

A SharePoint Developer Introduction

Hands-On Lab

Lab Manual

SPCHOL305 - Developing a SharePoint 2010 Workflow with Initiation Forms in Visual Studio 2010 – C#

This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it.

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

© 2010 Microsoft. All rights reserved.

**Contents**

[SPCHOL 305 – Developing a SharePoint 2010 Workflow with Initiation Forms 2](#_Toc260147786)

[Lab Objective 2](#_Toc260147787)

[Additional Resources 2](#_Toc260147788)

[Getting Started 2](#_Toc260147789)

[Logging in to the Virtual Machine 2](#_Toc260147790)

[Locations 2](#_Toc260147791)

[Lab Pre-requisites 3](#_Toc260147792)

[Copying code samples from Word document 3](#_Toc260147793)

[Code Snippets 4](#_Toc260147794)

[Exercise 1 5](#_Toc260147795)

[Task 1: Create and Prepare Project 5](#_Toc260147796)

[Task 2: Create a new Workflow Activity for use by the Reusable Workflow 6](#_Toc260147797)

[Task 3: Add code to create a document library based on parameters passed to our activity. 9](#_Toc260147798)

[Task 4: Configure activity for deployment. 11](#_Toc260147799)

[Task 5: Add the SPDActivityDemo activity to be deployed with SPCHOL305Ex1. 15](#_Toc260147800)

[Task 6: Configure a Feature using Feature Designer 17](#_Toc260147801)

[Task 7: Add and code a Feature Receiver 17](#_Toc260147802)

[Task 8: Create a re-usable workflow using SharePoint Designer 19](#_Toc260147803)

[Task 9: Import the reusable workflow into Visual Studio 27](#_Toc260147804)

[Task 10: Associate the workflow with a list in SharePoint 33](#_Toc260147805)

[Exercise Summary 35](#_Toc260147806)

[Exercise 2 35](#_Toc260147807)

[Task 1: Ensure there is a Customer list in the site 35](#_Toc260147808)

[Task 2: Create and Prepare Project 36](#_Toc260147809)

[Task 2: Create a new Workflow Initiation Form to be used by the workflow. 39](#_Toc260147810)

[Task 3: Complete the Workflow Initiation Form. 40](#_Toc260147811)

[Task 4: Process Workflow Initiation Form data in Workflow1 41](#_Toc260147812)

[Task 5: Deploy and Debug Workflow1 42](#_Toc260147813)

[Exercise Summary 43](#_Toc260147814)

[Lab Summary 44](#_Toc260147815)

# SPCHOL 305 – Developing a SharePoint 2010 Workflow with Initiation Forms

Estimated time to complete this lab: **30 minutes**

## Lab Objective

**Exercise Overview**

The first exercise shows how to create a custom workflow activity used by a SharePoint Designer reusable workflow and deploy them together as a single \*.WSP. It also shows how to import that workflow into Visual Studio 2010.

The second exercise shows how to create a SharePoint Workflow with an Initiation Form and process the form data.

**Feature Overview**

SharePoint Designer allows creating a workflow that is reusable across multiple lists and multiple sites. The workflow can be exported as a SharePoint Solution File (\*.WSP).

## Additional Resources

This lab includes the following additional resources:

|  |  |  |
| --- | --- | --- |
| This Lab Manual | SPCHOL305\_Manual\_CS.docx | This document |
| Source Code | Completed\CS\Ex1  Completed\CS\Ex2 | Completed lab source code. |
| Resources | *Resources\CS* | Various resources used throughout this lab. |

## Getting Started

### Logging in to the Virtual Machine

Please log into the virtual machine as the following user:

Username: Administrator

Password: pass@word1

### Locations

This Hands-On Lab contains a number of additional resources in fixed locations. By default, it is assumed that the base HOL directory is ***C:\Content Packs\Packs\SharePoint 2010 Developer Labs 1.0\*Supporting Files\SPCHOL305\Resources**.

The default working folder for this lab is C:\SPHOLs\SPCHOL305.

### Lab Pre-requisites

1. Browse to base HOL directory ***Supporting Files\SPCHOL305\Resources***and execute the **optimize.ps1** PowerShell script:
   * **Right-click** on **optimize.ps1** and select **Run with PowerShell**:

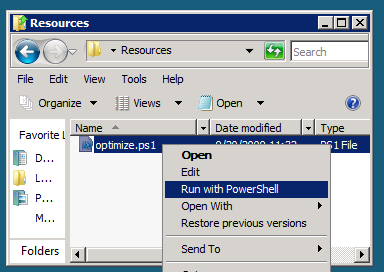


Figure 1 - Execute the PowerShell script

* + This will open a PowerShell window to execute the script. Please wait until the PowerShell script completes executing the script and closes the PowerShell window (this may take a few moments):

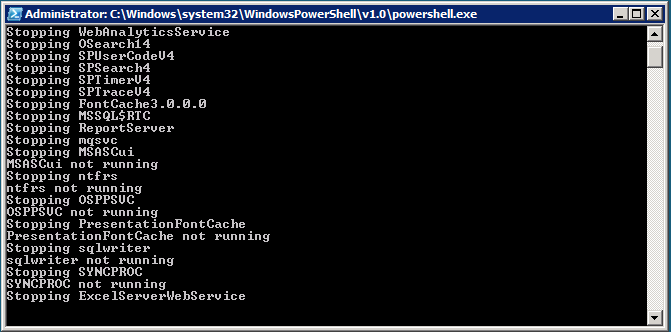


Figure 2 - PowerShell Window executing the script

### Copying code samples from Word document

Copying and pasting code from this Word document to Visual Studio is only safe for the sections of formatted code, e.g.:

Console.WriteLine("This is safe code!");

Code not in these sections may contain Unicode or other invisible characters that are not valid XML or C#/VB code, e.g.:

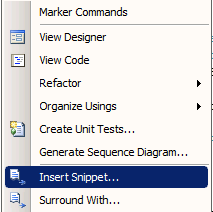
**Console.WriteLine(“This is NOT safe code !!”);**

### Code Snippets

You can also use Code Snippets to insert the appropriate code in the lab.

To use the required code snippet for this lab:

* **Right-click** on the code file where you want to insert the code snippet.
* Select **Insert Snippet:**

****

* Select code snippets from **My Code Snippets** gallery.

## Exercise 1

This exercise shows how to create a custom workflow activity used by a SharePoint Designer reusable workflow and deploy them together as a single \*.WSP. It also shows how to import that workflow into Visual Studio 2010.

### Task 1: Create and Prepare Project

In this task a project you will create an Empty Project solution and use SharePoint user controls.

1. Open **Visual Studio 2010** by going to **Start | All Programs | Microsoft Visual Studio 2010 | Microsoft Visual Studio 2010.**
2. In **Visual Studio 2010,** create a new project by going to **File | New | Project**.
3. Select the **Visual C# | SharePoint | 2010 |** **Empty SharePoint Project** project template.
4. Enter **SPCHOL305Ex1** in the Name textbox, and enter **C:\SPHOLS\SPCHOL305\CS\Ex1** in the **Location** textbox.

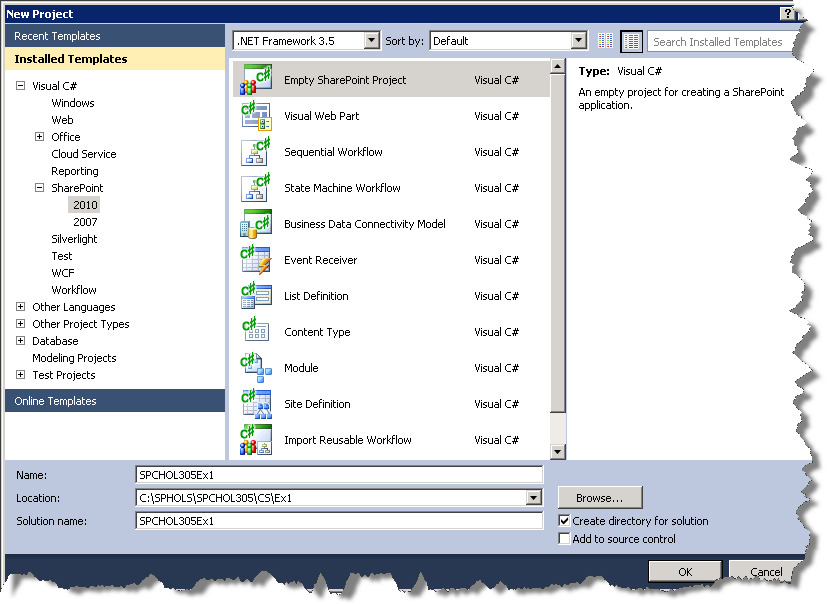


Figure 3 - New Project Dialog

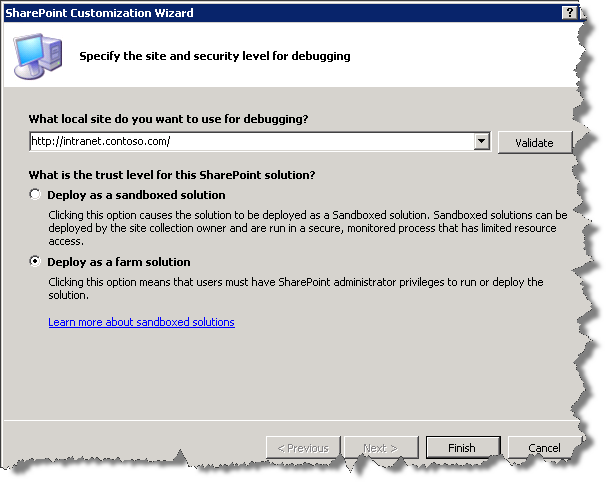
1. Click **OK**.
2. In the SharePoint Customization Wizard, change the URL to [**http://intranet.contoso.com**](http://intranet.contoso.com) and select **Deploy as a farm solution.** Then click **Finish**.  
   

Figure 4 - SharePoint Customization Wizard

7. Visual Studio will create the new SPCHOL305Ex1 project and add the necessary files.

### Task 2: Create a new Workflow Activity for use by the Reusable Workflow

1. In Visual Studio 2010, add a new project by going to **File | Add | New Project.**
2. Change the Framework to **.NET Framework 3.5**

Figure 5 - Set .NET Framework Version

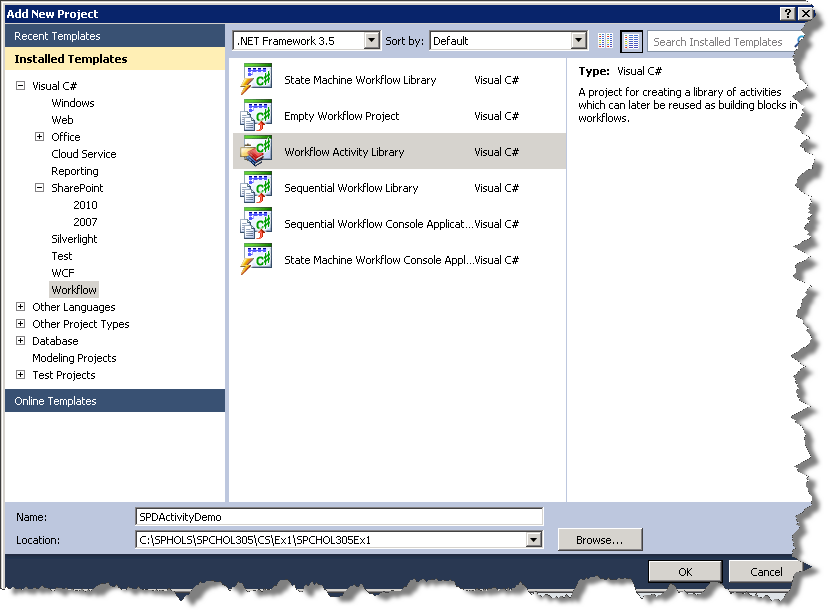
1. Select the **Visual C# | Workflow | Workflow Activity Library** project template.
2. Enter **SPDActivityDemo** in the **Name** textbox, and click **OK**. 

Figure 6 - New Project Dialog

1. In Solution Explorer,right-click the **SPDActivityDemo** project and select **Add Reference**.
2. Switch to the **Browse** tab. Enter **C:\Program Files\Common Files\Microsoft Shared\Web Server Extensions\14\ISAPI** in the File name textbox and hit **Enter**.
3. Select **Microsoft.SharePoint.dll** and **Microsoft.SharePoint.WorkflowActions.dll**, and click **OK**.

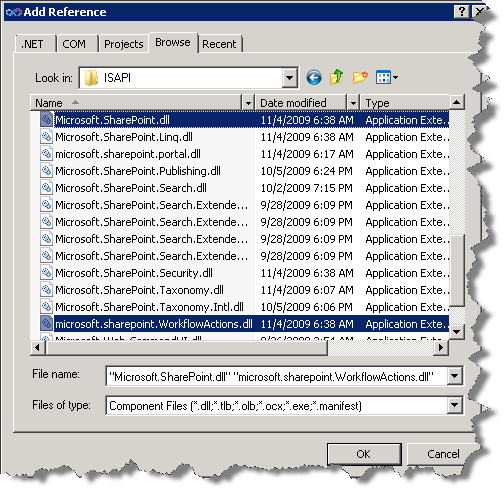


Figure 7 - Add Reference Dialog

1. Right-click on **Activity1.cs** in the Solution Explorer and select **Rename**.
2. Rename **Activity1.cs** to **CreateDocumentLibrary.cs.**

### Task 3: Add code to create a document library based on parameters passed to our activity.

1. Right-click on the **CreateDocumentLibrary.cs** file in the Solution Explorer and select **View Code**.
2. Change the **CreateDocumentLibrary base class** from SequenceActivity to **Activity**, as seen below.

public partial class CreateDocumentLibrary : Activity

1. Add the following using statements at the top of the file under the existing using statements.:

using Microsoft.SharePoint;

using Microsoft.SharePoint.Workflow;

using Microsoft.SharePoint.WorkflowActions;

**Code Snippet:** *My Code Snippets |* **spchol305\_ex1\_createdoclib\_namespace**

1. Add a new DependencyProperty named **UrlProperty** to the CreateDocumentLibrary class..

* *This will be the location where the document library will be created.*

public static DependencyProperty UrlProperty = DependencyProperty.Register("Url", typeof(string), typeof(CreateDocumentLibrary), new PropertyMetadata(""));

[DescriptionAttribute("Url of base site")]

[BrowsableAttribute(true)]

[DesignerSerializationVisibilityAttribute(DesignerSerializationVisibility.Visible)]

[ValidationOption(ValidationOption.Optional)]

public string Url

{

get

{

return ((string)(base.GetValue(CreateDocumentLibrary.UrlProperty)));

}

set

{

base.SetValue(CreateDocumentLibrary.UrlProperty, value);

}

}

**Code Snippet:** *My Code Snippets |* **spchol305\_ex1\_createdoclib\_urlproperty**

1. Add a new DependencyProperty to the **CreateDocumentLibrary** class named **DocLibNameProperty**.

* *This will be the name of the DocumentLibrary created by the activity.*

public static DependencyProperty DocLibNameProperty = DependencyProperty.Register("DocLibName", typeof(string), typeof(CreateDocumentLibrary), new PropertyMetadata(""));

[DescriptionAttribute("Used as doc lib name")]

[BrowsableAttribute(true)]

[DesignerSerializationVisibilityAttribute(DesignerSerializationVisibility.Visible)]

[ValidationOption(ValidationOption.Optional)]

public string DocLibName

{

get

{

return ((string)(base.GetValue(CreateDocumentLibrary.DocLibNameProperty)));

}

set

{

base.SetValue(CreateDocumentLibrary.DocLibNameProperty, value);

}

}

**Code Snippet:** *My Code Snippets |* **spchol305\_ex1\_createdoclib\_doclibproperty**

1. Add the following code beneath your **CreateDocumentLibrary** constructor:

protected override ActivityExecutionStatus Execute(ActivityExecutionContext executionContext)

{

CreateDocLib();

return ActivityExecutionStatus.Closed;

}

private void CreateDocLib()

{

using (SPSite sps = new SPSite(Url))

{

using (SPWeb spw = sps.RootWeb)

{

Guid ID = spw.Lists.Add(DocLibName, DocLibName + " Document Library",   
SPListTemplateType.DocumentLibrary);

SPList spdl = spw.Lists[ID];

spdl.OnQuickLaunch = true;

spdl.Update();

}

}

}

**Code Snippet:** *My Code Snippets |* **spchol305\_ex1\_createdoclib\_execute**

### Task 4: Configure activity for deployment.

1. Configure activity for a strong name. Add **SPDActivityDemo.snk** from Resources folder. (*Location at the start of this manual)*. To the **SPDActivityDemo** project.

