

# Advanced Programming Using Visual Basic .NET

## Update for VB .NET 2003

by Julia Case Bradley  
and Anita C. Millspaugh

This supplement addresses changes in Visual Studio, Visual Basic and the .NET platform introduced since this text was written. Although most features remain unchanged in the 2003 release of Visual Studio, you should be aware of some changes. And Microsoft continues to make numerous updates in the area of security. Many of these changes will affect your programs, especially those related to IIS and SQL Server (MSDE).

### Topics Covered

- Referring to base class events in the VS IDE.
- Installing MSDE.
- Web security.
- Creating a virtual directory.
- Web Services database Credentials property.
- Mobile devices.
- Setup and deployment projects.
- Running applications and/or Web pages from a Windows application.
- New security appendix from Programming in Visual Basic .NET, VB 2003 Update Edition.
- Customized error handling for Web pages.

### The Visual Studio IDE

In the Editor window, you select events of the current form or class differently. The Object list now shows the specific class, rather than *BaseClassEvents*. For example, to write code for the Closing event of frmMain, drop down the Object list and select (*frmMain events*), then you can drop down the Events list and choose *Closing*.

This change requires a modification to the step-by-step exercise in Chapter 12. On page 405 in "Add an Event Procedure", Step 1 refers to

(*BaseClassEvents*). Instead the reference should be (*ValidDate events*), because the name of the class is *ValidDate*.

## Chapter 3: Windows Database Applications

Major changes have occurred in Visual Studio 2003 for installation of MSDE. The changes affect the security issues that relate to SQL Server installations. You should update your MSDE if you have a previously installed version.

MSDE is no longer included on the Visual Studio .NET installation CDs but must be downloaded (free) from Microsoft. This is a much safer approach, because the download files can be kept up-to-date to maintain the integrity of the application as new security issues arise. One of the main concerns with the earlier version was the Slammer worm. It is now highly recommended that you set a password when installing the desktop engine (MSDE).

### ***Installing MSDE—Step-by-Step***

1. Use Windows Explorer or My Computer to open the C:\Program Files\Microsoft Visual Studio .NET 2003\Setup\MSDE folder. The msde\_readme.htm folder contains a link to download the software. (Or use <http://www.microsoft.com/downloads/details.aspx?FamilyId=A0DAC778-60A6-4B11-8AA8-BF12261A303A&displaylang=en>)
2. Download sql2kdeskp3.exe.
3. Double\_click the .exe file to extract all files. By default, the files will be extracted to c:\SQL2KSP3\msde.
4. Open an MS-DOS command prompt, change to the directory where you extracted the files, and run one of the following setup commands. If your computer is networked, use the second version of the command.

*Non-networked version:*

```
Setup InstanceName = "NetSDK" SAPWD = "yourPassword"
```


*Networked version:*

```
Setup InstanceName = "NetSDK" SAPWD = "yourPassword" disablenetworkprotocols=0
```

*Password note:* You can make up any password, which is required for installation. However, the text applications and Microsoft' sample applications use Windows Authentication, which does not require a

password to access the database files. The InstanceName must be NetSDK to automatically install the sample databases.

*Networking note:* Setup attempts to prevent all network access to the instance of MSDN for security purposes, but introduces another problem in the process. The `disablenetworkprotocols` argument instructs the setup application to *not* disable network access.

5. Restart the machine. The MSDE icon should now appear in the System Tray. 

## Load the Sample Databases


1. Navigate to the C:\Program Files\Microsoft Visual Studio .NET 2003\SDK\v1.1\Samples\Setup folder.
2. Double-click on ConfigSamples.Exe file to install the sample databases.
3. Reboot your system.
4. The databases can be found at C:\Program Files\Microsoft SQL Server\MSSQL\Data..

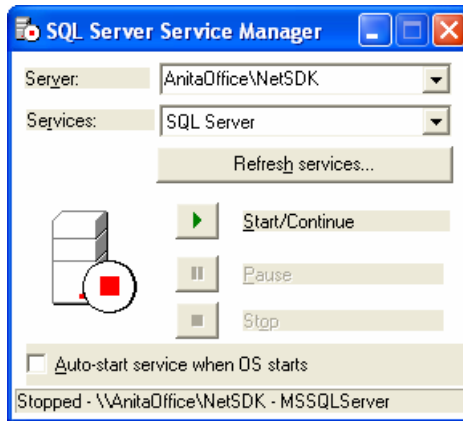
*Note:* If the sample databases will not load using this technique, you can load them using the `osql` utility to run the sql scripts found in the C:\Program Files\Microsoft Visual Studio .NET 2003\SDK\v1.1\Samples\Setup folder. To install from the scripts, use an MS-DOS command prompt in the script directory. You could copy the scripts to a folder on the root directory to make the job easier. From the script directory use the command:

```
osql /e /i scriptname.sql
```

For example, to install the pubs database use `osql /e /I instpubs.sql`.

