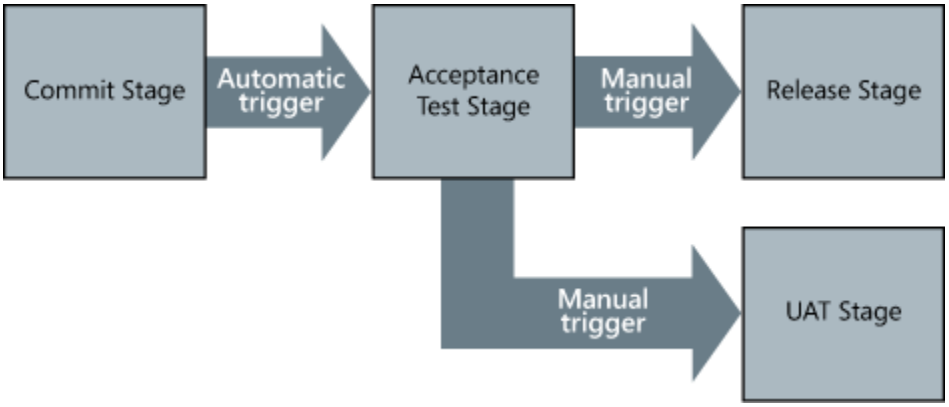
Hands-on Lab 2.3: Configuring the Pipeline

Table of Contents

Objectives

In parts one and two of the Orchestration HOL you orchestrated the pipeline. In part three you configure what will be an actual running instances of a pipeline that has a commit stage, an acceptance test stage, a release stage and a user acceptance test (UAT) stage. Here is the structure of the pipeline.



The commit stage automatically triggers the acceptance test stage. The release stage and the UAT stage are manually triggered.

Prerequisites

Here are the prerequisites for completing this lab.

- Complete Lab-2.2 Orchestrating the Remaining Stages.
- You will need Microsoft Visual Studio Team Foundation Server 2012 Power Tools.

> **Note:** The Power Tools are included in the Brian Keller VM.

Time

You should be able to complete this lab in about 40 minutes.

Exercise 1: Add the Customized Build Templates to the Project

In this exercise you add the workflow build templates that you created in the previous two labs to the project and check them in to TFS.

Task 1: Locate the Files

1. Navigate to Lab02_Orchestration\Start-Lab\TreyResearchBuildCustomization\TreyResearchBuildCustomization.
2. Locate the following two workflow files:
    - Start_CDPipelineCommitStageProcessTemplate.xaml
    - Start_CDPipelineGenericStageProcessTemplate.xaml

Task 2: Copy and Rename the Files

1. Select the two workflow files.
2. Navigate to the \BuildProcessTemplates folder that was created when you created the TreyResearch team project in Lab 1.
3. Paste the files into the folder.
4. Rename Start_CDPipelineCommitStageProcessTemplate.xaml to CDPipelineCommitStageProcessTemplate.xaml.
5. Rename Start_CDPipelineGenericStageProcessTemplate.xaml to CDPipelineGenericStageProcessTemplate.xaml.
6. Check in the two xaml files to TFS.

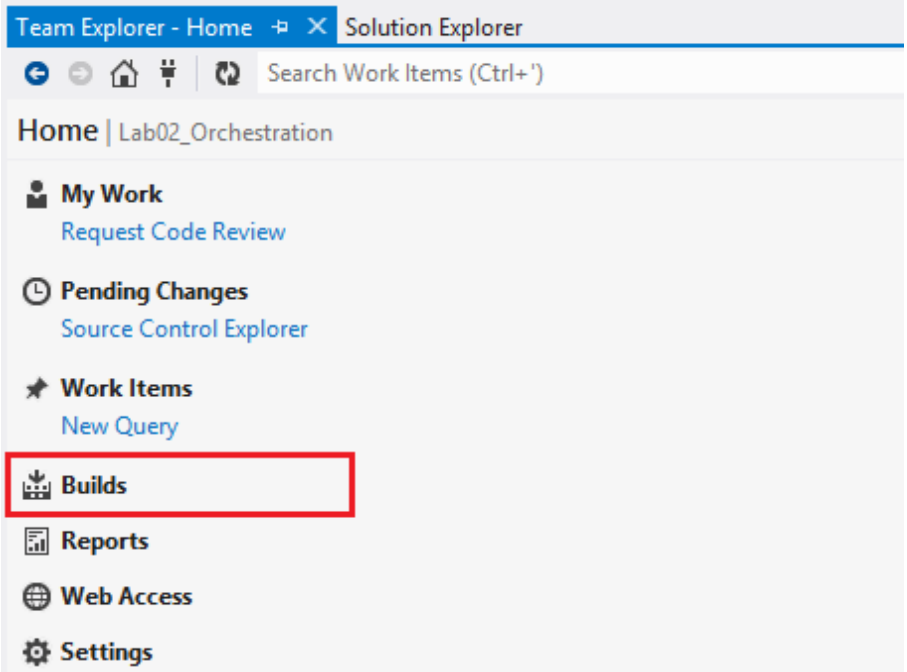
Exercise 2: Creating the Commit Stage for a Pipeline Instance

In this exercise you create the build definition for a particular commit stage that belongs to a particular pipeline instance.

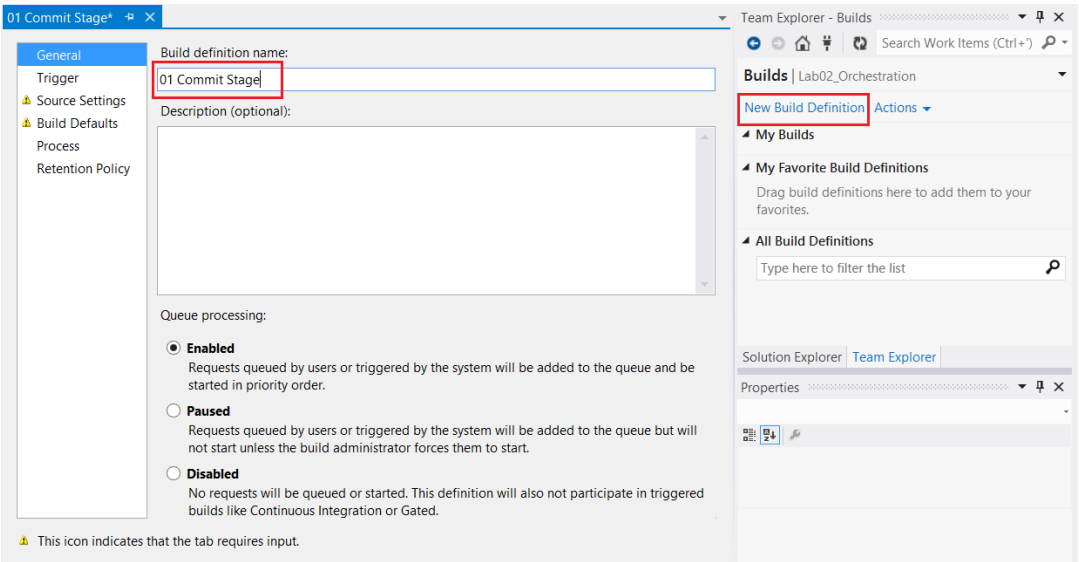
Task 1: Create the 01 Commit Stage Build Definition

In this task you create a build definition for a commit stage named 01 Commit Stage.

