overwrite the existing field value and replace with the "empty" value from sXML, basically erasing the field.

When the original field value is to be retained, the field in sXML must be removed or its value must be the same as the current data held in the object.

Example

Not available

Related information

"Request" on page 1161

"ApiRequest XML" on page 509

"Request: CreateByXML" on page 1164

"Request: UpdateByXML" on page 1203

Request: Update

```
Public Function Update() As WSInt32
```

Purpose

Update the database with data held in the objects properties.

Parameters

None

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The update process will update all fields in Request.

The ByPassMetadataCheck switch will stop any metadata validation in Request and also in Request UDFs

It is not recommended to set the ByPassWorkflow switch for a request using Workflow. Only set this switch where Workflow is not expected to be active for the Request.

Example

```
Dim proxy As New WebRequest.RequestWse
Dim myRequest as New WebRequest.ApiRequest
Dim iRet as Int32 = 0
Dim sNewId as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
'Get latest data into the object
myRequest = proxy.GetById("{f012868e-3717-11d4-8e11-00105a9e2ddf }").Value
'Make changes
With myRequest
   'Set property values as necessary
   Dim sId as New WebRequest.Identity
   sId.Id = "{bff37287-abbf-483f-8f0b-da17a8f24ce5 }"
   .Responsible = sId
   .ProductCost = 1200.00
   sId = New WebRequest.Identity
   sId = "{ddbd4f63-363b-48f4-9ee8-05a852ebfd5a }"
   .Product = sId
   .Quantity = 5
   sId = New WebRequest.Identity
   sId = "{4fb06751-9a3b-42f5-801a-73032b97edff }"
   .Assignment = sId
   .EstimatedHours = 100
   ...
End With
iRet = proxy.Update(myRequest).Value
```

```
...'Continue processing
```

Related information

"Request" on page 1161

Request: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, ByVal sRequestId As String)
As WSInt32
```

Purpose

Update the database using an XML string of the Request object containing update info.

Parameters

Parameter	Description
sXML	Request XML string with updated fields.
sRequestId	Request being updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The ApiRequest XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The update process will update all fields in Request.

The ByPassMetadataCheck switch will stop any meta data validation in Request and also in Request UDFs.

It is not recommended to set the ByPassWorkflow switch for a request using workflow. Only set this switch when workflow is not expected to be active for the Request.

It is recommended to remove the XML tags if you want to retain the original data in the database.

The method uses the following sequence to find the request ID:

1. If the sRequestId parameter is passed in, the method uses this value for the request ID.
2. If this fails, the method attempts to extract the request ID from <requestid> in the XML.
3. If this fails, the request ID is taken from the object properties.
4. If this fails, an attempt is made to look up the request ID using <requestnumber> in the XML.

Example

```
Dim proxy As New WebRequest.RequestWse
Dim sMyXML As String = ""
Dim wsRet As WebRequest.WSInt32
Dim sReqId As String = ""
sReqId = "{CE26F116-8FCF-4697-A0BF-00118BC3059A}"
UserToken.SetToken(proxy, mUserName, mPassword)
'Get Request XML structure with existing data if it's necessary.
sMyXML = proxy.GetByXML(sReqId).Value
'modify sMyXML and then pass it to UpdateByXML
'
wsRet = proxy.UpdateByXML(sMyXML)
```

Related information

"Request" on page 1161

"Request: GetByXML" on page 1177

"Request: GetXMLStructure" on page 1197

"ApiRequest XML" on page 509

Request: UploadAttachment - for WSE only

```
Public Function UploadAttachment(ByVal sRequestId As String, ByVal sName As
String, ByVal sDescription As String, ByVal sFileName As String, ByVal iShared
As Int32) As WSString
```

Purpose

Upload an attachement for a request that has an ID value that matches sRequestId.

Parameters

Parameter	Description
sRequestId	Request to which the attachment belongs.
sName	Name of the attachment
sDescription	Description of the attachment
sFileName	Filename of the attachment. The path to the file is part of sFileName
iShared	Switch indicating whether the attachment can be shared.

Returns

Type	Description
WSString	The returned string is the new AttachmentId Access the returned string through the .Value property of WSString.

Remarks

None

Example

```
Dim sMemStream As MemoryStream
Dim myDimeAttachment As DimeAttachment
Dim sRetval As String = ""
Dim sReqId As String = ""
Dim proxy As New WebRequest.RequestWse

UserToken.SetToken(proxy, mUserName, mPassword)

sReqId = "{BDBFEA1F-D881-44E5-9B04-49AF2A20F699}"

'Create the attachment
myDimeAttachment = New DimeAttachment("application/octet-stream",
TypeFormat.MediaType, "C:\ MyAttachment.doc")
proxy.RequestSoapContext.Attachments.Add(myDimeAttachment)
```

```
'Upload to Changepoint
```

sRetval = proxy.UploadAttachment(sReqId, " MyAttachment Word Doc", "Word Doc", " MyAttachment.doc", 1).Value

Related information

"Request" on page 1161

RequestUploadDownload: UploadAttachment - for WCF only

```
Public Function UploadAttachment(UserName As String, iShared As Int, sDesc As String, sFileName As String, sName As String, sRequestId As String, Data As Stream) AS WSString
```

Purpose

Upload an attachement for a request that has an ID value that matches sRequestId.

Parameters

Parameter	Description
Username	Username of logged-in User
iShared	Switch indicating whether the attachment can be shared.
sDesc	Description of the attachment
sFileName	Filename of the attachment. The path to the file is part of sFileName
sName	Name of the attachment
sRequestId	Request to which the attachment belongs.
Data	Stream input for the attachment to the Request object

Returns

Type	Description
WSString	The returned string is the new AttachmentId Access the returned string through the .Value property of WSString.

Remarks

The attachment must be passed in through the Data parameter as a stream in the method call.

RequestDemand

The RequestDemand object is a subobject of the Request object and reflects a request as a demand element.

Namespace

`http://changepoint.com/changepoint/CPWebService/Request`

URL

`http://webserver/CPWebService/Objects/CPRequest/RequestDemand.asmx`

Methods

Note: SaveRequestDemand has been removed as a separate method and the functionality is now part of Request.Add and Request.Update. Please fill in the Request.RequestDemand property as appropriate.

Properties

For more information, see the "ApiRequestDemand" section on page 553.

Related information

"Request" on page 1161

"Request: Add" on page 1163

"Request: Update" on page 1201

"ApiRequestDemand XML" on page 554

RequestDemand: GetList

```
Public Function GetList(ByVal iRetRows As Int16) As WSDataset
```

Purpose

Retrieves request demand records.

Parameters

Parameter	Description
iRetRows	Number of records to be returned. To return all, you must pass in -1.

Returns

Type	Description
WSDataSet	WSDataSet of request demand records, or WSDataSet object upon success and nothing upon error. Returns all rows when method called with parameter -1. Access the dataset through the .Value property of WSDataSet.

Remarks

Check log file upon error.

Example

```
Dim myProxy As New webRequestDemand.RequestDemandWse
Dim oRet As webRequestDemand.WSDataset
UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetList(500)
```

RequestDemand: GetByRequestId

```
Public Function GetByRequestId(ByVal sRequestId As String) As WSString
```

Purpose

Retrieves request demand records for a request

Parameters

Parameter	Description
sRequestId	The ID of the Request

Returns

Type	Description
WSString	Returns XML string of request demand records for the specific request in the property value of the WSString object. Returns request demand XML schema only when the method is called with an empty string or the passed request has no request demand records. Access the returned string through the .Value property of WSString.

Remarks

Check log file upon error.

Example

```
Dim myProxy As New webRequestDemand.RequestDemandWse
Dim oRet As webRequestDemand.WSString
UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetByRequestId("{FF636BB5-B108-4742-BA1D-E7D7BA9CDF3D}")
```

Related information

"ApiRequestDemand XML" on page 554

RequestTime

The RequestTime object allows users to add, retrieve, update, approve or reject request time information and submit request time, time, and standard time information for resources within the Changepoint database.

Namespace

```
http://changepoint.com/changepoint/CPWebService/RequestTime
```

URL

```
http://webserver/CPWebService/Objects/CPRequestTime/RequestTime.asmx
```

Methods

Properties

For more information, see the "ApiRequestTime" section on page 558.

RequestTime: Add

```
Public Function Add(ByRef sId As String, ByVal oRequestTime As ApiRequestTime)
As WSInt32
```

Purpose

Insert a new request time record.

Parameters

Parameter	Description
sId	ID of the request time record.
oRequestTime	Populated ApiRequestTime object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check log file upon error.

Example

```
Dim myProxy As New webRequestTime.RequestTimeWse
Dim myRequestTime As New webRequestTime.ApiRequestTime
Dim oRet As webRequestTime.WSInt32
Dim sId As String = String.Empty
UserToken.SetToken(myProxy, mUserName, mPassword)
With myRequestTime
   .Request = New webRequestTime.Identity
   .Request.Id = "{DFAB2126-BE7E-462D-813A-24CE67871964}"
   .Resource = New webRequestTime.Identity
   .Resource.Id = "{BE19F806-03FD-4DB3-B105-ECFD5E2F9195}"
   .Description = "Unit Testing"
   .RegularHours = 8
   .OvertimeHours = 2
   .StartTime = CDate("05/18/2007 9:00:00 AM")
   .TimeZone = "EST"
   .WorkLocationGroup = New webRequestTime.Identity
   .WorkLocationGroup.Id = "{6AAF1A2D-2535-40E3-8D03-07801AE85E41}"
   .WorkLocation = New webRequestTime.Identity
   .WorkLocation.Id = "{0175FA25-7CAA-4DC3-BB95-62A37A36134F}"
   .WorkCodeCategory = New webRequestTime.Identity
   .WorkCodeCategory.Id = "{AA9C83C1-6E1E-4974-9DEC-7C554CC429D2}"
   .WorkCode = New webRequestTime.Identity
   .WorkCode.Id = "{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}"
End With
oRet = myProxy.Add(sId, myRequestTime)
```

Related information

"RequestTime" on page 1209

RequestTime: ApproveOrReject

```
Public Function ApproveOrReject(ByVal sResourceId As String, ByVal sRequestId
As String, ByVal daStartDate As Date, ByVal daEndDate As Date, ByVal bAction
As Boolean, ByVal sReason As String) As WSInt32
```

Purpose

Approve or reject all request time records whose StartTime falls between the dates specified in the daStartDate and daEndDate parameters

Parameters

Parameter	Description
sResourceId	ID of the resource.
sRequestId	ID of the request.
daStartDate	Start date of the period.
daEndDate	End date of the period.
bAction	0 for rejection and 1 for approval.
sReason	Rejected reason. • String value required when bAction=0 (False) • Empty string when bAction=1 (True)

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Either sResourceId and sRequestId can be empty string:

- Approve or reject request time for a specified resource in sResourceId when sRequestId is an empty string.
- Approve or reject request time for a specified request in sRequestId when sResourceId is an empty string.

Check log file upon error.

Example

```
Dim myProxy As New webRequestTime.RequestTimeWse
Dim oRet As webRequestTime.WSInt32

UserToken.SetToken(myProxy, mUserName, mPassword)

oRet = myProxy.ApproveOrReject("{BE19F806-03FD-4DB3-B105-ECFD5E2F9195}", "",
"05/18/2009", "05/18/2009", True, "")
```

Related information

"RequestTime" on page 1209

RequestTime: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, ByRef sId As String) As
WSInt32
```

Purpose

Create RequestTime using an XML string of the RequestTime object in Changepoint.

Parameters

Parameter	Description
sXML	A new RequestTime XML string.
sId	ByRef parameter that returns the new RequestTimeId after the RequestTime has been added.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

ApiRequestTime XML structure can be obtained by GetXMLStructure or GetByXML methods.

The ByPassMetadataCheck switch will stop any meta data validation in RequestTime.

Example

```
Dim proxy As New WebRequestTime.RequestTimeWse
Dim wsRet As WebRequestTime.WSInt32
Dim sMyXML As String = ""
Dim iRet As Int32 = 0
UserToken.SetToken(proxy, mUserName, mPassword)
'Get RequestTime XML structure
sMyXML = proxy.GetXMLStructure().Value
'populate sMyXML with data and then pass it to CreateByXML
'
wsRet = proxy.CreateByXML(sMyXML)
```

Related information

"RequestTime" on page 1209

"RequestTime: GetByXML" on page 1219

"RequestTime: GetXMLStructure" on page 1222

"ApiRequestTime XML" on page 576.

RequestTime: Delete

```
Public Function Delete(ByVal sId As String) As WSInt32
```

Purpose

Deletes a request time record.

Parameters

Parameter	Description
sId	ID of the request time record.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check log file upon error.

Example

```
Dim myProxy As New webRequestTime.RequestTimeWse
Dim oRet As webRequestTime.WSInt32

UserToken.SetToken(myProxy, mUserName, mPassword)

oRet = myProxy.Delete("{FF636BB5-B108-4742-BA1D-E7D7BA9CDF3D}")
```

Related information

"RequestTime" on page 1209

RequestTime: Exists

```
Public Function Exists(ByVal sId As String) As WSBoolean
```

Purpose

Check whether this request time exists or not

Parameters

Parameter	Description
sId	The ID of the request time record.

Returns

Type	Description
WSBoolean	True if the request time exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check log file upon error

Example

```
Dim myProxy As New webRequestTime.RequestTimeWse
Dim oRet As webRequestTime.WSBoolean

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.Exists("{1F8DB2F7-35CE-4C1D-84C7-D234F1E2E5EC}")
```

Related information

"RequestTime" on page 1209

RequestTime: GetById

```
Public Function GetById(ByVal sId As String) As WSRequestTime
```

Purpose

Retrieves a request time record.

Parameters

Parameter	Description
sId	The ID of the request time record.

Returns

Type	Description
WSRequestTime	Returns an ApiRequestTime object. Access the object through the .Value property of WSRequestTime.

Remarks

Check log file upon error.

Example

```
Dim myProxy As New webRequestTime.RequestTimeWse
Dim oRet As webRequestTime.WSRequestTime

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetById("{1F8DB2F7-35CE-4C1D-84C7-D234F1E2E5EC}")
```

Related information

"RequestTime" on page 1209

RequestTime: GetByRequest

```
Public Function GetByRequest(ByVal sRequestId As String, ByVal daStartDate As
Date, ByVal daEndDate As Date) As WSDataset
```

Purpose

Retrieves all request time records whose start time falls between the dates specified in the daStartDate and daEndDate parameters for the target request.

Parameters

Parameter	Description
sRequestId	ID of the target request
daStartDate	Start date of the period
daEndDate	End date of the period

Returns

Type	Description
WSDataSet	WSDataset of request time records. Access the dataset through the .Value property of WSDataSet.

Remarks

Check log file upon error.

Example

```
Dim myProxy As New webRequestTime.RequestTimeWse
Dim oRet As webRequestTime.WSDataset

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetByRequest("{DFAB2126-BE7E-462D-813A-24CE67871964}",
"05/14/2007", "05/18/2007")
```

Related information

"RequestTime" on page 1209

RequestTime: GetByRes

```
Public Function GetByRes(ByVal sResourceId As String, ByVal daStartDate As
Date, ByVal daEndDate As Date) As WSDataset
```

Purpose

Retrieves all the request time records, where the start time falls between the dates specified in the daStartDate and daEndDate parameters for the target resource.

Parameters

Parameter	Description
sResourceId	ID of the target resource.
daStartDate	Start date of the period.
daEndDate	End date of the period

Returns

Type	Description
WSDataSet	WSDataset of request time records in the specifed period. Access the dataset through the .Value property of WSDataSet.

Remarks

Check log file upon error

Example

```
Dim myProxy As New webRequestTime.RequestTimeWse
Dim oRet As webRequestTime.WSDataset

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetByRes("{BE19F806-03FD-4DB3-B105-ECFD5E2F9195}",
"05/14/2007", "05/18/2007")
```

Related information

"RequestTime" on page 1209

RequestTime: GetByXML

```
Public Function GetByXML(ByVal sXML as String, ByVal sRequestTimeId as String)
As WSString
```

Purpose

Retrieve current RequestTime data based on the RequestTime ID passed and the fields in the XML string.

Parameters

Parameter	Description
sXML	The XML string containing fields where data is required.
sRequestTimeId	The RequestTime whose data is being retrieved.

Returns

Type	Description
WSString	XML string of RequestTime data. Access the returned string through the .Value property of WSString.

Remarks

The XML string sXML can be varied in the number of fields. If only a small subset of data is required, any unwanted fields can be removed from sXML. The returned XML string will mirror sXML in the fields contained in the XML.

