CS708 Lecture Notes

Visual Basic.NET Programming

Object-Oriented Programming

Web Technologies and ASP.NET

(Part I)

(Lecture Notes 5B)

Prof. Abel Angel Rodriguez

Section I. Introduction to Web Applications & Technologies

1.0 Overview

1.1 Introduction

- Due to the World-wide web, client/server web applications are becoming the norms. With the demands of E-Commerce technologies have been developed to make it easier to create complex web applications.
- In this section we examine these technologies and learn the basics required to create web applications using ASP.NET.

1.2 Web-Base Client/Server Applications

- Let's review the components for a Web-based Client/Server application:
- Client machines don't run a client program, but put as little as possible on the client machine.
- Instead of a client program, the client runs an HTML Browser Application such as Microsoft Internet Explore and Netscape Navigator which handle presenting the data to the user or ***User-Interface (UI)***.

Client or Browser:

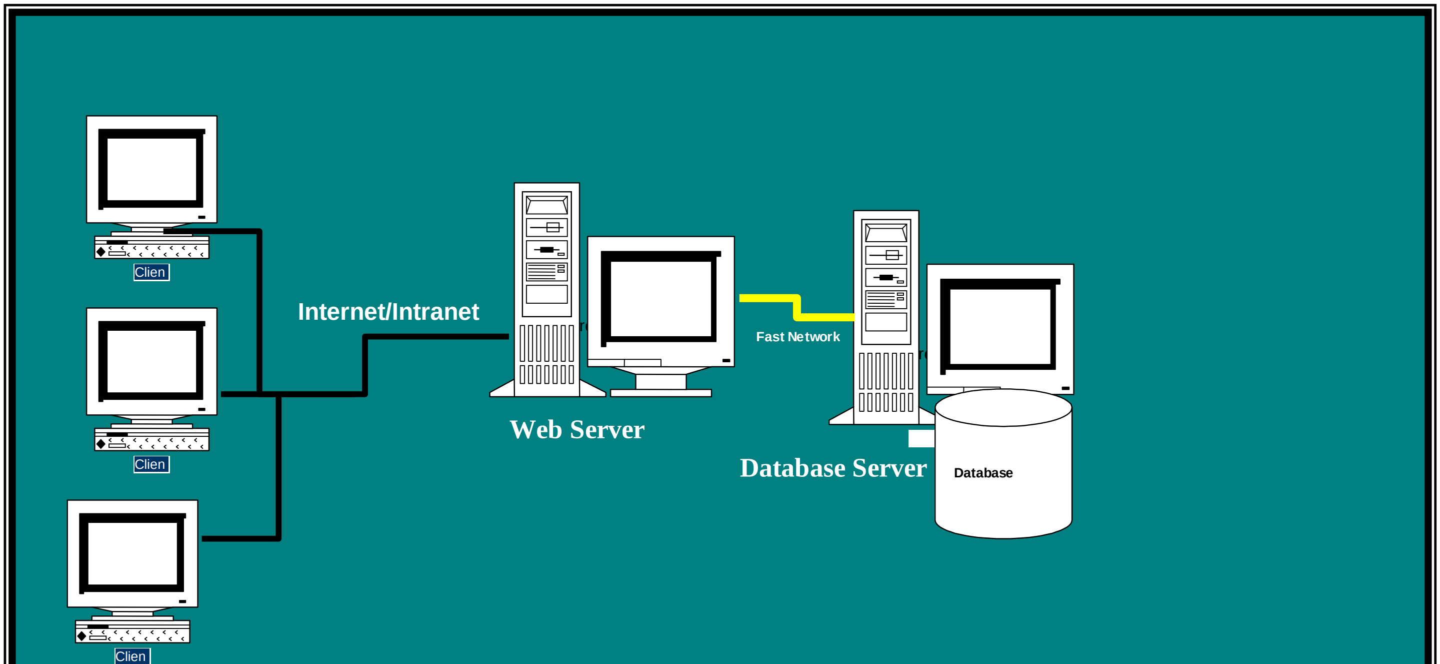
- There is only one client program, or program which runs on the client machine, which is the Browser application such as *Microsoft Internet Explorer* and *Netscape Navigator*.
- This means, programmers don't have to worry about writing a complex client program and all it's interaction with the operating system etc. The programmer only concern is writing HTML or other server side code to solve the problem at hand, not other Operating System related programming.
- Programming in HTML is easier; although for complex processing scripting languages are used such as VBScript, JavaScript, ASP.NET, CGI scripting etc (More on this in later). But writing complex combinations of HTML & scripting language is still easier than having to write an entire client application.
- Also, there is not installation of client application involved since there is only one client application, the Browser!

Web Server:

- The client code, actually resides on a Web Server which stores the actual HTML code that are requested by the Browser.
- The job of the web server is to provide the browsers with the requested HTLM pages. It is simply an HTML page distributor.
- Writing the basic client program means writing HTML code which will then reside on a Web Server and the server distributes the pages as requested by the client browsers.

Database Server:

- In the web server architecture, database servers are also used to store data requested by the clients.



Characteristics of Web Architecture

- We don't need Business Objects or Object-Oriented Programming to write an application on the Web Architecture. A matter of fact, originally, web applications were simply an HTML page with text/graphics that is loaded from the server to the client browser. The browser simply translated and displayed the text/graphics.
- But this limits how powerful a Web App can be, since there is **no processing** on the client, applications cannot perform any processing functionalities and are not powerful. To resolve this problem computer scientist developed the following technologies:

  - **Client- Side Scripting:** Browser built-in *Interpreters* (Translators) for scripting languages such as ***VBScript*** and ***JavaScript*** for client side processing.
    - **Advantage:**
      - Processing on the client
      - Web application can do business logic and programs can be powerful
    - **Disadvantage:**
      - Learn new language, VBScript or JavaScript
      - Powerful workstation to handle the processing

  - **Server-Side Scripting:** Web Server Built-in *Interpreters* (Translators) and Technology such as ***Active-Server Pages*** (uses VBScript or JavaScript) and ***CGI Script*** (can use VBScript, JavaScript, C/C++, Pearl & others).
    - **Advantage:**
      - No Processing on the Client
      - Less powerful Client computer
    - **Disadvantage:**
      - Learn new language, VBScript or JavaScript
      - And, learn a Active Server Pages or worst CGI Script

- These days the Server-Side technology has been in the fore front of the Web development industry. The advantages of being able to have as little processing in the client and allowing the important processing to be done at the server where is closer to the database has made it appealing to write most web applications using Server-Side Scripting.
- Technologies like Active Server Pages & Java Server Side Scripting has grown in popularity

Internet/Intranet

- The Web-Based Client/Server configuration is the standard for the ***Internet***.
- The Internet is a combination of thousands or Web-Based Client/Server architecture all connected together.
- Every time you log on to the Web, you are using a Web Browser to connect to a Web Server that has an address such as Microsoft.com that you enter in the Web Brower.
- The Web-Based architecture has become so popular that industry & governments are now adopting this architecture to develop their internal business applications.
- This lead to the terminology ***Intranet***.
- ***Intranet*** refers to a Web-Based Client/Server architecture that is private or internal to an organization. It is not public like the Internet.

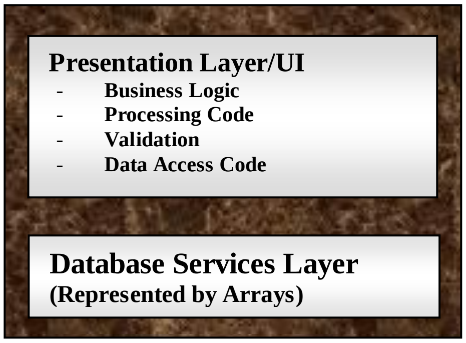
2.0 Web Application Architectures

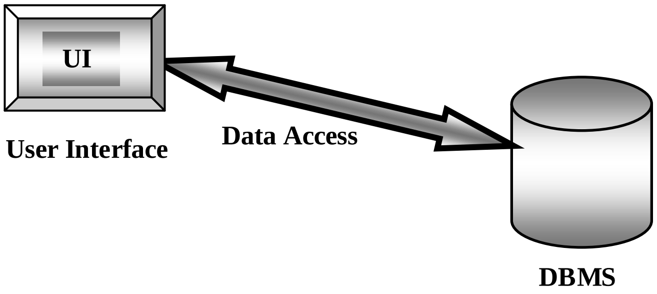
2.1 Overview

- In this section I will briefly provide an overview of the types of Web applications usually created.
- We will discuss applications that use where all the processing is done on the Web Forms and Object-Oriented architectures such as multitier object-oriented architecture we have been implementing with our Windows Applications

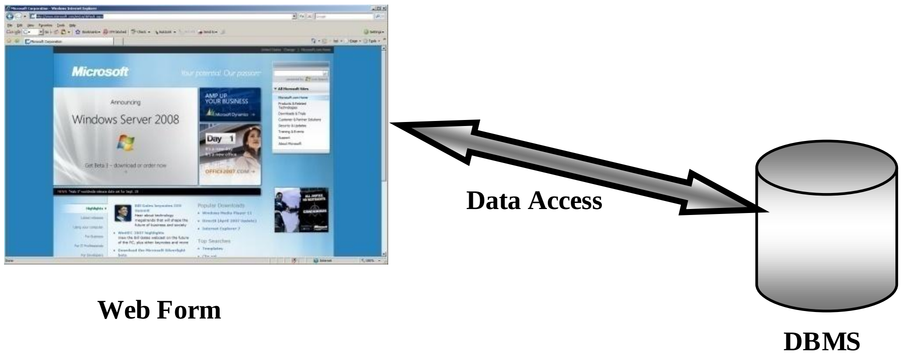
2.2 Standard Web Client/Server Applications Architecture

- From the beginning of CS608 I have been preaching the OOP model for creating applications.
- I stated that creating application s the conventional way of putting PROCESSING code on the FORMS is not what we want to do.
- This architecture & diagram looks as follows:



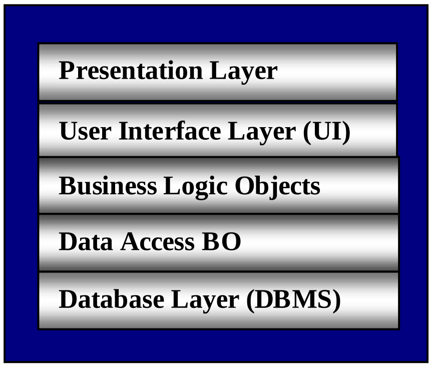


- With ASP.NET this approach is actually quite popular. Most books and courses on ASP.NET will simply focus on this time of Web Development
- Does this mean that we don't have OOP?
- Not really, since ASP.NET is a Object-Oriented environment, the WEB FORM, is an instance of an ASP.NET CLASS name PAGE. This PAGE CLASS contains all the properties and methods to create Web Pages. Some of these properties are ASP.NET OBJECTS.
- With that said, we have Object-Oriented Programming already embedded into the PAGE.
- With that said, we will approach learning ASP.NET and Web Development using a standard Web Form using this architecture at first.
- We will understand the Web Form, Web Controls, database connectivity etc., as follow:



2.3 Object-Oriented Client/Server Web Applications Architecture

- ❑ Although the previous architecture is suitable for situations where you need to create a Web Application fast and easy, large scale application required a better more scalable architecture.
- ❑ After we understand how ASP.NET web application work, then we will apply our SCALABLE, Client/Server application architecture model that we used for our WINDOWS APPLICATIONS:



Presentation Layer

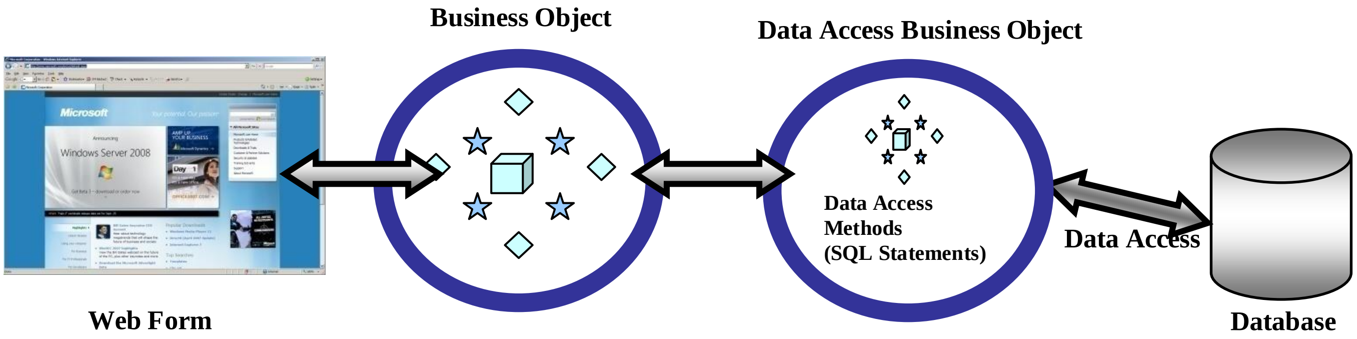
User Interface Layer (UI)

Business Logic Objects

Data Access BO

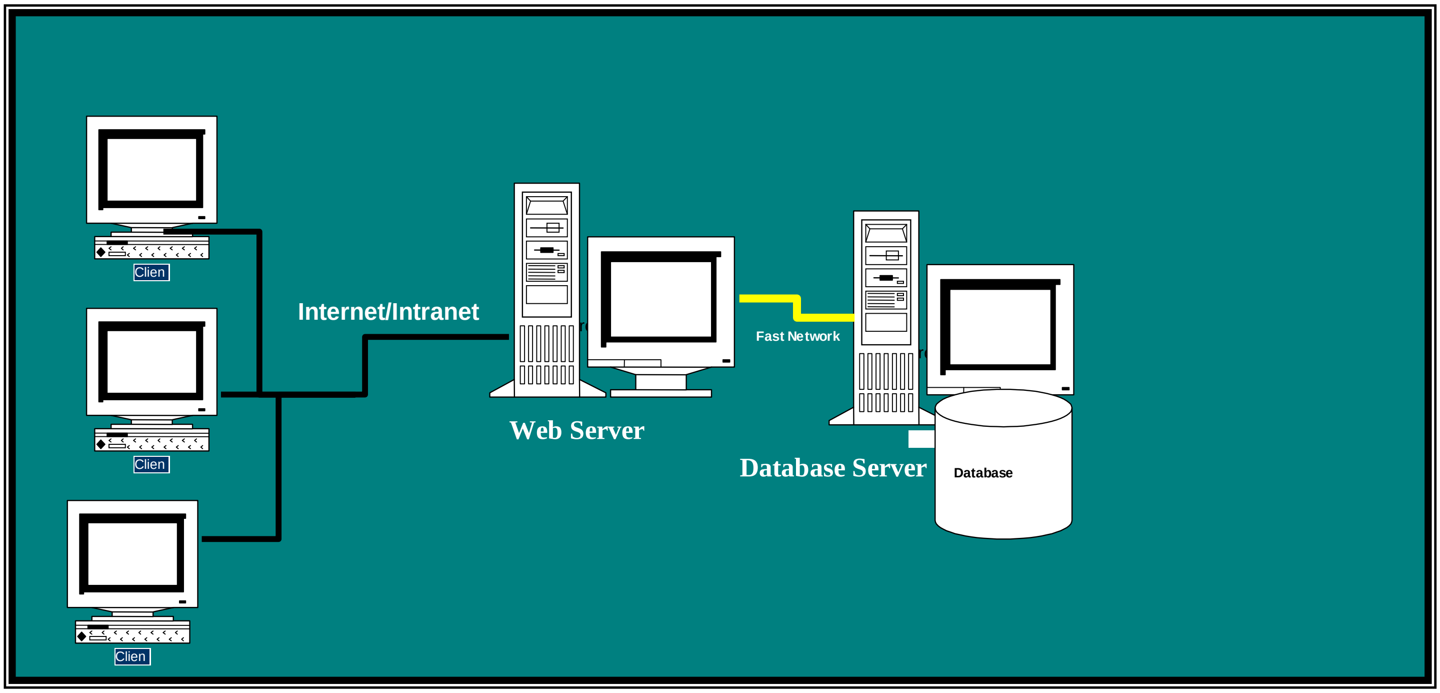
Database Layer (DBMS)

- ❑ Using this architecture we will encapsulate ALL PROCESSING, VALIDATION & DATA ACCESS into BUSINESS OBJECTS.
- ❑ This will allow us to create the following WEB application implementations:



Business Objects Placement in STANDARD Web Based Client/Server

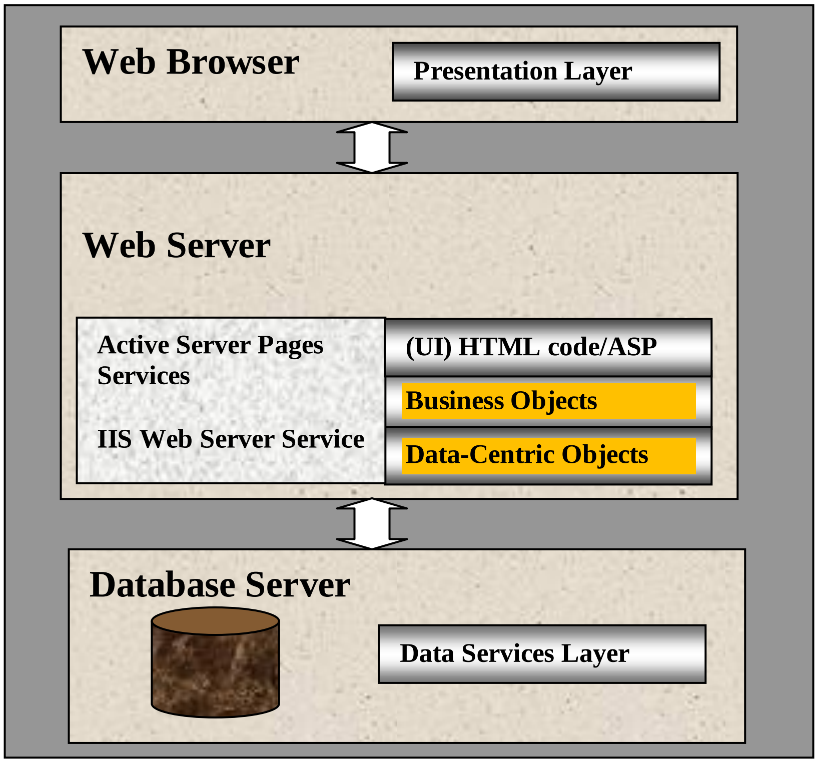
- A standard Web Client/Server architecture has the 3 basic Web components, Browser, Web Server & Database:



- Fitting our business Objects into the Web Architecture is done as follows:

1) Write the Business Objects in VB.NET, C#, C++, or Java.
2) Package the Business Objects in a ***Class Library or DLL COMPONENT***, enable **Remoting** & **Serialization**.
3) Have ASP.NET Server Side Technologies in the *Web Server* interact with the ***Business Objects*** in the ***DLL***.
4) Place the Business Objects as follows:

   - In a *Web-Based* configuration, the ***User-Interface (UI)*** is the Web Browser on the client, and the ***UI-Business Objects*** and the ***Data-Service Objects*** that handle database access resided in the Web Server. As usual, the ***Data Service layer*** or database itself resides on the Database Server.

