Microsoft® SharePoint® 2013

HOL010: Building Visual Studio-Based Workflows in SharePoint 2013

Hands-On Lab

Lab Manual

Contents

Building Visual Studio-Based Workflows in SharePoint 2013

Estimated time to complete this lab: **60 minutes**

Lab Objectives

The workflow model in SharePoint 2013 Preview edition has changed. All workflows, regardless of where they are designed are now declarative, with XAML being the only artifact uploaded to the server. This lab walks through the basics of creating a workflow for SharePoint Online 2013 Preview using Visual Studio 2012.

This lab will introduce you to the declarative development environment inside of Visual Studio. It will also introduce some new SharePoint 2013 Workflow features, such as the ability to query REST web services.

Technologies

- SharePoint 2013
- Visual Studio 2012

Audience

- SharePoint Developers
- SharePoint IT Professionals

Scenario

In this lab, you will build a workflow that will walk through certain aspects of a document creation process. It will watch for a status field to be updated to “Approved for Publishing.” Once that state is hit, the workflow will query the public Northwind OData API to retrieve a customer name from a customer ID. Finally, it will update the list item with the customer name retrieved from the web service.

Getting Started

Connect to the Lab Environment

Note: This lab exercise involves using an HTTP activity in a workflow that will obtain content from an externally hosted Web service. Therefore the hands on lab computer running the workflow must have internet connectivity enabled and correctly configured in order to complete this lab.

Consult your system administrator or lab coordinator to obtain the login information needed to connect to the lab environment.

Open the Lab Environment

To begin the lab log into your VM with the following credentials:

Username: **CONTOSO\garthf**
Password: **pass@word1**

Exercise 1 – Add the Necessary SharePoint Objects

The solution calls for two document libraries:

- Contracts – The library where users work on documents in progress.

The solution also calls for two site columns

- Reviewer – Person designated as the reviewer for the document.
- Approval Status – What state the document is in.

Estimated time to complete this exercise: **20 minutes**

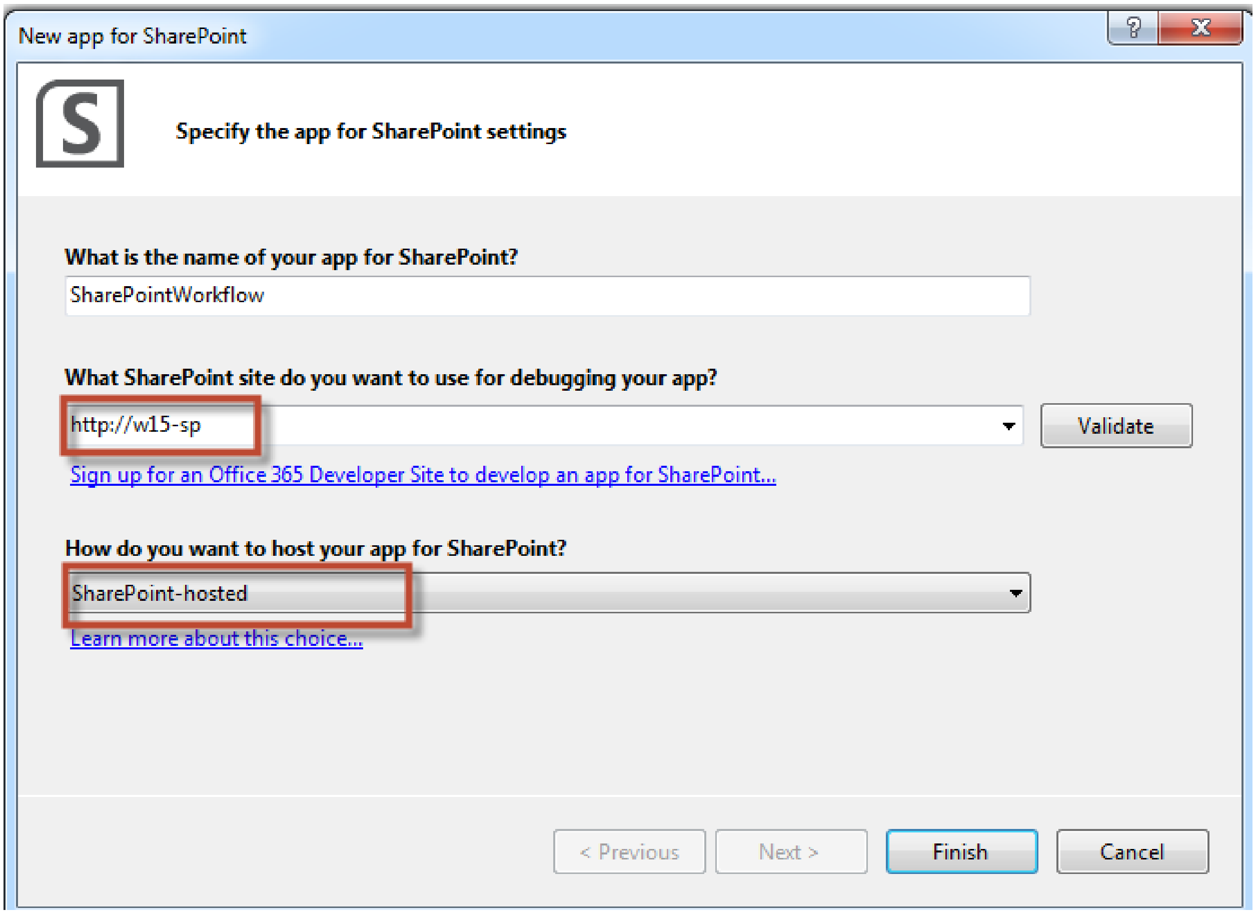
Scenario

In this exercise, you will create a SharePoint app, add columns, modify the columns, and add a Document Library with site columns.

Task 1 -- Create the App for SharePoint 2013 project

In this task, you will create a SharePoint app project.

1. From the **Start Menu**, select **All Programs**, and then click **Microsoft Visual Studio 2012**.
2. Select **File**, **New** and then click **Project**.
3. Under Templates, navigate to **Visual C#**, then **Office/SharePoint**, and then click **Apps**.
4. Select **App for SharePoint 2013**.
5. Name the app **SharePointWorkflow**.
6. Click **OK**.
7. In the **New App for SharePoint** window, enter **http://w15-sp** the URL for the local SharePoint server.
8. Choose to create a **SharePoint-hosted** app.


New app for SharePoint
?
X

Specify the app for SharePoint settings

What is the name of your app for SharePoint?

SharePointWorkflow

What SharePoint site do you want to use for debugging your app?

http://w15-sp

Validate
Sign up for an Office 365 Developer Site to develop an app for SharePoint...

How do you want to host your app for SharePoint?

SharePoint-hosted

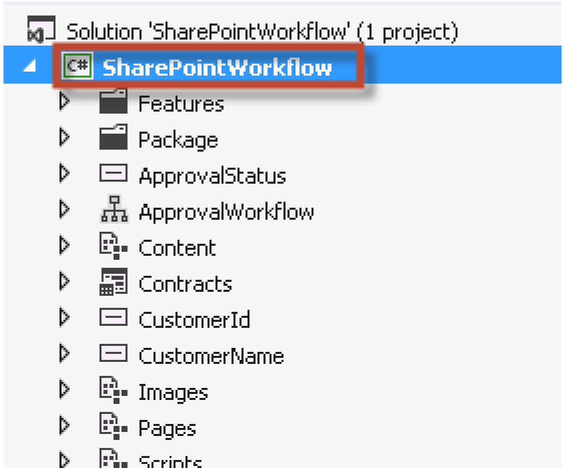
Learn more about this choice...
< Previous
Next >
Finish
Cancel

9. Click **Finish** to create the project.
   This may take several minutes.

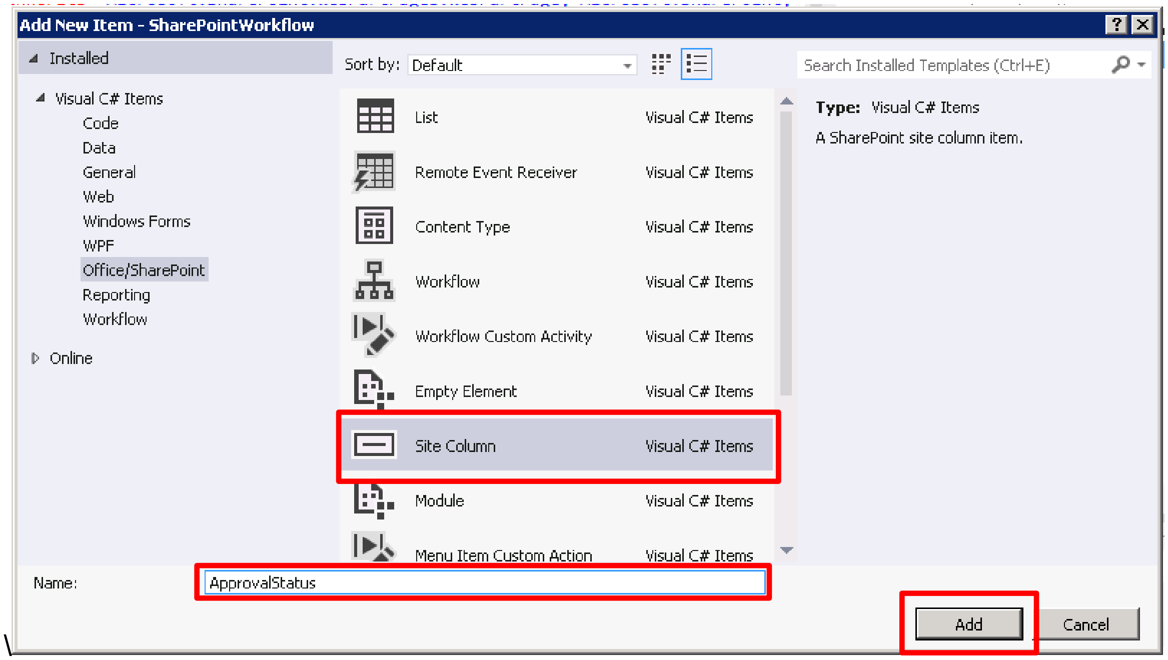
Task 2 – Add the Column Definitions to the project

Now that the project has been created, you will add and configure the site column definitions that the libraries will use in this task.

1. In **Solution Explorer**, right-click your project and select **Add**, and then click **New Item**.



2. Expand the **Visual C#** Items node and select **Office/SharePoint** and then click **Site Column**.
3. Name the site column **ApprovalStatus** and click **Add**.



Type: Visual C# Items

A SharePoint site column item.

Name: ApprovalStatus

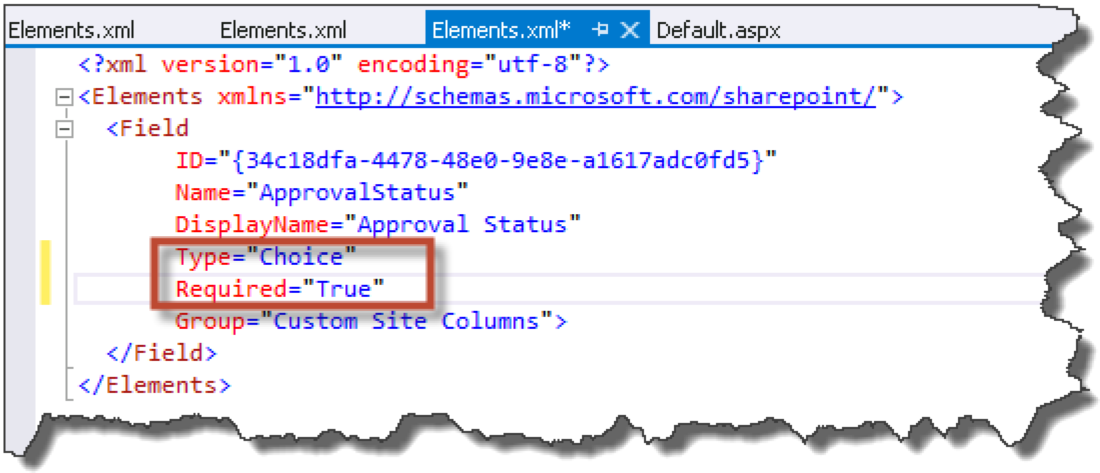
4. Repeat steps 1- 3 and add two additional site columns named:
   - **CustomerId**
   - **CustomerName**

Task 3 – Modify the Site Column Definitions

In this task you will modify the Site Column definitions.

1. In **Solution Explorer**, expand the **ApprovalStatus** site column and double-click **Elements.xml** to load it.

2. In the XML, change the **Type** attribute value from **Text** to **Choice** and the **Required** attribute to **True**.



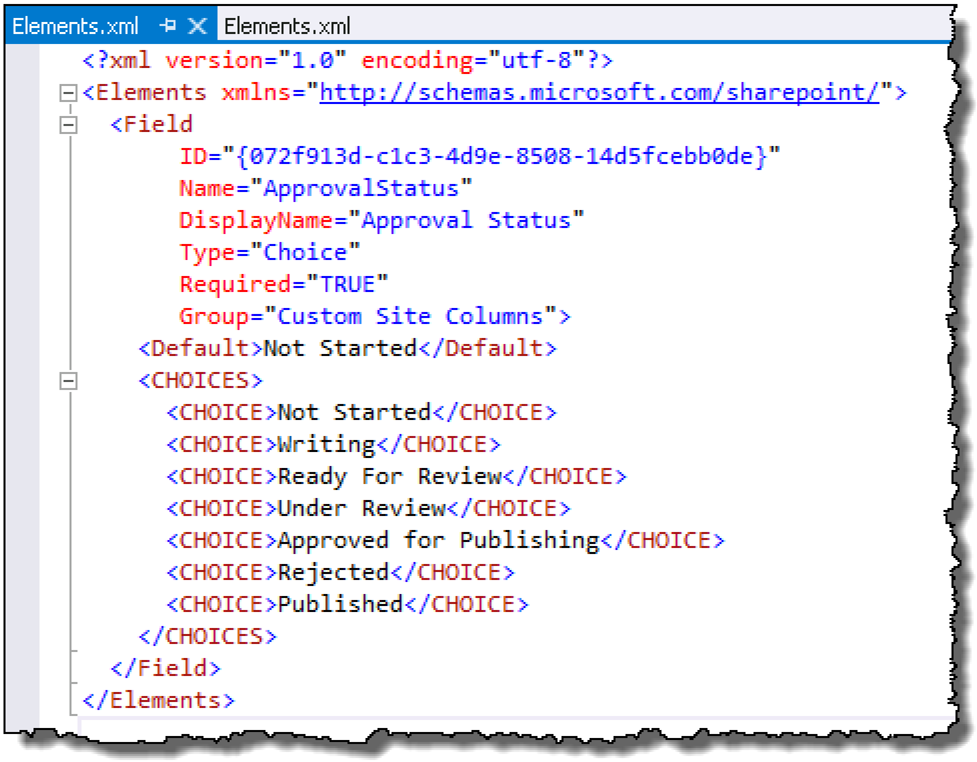
```
<?xml version="1.0" encoding="utf-8"?>
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
  <Field
       ID="{34c18dfa-4478-48e0-9e8e-a1617adc0fd5}"
       Name="ApprovalStatus"
       DisplayName="Approval Status"
       Type="Choice"
       Required="True"
       Group="Custom Site Columns">
  </Field>
</Elements>
```

   a. Finally, add the Default and Choices elements at the bottom of the Field node. **Right-click** above `</Field>`.

   b. Select **Insert Snippet**, then double-click **HOLSnippets**, double-click **HOL010** and then double-click **Snippet 1**.

   Note: All C #and HTML code samples in this lab are pre-loaded into Visual Studio 2012 code snippets that you can insert using these steps.

```
<Default>Not Started</Default>
<CHOICES>
   <CHOICE>Not Started</CHOICE>
   <CHOICE>Writing</CHOICE>
   <CHOICE>Ready For Review</CHOICE>
   <CHOICE>Under Review</CHOICE>
   <CHOICE>Approved For Publishing</CHOICE>
   <CHOICE>Rejected</CHOICE>
   <CHOICE>Published</CHOICE>
</CHOICES>
```



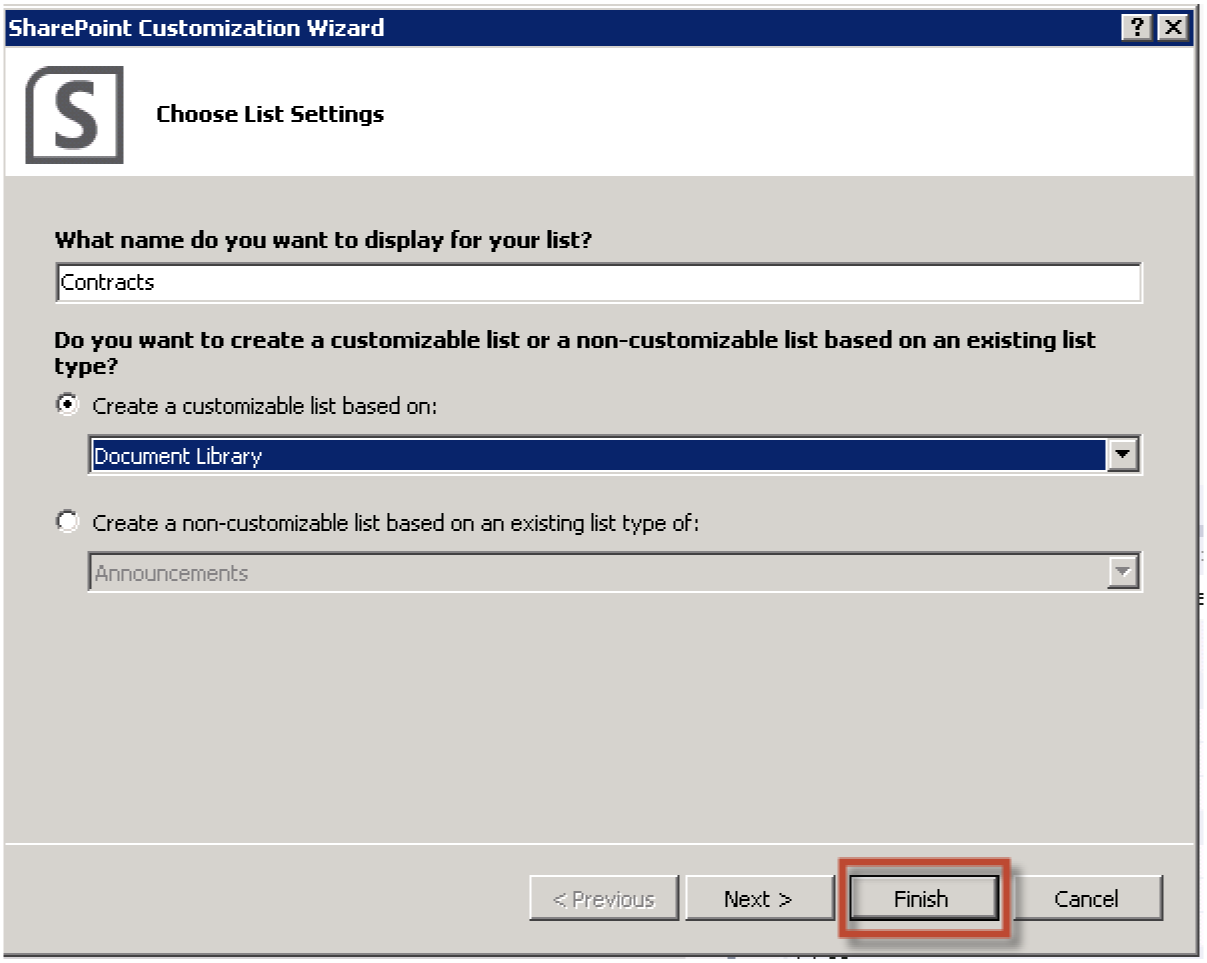
```
<?xml version="1.0" encoding="utf-8"?>
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
  <Field
       ID="{072f913d-c1c3-4d9e-8508-14d5fcebb0de}"
       Name="ApprovalStatus"
       DisplayName="Approval Status"
       Type="Choice"
       Required="TRUE"
       Group="Custom Site Columns">
    <Default>Not Started</Default>
    <CHOICES>
      <CHOICE>Not Started</CHOICE>
      <CHOICE>Writing</CHOICE>
      <CHOICE>Ready For Review</CHOICE>
      <CHOICE>Under Review</CHOICE>
      <CHOICE>Approved for Publishing</CHOICE>
      <CHOICE>Rejected</CHOICE>
      <CHOICE>Published</CHOICE>
    </CHOICES>
  </Field>
</Elements>
```

3. Save the **Elements.xml** file.

Task 4 – Add the Document Library

In this task you will add a list to the Document Library.

1. In **Solution Explorer**, right-click your project and select **Add**, **New Item**.
2. Under **Visual C# Items** and then **Office/SharePoint**, choose **List** from the available items.
3. Name the list **Contracts** and click **Add**.
4. In the Choose List Settings window, choose to create a customizable list based on **Document Library** and click **Finish**.



SharePoint Customization Wizard

Choose List Settings

What name do you want to display for your list?

Contracts

Do you want to create a customizable list or a non-customizable list based on an existing list type?

● Create a customizable list based on:

Document Library

○ Create a non-customizable list based on an existing list type of:

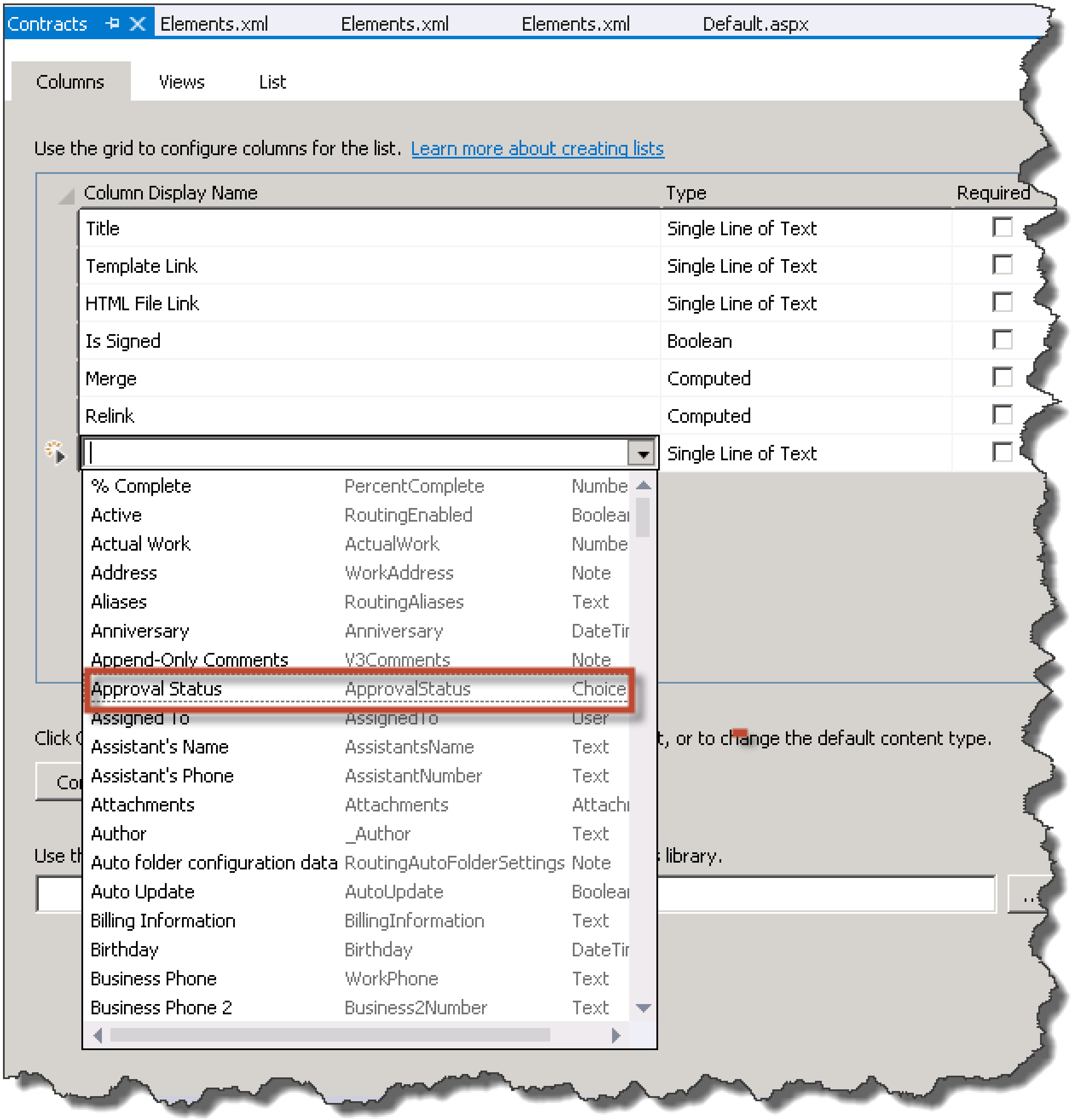
Announcements

Task 5 – Add the Site Columns to the Document Libraries

In this task, you will add site columns to the Document Library. In this lab:

- The contracts library will use all three of the site columns that you will add to the project.
- The approval status will be used to trigger the workflow.
- The CustomerId column will be used as input to the REST call to retrieve the customer details.
- The CustomerName column will be populated by the workflow based on the REST call results.

1. In **Solution Explorer**, double-click **Contracts** to load the editor.
2. Add the **Approval Status** column from the drop down list of columns.



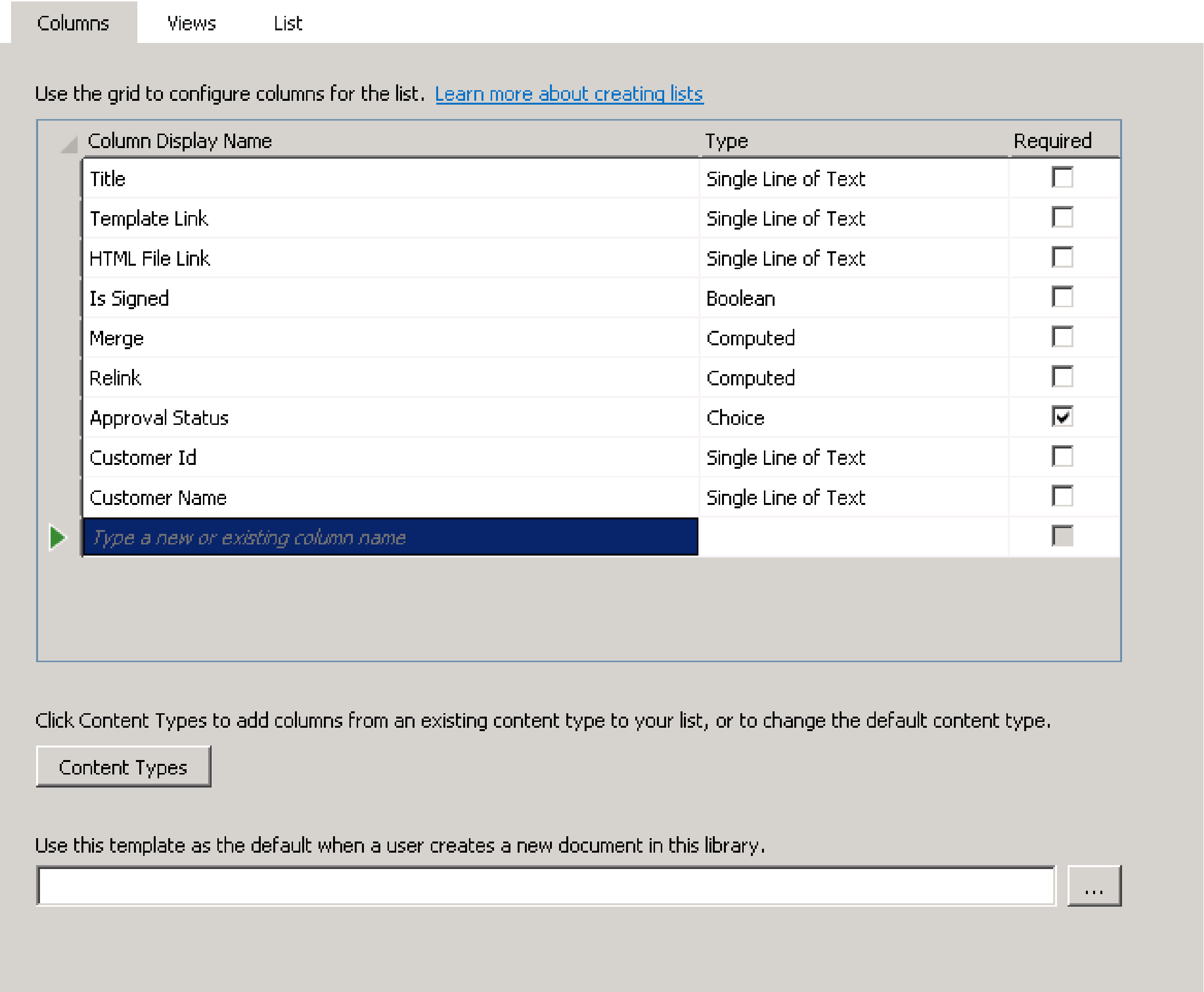
Use the grid to configure columns for the list. Learn more about creating lists

Column Display Name	Type	Required
Title	Single Line of Text	□
Template Link	Single Line of Text	□
HTML File Link	Single Line of Text	□
Is Signed	Boolean	□
Merge	Computed	□
Relink	Computed	□
	Single Line of Text	□

% Complete	PercentComplete	Numbe
Active	RoutingEnabled	Boolea
Actual Work	ActualWork	Numbe
Address	WorkAddress	Note
Aliases	RoutingAliases	Text
Anniversary	Anniversary	DateTi
Append-Only Comments	V3Comments	Note
Approval Status	ApprovalStatus	Choice
Assigned To	AssignedTo	User
Assistant's Name	AssistantsName	Text
Assistant's Phone	AssistantNumber	Text
Attachments	Attachments	Attach
Author	_Author	Text
Auto folder configuration data	RoutingAutoFolderSettings	Note
Auto Update	AutoUpdate	Boolea
Billing Information	BillingInformation	Text
Birthday	Birthday	DateTi
Business Phone	WorkPhone	Text
Business Phone 2	Business2Number	Text

3. Repeat the process for the following columns
    - Customer Id
    - Customer Name

    Your columns should now look like this:



Use the grid to configure columns for the list. Learn more about creating lists

Column Display Name	Type	Required
Title	Single Line of Text	□
Template Link	Single Line of Text	□
HTML File Link	Single Line of Text	□
Is Signed	Boolean	□
Merge	Computed	□
Relink	Computed	□
Approval Status	Choice	☑
Customer Id	Single Line of Text	□
Customer Name	Single Line of Text	□
Type a new or existing column name		□

Click Content Types to add columns from an existing content type to your list, or to change the default content type.

Use this template as the default when a user creates a new document in this library.

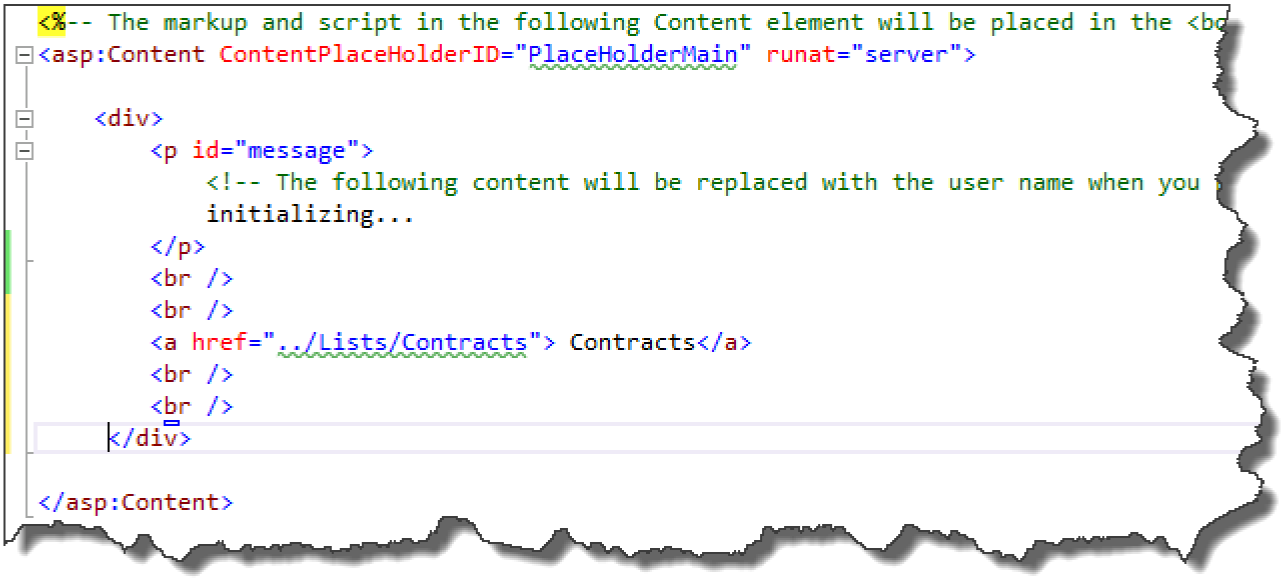
4. Save the list definition.

Task 6 – Add a link to the list in the Default.aspx page

Finally, you will need to add links to the library to the default page of the app so that users will be able to navigate to it. In this task, you will HTML code to the Default.aspx.

1. In **Solution Explorer**, expand **Pages** and double-click **Default.aspx**.
2. Add the following HTML to the `PlaceHolderMain` content placeholder, directly under the existing paragraph by navigating to **C:\Demo\HOL010\Snippets** and double-click **Snippet 03.**
    - Copy the text and paste content below `</p>`.

```
<br />
<br />
<a href="../Lists/Contracts"> Contracts</a>
<br />
<br />
```



```
<%-- The markup and script in the following Content element will be placed in the <bo
<asp:Content ContentPlaceHolderID="PlaceHolderMain" runat="server">

    <div>
        <p id="message">
            <!-- The following content will be replaced with the user name when you
            initializing...
        </p>
        <br />
        <br />
        <a href="../Lists/Contracts"> Contracts</a>
        <br />
        <br />
    </div>

</asp:Content>
```

3. Save the file.

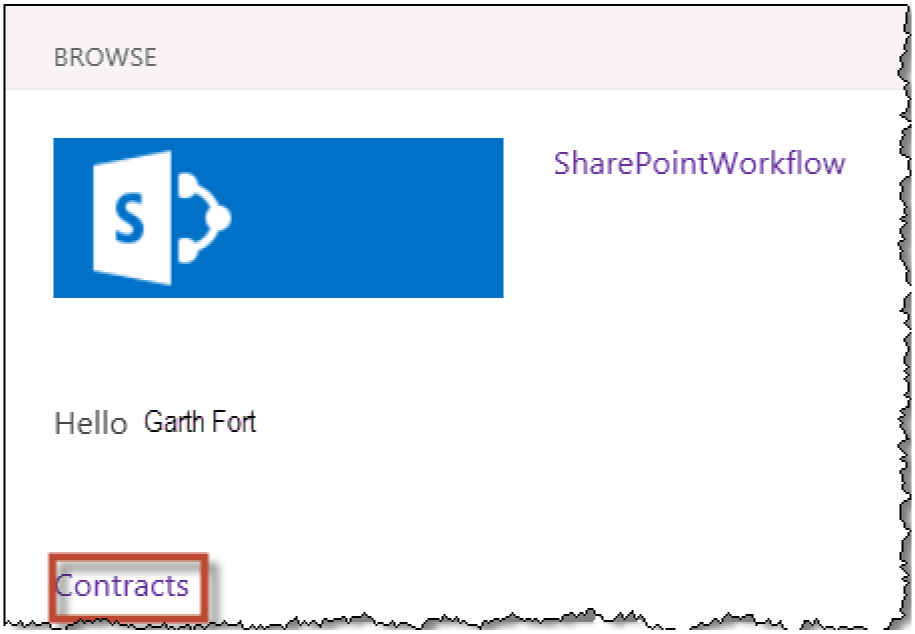
Task 7 – Debug your solution

In this task, you will debug your project. At this point all of the artifacts necessary for the workflow have been created. It is a good practice to make sure that everything is OK before you start building the workflow.

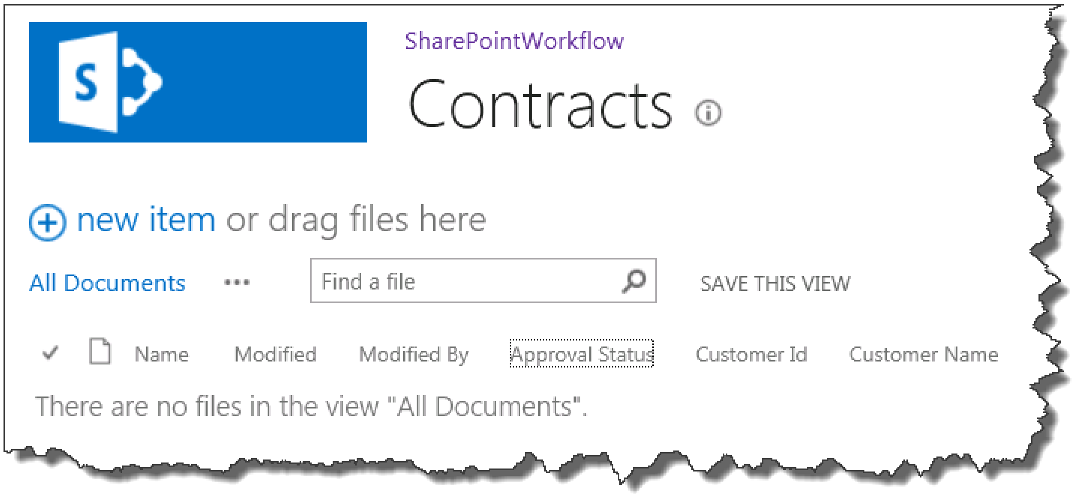
1. In the Visual Studio menu, select **Debug** and **Start Debugging** to start debugging. **Important Note:** do not use **F5** as it may cause problems in the HOL environment.

   Visual studio will automatically deploy your solution to the site

2. When Internet Explorer loads, click **Contract** on the **Site Contents** page.



3. Click each of the links to confirm that the lists loaded correctly.



There are no files in the view "All Documents".

4. When ready, close Internet Explorer to stop debugging.

In this exercise, you created a SharePoint app, added columns, modified the columns, and added a Document Library with site columns.

Exercise 2 – Creating the Workflow

Estimated time to complete this exercise: **40 minutes**

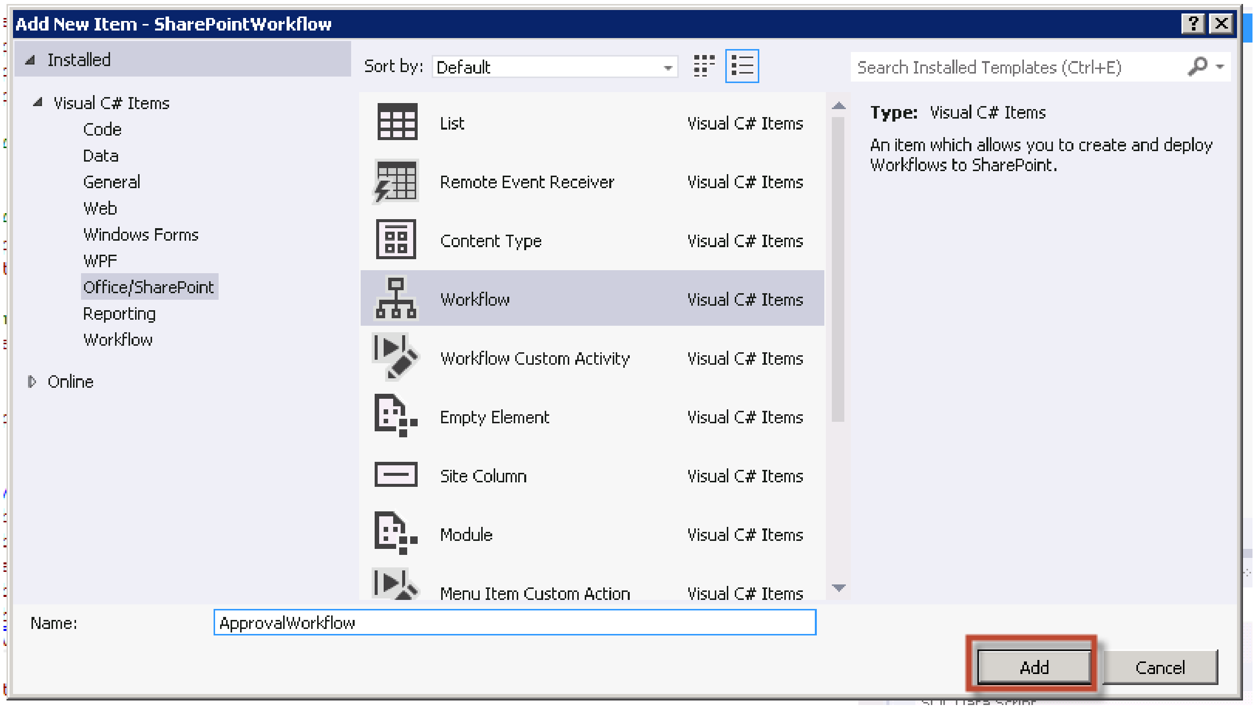
Scenario

Now that all of the SharePoint artifacts have been created, you can create the workflow. This workflow will run against the Contracts list and will update the Customer Name field once an item’s Approval Status is set to Ready For Publish.

Task 1 – Add the Workflow to the Project

In this task, you will add the workflow to the project.

1. In **Solution Explorer**, right-click your project and select **Add**, and then click **New Item**.
2. In the **Visual C# Items**, **Office/SharePoint** list, select **Workflow.**
3. Name the workflow, **ApprovalWorkflow**, and click **Add**.



Add New Item - SharePointWorkflow

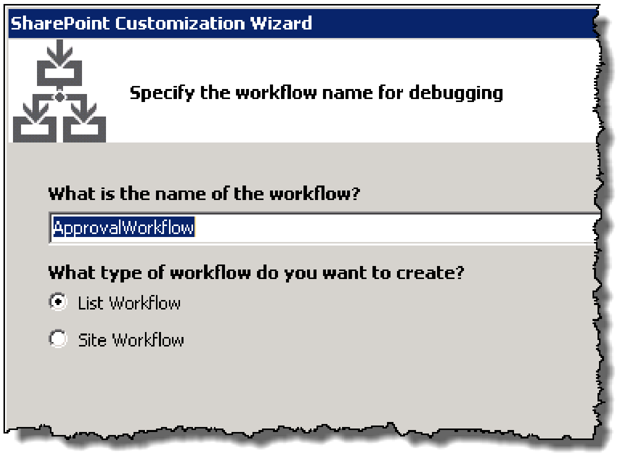
List	Visual C# Items
Remote Event Receiver	Visual C# Items
Content Type	Visual C# Items
Workflow	Visual C# Items
Workflow Custom Activity	Visual C# Items
Empty Element	Visual C# Items
Site Column	Visual C# Items
Module	Visual C# Items
Menu Item Custom Action	Visual C# Items

Type: Visual C# Items

An item which allows you to create and deploy Workflows to SharePoint.

Name: ApprovalWorkflow

4. In the **SharePoint Customization Wizard**, create a new **List Workflow** with the name, **ApprovalWorkflow**.
5. Click **Next**.



SharePoint Customization Wizard

Specify the workflow name for debugging

What is the name of the workflow?

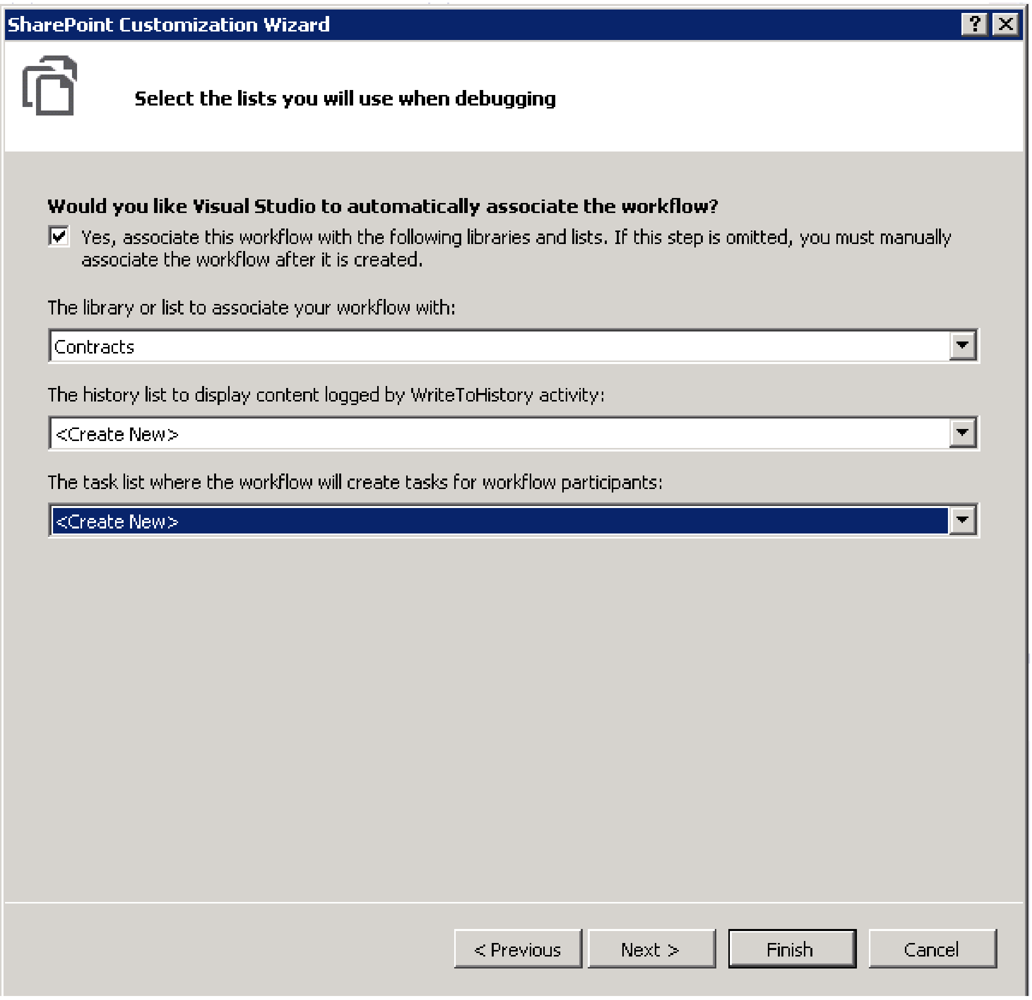
ApprovalWorkflow

What type of workflow do you want to create?

● List Workflow

○ Site Workflow

6. Associate the workflow with the **Contracts** list.
7. Choose `<Create New>` for the history list and task list.



SharePoint Customization Wizard

Select the lists you will use when debugging

Would you like Visual Studio to automatically associate the workflow?

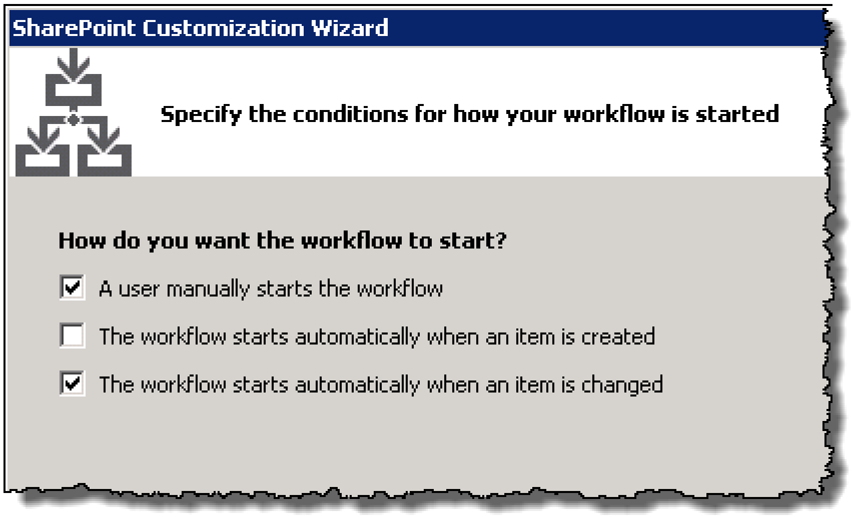
☑ Yes, associate this workflow with the following libraries and lists. If this step is omitted, you must manually associate the workflow after it is created.

The library or list to associate your workflow with: Contracts

The history list to display content logged by WriteToHistory activity: <Create New>

The task list where the workflow will create tasks for workflow participants: <Create New>

8. Click **Next**.
9. Choose to have the workflow start manually and automatically when an item is changed.



SharePoint Customization Wizard

Specify the conditions for how your workflow is started

How do you want the workflow to start?

☑ A user manually starts the workflow

☐ The workflow starts automatically when an item is created

☑ The workflow starts automatically when an item is changed

10. Click **Finish** to create the workflow.

Task 2 – Create the Workflow Variables

In this task, you will create four workflow variables.

The overall workflow needs 4 variables to track various items and statuses while executing. They are:

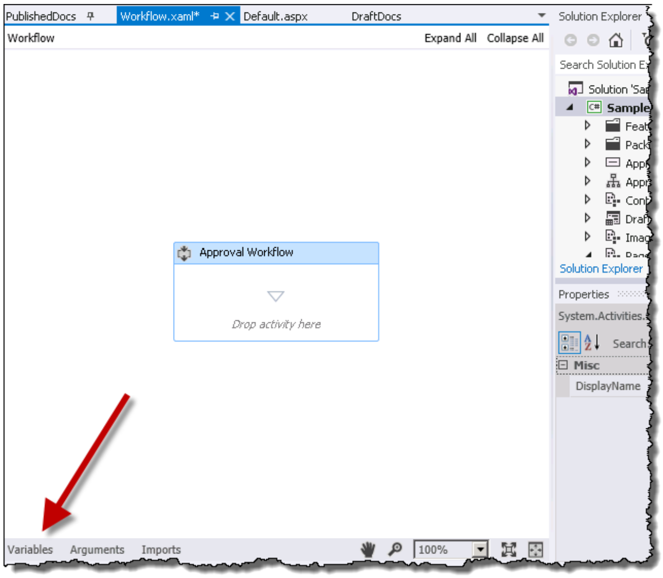
currentItemGuid The Guid of the item that triggered the workflow

currentListId The ID of the list where the workflow is running

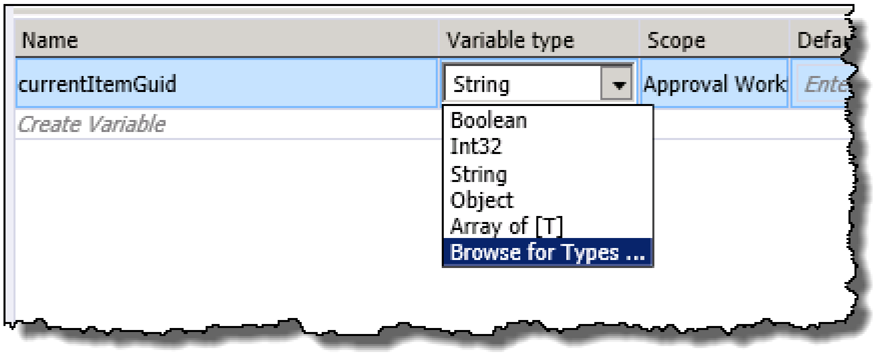
customerId The customer Id retrieved from the list item

customerName The customerName field is used to store the customer name temporarily before updating the Customer Name field in the list.

1. Select the Sequence activity on the workflow surface by clicking **Variables**.
2. While you are here, change the name of the activity from Sequence to **Approval Workflow** by clicking in the Sequence title.
3. While the cursor is still in the activity, click the **Variables** tab in the bottom left of the design surface.



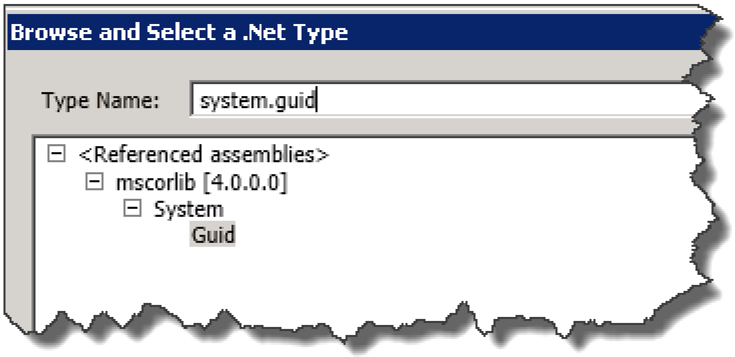
4. Click the **Create Variable** string in the variables window.
5. Change the name of the new variable from variable1 to **currentItemGuid**.
6. Change the variable type to **Guid**:
   a. Use the Variable type drop to select Browse **for Types**.



Name	Variable type	Scope	Defa
currentItemGuid	String	Approval Work	Ente
Create Variable			

b. In the Type Name field, type **System.guid**.

c. Click **OK**.



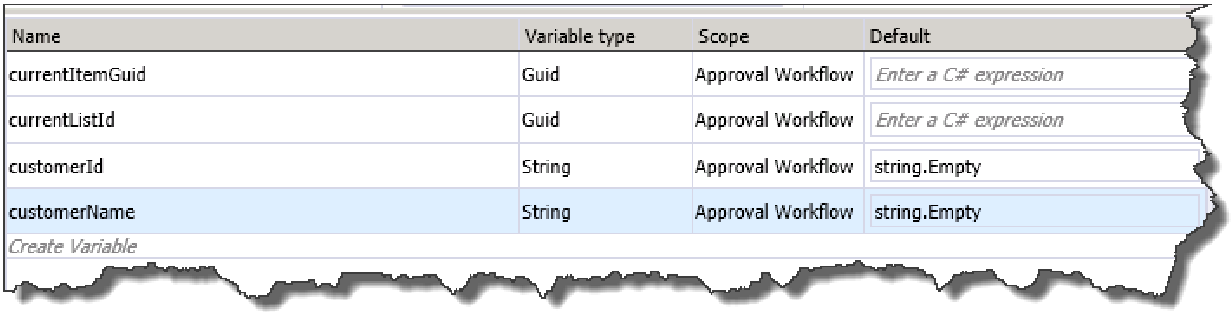
Browse and Select a .Net Type

Type Name: system.guid

The Guid variable type will now appear directly in the drop down.

7. Repeat the process to create the following variables

currentListId	Type = System.Guid	
customerId	Type = String	Default: string.Empty
customerName	Type=String	Default: string.Empty



Name	Variable type	Scope	Default
currentItemGuid	Guid	Approval Workflow	Enter a C# expression
currentListId	Guid	Approval Workflow	Enter a C# expression
customerId	String	Approval Workflow	string.Empty
customerName	String	Approval Workflow	string.Empty
Create Variable			

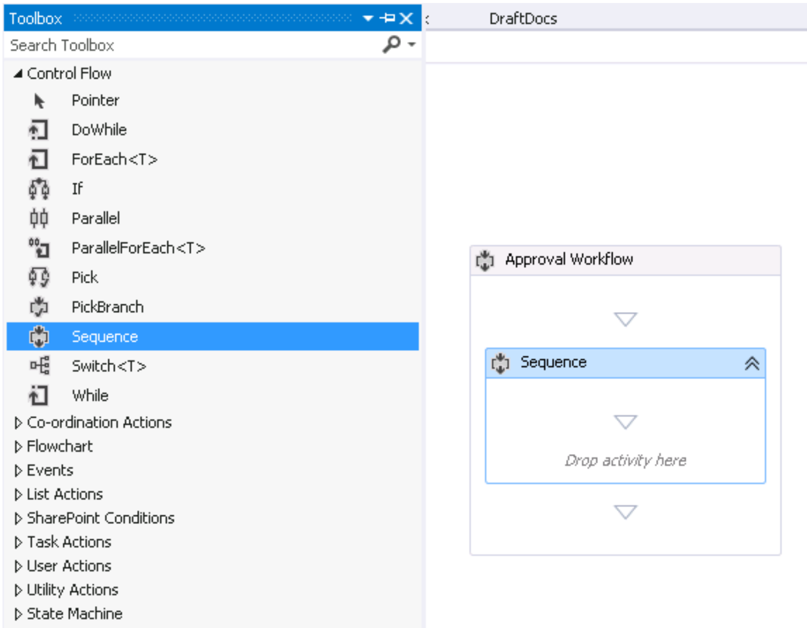
8. Save the solution.

9. Close the **Variables** pane by clicking the **Variables** tab in the bottom left of the designer.

Task 3 – Add the Activities to retrieve and save the current item GUID and the current list ID

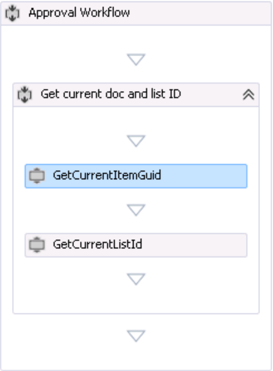
In this task, you will retrieve the item GUID and the List ID from SharePoint and store them in the `currentItemGuid` and `currentListId` variables.

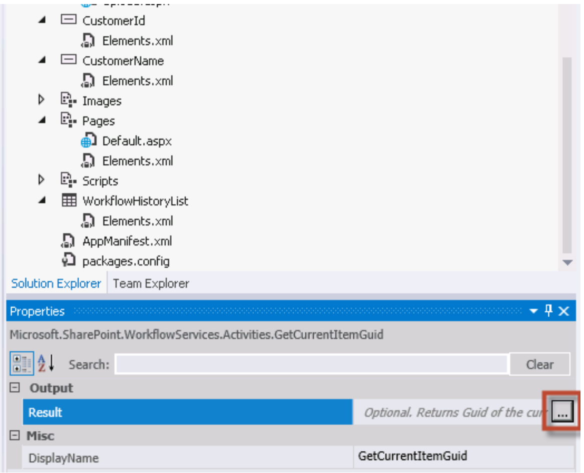
1. From the **View** menu, select **Toolbox**.
2. In the **Toolbox**, expand the **Control Flow** activities and drag a new **Sequence** activity inside the Approval Workflow activity.



3. Rename the sequence to **Get current doc and list ID**.
4. In the Toolbox, expand the **SP – Current Context** group and drag the **GetCurrentItemGuid** and **GetCurrentListId** activities to the sequence that you added in step 2.

   These activities will retrieve the corresponding Guid/ID and store it in variables that you specify.
5. Select the **GetCurrentItemGuid** activity. If the Properties window is not open, load it by clicking **View**, **Properties Window**.
6. In the Properties window, click the **ellipsis** by the Result property.





Microsoft.SharePoint.WorkflowServices.Activities.GetCurrentItemGuid

Result: Optional. Returns Guid of the cu

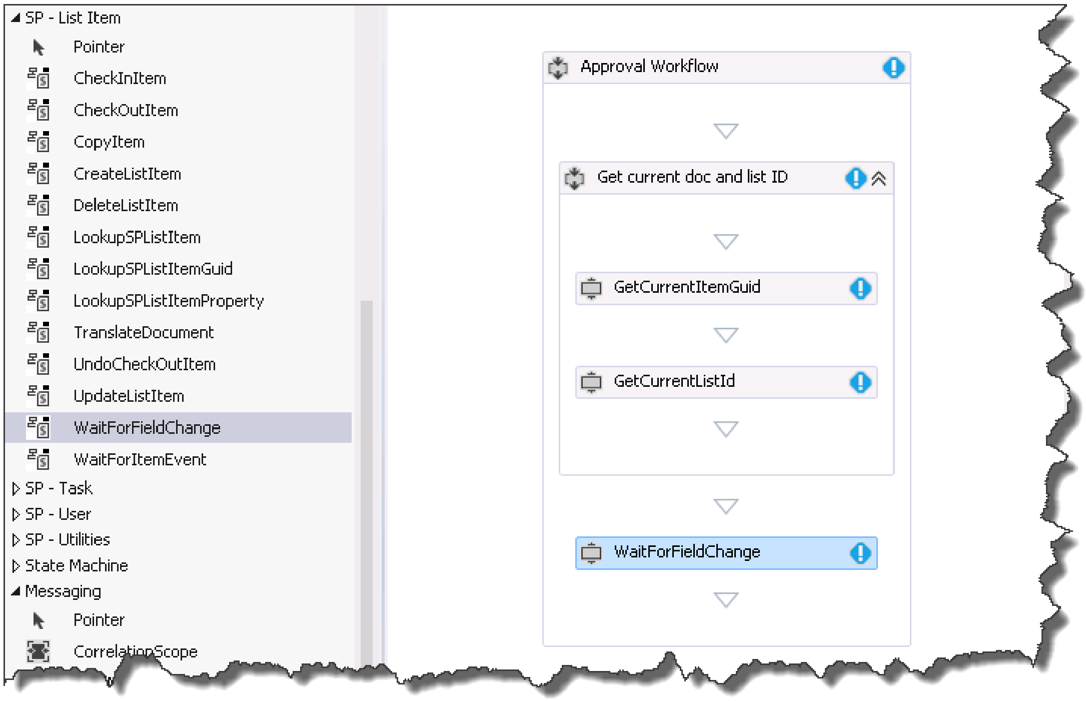
DisplayName: GetCurrentItemGuid

7. Start typing the **currentItemGuid** variable name.
8. Select **currentItemGuid** and click **OK**.
9. Repeat steps 5 through 8, adding a **GetCurrentListId** activity and storing the result in the **currentListId** variable.
10. Save the solution.

Task 4 – Wait until the document is “Approved For Publishing”

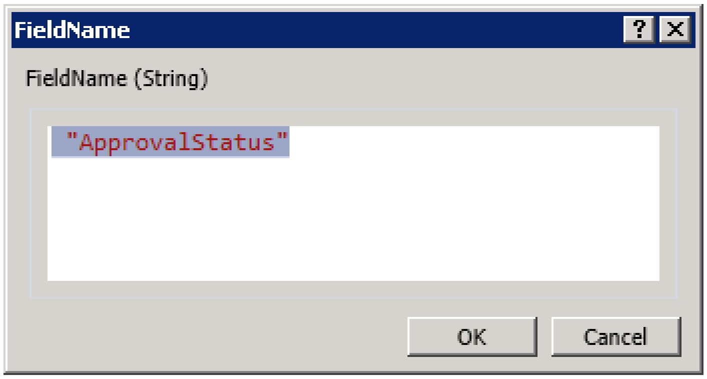
In this task, you will add a wait for field change activity to the workflow.

1. In the Toolbox, expand the **SP – List Item** activities.
2. Drag the **WaitForFieldChange** activity underneath the **Get current Doc and List ID** activity.



Note: The exclamation point next to the new activity. This is a signal to you that there is a problem with the activity. As you have yet to set the activity's properties, this is to be expected at this stage.

3. Rename the activity to **Wait Until Doc is Approved For Publishing**.
4. In the Properties Window
   a. Click the Ellipsis next to the field name.
   b. Add **"ApprovalStatus"** (with the quotes) in the text field of the FieldName dialog. Make sure there is not a space between the two words.

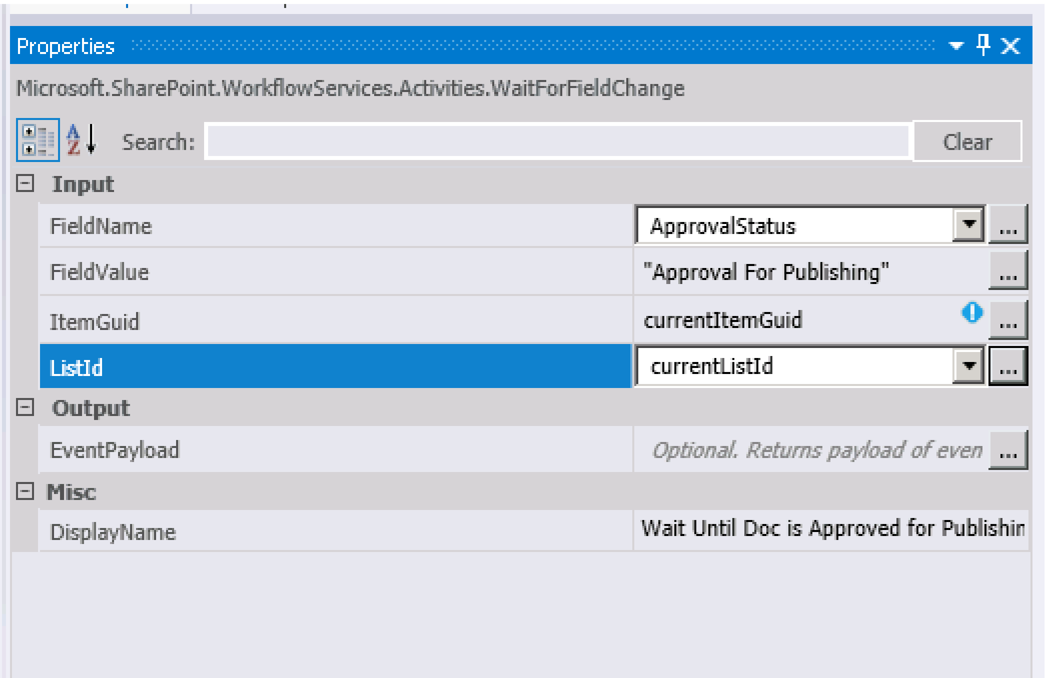


```
"ApprovalStatus"
```

   c. Click **OK**

Note: The Field Name will now appear in the drop down.

   d. Set the **FieldValue** to **"Approved For Publishing"** (with quotes).
   e. Set the **ItemGuid** to **currentItemGuid**.
   f. Set the **ListId** to **currentListId** by clicking the Ellipses and entering the variable name.



Microsoft.SharePoint.WorkflowServices.Activities.WaitForFieldChange

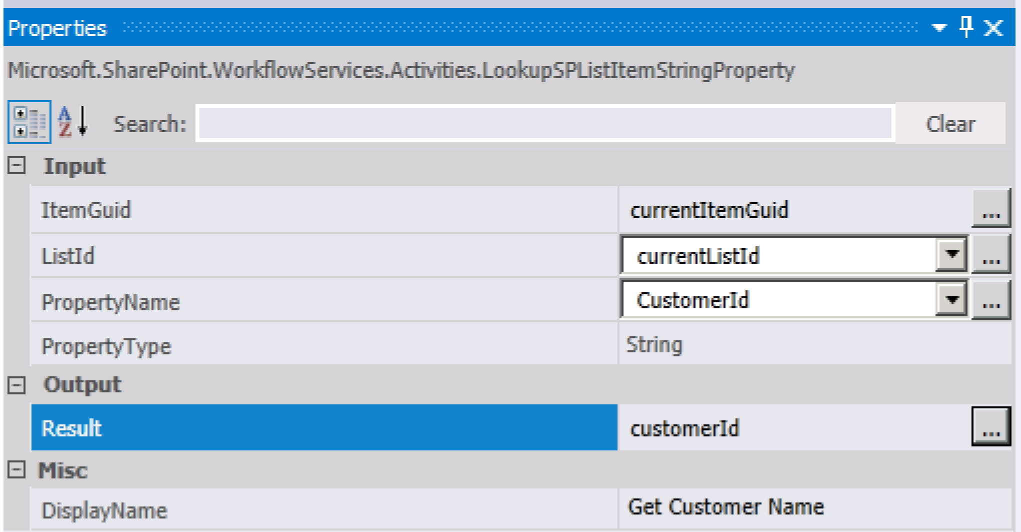
Input	
FieldName	ApprovalStatus
FieldValue	"Approval For Publishing"
ItemGuid	currentItemGuid
ListId	currentListId
Output	
EventPayload	*Optional. Returns payload of even*
Misc	
DisplayName	Wait Until Doc is Approved for Publishin

5. Save the Solution.

Task 6 – Load the Customer Id into a Workflow Variable

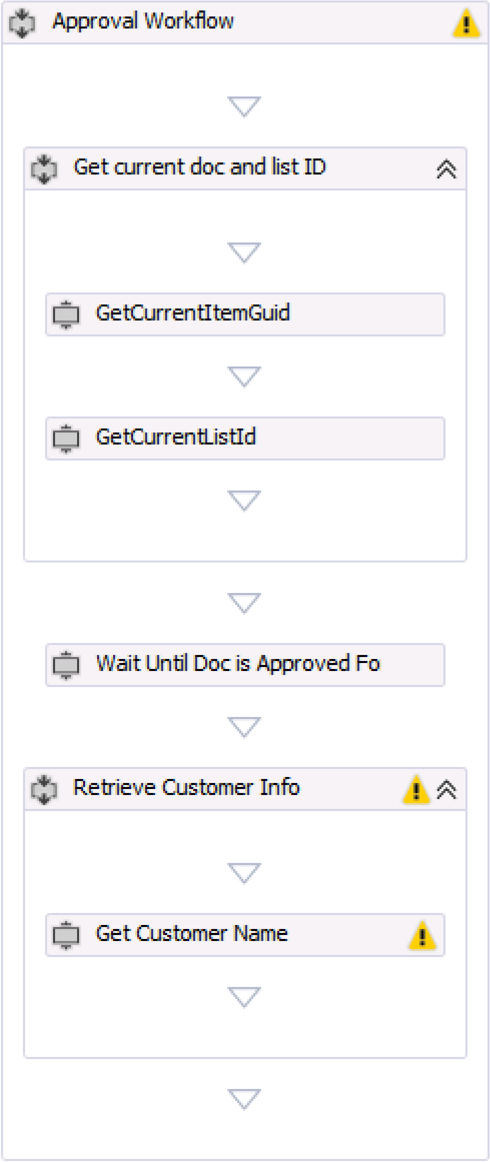
In this task, you will load the customer id into the workflow variable. It is now time to call the Northwind OData service and retrieve the customer based on the customer name.

1. In the Toolbox, expand **Control Flows** and drag a new **Sequence** activity to the bottom of the Approval Workflow Sequence.
2. Rename the activity to **Retrieve Customer Info.**
3. In the Toolbox, Expand **SP-List Item** and drag the **LookupSPListItemProperty** activity into the new sequence you just created.
4. Rename the activity to **Get Customer Name**.
5. In the properties window, set the following properties
    - **ItemGuid**: currentItemGuid
    - **ListId**: currentListId
    - **PropertyName**: “CustomerId” (with quotes).
    - **PropertyType**: String
    - **Result**: customerId



Microsoft.SharePoint.WorkflowServices.Activities.LookupSPListItemStringProperty

Input	
ItemGuid	currentItemGuid
ListId	currentListId
PropertyName	CustomerId
PropertyType	String
Output	
Result	customerId
Misc	
DisplayName	Get Customer Name



6. To add some debugging logic, from the **Toolbox**, expand **Primitives** and add a **WriteLine** activity directly under **Get Customer Name** (still within the Retrieve Customer Info sequence).
7. Change the **Text** property to read:

```
"Customer Id = " + customerId
```

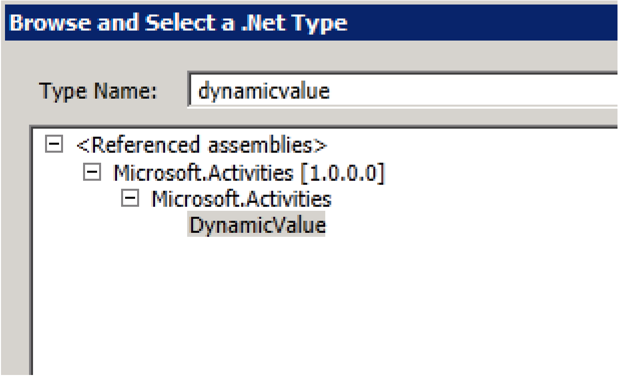
8. Save the solution.

Task 7 – Add a Variable to Store the Results of the Web Service Call

In this task, you will create a variable that is scoped to the Retrieve Customer Info sequence. Before calling the web service, you will need to create one more variable to store the result of the call. You may have noticed when you were creating the other workflow variables that they had a scope set to the

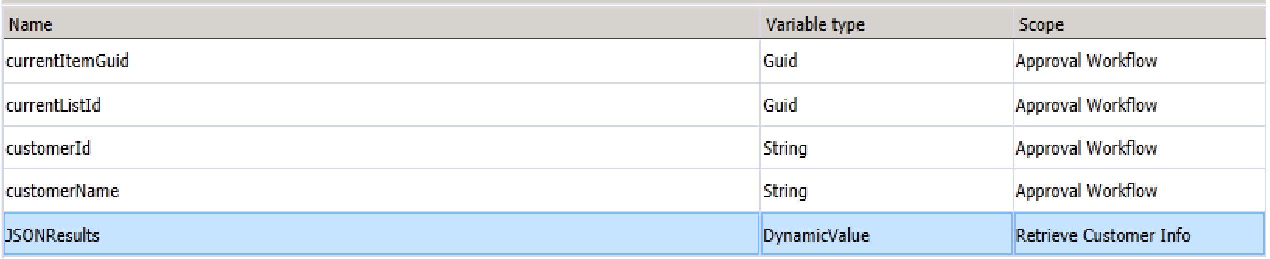
Approval Workflow sequence (as this was the sequence that was selected when you created them. This time, you will create the variable scoped to the Retrieve Customer Info sequence.

1. Select the **Retrieve Customer Info** sequence.
2. Click the **Variables** tab in the bottom left hand corner of the designer.
3. Add a new variable named **JSONResults**.
4. In the **Variable type** drop down, select **Browse for Types**.
5. Enter **dynamicvalue** for the **Type Name** and then select **DynamicValue**.


Browse and Select a .Net Type

Type Name: dynamicvalue

Note: The new variable is scoped only to the Retrieve Customer Info sequence. Only activities within that sequence will be able to access it.

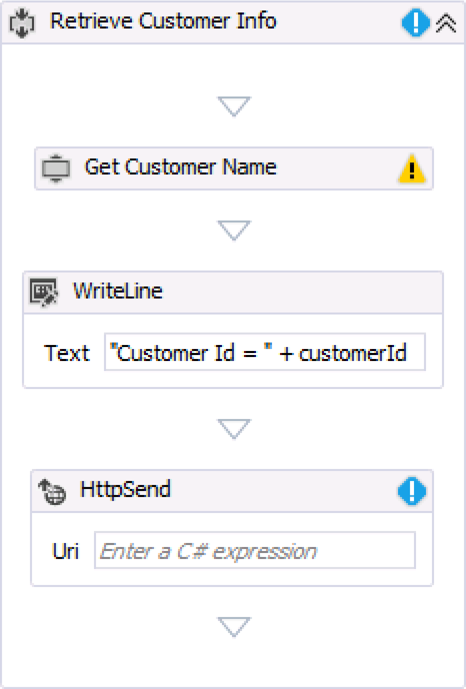


Name	Variable type	Scope
currentItemGuid	Guid	Approval Workflow
currentListId	Guid	Approval Workflow
customerId	String	Approval Workflow
customerName	String	Approval Workflow
JSONResults	DynamicValue	Retrieve Customer Info

Task 8 – Call the Northwind OData Web Service

In this task, you will call the Northwind OData Service.

1. In the Toolbox, expand **Messaging** and drag an **HttpSend** activity to the bottom of the **Retrieve Customer Info** activity.

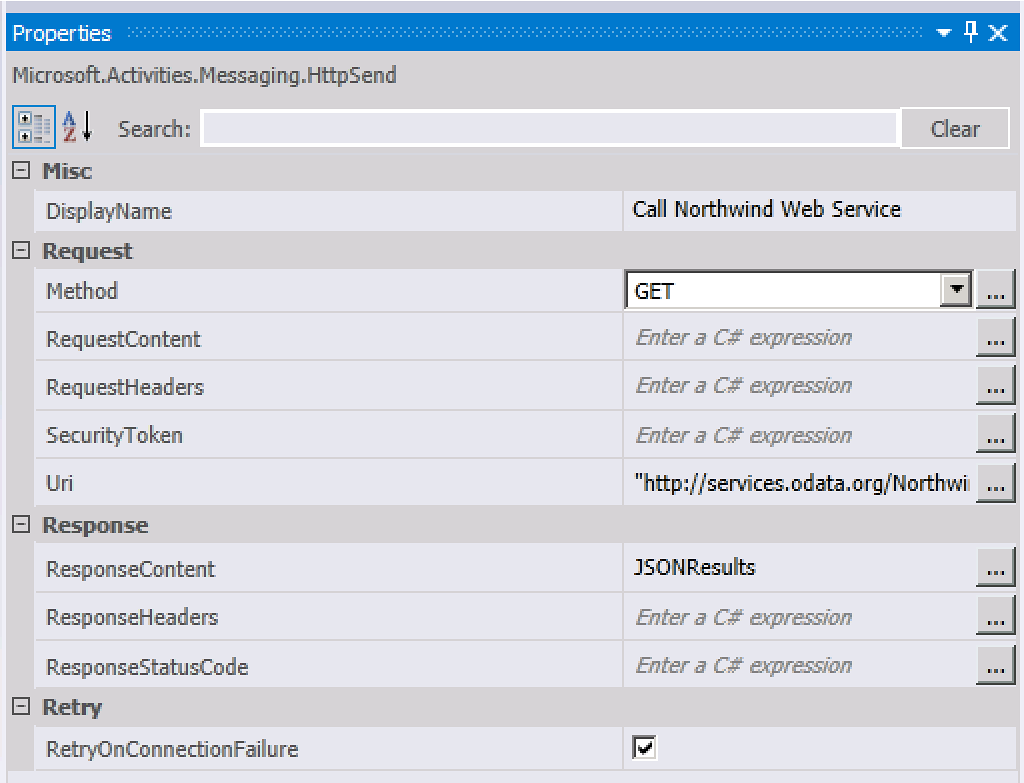


2. Rename the activity to **Call Northwind Web Service.**
3. In the Properties Window
   - For **Method**, choose **Get** from the dropdown.
4. Set **Uri** to the following by navigating to **C:\Demo\HOL010\Snippets** and double-click **Snippet 03.**
   - Copy the text and paste the content including quotes.

```
"http://services.odata.org/Northwind/Northwind.svc/Customers('" +
customerId + "')?$format=json"
```

Note: This URL incorporates the value of the customerId workflow variable

5. Set **ResponseContent** to the **JSONResults** variable you just created.



Microsoft.Activities.Messaging.HttpSend

Misc	
DisplayName	Call Northwind Web Service
Request	
Method	GET
RequestContent	Enter a C# expression
RequestHeaders	Enter a C# expression
SecurityToken	Enter a C# expression
Uri	"http://services.odata.org/Northwi
Response	
ResponseContent	JSONResults
ResponseHeaders	Enter a C# expression
ResponseStatusCode	Enter a C# expression
Retry	
RetryOnConnectionFailure	☑

Task 9 – Retrieve the CustomerName from the JSON Result Set and Update the List

In this task, you will setup customer name to return results in the JSON format.

The OData query will return results in the JSON format. Below are the actual results from the query for the customer Id **EASTC**.

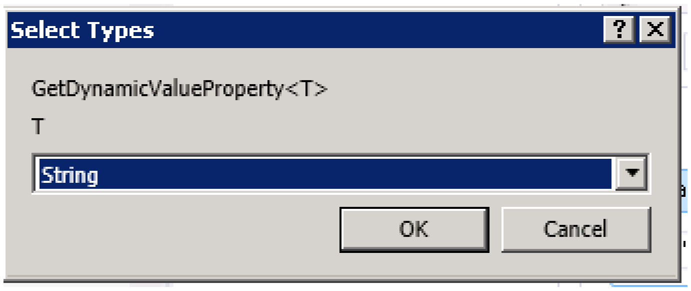
```
{
"d" : {
   "__metadata": {
         "uri":
"http://services.odata.org/Northwind/Northwind.svc/Customers('EASTC')", "type":
"NorthwindModel.Customer"
   },
         "CustomerID": "EASTC",
         "CompanyName": "Eastern Connection",
         "ContactName": "Ann Devon",
         "ContactTitle": "Sales Agent",
         "Address": "35 King George",
         "City": "London",
         "Region": null,
         "PostalCode": "WX3 6FW",
         "Country": "UK",
         "Phone": "(171) 555-0297",
         "Fax": "(171) 555-3373",
         "Orders": {
         "__deferred": {"uri":
"http://services.odata.org/Northwind/Northwind.svc/Customers('EASTC')/Orders"}},
   "CustomerDemographics": {
         "__deferred": {"uri":
"http://services.odata.org/Northwind/Northwind.svc/Customers('EASTC')/CustomerDem
ographics"}}
   }
}
```

Note: The very hierarchical nature of the results set. JSON is an open and text-based data exchange format that provides a standardized data exchange format well-suited for Ajax-style web applications. To learn more about JSON, see this MSDN article.

After spending some time looking at the above results, you will probably deduce that there is an object, d, with a bunch of children, such as CustomerID, CompanyName and ContactName. When querying JSON data, you would refer to the ContactName as d\ContactName as it is a child of the object, d.

1. In the **Toolbox**, expand **DynamicValue** and drag a **GetDynamicValueProperty<t>** to the bottom of the **Retrieve Customer Info** sequence.

2. When prompted to choose which type of property you are looking for, choose **String**.



Select Types

GetDynamicValueProperty<T>

T

String

6. In the properties pane:

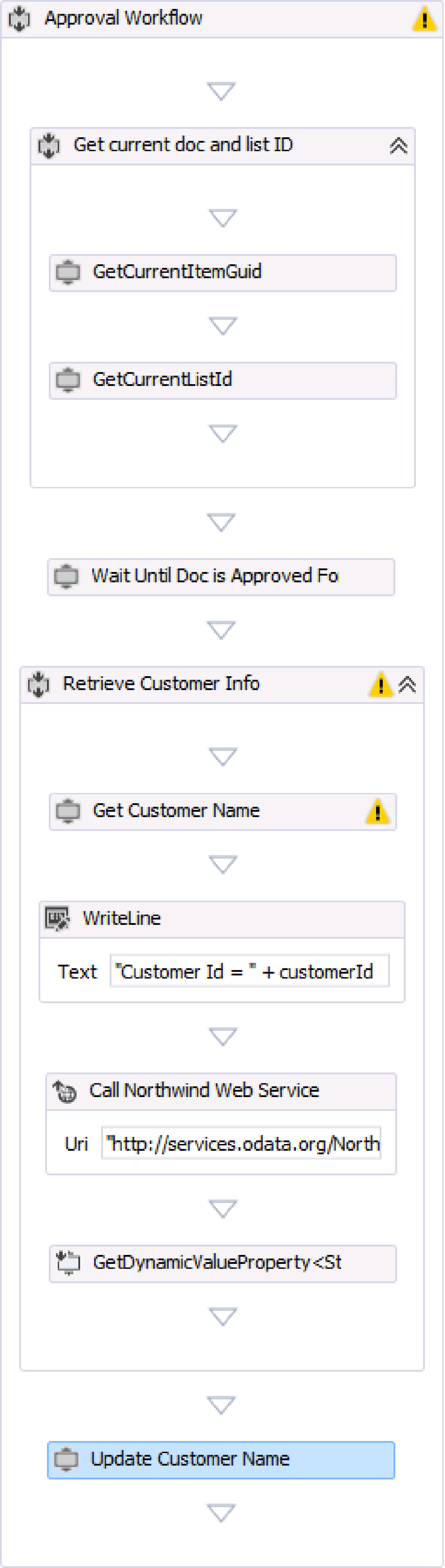
- Set the **PropertyName** to be retrieved from the results to **“d/ContactName”.**
- Set the **Result** of the activity (ie, the destination variable) to **customerName**

- Set the **Source** variable to **JSONResults**

Finally, you will add an activity to update the list item with the CustomerName

7. In the Toolbox, expand **SP – Current Context** and drag a **SetField** activity to the bottom of the Approval Workflow sequence.
8. Rename it to **Update Customer Name**.
9. Set its properties to the following:
    - For **FieldName**, Select **Customer Name** from the drop down.
    - Set **FieldValue** to the **customerName** variable.

Your completed workflow should now look like the screen shot below.



Text "Customer Id = " + customerId

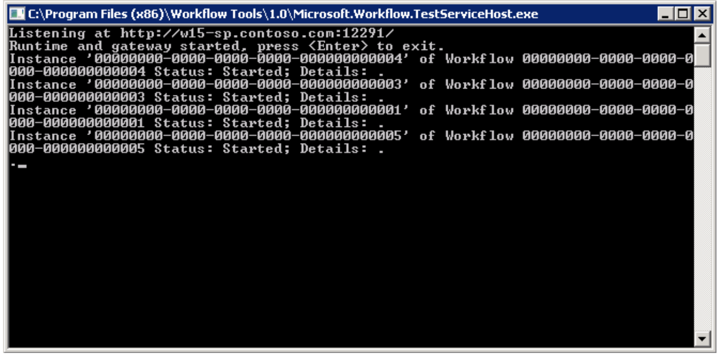
Uri "http://services.odata.org/North

Task 10 – Debug your App

Congratulations! Your workflow is now ready for test. It's now time to put it through its paces. In this task, you will debug your app.

1. Debug the solution by selecting **Debug**, and then **Start Debugging** from the Visual Studio menu. **Important Note:** do not use **F5** as it may cause problems in the HOL environment.
2. The Workflow Test Service Host will spin up which will allow you to debug the workflow. If you use the WriteLine workflow activity (located in the Toolbox in the Primitives group), this console is where you will see the output.

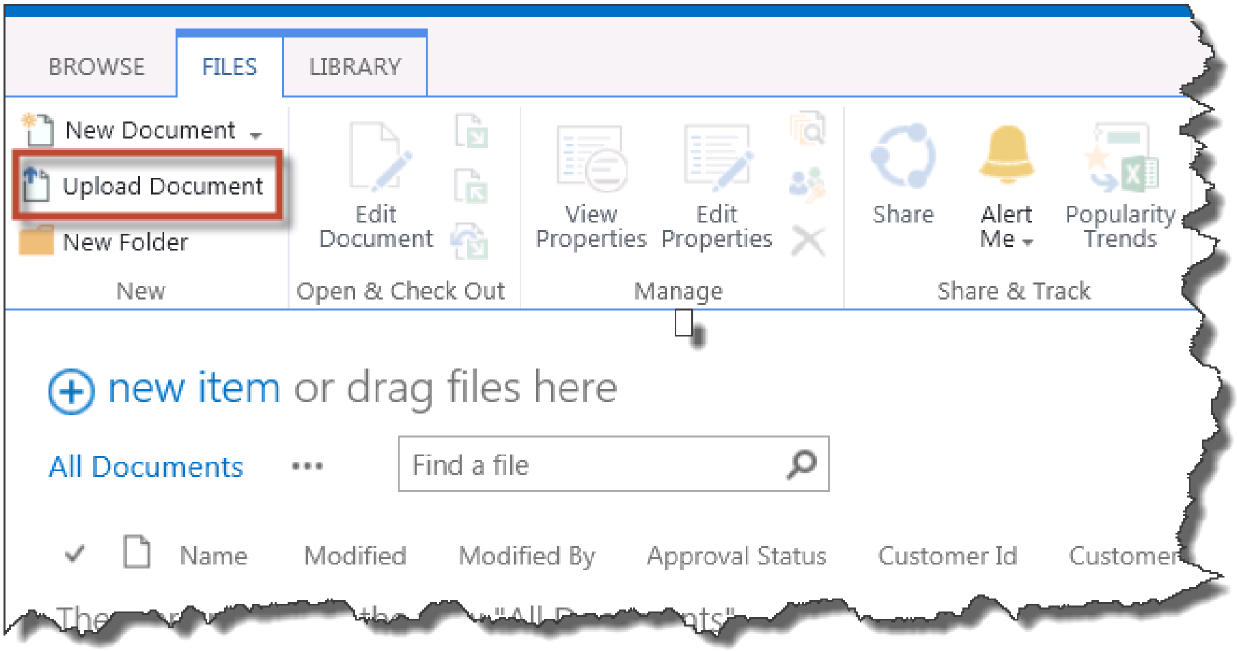
Note: Any WriteLine activities that you added will appear in this console.



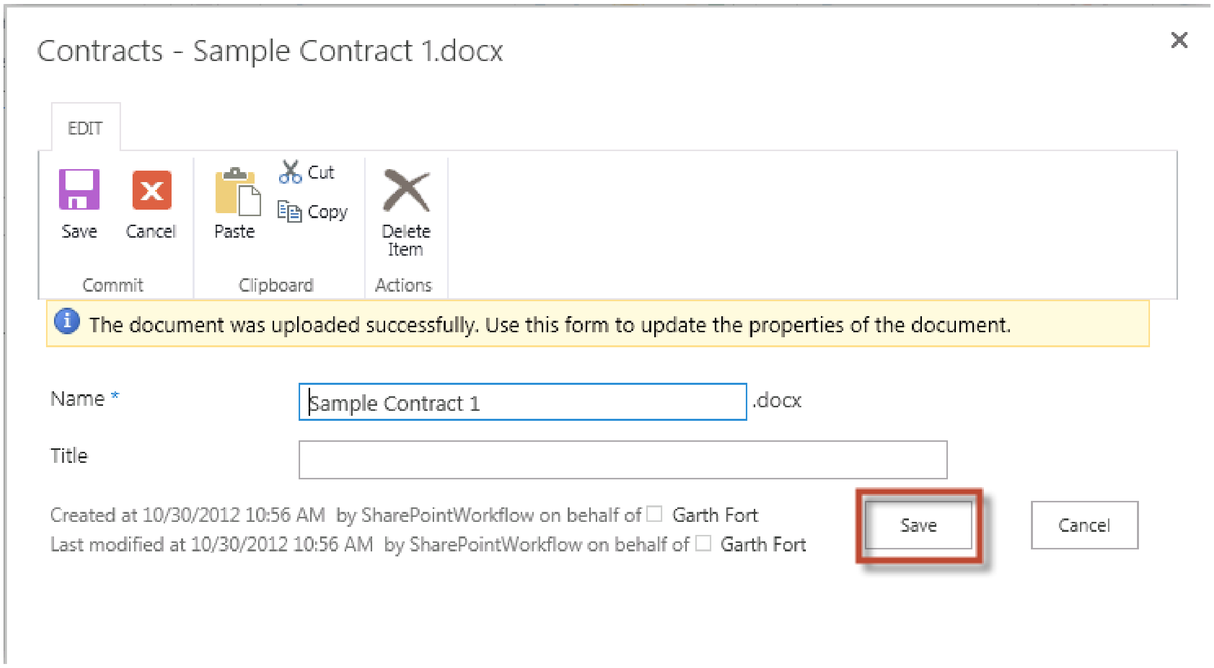
C:\Program Files (x86)\Workflow Tools\1.0\Microsoft.Workflow.TestServiceHost.exe

```
Listening at http://w15-sp.contoso.com:12291/
Runtime and gateway started, press <Enter> to exit.
Instance '00000000-0000-0000-0000-000000000004' of Workflow 00000000-0000-0000-0
000-000000000004 Status: Started; Details: .
Instance '00000000-0000-0000-0000-000000000003' of Workflow 00000000-0000-0000-0
000-000000000003 Status: Started; Details: .
Instance '00000000-0000-0000-0000-000000000001' of Workflow 00000000-0000-0000-0
000-000000000001 Status: Started; Details: .
Instance '00000000-0000-0000-0000-000000000005' of Workflow 00000000-0000-0000-0
000-000000000005 Status: Started; Details: .
._
```

3. Eventually, Visual Studio will load the Site Contents window in Internet Explorer. Within your app, click the **Contracts** link to navigate to the Contracts library.
4. Upload a new document to the library by clicking **Files**, and then click **Upload Document**.



5. Browse to **c:\demo\HOL010**, select **Sample Contract1.docx** and click **Open**, and then click **OK** and then **Save** to upload a sample file.



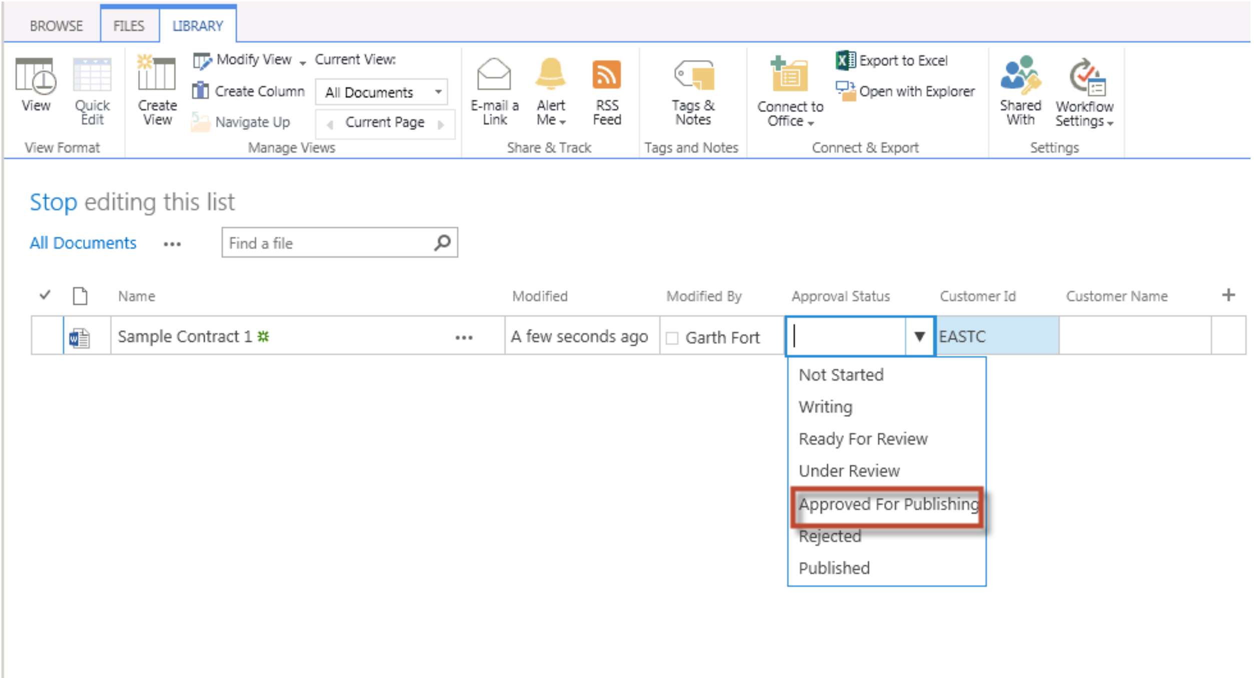
Contracts - Sample Contract 1.docx

The document was uploaded successfully. Use this form to update the properties of the document.

Name *	Sample Contract 1 .docx
Title	

Created at 10/30/2012 10:56 AM by SharePointWorkflow on behalf of □ Garth Fort
Last modified at 10/30/2012 10:56 AM by SharePointWorkflow on behalf of □ Garth Fort

6. Once the file is uploaded, navigate to the quick edit view of the list by clicking **Library**, and then click **Quick Edit** on the Contracts ribbon.
7. In the **CustomerID** field type **EASTC.**
8. In the **Approval Status** field, select **Approved For Publishing**.



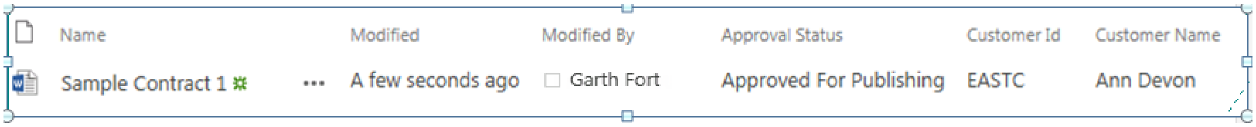
Stop editing this list

All Documents ••• Find a file

✓		Name	Modified	Modified By	Approval Status	Customer Id	Customer Name
		Sample Contract 1	A few seconds ago	□ Garth Fort		EASTC	

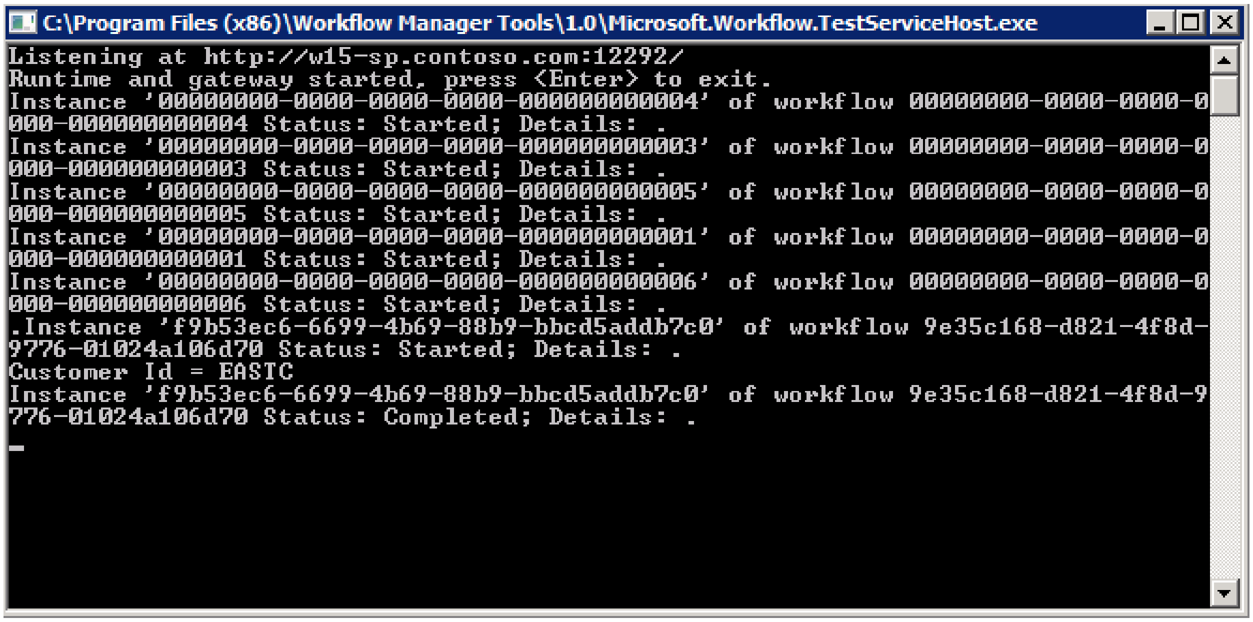
9. Back in the Ribbon, select **View** instead of Quick Edit in the **View Format** group.

10. If the Customer Name has not been updated, refresh the page. The REST query can take some seconds to commit, but almost always less than a minute.



	Name		Modified	Modified By	Approval Status	Customer Id	Customer Name
	Sample Contract 1	•••	A few seconds ago	Garth Fort	Approved For Publishing	EASTC	Ann Devon

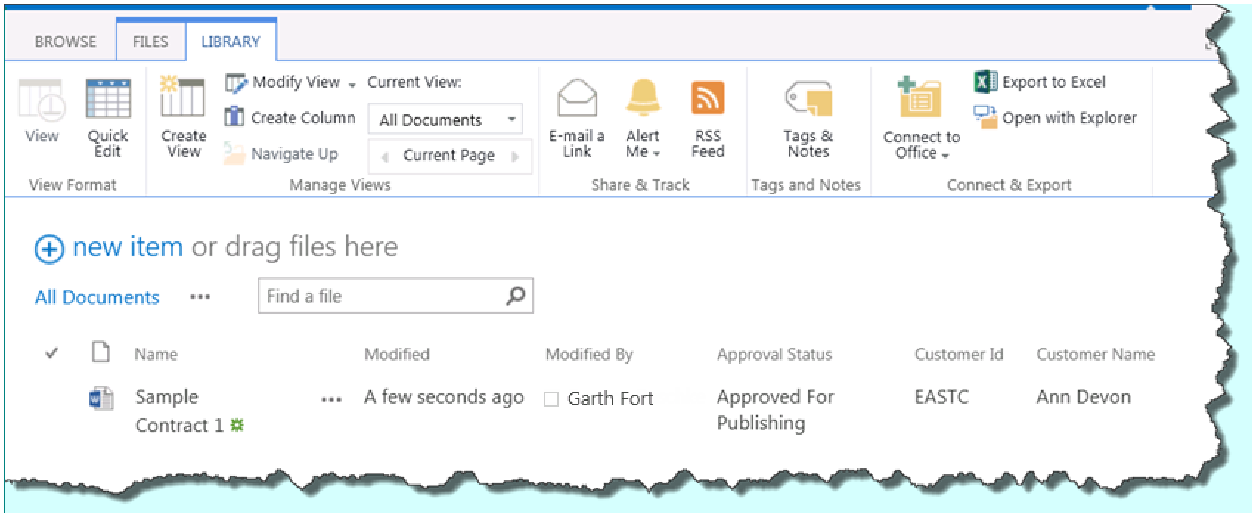
11. Switch to the console window to see the results of your write line statements and the notification that the workflow has completed.



C:\Program Files (x86)\Workflow Manager Tools\1.0\Microsoft.Workflow.TestServiceHost.exe

```
Listening at http://w15-sp.contoso.com:12292/
Runtime and gateway started, press <Enter> to exit.
Instance '00000000-0000-0000-0000-000000000004' of workflow 00000000-0000-0000-0
000-000000000004 Status: Started; Details: .
Instance '00000000-0000-0000-0000-000000000003' of workflow 00000000-0000-0000-0
000-000000000003 Status: Started; Details: .
Instance '00000000-0000-0000-0000-000000000005' of workflow 00000000-0000-0000-0
000-000000000005 Status: Started; Details: .
Instance '00000000-0000-0000-0000-000000000001' of workflow 00000000-0000-0000-0
000-000000000001 Status: Started; Details: .
Instance '00000000-0000-0000-0000-000000000006' of workflow 00000000-0000-0000-0
000-000000000006 Status: Started; Details: .
.Instance 'f9b53ec6-6699-4b69-88b9-bbcd5addb7c0' of workflow 9e35c168-d821-4f8d-
9776-01024a106d70 Status: Started; Details: .
Customer Id = EASTC
Instance 'f9b53ec6-6699-4b69-88b9-bbcd5addb7c0' of workflow 9e35c168-d821-4f8d-9
776-01024a106d70 Status: Completed; Details: .
_
```

Your SharePoint item should now look like this:

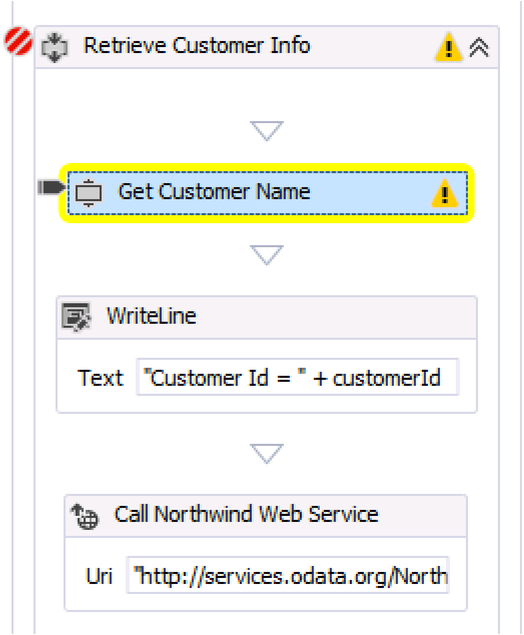


✓	Name		Modified	Modified By	Approval Status	Customer Id	Customer Name
	Sample Contract 1	•••	A few seconds ago	Garth Fort	Approved For Publishing	EASTC	Ann Devon

Task 11 – Add Break Statements to the Workflow

In this task, you will add break statements to the workflow and step through them in debugging mode.

1. Switch back to **Visual Studio** but do not stop debugging.
2. Right-click the **Retrieve Customer Info** activity and select **Breakpoint** and then click **Insert Breakpoint**.
3. Now, when you add a new document and repeat the steps from the above debugging task, the workflow will hit this breakpoint. Step into the sequences by pressing **F11**. At any point in time, you will be able to check on all of the workflow variables in scope by viewing the Locals window in the lower left-hand corner of your screen.



Text "Customer Id = " + customerId

Uri "http://services.odata.org/North

Note: The sequence that is about to be executed (in this case, **Get Customer Name**) is highlighted in yellow.

In this exercise, you created the workflow. This workflow ran against the Contracts list and updated the Customer Name field once an item's Approval Status was set to Ready For Publish.

Summary

In this hands-on lab, you were introduced to the new SharePoint Server 2013 Preview workflow development environment in Visual Studio 2012. You built a workflow that walked through certain aspects of a document creation process. It watched for a status field to be updated to "Approved for Publishing." Once that state was hit, the workflow queried the public Northwind OData API to retrieve a customer name from a customer Id. Finally, it updated the list item with the customer name retrieved from the web service.