



Web Services API Guide

AgilePoint BPMS v5.0 R2

Document Revision r5.6.0

June 2014

Contents

Web Services API.....	9
Preface.....	10
Disclaimer of Warranty.....	10
Copyright.....	10
Trademarks.....	10
Government Rights Legend.....	10
Virus-free software policy.....	10
Document Revision Numbers.....	10
AgilePoint Documentation in PDF and HTML.....	11
Opening the Documentation Library.....	11
Finding Information in the Documentation Library.....	12
Downloading Files and Sharing Links from the Documentation Library.....	12
Contacting AgilePoint Sales.....	13
Contacting Customer Support.....	13
Basic Programming Tasks.....	14
Code Examples.....	14
AgilePoint Server Running in IIS Mode.....	14
AgilePoint Server Running in Windows Service Mode.....	15
(Windows Service Installation Only) Run the Windows Service Client Utility Tool.....	17
Authentication.....	17
Namespace Reference.....	18
Catching Exceptions.....	18
Making Calls.....	18
Methods.....	20
Common Methods.....	20
Check Authentication.....	20
Surrogate.....	21
Surrogate With Application Name and Locale.....	23
Set Client Application Name.....	24
Set Client Locale.....	25
Process Definition Methods.....	26
CheckOut Process Definition.....	27
Check In Process Definition.....	28
Create Process Definition.....	30
Delete Process Definition.....	31
Get Base Process Definition ID.....	32
Get Process Definition By Base Process Definition ID.....	33
Get Process Definition Graphics.....	34
Get Process Definition Name and Version.....	35
Get Process Definitions.....	37

Get Process Definition XML.....	38
Get Released Process Definition ID.....	39
Get Released Process Definitions.....	40
Release Process Definition.....	41
Uncheckout Process Definition.....	42
Update Process Definition.....	43
Methods for Process Instances.....	44
Cancel Process Instance.....	44
Create Process Instance.....	45
Create Process Instance (Extended with Initiator and Work Object Info).....	48
Create Process Instance (Extended with Initiator).....	51
Create Process Instance (Extended Method).....	54
Delete Process Instance.....	58
Get Events By Process Instance ID.....	59
Get Process Instance.....	60
Get Process Instance Attribute.....	61
Get Process Instance Attributes.....	63
Merge Process Instances.....	64
Migrate Process Instances.....	66
Promote Process Instance.....	68
Query Process Instances.....	69
Query Process Instances (Extended Method).....	71
Resume Process Instance.....	72
Rollback Process Instance.....	73
Split Process Instance.....	74
Start Process Instance.....	77
Suspend Process Instance.....	78
Update Process Instance.....	79
Methods for Activity Instances.....	80
Cancel Activity Instance.....	81
Get Activity Instance.....	82
Get Activity Instance Status.....	83
Get Activity Instances By Process Instance ID.....	84
Query Activity Instances.....	85
Rollback Activity Instance.....	86
Rollback Activity Instances.....	88
Methods for Manual Work Items (Tasks).....	89
Assign Work Item.....	89
Assign Work Item (Extended Method).....	90
Cancel Work Item.....	92
Cancel Work Item (Extended Method).....	93
Complete Work Item.....	94
Complete Work Item (Extended Method).....	95
Create Linked Work Item.....	97
Create Linked Work Item (Extended Method).....	99
Create Pseudo Work Item.....	102
Create Work Item.....	104
Get Work Item.....	106
Get Work List By User ID.....	107
Query Work List.....	109
Query Work List (Extended Method).....	110
Reassign Update Work Item.....	112
Reassign Work Item.....	114
Reassign Work Item (Extended Method).....	115

Undo Assign Work Item.....	117
Undo Assign Work Item (Extended Method).....	118
Update Work Item.....	119
Methods for Automatic Work Items (Procedures).....	121
Cancel Procedure.....	121
Complete Procedure.....	122
Get Procedure.....	123
Query Procedure List.....	124
User Delegation.....	126
Activate Delegation.....	126
Add Delegation.....	127
Cancel Delegation.....	128
Get Delegation.....	129
Get Delegations.....	130
Remove Delegation.....	132
Update Delegation.....	133
Methods for Notifications.....	134
Cancel Mail Deliverable.....	134
Get Expecting Send Mail Deliverable.....	135
Get Mail Deliverables.....	136
Resend Mail Deliverable.....	137
Methods for Events.....	138
Get Event.....	139
Send Mail.....	140
Send Mail.....	140
Send Mail (Extended Method).....	142
Send Mail (Extended Method with Priority).....	144
Methods for Custom Attributes.....	146
Get Custom Attribute.....	146
Get Custom Attributes.....	147
Get Custom Attributes (Extended Method).....	149
Remove Custom Attribute.....	150
Remove Custom Attributes.....	151
Set Custom Attribute.....	152
Set Custom Attributes.....	154
Methods for Archiving and Restoring Processes.....	155
Archive Process Instance.....	155
Restore Process Instance.....	156
Query Archived Process Instances.....	158
Group, Role, and Rights.....	159
Add Group.....	159
Add Group Member.....	161
Add Role.....	163
Add Role Member.....	165
Enabled Group Member.....	168
Get Access Right Names.....	169
Get Access Rights.....	170
Get Group.....	171
Get Group Members.....	173
Get Groups.....	174
Get Role.....	175
Get Roles.....	176
Query Role Members.....	177
Remove Group.....	178

Remove Group Member.....	179
Remove Role.....	181
Remove Role Member.....	182
Update Group.....	183
Update Role.....	185
Organization Properties.....	187
Get Organization Properties.....	187
Remove Organization Properties.....	188
Update Organization Properties.....	189
Other Web Services.....	191
Query Audit Trail.....	191
Query Database.....	192
Query Database (Extended Method).....	193
Administrative Service.....	194
Get All EMail Templates.....	194
Get Database Information.....	195
Get Domain Groups.....	196
Get Domain Group Members.....	198
Get Domain Name.....	199
Get Domain Users.....	200
Get EMail Template.....	201
Get Locale.....	202
Get Register User.....	203
Get Register Users.....	204
Get Register User Icons.....	205
Get Sender Email Address.....	206
Get SMTP Server.....	207
Get System Performance Information.....	208
Get System User.....	209
Query Register Users.....	210
Register User.....	211
Unregister User.....	212
Update Registered User.....	213
Update Registered User Icon.....	214
Report Configuration Methods.....	215
Add Report Configuration.....	215
Get All Report Configurations.....	217
Get Report Configuration.....	218
Read Configuration.....	219
Remove Report Configure.....	219
Update Report Configuration.....	220
Component Administration Methods.....	222
Get Server Component.....	222
Get Server Component Names.....	223

Classes.....	224
KeyValue.....	224
Description.....	224
Syntax.....	224
Constructors.....	224
Namespace and Assembly.....	224
Properties.....	224
NameValue.....	225

Description.....	225
Syntax.....	225
Constructors.....	225
Namespace and Assembly.....	225
Properties.....	226
IWFWorkflowService.....	226
Description.....	226
Syntax.....	226
Constructors.....	226
Namespace and Assembly.....	227
IWFTrackingEventPublisher.....	227
Description.....	227
Syntax.....	227
Constructors.....	227
Namespace and Assembly.....	227
Properties.....	227
RegisteredUser.....	228
Description.....	228
Syntax.....	228
Constructors.....	228
Namespace and Assembly.....	228
Properties.....	229
WFAccessRights.....	230
Description.....	230
Syntax.....	230
Constructors.....	230
Namespace and Assembly.....	231
WFAgilePart.....	231
Description.....	231
Syntax.....	231
Constructors.....	231
Namespace and Assembly.....	231
WFAgilePartDescriptor.....	231
Description.....	231
Syntax.....	232
Constructors.....	232
Namespace and Assembly.....	232
Properties.....	232
WFAgileWork.....	237
Description.....	237
Syntax.....	237
Constructors.....	238
Namespace and Assembly.....	238
WFAgileWorkDescriptor.....	238
Description.....	238
Syntax.....	238
Constructors.....	238
Namespace and Assembly.....	238
Properties.....	239
WFAny.....	247
Description.....	247
Syntax.....	247
Constructors.....	247
Namespace and Assembly.....	247

Properties.....	248
WFEvent.....	248
Description.....	248
Syntax.....	248
Constructors.....	248
Namespace and Assembly.....	249
Properties.....	249
WFIntegratedApplication.....	254
Description.....	254
Syntax.....	254
Constructors.....	255
Namespace and Assembly.....	255
Properties.....	255
WFIntegratedApplicationDescriptor.....	256
Description.....	256
Syntax.....	256
Constructors.....	257
Namespace and Assembly.....	257
Properties.....	257
WFPartialRollbackInstruction.....	257
Description.....	257
Syntax.....	257
Constructors.....	257
Namespace and Assembly.....	258
Properties.....	258
WFProcessMergingInstruction.....	258
Description.....	258
Syntax.....	258
Constructors.....	258
Namespace and Assembly.....	259
Properties.....	259
WFProcessMigrationInstruction.....	259
Description.....	259
Syntax.....	260
Constructors.....	260
Namespace and Assembly.....	260
Properties.....	260
WFProcessPluggableAdapter.....	261
Description.....	261
Syntax.....	261
Constructors.....	262
Namespace and Assembly.....	262
Properties.....	262
WFProcessPluggableAdapterDescriptor.....	264
Description.....	264
Syntax.....	264
Constructors.....	264
Namespace and Assembly.....	264
Properties.....	264
WFProcessSplittingInstruction.....	266
Description.....	266
Syntax.....	266
Constructors.....	266
Namespace and Assembly.....	266

Properties.....	267
WFQueryExpr.....	267
Description.....	267
Syntax.....	267
Constructors.....	267
Namespace and Assembly.....	267
Properties.....	268
WFTimeDuration.....	269
Description.....	269
Syntax.....	269
Constructors.....	269
Namespace and Assembly.....	269
Properties.....	270
WFTimeUnit.....	271
Description.....	271
Syntax.....	271
Constructors.....	271
Namespace and Assembly.....	271
Properties.....	271

Data Types.....272

Web Services API

This document describes the AgilePoint remote API, which you can access using a web service or a Windows service (WCF).

The AgilePoint remote API is designed and implemented using the Microsoft .Net web service framework, but you can view the WSDL using Internet Explorer by connecting AgilePoint Server at a URL in the format:

- **Workflow API** - [http://\[qualified machine name\]:\[port\]/\[AgilePoint virtual directory\]/workflow.asmx?WSDL](http://[qualified machine name]:[port]/[AgilePoint virtual directory]/workflow.asmx?WSDL)
- **Administration API** - [http://\[qualified machine name\]:\[port\]/\[AgilePoint virtual directory\]/admin.asmx?WSDL](http://[qualified machine name]:[port]/[AgilePoint virtual directory]/admin.asmx?WSDL)

WSDL is a web service standard that allows any client application to consume the AgilePoint API. However with .NET, Visual Studio or a command line can help to generate a web service proxy class that is used to consume the AgilePoint web service using the .NET object model, rather than HTTP and XML. With AgilePoint Developer, AgilePoint provides a prebuilt web service proxy to simplify the process even further. Using the AgilePoint web service proxy also helps with upgrade and deployment because updates will occur using the AgilePoint upgrade kit.

Preface

Disclaimer of Warranty

AgilePoint, Inc. makes no representations or warranties, either express or implied, by or with respect to anything in this document, and shall not be liable for any implied warranties of merchantability or fitness for a particular purpose or for any indirect, special or consequential damages.

Copyright

Copyright © 2013 AgilePoint, Inc. All rights reserved.

Trademarks

AgilePoint, Inc. and AgilePoint's products are trademarks of AgilePoint Inc. References to other companies and their products use trademarks owned by the respective companies and are for reference purpose only.

Government Rights Legend

Use, duplication or disclosure by the U.S. Government is subject to restrictions set forth in the applicable license agreement and as provided in DFARS 227.7202-1(a) and 227.7202-3(a) (1995), DFARS 252.227-7013(c)(1)(ii) (Oct 1988), FAR 12.212(a) (1995), FAR 52.227-19, or FAR 52.227-14, as applicable.

Virus-free software policy

AgilePoint recognizes that viruses are a significant security consideration for our customers. To date, we have had no report of AgilePoint BPMS carries any virus. AgilePoint takes the following measures to ensure our software is free of viruses upon delivery:

- AgilePoint is built on top of Microsoft .NET framework. The pre-compiled executable is a .NET Common Language Runtime (CLR) application, not a native machine binary. As far as is known at this time, there are no viruses that infect .NET CLR executables.
- The virtual environment for the product packaging process is fully isolated and protected, and anti-virus software is installed and running during packaging.
- The deliverable package is scanned by anti-virus software before upload to our customer download site.

Document Revision Numbers

AgilePoint documentation uses the revision number format **rX.Y.Z**. The letters and numbers in this revision number can be interpreted as follows:

- **r** - Indicates "revision." This helps to differentiate the document *version* numbers, which start with **v**.
- **X** - The major version number for AgilePoint BPMS to which this document refers. For example, AgilePoint releases 5.0, 5.0 SP1, and 5.5 would all have an **X** value of **5**.
- **Y** - The major document revision number. This number typically changes only when either there is a new AgilePoint release, or there are major changes to the document.
- **Z** - The minor document revision number. This number is incremented each time the document is republished.

AgilePoint Documentation in PDF and HTML

AgilePoint documentation is provided in both print-friendly (PDF) and web-based (HTML) formats.

Advantages of HTML Documentation

- HTML is the **primary delivery format** for AgilePoint documentation.
- Unified, global **search** across all documentation. PDF documents allow you to search only within the context of a given PDF file.
- **All hyperlinks supported**. Links in PDFs are only supported in certain contexts.
- "One-stop shopping" for all information related to AgilePoint BPMS.
- The HTML documentation is updated more frequently than the PDF documentation. Web-based documentation is updated periodically between AgilePoint releases to address errors and omissions, but the PDF documentation is updated only at the time of a software release.

Advantages of PDF Documentation

PDFs can be more easily **printed**, **archived**, and **transferred** (such as by FTP or email) than HTML documentation.

For more information, see [Downloading Files and Sharing Links from the Documentation Library](#) in the [Documentation Library](#).

Opening the Documentation Library

To open the AgilePoint Documentation Library, do the following.

Prerequisites

You must have a valid account on the AgilePoint Support Portal.

Instructions

1. Log on to the AgilePoint Support Portal.
2. Click **Documentation**.
3. On the **Documentation** page, click the documentation library for your AgilePoint release.
 - For AgilePoint BPMS v5.0 SP1 and higher, the web-based documentation library opens in a new tab or window in your web browser.

- For releases prior to v5.0 SP1, a download starts for a Zip file with the PDF documentation for your release.

Finding Information in the Documentation Library

The information in this topic will help you to locate information in the AgilePoint Documentation Library.

Using the Table of Contents

The table of contents in the AgilePoint Documentation Library is divided by content areas. For example, the Installation section includes all the information you need to install AgilePoint BPMS. The AgilePoint API section includes information about the AgilePoint APIs.

You can use the Table of Contents to explore the AgilePoint documentation content and find the information you want.

Searching

The web-based documentation includes a centralized search for all documentation content. To search for information:

1. In the AgilePoint Documentation Library, click the **Search** tab. In the Search box, enter **1 search term**, and click **Search**.

The search results display in alphabetical order by topic title.

It is important to understand that the third-party software AgilePoint uses to generate web-based documentation allows only 1 search term. More than 1 search term will cause the search to fail.

AgilePoint recommends using a relatively unique search term to find the information you need. For example, entering a common term, such as "process," will return a high percentage of the total documentation topics in the search results.

2. Browse the list of topic titles to find the information you want.

Printing

The PDF documentation is provided mainly for the purpose of printing and archiving. To print a set of information:

1. Navigate to the main page of the Documentation Library from which you want to print.
2. In the list of documents, click the document name in the **PDF** column.
3. From your PDF reader software, print the portion of the document you want.

Downloading Files and Sharing Links from the Documentation Library

You can download and share files AgilePoint's documentation library as you would in any other web page. Note that if you send links to recipients, they must have a Support Portal login to view the file.

These procedures are common examples based on Internet Explorer with the Adobe Reader plug-in. Exact procedures may vary depending on your web browser, PDF viewer, and email client configuration.

Share a Link to an HTML Topic

1. Navigate to the topic you want to share.
2. Copy the URL in the Location box in your web browser.
3. Paste the URL in an email, IM client, etc.

Share a Link to a PDF Document

1. In Internet Explorer, navigate to the Documentation Library home page.
2. In the **PDF** column, right-click the name of the PDF file you want to share.
3. In the quick menu, click **Copy shortcut**.
4. Paste the URL in an email, IM client, etc.

Save a Copy of a PDF Document

1. In Internet Explorer, [open the Documentation Library home page](#).
2. In the **PDF** column, click the name of the PDF file you want to share.
3. In the Adobe Reader plug-in, click **Save** button.

Contacting AgilePoint Sales

AgilePoint is a leading Business Process Management System (BPMS) provider created by a team of driven people who strive to incorporate the principles of relentless innovation for the benefit of our customers. Our mission is to help companies of any size attain and sustain operational success through process excellence.

Headquarters: AgilePoint Corporation 1916C Old Middlefield Way Mountain View, CA 94043, USA

Tel: (650) 968 - 6789

Fax: (650) 968 - 6785

Email: info@agilepoint.com

Web site: www.agilepoint.com

International: For AgilePoint EMEA and AgilePoint Asia Pacific, please call the AgilePoint Corporate Office for contact information.

Contacting Customer Support

To contact AgilePoint Support, please submit a ticket on the AgilePoint Support Portal: <http://support.agilepoint.com/SupportPortal/>

If you do not have a Support Portal account, you can send an email to request one: support@agilepoint.com

Basic Programming Tasks

This section describes the basic programming tasks that are required for the AgilePoint API:

- Authentication
- Adding a namespace and reference
- Catching exceptions
- Making calls

Code Examples

This section provides general code examples for the remote APIs. The code varies slightly depending upon whether AgilePoint Server is installed in IIS or Windows Service mode.

AgilePoint Server Running in IIS Mode

GetAdminService

```
public IWFAdminService GetAdminService()
{
    // initiated an object of proxy class of AgilePoint Web Service
    IWFAdminService svc = new AdminService("http://[machine]/[virtual
        directory]");
    // set URL
    svc.CookieContainer = new System.Net.CookieContainer();
    // set Credentials
    svc.Credentials = new System.Net.NetworkCredential(userName,
password,
        domainName);
    // or take default credential
    // svc.Credentials = System.Net.CredentialCache.DefaultCredentials;

    try
    {
        // set client application name
        svc.SetClientAppName("Samples");
        // set locale
        svc.SetClientLocale("en-US"); // AgilePoint support 13
languages
        // check authenticated and return qualified user name
        string qualifiedDomainUserName = svc.CheckAuthenticated();
    }

    catch(Exception ex)
    {
        string error = ShUtil.GetSoapMessage(ex)
        //log/throw the exception
    }
}
```

```
    Finally
    {
        return svc;
    }
}
```

GetWorkflowService

```
public IWFWorkflowService GetWorkflowService ()
{
    // initiated an object of proxy class of AgilePoint Web Service
    IWFWorkflowService svc = new WorkflowService("http://[machine]/[virtual
directory]");

    // set URL
    svc.CookieContainer = new System.Net.CookieContainer();

    // set Credentials
    svc.Credentials = new System.Net.NetworkCredential(userName,
password,
    domainName);

    // or take default credential
    // svc.Credentials = System.Net.CredentialCache.DefaultCredentials;
    try
    {
        // set client application name
        svc.SetClientAppName("Samples");

        // set locale
        svc.SetClientLocale("en-US"); // AgilePoint supports 13
languages

        // check authenticated and return qualified user name
        string qualifiedDomainUserName = svc.CheckAuthenticated();

    }

    catch(Exception ex)
    {
        string error = ShUtil.GetSoapMessage(ex)
        //log/throw the exception
    }

    Finally
    {
        return svc;
    }
}
```

AgilePoint Server Running in Windows Service Mode

GetAdminService

```
public IWFAAdminService GetAdminService()
{
    string user = this.Context.User.Identity.Name;

    // Set Credentials - Windows Authentication
    System.Net.ICredentials credentials =
        System.Net.CredentialCache.DefaultCredentials;
    // In case of form authentication
    //System.Net.NetworkCredential credentials = new
        System.Net.NetworkCredential(userName, password, domain);

    string locale = "en-us";

    string adminBinding =
        (String)ConfigurationSettings.AppSettings["AdminBindingUsed"];
    IWFAAdminService m_adm = new WCFAdminProxy("MyApplicationName",
        "", locale, user, credentials, adminBinding);

    Return m_adm;
}
```

GetWorkflowService

```
public IWFWWorkflowService GetWorkflowService ()
{
    string user = this.Context.User.Identity.Name;

    // Set Credentials - Windows Authentication
    System.Net.ICredentials credentials =
        System.Net.CredentialCache.DefaultCredentials;
    // In case of form authentication
    //System.Net.NetworkCredential credentials = new
        System.Net.NetworkCredential(userName, password, domain);

    string locale = "en-us";

    string workFlowBinding =
        (String)ConfigurationSettings.AppSettings["WorkFlowBindingUsed"];
    IWFWWorkflowService m_api = new
    WCFWorkflowProxy("MyApplicationName",
        "", locale, user, credentials, workFlowBinding);

    return m_api;
}
```


(Windows Service Installation Only) Run the Windows Service Client Utility Tool

If you are using AgilePoint Server in a Windows Service environment, you must run the Windows Service Client Utility Tool on your application configuration file to configure the binding with the AgilePoint Server.



Note: AgilePoint recommends creating a backup copy of the configuration file for the application you are configuring before using this utility.

Open the Client Utility Tool on Your Machine

On the AgilePoint Server machine, navigate to [\[AgilePoint Server installation folder\]\SVCUtilityTool\AgilePointWindowsServiceClientUtilityTool.exe](#)

This is only installed on the AgilePoint Server machine.

Open the Client Utility Tool from the AgilePoint Installation Program

Open the AgilePoint installation Setup.exe file, and click **AgilePoint Utilities and Other Support Files > AgilePoint Windows Service Client Utility Tool**.

More Information

- [Run the Windows Service Client Utility Tool](#)

Authentication

To communicate with the AgilePoint API, you must establish a session, which associates an authenticated user with a set of calls. The client must provide credentials to the AgilePoint Server using Windows authentication.

There are two main ways to provide credentials to the AgilePoint Server: Specify the user's credentials, or use the default credentials.

Specify the Credentials for a User

Specify the user name, password, and domain name for a user. The domain name could be a Windows Domain Name or a Local Host Name. The following example shows the syntax for establishing credentials in this way:

```
System.Net.ICredentials = new
    System.Net.NetworkCredential(userName, password, domainName);
```

Use the Default Credentials

Use the default credentials for a user. The default is the system credentials for the current security context in which the application is running. For a client-side application, these are usually the Windows credentials (user name, password, and domain) of the user running the application. For ASP.NET applications, the default credentials are the user credentials of the authenticated user, or the user being impersonated.

For more information, see [Surrogate](#) in the [Documentation Library](#).

```
System.Net.ICredentials =
    System.Net.CredentialCache.DefaultCredentials;
```

Namespace Reference

With an out-of-the-box web service proxy included with AgilePoint, a reference to the namespace must be created. There are two methods to create a namespace reference:

1. Use the AgilePoint pre-compiled web service proxy. This method is recommended because it allows for easier upgrades. If you use AgilePoint proxy, you do not need to change your references to it when you upgrade the AgilePoint server because the new functions are included with the AgilePoint proxy.

- a. Add the following commands to your assembly file:

- Ascentn.workflow.shared
- Ascentn.workflow.WFBase
- Ascentn.workflow.WFXML
- Ascentn.AgilePoint.WCFClient

Example:

```
using Ascentn.Workflow.Base;  
using Ascentn.AgilePoint.WCFClient;
```

- b. Add a namespace using Ascentn.WorkflowBase
2. Use Microsoft Visual Studio to generate a web service proxy. The disadvantage to this method is that you must regenerate the proxy each time you upgrade the AgilePoint Server.

Catching Exceptions

The exception that a web service throws contains a lot of information, and most of it is not easy to read. AgilePoint Server tags the readable message for end-users.

Call the following function to extract the error message:

```
String error = ShUtil.GetSoapMessage(ex);
```

ex is the exception object that contains error message from AgilePoint Server

Making Calls

Calls within the AgilePoint API fall into two categories – synchronous or asynchronous.

Synchronous Call

Synchronous calls are used for short transactions. In a synchronous call, a request is sent to the AgilePoint Server, the server acknowledges the request, and then acts upon it immediately.

Asynchronous Call

Synchronous calls are used for short transactions. In a synchronous call, a request is sent to the AgilePoint Server, the server acknowledges the request, and then acts upon it immediately.

Asynchronous calls are used for longer transactions. In an asynchronous call, a request is sent to the AgilePoint Server, the server acknowledges the request, but it does not immediately act upon it immediately. The server creates a WFEvent object, which contains the call's status, and returns the WFEvent object to the client. The client can call GetEvent(EventID) to retrieve the WFEvent object with the updated status. The status can be:

- **Processed** – The transaction was completed successfully.
- **Failed** – The transaction failed, and was not completed.
- **Sent** – The call was received, but it has not been acted upon.

Completing an asynchronous call could take any amount of time, from one second to several days. As a best practice a user interface should handle GetEvent() calls to update end users or the application itself regarding the status of asynchronous calls. You might, for example, use Ajax to check status to display on an ASP.NET page in real time.

Methods

This section includes references for all methods within the AgilePoint Web Service API.

Common Methods

This section describes some commonly used methods for the Web Services API.

Check Authentication

API Type

Web Services

Description

This call is used to verify whether the specified user is a registered user on the AgilePoint Server.

Syntax

```
public virtual string CheckAuthenticated()
```

Parameters

Name	Description
None	Not Applicable

Output

If the user is a registered user, the qualified user name in the format of DomainName\UserName is returned. Otherwise, null is returned.

Example

```
//This example is for an ASP.net application.//
public static string Connect(
System.Web.SessionState.HttpSessionState session,
System.Net.ICredentials credential, string appName, string locale)
{
    string url =
    System.Configuration.ConfigurationManager.AppSettings.Get("ServerUrl");

    WorkflowService svc = new WorkflowService(url);
    AdminService adm = new AdminService(url);
    System.Net.CookieContainer cookieContainer = new
    System.Net.CookieContainer();
    svc.Credentials = credential;
```

```
svc.CookieContainer = cookieContainer;
svc.SetClientAppName(appName);
svc.SetClientLocale(locale);
string userName = svc.CheckAuthenticated();
adm.Credentials = credential;
adm.CookieContainer = cookieContainer;
adm.SetClientAppName(appName);
adm.SetClientLocale(locale);

//Assume the ASP.net is on session
WFCommonPage.SetAdm(session, adm);
WFCommonPage.SetAPI(session, api);

//return fully qualified Domain username
return userName;
}
```

Supported Versions

3.2.0.4 and higher

Surrogate

API Type

Web Services

Description

IIS does not support users who do not use Windows Active Directory authentication. To address this issue, AgilePoint uses a special type of user called an impersonator. Impersonators enable client applications to authenticate end users who use Active Directory authentication, as well as those who do not.