Example

Not available

Related information

"RequestTime" on page 1209

RequestTime: GetIdsWithinPeriod

```
Public Function GetIdsWithinPeriod(ByVal daStartDate As Date, ByVal daEndDate
As Date) As WSDataset
```

Purpose

Retrieves all the request time IDs from request time records, where the start time falls between the dates specified in the daStartDate and daEndDate parameters.

Parameters

Parameter	Description
daStartDate	Start date of the period.
daEndDate	End date of the period.

Returns

Type	Description
WSDataSet	WSDataset of request time IDs in the specified period. Access the dataset through the .Value property of WSDataSet.

Remarks

Check log file upon error

Example

```
Dim myProxy As New webRequestTime.RequestTimeWse
Dim oRet As webRequestTime.WSDataset

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetIdsWithinPeriod("5/14/2007","5/18/2007")
```

Related information

"RequestTime" on page 1209

RequestTime: GetList

```
Public Function GetList(ByVal iRetRows As Int16) As WSDataset
```

Purpose

Retrieves request time records.

Parameters

Parameter	Description
iRetRows	Number of records to be returned. To return all, you must pass in -1.

Returns

Type	Description
WSDataSet	WSDataset (RequestId, RequestNum, CustomerId, EngagementId, ResourceId, RName, RequestTimeId). Access the dataset through the .Value property of WSDataSet.

Remarks

Check log file upon error

Example

```
Dim myProxy As New webRequestTime.RequestTimeWse
Dim oRet As webRequestTime.WSDataset

UserToken.SetToken(myProxy, mUserName, mPassword)
oRet = myProxy.GetList (-1)
```

Related information

"RequestTime" on page 1209

RequestTime: GetXMLStructure

```
Public Function GetXMLStructure()As WSString
```

Purpose

Retrieve the XML structure for the RequestTime object.

Parameters

None

Returns

Type	Description
WSString	An XML string of the RequestTime object. Access the returned string through the .Value property of WSString.

Remarks

None

Example

Not available

Related information

"RequestTime" on page 1209

"ApiRequestTime XML" on page 576

RequestTime: Submit

```
Public Function Submit(ByVal sResourceId As String, ByVal daStartDate As Date,
ByVal daEndDate As Date) As WSInt32
```

Purpose

Submit time, request time and standard time where the start time falls between the dates specified in the daStartDate and daEndDate parameters for a target resource.

Parameters

Parameter	Description
sResourceId	ID of the target resource.
daStartDate	Start date of the period
daEndDate	End date of the period

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check log file upon error

Example

```
Dim myProxy As New webRequestTime.RequestTimeWse
Dim oRet As webRequestTime.WSInt32

UserToken.SetToken(myProxy, mUserName, mPassword)

oRet = myProxy.Submit("{BAA411CF-F648-47CD-A4F1-12C259EE25BF}",
"5/14/2007","5/18/2007")
```

Related information

"RequestTime" on page 1209

RequestTime: Update

```
Public Function Update(ByVal oRequestTime As ApiRequestTime) As WSInt32
```

Purpose

Update a request time record.

Parameters

Parameter	Description
oRequestTime	Populated ApiRequestTime object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

- Ensure all mandatory properties are set.

- Get the requesttime record to beupdated by using the GetById method
- Set the HistoryComments property when FedAudit flag of the Engagement is on
- Check the log file upon error

Example

```
Dim myProxy As New webRequestTime.RequestTimeWse
Dim myRequestTime As New webRequestTime.ApiRequestTime
Dim oRet As webRequestTime.WSInt32
UserToken.SetToken(myProxy, mUserName, mPassword)
myRequestTime = myProxy.GetById("{1F8DB2F7-35CE-4C1D-84C7-
D234F1E2E5EC}").value
'Update Description, OvertimeHours, WorkLocationgroup, WorkLocation
With myRequestTime
   .Description = "Modified through WS API"
   .OvertimeHours = 4
   .WorkLocationGroup = New webRequestTime.Identity()
   .WorkLocationGroup.Id = "{6AAF1A2D-2535-40E3-8D03-07801AE85E41}"
   .WorkLocation = New webRequestTime.Identity()
   .WorkLocation.Id = "{0175FA25-7CAA-4DC3-BB95-62A37A36134F}"
End With
oRet = myProxy.Update(myRequestTime)
```

Related information

"RequestTime" on page 1209

RequestTime: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, ByVal sId As String) As
WSInt32
```

Purpose

Update the database using an XML string of the RequestTime object containing update info.

Parameters

Parameter	Description
sXML	RequestTime XML string with updated fields.
sId	RequestTime being updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The ApiRequestTime XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The update process will update all fields in RequestTime.

The ByPassMetadataCheck switch will stop any meta data validation in RequestTime.

It is recommended to remove the XML tags if you want to retain the original data in the database.

The method uses the following sequence to find the RequestTime ID:

1. If the sId parameter is passed in, the method uses this value for the RequestTime ID.
2. If this fails, the method attempts to extract the RequestTime ID from <requesttimeid> in the XML.
3. If this fails, the RequestTime ID is taken from the object properties.

Example

```
Dim proxy As New WebRequestTime.RequestTimeWse
Dim sMyXML As String = ""
Dim wsRet As WebRequestTime.WSInt32
Dim sReqTimeId As String = ""
sReqTimeId = "{CE26F116-8FCF-4697-A0BF-00118BC3059A}"
UserToken.SetToken(proxy, mUserName, mPassword)
'Get RequestTime XML structure with existing data if it's necessary.
sMyXML = proxy.GetByXML(sReqTimeId).Value
'modify sMyXML and then pass it to UpdateByXML
'
wsRet = proxy.UpdateByXML(sMyXML)
```

Related information

"RequestTime" on page 1209

"RequestTime: GetByXML" on page 1219

"RequestTime: GetXMLStructure" on page 1222

"ApiRequestTime XML" on page 576

Resource

The ApiResource object allows users to create, retrieve, update and delete Changepoint resources as well as general resource information, such as: addresses, payroll information and billing rates. With the migration to .NET in version 12.0, the interface IResourceBase inherits IBusinessPropertyBase and the class ResourceClass implements IResourceBase.

The Resource object is composed of several objects: ApiResourceAddress, ApiResourcePayroll, and a collection of ApiResourceRate objects.

Each of the subobjects is capable of updating data and retrieving current data into their properties. In some cases such as ApiResourceRate, new rates can also be added. It is recommended that smaller subobjects should be used when a specific change is only in a single area under the smaller object's control.

Namespace

`http://changepoint.com/changepoint/CPWebService/Resource`

URL

`http://webserver/CPWebService/Objects/CPResource/Resource.asmx`

Methods

Properties

For more information, see "ApiResource" on page 579.

Related information

"ApiResource XML" on page 588

"ResourceAddress" on page 1266

"ResourcePayroll" on page 1271

"ResourceRate" on page 1275

Resource: Add

```
Public Function Add(ByRef sId As String, ByVal oResource As ApiResource) As
WSInt32
```

Purpose

Adds a new resource to Changepoint.

Parameters

Parameter	Description
sId	ByRef parameter that will return the new ResourceId after the resource has been added.
oResource	ApiResource object that contains all the data for the new resource being added to Changepoint

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Adds a Resource to Changepoint. Mandatory fields are the following fields:

- First Name
- Last Name
- Name (updated automatically when first, middle and last name are set)
- BaseCurrency

  Note: The Add method does not set the TerminationDate when creating a resource. To set the TerminationDate when you create a resource, call the Update method immediately after the Add method.

Error numbers are documented in the file "cpExceptions.txt".

The Resource object also contains Address and Payroll objects that can be accessed through Resource.Address and Resource.Payroll.

New rates can be added to the object using one of two methods: UpdateRates or SetPropertiesByXml.

Assigning a new resource to a GlobalWorkgroup and Workgroup causes the resource to become "active" in Changepoint, otherwise resources are added with an "unassigned" state.

Example

The following example uses separate objects to get data into Changepoint. Another method that avoIds the separate update call through the ResourceRate object is to use an XML string containing all data for the resource (Address, Payroll and Rate data) and use the SetPropertiesByXML which will automatically load the object and all internal objects with the data held in the XML.

```
Dim proxy As New WebResource.ResourceWse
Dim myResource as New WebResource.ApiResource
Dim mAddress As New WebResource.ApiResourceAddress
Dim mPayroll As New WebResource.ApiResourcePayroll

Dim iRet as Int32 = 0
Dim sNewId as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
With myResource
   .FirstName = "John"
   .LastName = "Smith"
   .BaseCurrency = "CAD"
   ...
End With
With mAddress
   .AddressLine = "131 Winding Lane"
   .City = "Toronto"
   Dim mIdent as New WebAddress.Identity
   mIdent.Id = "{06810a99-77ba-41ce-91be-2f8747ccc65f}"
   mIdent.Name = "Ontario"
   .StateProvince = mIdent
   ...
End With
With mPayroll
   .HoursPerDay = 8.0
   .NonWorkingDay = "1000001"
   .AnnualVacationHours = 240
End With
```

```
'Add address and Payroll data to the object
myResource.Address = mAddress
myAddress.Payroll = mPayroll

'Add the new resource to Changepoint
iRet = proxy.Add(sNewId, myResource).Value
If iRet = 0 And sNewId.Length > 0 Then
'add a new rate
   Dim myNewRate as WebResource.ApiResourceRate

   proxy = New WebResource.ApiResourceRateWse
   myNewRate = New WebResource.ApiResourceRate
   With myNewRate
      .ResourceId = sNewId
      .Currency = "CAD"
      .EffectiveDate = Now.Date
      .HourlyBilling = 80.00
      .DailyBilling = 500.00
   End With
   'Add the rate data to Changepoint
   iRet = proxy.Update(myNewRate).Value
   If iRet <> 0 then
   'Handle error
   End If
EndIf
...'Continue processing
```

Related information

"Resource" on page 1227

Resource: CanUnassign

```
Public Function CanUnassign(ByVal sResourceId as String, ByVal dEffDate as
Date) As WSInt32
```

Purpose

Checks whether the resource specified can be unassigned successfully.

Parameters

Parameter	Description
sResourceId	ResourceId of the resource to be deleted
dEffDate	Effective date the unassignment will take place

Returns

Type	Description
WSInt32	0 = Success nonzero = cannot unassign, check list of errors for the specific reason. Access the returned Int32 through the .Value property of WSInt32.

Remarks

The deletion process first unassigns the resource before deleting. Note deletions are "soft" deletes, where the deleted flag is 1.

The process of unassigning a resource will be halted if:

- the resource is actively involved in projects or tasks where time has been booked and/or billings have been booked to contracts that are active
- the resource is a project manager and there are active projects under their control
- there is a future change to the resource set to occur (such as a transfer or even a future unassignment)

Example

```
Dim proxy As New WebResource.ResourceWse
Dim iRet as Int32 = 0
Dim sResId as String = ""

sResId = "{5B401107-53D3-4328-8369-3F6F2BDAA86C}"
UserToken.SetToken(proxy, mUserName, mPassword)

'Check if the resource can be unassigned
iRet = proxy.CanUnassign (sResId, "05/25/2007").Value
'Unassign the resource
If iRet = 0 then
```

```
   iRet = proxy.UnassignResource(sResId, "05/25/2007", False).Value
End If
...'Continue processing
```

Related information

"Resource" on page 1227

Resource: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, ByRef sId As String) As
WSInt32
```

Purpose

Create a resource using an XML string of the Resource object in Changepoint.

Parameters

Parameter	Description
sXML	A new resource XML string.
sId	ByRef parameter that returns the new ResourceId after the resource has been added.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The ApiResource XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The ByPassMetadataCheck switch will stop any meta data validation in Resource and also in Resource UDF's.

For details of the use of UDF default values, see "UDF default values logic for CreateByXML" on page 746.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim wsRet As WebResource.WSInt32
Dim sMyXML As String = ""
Dim iRet As Int32 = 0
UserToken.SetToken(proxy, mUserName, mPassword)
'Get Resource XML structure
sMyXML = proxy.GetXMLStructure().Value
'populate sMyXML with data and then pass it to CreateByXML
'
wsRet = proxy.CreateByXML(sMyXML)
```

Related information

"Resource" on page 1227

"Resource: GetByXML" on page 1240

"Resource: GetXMLStructure" on page 1252

"ApiResource XML" on page 588

Resource: Delete

```
Public Function Delete(ByVal sId as String) As WSInt32
```

Purpose

Deletes an existing resource from Changepoint.

Parameters

Parameter	Description
sId	ResourceId of the resource to be deleted.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The deletion process first unassigns the resource before deleting. Note deletions are "soft" deletes, where the deleted flag is 1.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim iRet as Int32 = 0
Dim sResId as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)

sResId = "{5B401107-53D3-4328-8369-3F6F2BDAA86C}"
iRet = proxy.Delete(sResId).Value
Returns iRet
```

Related information

"Resource" on page 1227

Resource: Exists

```
Public Function Exists(ByVal sId as String) As WSBoolean
```

Purpose

Checks whether the Resource exists in the database and has not been deleted.

Parameters

Parameter	Description
sId	Resource needing confirmation whether they still exist in Changepoint.

Returns

Type	Description
WSBoolean	True if the resource exists, else False Access the returned Boolean through the .Value property of WSBoolean.

Remarks

This method checks against the deleted flag of the Resource (resource records are soft deleted in Changepoint).

Example

```
Dim proxy As New WebResource.ResourceWse
Dim bRet as Boolean

UserToken.SetToken(proxy, mUserName, mPassword)
'Check if the resource exists in Changepoint
bRet = proxy.Exists("{5B401107-53D3-4328-8369-3F6F2BDAA86C}").Value
```

Related information

"Resource" on page 1227

Resource: GetAllByWorkgroup

```
Public Function GetAllByWorkgroup(ByVal sWorkgroupId As String, ByVal
sWorkgroupName as String) As WSDataSet
```

Purpose

Retrieves all resources who belong to the specified sWorkgroupId or Workgroup Identified by sWorkgroupName

Parameters

Parameter	Description
sWorkgroupId	The WorkgroupId whose resources are being extracted. An empty string is allowed.
sWorkgroupName	The name of the workgroup. An empty string is allowed.

Returns

Type	Description
WSDataSet	WSDataset (WorkgroupId, ResourceId, ResourceName), or WSDataset (ResourceId, ResourceName). Access the returned dataset through the .Value property of WSDataSet.

Remarks

This method relies on the parameters passed in for Workgroup information, if it is missing it falls back to the Workgroup of the resource object. If there is none it will return an error.

If sWorkgroupName is used, the WorkgroupId is returned as part of the dataset becaause there is a possibility of duplicate Workgroup names. When sWorkgroupId is passed, the returned the dataset consists of (ResourceId, ResourceName).

Example

```
Dim proxy As New WebResource.ResourceWse
Dim ds as New DataSet

UserToken.SetToken(proxy, mUserName, mPassword)
'Get all the resources in the Workgroup MyWorkgroupName
ds = proxy.GetAllByWorkgroup("", "MyWorkgroupName").Value
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count > 0 then
   ...
   Else
   ...
End If
```

Related information

"Resource" on page 1227

Resource: GetAllNames

```
Public Function GetAllNames() As WSDataSet
```

Purpose

Retrieve the names of all resources not deleted from Changepoint.

Parameters

None

Returns

Type	Description
WSDataSet	WSDataSet (ResourceId, Name) Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

```
Dim proxy As New WebResource.ResourceWse
Dim ds as DataSet

UserToken.SetToken(proxy, mUserName, mPassword)
ds = proxy.GetAllNames.Value
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count > 0 Then
   'Iterate through the dataset as necessary
End If
```

Related information

"Resource" on page 1227

Resource: GetById

```
Public Function GetById(ByVal sId as String) As WSResource
```

Purpose

Retrieves resource, resource address, resource payroll, rate data into the object specified by ResourceId.

Parameters

Parameter	Description
sId	ResourceId of the resource whose data is to be retrieved.

Returns

Type	Description
WSResource	0 = Success nonzero = Error Access the returned Resource object through the .Value property of WSResource.

Remarks

This method fills the object with current data from the database. In the Resource object this method also fills the inner Address and Payroll objects along with existing Rate data.

