AgilePoint Developer User's Guide

AgilePoint BPMS v5.0 SP4

Document Revision r5.5.5

August 2013

Contents

Preface

Disclaimer of Warranty

Copyright

Trademarks

Government Rights Legend

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 in 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 team**, 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

AgilePoint Developer

AgilePoint Developer is a component of AgilePoint BPMS used by software developers to create reusable modules and extensions to develop highly complex and customized workflow management solutions. In short, AgilePoint Developer enables you to create any process management customizations you want that aren't included with AgilePoint BPMS out of the box. It is a Microsoft Visual Studio.NET add-in.

This document is a user's guide and reference to the features, functionality, usage, configuration, and administration of the AgilePoint Developer component of the AgilePoint BPMS Suite.

The following languages are currently supported by the AgilePoint Developer project templates: English, Japanese, Korean, Simplified Chinese, Traditional Chinese, and Hebrew. Other languages (such as Spanish) are not currently supported, but can be enabled with some manual work. If your language is not supported an error will be returned. For more information, contact AgilePoint Support.

Converting a Web Application to AgilePoint Format

Prerequisites

- This command is only available when a non-AgilePoint web application project is open in AgilePoint Developer.
- This option only works for traditional AgilePoint web applications, not for the AgilePoint Web Application for Azure format.

Note:

To add additional Web form pages to the application, click Tools > Convert to AgilePoint Web Application.

Project Templates

This section provides a general overview of the features, functionality, purpose, and intended usage of the project templates included with the AgilePoint Developer component of the AgilePoint BPMS Suite.

These project templates can be accessed using the **File > New > Project** menu command.

AgilePart Project Template

This project template is used to create a new custom AgilePart. AgilePart assemblies must be deployed into the AgilePoint Server's GAC. Deploying the AgilePart assemblies can be done manually via drag and drop or by using the AgilePart Deployment Utility.

AgilePoint Envision provides the AgilePart and AgileWork AgileShapes, which enables you to associate your own custom AgileParts or AgileWorks (which you build in AgilePoint Developer) with a common, pre-built AgileShape in Envision. This allows you to use the .NET code for a custom AgilePart or AgileWork without creating a custom Visio shape. This is a shortcut that is often used to test custom AgileParts or AgileWorks. Usually in a production environment, organizations will take the extra step to create a unique AgileShape to associate with the custom AgilePart or AgileWork. These AgileShapes include all of the core properties of AgileParts and AgileWorks. They can also include additional custom properties.

AgileWork Project Template

This project template is used to create a new custom AgileWork. AgileWork assemblies (like AgilePart assemblies) must be deployed into the AgilePoint Server's GAC. Deploying the AgileWork assemblies must be done manually via drag and drop or by using the .NET Framework tool gacutil.exe.

AgilePoint Envision provides the AgilePart and AgileWork AgileShapes, which enables you to associate your own custom AgileParts or AgileWorks (which you build in AgilePoint Developer) with a common, pre-built AgileShape in Envision. This allows you to use the .NET code for a custom AgilePart or AgileWork without creating a custom Visio shape. This is a shortcut that is often used to test custom AgileParts or AgileWorks. Usually in a production environment, organizations will take the extra step to create a unique AgileShape to associate with the custom AgilePart or AgileWork. These AgileShapes include all of the core properties of AgileParts and AgileWorks. They can also include additional custom properties.

The main differences of an AgileWork component as compared to an AgilePart are:

- An AgileWork component must inherit from the WFAgileWork base class.
- An AgileWork component's design-time support is created using the WFAgileWorkDescriptor class. (It is very similar to the WFAgilePartDescriptor class).
- All AgileWorks inherit the following server-side events from the WFAgileWork base class:
    - AssignWorkItem
    - CancelWorkItem
    - CompleteWorkItem
    - EnterActivityInstance
    - LeaveActivityInstance
    - ReassignWorkItem

Good to Know

- For AgileWork developers, at runtime, user-defined properties are processed and assigned to the WFManualWorkItem.ClientData property as an XML-serialized string of type NameValue[].

Queue Based AgileParts

Queue Based AgileParts provide an event-driven extension to traditional AgileParts that allow a mechanism for connecting from Windows Azure to your internal, third-party systems (for example, SAP or Oracle), while minimizing security risks and maximizing performance.

AgilePoint can connect to many third-party systems that cannot currently run on Windows Azure due either to limitations of the Windows Azure platform, or current limitations of the software. So, they must for now continue to run on your internal network or data center.

Connecting from AgilePoint BPMS for Azure to these third-party systems is not ideal using the traditional AgilePart architecture because a) connecting directly to individual systems creates a security risk, and b) Internet communication tends to be slower and less reliable than internal networks. Queue Based AgileParts extend the traditional AgilePart Framework to an event-driven model, in which communication with third-party systems is extrapolated to the AgilePoint Service Bus component.

When to Use Queue Based AgileParts

This topic provides guidance for when to use Queue Based AgileParts, and when not to.

When to Use Queue Based AgileParts

Use Queue Based AgileParts when an AgilePart operation:

- Requires more than a few seconds to execute.
- When a call goes across service boundaries — for example from Windows Azure to your local intranet.

When Not to Use Queue Based AgileParts

Queue Based AgileParts are not recommended when an AgilePart operation:

- Requires only a few seconds or less to execute.
- Occurs within your local network.

Creating a Queue Based AgilePart

To create a Queue Based AgilePart, do the following.

Prerequisites

- AgilePoint Service Bus installed.

Navigation

1. Open Microsoft Visual Studio.

2. In Visual Studio, click **New Project**.

Instructions

1. In the **New Project** window, click **Queue Based AgilePart**.
2. Click **OK**.

Instructions are provided in comments within the project.

Queue Based AgilePart Properties

This topic describes the properties that are common to Queue Based AgileParts.

Field Definitions

Field Name	Definition
Retries	Definition: Specifies the number of times to retry an action for a Queue Based AgilePart if it fails. Allowed Values: Any integer. Default Value: 0 Custom Attributes: No Property Group: Status and Error Message (Queue Based AgileParts Only)
HandleException	Definition: Specifies the action to take when an exception occurs. Allowed Values: • **Ignore** - Writes an error to the error log, and moves the process forward. • **SuspendProcess** - Writes an error to the error log and suspends the process. • **UserDefined** - Uses a custom dll you provide to determine the appropriate action. Default Value: Ignore

Field Name	Definition
	Custom Attributes: No Property Group: Status and Error Message (Queue Based AgileParts Only)
Optimizing	Definition: Specifies the action to take when an exception occurs. Allowed Values: • **Performance** - A message from an AgilePart is sent directly to a third-party system without a message queue. When this is set, the Retries property is always treated as if the value were 0. • **Scalability** - A message from an AgilePart goes through the internal message queue within AgilePoint Server. • **Connectivity** - A message from an AgilePart goes through the Azure message queue in the cloud. This requires the AgilePoint Broker to be running in your internal environment. • **Configuration** - The value is set based on the value of the process attribute, QueuebaseAgilePartOptimizing. You can set the value of this attribute at the process model, application, or global level to one of the following values: • **Performance** • **Scalability** • **Connectivity** To determine the value the process attribute to use, AgilePoint checks QueuebaseAgilePartOptimizing in the following context: • **[Application].[process model Name].QueuebaseAgilePartOptimizing** - The process attribute within the context of a specific process model within a specific application. • **[Application].QueuebaseAgilePartOptimizing** - The process attribute within the context of a specific application.

Field Name	Definition
	• **Global.Setting.QueuebaseAgilePartOptimizing** - The global process attribute. Even if this process attribute is set, individual AgileParts can have a different property if desired. Default Value: Configuration, or Performance. If the process attribute QueuebaseAgilePartOptimizing has a value, that value is used, and Configuration is considered the default setting. If the value of QueuebaseAgilePartOptimizing is not defined, this property is treated as if Performance were the default value. Custom Attributes: No, but the Configuration option uses a process attribute to determine its behavior. Property Group: AgilePart (Queue Based AgileParts Only)

Communicating with AgilePoint Server

To communicate securely with your internal data sources, Queue Based AgileParts running in AgilePoint BPMS for Azure can use a message queue on Windows Azure, and an on premises component called the AgilePoint Service Bus. The message queue organizes data requests in a linear order. The AgilePoint Service Bus then uses an outbound only HTTP or HTTPS connection to send and receive data from the queue.

AgilePoint Service Bus

The AgilePoint Service Bus is a component that runs in your environment on-premises or in your data center to manage communication between the AgilePoint message queues and your internal data sources. In AgilePoint BPMS for Azure, the AgilePoint Service Bus provides a single point of communication between all your Queue Based AgileParts and your internal data sources.

Having one source for input and output between AgilePoint BPMS for Azure and your internal systems has several advantages:

- You do not need to open only any ports to your internal network.
- The cloud-based message queue is always on, meaning that even if your Internet connection goes down, your processes will simply pick up where they left off when it returns.

The AgilePoint Service Bus uses a flexible architecture, which enables it to:

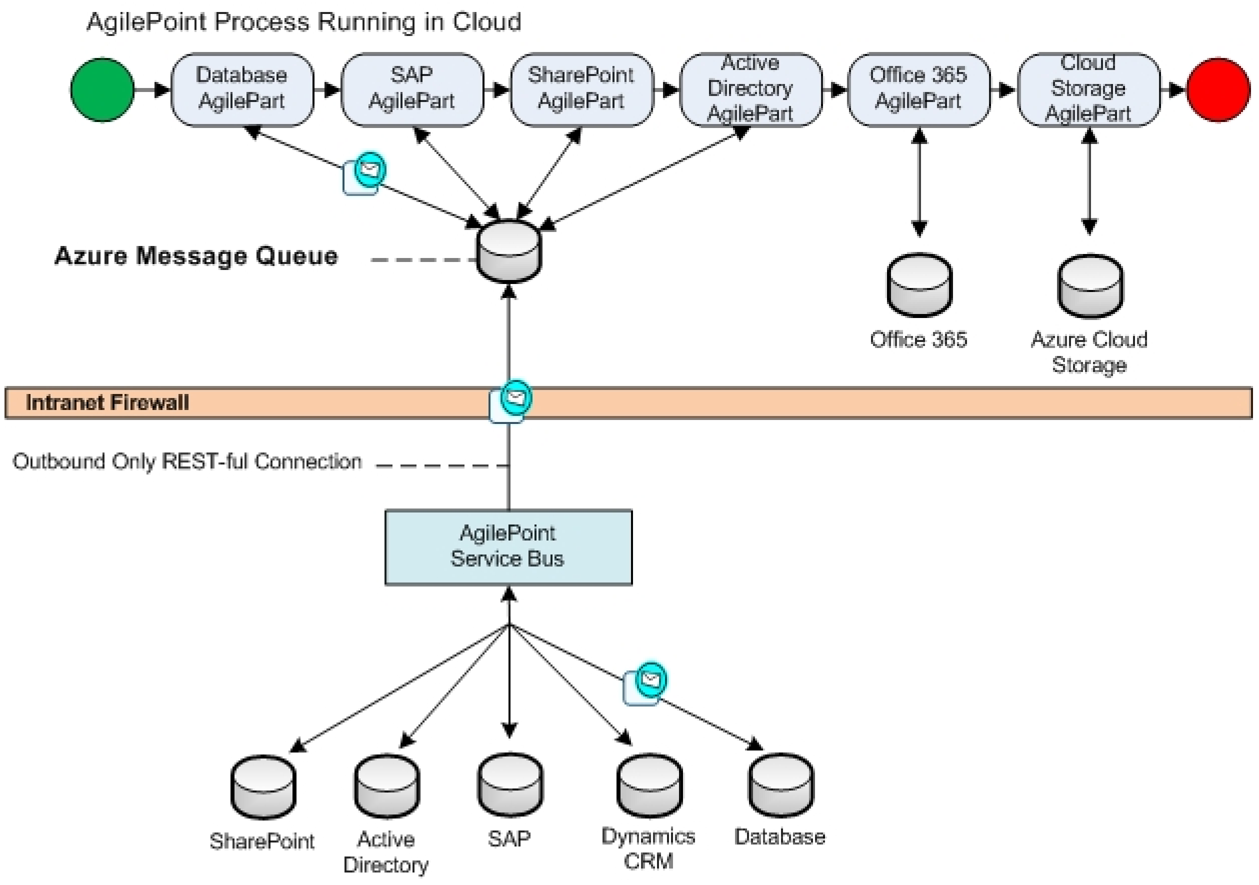
- Run as a Windows service on any .NET enabled machine.
- Run on multiple machines to support failover.
- Be configured to use multiple threads for scalability.

Message Queues

AgilePoint BPMS for Azure provides 2 message queues: the Internal Message Queue, and the Azure Message Queue. Queue Based AgileParts can be set to use either message queue, or no message queue.

Azure Message Queue

The Azure Message Queue uses a standard feature for Windows Azure that provides a common queue capability to connect a Windows Azure application to external systems.



The following are the main advantages of the Azure Message Queue:

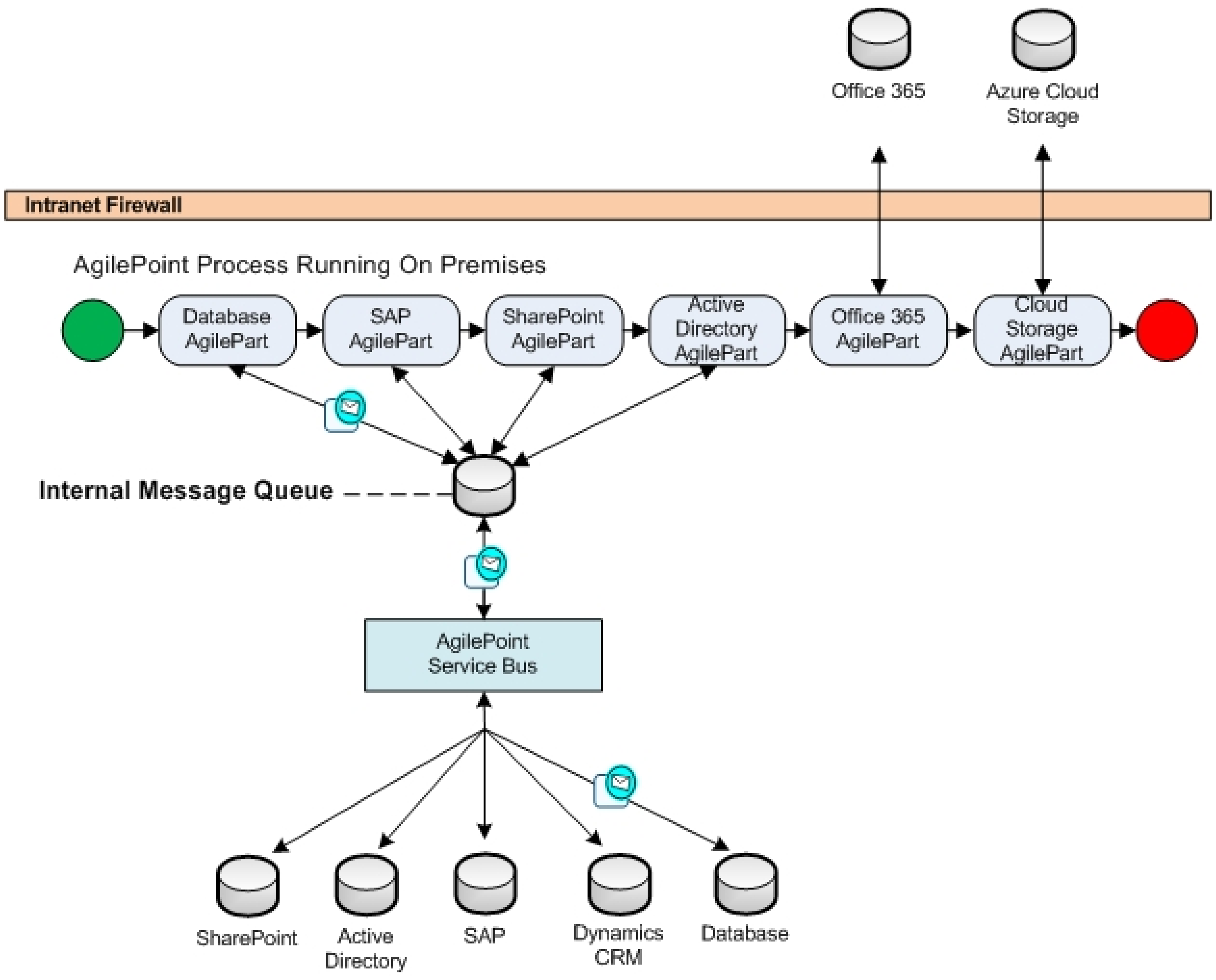
- The Azure Message Queue enables AgilePoint BPMS for Azure systems to connect to third-party systems within your intranet, thus improving the **connectivity** of a Queue Based AgilePart.
- An Internet connection is often slower and less stable than an organization's internal network connection. Decoupling AgilePoint applications in the cloud and your internal systems helps to improve the overall performance of the system.
- Messages between AgilePoint BPMS for Azure and your internal systems are fed through a single, secure port in your firewall, instead of sending individual messages directly to your internal data sources.
- Cloud messages are passed through REST-full, HTTP or SSL connection.
- The messages through cloud message queue are compressed to improve the performance.

- Messages in the cloud message queue can be encrypted using SSL.

For development machines that need access to the Azure Message Queue in a local environment, there is an Azure Message Queue emulator that runs on a local machine.

Internal Message Queue

The Internal Message Queue resides on the AgilePoint Server database. This queue can run in an on premises environment, or on AgilePoint for Azure.

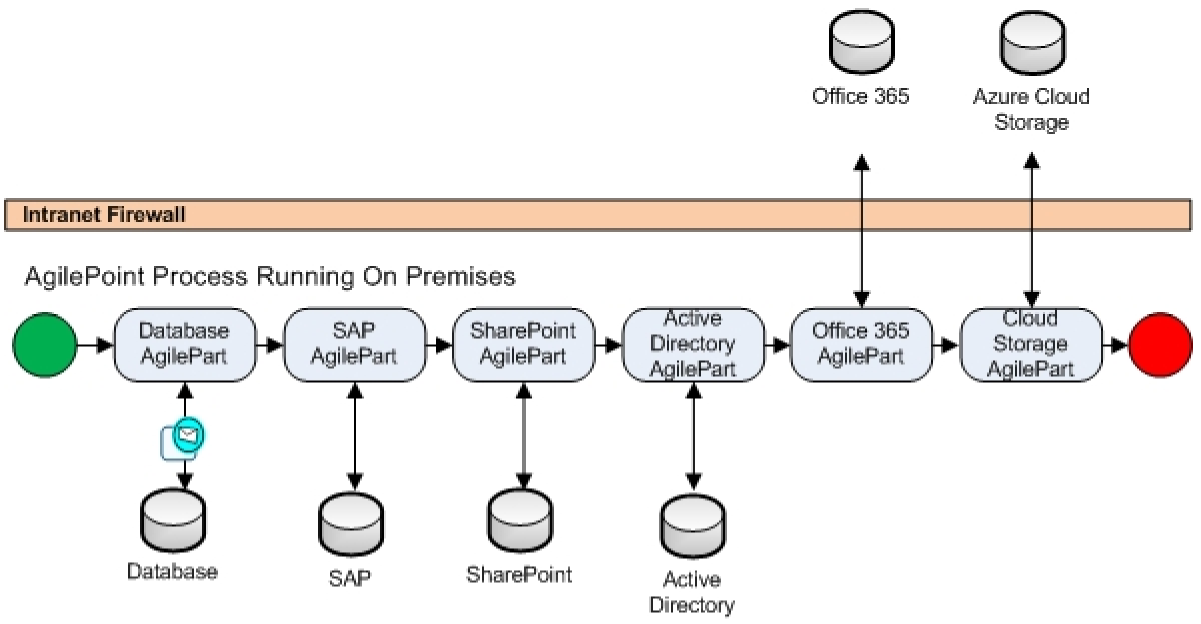


The benefit of the internal message queue is that it can move processes that take a long time out of the main processing thread, thus improving an application's **scalability**.

For example, it is common for a Facebook call to take several minutes or longer. Knowing this, you might want to use the Internal Message Queue for an AgilePart that makes a Facebook call, so that your process thread is not held up by the long processing time.

No Message Queue

Prior to the development of Queue Based AgileParts, AgileParts did not use a queue at all. Queue Based AgileParts can also be set to not use a queue. Not using a queue improves system performance because each AgilePart can connect to third-party systems independently, without an intermediary component.



Requirements

The following FAQ describes the minimal additional requirements for Queue-Based AgileParts.

What Additional Effort or Resources Are Required?

If you are using AgilePoint BPMS for Azure, you must install the AgilePoint Service Bus software component on your internal systems to manage the message queue between AgilePoint BPMS for Azure and your internal data sources.

What Other Special Considerations Are Required?