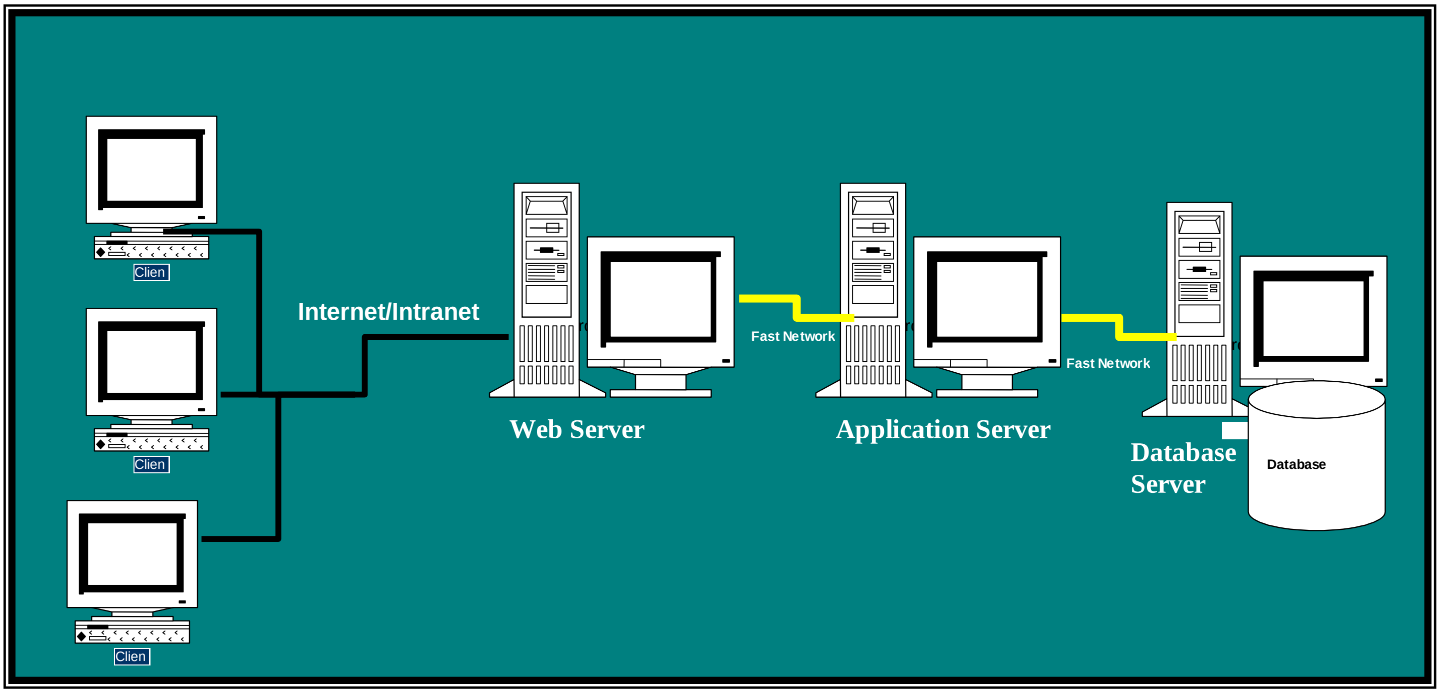


Business Objects Placement in N-TIER Web Based Client/Server

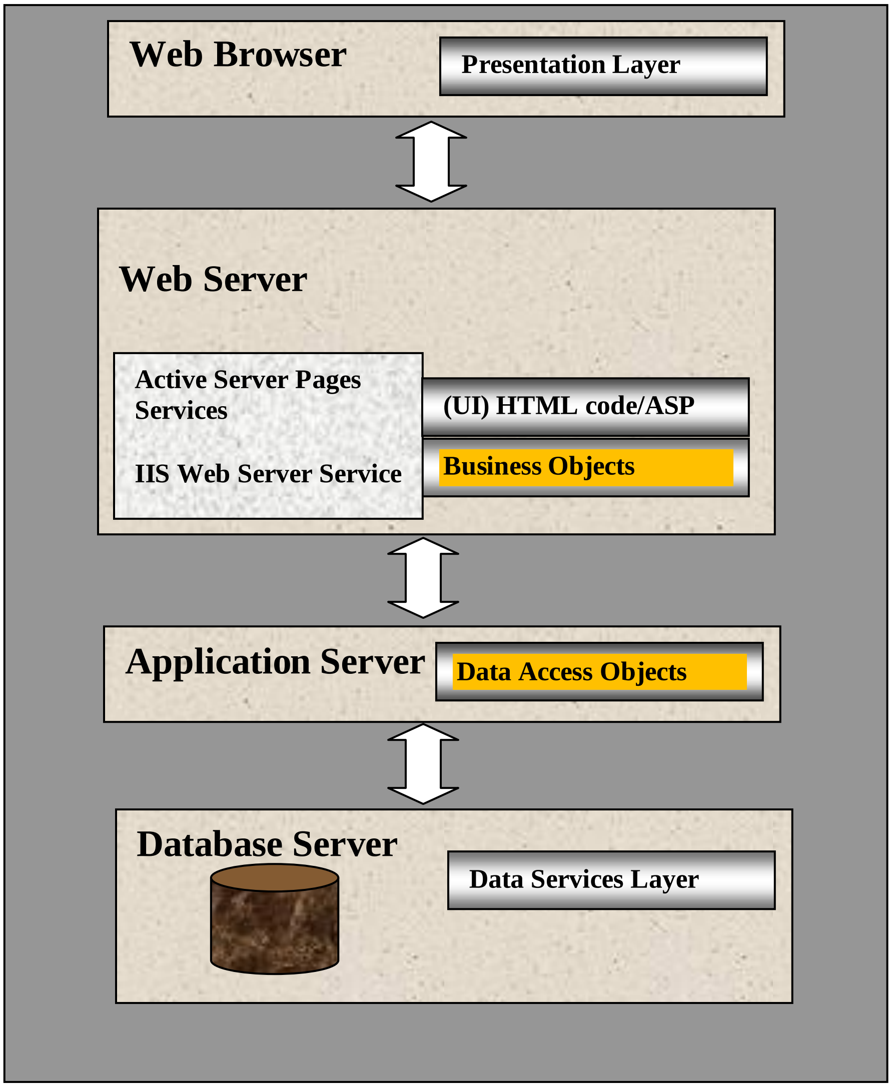
- ❑ As the demand for faster processing and the eCommerce growth, the Web-Based Client/Server architecture has evolved into a combination of the ***Three-Tier*** & ***Web-Based Client/Server*** to form the ***n-tier Web Based Client/Server Architecture***.
- ❑ In this architecture, the **Application Server** Tier is added, thus yielding all the benefits of managing faster connection, business object management etc.

Application Server:

- The application server is used to manage the connections and data access to the database servers.
- This server has all the advantages listed for the Three-Tier Client/Server configuration.
- You can add several Application Servers or Server Farms for load balancing.



- In this configuration, the ***User-Interface (UI)*** is the Web Browser on the client, and the User Interface Related Business Objects or ***UI-Centric Business Objects*** are kept at the Web Server closer to the HTML code.
- On the other hand, the ***Data-Centric Business Objects*** are placed and managed in the Application Server.
- As usual, the ***Data Service layer*** or database itself resides on the Database Server



3.0 Introduction to ASP.NET

3.1 ASP.NET

- In the infancy of web technology web pages were static
- Static web pages refer to HTML pages in which the content of the page is determined before the request is made. The output is always the same and never changes.
- The diagram below illustrates communication between browser and web server to generate static pages

3.2 Important Facts About ASP.NET

- In the infancy of web technology web pages were static
- Static web pages refer to HTML pages in which the content of the page is determined before the request is made. The output is always the same and never changes.

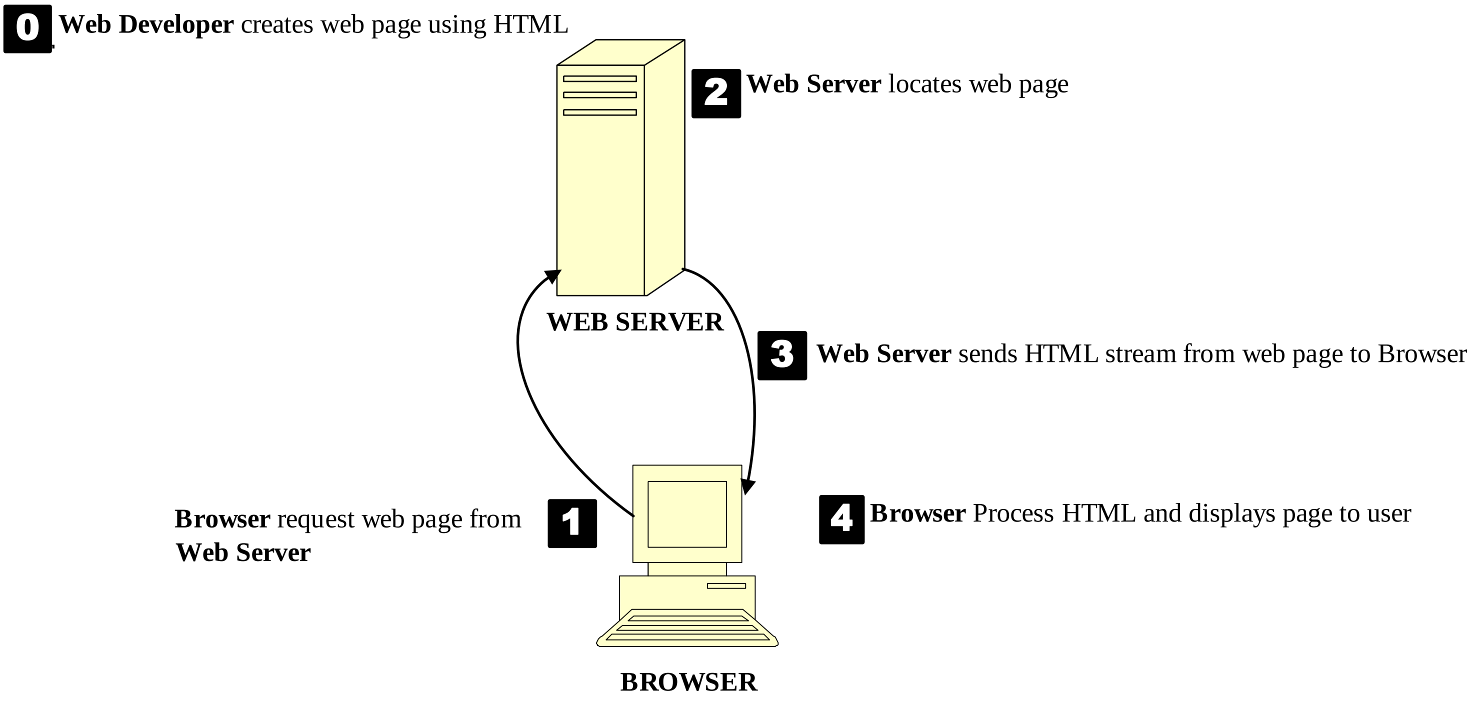
The diagram below illustrates communication

ASP.NET is Object-Oriented

- Development work can be done in terms of OBJECTS & EVENTS
- (Create object & Use them (Set & get proper…)

3.1 Static Web Pages

- In the infancy of web technology web pages were static
- Static web pages refer to HTML pages in which the content of the page is determined before the request is made. The output is always the same and never changes.
- The diagram below illustrates communication between browser and web server to generate static pages



0 **Web Developer** creates web page using HTML

1 **Browser** request web page from **Web Server**

2 **Web Server** locates web page

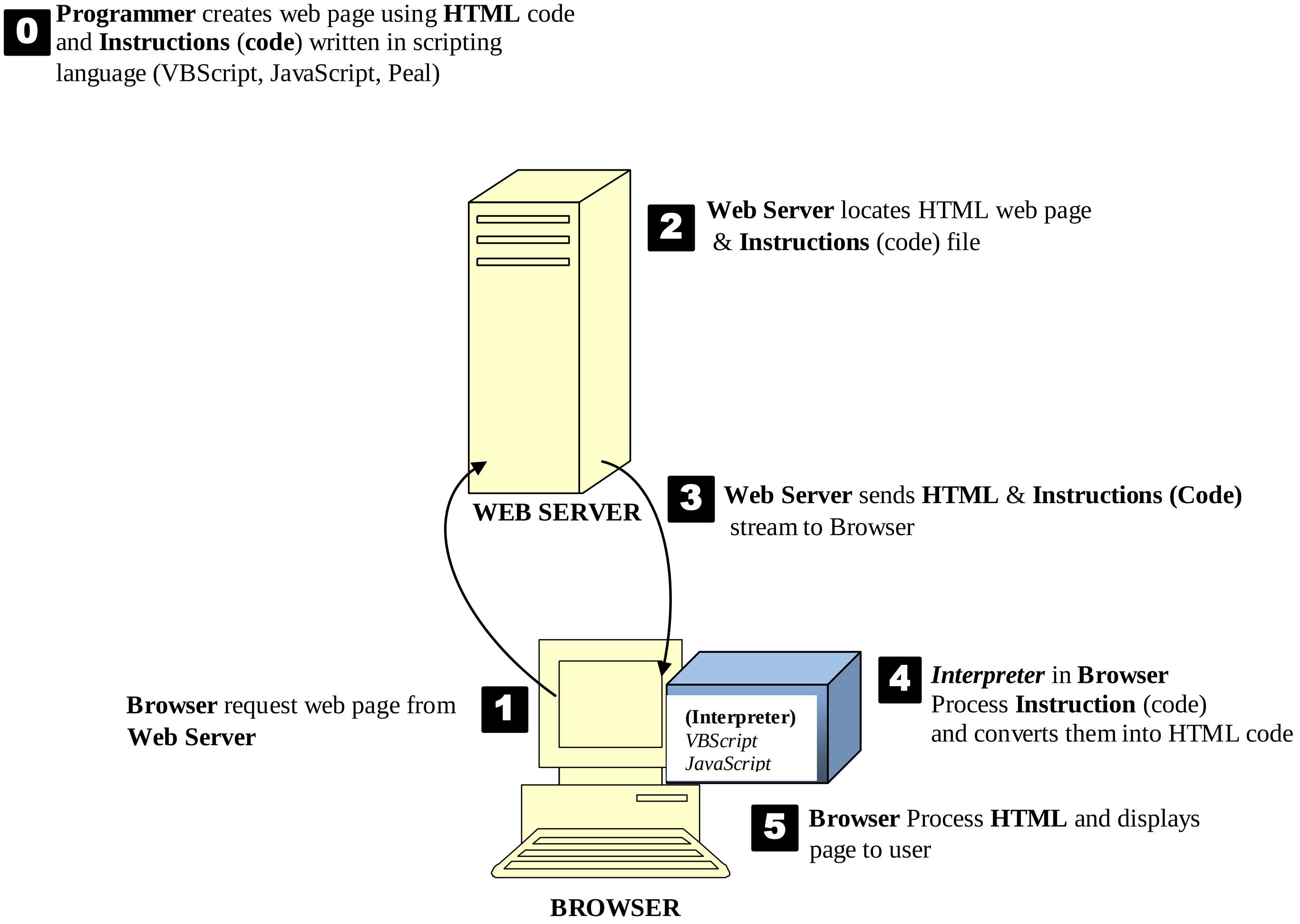
3 **Web Server** sends HTML stream from web page to Browser

4 **Browser** Process HTML and displays page to user

Advantages/Characteristics	Disadvantages
▪ Easy to program (HTML only) ▪ Fast, quick to copy small file over network. ▪	▪ Output always the same ▪ Content is completely determined before the page is requested. ▪ Cannot personalize the web page to the user profile ▪ No security ▪ Code is available to anyone who wishes to copy and use it for themselves ▪ In short, not ***dynamic*!**

3.2 Client-Side Dynamic Web Pages

- To work around the limitations of static web pages, technology was provided to the browser in the form of plug-ins, such as (Translators) for scripting languages such as ***VBScript*** and ***JavaScript*** for client side processing.
- This technology is known as Client-Side scripting and provided programmers with the ability to make the static web pages into dynamic web pages.
- The diagram below illustrates communication between browser and web server to generate client-side dynamic web pages:



0 **Programmer** creates web page using **HTML** code and **Instructions** (**code**) written in scripting language (VBScript, JavaScript, Peal)

1 **Browser** request web page from **Web Server**

2 **Web Server** locates HTML web page & **Instructions** (code) file

3 **Web Server** sends **HTML** & **Instructions (Code)** stream to Browser

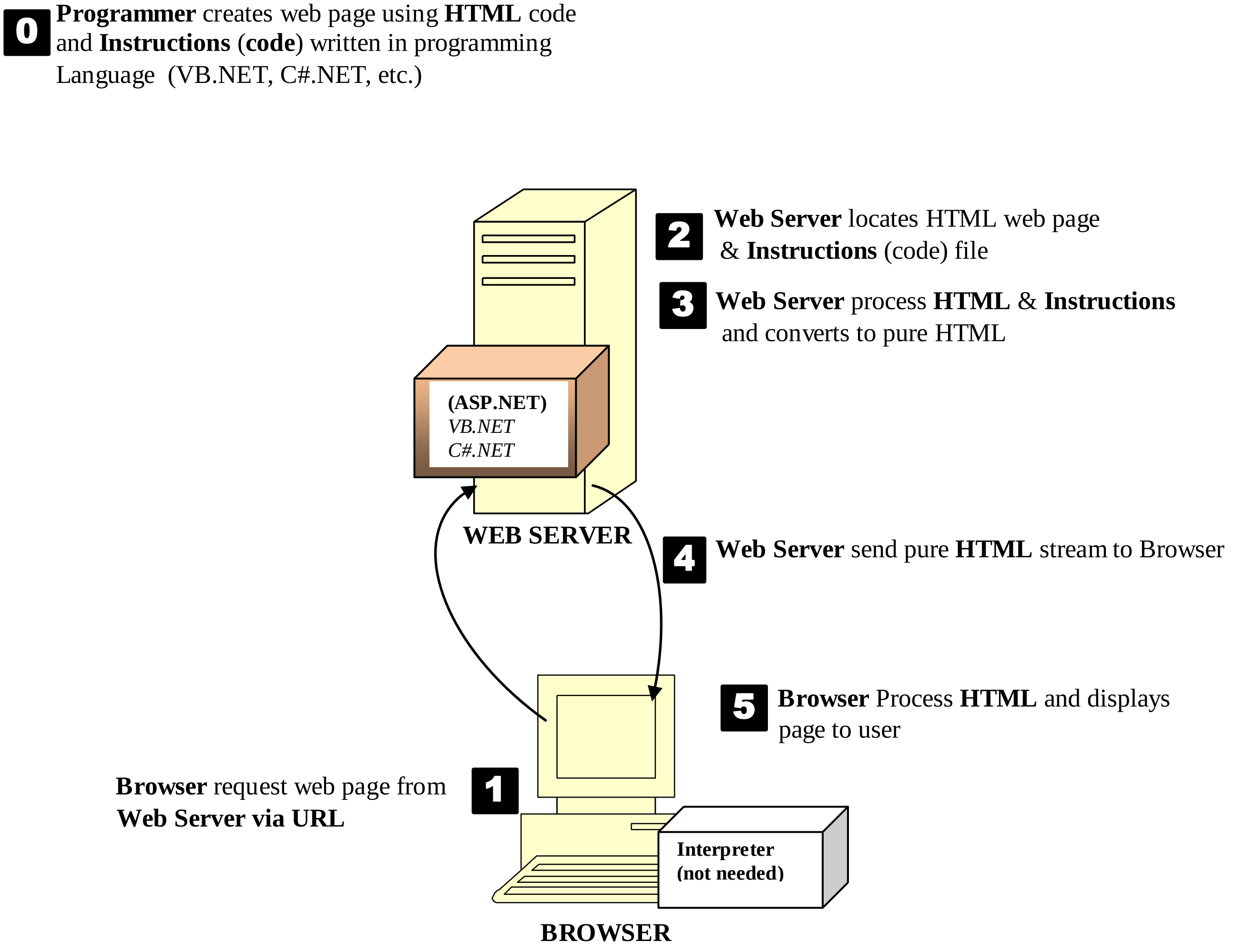
4 ***Interpreter*** in **Browser** Process **Instruction** (code) and converts them into HTML code

5 **Browser** Process **HTML** and displays page to user

Advantages/Characteristics	Disadvantages
▪ Generated dynamically upon request ▪ Output not the same, can be personalized, security etc.	▪ Requires plug-in or Interpreter to be installed, version control etc. ▪ Fat-client ▪ Difficult to program (Must learn VBScript or JavaScript) ▪ HTML & Instruction code sent to browser ▪ Network bandwidth limitations, take too long to download ▪ Each browser type interpret code differently, so you may need to program for different versions of browsers. ▪ Since interpreted in client, issues communication with using database server etc. ▪ Code not secured, can be seen in the client by anyone.

3.3 Server-Side Dynamic Web Pages

- Server side scripting provides the flexibility and dynamic requirements of the web-base client/server architecture.
- The diagram below illustrates communication between browser and web server to generate Server-side dynamic web pages:



0 **Programmer** creates web page using **HTML** code and **Instructions** (**code**) written in programming Language (VB.NET, C#.NET, etc.)

2 **Web Server** locates HTML web page & **Instructions** (code) file

3 **Web Server** process **HTML** & **Instructions** and converts to pure HTML

4 **Web Server** send pure **HTML** stream to Browser

5 **Browser** Process **HTML** and displays page to user

1 **Browser** request web page from **Web Server via URL**

Advantages/Characteristics	Disadvantages
▪ Generated dynamically upon request ▪ Server-side processing ▪ Output not the same, can be personalized, security etc. ▪ Only HTML sent to browser, browser type no longer an issue ▪ Fast, quick to copy small file over network. ▪ Original code is safe and secure in server; user cannot see the source code. ▪ Easier to communicate with database servers etc.	▪ Difficult to program (Must learn VBScript or JavaScript or VB.NET, or C#) ▪ Must understand ASP.NET, and .NET Framework.

3.x Designing a Web Page

- ❑ In this section we provide a brief overview of the process required to create and use WEB FORM.

Thinking Object-Oriented

- ❑ If you have been keeping up with my method of teaching programming, you should have the following in mind:

  - ▪ ASP.NET Web Forms are Objects, therefore our 3 step rule for Object-Oriented program still holds:

    1. Create Class
    2. Create Object
    3. User Object (Get/Set Properties, call method, program event-handlers etc.)

  - ▪ We can modify these rules to fit Web Development as follows;

    1. **Create Web Page Class** – When you create a WEB FORM, you are actually working with an inherited class from the Microsoft's PAGE CLASS, which is located in the ***`System.Web.UI.Page`*** Namespace

    2. **Create Object** – When the web application is running, an OBJECT of the WEB PAGE is created by ASP.NET.

