



# Visual Studio Build Customization

---

## What's new in Team Foundation Build 2012 Hands-On Lab

Monday, June 25, 2012

**Visual Studio ALM Rangers**

Fabio Stawinski, Mike Fourie

**Microsoft Corporation**



Please consider the environment before printing this document

### **Visual Studio ALM Rangers**

This content was created by the Visual Studio ALM Rangers, a special group with members from the Visual Studio Product Team, Microsoft Services, Microsoft Most Valued Professionals (MVPs) and Visual Studio Community Leads.

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

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

Microsoft grants you a license to this document under the terms of the Creative Commons

Attribution 3.0 License. All other rights are reserved.

© 2012 Microsoft Corporation.

Microsoft, Active Directory, Excel, Internet Explorer, SQL Server, Visual Studio, and Windows are trademarks of the Microsoft group of companies.

All other trademarks are property of their respective owners.

## Table of Contents

Introduction.....	5
Visual Studio ALM Rangers.....	5
Prerequisites .....	5
Paths.....	5
Suggested Focus and Time Map.....	5
Exercise 1: Creating a new Team Project.....	6
Exercise 2: Team Explorer Build Panel.....	11
Exercise 3: Queuing builds and the build context menu .....	21
Exercise 4: Build Summary Extensibility .....	25

## Table of Figures

Figure 1 – Click Microsoft Visual Studio 2012 to start the application .....	6
Figure 2 – Opening the VS Team Explorer .....	6
Figure 3 – Creating a new Team Project .....	7
Figure 4 – Creating a new Team Project .....	7
Figure 5 – Select a Process Template .....	8
Figure 6 – Skip SharePoint configuration.....	8
Figure 7 – Specify Source Control Settings .....	9
Figure 8 – Project Creation progress .....	9
Figure 9 – Home Panel.....	10
Figure 10 – New Project menu .....	11
Figure 11 – New Project Window .....	11
Figure 12 – Adding project in the source control .....	12
Figure 13 – Application running successfully.....	12
Figure 14 – Access to Builds Panel.....	13
Figure 15 – New Build Definition link .....	13
Figure 16 – Gated Check-in build setup.....	14
Figure 17 – Drop folder.....	15
Figure 18 – Items to build .....	15
Figure 19 – Build Definition recently created .....	16
Figure 20 – Running the build.....	16
Figure 21 – Queue Build window.....	17
Figure 22 – Build started.....	17
Figure 23 – Failed Build .....	18
Figure 24 – Pending changes .....	18
Figure 25 – Successfully Checked-in .....	19
Figure 26 – Builds list .....	19
Figure 27 – Add to Favorites.....	19
Figure 28 – Favorite Builds .....	20
Figure 29 – New build definition .....	21
Figure 30 – Builds queued .....	21
Figure 31 – Context menu - Completed builds.....	22
Figure 32 – Context menu - Queued builds.....	22
Figure 33 – Context menu - Open Item .....	23
Figure 34 – Build Request window .....	23
Figure 35 – Visual Studio 2010 build window for comparison .....	24
Figure 36 – Build Request after retrying.....	24
Figure 37 – View the template our builds use .....	25
Figure 38 – Add and configure the WriteCustomSummaryInformation activity.....	25
Figure 39 – Custom summary section displayed .....	26

## Introduction

Welcome to the **What's new in Team Foundation Build 2012** Hands-on Lab (HOL). This HOL was created by the Visual Studio ALM Rangers team and will describe the main changes made in the new version of Visual Studio and its integration with Team Foundation Server.

## Visual Studio ALM Rangers

Visual Studio ALM Rangers is a special group with members from the Visual Studio Product group, Microsoft Services, Microsoft Most Valued Professionals (MVP) and Visual Studio Community Leads. Their mission is to provide out of band solutions to missing features and guidance. A growing [Rangers Index](#) is available online<sup>1</sup>. This guide targets all users of Visual Studio.

## Prerequisites

To complete these hands-on-lab walk-through scenarios you need the following environment:

- Visual Studio ALM Rangers Base Image

or

- A single server (physical or virtual) environment that has the following software installed and configured

Software	Version
Microsoft Windows	7, 8, Server 2008 / R2
Visual Studio	Visual Studio 2012
Team Foundation Server	Team Foundation Server 2012

## Paths

The HOL refers to the working directory as **<HOL\_PATH>** in this document, which by default refers to the following path on Rangers VMs: **C:\HOL**. We suggest you use this path as longer paths may encounter issues with file length limits.

## Suggested Focus and Time Map

If you are intending to follow the Hands-on Lab (HOL) step by step, use these times as a guideline. If, however, you are intending to investigate each step in detail, do double the times as a bare minimum.

Topic	Duration in minutes	Page
Exercise 1: Creating a new Team Project	5	6
Exercise 2: Team Explorer Build Panel	15	11
Exercise 3: Queuing builds and build context menu	15	21
Exercise 4: Build Summary Extensibility	10	25
<b>TOTAL</b>	<b>45 min</b>	

**Table 1 – Suggested focus and time map**

*We wish you a pleasant and interesting journey!*

---

<sup>1</sup> Rangers Index - [http://blogs.msdn.com/b/willy-peter\\_schaub/archive/2010/06/18/introducing-the-visual-studio-alm-rangers-an-index-to-all-rangers-covered-on-this-blog.aspx](http://blogs.msdn.com/b/willy-peter_schaub/archive/2010/06/18/introducing-the-visual-studio-alm-rangers-an-index-to-all-rangers-covered-on-this-blog.aspx)

## Exercise 1: Creating a new Team Project



### OBJECTIVE

In this Exercise, we start and setup the environment to use in this HOL and we also get to know the new Team Explorer and its Home section.

1. Start Visual Studio: **Start, All Programs, Microsoft Visual Studio 2012, Visual Studio 2012** (if your user account is not an administrator, right-click and use the “Run as Administrator” option)

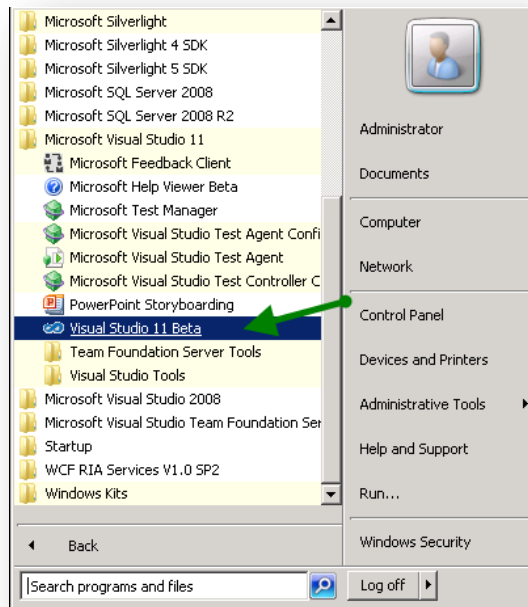


Figure 1 – Click Microsoft Visual Studio 2012 to start the application

2. Click **View, Team Explorer**

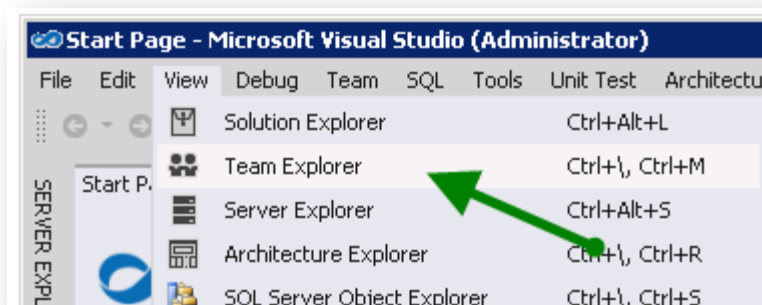


Figure 2 – Opening the VS Team Explorer

3. Connect to an available **Team Foundation Server** of your choice
4. Now go to Team Explorer and select **New Team Project** from the drop down list.