WebPassword cannot be retrieved, only set during an update.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim mResource As New WebResource.ApiResource
Dim iRet as Integer
Dim sResourceId as String = ""
Dim sFirstName as String = ""
Dim sLastName as String = ""
Dim sName as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
'Retrieve current resource data into the object
mResource = proxy.GetById("{5B401107-53D3-4328-8369-
3F6F2BDAA86C}").Value
'Extract resource data
If iRet = 0 then
   With myResource
      sResourceId = .ResourceId
      sFirstName = .FirstName
      sLastName = .LastName
      sName = .Name
      ...
   End With
Else
   Returns iRet
End If
```

Related information

"Resource" on page 1227

Resource: GetByXML

```
Public Function GetByXML(ByVal sXML as String, ByVal sResourceId as String) As
WSString
```

Purpose

Retrieve current resource data based on the resource ID passed and the fields in the XML string.

Parameters

Parameter	Description
sXML	The XML string containing fields where data is required.
sResourceId	The resource whose data is being retrieved.

Returns

Type	Description
WSString	Returns an XML string with resource data filled in. Access the returned string through the .Value property of WSString.

Remarks

The XML string sXML can be varied in the number of fields. If only a small subset of data is required, any unwanted fields can be removed from sXML. The returned XML string will mirror sXML in the fields contained in the XML.

Example

Not available

Related information

"Resource" on page 1227

Resource: GetDefaultEffDate

```
Public Function GetDefaultEffDate(ByVal sResourceId As String) As WSDate
```

Purpose

Returns the default effective date that is used by the system for setting effective date values.

Parameters

Parameter	Description
sResourceId	Resource whose default effective date is to be retrieved.

Returns

Type	Description
WSDate	The default effective date. Access the date through the .Value property of WSDate.

Remarks

None

Example

```
Dim proxy As New WebResource.ResourceWse
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As WebResource.WSDate = proxy.GetDefaultEffDate("{7C8907BA-EDD5-401A-919E-FACF4261ABD3}")
```

Related information

"Resource" on page 1227

Resource: GetFullName

```
Public Function GetFullName(ByVal sFirstName As String, ByVal sMiddleName As String, ByVal sLastName As String) As WSString
```

Purpose

Returns a formatted resource name based on the passed in parameters for first, middle and last names.

Parameters

Parameter	Description
sFirstName	First name of the resource
sMiddleName	Middle name of the resource
sLastName	Last name of the resource.

Returns

Type	Description
WSString	The resource name as a string. Access the returned string through the .Value property of WSString.

Remarks

None

Example

```
Dim proxy As New WebResource.ResourceWse
Dim myResourceName as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
myResourceName = proxy.GetFullName ("John", "D","Smith")
```

Related information

"Resource" on page 1227

Resource: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As WSString
```

Purpose

Returns the ResourceId based on the UDF Text field and value.

Parameters

Parameter	Description
sUDFField	UDF text field name (for example, Text1)
sUDFValue	UDF text field value

Returns

Type	Description
WSString	String with the ResourceId or an empty string. Access the returned string through the .Value property of WSString

Remarks

None

Example

Not available

Related information

"Resource" on page 1227

Resource: GetList

```
Public Function GetList(ByVal iRetRows as Short) As WSDataSet
```

Purpose

Returns a list of all resources in Changepoint who are not deleted in the system.

Parameters

Parameter	Description
iRetRows	The number of records to be returned. To return all, you must pass in -1.

Returns

Type	Description
WSDataSet	WSDataSet (ResourceId, Name) Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

```
Dim proxy As New WebResource.ResourceWse
Dim ds as DataSet
Dim sResourceId as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
'Get all resources
ds = proxy.GetList(-1).Value
'Iterate through the dataset
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count > 0 then
   For each mRow As DataRow in ds.Tables(0).Rows
      If mRow.Item("Name").ToString = "John Smith" Then
         sResourceId = mRow.Item("ResourceId")
         Exit For
      End If
   Next mRow
Else
    Returns sResourceId
End If
```

Related information

"Resource" on page 1227

Resource: GetRate

```
Public Function GetRate(ByVal sResourceRateId As String, ByVal sResourceId As
String) As WSString
```

Purpose

Returns a rate as an XML string based on sResourceRateId and sResourceId.

Parameters

Parameter	Description
sResourceRateId	ID of the resource rate being extracted.
sResourceId	ResourceId of the resource whose rate is being extracted.

Returns

Type	Description
WSString	An XML string of the resource rate. If no rate is found, an XML string is returned with no values in the fields. Access the returned string through the .Value property of WSString.

Remarks

If no ResourceId is specified (sResourceId) then the object's ResourceId value is used, again if there is none, data is extracted based on sResourceRateId alone.

When there is a ResourceId specified in sResourceId, the provided rate is checked for validity against the rates for the Resource in sResourceId.

If only the ResourceRateId is used confirm in the returned XML the validity of the related resource.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim sResourceName as String = "John Q Smith"
Dim sResId as String = ""
Dim sResRateId as String = "{7143D807-D450-490F-AD20-629C43D54266}"
Dim sMyRateXml as String = ""
Dim ds as New DataSet
UserToken.SetToken(proxy, mUserName, mPassword)

'Get the ResourceId, as we have none to link to
ds = proxy.GetResourceIdsByName(sResourceName, "111").Value
'Note, for this example there is the assumption the name is unique therefore
'only a single record is returned.
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count = 1 Then
   sResId = ds.Tables(0).Rows(0).Item("ResourceId").ToString
End If
```

```
if sResId <> "" Then
   sMyRateXml = proxy.GetRate (sResRateId, sResId).Value
End If
```

Related information

"Resource" on page 1227

Resource: GetResourceIdsByName

```
Public Function GetResourceIdsByName(ByVal ResourceName As String, Byval
sNameFormat as String) As WSDataSet
```

Purpose

Returns all resources that match the criteria passed in, and are not deleted.

Parameters

Parameter	Description
ResourceName	Name of a resource
sNameFormat	Denotes the format of ResourceName where "John Smith" can be denoted with "101" indicating the name string consists of a first name and a last name with no mIddle name. If sNameFormat is an empty string the format "111" will be used.

Returns

Type	Description
WSDataSet	WSDataSet (ResourceId) Access the returned dataset through the .Value property of WSDataSet.

Remarks

There may be multiple records with the same names, since names are not unique.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim ds as New DataSet
UserToken.SetToken(proxy, mUserName, mPassword)
```

```
ds = proxy.GetResourceIdsByName("John Q Smith", "111").Value
```

Related information

"Resource" on page 1227

Resource: GetResourcesByUserDefinedId

```
Public Function GetResourcesByUserDefinedId(ByVal sUserDefinedId As String) As
WSDataSet
```

Purpose

Returns a dataset of all resources that match the UserDefinedId as specified in the parameter sUserDefinedId and are not deleted in Changepoint.

Parameters

Parameter	Description
sUserDefinedId	UserDefinedId of the resource.

Returns

Type	Description
WSDataSet	WSDataSet (ResourceId, Name) Access the returned dataset through the .Value property of WSDataSet.

Remarks

There is a possibility of multiple records being returned as the UserDefinedId is not guaranteed to be unique in Changepoint.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim ds as New DataSet

UserToken.SetToken(proxy, mUserName, mPassword)
ds = proxy.GetResourcesByUserDefinedId("Resource99").Value
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count > 0 Then
   ' Iterate through the dataset, until desired data is found
   ...
```

```
End If
```

Related information

"Resource" on page 1227

Resource: GetResTypes

```
Public Function GetResTypes() As WSDataSet
```

Purpose

Returns all Resource types in Changepoint

Parameters

None

Returns

Type	Description
WSDataSet	WSDataSet. (Code, Description). Access the returned dataset through the .Value property of WSDataSet.

Remarks

In the returned dataset, the "Code" field will be a three-character code and is the unique Identifier for the Resource type

Example

```
Dim proxy As New WebResource.ResourceWse
Dim ds as New DataSet

UserToken.SetToken(proxy, mUserName, mPassword)
ds = proxy.GetResourcesByUserDefinedId("Resource99").Value
If ds.Tables.Count > 0 AndAlso ds.Tables(0).Rows.Count > 0 Then
   ' Iterate through the dataset, until desired data is found
   ...
End If
```

Related information

"Resource" on page 1227

Resource: GetUDF

```
Public Function GetUDF(ByVal sWorkgroupId as String, ByVal sActionResourceId
as String, ByVal sResourceId as String, ByVal UDFOption As CPUDFReturnType) As
WSString
```

Purpose

Returns UDF (configurable field) data for a specified resource as an XML string.

Parameters

Parameter	Description
sWorkgroupId	Workgroup of the resource
sActionResourceId	Resource querying for information
sResourceId	Resource whose UDF data is required
UDFOption	Option that determines the UDF data to be returned.

Returns

Type	Description
WSString	An XML string of UDF's Access the returned string through the .Value property of WSString.

Remarks

The CPUDFReturnType Values are:

- WithStructure = 0 – returns UDF field data with full field structure
- OnlyValues = 1 – returns only UDF fields with data, does not include any field structure or options
- OnlyStructure = 2 – returns only UDF XML structure, no field data
- OnlyStructureAndOptions = 3 – returns UDF structure and option data without field data

- WithStructureAndOptions = 4 – returns UDF field data with full structure and all field options (except entity-based and conditional codes)

Example

```
Dim proxy As New WebResource.ResourceWse
Dim sRetUDF as String
Dim sWorkgroup as String = ""
Dim sResId as String = ""

sWorkgroup = "{79d28dde-d4f7-42c9-800b-cbfaf527259c}"
sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"

UserToken.SetToken(proxy, mUserName, mPassword)

'Get the UDF XML string
sRetUDF = proxy.GetUDF(sWorkgroup, myCon.ChangepointUserId, sResId,
CPUDFReturnType.OnlyValues).Value
...'Continue processing
```

Related information

"Resource" on page 1227

Resource: GetUDFCodeOptions

```
Public Function GetUDFCodeOptions(ByVal sResourceId as String, ByVal sCodeName
as String, ByVal sSearchString as String) As WSString
```

Purpose

Returns UDF data for a resource as an XML string.

Parameters

Parameter	Description
sResourceId	Id of the resource
sCodeName	Name of the UDF code. Must be of the form "Code#", for example, "Code10"
sSearchString	Search string.

Returns

Type	Description
WSString	An XML string of the option list that exists for a specified code. Access the returned string through the .Value property of WSString.

Remarks

The search string matches all the options in sCodeName and returns all options containing sSearchString in the option text.

This method uses the login User ResourceId to extract the UDFCode options. Although GetById must be called to retrieve current data into the object, the ResourceId passed into GetById is not used to retrieve UDF Code options.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim sRetUDFOptions as String
Dim sCodeName as String = ""
Dim sSearchStr as String = ""
Dim sResourceId as String = ""

sCodeName = "Code1"
UserToken.SetToken(proxy, mUserName, mPassword)
'Get the UDF Code options as an XML string
sRetUDFOptions = proxy.GetUDFCodeOptions(sResourceId, sCodeName,
sSearchStr).Value
Returns sRetUDFOptions
```

Related information

"Resource" on page 1227

Resource: GetWorkgroupInfo

```
Public Function GetWorkgroupInfo(ByVal sId as String) As WSDataSet
```

Purpose

Retrieves the current Globalworkgroup and Workgroup the specified resource (sId) is related to

Parameters

Parameter	Description
sId	The Resource whose workgroup info is required. The method will throw an error if a valid GUID string is not passed in the parameter.

Returns

Type	Description
WSDataSet	The returned dataset contains the fields GlobalWorkgroupId, GlobalWorkgroup, WorkgroupId, Workgroup. The returned DataSet is accessed through the .Value property of WSDataSet.

Remarks

This method relies solely on the value in sId for the ResourceId. If an empty string or invalid GUID string is passed, an error is raised.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim ds as DataSet
Dim sResId as String = ""

sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
UserToken.SetToken(proxy, mUserName, mPassword)
ds = proxy.GetWorkgroupInfo(sResId).Value
Returns ds
```

Related information

"Resource" on page 1227

Resource: GetXMLStructure

```
Public Function GetXMLStructure() As WSString
```

Purpose

Returns the XML structure of the ApiResource object

Parameters

None

Returns

Type	Description
WSString	An XML string of the ApiResource object. Access the returned XML String through the .Value property of WSString.

Remarks

Some fields in the structure will have defaulted data, otherwise fields are empty.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim sStructure as String = ""
UserToken.SetToken(proxy, mUserName, mPassword)
sStructure = proxy.GetXMLStructure
```

Related information

"Resource" on page 1227

"ApiResource XML" on page 588

Resource: SaveUDF

```
Public Function SaveUDF(ByVal sXML as String, ByVal bypassMetadata as
CPMetadataCheck) As WSInt32
```

Purpose

Saves (Insert/Update) UDF information for a resource.

Parameters

Parameter	Description
sXMLUDF	UDF string in XML format to be saved.
bypassMetadata	Determines the level of metadata checking for sXMLUDF. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. If HaveErrors is True, then check WSException.Message and Logs.

To obtain the correct UDF XML format, call the GetUDF method.

Example

Not available

Related information

"Resource" on page 1227

"Resource: GetUDF" on page 1249

Resource: SetPropertiesByXML

```
Public Function SetPropertiesByXML(ByVal sXML as String, ByVal bNotInitialize
as Boolean) As WSResource
```

Purpose

Set the properties of the object via an XML string

Parameters

Parameter	Description
sXML	XML string of the Resource object with data to load into the object
bNotInitialize	Switch that initializes the object. Normal use is to set as "True"

Returns

Type	Description
WSResource	0 = Success nonzero = Error Access the returned Resource object through the .Value property of WSResource.

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods. If you use the SetPropertiesByXML method, you must pass in the entire XML string.

The XML string can be the entire resource object with all or some fields filled.

Empty fields in the XML string will result in the object's matching field being overwritten with an empty value (in essence you will clear the field). If data in the object is to be retained, remove the field from the XML string.

At a minimum the ResourceId and ByPassMetaDataCheck fields should be present in the XML string

When called from an empty object, the method will first retrieve current data into the object before applying the new field values as held in sXML.

In the case where properties in the object have been altered prior to SetPropertiesByXML being called, the method will try to preserve those values when retrieving base data.

It is best to load an empty object with data first before applying any changes.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim myResource as New WebResource.ApiResource
Dim iRet as Int32 = 0
Dim sResId as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)

'Load the object with any changes, refer to ApiResource XML for XML that can
be 'used in sUpdateXML
myResource = proxy.SetPropertiesByXML(sUpdateXML, True).Value
If proxy.Update(myResource).Value <> 0 Then
… 'Handle error
End If
```

Related information

"Resource" on page 1227

"Resource: CreateByXML" on page 1233

"Resource: UpdateByXML" on page 1260

Resource: UnassignResource

```
Public Function UnassignResource(ByVal sResourceId As String, ByVal
dEffectiveDate As Date, ByVal bCheckIfCanUnassign As Boolean) As WSInt32
```

Purpose

Unassign a specified resource in Changepoint

Parameters

Parameter	Description
sResourceId	ID of the resource to be unassigned
dEffectiveDate	Date on which the unassignment is to occur. Default = 1/1/1753
bCheckIfCanUnassign	Switch indicating whether a check is made whether the resource can be unassigned.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Unassigning a resource is similar to a deletion from Changepoint. This process will be halted if:

- the resource is actively involved in projects or tasks where time has been booked and/or billings have been booked to contracts that are active.
- the resource is a "manager" and there are active projects under their control the process will not continue.
- there is a future change to the resource set to occur (such as a transfer or even future "unassignment")

Example

```
Dim proxy As New WebResource.ResourceWse
Dim myResource As New ApiResource
Dim sResId As String = ""
Dim iRet As Int32 = 0

sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
UserToken.SetToken(proxy, mUserName, mPassword)
'Unassign the resource today, and also check if the resource can be
'unassigned before proceeding
iRet = proxy.UnassignResource(sResId, Now, True)
```

Related information

"Resource" on page 1227

Resource: Update

