Agenda

1. *Introduction to ASP.NET Web Application*
2. *Advantages of IIS Applications*
3. *Creating web application in IIS*
4. *Converting File System application to IIS Application*
5. *Using Virtual Directory*
6. *Publishing ASP.NET Websites*

IIS / ASP.NET Web Application

An **Web Application** is collection of files including Pages, HTTP Handlers, HTTP Modules and code within the scope of a physical directory or its sub directory for which a **virtual directory** is created.

- Default Physical Directory of a website in IIS: **c:\Inetpub\wwwroot.** i.e., by default only the content present in this directory or its subdirectories can be accessed by clients on internet.
- Irrespective of the location of the physical folder a virtual directory can be created in IIS. Only then the content of it can be accessed by clients on internet.
- A virtual directory can be created either under root directory of the website or under another virtual directory.
- If the same content has to be accessible from different URL's, multiple virtual directories can be created for the selected physical directory i.e. for one physical directory we can have more than one virtual directory.

Advantages of Web Applications

1. Because of web application, content of any folder on server machine can be published on internet.
2. In ASP.Net for every web application a new App-Domain is created and thus every application has its own sessions and applications data which is not shared with web pages in other web applications.
3. Every web application can have its own configuration settings including security, application mapping, Version of ASP.NET, Master Pages, global.asax file etc.

Note: The root directory of the web application in ASP.NET has some special features as some of the content can be placed only in that directory and not in sub directories.

Using VS.NET when we create an ASP.NET website with Location = IIS, It creates

1. A sub folder in c:\inetpub\wwwroot i.e. the default directory of the website. The name of the folder is the name of the application.
2. A Virtual Directory is created under the root directory of the website.

Creating New Web Application in IIS

1. Start→Programs→Administrative Tools→Internet Information Server
2. Select Default Website → View Application (right panel)
3. Right click → Add Application
4. Specify Alias and Physical directory → OK

Converting File System application to IIS Application

1. Create a new Web Application in VS.NET using File System
2. Close VS.NET.
3. Copy the complete folder with all its subdirectories to the target machine or any other folder on same machine.
4. For this folder create an IIS Application as mentioned in steps above.

5. Start VS.NET in Administrator mode
6. Go to File → Open → WebSite → Select Local IIS in left panel → select the Web Application in right Panel.

Using Virtual Directory

1. Create physical folders D:\WA1, D:\WA2 and D:\Images
2. Make "D:\WA1" as a Web Application = "/WA1".
3. Make "D:\WA2" as a Web Application = "/WA2".
4. Go to IIS → Right Click on WA1 → Add Virtual Directory → Alias = "Images" → Physical Directory = "D:\Images"
5. Repeat previous step for WA2.
6. In browser we can visit the Images in d:\Images folder either using
    a. http://localhost/WA1/images/Logo.gif
    b. http://localhost/WA2/images/Logo.gif

To Publish Website

Build and Publish Website…

Provide Target Location eg: c:\temp\MyWebsitePublished

a) Allow this Precompiled Site to be updatable:
    a. If this is checked, then the content of aspx files is retained as it is in the development website. The only change is done in <%Page ….Inherits = "ClassName, **AssemblyName**" where AssemblyName is the name of DLL in bin folder of the website.
    b. If this is Un-Checked the content of ASPX file is also compiled and included in the DLL files generated in the bin folder.
b) Use fixed Naming and Single Page Assemblies
    a. If unchecked, for all aspx pages only one DLL file is generated.
    b. If checked, for every aspx page separate DLL file is generated.

After publishing the website

1. The content of the folder as it is can be uploaded to the web server using FTP.
2. Using a Setup Project, MSI file can be created for installing on the web server.
    a. Add a new WebSetup Project to the solution.
    b. Copy the published content into the Web Application Folder
    c. Build the Setup project.