    3. **User Object** – As you web page is running, you CODE WILL USE THE WEB FORM PAGE OBJECT by GETTING/SETTING PROPERTIES, calling METHODS, executing EVENT-HANDLERS etc.

User-Interface and Code (Graphical & Visual)

- ❑ As with WINDOWS APPLICATIONS, WEB APPLICATION have a GRAPHICAL aspect or what the user sees (Web Form), and there is the CODING portion which performs the processing and works behind the scenes.
- ❑ In WINDOWS APPLICATION, we only dealt with TWO aspects, GRAPHICAL AND CODE, in WEB FORMS, there are actually THREE :

  1. **GRAPHICAL** – What the user sees (Web Form, Controls etc.)

  2. **HTML SOURCE CODE** – What is being processed by the BROWSER

  3. **VB.NET or C# Code** – PROCESSING CODE EXECUTED ON THE SERVER. This code can be kept in a Code-behind file or embedded into the HTML code)

- ❑ To create a Web Page, you simply perform the following:

  1. Create a Web Application using Visual Studio 2005
  2. The application will contain a DEFAULT WEB FORM
  3. Design your interface by ADDING user-interface CONTROLS to the WEB FORM
  4. Write VB.NET or C# CODE to support the WEB FORM.
  5. Run the Web Application

4.0 Creating a Web Site Using Visual Studio IDE

4.1 Type of Web Sites available in Visual Studio 2005

- As previously discuss, a web application is composed of three major components:

1. Browser
2. Web Server
3. Database Server

- The second component is the WEB SERVER or the host which stores the web pages
- Visual Studio 2005 provides 3 types of Web Sites to HOST your WEB PAGES, these described as follows:

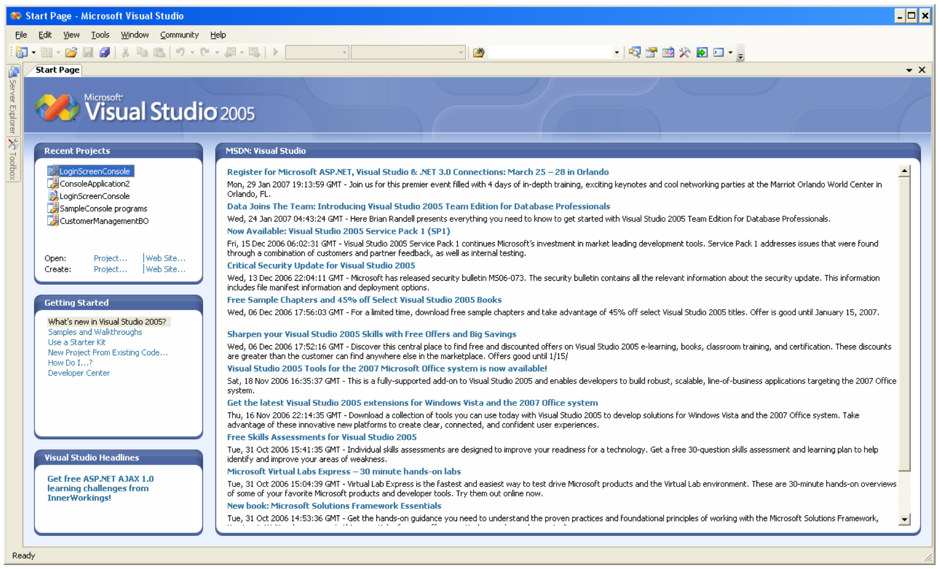
Web Sites	Description
File System	▪ WEB SITE & WEB PAGES are managed and executed under the ASP.NET Development WEB SERVER ▪ This Dev Web Server stores the entire site in a FILE ▪ Not managed by Windows Operating System ▪ This server is simple, flexible and ideal for class room and lab environment ▪ Since is in a file, it is portable.
HTTP	▪ This method uses an actual Production Web Server, such as Microsoft Internet Information Server (IIS) ▪ This is a professional-quality WEB SERVER, which requires security management, settings, permissions etc. ▪ This type of server is more complex to manage. ▪ Nevertheless, it already available in the Windows Operating System and ready to use ▪ This is NOT A FILE, IT IS NOT EASILY PORTABLE. A web site you create on your home computer would have to be recreated in another computer, say school lab etc.
FTP	▪ WEB SITE & WEB PAGES are stored in a REMOTE SERVER or REMOTE WEB SITE using FTP (File Transfer Protocol) ▪ Requires the username/password, server & port of the target machine. ▪ Visual Studio will actually create the Web Site in the remote server once the required credential and information is provided

4.2 Creating FILE SYSTEM WEB SITE using the IDE

- You should be familiar with using the IDE to create Windows Application. Now we follow the same road-map in order to create our web applications
- The steps to using the ***IDE*** to program our applications as follows:

Step 1: Open the *Visual Studio IDE* and invoke the *Start Page*:

- The screen looks as follows:


Start Page - Microsoft Visual Studio
File Edit View Tools Window Community Help
Server Explorer
Toolbox
Start Page
Microsoft Visual Studio 2005
Recent Projects
LoginScreenConsole
ConsoleApplication2
LoginScreenConsole
SampleConsole programs
CustomerManagementBO
Open: Project... | Web Site...
Create: Project... | Web Site...
Getting Started
What's new in Visual Studio 2005?
Samples and Walkthroughs
Use a Starter Kit
New Project From Existing Code...
How Do I...?
Developer Center
Visual Studio Headlines
Get free ASP.NET AJAX 1.0 learning challenges from InnerWorkings!
MSDN: Visual Studio
Ready

Register for Microsoft ASP.NET, Visual Studio & .NET 3.0 Connections: March 25 – 28 in Orlando
Mon, 29 Jan 2007 19:13:59 GMT - Join us for this premier event filled with 4 days of in-depth training, exciting keynotes and cool networking parties at the Marriot Orlando World Center in Orlando, FL.

Data Joins The Team: Introducing Visual Studio 2005 Team Edition for Database Professionals
Wed, 24 Jan 2007 04:43:24 GMT - Here Brian Randell presents everything you need to know to get started with Visual Studio 2005 Team Edition for Database Professionals.

Now Available: Visual Studio 2005 Service Pack 1 (SP1)
Fri, 15 Dec 2006 06:02:31 GMT - Visual Studio 2005 Service Pack 1 continues Microsoft's investment in market leading development tools. Service Pack 1 addresses issues that were found through a combination of customers and partner feedback, as well as internal testing.

Critical Security Update for Visual Studio 2005
Wed, 13 Dec 2006 22:04:11 GMT - Microsoft has released security bulletin MS06-073. The security bulletin contains all the relevant information about the security update. This information includes file manifest information and deployment options.

Free Sample Chapters and 45% off Select Visual Studio 2005 Books
Wed, 06 Dec 2006 17:56:03 GMT - For a limited time, download free sample chapters and take advantage of 45% off select Visual Studio 2005 titles. Offer is good until January 15, 2007.

Sharpen your Visual Studio 2005 Skills with Free Offers and Big Savings
Wed, 06 Dec 2006 17:52:16 GMT - Discover this central place to find free and discounted offers on Visual Studio 2005 e-learning, books, classroom training, and certification. These discounts are greater than the customer can find anywhere else in the marketplace. Offers good until 1/15/

Visual Studio 2005 Tools for the 2007 Microsoft Office system is now available!
Sat, 18 Nov 2006 16:35:37 GMT - This is a fully-supported add-on to Visual Studio 2005 and enables developers to build robust, scalable, line-of-business applications targeting the 2007 Office system.

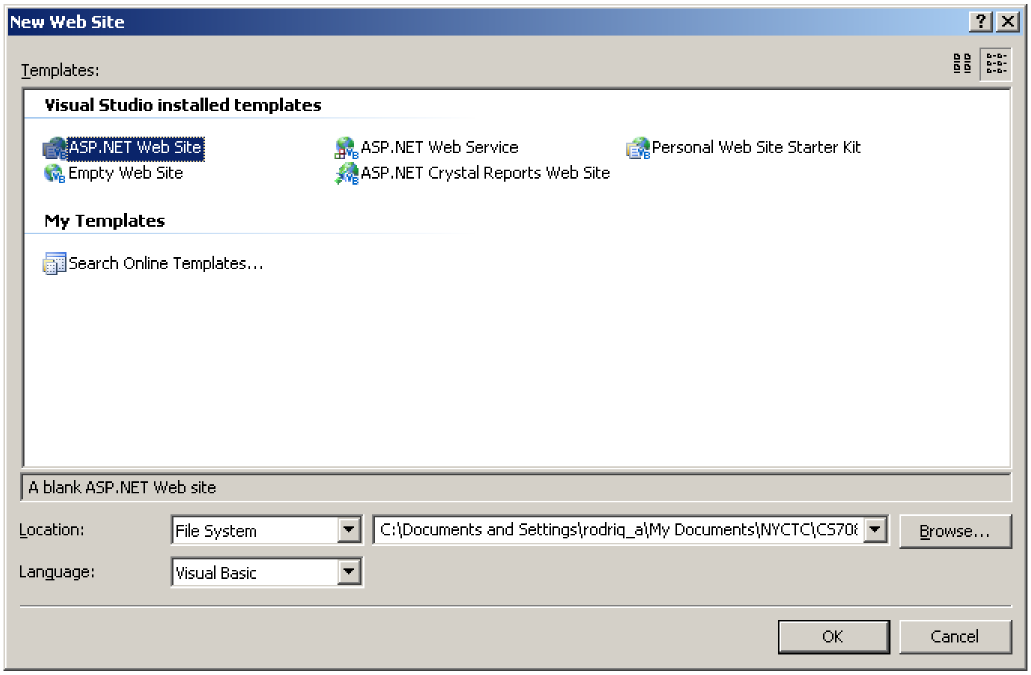
Get the latest Visual Studio 2005 extensions for Windows Vista and the 2007 Office system
Thu, 16 Nov 2006 22:14:35 GMT - Download a collection of tools you can use today with Visual Studio 2005 to develop solutions for Windows Vista and the 2007 Office system. Take advantage of these innovative new platforms to create clear, connected, and confident user experiences.

Free Skills Assessments for Visual Studio 2005
Tue, 31 Oct 2006 15:41:35 GMT - Individual skills assessments are designed to improve your readiness for a technology. Get a free 30-question skills assessment and learning plan to help identify and improve your areas of weakness.

Microsoft Virtual Labs Express – 30 minute hands-on labs
Tue, 31 Oct 2006 15:04:39 GMT - Virtual Lab Express is the fastest and easiest way to test drive Microsoft products and the Virtual Lab environment. These are 30-minute hands-on overviews of some of your favorite Microsoft products and developer tools. Try them out online now.

New book: Microsoft Solutions Framework Essentials
Tue, 31 Oct 2006 14:53:36 GMT - Get the hands-on guidance you need to understand the proven practices and foundational principles of working with the Microsoft Solutions Framework,

- The "New Web Site screen will display as follows:


New Web Site
Templates:
Visual Studio installed templates
ASP.NET Web Site
ASP.NET Web Service
Personal Web Site Starter Kit
Empty Web Site
ASP.NET Crystal Reports Web Site
My Templates
Search Online Templates...
Browse...
OK
Cancel

A blank ASP.NET Web site

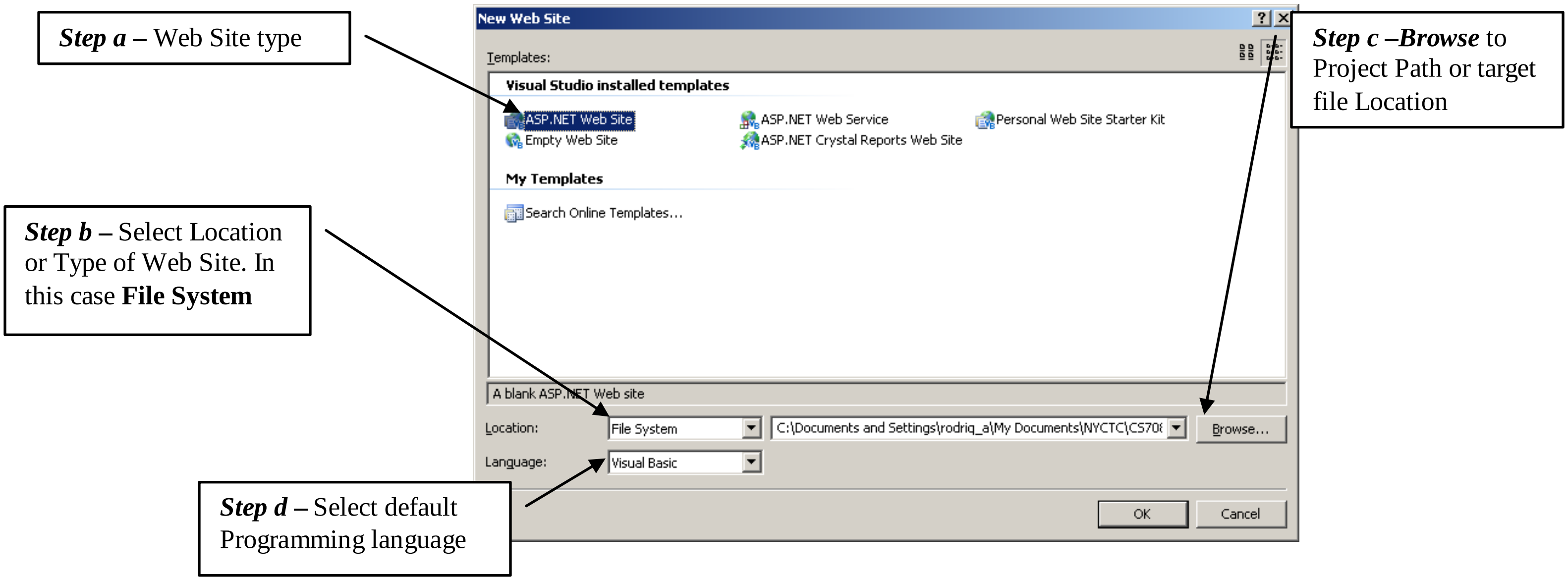
Location: File System | C:\Documents and Settings\rodriq_a\My Documents\NYCTC\CS708

Language: Visual Basic

Step 2: In the New Web Site Dialog Create the Web Site & Location

- To create your Web Site follow these steps:

  a). In the *Templates* box select: ***"ASP.NET Web Site"***
  b). In the *Template* box select: ***"File System***"
  c). Click ***"Browse"*** button and select the path
  d). Select the DEFAULT Language you will use for the CODE. You can set a default, but can still use other language



Location: File System — C:\Documents and Settings\rodriq_a\My Documents\NYCTC\CS70(

Language: Visual Basic

Step a – Web Site type

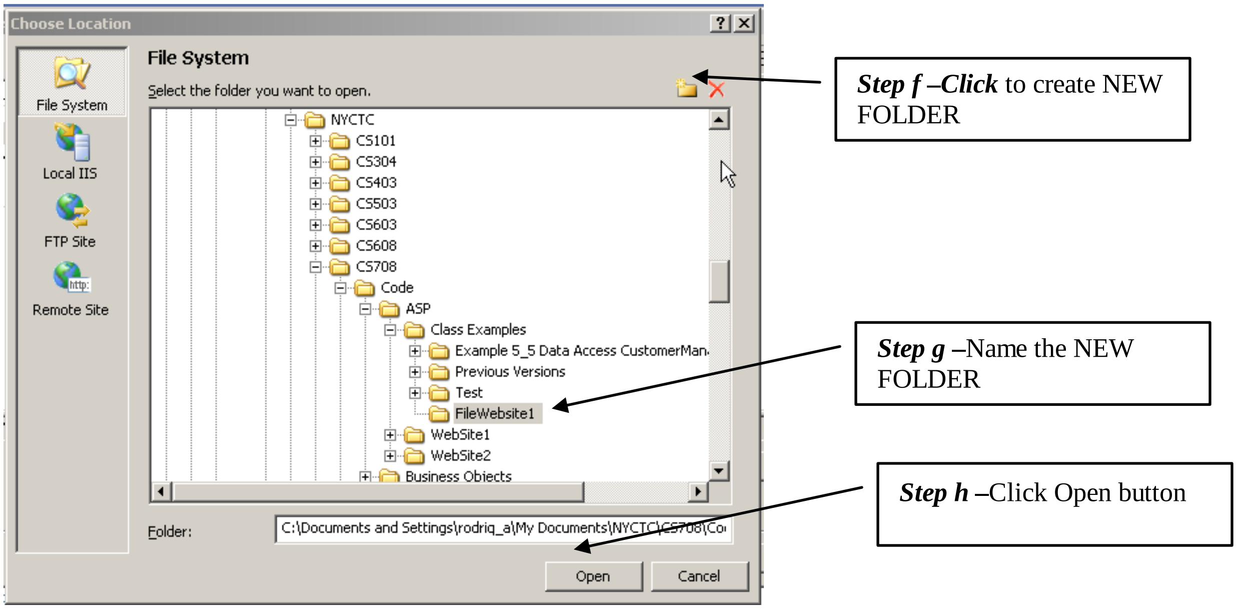
Step b – Select Location or Type of Web Site. In this case **File System**

Step c –Browse to Project Path or target file Location

Step d – Select default Programming language

- Clicking the Browse button will allow you to navigate and create a folder for your project

  e). Browse to the desired location
  f). Click the ***"Create New Folder ICON"*** on the top right-hand side of the '*Choose Location*" screen.
  g). Name the folder
  h). Click Open button to return to the "***New Web Site***" screen
  i). Click ***OK***



Folder: C:\Documents and Settings\rodriq_a\My Documents\NYCTC\CS708\Co

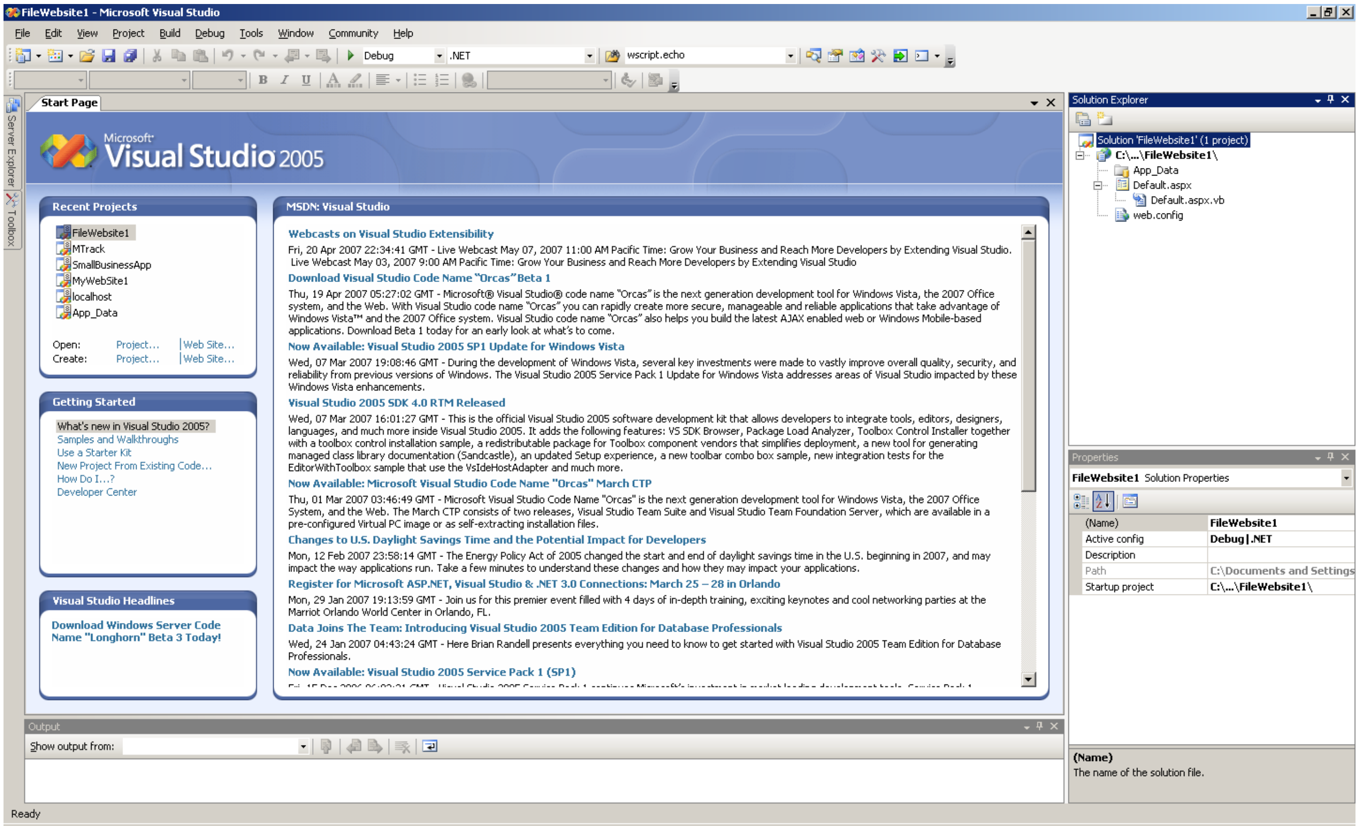
Step f –Click to create NEW FOLDER

Step g –Name the NEW FOLDER

Step h –Click Open button

Step 3: The IDE Main Screen.