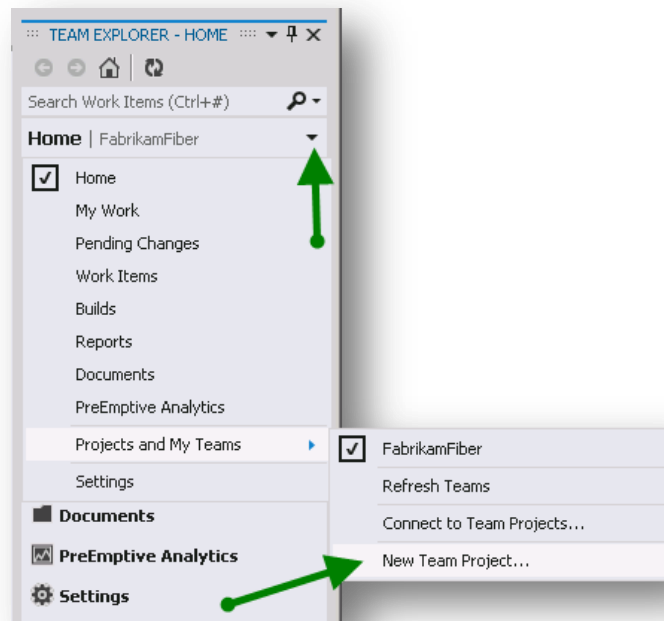


Figure 3 – Creating a new Team Project

5. Set the name of the new Team Project, e.g. **vNextHOL** and then click **Next**

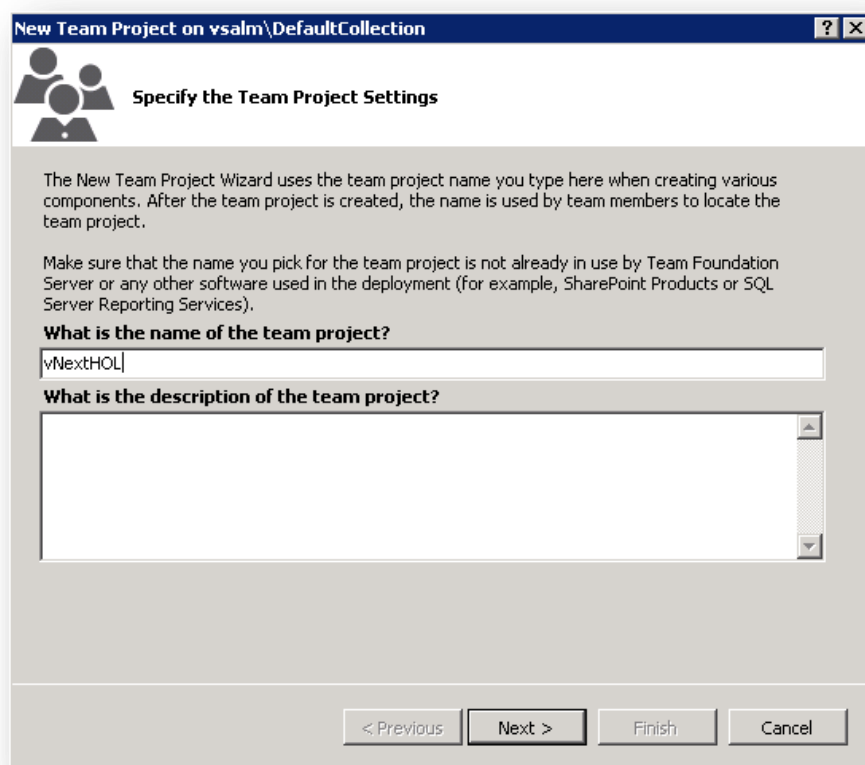


Figure 4 – Creating a new Team Project

6. Select any **Process Template**, then click **Next**

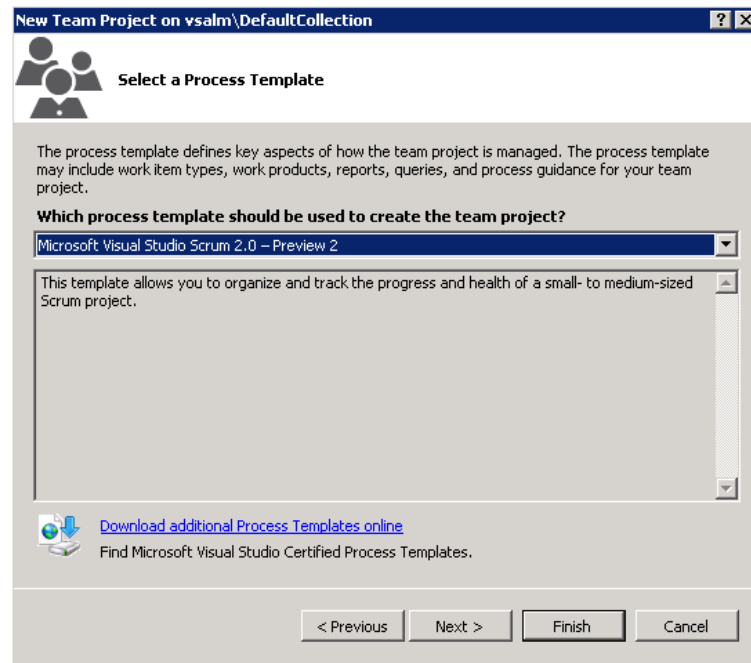


Figure 5 – Select a Process Template

7. If prompted, select **Do not configure a SharePoint site at this time** and click **Next**

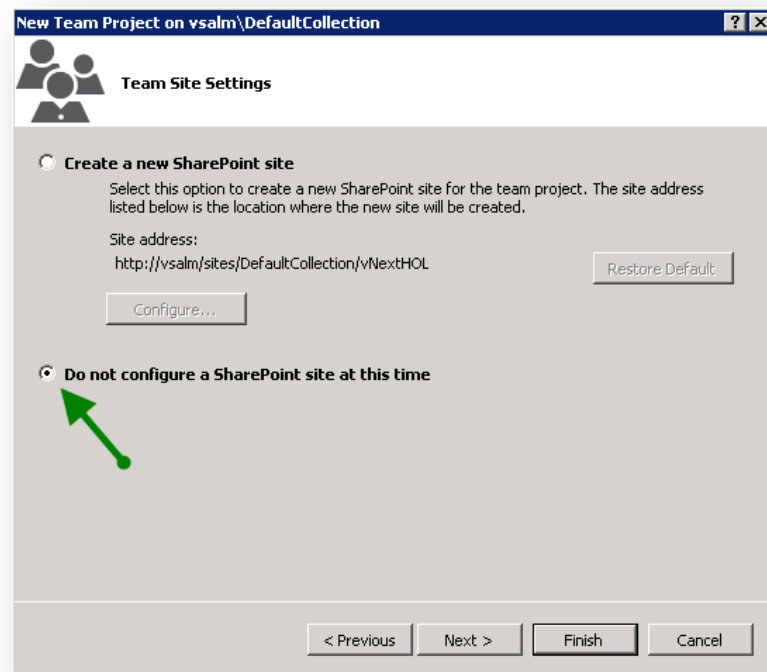


Figure 6 – Skip SharePoint configuration

8. In the screen Specify Source Control Settings, select **Create an empty source control folder** and click **Finish**. *It may take a few minutes to create your project.*



