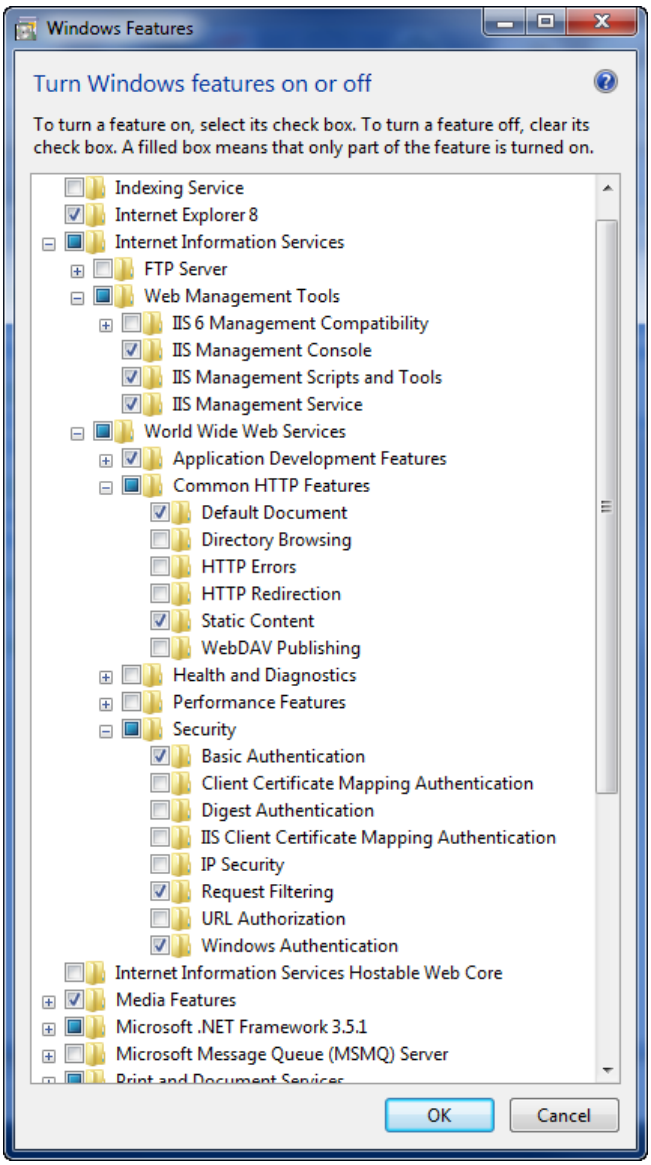
Classic ASP and C# COM

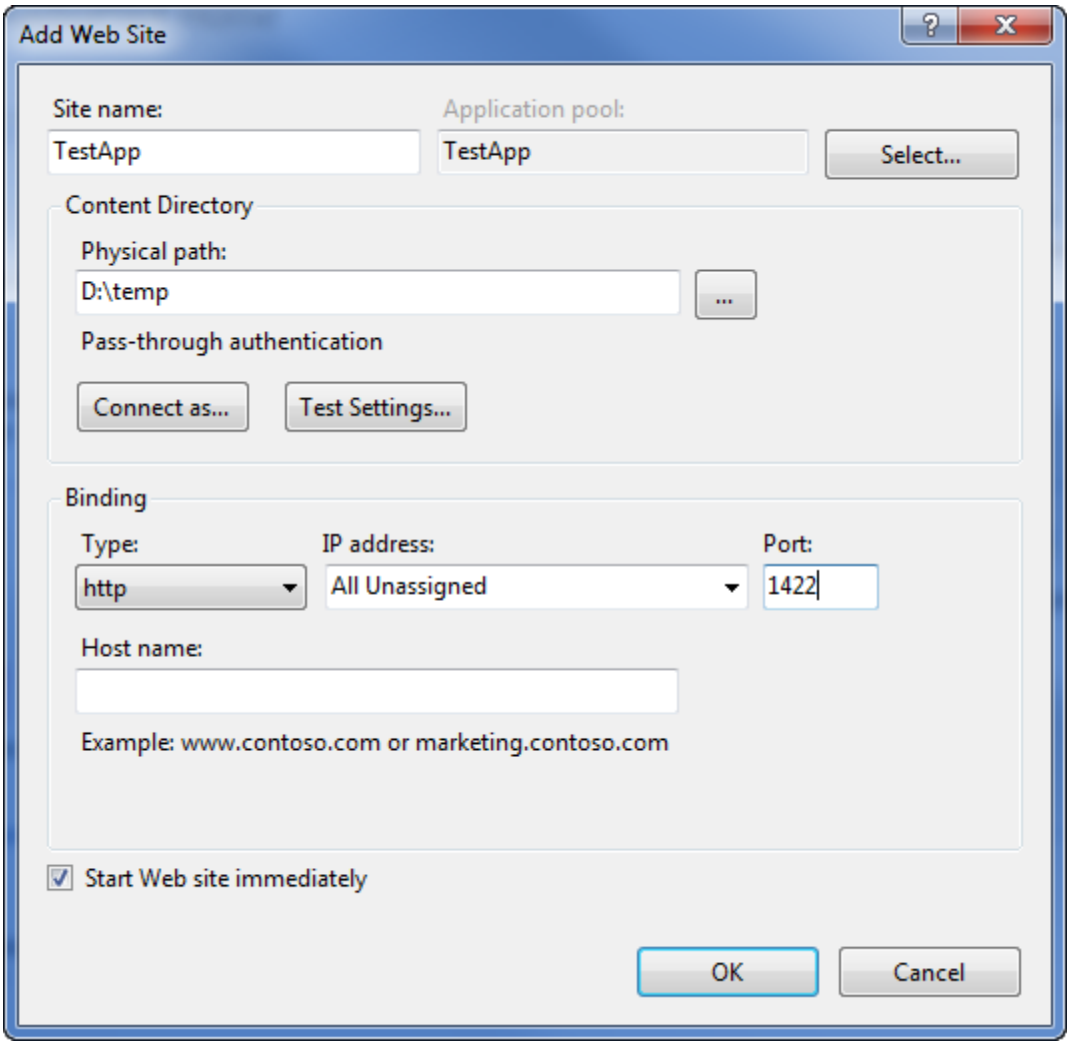
Debugging Classic ASP in Visual Studio 2005/2008/2010

Windows 7 - Control Panel >> Programs >> Turn Windows Features on or off

- Under Internet Information Services, enable IIS Management Console, IIS Management Scripts & Tools, and IIS Management Service.
- Under World Wide Web Services:
  - Application Development Features - enable everything
  - Common HTTP Features - enable Default Document and Static Content
  - Security - enable Basic Authentication, Request Filtering and Windows Authentication
- Click OK