```
Public Function Update(ByVal oResource As ApiResource) As WSInt32
```

Purpose

Update Resource data in the database with data held in the object

Parameters

None

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Update will write all data held in the Resource object including the internal objects for Address, Payroll and any Rate objects.

The API does not verify if data in the resource objects or subobjects have been modified. When an Update() method is called, the entire object and subobject collections are updated regardless whether modifications have been done or not in the object or subobject data. The UpdatedOn and UpdatedBy fields in all related tables such as ResourceRate are overwritten, each time the Update() method is used.

This process carries a higher overhead than calling update from the smaller objects such as Address. Therefore it is a good Idea to use Update when there are large changes across the entire Resource. If changes are centered on an area such as Address or Payroll, it is quicker to instantiate an instance of those objects and run an update from the Address or Payroll objects.

Depending on requirements, metadata checking can be turned on or off by using the ByPassMetaDataCheck switch. Setting this switch will halt any metadata checking in the object, including UDFs.

The following example uses separate objects to get data into Changepoint. Another method that avoids the separate update call through the ResourceRate object is to use an XML string that contains all data for the resource (Address, Payroll and Rate data) and use the SetPropertiesByXML which will automatically load the object and all internal objects with the data held in the XML.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim myResource as New WebResource.ApiResource
Dim mAddress As New WebResource.ApiResourceAddress
Dim mPayroll As New WebResource.ApiResourcePayroll

Dim iRet as Int32 = 0
Dim sNewId as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
myResource = proxy.GetById("{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}").Value
With myResource
   .Title = "Sales Manager"
   .TitleEffectiveDate = "08/15/2007"
   .BaseCurrency = "USD"
   ...
End With
With mAddress
   .AddressLine = "10 Post Road"
   .City = "Denver"
   Dim mIdent as New WebAddress.Identity
   mIdent.Id = "{06810a99-77ba-41ce-91be-2f8747ccc65f}"
   mIdent.Name = "Colorado"
   .StateProvince = mIdent
   ...
End With
With mPayroll
   .HoursPerDay = 8.0
   .NonWorkingDay = "1000001"
   .AnnualVacationHours = 280
End With
'Add address and Payroll data to the object
myResource.Address = mAddress
myAddress.Payroll = mPayroll

'Update the resource
iRet = proxy.Update(myResource).Value
If iRet = 0 Then
   'add a new rate
   Dim myNewRate as WebResource.ApiResourceRate

   proxy = New WebResource.ApiResourceRateWse
   UserToken.SetToken(proxy, mUserName, mPassword)
   myNewRate = New WebResource.ApiResourceRate
   With myNewRate
      .ResourceId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
      .Currency = "USD"
```

```
      .EffectiveDate = "08/15/2007"
      .HourlyBilling = 100.00
      .DailyBilling = 800.00
   End With
   'Add the rate data to Changepoint
   iRet = proxy.Update(myNewRate).Value
   If iRet <> 0 then
      'Handle error
   End If
EndIf
...'Continue processing
```

Related information

"Resource" on page 1227

Resource: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, ByVal sResourceId As String)
As WSInt32
```

Purpose

Update a resource using an XML string containing new data.

Parameters

Parameter	Description
sXML	The XML string of the resource object with new data contained in the fields.
sResourceId	The date on which the unassignment is to occur

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Performs the same function as Update except any resource can be updated through this function. The XMl sent in the parameter should be of the form in ApiResource XML. The ApiResource XML structure can be obtained by the GetXMLStructure or GetByXML methods.

Ensure that any fields that are not updated are removed from the XML string.

The update process will first get any current data before applying the updated data in the XML string.

This method does not alter any data in the Resource object itself.

It is also possible to add new rates as well as update existing ones. New rates should have empty <ResourceRateId> tags in the XML string

The method uses the following sequence to find the resource ID:

1. If the sResourceId parameter is passed in, the method uses this value for the resource ID.
2. If this fails, the method attempts to extract the resource ID from <resourceid> in the XML.
3. If this fails, the resource ID is taken from the object properties.
4. If this fails, an attempt is made to look up the resource ID using <userdefinedresourceidd> in the XML. If <userdefinedresourceid> has a value, but the value is invalid or duplicated, then you will get an error and no further attempts are made to look up the resource ID.
5. If <userdefinedresourceid> is empty, an attempt is made to look up the resource ID using <firstname> and <lastname> in the XML

Example

```
Dim proxy As New WebResource.ResourceWse
Dim sMyXML As String = ""
Dim wsRet As WebResource.WSInt32
Dim sResId As String = ""
sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
UserToken.SetToken(proxy, mUserName, mPassword)
'Get Resource XML structure with existing data if it's necessary.
sMyXML = proxy.GetByXML(sResId).Value
'modify sMyXML and then pass it to UpdateByXML
'
wsRet = proxy.UpdateByXML(sMyXML)
```

Related information

"Resource" on page 1227

"Resource: GetByXML" on page 1240

"Resource: GetXMLStructure" on page 1252

"ApiResource XML" on page 588

Resource: UpdateLoginInfo

```
Public Function UpdateLoginInfo(ByVal sId As String, ByVal sNtLogin As String)
As WSInt32
```

Purpose

Update the basic login info for a resource

Parameters

Parameter	Description
sId	ResourceId of the resource whose login info is being updated
sNtLogin	New NT Login as a string

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Before changing the login data, the method does a search for other resources that have the same logins. If there are others an error is raised.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim sResId As String = ""
```

```
UserToken.SetToken(proxy, mUserName, mPassword)
sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
iRet = proxy.UpdateLoginInfo(sResId, "JohnsNtL0g1n").Value
```

Related information

"Resource" on page 1227

Resource: UpdateRates

```
Public Function UpdateRates(ByVal sRateXML as String) As WSInt32
```

Purpose

To update or add one or more rates directly from an XML string

Parameters

Parameter	Description
sRateXML	XML string containing one or more ResourceRate object

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The XML string passed in the parameter is of the form

<resourcerates><rate>…</rate><rate>…</rate>…</resourcerates> this is the same form used in the XML for the ApiResource object.

This method will extract the rate objects and update the data for each. Ensure the <resourceId> and <resourcerateId> tags are filled with the correct values.

You cannot create a new rate using this method.

If the XML string is empty, the method will attempt to save the rates held within the object itself.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim iRet as Int32 = 0

UserToken.SetToken(proxy, mUserName, mPassword)

iRet = proxy.UpdateRates(sMyUpdateRateXML).Value
```

Related information

"Resource" on page 1227

"ApiResource XML" on page 588

Resource: UpdateRolesAndFeatures

```
Public Function UpdateRolesAndFeatures(ByVal sResourceId As String, ByVal
sRoles As String, ByVal sFeatures As String) As WSInt32
```

Purpose

Replace the resource's roles and features with passed in data.

Parameters

Parameter	Description
sResourceId	Resource whose roles and features are being modified.
sRoles	String of role IDs separated by the pipe character (|)
sFeatures	String of feature IDs separated by the pipe character (|)

Returns

Type	Description
WSInt32	0 = Succes nonzero = Error Access the returned int32 through the .Value property of WSInt32.

Remarks

In both sRoles and sFeatures, the update process removes any existing roles or features from the resource. If you pass in an empty string "" for sRoles, all roles assigned to the resource are removed. If you pass in an empty string "" for sFeatures, all features assigned to the resource are removed.

Example

Not available

Related information

"Resource" on page 1227

Resource: UpdateWebPassword

```
Public Function UpdateWebPassword(ByVal sResourceId As String, ByVal sPassword
As String) As WSInt32
```

Purpose

Allow changes to the web password for a specified resource

Parameters

Parameter	Description
sResourceId	ResourceId of resource whose password is being changed.
sPassword	New password

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned int32 through the .Value property of WSInt32.

Remarks

The password will be encrypted and saved to the database; therefore, it is not necessary to pass in an encrypted form of the new password.

Example

```
Dim proxy As New WebResource.ResourceWse
Dim sNewPassword as String = "J0hns563*7"
Dim iRet as Int32 = 0
Dim sResId As String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
sResId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"

iRet = proxy.UpdateWebPassword(sResId, sNewPassword).Value
```

Related information

"Resource" on page 1227

ResourceAddress

The ResourceAddress object contains common address information. IResourceAddress inherits properties from the IAddressBase interface.

Namespace

`http://changepoint.com/changepoint/CPWebService/ResourceAddress`

URL

`http://webserver/CPWebService/Objects/CPResource/ResourceAddress.asmx`

Methods

There is no Add method because a new address record is automatically created in the database when a new resource is created in Changepoint whether through the API or the user interface.

Properties

For more information, see "ApiResourceAddress" on page 631

Related information

"ApiResourceAddress XML" on page 634

"Resource" on page 1227

ResourceAddress: GetById

```
Public Function GetById(ByVal sId as String) As WSResourceAddress
```

Purpose

Retrieve current resource address data

Parameters

Parameter	Description
sId	ID of the entity (resource).

Returns

Type	Description
WSResourceAddress	Returns an ApiResourceAddress object. Access the object through the .Value property of WSResourceAddress.

Remarks

Retrieves current data into the object. Any existing data is replaced with fresh data from the database.

Example

Not available

Related information

"ResourceAddress" on page 1266

ResourceAddress: New

```
Public Function New()
```

Purpose

Instantiate a new object of ApiResourceAddress Type

Parameters

None

Returns

Not applicable

Remarks

Instantiates a new ApiResourceAddress object.

Example

Not available

Related information

"ResourceAddress" on page 1266

ResourceAddress: SetPropertiesByXML

```
Public Function SetPropertiesByXML(ByVal sXML as String, ByVal bNotInitialize
As Boolean) As WSResourceAddress
```

Purpose

Sets the properties of the object using an XML string

Parameters

Parameter	Description
sXML	XML string of the resource address object with data in the fields.
bNotInitialize	Switch that initializes the object. Normal use is to set as "True."

Returns

Type	Description
WSResourceAddress	Returns an ApiResourceAddress object. Access the object through the .Value property of WSResourceAddress.

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods of the Resource object. If you use the SetPropertiesByXML method, you must pass in the entire XML string.

The XML string is returned in structure form if the sEntityId parameter has no value.

The minimal XML that can be passed to this method is:

```
<root>
   <address>
   <resourceId></resourceId>
   </address >
</root>
```

If this method is called from an empty object, the object will fill itself first with the most recent data before applying any changes from sXML.

Be aware that empty fields in sXML will serve to clear fields in the object. Therefore to preserve values, either remove the tags from the XML string or ensure the tag values are the same as currently in the database.

Example

Not available

Related information

"ResourceAddress" on page 1266

"Resource: CreateByXML" on page 1233

"Resource: UpdateByXML" on page 1260

ResourceAddress: Update

```
Public Function Update() As WSInt32
```

Purpose

Update the database with the data held in the object properties.

Parameters

None

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

```
Dim proxy As New WebResourceAddress.ResourceAddressWse
Dim myAddress as New WebResourceAddress.ApiResourceAddress
Dim iRet as Int32 = 0

UserToken.SetToken(proxy, mUserName, mPassword)

With myAddress
   .ResourceId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
   .AddressLine = "131 Winding Lane"
   .City = "Toronto"
   Dim mIdent as New WebResourceAddress.Identity
   mIdent.Id = "{06810a99-77ba-41ce-91be-2f8747ccc65f}"
   .StateProvince = mIdent
   ...
End With
iRet = proxy.Update(myAddress).Value
```

Related information

"ResourceAddress" on page 1266

ResourcePayroll

The ApiResourcePayroll object contains resource payroll information.

Namespace

`http://changepoint.com/changepoint/CPWebService/ResourcePayroll`

URL

`http://webserver/CPWebService/Objects/CPResource/ResourcePayroll.asmx`

Methods

Properties

For more information, see "ApiResourcePayroll" on page 639.

Related information

"ApiResourcePayroll XML" on page 642

"Resource" on page 1227

ResourcePayroll: GetById

```
Public Function GetById(ByVal sId as String) As WSResourcePayroll
```

Purpose

Retrieve current resource payroll data

Parameters

Parameter	Description
sId	The ID of the entity (resource).

Returns

Type	Description
WSResourcePayroll	Returns an ApiResourcePayroll object. Access the object through the .Value property of WSResourcePayroll.

Remarks

Retrieves current data into the object. Any existing data is replaced with fresh data from the database.

Example

Not available

Related information

"ResourcePayroll" on page 1271

ResourcePayroll: New

```
Public Function New()
```

Purpose

Instantiate a new object of ApiResourcePayroll Type

Parameters

None

Returns

Not applicable

Remarks

Instantiates a new ApiResourcePayroll object.

The ByPassMetadataCheck flag is set to False as default

Example

Not available

Related information

"ResourcePayroll" on page 1271

ResourcePayroll: SetPropertiesByXML

```
Public Function SetPropertiesByXML(ByVal sXML as String, ByVal bNotInitialize
As Boolean) As WSResourcePayroll
```

Purpose

Set the properties of the object using an XML string

Parameters

Parameter	Description
sXML	XML string of the ApiResourcePayroll object with data in the fields.
bNotInitialize	Switch that initializes the object. Normal use is to set as "True."

Returns

Type	Description
WSResourcePayroll	Returns an ApiResourcePayroll object. Access the returned object through the .Value property of WSResourcePayroll.

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods of the Resource object. If you use the SetPropertiesByXML method, you must pass in the entire XML string.

The minimal XML that can be passed to this method is:

```
<root>
   <payroll>
      <resourceId></resourceId>
```

```
   </ payroll >
</root>
```

If this method is called from an empty object, the object will fill itself first with the most recent data first before applying any changes from sXML.

Note that empty fields in sXML will basically clear fields in the object. Therefore to preserve values, either remove the tags from the XML string or ensure the tag values are the same as currently in the database.

Example

Not available

Related information

"ResourcePayroll" on page 1271

"Resource: CreateByXML" on page 1233

"Resource: UpdateByXML" on page 1260

ResourcePayroll: Update

```
Public Function Update() As WSInt32
```

Purpose

Update the database with the data held in the object properties.

Parameters

None

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 object through the .Value property of WSInt32.

Remarks

None

Example

```
Dim proxy As New WebResourcePayroll.ResourcePayrollWse
Dim myPayroll as New WebResourcePayroll.ApiResourcePayroll
Dim iRet as Int32 = 0

UserToken.SetToken(proxy, mUserName, mPassword)

With myPayroll
   .ResourceId = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
   .HoursPerDay = 8.0
   .NonWorkingDay = "1000001"
   .AnnualVacationHours = 240
   ...
End With
iRet = proxy.Update(myPayroll).Value
...'Continue processing
```

Related information

"ResourcePayroll" on page 1271

ResourceRate

The ApiResourceRate object allows users to get, update and add resource rate information.

Namespace

`http://changepoint.com/changepoint/CPWebService/ResourceRate`

URL

`http://webserver/CPWebService/Objects/CPResource/ResourceRate.asmx`

Methods

Properties

For more information, see "ApiResourceRate" on page 648.

Related information

"ApiResourceRate XML" on page 650

"Resource" on page 1227

ResourceRate: Add

```
Public Function Add(ByRef sId As String, ByVal oResourceRate As
ApiResourceRate)As WSInt32
```

Purpose

Add a new rate to Changepoint

Parameters

Parameter	Description
oResourceRate	ResourceRate object that contains all the properties for the new resource rate being added to Changepoint (see Resource.Add).
sId	New ResourceRateId.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Ensure the ResourceId field is populated in the object or else the new rate will fail to be added to Changepoint.

Example

```
Dim proxy As New WebResourceRate.ResourceRateWse
Dim myRate As New WebResourceRate.ApiResourceRate
```

```
Dim iRet As New Int32 = 0
Dim sNewId As String = ""

UserToken.SetToken(proxy, mUserName, mPAssword)

With myRate
   .Resource = "{cf4f66f2-a6e8-4686-9408-17dacc73f7e3}"
   .Currency = "CAD"
   .EffectiveDate = Now.Date
   .HourlyBilling = 80.00
   .DailyBilling = 500.00
End With
iRet = proxy.Add(sNewId, myRate).Value
If iRet <> 0 or sNewId.Length = 0 Then
   ...'Handle error
End If
...'Continue processing
```

Related information

"ResourceRate" on page 1275

ResourceRate: GetById

```
Public Function GetById(ByVal sId as String) As WSInt32
```

Purpose

Retrieve current resource rate data into the object.

Parameters

Parameter	Description
sId	ID of the entity (ResourceRateId).

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Retrieves current data into the object. Any existing data is replaced with fresh data from the database.

Example

Not available

Related information

"ResourceRate" on page 1275

ResourceRate: GetXMLStructure

```
Public Function GetXMLStructure() As WSString
```

Purpose

Returns the XML structure of the ApiResourceRate object

Parameters

None

Returns

Type	Description
WSString	An XML string of the ApiResourceRate object. Access the returned string through the .Value property of WSString.

Remarks

Some fields in the structure will have defaulted data, otherwise fields are empty.

Example

```
Dim proxy As New WebResourceRate.ResourceRateWse
Dim oRet As New WebResourceRate.WSString
Dim sStructure as String = ""
UserToken.SetToken(proxy, mUserName, mPassword)
oRet = proxy.GetXMLStructure()
```

Related information

"ResourceRate" on page 1275

"ApiResource XML" on page 588

ResourceRate: New

```
Public Function New()
```

Purpose

Instantiate a new object of ApiResourceRate Type

Parameters

None

Returns

Not applicable

Remarks

Instantiates a new ApiResourceRate object.

The ByPasssMetadataCheck flag is set to False as default

Example

Not available

Related information

"ResourceRate" on page 1275

ResourceRate: SetPropertiesByXML