Queue-Based AgileParts do not need any further knowledge, intervention, or resources beyond those required for traditional AgileParts. In particular:

- No additional software to install in the cloud, aside from AgilePoint BPMS for Azure.
- No special network connection.
- No additional development work compared to traditional AgileParts.
- No cloud software development experience required.

Using Queue Based AgileParts with On-Premises AgilePoint

Queue Based AgileParts can be used either in AgilePoint BPMS for Azure or on premises AgilePoint. If AgilePoint Server is on premises, a built-in AgilePoint Service Bus simply runs within AgilePoint Server, and uses the Internal Message Queue.

Configuring the AgileConnector for Queue Based AgileParts

This Queue Based AgilePart AgileConnector tells AgilePoint Server whether to place the messages to and from Queue Based AgileParts in the message queue for AgilePoint for AgilePoint for Azure or On Premises AgilePoint.

Instructions

1. In a text editor, open the netflow.cfg for AgilePoint.
2. Add the following section:

```
<application name="QueuedWorkItemHandler"
        assemblyName="AgilePoint.AgileConnector.QueuedWorkItemHandler"
        className="AgilePoint.AgileConnector.QueuedWorkItemHandler.
        QueuedAutomaticWorkItemHandler">
   <ThreadPool>[number of threads]</ThreadPool>
   <TraceMode>[True or False]</TraceMode>
</application>
```

3. Restart AgilePoint Server.

Note: Only 1 AgilePoint Service Bus instance can be started on AgliePoint Server at a time.

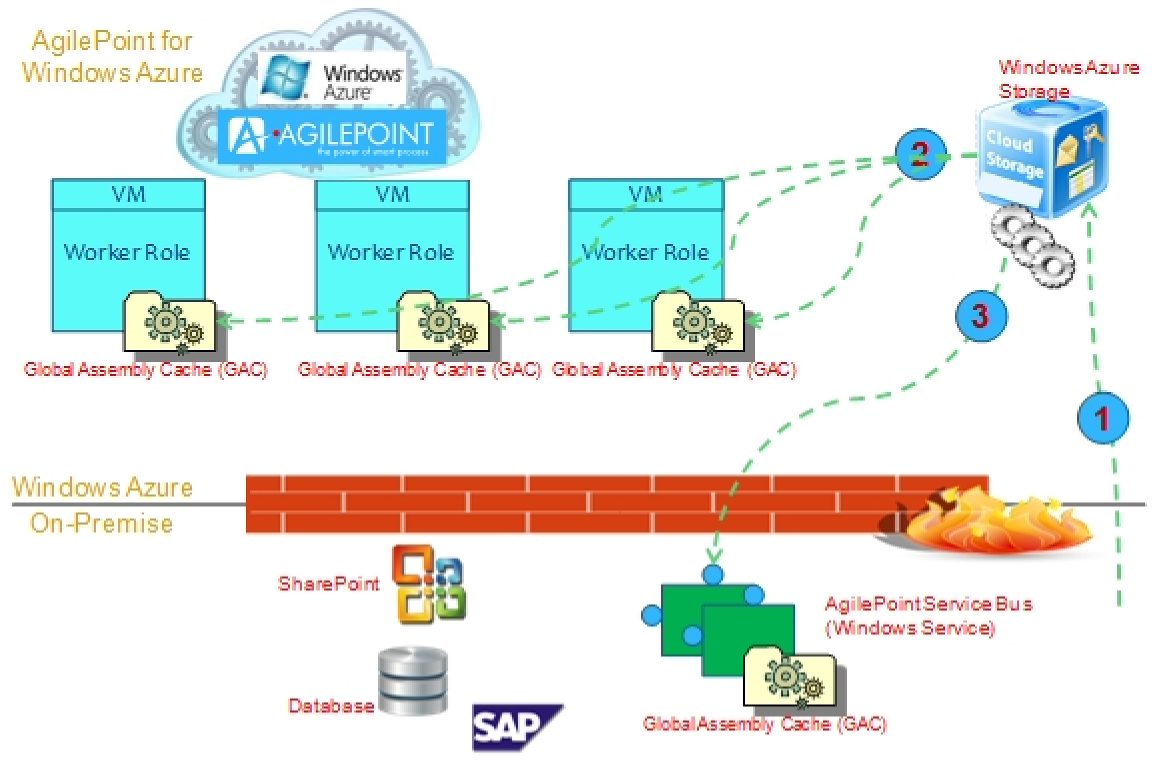
Element Definitions

Field Name	Definition
MessageQueue	Definition: Specifies whether to use the message queue to use for Queue Based AgileParts. The value of this element is only important for AgilePoint Server running on premises. Allowed Values: • **Default** - Specifies to use the default message queue. • For AgilePoint BPMS for Azure, this means use the Azure queue. • For on premises AgilePoint BPMS for Azure, this means use the build-in queue in the on premises database. • **Azure** - Specifies to use the message queue on Windows Azure message queue. • For AgilePoint BPMS for Azure, this means use the Azure queue. • Messages will go through local Windows Azure Storage Emulator. This requires

Field Name	Definition
	the AgilePoint Service Bus running in the network to read and process the messages. <u>Default Value:</u> Default

Deployment for Queue Based AgileParts

Once you create a Queue Based AgilePart, you deploy it to Azure storage. It is then picked up by components in the AgilePoint BPMS for Azure system as it is needed.


AgilePoint for Windows Azure
Windows Azure
AGILEPOINT
VM
Worker Role
VM
Worker Role
VM
Worker Role
Global Assembly Cache (GAC)
Global Assembly Cache (GAC)
Global Assembly Cache (GAC)
Windows Azure Storage
Cloud Storage
2
3
1
Windows Azure
On-Premise
SharePoint
Database
SAP
AgilePoint Service Bus (Windows Service)
Global Assembly Cache (GAC)

1. A software developer or system administrator uploads a Queue Based AgilePart to a folder in your Azure storage account.

   For more information, seePublishing a Project to AgilePoint BPMS for Azure
2. The worker roles for AgilePoint BPMS for Azure download the Queue Based AgilePart from Azure storage and publish it to the Microsoft .NET GAC. The AgilePoint BPMS engine can then load the Queue Based AgilePart when a process needs it to run.
3. When the Queue Based AgilePart needs to access an on-premises data source, the AgilePart is automatically downloaded to the GAC for the AgilePoint Service Bus.

Customizable Security

For AgilePoint BPMS for Azure production environments, you must use AgilePoint's highly customizable security framework, available from AgilePoint Developer. This framework enables AgilePoint BPMS to authenticate using your authentication mechanism.

This model has a client and server component:

- **Service proxy factory** - The client-side component.
- **Service binding factory** - The server-side component.

The customizable security framework supports WCF and REST services.

Service Proxy Factory

The Service Proxy Factory is an interface that supports connecting to AgilePoint services when a user has been authenticated using any authentication provider. The provider could be Windows authentication (the default), Windows Live ID, or a third party, such as Facebook or Google.

The Service Proxy Factory uses the user name as a surrogate for the credential for the impersonator user to determine the context for the user.

The Service Proxy Factory and Service Binding Factory are required for AgilePoint BPMS for Azure production environments.

Creating a Service Proxy Factory

To create a Service Proxy Factory, do the following.

Prerequisites

- AgilePoint BPMS for Azure.

Navigation

1. Open Microsoft Visual Studio.
2. In Visual Studio, click **New Project**.

Instructions

1. In the **New Project** window, click **AgilePoint Service Proxy Factory**.
2. Click **OK**.
3. On the **Service Proxy Factory Wizard**, complete the following fields as required, and click **OK**.

Field Name	Definition
Add Sample Custom Dialog	Specifies whether to create a configuration dialog for the Service Proxy Factory.

Field Name	Definition
Custom Dialog Type	Specifies the type of dialog to create: **Windows Form** or **Windows Presentation Foundation**.

4. Open the file **AgilePointServiceProxyFactory.cs** to set up your Service Proxy Factory. Comments are provided in the file.
5. When you generate the project, it is important to keep the default naming:
    - **Assembly name** - AgilePoint.Azure.ServiceProxyFactory.dll
    - **Class name** - AgilePointServiceProxyFactory

Note: If you want to use a custom name for Service Proxy Factory or Service Binding Factory, contact AgilePoint Professional Services.

Service Binding Factory

The binding factory is an interface that enables you to specify endpoints for your client application to connect to AgilePoint Server.

- BasicHttpBinding
- WebHttpBinding
- WsHttpBinding
- NetTcpBinding

The binding factory also defines the credentials for the impersonator user a client application uses to connect to AgilePoint Server.

Creating a Service Binding Factory

To create a Service Binding Factory, do the following.

Prerequisites

- AgilePoint BPMS for Azure.

Navigation

1. Open Microsoft Visual Studio.
2. In Visual Studio, click **New Project**.

Instructions

1. In the **New Project** window, click **AgilePoint Service Binding Factory**.
2. Click **OK**.
3. Open the file **BasicHttpBindingFactory.cs** to set up the following methods:
    - OpenBindings
    - OpenServiceHost

- GetSecurityContext

Instructions are provided in comments in the file.

Unlike the Service Proxy Factory, there is no naming restriction for Service Binding Factory.

AgileStub Project Template

This project template is used to create a new AgileStub that will customize the run time behavior of a specific process model.

An AgileStub is connected and initiated when a new process instance is initiated. An AgileStub's start and end lifecycle is one to one with the process instance life-time.

AgileConnector Project Template

This project template is used to create a new custom AgileConnector. AgileConnector assemblies must be deployed to the bin folder for AgilePoint Server.

AgilePoint Web Application Project Template

ASP.NET Web applications can be used in conjunction with AgilePoint as a foundation for workflow deployments.

This project template is used to create a new AgilePoint-enabled ASP.NET Web application that will implement the user interfaces of a specific process model based on the Generic process template from AgilePoint Envision.

If specified, it will also create mobile pages for the web application. This enables easy support for mobile devices.