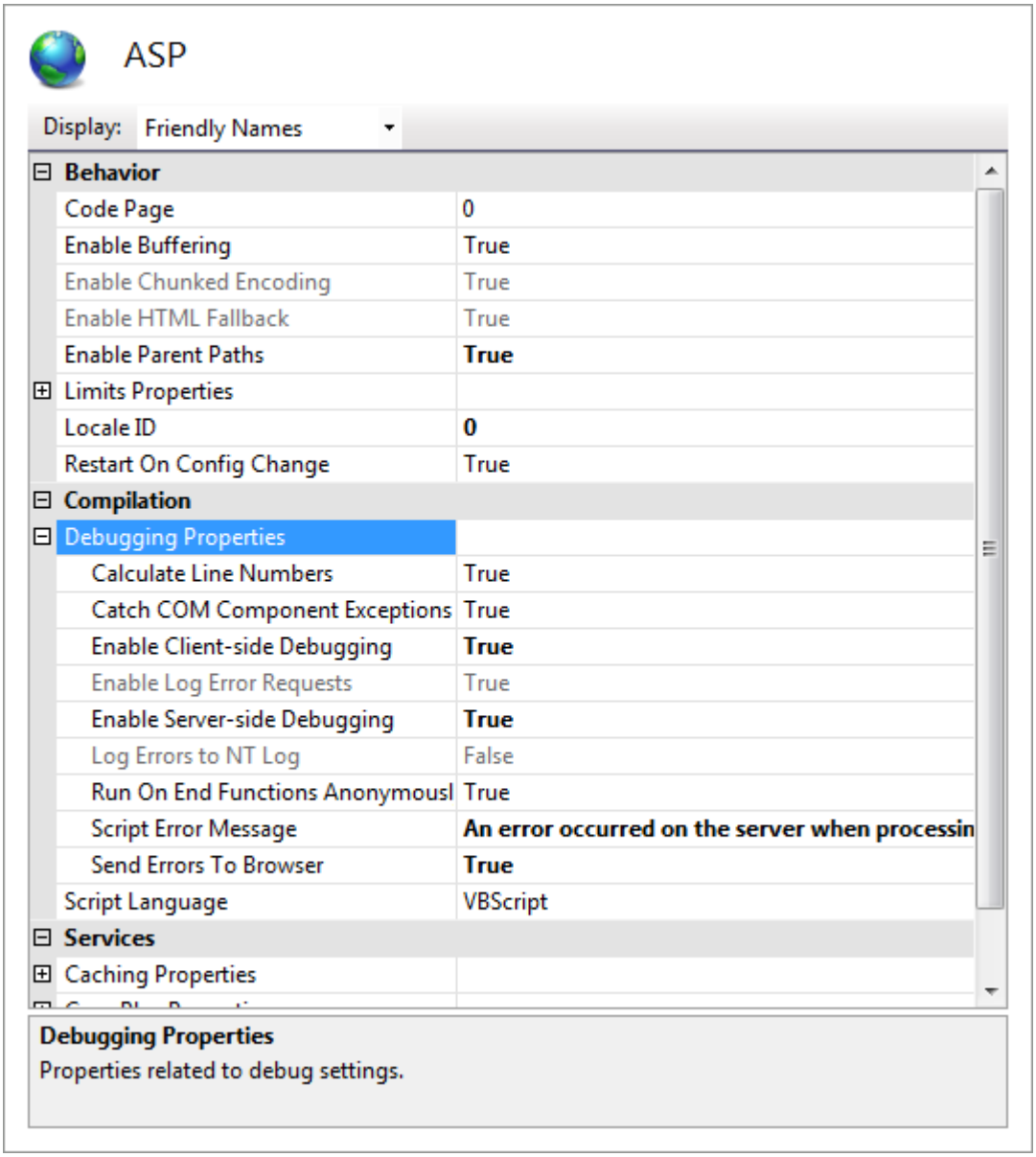
In IIS, add a website that points to the physical location of your files. I'd recommend using a specific port for running the site - see the example below for creating a site named TestApp running on port 1422 - you'll use http://localhost:1422 to browse