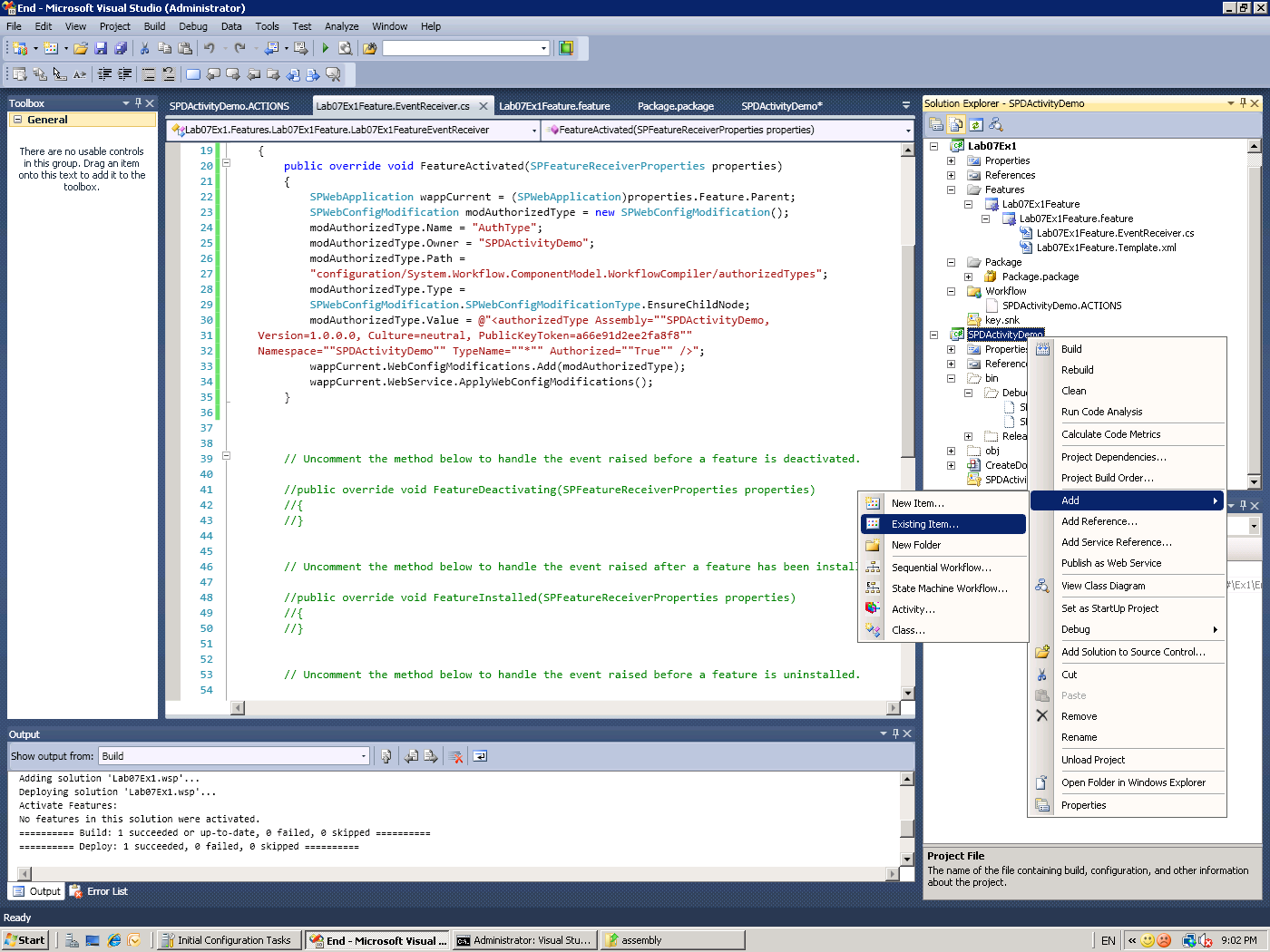


Figure 8 - Add Existing Item

1. **Right-click** on the SPDActivityDemo project in the Solution Explorer and select **Properties.**
2. Click on the **Signing** tab, check “**Sign the Assembly,** and then select **SPDActivityDemo.snk.**

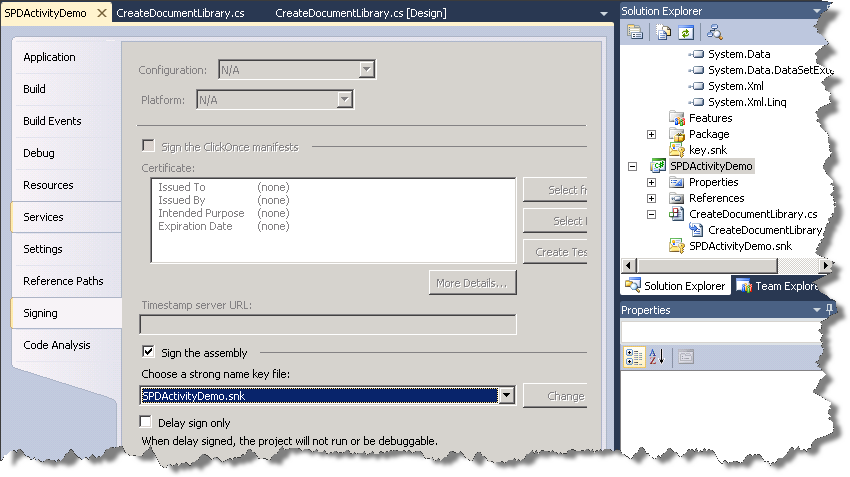


Figure 9 - Signing

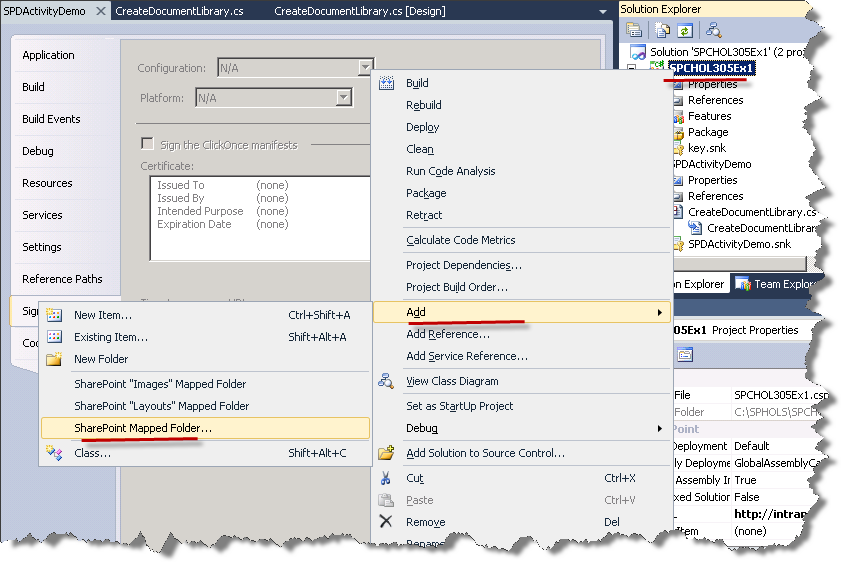
1. Build your project **(CTRL-SHIFT-B)** and fix any errors.
2. **Right-click** on the **SPCHOL305Ex1** project and click **Add**, **SharePoint Mapped Folder**

Figure 10 - SharePoint Mapped Folder

1. Select **Template\1033\Workflow** in the Add SharePoint Mapped Folder dialog and select **OK**.

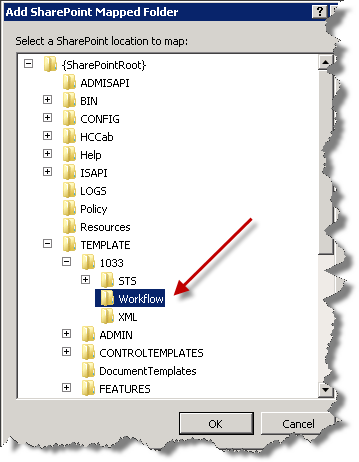


Figure 11 - Add SharePoint Mapped Folder dialog

1. In the Solution Explorer, expand the newly added **Workflow** folder.
2. **Right-click** on the **SPCHOL305Ex1** folder (if it exists) and select **Delete**.
3. **Right-click** on the **Workflow** folder and select **Add | New Item.**
4. Select the **XML File** template and name the file **SPDActivityDemo.ACTIONS** then press **Add**.

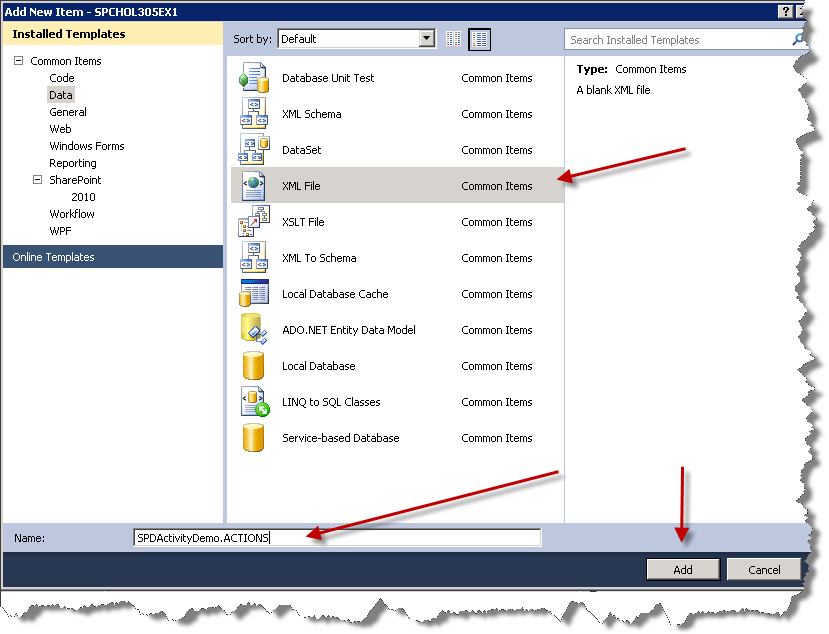


Figure 12 - Add New Item dialog

1. Replace the contents of **SPDActivityDemo.ACTIONS** with the following

<?xml version="1.0" encoding="utf-8"?>

<WorkflowInfo>

<Actions Sequential="then" Parallel="and">

<Action Name="Create Document Library"

ClassName="SPDActivityDemo.CreateDocumentLibrary"

Assembly="SPDActivityDemo, Version=1.0.0.0,

Culture=neutral, PublicKeyToken=a66e91d2ee2fa8f8"

AppliesTo="all"

Category="Labs">

<RuleDesigner Sentence="Document Library Name %1 to site %2.">

<FieldBind Field="DocLibName" Text="Document Library Name"

DesignerType="TextArea" Id="1"/>

<FieldBind Field="Url" Text="Url of base site" Id="2"

DesignerType="TextArea"/>

</RuleDesigner>

<Parameters>

<Parameter Name="DocLibName" Type="System.String, mscorlib"

Direction="In" />

<Parameter Name="Url" Type="System.String, mscorlib"

Direction="In" />

</Parameters>

</Action>

</Actions>

</WorkflowInfo>

**Code Snippet:** *My XML Snippets |* **spchol305\_ex1\_workflowactions\_xml**

1. Build **SPDActivityDemo** Project

### Task 5: Add the SPDActivityDemo activity to be deployed with SPCHOL305Ex1.

1. In the Solution Explorer expand the **Package** folder under the **SPCHOL305Ex1** project.
2. **Double-click** on the Package.package file to bring up the Package designer.
3. **Click** on the **Advanced** tab on the bottom of the designer



Figure 13 - Advanced Package tab

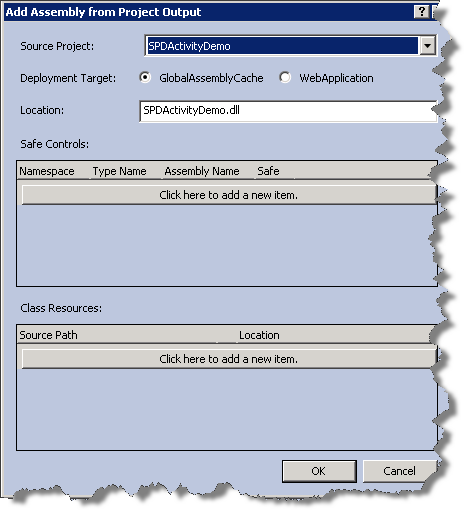
1. **Click** the **Add** button and select **Add Assembly from Project Output.**
2. In the **Add Existing Assembly from Project Output** dialog **click** the Source Project combo and select SPDActivityDemo**\bin**. 