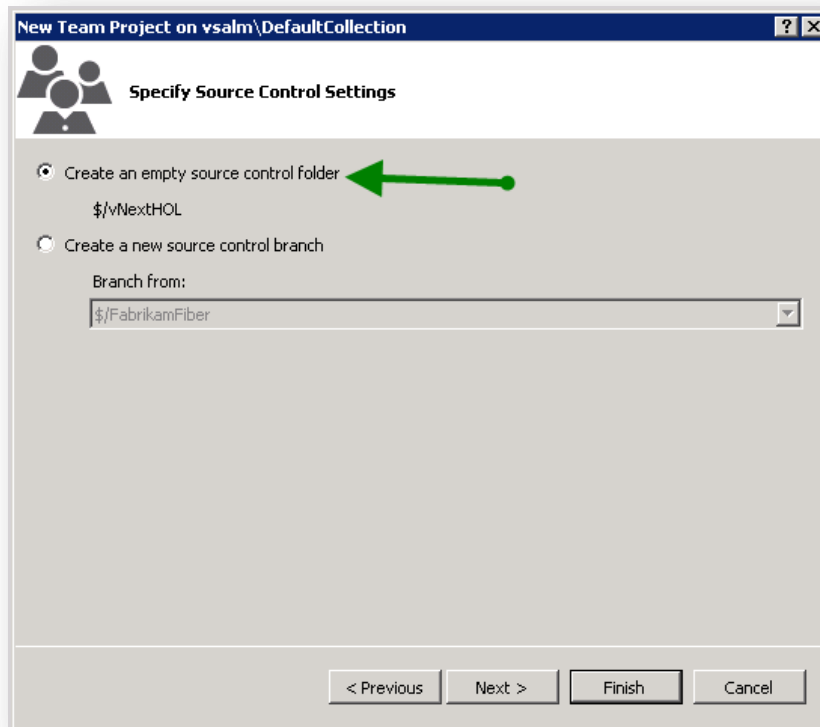


Figure 7 – Specify Source Control Settings

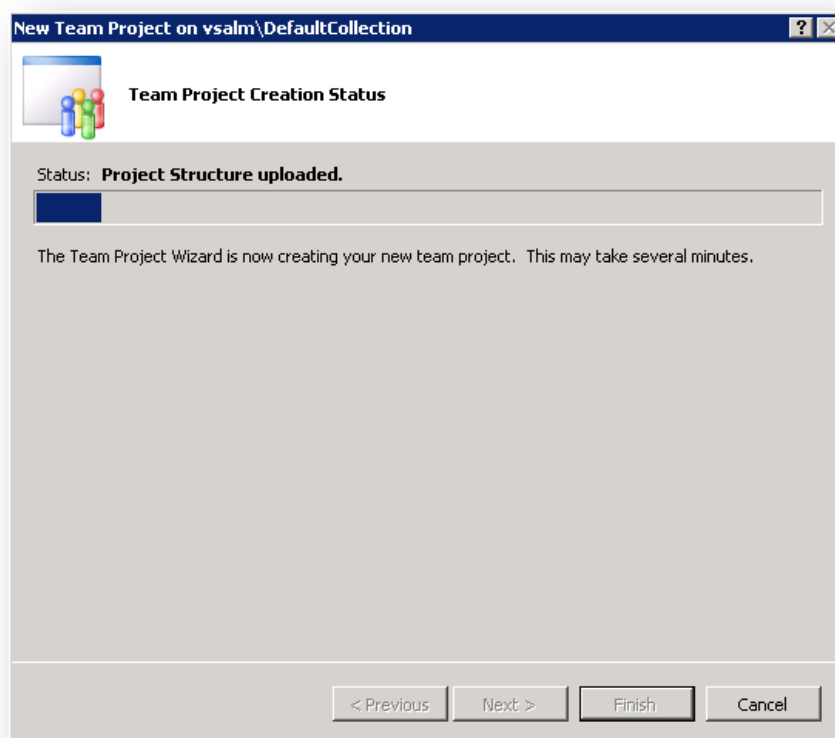


Figure 8 – Project Creation progress

9. When the project has been created click **Close**
10. **Team Explorer** will load your new **Team Project**

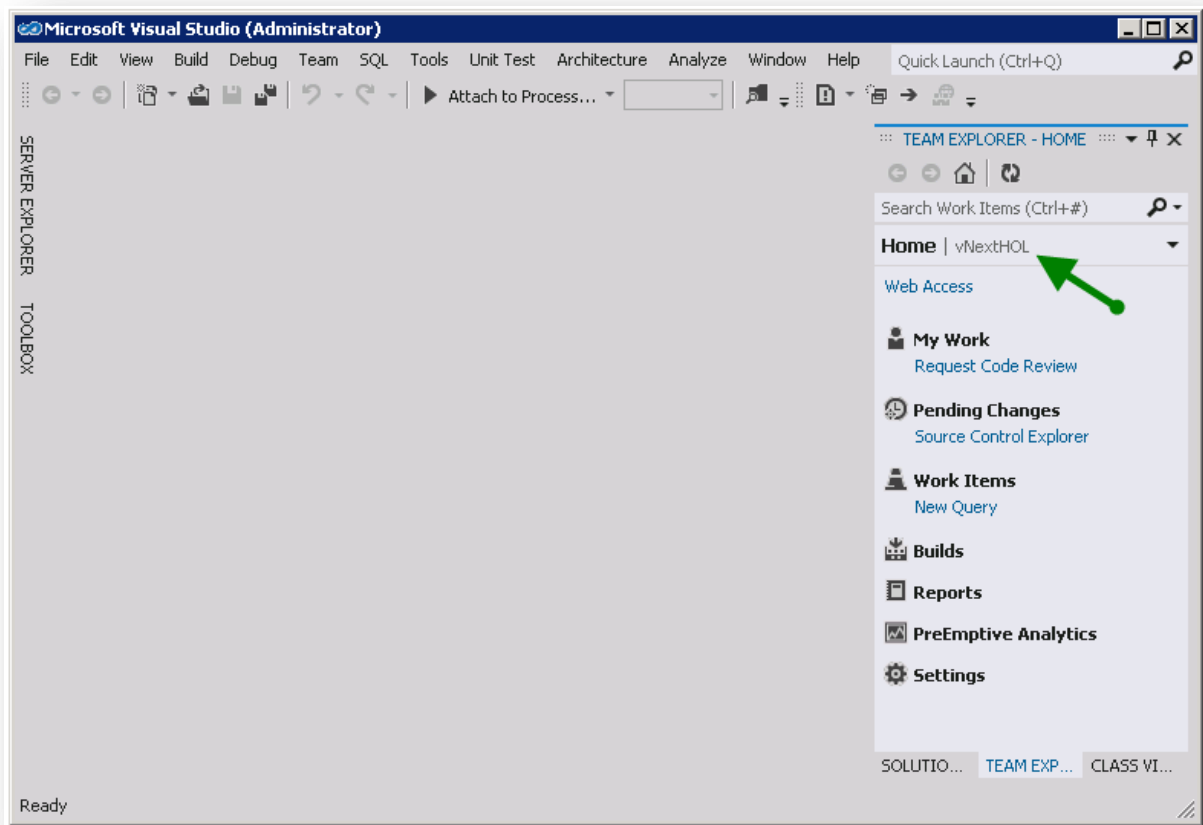


Figure 9 – Home Panel



### REVIEW

We have created a new Team Project, opened Team Explorer and viewed the new Home panel.

## Exercise 2: Team Explorer Build Panel



### OBJECTIVE

In this Exercise, we'll explore the Build panel included in the new Team Explorer. We will create a console application to be used as a sample and also create a new build definition to build it. Finally we finish by running the build created.

1. Still connected to the Team Project **vNextHOL**, create a new console application. In **Visual Studio 2012** click **File, New, Project**

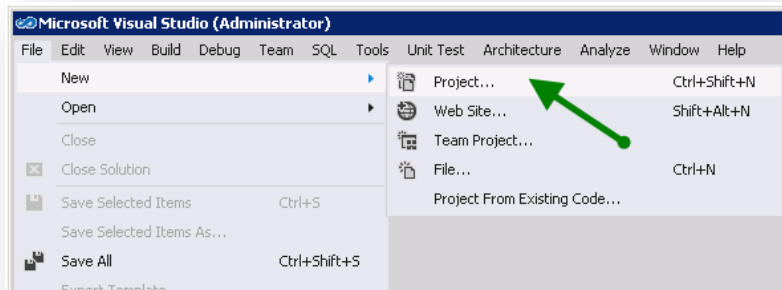


Figure 10 – New Project menu

2. In the “**New Project**” window, select **Console Application** under Visual C#, set the name to “**vNextConsole**”, check “**Create directory for solution**” and “**Add to source control**” and then click **OK**.

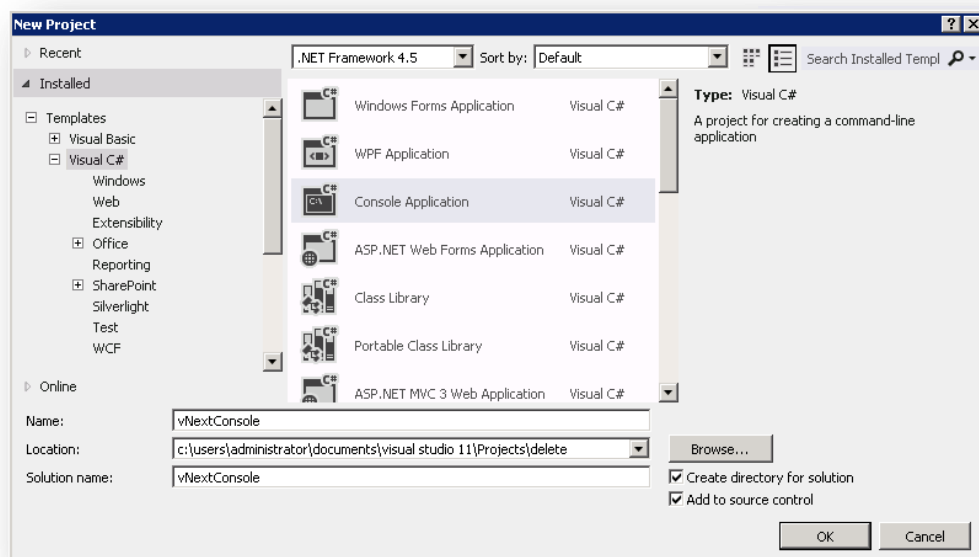


Figure 11 – New Project Window

3. Now you must select where this Console Application will be located in source control. Keep the default values and click **OK**.