## Start or Stop the SQL Server Service Manager

1. Double-click on the MSSQL Server icon  in the tray on the task bar. If the icon has a white circle you must Start the service using the following steps:
  - Type in your machine name followed by the MSDE instance name.



- Select SQL Server from the Services list.
- Click on the *Start* button to start the server. The icon will become green.
- To stop the service, click on the *Stop* button.

*Note:* If you receive a message that the server cannot be found, you may need to use the `disablenetworkprotocols` argument when you set up the MSDE instance. See the networking note on Page 2.

## ***Deleting the Sample Files***

If you find that you want to replace the sample databases, you cannot simply run the `ConfigSamples` application or the `osql` utility a second time. You must first remove the old files. Delete both the `.mdf` and the `.ldf` files from `C:\Program Files\Microsoft SQL Server\MSSQL\Data`. You can then install the samples again.

## ***IIS Security***

If your machine is running Internet Information Services (IIS) 6.0 you may get a "Login failed for user nnnnn\NetSDK". The default account for IIS does not have access to SQL Server. You can correct this by turning on impersonation. This requires an identity tag in the authorization section of the `web.config` file.

```
<identity impersonate = "true"/>
```

In addition, anonymous access must be turned off for the virtual directory. For more information see "Database Security for Web Applications" in Chapter 8, page 294, the "Security" section later in this supplement, or <http://localhost/iishelp>.

## ***Upsizing Access Database Files***

You can convert your Access .mdb files to SQL Server databases. From within Access, use the Upsizing Wizard from the *Tools/Database Utilities* menu. Select the option for creating a new database file.

## Chapter 4: Related Tables

Note that you do not need to set a relationship when you are using a filter. However, you must set the relationship when you use the *GetChildRows* method or the *GetParentRow* method.

## Chapter 6: Using Web Forms—ASP .NET

Database security for Web applications is very tight. Make sure to refer to the security instructions on page 294 to access a database.

The technique for creating a virtual directory is greatly simplified:

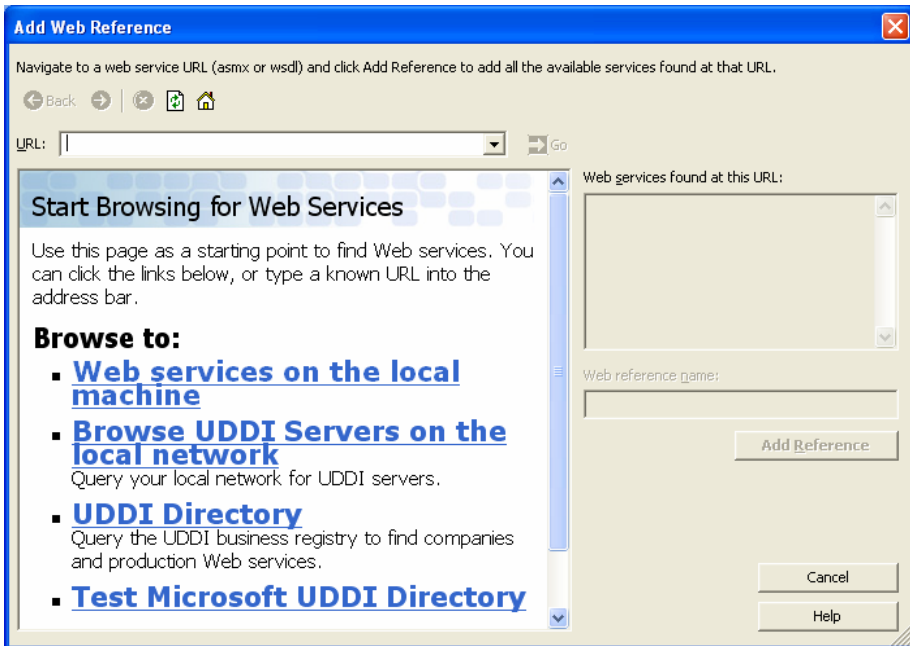
### ***Creating a Virtual Directory***

After you move a project to a new computer, you must open the Internet Services Manager to create a virtual directory. To access the Internet Services Manager you can right-click on MyComputer, select *Manage*, then double-click on *Services and Applications*. *Internet Information Services* should be the last item. Click on *Web Sites*, then *Default Web Site*. Select the folder for your project, right-click and display the Properties dialog box. On the *Directory* tab under *Application Settings* click on the *Create* button and then click on *OK*. Notice that the icon for the folder changes to indicate the virtual directory.

