



HEDGEHOG DEVELOPMENT

Getting Started with Team Development for Sitecore[®]

CONTENTS

1 Introduction.....	4
2 Why use TDS.....	5
3 Installing TDS.....	7
4 Using TDS with an existing Sitecore project.....	10
4.1 Assumptions.....	10
4.2 Steps.....	11
5 Using TDS to bring a new developer onto a project	16
5.1 Assumptions.....	16
5.2 Steps.....	16
5.2.1 Using project configurations.....	17
5.2.2 Using .user file.....	18
6 Using TDS to pull item changes from Sitecore into Source Code Control.....	23
6.1 Assumptions.....	23
6.2 Steps.....	23
7 Using TDS to push items from Source Code Control into Sitecore.....	26
7.1 Assumptions.....	26
7.2 Steps.....	26
8 Setting up packaging of TDS items.....	29
8.1 Assumptions.....	29
8.2 Steps.....	29
9 Excluding TDS items from specific build configurations.....	32
9.1 Assumptions.....	32
9.2 Steps.....	32



9.2.1 “Recommended” Steps.....	32
9.2.2 “Alternate” Steps.....	33
10 Frequently Asked Questions.....	35

Chapter 1

Introduction

The following document describes the best way to start using Team Development for Sitecore (TDS). The document includes steps to perform many different actions that developers will need to do to get started with TDS.

The document contains the following chapters:

- Chapter 1 – Introduction
- Chapter 2 – Why use TDS?
- Chapter 3 – Installing TDS
- Chapter 4 – Using TDS with an existing Sitecore project
- Chapter 5 – Using TDS to bring a new developer onto an existing project
- Chapter 6 – Using TDS to pull item changes from Sitecore into Source Code Control
- Chapter 7 – Using TDS to push items from Source Code Control into Sitecore
- Chapter 8 – Creating Update Packages
- Chapter 9 – Excluding TDS items from specific build configurations
- Chapter 10 – Additional Resources

Chapter 2

Why use TDS?

Team Development for Sitecore (TDS) gives you the ability to keep track of all the Sitecore items in your project, by bringing them into a Microsoft compliant Source Code Control system. Using TDS, you will improve the control and overall performance of the team and your project.

Having everything in a Source Code Management System gives you one master repository for the project. The Source Code Management System becomes the master and central integration point of all the code and Sitecore items for a team. Even if you are a team of 1, using a Source Code Control system is the smart way to work, and provides many benefits, including:

- Complete accountability and audit trail for all changes to all Sitecore items and .NET code in a project.
- All of your Sitecore items and code are stored in a single repository, meaning backups are easy to do, making your work safe.
- Developers can work remotely or disconnected from the network and then sync and merge changes at a later date to the source code control system.
- You can easily compare versions of items and code.
- You can rollback to prior versions if there is a problem.
- You can easily merge changes.
- Branching and other advanced source code management techniques can be leveraged.
- Continuous integration and automated builds

TDS helps new developers on a team get up to speed quickly by creating a new, clean Sitecore environment, getting latest from the Source Code Management System and then deploying to Sitecore. They also have the ability to easily see what other team members have been doing by reviewing the check-in comments.

Without TDS each developer is dependent on coordinating and then packaging up changes with the entire team. We have even seen partners go so far as design their projects so that developers each work in separate areas of a project to reduce possibilities for collision. This is not the optimal way to work, to say the least.

We truly believe that TDS will change the way the Sitecore projects are developed, and that TDS will become the standard Best Practices methodology for Sitecore development.

Chapter 3

Installing TDS

3.1 Assumptions

1. All running applications have been shut down prior to beginning the install process.
2. You have downloaded the zip file from the www.hhogdev.com that contains 2 versions of TDS to match up with the version of Visual Studio that you are running.
3. You have matched the version of Visual Studio with the version of TDS.

Visual Studio 2008	HedgehogDevelopmentTDS.msi
Visual Studio 2010	HedgehogDevelopmentTDS_2010.msi

NOTE

If you have started a project during a trial period of TDS and wish to install a purchased license,

- For 32 bit version just run the TDSLicenseInstaller.exe located here: C:\Program Files (x86)\Hedgehog Development\Team Development for Sitecore (VS2010). This will add your new license of TDS while not deleting your existing Sitecore development
- If you are operating on the 64 bit version you must uninstall and reinstall TDS, however; this will not delete your TDS project.



NOTE

By default TDS wants to install the project templates (TDSProject.zip) in the My Documents folder on your C: drive. Located:

C:\Documents and Settings\UserName\My Documents\Visual Studio 2008\Templates\ProjectTemplates

If you have installed the My Documents folder in a different directory, the Visual Studio path does not get updated.

You should make sure that the Visual Studio User Project Template Location in Visual Studio is pointing to the desired location. You can do so by:

1. In Visual Studio go to Tools>>Options
2. Under the Projects and Solutions highlight the General settings
3. Locate the Visual Studio user project template location: verify that this path is correct. It should be: Path to directory\Visual Studio 2008\Templates\ProjectTemplates
4. Click OK when you are done and now you should see the TDS Project Template when you try to create a new project.

If you are running Visual Studio in Safe Mode, none of your add-ins will load

3.2 Steps

The installation wizard steps are as follows:

1. Click next to begin the process of installing TDS.
2. Accept License Agreement, then Click Next.



Figure 1 TDS License Agreement

3. Insert the provided trial or purchased license key. Make sure you use the correct company name that was provided with the license key.

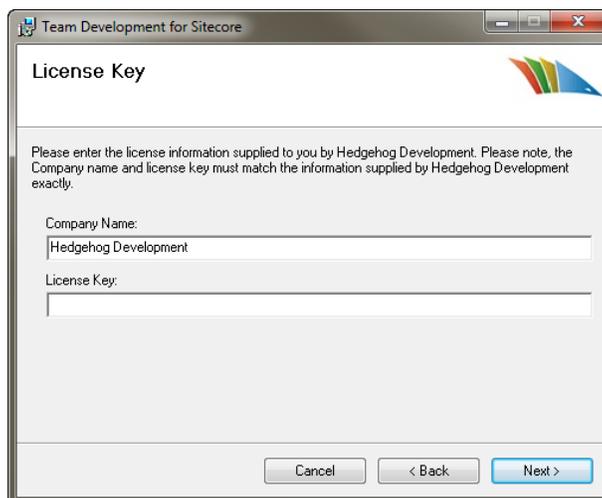


Figure 2 Insert your License Key and Company Name

4. Select the path to the location where you want TDS installed.
5. Confirm you wish to install TDS, by clicking Next.

Chapter 4

Starting to use TDS with an existing Sitecore Project

Syncing TDS with an existing Sitecore project is quick and simple. There is no need to start a Sitecore project over, or laboriously move existing code just because you've downloaded a new plug in. TDS allows you to seamlessly integrate an existing Sitecore project with TDS, through the Get Items option.