Figure 14 - Add Existing Assebly from Project Output dialog

1. Click the button under Safe Controls. Within Safe Controls add the following:

Assembly Name:

**SPDActivityDemo, Version=1.0.0.0, Culture=neutral, PublicKeyToken= a66e91d2ee2fa8f8**

Name Space: **SPDActivityDemo**

Safe: **Checked**

Type Name: **\***

1. When you have finished filling out the safe control details, press **Enter** to confirm them.
2. Click **OK**

### Task 6: Configure a Feature using Feature Designer

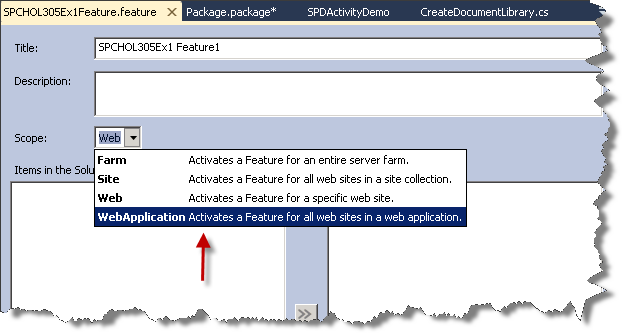
1. **Right-click** on the Features folder in the Solution Explorer under the SPCHOL305Ex1 project and select **Add Feature.**
2. **Right-click** on **Feature1** and rename it to **SPCHOL305Ex1Feature**.
3. In Feature designer change the scope of the **SPCHOL305Ex1Feature** scope to **WebApplication**. 

Figure 15 - Feature Designer

### Task 7: Add and code a Feature Receiver

1. **Right-click** on the **SPCHOL305Ex1Feature** feature in the Solution Explorer and select **Add Event Receiver**
2. Add a using statement to the top of the code:

using Microsoft.SharePoint.Administration;

1. Add the following code to your FeatureReceiver class declaration:

public override void FeatureActivated(SPFeatureReceiverProperties properties)

{

SPWebApplication wappCurrent = (SPWebApplication)properties.Feature.Parent;

SPWebConfigModification modAuthorizedType = new SPWebConfigModification();

modAuthorizedType.Name = "AuthType";

modAuthorizedType.Owner = "SPDActivityDemo";

modAuthorizedType.Path =

"configuration/System.Workflow.ComponentModel.WorkflowCompiler/authorizedTypes";

modAuthorizedType.Type =

SPWebConfigModification.SPWebConfigModificationType.EnsureChildNode;

modAuthorizedType.Value = "<authorizedType Assembly=\"SPDActivityDemo, "

+ "Version=1.0.0.0, Culture=neutral, PublicKeyToken=a66e91d2ee2fa8f8\" "

+ "Namespace=\"SPDActivityDemo\" TypeName=\"\*\" Authorized=\"True\" />";

wappCurrent.WebConfigModifications.Add(modAuthorizedType);

wappCurrent.WebService.ApplyWebConfigModifications();

}

**Code Snippet:** *My code Snippets |* **spchol305\_ex1\_spchol305feature\_receiver**

1. Build and Deploy the **SPCHOL305Ex1Feature** project by **Right-click**ing on the project name and selecting Deploy.

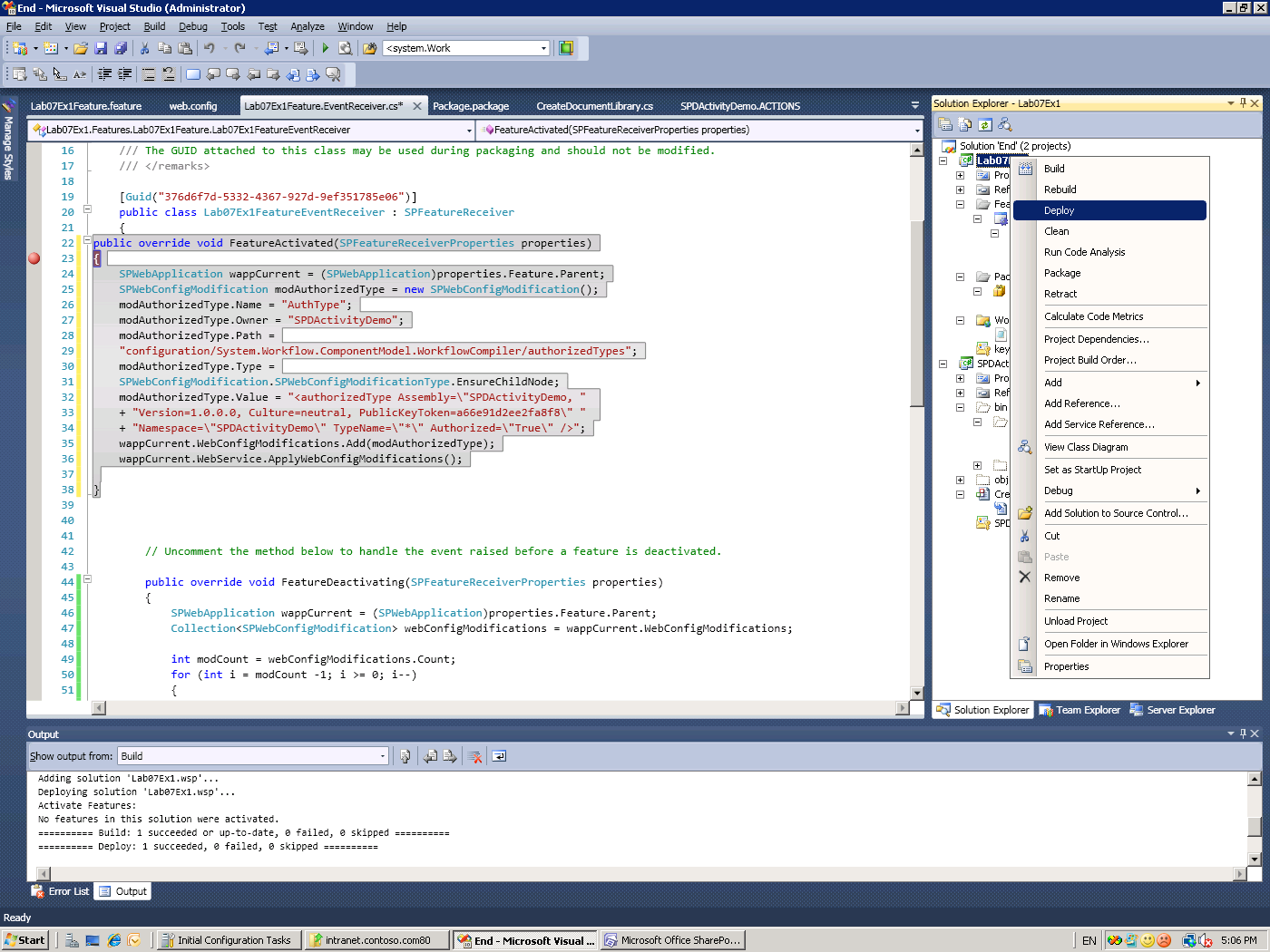


Figure 16 - Deploy Solution

### Task 8: Create a re-usable workflow using SharePoint Designer

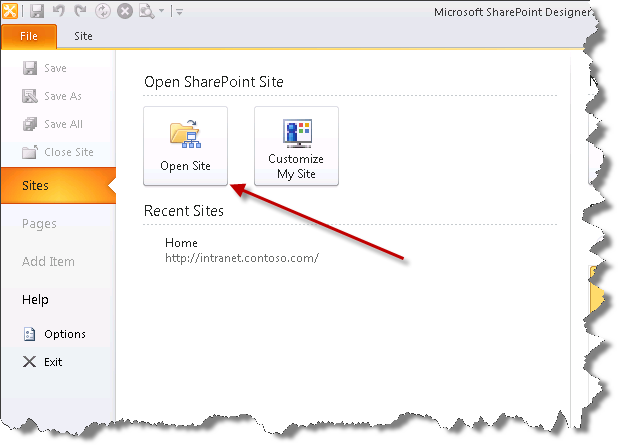
1. Open **SharePoint Designer 2010** by going to **Start | All Programs | SharePoint | Microsoft SharePoint Designer 2010**
2. Click on **Sites** and then **Open Site**.  
     
   

Figure 17 - Sites menu button in SharePoint Designer 2010

If you are asked for credentials use:

**Username**: Administrator

**Password**: pass@word1

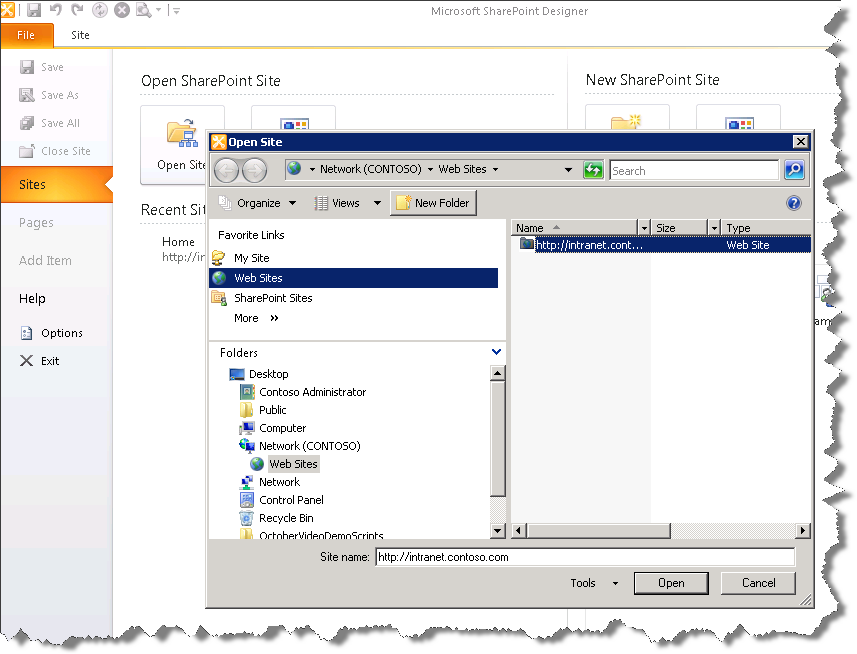
1. Change **Site name** to [*http://intranet.contoso.com*](http://intranet.contoso.com) and select **Open**  
     
   

Figure 18 - Open Site dialog

1. Click the **Reusable Workflow** button and when prompted name the workflow **SPDWorkflow** and then **click** the **OK** button to create the reusable workflow.

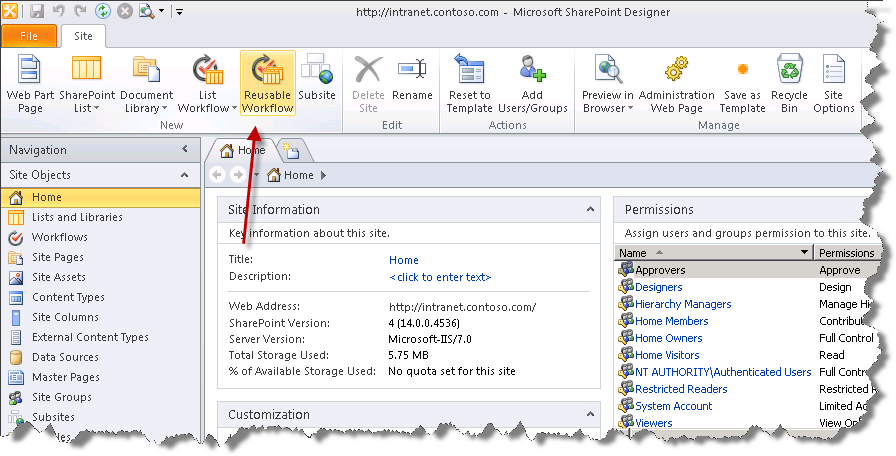


Figure 19 - Create Reusable Workflow

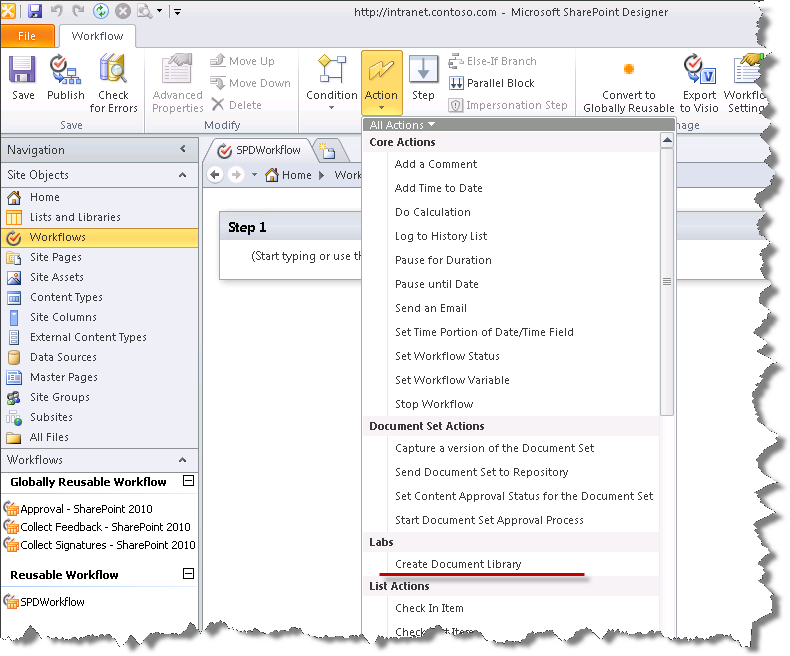
1. **Click** the **Actions** button in the ribbon and scroll down to **Labs** and **click Create a Document Library**. Note: This is the activity we made in Visual Studio previously. 

Figure 20 - Add Create Document Library Action

1. Click into Step 1. In the box next to *Document Library Name* click the Fx button.
2. Select **Current Item** as the Data Source, and **Title** as the value of Field from source. Then Press OK.

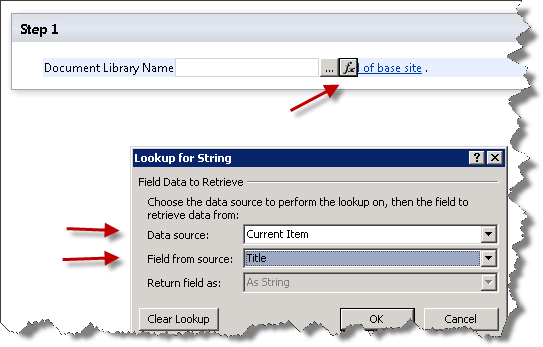


Figure 21 - Lookup for String dialog

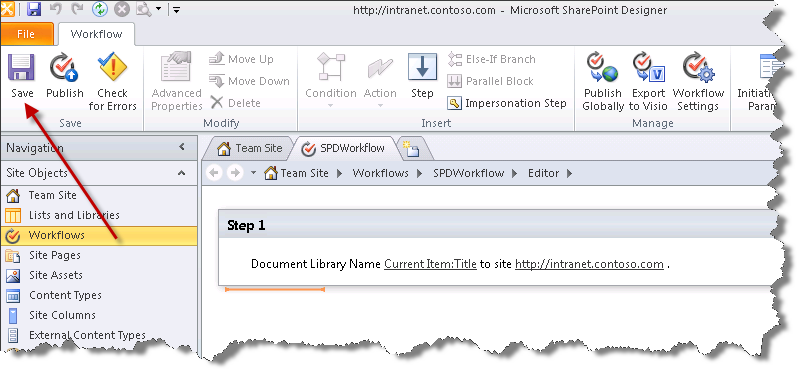
1. Type in **http://intranet.contoso.com** as the URL of the base site.
2. Click **Save**, on the ribbon.  
   