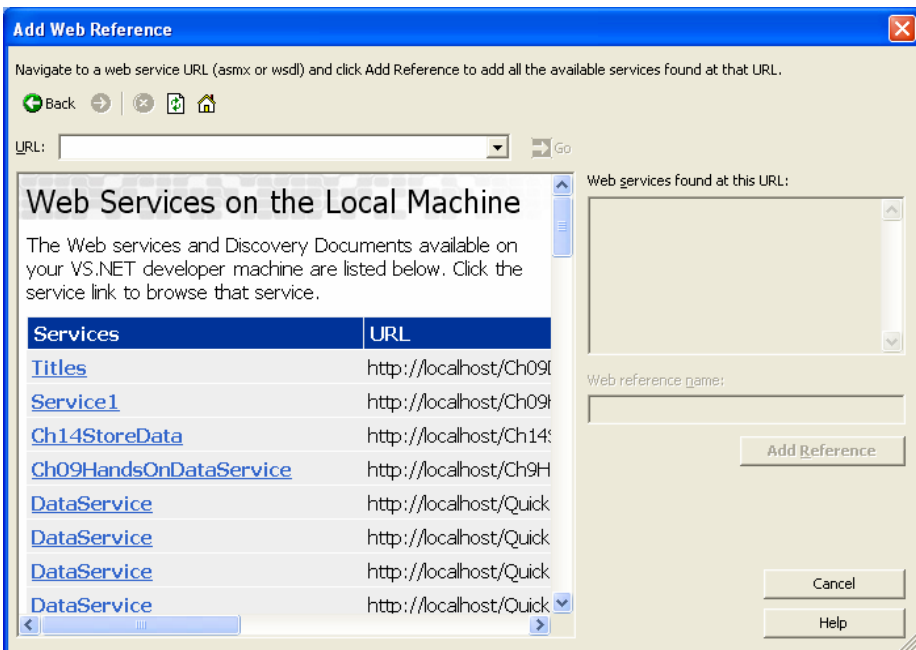
## Chapter 9: Web Services

### ***Browsing for Web Services***

In the first release of Visual Studio, it was necessary to know the URL of your Web Service to test it. Now when you add a Web Reference to your project, the *Add Web Reference* dialog box has a link allowing you to browse to a service on the local machine.



The second screen lists many services. If you have not changed the name of your service, it will be called *Service1*. You can verify that it is the proper service by checking the link under the URL column.



## ***Accessing Data using a Web Service***

When accessing a database using a Web Service, you may get the error "*The request failed with HTTP status 401: Access Denied*". This section explains how to avoid this problem which is caused when the Anonymous access is turned off. Recall that we turn off the Anonymous access as part of the process of retrieving data from a database. If you have not turned off the Anonymous Access you will receive another error such as "*Object reference not set to an instance of an object*". Refer to [www.support.com](http://www.support.com) Knowledge Base article 811318 for more information.

In order to use a database with a Web Service you now must set the Credentials property. You have the option of creating an instance of a credential with a specific user, password, and domain. However, for our purposes we are going to just use the default credential.

You must add the following code statement to your project.

```
WebServiceName.Credentials = System.Net.CredentialCache.DefaultCredentials
```

Therefore, the code on page 334 becomes:

```
Private Sub Page_Load(By Val sender As System.Object, _  
    ByVal e As System.EventArgs) Handles MyBase.Load  
    'Set the data source for the grid from the Web Service  
    Dim wsTitles as New localhost.TitlesData()  
  
    wsTitles.Credentials = System.Net.CredentialCache.DefaultCredentials  
    dgrTitles.DataSource = wsTitles.TitlesData  
    Me.DataBind()  
End Sub
```

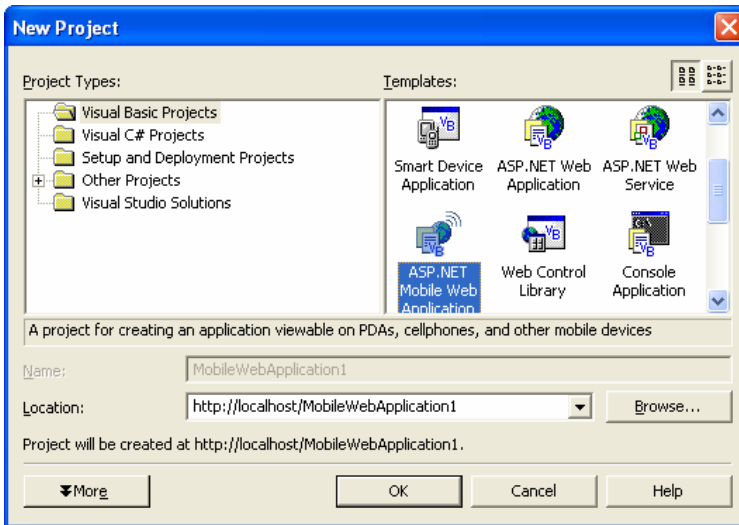
You also need to add the credentials to the Hands-On solution Page\_Load event procedure shown on page 339.

```
If Not IsPostBack Then  
    'Set the credentials  
    objData.Credentials = System.Net.CredentialCache.DefaultCredentials  
    'Get dataset  
    dsStoreSales = objData.getStoreSales()
```

