Create an installer for website with WIX, part 3

16 Replies

In part 1 we saw how we can use a MS Build script to harvest all the files that needs to be installed and to create our msi installer. In part 2 we saw how we can add our own custom UI.

With only the files installed you don't have a website running. At least not an ASP.NET MVC website. You have to register the website in the Internet Information Services (IIS), create an app pool with the correct framework selected and activate (start) the website.

This post will handle that part of the creation of the installer where we extend the project we used in part 1 & part 2.

Add the IISConfiguration WiX file

We start by adding a new item in our setup project, choose for an installer file and name it IISConfiguration.wxs. The template will create an empty WiX page with just an empty "Fragment" element.

We'll start with adding a component for the application pool.

We add a "DirectoryRef" element that we point to our installation folder so that this will be run when handling the installation directory in the installer.

In the "Component" element we add the "iis:WebAppPool" element. Give it an id and name, the name will be displayed in IIS. Choose the Identity that will start the application pool.

You can choose "networkService", "localService", "localSystem","applicationPoolIdentity" or "other". With "other'" you'll have to create a user with a username and password that you have to assign to the "iis:WebAppPool" element "users" parameter (not in this example).

NOTE: I had to install the 3.8 version of the WiX toolset to be able to assign the "applicationPoolIdentity". In previous versions this was apparently not possible.

If you work in Visual Studio you'll notice the editor adds an red line under the "iis:WebAppPool" element.

We have to add the IISExtension namespace in the Wix declaration.

Add the install website component

Next we’ll add a second component to our file to create the website in IIS.

Here we’ll have to an unique id, a description of the website ‘(will be shown in IIS), the directory where the website is installed (our “INSTALLFOLDER”) and the auto start variables.

In the “iis:Website” element you can add one or more “iis:WebAddress” elements to assign DNS names and TCP ports to the website.

Last but not least, we’ll have to connect the correct application pool to the website by adding a “iis:WebApplication” with a unique id and name (I took the website id) and a reference to the “iis:WebAppPool” element we created before.

Create and reference component group

It’s best practice to group these two components in a component group so you only have to add one reference to iis configuration elements.

Add this “ComponentGroup” in the “Target” element.

In the product.wxs file we have to place a reference to this component group in the “Feature” tag.

Update the MS Build file

Because we created an extra file to include in the build process we must be sure that the file is added in the MS Build file.

We have to update the item groups that contain the list of files to include as shown above.

Now open up the Developer Command Prompt for Visual Studio 2012 again and change the prompt to the setup project folder and type the next statement:

msbuild /t: Build;PublishWebsite;Harvest;WIX;DeleteTmpFiles setup.build

and hit enter.

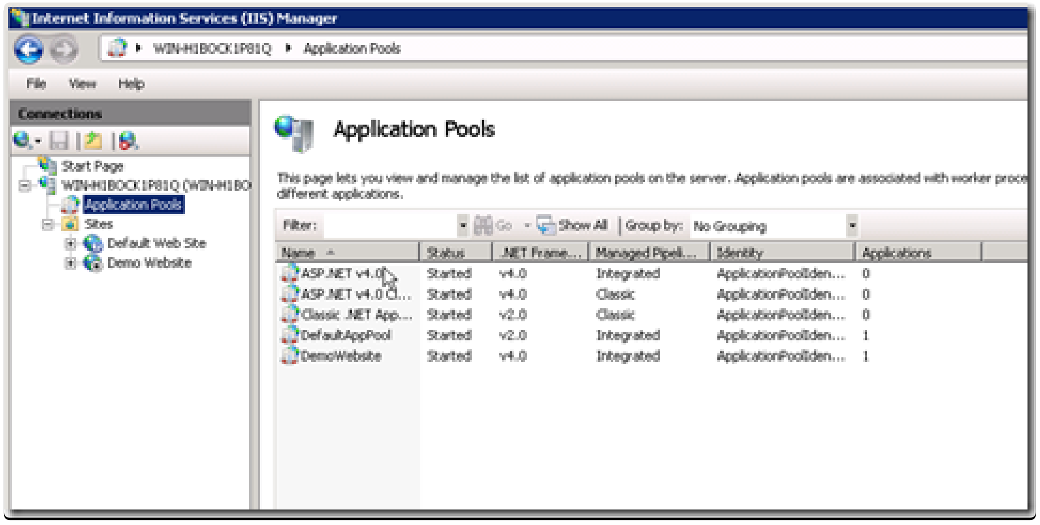
You'll see the build process will return an error. We added a reference to the IISExtensions library in our IISConfiguration.wxs file but we didn't told the candle.exe and light.exe tools to take that into account.

Add in both commands the –ext WixIISExtension flag and run the MS build script again. This time it shouldn't be returning any errors.

Test the installer

Let's run the installer and see if it really adds our website.

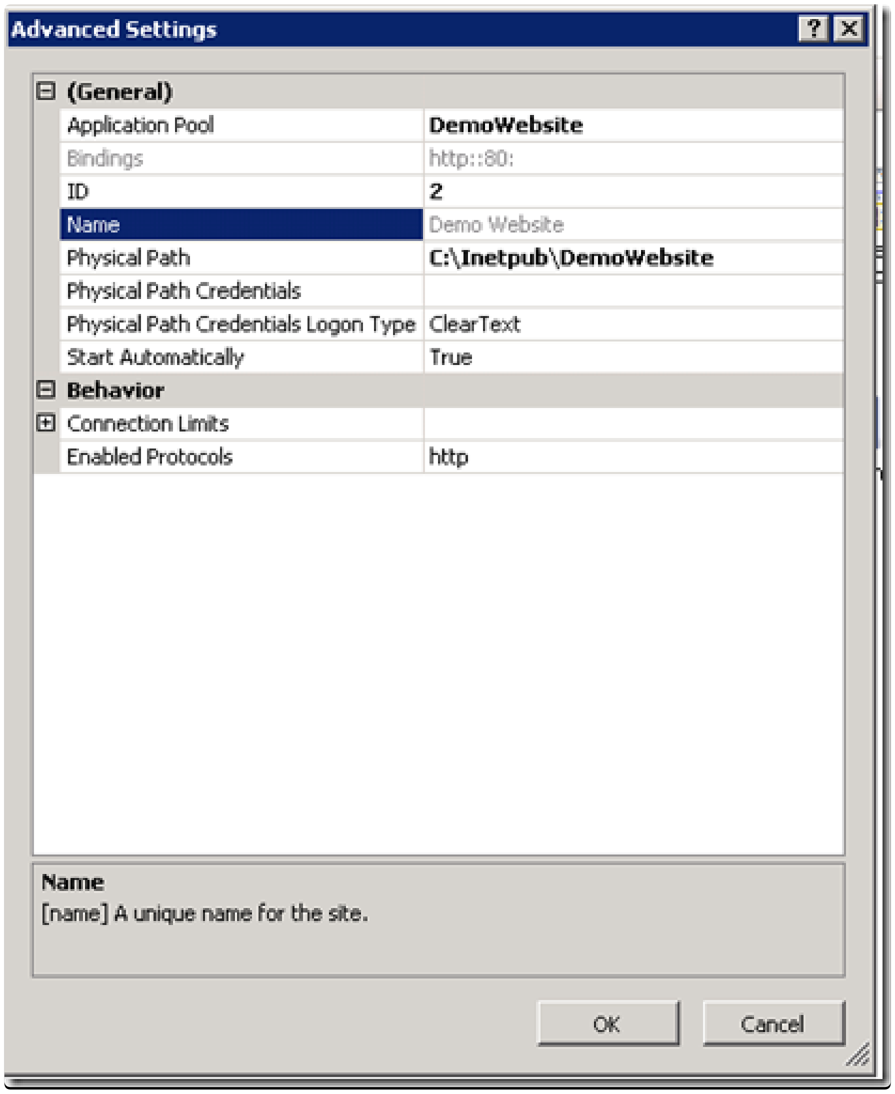
During the installation you'll see a message passing by that it's configuring IIS. Complete the installation and open up the IIS manager (run inetmgr).



This page lets you view and manage the list of application pools on the server. Application pools are associated with worker proce different applications.

Name	Status	.NET Frame...	Managed Pipeli...	Identity	Applications
ASP.NET v4.0	Started	v4.0	Integrated	ApplicationPoolIden...	0
ASP.NET v4.0 Cl...	Started	v4.0	Classic	ApplicationPoolIden...	0
Classic .NET App...	Started	v2.0	Classic	ApplicationPoolIden...	0
DefaultAppPool	Started	v2.0	Integrated	ApplicationPoolIden...	1
DemoWebsite	Started	v4.0	Integrated	ApplicationPoolIden...	1

We'll see a new Demo Website website on the left hand side and a new application pool DemoWebsite on the right hand side running .NET 4.0 under the correct application pool identity.



⊟ (General)	
Application Pool	DemoWebsite
Bindings	http::80:
ID	2
Name	Demo Website
Physical Path	C:\Inetpub\DemoWebsite
Physical Path Credentials	
Physical Path Credentials Logon Type	ClearText
Start Automatically	True
⊟ Behavior	
⊞ Connection Limits	
Enabled Protocols	http

Name
[name] A unique name for the site.

If we open up the advanced settings of the website we'll see the correct physical path is used and the binding is correct.

You may have noticed that our website isn't started (the stop sign on the website icon). This isn't a fault of the installer, it tried to start the website but because we choose port 80 in our installer IIS noticed there is another site running on that port (Default Web Site) and refuses to start our installed website.

If you change the port in the installer (or remove/stop the default webs site before installing) you'll notice it starts upon install.

Next parts

This concludes the installation in IIS. Next part we'll handle the creation of the database using the entered credentials and running SQL scripts from the installer.

- Install the .NET 4.5 framework if that isn't installed already
- Install the MVC 4 framework if that isn't installed already.
- Create a folder and copy all needed files to run the application (done)
- Create a new database on an existing SQL server and prefill the database with the correct tables and values. (the connection details and database name should be entered by the end user running the installer)
- Create a new website in IIS 7.5 (create website and application pool running under .NET 4.5) (done)
- Alter the config file so the correct connection settings are used(entered by the end user)

Complete source code

Can be found on Github.

Other posts in this series:

- Create an installer for website with WIX, part 1 (install files, create MS Build script)
- Create an installer for website with WIX, part 2 (Custom UI)
- Create an installer for website with WIX, part 3 (Install website in IIS)
- Create an installer for website with WIX, part 4 (Create database and run scripts)