Creating an AgilePoint Web Application

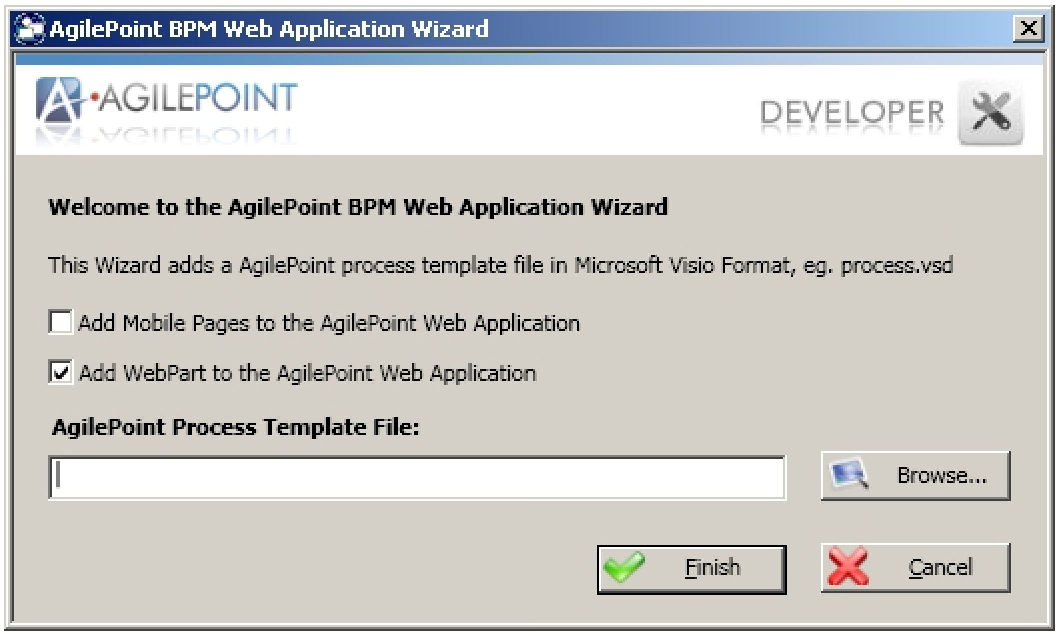
To create a new AgilePoint web application:

1. In AgilePoint Developer, click **File > New > Web Site**.
2. Specify the **Name** and **Location** for the web application, and click **OK**.
3. Complete the fields on the AgilePoint BPM Web Application Wizard.
4. Click **Finish**.

AgilePoint Developer creates the necessary skeleton in Visual Studio for the project.

AgilePoint BPM Web Application Window

This window enables you to specify the options for a web application.


AgilePoint BPM Web Application Wizard

Welcome to the AgilePoint BPM Web Application Wizard

This Wizard adds a AgilePoint process template file in Microsoft Visio Format, eg. process.vsd

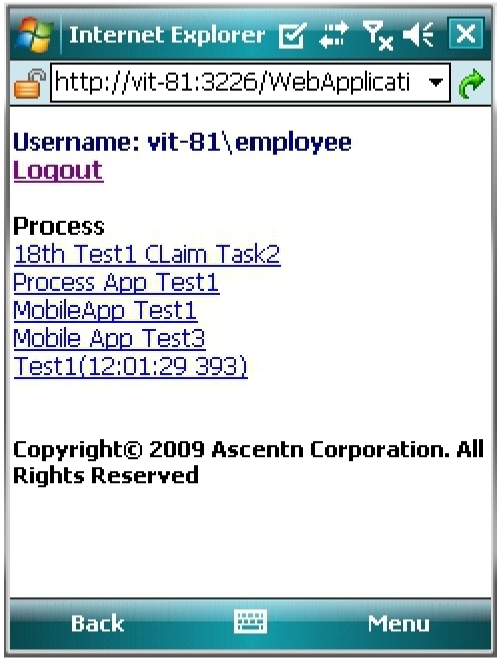
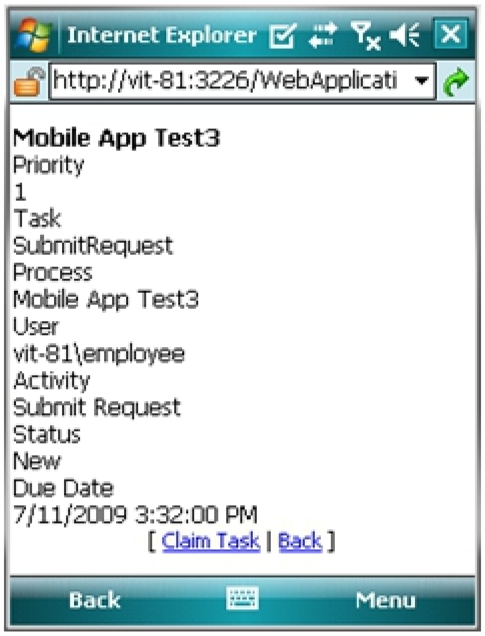
☐ Add Mobile Pages to the AgilePoint Web Application

☑ Add WebPart to the AgilePoint Web Application

AgilePoint Process Template File:

Field Definitions

Field Name	Definition
Add Mobile Pages to the AgilePoint Web Application	If selected, AgilePoint Developer creates mobile versions of all files for the web application. This simplifies development for mobile devices. AgilePoint Developer adds the following mobile files: • **Mobile_LognonForm** - A form page where the user can log on to the web application. • **Mobile_Default** - A default mobile template for an ASP.NET page. • **Mobile_TasksPage** - A mobile page that displays the user's Task List. There is also a detail page that displays details about a task when selected in the Task List. • A mobile page for the Work To Perform property of each activity that has the **EnableMobile** property set to **True**. The following images show the mobile Task List and Task Detail pages:

Field Name	Definition
	 Internet Explorer http://vit-81:3226/WebApplicati **Username: vit-81\employee** Logout **Process** 18th Test1 CLaim Task2 Process App Test1 MobileApp Test1 Mobile App Test3 Test1(12:01:29 393) **Copyright© 2009 Ascentn Corporation. All Rights Reserved** Back Menu  Internet Explorer http://vit-81:3226/WebApplicati **Mobile App Test3** Priority 1 Task SubmitRequest Process Mobile App Test3 User vit-81\employee Activity Submit Request Status New Due Date 7/11/2009 3:32:00 PM [Claim Task | Back] Back Menu
Add Web Part to the AgilePoint Web Application	If selected, AgilePoint Developer creates the following Web Parts for the web application: • Task List • Process Instance List
AgilePoint Process Template File	Specifies the process model you want to associate with the web application.

Field Name	Definition
	When you create the process template file, skeleton ASP.NET pages are created for each of the Work to Perform properties of the Manual Activities specified in the process model.

Web Application Project Template for AgilePoint BPMS for Azure

You can create ASP.NET web applications to serve as a user interface for your AgilePoint BPMS for Azure process-driven applications. Your ASP.NET applications can run on premises, in your data center, or on Windows Azure.

Creating an Azure-Enabled Web Application Project

To create an ASP.NET Web Application project for AgilePoint BPMS for Azure, do the following.

Prerequisites

- AgilePoint BPMS for Azure.

Navigation

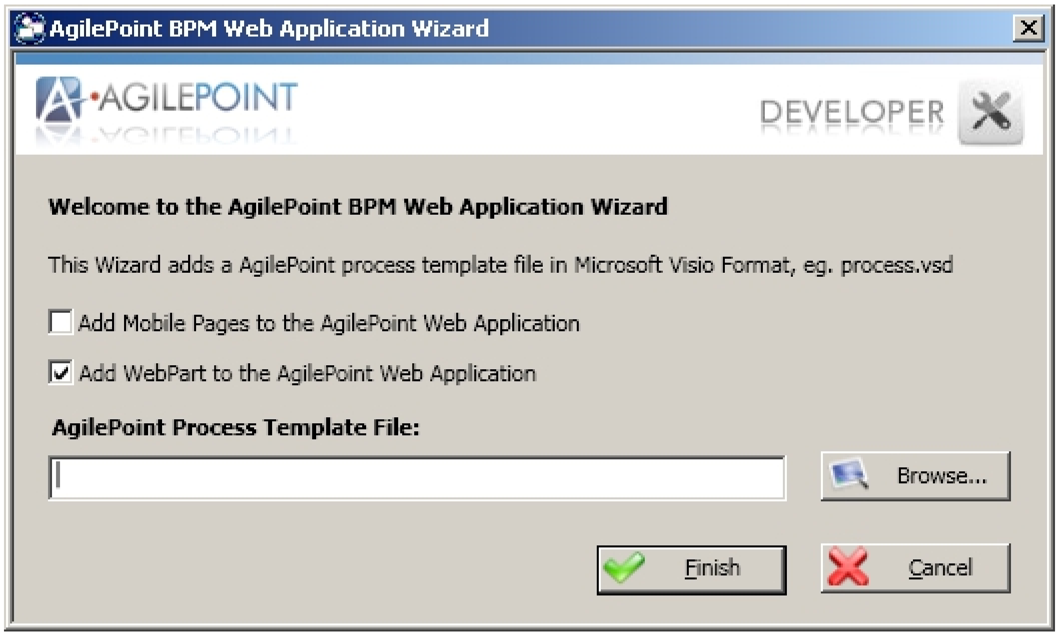
1. Open Microsoft Visual Studio.
2. In Visual Studio, click **New Project**.

Instructions

1. In the **New Project** window, click **AgilePoint Web Application for Azure**.
2. Click **OK**.
3. Complete the fields on the AgilePoint BPM Web Application Wizard.
4. Click **Finish**.

AgilePoint BPM Web Application Window

This window enables you to specify the options for a web application.



Welcome to the AgilePoint BPM Web Application Wizard

This Wizard adds a AgilePoint process template file in Microsoft Visio Format, eg. process.vsd

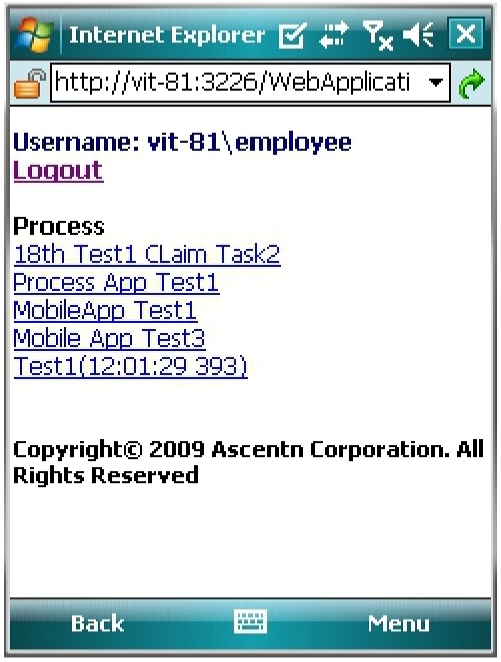
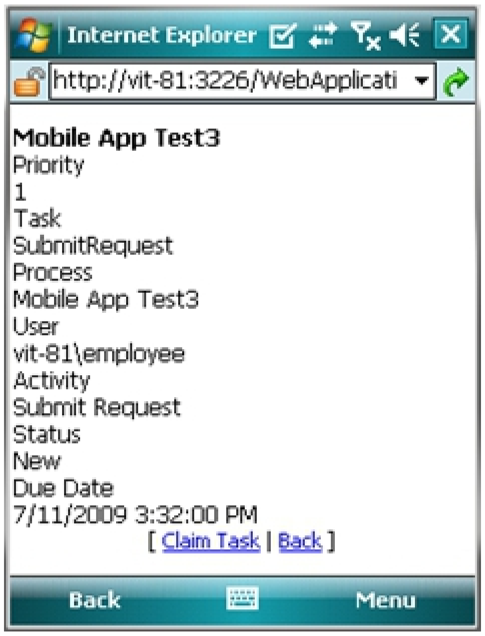
☐ Add Mobile Pages to the AgilePoint Web Application