## Chapter 14: Looking Ahead

The Microsoft Mobile Internet Toolkit is now integrated into the Visual Studio IDE. You can use the Mobile Web Application template to create any of the sample applications for mobile devices in the chapter.

To begin a new mobile Web application, select the ASP.NET Mobile Web Application template on the *New Project* dialog box. Notice that the project location is *localhost*, which by default is C:\InetPub\wwwroot.



Also notice the Smart Device Application template in the *New Project* dialog box. Smart Device applications are used to create Windows applications that can run on the Pocket PC or a Windows CE device.

The mobile applications work with a large number of devices. You can find more details by searching for "mobile controls" at [msdn.microsoft.com](http://msdn.microsoft.com). At the time of this writing, a listing of the devices and emulators is found at the following URL:

<http://www.asp.net/mobile/testeddevices.aspx?tabindex=6>

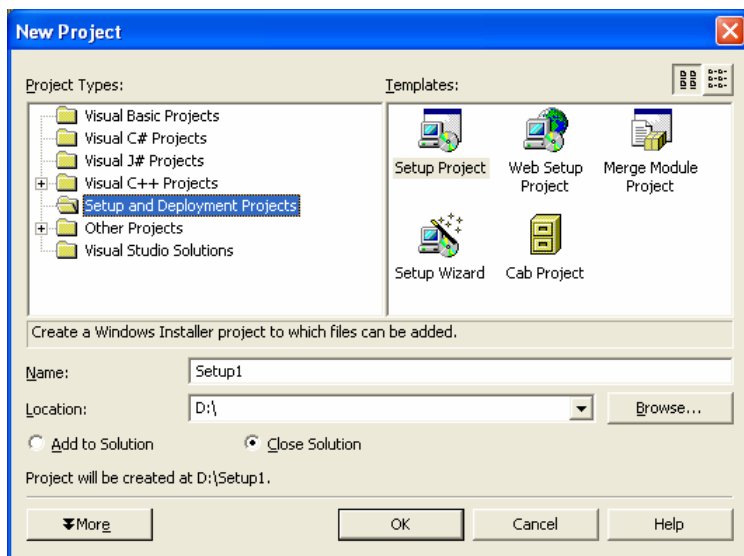
## Setup and Deployment Projects

You probably are already aware that you can copy an exe or dll file from the bin folder of a project and use the application on another computer. This is often referred to as an *XCopy deployment process*. However, you may want to create a CD or other media to allow your users to run a Setup file (an .msi file).

One of the advantages of using an msi file is that the user can also uninstall the application at a future date.

Visual Studio includes a feature that allows you to create a Setup that you can use to distribute your applications. In the *New Project* dialog box, the *Setup and Deployment* project type contains templates for setting up Windows projects and Web projects. The *Setup Project* template allows you to add files to a Windows Installer project. You also have the option of creating a Windows Installer for a Web project or for creating a Cab project. The Setup Wizard walks you through the process.

Microsoft Windows Installer is an application that you use to install and configure your programs. A Cab file, also referred to as a *.cab file*, stands for a cabinet file and is a compressed file that is similar to a zip file. An advantage of *.cab* files is that they can be split, which allows you to store the setup files for a large application on multiple floppy disks. A merge module contains dependencies for a project in a separate file.