to the site. Note that an Application Pool is created with the same name - TestApp.



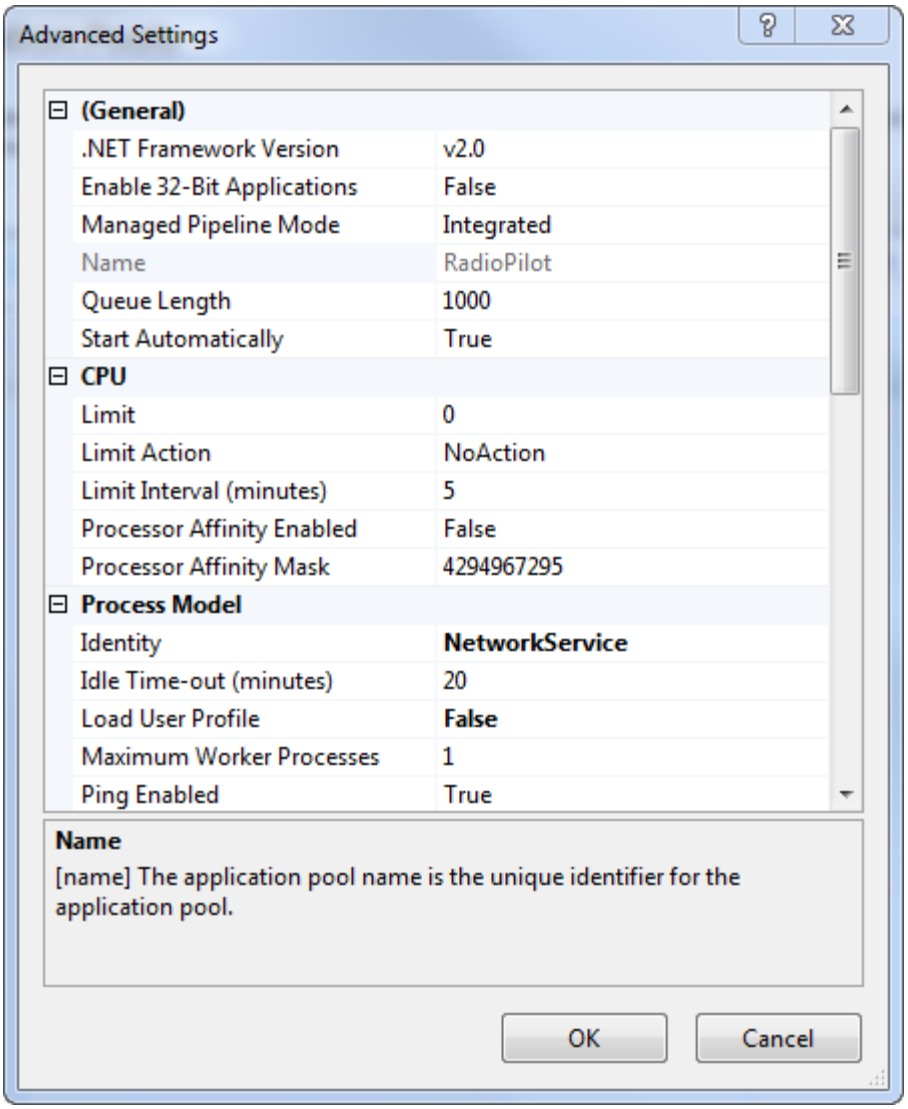
Next, in IIS Manager Features View, double-click Default Document and ensure the Default.asp is in the list. Backup to the Features View and double-click - ASP icon.