☑ Add WebPart to the AgilePoint Web Application

AgilePoint Process Template File:

Field Definitions

Field Name	Definition
Add Mobile Pages to the AgilePoint Web Application	If selected, AgilePoint Developer creates mobile versions of all files for the web application. This simplifies development for mobile devices. AgilePoint Developer adds the following mobile files: • **Mobile_LognonForm** - A form page where the user can log on to the web application. • **Mobile_Default** - A default mobile template for an ASP.NET page. • **Mobile_TasksPage** - A mobile page that displays the user's Task List. There is also a detail page that displays details about a task when selected in the Task List. • A mobile page for the Work To Perform property of each activity that has the **EnableMobile** property set to **True**. The following images show the mobile Task List and Task Detail pages:

Field Name	Definition
	 Internet Explorer http://vit-81:3226/WebApplicati Username: vit-81\employee Logout Process 18th Test1 CLaim Task2 Process App Test1 MobileApp Test1 Mobile App Test3 Test1(12:01:29 393) Copyright© 2009 Ascentn Corporation. All Rights Reserved  Internet Explorer http://vit-81:3226/WebApplicati Mobile App Test3 Priority 1 Task SubmitRequest Process Mobile App Test3 User vit-81\employee Activity Submit Request Status New Due Date 7/11/2009 3:32:00 PM [Claim Task | Back]
Add Web Part to the AgilePoint Web Application	If selected, AgilePoint Developer creates the following Web Parts for the web application: • Task List • Process Instance List
AgilePoint Process Template File	Specifies the process model you want to associate with the web application.

Field Name	Definition
	When you create the process template file, skeleton ASP.NET pages are created for each of the Work to Perform properties of the Manual Activities specified in the process model.

Publishing a Project to AgilePoint BPMS for Azure

To publish an AgilePoint Developer project to AgilePoint BPMS for Azure, do the following.

Prerequisites

- An AgilePoint Developer project enabled for AgliePoint BPMS for Azure, which has been set up and built in Microsoft Visual studio.

Navigation

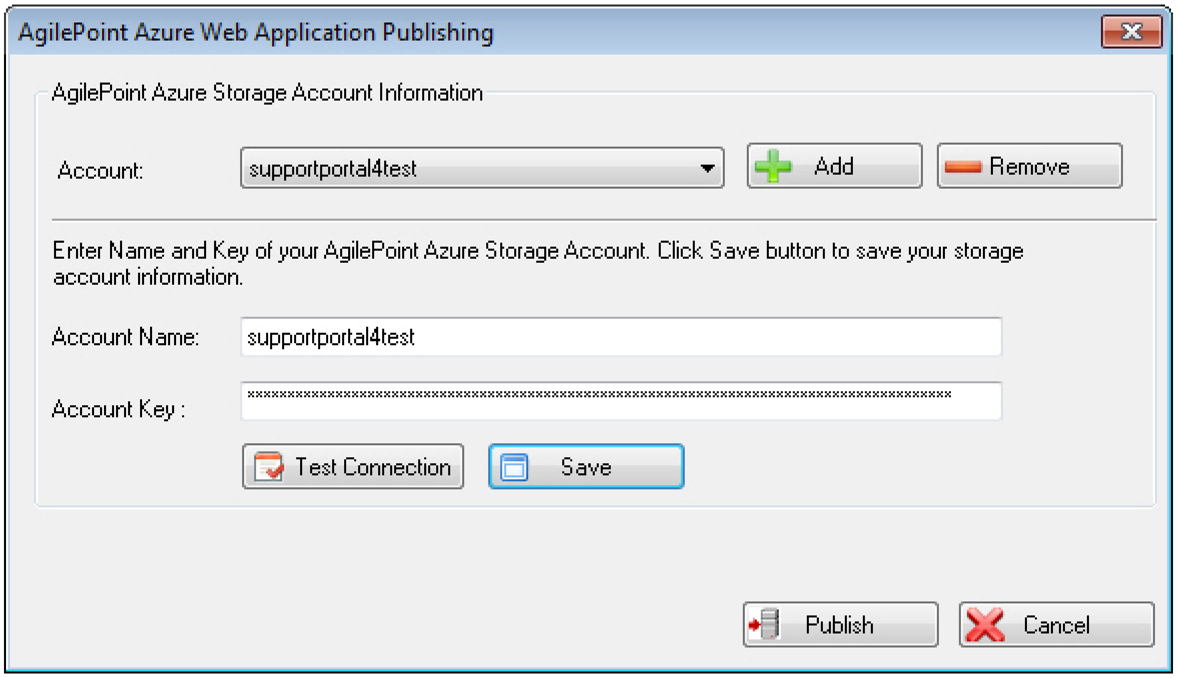
1. In Visual Studio, open and build your ASP.NET web application that is enabled for AgilePoint BPMS for Azure.

Instructions

1. In the **Solution Explorer** pane, right-click the project name, and click **Publish to AgilePoint Azure Storage**.
2. Complete the fields on the AgilePoint Azure Web Application Publishing window.
3. Click **Publish**.

AgilePoint Azure Web Application Publishing

Specifies the storage account to which to publish an ASP.NET Web Application project for AgilePoint BPMS for Azure.


AgilePoint Azure Web Application Publishing

AgilePoint Azure Storage Account Information

Account: supportportal4test

Enter Name and Key of your AgilePoint Azure Storage Account. Click Save button to save your storage account information.

Account Name: supportportal4test

Account Key : **

Navigation

1. In Visual Studio, open and build your ASP.NET web application that is enabled for AgilePoint BPMS for Azure.
2. In the **Solution Explorer** pane, right-click the project name, and click **Publish to AgilePoint Azure Storage**.

Field Definitions

Field Name	Definition
Account	Definition: Azure storage account to which to publish the web application. Allowed Values: A valid Azure storage account name. Default Value: None
Account Name	Definition: Specifies the name of your Azure storage account.. Allowed Values: A valid Azure storage account name.

Field Name	Definition
	<u>Default Value:</u> None
Account Key	<u>Definition:</u> Specifies the account key for your Azure storage account. <u>Allowed Values:</u> A valid Azure storage account key. <u>Default Value:</u> None

Deploying a Project to On-Premises AgilePoint BPMS

To publish an AgilePoint Developer project to on-premises AgilePoint BPMS, do the following.

Prerequisites

- An AgilePoint Developer project that has been set up and built in Microsoft Visual studio.

Navigation

1. In Visual Studio, open and build your ASP.NET web application.

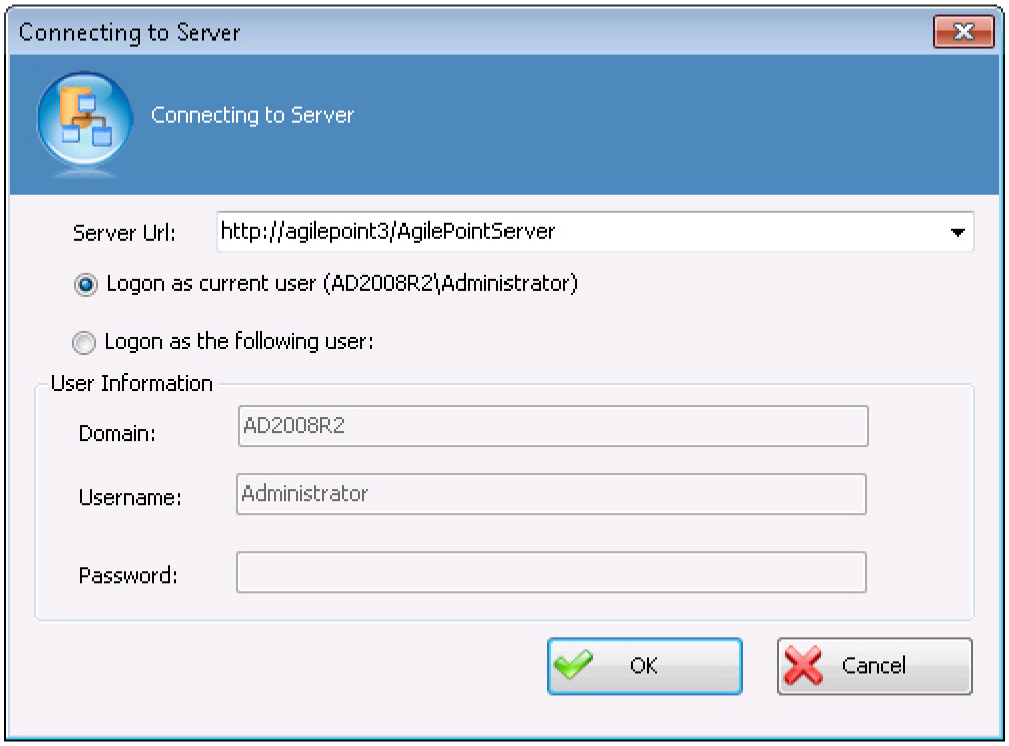
Instructions

1. In the **Solution Explorer** pane, right-click the project name, and click **Deploy to AgilePoint Server On Premises**.
2. Complete the fields on the Connecting to Server window.
3. Click **OK**.
4. On the **Open** window, select the assembly you want to deploy, and click **Open**.

   The assembly is deployed to AgilePoint Server.

Connecting to Server Window

Enables AgilePoint Developer to bind and authenticate to AgilePoint Server.


Connecting to Server

Connecting to Server

Server Url: http://agilepoint3/AgilePointServer

(●) Logon as current user (AD2008R2\Administrator)

( ) Logon as the following user:

User Information

Domain: AD2008R2

Username: Administrator

Password:

Navigation

1. In Visual Studio, open and build your ASP.NET web application.
2. In the **Solution Explorer** pane, right-click the project name, and click **Deploy to AgilePoint Server On Premises**.