4.1 Assumptions

To perform this operation in TDS there are two mandatory conditions that must be met. They are as follows:

1. TDS has been installed.
2. There is an existing Sitecore installation with a project currently under development. The team wishes to use TDS to gain more control over their Sitecore Items.
3. (Optional) There is an existing solution with custom application code in a Web Application Project.

NOTE

If you have started a project during a trial period of TDS and wish to install a purchased license,

- For 32 bit version just run the TDSLicenseInstaller.exe located here: C:\Program Files (x86)\Hedgehog Development\Team Development for Sitecore (VS2010). This will add your new license of TDS while not deleting your existing Sitecore development
- If you are operating on the 64 bit version you must uninstall and reinstall TDS, however; this will not delete your TDS project.

4.2 Steps

To start an existing Sitecore project on TDS is as extremely simple. To open an existing project just follow the steps below:

1. In Visual Studio open the “New Project” dialog. You may want to add this new project to your existing solution, or create it stand alone.
2. Choose Project type: “TDS Project” and choose Template: “TDS Project” then specify the desired name, location and Source Code Control (SCC) options (if applicable) [see Figure 3]

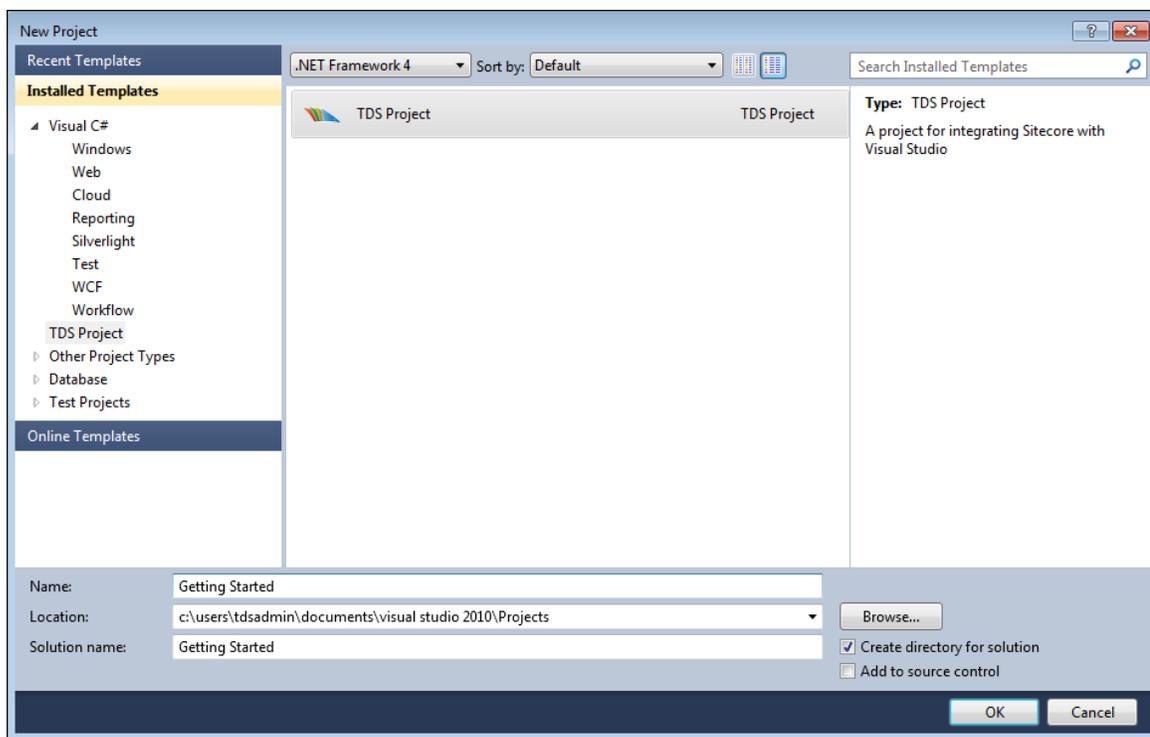


Figure 3 New project dialog box, which you can create a TDS project

NOTE

By default TDS wants to install the project templates (TDSProject.zip) in the My Documents folder on your C: drive. Located:



C:\Documents and Settings\UserName\My Documents\Visual Studio 2008\Templates\ProjectTemplates

If you have installed the My Documents folder in a different directory, the Visual Studio path does not get updated.

You should make sure that the Visual Studio User Project Template Location in Visual Studio is pointing to the desired location. You can do so by:

5. In Visual Studio go to Tools>>Options
6. Under the Projects and Solutions highlight the General settings
7. Locate the Visual Studio user project template location: verify that this path is correct. It should be: Path to
directory\Visual Studio
2008\Templates\ProjectTemplates
8. Click OK when you are done and now you should see the TDS Project Template when you try to create a new project.

If you are running Visual Studio in Safe Mode, none of your add-ins will load

3. Right click on the project node in the solution explorer and choose properties.
4. The following step is optional. It only applies to developers that have a source web project. If you do not have a source web project, skip this step. In the “General” tab, choose the source web project from the dropdown
5. In the Build Tab fill in the following:
 - a. Sitecore Web URL. This is the URL to the root of **your Sitecore development environment.**
 - b. Sitecore Deploy Folder. This is the UNC path to the web root of the file system you are connecting TDS to.

- c. Check the “Install Sitecore Connector” checkbox [See Figure 4]
6. Save the changes to the project.