1. Open Visual Studio. In Team Explorer, go to the **Builds** tab.



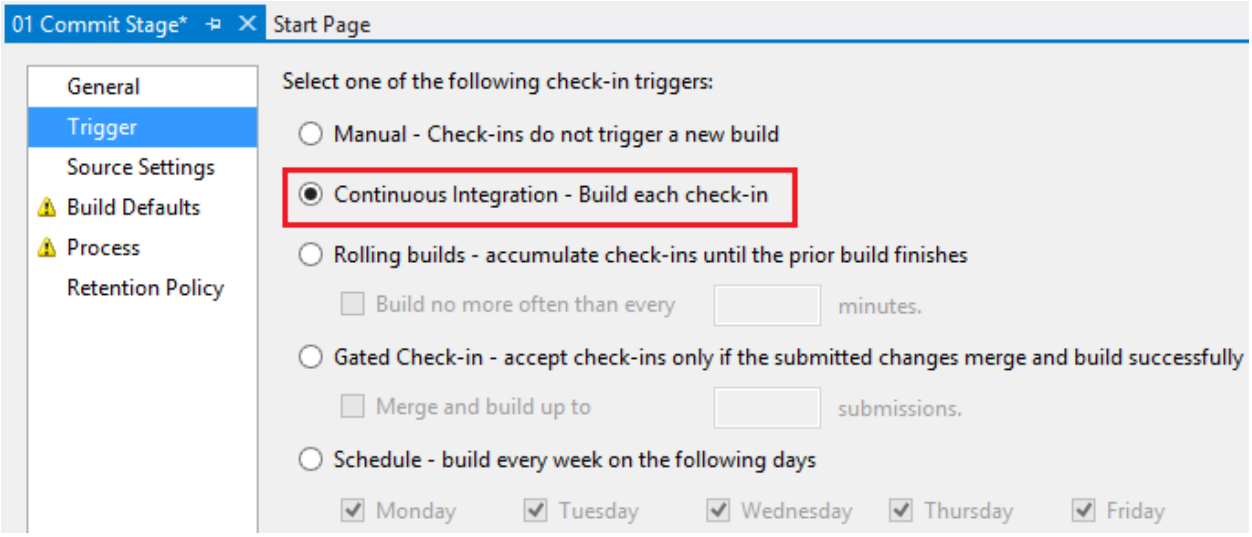
2. Click **New Build Definition**. A new build definition appears. Name it **01 Commit Stage**.



Task 2: Set the Check-in Trigger

In this task you set the type of check-in that triggers a build.

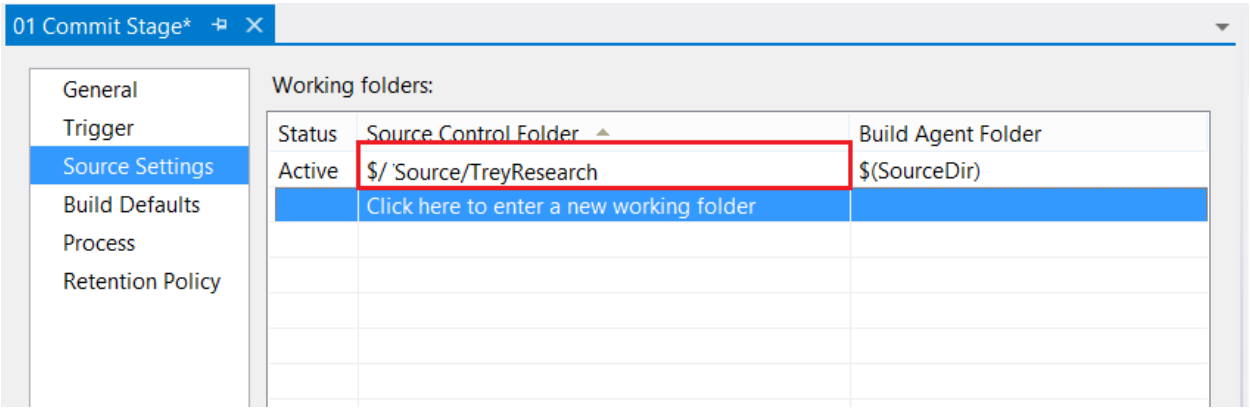
1. Select the **Trigger** tab in the build definition.
2. Select **Continuous Integration – Build each check-in**.



Task 3: Set the Working Folder

In this task you specify the folders to include in the build.

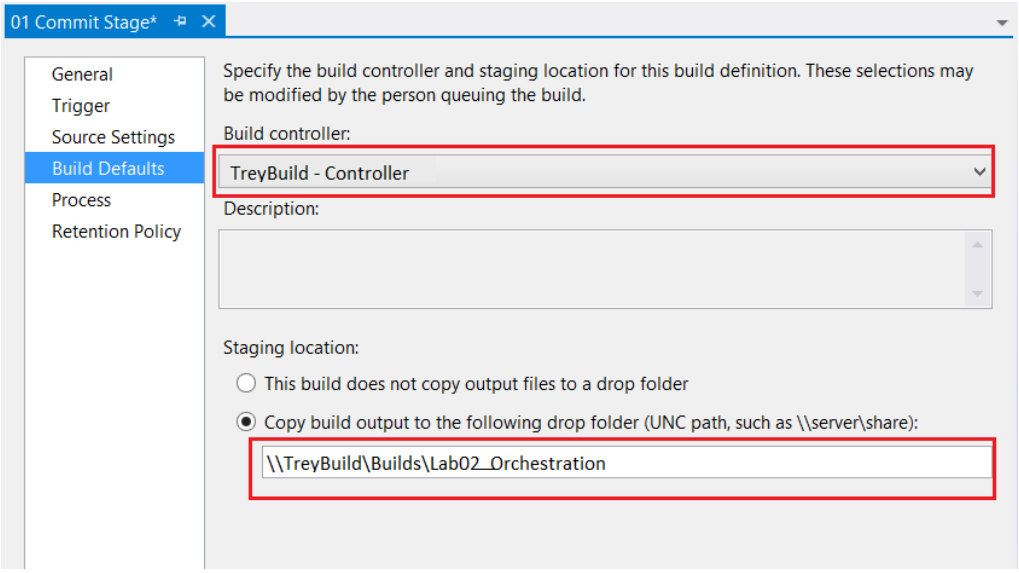
1. In the build definition, click the **Source Settings** tab.
2. In the **Active** row, set the source control folder to match the text highlighted in red in the following screenshot.



Task 4: Set the Build Controller and Drop Location

1. Click the **Build Defaults** tab.
2. In the **Build Controller** combo box, select the name of the build controller that you want to use for this build definition.
3. Under **Staging location**, select **Copy build output to the following drop folder (UNC path, such as \\server\share)**.

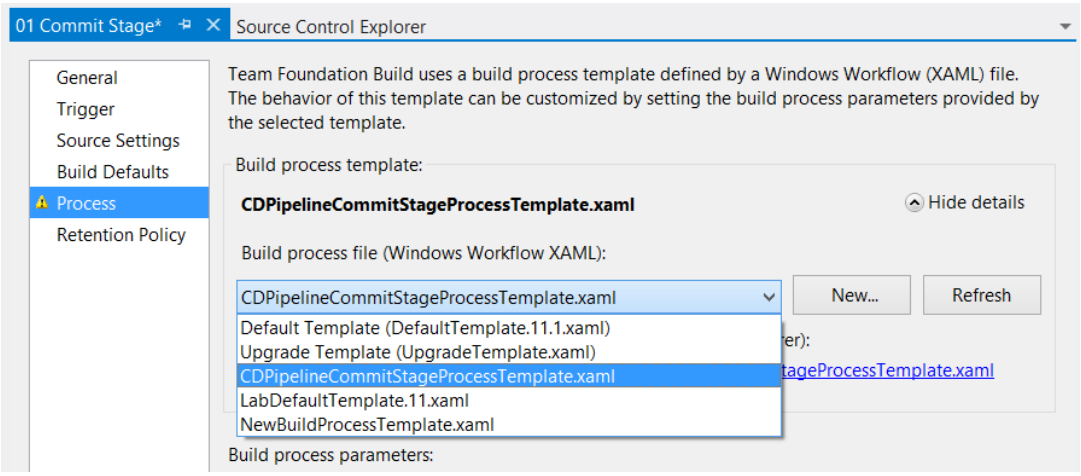
4. In the text box, enter the name of the drop folder. This is where the commit stage will place the binaries.