```
Public Function SetPropertiesByXML(Byref sXML as String, ByVal bNotInitialize
As Boolean) As WSResourceRate
```

Purpose

Set the properties of the object using an XML string.

Parameters

Parameter	Description
sXML	XML string of the resource rate object with data in the fields.
bNotInitialize	Switch that initializes the object. Normal use is to set as "True."

Returns

Type	Description
WSResourceRate	Returns an ApiResourceRate object. Access the object through the .Value property of WSResourceRate.

Remarks

This legacy method has been superseded by the CreateByXML and UpdateByXML methods of the Resource object. If you use the SetPropertiesByXML method, you must pass in the entire XML string.

The minimal XML that can be passed to this method is:

```
<root>
   <rate>
      <resourceId></resourceId>
   </rate>
</root>
```

If this method is called from an empty object, the object will fill itself first with the most recent data first before applying any changes from sXML.

Be aware that empty fields in sXML will serve to "clear" fields in the object. Therefore to preserve values remove the tags from the XML string or ensure the tag values are the same as currently in the database.

Example

Not available

Related information

"ResourceRate" on page 1275

"Resource: CreateByXML" on page 1233

"Resource: UpdateByXML" on page 1260

ResourceRate: Update

```
Public Function Update(ByVal oResourceRate As ApiResourceRate) As WSInteger
```

Purpose

Update the database with the data held in the properties object

Parameters

None

Returns

Type	Description
WSInteger	0 = Success nonzero = Error Access the returned Integer through the .Value property of WSInteger.

Remarks

If the <resourcerateId> tag is empty, the update process will consider the rate data as "new" and will add a new rate.

Example

```
Dim proxy As New WebResourceRate.ResourceRateWse
Dim myRate as New WebResourceRate.ApiResourceRate
Dim iRet as Int32 = 0

UserToken.SetToken(proxy, mUserName, mPassword)
myRate = proxy.GetById("{06810a99-77ba-41ce-91be-2f8747ccc65f}").Value
'Edit the rate
With myRate
   .Currency = "CAD"
   .EffectiveDate = Now.Date
   .HourlyBilling = 80.00
   .DailyBilling = 500.00
   ...
```

```
End With
iRet = proxy.Update(myRate).Value
...'Continue processing
```

Related information

"ResourceRate" on page 1275

Skill

The Skill object allows users to view, create, and edit skills for resources.

Namespace

`http://changepoint.com/changepoint/CPWebService/Skill`

URL

`http://webserver/CPWebSercive/Objects/CPSkill/Skill.asmx`

Methods

Properties

For more information, see "ApiSkill" on page 658.

Related information

"SkillCategory " on page 1287

"SkillCompetency" on page 1291

Skill: Add

```
Public Function Add(ByVal mySkill As ApiSkill) As WSString
```

Purpose

Add a new skill to Changepoint

Parameters

Parameter	Description
mySkill	ApiSkill object.

Returns

Type	Description
WSString	Returns a new Skill ID on success, else an empty string. Access the returned string through the .Value property of WSString.

Remarks

The SkillCategoryId field in ApiSkill must contain a valid GUID value.

Example

```
Dim mySkill as New WebSkill.ApiSkill
Dim proxy As New WebSkill.SkillWse
Dim skillCatProxy As New WebSkillCategory.SkillCategoryWse
Dim sNewId as String = ""
Dim ds as New DataSet
Dim mySkillCat as New WebSkill.ApiSkillCategory

UserToken.SetToken(proxy, mUserName, mPassword)
UserToken.SetToken(skillCatProxy, mUserName, mPassword)
ds = skillCatProxy.GetSkillCategories().Value
If ds.tables.Count > 0 AndAlso ds.Tables(0)
   For Each mRow As DataRow in ds.tables(0).Rows
      If mRow.Item("Name").ToString = "SkillCategory1" Then
         mySkill.SkillCategoryId = mRow.Item ("Code").ToString
         Exit For
      End If
   Next mRow
      Else
      ...
End If
'If the Id field is empty then the skill category was not found and the
process stops
If mySkill.SkillCategoryId.Length = 0 Then
   ...'Handle error
Else
   With mySkill
```

```
      .Description = "A test skill description"
      .Name = "MyTestSkill-1"
   End With
   'Add the new skill to Changepoint
   sNewId = proxy.Add(mySkill).Value
End If
...'Continue processing
```

Related information

"Skill" on page 1282

Skill: GetSkill

```
Public Function GetSkill(ByVal sSkillId As String) As WSSkill
```

Parameters

Parameter	Description
sSkillId	Changepoint Skill Id.

Remarks

Type	Description
WSSkill	Returns an ApiSkill object. Access the object through the .Value property of WSSkill.

Example

Not available

Related information

"Skill" on page 1282

Skill: GetSkills

```
Public Function GetSkills(ByVal sSkillCategoryId As String) As WSDataSet
```

Parameters

Parameter	Description
sSkillCategoryId	Changepoint Skill Category Id.

Remarks

Type	Description
WSDataSet	WSDataset (SkillCode, Name) Access the dataset through the .Value property of WSDataSet.

Example

Not available

Related information

"Skill" on page 1282

Skill: Update

```
Public Function Update(ByVal mySkill As ApiSkill) As WSInt32
```

Purpose

Update a skill in Changpoint

Parameters

Parameter	Description
mySkill	ApiSkill object contains data to be updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

```
Dim proxy As New WebSkill.SkillWse
Dim skillCatProxy As New WebSkillCategory.SkillCategoryWse
Dim mySkill as New ApiSkill
Dim sId as String = ""
Dim ds as New DataSet
Dim mySkillCat as New ApiSkillCategory
Dim iErr as Int32 = 0

UserToken.SetToken(proxy, mUserName, mPassword)
UserToken.SetToken(skillCatProxy, mUserName, mPassword)

'get a list of skill categories and extract necessary code
ds = skillCatProxy.GetSkillCategories().Value
If ds.tables.Count > 0 AndAlso ds.Tables(0)
   For Each mRow As DataRow in ds.tables(0).Rows
      If mRow.Item("Name").ToString = "SkillCategory1" Then
         mySkill.SkillCategoryId = mRow.Item ("Code").ToString
         Exit For
      End If
   Next mRow
      Else
      ...
End If
'If the Id field is empty then the skill category was not found and the
process stops
If mySkill.SkillCategoryId.Length = 0 Then
...'Handle error
Else
   ds = New DataSet
   'Retrieve all skills in the category
   ds = proxy.GetSkills(mySkill.SkillCategoryId).Value
   'Iterate and extract the required skill code
   If ds.tables.Count > 0 AndAlso ds.Tables(0)
      For Each mRow As DataRow in ds.tables(0).Rows
         If mRow.Item("Name").ToString = "MyTestSkill-1" Then
```

```
            sId = mRow.Item("SkillCode").ToString
            Exit For
        End If
...
```

Related information

"Skill" on page 1282

SkillCategory

The SkillCategory object represents skill categories within the Changepoint database.

Namespace

`http://changepoint.com/changepoint/CPWebService/SkillCategory`

URL

`http://webserver/CPWebService/Objects/CPSkill/SkillCategory.asmx`

Methods

Properties

For more information, see "ApiSkillCategory" on page 663

SkillCategory: Add

```
Public Function Add(ByVal mySkillCategory As ApiSkillCategory) As WSString
```

Purpose

Create a new skill category in Changepoint

Parameters

Parameter	Description
mySkillCategory	ApiSkillCategory object.

Returns

Type	Description
WSString	Returns the new SkillCategoryId on success else an empty string. Access the returned string through the .Value property of WSString.

Remarks

None

Example

```
Dim mySkillCategory as New WebSkillCategory.ApiSkillCategory
Dim proxy As New WebSkillCategory.SkillCategoryWse
Dim sNewId as String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
mySkillCategory.Name = "SkillCategory1"
mySkillCategory.Description = "Test Category"
sNewId = proxy.Add(mySkillCategory).Value
if sNewId.Length = 0 Then
   ...'Handle error
End if
```

Related information

"SkillCategory " on page 1287

SkillCategory: GetSkillCategories

```
Public Function GetSkillCategories() As WSDataSet
```

Purpose

Returns recordsets of Skill Category Code, Name.

Parameters

None

Returns

Type	Description
WSDataset	WSDataset (SkillCategoryCode, Name.) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"SkillCategory " on page 1287

SkillCategory: GetSkillCategory

```
Public Function GetSkillCategory(ByVal sSkillCategoryId As String) As
WSSkillCategory
```

Parameters

Parameter	Description
sSkillCategoryId	Changepoint Skill Category ID.

Returns

Type	Description
ApiSkillCategory	Returns an ApiSkillCategory object. Access the object through the .Value property of WSSkillCategory.

Remarks

None

Example

Not available

Related information

"SkillCategory " on page 1287

SkillCategory: Update

```
Public Function Update(ByVal mySkillCategory As ApiSkillCategory) As WSInt32
```

Purpose

Update an existing skill category in Changepoint

Parameters

Parameter	Description
mySkillCategory	ApiSkillCategory object.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

```
Dim mySkillCategory as New WebSkillCategory.ApiSkillCategory
Dim proxy As New WebSkillCategory.SkillCategoryWse
Dim sNewId As String = ""
Dim sSkillCatId As String = ""
Dim iErr As Int32 = -1
Dim ds As New DataSet

UserToken.SetToken(proxy, mUserName, mPassword)
```

```
'Get a list of categories, and extract the correct code value.
'If you already have the code this part can be skipped
ds = proxy.GetSkillCategories().Value
If ds.tables.Count > 0 AndAlso ds.Tables(0).Rows.Count > 0 Then
   For Each mRow As DataRow in ds.Tables(0).Rows
      If mRow.Item("Name").ToString = "SkillCategory1" Then
            sSkillCatId = mRow.Item ("Code").ToString
            Exit For
      End If
   Next mRow
   Else
...
End If
'Retrieve the required skill category
If sSkillCatId.Length > 0 Then
   mySkillCategory = proxy.GetSkillCategory(sSkillCatId).value
End If

'Update the data with new info
mySkillCategory.Name = "SkillUpdatedCategory1"
mySkillCategory.Description = "Update Test Category"

'Save the data
iErr = proxy.Update(mySkillCategory).Value
If iErr <> 0 Then
   ...'Handle error
End if
```

Related information

"SkillCategory " on page 1287

SkillCompetency

The SkillCompetency object allows users to create, retrieve and update competencies for skills within the Changepoint database.

Namespace

`http://changepoint.com/changepoint/CPWebService/SkillCompetency`

URL

`http://webserver/CPWebService/Objects/CPSkill/SkillCompetency.asmx`

Methods

Properties

For more information, see "ApiSkillCompetency" on page 668.

SkillCompetency: Add

```
Public Function Add(ByVal mySkillCompetency As ApiSkillCompetency) As WSString
```

Purpose

Add a new skill competency to Changepoint.

Parameters

Parameter	Description
mySkillCompetency	ApiSkillCompetency object.

Returns

Type	Description
WSString	String (a new SkillCompetencyId) Access the returned string through the .Value property of WSString.

Remarks

None

Example

Both SkillId and SkillCategoryId fields must contain a valid GUID value.

```
Dim mySkillCompetency As New WebSkillCompetency.ApiSkillCompetency
Dim proxy As New WebSkillCompetency.SkillCompetencyWse
Dim iErr As Int32 = 0
Dim sNewId As String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
```

```
mySkillCompetency.SkillId = "{0609965a-3344-4126-89dd-0aece80ec474}"
mySkillCompetency.SkillCategoryId = "{2435a154-a3d9-4a3d-a317-26ba9080b03e}"
mySkillCompetency.Name = "Test Skill Competency"
mySkillCompetency.Description = "Test description"
sNewId = proxy.Add(mySkillCompetency).Value
...'Continue processing
```

Related information

"SkillCompetency" on page 1291

SkillCompetency: GetSkillCompetencies

```
Public Function GetSkillCompetencies(ByVal sSkillId As String) As WSDataSet
```

Parameters

Parameter	Description
sSkillId	Changepoint Skill ID.

Returns

Type	Description
WSDataset	WSDataset (SkillCompetencyCode, Name) Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"SkillCompetency" on page 1291

SkillCompetency: GetSkillCompetency

```
Public Function GetSkillCompetency(ByVal sSkillCompetencyId As String) As
WSSkillCompetency
```

Parameters

Parameter	Description
sSkillCompetencyId	Changepoint Skill Competency Id.

Returns

Type	Description
WSSkillCompetency	Returns an ApiSkillCompetency object. Access the object through the .Value property of WSSkillCompetency.

Remarks

None

Example

Not available

Related information

"SkillCompetency" on page 1291

SkillCompetency: Update

```
Public Function Update(ByVal mySkillCompetency As ApiSkillCompetency) As
WSInt32
```

Purpose

Update Changepoint with updated information for skill competency

Parameters

Parameter	Description
mySkillCompetency	ApiSkillCompetency object containing the data to be updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

```
Dim mySkillCompetency As New WebSkillCompetency.ApiSkillCompetency
Dim proxy As New WebSkillCompetency.SkillCompetencyWse
Dim iErr As Int32 = 0
Dim sNewId As String = ""

UserToken.SetToken(proxy, mUserName, mPassword)
   'If the SkillCompetencyId is not known, only the name then use
GetSkillCompetencies and
   'locate the required SkillCompetency and extract the Id value
mySkillCompetency = proxy.GetSkillCompetency("{8c17755c-582b-4579- 8774-
72643a9b1cd3 }").value
mySkillCompetency.Name = "Test Skill Competency updated"
mySkillCompetency.Description = "Update test description"
iErr = proxy.Update(mySkillCompetency).Value
If lErr <> 0 Then
   ...'Handle error
End If
...'Continue processing
```

Related information

"SkillCompetency" on page 1291

Task

The Task object allows users to add, retrieve, update or delete Task information within the Changepoint database.

Namespace

```
http://changepoint.com/changepoint/CPWebService/Task
```

URL

```
http://webserver/CPWebService/Objects/CPTask/Task.asmx
```

Methods

Properties

For more information, see "ApiTask" on page 672.

Related information

"ApiTask XML" on page 676

"Project" on page 1143

"TaskAssignment" on page 1309

Task: Add

```
Public Function Add(ByRef sId As String, ByVal oTask As ApiTask) As WSInt32
```

Purpose

Add a new task to Changepoint database by assigned task ID.

Parameters

Parameter	Description
sId	Task ID for new task. If no value is assigned, the system will create a new GUID and return it.
oTask	ApiTask object which will be added.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim objApi As New webTask.ApiTask()
With ojbApi
.Name="Richard Task960"
.Project=New webTask.Identity()
.Project.Id= "{…}"
.WorkCode=New webTask.Identity()
.WorkCode.Id= "{…}"
.WorkCodeCategory =New webTask.Identity()
.WorkCodeCategory.Id = "{…}"
...
End With
Dim proxy As New webTask.TaskWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = ""
Dim oRet As webTask.WSInt32 = proxy.Add(sId, objApi)
```

Related information

"Task" on page 1295

Task: Delete

```
Public Function Delete(ByVal sId As String) As WSInt32
```

Purpose

Delete existing task from Changepoint database.

Parameters

Parameter	Description
sId	ID of task to be deleted.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTask.TaskWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webTask.WSInt32 = proxy.Delete(sId)
```

Task: Exists

```
Public Function Exists(ByVal sId As String) As wsBoolean
```

Purpose

Check if the task existed.

Parameters

Parameter	Description
sId	ID of task to be checked.

Returns

Type	Description
wsBoolean	True if the task exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTask.TaskWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webTask.wsBoolean = proxy.Exists(sId)
```

Related information

"Task" on page 1295

Task: GetBaselineHistory

```
Public Function GetBaselineHistory(ByVal sTaskId As String) As WSDataSet
```

Purpose

Retrieves baseline history information for a task

Parameters

Parameter	Description
sTaskId	ID of the Task

Returns

Type	Description
WSDataset	WSDataset containing all the fields in the DS_TaskBaseline view in the database. Access the dataset through the .Value property of WSDataSet.

Remarks

Contains the current baseline and all historical baselines for the task in descending order by createdon date. Check return object WSException.HaveErrors before reading the value.

Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTask.TaskWse
'set the SOAP header with UsernameToken which include login user and access
token
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webTask.WSDataset = proxy.GetBaselineHistory(sTaskId)
...
```

Related information

"Task" on page 1295

Task: GetById

```
Public Function GetById(ByVal sId As String) As wsTask
```

Purpose

Retrieve a Task object from the Changepoint database.

Parameters

Parameter	Description
sId	ID of task to be retrieved.

Returns

Type	Description
WSTask	Returns an ApiTask object. Access the object through the .Value property of WSTask.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTask.TaskWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webTask.WSTask = proxy.GetById(sId)
```

Related information

"Task" on page 1295

Task: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As WSString
```

Purpose

Returns the TaskId based on the UDF Text field and value.

Parameters

Parameter	Description
sUDFField	UDF text field name (for example, Text1)
sUDFValue	UDF text field value

Returns

Type	Description
WSString	String with the TaskId or an empty string. Access the returned string through the .Value property of WSString

Remarks

None

Example

Not available

Related information

"Task" on page 1295

Task: GetList

```
Public Function GetList(ByVal iRetRows As Int16) As WSDataset
```

Purpose

Retrieve Task list from Changepoint database.

Parameters

Parameter	Description
iRetRows	The number of records to be returned. To return all, you must pass in -1.

Returns

Type	Description
WSDataset	WSDataset of tasks. Access the dataset through the .Value property of WSDataSet.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTask.TaskWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As webTask.WSDataset = proxy.GetList(100)
```

Related information

"Task" on page 1295

Task: GetUDF

```
Public Function GetUDF(ByVal entity As CPEntity, ByVal retOption As
CPUDFReturnType, ByVal entityId As String, ByVal actionResourceId As String)
As WSString
```

Purpose

Retrieve Task UDF (configurable field) information.

Parameters

Parameter	Description
entity	CPEntity, CPEntity.Task = 5.
retOption	The CPUDFReturnType Values are: • WithStructure = 0 – returns UDF field data with full field structure • OnlyValues = 1 – returns only UDF fields with data, does not include any field structure or options • OnlyStructure = 2 – returns only UDF XML structure, no field data • OnlyStructureAndOptions = 3 – returns UDF structure and option data without field data • WithStructureAndOptions = 4 – returns UDF field data with full structure and all field options (except entity-based and conditional codes)
entityId	Task ID for the UDF.
actionResourceId	ResourceId for calling the function, default is login userid.

Returns

Type	Description
WSString	XML string of UDF information. Access the returned string through the .Value property of WSString.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTask.TaskWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webTask.wsString = proxy.GetUDF(5, 1, sId, "")
```

Related information

"Task" on page 1295

Task: HasAssignments

```
Public Function HasAssignments(ByVal sId As String) As WSBoolean
```

Purpose

Check if the task has assignments.

Parameters

Parameter	Description
sId	Task ID for checking.

Returns

Type	Description
WSBoolean	True if the task has assignments, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTask.TaskWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webTask.wsBoolean = proxy.HasAssignments(sId)
```

Related information

"Task" on page 1295

Task: SaveBaseline

```
Public Function SaveBaseline(ByVal sTaskId As String) As WSInt32
```

Purpose

Saves current PlannedStart, PlannedFinish, PlannedHours information for a task into the baseline fields.

Parameters

Parameter	Description
sTaskId	ID of the Task

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Current Planned information is saved to the Baseline fields in the task and a new record is created in the Baselines table. Check return object WSException.HaveErrors before reading the value.

Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTask.TaskWse
'set the SOAP header with UsernameToken which include login user and access
token
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webTask.WSInt32 = proxy.SaveBaseline(sTaskId)
...
```

Related information

"Task" on page 1295

Task: SaveUDF

```
Public Function SaveUDF(ByVal sXMLUDF As String, ByVal needValidate As
Boolean, ByVal bypassMetadata As CPMetadataCheck) As WSInt32
```

Purpose

Save (Insert/Update) UDF information for Task object.

Parameters

Parameter	Description
sXMLUDF	UDF fields in XML string
needValidate	Flag for validation required.
bypassMetadata	Options for Metadata checking. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTask.TaskWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sXML as String = "<root><udf>…</root>"
Dim oRet As webTask.wsInt32 = proxy.SaveUDF(sXML, True, 0)
```

Related information

"Task" on page 1295

Task: Update

```
Public Function Update(ByVal oTask As ApiTask) As WSInt32
```

Purpose

Update Task object to Changepoint database.

Parameters

Parameter	Description
Task	ApiTask object to be updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The Task object will be updated with all the properties assigned. Note that the Baseline fields are no longer saved as part of the Update method. Instead, see the new SaveBaselines( ) method.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTask.TaskWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webTask.wsTask = proxy.GetById(sId)
If Not oRet.WSException.HaveErrors then
Dim objApi As webTask.APITask = oRet.value
```

```
With ojbApi
'update with new data
.WorkCode = New webTask.Identity()
.WorkCode.Id= "{…}"
.WorkCodeCategory = New webTask.Identity()
.WorkCodeCategory.Id = "{…}"
…
End With
Dim iRet as webTask.WSInt32 = proxy.Update(objApi)
End If
```

Related information

"Task" on page 1295

TaskAssignment

The TaskAssignment object allows users to add, retrieve, update or delete TaskAsssignment information within the Changepoint database.

Namespace

`http://changepoint.com/changepoint/CPWebService/TaskAssignment`

URL

`http://webserver/CPWebService/Objects/CPTaskAssignment/TaskAssignment.asmx`

Methods

Properties

For more information, see "ApiTaskAssignment" on page 691.

Related information

"ApiTaskAssignment XML" on page 694

"Project" on page 1143

"Task" on page 1295

TaskAssignment: Add

```
Public Function Add(ByRef sId As String, ByVal oTaskAssignment As
ApiTaskAssignment) As WSInt32
```

Purpose

Add a new TaskAssignment to the Changepoint database.

Parameters

Parameter	Description
sId	TaskAssignment ID for new TaskAssignment, if it is not assigned value before passing, the system will create a new GUID and return it.
oTaskAssignment	ApiTaskAssignment object which will be added.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

When "-15" is returned, it indicates that task planned hours is updated with the total planned hours of task assignments. This is a necessary step for the API to update the task planned

hours, so this should be treated as notification.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim objApi As New webTaskAssignment.ApiTaskAssignment()
With ojbApi
.Task=New webTask.Identity()
.Task.Id= "{…}"
.Resource=New webTask.Identity()
.Resource.Id= "{…}"
.WorkCode=New webTask.Identity()
.WorkCode.Id= "{…}"
.WorkCodeCategory =New webTask.Identity()
.WorkCodeCategory.Id = "{…}"
...
End With
Dim proxy As New webTaskAssignment.TaskAssignmentWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = ""
Dim oRet As webTaskAssignment.WSInt32 = proxy.Add(sId, objApi)
```

Related information

"TaskAssignment" on page 1309

TaskAssignment: Delete

```
Public Function Delete(ByVal sId As String) As WSInt32
```

Purpose

Delete existing TaskAssignment from Changepoint database.

Parameters

Parameter	Description
sId	TaskAssignment ID for deleting.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTaskAssignment.TaskAssignmentWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webTask.WSInt32 = proxy.Delete(sId)
```

Related information

"TaskAssignment" on page 1309

TaskAssignment: Exists

```
Public Function Exists(ByVal sId As String) As wsBoolean
```

Purpose

Check if the TaskAssignment exists.

Parameters

Parameter	Description
sId	TaskAssignment ID for checking.

Returns

Type	Description
WSBoolean	True if the task assignment exists, else False. Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTaskAssignment.TaskAssignmentWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webTask.wsBoolean = proxy.Exists(sId)
```

Related information

"TaskAssignment" on page 1309

TaskAssignment: GetBaselineHistory

```
Public Function GetBaselineHistory(ByVal sTaskAssignmentId As String) As
WSDataSet
```

Purpose

Retrieves Baseline History information for a TaskAssignment

Parameters

Parameter	Description
sTaskAssignmentId	ID of the TaskAssignment

Returns

Type	Description
WSDataset	WSDataset containing all the fields in the DS_TaskAssignmentBaseline view in the database. Access the dataset through the .Value property of WSDataSet.

Remarks

Contains the current baseline and all historical baselines for the task in descending order by createdon date. Check return object WSException.HaveErrors before reading the value.

Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTaskAssignment.TaskAssignmentWse
'set the SOAP header with UsernameToken which include login user and access
token
UserToken.SetToken(myProxy, mUserName, mPassword)
Dim oRet As webProject.WSDataset = proxy.GetBaselineHistory(sTaskAssignmentId)
...
```

Related information

"TaskAssignment" on page 1309

TaskAssignment: GetById

```
Public Function GetById(ByVal sId As String) As wsTask
```

Purpose

Retrieve TaskAssignment object from Changepoint database.

Parameters

Parameter	Description
sId	ID of TaskAssignment which will be retrieved.

Returns

Type	Description
WSTask	Returns an ApiTask object. Access the object through the .Value property of WSTask.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTaskAssignment.TaskAssignmentWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webTaskAssignment.WSTaskAssignment = proxy.GetById(sId)
```

Related information

"TaskAssignment" on page 1309

TaskAssignment: GetIdByUDFText

```
Public Function GetIdByUDFText(ByVal sUDFField As String, ByVal sUDFValue As
String) As WSString
```

Purpose

Returns the TaskAssignmentId based on the UDF Text field and value.

Parameters

Parameter	Description
sUDFField	UDF text field name (for example, Text1)
sUDFValue	UDF text field value

Returns

Type	Description
WSString	String with the TaskAssignmentId or an empty string. Access the returned string through the .Value property of WSString

Remarks

None

Example

Not available

Related information

"TaskAssignment" on page 1309

TaskAssignment: GetList

```
Public Function GetList(ByVal iRetRows As Int16) As WSDataset
```

Purpose

Retrieve TaskAssignment list from Changepoint database.

Parameters

Parameter	Description
iRetRows	The number of records to be returned. To return all, you must pass in -1.

Returns

Type	Description
WSDataset	WSDataset of task assignments. `Access the dataset through the .Value property of WSDataSet.`

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTaskAssignment.TaskAssignmentWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim oRet As webTaskAssignment.WSDataset = proxy.GetList(100)
```

Related information

"TaskAssignment" on page 1309

TaskAssignment: GetUDF

```
Public Function GetUDF(ByVal entity As CPEntity, ByVal retOption As
CPUDFReturnType, ByVal entityId As String, ByVal actionResourceId As String)
As WSString
```

Purpose

Retrieve TaskAssignment UDF (configurable field) information.

Parameters

Parameter	Description
entity	CPEntity, CPEntity.TaskAssignment = 6.
retOption	The CPUDFReturnType Values are: • WithStructure = 0 – returns UDF field data with full field structure • OnlyValues = 1 – returns only UDF fields with data, does not include any field structure or options • OnlyStructure = 2 – returns only UDF XML structure, no field data • OnlyStructureAndOptions = 3 – returns UDF structure and option data without field data • WithStructureAndOptions = 4 – returns UDF field data with full structure and all field options (except entity-based and conditional codes)

Parameter	Description
entityId	TaskAssignment ID for the UDF.
actionResourceId	ResourceId for calling the function, default is login userid.

Returns

Type	Description
WSString	XML string of UDF information. Access the returned string through the .Value property of WSString.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTaskAssignment.TaskAssignmentWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webTaskAssignment.wsString = proxy.GetUDF(6, 1, sId, "")
```

Related information

"TaskAssignment" on page 1309

TaskAssignment: SaveUDF

```
Public Function SaveUDF(ByVal sXMLUDF As String, ByVal needValidate As
Boolean, ByVal bypassMetadata As CPMetadataCheck) As WSInt32
```

Purpose

Save (Insert/Update) UDF information for TaskAssignment object.

Parameters

Parameter	Description
sXMLUDF	The UDF fields in XML string
needValidate	The flag for validation required.
bypassMetadata	The options for Metadata checking. Value range: • CPMetadataCheck.CheckAll = 0 • CPMetadataCheck.BypassAll = 1 • CPMetadataCheck.OnlyMandatory = 2

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTaskAssignment.TaskAssignmentWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sXML as String = "<root><udf>…</root>"
Dim oRet As webTask.wsInt32 = proxy.SaveUDF(sXML, True, 0)
```

Related information

"TaskAssignment" on page 1309

TaskAssignment: StatusTask

```
Public Function StatusTask(ByVal sTaskAssignmentId As String, ByVal
dRemainingHours As Double, ByVal dtForecastStart As DateTime, ByVal
```

```
dtForecastFinish As DateTime, ByVal dPercentComplete as Double, ByVal
sResComments As String, ByVal sMgrComments As String) As WSBoolean
```

Purpose

Provides ability to status a task.

Parameters

Parameter	Description
sTaskAssignmentId	Task assignment ID for statusing.
dRemainingHours	Remaining hours to be set for the task.
dtForecastStart	Forecast start date to be set for the task.
dtForecastFinish	Forecast finish date to be set for the task.
dPercentComplete	Percent complete value to be set for the task.
sResComments	Resource comments
sMgrComments	Manager comments

Returns

Type	Description
WSBoolean	True if the task status was updated, else False Access the returned Boolean object through the .Value property of WSBoolean.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there has been an error.

To pass in a non value for dRemainingHours or for dPercentComplete you must pass in -1. When -1 is passed in for dRemainingHours then the Percent complete will be calculated. If -1 is passed in for dPercentComplete then Remaining Hours will be calculated.

If both Remaining hours and Percent complete are provided, the Remaining hours will be honored.

Task can only be statused by the resource assigned to the task or the project manager or the project plan editor.

Manager comments can only be set by the project manager

Example

```
Dim proxy As New webTaskAssignment.TaskAssignmentWse()
'set the SOAP header with UsernameToken which include login user and
access token
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim dRHours as Double = 40.0
Dim dRHours as DateTime = "5/18/2007"
Dim dtFFinish as DateTime = "6/19/2007"
Dim dPComplete as Double = 25.0
Dim sRComments as String = "Resource comments"
Dim sMComments as String = "Manager comments"

Dim iRet as webTaskAssignment.wsBoolean = proxy.StatusTask(sId, dRHours,
dFStart, dFFinish, dPComplete, sRComments, sMComments )
```

Related information

"TaskAssignment" on page 1309

TaskAssignment: Update

```
Public Function Update(ByVal oTaskAssignment As ApiTaskAssignment) As WSInt32
```

Purpose

Update TaskAssignment object to Changepoint database.

Parameters

Parameter	Description
oTaskAssignment	ApiTaskAssignment object which will be updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

When "-15" is returned, it indicates that task planned hours is updated with the total planned hours of task assignments. This is necessary step for the API to update the task planned hours, so this should be treated as notification.

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and logs if there is an error.

Example

```
Dim proxy As New webTaskAssignment.TaskAssignmentWse()
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, mUserName, mPassword)
Dim sId as String = "{…}"
Dim oRet As webTaskAssignment.wsTaskAssignment = proxy.GetById(sId)
If Not oRet.WSException.HaveErrors then
Dim objApi As webTaskAssignment.APITaskAssignment = oRet.value
With ojbApi
'update with new data
.WorkCode = New webTaskAssignment.Identity()
.WorkCode.Id= "{…}"
.WorkCodeCategory = New webTaskAssignment.Identity()
.WorkCodeCategory.Id = "{…}"
...
End With
Dim iRet as webTaskAssignment.WSInt32 = proxy.Update(objApi)
End If
```

Related information

"TaskAssignment" on page 1309

Time

The Time object allows users to add, retrieve, update, approve or reject time information, and submit time, request time and standard time information for resources within the Changepoint

database.

Namespace

`http://changepoint.com/changepoint/CPWebService/Time`

URL

`http://webservice/CPWebService/Objects/CPTime/Time.asmx`

Methods

Properties

For more information, see "ApiTime" on page 709.

Time: Add

```
Public Function Add(ByVal oTime As ApiTime) As WSString
```

Purpose

Adds a new time record into Changepoint.

Parameters

Parameter	Description
oTime	ApiTime object, which contains time information that will be added.

Returns

Type	Description
WSString	Returns TimeId as a string. Access the returned string through the .Value property of WSString.

Remarks

Returns empty if error occurred and the error is written to the log file. If returned empty, oTime is null, no time is to be added.

Example

```
Dim proxy As New WebTime.TimeWse
Dim mTime As New WebTime.ApiTime
Dim sRet As String

UserToken.SetToken(proxy, mUserName, mPassword)
With mTime
   .AdjustedOvertimeHours = 0
   .AdjustedRegularHours = 0
   .AdjustmentStatus = False
   .BookedByResourceId = "{7E4F8CE9-363B-11D4-8AB1-0001023D3221}"
   .Description = "Case 1"
   .OvertimeHours = 2
   .ResourceId = "{7E4F8CE9-363B-11D4-8AB1-0001023D3221}"
```

