## Exercise 3: Using Visual Studio to Deploy Applications

The first time you deploy a service to Windows Azure using Visual Studio, you need to create the necessary credentials to access your account. This involves generating a self-signed certificate that you can upload to the **Windows Azure Platform Management Portal**. Once you set up your account information in Visual Studio, you can deploy your current solution in the background.

In this exercise, you set up the credentials to authenticate with the Windows Azure Management Service and then deploy the *MyTodo* application from Visual Studio.

### Task 1 – Preparing the Solution for Deployment

When you deploy your service using Visual Studio, the Windows Azure Tools upload the service package and then automatically start it. You will not have a chance to update the configuration settings before the service starts. Therefore, you must configure all the necessary settings before you publish the service.

In this task, you update the storage connection strings to point to your Windows Azure storage account.

1. Open Microsoft Visual Studio 2010 as administrator.
2. Open ***MyTodo.sln*** solution file (locates in *Ex3-DeployingWithVisualStudio\Begin*).
3. In Solution Explorer, expand the **Roles** node inside the *MyTodo* cloud project and then double-click the ***MyTodo.WebUx*** role.
4. In the ***MyTodo.WebUx* [Role]** window, switch to the **Settings** tab and configure the necessary Azure Storage account details replacing [YOUR\_ACCOUNT\_NAME] with the name of your storage account and [YOUR\_ACCOUNT\_KEY] with the shared key. Do this for both *DataConnectionString* and *DiagnosticsConnectionString* settings. These are the same values that you used in previous exercises to configure the application.
5. Press **CTRL + S** to save your changes.

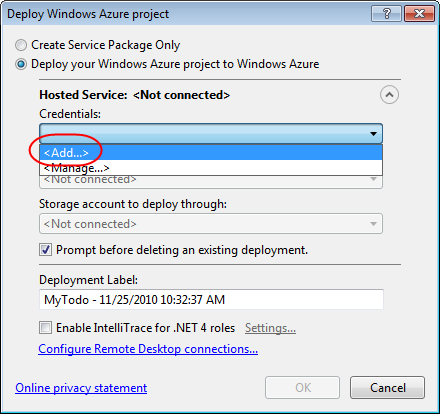
**Task 2 – Generating a Self-Signed Certificate in Visual Studio**

In this task, you create a second management certificate using Visual Studio—the first one you created earlier using a command line tool—and then upload it to the Windows Azure Platform Management Portal.

1. If it is not already open, launch Microsoft Visual Studio 2010 as administrator. To do this, in **Start | All Programs | Microsoft Visual Studio 2010**, right-click the **Microsoft Visual Studio 2010** shortcut and choose **Run as administrator**.
2. In the **File** menu, choose **Open** and then **Project/Solution**. In the **Open Project** dialog, browse to **Ex3-DeployingWithVisualStudio\Begin** in the **Source** folder of the lab, select **MyTodo.sln** in the folder for the language of your preference (Visual C# or Visual Basic) and click **Open**.

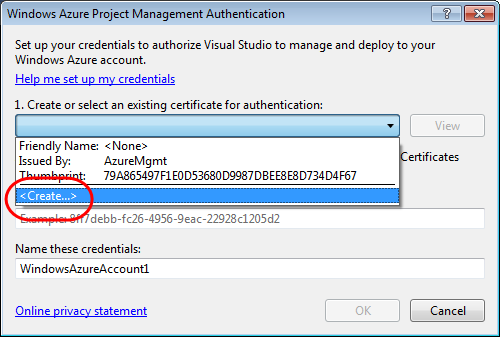
|  |
| --- |
| **C:\WindowsAzurePlatformTrainingKit\Labs\WindowsAzureDeploymentVS2010\Lab.html\local\note.gifNote:** |
| This is the same solution deployed earlier except for a legend in its footer area to indicate that it was deployed using Visual Studio. |

1. In **Solution Explorer**, right-click the **MyTodo** cloud project and select **Publish**.
2. In the **Deploy Windows Azure project** dialog, you first need to create the necessary credentials to access your Windows Azure account. First, select the option labeled **Deploy your Windows Azure project to Windows Azure**. Next, to add a new set of credentials to your configuration, in the **Credentials** drop down list, select **Add**.