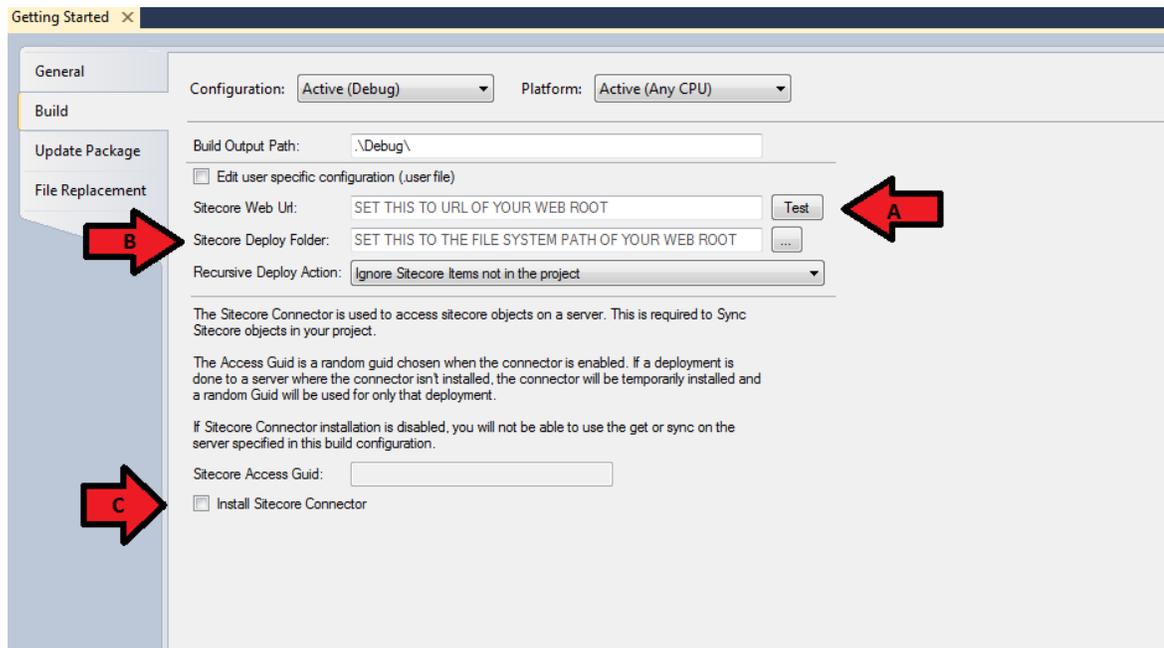


Figure 4 TDS project build tab, before a project has been started

7. Right click on the project node in the solution explorer and select “Install Sitecore Connector”.
8. Right Click on the project node in the solution explorer and select “Get Sitecore Items”. This will allow you to begin building the tree of Sitecore items you wish to control.

NOTE

You should only put the nodes (and their parents) that you wish to control, into TDS. I.e. avoid putting Sitecore system nodes into TDS if there aren't changes to them.



9. Start by right clicking on the “Templates” item in the solution explorer and select “Get Sitecore Items”.
10. Right click on the “User Defined” item in the solution explorer and select “Get Sitecore Items”.
11. In the dialog box use the Shift or Control to select the items you wish.
12. Right click and select “Get All Children” to select all the children.

NOTE

Choosing “Get All Children” on the items sets the “Child Item Synchronization” property to “Keep all Children Synchronized”.

This changes the behavior of the item as follows:

- TDS will automatically look for new items under the “Keep all Children Synchronized” item during synchronization.
- If the Recursive Deploy Action is changed from its default setting of “Ignore Sitecore Items not in the project”. During deployment, any item in Sitecore that is not in the project will be removed from Sitecore. These items may optionally be sent to the Sitecore Recycle bin.

13. You may go back to other Sitecore folders and get items, as well as check the TDS project into SCC.

NOTE

Some items may need to be added to Sitecore during a deployment only if they aren’t already there. You can mark an item as “Deploy Once” by right clicking the item in the solution explorer, choosing “Properties” and setting the “Item Deployment” property to “Deploy Once”. This will cause TDS to create the item if it isn’t in Sitecore. If it exists, the items properties aren’t changed. This feature is used to help build a skeleton content tree during deployment.



An informative troubleshooting video about verifying access to your deployment environments can be found at:

<http://www.youtube.com/watch?v=ufJfUbOPCj0&feature=youtu.be&hd=1>

Chapter 5

Using TDS to bring a new developer onto an existing project

It is inevitable, at some point you have to bring a new developer onto a project. Before TDS, this process took too long, and involved more work than it should. Now, with TDS, bringing a new developer onto a project has never been simpler. There are two different ways you can complete this action, both of which work equally well. The ease at which you can bring a developer onto a project will save you invaluable time and effort; while at the same time making it easier on the new developer to get acclimated with the code of a new project.

5.1 Assumptions

The following assumptions must be met before you can bring on a new developer. They are as follows:

1. There is an existing solution with a TDS project and a web application project available on the developer's local Sitecore development environment.
2. There is an existing blank development Sitecore installation available for the developer to use and it hasn't been used by any other developer.
3. The Sitecore development installation can be local, on another computer, or on a Virtual Machine (VM).

5.2 Steps

There are two different configurations in which one can bring a new developer onto a project. First you can do it using project configurations or you can do it using .user file. No method is recommended over the other, the method used is strictly up to the lead developer's preference.

5.2.1 Using project configurations

Using project configurations to bring a new developer onto a project is quick and simple. The process is transparent, which makes it easy for new developers to quickly catch up on all the existing code of the project. To bring a new developer onto a project just follow these steps:

1. Open the existing TDS project.
2. Right click on the solution in the solution explorer and select “Configuration Manager...”
3. On the row with the TDS project, choose “<New...>” for the “Configuration” dropdown.
4. Choose the typical developer configuration in the “Copy settings from” dropdown (usually Debug).
5. Check “Create new solution configuration” then click OK.
[See Figure 5]

NOTE

When a solution is checked into SCC, the currently selected configuration is preserved, and when other developers get the solution they begin using the selected configuration. This can cause confusion if a developer deploys their changes to another developer’s server. Therefore if all developers are using the same solution configuration name, but have custom settings locally, there is less chance of problems.

