INNOVATE

Microsoft Dynamics™ GP

Creating a Windows® service that uses Microsoft Dynamics GP eConnect to integrate data

Article

Create a Windows Service that uses the .NET FileSystemWatcher class to monitor a directory for file operations

Date: April, 2008

Microsoft Dynamics™

Table of Contents

Introduction

In a fast-paced environment, Customers and Partners look for a solution to import their data in a timely and accurate fashion. This document explains how to use Microsoft Dynamics™ GP eConnect, Microsoft® Visual Basic® .NET and Microsoft Visual Studio® .NET to develop a Windows® Service to automate the process of importing data. You should be familiar with the navigation in Visual Studio. NET and be familiar with referencing .NET assemblies in Visual Basic. NET. Also, you should have experience with developing and deploying .NET applications to end users' computers.

Overview

1. First, create a new project in Visual Studio .NET and choose Windows Service.
2. To use Microsoft Dynamics GP eConnect, add a reference to the Microsoft.Dynamics.GP.eConnect.dll and Microsoft.Dynamics.GP.eConnect.Serialization.dll.
3. After eConnect has been referenced in the project, the next step is to copy and paste all of the code from this document into the project.
4. After the code has been copied into the Windows Service project, use Visual Studio .NET to build the project. This creates the setup files that must be installed on the client computers when installing the Windows Service.
5. To use the Windows Service, start the service within Services in the Administrative Tools section of the Control Panel. After the service has been started, the project is then active and able to import data.

Start a new application in Visual Basic .NET

1. Start Visual Studio .NET.
2. On the **File** menu, point to **New** and then click **Project**.
3. In the **New Project** dialog box, click **Visual Basic Projects** and then click **Windows Service**. Set the project name to eConnectWatcher and the folder location to C:\, and then click **OK**.
4. In the **Solution Explorer** window, right-click the Service1.vb class and choose **Rename**. Enter FileMonitor.vb as the new name.

Add References needed by the new application

1. In the **Solution Explorer** window, right-click the References folder and choose **Add Reference**.
2. In the **Add Reference** window, be sure the .NET tab is selected. Scroll down to the System.EnterpriseServices component and select it. Click **Select** to add this component to the **Selected Components** field of the window.
3. Click the **Browse** button. The **Select Component** window is displayed. Browse to the C:\Program Files\Common Files\Microsoft Shared\eConnect 10\Objects\Dot Net folder. Select the Microsoft.Dynamics.GP.eConnect.dll file and click **Open**.
4. Verify that both of these items are in the **Selected Components** field on the **Add Reference** window. Click **OK**.

Add code to the project

Add Imports statements and set the Service Name

1. Open the Code Editor for FileMonitor.vb by clicking the "click here to switch to code view" hyperlink in the design window.
2. Type the following lines of code immediately after the Imports System.ServiceProcess statement at the top:

```
Imports System.IO
Imports Microsoft.Dynamics.GP.eConnect
```

   Your code should look like the following example:

```
Imports System.ServiceProcess
Imports System.IO
Imports Microsoft.Dynamics.GP.eConnect
```

3. Click the "+" sign next to the Component Designer generated code to open that region of code. Scroll down to the end of this region in the code window. Several lines above the #End Region is the code which sets the Service Name:

```
Me.ServiceName = "Service1"
```

4. Change this ServiceName property to "eConnectWatcher". Your code should look like the following example:

```
<System.Diagnostics.DebuggerStepThrough()> Private Sub InitializeComponent()
    components = New System.ComponentModel.Container()
    Me.ServiceName = "eConnectWatcher"
End Sub
```

Declare a FileSystemWatcher object and add code to the OnStart event

1. Scroll back up to the top of the expanded Component Designer generated code region and click the "-" sign to collapse that region of code.
2. In the blank line immediately following the Component Designer generated code region, type the following line of code:

```
Private xmlWatcher As New FileSystemWatcher
```