- Enable Parent Paths - set to True
- Expand Debugging Properties

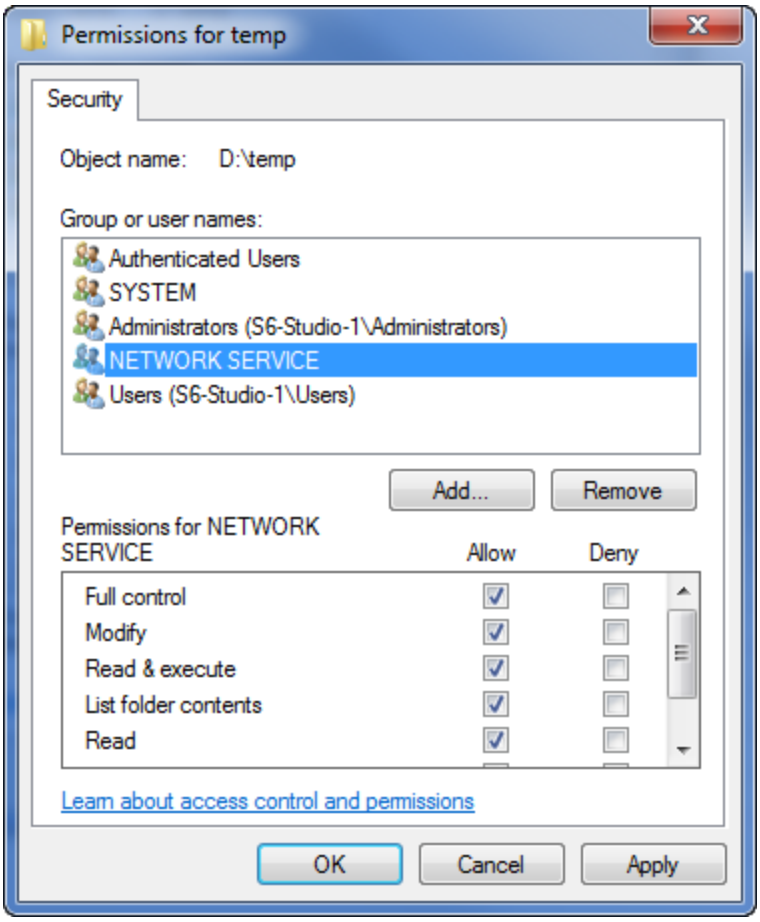
- Enable Client-side Debugging - set to True
- Enable Server-side Debugging - set to True
- Send Errors to Browser - set to True