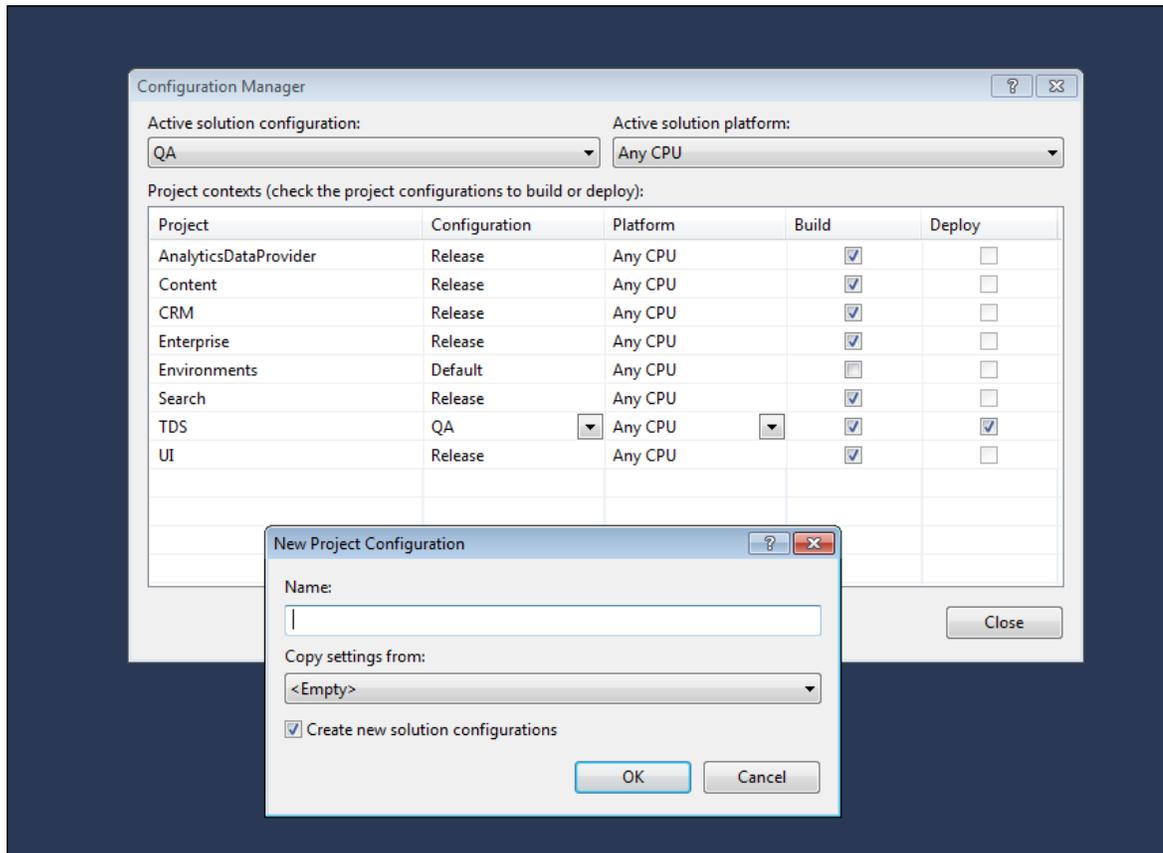


Figure 5 TDS Configuration Manager allows you to specify the fields you wish to create and where you would like to duplicate the settings from

6. Verify the new configuration is selected in the “Solution Configuration” on the standard toolbar.
7. Right click on the TDS project in the Solution Explorer and choose “Properties”
8. Click on the “Build” tab and enter the url of the blank Sitecore installation for “Sitecore Web url” then enter the file system path to the root of the Sitecore web site for “Sitecore Deploy Folder”. If this is a network share, verify your ability to write and modify files in the share.

9. Click “Install Sitecore Connector” to enable installing the connector.

IMPORTANT

This setting should not be used in a production environment.

10. Click “Save” or close the project properties window.
11. Right click on the TDS project in the solution explorer and choose “Deploy”

NOTE

The time this takes depends on the size of the project.

12. When the deployment is complete, the Visual Studio status bar will show “Deploy Succeeded”.

5.2.2 Using .user file

Using .user files to bring new developers onto a project is just as quick and straight forward as using configuration files. However, .user files are easier to work with, since all developers can use the same configurations e.g. “debug”. To bring a developer onto a project using .user files all you need to do is:

1. Open the existing TDS project.
2. Right click on TDS project in solution explorer and choose “Edit .user File”
3. This will open the existing .user file or create a new one.

IMPORTANT

This .user file should ***not*** be put under source code control or used on any other developer’s machine.

4. Add the following lines to the .user file:



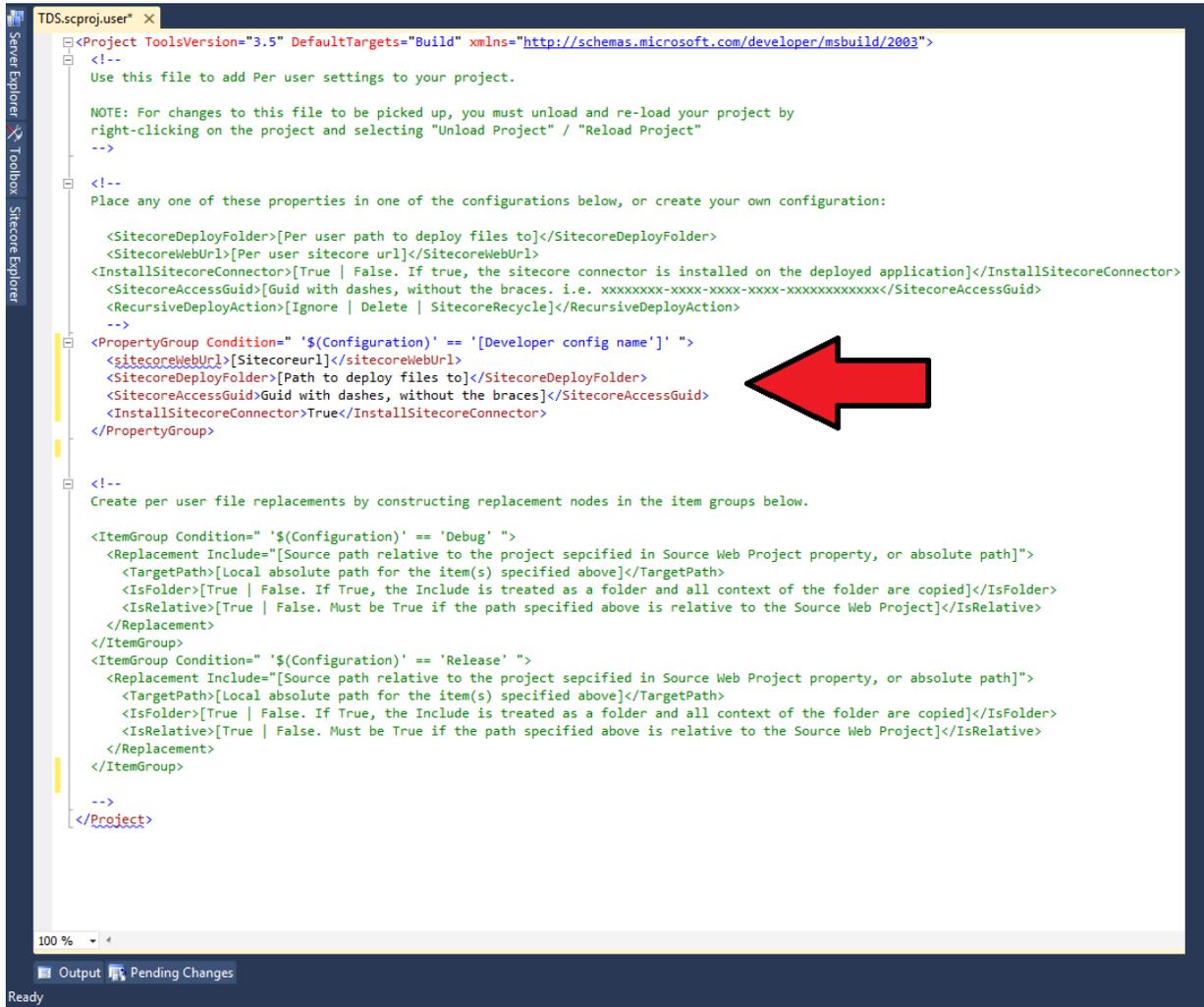
```
<PropertyGroup Condition=" '$(Configuration)' ==
'[Developer config name]' ">
  <SitecoreWebUrl>[Sitecore
url]</SitecoreWebUrl>
  <SitecoreDeployFolder>[Path to deploy
files to]</SitecoreDeployFolder>
  <SitecoreAccessGuid>[Guid with dashes,
without the braces. i.e. xxxxxxxx-xxxx-xxxx-xxxx-
xxxxxxxxxxxx]</SitecoreAccessGuid>

  <InstallSitecoreConnector>True</InstallSitecore
Connector>
</PropertyGroup>
```

NOTE

These must be placed within the <Project> node.