Field Definitions

Field Name	Definition
AgilePoint Server URL	Definition: The URL of your AgilePoint Server. Allowed Values: A valid AgilePoint Server URL. Default Value: In AgilePoint BPMS for Azure, the default value is the value of the default AgilePoint evaluation environment. Custom Attributes: No

Field Name	Definition
Logon as current user	Authenticates using the local machine's Windows user account. If you select this option, you do not need to complete the remaining fields.
Logon as the following user	Authenticates using the credentials you specify.
Domain	Definition: The authentication domain. This does not apply to Forms-Based Authentication.
Username	Definition: The user ID of the AgilePoint Server authentication account. Allowed Values: A valid user name for an AgilePoint Server administrator account. Default Value: None Custom Attributes: No
Password	The password for the authentication account.

AgilePoint Web Controls

The following information provides details on how to leverage the built-in AgilePoint Web Controls to create process-enabled Web Applications.

The main difference between the AgilePoint Web Controls and the standard ASP.NET Web Controls is the additional AgilePoint specific properties that allow you to specify a binding name. With the AgilePoint Web Controls no code is needed to save and retrieve the values entered by the user. AgilePoint will automatically save the values to a custom attribute that can be accessed from anywhere in the process.

Adding AgilePoint Web Controls to Your Project

1. In Microsoft Visual Studio, click the Toolbox button.
2. Right-click in the Toolbox and click Add Tab.
3. Name the new tab AgilePoint Web Controls (or your preferred name).
4. Right-click on the AgilePoint Web Controls tab and click Choose Items.
5. From the .NET Framework Components tab, deselect all the checked items and scroll down and select all the AgilePoint Workflow controls. The AgilePoint Workflow controls can be identified in the list with the Namespace Ascentn.Workflow.WebControls.
6. Click OK.
7. Expand the AgilePoint Web Controls tab. The list of AgilePoint Web Controls are displayed. You can now make use of these Web Controls to build your .aspx pages.

ConfirmButton

The implementation of this Web Control allows you to call your custom JavaScript function and post back to the server to execute managed code.

Properties

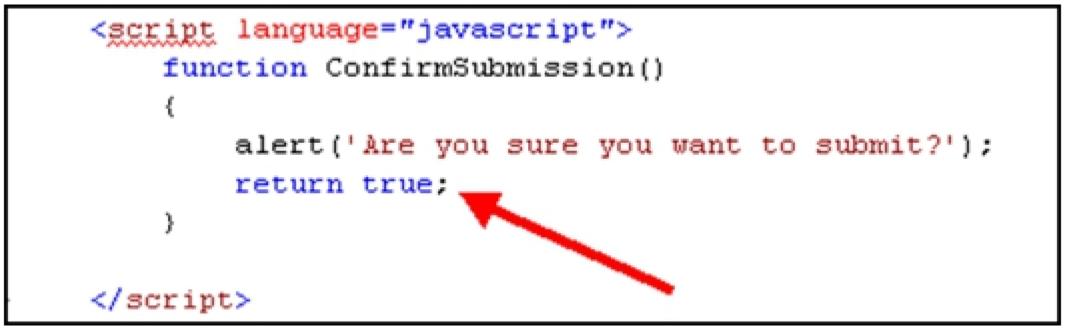
Field Name	Definition
onBeforeSubmit	Specifies the name of the JavaScript function. A () is required at the end.

Inherits

- System.Web.UI.WebControls.Button

Usage

- Insert similar JavaScript as below in the .aspx file. The last line of the JavaScript function must include "return true" in order to run server-side post back. If not, after the confirmation message dialog is displayed, no action will be taken.



```
<script language="javascript">
    function ConfirmSubmission()
    {
        alert('Are you sure you want to submit?');
        return true;
    }

</script>
```

- The ConfirmButton Web Control does not support the ASP.NET RequiredFieldValidator control. This is a limitation of this AgilePoint Web Control, if you have RequiredFieldValidators setup for other input Web Controls on the page, then ConfirmButton will not support the validation action.

DatePicker

The implementation of this Web Control provides a convenient way for the user to pick a date.

Properties

Field Name	Definition
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control. Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.
Date	The default value of the Date Picker.

Inherits

- System.Web.UI.Control
- INamingContainer
- WFDataBindingControl

Usage

Not applicable.

WFcheckbox

The implementation of this Web Control shows a check box on the page at run time.

Properties

Field Name	Definition
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control. Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.

Inherits

- System.Web.UI.WebControls.check box
- WFDataBindingControl

Usage

Not applicable.

WFFileAttachment

The implementation of this Web Control uploads a single file, either to the system directory or SharePoint during run time.

If the file already exists, information about the file is shown and the delete option is presented.

Properties

Field Name	Definition
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control.

Field Name	Definition
	Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.
FileExtension	Determines the file extension type (e.g. .jpg, .jpg, .doc). If this property is left null, it matches any type of file.
Location	When uploading a file to the File System, the default path is set to the root directory. A relative path is accepted (e.g. /DirectoryName). If you forget to type the full path, the file will automatically be uploaded to the root directory. When uploading to SharePoint, enter the SharePoint URL (e.g.: http: //[hostname]:8080/DemoDirectory).
OverWrite	Determines whether the file will be overwritten if it already exists.
Repository	Staging area for the file.

Inherits

- System.Web.UI.Design.ControlDesigner

Usage

- As a general rule do not use the & character in file names. If the uploading file name contains an "&" character, it will be changed to "_", because the "&" is the XML parser's primary key.
- The user who uploaded the file is the only user who can see the information in the file or delete the file.
- An empty file cannot be uploaded.

WFRadioButton

The implementation of this Web Control shows a radio button on the page at run time.

Properties

Field Name	Definition
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control.

Field Name	Definition
	Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.

Inherits

- System.Web.UI.WebControls.RadioButton
- WFDataBindingControl

Usage

- Usually WFRadioButton is used to bind with Single Condition. The data type of WFRadioButton is Boolean only. Single Condition only supports Boolean data type.
- Only set the binding name to the radio button that represents 'yes' or 'true'. For the other radio button in the group (the one represents 'no' or 'false'), do not set the binding name/leave the binding name property empty. If you set the binding name for both radio buttons in the group, the auto-binding does not work. See ManagerApproval.aspx for example.
- WFRadioButton is not the same as WFRadioButtonList. WFRadioButtonList only supports string data type. Do not use WFRadioButtonList to bind with a Single Condition, rather use a Multiple Condition AgileShape.

WFDropDownList

The implementation of this Web Control shows a drop-down list for which to select a value on the page at run time.

Properties

Field Name	Definition
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control. Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.

Inherits

- System.Web.UI.WebControls.DropDownList
- WFDataBindingControl

Usage

Not applicable.

WFGridView

The implementation of this Web Control shows the associated XML Schema content in a repeating table view at run time. Add, update and delete is enabled.

Properties

Field Name	Definition
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control. Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.

Inherits

- System.Web.UI.WebControls.GridView
- WFDataBindingControl

Usage

After Setting the BindingName property, add the desired fields in accordance with the XML schema.

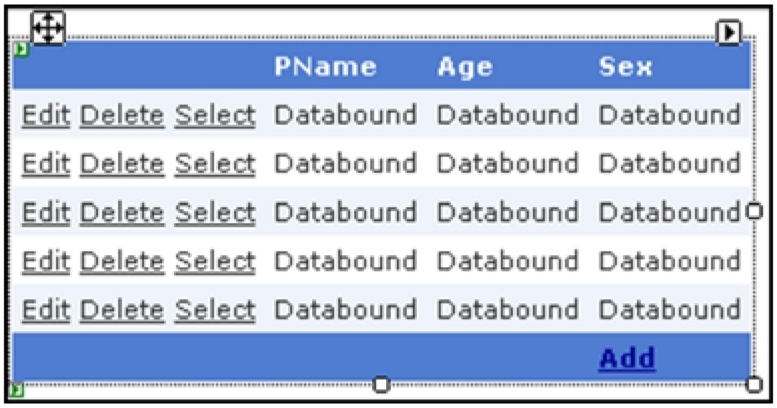
The structure of selected XML node must be similar to the following:

```
<Persons>
        <Person>
                <PName>Tom</PName>
                <Age>23</Age>
                <Sex>Male</Sex>
        </Person>
        <Person>
                <PName>Jack</PName>
                <Age>25</Age>
                <Sex>Male</Sex>
        </Person>
</Persons>
```

The value of AutoGenerateColumns property should not be true; otherwise it will generate an error.

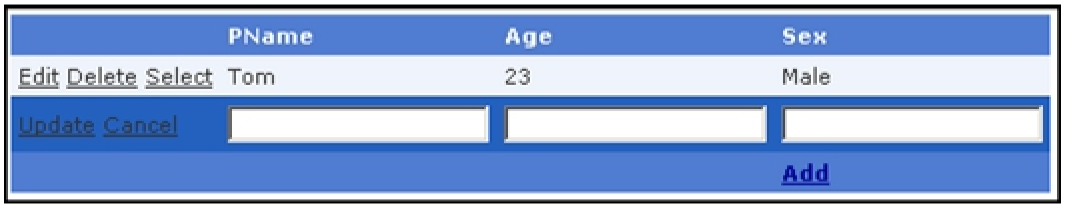
Button can be added, updated or deleted by setting the CommandName. Without coding in the event of RowUpdated and RowDeleted, WFGridView will save the changes to the data source automatically.

Add a **Submit** button that calls GetBoundDataItem() to save the changes to XML.



	PName	Age	Sex
Edit Delete Select	Databound	Databound	Databound
Edit Delete Select	Databound	Databound	Databound
Edit Delete Select	Databound	Databound	Databound
Edit Delete Select	Databound	Databound	Databound
Edit Delete Select	Databound	Databound	Databound
			Add

At run time the Web Control shows the XML content.



	PName	Age	Sex
Edit Delete Select	Tom	23	Male
Update Cancel			
			Add

Click the corresponding button to add, update, or delete the repeating section.

Click the **Submit** button, all changes will be saved to AgilePoint Server.

WFTaskGridControl

The implementation of this Web Control shows a user Task List on the page at run time. The AgilePoint Task List supports the following functions:

- Pool Function (not in column)
- Reassign Task
- Cancel Task
- Multiple Complete
- Multiple Cancel
- Rollback
- At design time, columns must be added manually, as a result the value of the AutoGenerateColumns property must be set to False, otherwise it will throw an error.
- At run time, you can view the tasks in the task grid view.
- You can do sorting and grouping by clicking the title.
- Upon selecting the check box for a task, the row will become bold.
- When right-clicking on a row, a menu appears with several functions.

- You can format the style of grid view as you want by setting the grid view's property or "auto Format" command.

Properties

Field Name	Definition
AutoGeneratecheckboxColumn	Determines whether a check box column will automatically be generated at run time.