Impersonators must meet the following requirements:

1. The impersonator must be a Windows Active Directory user to pass IIS authentication.
2. The impersonator must be registered on the AgilePoint Server. The registration can be done through Enterprise Manager.
3. The impersonator must be registered for the application under the Extension of AgilePoint Server Configuration. The application name is case sensitive.
4. The impersonator does not need to be the administrator for AgilePoint Server, or even have workflow execution rights on AgilePoint.

This Surrogate function allows the impersonator to act as a surrogate for the specified user to complete IIS authentication. Once the authentication has passed, the web service API will be called based on the rights granted to the specified user on AgilePoint Server.

This function is called before calling any other AgilePoint Web Service API.

Syntax

```
public virtual void Surrogate(string userName)
```

Parameters

Name	Description
userName	<u>Definition:</u> The user name for the user. <u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.

Output

None.

Example

```
public IWFWorkflowService GetWorkflowService(
System.Net.ICredentials credentials, string
username)
{
    IWFWorkflowService svc = GetAdm(credentials);
    svc.Surrogate(username);
    return svc;
}

//Web service using Impersonator credentials that is registered in the AgilePoint
Server
//configuration with application name
public IWFWorkflowService GetWorkflowService(System.Net.ICredentials credentials)
{
    string url = ... // AgilePoint Server web service Url
    System.Net.CookieContainer cookieContainer = new
    System.Net.CookieContainer();
    WFWorkflowService svc = new WFWorkflowService(url);
    svc.CookieContainer = cookieContainer;
    svc.Credentials = credentials;
    svc.SetClientAppName([your application name]);
    //Current Locale
    svc.SetClientLocale(Thread.CurrentThread.CurrentUICulture.Name);
    return svc;
}
```

Supported Versions

3.2.0.4 and higher

Surrogate With Application Name and Locale

API Type

Web Services

Description

This function is similar to Surrogate, with the ability to set the application name and locale at the same time. Calling the functions SetClientAppName and SetClientLocale is not needed if this function is called.

Syntax

```
public virtual void Surrogate(string userName, string AppName, string Locale)
```

Parameters

Name	Description
userName	<u>Definition:</u> The user name for the user. <u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.
AppName	<u>Definition:</u> Specifies the name of the application. <u>Type</u> string <u>Allowed Values:</u> A valid, case-sensitive application name.
Locale	<u>Definition:</u> Specifies the client locale. <u>Type</u> string <u>Allowed Values:</u> A valid locale in standardized format. <u>Example:</u>

Name	Description
	en-US

Output

None.

Example

```
public IWFlowService GetWorkflowService(System.Net.ICredentials
credentials, string userName, string AppName, string Locale)
{
    IWFlowService svc = GetWorkflowService(credentials);
    svc.Surrogate(userName, surrogate AppName, surrogate Locale);
    return svc;
}

public IWAdminService GetWorkflowService(System.Net.ICredentials credentials)
{
    //The body is the same as Surrogate
}
```

Supported Versions

3.2.0.4 and higher

Set Client Application Name

API Type

Web Services

Description

This call is used to set the current application name. The application can be a web/Windows/Windows service application that calls the AgilePoint server for business process actions.

Syntax

```
public virtual void SetClientAppName(string AppName)
```

Parameters

Name	Description
AppName	<u>Definition:</u> Specifies the name of the application. <u>Type</u> string

Name	Description
	<u>Allowed Values:</u> A valid, case-sensitive application name.

Output

None.

Example

[See previous example](#)

Supported Versions

3.2.0.4 and higher

Set Client Locale

API Type

Web Services

Description

This call is used to set the locale for the client application that calls AgilePoint Server. A client application can be a web/Windows/Windows service application.

Syntax

```
public virtual void SetClientLocale(String Locale)
```

Parameters

Name	Type	Description
Locale	<u>Definition:</u> Specifies the client locale. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid locale in standardized format. <u>Example:</u> en-US	

Output

None.

Example

None.

Supported Versions

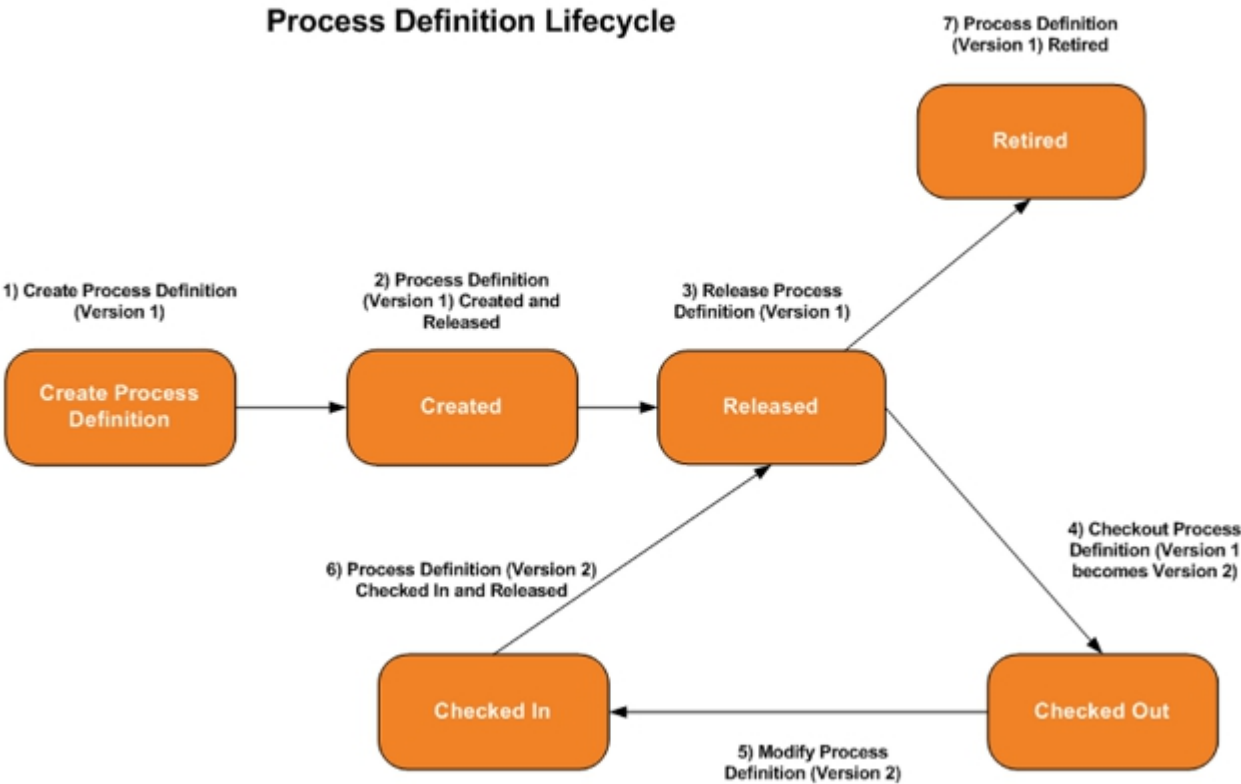
3.2.0.4 and higher

Process Definition Methods

This section describes how process definitions are managed. The AgilePoint server supports version control to ensure that process definitions are archived and maintained for later use. A process definition can spawn many process instances. Employing version controls allows the AgilePoint Server to store multiple versions of a single process definition.

The project lifecycle of a process definition can be defined by the following steps:

1. **Create Process Definition** – Create a process definition. The version for the initial process definition is version 1.
2. **Created** – Confirm that the process definition has been created.
3. **Release process definition** – Once a process definition has been created, it must be "released." This allows the process definition to be checked out and edited/modified.
4. **Check out process definition** – If changes or modifications are made, the process definition is "checked out." Changes can then be made to the process definition.
5. **Modify process definition** – After a process definition has been checked out, changes can be made to the process definition.
6. **Check in process definition** – Once all changes/modifications have been made, the process definition is checked in, so the changes are committed to the process definition. The process definition is now version 2.
7. **Retire process definition** – Once version 2 of the process definition has been checked in, the previous version 1 is retired. Retired process definitions can also be deleted.



CheckOut Process Definition

API Type

Web Services

Description

This method is used to manage process definition versioning by setting the process definition status to **CheckedOut** based on a given process definition ID. Only process definitions with the status of **Released** can transition into the **CheckedOut** status.

Syntax

```
public virtual string CheckoutProcDef(string processTemplateID)
```

Parameters

Name	Description
processTemplateID	<u>Definition:</u> The unique identifier for the process definition to be checked out for modification. <u>Type</u>

Name	Description
	string <u>Allowed Values:</u> A valid process template ID

Output

The process definition, in XML format, that has been checked out.

Example

```
IWFWorkflowService svc = GetWorkflowService();

try
{
    string processTemplateID = ... // process definition to be checked out
    string processDefinitionXML =
        svc.CheckoutProcDef(processTemplateID);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage( ex ) );
}
```

Supported Versions

3.2.0.4 or higher

Check In Process Definition

API Type

Web Services

Description

Checks in the process definition to the AgilePoint Server and returns the process definition identifier. This method accepts a string with the updated process definition in XML format.

Syntax

```
public virtual string CheckinProcDef(string xml)
```

Parameters

Name	Description
xml	<u>Definition:</u> Specifies a process definition in XML format.

Name	Description
	To generate the process definition file in XML format, in AgilePoint Envision, click File > Export & Import > Save As Deploying File(xml). You can also download the process definition XML from AgilePoint Enterprise Manager. <u>Type</u> string <u>Allowed Values:</u> A string that contains process definition in XML format.

Output

A new process definition ID.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string xml = ...// Process definition XML as string
WFProcessDefinition pd = new WFProcessDefinition();
GraphicImage g = new GraphicImage();
ProcDefXmlParser parser = new ProcDefXmlParser(new
WFDefaultActivityInstantiator(), pd, g);
parser.Parse(xml);

if (release process definition immediately)
{
    pd.ReleaseDate = DateTime.Now;
    pd.Version = .. // new version
    string procDefID = svc.CheckinProcDef( xml );
    svc.ReleaseProcDef(processDefinitionID);
}

else if(release process definition at specific date in the future)
{
    pd.ReleaseDate = ...// a specific date in the future
    pd.Version = .. // new version
    string processDefinitionID = svc.CheckinProcDef( xml );
    svc.ReleaseProcDef(processDefinitionID);
}

else // not release process definition
{
    pd.ReleaseDate = Constants.NullDate;
    string processDefinitionID = svc.CheckinProcDef( xml );
}
```

Supported Versions

3.2.0.4 or higher

Create Process Definition

API Type

Web Services

Description

Adds a new process definition to the AgilePoint Server.

Syntax

```
public virtual string CreateProcDef(string xml)
```

Parameters

Name	Description
xml	<u>Definition:</u> Specifies a process definition in XML format. To generate the process definition file in XML format, in AgilePoint Envision, click File > Export & Import > Save As Deploying File(xml). You can also download the process definition XML from AgilePoint Enterprise Manager. <u>Type</u> string <u>Allowed Values:</u> A string that contains process definition in XML format.

Output

Unique ID of the process definition, which the AgilePoint system generates.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string xml = ...// Process definition XML as string

if( set release date )
{
    WFProcessDefinition pd = new WFProcessDefinition();
    GraphicImage g = new GraphicImage();
    ProcDefXmlParser parser = new ProcDefXmlParser(new
    WFDefaultActivityInstantiator(), pd, g);
    parser.Parse(xml);
}
```

```
pd.ReleaseDate = DateTime.Now; //Set release date
ProcDefXmlWriter w = new ProcDefXmlWriter(pd, g);
xml = w.WriteToString();
string procDefID = svc.CreateProcDef(xml);
}

else // not to release process template
{
    string procDefID = svc.CreateProcDef(xml);
}
```

Supported Versions

3.2.0.4 and higher

Delete Process Definition

API Type

Web Services

Description

Deletes the process definition and all of the process instances associated with the process definition. The process definition cannot be deleted if one or more process instances associated with the process definition is running or suspended. The function may take a long time to execute if there are many process instances associated with the process definition.

Syntax

```
public virtual void DeleteProcDef(string processTemplateID)
```

Parameters

Name	Description
processTemplateID	<u>Definition:</u> The unique identifier for the process definition to be checked out for modification. <u>Type</u> string <u>Allowed Values:</u> A valid process template ID

Output

None.

Example

```
IWFWorkflowService svc = GetWorkflowService();  
  
string processTemplateID = ..// The unique identifier of the process definition to  
be deleted  
svc.DeleteProcDef(processTemplateID);
```

Supported Versions

3.2.0.4 and higher

Get Base Process Definition ID

API Type

Web Services

Description

Retrieves the ID for the first version of the process definition, called the base process definition. All subsequent process definition versions have the same base process definition ID. This call retrieves the base process definition ID with the specified process definition name.

Syntax

```
public virtual string GetBaseProcDefID(string procDefName)
```

Parameters

Name	Description
procDefName	<u>Definition:</u> The name of the process definition. <u>Type</u> string <u>Allowed Values:</u> A single line of text.

Output

string that contains the base process definition ID.

Example

```
IWFWorkflowService svc = GetWorkflowService();  
string procDefName = "CreateRequest";  
string baseProcessDefinitionID=
```



```
svc.GetBaseProcDefID(procDefName);
```

Supported Versions

3.2.0.4 and higher

Get Process Definition By Base Process Definition ID

API Type

Web Services

Description

Retrieves all process definitions by a specified base process definition ID.

Syntax

```
public virtual WFBaseProcessDefinition[] GetProcDefByBasePID(string baseprocessTemplateID)
```

Parameters

Name	Description
baseprocessTemplateID	<u>Definition:</u> The ID of the base process definition. <u>Type</u> string <u>Allowed Values:</u> A valid base process definition ID.

Output

Array of WFBaseProcessDefinition objects.

Example

```
// This is console application
IWfWorkflowService svc = GetWorkflowService();

//Base process definition ID.
string baseprocessTemplateID = ... // for example "1e3d514d43d3465cae6ec3bbbd409168";

try
{
    //Returns Array of WFBaseProcessDefinition for all versions of
    process definition WFBaseProcessDefinition[] processDefinitions =
    svc.GetProcDefByBasePID(baseprocessTemplateID);
    for (int i = 0; i < processDefinitions.Length; i++)
```

```
{
    Console.WriteLine("Definition ID: '" + processDefinitions[i].DefID + "' ");
    Console.WriteLine("Definition Name: '" +
        processDefinitions[i].DefName + "' ");
}
}

catch (Exception ex)
{
    Console.WriteLine(ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Process Definition Graphics

API Type

Web Services

Description

Retrieves graphical data for the process definition in XML format. The graphical representation of the process is XML-serialized by the class Graphic Image. The graphical data is used to display the process visually.

Syntax

```
public virtual string GetProcDefGraphics(string processID)
```

Parameters

Name	Description
processID	<u>Definition:</u> The process definition ID for a released process definition. <u>Type</u> string <u>Allowed Values:</u> A valid process definition ID.

Output

Graphics object in XML format.

Example

```
// This is console application sample
IWfWorkflowService svc = GetWorkflowService();
//process definition ID or process instance ID
string processID = ... // for example,
"42544811EC2D4FC18E6BA15CC9FE28DF";

try
{
    //returns an image of a process definition as string.
    string procDefGraphicsXML = svc.GetProcDefGraphics(processID);
    GraphicImage g = new GraphicImage();
    g.FromXml(procDefGraphicsXML);
    byte[] images = g.Image // process image
    NamedRectangle[] shapes = g.Shapes;
}

catch (Exception ex)
{
    Console.WriteLine(ShUtil.GetSoapMessage(ex));
}

/* This example produces the following results:
process definition Graphics:
<?xml version="1.0" encoding="utf-8"?><Graphics
left="3.33333333333333" right="5.76002857553708"
top="10.3848753378378" bottom="7"><Shapes><Shape
left="3.33333333333333" right="3.83333333333333"
top="10.38541666666667" bottom="9.88541666666667" name="Start"
/><Shape left="5.26041666666667"
right="5.76041666666667" top="7.5" bottom="7" name="Stop"
/><Shape left="4.30208333333333"
right="5.30208333333333" top="9.16666666666667"
bottom="8.66666666666667" name="Text File Writer.3"
/></Shapes><Image>@64R0lGODlh6QBFAXAAACwAAAAA6QB
FAYcAAAAAMDawkJCQsLCydAAClAACqAACuAACyAAC2AAC6AAC+\nAADCAADGAADK
AAA8PDzOAADSAA
.....\nTEGgS0+D2pKSHjUEWOzmRMPZljdStSBbQshYsxjRfu7xD
REAAv3vgai7gAEaA3sj3bghhl2dbEN\NDAM5bsCFff6yIluYAAZ8WtowAasI
wudCT76YxwltYQa9/Wlww8ADCMxp+/SHxwvUm4UYBGq02d2S\nWJwAf+brX/9H0ic
BCEbggcpcuAAbuvO+YgBV8acwAwNO3PjK+j4EKZzhNwp1PZ4dvfBUAoMEjgHG
b\nBGNA0rYGalW+cpazvMA32cOlMo6cgAAAOw==</Image></Graphics>
*/
```

Supported Versions

3.2.0.4 and higher

Get Process Definition Name and Version

API Type

Web Services

Description

Retrieves the process definition name and version.

Syntax

```
public virtual KeyValue GetProcDefNameVersion(string processTemplateID)
```

Parameters

Name	Description
processTemplateID	<u>Definition:</u> The process definition ID for a released process definition. <u>Type</u> string <u>Allowed Values:</u> A valid process definition ID.

Output

[KeyValue](#) object, where Key contains process definition name and Value contains version.

Example

```
// This is console application sample
IWFFWorkflowService svc = GetWorkflowService();

//process definition ID for a process.
string processTemplateID = ... // for example "1e3d514d43d3465cae6ec3bbbd409168";

try
{
    //Returns KeyValuepair, for example "process definition
    Name-process definition Version"
    KeyValue keyValue = _ svc.GetProcDefNameVersion(processTemplateID);
    Console.WriteLine("process definition Name: '" +
        keyValue.Key.ToString() + "' ");
    Console.WriteLine("process definition Version: '" +
        keyValue.Value.ToString() + "' ");
}

catch (Exception ex)
{
    Console.WriteLine( ShUtil.GetSoapMessage(ex) );
}
```

Supported Versions

3.2.0.4 and higher

Get Process Definitions

API Type

Web Services

Description

Retrieves all of process definition objects.

Syntax

```
public virtual WFBaseProcessDefinition[] GetProcDefs()
```

Parameters

Name	Description
None	Not Applicable

Output

An array of WFBaseProcessDefinition objects.

Example

```
IWFWorkflowService svc = GetWorkflowService();

try
{
    //Returns Array of WFBaseProcessDefinition type.
    WFBaseProcessDefinition[] processDefinitions = svc.GetProcDefs();
    for (int i = 0; i < processDefinitions.Length; i++)
    {
        Console.WriteLine("Definition ID: '" +
            processDefinitions[i].DefID + "' ");
        Console.WriteLine("Definition Name: '" +
            processDefinitions[i].DefName + "' ");
    }
}

catch (Exception ex)
{
    Console.WriteLine(ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Process Definition XML

API Type

Web Services

Description

Retrieves a process definition in XML format.

Syntax

```
public virtual string GetProcDefXml(string processTemplateID)
```

Parameters

Name	Description
processTemplateID	<u>Definition:</u> The process definition ID for a released process definition. <u>Type</u> string <u>Allowed Values:</u> A valid process definition ID.

Output

string that contains XML format of the process definition.

Example

```
// This is console application sample
IWFWorkflowService svc = GetWorkflowService();
//process definition ID for a process.
string processTemplateID = ..// for example "42544811EC2D4FC18E6BA15CC9FE28DF";

try
{
    //Returns process definition in XML format.
    string procDefXML = svc.GetProcDefXml(processTemplateID);
    Console.WriteLine("process definition XML: \n");
    Console.WriteLine(procDefXML);
}

catch (Exception ex)
{
    Console.WriteLine( ShUtil.GetSoapMessage(ex));
}
```

```

}

/*
This example produces the following results:
process definition XML:
<?xml version="1.0" encoding="utf-8"
standalone="no"?>\n<?wfmc-xpdl xmlns="http://www.wfmc.org/2002/XPDL1.0"
xmlns:xpdl="http://www.wfmc.org/2002/XPDL1.0"
xsi:schemaLocation="http://www.wfmc.org/2002/XPDL1.0"?>\n<!--Process
Definition, Copyright 2003-2004 AgilePoint Inc., All
Rights Reserved.-->\n<ProcessDefinition
defName="TextFileWriterProcess" preVersion="" version="1.0"
description="" owner="Bipin.Shah" docRef=""
.....\nTEGgS0+D2pKSHjUEWOzmRMPZljdStSBbQshYsxjRfu7
xDREAaV3vGgai7gAEaA3sj3bghhl2dbEN\nDAM5bsCFff6
yIluYAAZ8WtowAasIwudCT76YxwltYQa9/Wlww8ADCmXp+/SHxwvUm4UYBGq02d2S\nWJwAf+brX/9H
0icBCEbggcPuAAbuV0+YgBV8acwAwNO3PjK+j4EKZzhNwp
1PZ4dvfBUAoMEjgHGb\nBGNA0rYGa1W+cpazvMA32cOlMo6cgAAA0w==</Image>\n
</Graphics>\n</ProcessDefinition>
*/

```

Supported Versions

3.2.0.4 and higher

Get Released Process Definition ID

API Type

Web Services

Description

Retrieves the released process definition ID by a specified process definition name.

Syntax

```
public virtual string GetReleasedPID(string procDefName)
```

Parameters

Name	Description
procDefName	<u>Definition:</u> The name of the process definition. <u>Type</u> string <u>Allowed Values:</u> A single line of text.

Output

The ID for the released process definition.

Example

```
//GetReleasedPID
IWfWorkflowService svc= GetWorkflowService();
string procDefName = "Budget Request Approval Process";
string processDefinitionID =
svc.GetReleasedPID(procDefName);
Console.WriteLine("Process definition ID=" + processDefinitionID);
```

Supported Versions

3.2.0.4 and higher

Get Released Process Definitions

API Type

Web Services

Description

Retrieves the names and IDs of all released process definitions.

Syntax

```
public virtual KeyValue[] GetReleasedProcDefs()
```

Parameters

Name	Description
None	Not Applicable

Output

[KeyValue](#) array for pairs of process definition IDs and process definition names.

Example

```
KeyValue[] defs = svc.GetReleasedProcDefs();
for (int i = 0; i < defs.Length; i++)
{
    Console.WriteLine("Key=" + defs[i].Key.ToString() + "*****"+"Value=" +
    defs[i].Value.ToString());
}
```


Supported Versions

3.2.0.4 and higher

Release Process Definition

API Type

Web Services

Description

Releases a process definition from the AgilePoint Server.

Syntax

```
public virtual void ReleaseProcDef(string processTemplateID)
```

Parameters

Name	Description
processTemplateID	<u>Definition:</u> The unique identifier for the process definition to be checked out for modification. <u>Type</u> string <u>Allowed Values:</u> A valid process template ID

Output

None.

Example

```
//Sample for using Workflow.ReleaseProcDef using Console Application

try
{
    IWFWorkflowService svc = GetWorkflowService();
    string processTemplateID = ...
    svc.ReleaseProcDef(processTemplateID);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage( ex ) );
}
```

```
}
```

Supported Versions

3.2.0.4 and higher

Uncheckout Process Definition

API Type

Web Services

Description

Undoes a check-out for a process definition. This method returns the status of a process definition from CheckedOut to Released without making changes to the process definition, or changing the version number.

Syntax

```
public virtual void UnCheckOutProcDef(string processTemplateID)
```

Parameters

Name	Description
processTemplateID	<u>Definition:</u> The unique identifier for the process definition to be checked out for modification. <u>Type</u> string <u>Allowed Values:</u> A valid process template ID

Output

None.

Example

```
try
{
    IWFWorkflowService svc = GetWorkflowService();
    string processTemplateID = ...
    string processDefinitionXML =
    svc.UncheckoutProcDef(processTemplateID);
}
catch (Exception ex)
```

```
{  
    Console.WriteLine("Message:\n" + ShUtil.GetSoapMessage( ex ) );  
}
```

Supported Versions

3.2.0.4 and higher

Update Process Definition

API Type

Web Services

Description

Updates a process definition without using version control. This method is intended for minor changes only, such as typographical errors. Warning: Changes made using this method circumvent version control, meaning changes are not tracked, and versions cannot be managed. Do not use this call for making any major changes to the process definition.

Syntax

```
public virtual string UpdateProcDef(string xml)
```

Parameters

Name	Type	Description
xml	<u>Definition:</u> Specifies a process definition in XML format. To generate the process definition file in XML format, in AgilePoint Envision, click File > Export & Import > Save As Deploying File(xml). You can also download the process definition XML from AgilePoint Enterprise Manager. <u>Type</u> string <u>Allowed Values:</u> A string that contains process definition in XML format.	

Output

Returns the unique identifier for the process definition that is updated.

Example

```
// This is console application sample
IWfWorkflowService svc = GetWorkflowService();
string processDefinitionXML = ..// see previous description of how to
get process definition XML

try
{
    //Update Process definition using updated process xml string
    string processDefinitionID = svc.UpdateProcDef(xml);
}

catch (Exception ex)
{
    Console.WriteLine("Exception:" + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

4.0.1 and higher

Methods for Process Instances

This section describes service calls related to process instances.

Cancel Process Instance

API Type

Web Services

Description

Cancels the process instance based on a specified process instance identifier. This method cancels all automatic work items, manual work items, and child process instances.

Syntax

```
public virtual WfEvent CancelProcInst(string processInstanceID)
```

Parameters

Name	Description
processInstanceID	<u>Definition:</u>

Name	Description
	Specifies the unique ID of a process instance. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid process instance ID

Output

WFEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string processInstanceID = ..// the ID of the process instance to be
canceled.

try
{
    WFEvent evt = svc.CancelProcInst(processInstanceID);
}

catch( Exception ex)
{
    base.ShowMessage( base.GetSoapMessage(ex) );
}
```

Supported Versions

3.2.0.4 and higher

Create Process Instance

API Type

Web Services

Description

Creates a process instance for a specified process definition ID and parameters.