5. Replace the text within brackets with the proper values. For an example see Figure 6.



```

TDS.sproj.user x
<Project ToolsVersion="3.5" DefaultTargets="Build" xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
  <!--
  Use this file to add Per user settings to your project.

  NOTE: For changes to this file to be picked up, you must unload and re-load your project by
  right-clicking on the project and selecting "Unload Project" / "Reload Project"
  -->
  <!--
  Place any one of these properties in one of the configurations below, or create your own configuration:

  <SitecoreDeployFolder>[Per user path to deploy files to]</SitecoreDeployFolder>
  <SitecoreWebUrl>[Per user sitecore url]</SitecoreWebUrl>
  <InstallSitecoreConnector>[True | False. If true, the sitecore connector is installed on the deployed application]</InstallSitecoreConnector>
  <SitecoreAccessGuid>[Guid with dashes, without the braces. i.e. xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx]</SitecoreAccessGuid>
  <RecursiveDeployAction>[Ignore | Delete | SitecoreRecycle]</RecursiveDeployAction>
  -->
  <PropertyGroup Condition=" '$(Configuration)' == '[Developer config name]' " >
    <SitecoreWebUrl>[Sitecoreurl]</sitecoreWebUrl>
    <SitecoreDeployFolder>[Path to deploy files to]</SitecoreDeployFolder>
    <SitecoreAccessGuid>Guid with dashes, without the braces]</SitecoreAccessGuid>
    <InstallSitecoreConnector>True</InstallSitecoreConnector>
  </PropertyGroup>

  <!--
  Create per user file replacements by constructing replacement nodes in the item groups below.

  <ItemGroup Condition=" '$(Configuration)' == 'Debug' " >
    <Replacement Include="[Source path relative to the project sepcified in Source Web Project property, or absolute path]">
      <TargetPath>[Local absolute path for the item(s) specified above]</TargetPath>
      <IsFolder>[True | False. If True, the Include is treated as a folder and all context of the folder are copied]</IsFolder>
      <IsRelative>[True | False. Must be True if the path specified above is relative to the Source Web Project]</IsRelative>
    </Replacement>
  </ItemGroup>
  <ItemGroup Condition=" '$(Configuration)' == 'Release' " >
    <Replacement Include="[Source path relative to the project sepcified in Source Web Project property, or absolute path]">
      <TargetPath>[Local absolute path for the item(s) specified above]</TargetPath>
      <IsFolder>[True | False. If True, the Include is treated as a folder and all context of the folder are copied]</IsFolder>
      <IsRelative>[True | False. Must be True if the path specified above is relative to the Source Web Project]</IsRelative>
    </Replacement>
  </ItemGroup>
  -->
</Project>
  
```

Figure 6 An example of a .user file with the lines that must be added to bring a new developer onto a project

6. Save and Close the .user file.
7. Right click on the TDS project in the solution explorer and choose “Unload Project”.
8. Right click on the TDS project in the solution explorer and choose “Reload Project”.



NOTE

After any change to the .user file, the project must be re-uploaded and reloaded.

9. Right click on the TDS project in the solution explorer and choose “Properties”.
10. Go to the build tab for your configuration and verify the properties are correct. These values cannot be edited in the properties window.

NOTE

It is a good idea to add additional blank configurations to the .user file to prevent developers from accidentally deploying to the wrong place. The following is an example of this:

```
<PropertyGroup Condition="
'$ (Configuration)' == 'QA' ">
  <SitecoreWebUrl></SitecoreWebUrl>
  <SitecoreDeployFolder></SitecoreDeployFolder>
  <SitecoreAccessGuid></SitecoreAccessGuid>
</PropertyGroup>
```

IT should take additional measures, like limiting access to the QA shares, so developers can not deploy to QA unless they have permission to do so.

Chapter 6

Using TDS to pull item changes from Sitecore into Source Code Control

Your Source Code Control (SCC) is an important aspect of any development project. The ability to move your Sitecore items into Source Control is essential to obtaining as much control over your project, as possible. In the past you had to package any changes to Sitecore items and deploy them to make desired changes. Now with TDS you can bring your Sitecore items directly into your Source Control. This gives you the ability to treat all your Sitecore items as regular code, which can be easily altered to do anything you want. All of this allows you to have an authoritative source of code for your project; we like to call it “the single source of truth”.

6.1 Assumptions

The two assumptions that must be met before you can pull item changes from Sitecore into SCC are:

1. You have gone through one of the following scenarios described above and connected TDS to your Sitecore environment
 - Scenario: Starting to use TDS with an existing Sitecore Project
 - Scenario: Using TDS to bring a new Sitecore developer onto a project
2. There are changes to items in Sitecore

6.2 Steps

There are eight steps involved with pulling item changes from Sitecore into SCC. These steps enable you to treat all your Sitecore items like regular code, giving you the greatest control possible over your Sitecore development. The steps are as follows:

1. In the Solution Explorer, right click on the Sitecore Project node or any Sitecore item under the project and choose “Sync with Sitecore”.

The sync window will then open and show the current sync status. [See Figure 7]