- This screen is where you will create all your Web Form and develop your site.


FileWebsite1 - Microsoft Visual Studio
File Edit View Project Build Debug Tools Window Community Help
Debug
.NET
wscript.echo
Server Explorer
Toolbox
Start Page
Microsoft Visual Studio 2005
Recent Projects
FileWebsite1
MTrack
SmallBusinessApp
MyWebSite1
localhost
App_Data
Open: Project... | Web Site...
Create: Project... | Web Site...
Getting Started
What's new in Visual Studio 2005?
Samples and Walkthroughs
Use a Starter Kit
New Project From Existing Code...
How Do I...?
Developer Center
Visual Studio Headlines
Download Windows Server Code Name "Longhorn" Beta 3 Today!
MSDN: Visual Studio
Solution Explorer
Solution 'FileWebsite1' (1 project)
C:\...\FileWebsite1\
App_Data
Default.aspx
Default.aspx.vb
web.config
Properties
FileWebsite1 Solution Properties
Output
Show output from:
Ready

Webcasts on Visual Studio Extensibility

Fri, 20 Apr 2007 22:34:41 GMT - Live Webcast May 07, 2007 11:00 AM Pacific Time: Grow Your Business and Reach More Developers by Extending Visual Studio. Live Webcast May 03, 2007 9:00 AM Pacific Time: Grow Your Business and Reach More Developers by Extending Visual Studio

Download Visual Studio Code Name "Orcas" Beta 1

Thu, 19 Apr 2007 05:27:02 GMT - Microsoft® Visual Studio® code name "Orcas" is the next generation development tool for Windows Vista, the 2007 Office system, and the Web. With Visual Studio code name "Orcas" you can rapidly create more secure, manageable and reliable applications that take advantage of Windows Vista™ and the 2007 Office system. Visual Studio code name "Orcas" also helps you build the latest AJAX enabled web or Windows Mobile-based applications. Download Beta 1 today for an early look at what's to come.

Now Available: Visual Studio 2005 SP1 Update for Windows Vista

Wed, 07 Mar 2007 19:08:46 GMT - During the development of Windows Vista, several key investments were made to vastly improve overall quality, security, and reliability from previous versions of Windows. The Visual Studio 2005 Service Pack 1 Update for Windows Vista addresses areas of Visual Studio impacted by these Windows Vista enhancements.

Visual Studio 2005 SDK 4.0 RTM Released

Wed, 07 Mar 2007 16:01:27 GMT - This is the official Visual Studio 2005 software development kit that allows developers to integrate tools, editors, designers, languages, and much more inside Visual Studio 2005. It adds the following features: VS SDK Browser, Package Load Analyzer, Toolbox Control Installer together with a toolbox control installation sample, a redistributable package for Toolbox component vendors that simplifies deployment, a new tool for generating managed class library documentation (Sandcastle), an updated Setup experience, a new toolbar combo box sample, new integration tests for the EditorWithToolbox sample that use the VsIdeHostAdapter and much more.

Now Available: Microsoft Visual Studio Code Name "Orcas" March CTP

Thu, 01 Mar 2007 03:46:49 GMT - Microsoft Visual Studio Code Name "Orcas" is the next generation development tool for Windows Vista, the 2007 Office System, and the Web. The March CTP consists of two releases, Visual Studio Team Suite and Visual Studio Team Foundation Server, which are available in a pre-configured Virtual PC image or as self-extracting installation files.

Changes to U.S. Daylight Savings Time and the Potential Impact for Developers

Mon, 12 Feb 2007 23:58:14 GMT - The Energy Policy Act of 2005 changed the start and end of daylight savings time in the U.S. beginning in 2007, and may impact the way applications run. Take a few minutes to understand these changes and how they may impact your applications.

Register for Microsoft ASP.NET, Visual Studio & .NET 3.0 Connections: March 25 – 28 in Orlando

Mon, 29 Jan 2007 19:13:59 GMT - Join us for this premier event filled with 4 days of in-depth training, exciting keynotes and cool networking parties at the Marriot Orlando World Center in Orlando, FL.

Data Joins The Team: Introducing Visual Studio 2005 Team Edition for Database Professionals

Wed, 24 Jan 2007 04:43:24 GMT - Here Brian Randell presents everything you need to know to get started with Visual Studio 2005 Team Edition for Database Professionals.

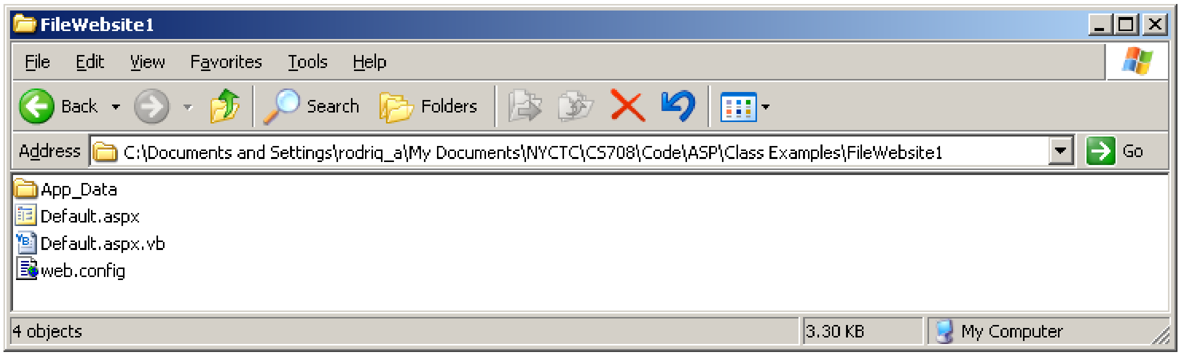
Now Available: Visual Studio 2005 Service Pack 1 (SP1)

(Name)	FileWebsite1
Active config	Debug | .NET
Description	
Path	C:\Documents and Settings
Startup project	C:\...\FileWebsite1\

(Name)
The name of the solution file.

Web Site Local File Structure

- ❑ Our Website looks as follows in the file system:



Address: C:\Documents and Settings\rodriq_a\My Documents\NYCTC\CS708\Code\ASP\Class Examples\FileWebsite1

App_Data
Default.aspx
Default.aspx.vb
web.config

4 objects | 3.30 KB

4.3 Visual Studio 2005 IDE Web Components

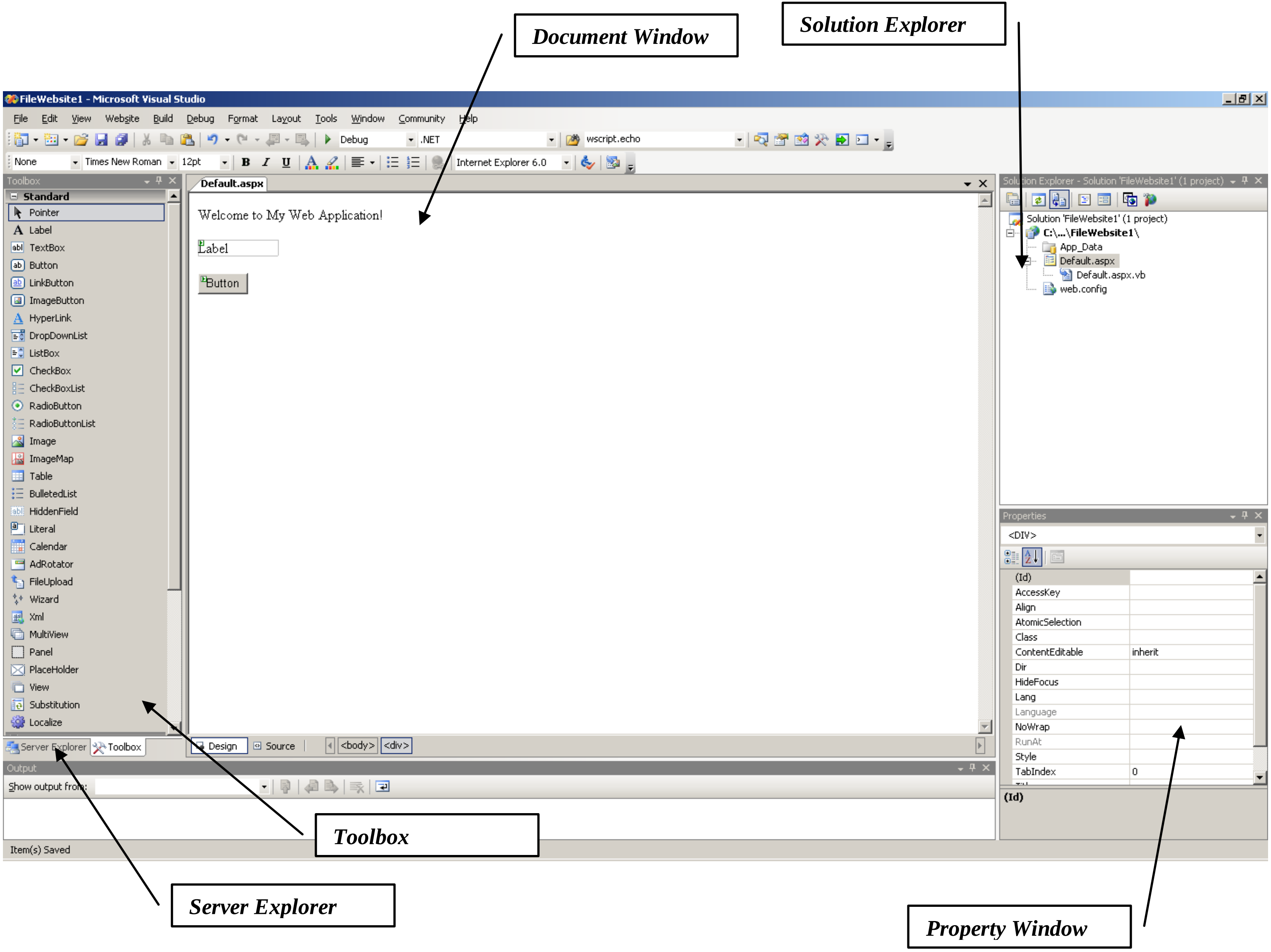
- In this section we take a look at the components available via Visual Studio 2005 to create Web Applications.
- You will notice that some of these components are similar to the Windows Applications you created before.

Main Web Application Components

- This screen is composed some basic Window items such as ***Title bar***, ***Menu bar***, ***Menu Toolbars*** and a ***Status Bar***, like most windows application.
- In addition the same application structure that are important for WINDOWS APPLICATIONS ARE USED FOR WEB APPLICATIONS such as

  - ***Document Window*** – Area where Web Forms is designed
  - ***Solution Explorer Window*** – List the FILES, FOLDERS, WEB FORMS, CLASSES etc., in the Web Application
  - ***Properties Window*** – Allows you to SET properties for the Objects (elements) in the Solution Explorer of a Control on a Web Form in the document window.
  - ***Toolbox*** – Contains the graphical controls to design and build your web pages
  - ***Server Explorer*** – Allows you to manage other services on the local computer or network, such as SQL Databases, Event logs, Services, etc.

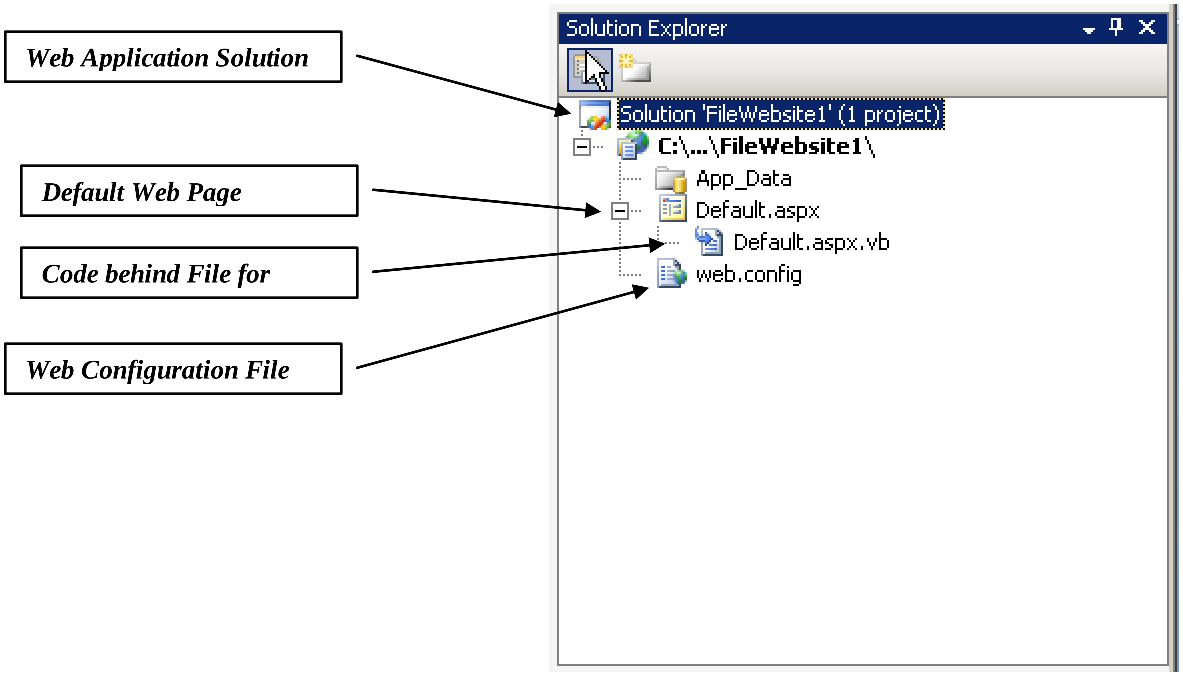
- The major components are shown below:



(Id)	
AccessKey	
Align	
AtomicSelection	
Class	
ContentEditable	inherit
Dir	
HideFocus	
Lang	
Language	
NoWrap	
RunAt	
Style	
TabIndex	0

Solution Explorer & Main Application Files

- The Solution Explore contains a listing of all the files, web forms or page and objects which make up our applications:



Main Application Files

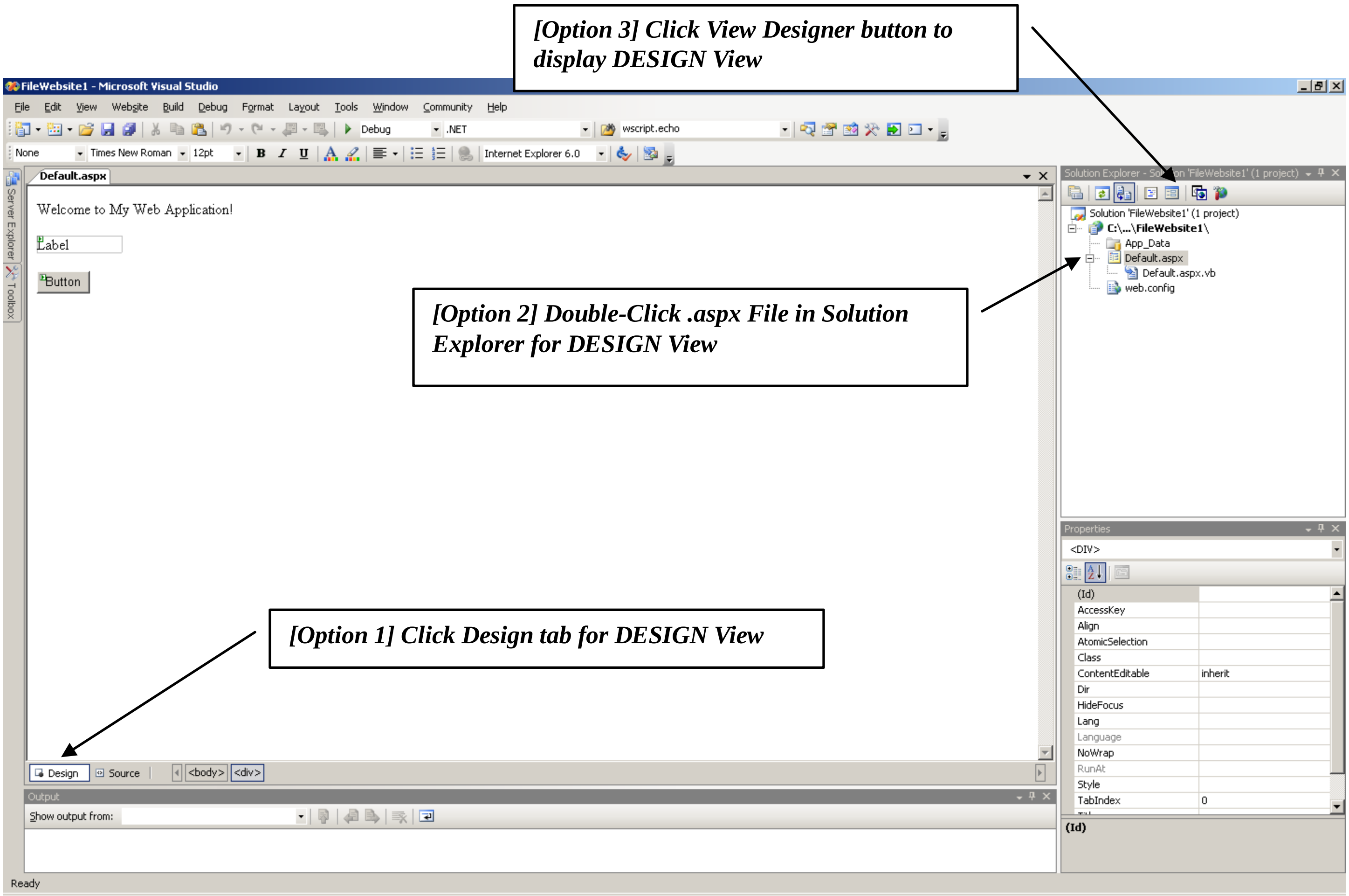
- There are several main application files we need to discuss:

File	Extension	Description
Default.aspx	.aspx	▪ This is the DEFAULT WEB PAGE AUTOMATICALLY ADDED TO YOUR APPLICATION
Default.aspx.vb	.aspx.vb	▪ This is the CODE-BEHIND file automatically created for the Default.aspx page ▪ Every Web Form will contain an associated code-behind file with .vb extension
Web.config	.config	▪ XML-configuration file for the entire application. ▪ Here you configure settings for the following: - Database connectivity (connection string) - Customizing security - State management - Memory management etc. ▪ This file may not be available when you create a web application, if so, created manually
Global.asax	.asax	▪ Global application file. ▪ Here you can declare global variables, etc. ▪ This file includes GLOBAL EVENT-HANDLERS, which you can use to execute code for global events such as: - Application_Start() – Executes when application starts - Application_End() – Executes just before application ends - Application_Error() – Executes when an unhandled exceptions occurs.