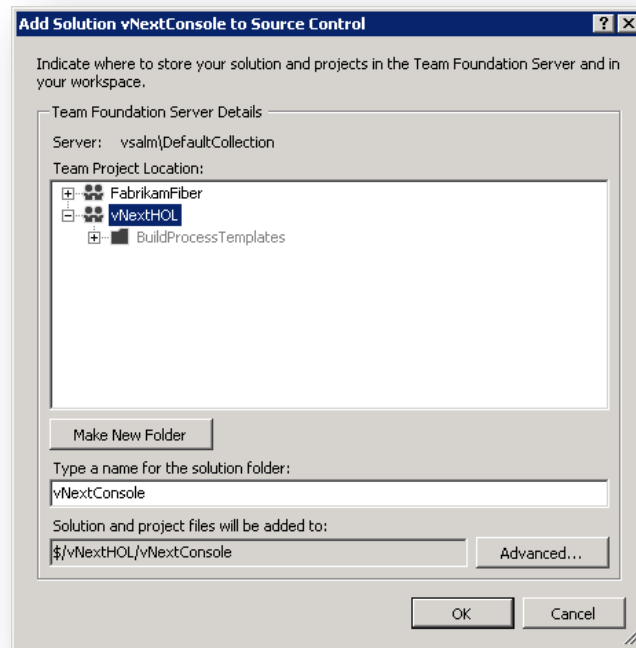


Figure 12 – Adding project in the source control

4. Add two lines in the Main method of **Program.cs** file as follows:

```
static void Main(string[] args)
{
    System.Console.WriteLine("HOL vNext");
    System.Console.ReadKey();
}
```

5. **Run** the Console Application for the first time by clicking the Run button or pressing F5 to verify if everything is ok

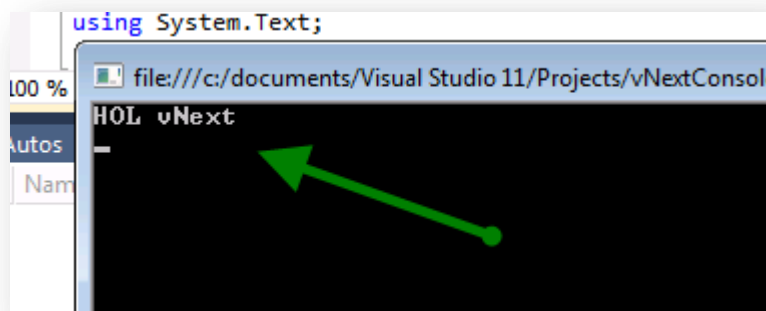


Figure 13 – Application running successfully

6. Now let's create a new build definition, click on **Builds** from the **Home** drop down list

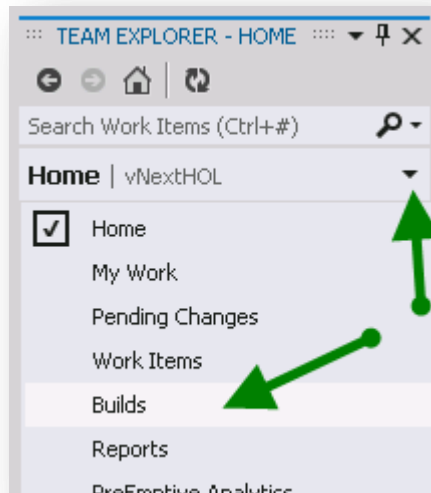


Figure 14 – Access to Builds Panel

7. Click on “New Build Definition”

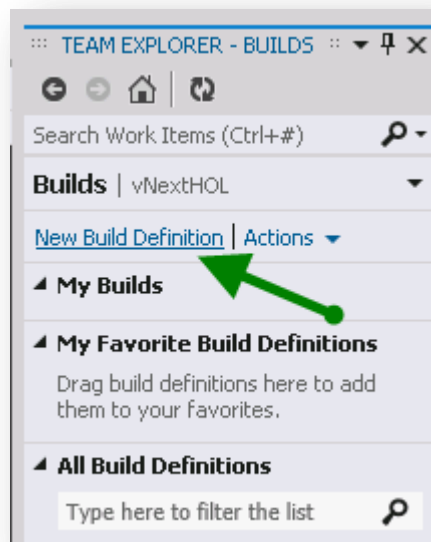


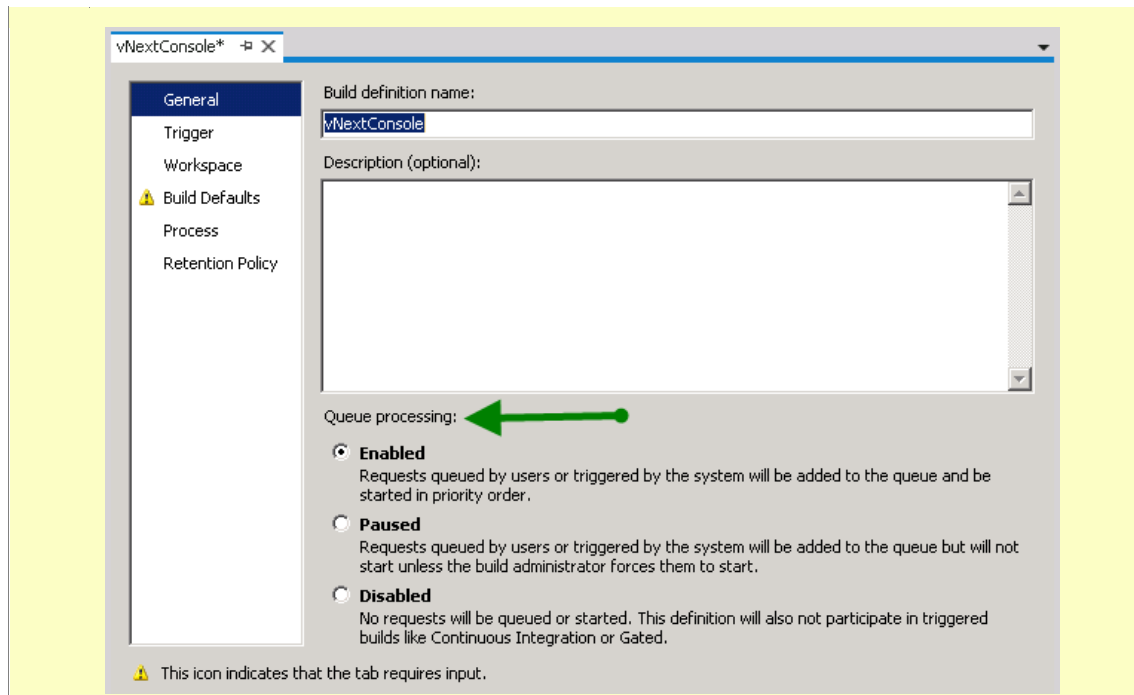
Figure 15 – New Build Definition link

8. In the **General tab**, set the name of the build definition to **vNextConsole**

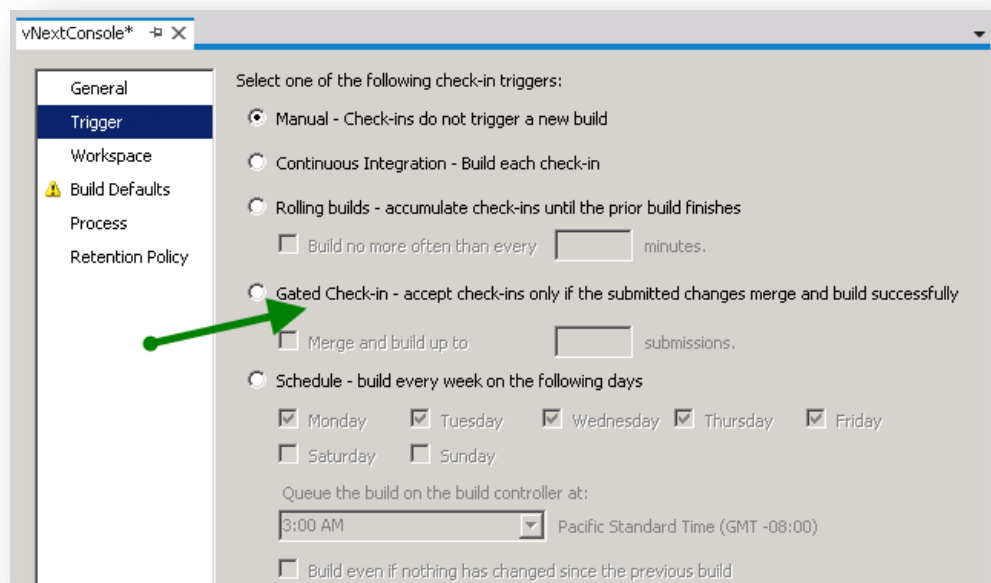


**More about Queue processing**

There are three types of queue processing, Enabled, Disabled, and the new Paused option. The latter can be queued normally but only the administrator can force it to start.