Syntax

```
public virtual WFEvent CreateProcInst(string ProcessID, string ProcessInstID, string ProcInstName, string WorkObjID, string SuperProcInstID, bool blnStartImmediately)
```

Parameters

Name	Description
ProcessID	<u>Definition:</u> The process definition ID for a released process definition. <u>Type</u> string <u>Allowed Values:</u> A valid process definition ID.
ProcessInstID	<u>Definition:</u> A process instance ID for the process instance you are creating. <u>Type</u> string <u>Allowed Values:</u> A unique, 32 character process instance ID. If you set this value to null, the AgilePoint Server generates the ID.
ProcInstName	<u>Definition:</u> A unique process name that is associated with the process definition. <u>Type</u> string <u>Allowed Values:</u> A unique process instance name up to 1024 characters.
WorkObjID	<u>Definition:</u> An ID for an object, such as a document, that is associated with the process instance. <u>Type</u>

Name	Description
	string <u>Allowed Values:</u> A valid, unique 256-character ID. Even though the field size is 256 characters, in common practice, this will usually return a 32-character GUID.
SuperProcInstID	<u>Definition:</u> A process instance ID that acts as a parent process instance of the process instance that is intended to create. In other words, this is the ID of the process instance on which you want to base your new process instance. <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-character process instance ID.
bInStartImmediately	<u>Definition:</u> An obsolete, legacy parameter that must be true. <u>Type</u> bool <u>Allowed Values:</u> True

Output

[WFEvent](#) object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string processDefinitionName = "EmployeeOnboardProcess";

// get UUID of released process definition
string ProcessID = svc.GetReleasedPID(processDefinitionName);
```

```
// assign UUID of process instance
string ProcessInstID = UUID.GetID();

// process instance name that has to be unique within process definition ID
string ProcInstName = string.Format("{0}-{1}",
processDefinitionName DateTime.Now.Ticks );

// work object ID
string WorkObjID = UUID.GetID();

// create process instance
WFEvent event = svc.CreateProcInst(ProcessID, ProcessInstID, ProcInstName,
WorkObjID, null, true);
```

Supported Versions

3.2.0.4 and higher

Create Process Instance (Extended with Initiator and Work Object Info)

API Type

Web Services

Description

Creates a process instance with a specified workObjInfo value. The workObjInfo parameter provides additional information about a work object, such as a URL for a document.

Syntax

```
public virtual WFEvent CreateProcInstEx(string ProcessID, string ProcessInstID, string ProcInstName, string
WorkObjID, string WorkObjInfo, string SuperProcInstID, string Initiator, string CustomID, NameValue[] Attributes,
bool blnStartImmediately)
```

Parameters

Name	Description
ProcessID	<u>Definition:</u> The process definition ID or process template name for a released process definition. <u>Type</u> string <u>Allowed Values:</u> A valid process definition ID or process template name.

Name	Description
ProcessInstID	<u>Definition:</u> A process instance ID for the process instance you are creating. <u>Type</u> string <u>Allowed Values:</u> A unique, 32 character process instance ID. If you set this value to null, the AgilePoint Server generates the ID.
ProcInstName	<u>Definition:</u> A unique process name that is associated with the process definition. <u>Type</u> string <u>Allowed Values:</u> A unique process instance name up to 1024 characters.
WorkObjID	<u>Definition:</u> An ID for an object, such as a document, that is associated with the process instance. <u>Type</u> string <u>Allowed Values:</u> A valid, unique 256-character ID. Even though the field size is 256 characters, in common practice, this will usually return a 32-character GUID.
WorkObjInfo	<u>Definition:</u> Usually this parameter is used to hold supplemental information about the work object, such as a URL for a document, within the process instance. <u>Type</u> string <u>Allowed Values:</u>

Name	Description
	A string up to 1024 characters.
SuperProcInstID	<u>Definition:</u> A process instance ID that acts as a parent process instance of the process instance that is intended to create. In other words, this is the ID of the process instance on which you want to base your new process instance. <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-character process instance ID.
Initiator	<u>Definition:</u> Specifies the user who initiates a process. <u>Type</u> string <u>Allowed Values:</u> A valid user name.
CustomID	<u>Definition:</u> A work object ID specified within a process instance. <u>Type</u> string <u>Allowed Values:</u> One valid work object ID.
Attributes	<u>Definition:</u> Name-value pairs associated with a custom ID. <u>Type</u> NameValue <u>Allowed Values:</u> A valid custom ID with an associated name.
blnStartImmediately	<u>Definition:</u> An obsolete, legacy parameter that must be true.

Name	Description
	<u>Type</u> bool <u>Allowed Values:</u> True

Output

WFEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
... see previous sample
string workObjectInfo = ..// for example, a XML-serialized of an object
WFEvent evt = svc.CreateProcInstEx(
    ProcessID,
    ProcessInstID,
    ProcInstName,
    WorkObjID,
    WorkObjInfo,
    parentProcessInstID,
    Initiator,
    WorkObjID,
    ds.ToArray(),
    true);
```

Supported Versions

4.0.1 and higher

Create Process Instance (Extended with Initiator)

API Type

Web Services

Description

Creates a process instance in which the user name for the user who initiates the process is specified.

Syntax

```
public virtual WFEvent CreateProcInstEx(string ProcessID, string ProcessInstID, string ProcInstName,
string WorkObjID, string SuperProcInstID, string Initiator, string CustomID, NameValue[] Attributes,
bool blnstartImmediately)
```

Parameters

Name	Description
ProcessID	<u>Definition:</u> The process definition ID or process template name for a released process definition. <u>Type</u> string <u>Allowed Values:</u> A valid process definition ID or process template name.
ProcessInstID	<u>Definition:</u> A process instance ID for the process instance you are creating. <u>Type</u> string <u>Allowed Values:</u> A unique, 32 character process instance ID. If you set this value to null, the AgilePoint Server generates the ID.
ProcInstName	<u>Definition:</u> A unique process name that is associated with the process definition. <u>Type</u> string <u>Allowed Values:</u> A unique process instance name up to 1024 characters.
WorkObjID	<u>Definition:</u> An ID for an object, such as a document, that is associated with the process instance.

Name	Description
	<u>Type</u> string <u>Allowed Values:</u> A valid, unique 256-character ID. Even though the field size is 256 characters, in common practice, this will usually return a 32-character GUID.
SuperProcInstID	<u>Definition:</u> A process instance ID that acts as a parent process instance of the process instance that is intended to create. In other words, this is the ID of the process instance on which you want to base your new process instance. <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-character process instance ID.
Initiator	<u>Definition:</u> Specifies the user who initiates a process. <u>Type</u> string <u>Allowed Values:</u> A valid user name.
CustomID	<u>Definition:</u> A work object ID specified within a process instance. <u>Type</u> string <u>Allowed Values:</u> One valid work object ID.
Attributes	<u>Definition:</u> Name-value pairs associated with a custom ID. <u>Type</u> NameValue

Name	Description
	<u>Allowed Values:</u> A valid custom ID with an associated name.
blnStartImmediately	<u>Definition:</u> An obsolete, legacy parameter that must be true. <u>Type</u> bool <u>Allowed Values:</u> True

Output

WFEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
... see previous sample
string Initiator = ..// System.Environment.UserName
WFEvent evt = svc.CreateProcInstEx(
    ProcessID,
    ProcessInstID,
    ProcInstName,
    WorkObjID,
    parentProcessInstID,
    Initiator,
    WorkObjID,
    ds.ToArray(),
    true);
```

Supported Versions

3.2.0.4 and higher

Create Process Instance (Extended Method)

API Type

Web Services

Description

Creates a process instance that can have additional input arguments added to the function.

Syntax

```
public virtual WFEvent CreateProcInstEx(string ProcessID, string ProcessInstID, string ProcInstName, string WorkObjID, string SuperProcInstID, string CustomID, NameValue[] Attributes, bool blnstartImmediately)
```

Parameters

Name	Description
ProcessID	<u>Definition:</u> The process definition ID or process template name for a released process definition. <u>Type</u> string <u>Allowed Values:</u> A valid process definition ID or process template name.
ProcessInstID	<u>Definition:</u> A process instance ID for the process instance you are creating. <u>Type</u> string <u>Allowed Values:</u> A unique, 32 character process instance ID. If you set this value to null, the AgilePoint Server generates the ID.
ProcInstName	<u>Definition:</u> A unique process name that is associated with the process definition. <u>Type</u> string <u>Allowed Values:</u> A unique process instance name up to 1024 characters.

Name	Description
WorkObjID	<u>Definition:</u> An ID for an object, such as a document, that is associated with the process instance. <u>Type</u> string <u>Allowed Values:</u> A valid, unique 256-character ID. Even though the field size is 256 characters, in common practice, this will usually return a 32-character GUID.
SuperProcInstID	<u>Definition:</u> A process instance ID that acts as a parent process instance of the process instance that is intended to create. In other words, this is the ID of the process instance on which you want to base your new process instance. <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-character process instance ID.
CustomID	<u>Definition:</u> A work object ID specified within a process instance. <u>Type</u> string <u>Allowed Values:</u> One valid work object ID.
Attributes	<u>Definition:</u> Name-value pairs associated with a custom ID. <u>Type</u> NameValue <u>Allowed Values:</u> A valid custom ID with an associated name.
bInStartImmediately	<u>Definition:</u>

Name	Description
	An obsolete, legacy parameter that must be true.
	Type
	bool
	Allowed Values:
	True

Output

WFEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string processDefinitionName = "EmployeeOnboardProcess";

// get UUID of released process definition
string ProcessID =
svc.GetReleasedPID(processDefinitionName);

// assign UUID of process instance
string ProcessInstID = UUID.GetID();

// process instance name that has to be unique within process definition ID
string ProcInstName = string.Format("{0}-{1}",
processDefinitionName DateTime.Now.Ticks );

// work object ID
string WorkObjID = UUID.GetID();
//parent process instance ID is required if this is to create a sub
process. If not, just provide null
string SuperProcInstID = .. // for example,
"09315f0ae769429bbfb243f888bcb09f" or null
List<NameValue> ds = new List<NameValue>();
ds.Add(new NameValue("CustomAttrKey1", "CustomAttrValue1"));
ds.Add(new NameValue("CustomAttrKey2", true));
ds.Add(new NameValue("CustomAttrKey3", 12345));
WFEvent evt = svc.CreateProcInstEx(
    ProcessID,
    ProcessInstID,
    ProcInstName,
    WorkObjID,
    SuperProcInstID,
    WorkObjID,
    ds.ToArray(),
    true);
```

Supported Versions

3.2.0.4 and higher

Delete Process Instance

API Type

Web Services

Description

Deletes a process instance. This method removes the specified process instance and all the associated data from the database, such as work items, email, and activity instances associated with this process instance. It may take some time to complete this transaction.

Syntax

```
public void DeleteProcInst(string processInstanceID)
```

Parameters

Name	Description
processInstanceID	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid process instance ID

Output

None.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string processInstanceID = ..// the ID of the process instance to be
canceled.

try
{
    WFEvent evt = svc.DeleteProcInst(processInstanceID);
}

catch( Exception ex)
{
}
```

```
base.ShowMessage ( base.GetSoapMessage (ex) );  
}
```

Supported Versions

4.0.1 and higher

Get Events By Process Instance ID

API Type

Web Services

Description

Retrieves all the events that have occurred for a specified process instance.

Syntax

```
public virtual WFEvent[] GetEventsByProcInstID(string processInstanceID)
```

Parameters

Name	Description
processInstanceID	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID

Output

An array of [WFEvent](#) objects.

Example

```
IWFWorkflowService svc = GetWorkflowService();  
string processInstanceID = ... // process instance ID  
  
try  
{  
    WFEvent[] events = svc.GetEventsByProcInstID(processInstanceID);  
    for (int i = 0; i < events.Length; i++)  
    {  
        Console.WriteLine("Event ID: '" + events[i].EventID + "'");  
        Console.WriteLine("Event Name: '" + events[i].EventName + "'");  
    }  
}
```

```
}  
catch (Exception ex)  
{  
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));  
}
```

Supported Versions

3.2.0.4 and higher

Get Process Instance

API Type

Web Services

Description

Retrieves basic information about a specified process instance.

Syntax

```
public virtual WFBBaseProcessInstance GetProclnst(string processInstanceId)
```

Parameters

Name	Description
processInstanceId	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID

Output

WFBBaseProcessInstance object that contains basic information about a process instance. It returns null if the process instance ID does not exist.

Example

```
// This is sample code for console application  
IWFWorkflowService svc = GetWorkflowService();  
string processInstanceId = ...// process instance ID  
  
try  
{
```

```
//Returns an instance of WFBaseProcessInstance type.  
WFBaseProcessInstance processInstance =  
    svc.GetProcInst(processInstanceId);  
}  
  
catch (Exception ex)  
{  
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));  
}
```

Supported Versions

3.2.0.4 and higher

Get Process Instance Attribute

API Type

Web Services

Description

Retrieves a single attribute for a specified process instance.

Syntax

```
public virtual KeyValue GetProcInstAttr(string processInstanceId, string attributeName)
```

Parameters

Name	Description
processInstanceId	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID
attributeName	<u>Definition:</u> The name of the process instance attribute you want. <u>Type</u> string <u>Allowed Values:</u> • DefID - The ID of the process definition.

Name	Description
	• DefName - The name of the process definition. • ProcInstName - The name of the process instance. • Status - The current status of the process instance. • SuperProcInstID - The parent process instance ID. • workObjectID - The ID of the work object. • StartDate - The date and time when the process instance was started. • DueDate - The date that the process instance is expected to be complete. • LastModifiedDate - The date and time that the last modification was made to the process instance.

Output

Returns the [KeyValue](#) for the attribute associated with the process instance as a name-value" pair.

Example

```
//This is console application sample
IWfWorkflowService svc = GetWorkflowService();
string processInstanceId = ...// for example,
"1e3d514d43d3465cae6ec3bbbd409168";
string attributeName = "DefName";

try
{
    //Returns attribute associated with the Process Instance as
    "Name-Value" pair.
    KeyValue processInstanceAttribute =
    svc.GetProcInstAttr(processInstanceId, attributeName);
    Console.WriteLine("{0}={1}", processInstanceAttribute.Name, +
    processInstanceAttribute.Value);
}

catch (Exception ex)
{
    Console.WriteLine(ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Process Instance Attributes

API Type

Web Services

Description

Retrieves multiple attributes of a process instance.

Syntax

```
public virtual NameValue[] GetProcInstAttrs(String processInstanceId)
```

Parameters

Name	Description
processInstanceId	<u>Definition:</u> A process instance ID for the process instance you are creating. <u>Type</u> string <u>Allowed Values:</u> A unique, 32 character process instance ID. If you set this value to null, the AgilePoint Server generates the ID.

Attributes

Name	Description
DefID	The ID of the process definition.
DefName	The name of the process definition.
ProcInstName	The name of the process instance.
Status	The current status of the process instance.
SuperProcInstID	The parent process instance ID.
workObjectID	The ID of the work object.
StartDate	The date and time when the process instance was started.

Name	Description
DueDate	The date that the process instance is expected to be complete
LastModifiedDate	The date and time that the last modification was made to the process instance.

Output

Array of [NameValue](#) objects that holds the values of all the requested attributes.

Example

```
// This is console application sample
IWfWorkflowService svc = GetWorkflowService();
string processInstanceId = ...// process instance ID
NameValue[] attributes = svc.GetProcInstAttrs(processInstanceId);
for (int i = 0; i < attributes.Length; i++)
{
    NameValue nv = attributes[i];
    Console.WriteLine("Process Instance Attribute, {0}={1}", nv.Name, nv.Value);
}
```

Supported Versions

3.2.0.4 and higher

Merge Process Instances

API Type

Web Services

Description

Merges 2 or more process instances into one process instance.

These process instances should be based on the same process definition.

Syntax

```
public virtual string MergeProcInsts(WFProcessMergingInstruction instruction)
```

Parameters

Name	Description
instruction	<u>Definition:</u> Specifies the instructions for merging two process instances. <u>Type</u>

Name	Description
	WFProcessMergingInstruction
	<u>Allowed Values:</u>
	A WFProcessMergingInstruction object.

Output

The process instance ID of the merged process instance.

Example

```
string MergeProcessInstances(IWFWorkflowService svc, string[]
processInstanceIDs)
{
    IWFWorkflowService svc = GetWorkflowService();

    // suspends all of process instances to be merged.
    foreach (string id in processInstanceIDs)
    {
        svc.SuspendProcInst(id);
    }

    // query process instances
    string inExpr = ShUtil.Merge(processInstanceIDs, true);
    WFQueryExpr queryExpr = new WFQueryExpr("PROC_INST_ID", SQLExpr.IN,
    WFFAny.Create(inExpr), true);
    WFBaseProcessInstance[] pis = svc.QueryProcInsts(queryExpr);

    // merge custom attributes
    NameValue[] mergedCustomAttributes = GetMergeCustomAttributes(api, pis);
    string procInstID = UUID.GetID();
    string procInstName = pis[0].DefName + "_" + DateTime.Now.ToString()
    + " - Merged";
    string workObjectID = procInstID + " - Merged";

    WFProcessMergingInstruction instruction = new
    WFProcessMergingInstruction();
    instruction.MergingProcessInstanceIDs = processInstanceIDs;
    instruction.MergedProcessInstance = new
    WFProcessMergingInstruction.MergedProcessParameter(
        procInstID,
        procInstName,
        workObjectID,
        null,
        mergedCustomAttributes);
    instruction.Validate();
    return svc.MergeProcInsts(instruction);
}

// sample code for merging custom attributes
private NameValue[] GetMergeCustomAttributes(IWFWorkflowService svc,
WFBaseProcessInstance[] pis)
{
    List<string> workObjectIDs = new List<string>();
    foreach (WFBaseProcessInstance pi in pis)
    {
```

```

        workObjectIDs.Add(pi.WorkObjectID);
    }
    KeyValue[] items = svc.GetCustomAttributesEx(workObjectIDs.ToArray());
    Dictionary<string, System.Xml.XmlDocument> dss =
    new Dictionary<string, System.Xml.XmlDocument>();
    foreach (KeyValue item in items)
    {
        WFCustomAttributes ds = new WFCustomAttributes();
        ds.AttrXml = item.Value;
        System.Xml.XmlDocument xmlDoc = new System.Xml.XmlDocument();
        xmlDoc.LoadXml(ds["//"] as string);
        dss[item.Key] = xmlDoc;
    }
    // master processInstanceID and workObjectID
    string masterProcessInstanceID = pis[0].ProcInstID;
    string masterWorkObjectID = pis[0].WorkObjectID;
    System.Xml.XmlDocument masterXmlDoc = dss[masterWorkObjectID];
    System.Xml.XmlNamespaceManager nsm =
    ShUtil.GetNamespaces(masterXmlDoc);
    System.Xml.XmlNode titleNode =
    masterXmlDoc.SelectSingleNode("/pd:issueTracking/pd:issueTitle", nsm);
    titleNode.InnerText = "Title - Merged";
    System.Xml.XmlNode descriptionNode =

    masterXmlDoc.SelectSingleNode("/pd:issueTracking/pd:description", nsm);
    descriptionNode.InnerText = "Description - Merged";
    System.Xml.XmlNode parent =
    masterXmlDoc.SelectSingleNode("/pd:issueTracking/pd:Persons", nsm);

    // merge rest of children
    foreach (string key in dss.Keys)
    {
        if (key == masterWorkObjectID) continue;
        System.Xml.XmlDocument secondaryXmlDoc = dss[key];
        System.Xml.XmlNamespaceManager nsmgr =
        ShUtil.GetNamespaces(secondaryXmlDoc);
        System.Xml.XmlNode node =
        secondaryXmlDoc.SelectSingleNode("/pd:issueTracking/pd:Persons", nsmgr);
        parent.InnerXml += node.InnerXml;
    }
    return NameValue.Array("//", masterXmlDoc.OuterXml);
}
#endregion

```

Supported Versions

4.5 and higher

Migrate Process Instances

API Type

Web Services

Description

Migrates a process definition from one version to another version.

Syntax

```
public virtual void MigrateProcInst(WFProcessMigrationInstruction instruction, string processInstanceId, string reserved)
```

Parameters

Name	Description
instruction	<u>Definition:</u> Specifies the instructions for migrating a process to a new version. <u>Type</u> WFProcessMigrationInstruction <u>Allowed Values:</u> A WFProcessMigrationInstruction object.
processInstanceId	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID
reserved	<u>Definition:</u> Reserved for future use. <u>Type</u> string <u>Allowed Values:</u> The literal string null. You must pass this as a literal string at this time. This field will be used for other purposes in a later release.

Output

```
IWFWorkflowService svc = GetWorkflowService();  
WFProcessMigrationInstruction pmi = new  
WFProcessMigrationInstruction();  
  
// some code...
```

```
svc.MigrateProcInst(pmi, currentProcessInstanceID, null);  
  
// some more code...
```

Example

```
IWFWorkflowService svc = GetWorkflow();  
string processInstanceID = "DB50CFEFDE464A78AAAA9BD7D6E6D9D0";  
WFProcessMigrationInstruction pmi = new  
WFProcessMigrationInstruction();  
  
//add the correct CurrentActivityUniqueName  
string currentActivityUniqueName = "BudgetRequest";  
  
//add the correct TargetActivityUniqueName  
string targetActivityUniqueName = "BudgetRequestNew";  
bool bCurrentActivated = false;  
pmi.AddMatchingActivity(currentActivityUniqueName, targetActivityUniqueName,  
bCurrentActivated);  
svc.MigrateProcInst(pmi, currentProcessInstanceID, null);
```

Supported Versions

4.0.1 and higher

Promote Process Instance

API Type

Web Services

Description

Promotes a process instance. This method is obsolete.

Syntax

```
public virtual WFEvent PromoteProcInst(String processInstanceID)
```

Parameters

Name	Description
processInstanceID	<u>Definition:</u> A process instance ID for the process instance you are creating. <u>Type</u> string <u>Allowed Values:</u> A unique, 32 character process instance ID.

Name	Description
	If you set this value to null, the AgilePoint Server generates the ID.

Output

[WFEvent](#) object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

None.

Supported Versions

3.2.0.4 and higher

Query Process Instances

API Type

Web Services

Description

Retrieves a list of process instances that match a specified query expression. The WFQueryExpr string is used to generate a query expression, and the client application specifies the query terms.

Syntax

```
public virtual WFBBaseProcessInstance[] QueryProcInsts(WFQueryExpr expr)
```

Parameters

Name	Description
expr	<u>Definition:</u> Specifies the where clause of a SQL query expression. <u>Type</u> WFQueryExpr

Name	Description
	<u>Allowed Values:</u> A valid WFQueryExpr object.

Output

An array of WFBaseProcessInstance objects. It returns null if nothing matches to the query expression.

Example

```
IWFWorkflowService svc = GetWorkflowService();

// query all running process instance
string status = WFProcessInstance.RUNNING;
WFAny any = WFAny.Create(status);
WFQueryExpr expr = new WFQueryExpr("STATUS", SQLExpr.EQ, any, true);

try
{
    // Calling the QueryProcInsts WebMethod, passing the expression as the
    argument.
    WFBaseProcessInstance[] result = svc.QueryProcInsts(expr);

    if (result != null)
    {
        // Iterating through the list of the Process Instance
        foreach (WFBaseProcessInstance processInstance in result)
        {
            //Displaying the Process Instance Details on Console.
            Console.WriteLine("ApplName-->" +
                processInstance.ApplName);
            Console.WriteLine("DefName-->" + processInstance.DefName);
            Console.WriteLine("DefID-->" + processInstance.DefID);
            Console.WriteLine("CompletedDate-->" +
                processInstance.CompletedDate);
            Console.WriteLine("LastModifiedBy-->" +
                processInstance.LastModifiedBy);
        }
    }
}
catch(Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Query Process Instances (Extended Method)

API Type

Web Services

Description

Retrieves a list of process instances with a SQL query expression specified by the client application.

Syntax

```
public virtual WFBaseProcessInstance[] QueryProcInstsEx(string sqlWhereClause)
```

Parameters

Name	Description
sqlWhereClause	<u>Definition:</u> The where clause of the SQL statement you want to query. <u>Type</u> string <u>Allowed Values:</u> The where clause of a SQL statement.

Output

An array of WFBaseProcessInstance objects. It returns null if nothing matches the SQL query expression.

Example

```
// Console application sample code to illustrate QueryProcInstsEx API.
IWfWorkflowService svc = GetWorkflowService();

// SQL Expression
string sqlWhereClause = "STATUS in ('Running','Canceled')";

try
{
    //Calling QueryProcInstsEx WebMethod with sql query expression as argument.
    WFBaseProcessInstance[] result = svc.QueryProcInstsEx(where);

    if (result != null)
    {
        // Iterating through the list of the Process Instance
        foreach (WFBaseProcessInstance processInstance in result)
        {

```

```
//Displaying the Process Instance Details on Console.
Console.WriteLine("ApplName-->" + processInstance.ApplName);
Console.WriteLine("DefName-->" + processInstance.DefName);
Console.WriteLine("DefID-->" + processInstance.DefID);
Console.WriteLine("CompletedDate-->" +
processInstance.CompletedDate);
Console.WriteLine("LastModifiedBy-->" +
processInstance.LastModifiedBy);
}
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Resume Process Instance

API Type

Web Services

Description

Resumes a suspended process instance. The process instance status is changed to running, and the statuses of all the work items (tasks) become Active.

Syntax

```
public virtual WFEvent ResumeProcInst(String processInstanceId)
```

Parameters

Name	Description
processInstanceId	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID

Output

WFEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
// This is console application sample
IWfWorkflowService svc = GetWorkflowService();

try
{
    string processInstanceId = ...// process instance to be suspended.
    WFEvent event = svc.ResumeProcInst(processInstanceId);
}

catch (Exception ex)
{
    Console.WriteLine("Failed: " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Rollback Process Instance

API Type

Web Services

Description

Rolls a process instance back to a previous specified activity, or skips a specified activity if has not yet been completed. When this method is invoked, the current or skipped activity becomes canceled. When skipping, the process moves forward regardless of the activity's status.

Syntax

```
public virtual WFEvent RollbackProcInst(String activityInstanceId)
```

Parameters

Name	Description
activityInstanceId	<u>Definition:</u>

Name	Description
	The unique ID for an activity instance. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid activity instance ID.

Output

WFEEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
//This is console application sample
IWfWorkflowService svc = GetWorkflowService();
string activityInstanceID = ..// target activity instance to roll back

try
{
    WFEEvent evt = workflowService.RollbackProcInst(activityInstanceID);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Split Process Instance

API Type

Web Services

Description

Splits one process instance into 2 or more process instances. The original process is canceled.

Syntax

```
public virtual string[] SplitProcInst(WFProcessSplittingInstruction instruction)
```

Parameters

Name	Description
instruction	<u>Definition:</u> Specifies the instructions for splitting a process instance. <u>Type</u> WFProcessSplitting <u>Allowed Values:</u> A WFProcessSplitting object.

Output

A collection of strings that contain the process instance IDs for the process instances that were created from the split.