Task 5: Select the Build Process template

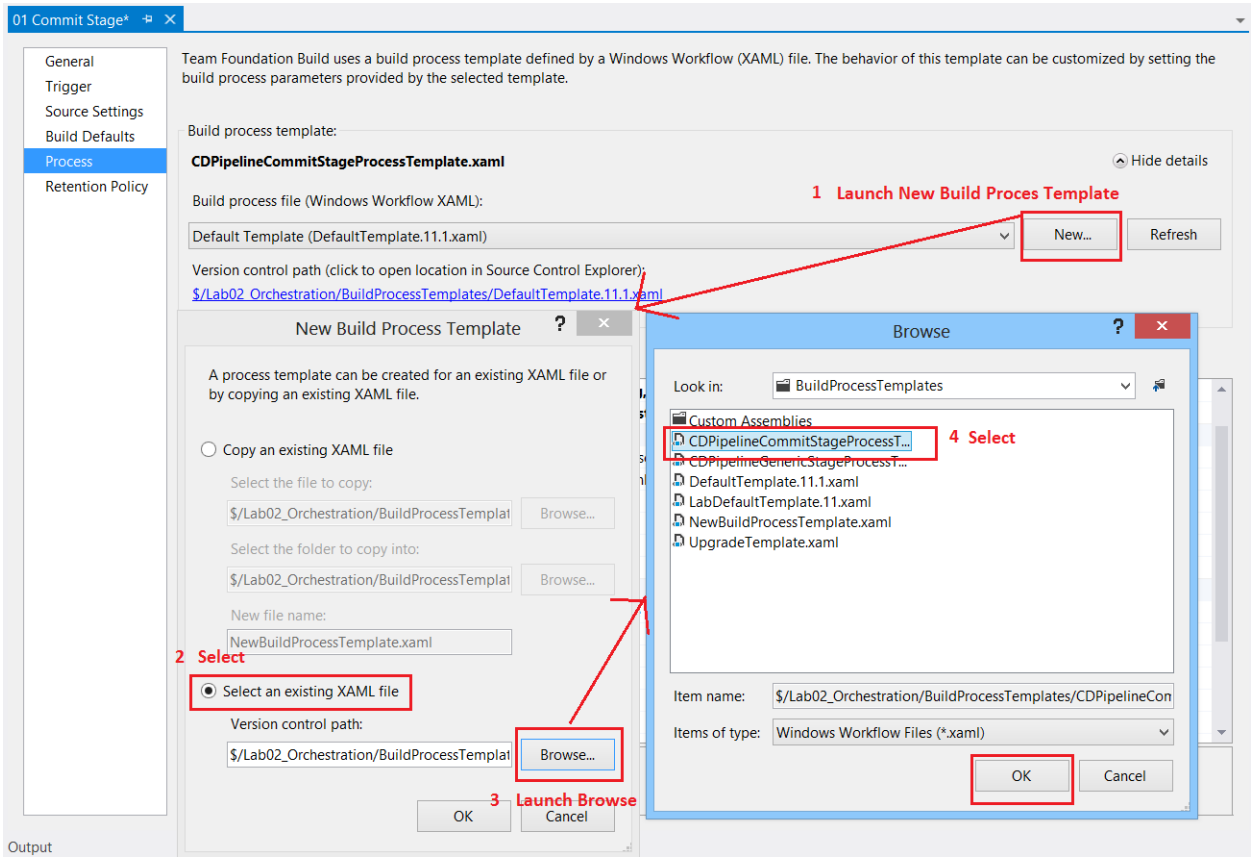
In this task you select the correct build process template.

1. Click the **Process** tab.
2. Change the build process template from the default template to CDPipelineCommitStageProcessTemplate.xaml. Look for it in the combo box named **Build process file (Windows Workflow XAML)**. If the template is there, select it.



3. If the template isn't available, click **New**. The **New Build Process Template Editor** opens.
4. Select **Select an existing XAML file**.

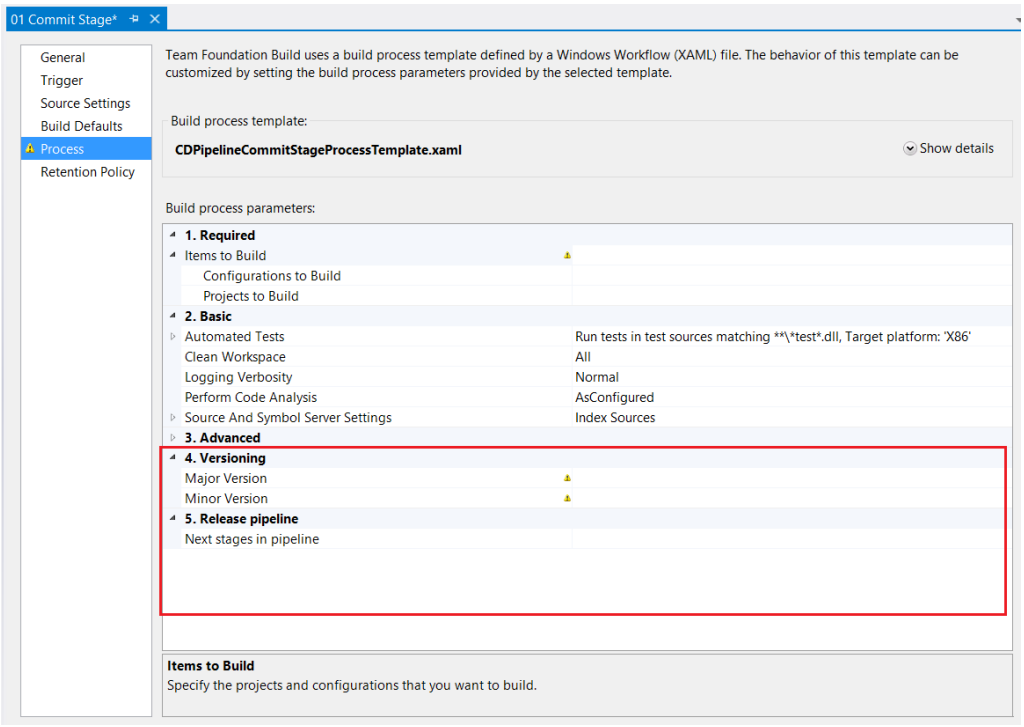
5. Browse to the **CDPipelineCommitStageProcessTemplate.xaml** file. Select it.
6. Click **OK**. Wait for the template to load.



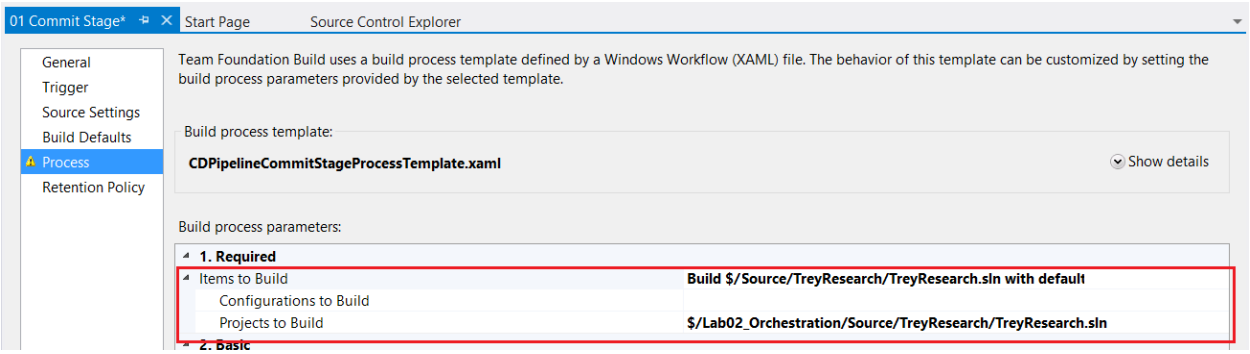
Task 6: Set the Build Process Parameters

In this task you set various build process parameters such as the items to build, the configurations to build, and the MSBuild platform to use.