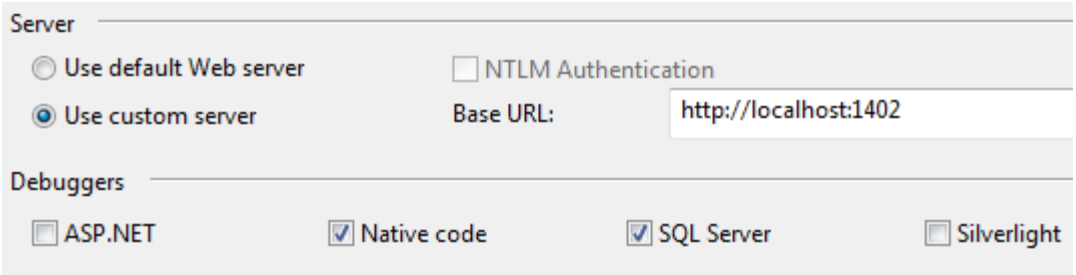
Now go back to IIS Connections and select the Application Pools node, select the Application Pool you created for your site and then select Advance Settings. In Advanced Settings, change the Identity to NetworkService (this makes it easy to set you security next).



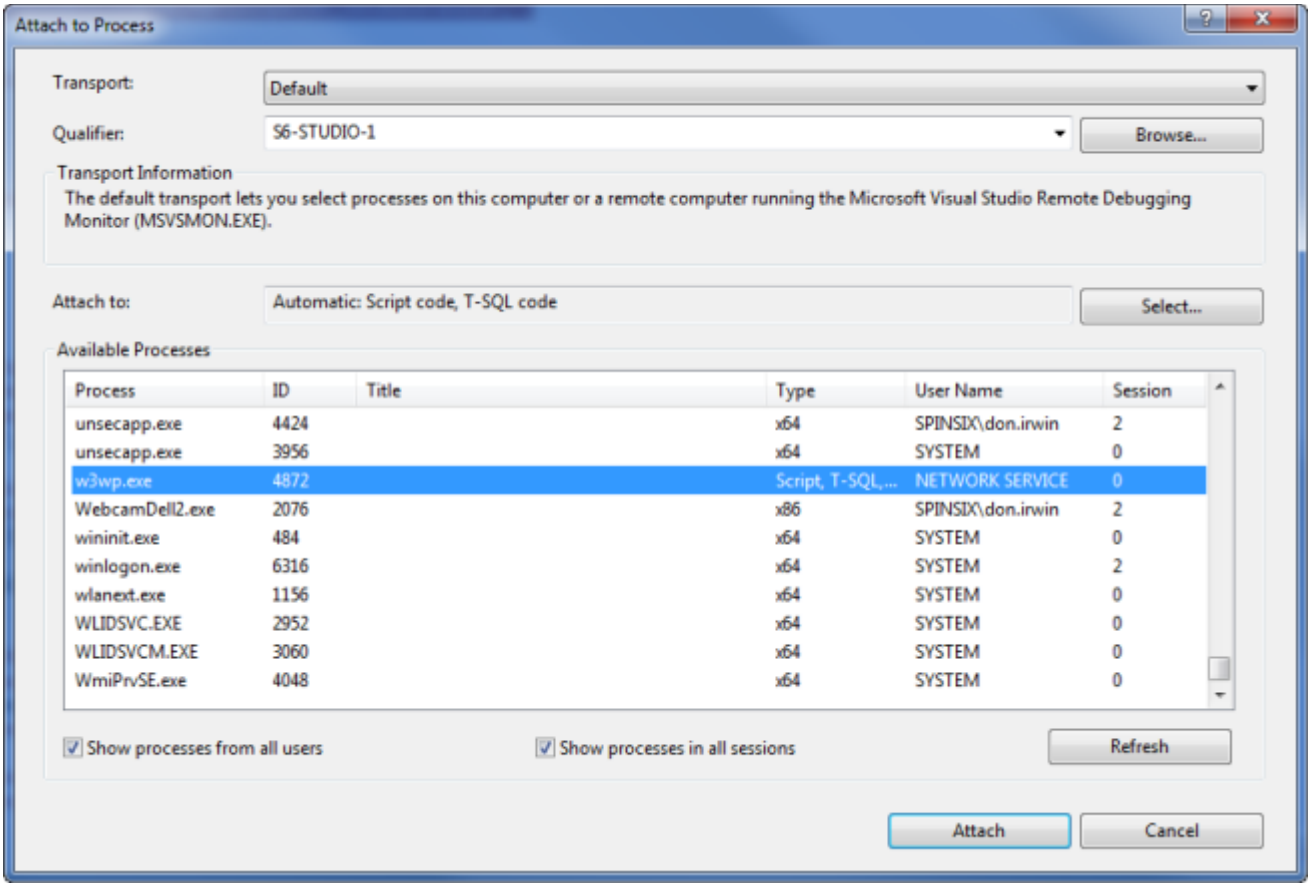
Now we need to give the User "Network Service" full permissions on the physical site. Go to Explorer, right-click on the top folder, select Properties >> Security. Add the user NETWORK SERVICE and assing full control.