ContextMenuCssClass	This property enables the inclusion of a CSS style sheet. The following is an example.
	`.RightMenu`
	`{`
	`    border-right: 2px outset;`
	`    border-top: 2px outset;`
	`    border-left: 2px outset;`
	`    border-bottom: 2px outset;`
	`    background-color: buttonface;`
	`}`
	`.RightMenu hr`
	`{`
	`    width: 100px;`
	`}`
	`.RightMenu ul`
	`{`
	`    list-style: none; margin:0;`
	` padding:0;`
	`}`
	`.RightMenu ul li`
	`{`
	`    vertical-align: bottom;`
	`}`
	`.RightMenu A { color: MenuText;`
	`    text-decoration: none; display:`
	`    block; width: 100px;`
	`    text-align:center; line-`
	`height:20px }`
	`.RightMenu A:link { color: MenuText;`
	`    text-decoration: none; }`
	`.RightMenu A:active { color: MenuText;`
	`    text-decoration: none; }`
	`.RightMenu A:visited { color: MenuText;`
	`    text-decoration: none; }`
	`.RightMenu A:hover { color:`
	` HighlightText;`
	`    background-color: Highlight; }`

ContextMenus	The ContextMenus property provides an editor for maintaining the menu items. To separate the menu items, enter <hr /> into the Text property. If you set the BoundCommandName property, you must set a ButtonField in columns property which is for the command of the menu item.

Field Name	Definition
	If you set the NavigateUrl you can use for example: ProcessView.aspx?PIID={ProcInstID} in the {} is the fields in DataKeyNames property. And if desired the URL will open in a new window by setting the target property to _blank.
GroupBackColor	Determines the background color of the title bar when grouping.
GroupFontColor	Determines the font of the title bar when grouping.

Inherits

- System.Web.UI.WebControls.GridView

Usage

You must create a QueryTaskListData event by double-clicking the QueryTaskListData property.

Here is some sample code in the QueryTaskListData Event.

```
IWFWorkflowService api = GetAPI();
        string statusList = string.Format("{0};{1};{2}",
                WFManualWorkItem.ASSIGNED, WFManualWorkItem.OVERDUE,
                WFManualWorkItem.PSEUDO);
        WFManualWorkItem[] wks =
                api.GetWorkListByUserID(GetCurrentUser(), statusList);

        if (wks == null) wks = new WFManualWorkItem[0];
        TaskGridControl.QueryTaskListEventArgs qArgs =
                e as TaskGridControl.QueryTaskListEventArgs;
qArgs.Data = wks;
```

You can write TaskGridControl1.Binding(); in the update page function.

If you have set a command in the Context Menu, you need to design the RowCommand event by double-clicking the RowCommand property.

Below is sample code:

```
int index;
        if (!int.TryParse(e.CommandArgument as string,
                out index)) return;
        DataKey data = TaskGridControl1.DataKeys[index];
        string wid = data.Values["WorkItemID"] as string;
        string TaskForm = data.Values["Name"] as string;
        string pid = data.Values["ProcInstID"] as string;
        switch (e.CommandName)
        {
            case "DoTask":
                Response.Redirect("../" + TaskForm +
                        ".aspx?WID=" + wid);
                break;
            case "ViewProcess":
                string url = "ProcessViewer.aspx?PIID=" + pid;
                string script = @"<script
                        language='javascript'> " +
                        @"var szFeatures = 'scrollbars=yes,
                        width=770,height=500,resizable=yes';" +
                        @"window.top.open('" + url + "',
                        'ProcessViewer', szFeatures);" + @"
                        </script> ";
                Page.ClientScript.RegisterStartupScript(Page.GetType(),
                        "", script);
                break;
            case "CreateLinkWorkItem":
                IWFWorkflowService api = base.GetAPI();
                api.CreateLinkedWorkItem(wid, TaskForm,
                        "Administrator", null, null);
                break;
            case "Test":
                int[] i = this.TaskGridControl1.GetSelectedIndices();
                IWFWorkflowService api1 = base.GetAPI();
                foreach (int s in i)
                {
                    DataKey d = TaskGridControl1.DataKeys[s];
                    string id = data.Values["WorkItemID"] as string;
                    //api1.CompleteWorkItem(id);
                }
                break;
            default:
                break;
        }
        UpdatePage();
```

Note: If you want to get the rows which check box selected you just need to call the function like this: int[] i = this.TaskGridControl1.GetSelectedIndices();

WFTextBox

The implementation of this Web Control shows a text box for entering data on the page at run time.

Properties

Field Name	Definition
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control. Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.

Inherits

- System.Web.UI.WebControls.TextBox
- WFDataBindingControl

Usage

Not applicable.

WFProcessTemplateDropDownList

The implementation of this Web Control shows a drop-down list to allow the user to select a process template that is saved to a custom attribute or XML field that is defined in the Binding Name property.

Properties

Field Name	Definition
AppNames	The process templates to be returned from the database and shown in the drop-down list that match the Application Name (defined in AgilePoint Envision process template **Application** property). If AppNames is left blank, it will return all the process templates from the database. There is filtering capability, for example if you have 1000 process templates, you can type an Application Name into this property or use wildcard characters (e.g. *Application). **Note:** AgilePoint will not save the value selected from the drop-down list into a custom process attribute.
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control. Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.

Inherits

- WFDropDownList

Usage

Not applicable.

WFComment

The implementation of this Web Control records and displays comments on the page. It consists of two parts, one is a text box where the user can input comments, and the other is a display box where comment history is shown. The comment history box will be hidden if previous comments do not exist.

If the user wants to input special characters in the comment, the page attribute ValidateRequest should be set to **false**:

Properties

Field Name	Definition
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control. Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.
ShowCommentList	Determines whether the comment list is visible or not (i.e. True/False).
CommentBackColor	Determines the background color of the comment title bar.
CommentCaption	Determines the caption of the comment title bar.
CommentTitleAlign	Determines the caption alignment in the comment title bar.
BarColor	Determines the color of the separator bar in the comment list at run time.
BarHeight	Determines the height of the separator bar in the comment list at run time.
CommentListBackColor	Determines the background color of the comment list title bar.
CommentListCaption	Determines the caption of the comment list title bar.
CommentListTitleAlign	Determines the caption alignment in the comment list title bar.
CustomFormat	Determines the style of appearance of Comment Name and Date. Parameter {0} represents date and

Field Name	Definition
	{1} represents username, you can change the format as desired, but the two parameters cannot be omitted.
Height	Determines the height of the comment list. The minimum height should be 100px.
Width	Determines the width of the comment list. The minimum width is 400px.

Inherits

- System.Web.UI.WebControls.CompositeControl
- WFDataBindingControl

Usage

Not applicable.

WFRadioButtonList

The implementation of this Web Control displays multiple radio buttons on the page.

Properties

Field Name	Definition
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control. Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.

Inherits

- System.Web.UI.WebControls.RadioButtonList
- WFDataBindingControl

Usage

WFRadioButtonList only supports string data type. Do not use WFRadioButtonList to bind with a Single Condition, rather use a Multiple Condition.

WFListBox

The implementation of this Web Control displays a list box for which to select a value on the page.

Properties

Field Name	Definition
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control. Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.

Inherits

- System.Web.UI.WebControls.Listbox
- WFDataBindingControl

Usage

Not applicable.

WFUserNameDropDownList

The implementation of this Web Control displays drop-down list for which to select a User Name on the page.

Properties

Field Name	Definition
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control. Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.

Field Name	Definition
Filter	Allows a wildcard for user full name, GROUP(groupName) ROLE(roleName).

Inherits

- System.Web.UI.WebControls.DropDownList
- WFDataBindingControl

Usage

Not applicable.

WFRadioButtonPair

The implementation of this Web Control displays a pair of radio buttons (e.g. Approve/Reject).

Properties

Field Name	Definition
Alignment	Determines whether the radio buttons are displayed vertical or horizontal.
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control. Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.
Decorate	Determines whether Approve/Reject or Yes/No is shown.

Inherits

- System.Web.UI.WebControls.WebControl
- WFDataBindingControl

Usage

This Web Control is recommended to be used with a Single Condition AgileShape.

WFUserNameListBox

The implementation of this Web Control displays a list box that includes a list of user names for which the user can make a selection.

Properties

Field Name	Definition
BindingName	The BindingName property allows you to type in a name that becomes the custom attribute name. You can also select the XPath of the data element contained within the XML Schema file for which to bind this control. Click on the **BindingName** property, the **Select XPath from XML Schema** window appears. Choose an XPath node for which to bind and click **OK**.
Filter	Allows a wildcard for user full name, GROUP(groupName) ROLE(roleName).

Inherits

- System.Web.UI.WebControls.ListBox
- WFDataBindingControl

Usage

Not applicable.

WFProcessViewer

The implementation of this Web Control shows the real-time status of a process instance on the page.

The control displays the process image and the status of activities in the AgilePoint process. A pop-up displays activity information for the current running activity.

Properties

Field Name	Definition
ProcessImageCacheFolder	The folder on the machines file system that will be used to store the process image. If a folder is not already created, one will be created for you. You can name the folder as desired. This folder path must be to a sub folder and formatted as a relative path to the

Field Name	Definition
	AgilePoint Web Application virtual folder. It cannot be a special site folder, such as **App_Code** or **Bin**. **Note:** The path must start with ~/. **Note:** Only one process image file will be created for one released process model.

Inherits

- System.Web.UI.WebControls.WebControl
- System.Web.UI.ICallbackEventHandler

Usage

- A single page can not contain more than one WFProcessViewer.
- It is required to drag the WFProcessViewer control into a .aspx page that inherits the Ascentn.Workflow.WebControls.WFWorkSheetPage class.

Task List Web Part for AgilePoint Web Applications

The AgilePoint ASP.NET web application template provides a page with an ASP.NET version of the Task List Web Part.

Similar to the SharePoint Task List Web Part, the **Task** name opens a drop-down with a list of task management features.

Users can also reassign a selected task or cancel a selected task or process using the **Global actions** list.

Manual Setup

The following steps must be completed before running an ASP.NET web application with the Task List Web Part:

1. Set up the personalization database for the Web Part. Execute the command aspnet_regsql as follows:

   ```
   "C:\WINDOWS\Microsoft.NET\Framework\v2.0.50727\aspnet_regsql.exe"
        -E -S [hostname] -A all
   ```

   Where [localhost] points to the default MS SQL Server instance running on the local machine.

   Running this command creates a database with name **aspnetdb**. This can be changed to any value and the same should be reflected in the connection string in the next step. This step is a one-time execution as the personalization database will work for multiple Web Parts.

2. Open the web.config file and replace the connection string **LocalSqlServer** under the <connectionStrings> node with the location you defined in the previous step:

   ```
   <add name="LocalSqlServer"
        connectionString="Data Source=[hostname];Initial
        Catalog=aspnetdb;trusted_connection=true"
        providerName="System.Data.SqlClient" />
   ```

Settings

You can modify the following settings.