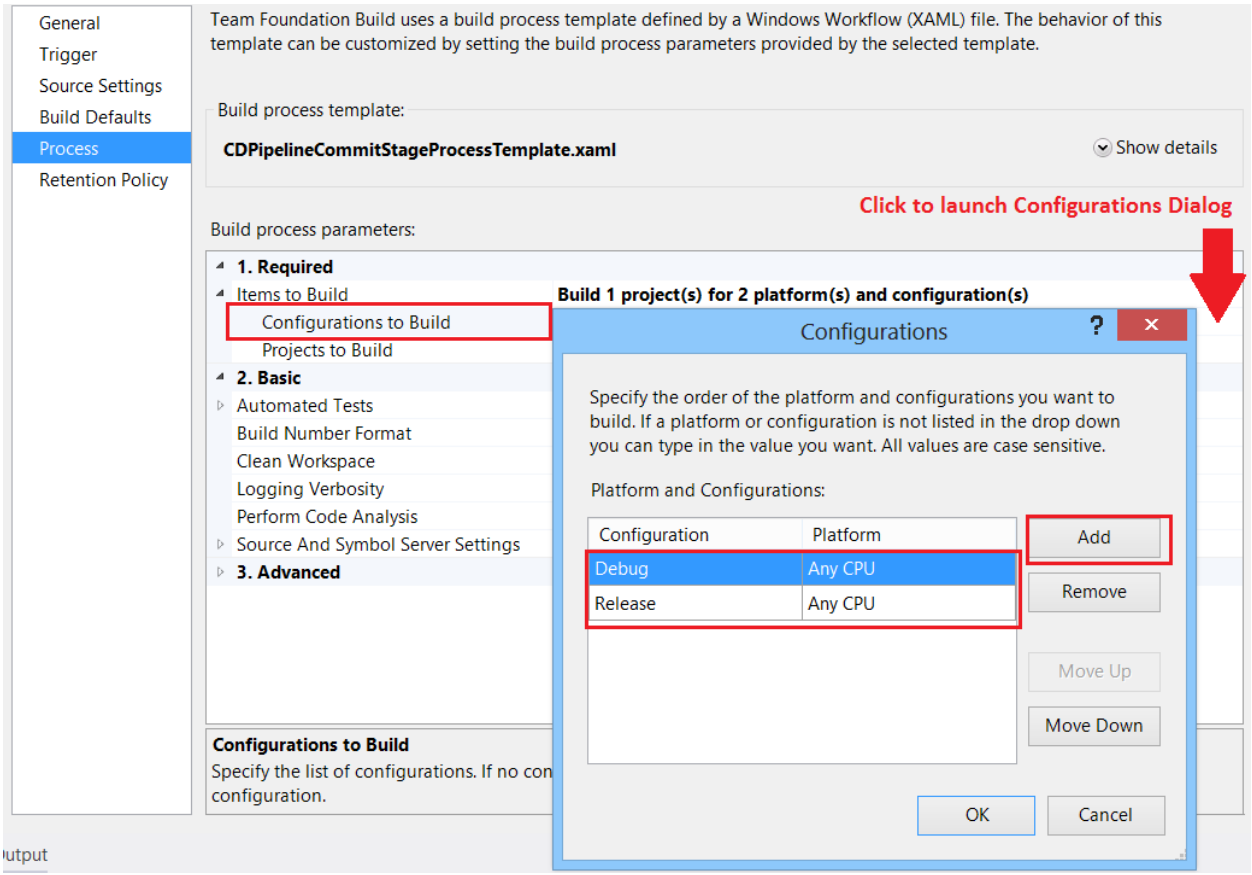
1. Notice that the **Versioning** section contains the arguments you created in Lab2.1 Orchestrating the Commit Stage.



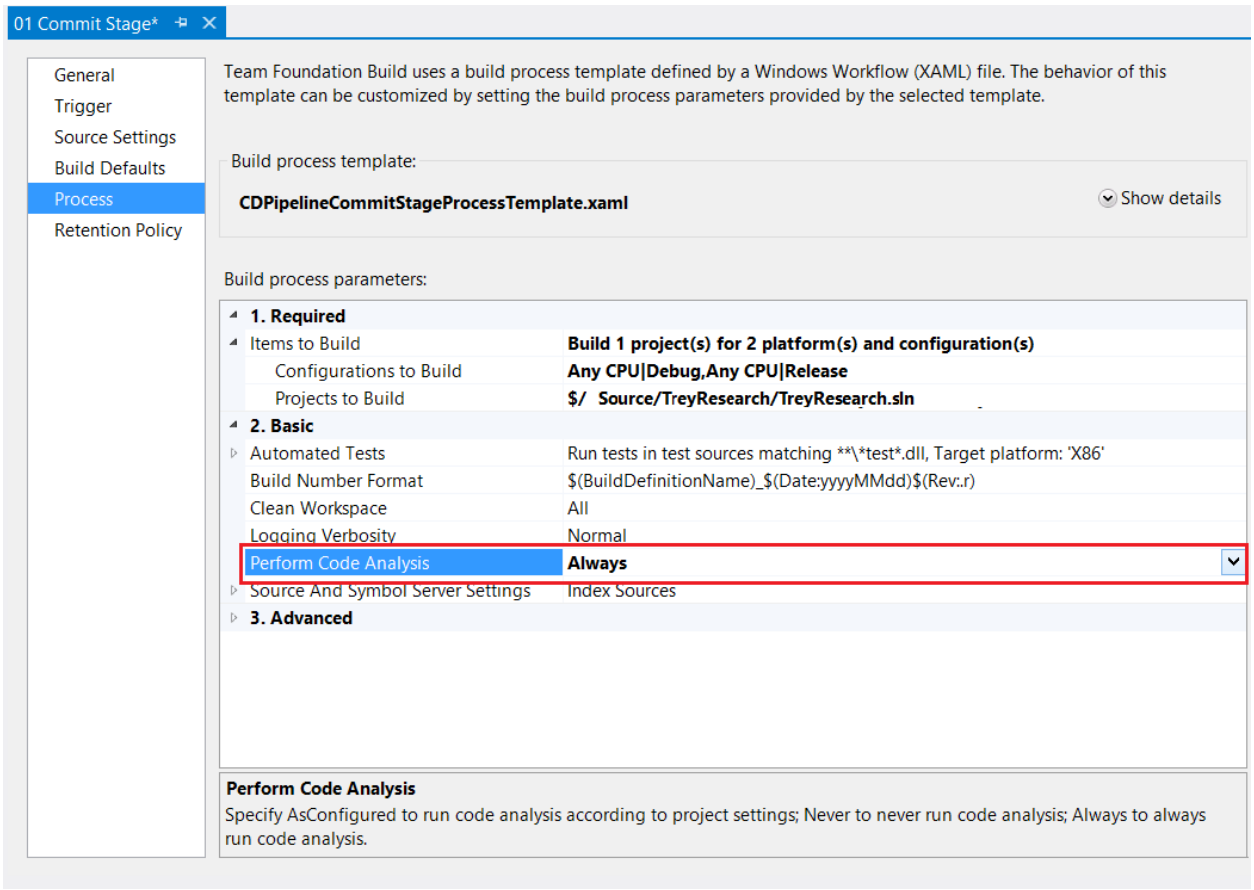
2. Go to the **Required** section of **Build process parameters**. For **Items to Build**, browse to **TreyResearch.sln** and select it.



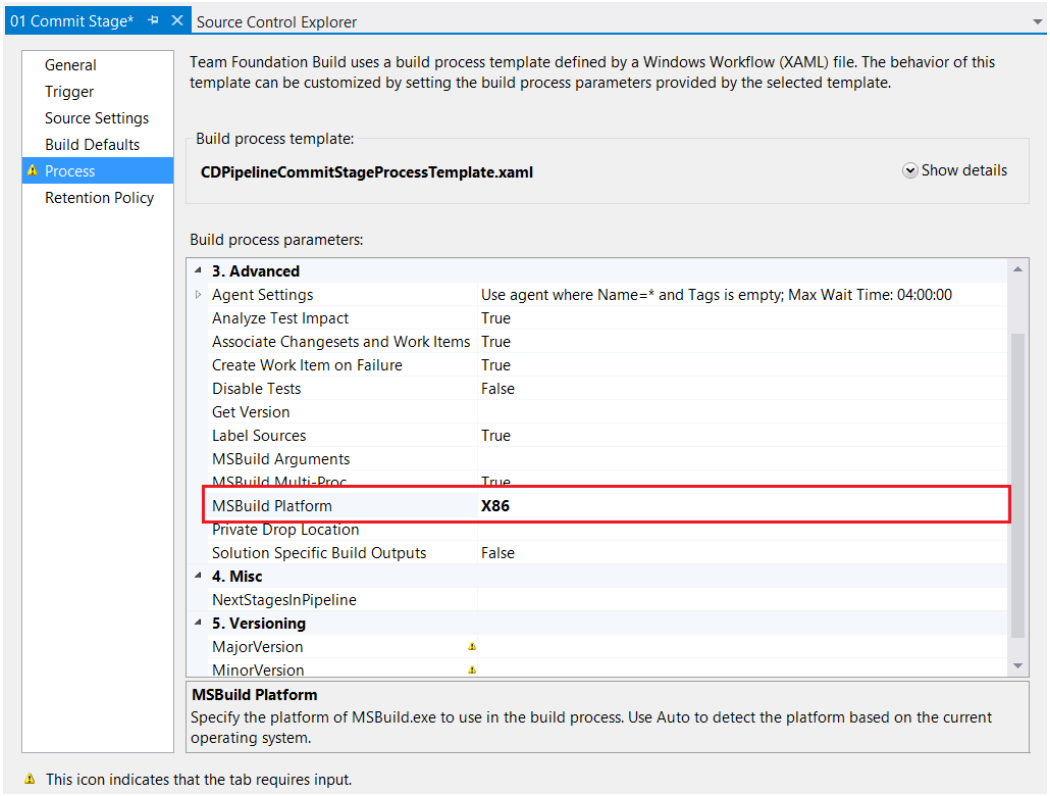
3. Click the **ellipses (…)** in the **Configurations to Build** row. The **Configurations** dialog box appears.
4. In the Configuration column, Click **Add**. Select **Release**. Set platform to **Any CPU**.
5. (Optional) Click **Add**. Select **Debug**. The debug configuration is only used to investigate problems with the application.
6. Click **OK**.