3. Type the following lines of code in the OnStart Sub (the 2 lines of default comments can be deleted if desired):

```
xmlWatcher.Path = "C:\Import\"
xmlWatcher.Filter = "*.xml"

AddHandler xmlWatcher.Created, AddressOf xmlImport
xmlWatcher.EnableRaisingEvents = True
```

Your code should look like the following example:

```
Component Designer generated code
    Private xmlWatcher As New FileSystemWatcher
```

```
    Protected Overrides Sub OnStart(ByVal args() As String)
        xmlWatcher.Path = "C:\Import\"
        xmlWatcher.Filter = "*.xml"

        AddHandler xmlWatcher.Created, AddressOf xmlImport
        xmlWatcher.EnableRaisingEvents = True
    End Sub
```

Add code to the OnStop event

1. Type the following lines of code in the OnStop Sub (the line of default comments can be deleted if desired):

```
xmlWatcher.EnableRaisingEvents = False
xmlWatcher.Dispose()
```

Your code should look like the following example:

```
Protected Overrides Sub OnStop()
    xmlWatcher.EnableRaisingEvents = False
    xmlWatcher.Dispose()
End Sub
```

Add a handler for the xmlWatcher.Created event

The signature of this method must match the signature of the FileSystemEventHandler delegate which is calling it. This signature is:

```
Delegate Sub FileSystemEventHandler(sender As Object, e As System.IO.FileSystemEventArgs)
```

1. Copy the code from the following file into the project after the end of the OnStop event sub.

   Note: To view attachments, you must be viewing this document in Adobe Reader 6.0 or later, or in the full version of Acrobat. Right-click the paperclip icon to the left of the desired file, and choose to open or save the file.

   - XMLWatcher.Created.txt

Add and configure an installer for the service

1. Switch to the Design view of the FileMonitor.vb class.
2. Right-click the Design window and choose **Add Installer**.

Build, install and test the service

1. On the **Build** menu, click **Build Solution**.
2. Launch a Visual Studio .NET Command prompt by clicking **Start**, point to **Programs | Microsoft Visual Studio .NET 2005 | Visual Studio .NET Tools | Visual Studio .NET 2005 Command Prompt**.
3. Navigate to the directory containing the eConnectWatcher service executable file by typing the following command:

   ```
   cd\eConnectWatcher\bin
   ```

4. Install the service by typing the following command:

   ```
   installutil eConnectWatcher.exe
   ```

5. When prompted for a username and password, enter the credentials for an account with local Administrator rights.
6. Close the Visual Studio .NET 2005 Command Prompt window by typing "Exit".
7. Using Windows Explorer, create a new folder named Import at the root of C:\. Inside the Import folder, create a new folder named Complete.
8. To open the Services Management Console, click **Start** and point to **Run**. Enter the following command:

   ```
   services.msc
   ```

9. Scroll down to the eConnectWatcher service. Right-click the eConnectWatcher service and point to **Start**. Then close the Services Management Console.
10. Test the service. Use Windows Explorer to copy one of the eConnect sample XML documents from the C:\Program Files\Common Files\Microsoft Shared\eConnect 10\XML Sample Documents\Incoming folder to the C:\Import folder. The service that you created should pull the xml document from the Import folder and use eConnect to integrate the data into Microsoft Dynamics GP 10.0.

Microsoft Dynamics is a line of integrated, adaptable business management solutions that enables you and your people to make business decisions with greater confidence. Microsoft Dynamics works like and with familiar Microsoft software, automating and streamlining financial, customer relationship and supply chain processes in a way that helps you drive business success.

U.S. and Canada Toll Free 1-888-477-7989

Worldwide +1-701-281-6500

www.microsoft.com/dynamics

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, this document should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This document is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

© 2008 Microsoft Corporation. All rights reserved.

Microsoft, the Microsoft Dynamics Logo, Microsoft Dynamics, Visual Basic, Visual Studio, and Windows are either registered trademarks or trademarks of Microsoft Corporation.