Figure 22 - Configue Workflow Action

1. Once saved click the **Publish** button in the ribbon. Wait for the workflow to publish.

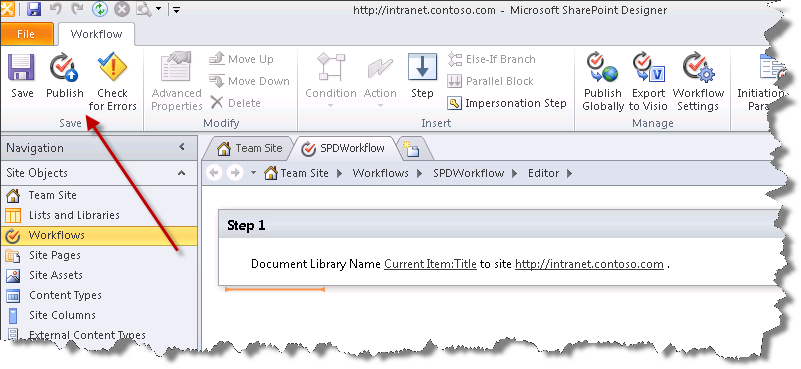


Figure 23 - Publish Workflow

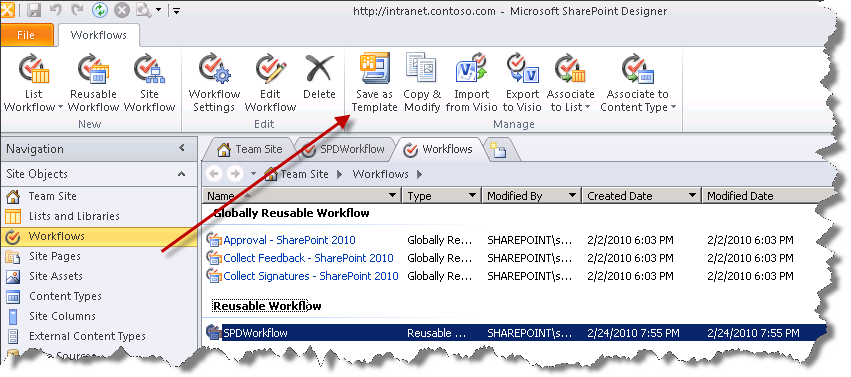
1. Once saved click on **Workflows** in the left navigation.
2. Click the workflow again beneath **Reusable Workflow** to bring up the workflow properties. Click **Save as Template** to save the .WSP file in the **Site Assets Library** SharePoint list.  
   

Figure 24 - Save As Template

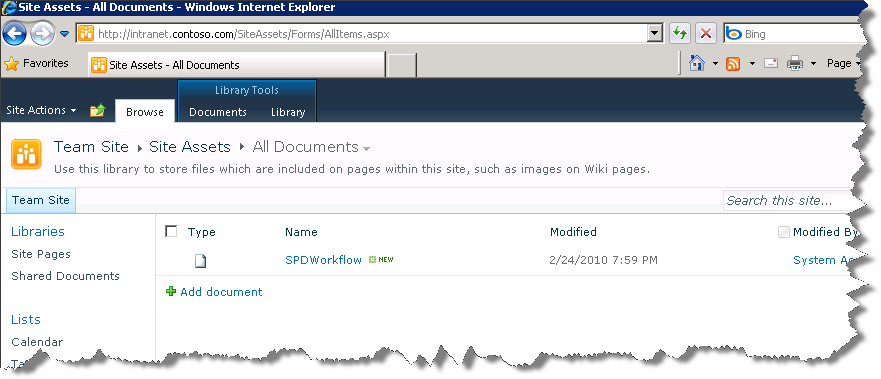
1. Open a web browser at [**http://intranet.contoso.com**](http://intranet.contoso.com)
2. Click **Site Actions -> View All Site Content -> Site Assets Library**. 

Figure 25 - Site Assets Library

1. Click on the **SPDWorkflow** dropdown menu and click **Send To -> Download a Copy**.

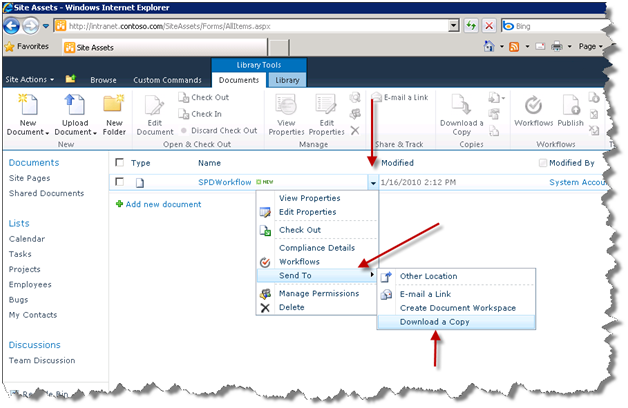


Figure 26 - Download WSP

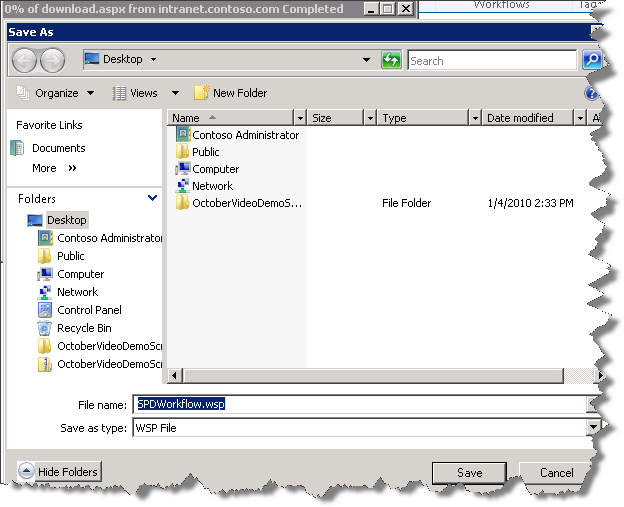
1. Save the WSP to a location that you can access in a subsequent step (like the Desktop) 

Figure 27 - Save As Dialog

### Task 9: Import the reusable workflow into Visual Studio

1. Back in Visual Studio 2010, **right-click** on the SPCHOL305Ex1 solution in the Solution **Explorer** window **Add | New Project.**
2. Select the **Visual C# | SharePoint | 2010 |** **Import Reusable Workflow** project template.
3. Change the **Name** to *SPDWorkflowImport*.

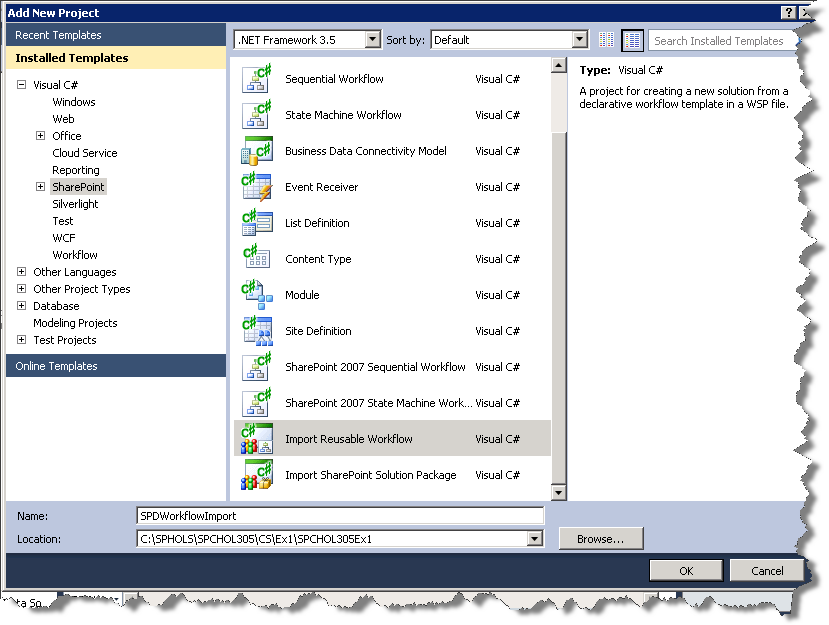


Figure 28 - New Project Dialog

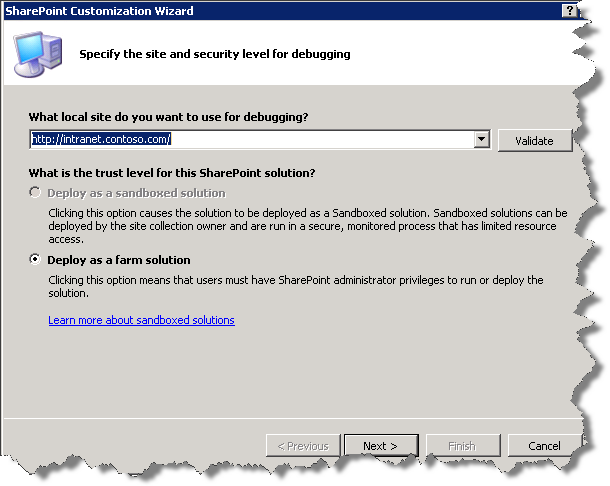
1. Click **Ok**.
2. Make sure the URL is [*http://intranet.contoso.com*](http://intranet.contoso.com). Click **Next**.  
   

Figure 29 - SharePoint Customization Wizard

1. When prompted for the .WSP file to import, browse to the SPDWorkflow.wsp file you saved previously**.** Click **Next.**

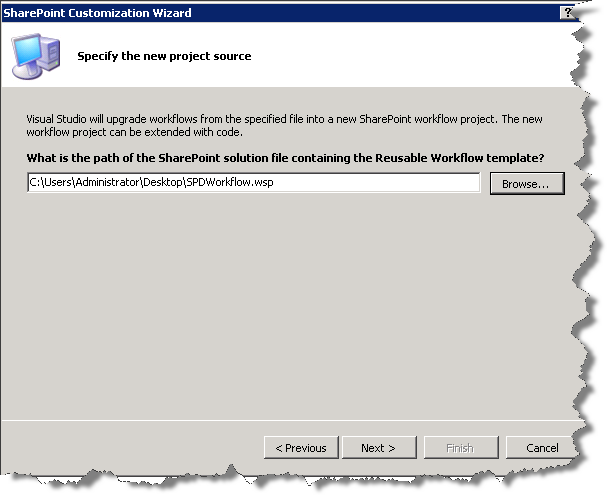


Figure 30 - SharePoint Customization Wizard

1. Click **Finish. NOTE – If you are using Visual Studio 2010 Beta 2, there will be an error and you will not be able to complete this lab. If you are using a later version of Visual Studio 2010 you may continue.**

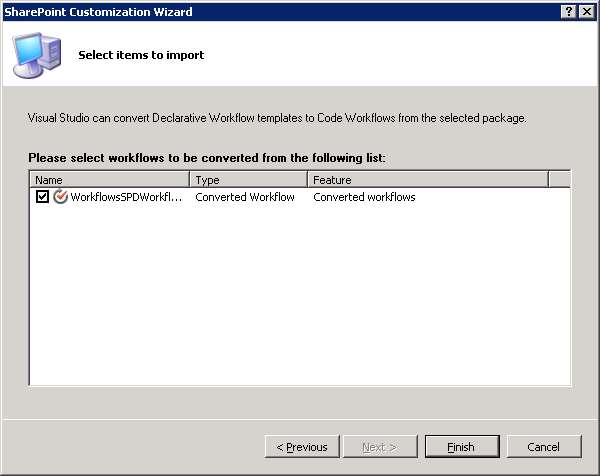


Figure 31 - SharePoint Customization Wizard

1. **Double-click** on Package in the **SPCHOL305Ex1** project and add the Converted workflows feature to the Package.

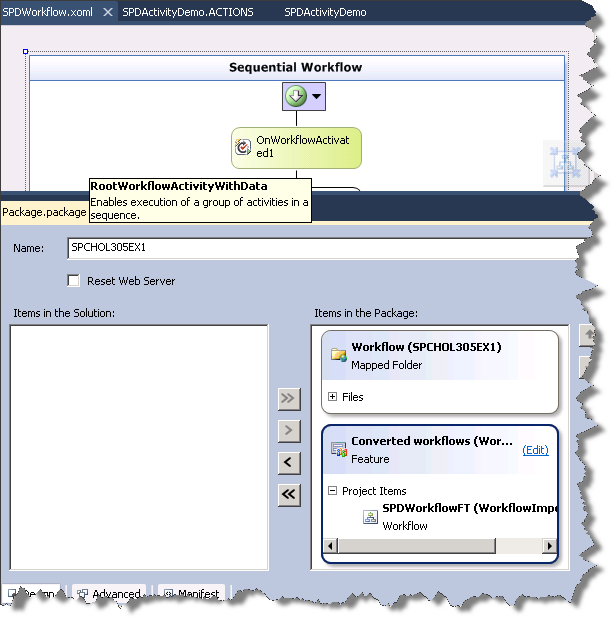


Figure 32 - SharePoint Package Explorer

1. Right-click on the **SPDWorkflowImport** project and select **Add Reference**
2. Select the project reference **SPDActivityDemo** and click **OK**.