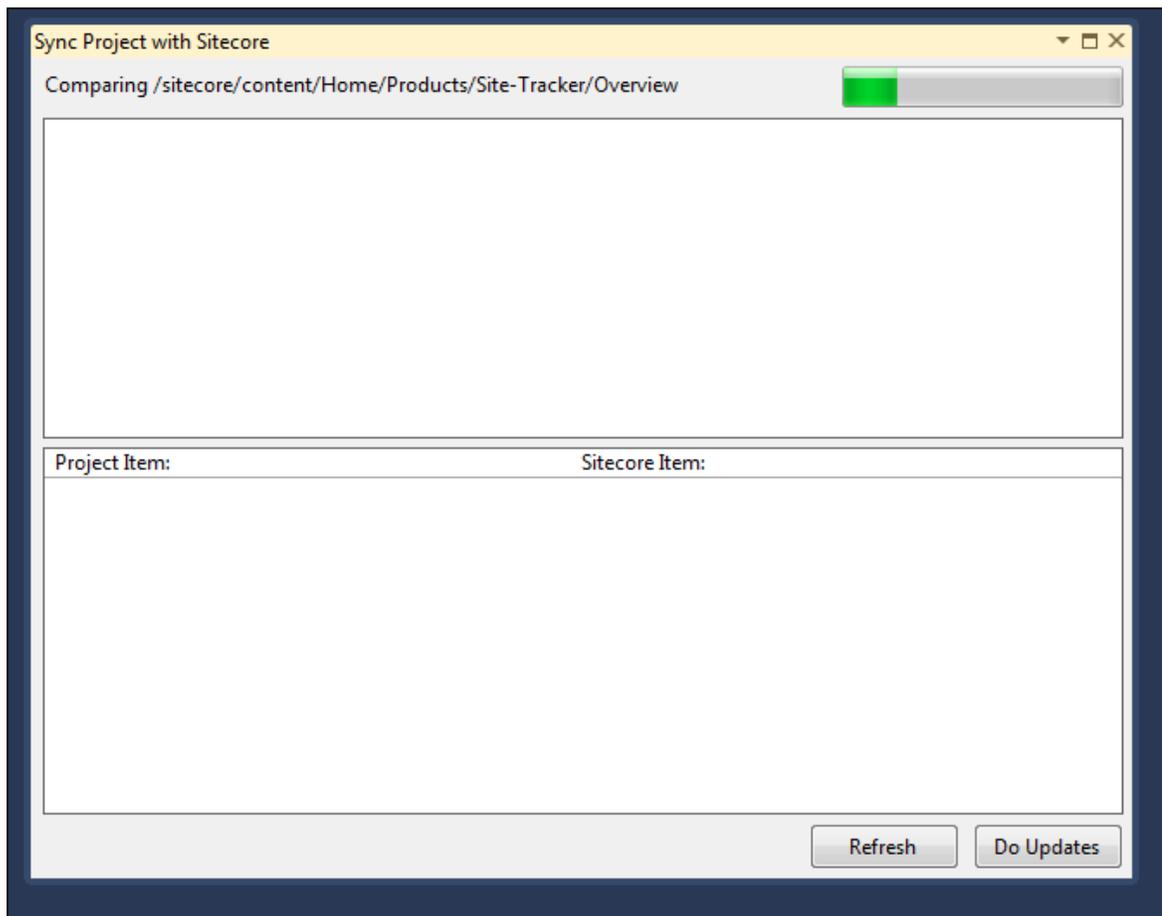


Figure 7 Loading of the Sync project with Sitecore feature

2. When sync is done, any nodes with changes will be expanded in the tree.
3. Clicking on a node in the tree will show a dropdown that allows the user to choose a valid sync action. To see an example of this, choose a node not shown as "Same".
4. Choose the action "Update Project"
5. Set the sync action for other nodes you wish to update.
6. Click "Do Updates"



7. Check the items into SCC.

Chapter 7

Using TDS to push item changes from Source Code Control into Sitecore

When pushing your changes into Sitecore you want to make sure that the code, from all your developers, is synced so you don't have to go through and do a complete debug of the code. You do not want new code to get into Sitecore that you did not originally want or know about. TDS solves this problem for you; it puts together all the different code from your developers and builds them together. This saves you enormous amount of time and avoids many of the headaches that can be involved with syncing.

7.1 Assumptions

There are two assumptions that must be met to push item changes from SCC into Sitecore. They are as follows:

1. You have gone through one of the following scenarios described above and connected TDS to your Sitecore environment
 - Scenario: Starting to use TDS with an existing Sitecore Project
 - Scenario: Using TDS to bring a new Sitecore developer onto a project
2. There are changes to the Sitecore schema in Source Code Control (SCC).

7.2 Steps

Pushing item changes from SCC to Sitecore is simple and transparent. The process relieves much of the stress originally on developers to package all their updates.

There are nine steps included in pushing item changes from SCC into Sitecore.

They are:

1. Get the latest version of the TDS project from SCC.
2. In the Solution Explorer, right click on the Sitecore Project node or any Sitecore item under the project and choose "Sync with Sitecore".
3. The sync window will open and show the current sync status.



4. When sync is done, any nodes with changes will be expanded in the tree.
5. Locate the changed nodes in the tree. These nodes will be shown as:
 - Different
 - Project Item Newer
 - Missing in Sitecore
6. You may examine the differences by clicking on any different node. The differences for the selected node will be shown in the lower half of the Sync window.
7. Choose any nodes you wish to push to Sitecore. These can be multi-selected by using <Control> or <Shift>
8. Choose “Update Sitecore” or “Add to Sitecore” in the action dropdown. [See Figure 8]
9. Click “Do Updates” when all changes are selected.

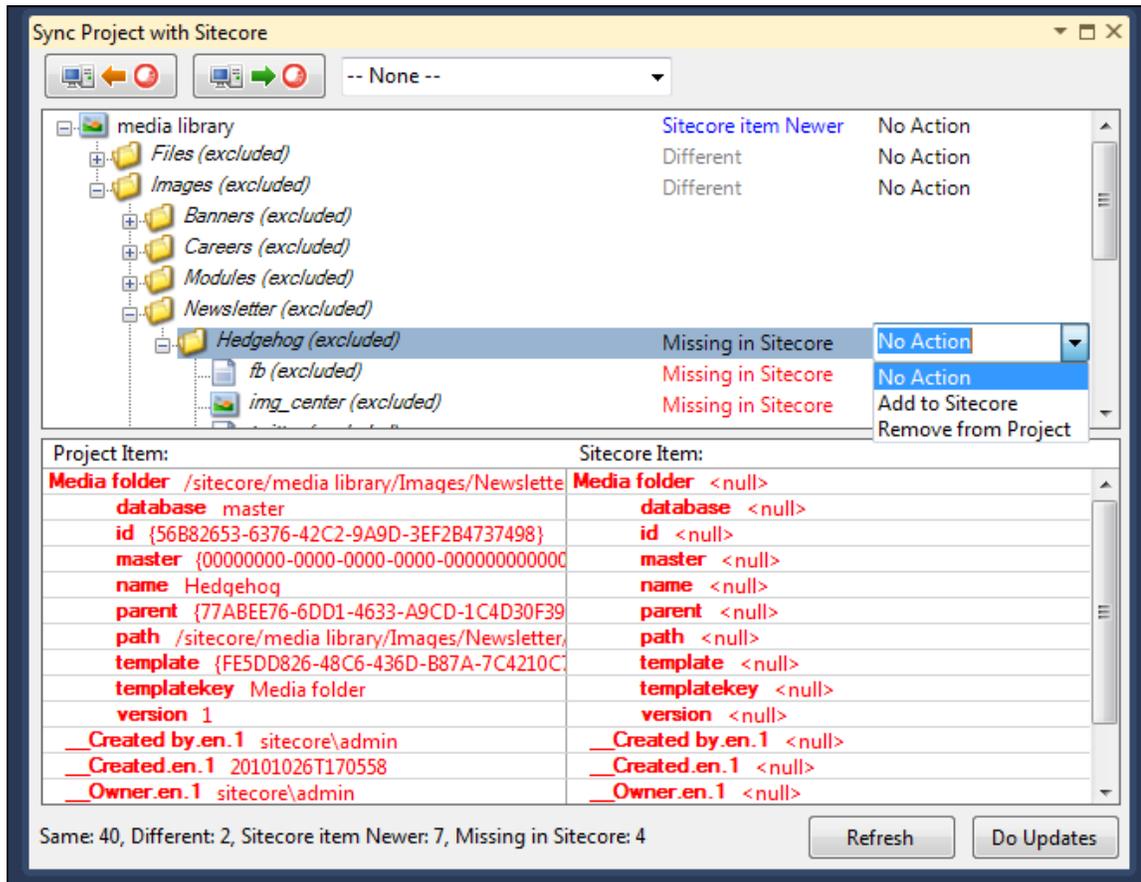


Figure 8 Sync Project with Sitecore window shows you items that may be different or not current in Sitecore