Finally to Visual Studio 2010 - run Visual Studio as an administrator! Then, open your website - File >> Open Web Site >> File System and select the root folder then click open. Now in Solution Explorer, right-click on the site root and select Property Pages. Under Build, select "No Build", then under Start Options >> Server, select "Use custom server" and enter the base URL with the port number for the site you created earlier: http://localhost:1422 - click OK.



Now run your site - Debug (F5) - either the default page or the page you specified in Properties >> Start Options should launch in a browser now. Go back to Visual Studio 2010 and select Debug >> Attach to Process. At the bottom of the dialog, select bot "Show Processes..." check boxes - then in the "Available Processes" list, scroll to the bottom and select w3wp.exe and then click "Attach". If you get a warning select YES/Continue.



You should now be able to set breakpoints in your code and step through the site.

C# Example Code Snippet (ClassicASPCOM.dll) built with Strong Name and Registered for COM Interop in Visual Studio 2005 (.NET Framework 2.0) and Windows 7

```
using System.Runtime.InteropServices;

namespace ComNamespace1
{
   [Guid("EAA4976A-45C3-4BC5-BC0B-E474F4C3C83F")]
   public interface ComClass1Interface
   {
      [DispId(1)]
      string GetMessage(string Message);
   }

   [Guid("7BD20046-DF8C-44A6-8F6B-687FAA26FA71"),
      InterfaceType(ComInterfaceType.InterfaceIsIDispatch)]
   public interface ComClass1Events
   {

   }

   [Guid("0D53A3E8-E51A-49C7-944E-E72A2064F938"),
      ClassInterface(ClassInterfaceType.None),
      ComSourceInterfaces(typeof(ComClass1Events))]
   public class ComClass1 : ComClass1Interface
   {
       public string GetMessage(string Message)
       {
          return "Your Message: " + Message;
       }
   }
}
```

Exporting to COM to be called from Classic ASP

- Exposing Visual C# objects to COM requires declaring a class interface, an events interface if it is required, and the class itself. Class members must follow these rules to be visible to COM
- The class must be public
- Properties, methods, and events must be public
- Properties and methods must be declared on the class interface
- Events must be declared in the event interface
- COM does not support static methods
  This rule ignores property and event accessors, operator overloading methods, or methods that are marked by using either the System.Runtime.InteropServices.ComRegisterFunctionAttribute attribute or the System.Runtime.InteropServices.ComUnregisterFunctionAttribute attribute
- By default, the following are visible to COM: assemblies, public types, public instance members in public types, and all members of public value types
- Other public members in the class that are not declared in these interfaces will not be visible to COM, but they will be visible to other .NET Framework objects
- To expose properties and methods to COM, you must declare them on the class interface and mark them with a DispId attribute, and implement them in the class. The order in which the members are declared in the interface is the order used for the COM vtable
- To expose events from your class, you must declare them on the events interface and mark them with a DispId attribute. The class should not implement this interface
- The class implements the class interface; it can implement more than one interface, but the first implementation will be the default class interface. Implement the methods and properties exposed to COM here. They must be marked public and must match the declarations in the class interface. Also, declare the events raised by the class here. They must be marked public and must match the declarations in the events interface

Register Component for COM Interop

- With the project node selected in Solution Explorer, from the Project menu, click Properties (or right-click the project node in Solution Explorer, and click Properties)
- In the Project Designer, click the Build tab
- Select the Register for COM interop check box

Sign assembly

- With the project node selected in Solution Explorer, from the Project menu, click Properties (or right-click the project node in Solution Explorer, and click Properties)
- In the Project Designer, click the Signing tab
- Select the Sign the assembly check box
- Specify a new key file. In the Choose a strong name key file drop-down list, select <New...>. Note that new key files are always created in the .pfx format
- The Create Strong Name Key Dialog Box appears
- In the Create Strong Name Key dialog box, enter a name and password for the new key file, and then click OK

Create a Public/Private Key Pair (Strong Name)