Example

```
// This is console application sample
public string[] SplitProcessInstance(string processInstanceID)
{
    IWFWorkflowService svc = GetWorkflowService();
    WFBaseProcessInstance processInstance =
    svc.GetProcInst(processInstanceID);
    WFCustomAttributes ds = ds = new
    WFCustomAttributes(processInstance.WorkObjectID);
    ds.AttrXml = svc.GetCustomAttrs(processInstance.WorkObjectID);
    string xml = ds["//"] as string;
    Console.WriteLine("Splitting custom attributes ...");
    List<NameValue[]> splittedCustomAttributes =
    GetSplittedCustomAttributes(xml);
    WFProcessSplittingInstruction instruction;
    instruction = new WFProcessSplittingInstruction();
    instruction.SplittingProcessInstanceID = procInstID;
    List<string> workObjectIDs = new List<string>();
    for (int i = 0; i < splittedCustomAttributes.Count; ++i)
    {
        string SplittingProcessInstanceID = UUID.GetID();
        string splittedProcInstName = string.Format("{0} - Splitted -
{1}", pi.ProcInstName, i + 1);
        string splittedWorkObjectID = string.Format("{0} - Splitted -
{1}", pi.WorkObjectID, i + 1);
        instruction.Add(
            SplittingProcessInstanceID,
            splittedProcInstName,
            splittedWorkObjectID,
```

```

        null,
        splittedCustomAttributes[i]);
    }
    instruction.Validate();
    Console.WriteLine("Suspending process instance...");
    svc.SuspendProcInst(procInstID);
    System.Threading.Thread.Sleep(1000);
    string[] ids = svc.SplitProcInst(instruction);
    return ids;
}

// function to split custom attributes
private List<NameValue[]> GetSplittedCustomAttributes(string xml)
{
    System.Xml.XmlDocument xmlDoc = new System.Xml.XmlDocument();
    xmlDoc.PreserveWhitespace = true; xmlDoc.LoadXml(xml);
    System.Xml.XmlNamespaceManager nsm = ShUtil.GetNamespaces(xmlDoc);
    System.Xml.XmlNode parent =
        xmlDoc.SelectSingleNode("/pd:issueTracking/pd:Persons", nsm);

    // split repeating notes - 'pd:issueTracking/pd:Persons/pd:Person'
    System.Xml.XmlNodeList nodes = parent.SelectNodes("pd:Person", nsm);

    if (nodes == null || nodes.Count < 2)
    {
        throw new InvalidOperationException("Failed to get splitted
        customAttributes,
        'pd:issueTracking/pd:Persons/pd:Person' does not have
        multiple node.");
    }

    // remove child nodes
    System.Xml.XmlNode titleNode =
        xmlDoc.SelectSingleNode("/pd:issueTracking/pd:issueTitle", nsm);
    string title = titleNode.InnerText;
    System.Xml.XmlNode descriptionNode =
        xmlDoc.SelectSingleNode("/pd:issueTracking/pd:description", nsm);
    string description = descriptionNode.InnerText;
    List<NameValue[]> splittedCustomAttributes = new
    List<NameValue[]>();
    for( int i = 0; i < nodes.Count; ++i)
    {
        //change title
        titleNode.InnerText = title + "-Splitted-" + i;
        descriptionNode.InnerText = description + "- Splitted-" + i;
        parent.RemoveAll();
        parent.AppendChild(nodes[i]);
        splittedCustomAttributes.Add( NameValue.Array("//",
            xmlDoc.OuterXml) );
    }
    return splittedCustomAttributes;
}

```

Supported Versions

4.5 and higher

Start Process Instance

API Type

Web Services

Description

Starts a process instance that is not set to start immediately on creation. This method is obsolete.

Syntax

```
public virtual WFEvent StartProcInst(string processInstanceId)
```

Parameters

Name	Description
processInstanceId	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID

Output

[WFEvent](#) object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

None.

Supported Versions

3.2.0.4 and higher

Suspend Process Instance

API Type

Web Services

Description

Suspends a process instance. The process instance status is changed to Suspended, and the statuses of all the work items (tasks) become Pending.

Syntax

```
public virtual WFEvent SuspendProcInst(string processInstanceId)
```

Parameters

Name	Description
processInstanceId	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid process instance ID

Output

[WFEvent](#) object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
// This is console application sample
IWfWorkflowService svc = GetWorkflowService();

try
{
    string processInstanceId = ...// process instance to be suspended.
    WFEvent event = svc.SuspendProcInst(processInstanceId);
}
```

```
catch (Exception ex)
{
    Console.WriteLine("Failed: " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Update Process Instance

API Type

Web Services

Description

Updates attributes of a workflow process instance. The attributes that can be updated are listed in the attribute table.

Syntax

```
public void UpdateProclnst(string processInstanceId, NameValue[] attributes)
```

Parameters

Name	Description
processInstanceId	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID
attributes	<u>Definition:</u> Name-value pairs associated with a custom ID. <u>Type</u> NameValue <u>Allowed Values:</u> A valid custom ID with an associated name.

Attributes

Name	Description
ProcInstName	The name of the process instance.
DueDate	The date that the process instance is expected to be complete
workObjectID	The ID of the work object.

Output

None.

Example

```
// This is console application sample to update process instance name
IWfWorkflowService svc = GetWorkflowService();
string processInstanceID = ... // process instance ID
string newProcessInstanceName = "[new process instance name]";
DateTime newDueDate = DateTime.Now.AddDays(7.0);

try
{
    WfBaseProcessInstance inst = svc.GetProcInst(processInstanceID);
    NameValue[] attributes = new NameValue[]
    {
        new NameValue("ProcInstName", newProcessInstanceName),
        new NameValue("DueDate", newDueDate)),
    };

    // update process instance
    svc.UpdateProcInst(processInstanceID, attributes);

    // check if it has been updated.
    string processInstanceName =
    svc.GetProcInst(processInstanceID).ProcInstName;
    Console.WriteLine("New Process Instance Name= '{0}'",
    processInstanceName);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Methods for Activity Instances

This section describes service calls related to activity instances.

Cancel Activity Instance

API Type

Web Services

Description

Cancels a manual activity instance along with all manual work items associated with the specified manual activity instance ID. Note that an activity instance can be associated with one or more manual work items. Once the manual activity instance is canceled, the process instance will move forward to the next activity.

Syntax

```
public virtual WFEvent CancelActivityInst(string activityInstanceID)
```

Parameters

Name	Description
activityInstanceID	<u>Definition:</u> The unique ID for an activity instance. <u>Type</u> string <u>Allowed Values:</u> A valid activity instance ID.

Output

[WFEvent](#) object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string activityInstanceID = ...// activity instance needs to be
canceled.

try
{
    WFEvent evt = svc.CancelActivityInst(activityInstanceID);
```

```
    }  
catch (Exception ex)  
{  
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));  
}
```

Supported Versions

3.2.0.4 and higher

Get Activity Instance

API Type

Web Services

Description

Retrieves basic information for a specified activity instance.

Syntax

```
public virtual WFBBaseActivityInstance GetActivityInst(string activityInstanceId)
```

Parameters

Name	Description
activityInstanceId	<u>Definition:</u> The unique ID for an activity instance. <u>Type</u> string <u>Allowed Values:</u> A valid activity instance ID.

Output

WFBBaseActivityInstance object.

Example

```
IWFWorkflowService svc = GetWorkflowService();  
string activityInstanceId = ..// activity instance  
  
try  
{  
    WFBBaseActivityInstance activityInstance =  
        _svc.GetActivityInst(activityInstanceId);  
}
```

```
Console.WriteLine("DisplayName" + activityInstance.DisplayName);
Console.WriteLine("ID" + activityInstance.ID);
Console.WriteLine("CompletedDate" + activityInstance.CompletedDate);
Console.WriteLine("DueDate" + activityInstance.DueDate);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Activity Instance Status

API Type

Web Services

Description

Retrieves all the status of all activity instances for a specified process instance.

Syntax

```
public virtual KeyValue[] GetActivityInstStatus(string procInstID)
```

Parameters

Name	Description
procInstID	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID

Output

An array of [KeyValue](#) objects that holds pairs of activity definition names and statuses. The statuses can be Passed, Active, Pending, Activated, Canceled, or null.

Example

```
// This is console application sample.
IWFWorkflowService svc = GetWorkflowService();
```

```
string procInstID = ..// for example, "02C3FA88ADE04750A34B5B3168C25793";

try
{
    KeyValue[] resultList = svc.GetActivityInstStatus(procInstID);
    foreach (KeyValue result in resultList)
    {
        System.Console.WriteLine("Activity Definition ID: '{0}'", result.Key);
        System.Console.WriteLine("Status: '{0}'",
            result.Value);
    }
}

catch( Exception ex )
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Activity Instances By Process Instance ID

API Type

Web Services

Description

Retrieves the status of all activity instances for a specified process instance.

Syntax

```
public virtual WFBaseActivityInstance[] GetActivityInstsByPIID(string processInstanceID)
```

Parameters

Name	Description
processInstanceID	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID

Output

An array of WFBaseActivityInstance objects.

Example

```
// This is console application sample
IWfWorkflowService svc = GetWorkflowService();
string processInstanceId = ..//"02C3FA88ADE04750A34B5B3168C25793";

try
{
    WFBaseActivityInstance[] activityInstance =
    svc.GetActivityInstsByPIID(processInstanceId);
    foreach (WFBaseActivityInstance activity in activityInstance)
    {
        System.Console.WriteLine("Activity DispalyName: '{0}'",
activity.DisplayName);
        System.Console.WriteLine("CompletedDate: '{0}'", activity.CompletedDate);
    }
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex) );
}
```

Supported Versions

3.2.0.4 and higher

Query Activity Instances

API Type

Web Services

Description

Retrieves activity instances that match a query expression.

Syntax

```
public virtual WFBaseActivityInstance[] QueryActivityInsts(WFQueryExpr expr)
```

Parameters

Name	Description
expr	<u>Definition:</u> Specifies the where clause of a SQL query expression.

Name	Description
	<u>Type</u> WFQueryExpr <u>Allowed Values:</u> A valid WFQueryExpr object.

Output

An array of WFBBaseActivityInstance objects.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string processInstanceID = ...
WFAny any = WFAny.Create(processInstanceID);
WFQueryExpr expr = new WFQueryExpr("PROC_INST_ID", SQLExpr.IN, any,
true);

try
{
    WFBBaseActivityInstance[] ais = svc.QueryActivityInsts(expr);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Rollback Activity Instance

API Type

Web Services

Description

Rolls back a manual activity instance to the token position EN – that is, the state where the activity is entered. All work items associated with the manual activity instance with the status of NEW, OVERDUE, or ASSIGNED are canceled.

Syntax

```
public virtual WFEvent RollbackActivityInst(string activityInstanceID)
```

Parameters

Name	Description
activityInstanceID	<u>Definition:</u> The unique ID for an activity instance. <u>Type</u> string <u>Allowed Values:</u> A valid activity instance ID.

Output

[WFEvent](#) object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
// This is console application sample
IWFWorkflowService svc = GetWorkflowService();
string activityInstanceID = ...

try
{
    //Rolling back the activity instance
    WFEvent evt = svc.RollbackActivityInst(activityInstanceID);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Rollback Activity Instances

API Type

Web Services

Description

Rolls back a process instance according to a specified instruction. The class `WFPartialRollbackInstruction` is used to specify detailed information about the instruction.

Syntax

```
public virtual WFEvent RollbackActivityInsts(WFPartialRollbackInstruction instruction)
```

Parameters

Name	Description
instruction	<u>Definition:</u> Specifies the instructions for the partial rollback. <u>Type</u> <code>WFPartialRollbackInstruction</code> <u>Allowed Values:</u> A <code>WFPartialRollbackInstruction</code> object.

Output

`WFEvent` object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
//Sample for partial rollback
IWfWorkflowService svc = GetWorkflowService();

// PartialRollback unit
WFPartialRollbackInstruction.PartialRollbackUnit unit1 =
    new WFPartialRollbackInstruction.PartialRollbackUnit();
unit1.DestinationActivityInstanceID =
    ... // destination activity instance ID
```



```

unit1.SourceActivityInstanceIDs =
    new string[] { ... }; // array of source activity instance ID
WFPartialRollbackInstruction.PartialRollbackUnit unit2 =
    new WFPartialRollbackInstruction.PartialRollbackUnit();
unit2.DestinationActivityInstanceID =
    ... // destination activity instance ID
unit2.SourceActivityInstanceIDs =
    new string[] { ... }; // array of source activity instance ID
WFPartialRollbackInstruction instruction =
    new WFPartialRollbackInstruction();
instruction.PartialRollbackUnits =
    new WFPartialRollbackInstruction.PartialRollbackUnit[]
    {unit1,unit2};

try
{
    //Rolling back the activity instance
    WFEvt evt = workflowService. RollbackActivityInsts(instruction);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

```

Supported Versions

4.6 and higher

Methods for Manual Work Items (Tasks)

This section describes service calls related to manual work items (tasks).

Assign Work Item

API Type

Web Services

Description

Assigns a work item to a user, which often means claiming a work item for oneself. This is often used with task pools where work items are created, and then multiple users are notified, but the work item is not immediately assigned to a user. A user then claims the work item, or his manager assigns it to him. The user must have privileges to claim or assign the work item.

Syntax

```
public virtual WFEvt AssignWorkItem(string workItemID)
```

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.

Output

[WFEvent](#) class that represents the workflow event instance that is created when the task is assigned.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string workItemID = ...// work item ID

try
{
    WFEvent evt = svc.AssignWorkItem(workItemID);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Assign Work Item (Extended Method)

API Type

Web Services

Description

Assigns a work item to a user, which often means claiming a work item for oneself. This is often used with task pools where work items are created, and then multiple users are notified, but the work item is not immediately assigned to a user. A user then claims the work item, or his manager assigns it to him. The user must have privileges to claim or assign the work item. This method extends `AssignWorkItem()` by allowing you to specify client data.

Syntax

```
public virtual WFEvent AssignWorkItemEx(string workItemID, string clientData)
```

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.
clientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A string that contains the client data. If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.

Output

WFEvent object representing the workflow event instance raised by the invocation of the work assignment.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string workItemID = ...// work item ID
string clientData = null;

try
{
    WFEvent evt = svc.AssignWorkItemEx(workItemID, clientData);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Cancel Work Item

API Type

Web Services

Description

Cancels a manual work item based on a specified manual work item identifier. Only the following manual work item status can transition to a Canceled status: Assigned, New, Pseudo, and Overdue.

Syntax

```
public virtual WFEvent CancelWorkItem(string workItemID)
```

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.

Output

WFEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();  
string workItemID = ... //  
  
try
```

```
{
    WFEvt evt = svc.CancelWorkItem(workItemID);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Cancel Work Item (Extended Method)

API Type

Web Services

Description

Cancels a manual work item based on a specified manual work item identifier. This method also contains an added routine to track the method call duration. Only the following manual work item status can transition to a Canceled status: Assigned, New, Pseudo, and Overdue.

Syntax

public virtual WFEvt CancelWorkItemEx(string workItemID, string clientData)

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.
clientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u> string <u>Allowed Values:</u>

Name	Description
	A string that contains the client data. If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.

Output

WFEEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string workItemID = ...// work item ID
string clientData = null;

try
{
    WFEEvent evt = svc.CancelWorkItemEx(workItemID, clientData);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Complete Work Item

API Type

Web Services

Description

Marks a work item as completed.

Syntax

```
public virtual WFEEvent CompleteWorkItem(string workItemID)
```

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.

Output

[WFEvent](#) object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string workItemID = ...// work item ID

try
{
    WFEvent evt = svc.CompleteWorkItem(workItemID);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Complete Work Item (Extended Method)

API Type

Web Services

Description

Marks a work item as completed with client data.

Syntax

```
public virtual WFEvent CompleteWorkItemEx(string workItemID, string clientData)
```

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.
clientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u> string <u>Allowed Values:</u> A string that contains the client data. If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.

Output

WFEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();
```



```
string workItemID = ...// work item ID
string clientData = null;

try
{
    WFEvt evt = svc. CompleteWorkItemEx(workItemID, clientData);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Create Linked Work Item

API Type

Web Services

Description

Creates a manual work item that is linked to another manual work item. The work item you create does not depend on the completion of the work item to which it is linked. In other words, the original (source) work item can be marked as completed before new work item is completed.

Syntax

```
public virtual WFEvt CreateLinkedWorkItem(string SourceWorkItemID, string WorkToPerform, string UserID,
WFTimeDuration duration, string ClientData)
```

Parameters

Name	Description
SourceWorkItemID	<u>Definition:</u> An ID that represents the original, or source, work item. <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-character ID.
WorkToPerform	<u>Definition:</u> Represents the task that performed by the participants of the activity.

Name	Description
	<u>Type</u> string <u>Allowed Values:</u> A valid work to perform name.
UserID	<u>Definition:</u> Specifies the user ID. <u>Type</u> string <u>Allowed Values:</u> A valid user ID.
duration	<u>Definition:</u> Specifies the duration settings of a work item. <u>Type</u> WFTimeDuration <u>Allowed Values:</u> A valid WFTimeDuration object.
ClientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u> string <u>Allowed Values:</u> A string that contains the client data. If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.

Output

None.

Example

```
IWFWorkflowService svc = base.GetWorkflowService();
// get existing work item
string workItemID = ..// for example,
"90CF843AC57644058A391FBFA030F607"
```

```
try
{
    // Get the source WFManualWorkItem object
    WFManualWorkItem SourceWorkItemID = svc.GetWorkItem(workItemID)
    string WorkToPerform = SourceWorkItemID.Name; //different
    WorkToPerform can be used if desired
    WFTimeDuration duration = new WFTimeDuration();
    duration.Length = "15"; //for example, 15 days
    duration.Unit = WFTimeUnit.DAY;
    string UserID = @"[DOMAIN NAME]\username"; //the participant of the
    linked work item
    WFEvent evt = svc.CreateLinkedWorkItem(
        SourceWorkItemID.WorkItemID,
        WorkToPerform,
        UserID,
        duration,
        null);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Create Linked Work Item (Extended Method)

API Type

Web Services

Description

Creates a manual work item that is linked to another manual work item. The extended parameter bDependent is used to specify the dependency between the original work item and the linked work item. If bDependent is false, the work items are independent, just like CreateLinkedWorkItem(). If bDependent is true, the original (source) work item cannot be marked as completed before new work item has been completed.

Syntax

```
public virtual WFEvent CreateLinkedWorkItemEx(string SourceWorkItemID, string WorkToPerform, string
UserID, WFTimeDuration duration, string ClientData, bool bDependent)
```

Parameters

Name	Description
SourceWorkItemID	<u>Definition:</u> An ID that represents the original, or source, work item.

Name	Description
	<u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-character ID.
WorkToPerform	<u>Definition:</u> Represents the task that performed by the participants of the activity. <u>Type</u> string <u>Allowed Values:</u> A valid work to perform name.
UserID	<u>Definition:</u> Specifies the user ID. <u>Type</u> string <u>Allowed Values:</u> A valid user ID.
duration	<u>Definition:</u> Specifies the duration settings of a work item. <u>Type</u> WFTimeDuration <u>Allowed Values:</u> A valid WFTimeDuration object.
ClientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u> string <u>Allowed Values:</u> A string that contains the client data.

Name	Description
	If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.
bDependent	<u>Definition:</u> Specifies whether the target work item waits for a linked work item to be completed or canceled before moving forward. <u>Type</u> bool <u>Allowed Values:</u> • True - The source work item waits until the linked work item is either completed or canceled, before it can be completed or canceled. • False - The source work item can be completed or canceled regardless of whether the linked work item is completed or canceled.

Output

WFEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = base.GetWorkflowService();

// get existing work item
string workItemID = ..// for example, "90CF843AC57644058A391FBFA030F607"

try
{
    // Get the source WFManualWorkItem object
    WFManualWorkItem SourceWorkItem = svc.GetWorkItem(workItemID)
    string WorkToPerform = sourceWorkItem.Name; //different
    WorkToPerform can be used if desired
    WFTimeDuration duration = new WFTimeDuration("15", WFTimeUnit.DAY, false);
    string UserID = @"[DOMAIN NAME]\username"; //the participant of the
    linked work item
    WFEvent evt = svc.CreateLinkedWorkItem(
        SourceWorkItem.WorkItemID,
        WorkToPerform,
```

```
        UserID,  
        duration,  
        null,  
        true);  
    }  
  
    catch( Exception ex)  
    {  
        Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));  
    }
```

Supported Versions

4.6 and above

Create Pseudo Work Item

API Type

Web Services

Description

Creates a task by a specific AgileWork or other module that has the following characteristics:

- It does not have to be completed in order for a process to advance to the next steps.
- Unless specifically canceled, it remains active through the duration of the entire process, not just the duration of the AgileWork or other module that created it.

This provides a way for tasks to be included in a user's or manager's task list purely for monitoring purposes.

Syntax

```
public virtual WFEEvent CreatePseudoWorkItem(string SourceWorkItemId, string WorkToPerform, string UserID,  
WFTimeDuration duration, string ClientData, bool bReserved)
```

Parameters

Name	Description
SourceWorkItemId	<u>Definition:</u> An ID that represents the original, or source, work item. <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-character ID.
WorkToPerform	<u>Definition:</u>

Name	Description
	Represents the task that performed by the participants of the activity. <u>Type</u> string <u>Allowed Values:</u> A valid work to perform name.
UserID	<u>Definition:</u> Specifies the user associated with the work item. <u>Type</u> string <u>Allowed Values:</u> A valid user ID.
duration	<u>Definition:</u> Specifies the duration settings of a work item. <u>Type</u> WFTimeDuration <u>Allowed Values:</u> A valid WFTimeDuration object.
ClientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u> string <u>Allowed Values:</u> A string that contains the client data. If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.

Output

[WFEvent](#) object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.

- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = base.GetWorkflowService();

// get existing work item
string workItemID = ..// for example, "90CF843AC57644058A391FBFA030F607"

try
{
    // Get the source WFManualWorkItem object
    WFManualWorkItem SourceWorkItem = svc.GetWorkItem(workItemID)
    string WorkToPerform = sourceWorkItem.Name; //different
    WorkToPerform can be used if desired
    WFTimeDuration duration = new WFTimeDuration("15", WFTimeUnit.DAY, false);
    string UserID = @"[DOMAIN NAME]\username"; //the participant of the linked work
item
    WFEvt evnt = svc.CreatePseudoWorkItem (
        SourceWorkItem.WorkItemID,
        WorkToPerform,
        UserID,
        duration,
        null,
        false);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Create Work Item

API Type

Web Services

Description

Creates a manual work item.

Syntax

```
public virtual WFEvt CreateWorkItem(string ActivityInstanceID, string WorkToPerform, string UserID,
WFTimeDuration Length, string ClientData)
```


Parameters

Name	Description
ActivityInstanceId	<u>Definition:</u> The unique ID for an activity instance. <u>Type</u> string <u>Allowed Values:</u> A valid activity instance ID.
WorkToPerform	<u>Definition:</u> Represents the task that performed by the participants of the activity. <u>Type</u> string <u>Allowed Values:</u> A valid work to perform name.
UserID	<u>Definition:</u> Specifies the user associated with the work item. <u>Type</u> string <u>Allowed Values:</u> A valid user ID.
Length	<u>Definition:</u> Specifies the length of time duration. <u>Type</u> string <u>Allowed Values:</u> A valid time duration length.
ClientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u>

Name	Description
	string <u>Allowed Values:</u> A string that contains the client data. If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.

Output

[WFEvent](#) object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string ActivityInstanceID = ... // for example, "0172460E0AF943C6A6520044452BCAB3";
string WorkToPerform = ... // for example, "SubmitRequest";
//different WorkToPerform can be used if desired

WFTimeDuration Length = new WFTimeDuration("15", WFTimeUnit.DAY, true );// business
time
string UserID = @"[DOMAIN NAME]\username"; //the participant of the linked work
item

try
{
    WFEventevt = svc.CreateWorkItem(ActivityInstanceID,
    WorkToPerform, UserID, Length, null);
}

catch(Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Work Item

API Type

Web Services

Description

Retrieves the manual work item object for a specified ID.

Syntax

```
public virtual WFManualWorkItem GetWorkItem(string workItemID)
```

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.

Output

WFManualWorkItem object.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string workItemID = ...// for example, "54A648A0A3004A02981E7F0848820FE7";

try
{
    WFAutomaticWorkItem workItem = svc.GetWorkItem(workItemID);
    Console.WriteLine("{0}", workItem.Name);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Work List By User ID

API Type

Web Services

Description

Retrieves a work item collection by specifying a user name and work item status.

Syntax

```
public virtual WFManualWorkItem[] GetWorkListByUserID(string UserName, string Status)
```

Parameters

Name	Description
UserName	<u>Definition:</u> Specifies a qualified user name of the instance. A qualified user name formats as [Domain Name]\[Logon Username] or [Local host name]\[Logon Username]. <u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.
Status	<u>Definition:</u> The status of the work item. <u>Type</u> string <u>Allowed Values:</u> • New - The work item is in a pool, more than one participant shares the work item and needs to be assigned. • Assigned - The work item is assigned to a participant and needs to be performed. • Removed - The work item was originally in a pool (status is new), and didn't get assigned to the user. • Completed - The work item is completed. • Reassigned - The work item is re-assigned to the other user. • Canceled - The work item is canceled. • Overdue - The work item is overdue. • Carbon - The work item is a carbon copy that does not affect process instance running.

Output

Array of WFManualWorkItem objects.

Example

```
//Get all WFManualWorkItem assigned to user
IWFWorkflowService svc = GetWorkflowService();
string userID = ...// for example, @"Demo3\Administrator";
string Status= string.format("{0};{1}",
WFManualWorkItem.ASSIGNED, WFManualWorkItem.OVERDUE);

try
{
    WFManualWorkItem[] workItems = svc.GetWorkListByUserID(userID,
    Status);
    foreach (WFManualWorkItem workItem in workItems)
    {
        Console.WriteLine("{0}", workItem.ApplName);
        Console.WriteLine("{0}", workItem.AssignedDate);
        Console.WriteLine("{0}", workItem.DefName);
        Console.WriteLine("{0}", workItem.DueDate);
    }
}

catch(Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Query Work List

API Type

Web Services

Description

Retrieves a list of manual work items that match a specified query expression.

Syntax

```
public virtual WFManualWorkItem[] QueryWorkList(WFQueryExpr expr)
```

Parameters

Name	Description
expr	<u>Definition:</u>

Name	Description
	Specifies the where clause of a SQL query expression. <u>Type</u> WFQueryExpr <u>Allowed Values:</u> A valid WFQueryExpr object.

Output

An array of WFManualWorkItem objects that contain the work item data.

Example

```
IWFWorkflowService svc = GetWorkflowService();
WFAny any = WFAny.Create(WFManualWorkItem.ASSIGNED);
WFQueryExpr expr = new WFQueryExpr("WF_MANUAL_WORKITEM.STATUS",
SQLExpr.EQ, any, true);

try
{
    WFManualWorkItem[] workItems = svc.QueryWorkList(expr);

    // Iterating through the list of the ManualWorkItem
    foreach (WFManualWorkItem workItem in workItems)
    {
        Console.WriteLine("{0}", workItem.ApplName);
        Console.WriteLine("{0}", workItem.AssignedDate);
        Console.WriteLine("{0}", workItem.DefName);
        Console.WriteLine("{0}", workItem.DueDate);
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Query Work List (Extended Method)

API Type

Web Services

Description

Retrieves a list of manual work items that match a SQL statement.

Syntax

```
public virtual WFManualWorkItem[] QueryWorkListEx(string sqlWhereClause)
```

Parameters

Name	Description
sqlWhereClause	<u>Definition:</u> The where clause of the SQL statement you want to query. <u>Type</u> string <u>Allowed Values:</u> The where clause of a SQL statement.

Output

An array of WFManualWorkItem objects.

Example

```
IWFWorkflowService svc = GetWorkflowService();

//Query Expression string for Comparison of the WORK_ITEM_ID
string sqlWhereClause = "WORK_ITEM_ID in ('0006EE0244ED431CB93F6253060DD21F', ...)";

try
{
    // Calling the QueryWorkListEx API with argument sql query expression string.
    WFManualWorkItem[] workItems = _svc.QueryWorkListEx(sqlWhereClause);

    // Iterating through the list of the ManualWorkItem
    foreach (WFManualWorkItem workItem in workItems)
    {
        Console.WriteLine("{0}", workItem.ApplName);
        Console.WriteLine("{0}", workItem.AssignedDate);
        Console.WriteLine("{0}", workItem.DefName);
        Console.WriteLine("{0}", workItem.DueDate);
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Reassign Update Work Item

API Type

Web Services

Description

Reassigns a manual work item to another participant.