Document Window & Available Views

- The document window is the location where you will design and create your Web Form.
- Visual Studio provides server designers and views to work with the various FILES in the SOLUTION EXPLORER
- We will now discuss three important vies:

1. ***Design View*** – Visually design and create your WEB FORM using CONTROLS from the Toolbox.
    - There are three available methods to display Design View:

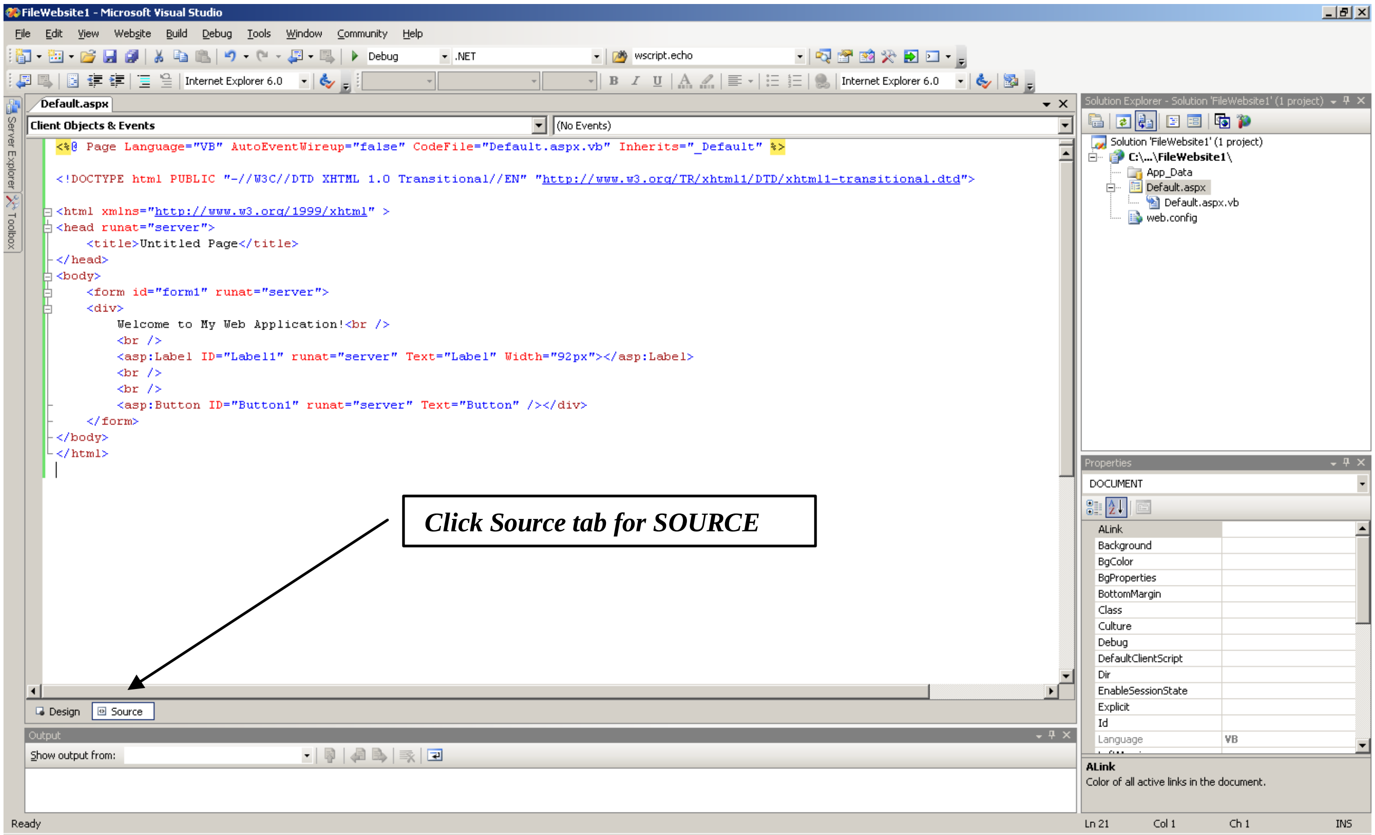


[Option 3] Click View Designer button to display DESIGN View

[Option 2] Double-Click .aspx File in Solution Explorer for DESIGN View

[Option 1] Click Design tab for DESIGN View

2. ***Source or HTML View*** – Directly edit your WEB FORM HTML code:



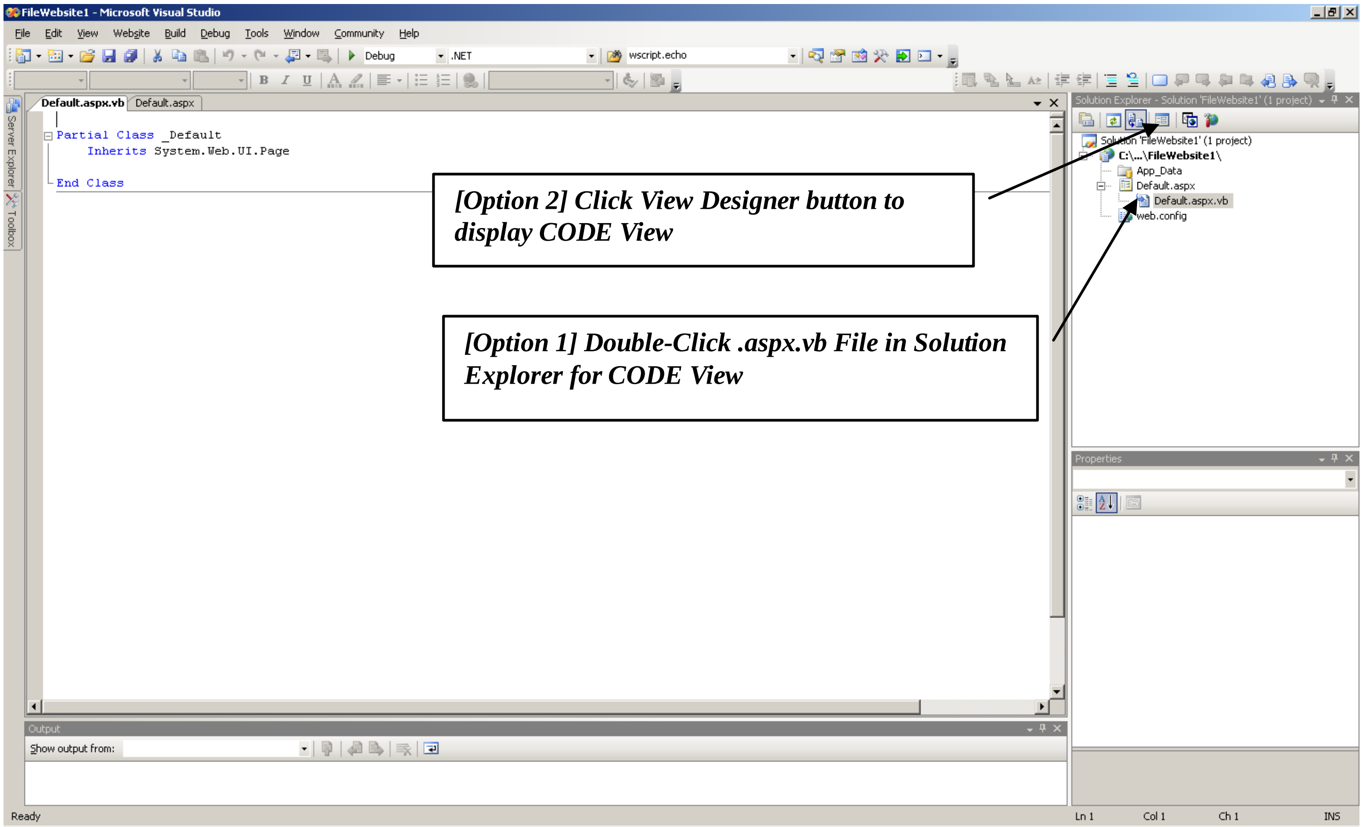
```
<%@ Page Language="VB" AutoEventWireup="false" CodeFile="Default.aspx.vb" Inherits="_Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Untitled Page</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        Welcome to My Web Application!<br />
        <br />
        <asp:Label ID="Label1" runat="server" Text="Label" Width="92px"></asp:Label>
        <br />
        <br />
        <asp:Button ID="Button1" runat="server" Text="Button" /></div>
    </form>
</body>
</html>
```

Click Source tab for SOURCE

3. ***Code Editor View (Code-behind View)*** – Editor where you will WRITE CODE (VB.NET, C#, etc.) to support the user-interface WEB FORM. This code is places in the CODE-BEHIND FILE with **aspx**.**vb** extension.


FileWebsite1 - Microsoft Visual Studio
File Edit View Website Build Debug Tools Window Community Help
Debug
.NET
wscript.echo
Default.aspx.vb
Default.aspx
Server Explorer
Toolbox
Solution Explorer - Solution 'FileWebsite1' (1 project)
Solution 'FileWebsite1' (1 project)
C:\...\FileWebsite1\
App_Data
Default.aspx
Default.aspx.vb
web.config
Properties
Output
Show output from:
Ready
Ln 1
Col 1
Ch 1
INS

```
Partial Class _Default
    Inherits System.Web.UI.Page

End Class
```

[Option 2] Click View Designer button to display CODE View

[Option 1] Double-Click .aspx.vb File in Solution Explorer for CODE View

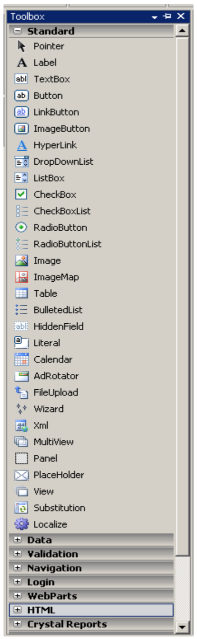
Toolbox

- The Toolbox contains the various types of controls for your application.
- Toolbox is divided into several sections, examples:

  - Standard: Contains all ASP.NET CONTROLS (Run on Server)

  - HTML: Contains the classic HTML controls (Run on client only)

  - Data: Contains CONTROLS to perform data access or connect to database.



Toolbox

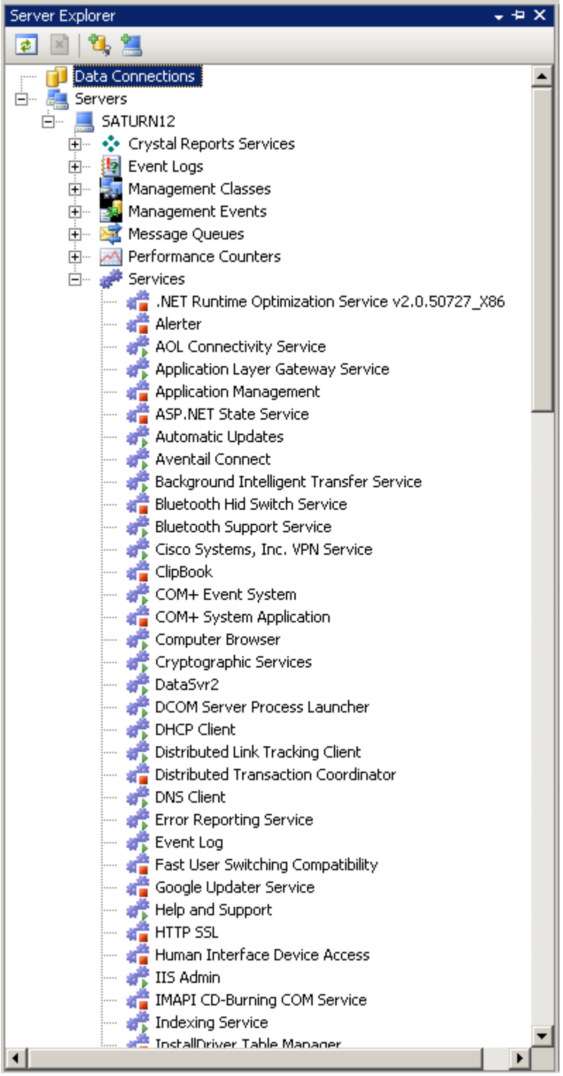
- Standard
  - Pointer
  - Label
  - TextBox
  - Button
  - LinkButton
  - ImageButton
  - HyperLink
  - DropDownList
  - ListBox
  - CheckBox
  - CheckBoxList
  - RadioButton
  - RadioButtonList
  - Image
  - ImageMap
  - Table
  - BulletedList
  - HiddenField
  - Literal
  - Calendar
  - AdRotator
  - FileUpload
  - Wizard
  - Xml
  - MultiView
  - Panel
  - PlaceHolder
  - View
  - Substitution
  - Localize
- Data
- Validation
- Navigation
- Login
- WebParts
- HTML
- Crystal Reports

Server Explorer

- Server Explorer Windows provides an interface for your application to connect to SERVICES in your local or network computer such as DATABASES, FILES, Services, etc.
- Server Explorer not only lets you have access to these services but can allow you to CONFIGURE & MODIFY the services.
- For example, you can do the following with an SQL Server database just as if you were doing it with the SQL Server Management Console or Enterprise Manager:

  - Create database
  - Connect to a database
  - Create tables, relationships etc
  - Create & Execute queries, stored procedures etc.

- In summary, you can work from your Visual Studio 2005 IDE to create your application and create & mange the DATABASE.



Server Explorer

- Data Connections
- Servers
  - SATURN12
    - Crystal Reports Services
    - Event Logs
    - Management Classes
    - Management Events
    - Message Queues
    - Performance Counters
    - Services
      - .NET Runtime Optimization Service v2.0.50727_X86
      - Alerter
      - AOL Connectivity Service
      - Application Layer Gateway Service
      - Application Management
      - ASP.NET State Service
      - Automatic Updates
      - Aventail Connect
      - Background Intelligent Transfer Service
      - Bluetooth Hid Switch Service
      - Bluetooth Support Service
      - Cisco Systems, Inc. VPN Service
      - ClipBook
      - COM+ Event System
      - COM+ System Application
      - Computer Browser
      - Cryptographic Services
      - DataSvr2
      - DCOM Server Process Launcher
      - DHCP Client
      - Distributed Link Tracking Client
      - Distributed Transaction Coordinator
      - DNS Client
      - Error Reporting Service
      - Event Log
      - Fast User Switching Compatibility
      - Google Updater Service
      - Help and Support
      - HTTP SSL
      - Human Interface Device Access
      - IIS Admin
      - IMAPI CD-Burning COM Service
      - Indexing Service
      - InstallDriver Table Manager

5.0 Web Form Life-Cycle

5.1 Important Facts about ASP.NET

- Now we will analyze one the most important aspects of ASP.NET
- Before we look into how to process a WEB PAGE, we need to understand some important properties of the ASP.NET technology and the communication rules between the BROWSER & the SERVER.

WEB FORM EXECUTE ON SERVER

- Web Form execute on Server only
- User performs the task in the BROWSER, but the CODE EXECUTES ON THE SERVER
- User actions happen on the client or BROWSER, PROCESSING of these actions happen on the WEB SERVER
- WEB FORMS ARE RE-CREATED WITH EVERY ROUND TRIP.

EVENT-DRIVEN & Server Side Execution

- ASP.NET is event-driven, everything in your web-page is executed in RESPONSE TO AN EVENT.
- Because EVENTS are initially requested in the BROWSER, but executed on the SERVER, there is a delay or lag in the process of triggering events and reacting to them.
- WEB APPLICATIONS are not as ROBUST AS WINDOWS APPLICATIONS when it comes to EVENT-DRIVEN PROGRAMMING.
- WEB APPLICATIONS actually SIMULATE EVENTS-DRIVEN PROGRAMMING due to the limitations of the WEB PAGE LIFE-CYCLE and a process called POSTBACK (More on this below)

WEB APPLICATIONS ARE STATELESS (VIEW STATE)

- Web Page lifetime only exist for ONE PAGE REQUEST
- WEB SERVER DISCARDS ALL WEB PAGE INFORMATION after rendering and sending to the BROWSER THE HTML PAGE
- DISCARDING SERVER RESOURCES after each request ensures that the WEB SERVER can support hundreds or thousands of REQUESTS WITHOUT BOGGING DOWN.

VIEW STATE & WEB CONTROLS

- The state or content of all Web Controls are stored or kept via ***VIEW STATE*** mechanism on the SERVER
- Form Data is stuffed into a HIDDEN PROPERTY, ENCRYPTED and SENT TO SERVER.
- View State enabled by default
- View State can be granular (Form-level or Control-level)
- VIEW STATE only works with WEB PAGE & WEB CONTROL PROPERTIES

Final Comments on VIEW STATE

- Using VIEW STATE mechanism is a great solution, because server resources are discarded and server is freed to handle other requests.
- Nevertheless, there is a price:
- Because form data or VIEW STATE is stored on the WEB FORM, it results in a LARGER WEB PAGE SIZE
- BROWSER receives a LARGE PAGE.
- BROWSER has to package & SEND THE HIDDEN-VIEW STATE BACK to SERVER with the NEXT POSTBACK
- Therefore with LARGE PAGES, it TAKES LONGER TO SEND/RECEIVE

POSTBACK

- PAGE and ALL USER-SUPPLIED INFORMATION IS SENT BACK TO SERVER when certain action in the BROWSER is performed
- Round trip from BROWSER to SERVER to BROWSER in response to a WEB FORM being submitted to Web Server(after the page has been displayed in browser for the first time.)
- All Data in controls are sent to server via View State
- Initiates when user clicks a BUTTON in the Client, or trigger a WEB CONTROL, example check box etc.

POSTBACK & PAGE OBJECT PROPERTY

- The PAGE CLASS, contains a BOOLEAN PROPERTY named **IsPostback**.
- This is a very important PROPERTY for programming in ASP.NET.
- The value of this property is as follows:

  - **FALSE:** NEW PAGE REQUEST (First time the Page is Loaded? First time page is displayed by browser
  - **TRUE:** Postback has occurred, Web Form has been sent to server, server has perform the required operation, Web Form sent back to Browser.

POSTBACK & WEB CONTROLS

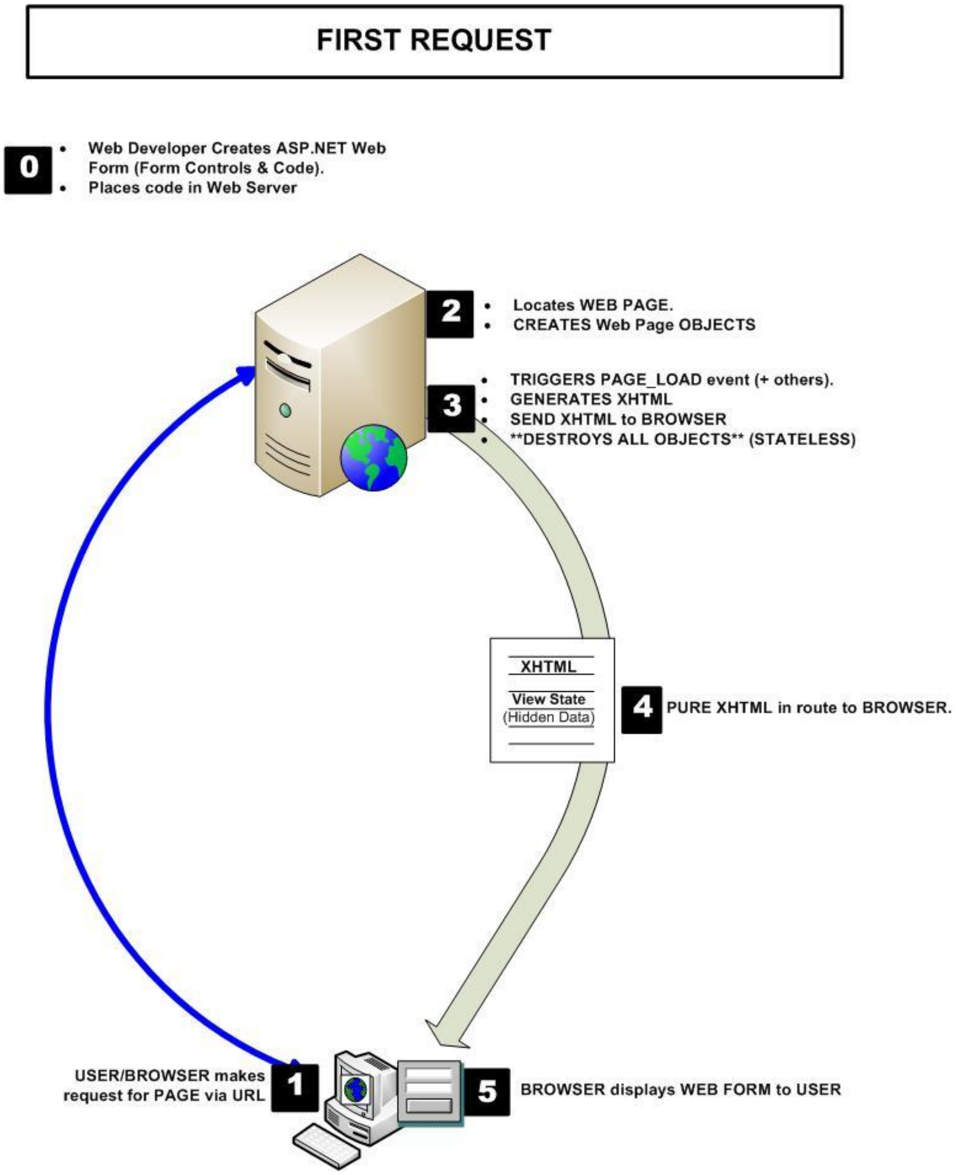
- All Web Controls have a PROPERTY named AUTOPOSTBACK. Set to FALSE by Default
- If this AutoPostBack is set to TRUE on a CONTROL, the control will initiate a POSTBACK.
- With this PROPERTY SET TO TRUE, changes to controls such as CheckBox, ListBox, TextBox, etc, can generate POSTBACK which will allow the WEB SERVER to execute EVENT-HANDLERS FOR THE CONTROLS.