- Create a strong-name key (cryptographic key pair) and store it in the file PayPalAPCOMKeyPair

sn -k PayPalAPCOMKeyPair.snk

- Extract the public key from PayPalAPCOMKeyPair.snk and put it into PayPalAPCOMPublicKey.snk
  sn -p PayPalAPCOMKeyPair.snk PayPalAPCOMPublicKey.snk
  sn -tp PayPalAPCOMPublicKey.snk

- Get the public key stored in the file PayPalAPCOMPublicKey.snk
  sn -tp PayPalAPCOMPublicKey.snk

Public key is
0024000004800000940000000602000000240000525341310004000001000100852fb0f89f9ad1
8db88b3f6ec7c2c475bd079c9e6d16293257986b7aed572b08cfd994b92a594445285d384b187f
8f426d102a03dce3699b022b2c6ed17a90c9f2db2306179b1c09038e91a9249a1c7ea7732f854e
25a919da7a924733bd90a9be39ad0c6c211470fa4f5813bcb11c50b0eca90198aed042e79569a7
d67ebfc0

Public key token is 9cdbbf45e4ede757

COM Installation Batch Script
Regasm ClassicASPCOM.dll
Regasm ClassicASPCOM.dll /codebase
Regasm ClassicASPCOM.dll /tlb
gacutil/i ClassicASPCOM.dll

COM Uninstallation Batch Script
iisreset
Regasm/u ClassicASPCOM.dll
Regasm/u ClassicASPCOM.dll /tlb
gacutil/u ClassicASPCOM.dll

'VBScript (ClassicASPHome.asp)

```
<%
Response.Write("Classic ASP Page:")
%>
<br />
<br />
<%
Dim com
Set com = Server.CreateObject("ComNamespace1.ComClass1")
If TypeName(com.error) <> "String" Then
   Response.Write(com.GetMessage("Hello World!"))
Else
%>
   Exception message:
<%
   Response.Write com.error
End if
%>
```

Certificate

Log In to Developer website https://developer.paypal.com/

PayPal account:
######@email.com
PayPal account password:
****

API username:
#*#*#*@email.com
API password:
****

https://www.sandbox.paypal.com/

##*#@email.com
****

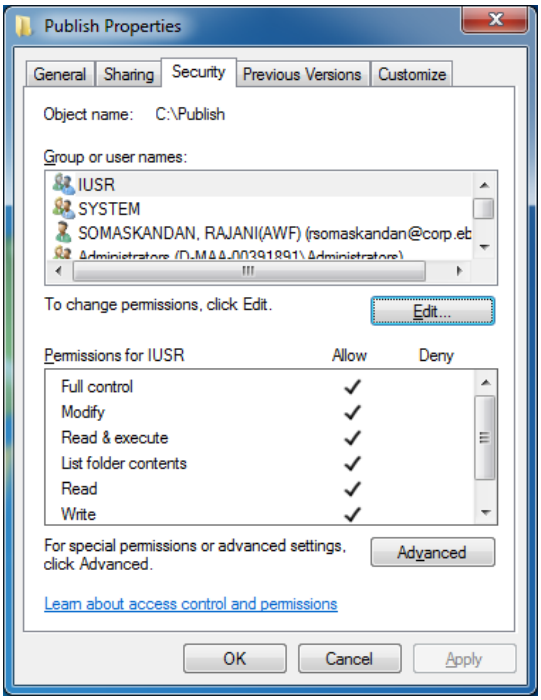
```
Dim CertificateSubjectDistinguishedName
    CertificateSubjectDistinguishedName = "C=US, S=TX, L=Austin, O=SDK Seller,
CN= #*#*#*@email.com"

    Dim configHashtable
    Set configHashtable = Server.CreateObject("System.Collections.Hashtable")
    configHashtable.Add "mode", "sandbox"
    configHashtable.Add "account1.apiUsername", "#*#*#*@email.com"
    configHashtable.Add "account1.apiPassword", " ***************"
    configHashtable.Add "account1.applicationId", "###-****************"
    configHashtable.Add "account1.apiCertificate",
CertificateSubjectDistinguishedName
    configHashtable.Add "account1.privateKeyPassword", "********"
```

>openssl pkcs12 -export -in cert_key_pem.txt -inkey cert_key_pem.txt -out cert_key.p12

Loading 'screen' into random state - done
Enter Export Password:
Verifying - Enter Export Password:

Publish Classic ASP

- Ensure IUSR is added to the publish folder



- Ensure IIS_IUSRS is added to the publish folder