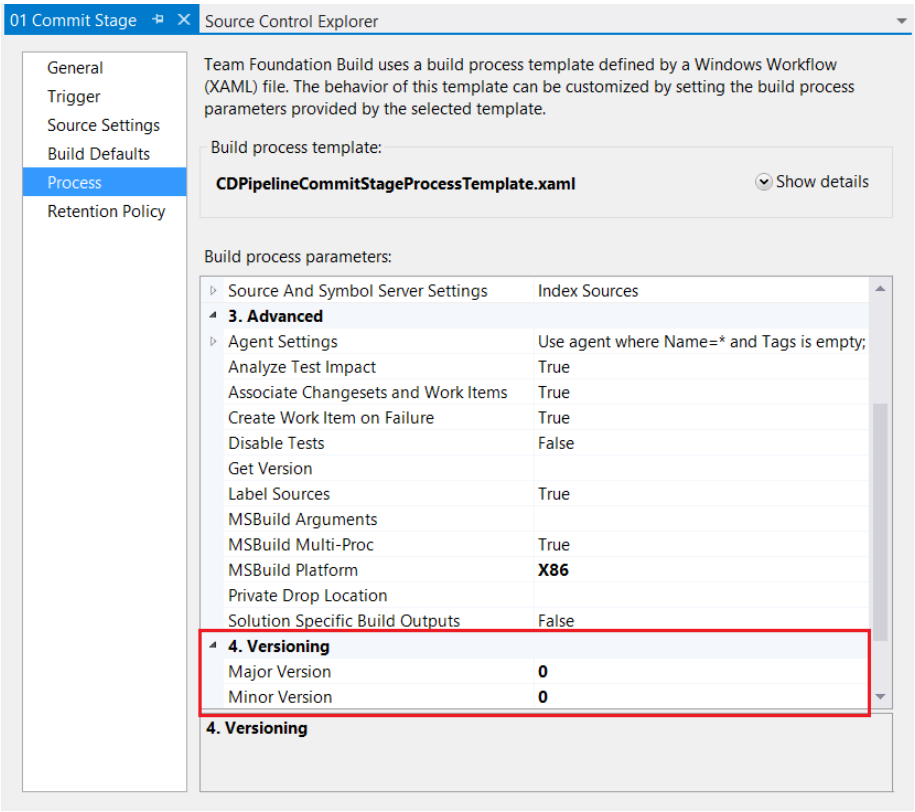
7. Go to the **Basic** section of **Build process parameters**. Set the **Perform Code Analysis** parameter to **Always**. Performing code analysis in the commit stage is, in general, a good practice to follow.



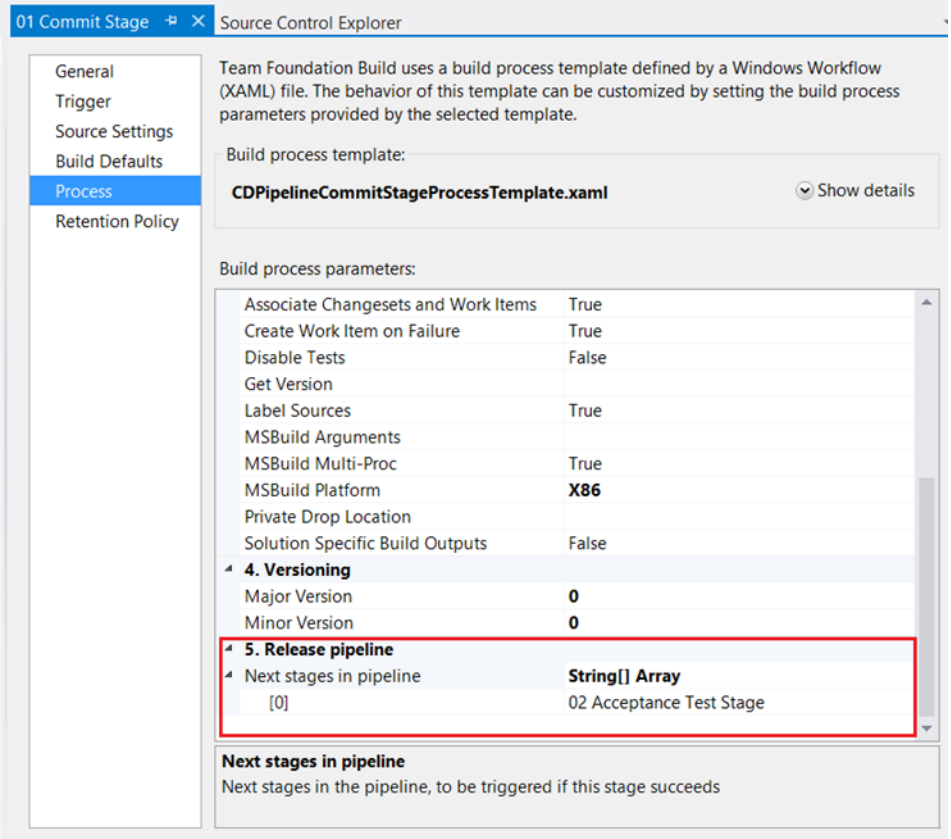
8. (Only if you are implementing the Advanced Phone options) Go to the **Advanced** section of **Build process parameters**. Set the **MSBuild Platform** parameter to **X86**. This setting is required to support the Windows Phone 8 component of the Trey Research application.



9. Go to the **Versioning** section of **Build process parameters**. Set both the **Major Version** and **Minor Version** parameters to **0**.



10. Go to the **Release Pipeline** section of **Build process parameters**. For the parameter array **Next stages in pipeline**, under **String[] Array**, add **02 Acceptance Test Stage**. This is the next stage of the pipeline.