Linking to Tasks in Multiple Process Templates

By default, the Task List Web Part displays the options **Open Task** and **Take Assignment and Open Task** for the process instance whose process template is attached with the AgilePoint Web Application. This is defined in the following setting in the web.config:

```
<add key="ProcessModel" value="LeaveRequest_ASP" />
```

If you want to add additional process templates, you can add the process template names to the **value** property of this node, separated by semicolons (;).

Note: Task List Web Part will not display **Open Task** or **Take Assignment and Open Task** option if the file does not exist in the specified path.

Linking to Tasks in Multiple Web Applications

You can also display the **Open Task** option for other web applications. Add the following **appSetting** node. The **key** is the name of the process template. The **value** is the base URL of the web application.

```
<add key="BudgetRequest" value="http://[hostname]:[port]/BudgetRequest"/>
```

Setting the Maximum Number of Tasks to Display

You can modify the number of tasks to display on the Task List Web Part using the following appSettings tag entry:

```
<add key="PageSize" value="5" />
```

The default value is 5, but this number can be changed.

Filter Tasks by Application Name

Tasks can be filtered based on the application name using the following appSettings tag entry:

```
<add key="ExcludedApplication" value="SPSIntegration" />
```

This can accept multiple values with semicolon (;) separated application names.

Show or Hide the Complete Selected Tasks Option

You can show or hide the Complete Selected Tasks option in the Global Tasks menu by changing the value of the following appSettings node:

```
<add key="ShowCompleteSelectedTask" value="false" />
```

Limitations

The following limitations apply to the Task List Web Part:

- While grouping, the user cannot filter, and vice versa.
- The user cannot group the Select and Tasks columns.

Exception Handling for AgileParts

AgilePoint makes available to customers options for flexibility in exception handling for out-of-the-box, and custom AgileParts by enabling the ability to implement custom logic. For out-of-the-box exception handling functionality, AgilePoint writes AgilePart exception information (e.g. error message and status) to custom attributes to be viewed via AgilePoint Enterprise Manager or reported on later. The following information will show how to extend the out-of-the-box exception handling capabilities for AgileParts to call on a custom exception handler AgileConnector to perform more refined custom exception handling.

AgilePoint Envision includes a drop-down property called **ExceptionHandlerScope** for AgileParts that enables the user to select either **Local** or **Global**. If **Local** is selected, the exception handling will be handled as normal by reporting exception data to the **SaveErrorMessageTo** and **SaveStatusTo** custom attributes. If **Global** is selected, the AgilePart will continue to process as normal by reporting exception data to the **SaveErrorMessageTo** and **SaveStatusTo** custom attributes, but also includes the extended ability to call on a custom AgileConnector with custom exception handling logic to handle the exception(s) as desired. Now users have the extended ability to implement whatever logic is required to handle exceptions (e.g. custom error handling based on error message or error code, database interactivity, etc.).

Note: To use the Global Exception Handling option, you do not need to do anything extra in the AgilePart code, the implementation is determined at the process model layer as shown below.

Developing the Custom Exception Handler AgileConnector

1. Open AgilePoint Developer and create a new .NET Class Library project. Add the following 3 .NET References:
   - System.EnterpriseServices
   - System.Web
   - System.Web.Services
2. Add the following AgilePoint References:
   - Ascentn.Workflow.Share
   - Ascentn.Workflow.WFBase
3. Include the **Using Clause**: Ascentn.Workflow.Base
4. Implement the Base Class.
5. Implement the custom code.

For more information, see AgileConnector Sample for Custom Exception Handling in the Documentation Library.

Deploying the Custom Exception Handler AgileConnector

1. Copy the assembly (e.g. CustomExceptionHandler.dll) to the bin folder of AgilePoint Server.
2. Open the AgilePoint Server Configuration tool by selecting **Start > Programs > AgilePoint > AgilePoint Server Configuration**.

3. Click the **Extension** link.
4. Click **Add**. The **Global Extended Module** window appears.
5. In the **Name** field, type **CustomExceptionHandler**.
6. In the **Assembly** field, click the **Ellipses** button.
7. Select the **CustomExceptionHandler.dll** and click **Open**. The **Class Name** field is automatically populated.
8. Click **OK**. The AgilePoint Configuration window reappears with CustomExceptionHandler in the list of extensions.
9. If there are additional configuration properties associated with this AgileConnector, click **Configure**. The custom form window appears.
10. Configure the rules.
11. Click **OK**.

The configuration information is stored in the AgilePoint Server Configuration file.

Implementation

After development and deployment of the Global Exception Handler AgileConnector, the implementation is done at the process model layer via AgilePoint Envision. All out-of-the-box and custom AgileParts include a drop-down property activity called **ExceptionHandlerScope**. This drop-down is used to select either **Local** or **Global** where if **Local** is selected, the exception handling will be handled as normal by reporting exception data to the **SaveErrorMessageTo** and **SaveStatusTo** custom attributes and if **Global** is selected, the AgilePart will continue to process as normal by reporting exception data to the **SaveErrorMessageTo** and **SaveStatusTo** custom attributes, but also calls on a custom AgileConnector with custom exception handling logic to handle the exception(s) as desired.

Note: To use the Global Exception Handling option, you do not need to do anything extra in the AgilePart code, the implementation is determined at the process model layer as shown below.

Testing and Debugging Custom AgileShape Projects

Custom AgileShapes (including AgileParts and AgileWorks) are not stand-alone applications, and are only intended to be fully executed within the context of an AgilePoint Server instance. Like other "hosted" code (e.g. ASP.NET code-behind code; ASP.NET user and server controls; SharePoint modules; etc.), the behavior of the code at run time depends upon the state of the server. Although this makes unit testing more difficult than for a non-hosted application, it is an unfortunate consequence of the hosted-component model.

Therefore, at this time our official recommendation is to test AgileShapes using actual process models on a real AgilePoint Server installation, because this most closely emulates a production environment. This can be done either manually and/or using custom test applications that use the AgilePoint APIs to start and monitor the test processes.

However, there are certain steps you can take to make unit testing of your code more feasible. The first step is to separate as much of your AgileShape's code as possible into static methods and/or into separate class libraries. This should allow such separated code to be more easily unit testable outside of the hosting environment.

Another alternative is to implement a "fake server" by creating one or more custom classes that implement the AgilePoint interfaces required by your code at run time (e.g. IWFApi or IWFWorkflowService). The complexity of implementing such a solution will depend on how closely you need the "fake server" to mimic a real server. Although this would likely require more time than most developers would wish to spend, this would be the most comprehensive way to make AgilePoint-hosted components fully unit testable.

Ensuring You Are Testing the Correct Version

When testing and debugging custom AgileShapes, you must ensure that AgilePoint Envision is using the correct version:

1. When making minor implementation changes to an AgilePart or AgileWork, use the same assembly version number when you recompile. (I.e. make sure that the AssemblyInfo file in your project uses a hard-coded version number, and does not include any wildcards.)
2. After compiling a new version, add it to the GAC, or use the AgilePart Deployment Utility to deploy it.
3. If AgilePoint Server is already running and has already loaded an older copy of the assembly into memory, then you will need to reset IIS (e.g. run "iisreset") or restart the AgilePoint Server application in order to force AgilePoint Server to load the newer copy of the assembly into memory the next time it is needed.
4. If you compile a new copy of the assembly with a new assembly version number (as opposed to #1 above), then you will also need to unregister and reregister the AgileShape's assembly in Envision. However, if you followed the recommendations in #1 above, then you do not need to unregister or reregister the AgileShape's assembly from Envision (other than the single, initial registration of the assembly before you first use it.

Process Merging and Splitting

This section provides information about the Process Splitting and Merging functionality.

Download http://download.agilepoint.com/pub/ProcessSplittingMergingSample.zip for more information on implementing this functionality using the AgilePoint Web Service API. The AgilePoint Web Service API document also provides information about this functionality.

Process Merging

This feature can merge multiple process instances (at least two process instances) into one process instance, and the original instances will be canceled.

Note: These process instances should be based on the same process template.

Note: It is required to suspend the process instances before performing the merge.

Process Splitting

This feature can split one process instance into multiple process instances (at least two process instances to begin with). The original process instance will still keep running.

The split process instances should be based on one process instance.

Remote API

This section provides information about the AgilePoint remote API. With AgilePoint Developer, AgilePoint can be developed and deployed as an embedded or autonomous application to meet different requirements.

- An embedded BPM application is usually tightly integrated with its surrounding system through AgilePoint API and functionality of an embedded BPMS can be exhibited through various surrounding applications.
- An autonomous AgilePoint Application is hosted by Microsoft IIS, running independently as a Microsoft .NET Web Application, and provides XML Web Service to its client Web Application. Exposing BPM functionality in this manner has a number of advantages. The greatest is that the Web Service can be provided to many applications independent of hardware, operating system, and programming languages boundaries. It makes application debugging easier because requests / responses are sent / received as human-readable content.

For more information, see AgilePoint API in the Documentation Library.

Accessing AgilePoint Web Service API

This section provides information about accessing the AgilePoint Remote API from a 3rd party application such as a SharePoint component or ASP.NET application in a separate machine through Impersonation. This method is used if you want to implement your own custom component to access AgilePoint through Web Service API. These components can be located in a physically separate machine (for example, calling from a SharePoint Web Part).

When accessing AgilePoint Server through the Remote API, an HTTP - 401 authentication error may occur. For the SharePoint component, it runs under the identity that is specified in the SharePoint Application Pool. In some case, the identity could be set using the NetworkService or LocalSystem. These identities will not be able to get authenticated remotely through IIS and therefore returned the HTTP - 401 error.

In a multiple-server environment, it is recommend to set up a dedicated user credential that the 3rd party application code can impersonate to get authenticated cross machine in order to access the AgilePoint Web Service. This dedicated user credential is typically stored in the web.config file of the application which can be retrieved before accessing the Web Service.

AgilePoint also provides an API called Surrogate() to allow the custom code to set the login user identity after the initial authentication through the impersonation. This will allow the remaining operation with the Web Service to run under the correct login identity.

Surrogating

1. In your custom coding for surrogating, at **api.SetClientAppName("MyApplication")**, put in your desired application name (case sensitive).
2. Open the **AgilePoint Server Configuration** and click the **Extension** link, then click the **Add** button.
3. Enter the application name (case sensitive) you used in the custom coding and also the Impersonator name (see picture below). The Impersonator has to be a registered user and preferably an administrator in AgilePoint Server.