Syntax

```
public virtual WFEvent ReassignUpdateWorkItem(string workItemID, string originalUserID, string newAssigneeUserID, string clientData)
```

Parameters

Name	Description
WorkItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.
originalUserID	<u>Definition:</u> The user ID for the original user assigned the work item. <u>Type</u> string <u>Allowed Values:</u> A valid user ID.
newAssigneeUserID	<u>Definition:</u> The user name for a user to whom you want to assign a work item. <u>Type</u> string <u>Allowed Values:</u>

Name	Description
	A valid user name.
clientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u> string <u>Allowed Values:</u> A string that contains the client data. If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.

Output

WFEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string WorkItemID = ...// "0006EE0244ED431CB93F6253060DD21F"; // Work item ID
string originalUserID = ...// @"[DOMAIN NAME]\[user name]"; // new user ID
string newAssigneedUserID = ...// @"[DOMAIN NAME]\[user name]"; // new user ID

try
{
    WFEvent evt = ReassignUpdateWorkItem(WorkItemID, originalUserID,
    newAssigneedUserID, null);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

4.6 and higher

Reassign Work Item

API Type

Web Services

Description

Reassigns a work item to another participant, and update the user name.

Syntax

```
public virtual WFEvent ReassignWorkItem(string WorkItemID, string UserName)
```

Parameters

Name	Description
WorkItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.
UserName	<u>Definition:</u> The user name for the user. <u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.

Output

WFEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string WorkItemID = ...// "0006EE0244ED431CB93F6253060DD21F"; // Work item ID
string UserName = ...// @"[DOMAIN NAME]\[user name]"; // new user ID

try
{
    WFEEvent evt = svc.ReassignWorkItem(WorkItemID, UserName);
}

catch(Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Reassign Work Item (Extended Method)

API Type

Web Services

Description

Reassigns a work item to another participant, and update the user name. The extended method includes client data.

Syntax

public virtual WFEEvent ReassignWorkItemEx(string WorkItemID, string UserName, string ClientData)

Parameters

Name	Description
WorkItemID	Definition: An ID that represents a work item (task). Type string Allowed Values: A valid, unique 32-byte work item (task) ID.
UserName	Definition: The user name for the user.

Name	Description
	<u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.
ClientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u> string <u>Allowed Values:</u> A string that contains the client data. If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.

Output

WFEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
string ClientData= "<?xml version='1.0' ... ..";
string WorkItemID = ...// for example, "0006EE0244ED431CB93F6253060DD21F"; // Work
item ID
string UserName = ...// for example @"[DOMAIN NAME]\[USER NAME]"; // new user ID

try
{
    WFEvent evt = _svc.ReassignWorkItemEx(WorkItemID,
    UserName, ClientData);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Undo Assign Work Item

API Type

Web Services

Description

Unassigns a work item that was previously assigned to a user. This method applies to work items that can be assigned to members of task groups, where a work item can be assigned to or claimed by any of a group of users.

Syntax

```
public virtual WFEvent UndoAssignWorkItem(string workItemID)
```

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.

Output

[WFEvent](#) object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();  
string workItemID = ...// for example, "03ABD59A0EB74D7A8741709478E83877";
```

```
try
{
    WFEvent evt = svc.UndoAssignWorkItem(workItemID);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Undo Assign Work Item (Extended Method)

API Type

Web Services

Description

Unassigns a work item that was previously assigned to a user. This method applies to work items that can be assigned to members of task groups, where a work item can be assigned to or claimed by any of a group of users. The extended method includes client data.

Syntax

```
public virtual WFEvent UndoAssignWorkItemEx(string workItemID, string clientData)
```

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.
clientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u> string

Name	Description
	<u>Allowed Values:</u> A string that contains the client data. If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.

Output

WFEvent object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
string url = "http://[hostname]:[port]/AgilePointServer";
string workItemID = ...// for example,
"03ABD59A0EB74D7A8741709478E83877";
string clientData = ...//

try
{
    WFEvent evt = svc.UndoAssignWorkItemEx(workItemID, clientData);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Update Work Item

API Type

Web Services

Description

Updates a manual work item or automatic work item.

Syntax

```
public virtual void UpdateWorkItem(string workItemID, NameValue[] attributes)
```

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.
attributes	<u>Definition:</u> A NameValue array that contains the attributes that needs to be updated in the work item. <u>Type</u> NameValue <u>Allowed Values:</u> For a manual work item, the following attributes can be updated: • NAME • ORIGINAL_USER_ID • CLIENT_DATA, POOL_ID • POOL_INFO • STATUS • USER_ID • PRIORITY • DUE_DATE For an automatic work item, the following attributes can be updated: • DUE_DATE • STATUS - if the value is Canceled, Completed, Overdue, Running, or Waiting.

Output

None.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string workItemID = ...// work item ID of manual work item or automatic work item.

try
{
    NameValue[] attributes = NameValue.Array(
        "NAME", "[New Name]",
        "DUE_DATE", [DateTime]); // for example, DateTime.Now.AddDays(3.0)
    svc.UpdateWorkItem(workItemID, attributes);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Methods for Automatic Work Items (Procedures)

This section describes service calls related to automatic work items.

Cancel Procedure

API Type

Web Services

Description

Cancels an automatic work item based on supplied specified automatic work item identifier.

Syntax

```
public virtual WFEvent CancelProcedure(string workItemID)
```

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task).

Name	Description
	<u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.

Output

[WFEvent](#) object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string workItemID = ...//

try
{
    WFEvent evt = svc.CancelProcedure(workItemID);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Complete Procedure

API Type

Web Services

Description

Marks an automatic work item as completed by an asynchronous activity.

Syntax

```
public virtual WFEvent CompleteProcedure(string workItemID)
```

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.

Output

[WFEvent](#) object that provides the status of the transaction. The possible statuses are:

- **Sent** - Indicates event has been sent to engine for processing.
- **Failed** - Indicates event failed to process.
- **Processed** - Indicates event has been processed successfully.
- **Canceled** - Indicates event was canceled.
- **Deferred** - Indicates event does not need to be sent immediately.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string workItemID = ..//

try
{
    svc.CompleteProcedure(workItemID);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Procedure

API Type

Web Services

Description

Retrieves work item data by a specified work item ID.

Syntax

```
public virtual WFAutomaticWorkItem GetProcedure(string workItemID)
```

Parameters

Name	Description
workItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.

Output

WFAutomaticWorkItem object.

Example

```
IWFWorkflowService svc = GetWorkflowService()
string workItemID = ..// for example,
"54A648A0A3004A02981E7F0848820FE7";

try
{
    WFAutomaticWorkItem wItem = svc.GetProcedure(workItemID);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Query Procedure List

API Type

Web Services

Description

Retrieves a list of automatic work items that match a specified query expression.

Syntax

```
public virtual WFAutomaticWorkItem[] QueryProcedureList(WFQueryExpr expr)
```

Parameters

Name	Description
expr	<u>Definition:</u> Specifies the where clause of a SQL query expression. <u>Type</u> WFQueryExpr <u>Allowed Values:</u> A valid WFQueryExpr object.

Output

An array of automatic work items.

Example

```
IWFWorkflowService svc = GetWorkflowService();

try
{
    //WebMethod with sql query expression as argument.
    WFAny any = WFAny.Create(WFAutomaticWorkItem.WAITING);
    WFQueryExpr expr = new WFQueryExpr("STATUS", SQLExpr.EQ, any, true);
    WFAutomaticWorkItem[] result = svc.QueryProcedureList(expr);

    if (result != null)
    {
        // Iterating through the list of the automatic work item
        foreach (WFAutomaticWorkItem re in result)
        {
            Console.WriteLine("ActivityInstID-->" +
                re.ActivityInstID);
            Console.WriteLine("ApplName-->" + re.ApplName);
            Console.WriteLine("ProcInstID-->" + re.ProcInstID);
            Console.WriteLine("CreatedDate-->" + re.CreatedDate);
        }
    }
}

catch (Exception ex)
{
}
```

```
Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));  
}
```

Supported Versions

3.2.0.4 and higher

User Delegation

This section describes service calls related to user delegation processes.

Activate Delegation

API Type

Web Services

Description

Activates a delegation.

Syntax

```
public virtual void ActivateDelegation(string delegationID)
```

Parameters

Name	Description
delegationID	<u>Definition:</u> The unique ID of a delegation object. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid delegation ID.

Output

None.

Example

```
IWFAdminService svc = GetAdminService();  
string delegationID = ...;  
  
try  
{
```

```
        svc.ActivateDelegation(delegationID);
    }

    catch (Exception ex)
    {
        Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
    }
}
```

Supported Versions

3.2.0 and higher

Add Delegation

API Type

Web Services

Description

Creates a rule for delegating one user's tasks to another user. This method uses the WFDelegation class. For more information, see the AgilePoint Class Reference.

Syntax

```
public virtual WFDelegation AddDelegation(WFDelegation delegation)
```

Parameters

Name	Description
delegation	<u>Definition:</u> An object that specifies the details of the delegation rule, including the user whose tasks will be delegated and the designated user to whom to delegate the tasks. <u>Type</u> WFDelegation <u>Allowed Values:</u> A valid WFDelegation object.

Output

WFDelegation object.

Example

```
IWFAdminService svc = GetAdminService();
WFDelegation delegation = new WFDelegation();
```

```
//Set the object properties.
delegation.FromUser = "Demo3\\Andy";
delegation.ToUser = "Demo3\\Joe";
delegation.StartDate = DateTime.Now;
delegation.EndDate = DateTime.Parse("27/10/2009");
delegation.Description = "Delegating Andy's task to Joe";

try
{
    WFDelegation delegation = adminService.AddDelegation(delegation);
    Console.WriteLine("Delegation ID: {0}", delegation.DelegationID);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Cancel Delegation

API Type

Web Services

Description

Cancels a currently operating delegation.

Syntax

```
public virtual void CancelDelegation(string delegationID)
```

Parameters

Name	Description
delegationID	<u>Definition:</u> The unique ID of a delegation object. <u>Type</u> string <u>Allowed Values:</u> A valid delegation ID.

Output

None.

Example

```
IWFAdminService svc = GetAdminService();
string delegationID = ...;

try
{
    svc.CancelDelegation(delegationID);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Delegation

API Type

Web Services

Description

Retrieves a delegation object.

Syntax

```
public virtual WFDelegation GetDelegation(string delegationID)
```

Parameters

Name	Description
delegationID	<u>Definition:</u> The unique ID of a delegation object. <u>Type</u> string <u>Allowed Values:</u> A valid delegation ID.

Output

WFDelegation object that specifies the user whose tasks will be delegated and the designated user to whom to delegate tasks.

Example

```
IWFAdminService svc = GetAdminService();
string delegationID = ...; // for example, "C9A40F4BDA26481FB822C398C4387901"

try
{
    WFDelegation delegation = svc.GetDelegation(delegationID);
    Console.WriteLine("Delegation Id:{0}; From User:{1}; To User:{2}; Status:{3}",
                      delegation.DelegationID,
                      delegation.FromUser,
                      delegation.ToUser,
                      delegation.Status);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/* Output
Delegation Id:C9A40F4BDA26481FB822C398C4387901; From
User:VITBDC\yuvarajn; To User:VITBDC\ravis; Status:Canceled
*/
```

Supported Versions

3.2.0.4 and higher

Get Delegations

API Type

Web Services

Description

Retrieves a list of delegation objects that match the specified parameters. You can leave the parameters null to indicate any.

Syntax

```
public virtual WFDelegation[] GetDelegations(string FromUser, string ToUser, string Status)
```

Parameters

Name	Description
FromUser	<u>Definition:</u> Specifies a user from whom to delegate tasks. <u>Type</u> string <u>Allowed Values:</u> A valid user name.
ToUser	<u>Definition:</u> Specifies the user who will receive the delegated tasks. <u>Type</u> string <u>Allowed Values:</u> A valid user name.
Status	<u>Definition:</u> The status of the associated item. <u>Type</u> string <u>Allowed Values:</u> A valid status.

Output

An array of WFDelegation objects.

Example

```
IWFAdminService svc = GetAdminService();
string FromUser = ...// for example, @"vitbdc\yuvarajn"
string ToUser = ...// for example, null for any
string Status = WFDelegation.ACTIVE;

try
{
    WFDelegation[] delegations = svc.GetDelegations( FromUser, ToUser,
    Status);
    foreach(WFDelegation delegation in delegations)
```

```
{
    Console.WriteLine("Delegation Id:{0}; From User:{1}; To User:{2};
        Status:{3}, {4}=>{5}",
            delegation.DelegationID,
            delegation.FromUser,
            delegation.ToUser,
            delegation.Status,
            delegation.StartDate,
            delegation.EndDate);
}
}
catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Remove Delegation

API Type

Web Services

Description

Removes a delegation from the AgilePoint system.

Syntax

```
public virtual void RemoveDelegation(string delegationID)
```

Parameters

Name	Description
delegationID	<u>Definition:</u> The unique ID of a delegation object. <u>Type</u> string <u>Allowed Values:</u> A valid delegation ID.

Output

None.

Example

```
IWFAdminService svc = GetAdminService();
string delegationID = ...;

try
{
    svc.RemoveDelegation(delegationID);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Update Delegation

API Type

Web Services

Description

Updates a delegation object that has already been created.

Syntax

WFDelegation UpdateDelegation(WFDelegation delegation)

Parameters

Name	Description
delegation	<u>Definition:</u> An object that specifies the details of the delegation rule, including the user whose tasks will be delegated and the designated user to whom to delegate the tasks. <u>Type</u> WFDelegation <u>Allowed Values:</u> A valid WFDelegation object.

Output

Returns an updated instance of WFDelegation.

Example

```
IWFAdminService svc = GetAdminService();
WFDelegation delegation = new WFDelegation();
delegation.DelegationID = ...// unique ID
delegation.FromUser = ...// from user name
delegation.ToUser = ...// to user name
delegation.StartDate = ... // start date
delegation.EndDate = ...// end date

try
{
    WFDelegation updatedDelegation = svc.UpdateDelegation( delegation );
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Methods for Notifications

This section describes service calls related to email notifications.

Cancel Mail Deliverable

API Type

Web Services

Description

Cancels the failed mail deliverable record based on a given message identifier. Note that canceling the failed mail deliverable record prevents it from being recycled or present on a given interval by the AgilePoint engine.

Syntax

```
public virtual void CancelMailDeliverable(string mailID)
```

Parameters

Name	Description
mailID	<u>Definition:</u> Specifies the unique ID for an email notification. <u>Type</u> string <u>Allowed Values:</u> A valid email notification ID.

Output

None.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string mailID = ...

try
{
    svc.CancelMailDeliverable(mailID);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Expecting Send Mail Deliverable

API Type

Web Services

Description

Retrieves all the failed and scheduled to resend email notifications.

Syntax

```
public virtual WFMailDeliverable[] GetExpectingSendMailDeliverable()
```

Parameters

Name	Description
None	Not Applicable

Output

Array of WFMailDeliverable objects.

Example

```
IWFWorkflowService svc = GetWorkflowService();

try
{
    //Returns Array of WFMailDeliverable type
    WFMailDeliverable[] mailDeliverables =
    svc.GetExpectingSendMailDeliverable();
    foreach(WFMailDeliverable m in mailDeliverables)
    {
        Console.WriteLine("Mail ID: '{0}'", m.ID);
        Console.WriteLine("Process Instance ID: '{0}'", m.ProcInstID);
        Console.WriteLine("E-Mail Subject: '{0}'", m.Mail.Subject);
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Mail Deliverables

API Type

Web Services

Description

Retrieves all the mail deliverables for a process instance.

Syntax

```
public virtual WFMailDeliverable[] GetMailDeliverables(string processInstanceId)
```


Parameters

Name	Description
processInstanceId	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID

Output

Array of WFMailDeliverable objects.

Example

```
//Process Instance ID associated with the Process Instance.
IWfWorkflowService svc = GetWorkflowService();
string processInstanceId = ...// for example, "1e3d514d43d3465cae6ec3bbbd409168";

try
{
    WFMailDeliverable[]emailNotifications =
        svc.GetMailDeliverables(processInstanceId);
    foreach(WFMailDeliverable m in emailNotifications)
    {
        Console.WriteLine("Mail ID: '{0}'", m.ID);
        Console.WriteLine("Process Instance ID: '{0}'", m.ProcInstID);
        Console.WriteLine("E-Mail Subject: '{0}'", m.Mail.Subject);
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Resend Mail Deliverable

API Type

Web Services

Description

Resends the mail deliverable with a specified mail ID.

Syntax

```
public virtual void ResendMailDeliverable(string mailID)
```

Parameters

Name	Description
mailID	<u>Definition:</u> Specifies the unique ID for an email notification. <u>Type</u> string <u>Allowed Values:</u> A valid email notification ID.

Output

None.

Example

```
//Sample for using resendMailDeliverable
IWfWorkflowService svc = GetWorkflowService();
string mailID = ... // for example,
"149C3974240F47D3B28EB6D4A3CDCD3F"

try
{
    svc.ResendMailDeliverable(mailID);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Methods for Events

This section describes service calls related to workflow events in the Web Service API.

Get Event

API Type

Web Services

Description

Retrieves an event object. This service call is usually used to check if a service call has been completed.

Syntax

```
public virtual WFEvent GetEvent(string eventID)
```

Parameters

Name	Description
eventID	<u>Definition:</u> Specifies a unique ID for an event. <u>Type</u> string <u>Allowed Values:</u> A unique, 32-character ID.

Output

[WFEvent](#) object.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string eventID = ...// for example, "049C3974240F47D3BA8EB6D4A3CDCD3F";

try
{
    WFEvent evt = _workflowAPI.GetEvent(eventID);
    Console.WriteLine("Event ID: '{0}'", evt.EventID);
    Console.WriteLine("Event Name: '{0}'", evt.EventName);
    Console.WriteLine("Event Status: '{0}'", evt.Status);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Send Mail

This section describes service calls related to sending email using AgilePoint.

Send Mail

API Type

Web Services

Description

Sends an email through AgilePoint Server.

Syntax

```
public virtual void SendMail(String To, String CC, String Subject, String Body)
```

Parameters

Name	Description
To	<u>Definition:</u> Specifies the To portion of an email. <u>Type</u> string <u>Allowed Values:</u> A string that includes one or more email addresses in SMTP format.
CC	<u>Definition:</u> Specifies the CC portion of an email. <u>Type</u> string <u>Allowed Values:</u> A string that includes one or more email addresses in SMTP format.

Name	Description
Subject	<u>Definition:</u> The subject of an email. <u>Type</u> string <u>Allowed Values:</u> A single line of text.
Body	<u>Definition:</u> Specifies the body portion of an email. <u>Type</u> string <u>Allowed Values:</u> Free text.

Output

None.

Example

```
//Sample for using Workflow.SendMail
IWFFWorkflowService svc = GetWorkflowService();

try
{
    // email recipients
    string To = "bill@tusca.com";

    // CC
    string CC = "bob@tusca.com";

    //Subject of the Mail
    string Subject = "This is email Subject";

    //Body of the Mail
    string Body = "This email Body";
    svc.SendMail(To, CC, Subject, Body);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Send Mail (Extended Method)

API Type

Web Services

Description

Sends an email through AgilePoint Server. The extended method enables you to send attachments.

Syntax

```
public virtual void SendMailEx(String From, String To, String CC, String Subject, String Body , String Attachments)
```

Parameters

Name	Description
From	<u>Definition:</u> Specifies the From portion of an email. <u>Type</u> string <u>Allowed Values:</u> A string that includes one or more email addresses in SMTP format.
To	<u>Definition:</u> Specifies the To portion of an email. <u>Type</u> string <u>Allowed Values:</u> A string that includes one or more email addresses in SMTP format.
CC	<u>Definition:</u> Specifies the CC portion of an email. <u>Type</u> string <u>Allowed Values:</u>

Name	Description
	A string that includes one or more email addresses in SMTP format.
Subject	<u>Definition:</u> The subject of an email. <u>Type</u> string <u>Allowed Values:</u> A single line of text.
Attachments	<u>Definition:</u> File attachments included with the email. <u>Type</u> string <u>Allowed Values:</u> A valid path and file name on the AgilePoint Server. This parameter must use a file path from the file system (for example, C:\file.txt) on the machine where AgilePoint Server is installed. If there is no attachment, you can pass null or String.Empty.

Output

None.

Example

```
//Sample for using Workflow.SendMailEx
IWFWorkflowService svc = GetWorkflowService();

try
{
    string From = "john@tusca.com"; // or <Full Name>"email address"
    string To = "bill@tusca.com";
    string CC = "bob@tusca.com";
    string Subject = "Mail Subject";
    string Body = "Mail Body";
    string Attachments = "c:\\Temp\\Tempdoc.doc";

    //Send Mail
    svc.SendMailEx(From, To, CC, Subject, Body, Attachments);
}

catch (Exception ex)
{
}
```

```
Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));  
}
```

Supported Versions

4.6 and higher

Send Mail (Extended Method with Priority)

API Type

Web Services

Description

Sends an email through AgilePoint Server. The extended method enables you to send attachments.

Syntax

```
public virtual void SendMailEx(String From, String To, String CC, String Subject, String Body , String  
Attachments, Enum priority)
```

Parameters

Name	Description
From	<u>Definition:</u> Specifies the From portion of an email. <u>Type</u> string <u>Allowed Values:</u> A string that includes one or more email addresses in SMTP format.
To	<u>Definition:</u> Specifies the To portion of an email. <u>Type</u> string <u>Allowed Values:</u> A string that includes one or more email addresses in SMTP format.
CC	<u>Definition:</u> Specifies the CC portion of an email.

Name	Description
	<u>Type</u> string <u>Allowed Values:</u> A string that includes one or more email addresses in SMTP format.
Subject	<u>Definition:</u> The subject of an email. <u>Type</u> string <u>Allowed Values:</u> A single line of text.
Attachments	<u>Definition:</u> File attachments included with the email. <u>Type</u> string <u>Allowed Values:</u> A valid path and file name on the AgilePoint Server. This parameter must use a file path from the file system (for example, C:\file.txt) on the machine where AgilePoint Server is installed. If there is no attachment, you can pass null or String.Empty.
priority	<u>Definition:</u> The email's priority. <u>Type</u> enum <u>Allowed Values:</u> • high • normal • low

Output

None.

Example

```
//Sample for using Workflow.SendMailEx
IWFWWorkflowService svc = GetWorkflowService();

try
{
    string From = "john@tusca.com"; // or <Full Name>"email address"
    string To = "bill@tusca.com";
    string CC = "bob@tusca.com";
    string Subject = "Mail Subject";
    string Body = "Mail Body";
    string Attachments = "c:\\Temp\\Tempdoc.doc";
    enum priority = "high";

    //Send Mail
    svc.SendMailEx(From, To, CC, Subject, Body, Attachments);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

5.0 and higher

Methods for Custom Attributes

This section describes service calls related to custom attributes.

Get Custom Attribute

API Type

Web Services

Description

Retrieves a single custom attribute.

Syntax

public virtual object GetCustomAttr(string customID, string attrName);

Parameters

Name	Description
customID	<u>Definition:</u>

Name	Description
	A work object ID specified within a process instance. <u>Type</u> string <u>Allowed Values:</u> One valid work object ID.
attrName	<u>Definition:</u> The name of an item, such as a property or attribute in a name/value pair. <u>Type</u> string <u>Allowed Values:</u> A valid name.

Output

Custom attribute value (can be string, integer, float, double, bool, and/or DateTime).

Example

```
IWFWorkflowService svc = GetWorkflowService();
string customID = ...// for example, "013933F128C3415F81D6F545594D4CB6";
string attrName = ...// for example, "/pd:myFields/pd:Name" or "Approval"

try
{
    Object obj = svc.GetCustomAttr(customID, attrName);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Custom Attributes

API Type

Web Services

Description

Retrieves a collection of custom attributes for a specified custom ID in XML format.

Syntax

```
public virtual string GetCustomAttrs(string customID)
```

Parameters

Name	Description
customID	<u>Definition:</u> A work object ID specified within a process instance. <u>Type</u> string <u>Allowed Values:</u> One valid work object ID.

Output

string that contains all the attributes for the custom ID in XML format.

Example

```
// get custom attributes in xml format
IWfWorkflowService svc = GetWorkflowService();
string customID = ...// for example, "013933F128C3415F81D6F545594D4CB6";

try
{
    string xml = svc.GetCustomAttrs(sessionID, string processInstanceID);
    Console.WriteLine("AttributeXMLstring={0}", resultAttrXML);
    WfCustomAttributes attrs = new WfCustomAttributes();
    attrs.AttrXml = xml; // de-serialize xml
    string[] attributeNames = attrs.GetNames();// get attribute names
    Object value = attrs["MyAttributeName"]; // retrieve attribute value
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Custom Attributes (Extended Method)

API Type

Web Services

Description

Retrieves all custom attributes for a specified a set of custom IDs.

Syntax

```
public virtual KeyValue[] GetCustomAttrsEx(string[] customIDs)
```

Parameters

Name	Description
customIDs	<u>Definition:</u> Multiple work object IDs specified within a process instance. <u>Type</u> string <u>Allowed Values:</u> An array of valid work object IDs.

Output

An array of KeyValue values. The key is a custom ID, and value is a string of custom attributes in XML format.

Example

```
IWFWorkflowService svc = GetWorkflowService();

//Array of custom ID
string[] customIDs = ...// for example,
{"InfoPath:011eaf6c46ac4723b25b4db5772d9912", ...};

try
{
    KeyValue[] keyValues = svc.GetCustomAttrsEx(customIDs);
    foreach (KeyValue kv in keyValues)
    {
        Console.WriteLine("ID:'{0}'", kv.Key);
        Console.WriteLine("Custom Attributes XML: '{0}'", kv.Value);
        WFCustomAttributes attrs = new WFCustomAttributes();
        Attrs.AttrXml = kv.Value;
    }
}
```

```
}  
catch (Exception ex)  
{  
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));  
}
```

Supported Versions

3.2.0.4 and higher

Remove Custom Attribute

API Type

Web Services

Description

Removes a custom attribute from a custom ID.

Syntax

```
public virtual void RemoveCustomAttr(string customID, string attributeName)
```

Parameters

Name	Description
customID	<u>Definition:</u> A work object ID specified within a process instance. <u>Type</u> string <u>Allowed Values:</u> One valid work object ID.
attributeName	<u>Definition:</u> The name of the process instance attribute you want. <u>Type</u> string <u>Allowed Values:</u> A valid process instance attribute.

Output

None.

Example

```
//Sample for using Workflow.RemoveCustomAttr
IWFFrameworkService svc = GetWorkflowService();
string customID = ..// for example, "InfoPath:011eaf6c46ac4723b25b4db5772d9912"
string attributeName = ...// for example, "//pd:purchaseOrder/pd:secondApproval"

try
{
    svc.RemoveCustomAttr(customID, attributeName);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Remove Custom Attributes

API Type

Web Services

Description

Removes multiple custom attributes from a custom ID.

Syntax

```
public virtual void RemoveCustomAttrs(string customID, string[] namesArray)
```

Parameters

Name	Description
customID	<u>Definition:</u> A work object ID specified within a process instance. <u>Type</u> string <u>Allowed Values:</u>

Name	Description
	One valid work object ID.
namesArray	<u>Definition:</u> The the names of one or more items, such as a properties or attributes. <u>Type</u> string <u>Allowed Values:</u> An array of valid names.