9. Go to the **Trigger** tab, here we have something new related to gated check-ins. It's possible to set the gated check-ins to simultaneously build up to a defined number of submissions. **Leave your build as Manual**



**Figure 16 – Gated Check-in build setup**

10. Jump to **Build Defaults** tab directly and set a drop folder. You can use an existing share or create a new shared folder.

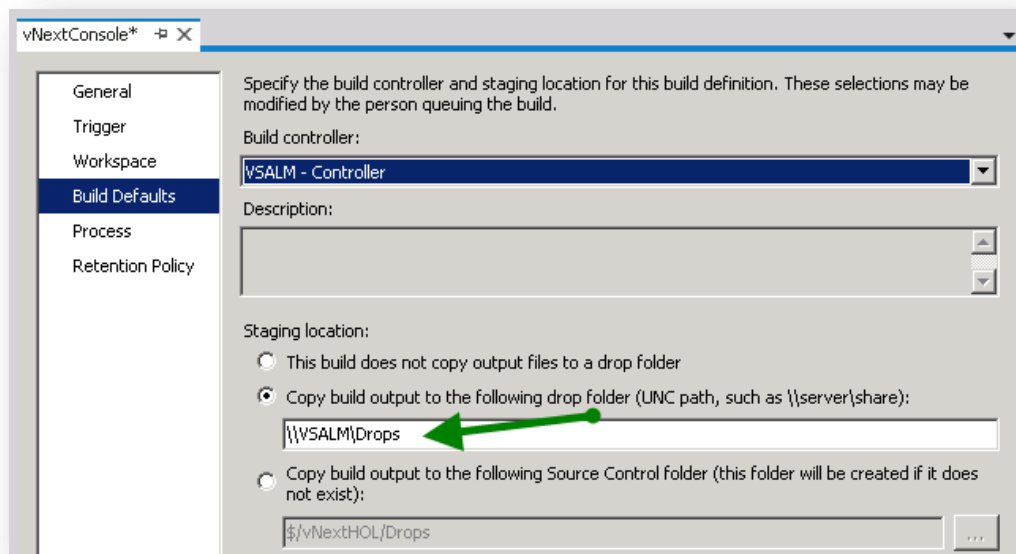


Figure 17 – Drop folder

- Go to **Process** tab, and confirm that the solution file is set in the “Items to Build” field otherwise set it.

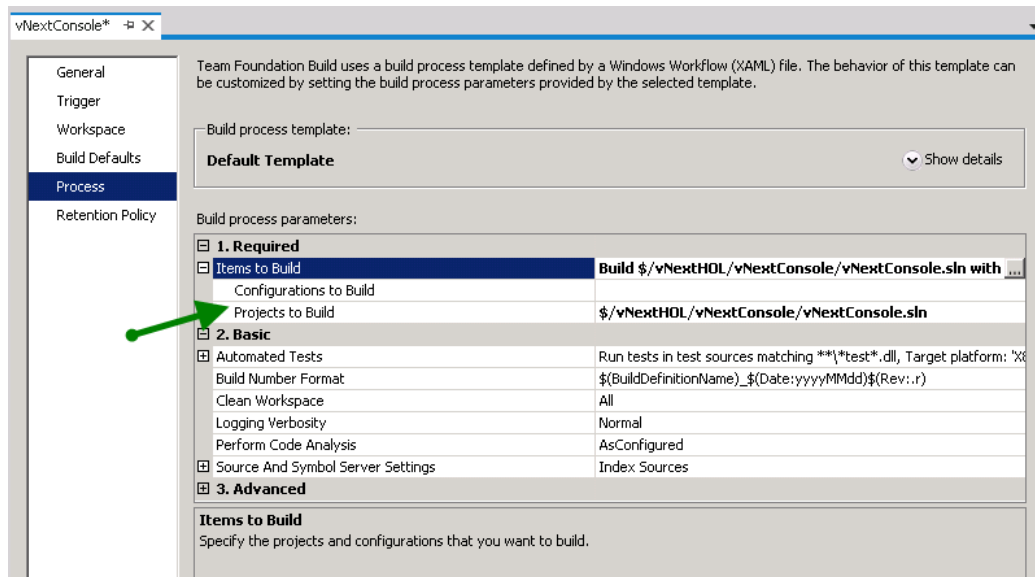


Figure 18 – Items to build

- Now **save** it and your build definition is ready to run.

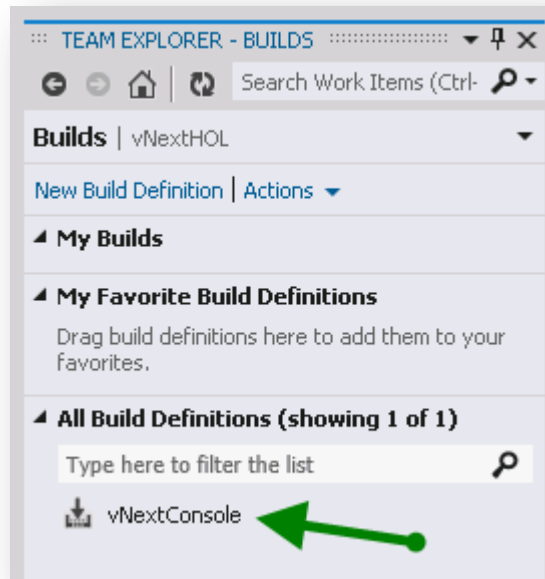


Figure 19 – Build Definition recently created

13. Run the build. **Right click** the build name and select **Queue New Build...** option

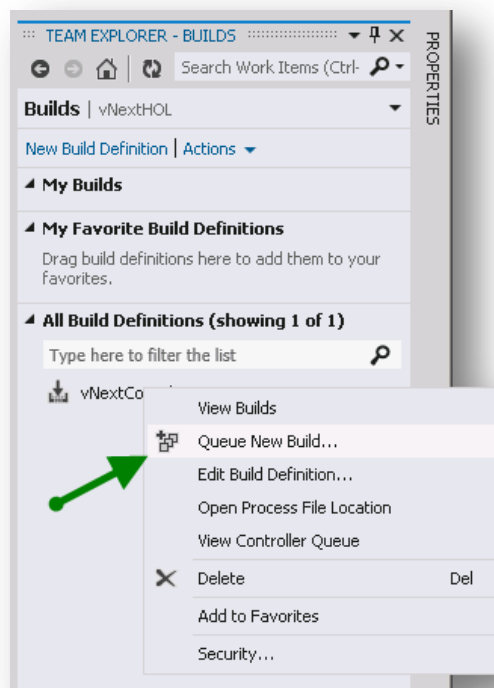


Figure 20 – Running the build

14. Once selected the item above, you will see the **Queue Build** window, where you can customize some parameters, keep everything as it, and click **Queue**.



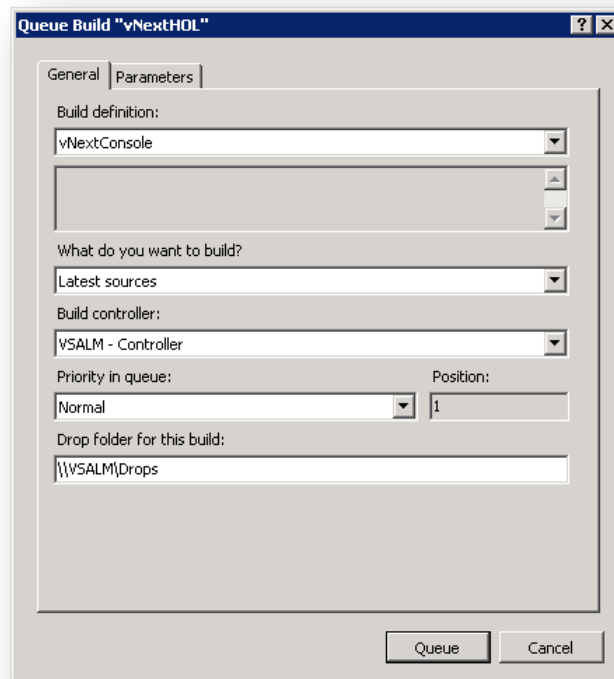


Figure 21 – Queue Build window

15. Right after you queue a build, you can see the builds you have queued below “My Builds”.

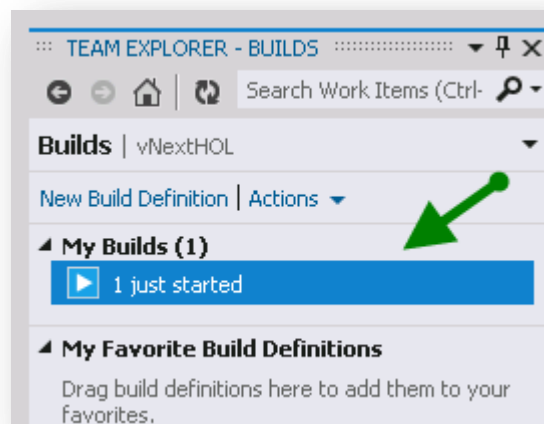


Figure 22 – Build started

16. Your build should fail. This happened because the new project has not yet been checked in. You must check in all files you have included in your solution.