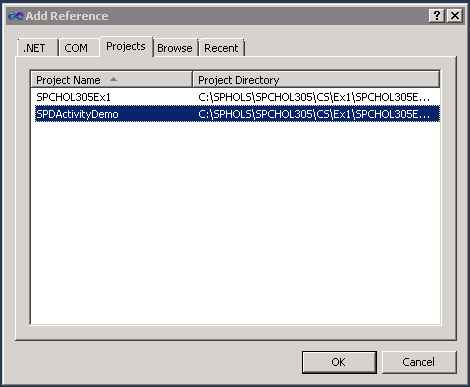


Figure 33 - Add Assembly Reference

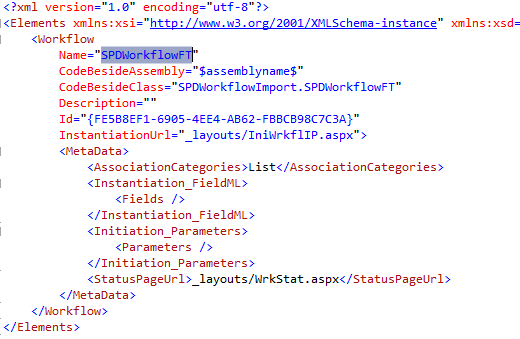
1. Under the **SPDWorkflowImport** project, expand the **Workflows\SPDWorkflowFT** node.
2. Open the **Elements.xml**
3. Change the workflow **Name** to **SPDWorkflowFT**

Figure 34 - Setting the imported workflow name

1. Save the project, build the solution and deploy the **SPCHOL305Ex1** project.

### Task 10: Associate the workflow with a list in SharePoint

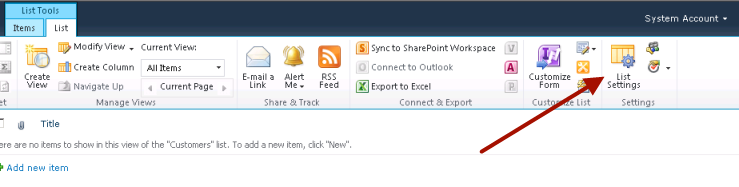
1. Browse to <http://intranet.contoso.com>.
2. Click on **Site Actions | More Options**
3. Select **Custom List** from the items list.
4. Name the list *Customers* and press **Create**.
5. Open List Settings

Figure 35 - List Settings

1. Under **Permissions and Management** click **Workflow Settings.**
2. Configure the list workflow settings as follows:
   * Select a workflow template: **SPDWorkflowFT**
   * Type a unique name for this workflow: **Create Doc Lib for Customer**
   * Select a task list: **Tasks**
   * Select a history list: **Workflow History**
   * Start options: **Start this workflow when a new item is created**

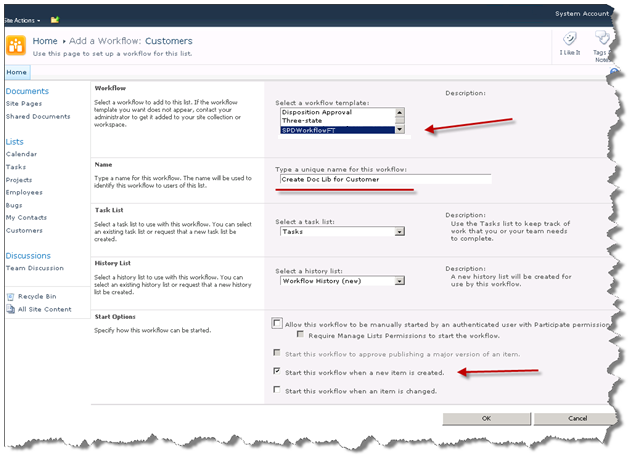


Figure 36 - Workflow Settings

1. Click **OK**
2. Navigate to the **Customers** list and select **Items | New Item**.

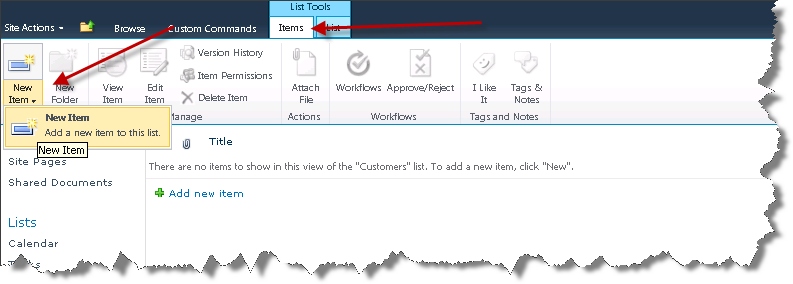


Figure 37 - Create New Item

1. Enter **Northwind** as name of the new customer and press Save when the **Customers – New Item** dialog appears.

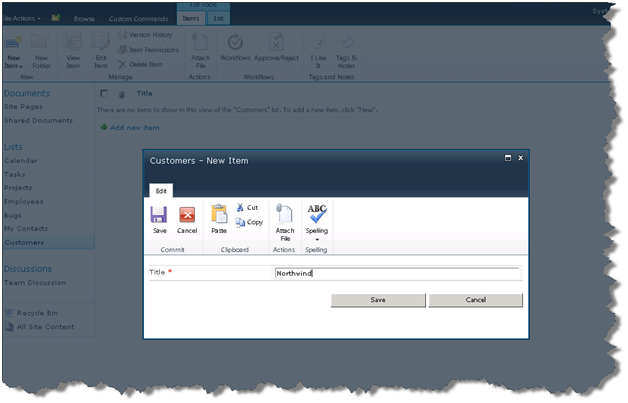


Figure 38 - New Item Dialog

1. Wait while a new document library of the same name should be created automatically.

### Exercise Summary

In this exercise you built a new Workflow activity for use in SharePoint Designer workflows. You built a reusable workflow in SharePoint Designer and used the custom activity. You then imported that reusable workflow into Visual Studio 2010 and deployed the final workflow back into SharePoint.

## Exercise 2

This exercise shows how to create a SharePoint Workflow with an Initiation Form and process the form data.

### Task 1: Ensure there is a Customer list in the site

1. Browse to the site <http://intranet.contoso.com>
2. Check that a **Custom List** called **Customer** exists; if it does not exist create it.

### Task 2: Create and Prepare Project

In this task a project you will create a Sequential Workflow solution and use SharePoint user controls.

1. In Visual Studio 2010 and from the menu select **File | New | New Project**.
2. Select the **Visual C# | SharePoint | 2010 |** **Sequential Workflow** project template.
3. Change the **Name** to *SPCHOL305Ex2*.

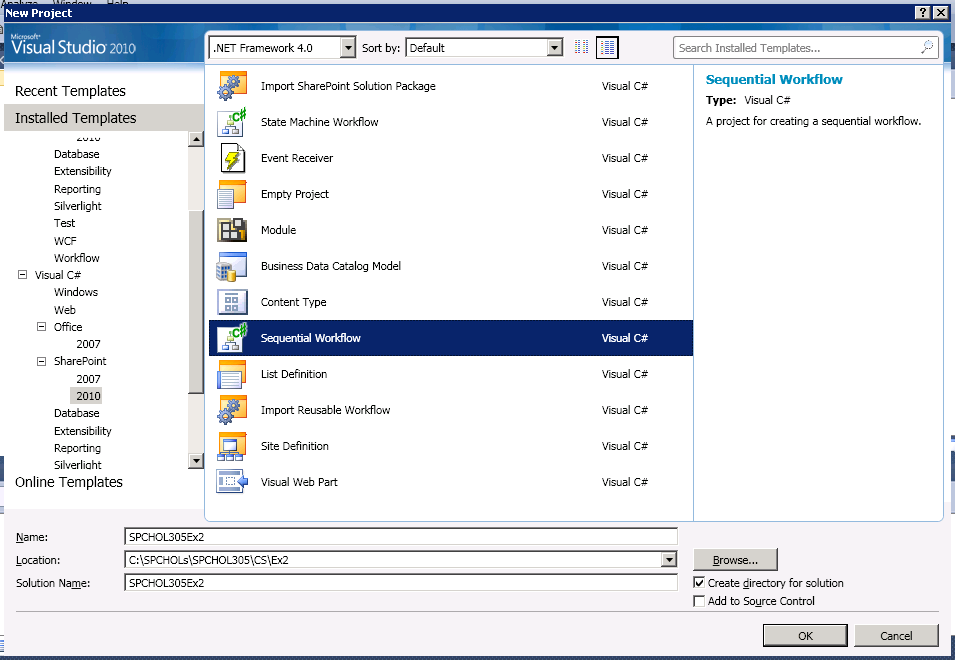


Figure 39 - New Project Dialog

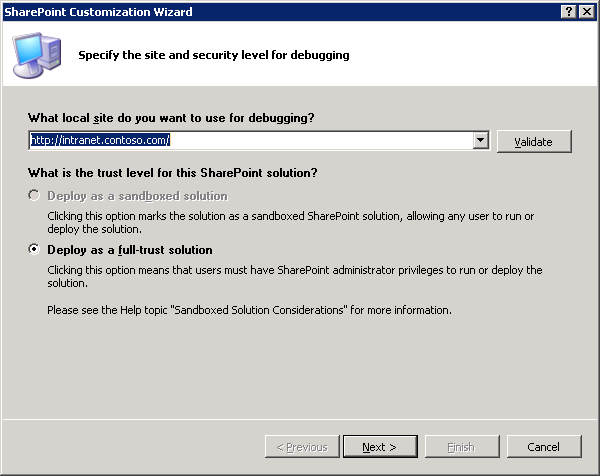
1. Click **Ok**.
2. Change the URL to *http://intranet.contoso.com*.  
   

Figure 40 - SharePoint Customization Wizard

Click **Next**.

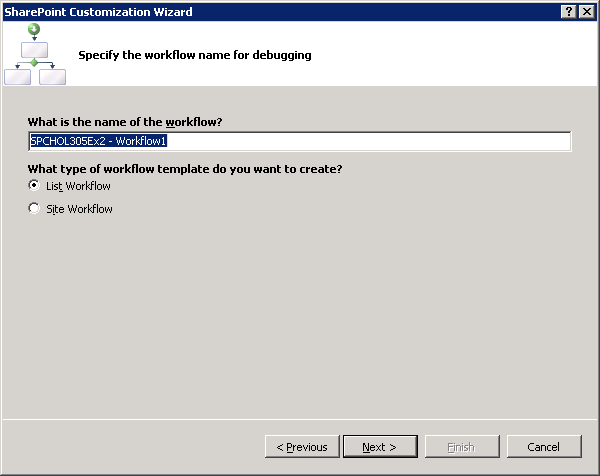


Figure 41 - SharePoint Customization Wizard

Click **Next.**

1. Change **The Library or list to associate your workflow with:** to **Customer**

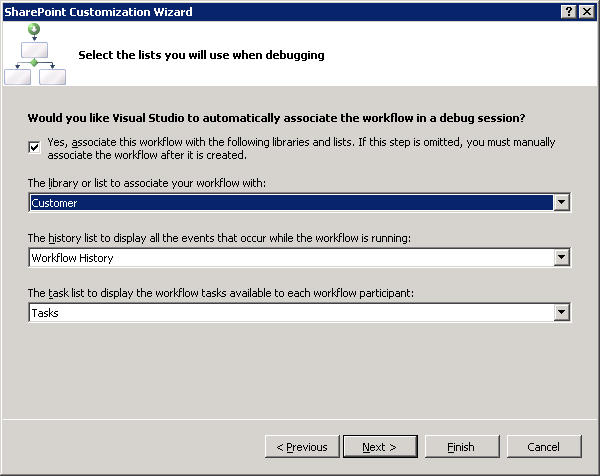


Figure 42 - SharePoint Customization Wizard

Click **Next.**

1. *Deselect* **The workflow starts automatically when an item is created**

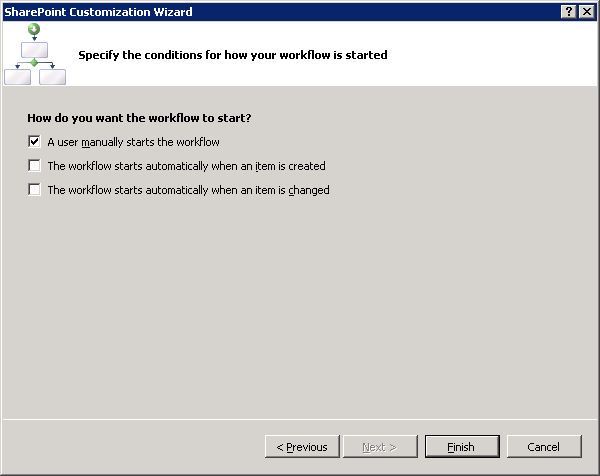


Figure 43 - SharePoint Customization Wizard

1. Click **Finish**.

### Task 2: Create a new Workflow Initiation Form to be used by the workflow.

1. **Right-click** Workflow1 in the *SPCHOL305Ex2* workflow project, **Add, New Item.**

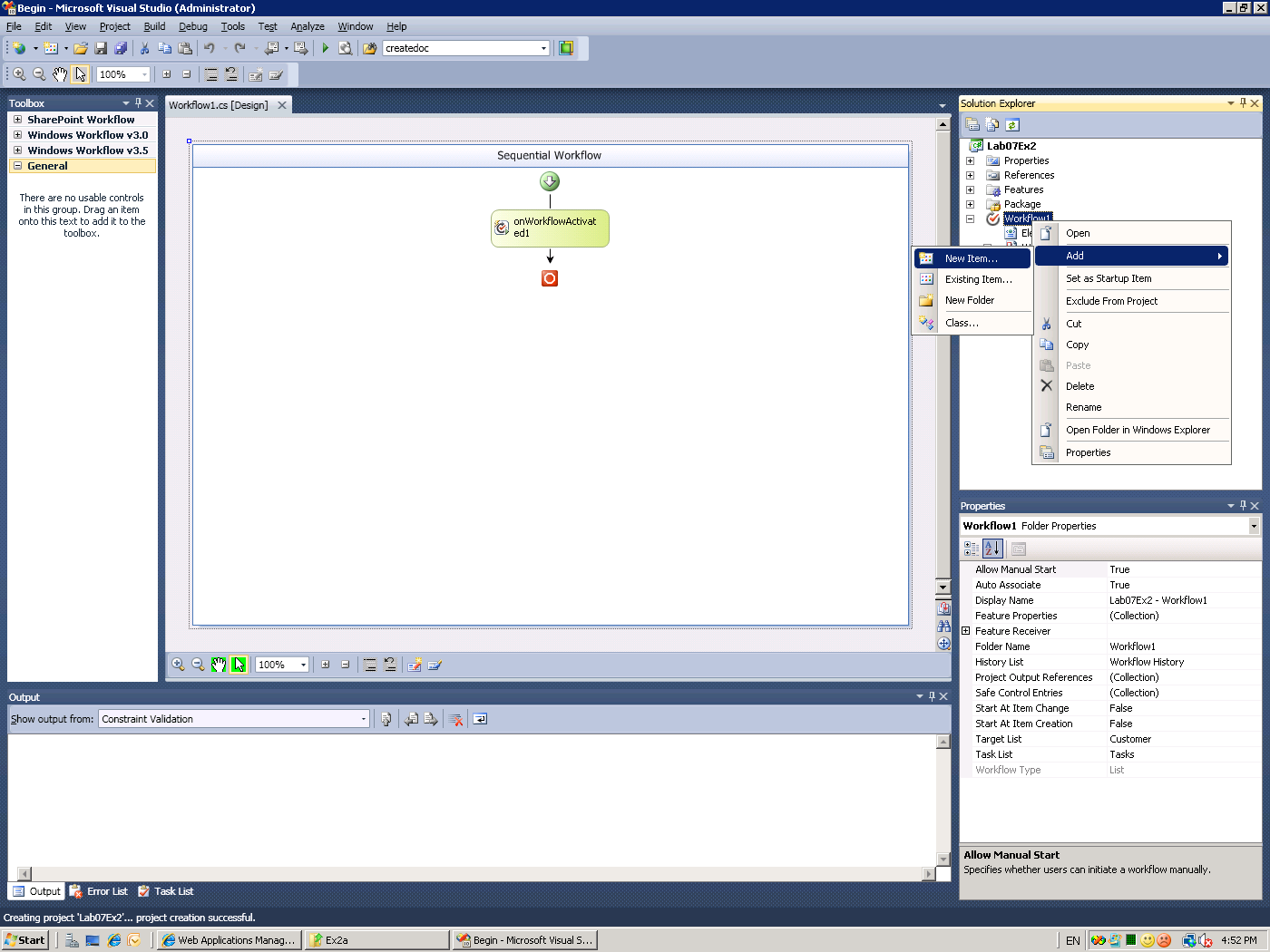


Figure 44 - Add New Item

1. Select the **Visual C# | SharePoint | 2010 |** **Workflow Initiation Form** item template.
2. Change the **Name** to *WorkflowIntiationForm.aspx*.