Chapter 8

Creating Update Packages

Packaging your code is what makes developers want to tear the hair out of their heads. Making sure all your code syncs properly is the biggest headache involved with deployment. Doing manual builds to then be pushed into QA or Production is extremely tedious and time consuming. With TDS the packaging is now automatic, you no longer need to spend unnecessary time syncing code and debugging packages. TDS dramatically streamlined the workflow process in development.

8.1 Assumptions

There are two assumptions involved with setting up packaging of TDS items. They are:

1. You have gone through one of the following scenarios described above and connected TDS to your Sitecore environment
 - Scenario: Starting to use TDS with an existing Sitecore Project
 - Scenario: Using TDS to bring a new Sitecore developer onto a project
2. You have Sitecore 6.1+ installed.

8.2 Steps

The automatic packaging process, done by TDS, makes manually syncing all your changes a thing of the past. There are nine steps that need to be followed to set up packaging of TDS items. They are:

1. Get the latest version of the TDS project from SCC.
2. Choose the Solution Configuration you wish to generate an update package for.
3. In the Solution Explorer, right click on the TDS project and select “Properties”
4. Select the “Update Package” tab on the left hand side of the property page.



5. Enable the check box “Generate package during build”
6. Fill in “Package Author”, “Package Publisher”, “Package Version” and “Package Readme”. These text fields are incorporated into the package when it is built.
7. If you want you can check “Generate separate code and item packages”. [See Figure 9]

NOTE

You can perform this action if you deem it desirable to create separate code and item packages in a multi-server environment, where the code runs on a “Content delivery server” and editors modify the content on a separate “Content management server”.

8. Set the “Sitecore Assembly Path”. This is a path to a folder that contains the Sitecore assemblies used to create the package. The required assemblies are: **Sitecore.Kernel.Dll**, **Sitecore.Logging.Dll**, **Sitecore.Update.Dll** and **Sitecore.Zip.Dll**.

IMPORTANT

The assemblies must be from the version of Sitecore you are deploying to. Packaging will not run without the assemblies.

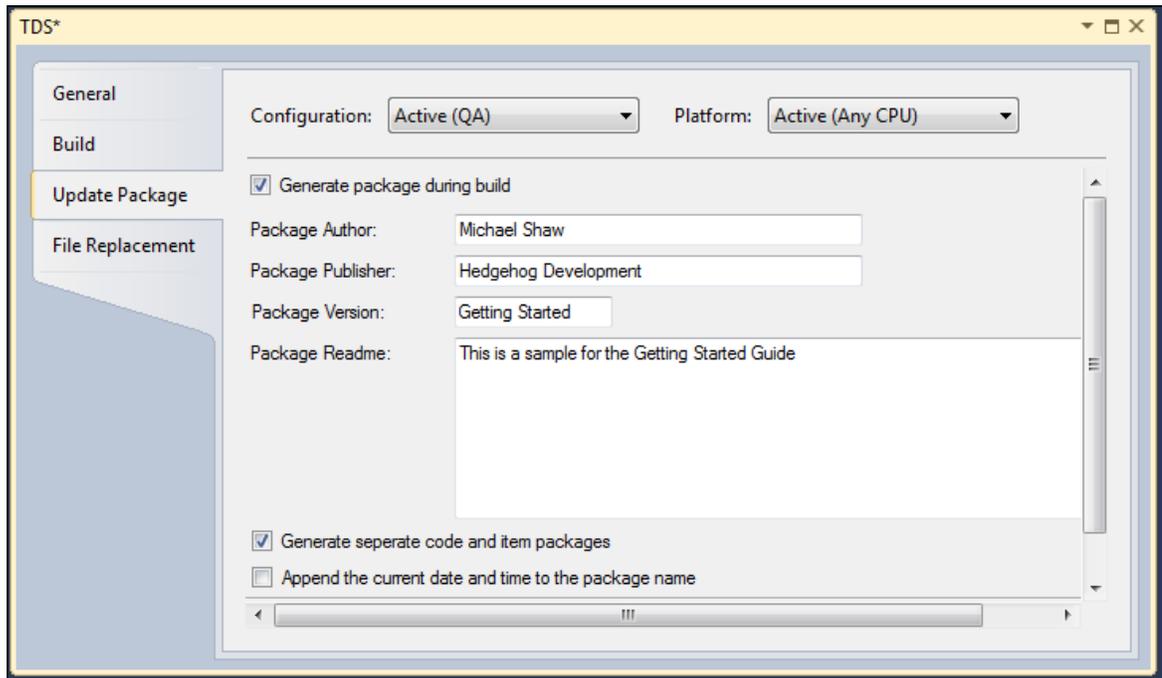


Figure 9 TDS Update Package tab allows you to easily create and customize packages

9. Build the application.

NOTE

The .updatepackage is generated in the [Project Root]\Package_[Configuration]

The .updatepackages are installed using Sitecore update installation wizard deployment tool included located by default at /sitecore/admin/UpdateInstallationWizard.aspx

Chapter 9

Excluding TDS items from specific build configurations

There are many times where you have test content, content that isn't complete, or just content you don't want pushed to QA or Production. It would be a problem if all this content automatically got pushed, even if that wasn't what you intended to happen. It is for this reason the ability to exclude items from a build is so necessary to Sitecore development.

9.1 Assumptions

There are two assumptions. They are:

1. You have gone through one of the following scenarios described above and connected TDS to your Sitecore environment
 - Scenario: Starting to use TDS with an existing Sitecore Project
 - Scenario: Using TDS to bring a new Sitecore developer onto a project
2. You have Sitecore 6.1+ installed.

9.2 Steps

There are two different sets of steps one can follow to exclude TDS items from specific build configurations, both of which are extremely quick and easy to follow.

9.2.1 "Recommended" steps

1. Get the latest version of the TDS project from SCC.
2. Expand TDS items in the solution explorer and select a TDS item that shouldn't be included in one or more project configurations.
3. Press 'F4' or right click on the item and select 'Properties'
4. In the property window, click on the "Exclude Item From" property.