5.2 ASP.NET Web Page (Web Form) Processing Life-cycle

- ❑ Now we will analyze one the most important aspects of ASP.NET

Processing Life-Cycle for First Request for a Web Form

- ❑ Let’s analyze in detail what happens when the USER makes the first request for a WEB FORM

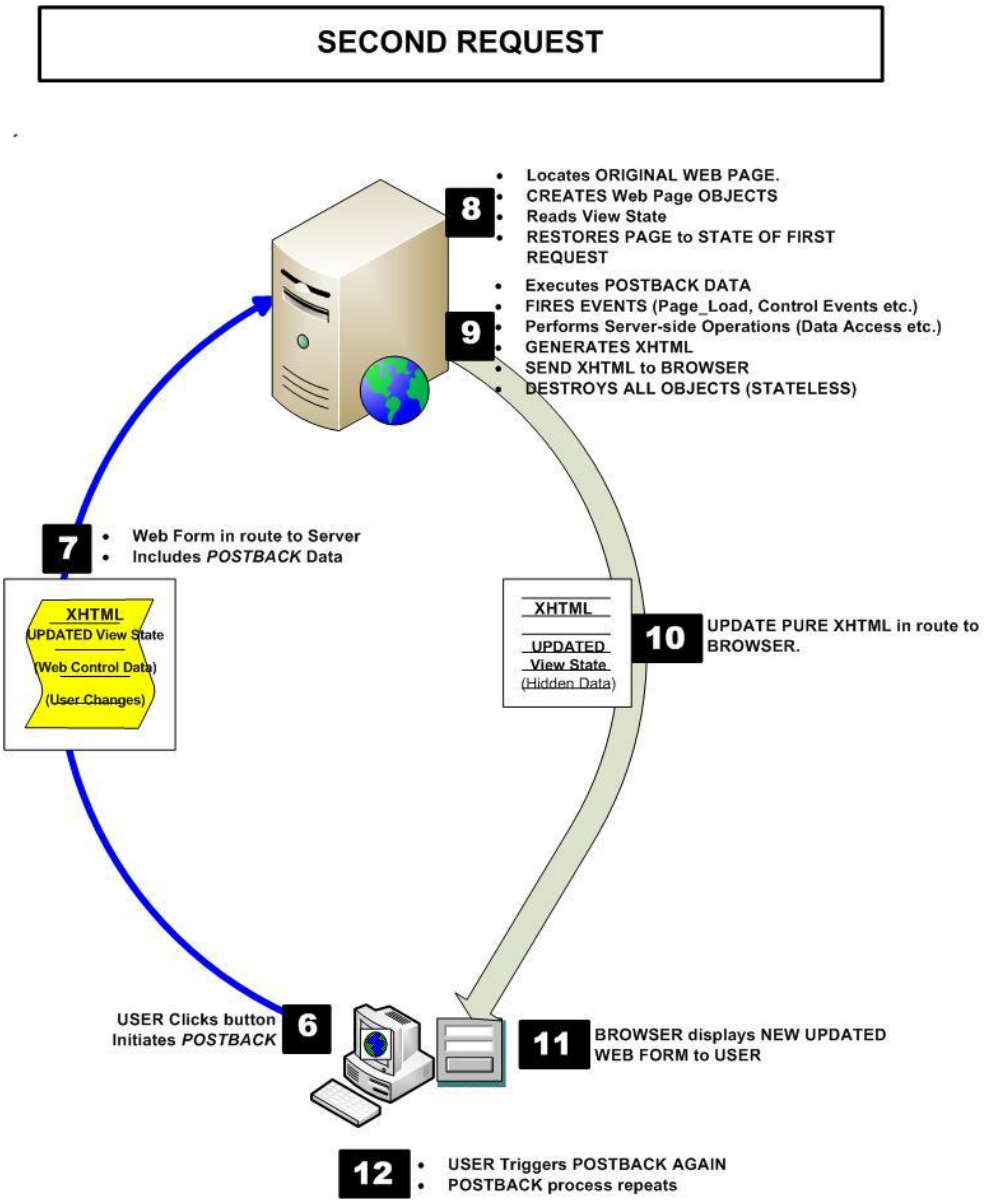

FIRST REQUEST
0
• Web Developer Creates ASP.NET Web Form (Form Controls & Code).
• Places code in Web Server
2
• Locates WEB PAGE.
• CREATES Web Page OBJECTS
3
• TRIGGERS PAGE_LOAD event (+ others).
• GENERATES XHTML
• SEND XHTML to BROWSER
• **DESTROYS ALL OBJECTS** (STATELESS)
XHTML
View State
(Hidden Data)
4
PURE XHTML in route to BROWSER.
USER/BROWSER makes request for PAGE via URL
1
5
BROWSER displays WEB FORM to USER

- Detailed explanation of steps 1 – 5:

Step	Action by Web Server or Browser	Reaction by Web Server (ASP) or Browser
0	Web Developer Creates Web Form, with Server Side Controls & Code behind-file	▪ Developer copies ASP.NET Web Form to WEB SERVER. ▪ WEB FORM is now stored in Web Server for retrieval by BROWSERS
1	Browser or Client makes request for Web Form	▪ Request is sent to server via **URL**
2	Web Server locates and initializes the Web Form	▪ PAGE RUNS FOR THE FIRST TIME **1.** Server Receives Request via **URL** **2.** Server locates the Web Form **3.** Web Server CREATES OBJECTS of Web Form (instantiates Web Form object & associated objects)
3	Web Server Process Web Form	**1.** Page Events are fired: - **Page_Load** - Other initialization events **2.** RENDERS or GENERATES an **XHTML** PAGE from WEB PAGE OBJECT **3.** SENDS pure **XHTML** to Browser **4.** DESTROYS WEB PAGE OBJECTS
4		▪ PURE HTML in route to Browser
5	BROWSER displays WEB FORM to user	▪ HTML is processed by BROWSER and displayed to User

Processing Life-Cycle for First Request for a Web Form

- Let's analyze in detail what happens when the USER triggers a **POSTBACK** an makes the second request for an UPDATED WEB FORM:


SECOND REQUEST
8
Locates ORIGINAL WEB PAGE.
CREATES Web Page OBJECTS
Reads View State
RESTORES PAGE to STATE OF FIRST REQUEST
9
Executes POSTBACK DATA
FIRES EVENTS (Page_Load, Control Events etc.)
Performs Server-side Operations (Data Access etc.)
GENERATES XHTML
SEND XHTML to BROWSER
DESTROYS ALL OBJECTS (STATELESS)
7
Web Form in route to Server
Includes POSTBACK Data
XHTML
UPDATED View State
(Web Control Data)
(User Changes)
XHTML
UPDATED
View State
(Hidden Data)
10
UPDATE PURE XHTML in route to BROWSER.
USER Clicks button
Initiates POSTBACK
6
11
BROWSER displays NEW UPDATED WEB FORM to USER
12
USER Triggers POSTBACK AGAIN
POSTBACK process repeats

❑ Detailed explanation of steps 6 – 12:

Step	Action by Web Server or Browser	Reaction by Web Server (ASP) or Browser
6	User clicks a button or does something to trigger *POSTBACK*	▪ **BROWSER** INITIATES A ***POSTBACK*** **1.** User information is packed into VIEW STATE hidden property **2.** Web Page is sent to WEB-SERVER including VIEW STATE DATA
7		▪ Web Form is in route to server ▪ Includes FORM DATA from all WEB CONTROLS
8	Web Server receives the MODIFIED WEB FORM DATA from BROWSER .	▪ PAGE RUNS FOR THE SECOND TIME ▪ PAGE INITIALIZED AND CREATED TO SAME STATE THE LAST TIME SENT TO CLIENT: **1.** Locates the ORIGINAL Web Form on the server **2.** Web Server CREATES OBJECTS of Web Form (instantiates Web Form object & associated objects) **3.** Web Form is in state when FIRST REQUESTED **4.** LOADS VIEW STATE information from memory to a COLLECTION **5.** DESERIALIZES VIEW STATE Collection data & UPDATES ALL CONTROLS with VIEW STATE data. **6.** WEB PAGE is now in same STATE as it was when SENT TO BROWSER the LAST TIME (before POSTBACK)
9	Server executes or begins ***POSTBACK*** process	▪ POSTBACK PROCESS IS EXECUTED IN SERVER: **1.** MODIFIES WEB PAGE with POSTBACK DATA from BROWSER **2.** WEB PAGE reflects the CURRENT STATE AS IT APPEARS TO USER in BROWSER **3.** EVENTS for all WEB CONTROLS are fired and processed **4.** ASP.NET TRIGGERS THE APPROPRIATE EVENT HANDLERS fired: - Page_Load Event is triggered - Other WEB CONTROL EVENTS **5.** Server Side operations can take place (updating database, reading data from file, move to new page, etc.) **6.** POPULATE WEB CONTROL OBJECTS with NEW DATA from server side operations (database, file etc.) **7.** ASP.NET examines all PROPERTIES of all WEB CONTROLS and verifies if it has changed from its INITIAL STATE. **8.** SAVES VIEW STATE information (Web Control's data) to a Key/Value COLLECTION **9.** ENCRYPTS & SERIALIZES the data from Collection to a STRING and APPENDS to VIEW STATE HIDDEN FIELD

Step	Action by Web Server or Browser	Reaction by Web Server (ASP) or Browser
		10. RENDERS or GENERATES a PURE HTML PAGE from the WEB PAGE **11.** SENDS pure **XHTML** to BROWSER **12.** DESTROYS WEB PAGE OBJECTS (includes Web Form, Controls etc.)
10		▪ NEW MODIFIED PURE HTML in route to Browser
11	BROWSER displays NEW MODIFIED WEB FORM to user	▪ NEW HTML is processed by BROWSER and displayed to User
12	User clicks a button or does something to trigger *POSTBACK*	▪ BACK TO STEP 6, ***POSTBACK*** PROCESS REPEATS

6.0 Sample Program

6.1 Sample Program #1 – Customer Management Retail Application /MS Access Database

- We now implement our Single-Tier Client/Server example of a Small Business Customer Management Application as a WEB-BASED CLIENT/SERVER APPLICATION.

Single-Tier Client/Server

- The requirements for Sample program #1. are as follows:

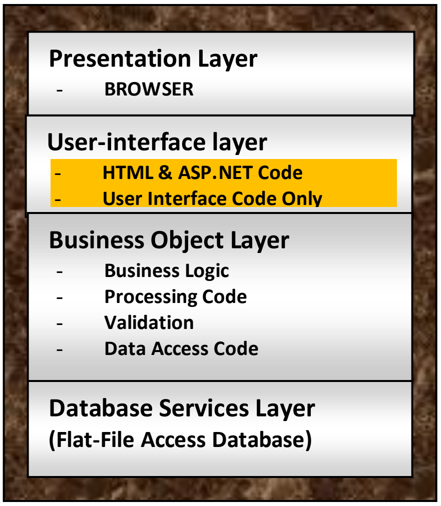
Example #1 – ASP.NET Small Business Customer Retail Management Application (MS ACCESS version)

Problem statement:

- Implement the Small Business Customer Management application as a WEB APPLICATION using ASP.NET
- The requirements are as follows:

Application Architecture – Programming Methodology

- We will NOW implement the 4-tiered layer Web-based Application Architecture:



Presentation Layer
- BROWSER

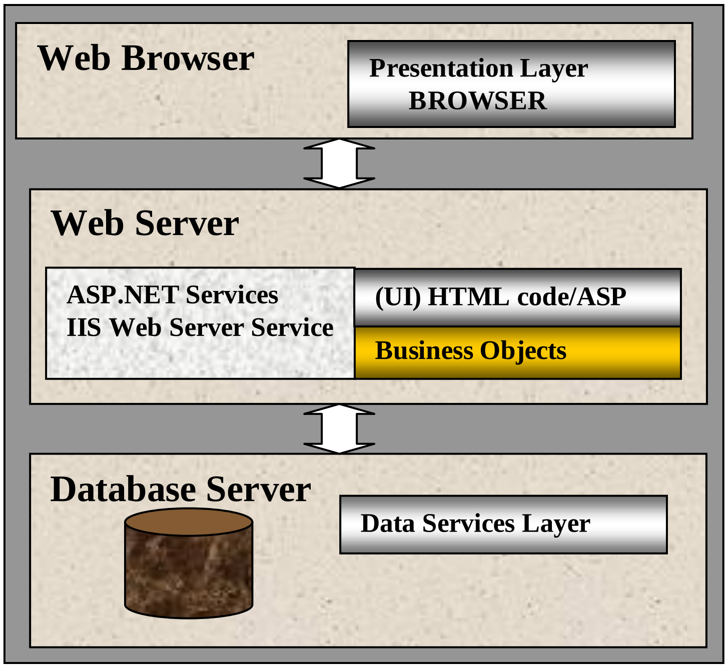
User-interface layer
- HTML & ASP.NET Code
- User Interface Code Only

Business Object Layer
- Business Logic
- Processing Code
- Validation
- Data Access Code

Database Services Layer
(Flat-File Access Database)

Client/Server Architecture – Web-based Client/Server

- This architecture looks as follows:



Presentation Layer – Client Process requirements:

- NO NEED TO CREATE A CLIENT PROGRAM, THE CLIENT IS THE BROWSER
- No coding or configuration required at this time

User-Interface Layer (THIS IS WHERE MOST OF THE PROGRAMMING WILL TAKE PLACE)

- The user-interface code is the **XHTML** & ASP.NET CODE needed to create our WEB PAGE
- We will create XHTML and ASP.NET code using VB.NET to implement our WEB FORM

New requirement for User-Interface Web Form

- In the previous Windows Application, we used a LISTBOX to display all customers
- In this WEB version, our WEB FORM will use **GRIDVIEW** control to display the customer records from DATABASE

Business Object Layer Requirements

- This code HAS ALREADY BEEN IMPLEMENTED IN THE PREVIOUS EXAMPLE
- Keep the DLL COMPONENT, Object Model, Business Rules & DATA ACCESS CODE

NEW Requirements for Business Object Layer

- We will be implementing DATABINDING in order to POPULATE THE GRIDVIEW CONTROL.
- Here is the issue:

  - Databinding is usually done by BINDING DATASETS, DATAREADERS or DATATABLES to a GRIDVIEW
  - Nevertheless, you can also BIND COLLECTIONS as well.
  - SINCE NONE OF OUR DATA ACCESS CODE RETURN DATASETS, DATAREADERS OR DATATABLES, WE WILL USE OUR COLLECTION (***clsCustomerList***) TO BIND TO THE GRIDVIEW CONTROL
  - **UNFORTUNATELY, the .NET DATABINDING MECHANISM DOES NOT SUPPORT BINDING OF DICTIONARY TYPE COLLECTIONS!**
  - **BINDING ONLY WORKS FOR ILIST TYPE COLLECTION OR ARRAYS**
  - **THEREFORE, WE CANNOT USE clsCustomerList TO DIRECTLY BIND TO THE GRIDVIEW CONTROL**

- Solution or workaround:

  - We will implement A METHOD INSIDE THE COLLECTION CLASS, this method will RETURN A CONVERTED VERSION OF OUR DICTIONARY COLLECTION AS AN ARRAY!!
  - So we will convert the collection to an ARRAY and return this ARRAY so we can BIND IT TO THE GRIDVIEW
  - The code looks as follows:

```
'********************************************************************
'Function returns array version of the Dictionary collection
'Note this function returns an ARRAY OF CUSTOMER [clsCustomer()]
Public Function ToArray() As clsCustomer()
    'create empty array of class objects. Size is obtained from collection
    Dim arrCustomerList(MyBase.Dictionary.Count - 1) As clsCustomer
    'Copy Dictionary Collection to Empty array
    MyBase.Dictionary.Values.CopyTo(arrCustomerList, 0)
    'Return populated array
    Return arrCustomerList
End Function
```

Data Service Layer – Database Requirements

- Continue to use the ***smallbusinessapp.mdb*** database

HOW IT'S DONE:

Part I – Create The Web Application:

Web User-Interface Layer

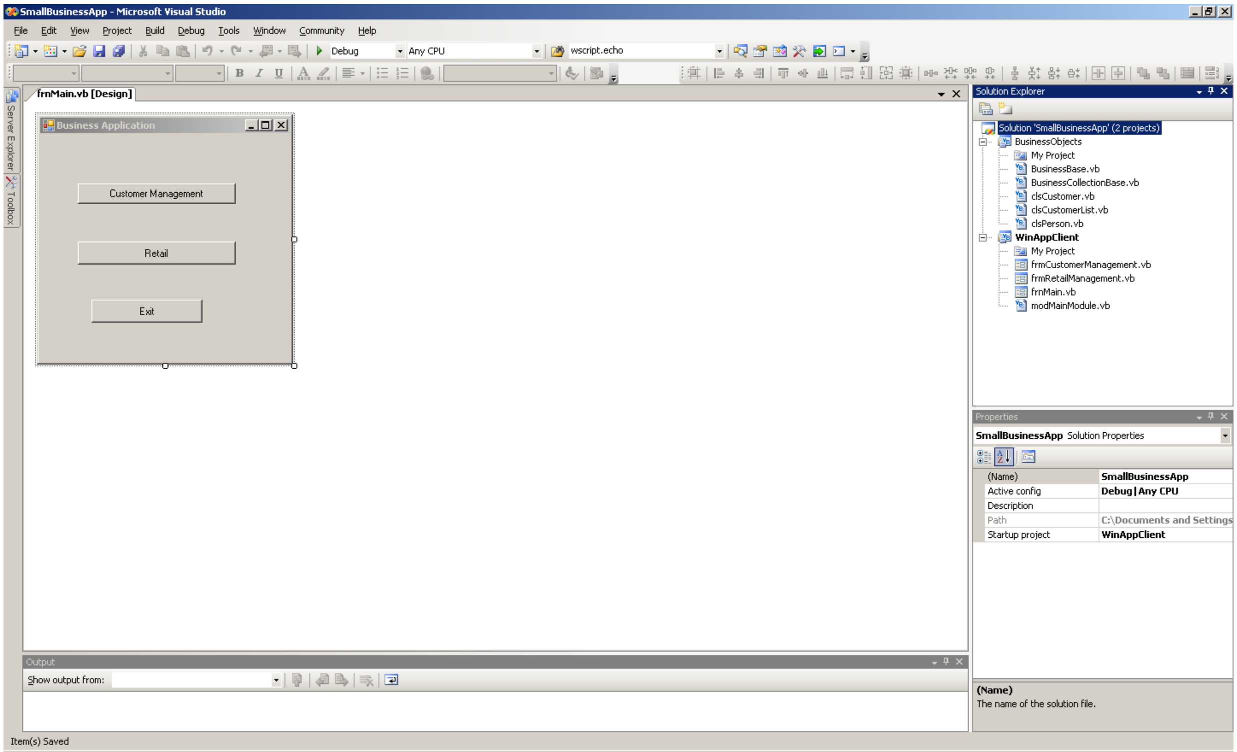
Implementing Application as a Web Application using ASP.NET & XHTML

- We are now ready to add the a WEB FORM and add our ASP.NET CODE.
- At this point, the **Customer Retail Management Application** has been upgraded to contain all the Business Logic, Validation & Data Access methods with ADO.NET code.
- Our application is a full ***Single-tier Client/Server*** application managing an MS ACCESS DATABASE.
- Using Visual Studio 2005, we will upgrade our application by doing the following:

  - Create Web Site
  - Create Web Form
  - Design and implement using ASP.NET
  - Test the application

Step 1: Open the Customer Management Application with BO & Data Access Code

- Open the last version of the Customer Management Retail Application.
- This version contains all the Business Rules, Validation, Data Access methods, and Data Access Code to continue to save and load from an Access Database.
- When you open the application, the application will contain the DLL & Windows Client Projects as follows:

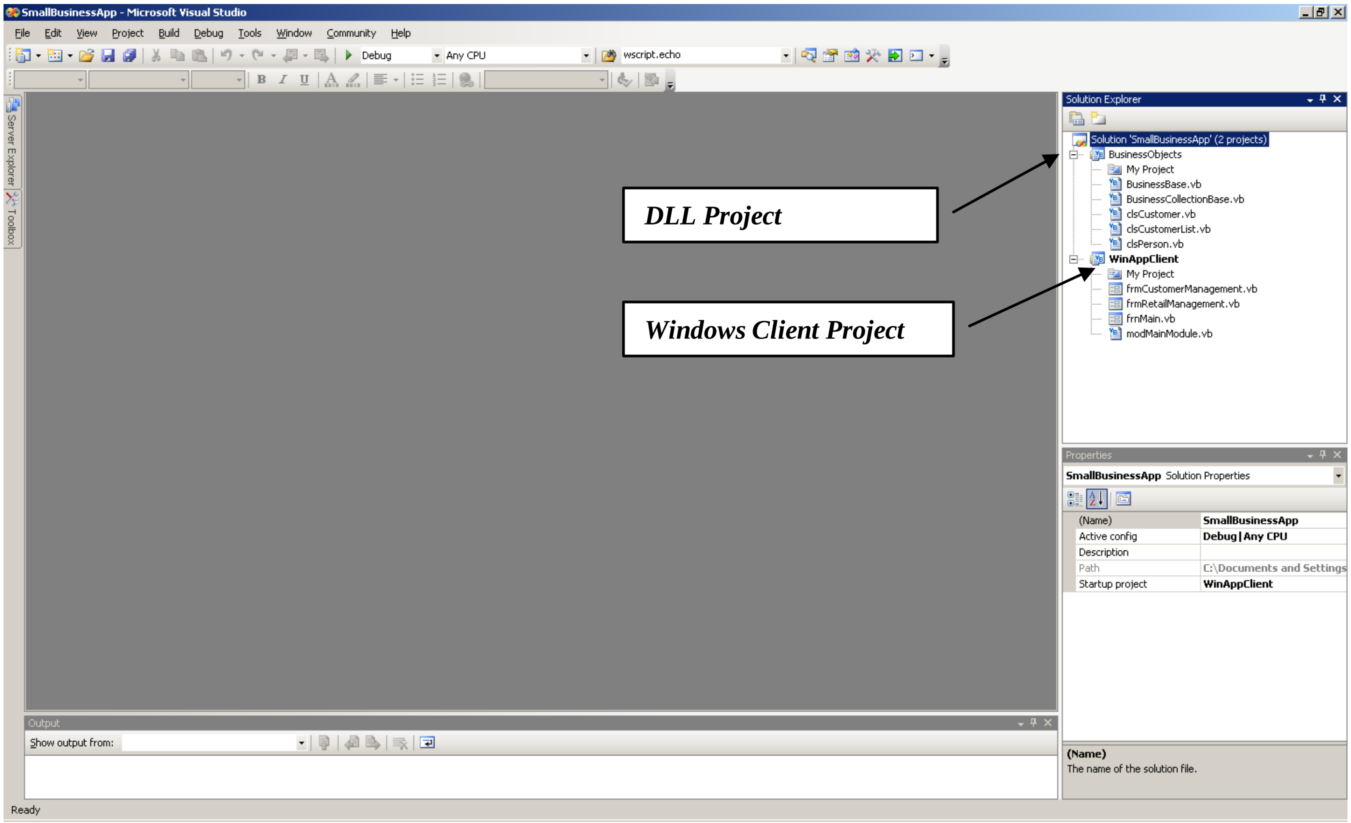


SmallBusinessApp Solution Properties

(Name)	SmallBusinessApp
Active config	Debug|Any CPU
Description	
Path	C:\Documents and Settings
Startup project	WinAppClient

(Name)
The name of the solution file.