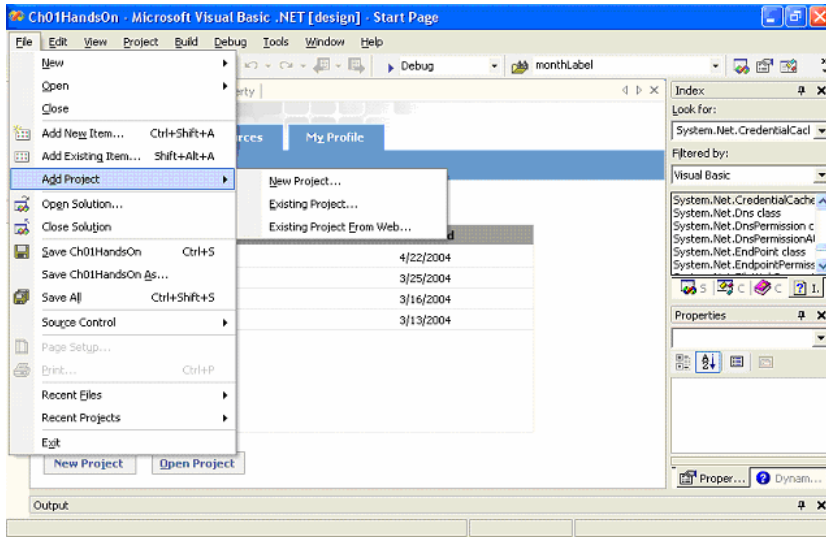


## ***Creating a Setup and Deployment Project—Step-by-Step***

### **Create a Project**

1. Open the project that you want to distribute. For this sample use the Ch01HandsOn from your text.
2. Choose an icon file (.ico extension) to use for the application and add it to the project's bin folder.

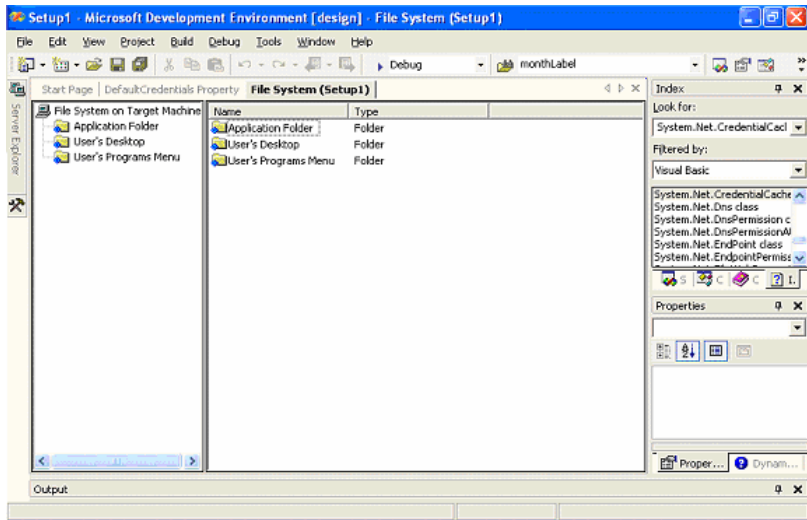
3. From the *File* menu, select *Add Project/New Project*.



4. Select *Setup and Deployment Projects* from the *Project Types*.
5. Select *Setup* from the *Templates* pane.
6. Set the desired name and location for your files.
7. Click OK.

## Create the Install program

1. The Setup project displays three folders: the Application Folder, the User's Desktop, and the User's Program Menus.



2. Open the Applications Folder. The properties for the Application folder set the Location to Program Folders. You may change this if you wish.
3. Use the *Action* menu to *Add/Project Output* to add the executable file to the Applications folder. You may also right-click on the Applications folder to select *Add*.
4. Open the User's Desktop folder.
5. From the *Action* menu or the context menu select *Create Shortcut to User's Desktop*.
6. In the Properties window, select *Icon* and browse to the icon in the Applications Folder.

## Test the Install Program

1. Right-click on *Setup project and Build the project* or select *Build* from the *Project* menu.
2. Right-click on *Setup project* and select *Install* from the menu.
3. After the installation completes, go to your desktop and double-click the icon to run the application.

## ***Installing the .NET Framework on the Target Computer***

One consideration is whether the systems installing your application will already have the .NET framework installed. Remember that the framework is necessary to run any .NET application. The framework is available as a free download from Microsoft but you may wish to save your users the trouble and include it on the Setup disk.

The Dotnetfx.exe file is the redistributable package for the .NET framework. You can create a single setup program that can first install the framework and then the application using the Setup.exe Bootstrapper.

<http://www.microsoft.com/downloads/details.aspx?FamilyID=bf253cfd-1efc-4fc5-ba7e-6a6f21403495&displaylang=en>

Want more details? Watch the online seminar presentation at:

<http://www.microsoft.com/seminar/shared/asp/view.asp?url=/Seminar/en/20030218VB004/manifest.xml>

*Note:* Each of the above URLs must be entered on a single line.

## Using the Process Class

The Process class allows you to run applications from your .NET project. You also can use it to start a Web browser and navigate to a specific page. Depending on the level of security on the network, you may be able to use the Process class to have your project send email. The Process class is in the System.Diagnostics namespace. The Start method contains the URL of a Web page or the path of the program you want to run.

```
System.Diagnostics.Process.Start("www.msn.com")
```

```
System.Diagnostics.Process.Start("C:\Windows\System32\mspaint.exe")
```

```
System.Diagnostics.Process.Start("mailto:someone@msn.com")
```

## .NET Security

*Note: Most of the following is taken from the updated introductory text, Programming in Visual Basic .NET, Visual Basic 2003 Update Edition by Bradley and Millsbaugh. It is Appendix D of the revised text.*

As a programmer, you must be aware of many aspects of security. You must not allow any unauthorized access to programs or data. But you must be able to

access all needed resources while you are developing applications. For both sides of the security issue you need a basic understanding of security topics.