Figure 23 – Failed Build

17. To do this, go to the **Pending Changes** panel in Team Explorer and check in all files listed there by clicking the “**Check In**” button

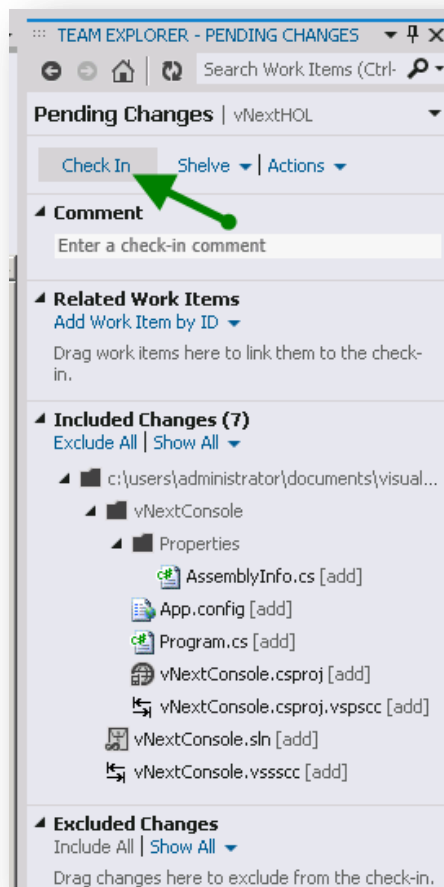


Figure 24 – Pending changes

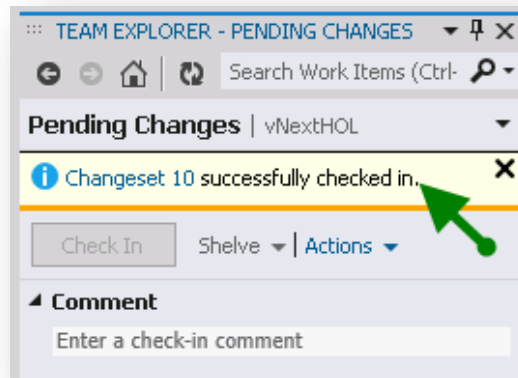


Figure 25 – Successfully Checked-in

18. Go back to **Builds** panel and run the build over again

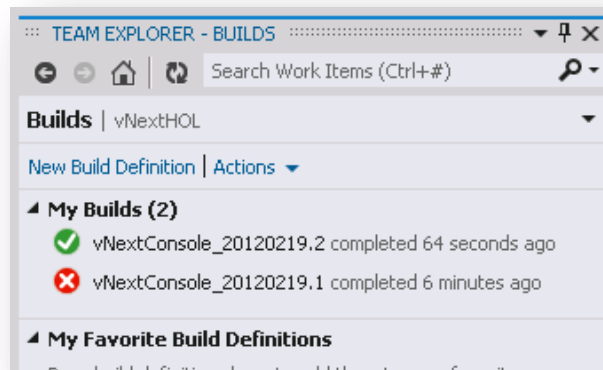


Figure 26 – Builds list

19. Now right click the build definition and select the option **Add to Favorites**, or drag and drop the build definition on the **Favorite Build Definitions** area.

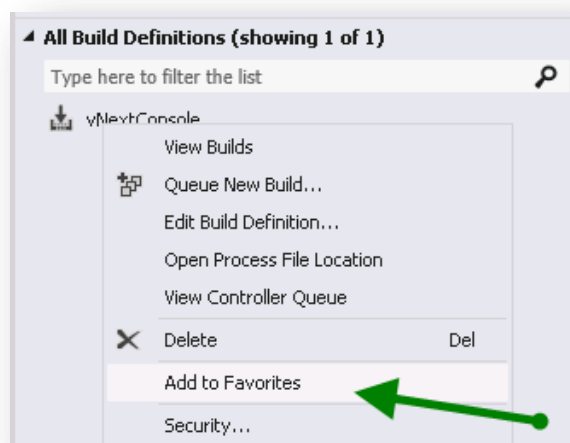


Figure 27 – Add to Favorites

20. Check out the **Favorite Build Definitions** area, and notice that is possible to see the last builds and also the Favorite builds are marked with a star in the list of "All Build Definitions".

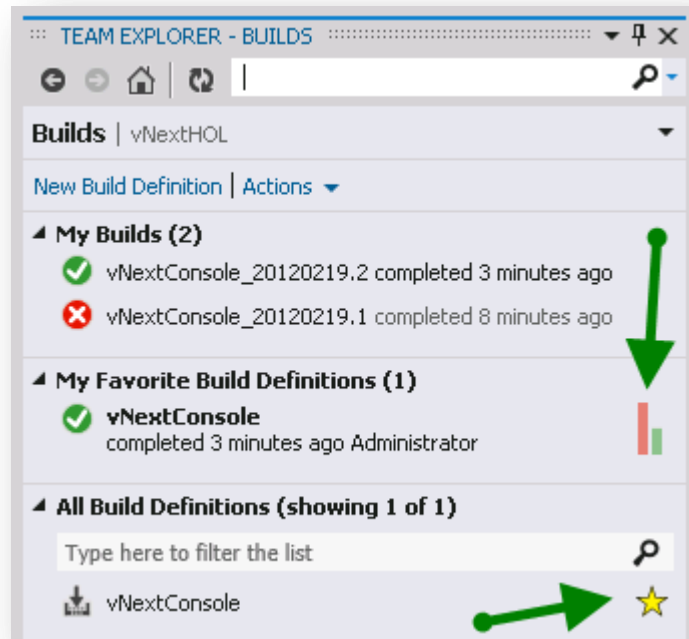


Figure 28 – Favorite Builds



#### REVIEW

We have created a new console application, gone through the Team Explorer Builds panel, created a new build definition, selected a favorite build, and looked through the My Builds section.

## Exercise 3: Queuing builds and the build context menu



### OBJECTIVE

In this Exercise, we will create another build definition to show how to queue different builds at the same time, and explain the My Builds section (Position in the queue and priority), completed build and build to be run. We are going to compare the context menus of running versus completed builds and will also have a look at the Build Request and Build Detail windows.

1. Follow the steps to create a build found in the previous exercise and create a new build definition called **"Sample Build 2"**. For simplicity, it can build the same console application.

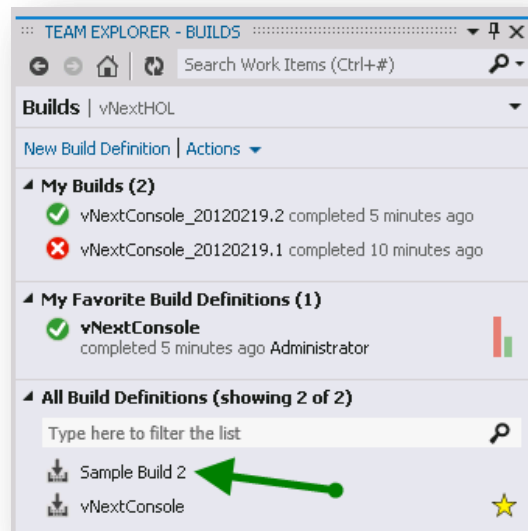


Figure 29 – New build definition

2. Queue both builds definitions 3 times each. Notice the build **position** and the **priority**.

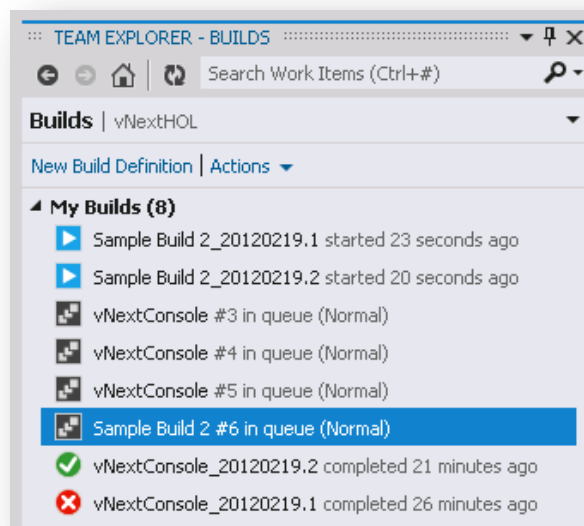


Figure 30 – Builds queued



### Build position and priority

Notice that in My Builds section, it is possible to see what is the build position and the priority as well. Obviously the builds run from high priority to low priority, and the build positions are given according to the priority.