**Figure 46**  
*Adding Windows Azure account credentials in Visual Studio*

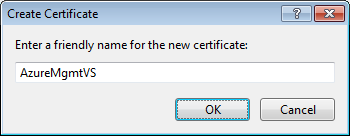
1. To create the credentials, you require a certificate. If Visual Studio cannot find a suitable certificate in your personal certificate store, it will prompt you to create a new one; otherwise, in the **Windows Azure Project Management Authentication** dialog, expand the drop down list labeled **Create or select an existing certificate for authentication** and then select **Create**.



**Figure 47**  
*Creating a new certificate for authentication*

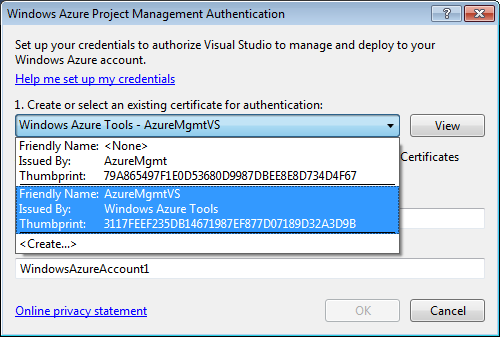
|  |
| --- |
| **C:\WindowsAzurePlatformTrainingKit\Labs\WindowsAzureDeploymentVS2010\Lab.html\local\note.gifNote:** |
| The drop down list contains all the certificates that are suitable for authentication with the Azure Management API. This list includes the certificate you created earlier, during the PowerShell deployment exercise. Nevertheless, in this exercise, you create a new certificate to walk through the procedure required to generate certificates in Visual Studio. |

1. In the **Create Certificate** dialog, enter a suitable name for the certificate, for example, *AzureMgmtVS*, and then click **OK**.



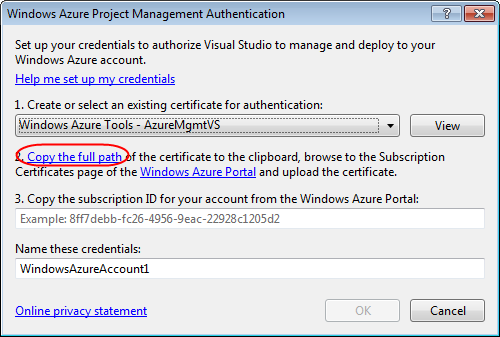
**Figure 48**  
*Creating a new management certificate*

1. Back in the **Windows Azure Project Management Authentication** dialog, ensure that the newly created certificate is selected. Notice that the issuer for this certificate is the Windows Azure Tools.



**Figure 49**  
*Selecting a certificate for the credentials*

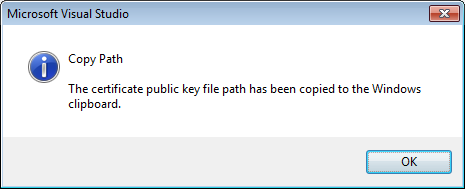
1. Now, click the link labeled **Copy the full path**. This copies the path of the certificate public key file to the clipboard.



**Figure 50**  
*Copying the path of the certificate public key file to the clipboard*

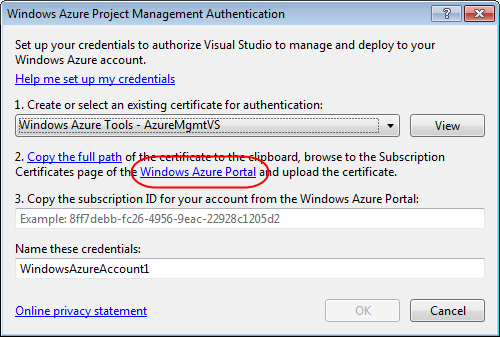
|  |
| --- |
| **C:\WindowsAzurePlatformTrainingKit\Labs\WindowsAzureDeploymentVS2010\Lab.html\local\note.gifNote:** |
| Visual Studio stores the public key file for the certificate it generates in a temporary folder inside your local data directory. |

1. Click **OK** to dismiss the confirmation message box and then save the path in the clipboard to a safe location. You will need this value shortly, when you upload the certificate to the management portal.