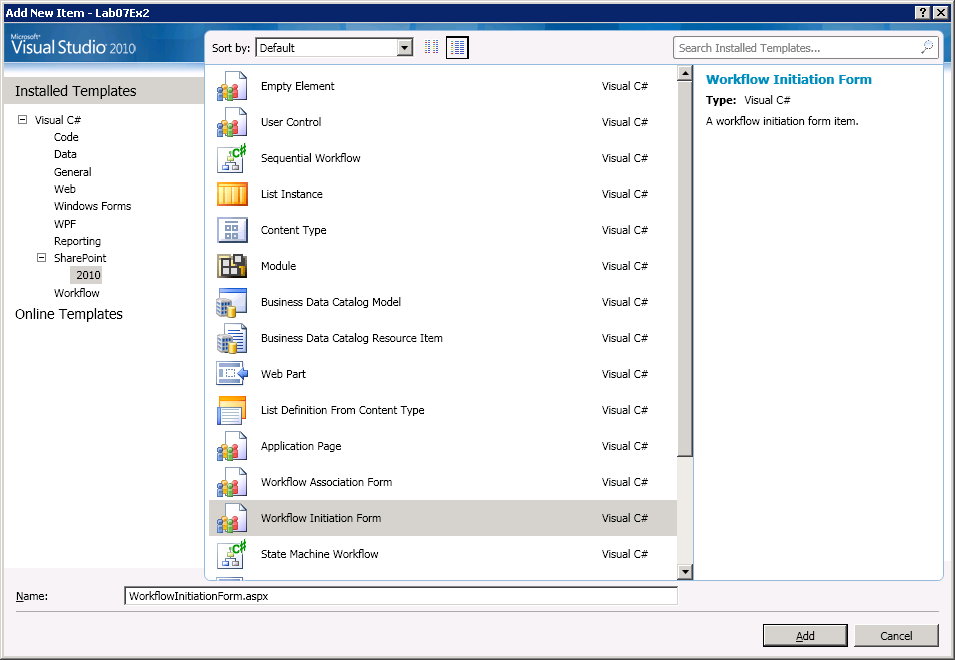


Figure 45 – New Item Dialog

1. Click **Add**.

### Task 3: Complete the Workflow Initiation Form.

1. Add the following code to the **Main** content placeholder

<br />

<br />

<br />

<asp:CheckBox ID="Checkbox" runat="server" Text="Fast Track Customer" />

**Code Snippet:** *My HTML Snippets |* **spchol305\_ex2\_initiationform\_html**

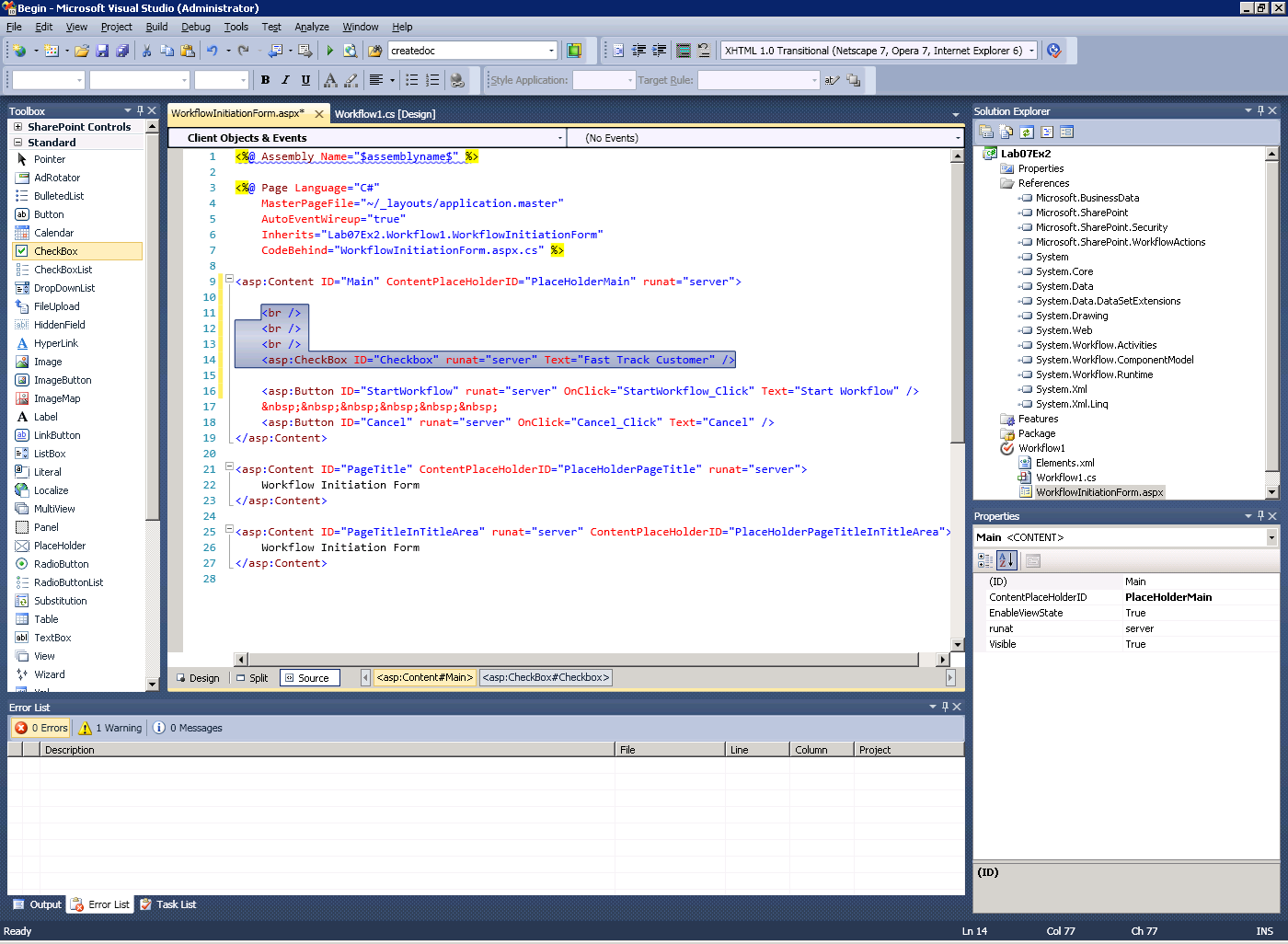
****

Figure 46 - Workflow Initiation Form

1. Open the **WorkflowInitiationForm.aspx.cs.** In the **GetInitiationData** method **replace** *return string.Empty;* with the following code

return Checkbox.Checked.ToString();

**Code Snippet:** *My Code Snippets |* **spchol305\_ex2\_workflowinitiationform**

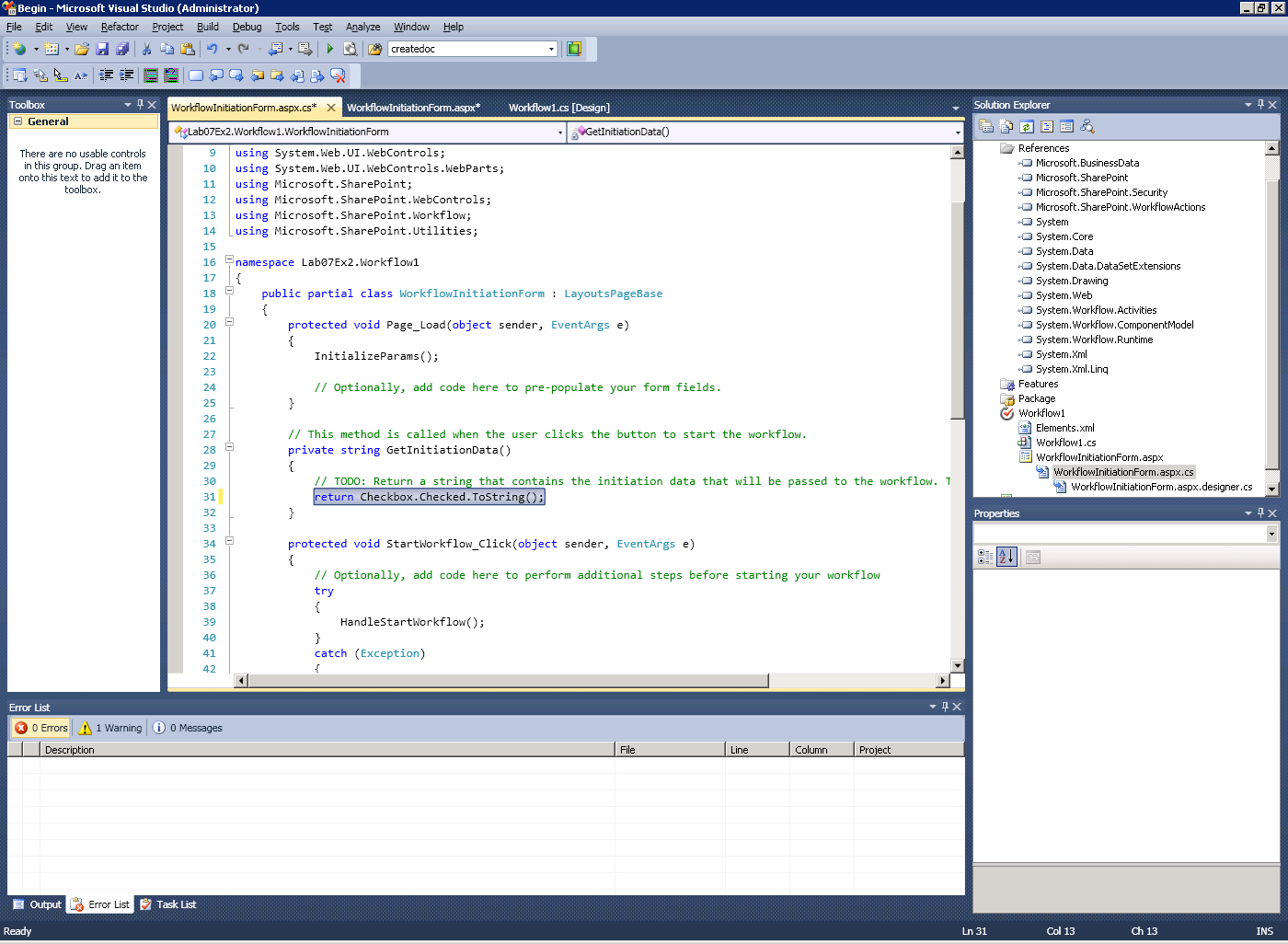


Figure 47 - Workflow Initiation Form Code

### Task 4: Process Workflow Initiation Form data in Workflow1

1. Open the **View** **Designer** of **Workflow1.cs.** From the **Toolbox,** add a **Code** activity (Windows Workflow v3.0) after the onWorkflowActivated1 activity.

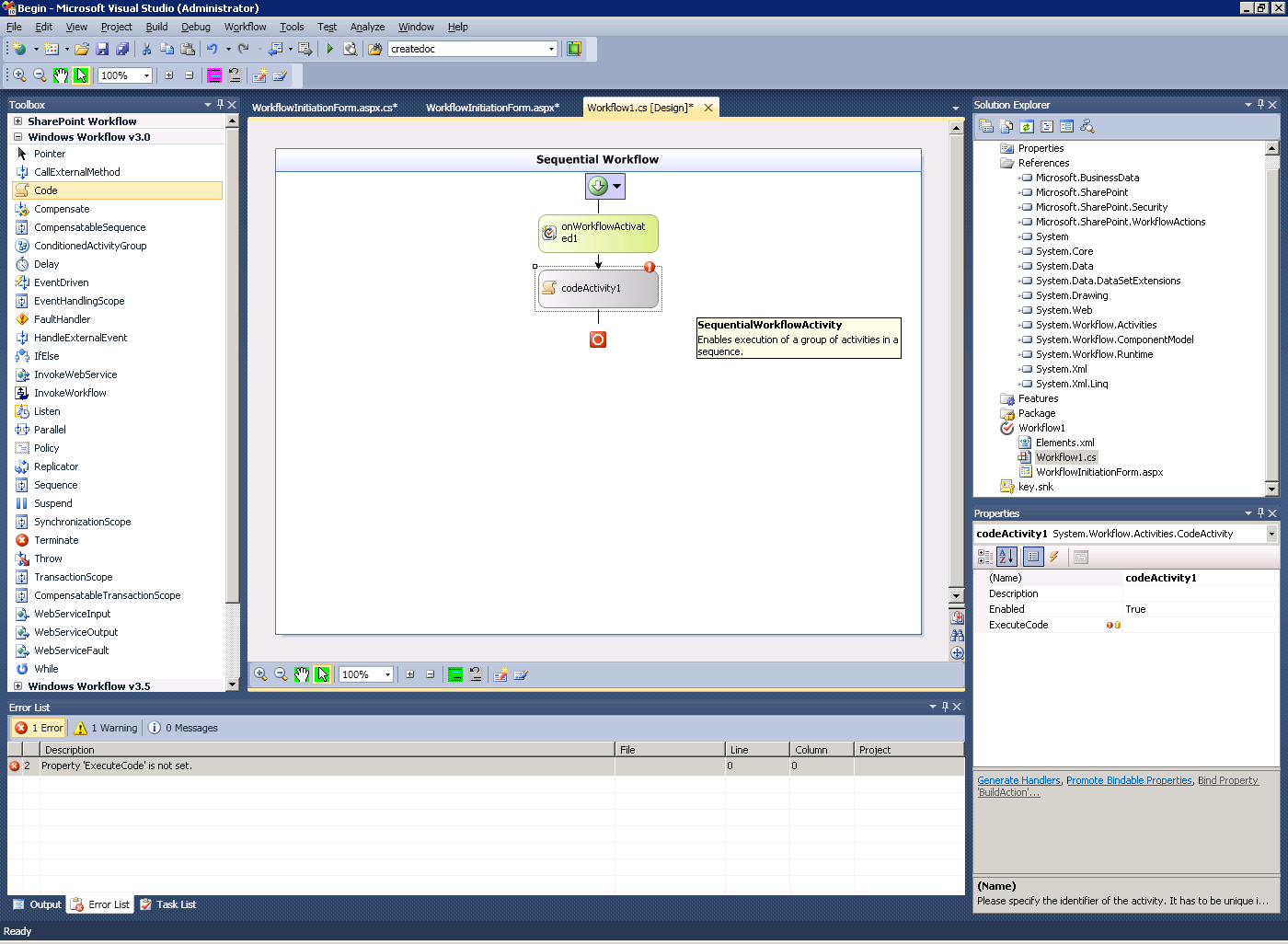


Figure 48 - Workflow Designer

1. **Double-click** **codeActivity1** to generate an ExecuteCode event in code behind. Add the following code to the codeActivity1\_ExecuteCode method. Set a breakpoint in the method.