3. Test clicking a completed build with the right mouse button.

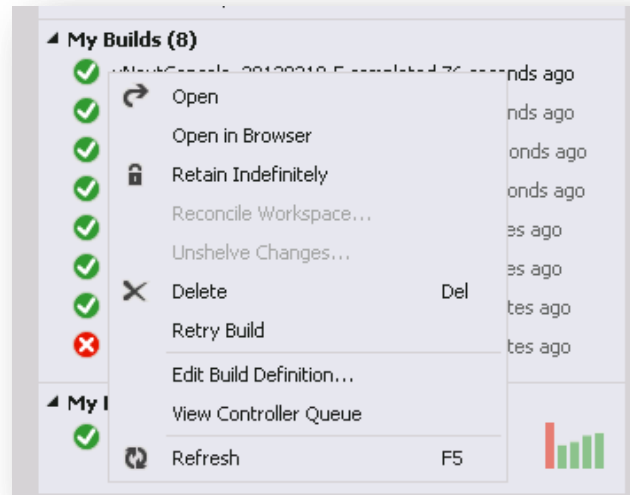


Figure 31 – Context menu - Completed builds



### Context menu – Completed builds

In this menu there is an item called “**Retain Indefinitely**” that when clicked will ensure your build is not deleted (as otherwise it would be according to the retention policy in the build definition).

There is another interesting option in this menu, “**Retry Build**”; this will run a build with the same parameters as it ran before.

4. Queue some more builds and test the right click over a running build.

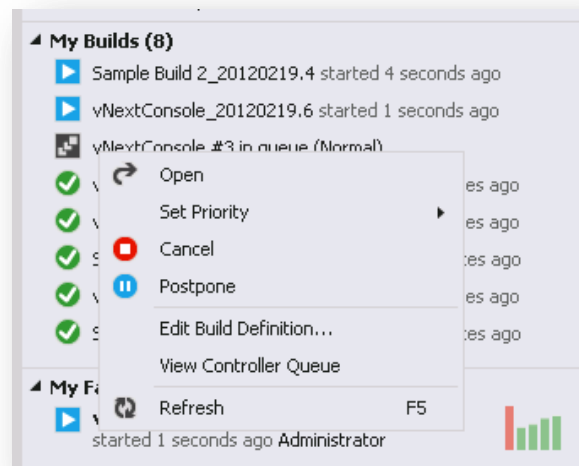


Figure 32 – Context menu - Queued builds.



### Context menu – Queued builds

It's possible to set the build priority easily selecting the menu “**Set priority**” and choosing some priority item. Also is possible to “**Open**” the build request to see more details about it, “**Cancel**” or even “**Postpone**” a build.

- Using the context menu of some completed build, select the menu “**Open**”

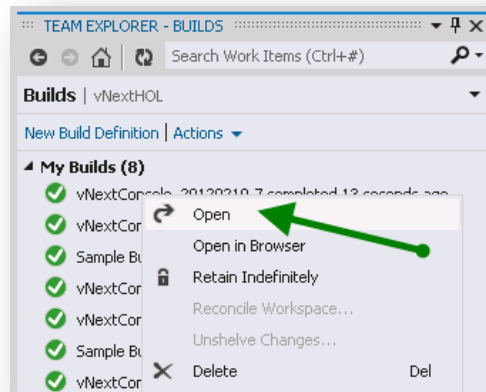


Figure 33 – Context menu - Open Item

- Note the new **Build Request** window for a succeeded build showing the Actions menu with Retry and Queue Details