```
    .RegularHours = 8
    .StartTime = "5/18/2007"
    .TaskId = "{61396453-4DBF-4FD5-9F46-299BD2374A35}"
    .TimeDate = "5/18/2007"
    .WorkCodeCategoryId = "{AA9C83C1-6E1E-4974-9DEC-7C554CC429D2}"
    .WorkCodeId = "{E716CDE8-72A0-481D-9D29-99FBD74A9AFA}"
    .WorkLocationGroupID = "{6AAF1A2D-2535-40E3-8D03-07801AE85E41}"
    .WorkLocationID = "{0175FA25-7CAA-4DC3-BB95-62A37A36134F}"
End With

sRet = proxy.Add(mTime).Value
```

Related information

"Time" on page 1322

Time: ApproveRejectTimes

```
Public Function ApproveRejectTimes(ByVal ResourceId As String, ByVal TaskId As
String, ByVal StartDate As Date, ByVal EndDate As Date, ByVal bAction As
Boolean, ByVal Reason As String) As WSInt32
```

Purpose

Approve or reject time records whose StartTime falls between the dates specified in the daStartDate and daEndDate parameters

Parameters

Parameter	Description
sResourceId	Changepoint assigned Resource ID. Cannot be empty.
sTaskId	Task ID in Changepoint. Cannot be empty.
daStartDate	Start date.
daEndDate	End date.
bAction	0 for rejection. 1 for approval.
Reason	The Reason cannot be empty if bAction equals zero.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None

Example

Not available

Related information

"Time" on page 1322

Time: CreateByXML

```
Public Function CreateByXML(ByVal sXML As String, ByRef sId As String) As
WSInt32
```

Purpose

Create a Time record using an XML string of the Time object in Changepoint.

Parameters

Parameter	Description
sXML	A new Time XML string.
sId	ByRef parameter that returns the new TimeId after the Time has been added.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

ApiTime XML structure can be obtained by GetXMLStructure or GetByXML methods.

The ByPassMetadataCheck switch will stop any meta data validation in Time.

Example

```
Dim proxy As New WebTime.TimeWse
Dim wsRet As WebTime.WSInt32
Dim sMyXML As String = ""
Dim iRet As Int32 = 0
UserToken.SetToken(proxy, mUserName, mPassword)
'Get Time XML structure
sMyXML = proxy.GetXMLStructure().Value
'populate sMyXML with data and then pass it to CreateByXML
'
wsRet = proxy.CreateByXML(sMyXML)
```

Related information

"Time" on page 1322

"Time: GetByXML" on page 1330

"Time: GetXMLStructure" on page 1340

"ApiTime XML" on page 713

Time: Delete

```
Public Function Delete(ByVal sId as String) As WSInt32
```

Purpose

Delete the specified Time

Parameters

Parameter	Description
sId	The ID of the Time to delete.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

If sId is an empty string, the object's TimeId is used; otherwise, an error is returned.

Because the method "falls back" always ensure the TimeId is passed unless the Time (that the object refers to) is the Time to be deleted.

Example

```
Dim proxy As New WebTime.TimeWse
Dim iRet as Int32 = 0

UserToken.SetToken(proxy, mUserName, mPassword)
iRet = proxy.Delete("{f012628c-3717-11d4-8e11-00105a9e2ddf}")
If iRet = 0 Then
   ...'Continue processing
Else
   ...'Handle the error
End If
```

Related information

"Time" on page 1322

Time: Exists

```
Public Function Exists(ByVal sId as String) As WSBoolean
```

Purpose

Checks whether the specified TimeId exists in Changepoint.

Parameters

Parameter	Description
sId	TimeId

Returns

Type	Description
WSBoolean	True if the specified Time exists. Access the returned Boolean through the .Value property of WSBoolean.

Remarks

The method does not "fall back" and use the object's TimeId if the parameter is an empty string. The method will return false.

Example

Not available

Related information

"Time" on page 1322

Time: GetById

```
Public Function GetById(ByVal sId as String) As WSTime
```

Purpose

Retrieve current Time data into the object

Parameters

Parameter	Description
sId	The ID of the Time whose data is being retrieved.

Returns

Type	Description
WSTime	Returns an ApiTime object. Access the object through the .Value property of WSTime.

Remarks

None

Example

Not available

Related information

"Time" on page 1322

Time: GetByXML

```
Public Function GetByXML(ByVal sXML as String, ByVal sTimeId as String) As
WSString
```

Purpose

Retrieve current Time data based on the Time ID passed and the fields in the XML string.

Parameters

Parameter	Description
sXML	The XML string containing fields where data is required.
sTimeId	The Time whose data is being retrieved.

Returns

Type	Description
WSString	XML string of Time data. Access the returned string through the .Value property of WSString.

Remarks

The XML string sXML can be varied in the number of fields. If only a small subset of data is required, any unwanted fields can be removed from sXML. The returned XML string will mirror sXML in the fields contained in the XML.

Example

Not available

Related information

"Time" on page 1322

Time: GetList

```
Public Function GetList(ByVal iRetRows as Int16) As WSDataSet
```

Purpose

Retrieve a list of Time records based on the parameter iRetRows.

Parameters

Parameter	Description
iRetRows	The number of records to be returned. To return all, you must pass in -1.

Returns

Type	Description
WSDataSet	WSDataSet with columns in the Time table. Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Time" on page 1322

Time: GetTasksWithinPeriod

```
Public Function GetTasksWithinPeriod(ByVal StartDate As Date, ByVal EndDate As
Date) As WSDataSet
```

Purpose

Retrieve task information in time records, where the start time falls between the dates specified in the daStartDate and daEndDate parameters.

Parameters

Parameter	Description
daStartDate	Cannot be empty.
daEndDate	Cannot be empty.

Returns

Type	Description
WSDataSet	WSDataSet (TaskId, TaskName) Access the returned dataset through the .Value property of WSDataSet.

Remarks

Errors are returned in the .WSException property of WSDataSet.

Example

Not available

Related information

"Time" on page 1322

Time: GetTimeByRes

```
Public Function GetTimeByRes(ByVal ResourceId As String, ByVal StartDate As
Date, ByVal EndDate As Date) As WSDataSet
```

Purpose

Retrieve time records, where the start time falls between the dates specified in the daStartDate and daEndDate parameters for the target resource.

Parameters

Parameter	Description
ResourceId	Changepoint assigned Resource ID
StartDate	Start date.
EndDate	End date.

Returns

Type	Description
WSDataSet	WSDataSet (TimeId, TaskId, ResourceId, Description). Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Time" on page 1322

Time: GetTimeByTask

```
Public Function GetTimeByTask(ByVal TaskId As String, ByVal StartDate As Date,
ByVal EndDate As Date) As WSDataSet
```

Purpose

Retrieve time records, where the start time falls between the dates specified in the StartDate and EndDate Parameters for the target task.

Parameters

Parameter	Description
TaskId	Changepoint created task Id.
StartDate	Start date.
EndDate	End date.

Returns

Type	Description
WSDataSet	WSDataSet (TimeId, TaskId, ResourceId, Description). Access the dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Time" on page 1322

Time: GetTimeCodes

```
Public Function GetTimeCodes(ByVal GlobalWorkgroupId As String, ByVal
WorkgroupId As String, ByVal CodeType As String) As WSDataSet
```

Purpose

Retrieve time code information for a workgroup.

Parameters

Parameter	Description
GlobalWorkgroupId	GlobalWorkgroup Id.
WorkgroupId	Workgroup Id.
CodeType	Code Types: • CODE1 • CODE2 • CODE3

Returns

Type	Description
WSDataSet	WSDataSet of time code information. Access the returned dataset through the .Value property of WSDataSet.

Remarks

Errors are accessed through the .WSException property in WSDataSet.

Example

Not available

Related information

"Time" on page 1322

Time: GetTimeIdsWithinPeriod

```
Public Function GetTimeIdsWithinPeriod(ByVal StartDate As Date, ByVal EndDate
As Date) As WSDataSet
```

Purpose

Retrieve time records, where the start time falls between the dates specified in the StartDate and EndDate Parameters.

Parameters

Parameter	Description
StartDate	Start date
EndDate	End date

Returns

Type	Description
WSDataSet	WSDataSet (TimeId) Access the returned dataset through the .Value property of WSDataSet

Remarks

None.

Example

Not available

Related information

"Time" on page 1322

Time: GetWorkCodeCategories

```
Public Function GetWorkCodeCategories(ByVal sResourceId As String, ByVal
sTaskId As String, Optional ByVal sSearchString As String = "") As WSDataSet
```

Purpose

Returns a lists of work code categories based on the project or task level settings.

Parameters

Parameter (*=required)	Description
sResourceId	Resource ID
sTaskId	Task ID
sSearchString	Search string

Returns

A dataset with two columns (WorkCodeCategoryId, Name). Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Time" on page 1322

Time: GetWorkCodes

```
Public Function GetWorkCodes(ByVal sResourceId As String, ByVal sTaskId As
String, ByVal sWorkCodeCategoryId As String, Optional ByVal sSearchString As
String = "") As WSDataSet
```

Purpose

Returns a list of the work codes based on project or task level settings.

Parameters

Parameter (*=required)	Description
*sResource Id	Resource ID
*sTaskId	Task ID
*sWorkCodeCategoryId	Work code category ID
sSearchString	Search string

Returns

A dataset with two columns (WorkCodeId, Name). Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Time" on page 1322

Time: GetWorkLocationGroups

```
Public Function GetWorkLocationGroups(ByVal sResourceId As String, ByVal
sTaskId As String, Optional ByVal sSearchString As String = "") As WSDataSet
```

Purpose

Returns a list of work location groups based on the project or task level settings.

Parameters

Parameter (*=required)	Description
*sResourceId	Resource ID
*sTaskId	Task ID
sSearchString	Search string

Returns

A dataset with two columns (WorkLocationGroupId, Name). Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Time" on page 1322

Time: GetWorkLocations

```
Public Function GetWorkLocations(ByVal sResourceId As String, ByVal sTaskId As
String, ByVal sWorkLocationGroupId As String, Optional ByVal sSearchString As
String = "") As WSDataSet
```

Purpose

Returns a list of work locations based on the project level settings.

Parameters

Parameter (*=required)	Description
*sResourceId	ResourceID
*sTaskId	Task ID
*sWorkLocationGroupId	Work location group ID
sSearchString	Search string

Returns

A dataset with two columns (WorkLocationId, Name). Access the returned dataset through the .Value property of WSDataSet.

Remarks

None

Example

Not available

Related information

"Time" on page 1322

Time: GetXMLStructure

```
Public Function GetXMLStructure()As WSString
```

Purpose

Retrieve the XML structure for the Time object.

Parameters

None

Returns

Type	Description
WSString	An XML string of the Time object. Access the returned string through the .Value property of WSString.

Remarks

None

Example

Not available

Related information

"Time" on page 1322

"ApiTime XML" on page 713

Time: SubmitTimes

```
Public Function SubmitTimes(ByVal ResourceId As String, ByVal TaskId As
String, ByVal StartDate As Date, ByVal EndDate As Date) As WSInt32
```

Purpose

Submit time, request time and standard time where the start time falls between the dates specified in the StartDate and EndDate parameters for a target resource.

Parameters

Parameter	Description
ResourceId	Changepoint assigned Resource Id. Cannot be empty.
TaskId	Task ID in Changepoint. This is an unused parameter and should be defaulted to a null ID, e.g. 00000000-0000-0000-0000-000000000000.
StartDate	Start date.
EndDate	End date.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

None.

Example

Not available

Related information

"Time" on page 1322

Time: Update

```
Public Function Update(ByVal oTime As ApiTime) As WSInt32
```

Purpose

Update the general information of a time record.

Parameters

Parameter	Description
oTime	ApiTime object, which contains time information that will be updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Error number and message is written to the log file if the error occurs.

Example

```
Dim proxy As New WebTime.TimeWse
Dim mTime As New WebTime.ApiTime
Dim iRet As Int32

UserToken.SetToken(proxy, mUserName, mPassword)
mTime = proxy.GetTimeById("{A0A6F42C-B38D-4C45-AC6F-72EB638ADB62}").value

With mTime
   .OvertimeHours = 4
End With

iRet = proxy.Update(mTime).Value
```

Related information

"Time" on page 1322

Time: UpdateByXML

```
Public Function UpdateByXML(ByVal sXML As String, ByVal sTimeId As String) As
WSInt32
```

Purpose

Update the database using an XML string of the Time object containing update info.

Parameters

Parameter	Description
sXML	Time XML string with updated fields.
sTimeId	Time being updated.

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

The ApiTime XML structure can be obtained by the GetXMLStructure or GetByXML methods.

The update process will update all fields in Time.

The ByPassMetadataCheck switch will stop any meta data validation in Time.

It is recommended to remove the XML tags if you want to retain the original data in the database.

The method uses the following sequence to find the Time ID:

1. If the sTimeId parameter is passed in, the method uses this value for the Time ID.
2. If this fails, the method attempts to extract the Time ID from <timeid> in the XML.
3. If this fails, the Time ID is taken from the object properties.

Example

```
Dim proxy As New WebTime.TimeWse
Dim sMyXML As String = ""
Dim wsRet As WebTime.WSInt32
Dim sReqId As String = ""
sReqId = "{CE26F116-8FCF-4697-A0BF-00118BC3059A}"
UserToken.SetToken(proxy, mUserName, mPassword)
'Get Time XML structure with existing data if it's necessary.
sMyXML = proxy.GetByXML(sReqId).Value
'modify sMyXML and then pass it to UpdateByXML
'
wsRet = proxy.UpdateByXML(sMyXML)
```

Related information

"Time" on page 1322

"Time: GetByXML" on page 1330

"Time: GetXMLStructure" on page 1340

"ApiTime XML" on page 713

WSLogin

The WSLogin object provides a token credential to access Web Services API for valid Changepoint users.

Namespace

`http://changepoint.com/changepoint/CPWebService/WSLogin`

URL

`http://webserver/CPWebService/WSLogin.asmx`

Methods

WSLogin: GetVersion

```
Public Function GetVersion() As WSString
```

Purpose

Returns the API version number

Parameters

None

Returns

Type	Description
WSString	Welcome message with the Web Services API version number. Access the returned string through the .Value property of WSString.

Remarks

None

Example

Not available

Related information

"WSLogin" on page 1345

WSLogin: Login

```
Public Function Login(ByVal userLoginId As String) As WSUser
```

Purpose

Returns WebUser with Token to access the Web Services API.

Parameters

Parameter	Description
userLoginId	Resource email address in the Changepoint Resources table.

Returns

Type	Description
WSUser	Returns an ApiUser object. Access the object through the .Value property of WSUser.

Remarks

This function must be called the first time a user connects to the Web Services API or when the Login Token has expired.

This function should only be called from a SOAP request when the Username Token has been set in the SOAP header.

Example

```
Dim proxy As New webLogin.WSLoginWse
```

```
'set SOAP header with user name and password
UserToken.SetLoginToken(proxy, txtUserName.Text, txtPassword.Text)
Dim loginUser As webLogin.WSUser = proxy.Login(txtUserName.Text)
If Not loginUser.WSException.HaveErrors Then
   Dim myUser As webLogin.APIUser = loginUser.value
   MainForm.mUser = myUser
   MainForm.mUserName = myUser.UserId
   MainForm.mPassword = myUser.AccessToken
   MainForm.mGWorkgroupId = myUser.GlobalWorkgroupId
   MainForm.mWorkgroupId = myUser.WorkgroupId
End If

'After login using this Token to call web service to insert new Project
Dim objApi As New webProject.APIProject
With ojbApi
.Name="Richard Project36"
.Customer=New webProject.Identity()
.Customer.Id= "{…}"
.Engagement=New webProject.Identity()
.Engagement.Id = "{…}"
.Currency="CAD"
...
End With
Dim proxy As New webProject.ProjectWse
'set the SOAP header with UsernameToken
UserToken.SetToken(proxy, MainForm.mUserName, MainForm.mPassword)
Dim sId as String = ""
Dim oRet As webProject.WSInt32 = proxy.Add(sId, objApi)
```

Related information

"SOAP header example" on page 773

"WSLogin" on page 1345

"ApiUser class" on page 1351

WSLogin: TestConnection