- Another view with all Forms closed:



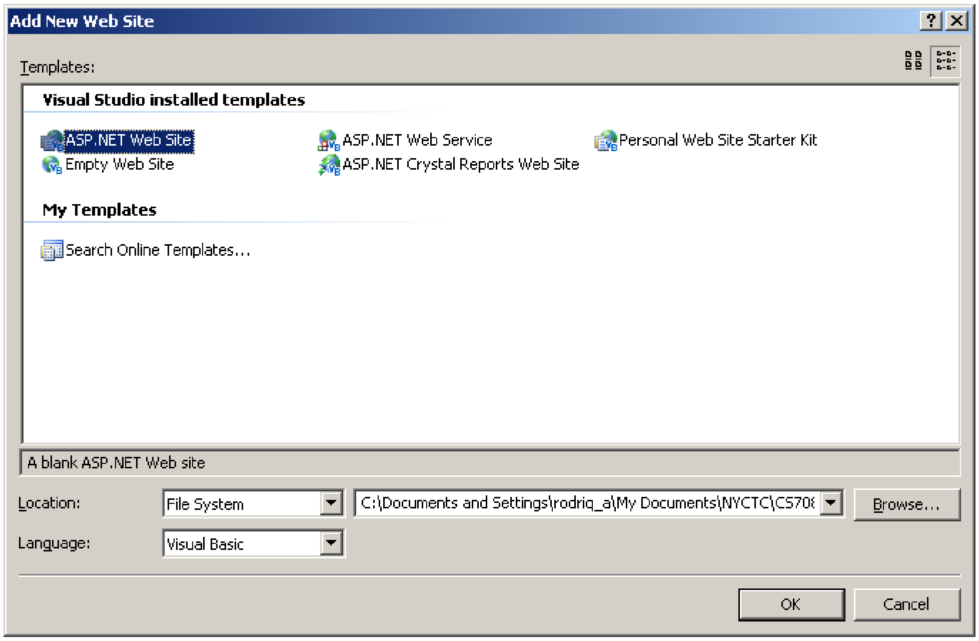
(Name)	SmallBusinessApp
Active config	Debug|Any CPU
Description	
Path	C:\Documents and Settings
Startup project	WinAppClient

(Name)
The name of the solution file.

Step 2: Add a FILE Web Site to Existing Project

- We add a web project to our existing application. No need to start from scratch , simply add the Web Project and our solution contains both type of clients.

a) In the main menu, select **File|Add|New Web Site….** To invoke the *Add New Web Site* screen
b) In the *Templates* box select: ***"ASP.NET Web Site"*, i**n the *Template* box select: ***"File System***"
c) Select Visual Basic as the DEFAULT Language


Add New Web Site
Templates:
Visual Studio installed templates
ASP.NET Web Site
ASP.NET Web Service
Personal Web Site Starter Kit
Empty Web Site
ASP.NET Crystal Reports Web Site
My Templates
Search Online Templates...
Browse...
OK
Cancel

A blank ASP.NET Web site

Location: File System   C:\Documents and Settings\rodriq_a\My Documents\NYCTC\CS70

Language: Visual Basic

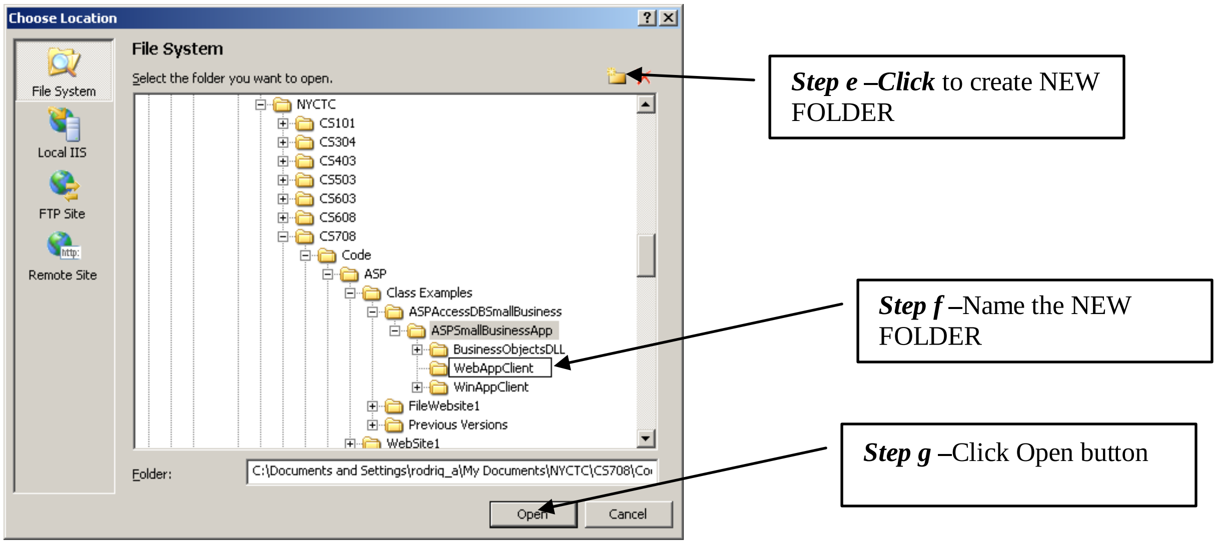
d) Click ***"Browse"*** and browse to the desired location

e) Click the ***"Create New Folder ICON"*** on the top right-hand side of the '*Choose Location*" screen

f) Name the folder "***WebAppClient***"

g) Click Open button to return to the "***New Web Site***" screen

h) Click ***OK*** in *Add New Web Site* screen to create the web site



Folder: C:\Documents and Settings\rodriq_a\My Documents\NYCTC\CS708\Co

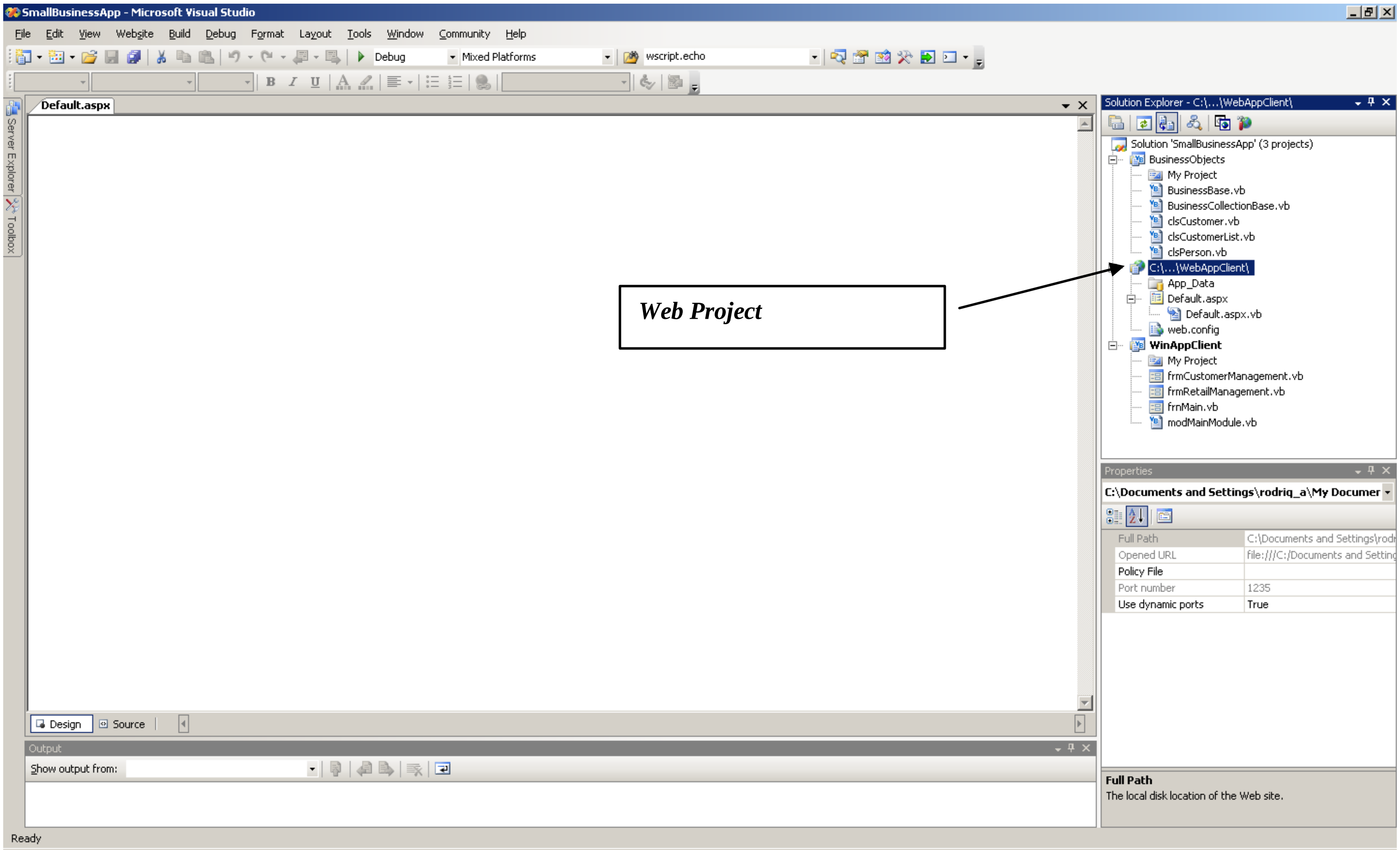
Step e –Click to create NEW FOLDER

Step f –Name the NEW FOLDER

Step g –Click Open button

a) Click ***OK*** in *Add New Web Site* screen to create the web site & the IDE will display as follows:

- Note we now have three project: DLL, Windows Client & new WEB CLIENT
- Project will automatically include the ***DEFAULT.ASPX*** WEB FORM, Code-behind file ***default.aspx.vb*** & ***Web.config***



C:\Documents and Settings\rodriq_a\My Documen

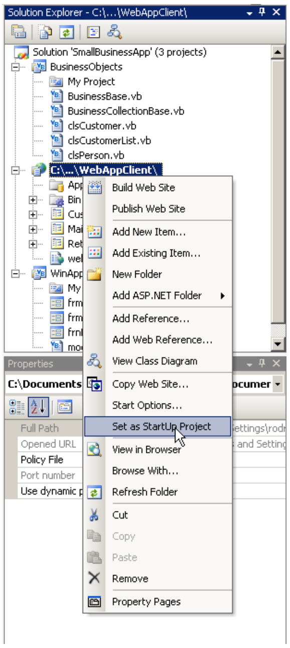
Full Path	C:\Documents and Settings\rodr
Opened URL	file:///C:/Documents and Setting
Policy File	
Port number	1235
Use dynamic ports	True

Full Path
The local disk location of the Web site.

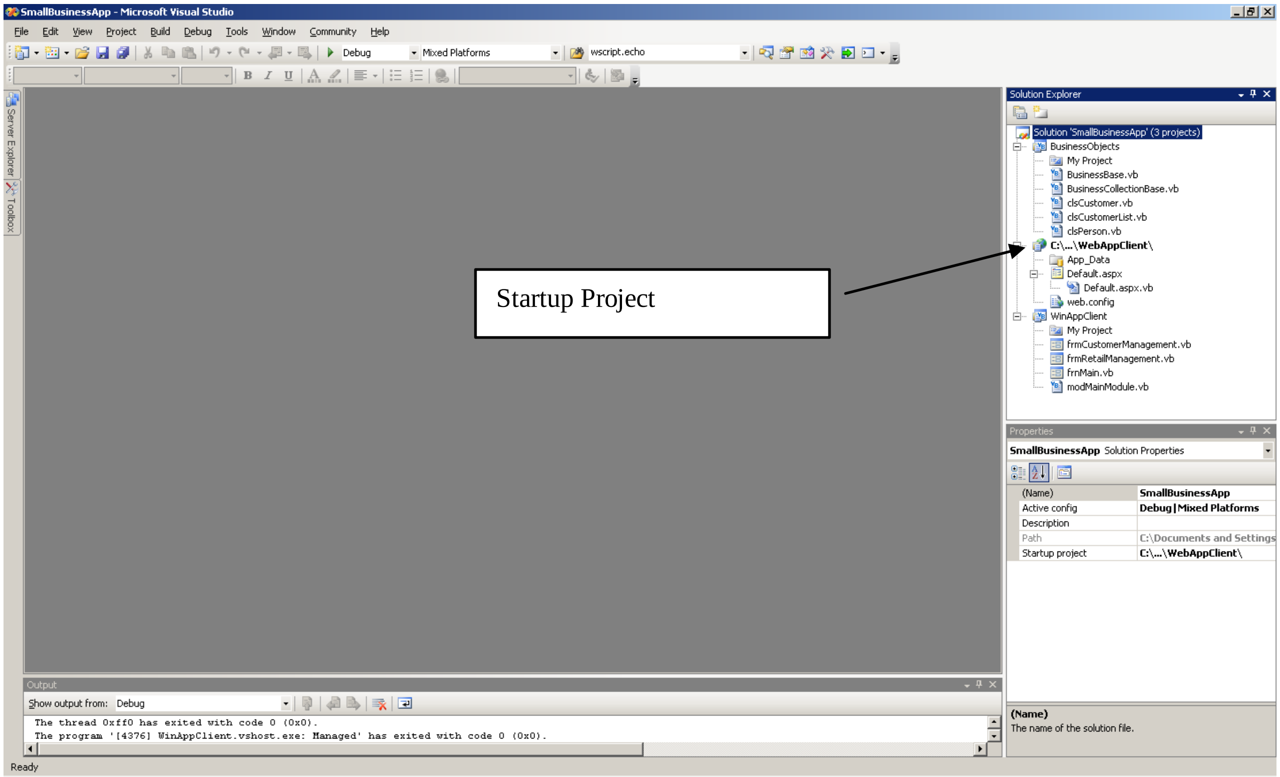
Web Project

Step 3: Set Web Project as Startup Project

- ❑ Currently it is the Windows Client or WinAppClient that is the STARTUP PROJECT, thus running or controlling this application.
- ❑ Since we are now managing this application as a web application we need to set it as the STARTUP PROJECT as follows:



a) In the Solution Explore, right-click on the Web Project **WebAppClient**
b) In the menu select "
c) The Web Project will now be IN BOLD



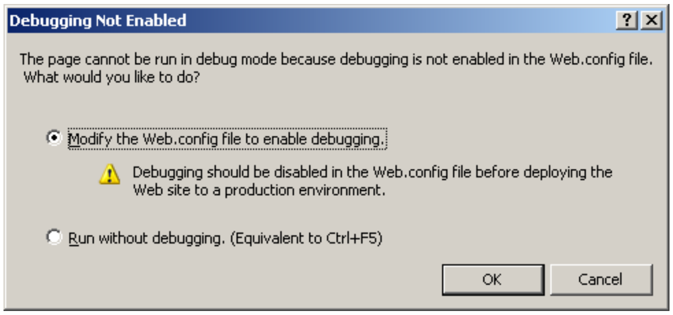
(Name)	SmallBusinessApp
Active config	Debug|Mixed Platforms
Description	
Path	C:\Documents and Settings
Startup project	C:\...\WebAppClient\

(Name)
The name of the solution file.

```
The thread 0xff0 has exited with code 0 (0x0).
The program '[4376] WinAppClient.vshost.exe: Managed' has exited with code 0 (0x0).
```

Step 4: [Optional] Set Web Project for Debugging

❑ When you try to compile, you may get a dialog box when executing the project that will ask you if you want to turn on debugging:



Debugging Not Enabled

The page cannot be run in debug mode because debugging is not enabled in the Web.config file. What would you like to do?

● Modify the Web.config file to enable debugging.

Debugging should be disabled in the Web.config file before deploying the Web site to a production environment.

○ Run without debugging. (Equivalent to Ctrl+F5)

❑ This will enable the debugging mechanism in Visual Studio for this project, that is you can SET BREAKPOINTS, etc..

❑ It is in the WEB.CONFIG file in a TAG name <compilation debug> where debugging is turned ON/OFF. This dialog will actually modify the Web.config file.

❑ Initially, the TAG looks as follows in the Web.config file:

```
<compilation debug="false" strict="false" explicit="true" />
```

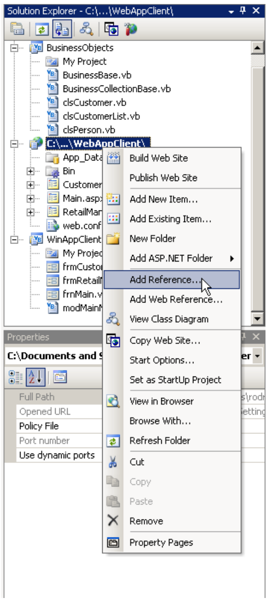
❑ After this dialog executes it looks as follows:

```
<compilation debug="true" strict="false" explicit="true"/>
```

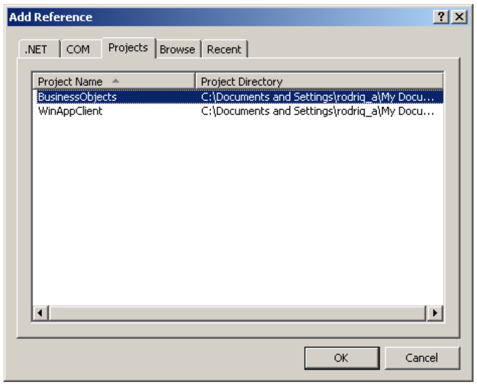
❑ NOTE THAT YOU CAN MAKE THIS MODIFICATION YOURSELF AND YOU WON'T GET THE DIALOG BOX.!