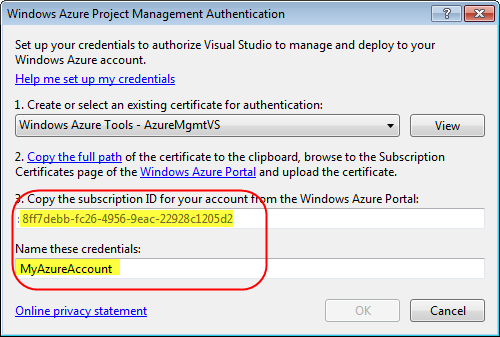
**Figure 51**  
*Confirmation that the file path was copied to the clipboard successfully*

1. Next, click the link labeled **Windows Azure Portal** to open a browser window and navigate to the Windows Azure Platform Management portal.



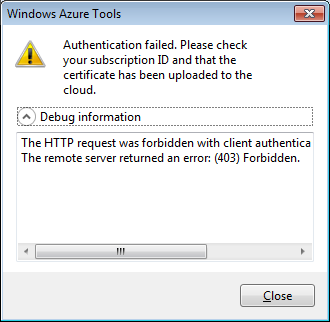
**Figure 52**  
*Opening the Developer Portal in your browser*

1. At the Management Portal, sign in and then upload the certificate that you generated in Visual Studio using the same procedure described in **Task 2** of the previous exercise. Refer to that section of the document for detailed instructions.
2. If you did not keep a record of the **Subscription ID** of your account, make a note of this value from the **Account** page now. You will require it for the next step.
3. At the **API Certificates** page, when selecting a certificate file from your local storage, make sure to use the path that you copied earlier from your clipboard to specify the certificate public key file to upload.
4. To complete the setup of your credentials, switch back to the **Windows Azure Project Management Authentication** dialog, enter your **subscription ID** and a name to identify the credentials and then click **OK**.



**Figure 53**  
*Completing the credential setup procedure*

1. After you confirm the creation of the new credentials, Visual Studio uses them to access the management service to verify that the information that you provided is valid and notifies you if authentication fails. If this occurs, verify the information that you entered and then re-attempt the operation once again.



**Figure 54**  
*Authentication failure while accessing the management service*

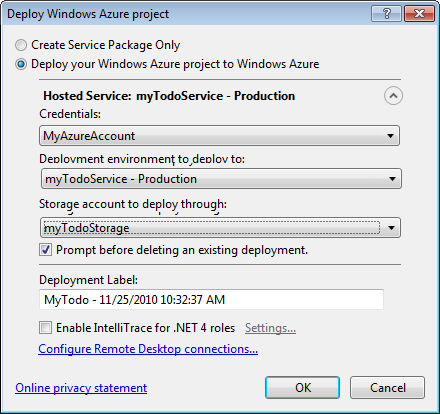
1. In the **Deploy Windows Azure project** dialog, click **Cancel**. You will restart the publishing process in the next task.

**Task 3 – Deploying a Service with the Windows Azure Tools**

In the previous task, you configured a set of credentials that provide access to your Windows Azure account. Visual Studio saves this information and allows you to reuse the credentials whenever you need to publish a service, without requiring you to enter your credentials again.

In this task, you use these credentials to publish the MyTODO application directly from Visual Studio.

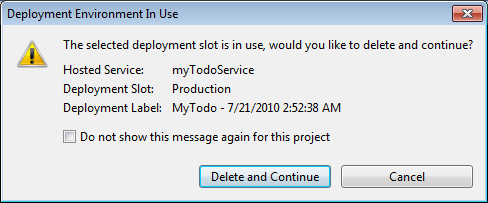
1. In **Solution Explorer**, right-click the **MyTodo** cloud project and select **Publish**.
2. In the **Deploy Windows Azure project** dialog, select the credentials that you created in the previous task.
3. Once you select the credentials, notice that the dialog populates the drop down list labeled **Deployment Environment to deploy to** with the information for all the services configured in your Windows Azure account, including production and staging slots for each service. Select the slot in this list where you wish to deploy the service.
4. Similarly, the dialog populates the drop down list labeled **Storage account to deploy through** with all the storage services that you have configured in your Windows Azure account. To deploy a service, Visual Studio first uploads the service package to Azure storage, and then deploys the service from there. Select the storage service that you wish to use for this purpose.
5. Leave the check box labeled **Prompt before deleting an existing deployment** checked.
6. If you wish, update the **Deployment Label** to identify the deployment in the Developer Portal UI.



**Figure 55**  
*Publishing a service from Visual Studio*

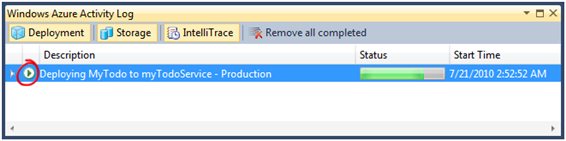
|  |
| --- |
| **C:\WindowsAzurePlatformTrainingKit\Labs\WindowsAzureDeploymentVS2010\Lab.html\local\note.gifNote:** |
| Although this is not covered in this lab, the IntelliTrace option enables you to capture detailed trace logs of your running service in the cloud that you can download to your desktop to perform historical debugging. This can be invaluable when troubleshooting issues that occur during role start up. Note that IntelliTrace requires the .NET Framework 4 and it is only available in Visual Studio Ultimate edition. |

1. Click **OK** to start the deployment process.
2. If the slot that you chose is already occupied by a previous deployment, Visual Studio warns you and asks for confirmation before it deletes it. Click **Delete and Continue** if you are certain that the current deployment is no longer needed and can be overwritten. Otherwise, click **Cancel** and repeat the operation choosing a different deployment slot.



**Figure 56**  
*Overwriting an existing deployment*

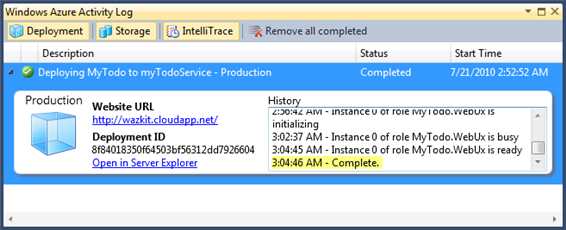
1. After you start a deployment, you can examine the Windows Azure activity log window to determine the status of the operation. If this window is not visible, in the **View** menu, point to **Other Windows**, and then select **Windows Azure Activity Log**.
2. By default, the log shows a descriptive message and a progress bar to indicate the status of the deployment operation.



**Figure 57**  
*Viewing summary information in the Windows Azure activity log*

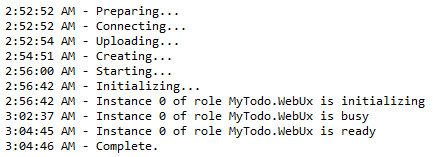
1. To view detailed information about the deployment operation in progress, click the green arrow on the left side of the activity log entry.

Notice that the additional information provided includes the deployment slot, **Production** or **Staging**, the **Website URL**, the **Deployment ID**, and a **History** log that shows state changes, including the time when each change occurred.



**Figure 58**  
*Viewing detailed information about a deployment operation*

1. Wait for the deployment operation to complete, which may take several minutes. While this is happening, you can examine the **History** panel on the right side to determine the status of the deployment. For a successful deployment, it should resemble the following sequence.



**Figure 59**  
*Deployment operation history log*

1. Once the deployment operation is complete, in the **Windows Azure Activity Log**, click the **Website URL** link for the completed operation to open the application in your browser and ensure that it is working properly. Notice the legend in the copyright notice at the bottom of the page indicating that this is the version that you deployed with Visual Studio.



**Figure 60**  
*Running the application deployed with Visual Studio*