```
Public Function TestConnection() As WSInt32
```

Purpose

Tests the web server is connected with Changepoint database.

Parameters

None

Returns

Type	Description
WSInt32	0 = Success nonzero = Error Access the returned Int32 through the .Value property of WSInt32.

Remarks

Check return object WSException.HaveErrors before reading the value. Check WSException.Message and Logs if there is an error.

If the integer value 0 is returned, it means the connection is good.

Example

Not available

Related information

"WSLogin" on page 1345

Changepoint custom classes

Namespace:

`changepoint.com/changepoint/CPWebService`

Properties are listed below by class.

CodeItems class

Property	Type	Description
Item()	CodeItem	Code Item

CodeItem class

Property	Type	Description
Selected	Boolean	Flag of selected
Text	String	Displayed text in code
Value	String	Value of code

CPCode class

Property	Type	Description
Caption	String	Label of code
Changed	Boolean	Flag of changes
CodeItems()	CodeItems	Code Item list
Conditional	Boolean	Conditional flag
DuplicateCheck	CPDuplicateType	User defined type
EntityBased	Boolean	Entity based flag
FieldFormat	CPFieldType	User defined type
FieldName	String	Code1 ~ CodeN
Hidden	Boolean	Metadata flag
ItemName	String	For example, EngagementCode1...
LockedAfterEntry	Boolean	Metadata flag
Mandatory	Boolean	Metadata flag
Maximum	Decimal	Metadata value
Minimum	Decimal	Metadata value
NotEditable	Boolean	Editable flag

Property	Type	Description
SelectedValue	String	Selected code
UnUsed	Boolean	Metadata flag
MultiSelect	Boolean	MultiSelect

CPFields class (collection of CPFieldItem)

Property	Type	Description
Item()	CPFieldItem	Field Item

CPFieldItem class

Property	Type	Description
Enabled	Boolean	Enabled
FieldFormat	CPFieldType	FieldFormat
FieldName	String	Field Name
Group	String	Group
Index	Integer	Index
Item	Object	Item

CPLog class

Property	Type	Description
Caption	String	Caption
Level	Bite	Level
LogTime	DateTime	Log Time
Msg	String	Msg

CPText class

Property	Type	Description
Caption	String	Label of text
Changed	Boolean	Flag of changes
DuplicateCheck	CPDuplicateType	User defined type
FieldFormat	CPFieldType	User defined type
FieldName	String	Metadata flag
Hidden	Boolean	Metadata flag
ItemaName	String	Metadata flag
LockedAfterEntry	Boolean	Metadata flag
Mandatory	Boolean	Metadata flag
Maximum	Decimal	Metadata value
Minimum	Decimal	Metadata value
NotEditable	Boolean	Metadata flag
Traced	Boolean	Traced flag
UnUsed	Boolean	Metadata flag
Value	Object	Current value

ApiUser class

For more information, see the "ApiUser" section on page 734.

The PublicKey in the web.config file provides you with the ability to make the token unique.

Identity class

For more information, see the "Identity" section on page 735.

Signature class

For more information, see the "Signature" section on page 736.

Changepoint custom data types

Namespace: changepoint.com/changepoint/CPWebService

Values are listed below by data type

CPDuplicateType

```
Public Enum CPDuplicateType As Byte
```

Value	Duplicate type
0	NotCheck
1	WarnIfDuplicate
2	RejectIfDuplicate
3	WarnIfSimilar

CPFieldType

```
Public Enum CPFieldType As Byte
```

Value	Field type
0	UnKnown
1	Code
2	Text
3	Numeric
4	DateTime

CPLogLevel

```
Public Enum CPLogLevelType As Byte
```

Value	Log level	Description
0	NoLog	No log is created in the log file
1	Source	Object and method information
2	ErrorMessage	Error message
3	InputParameters	Input parameter
4	Returns	Returns value
8	Checkpoint	Internal checkpoint

CPObjectAction

```
Public Enum CPObjectAction As Byte
```

Value	Object action
0	GetValues
1	Create
2	Update
3	Delete
6	GetStatus
9	Unchange

CPMetadataCheck

```
Public Enum CPMetadataCheck As Byte
```

Value	Metadata check
0	CheckAll
1	BypassAll
2	OnlyMandatory

ProductCalculationBaseEnum

```
Public Enum ProductCalculationBaseEnum
   Discount
   AdjustedPrice
End Enum
```

Value	Description
Discount	AdjustedPrice property is calculated based on the value provided in the Discount property.
AdjustedPrice	Discount property is calculated based on the value provided in the AdjustedPrice property.

ServiceCalculationBase

```
Public Enum ServiceCalculationBase
   Discount
   NegotiatedRate
End Enum
```

Value	Description
Discount	NegotiatedBillingRate property is calculated based on the value provided in the Discount property
NegotiatedRate	Discount property is calculated based on the value provided in the NegotiatedBillingRate property. The default value of ServiceCalculationBase is NegotiatedRate.

Web Services API return types

Namespace: changepoint.com/changepoint/CPWebService

Properties are listed below by return type

WSException

Property		Description
HaveErrors	Boolean	Flag of errors
LogLevel	CPLogLevelType	Log Level
LogPath	String	Log Path
Logs	CPLogs	Logs
Message	String	Error Message
Number	Long	Error Number
Source	String	Error source
Warnings	CPLogs	Warnings

WSAdditionalItemArray

Property	Type	Description
WSException	WSException	Exception
value	ApiInvAdditionalItem()	Array of ApiInvAdditionalItem objects

WSArrayList

Property	Type	Description
WSException	WSException	Exception
value	ArrayList	ArrayList

WSBoolean

Property	Type	Description
WSException	WSException	Exception
value	Boolean	Type of Boolean

WSCodes

Property	Type	Description
WSException	WSException	Exception
value	Cpcodes	Type of CPCodes

WSContact

Property	Type	Description
WSException	WSException	Exception
value	ApiContact	ApiContact object

WSCreditNote

Property	Type	Description
WSException	WSException	Exception
value	ApiCreditNote	ApiCreditNote object

WSCustomer

Property	Type	Description
WSException	WSException	Exception
value	ApiCustomer	ApiCustomer object

WSDataSet

Property	Type	Description
WSException	WSException	Exception
value	DataSet	DataSet

WSDate

Property	Type	Description
WSException	WSException	Exception
value	Date	Date

WSDecimal

Property	Type	Description
WSException	WSException	Exception
value	Decimal	Type of Decimal

WSDouble

Property	Type	Description
WSException	WSException	Exception
value	Double	Type of Double

WSEngagement

Property	Type	Description
WSException	WSException	Exception
value	ApiEngagement	ApiEngagement object

WSEngBillingRate

Property	Type	Description
WSException	WSException	Exception
value	ApiEngBillingRate	ApiEngBillingRate object

WSEngBillingRateHistory

Property	Type	Description
WSException	WSException	Exception
value	ApiEngBillingRateHistory	ApiEngBillingRateHistory object

WSEngFixedFee

Property	Type	Description
WSException	WSException	Exception
value	ApiEngFixedFee	ApiEngFixedFee object

WSEngFixedFeeItem

Property	Type	Description
WSException	WSException	Exception
value	ApiEngFixedFeeItem	ApiEngFixedFeeItem object

WSEngProduct

Property	Type	Description
WSException	WSException	Exception
value	ApiEngProduct	ApiEngProduct object

WSEngProjectedResource

Property	Type	Description
WSException	WSException	Exception
value	ApiEngProjectedResource	ApiEngProjectedResource object

WSEngRequestProcessingRule

Property	Type	Description
WSException	WSException	Exception
value	ApiEngRequestProcessingRule	ApiEngRequestProcessingRule object

WSEngRequestSLA

Property	Type	Description
WSException	WSException	Exception
value	ApiEngRequestSLA	ApiEngRequestSLA object

WSEngRevRec

Property	Type	Description
WSException	WSException	Exception
value	ApiEngRevRec	ApiEngRevRec object

WSEngSplitBillingRule

Property	Type	Description
WSException	WSException	Exception
value	ApiEngSplitBillingRule	ApiEngSplitBillingRule object

WSEngWorkCode

Property	Type	Description
WSException	WSException	Exception
value	ApiEngWorkCode	ApiEngWorkCode object

WSEngWorkLocation

Property	Type	Description
WSException	WSException	Exception
value	ApiEngWorkLocation	ApiEngWorkLocation object

WSExchangeRate

Property	Type	Description
WSException	WSException	Exception
value	ApiExchangeRate	ApiExchangeRate object

WSExpense

Property	Type	Description
WSException	WSException	Exception
value	ApiExpense	ApiExpense object

WSExpenseReport

Property	Type	Description
WSException	WSException	Exception
value	ApiExpenseReport	ApiExpenseReport object

WSExpenseWriteOffArray

Property	Type	Description
WSException	WSException	Exception
value	ApiInvWOExpenses()	Array of ApiInvWOExpenses objects

WSFunction

Property	Type	Description
WSException	WSException	Exception
value	ApiFunctionDescription	ApiFunctionDescription object

WSFunctionSkill

Property	Type	Description
WSException	WSException	Exception
value	CPFunctionSkill	Object of FunctionSkill

WSFunctionSkillArray

Property	Type	Description
WSException	WSException	Exception
value	ApiFunctionSkill()	Array of ApiFunctionSkill objects

WSInteger

Property	Type	Description
WSException	WSException	Exception
value	Integer	Type of Integer

WSInvoice

Property	Type	Description
WSException	WSException	Exception
value	ApiInvoice	ApiInvoice object

WSInvoicedExpenseArray

Property	Type	Description
WSException	WSException	Exception
value	ApiInvExpense()	Array of ApiInvExpense objects

WSInvoicedFixedFeeArray

Property	Type	Description
WSException	WSException	Exception
value	ApiInvFixedFee()	Array of ApiInvFixedFee objects

WSInvoicedProductArray

Property	Type	Description
WSException	WSException	Exception
value	ApiInvProduct()	Array of ApiInvProduct objects

WSInvoicedTimeArray

Property	Type	Description
WSException	WSException	Exception
value	ApiInvTime()	Array of ApiInvTime objects

WSKMVersion

Property	Type	Description
WSException	WSException	Exception
value	ApiKMVersion	ApiKMVersion object

WSKnowledgeManagement

Property	Type	Description
WSException	WSException	Exception
value	ApiKnowledgeManagement	ApiKnowledgeManagement object

WSLong

Property	Type	Description
WSException	WSException	Exception
value	Long	Type of long

WSOpportunity

Property	Type	Description
WSException	WSException	Exception
value	ApiOpportunity	ApiOpportunity object

WSPaymentsInfoArray

Property	Type	Description
WSException	WSException	Exception
value	ApiInvPaymentInfo()	Array of ApiInvPaymentInfo objects

WSProduct

Property	Type	Description
WSException	WSException	Exception
value	ApiProduct	ApiProduct object

WSProductCurrency

Property	Type	Description
WSException	WSException	Exception
value	ApiProductCurrency	ApiProductCurrency object

WSProductTeam

Property	Type	Description
WSException	WSException	Exception
value	ApiProductTeam	ApiProductTeam object

WSProductWriteOffArray

Property	Type	Description
WSException	WSException	Exception
value	ApiInvWOProduct()	Array of ApiInvWOProduct objects

WSProject

Property	Type	Description
WSException	WSException	Exception
value	ApiProject	Object of ApiProject

WSRequest

Property	Type	Description
WSException	WSException	Exception
value	ApiRequest	ApiRequest object

WSRequestTime

Property	Type	Description
WSException	WSException	Exception
value	ApiRequestTime	ApiRequestTime object

WSResource

Property	Type	Description
WSException	WSException	Exception
value	ApiResource	Object of ApiResource

WSSkill

Property	Type	Description
WSException	WSException	Exception
value	ApiSkill	ApiSkill object

WSSkillCategory

Property	Type	Description
WSException	WSException	Exception
value	ApiSkillCategory	ApiSkillCategory object

WSSkillCompetency

Property	Type	Description
WSException	WSException	Exception
value	ApiSkillCompetency	ApiSkillCompetency object.

WSString

Property	Type	Description
WSException	WSException	Exception
value	String	String

WSInvoicedSupportTimeArray

Property	Type	Description
WSException	WSException	Exception
value	ApiInvSupportTime()	Array of ApiInvSupportTime objects

WSSupportTimeWriteOffArray

Property	Type	Description
WSException	WSException	Exception
value	ApiInvWOSupport()	Array of ApiInvWOSupport objects

WSTask

Property	Type	Description
WSException	WSException	Exception
value	ApiTask	ApiTask object

WSTaskAssignment

Property	Type	Description
WSException	WSException	Exception
value	ApiTaskAssignment	ApiTaskAssignment object

WSTexts

Property	Type	Description
WSException	WSException	Exception
value	CpTexts	Type of CPTexts

WSTime

Property	Type	Description
WSException	WSException	Exception
value	ApiTime	ApiTime object

WSTimeWriteOffArray

Property	Type	Description
WSException	WSException	Exception
value	ApiInvWOTime()	Array of ApiInvWOTime objects

WSUser

Property	Type	Description
WSException	WSException	Exception
value	ApiUser	Object of ApiUser

WSXmlNode

Property	Type	Description
WSException	WSException	Exception
value	XmlNode	Type of XmlNode

Web Services API log file

The Changepoint API provides a log file for all methods. The log information will help you to identify any problems and trace user activity.

The default log file folder is C:\Program Files\changepoint\APILogs\; however, you can select another path by setting the LogPath property of the ApiConnection object.

The log file folder must be an existing folder and full control access must be given to all users.

The log file name incorporates the date that the log file was generated. For example, a Web Services API log generated on 2010-12-31 would be named ws101231.log. A COM API log on the same date would be named com101231.log.

> **Note:** The log files do not clear automatically. You must clear them periodically for disk capacity reasons.

Log levels

There are six log levels. Each log level defines a different type of log message. Level 0 means no log. Each level also includes the levels below it. The following table defines each log definition:

Level	Description	Example
0	No log (default)	
1	Source information (object/method name) or some other simple information	Source: COM API call ValidateConnection function.
2	Exception number/message or some checkpoint information	Number=8; Message="Connection String is missing or not correct."
3	Entered parameters	P1=#2004-05-01#; P2=#2004-10-01#
4	Returns methods	Returns=....
8	Checkpoint	ValidateChangePointUser is successful.

Web Services API log file example

```
4:37:10 PM Level1: Source: Changepoint.CPWebService.WSConnection.Login
4:37:10 PM Level1: Source:
Changepoint.CPWebService.WSConnection.getWebPassword
4:37:10 PM Level3: WebLoginId skim
4:37:10 PM Level8: TokenExpiry: 0
4:37:10 PM Level8: UseCache: True
4:37:10 PM Level8: CacheExpiry: 0
4:37:10 PM Level1: Log for this source is end.
4:37:10 PM Level1: Source: Changepoint.CPWebService.WSLogin.Login
4:37:10 PM Level3: userLoginId: skim
4:37:10 PM Level1: Log for this source is end.
4:37:32 PM Level1: Source: Changepoint.CPWebService.WSConnection.Connect
4:37:32 PM Level1: Source: Changepoint.CPWebService.WSConnection.getWebUser
4:37:32 PM Level8: SOAP User: {6EE89511-89D3-4832-AB48-1F7C82C0477E}
4:37:32 PM Level1: Log for this source is end.
```

Web Services API XML

Some Web Services API methods accept XML strings as parameters. The XML strings contain the information to be added, updated or deleted.

For information on XML templates, elements, and samples, see the following:

- "ApiContact XML" on page 50

- "ApiCustomer XML" on page 75
- "ApiEngFixedFeeItem XML" on page 178
- "ApiEngRequestProcessingRule XML" on page 212
- "ApiEngSplitBillingRule XML" on page 239
- "ApiEngWorkCode XML" on page 247
- "ApiEngWorkLocation XML" on page 252
- "ApiExpense XML" on page 266
- "ApiFunctionSkill XML" on page 304
- "ApiProject XML" on page 486
- "ApiResource XML" on page 588
- "ApiRequest XML" on page 509
- "ApiRequestDemand XML" on page 554
- "ApiOpportunity XML" on page 405
- "UDF XML" on page 742

Web Services API error messages

Most Web Services API error messages are the same as the COM API error messages. For more information, see the "Error messages in the COM API" section on page 746.

The exceptions are listed below. The following error messages are specific to the Web Services API.

```
-2000: This is not a SOAP request.
-2001: Login token is not valid, please login.
-2002: Login token expired, please login again.
-2003: Login user password is not provided by AccessToken.
```

Troubleshooting the Web Services API

The COM API troubleshooting information is also applicable to Web Services. For more information, see the "Troubleshooting the COM API" section on page 766.

In addition, the following troubleshooting issues are specific to the Web Services API.

Security token could not be authenticated or authorized

Possible causes: Invalid Changepoint user ID or password.

Solution: Verify the Changepoint resource setup for a valid user ID and password pair.