if (workflowProperties.InitiationData == "True")

{

System.Diagnostics.Debug.WriteLine("This is a fast track customer");

}

**Code Snippet:** *My Code Snippets |* **spchol305\_ex2\_codeactivity1**

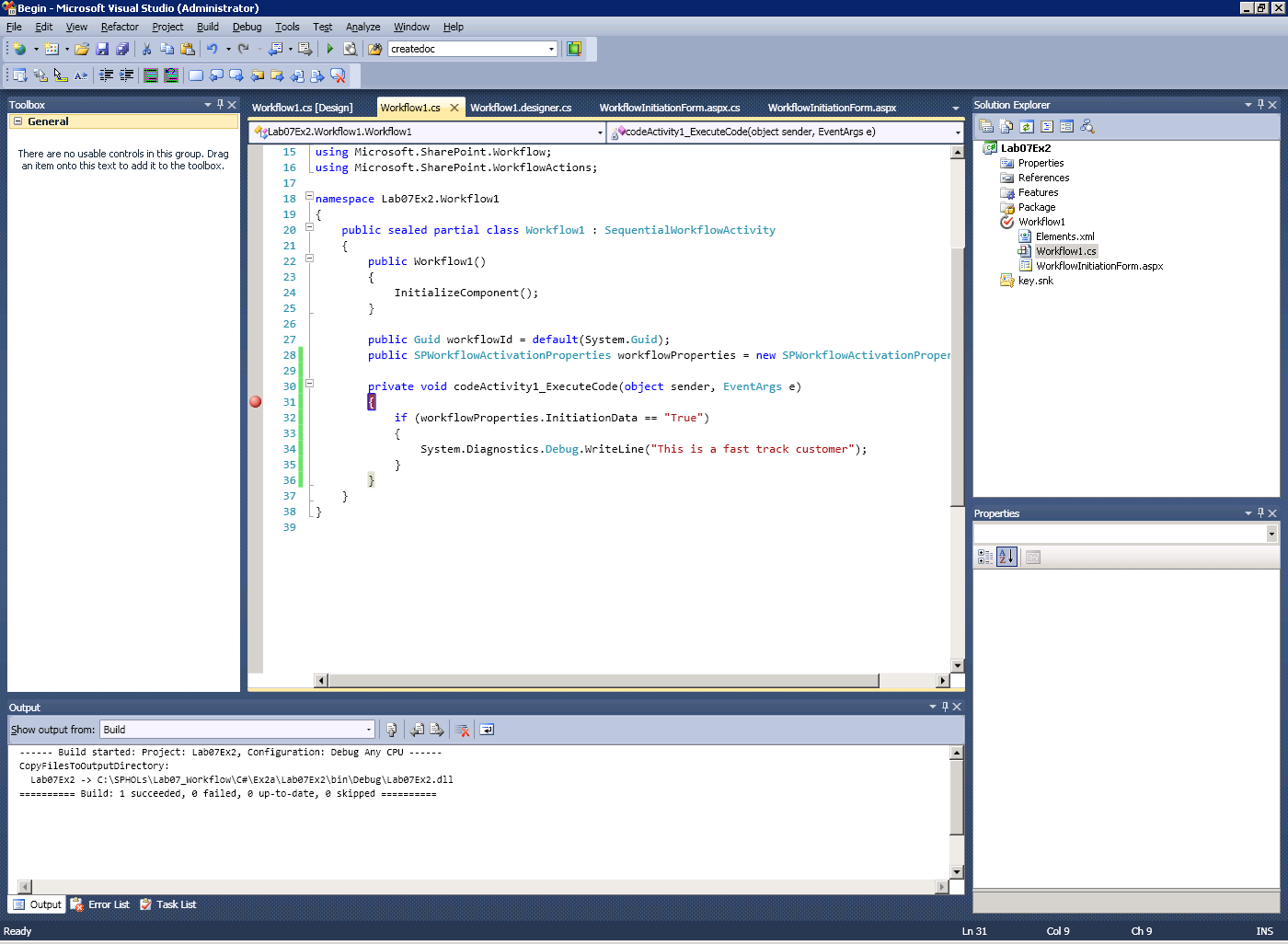


Figure 49 - Execute Code Method

### Task 5: Deploy and Debug Workflow1

1. Press **F5** to deploy and debug the new workflow
2. Navigate to the **Customer** list and create a new item in the list titled **Contoso**

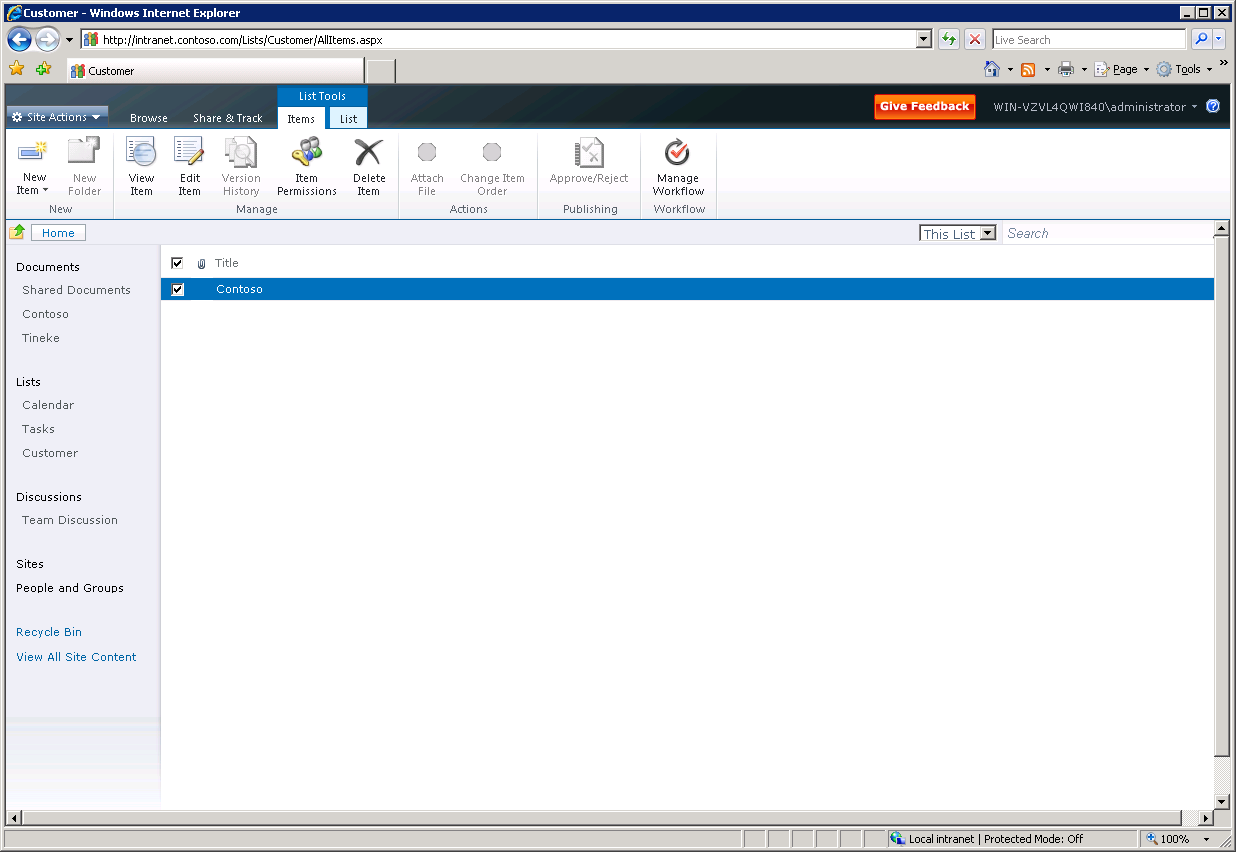


Figure 50 - Customer List

1. Click on the edit dropdown for the **Contoso** item and select **Workflows.** Start the **SPCHOL305Ex2****– Workflow1** on this item.

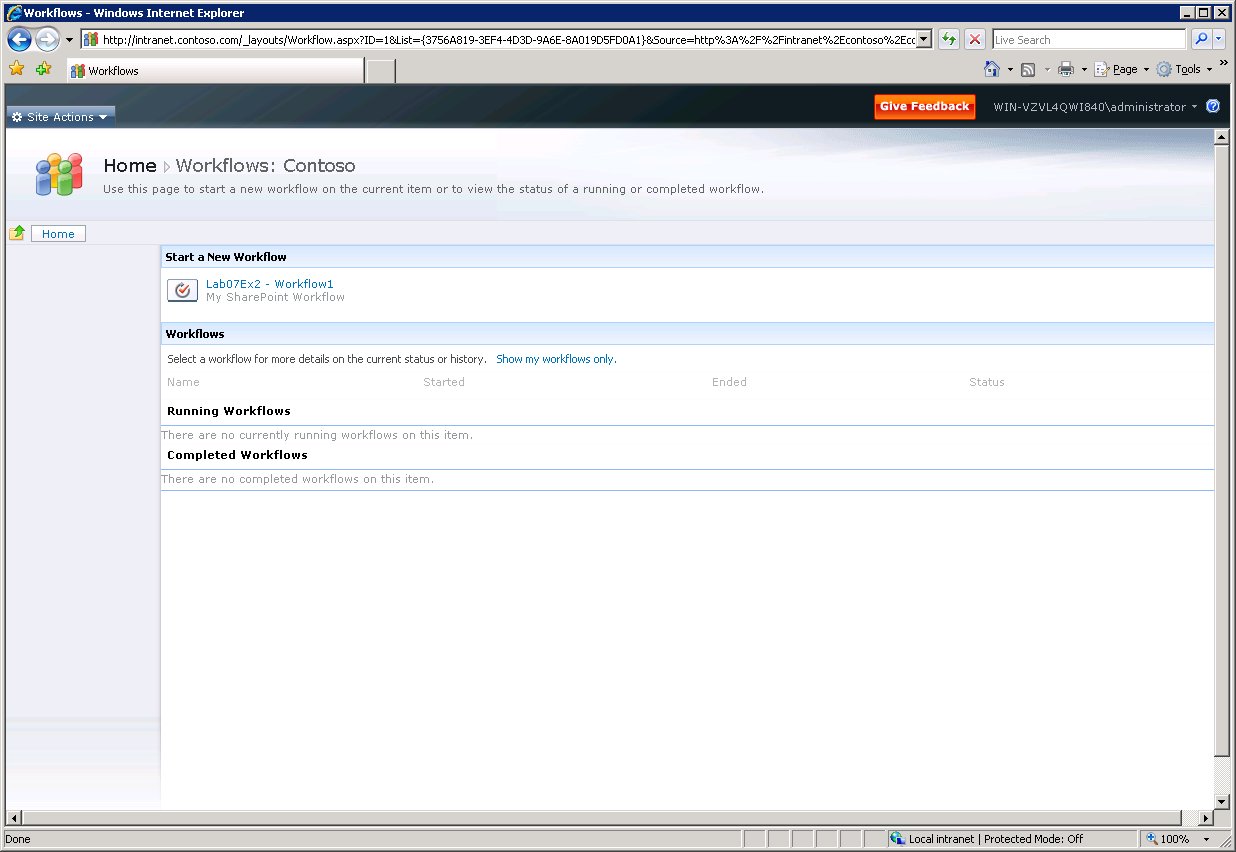


Figure 51 - Start Workflow

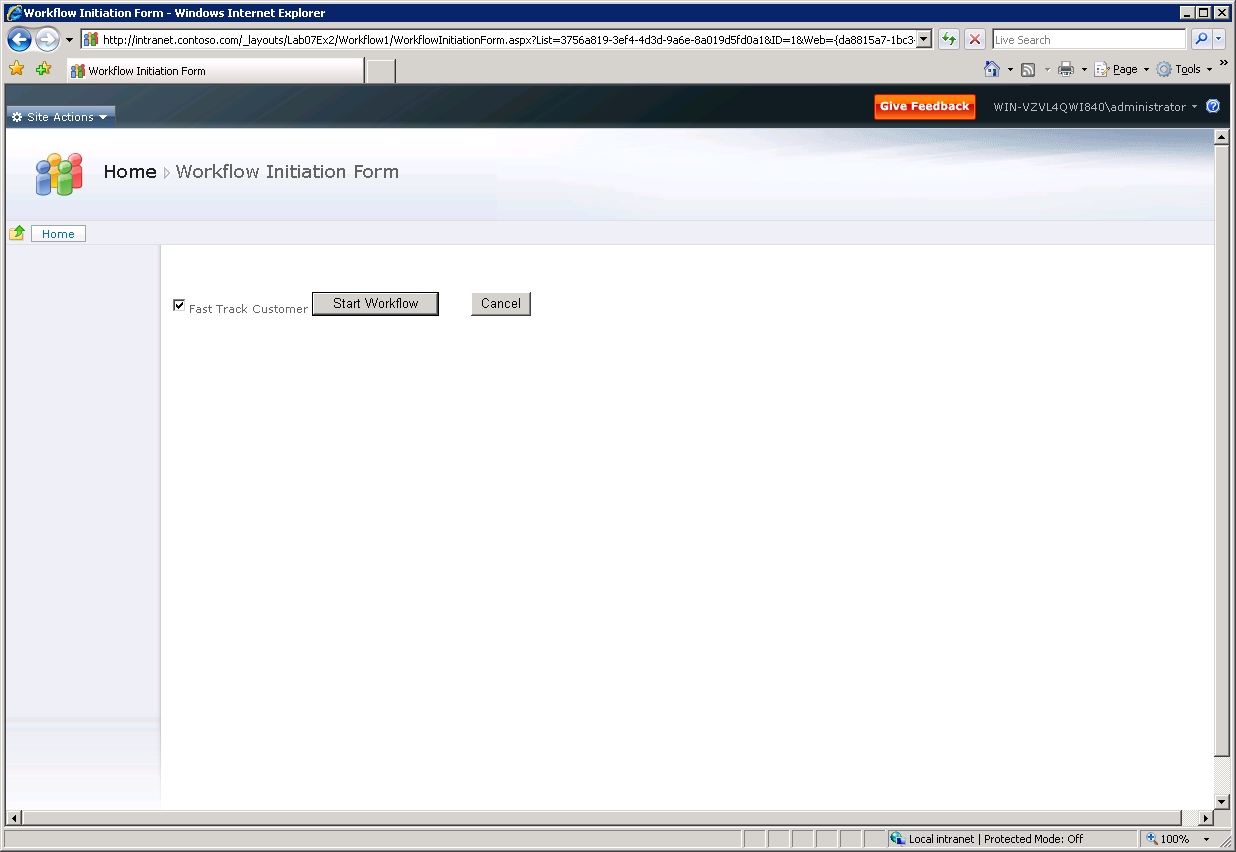
1. Check the Fast Track Customer **checkbox** and click the **Start Workflow** button.  
     