Step 5: IMPORTANT! Add Reference to DLL Component

- ❑ Now we need to tell the Web Project where the classes it will need are located.
- ❑ We need to REFERENCE OR POINT to the DLL PROJECT:



a) In the Solution Explore, right-click on the Web Project **WebAppClient**
b) In the menu select "***Add Reference….***"
c) This will invoke the "Add Reference" screen
d) Click the "Project" tab and select the **BusinessObjects** DLL
e) Click OK

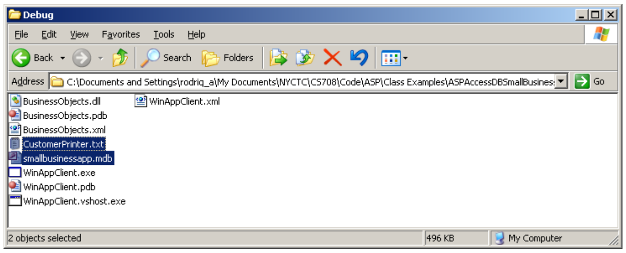


Project Name	Project Directory
BusinessObjects	C:\Documents and Settings\rodriq_a\My Docu…
WinAppClient	C:\Documents and Settings\rodriq_a\My Docu…

Step 6: IMPORTANT! Copy Database File & Printer File to the Root of the Solution

- ❑ The Web Client will automatically search for any Database File or text file in the ROOT DIRECTORY IT WAS CREATED IN.
- ❑ Since our program requires the ACCESS MDB file & the PRINTER file for Customer Printing, we need to COPY/PASTE these files to the root location
- ❑ Currently these files exist in the WINDOWS CLIENT PROJECT WinAppClient\BIN\DEBUG\ folder
- ❑ Do the following:

a) Navigate to the folder **WinAppClient\Bin\Debug\**
b) And SELECT COPY the database file ***smallbusinessapp.mdb*** & ***CustomerPrinter.txt*** files:

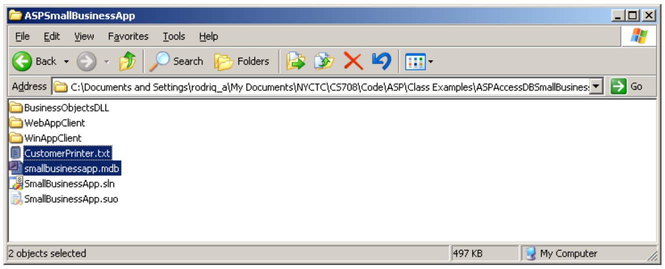


Address: C:\Documents and Settings\rodriq_a\My Documents\NYCTC\CS708\Code\ASP\Class Examples\ASPAccessDBSmallBusines

```
BusinessObjects.dll        WinAppClient.xml
BusinessObjects.pdb
BusinessObjects.xml
CustomerPrinter.txt
smallbusinessapp.mdb
WinAppClient.exe
WinAppClient.pdb
WinAppClient.vshost.exe
```

2 objects selected | 496 KB

c) PASTE these file in the Root directory of the SOLUTION \ASPAccessDBSmallBusinessApp\:



Address: C:\Documents and Settings\rodriq_a\My Documents\NYCTC\CS708\Code\ASP\Class Examples\ASPAccessDBSmallBusines

```
BusinessObjectsDLL
WebAppClient
WinAppClient
CustomerPrinter.txt
smallbusinessapp.mdb
SmallBusinessApp.sln
SmallBusinessApp.suo
```

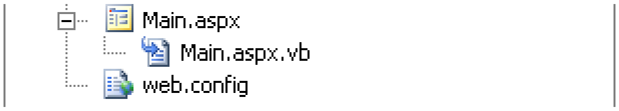
2 objects selected | 497 KB

❖ **SOLUTION ENVIRONMENT IS CONFIGURED, WE ARE NOW READY TO BEGIN TO DESIGN OUR WEB FORMS!**

Step 7: RENAME Defalut.aspx, to create the Main Web Form

- Instead of adding a new form and deleting the Default.aspx, we will rename this form, to see how that process works.
- We need to rename three items: 1) The Web Form file name (Solution Explorer), 2) HTML Source Code & 3) the Class Name in the aspx.vb Code-behind file:

a) In the Solution Explore, right-click on the Web Project **WebAppClient**
b) In the menu select "***Rename"*** and rename your Default.aspx to **Main.aspx**



c) Double-click the file, to open the newly renamed Web Form
d) Click the Source button to switch to SOURCE VIEW, in the XHTML code the first line looks as follows:

```
 <%@ Page Language="VB" AutoEventWireup="false" CodeFile="Default.aspx.vb"
Inherits="_Default" %>
```

e) MODIFY by renaming the value that indicates the name of the code-behind file: **CodeFile = Default.aspx.vb** & the value that indicates the class name **Inherits="_Defaut"** as follows:

```
<%@ Page Language="VB" AutoEventWireup="false" CodeFile="Main.aspx.vb"
Inherits="Main" %>
```

f) Finally we need to open the code-behind file (double-click on Main.aspx.vb in Solution Explorer):

```
Partial Class _Default
    Inherits System.Web.UI.Page

End Class
```

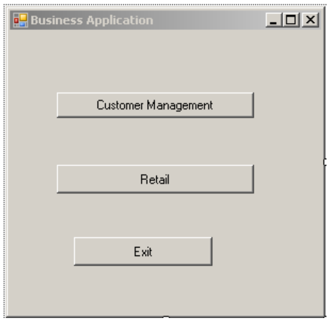
g) Rename from _Default to Main:

```
Partial Class Main
    Inherits System.Web.UI.Page

End Class
```

Step 8: Design the Main Page

- ❑ Begin to design what you main page will look like.
- ❑ Since we are trying to recreate the WINDOWS APPLICATION AS A WEB APPLICATION AS CLOSE AS POSSIBLE, currently the Windows version looks as follows:

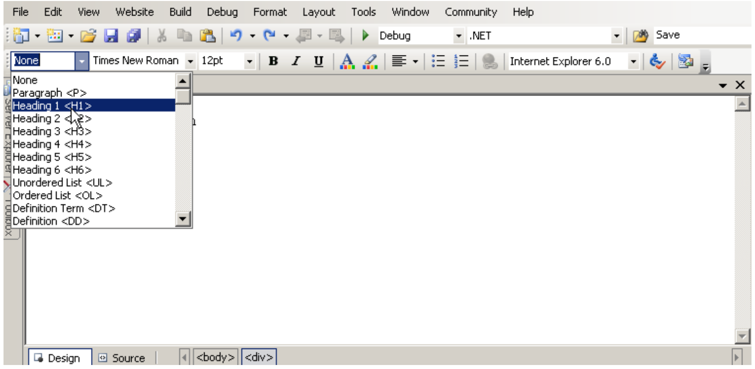


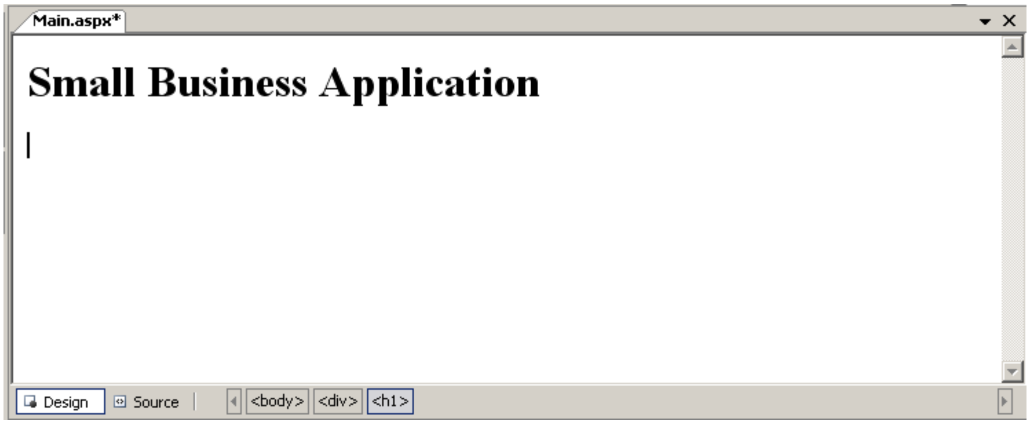
- ❑ I will use the following WEB controls:

  - ▪ HTML table – To align controls
  - ▪ One ASP control BUTTONS – for Customer Management
  - ▪ One ASP control BUTTONS – for Retail Management

- ❑ Note that we don't need an EXIT button because this is a web form in a BROWSER, we just close the browser.
- ❑ Design your WEB FORM as follows:

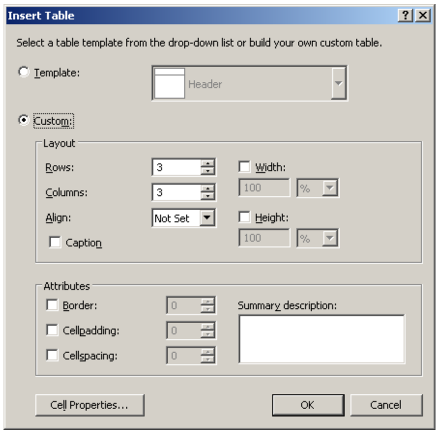
a) Open the Main.aspx Form and at the top type "**Small Business Application**", high-light the text and select Heading 1 in the style drop-down listbox in the toolbar:





Small Business Application

b) ADD A TABLE; From the Toolbox, drag an HTML table or in the Menu Bar select Layout|Insert table… to invoke the following dialog, SELECT a 3X3 table:



Select a table template from the drop-down list or build your own custom table.

○ Template: Header

● Custom:

Layout

Rows: 3

Columns: 3

Align: Not Set

□ Caption

□ Width: 100 %

□ Height: 100 %

Attributes

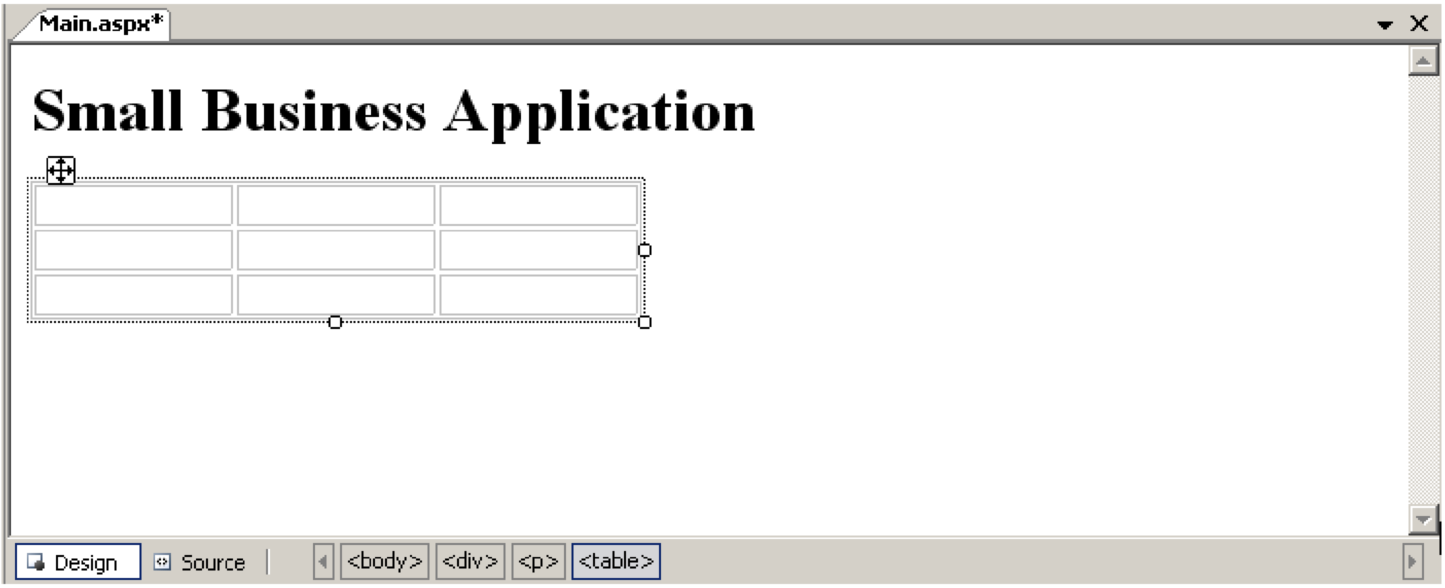
□ Border: 0

□ Cellpadding: 0

□ Cellspacing: 0

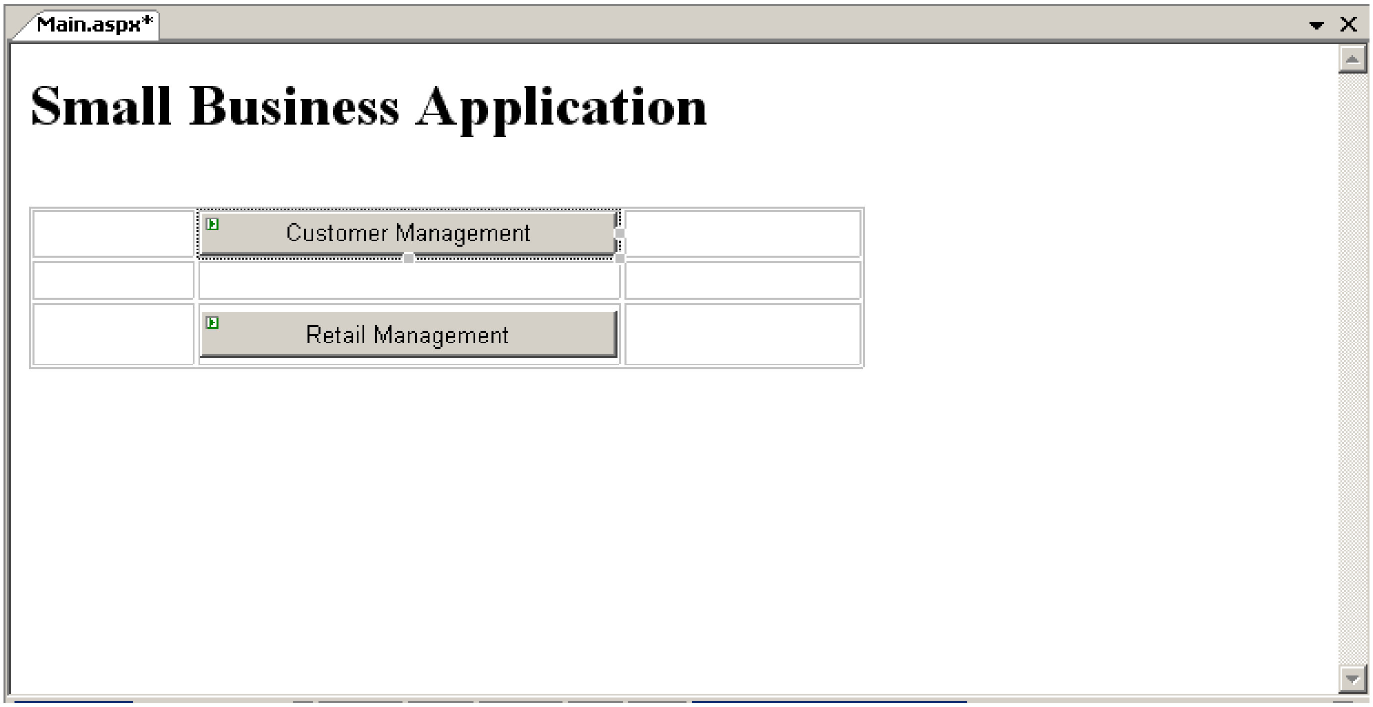
Summary description:

c) The form now looks as follows:



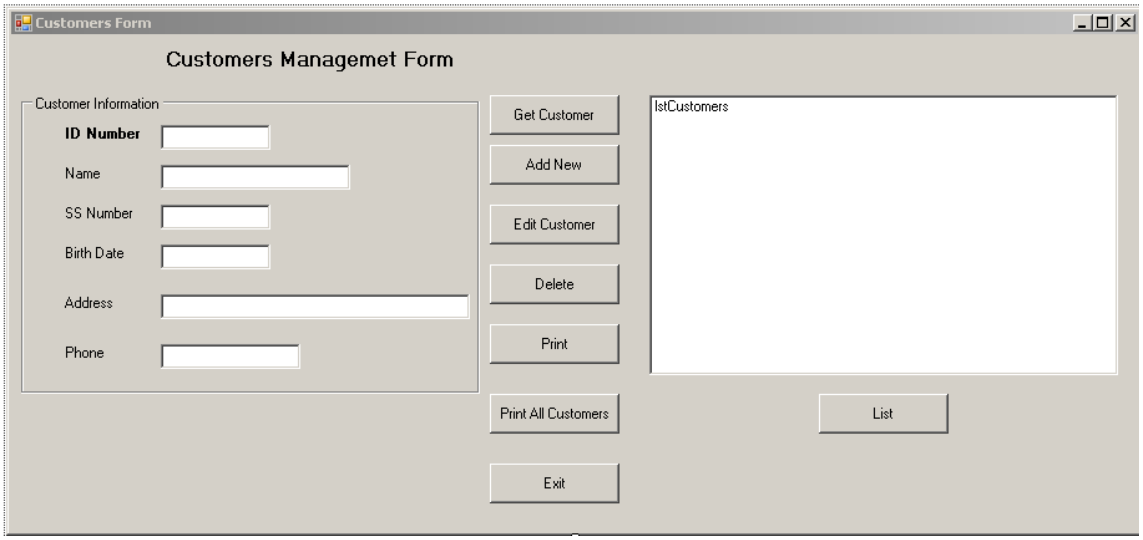
Small Business Application

d) ADD TWO BUTTONS from the toolbox and SET the PROPERTIES for ID & TEXT:



Step 9: Design the Customer Management Form

- Begin to design what you main page will look like.
- Since we are trying to recreate the WINDOWS APPLICATION AS A WEB APPLICATION AS CLOSE AS POSSIBLE, currently the Windows version looks as follows:

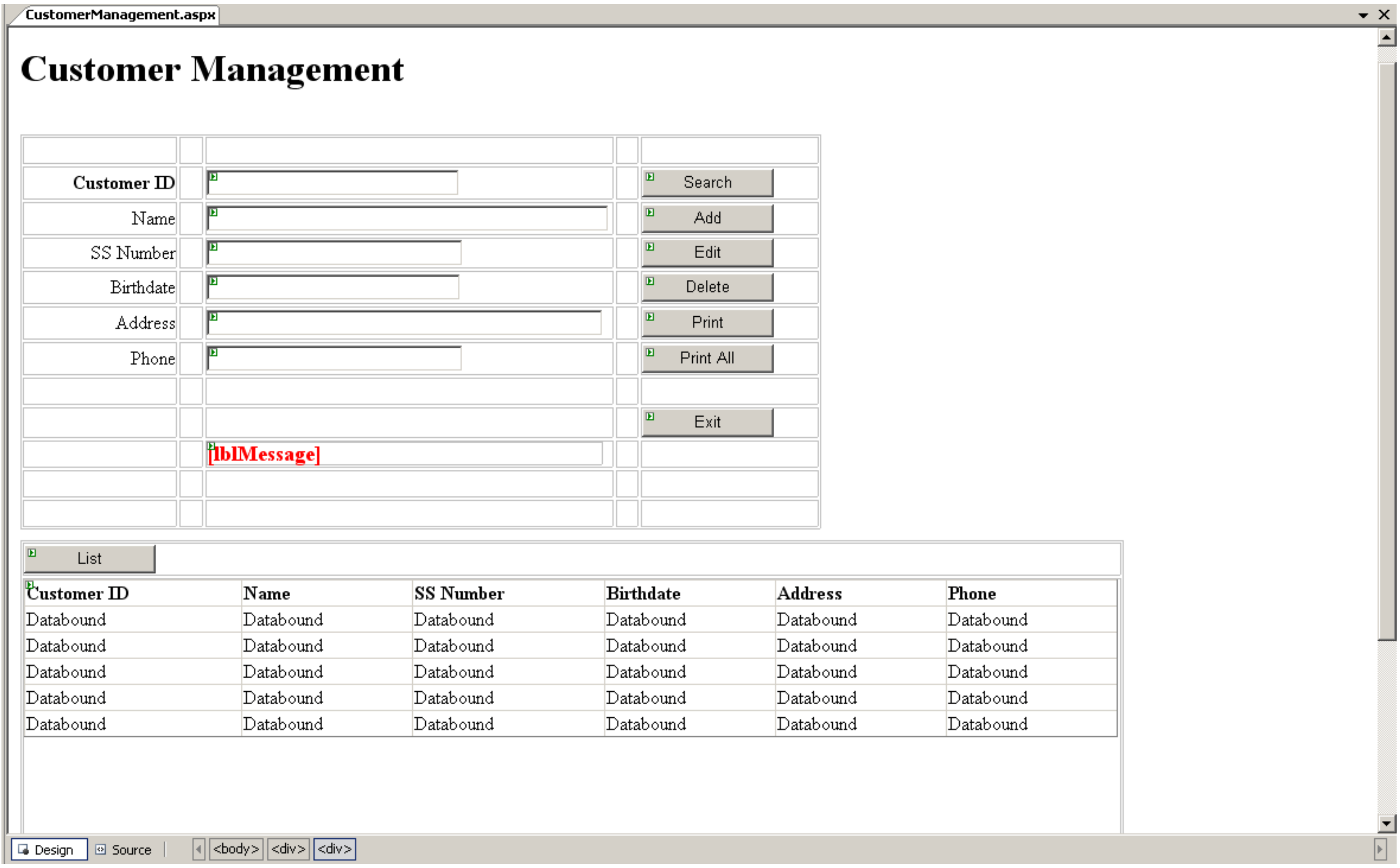


- I will use the following WEB controls:
  - 1 HTML TABLE – To align controls for Customer Information
  - 1 HTML TABLE – To align controls for GRIDVIEW CONTROL
  - 6 ASP TEXTBOX CONTROLS – for imputing Customer ID, Name, Social Security, Birth date, Address, & Phone
  - 7 ASP control BUTTONS – for various operations, Search, Add, Edit, Remove, Print, PrintAll & Exit
  - 1 LABEL – For displaying error messages or other messages (more on this later)
  - 1 GRIDVIEW CONTROL – For displaying records of all customers from the database

- Note that we don't need LABELS FOR IDENTIFYING the TEXTBOXES as we did in the Windows Application, we can type these directly in to the TABLE
- Design your WEB FORM as follows:

a) Add a Web Form and name it Customer Management as follows:

b) Design the Form using the controls indicated as follows:



Customer Management

Customer ID		Search
Name		Add
SS Number		Edit
Birthdate		Delete
Address		Print
Phone		Print All
		Exit
	[lblMessage]	

List

Customer ID	Name	SS Number	Birthdate	Address	Phone
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound
Databound	Databound	Databound	Databound	Databound	Databound