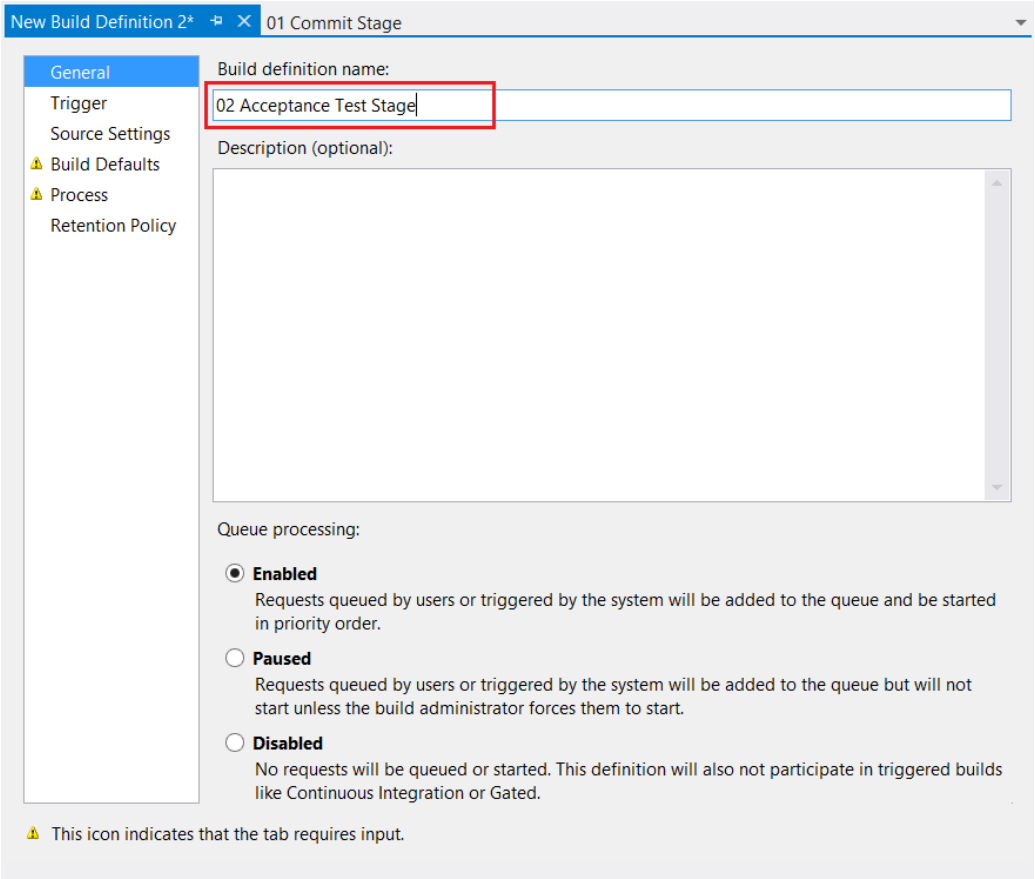
11. Save your work.

Exercise 3: Creating the Acceptance Test Stage for a Pipeline Instance

In this exercise you create a build definition for a particular acceptance test stage that belongs to a particular pipeline instance.

Task 1: Create the 02 Acceptance Test Stage Build Definition

1. Create a new build definition and name it **02 Acceptance Test Stage**. You can follow the same steps that you used in exercise 2, task 1. The following screen shot shows the **General** tab of the build definition.

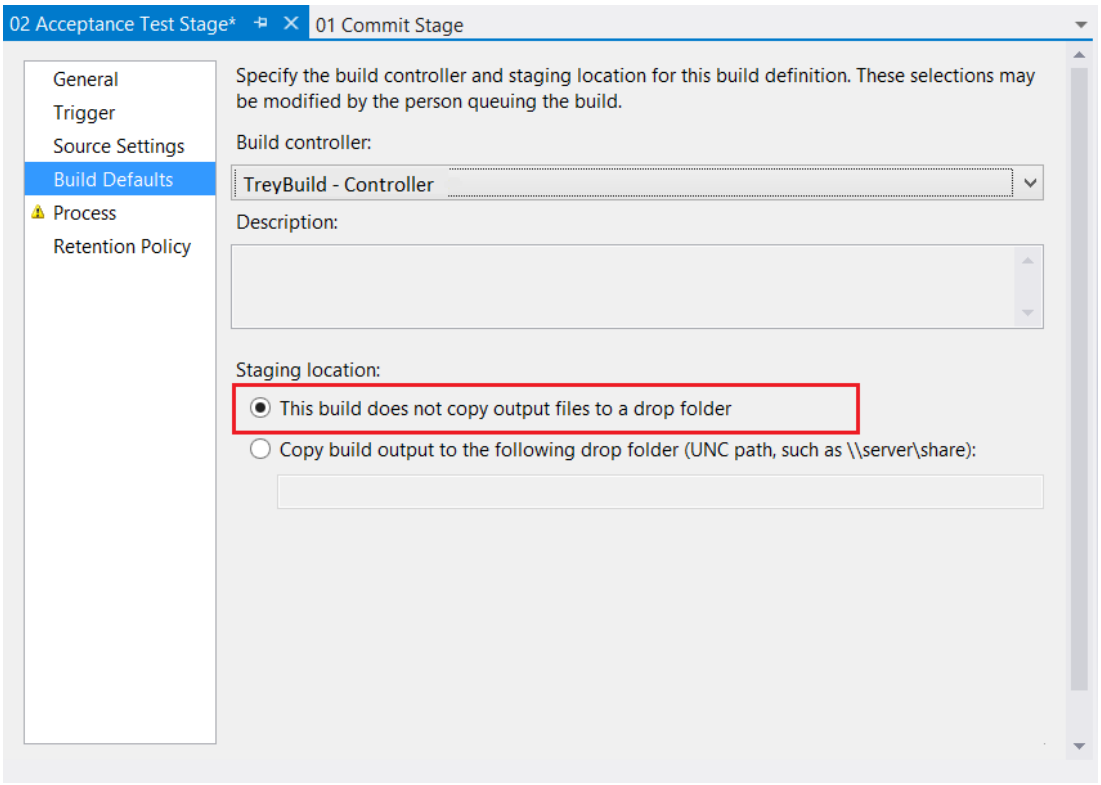


2. In the commit stage the **Trigger** and **Source** were explicitly set. For this stage nothing needs to change in the **Trigger** and **Source Settings** tabs. The acceptance test stage uses the default setting of **Manual** and uses the source settings that you already configured.

Task 2: Change the Build Defaults

In this task you change the staging location.

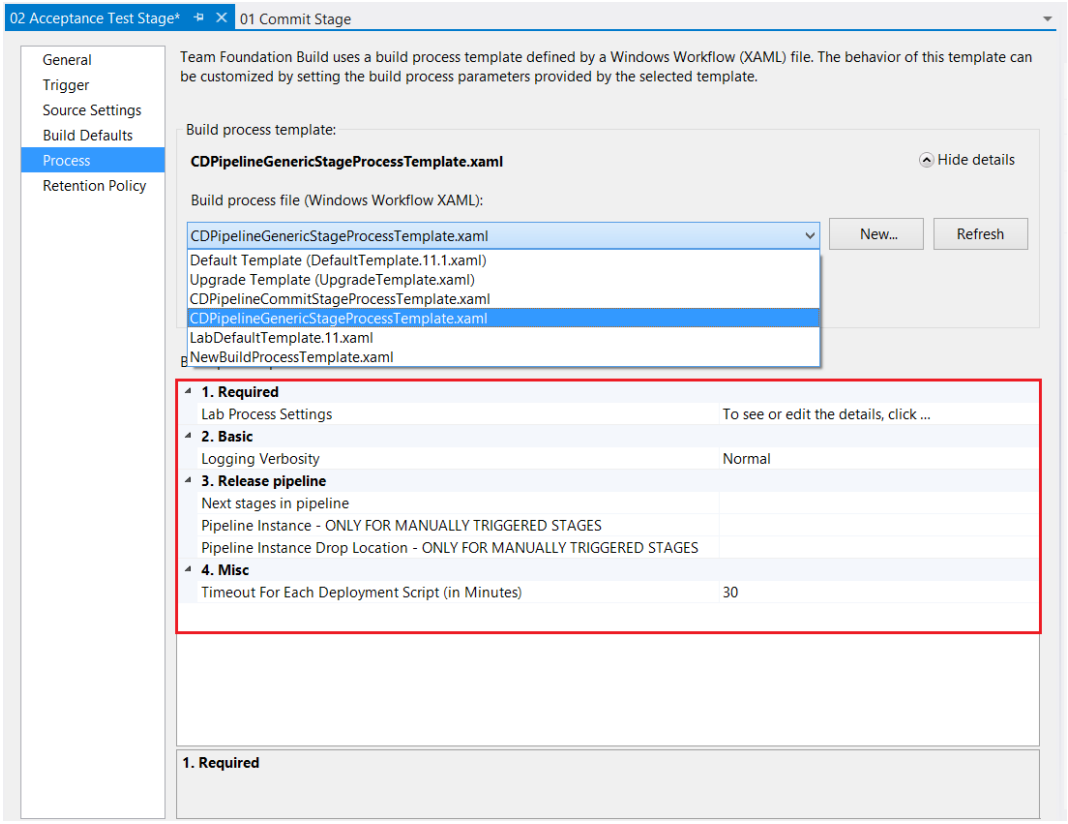
1. Select the **Build Defaults** tab.
2. For **Staging location**, select **This build does not copy output files to a drop folder**.