   

Figure 52 - Workflow Initiation Form

1. The break point we set in code should now be hit. Use **F10** to step through the code.

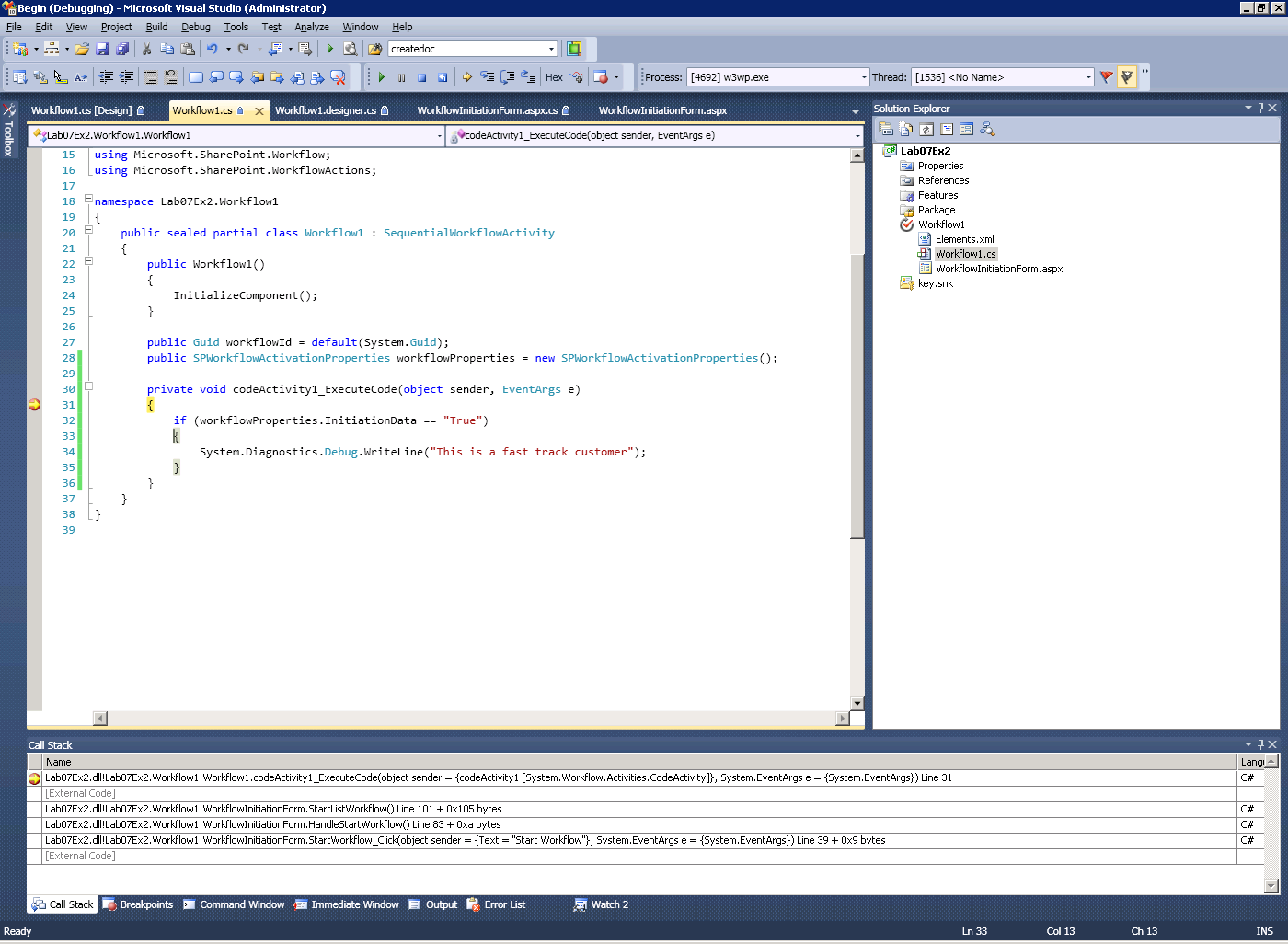


Figure 53 - Code Breakpoint

1. In the output window the value submitted in the Initiation Form has been captured

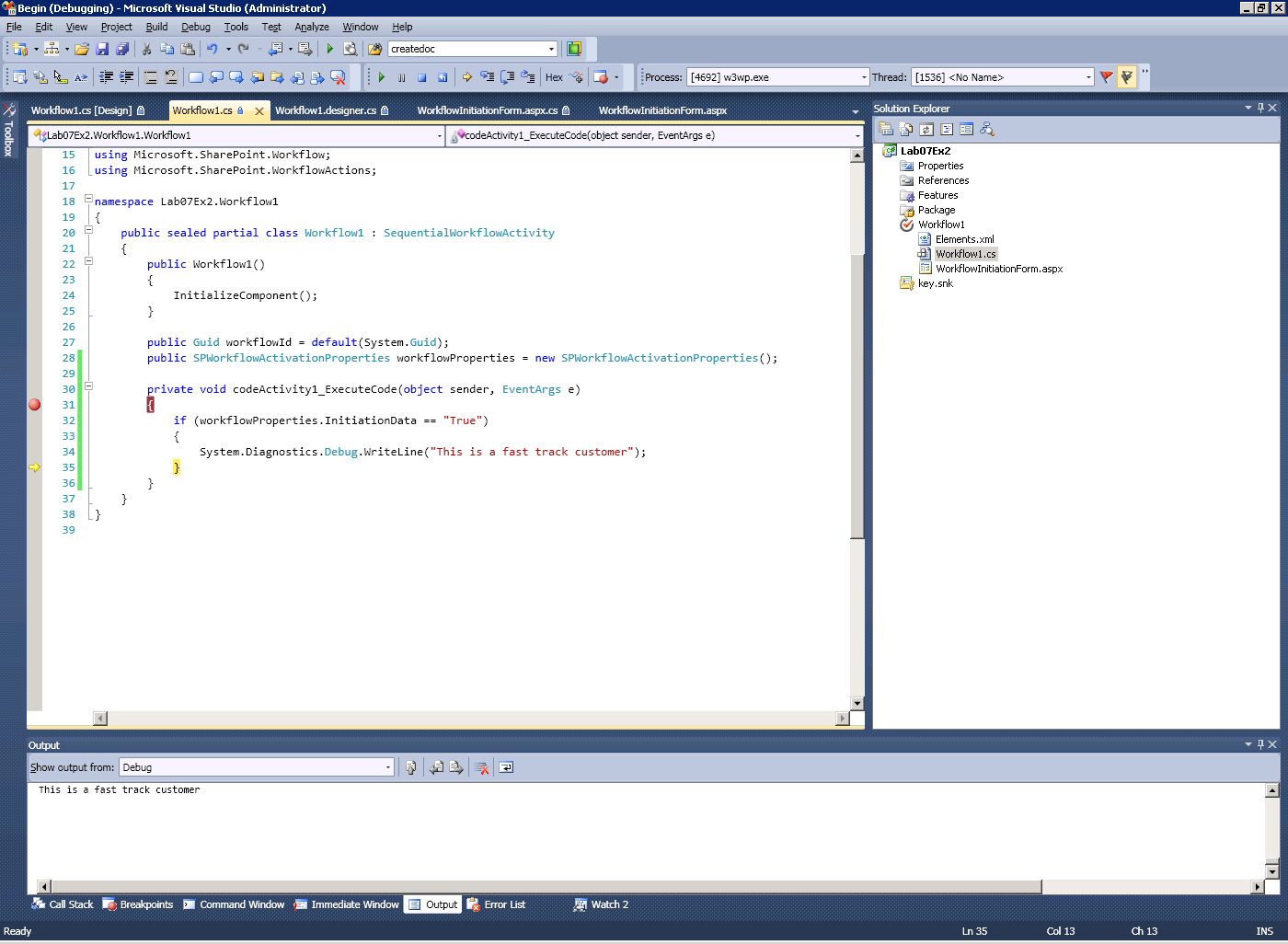
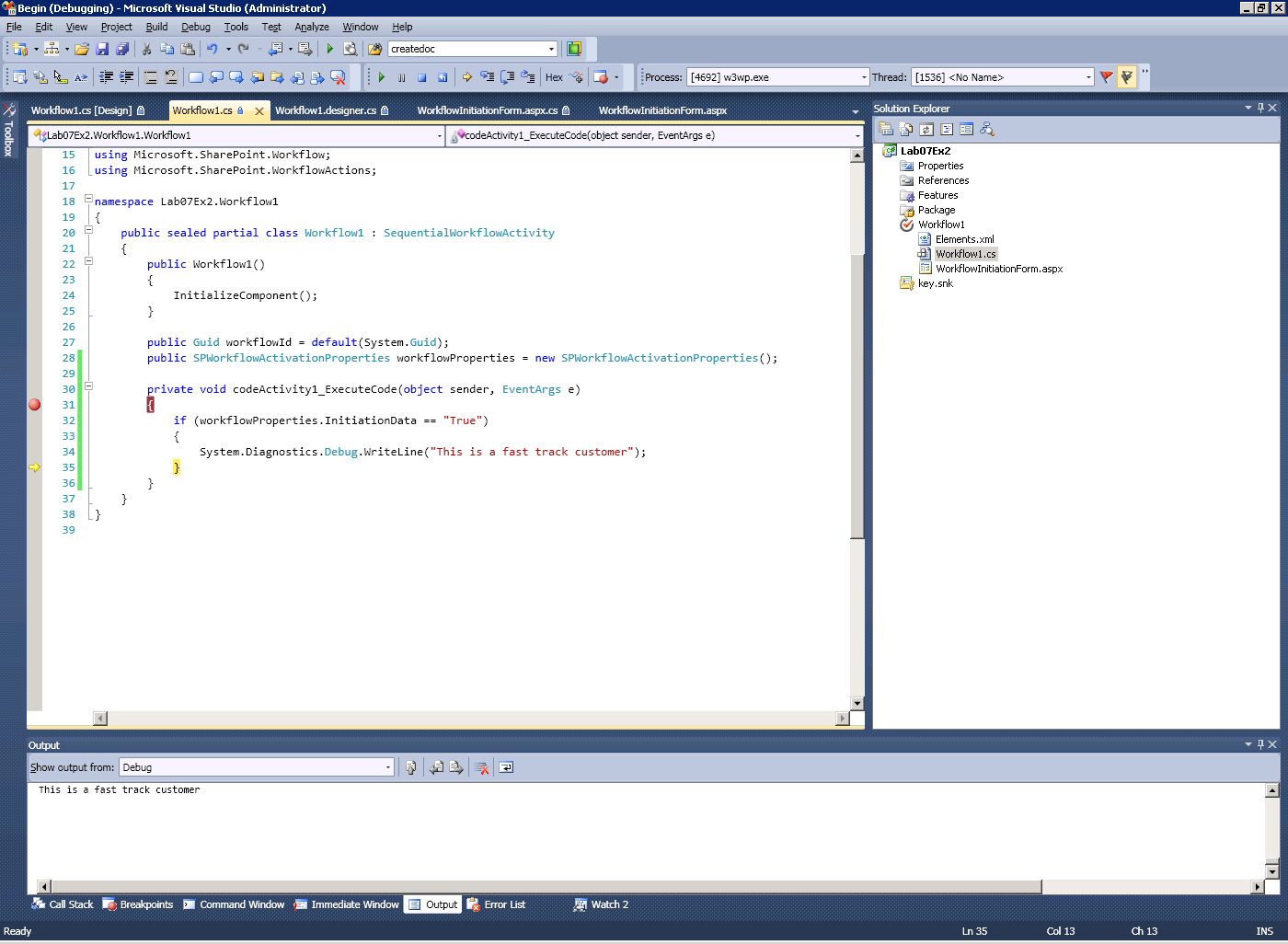


Figure 54 - Debug Output

1. Workflow complete

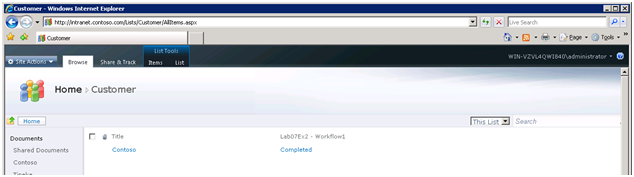


Figure 55 - Workflow Complete

### Exercise Summary

In this exercise you created a SharePoint Workflow with an Initiation Form and processed the form data.

## Lab Summary

In this lab you performed the following exercises.

Created a Workflow Action within Visual Studio 2010

Created an Empty SharePoint Project and included the Workflow Action in this solution

Deployed the Workflow Action as a WSP solution package

Used the Workflow Action to create a SharePoint Designer 2010 Reusable Workflow

Used SharePoint Designer to create a WSP package that can be imported back into Visual Studio 2010.

Imported a SharePoint Designer 2010 WSP in Visual Studio 2010.

Used Visual Studio 2010 to create an Initiation Form and associate this with a SharePoint Workflow

In this lab, you learned how to create a SharePoint Workflow Action. You learned how to include this Action in a SharePoint WSP solution package. You also used SharePoint Designer 2010 to create a reusable workflow has used the Visual Studio Workflow Action. You then use SharePoint Designer to generate a WSP that was imported back into Visual Studio.

You also used an Initiation form to capture user data and processed this data in the workflow.