Output

None.

Example

```
//Sample for using Workflow.RemoveCustomAttr
IWFFrameworkService svc = GetWorkflowService();
string customID = ...// for example,
"InfoPath:011eaf6c46ac4723b25b4db5772d9912"

// array of attribute name, for example,
// new string[] { // "//pd:purchaseOrder/pd:secondApproval", ... };
string[] namesArray = ...

try
{
    svc.RemoveCustomAttrs(customID, namesArray);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Set Custom Attribute

API Type

Web Services

Description

Sets the name and value for a custom attribute for a specified custom ID.

Syntax

```
public virtual SetCustomAttr(String customID, String attributeName, object val)
```

Parameters

Name	Description
customID	<u>Definition:</u> A work object ID specified within a process instance. <u>Type</u> string <u>Allowed Values:</u> One valid work object ID.
attributeName	<u>Definition:</u> The name of the process instance attribute you want. <u>Type</u> string <u>Allowed Values:</u> A valid process instance attribute.
val	<u>Definition:</u> The value of the custom attribute. <u>Type</u> object <u>Allowed Values:</u> A valid Types object.

Output

None.

Example

```
IWFWorkflowService svc = GetWorkflowService();  
string customID = ..// for example,  
"InfoPath:01leaf6c46ac4723b25b4db5772d9912"  
string attributeName = ...// for example,  
"//pd:purchaseOrder/pd:secondApproval"
```

```
bool val = true;

try
{
    svc.SetCustomAttr(customID, attributeName, val);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Set Custom Attributes

API Type

Web Services

Description

Sets names and values for multiple custom attributes for a specified custom ID.

Syntax

```
public virtual void SetCustomAttrs(string customID, NameValue[] nameValues)
```

Parameters

Name	Description
customID	<u>Definition:</u> A work object ID specified within a process instance. <u>Type</u> string <u>Allowed Values:</u> One valid work object ID.
attributes	<u>Definition:</u> Name-value pairs associated with a custom ID. <u>Type</u> NameValue <u>Allowed Values:</u>

Name	Description
	A valid custom ID with an associated name.

Output

None.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string customID = ..// for example, "InfoPath:011eaf6c46ac4723b25b4db5772d9912"

NameValue[] attributes = new NameValue[]
{
    new NameValue("CustomAttributeName1", "CustomAttributevalue1"),
    new NameValue("CustomAttributeName2", false),
    new NameValue("CustomAttributeName3", 10.0)
};

try
{
    svc.SetCustomAttrs(customID, attributes);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Methods for Archiving and Restoring Processes

This section describes service calls related to archiving and restoration of processes.

Archive Process Instance

API Type

Web Services

Description

Archives a process instance based on a specified process instance identifier by moving the set of process instance records from the current AgilePoint Database into the AgilePoint Archive Database. The process instance records and all of the associated data are then deleted from the AgilePoint Database. The process instance to be archived must be completed or canceled.

Syntax

```
public virtual void ArchiveProcInst(string processInstanceId)
```

Parameters

Name	Description
processInstanceId	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID

Output

None.

Example

```
// this is console application sample
IWFWorkflowService svc = GetWorkflowService();
string processInstanceId = ..// the ID of process instance to be
    archived

try
{
    svc.ArchiveProcInst(processInstanceId);
}

catch (Exception ex)
{
    Console.WriteLine("Failed!, {0}", ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Restore Process Instance

API Type

Web Services

Description

Restores a process instance and associated data from the ArchiveDatabase to the AgilePoint Server. The process instance records are written to the AgilePoint Database deleted from the AgilePoint Archive Database.

Syntax

```
public virtual void RestoreProcInst(string processInstanceId)
```

Parameters

Name	Description
processInstanceId	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID

Output

None.

Example

```
// this is console application sample code
IWfWorkflowService svc = GetWorkflowService();
string processInstanceId = ..// the ID of process instance to be
restored.

try
{
    svc.RestoreProcInst(processInstanceId);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex) );
}
```

Supported Versions

3.2.0.4 and higher

Query Archived Process Instances

API Type

Web Services

Description

Retrieves process instances that match a SQL query.

Syntax

```
public virtual WFBaseProcessInstance[] QueryArchivedProcInsts(string sqlWhereClause)
```

Parameters

Name	Description
sqlWhereClause	<u>Definition:</u> The where clause of the SQL statement you want to query. <u>Type</u> string <u>Allowed Values:</u> The where clause of a SQL statement.

Output

An array of WFBaseProcessInstance objects.

Example

```
// Build SQL Statement
string processDefinitionName = "Budget Request Approval Process";
string sqlWhereClause = string.Format("DEF_NAME = '{0}'",
processDefinitionName);

try
{
    WFBaseProcessInstance[] archivedProcessInstances =
        svc.QueryArchivedProcInsts(sqlWhereClause)
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Group, Role, and Rights

This section describes service calls related to the groups, roles, and rights.

Add Group

API Type

Web Services

Description

Adds a group to the AgilePoint system.

Syntax

```
public virtual WFGGroup AddGroup(string GroupName, string Description, string ResponsibleUser, bool Enabled)
```

Parameters

Name	Description
GroupName	<u>Definition:</u> Specifies the name of a group. <u>Type</u> string <u>Allowed Values:</u> A valid group name.
Description	<u>Definition:</u> A free text description of an entity. <u>Type</u> string <u>Allowed Values:</u> A string that can contain spaces and special characters.
ResponsibleUser	<u>Definition:</u>

Name	Description
	The user name for the responsible user of this group. The responsible user must be a registered AgilePoint user. <u>Type</u> string <u>Allowed Values:</u> A valid user name.
Enabled	<u>Definition:</u> Enables or disables an entity. <u>Type</u> bool <u>Allowed Values:</u> • True - Enables the entity. • False - Disables the entity.

Output

WFGroup object represented the group that is added.

Example

```
IWFAdminService svc = GetAdminService();
string GroupName = ...;
string Description = ...;
string ResponsibleUser = @"[Domain Name]\[Account Name]", // Group
Lead User Name
bool Enabled = true;

try
{
    WFGroup group = svc.AddGroup(GroupName, Description,
    ResponsibleUser, Enabled);
}

catch( Exception ex )
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Add Group Member

API Type

Web Services

Description

Adds a user as a member of a group.

Syntax

```
public virtual WFGroupMember AddGroupMember(string GroupName, string UserName, string Description, string ClientData, bool Enabled)
```

Parameters

Name	Description
GroupName	<u>Definition:</u> Specifies the name of a group. <u>Type</u> string <u>Allowed Values:</u> A valid group name.
UserName	<u>Definition:</u> The user name for the user. <u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.
Description	<u>Definition:</u> A free text description of an entity. <u>Type</u> string <u>Allowed Values:</u> A string that can contain spaces and special characters.

Name	Description
ClientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u> string <u>Allowed Values:</u> A string that contains the client data. If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.
Enabled	<u>Definition:</u> Enables or disables an entity. <u>Type</u> bool <u>Allowed Values:</u> • True - Enables the entity. • False - Disables the entity.

Output

WFGroupMember object that contains the data for the new group member.

Example

```
IWFAdminService svc = GetAdminService();
string GroupName = ...;
string Description = ...;
string UserName = @"[Domain Name]\[Account Name]", // Group Lead User Name
string ClientData = null;
bool Enabled = true;

try
{
    WFGroup group = svc.AddGroupMember(GroupName, UserName, Description,
    ClientData, Enabled);
}

catch( Exception ex )
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Add Role

API Type

Web Services

Description

Adds a role to the AgilePoint system.

Syntax

```
public virtual WFRole AddRole(String RoleName, String Description, int[] Rights, bool Enabled)
```

Parameters

Name	Description
RoleName	<u>Definition:</u> The name of a role. <u>Type</u> string <u>Allowed Values:</u> A valid role name.
Description	<u>Definition:</u> A free text description of an entity. <u>Type</u> string <u>Allowed Values:</u> A string that can contain spaces and special characters.
Rights	<u>Definition:</u> Specifies the rights that are assigned to the Role. <u>Type</u> WFAccessRights

Name	Description
	<u>Allowed Values:</u> An array of index values. See the table for appropriate indexes. WFAccessRights provides the enums for rights. See the sample code for more information.
Enabled	<u>Definition:</u> Enables or disables an entity. <u>Type</u> bool <u>Allowed Values:</u> • True - Enables the entity. • False - Disables the entity.

Access Rights

Index	Rights
0	Register and modify the user information
1	Unregister a user
2	Add and modify role information
3	Remove a role
4	Add and modify group information
5	Remove a group
6	Modify and view system information
7	Add a process definition
8	Check in and check out a process definition
9	Delete or disable a process definition
10	Release a process definition
11	Initiate a process
12	Suspend and resume a process
13	Resend and cancel an email notification
14	Cancel a process
15	Rollback a process
16	Reassign a task

Index	Rights
17	Cancel a task
18	Create a task
19	Add, remove and modify delegation
20	Add, remove and modify report configuration
21	Achieve and restore processes
22	Add, remove and modify shared custom attributes
23	View process details

Output

WFRole object for the role that is added.

Example

```
IWFAdminService svc = GetAdminService();
string RoleName = ...// for example, "Process Manager";
string Description = ...;
/*integer array specifying the access rights for this role.
 * 11 - Initiate a process
 * 14 - Cancel a process
 * 23 - view process details
 */
int[] Rights =
{
    WFAccessRights.InitiateProcessInstance,
    WFAccessRights.CancelProcessInstance,
    WFAccessRights.ViewProcessDetails
};

try
{
    WFRole role = svc.AddRole(RoleName, Description, Rights, True);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Add Role Member

API Type

Web Services

Description

Adds a user or a group to a role.

Syntax

```
public virtual WFRoleMember AddRoleMember(string RoleName, string Assignee, string AssigneeType, string ClientData, string ObjectID, string ObjectType);
```

Parameters

Name	Description
RoleName	<u>Definition:</u> The name of a role. <u>Type</u> string <u>Allowed Values:</u> A valid role name.
Assignee	<u>Definition:</u> The name of the assignee. <u>Type</u> string <u>Allowed Values:</u> A valid user name or group name, depending upon the assignee type.
AssigneeType	<u>Definition:</u> The type for the assignee for the task. <u>Type</u> string <u>Allowed Values:</u> • User - Assigns the task to a user. • Group - Assigns the task to a group.
ClientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u>

Name	Description
	string <u>Allowed Values:</u> A string that contains the client data. If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.
ObjectID	<u>Definition:</u> Reserved for future use. <u>Type</u> string <u>Allowed Values:</u> A null value.
ObjectType	<u>Definition:</u> Reserved for future use. <u>Type</u> string <u>Allowed Values:</u> A null value.

Output

WFRoleMember object.

Example

```
IWFAdminService svc = GetAdminService();

string RoleName = "Administrative";
string Assignee = @"Demo3\Administrator";
string AssigneeType = "User";

try
{
    WFRoleMember member = svc.AddRoleMember(RoleName, Assignee,
        AssigneeType, "", null, null);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Enabled Group Member

API Type

Web Services

Description

Enables or disables a user as a member of a group.

Syntax

```
public virtual WFGroupMember EnabledGroupMember(string GroupName, string userName, bool Enabled)
```

Parameters

Name	Description
GroupName	<u>Definition:</u> Specifies the name of a group. <u>Type</u> string <u>Allowed Values:</u> A valid group name.
userName	<u>Definition:</u> The user name for the user. <u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.
Enabled	<u>Definition:</u> Enables or disables an entity. <u>Type</u> bool <u>Allowed Values:</u>

Name	Description
	• True - Enables the entity. • False - Disables the entity.

Output

WFGGroupMember object.

Example

```
IWFAdminService svc = GetAdminService();

string GroupName = "Administrative";
string userName = @"Demo3\Administrator";
bool Enabled = false;

try
{
    svc.EnabledGroupMember(GroupName, userName, Enabled);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Access Right Names

API Type

Web Services

Description

Retrieves the names of all the access rights in the AgilePoint system.

Syntax

```
public virtual string[] GetAccessRightNames()
```

Parameters

Name	Description
None	Not Applicable

Output

An array of strings that contain the names of all the access rights for the system.

Example

```
IWFAdminService svc = GetAdminService();

try
{
    string[] permissionNames = svc.GetAccessRightNames();
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Access Rights

API Type

Web Services

Description

Retrieves the access rights for a specified user.

Syntax

```
public virtual int[] GetAccessRights(string userName)
```

Parameters

Name	Description
userName	<u>Definition:</u> The user name for the user. <u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.

Output

An array of integers that represent the access rights of the user.

Example

```
IWFAdminService svc = GetAdminService();
string userName = ...// for example, "Demo3\\ap_svc"

try
{
    string[] accessNames = svc.GetAccessRightNames();
    int[] userRights = svc.GetAccessRights(userName);
    Console.WriteLine("The user has the following rights:");
    foreach (int rightCode in userRights)
    {
        Console.WriteLine(accessNames[rightCode]);
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/*
This example produces the following results:
The user has the following rights:
Register and modify the user information
Unregister a user
Add and modify the role information
Remove a role
Add and modify the group information
Remove a group
Modify/View the system configuration
Add a process definition
Checkin and checkout a process definition
Delete and disable a process definition
...
*/
```

Supported Versions

3.2.0.4 and higher

Get Group

API Type

Web Services

Description

Retrieves a group object with the specified group name.

Syntax

```
public virtual WFGGroup GetGroup(string groupName)
```

Parameters

Name	Description
groupName	<u>Definition:</u> Specifies the name of a group. <u>Type</u> string <u>Allowed Values:</u> A valid group name.

Output

WFGGroup object. If the specified group does not exist, the output is null.

Example

```
IWFAdminService svc = GetAdminService();
string groupName = ...// for example, "Administrators";

try
{
    WFGGroup grpInfo = svc.GetGroup(groupName);
    Console.WriteLine("Group Name:{0}; Group Lead:{1}", grpInfo.Name,
        grpInfo.ResponsibleUser);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/*
This example produces the following results:
Group Name:Administrators; Group Lead:\Administrator
*/
```

Supported Versions

3.2.0.4 and higher

Get Group Members

API Type

Web Services

Description

Retrieves the members of a specified group.

Syntax

```
public virtual WFGroupMember[] GetGroupMembers(string groupName)
```

Parameters

Name	Description
groupName	<u>Definition:</u> Specifies the name of a group. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid group name.

Output

An array of WFGroupMember objects.

Example

```
IWFAdminService svc = GetAdminService();
string groupName = ...// for example, "Administrators";

try
{
    WFGroupMember[] grpMembers = adminService.GetGroupMembers(groupName);
    Console.WriteLine("AgilePoint Group {0} has {1} members:",
        groupName, grpMembers.Length);
    foreach (WFGroupMember grpMember in grpMembers)
    {
        Console.WriteLine("Member Name:{0}", grpMember.Member);
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

```
}

/*
This example produces the following results:
AgilePoint Group Administrators has 3 members:
Member Name:\Administrator
Member Name:VITBDC\yuvarajn
Member Name:Demo3\manager
*/
```

Supported Versions

3.2.0.4 and higher

Get Groups

API Type

Web Services

Description

Retrieves all the group objects in the system.

Syntax

```
public virtual WFGGroup[] GetGroups()
```

Parameters

Name	Description
None	Not Applicable

Output

An Array of WFGGroup objects.

Example

```
IWFAdminService svc = GetAdminService();

try
{
    WFGGroup[] apGroups = svc.GetGroups();
    Console.WriteLine("AgilePoint Group {0} has {1} members:",
        groupName, grpMembers.Length);
    Console.WriteLine("AgilePoint Groups:");
    foreach(WFGGroup grp in apGroups)
    {
        System.Console.WriteLine("Name:{0};Group Lead:{1} ", grp.Name,
            grp.ResponsibleUser);
    }
}
```

```
catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/*
This example produces the following results:
AgilePoint Groups:
Name:Administrators;Group Lead:\Administrator
Name:Domain Users;Group Lead:VITBDC\amarnv
*/
```

Supported Versions

3.2.0.4 and higher

Get Role

API Type

Web Services

Description

Retrieves a role object by name.

Syntax

```
public virtual WFRole GetRole(string roleName);
```

Parameters

Name	Description
roleName	<u>Definition:</u> The name of a role. <u>Type</u> string <u>Allowed Values:</u> A valid role name.

Output

WFRole object with the specified role name.

Example

```
IWFAdminService svc = GetAdminService();
```

```
string rolName = ...// for example, "Administrators"

try
{
    WFRole role = adminService.GetRole(rolName);
    Console.WriteLine("Name = '" + role.Name + "' Description = '" +
        role.Description + "'");
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/* This example produces the following results:
Name = 'Administrators' Description = 'Administrators have complete access to
maintain system' */
```

Supported Versions

3.2.0.4 and higher

Get Roles

API Type

Web Services

Description

Retrieves a list of all roles in the system.

Syntax

```
public virtual WFRole[] GetRoles()
```

Parameters

Name	Description
None	Not Applicable

Output

An array of WFRole objects that includes all roles.

Example

```
IWFAdminService svc = GetAdminService();

try
{
    WFRole[] roles = adminService.GetRoles();
    foreach (WFRole role in roles)
```



```

        {
            Console.WriteLine("Name = '{0}', Description = '{1}'", role.Name,
                               role.Description);
        }
    }

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/* This example produces the following results:
Name = 'Administrators' Description = 'Administrators have complete
access to maintain system'
Name = 'Process Runtime Managers' Description = 'Process Runtime
Managers have complete access to manage runtime processes'
Name = 'process definition Designers' Description = 'process
definition Designers have complete access to add, modify and remove process
templates' */

```

Supported Versions

3.2.0.4 and higher

Query Role Members

API Type

Web Services

Description

Retrieves the members assigned to a role that match a specified SQL statement.

Syntax

```
public virtual WFRoleMember[] QueryRoleMembers(string roleName, string sqlWhereClause)
```

Parameters

Name	Description
roleName	<u>Definition:</u> The name of a role. <u>Type</u> string <u>Allowed Values:</u> A valid role name.
sqlWhereClause	<u>Definition:</u>

Name	Description
	The where clause of the SQL statement you want to query. <u>Type</u> string <u>Allowed Values:</u> The where clause of a SQL statement.

Output

Returns array of WFRoleMember members of the role that match the specified SQL statement.

Example

```
IWFAdminService svc = GetAdminService();
string roleName = ...// for examples, "Users"
string sqlWhereClause = ...// for example, "ASSIGNEE_TYPE='User'"

try
{
    WFRoleMember[] roleMembers = svc.QueryRoleMembers(roleName, sqlWhereClause);
    foreach (WFRoleMember member in roleMembers)
    {
        Console.WriteLine("Assignee = '{0}', Created Date = '{1}'",
            member.Assignee,
            member.CreatedDate.ToShortDateString());
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/*
This example produces the following results:
Assignee = 'DEMO\\CONTROLLER' & Created Date = '9/18/2009'
Assignee = 'DEMO\\CFO' & Created Date = '9/18/2009'
*/
```

Supported Versions

3.2.0.4 and higher

Remove Group

API Type

Web Services

Description

Removes a group from the AgilePoint system.

Syntax

```
public virtual void RemoveGroup(string groupName)
```

Parameters

Name	Description
groupName	<u>Definition:</u> Specifies the name of a group. <u>Type</u> string <u>Allowed Values:</u> A valid group name.

Output

None.

Example

```
IWFAdminService svc = GetAdminService();
string groupName = ...// for examples, "Users"

try
{
    svc.RemoveGroup(groupName);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Remove Group Member

API Type

Web Services

Description

Removes a member from a group.

Syntax

```
public virtual void RemoveGroupMember(string GroupName, string UserName)
```

Parameters

Name	Description
GroupName	<u>Definition:</u> Specifies the name of a group. <u>Type</u> string <u>Allowed Values:</u> A valid group name.
UserName	<u>Definition:</u> The user name for the user. <u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.

Output

None.

Example

```
IWFAdminService svc = GetAdminService();
string GroupName = ...// for examples, "Engineers"
string UserName = ...// for example, @"VIT\ct-002"

try
{
    svc.RemoveGroupMember(GroupName, UserName);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Remove Role

API Type

Web Services

Description

Removes a role from the AgilePoint system.

Syntax

```
public virtual void RemoveRole(string roleName)
```

Parameters

Name	Description
roleName	<u>Definition:</u> The name of a role. <u>Type</u> string <u>Allowed Values:</u> A valid role name.

Output

None.

Example

```
IWFAdminService svc = GetAdminService();
string roleName = ...// for example, "Engineers"

try
{
    svc.RemoveRole (roleName);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Remove Role Member

API Type

Web Services

Description

Removes a user or a group from a specified role.

Syntax

```
public virtual void RemoveRoleMember(string RoleName, string Assignee, string AssigneeType, string ObjectID)
```

Parameters

Name	Description
RoleName	<u>Definition:</u> The name of a role. <u>Type</u> string <u>Allowed Values:</u> A valid role name.
Assignee	<u>Definition:</u> The name of the assignee. <u>Type</u> string <u>Allowed Values:</u> A valid user name or group name, depending upon the assignee type.
AssigneeType	<u>Definition:</u> The type for the assignee for the task. <u>Type</u> string

Name	Description
	<u>Allowed Values:</u> • User - Assigns the task to a user. • Group - Assigns the task to a group.
ObjectID	<u>Definition:</u> Reserved for future use. <u>Type</u> string <u>Allowed Values:</u> A null value.

Output

None.

Example

```
IWFAdminService svc = GetAdminService();
string RoleName = ...// for example, "Engineers"
string Assignee = ...// for example, @"VIT\ct-002"

try
{
    svc.RemoveRoleMember(RoleName, Assignee, "User", null);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Update Group

API Type

Web Services

Description

Updates information for a group.

Syntax

WFGroup UpdateGroup(string GroupName, string Description, string ResponsibleUser, bool Enabled)

Parameters

Name	Description
GroupName	<u>Definition:</u> Specifies the name of a group. <u>Type</u> string <u>Allowed Values:</u> A valid group name.
Description	<u>Definition:</u> A free text description of an entity. <u>Type</u> string <u>Allowed Values:</u> A string that can contain spaces and special characters.
ResponsibleUser	<u>Definition:</u> The user name for the responsible user of this group. The responsible user must be a registered AgilePoint user. <u>Type</u> string <u>Allowed Values:</u> A valid user name.
Enabled	<u>Definition:</u> Enables or disables an entity. <u>Type</u> bool <u>Allowed Values:</u>

Name	Description
	• True - Enables the entity. • False - Disables the entity.

Output

An updated WFGroup.

Example

```
IWFAdminService svc = GetAdminService();
string GroupName = ...// for example, "TestGroup1"
string Description = ... // for example, "This is new description of
the user group"
string ResponsibleUser = ...// for example, "DEMO3\\Administrator"

try
{
    WFGroup updatedGroup = Svc.UpdateGroup(GroupName, Description,
    ResponsibleUser, true);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Update Role

API Type

Web Services

Description

Updates information for a role.

Syntax

```
public virtual WFRole UpdateRole(string RoleName, string Description, int[] Rights, bool Enabled)
```

Parameters

Name	Description
RoleName	<u>Definition:</u> The name of a role.

Name	Description
	<u>Type</u> string <u>Allowed Values:</u> A valid role name.
Description	<u>Definition:</u> A free text description of an entity. <u>Type</u> string <u>Allowed Values:</u> A string that can contain spaces and special characters.
Rights	<u>Definition:</u> Specifies the rights that are assigned to the Role. <u>Type</u> WFAccessRights <u>Allowed Values:</u> An array of index values. See the table for appropriate indexes. WFAccessRights provides the enums for rights. See the sample code for more information.
Enabled	<u>Definition:</u> Enables or disables an entity. <u>Type</u> bool <u>Allowed Values:</u> True - Enables the entity. False - Disables the entity.

Output

An updated WFRole object.

Example

```
IWFAdminService svc = GetAdminService();
string RoleName = ...// for example, "TestRole"
```

```
string Description = ...// for example, "This is new description for TestRole"
List<int> list = new List<int>();
list.Add(WFAccessRights.AddModifyGroup);
list.Add(WFAccessRights.AddModifyRole);
list.Add(WFAccessRights.AddModifyUser);
list.Add(WFAccessRights.AddProcessTemplate);
list.Add(WFAccessRights.AddRemoveModifyDelegation);
int[] Rights = list.ToArray();

try
{
    WFRole updatedRole = svc.UpdateRole(RoleName, Description, Rights, true);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Organization Properties

This section describes service calls related to various organization properties.

Get Organization Properties

API Type

Web Services

Description

Retrieves organization properties such as Title, Department and Location.

Syntax

```
public virtual KeyValueType[] GetOrganizationProperties(string Name)
```

Parameters

Name	Description
Name	<u>Definition:</u> The name of an item, such as a property or attribute in a name/value pair. <u>Type</u> string

Name	Description
	<u>Allowed Values:</u> A valid name.

Output

An array of KeyValue objects.

Example

```
IWFAdminService svc = GetAdminService();
string name = ...// for example, "Departments"

try
{
    KeyValue[] organizationProperties =
        svc.GetOrganizationProperties(Name);
    foreach (KeyValue property in organizationProperties)
    {
        Console.WriteLine("Property Name = 'Department' Property Value = "
            + property.Value + "");
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Remove Organization Properties

API Type

Web Services

Description

Removes an organization property from AgilePoint.

Syntax

```
public virtual void RemoveOrganizationProperties(string Name)
```

Parameters

Name	Description
Name	<u>Definition:</u> The name of an item, such as a property or attribute in a name/value pair. <u>Type</u> string <u>Allowed Values:</u> A valid name.

Output

None.

Example

```
IWFAdminService svc = GetAdminService();
string Name = ...// for example, "Departments"

try
{
    svc.RemoveOrganizationProperties (Name);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Update Organization Properties

API Type

Web Services

Description

Updates organization properties in the AgilePoint the system.

Syntax

```
void UpdateOrganizationProperties(string Name, KeyValue[] list)
```

Parameters

Name	Description
Name	<u>Definition:</u> The name of an item, such as a property or attribute in a name/value pair. <u>Type</u> string <u>Allowed Values:</u> A valid name.
list	<u>Definition:</u> A list of properties. <u>Type</u> KeyValue <u>Allowed Values:</u> A list of key-value pairs.

Output

None.

Example

```
IWFAdminService svc = GetAdminService();
string Name = ...// for example, "Titles"
List<KeyValue> list = new List<KeyValue>();
list.Add( new KeyValue( "MANAGER", "Manager" ) );
list.Add( new KeyValue( "REG_SALES_DIRECTOR", "Regional Sales
Director" ) );
list.Add( new KeyValue( "GENERAL_MANAGER", "General Manager" ) );
KeyValue[] properties = list.ToArray();

try
{
    adm.UpdateOrganizationProperties("Titles",properties );
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Other Web Services

This section describes other additional web services.

Query Audit Trail

API Type

Web Services

Description

Retrieves all audit trail items.

Syntax

```
public virtual WFAuditTrailItem[] QueryAuditTrail(string where)
```

Parameters

Name	Description
where	<u>Definition:</u> The where clause of the SQL statement you want to query. <u>Type</u> string <u>Allowed Values:</u> The where clause of a SQL statement.