The .NET Framework includes many features for implementing security. And as time passes and hackers and virus writers discover ways to circumvent the security, Microsoft is forced to tighten security. Because of this fact, you may sometimes find that programs or procedures that worked previously no longer work after you apply updates to Windows, which include updates to the .NET Framework and CLR.

For programmers security means information assurance. The .NET Framework provides many object-oriented features to assist in the process. The topic of security can fill multiple books and courses; this appendix is intended as an overview for introductory programmers.

## *Authentication and Authorization*

The two topics of authentication and authorization are frequently lumped together because they are so closely related. **Authentication** determines who the user is and **authorization** decides if the user has the proper authority to access information.

Authentication is based on credentials. When you are working with a Windows application, you only need to be concerned with Windows authentication. However, if your application is Web Forms-based (ASP.NET) authentication might be IIS Authentication, Forms-based (not recommended, it is an HTML request for credentials) or it might use Microsoft Passport which is a centralized profile service for member sites. It is also possible to create a custom authentication method or use none at all. The settings in the Web.config file determine the method of authentication.

The following excerpt from the Web.config file describes each of the authentication methods:

```
<!-- AUTHENTICATION
```

```
  This section sets the authentication policies of the application. Possible modes are "Windows", "Forms", "Passport" and "None"
```

```
  "None" No authentication is performed.
```

```
  "Windows" IIS performs authentication (Basic, Digest, or Integrated Windows) according to its settings for the application. Anonymous access must be disabled in IIS.
```

```
  "Forms" You provide a custom form (Web page) for users to enter their credentials, and then you authenticate them in your application. A user credential token is stored in a cookie.
```

"Passport" Authentication is performed via a centralized authentication service provided by Microsoft that offers a single logon and core profile services for member sites.

-->

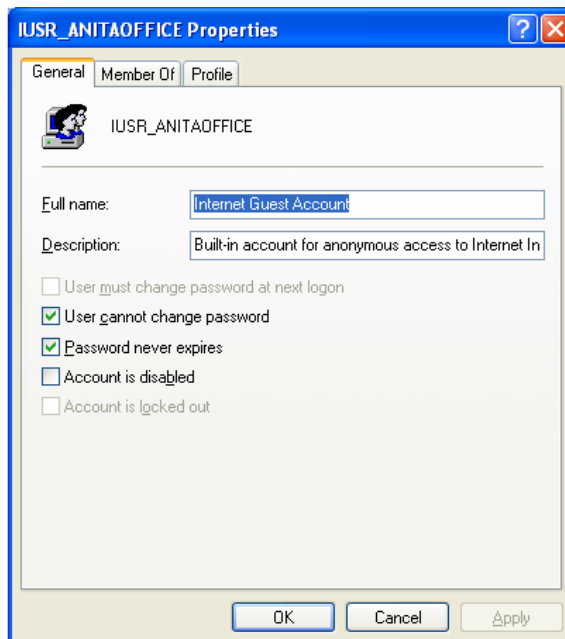
```
<authentication mode="Windows" />
```

For no authentication you can add the following line to the Web.config file:

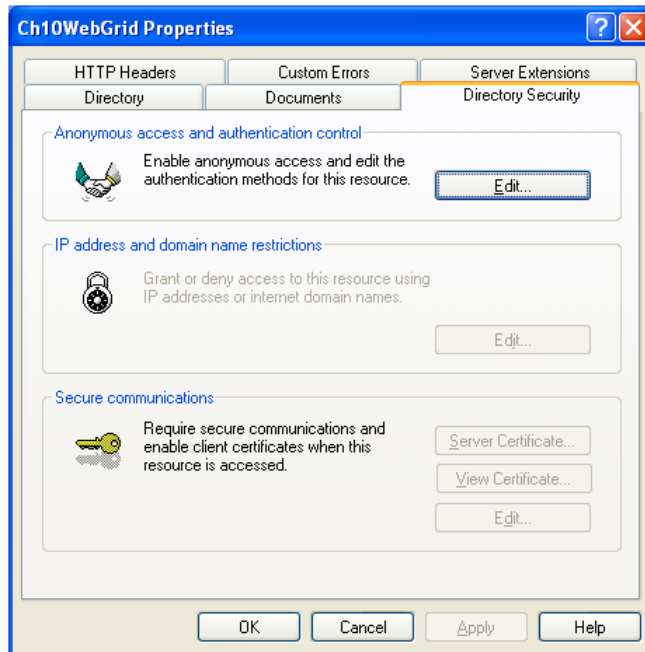
```
<authentication mode = "None"/>
```

## IIS Authentication

A Web Form runs on IIS (Internet Information Server). When IIS is installed, the Setup creates a user account for anonymous access. The username is `IUSR_computername`. For example, if the machine is called `SteveL`, the username is `IUSR_SteveL`. This same anonymous-logon user is then used for all IIS services installed on the computer. The password for the account is randomly generated. The following dialog can be displayed by going to *Control Panel / Administrative Services / Computer Management* and the expanding the *Local Users and Groups*.



The *Authentication Methods* dialog is accessed through IIS. From a project's properties dialog, select the *Directory Security* tab and then click on the *Edit* button under *Anonymous Access and authentication control*.

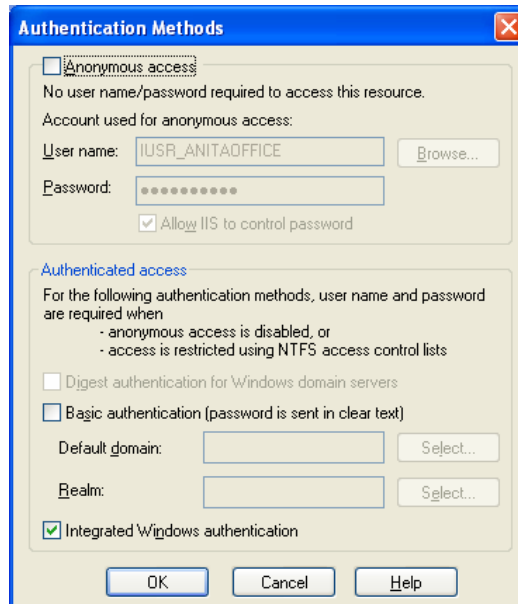


Anonymous access uses the anonymous user account. In the example programs, we have been removing the check mark for anonymous access and allowing Windows to perform the authentication.

Another authentication method is Basic authentication. This requires a Windows username and password. As you can see in the *Authentication Methods* dialog, this technique sends the password in clear text.

The Digest authentication, available on Windows domain servers, encrypts the password before it is sent across a network.

The final check box is the *Integrated Windows authentication* which requires users to run Internet Explorer 3.1 or later.



For more information see the following file from Microsoft:

<http://support.microsoft.com/support/kb/articles/q142/8/68.asp&NoWebContent=1>

If you are working with a SQL Server database or MSDE you should refer to:

<http://support.microsoft.com/support/kb/articles/Q247/9/31.ASP&NoWebContent=1>

## Authorization and Impersonation

After the user is authenticated, another step checks for authorization. If ASP.NET does not use impersonation, the ASP.NET user which is created when the Framework is installed runs without any privileges. If impersonation is turned on, ASP.NET takes on the identity from IIS. If anonymous access is turned off, ASP.NET takes on the credentials of the authenticated user, otherwise it impersonates the account that IIS uses.

This is where the following code applies.

```
<identity impersonate = "true"/>
```

For a specific user you can use:

```
<identity impersonate = "true" name = "Domain/username" password = "password"/>
```

The default authorization in the Web.config file has commented fields to specify users and roles.

```
<!-- AUTHORIZATION
```

```
    This section sets the authorization policies of the application. You can  
    allow or deny access to application resources by user or role.
```

```
    Wildcards: "*" mean everyone, "?" means anonymous  
    (unauthenticated) users.
```

```
-->
```

```
<authorization>
```

```
    <allow users="*" /> <!-- Allow all users -->
```

```
        <!-- <allow    users="[comma separated list of users]"  
            roles="[comma separated list of roles]"/>
```

```
        <deny     users="[comma separated list of users]"  
            roles="[comma separated list of roles]"/>
```

```
-->
```

```
</authorization>
```

## ***Writing Secure Code***

Programmers need to be aware of how hackers are able to gain access to a database or a network through code. Two primary areas of importance are string injections and error messages that may give away important information about a data source.

## **SQL Injection**

Proper validation of the code is extremely important. A system vulnerability occurs when code is "injected" into a string. A text box or a combo box control allows the user to type in information. It is the responsibility of the programmer to make sure that the code typed does not contain any scripting code or disruptive characters. When a program is working with a database there must be no way for the user to inject code into that database.

It is wise to validate the keystrokes to be sure that the input text contains only valid characters.

## **Error Messages**

Another technique that hackers use to find information about a database is to input wrong data hoping that the error message will give significant information. The default error messages indicate the name of the field containing an error. Plan your error messages so that you don't allow someone to determine valid field names or values.