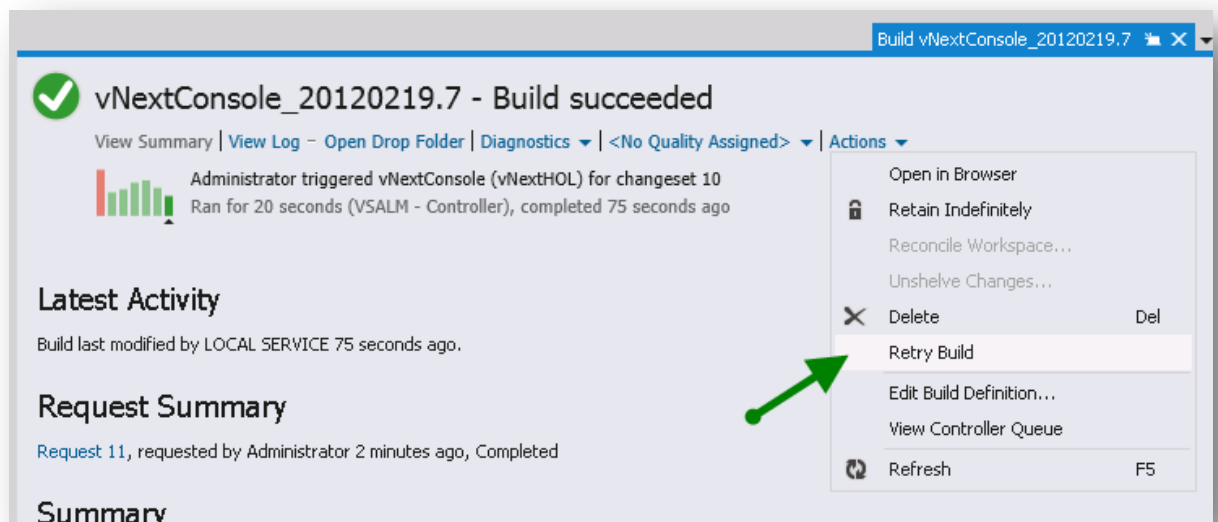


Figure 34 – Build Request window

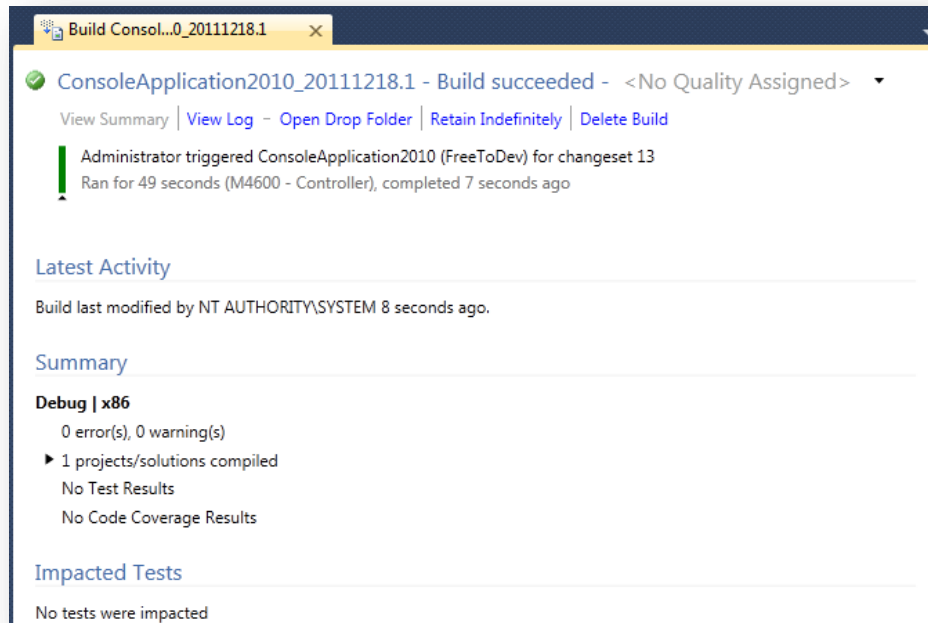


Figure 35 – Visual Studio 2010 build window for comparison

- Click the link “**Retry**” on the Actions menu and the build will restart

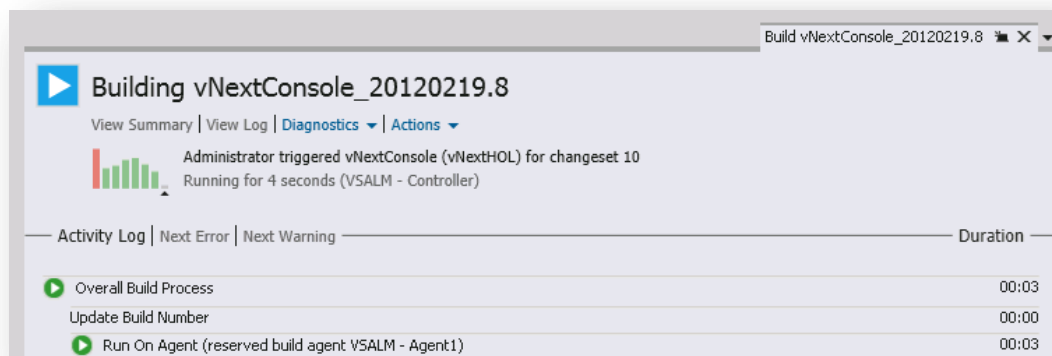


Figure 36 – Build Request after retrying



### Retry

This option uses the same build request with its parameters, providing a way to queue a new build with just one click. The Build Request window also offers a list of all builds started with this request in the Build Summary area.

- Click the Diagnostics link and select view Logs to see the log files generated by the build



## Exercise 4: Build Summary Extensibility



### OBJECTIVE

In this Exercise, we will add a build summary note to the Team Build using the new WriteCustomSummaryInformation activity.

1. Open the BuildProcessTemplates folder in Source Control Explorer and open the DefaultTemplate.xaml file.

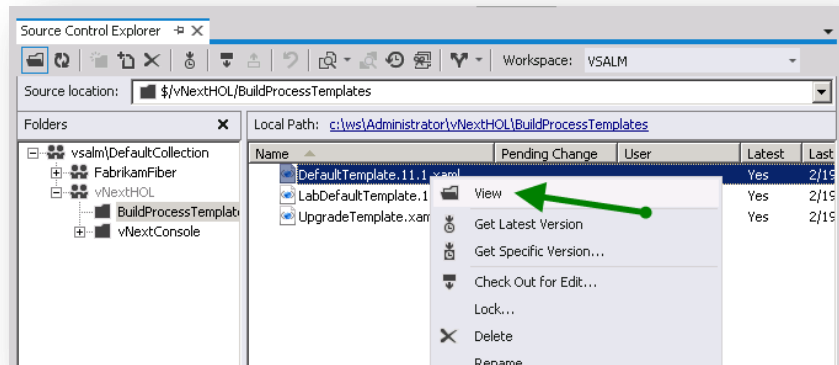


Figure 37 – View the template our builds use

2. Navigate to the end of the Run On Agent sequence and drag the WriteCustomSummaryInformation activity onto the build template. Set the Properties for the activity to those shown below. Note that we are using simple values for illustrative purposes only. In practice you could use calculated values to provide more value.

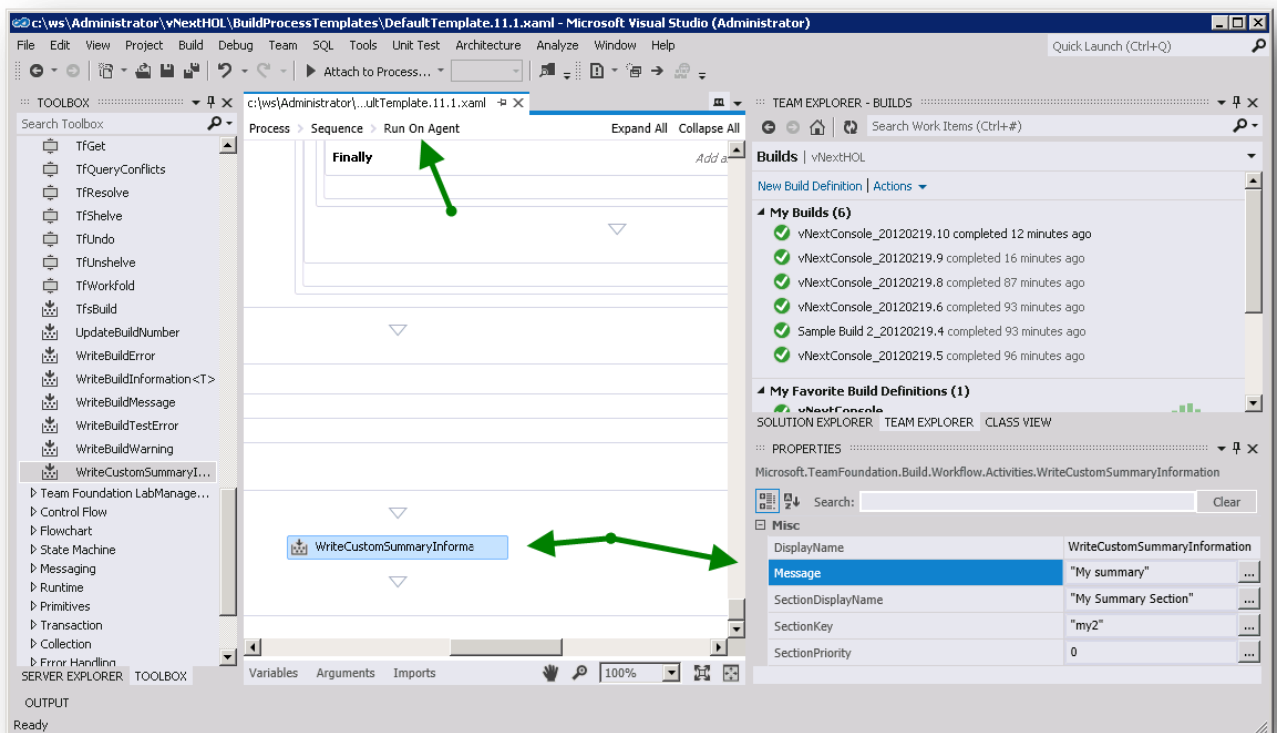


Figure 38 – Add and configure the WriteCustomSummaryInformation activity

3. Save the build template and check it in.

4. Run the VNextConsole build.

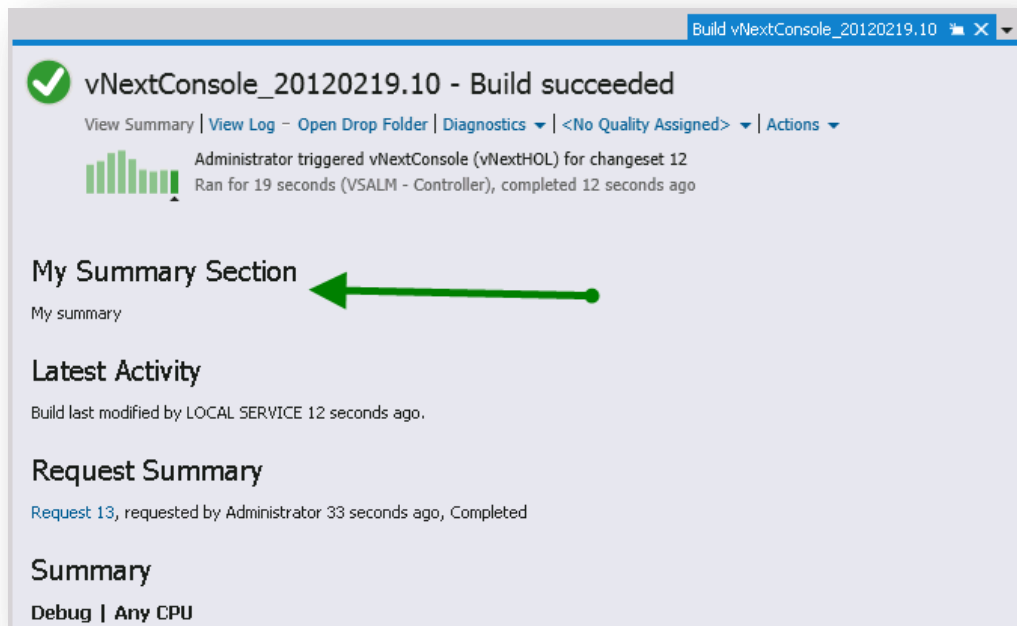


Figure 39 – Custom summary section displayed



### REVIEW

We have added and configured the new WriteCustomSummaryInformation activity to our build. This allows us to define and display custom information in the build reports.



LAB  
COMPLETE