Output

An array list of WFAuditTrailItem objects.

Example

```
// This is console application sample
IWfWorkflowService svc = GetWorkflowService();

// Checking where condition based on the CATEGORY COLUMN and PURPOSE
COLUMN .
string where = "CATEGORY = 0 AND PURPOSE='Check-in process"
```

```
definition'';  
  
try  
{  
    // calling the QueryAuditTrail web method it return a array of Dataset.  
    WFAuditTrailItem[] result = svc.QueryAuditTrail(where);  
    foreach (WFAuditTrailItem item in result)// Iterating through  
    WFAuditTrailItem  
    {  
        //Displaying the result on the console.  
        System.Console.Write("ItemCategory: {0} ,ItemDateOccurred: {1}",  
            item.Category,item.DateOccurred);  
        System.Console.Write("ItemDescription: {0} ,ItemObjectID: {1}",  
            item.Description, item.ObjectID);  
        System.Console.WriteLine("ItemPurpose: {0} ,ItemPerformer: {1}",  
            item.Purpose item.Performer);  
    }  
}  
  
catch (Exception ex)  
{  
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));  
}
```

Supported Versions

4.0.1 and higher

Query Database

API Type

Web Services

Description

Queries the database with any valid sql query and returns the dataset as a string in XML format.

Syntax

```
public virtual string QueryDatabase(string sql)
```

Parameters

Name	Description
sql	<u>Definition:</u> The where clause of the SQL statement you want to query. <u>Type</u> string <u>Allowed Values:</u>

Name	Description
	The where clause of a SQL statement.

Output

An XML string that contains the dataset with the results of the database query.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string sql = "SELECT * FROM WF_AUDIT_TRAILS where CATEGORY = 0 AND
PURPOSE='Check-in process definition'";

try
{
    // calling the QueryDatabase web method and passing the sql query as the
    argument.
    string xml = svc.QueryDatabase(sql);
    Console.WriteLine("{0}", xml); // Displaying the XML string on console.
    System.IO.StringReader sr = new System.IO.StringReader(xml);
    System.Data.DataSet ds = new System.Data.DataSet();
    ds.LoadXml(sr);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Query Database (Extended Method)

API Type

Web Services

Description

Queries the database with any valid SQL query and returns an array of 2 elements in XML string format.

Syntax

```
public virtual string[] QueryDatabaseEx(string sql)
```

Parameters

Name	Description
sql	<u>Definition:</u>

Name	Description
	The where clause of the SQL statement you want to query. <u>Type</u> string <u>Allowed Values:</u> The where clause of a SQL statement.

Output

2 elements in a string array, where string[0] is an XML string with the dataset of the query results and string[1] is an XML string with the schema.

Example

```
IWFWorkflowService svc = GetWorkflowService();
string sql = "SELECT * FROM WF_AUDIT_TRAILS where CATEGORY = 0 AND
PURPOSE='Check-in process definition'";

try
{
    // calling the QueryDatabase web method and passing the sql query as the
    argument.
    string[] xmls = svc.QueryDatabase(sql);
    Console.WriteLine("{0}", xmls[0]); // Displaying the XML string on console.
    Console.WriteLine("{0}", xmls[1]); // Displaying the schema on console.
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Administrative Service

This section describes service calls related to administrative functions. These calls are found under admin.wsdl.

Get All EMail Templates

API Type

Web Services

Description

Retrieves all the global email templates from the server.

Syntax

```
private KeyValue[] GetAllEmailTemplates()
```

Parameters

Name	Description
None	Not Applicable

Output

Returns all the global email templates.

Example

```
private KeyValue[] GetAllEmailTemplates()  
{  
    IWFAdminService api = GetAdminService();  
    return api.GetAllEmailTemplates();  
}
```

Supported Versions

3.2.0.4 and higher

Get Database Information

API Type

Web Services

Description

Retrieves the database information of the current server configuration.

Syntax

```
public virtual DatabaseInfo GetDatabaseInfo()
```

Parameters

Name	Description
None	Not Applicable

Output

A DatabaseInfo object that represents the database information of the system.

Example

```
IWFAdminService svc = GetAdminService();

try
{
    DatabaseInfo dbInfo = svc.GetDatabaseInfo();

    if (dbInfo != null)
    {
        Console.WriteLine("AgilePoint System Database Information:");
        Console.WriteLine("Vendor: {0}", dbInfo.Vendor);
        Console.WriteLine("Provider: {0}", dbInfo.Provider);
        Console.WriteLine("Connection string: {0}", dbInfo.Connstr);
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/*
This example produces the following results:
AgilePoint System Database Information:
Vendor: MSSQLDatabase
Provider:
Connection string: application name=AgilePoint Server;connection
lifetime=5;min pool
size=10;server=Demo3;database=AgilePointDB;trusted_Connection=yes
*/
```

Supported Versions

4.6 and higher

Get Domain Groups

API Type

Web Services

Description

Retrieves all the domain group objects.

Syntax

```
public virtual KeyValueType[] GetDomainGroups(string LDAPPath, string Filter)
```

Parameters

Name	Description
LDAPPath	<u>Definition:</u> The LDAP path to the domain. <u>Type</u> string <u>Allowed Values:</u> A valid LDAP path. If the value is null, the AgilePoint Server machine domain will be used.
Filter	<u>Definition:</u> A filter term for groups that is wildcard-enabled. <u>Type</u> string <u>Allowed Values:</u> A string in the format (name=[my wildcard filter]).

Output

Array of KeyValue objects. If the specified group is not found, the output is null.

Example

```
IWFAdminService svc = GetAdminService();
string LDAPPath = ...// for example, LDAP://ou=Sales,dc=Frabrikam,dc=com
string Filter = "A*";
Filter = string.Format("(name={0})", Filter);

try
{
    KeyValue[] grps = svc.GetDomainGroups(LDAPPath, Filter);
    foreach (KeyValue grp in grps)
    {
        Console.WriteLine("Group Name:{0}; Group Distinct Name:{1};", grp.Key,
            grp.Value);
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Domain Group Members

API Type

Web Services

Description

Retrieves the members of a domain group.

Syntax

```
public virtual DomainUser[] GetDomainGroupMembers(string groupDistinguishedName)
```

Parameters

Name	Description
groupDistinguishedName	<u>Definition:</u> Specifies the name of a group in LDAP format. <u>Type</u> string <u>Allowed Values:</u> A valid a group name in LDAP format.

Output

An array of DomainUser objects.

Example

```
IWFAdminService svc = GetAdminService();
string groupName = "Administrators";
string groupDistinguishedName = "LDAP://" + groupName;

try
{
    DomainUser[] grpUsers = svc.
        GetDomainGroupMembers(groupDistinguishedName);
    foreach (DomainUser usr in grpUsers)
    {
        Console.WriteLine("User Name:{0}; Full Name:{1};", usr.UserName,
            usr.FullName);
    }
}
```

```
}  
catch (Exception ex)  
{  
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));  
}  
/*  
This example produces the following results:  
User Name:Administrator; Full Name:Administrator  
User Name:vitbdc/yuvaraj; Full Name:Yuvaraj Nagarajan  
*/
```

Supported Versions

3.2.0.4 and higher

Get Domain Name

API Type

Web Services

Description

Retrieves the domain name to which AgilePoint Server connects.

Syntax

```
public virtual string GetDomainName()
```

Parameters

Name	Description
None	Not Applicable

Output

A string containing the domain name of the AgilePoint Server machine.

Example

```
IWFAdminService svc = GetAdminService();  
  
try  
{  
    string domainName = svc.GetDomainName();  
    Console.WriteLine("AgilePoint System Domain Name={0}", domainName);  
}  
  
catch (Exception ex)  
{  
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));  
}
```

```
/* Sample of output
AgilePoint System Domain Name=LDAP://dc=Frabrikam,dc=com
*/
```

Supported Versions

4.6 and higher

Get Domain Users

API Type

Web Services

Description

Retrieves all the user information in the domain that AgilePoint Server connects. It could be a local Windows system user, or a domain controller on the network.

Syntax

```
public virtual DomainUser[] GetDomainUsers(string LDAPPath, string Filter)
```

Parameters

Name	Description
LDAPPath	<u>Definition:</u> The LDAP path to the domain. <u>Type</u> string <u>Allowed Values:</u> A valid LDAP path. If the value is null, the AgilePoint Server machine domain will be used.
Filter	<u>Definition:</u> A filter term for groups that is wildcard-enabled. <u>Type</u> string <u>Allowed Values:</u> A string in the format (name=[my wildcard filter]).

Output

Array of DomainUser objects. If the specified group does not exist, the return value is null.

Example

```
IWFAdminService svc = GetAdminService();
string LDAPPath = ...// for example,
LDAP://ou=Sales,dc=Frabrikam,dc=com
string Filter = ""; // All Users

try
{
    DomainUser[] users = svc.GetDomainUsers(LDAPPath, Filter);
    foreach (DomainUser user in users)
    {
        Console.WriteLine("Full Name:{0}; Login Name:{1};", user.FullName,
            user.UserName);
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/*
This example produces the following results:
Full Name:Manager; Login Name:Demo3\Manager;
Full Name:Marcomm; Login Name:Demo3\Marcomm;
Full Name:SharePoint Administrator; Login Name:Demo3\sp_adm;
Full Name:Sujeet Kumar; Login Name:Demo3\sujeetk;
*/
```

Supported Versions

3.2.0.4 and higher

Get EMail Template

API Type

Web Services

Description

Retrieves an email templates with the specified template name from the server.

Syntax

```
private string GetEmailTemplate(string processTemplateID)
```

Parameters

Name	Description
processTemplateID	<u>Definition:</u> The unique identifier for the process definition to be checked out for modification. <u>Type</u> string <u>Allowed Values:</u> A valid process template ID

Output

Returns an email templates.

Example

```
private string GetEmailTemplate(string templateID)
{
    IWFAdminService api = GetAdminService();
    return api.GetEmailTemplate(templateID);
}
```

Supported Versions

3.2.0.4 and higher

Get Locale

API Type

Web Services

Description

Retrieves the default locale for the AgilePoint Server.

Syntax

```
public virtual string GetLocale()
```

Parameters

Name	Description
None	Not Applicable

Output

Locale abbreviation—for example, en-US.

Example

```
IWFAdminService svc = GetAdminService();
string activeDirectoryLdapPath = ...// for example,
LDAP://ou=Sales,dc=Frabrikam,dc=com
string userFilter = ""; // All Users

try
{
    string locale = svc.GetLocale();
    Console.WriteLine("Locale = '{0}'", locale);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/*
This example produces the following results:
Locale = 'en-US
*/
```

Supported Versions

4.6 and higher

Get Register User

API Type

Web Services

Description

Retrieves the user information for the registered user.

Syntax

```
public virtual RegisteredUser GetRegisterUser(string userName)
```

Parameters

Name	Description
userName	<u>Definition:</u> The user name for the user. <u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.

Output

RegisteredUser object.

Example

```
IWFAdminService svc = GetAdminService();

try
{
    RegisteredUser registerUser =
    svc.GetRegisterUser("domain\\wilson.goodman");
    Console.WriteLine("Name = '{0}', Department = '{1}'",
        registerUser.FullName,
        registerUser.Department);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

//This example produces the following results:
//Name = 'CFO' Department = 'Finance'
```

Supported Versions

3.2.0.4 and higher

Get Register Users

API Type

Web Services

Description

Retrieves all registered users.

Syntax

```
public virtual RegisteredUser[] GetRegisterUsers()
```

Parameters

Name	Description
None	Not Applicable

Output

Returns an array of registered users.

Example

```
IWFAdminService svc = GetAdminService();

try
{
    RegisteredUser[] registerUsers = svc.GetRegisterUsers();
    foreach (RegisteredUser regUser in registerUsers)
    {
        Console.WriteLine("Name = '{0}', Department = '{1}'",
            registerUser.FullName,
            registerUser.Department);
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Get Register User Icons

API Type

Web Services

Description

Retrieves an icon for a registered user.

Syntax

```
public virtual byte[] GetRegisteredUserIcon(string userName)
```

Parameters

Name	Description
userName	<u>Definition:</u> The user name for the user. <u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.

Output

Return an array of bytes that contains image data.

Example

```
For future use.
```

Supported Versions

4.6 and higher

Get Sender Email Address

API Type

Web Services

Description

Retrieves the sender email address of the AgilePoint Server.

Syntax

```
public virtual string GetSenderEmailAddress()
```

Parameters

Name	Description
None	Not Applicable

Output

The email address that is configured as the sender email address on the AgilePoint Server.

Example

```
IWFAdminService svc = GetAdminService();

try
{
    string senderEmailAddress = svc.GetSenderEmailAddress();
    Console.WriteLine("Sender EMail Address = '{0}'",
        senderEmailAddress);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

//This example produces the following results:
//Sender EMail Address = 'admin@your-domain.com'
```

Supported Versions

4.6 and higher

Get SMTP Server

API Type

Web Services

Description

Retrieves the SMTP server of the current server configuration.

Syntax

```
public virtual string GetSmtpServer();
```

Parameters

Name	Description
None	Not Applicable

Output

A string that contains the name of the AgilePoint system's SMTP server.

Example

```
IWFAdminService svc = GetAdminService();

try
{
    string smtpServer = svc.GetSmtpServer();
    Console.WriteLine("SMTP Server Name = '" + smtpServer + "'");
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

//This example produces the following results:
//SMTP Server Name = 'smtp.vitinfotech.com'
```

Supported Versions

4.6 and higher

Get System Performance Information

API Type

Web Services

Description

Retrieves system performance information for AgilePoint Server.

Syntax

```
public virtual WFSysPerfInfo GetSysPerfInfo()
```

Parameters

Name	Description
None	Not Applicable

Output

WFSysPerfInfo object.

Example

```
IWFAdminService svc = GetAdminService();

try
{
    WFSysPerfInfo sysPerfInfo = svc.GetSysPerfInfo();
}
```



```
    Console.WriteLine("1) ServerID = '{0}'", sysPerfInfo.ServerID);
    Console.WriteLine("2) MemoryAllocated = '{0}'",
        sysPerfInfo.MemoryAllocated);
    //...
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/*
This example produces the following results:
1) ServerID = 'DEMO/4356'
2) MemoryAllocated = '7329'
*/
```

Supported Versions

3.2.0.4 and higher

Get System User

API Type

Web Services

Description

Retrieves the name of the system user.

Syntax

```
public virtual string GetSystemUser()
```

Parameters

Name	Description
None	Not Applicable

Output

Returns the name of the system user as a string value.

Example

```
IWFAdminService svc = GetAdminService();

try
{
    string systemUser = svc.GetSystemUser();
    Console.WriteLine("AgilePoint System User= '{0}'", systemUser);
}
```

```
catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

//This example produces the following results:
//AgilePoint System User = 'Administrator'
```

Supported Versions

4.6 and higher

Query Register Users

API Type

Web Services

Description

Retrieves the list of registered users on the AgilePoint Server.

Syntax

```
public virtual RegisteredUser[] QueryRegisterUsers(string sqlWhereClause)
```

Parameters

Name	Description
sqlWhereClause	<u>Definition:</u> The where clause of the SQL statement you want to query. <u>Type</u> string <u>Allowed Values:</u> The where clause of a SQL statement.

Output

Returns a list of registered users.

Example

```
IWFAdminService svc = GetAdminService();
string sqlWhereClause = "DEPARTMENT in ('PublicUsers', ...)"

try
```

```
{
    RegisteredUser[] registeredUsers =
    adminService.QueryRegisterUsers(sqlWhereClause);
    foreach (RegisteredUser user in registeredUsers)
    {
        Console.WriteLine("User Name = '{0}', Email = '{1}'",
            user.UserName, user.EmailAddress);
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/*
This example produces the following results:
User Name = 'DEMO\Author' Email = 'author@DEMO.com'
User Name = 'DEMO\Employee' Email = 'employee@DEMO.com'
*/
```

Supported Versions

3.2.0.4 and higher

Register User

API Type

Web Services

Description

Registers a user on the AgilePoint system.

Syntax

```
public virtual void RegisterUser(RegisteredUser user)
```

Parameters

Name	Description
user	<u>Definition:</u> Specifies information about a user. <u>Type</u> RegisteredUser <u>Allowed Values:</u> A valid RegisteredUser object.

Output

None.

Example

```
IWFAdminService svc = GetAdminService();

try
{
    RegisteredUser user= new RegisteredUser();
    user.FullName = "Accountant";
    user.UserName = "DEMO\\Accountant";
    user.Department = "Accounts";
    user.EmailAddress = "accountant@DEMO.com";
    user.Locale = "en-US";
    svc.RegisterUser(user);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Unregister User

API Type

Web Services

Description

Removes a user's registration from the AgilePoint system. Note that this call does not remove the user from the local Windows system or the domain controller.

Syntax

```
void UnregisterUser(string userName)
```

Parameters

Name	Description
userName	<u>Definition:</u> The user name for the user. <u>Type</u> string

Name	Description
	<u>Allowed Values:</u> A valid user name for a registered AgilePoint user.

Output

None.

Example

```
IWFAdminService svc = GetAdminService();
string userName = ...// for example, "[Domain Name]\[User Account Name]"

try
{
    svc.UnregisterUser(userName);
}

catch( Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Update Registered User

API Type

Web Services

Description

Updates user data for a registered user.

Syntax

```
public virtual void UpdateRegisterUser(RegisteredUser user)
```

Parameters

Name	Description
user	<u>Definition:</u> Specifies information about a user. <u>Type</u>

Name	Description
	RegisteredUser <u>Allowed Values:</u> A valid RegisteredUser object.

Output

None.

Example

```
RegisteredUser user = new RegisteredUser();
user.UserName = "DEMO3\\cfo";
user.FullName = "Andy";
user.EmailAddress = "cfo@tusca.com";
user.Title = "CFO";
user.Department = "Marketing";
user.Manager = "DEMO3\\Administrator";
user.Locale = Thread.CurrentThread.CurrentUICulture.Name;
IWFAAdminService svc = GetAdminService();

try
{
    svc.UpdateRegisterUser( user );
}

catch( Exception ex )
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Update Registered User Icon

API Type

Web Services

Description

Updates the icon for a registered user.

Syntax

```
public virtual void UpdateRegisteredUserIcon(string userName, byte[] UserIcon)
```

Parameters

Name	Description
userName	<u>Definition:</u> The user name for the user. <u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.
UserIcon	<u>Definition:</u> A byte array representation of icon image. <u>Type</u> byte <u>Allowed Values:</u> A valid byte array.

Output

None.

Example

None.

Supported Versions

4.6 and higher

Report Configuration Methods

This section describes service calls related to reports.

Add Report Configuration

API Type

Web Services

Description

Adds a report configuration to the system.

Syntax

```
public virtual WfReportConfigure AddReportConfigure(string reportName, string configure)
```

Parameters

Name	Description
reportName	<u>Definition:</u> The name of a report. <u>Type</u> string <u>Allowed Values:</u> A valid report name.
configure	<u>Definition:</u> The report configuration in XML format. <u>Type</u> string <u>Allowed Values:</u> A valid report configuration in XML format.

Output

WfReportConfigure object.

Example

```
IWFAdminService svc = GetAdminService();
string reportName = ...//for example, "weekly task report"
string configure = ...// xml-serialization of WfReportConfiguration

try
{
    WfReportConfigure reportConfig =
        svc.AddReportConfigure(reportName, configure);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```


Supported Versions

3.2.0.4 and higher

Get All Report Configurations

API Type

Web Services

Description

Retrieves all report configurations from the system.

Syntax

```
public virtual WFReportConfigure[] GetAllReportConfigure()
```

Parameters

Name	Description
None	Not Applicable

Output

WFReportConfigure object.

Example

```
IWFAdminService svc = GetAdminService();

try
{
    WFReportConfigure[] reportConfigs = svc.GetAllReportConfigure();
    Console.WriteLine("This AgilePoint Server has {0} reports configured:", reportConfigs.Length);
    foreach (WFReportConfigure config in reportConfigs)
    {
        Console.WriteLine("Report Name: {0}", config.ReportName);
    }
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}

/*
This example produces the following results:
This AgilePoint Server has 1 reports configured:
Report Name:Average Process Time By Automatic Activities
*/
```

Supported Versions

3.2.0.4 and higher

Get Report Configuration

API Type

Web Services

Description

Retrieves a report configuration from the system.

Syntax

```
public virtual WfReportConfigure GetReportConfigure(string reportName)
```

Parameters

Name	Type	Description
reportName	<u>Definition:</u> The name of a report. <u>Type</u> string <u>Allowed Values:</u> A valid report name.	

Output

WfReportConfigure object.

Example

```
IWFAdminService svc = GetAdminService();
string reportName = ...;

try
{
    WfReportConfigure cfg = svc.GetReportConfigure(reportName);
    Console.WriteLine("Report Name: {0}, config:{1}", cfg.ReportName,
        cfg.Configure);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

```
}
```

Supported Versions

3.2.0.4 and higher

Read Configuration

API Type

Web Services

Description

Read an event service configuration.

Syntax

```
public string ReadConfiguration()
```

Parameters

Name	Description
None	Not Applicable

Output

[WFEvent](#) object.

Example

```
public string ReadConfiguration()
{
    IWFEventServiceConfiguration eventservice = GetEventService();
    string configuration = eventservice.ReadConfiguration();
}
```

Supported Versions

3.2.0.4 and higher

Remove Report Configure

API Type

Web Services

Description

Removes a report configuration from the system.

Syntax

```
public virtual void RemoveReportConfigure(string reportName)
```

Parameters

Name	Type	Description
reportName	<u>Definition:</u> The name of a report. <u>Type</u> string <u>Allowed Values:</u> A valid report name.	

Output

None.

Example

```
IWFAdminService svc = GetAdminService();  
string reportName = ...;  
  
try  
{  
    svc.RemoveReportConfigure(reportName);  
}  
  
catch (Exception ex)  
{  
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));  
}
```

Supported Versions

3.2.0.4 and higher

Update Report Configuration

API Type

Web Services

Description

Updates a report configuration in the AgilePoint system.

Syntax

```
public virtual WfReportConfigure UpdateReportConfigure(string reportName, string configure)
```

Parameters

Name	Description
reportName	<u>Definition:</u> The name of a report. <u>Type</u> string <u>Allowed Values:</u> A valid report name.
configure	<u>Definition:</u> The report configuration in XML format. <u>Type</u> string <u>Allowed Values:</u> A valid report configuration in XML format.

Output

WfReportConfigure object.

Example

```
IWFAdminService svc = GetAdminService();
string reportName = ...//for example, "weekly task report"
string configure = ...// xml-serialization of WfReportConfiguration

try
{
    WfReportConfigure reportConfig =
        svc.UpdateReportConfigure(reportName, configure);
}

catch (Exception ex)
{
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));
}
```

Supported Versions

3.2.0.4 and higher

Component Administration Methods

This section describes service calls related to server component administration.

Get Server Component

API Type

Web Services

Description

Retrieves a server component name.

Syntax

public virtual WFComponent GetServerComponent(string Name)

Parameters

Name	Description
Name	<u>Definition:</u> The name of an item, such as a property or attribute in a name/value pair. <u>Type</u> string <u>Allowed Values:</u> A valid name.

Output

WFComponent object.

Example

None.

Supported Versions

3.2.0.4 and higher

Get Server Component Names

API Type

Web Services

Description

Retrieves the server component names using the Admin Services.

Syntax

```
public virtual string[] GetServerComponentNames()
```

Parameters

Name	Description
None	Not Applicable

Output

An array of strings that contain the server component names.

Example

```
IWFAdminService svc = GetAdminService();  
  
try  
{  
    string[] names = svc.GetServerComponentNames();  
}  
  
catch (Exception ex)  
{  
    Console.WriteLine("Failed! " + ShUtil.GetSoapMessage(ex));  
}
```

Supported Versions

3.2.0.4 and higher

Classes

This section includes references for all classes within the AgilePoint Web Service API.

KeyValue

Description

A class that represents an object with name and value properties.

Syntax

```
public class KeyValue
```

Constructors

```
public KeyValue();  
public KeyValue(string key, string val);
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
Key	<u>Definition:</u> Specifies the key of the instance. <u>Type</u> <code>string</code> <u>Allowed Values:</u>

Property	Description
	A valid key.
Value	<u>Definition:</u> The value for an item, such as the value for an attribute in a name-value pair. <u>Type</u> string <u>Allowed Values:</u> A valid string value.

NameValue

Description

A class that represents an object with the name and value properties.

Syntax

```
public class NameValue
```

Constructors

```
public NameValue();  
public NameValue(string name,object value);
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.Share (in Ascentn.Workflow.Share.dll)

Properties

Property	Description
Name	<u>Definition:</u> The name of an item, such as a property or attribute in a name/value pair. <u>Type</u> string <u>Allowed Values:</u> A valid name.
Value	<u>Definition:</u> The value for an item, such as the value for an attribute in a key-value pair. <u>Type</u> object <u>Allowed Values:</u> A valid object value.

IWFWorkflowService

Description

A class that provides interfaces for the AgilePoint workflow API on the client side.

Syntax

```
public interface IWFWorkflowService
```

Constructors

Not Applicable.

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

IWFTrackingEventPublisher

Description

Provides an interface to publish a process event.

Syntax

```
public interface IWFTrackingEventPublisher
```

Constructors

Not Applicable.

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
AppNameFilter	<u>Definition:</u> Gets the application name filter. <u>Type</u>

Property	Description
	string <u>Allowed Values:</u> A valid application name filter.
EventTypeFilter	<u>Definition:</u> Gets the event type filter. <u>Type</u> string <u>Allowed Values:</u> A valid event type filter.

RegisteredUser

Description

A class that represents an AgilePoint registered user.

Syntax

```
public class RegisteredUser
```

Constructors

```
public RegisteredUser();  
public RegisteredUser(string user, string emailAddress, DateTime registeredDate, string fullName);
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
Department	<u>Definition:</u> Gets and sets department of the user. <u>Type</u> string <u>Allowed Values:</u> A valid department name.
EEmailAddress	<u>Definition:</u> Gets and sets the user's e-mail address. <u>Type</u> string <u>Allowed Values:</u> A valid e-mail address.
FullName	<u>Definition:</u> Specifies the full name of the user. <u>Type</u> string <u>Allowed Values:</u> A single line of text.
Manager	<u>Definition:</u> Specifies the manager of the user. <u>Type</u> string <u>Allowed Values:</u> A valid user name.
RegisteredDate	<u>Definition:</u> Gets and sets date registered. <u>Type</u>

Property	Description
	DateTime <u>Allowed Values:</u> A valid DateTime value.
Title	<u>Definition:</u> Specifies the job title of the user. <u>Type</u> string <u>Allowed Values:</u> A single line of text.
UserName	<u>Definition:</u> Specifies a qualified user name of the instance. A qualified user name formats as [Domain Name]\[Logon Username] or [Local host name]\[Logon Username]. <u>Type</u> string <u>Allowed Values:</u> A valid user name for a registered AgilePoint user.

WFAccessRights

Description

A class that represents different types of access rights.

Syntax

```
public enum WFAccessRights
```

Constructors

Not Applicable.

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

WFAgilePart

Description

An abstract class of AgilePart.

Syntax

```
public class WFAgilePart
```

Constructors

```
protected WFAgilePart();
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

WFAgilePartDescriptor

Description

An abstract class of WFAgilePartDescriptor.

Syntax

```
public class WFAgilePartDescriptor : WFAutomaticActivityDefinition, IWFPProcessDefinitionReference
```

Constructors

```
public WFAgilePartDescriptor();
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
AssemblyName	<u>Definition:</u> Gets and sets the full name of the assembly. <u>Type</u> string <u>Allowed Values:</u> A valid assembly name.
ClassName	<u>Definition:</u> Gets and sets the class name, including the namespace. <u>Type</u> string <u>Allowed Values:</u> A valid class name.
Description	<u>Definition:</u> A free text description of the activity.

Property	Description
	<u>Type</u> string <u>Allowed Values:</u> A string that can contain spaces and special characters.
DisplayName	<u>Definition:</u> Gets and sets the display name of the activity. <u>Type</u> string <u>Allowed Values:</u> A valid activity display name.
ExpectedDuration	<u>Definition:</u> Specifies an expected time duration of the activity. <u>Type</u> WFTimeDuration <u>Allowed Values:</u> A valid WFTimeDuration object.
ID	<u>Definition:</u> Specifies the activity ID. <u>Type</u> WFTimeDuration <u>Allowed Values:</u> A valid activity ID.
IncomingActivities	<u>Definition:</u> Gets the interfaces for all the incoming activities. <u>Type</u> WFTimeDuration <u>Allowed Values:</u> A valid WFTimeDuration object.
IncomingEmail	<u>Definition:</u>

Property	Description
	Specifies an email template that used for an email notification after the start of the activity. <u>Type</u> string <u>Allowed Values:</u> A valid email template.
IncomingMailTemplate	<u>Definition:</u> Specifies an email template that used for an email notification after the start of the activity. <u>Type</u> string <u>Allowed Values:</u> A valid email template.
Index	<u>Definition:</u> Specifies the ordering index of the activity. <u>Type</u> int <u>Allowed Values:</u> An integer.
Item	<u>Definition:</u> Specifies the value associated with the specified parameter name. <u>Type</u> object <u>Allowed Values:</u> A valid Types object.
Method	<u>Definition:</u> Specifies the name of the method for the AgilePart. <u>Type</u> string <u>Allowed Values:</u> A valid method name.

Property	Description
Name	<u>Definition:</u> The name of an item, such as a property or attribute in a name/value pair. <u>Type</u> string <u>Allowed Values:</u> A valid name.
OutgoingActivities	<u>Definition:</u> Gets the interfaces for all the outgoing activities. <u>Type</u> string <u>Allowed Values:</u> A valid interface.
OutgoingEmail	<u>Definition:</u> Specifies an email template that used for email notification after the completion of an activity. <u>Type</u> string <u>Allowed Values:</u> A valid email template.
OutgoingMailTemplate	<u>Definition:</u> Specifies an email template to use for email notification when leaving the activity. <u>Type</u> string <u>Allowed Values:</u> A valid email template.
Procedure	<u>Definition:</u> Specifies the name of the procedure (method). <u>Type</u> string <u>Allowed Values:</u>

Property	Description
	A valid procedure name.
Process	<u>Definition:</u> Gets the interface class and sets the process definition. <u>Type</u> string <u>Allowed Values:</u> A valid process definition.
SaveErrorMessageTo	<u>Definition:</u> Specifies an attribute name for an error message of the AgilePart execution. <u>Type</u> string <u>Allowed Values:</u> A valid attribute name.
SaveStatusTo	<u>Definition:</u> Specifies an attribute name for the status of the AgilePart execution. <u>Type</u> string <u>Allowed Values:</u> A valid name of an attribute that is boolean type.
Synchronous	<u>Definition:</u> Specifies the flag to determine if this is a synchronous call. <u>Type</u> bool <u>Allowed Values:</u> • True - The call is synchronous. • False - The call is asynchronous.
TimeoutActivities	<u>Definition:</u> Gets the interfaces for all the timeout activities.

Property	Description
	<u>Type</u> string <u>Allowed Values:</u> A valid interface.
TimeoutMailTemplate	<u>Definition:</u> Specifies an email template that used for email notification when the activity is overdue. <u>Type</u> string <u>Allowed Values:</u> A valid email template.
WaitAllIncoming	<u>Definition:</u> Gets and sets the flag to determine whether the activity must wait until all predecessor activities complete. <u>Type</u> string <u>Allowed Values:</u> • True - The activity waits until all predecessor activities complete. • False - The activity starts without waiting for predecessor activities complete.

WFAgileWork

Description

An abstract class of WFAgileWork.

Syntax

```
public class WFAgileWork
```

Constructors

```
protected WFAgileWork(WFProcessInstance currentProcessInstance, WFManualActivityInstance
currentActivityInstance);
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

WFAgileWorkDescriptor

Description

An abstract class of WFAgileWorkDescriptor Custom properties format: [Prefix]:[Assembly Information]:[Class Name]:[Parameter].

Syntax

```
public class WFAgileWorkDescriptor : WFManualActivityDefinition, IWFPProcessDefinitionReference
```

Constructors

```
public WFAgileWorkDescriptor();
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
AssemblyName	<u>Definition:</u> Gets and sets the full name of the assembly. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid assembly name.
AutoComplete	<u>Definition:</u> Gets and sets the flag that determines whether the work item is automatically marked as completed. <u>Type</u> <code>bool</code> <u>Allowed Values:</u> • True - The work item is completed automatically. • False - The work item is not completed automatically.
ClassName	<u>Definition:</u> Gets and sets the class name, including the namespace. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid class name.
CustomProperties	<u>Definition:</u> Specifies the custom properties. This member supports the AgilePoint Framework infrastructure and does not intend to use directly from your code. <u>Type</u> <code>string</code> <u>Allowed Values:</u>

Property	Description
	A valid custom properties.
Description	<u>Definition:</u> A free text description of the activity. <u>Type</u> string <u>Allowed Values:</u> A string that can contain spaces and special characters.
DisplayName	<u>Definition:</u> Gets and sets the display name of the activity. <u>Type</u> string <u>Allowed Values:</u> A valid activity display name.
ExpectedDuration	<u>Definition:</u> Specifies an expected time duration of the activity. <u>Type</u> WFTimeDuration <u>Allowed Values:</u> A valid WFTimeDuration object.
ID	<u>Definition:</u> Specifies the activity ID. <u>Type</u> WFTimeDuration <u>Allowed Values:</u> A valid activity ID.
IncomingActivities	<u>Definition:</u> Gets the interfaces for all the incoming activities. <u>Type</u> WFTimeDuration <u>Allowed Values:</u>

Property	Description
	A valid WFTimeDuration object.
IncomingEmail	<u>Definition:</u> Specifies an email template that used for an email notification after the start of the activity. <u>Type</u> string <u>Allowed Values:</u> A valid email template.
IncomingMailTemplate	<u>Definition:</u> Specifies an email template that used for an email notification after the start of the activity. <u>Type</u> string <u>Allowed Values:</u> A valid email template.
Index	<u>Definition:</u> Specifies the ordering index of the activity. <u>Type</u> int <u>Allowed Values:</u> An integer.
Item	<u>Definition:</u> Specifies the value associated with the specified parameter name. <u>Type</u> object <u>Allowed Values:</u> A valid Types object.
MaxParticipant	<u>Definition:</u> Specifies the maximum participants for the activity. <u>Type</u> int

Property	Description
	<u>Allowed Values:</u> An integer.
Name	<u>Definition:</u> The name of an item, such as a property or attribute in a name/value pair. <u>Type</u> string <u>Allowed Values:</u> A valid name.
Optional	<u>Definition:</u> Gets and sets the flag to indicate if the activity is optional. <u>Type</u> bool <u>Allowed Values:</u> • True - The activity is optional. • False - The activity is not optional.
OutgoingActivities	<u>Definition:</u> Gets the interfaces for all the outgoing activities. <u>Type</u> string <u>Allowed Values:</u> A valid interface.
OutgoingEmail	<u>Definition:</u> Specifies an email template that used for email notification after the completion of an activity. <u>Type</u> string <u>Allowed Values:</u> A valid email template.
OutgoingMailTemplate	<u>Definition:</u>

Property	Description
	Specifies an email template to use for email notification when leaving the activity. <u>Type</u> string <u>Allowed Values:</u> A valid email template.
Participant	<u>Definition:</u> Specifies the participant of the activity. <u>Type</u> string <u>Allowed Values:</u> A valid user name.
Participants	<u>Definition:</u> Specifies the participants of the activity. <u>Type</u> string <u>Allowed Values:</u> A valid multiple user names separated with a semicolon (;).
Process	<u>Definition:</u> Gets the interface class and sets the process definition. <u>Type</u> string <u>Allowed Values:</u> A valid process definition.
ReassigningEmail	<u>Definition:</u> Specifies the email template that used for an email notification when task is reassigned. <u>Type</u> string <u>Allowed Values:</u>

Property	Description
	A valid email template.
ReassigningMailTemplate	<u>Definition:</u> Specifies the email template for task reassignment. <u>Type</u> string <u>Allowed Values:</u> A valid email template.
RemindingEmail	<u>Definition:</u> Specifies an email template that used for reminding email notification. <u>Type</u> string <u>Allowed Values:</u> A valid email template.
RemindingMailAtBusinessTime	<u>Definition:</u> Gets and sets the flag that determines whether reminding an email must send in business time. <u>Type</u> bool <u>Allowed Values:</u> • True - Reminding emails are sent using business time. • False - Reminding email are sent using real time.
RemindingMailFrequency	<u>Definition:</u> Specifies the frequency of reminding an email notification. <u>Type</u> int <u>Allowed Values:</u> An integer.
RemindingMailStartTime	<u>Definition:</u>

Property	Description
	Specifies the start time in minutes of reminding an email notification before the task expires. <u>Type</u> int <u>Allowed Values:</u> An integer.
RemindingMailTemplate	<u>Definition:</u> Specifies the email template for task reminder emails. <u>Type</u> int <u>Allowed Values:</u> A valid email template.
ReuseParticipant	<u>Definition:</u> Gets and sets the flag that determines whether the participants must remain the same when the activity activated more than one time. <u>Type</u> bool <u>Allowed Values:</u> • True - The participants remains the same when the activity activated more than one time. • False - The participants do not remain the same when the activity activated more than one time.
TimeoutActivities	<u>Definition:</u> Gets the interfaces for all the timeout activities. <u>Type</u> string <u>Allowed Values:</u> A valid interface.
TimeoutEmail	<u>Definition:</u> Specifies the email template to use for email notification when task is overdue. <u>Type</u>

Property	Description
	string <u>Allowed Values:</u> A valid email template.
TimeoutMailTemplate	<u>Definition:</u> Specifies an email template that used for email notification when the activity is overdue. <u>Type</u> string <u>Allowed Values:</u> A valid email template.
WaitAllIncoming	<u>Definition:</u> Gets and sets the flag to determine whether the activity must wait until all predecessor activities complete. <u>Type</u> string <u>Allowed Values:</u> • True - The activity waits until all predecessor activities complete. • False - The activity starts without waiting for predecessor activities complete.
WaitWorkPerformed	<u>Definition:</u> Specifies whether the activity must be marked as completed by the application before proceeding. <u>Type</u> bool <u>Allowed Values:</u> • True - The activity must complete before proceeding. • False - The activity can proceed without waiting.
WorkPerformer	<u>Definition:</u> Specifies the WorkPerformer/job of the manual activity. <u>Type</u>

Property	Description
	string <u>Allowed Values:</u> A valid WorkPerformer/job name.
WorkToPerform	<u>Definition:</u> Represents the task that performed by the participants of the activity. <u>Type</u> string <u>Allowed Values:</u> A valid work to perform name.

WFAny

Description

A class that represents a primitive data type with type code.

Syntax

```
public class WFAny
```

Constructors

```
public WFAny();
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
Type	<u>Definition:</u> Gets and sets the type of a WFAny object. <u>Type</u> int <u>Allowed Values:</u> An integer.
Value	<u>Definition:</u> The value for an item, such as the value for an attribute in a key-value pair. <u>Type</u> object <u>Allowed Values:</u> A valid object value.

WFEvent

Description

A class that represents a workflow event.

Syntax

```
public class WFEvent
```

Constructors

```
public WFEvent();  
public WFEvent(string name);  
public WFEvent(string sender, string name);
```


Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
ActivityInstID	<u>Definition:</u> The unique ID for an activity instance. <u>Type</u> string <u>Allowed Values:</u> A valid activity instance ID.
AutoStart	<u>Definition:</u> Specifies if the process starts immediately after it is created. <u>Type</u> bool <u>Allowed Values:</u> • True - The process instance starts immediately after it is created. • False - The process instance does not start immediately after it is created.
clientData	<u>Definition:</u> Specifies the client data, which identifies a client for AgilePoint Server. <u>Type</u> string <u>Allowed Values:</u> A string that contains the client data.

Property	Description
	If this value is null, the system will keep existing client data. Otherwise the relevant data is overwritten.
CustomAttributes	<u>Definition:</u> Specifies custom attributes in XML format. <u>Type</u> NameValue <u>Allowed Values:</u> A valid NameValue pair.
Designated	<u>Definition:</u> Gets and sets the flag that determines if the event should be handled by a designated engine. <u>Type</u> bool <u>Allowed Values:</u> • True - The event is handled by a designated engine. • False - The event is not handled by a designated engine.
Diagnostic	<u>Definition:</u> Gets and sets the flag that determines if the process data will be deleted from the database after the process is completed. <u>Type</u> bool <u>Allowed Values:</u> • True - The process data is deleted from the database. • False - The process data is not deleted from the database.
EndDate	<u>Definition:</u> Specifies the completion date of delegation. <u>Type</u> DateTime <u>Allowed Values:</u>

Property	Description
	A valid DateTime value.
Entries	<u>Definition:</u> Gets the relay time of the event. <u>Type</u> <code>int</code> <u>Allowed Values:</u> A valid relay time.
Error	<u>Definition:</u> Gets and sets the error message of the event. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid error message.
EventID	<u>Definition:</u> Specifies a unique ID for an event. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A unique, 32-character ID.
EventName	<u>Definition:</u> Specifies the name of the event. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid event name.
ParamsXml	<u>Definition:</u> Gets and sets the parameters as XML. <u>Type</u> <code>string</code> <u>Allowed Values:</u>

Property	Description
	A parameter.
ParentProcInstID	<u>Definition:</u> Specifies parent process instance ID. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID.
ProcDefID	<u>Definition:</u> Specifies process template (process definition) ID. <u>Type</u> string <u>Allowed Values:</u> A valid process definition ID.
ProcInstID	<u>Definition:</u> Specifies the unique ID of a process instance. <u>Type</u> string <u>Allowed Values:</u> A valid process instance ID
ProcInstName	<u>Definition:</u> A unique process name that is associated with the process definition. <u>Type</u> string <u>Allowed Values:</u> A unique process instance name up to 1024 characters.
Sender	<u>Definition:</u> Specifies a sender of the event. <u>Type</u> string

Property	Description
	<u>Allowed Values:</u> A valid user name.
SentDate	<u>Definition:</u> Specifies a sent date of the event. <u>Type</u> DateTime <u>Allowed Values:</u> A valid DateTime value.
SourceWorkItemID	<u>Definition:</u> An ID that represents the original, or source, work item. <u>Type</u> string <u>Allowed Values:</u> A valid, unique 32-character ID.
Status	<u>Definition:</u> The status of the event. <u>Type</u> string <u>Allowed Values:</u> A valid status.
ThrowAwayInstance	<u>Definition:</u> Specifies the flag that determines if AgilePoint server should clear the process from cache. <u>Type</u> bool <u>Allowed Values:</u> True - AgilePoint Server clears the process from the cache. False - AgilePoint Server does not clear the process from cache.
UserID	<u>Definition:</u>

Property	Description
	Specifies the user ID. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid user ID.
WorkItemID	<u>Definition:</u> An ID that represents a work item (task). <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid, unique 32-byte work item (task) ID.
workObjectID	<u>Definition:</u> An ID for an object, such as a document, that is associated with the process instance. <u>Type</u> <code>string</code> <u>Allowed Values:</u> A valid, unique 256-character ID. Even though the field size is 256 characters, in common practice, this will usually return a 32-character GUID.

WFIntegratedApplication

Description

A class that represents an integrated application, such as an AgileConnector.

Syntax

```
public class WFIntegratedApplication : IWFIntegratedApplication
```

Constructors

```
public WFIntegratedApplication();
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
AppEventKind	<u>Definition:</u> Specifies the type of application event. <u>Type</u> IWFWorkflowService <u>Allowed Values:</u> A valid application event.
AppName	<u>Definition:</u> Specifies the name of the application. <u>Type</u> string <u>Allowed Values:</u> A valid, case-sensitive application name.
Category	<u>Definition:</u> Gets and sets a category name. <u>Type</u> string <u>Allowed Values:</u> A valid category name.

Property	Description
CustomConfiguration	<u>Definition:</u> Gets and sets custom configuration settings. <u>Type</u> System.Xml.NodeList <u>Allowed Values:</u> A valid set of custom configuration settings.
Description	<u>Definition:</u> A free text description of the application. <u>Type</u> string <u>Allowed Values:</u> A string that can contain spaces and special characters.
Impersonator	<u>Definition:</u> Specifies a user name of an impersonator user account. <u>Type</u> string <u>Allowed Values:</u> A valid impersonator user name.

WFIntegratedApplicationDescriptor

Description

An abstract class that describes the global control module.

Syntax

```
public class WFIntegratedApplicationDescriptor : IWFIntegratedApplicationDescriptor
```


Constructors

```
public WFIntegratedApplicationDescriptor(string integratedApplicationsXML);
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
integratedApplicationsXml	<u>Definition:</u> Gets the configuration settings for the integrated application. <u>Type</u> string <u>Allowed Values:</u> Valid configuration settings.

WFPartialRollbackInstruction

Description

A class that provides instructions for activating activity instances.

Syntax

```
public class WFPartialRollbackInstruction
```

Constructors

```
public WFPartialRollbackInstruction();
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
PartialRollbackUnits	<u>Definition:</u> An array of instances of the PartialRollbackUnit class. <u>Type</u> PartialRollbackUnit <u>Allowed Values:</u> One or more PartialRollbackUnit objects.

WFProcessMergingInstruction

Description

A class that provides instructions for merging process instances during runtime.

Syntax

```
public class WFProcessMergingInstruction
```

Constructors

Not Applicable.

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
MergingProcessInstanceIDs	<u>Definition:</u> An array that includes the process instance IDs of the process instances you are merging. <u>Type</u> string <u>Allowed Values:</u> An array of valid process instance IDs.
MergedProcessInstance	<u>Definition:</u> An instance of the MergedProcess Parameter class that includes the merged process instance. <u>Type</u> MergedProcessParameter <u>Allowed Values:</u> A valid An MergedProcess Parameter object.

WFProcessMigrationInstruction

Description

A class that provides instructions for merging process instances during runtime.

Syntax

```
public class WFProcessMigrationInstruction
```

Constructors

Not Applicable.

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Type	Description
MatchingActivityDefinition	<u>Definition:</u> Specifies information for activity migration. <u>Type</u> class <u>Allowed Values:</u> A valid class.	
SourceProcessDefinitionID	<u>Definition:</u> The ID of the original, or source, process definition. <u>Type</u> string <u>Allowed Values:</u> A valid process definition ID.	
TargetProcessDefinitionID	<u>Definition:</u>	

Property	Type	Description
	The ID of the target, or destination, process definition. <u>Type</u> string <u>Allowed Values:</u> A valid process definition ID.	
Action	<u>Definition:</u> Specifies the migration action. <u>Type</u> MigrationAction <u>Allowed Values:</u> A valid migration action.	
IncludeXmlData	<u>Definition:</u> Specifies whether a migration includes XML data. <u>Type</u> bool <u>Allowed Values:</u> • True - The migration includes an XML data. • False - The migration does not include an XML data.	

WFProcessPluggableAdapter

Description

A class that represents an AgileExtender.

Syntax

```
public WFProcessPluggableAdapter
```

Constructors

```
public WFProcessPluggableAdapter(WFProcessInstance currentProcessInstance)
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
ComponentTypeID	<u>Definition:</u> Specifies a unique type ID of an Agile Extender. <u>Type</u> string <u>Allowed Values:</u> A valid type ID for your AgileExtender.
Name	<u>Definition:</u> Specifies the name of an AgileExtender. <u>Type</u> string <u>Allowed Values:</u> A valid AgileExtender name.
DisplayName	<u>Definition:</u> Gets and sets the display name of an AgileExtender. <u>Type</u> string <u>Allowed Values:</u>

Property	Description
	A valid AgileExtender display name.
Description	<u>Definition:</u> A free text description of an AgileExtender. <u>Type</u> string <u>Allowed Values:</u> A string that can contain spaces and special characters.
AssemblyFullName	<u>Definition:</u> Gets the full name of an AgileExtender assembly. <u>Type</u> string <u>Allowed Values:</u> A valid AgileExtender assembly name.
ClassName	<u>Definition:</u> Gets the type of the AgileExtender. <u>Type</u> string <u>Allowed Values:</u> A valid AgileExtender type.
ProcessInstance	<u>Definition:</u> Gets a runtime process instance object. <u>Type</u> WFProcessInstance <u>Allowed Values:</u> A WFProcessInstance object.

WFProcessPluggableAdapterDescriptor

Description

A Design Time supporting class for Agile Extender. This Design Time class extends the class WFProcessPluggableAdapterDescriptor.

Syntax

```
public partial class MyAgileExtenderDescriptor : WFProcessPluggableAdaptterDescriptor
```

Constructors

```
public WFProcessPluggableAdaptterDescriptor()  
public WFProcessPluggableAdaptterDescriptor(bool designTime)
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
ComponentTypeID	<u>Definition:</u> Specifies a unique type ID of an Agile Extender. <u>Type</u> string <u>Allowed Values:</u> A valid type ID for your AgileExtender.
Name	<u>Definition:</u> Specifies the name of an AgileExtender.

Property	Description
	<u>Type</u> string <u>Allowed Values:</u> A valid AgileExtender name.
DisplayName	<u>Definition:</u> Gets and sets the display name of an AgileExtender. <u>Type</u> string <u>Allowed Values:</u> A valid AgileExtender display name.
Description	<u>Definition:</u> A free text description of an AgileExtender. <u>Type</u> string <u>Allowed Values:</u> A string that can contain spaces and special characters.
AssemblyFullName	<u>Definition:</u> Gets the full name of an AgileExtender assembly. <u>Type</u> string <u>Allowed Values:</u> A valid AgileExtender assembly name.
ClassName	<u>Definition:</u> Gets the type of the AgileExtender. <u>Type</u> string <u>Allowed Values:</u> A valid AgileExtender type.
Process	<u>Definition:</u>

Property	Description
	Gets the interface class and sets the process definition. <u>Type</u> IWFProcessDefinition <u>Allowed Values:</u> A IWFProcessDefinition class.
DesignTime	<u>Definition:</u> Specifies whether the object is used at design time. <u>Type</u> bool <u>Allowed Values:</u> • True - The object is used at design time. • False - The object is not used at design time.

WFProcessSplittingInstruction

Description

A class that provides instructions for splitting a process instance during runtime.

Syntax

```
public class WFProcessSplittingInstruction
```

Constructors

Not Applicable.

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
SplitProcessInstances	<u>Definition:</u> An array of instances of the SplitProcessParameter class. <u>Type</u> SplitProcessParameter <u>Allowed Values:</u> A valid SplitProcessParameter object.

WFQueryExpr

Description

A class that represents a query expression object.

Syntax

```
public class WFQueryExpr
```

Constructors

```
public WFQueryExpr();  
public WFQueryExpr(string columnName, int op, WFAny any, bool val);
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
Any	<u>Definition:</u> Gets and sets the WFAAny object to be compared. <u>Type</u> WFAAny <u>Allowed Values:</u> A valid WFAAny object.
ColumnName	<u>Definition:</u> Gets and sets the associated database column name. <u>Type</u> string <u>Allowed Values:</u> A valid database column name.
IsValue	<u>Definition:</u> Gets and sets the flag that indicates if it compares a value or column of the table. <u>Type</u> bool <u>Allowed Values:</u> • True - It compares a value or column of the table. • False - It does not compare a value or column of the table.
Name	<u>Definition:</u> The name of an item, such as a property or attribute in a name/value pair. <u>Type</u> string <u>Allowed Values:</u>

Property	Description
	A valid name.
Operator	<u>Definition:</u> Specifies the operator used for comparison. <u>Type</u> int <u>Allowed Values:</u> A valid operator.

WFTimeDuration

Description

A class that represents Time Duration with length, time unit, and business time.

Syntax

```
public class WFTimeDuration : Serializable
```

Constructors

```
public WFTimeDuration();
public WFTimeDuration(string length, WFTimeUnit unit, bool b);
public WFTimeDuration(int length, WFTimeUnit unit, bool b);
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
BusinessTime	<u>Definition:</u> Determines whether the system calculates the duration using your business time calendar. <u>Type</u> bool <u>Allowed Values:</u> • True - The system calculates duration using business time. • False - The calculates the duration based on real time.
Length	<u>Definition:</u> Specifies the length of time duration. <u>Type</u> string <u>Allowed Values:</u> A valid time duration length.
Unit	<u>Definition:</u> Specifies the time unit. <u>Type</u> WFTimeUnit <u>Allowed Values:</u> • hour • day • week • month • year

WFTimeUnit

Description

A class that represents time unit.

Syntax

```
public class WFTimeUnit
```

Constructors

```
public WFTimeUnit();
```

Namespace and Assembly

Requirement	Value
Namespace	Ascentn.Workflow.Base
Assembly	Ascentn.Workflow.WFBase (in Ascentn.Workflow.WFBase.dll)

Properties

Property	Description
Value	<u>Definition:</u> The value for an item expressed as an integer, such as a time unit. <u>Type</u> int <u>Allowed Values:</u> An integer.

Data Types

A list of all data types used in the AgilePoint Web Service API. Common types refer to the documentation on msdn.com

Name	Description
bool	See the documentation on MSDN .
byte	See the documentation on MSDN .
DateTime	See the documentation on MSDN .
enum	See the documentation on MSDN .
Guid	Instantiates the Guid class.
int	See the documentation on MSDN .
IWFProcessDefinition	Instantiates the IWFProcessDefinition class.
IWFWorkflowService	Instantiates the IWFWorkflowService class.
KeyValue	Instantiates the KeyValue class.
MergedProcessParameter	Instantiates the MergedProcessParameter class.
MigrationAction	Instantiates the MigrationAction class.
NameValue	Instantiates the NameValue class.
object	See the documentation on MSDN .
PartialRollbackUnit	Instantiates the PartialRollbackUnit class.
RegisteredUser	Instantiates the RegisteredUser class.
SplitProcessParameter	Instantiates the SplitProcessParameter class.
String	See the documentation on MSDN .
WFAccessRights	Instantiates the WFAccessRights class.
WFAny	Instantiates the WFAny class.
WFEvent	Instantiates the WFEvent class.
WFProcessInstance	Instantiates the WFProcessInstance class.
WFQueryExpr	Instantiates the WFQueryExpr class.
WFTimeDuration	Instantiates the WFTimeDuration class.
WFTimeUnit	See the documentation on MSDN .