You should make sure that you enclose your data access in a try/catch block and use a message box to display your desired message.

## Customized Error Handling for Web Pages

With a Web page you can avoid having the system display a message by setting up customized error handling.

- Create a Web Form that displays the message you want the user to see.
- In the Web.config file set the mode for customErrors.

```
<!-- CUSTOM ERROR MESSAGES
```

```
    Set customErrors mode="On" or "RemoteOnly" to enable  
    custom error messages, "Off" to disable.
```

```
    Add <error> tags for each of the errors you want to handle.
```

```
-->
```

```
<customErrors mode="On" />
```

- Add code in the Application\_Error event procedure in the Global.asax file.

```
Sub Application_Error(ByVal sender As Object, ByVal e As EventArgs)
```

```
    ' Fires when an error occurs
```

```
        Response.Redirect("ErrorForm.aspx")
```

```
End Sub
```

## Code Access Security

Basically, code access security determines what code is allowed to do on a computer; specifically, what resources such as hardware or files that the code can access. The settings of the computer ultimately determine if the code can execute or if a resource can be used by the code.

Code access security is based on several Permission classes. The Permission classes are organized into three types: Code Access, Identity, and Role-based. Examples of the code access type include the PrintingPermission class and the RegistryPermission class; SiteUdentityPermission and the URLIdentityPermission class fall into the identity permission type. The PrincipalPermission class is the role-based type for user credentials.

If your program has an OpenFileDialog component it may be necessary to request permission to access the files from a given system. The following code would appear at the top of the file.

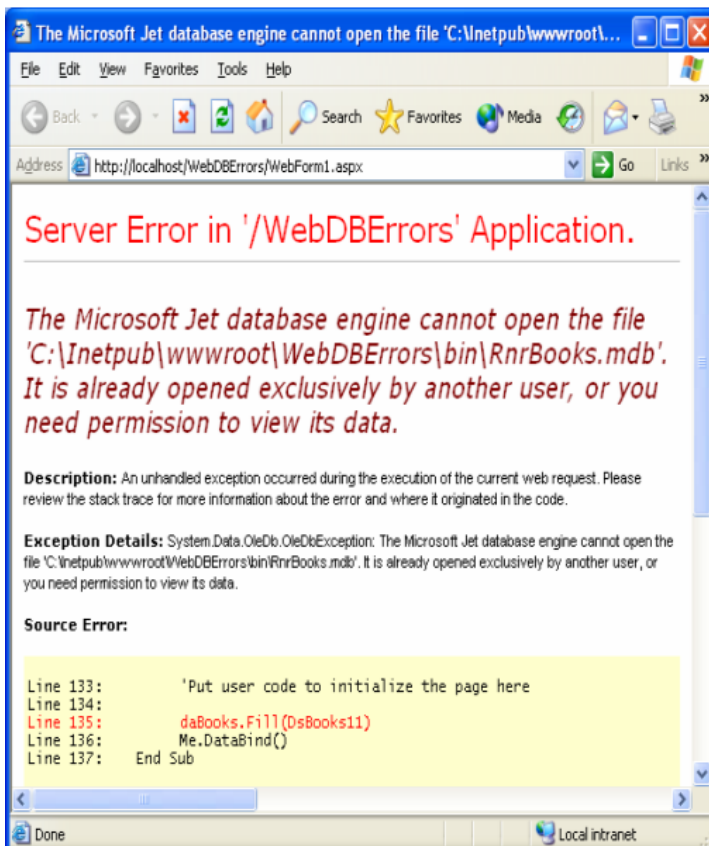
Imports System.Security.Permissions

<Assembly: FileDialogPermissionAttribute(  
SecurityAction.RequestMinimum, Unrestricted: = True)>

*Note:* Enter the statement on a single line.

## ***TroubleShooting***

If you receive an error trying to access a database in a Web application you must set the program up for Windows authentication and impersonation. The error message is *The Microsoft Jet database engine cannot open the file 'C:\Inetpub\wwwroot\filename\bin\databasename.mdb'. It is already opened exclusively by another user, or you need permission to view its data.*



1. In the Web.config file add a line following the Windows authentication line instructing the program to use impersonation.

After the line <authentication mode="Windows" /> type:

<identity impersonate = "true"/>

2. Disable Anonymous Access: Run *inetmgr*, expand to the default web site and locate your program. From the properties dialog, select the *Directory security* tab, click on the *Edit* button and deselect the *Anonymous Access* check box.

## Unspecified Error

When an Unspecified Error occurs on the Fill method of your data adapter, check to make sure that the *Anonymous access* box has been deselected in IIS.