NOTE

Excluding an item from a configuration will exclude all children of that item.

5. Press the dropdown button in the right-most portion of the property value area for “Exclude Item From”
6. Click the check box next to the configuration name you wish to exclude the item from. [See Figure 10]

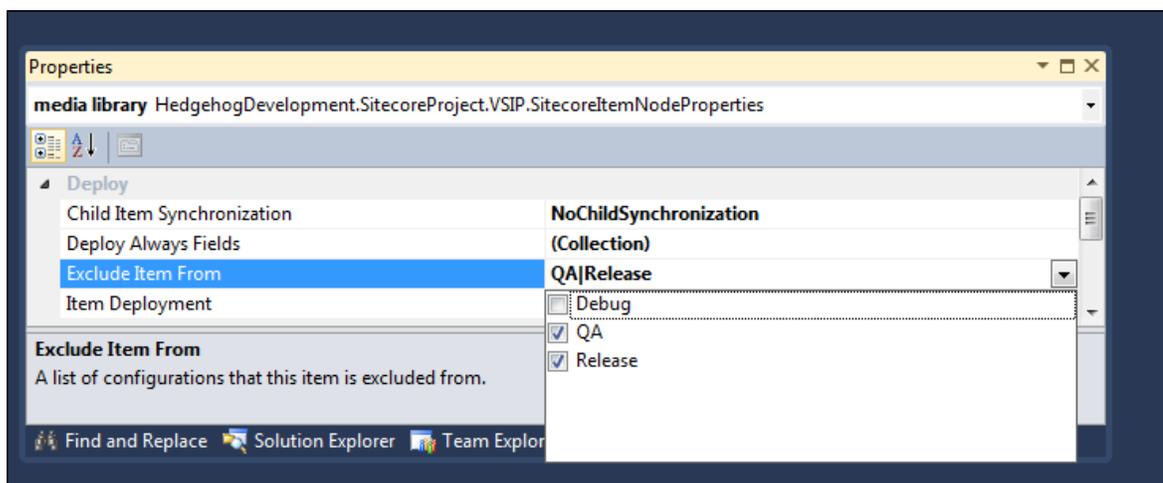


Figure 10 TDS Exclude Items functions allows you to Exclude certain items from areas of the workflow you do not wish for them to enter. This gives you greater control over your entire development.

9.2.2 Alternate Steps

1. Get the latest version of the TDS project from SCC.
2. Expand TDS items in the solution explorer and select a TDS item that shouldn't be included in one or more project configurations.
3. Right-click on the item and choose “Deployment Property Manager” (DPM)
4. The DPM shows the Sitecore item you selected along with all items under the selected item. In the “Current Config” column, there will be two

values: “Include” or “Exclude”.

NOTE

Including an item in a configuration will include all parents of that item

5. Clicking on a value under “Current Config” will allow you to change that value.
6. Selecting a different configuration in the “Solutions Configuration” dropdown will change the Include/Exclude values under “Current Config” to match the selected configuration. [See Figure 11]

NOTE

A solution configuration may use a project configuration with a name that is different than the solution configuration. Please use the “Configuration Manager” window to manage this.

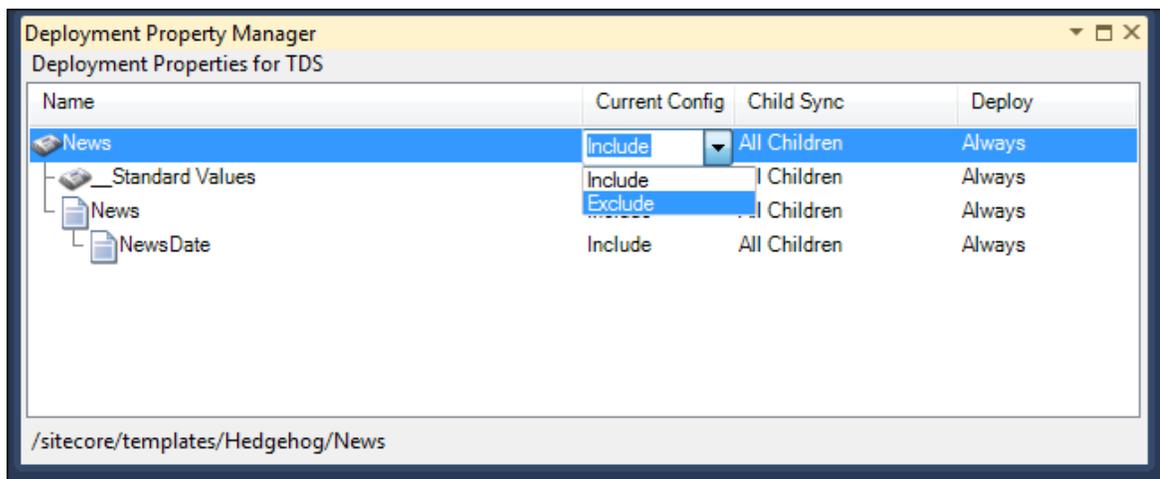


Figure 11 The TDS Deployment Property Manager allows you to distinguish the configuration for the parent node through all the children nodes.

Chapter 10

Additional Resources

On our website www.hhogdev.com you can find many other resources that can assist you while using TDS. We constantly update and add new things to our site to make sure our customers have the most update information available.

FAQ SECTION

Our FAQ section is stocked with important questions that our customers ask. We feel that sharing these questions will help everyone learn from them in the future. Most questions can be answered by viewing our extensive FAQ Section, which can be found by visiting <http://www.hhogdev.com/Products/Team-Development-for-Sitecore/FAQ.aspx>.

SAMPLE PROJECT

On our website we offer a sample project that you can download and test out. This project will enable you to see how we assemble our TDS projects, as well as go through a whole workflow using TDS. To download the sample project, just navigate to http://www.hhogdev.com/Products/Team-Development-for-Sitecore/~media/Files/Products/Team_Development/TDS-Sample.ashx.

INFORMATIVE VIDEOS

As new features are released Hedgehog releases videos that tell their customers how to take advantage of a particular feature. These videos can be extremely informative to developers that want to see how TDS features perform in a development setting. Just visit our Youtube channel, <http://www.youtube.com/user/hedgehogdevelopment>, to find many informative videos regarding TDS and much more.

White-Paper

Our top-notch developers have written a full document in which we provide valuable guidance, insights and tips on how to help simplify the frequently overwhelming and multi-faceted decision process you face when you set out to develop Sitecore in a team environment. This document is extremely helpful to developers having trouble integrating Sitecore in a team environment, to download this document just fill out the form found by visiting <http://www.hhogdev.com/Whitepapers/Sitecore/Team-Development-Best-Practices.aspx>.