Task 3: Select the Build Process Template

In this task you select the correct build process template.

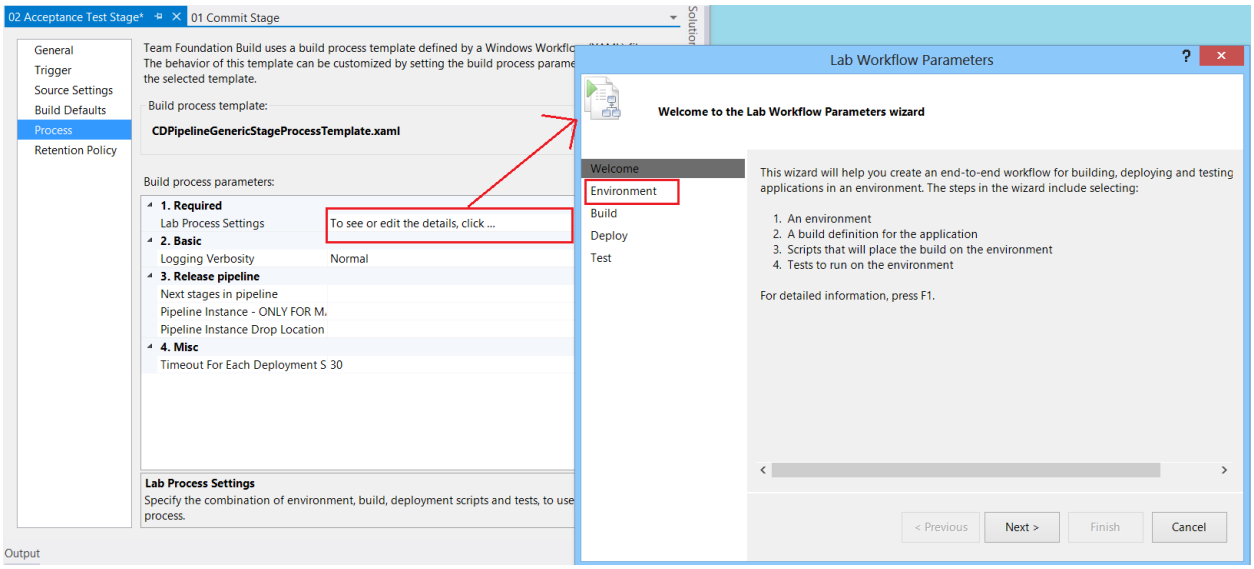
1. Select the **Process** tab.
2. Change the build process template from the default template to CDPipelineGenericStageProcessTemplate.xaml. You can follow the same steps that you used in exercise 2, task 5. The following screenshot shows the **Process** tab of the build definition.



Task 4: Set the Build Process Parameters

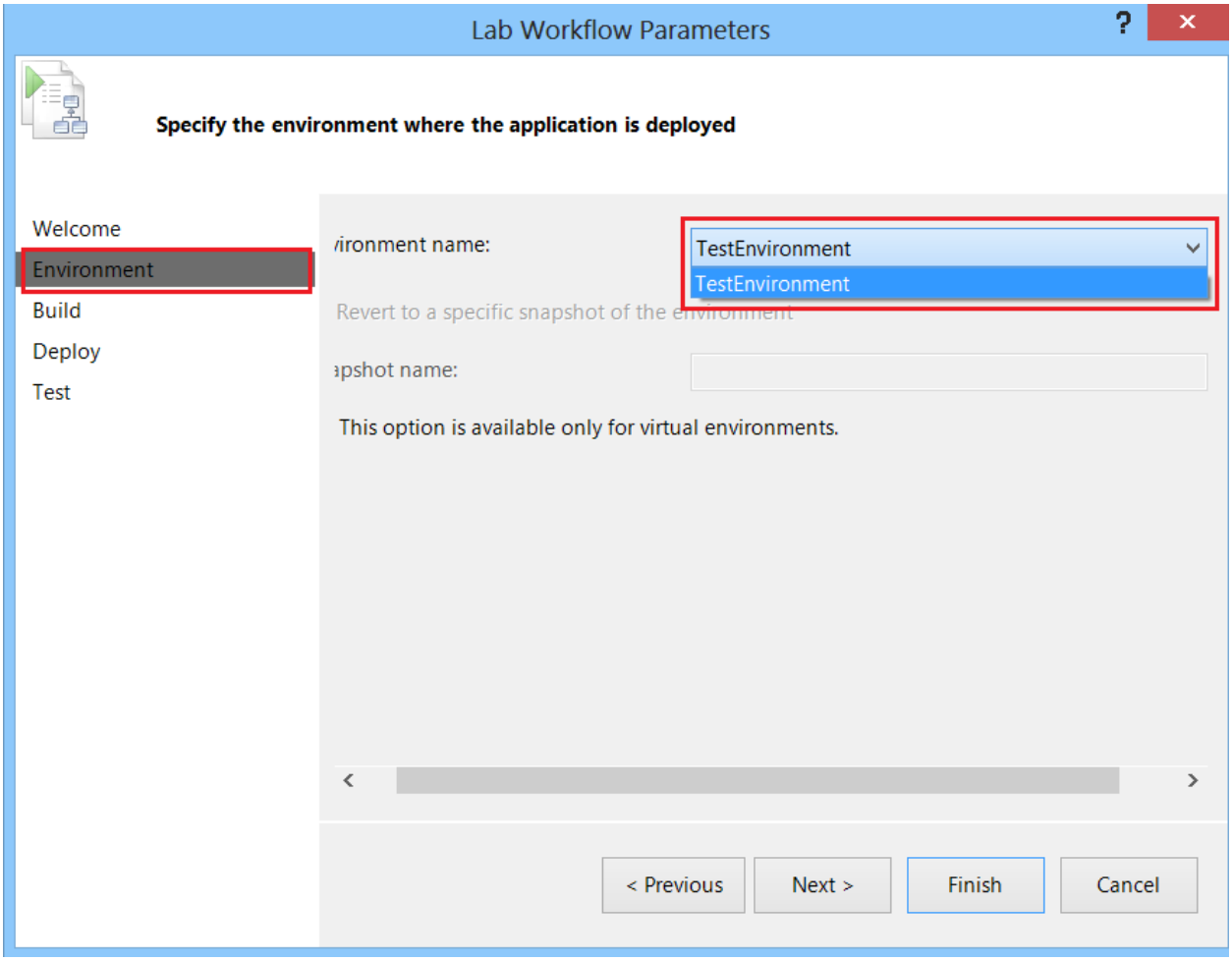
In this task you set various build process parameters such as the test environment and the versioning values.

1. Select the **Process** tab. In the **Required** section, click the **ellipses (…)**. The **Lab Workflow Parameters** wizard opens. Click **Next**.

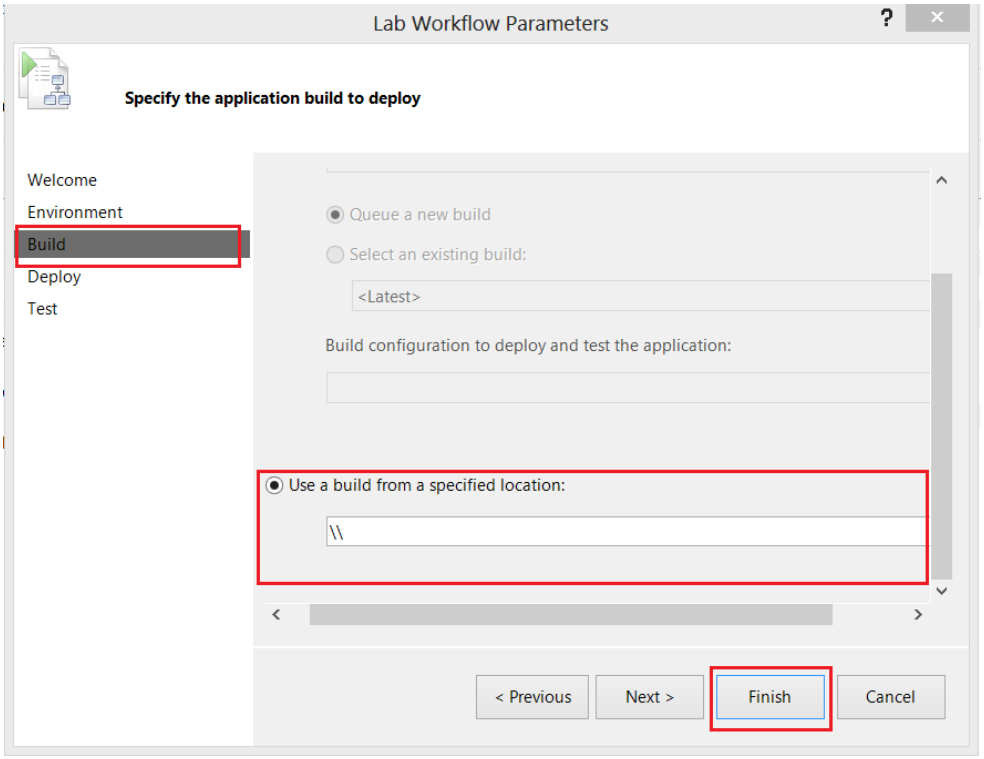


- Add the test environment. (To set up a new environment in Lab Management refer to **Lab 1 – StartingPoint**, exercise 5. If there is no environment, the acceptance test stage throws an error.) Click **Next**.

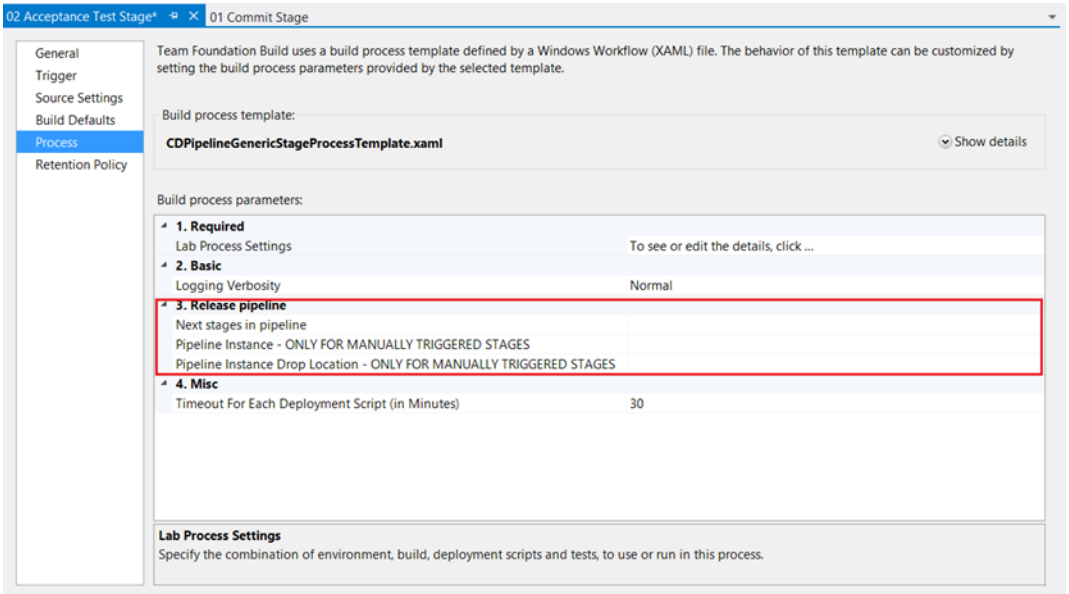
> **Note:** For more information on creating lab environments see Creating Lab Environments.



2. The **Build** tab should be selected. Select **Use a build from a specified location**. Provide a fake value for the parameter. The parameter isn't actually used. Click **Finish**. Save your work.



You have now configured the acceptance test stage. Because this stage uses the default **Manual** setting, the build location is passed as a parameter from the commit stage. Because the stages that follow the acceptance test stage are manually triggered, the **Next stages in pipeline** parameter is left empty for now.



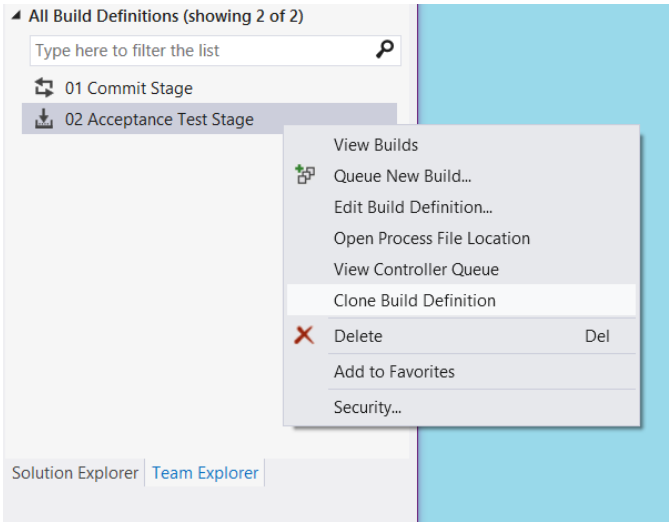
Exercise 4: Adding the Manual Stages

In this exercise you add the release stage and the UAT stage. They are both manually triggered. These stages are configured the same way as the 02 Acceptance Test Stage, so you can clone them from that build definition.

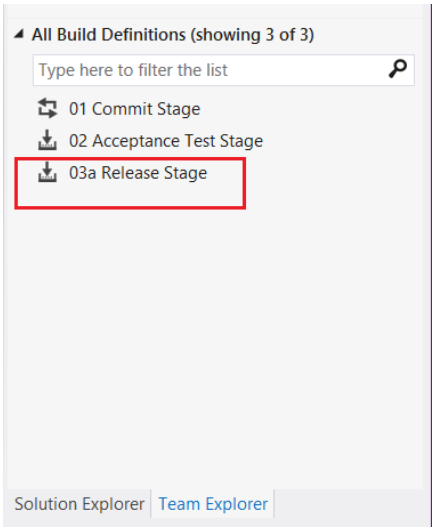
Task 1: Add the Manual Stage 03a Release Stage

In this task you add the definition for the release stage. You do this by using **Clone Build Definition**, which is included in Team Foundation Server 2012 Power Tools.

1. Right-click on the build definition named 02 Acceptance Test Stage. Click **Clone Build Definition**.



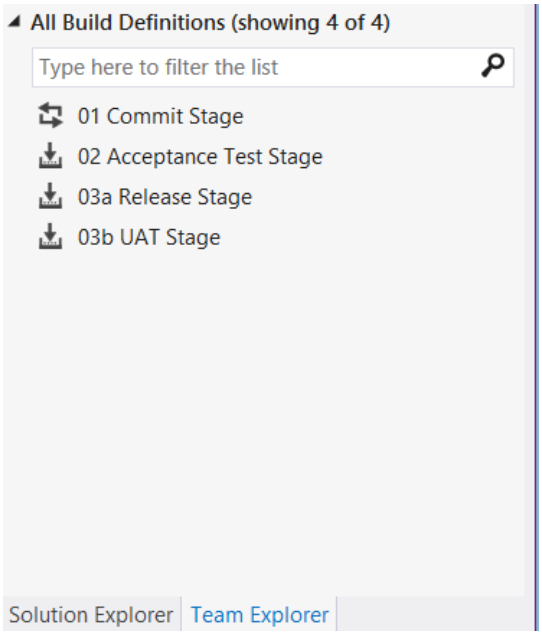
2. Rename the cloned build definition to **03a Release Stage** and save it.



Task 2: Add the Manual Stage 03b UAT Stage

In this task you add the definition for the UAT stage.

1. Right-click on the build definition named **02 Acceptance Test Stage** and clone it.
2. Rename the cloned build definition to **03b UAT Stage** and save it.



Summary

In this HOL you added the customized build templates that you created in the earlier labs to your project. You then defined a particular commit stage that belongs to a particular pipeline instance. You set the trigger so that a build occurs if there is a check-in. You also set the working folder, the build controller, and the drop location. Finally you set the build process parameters.

Next, you repeated the process for a particular acceptance test stage. You created its build definition, changed the build defaults and set the build process parameters.

Finally, you added two manually triggered stages. One is the release stage and the other is the UAT stage.

Copyright

examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred.