Reputation
CRM
Quality
Goals
ASP.NET
Innovation
Strategy
Solution
Analysis
Marketing
Management
Service
Team
Finance
Customers
Costs
Income
Plan

Jump2Learn
The Online Learning Place

ASP.NET

CHAPTER 1 : INTRODUCTION TO ASP.NET

AUTHORS

Dr. Ashoksinh V. Solanki
M.C.A., Ph.D.
COLLEGE OF APPLIED SCIENCE
AND PROFESSIONAL STUDIES, CHIKHLI

Mr. Yatin Solanki
M.C.A., Ph.D.(Pursuing)
DOLAT USHA INSTITUTE OF
APPLIED SCIENCES, VALSAD

Mr. Chetan Parmar
M.C.A., NET, SET
NARMADA COLLEGE OF SCIENCE
& COMMERCE, ZADESHWAR

Website : www.jump2learn.com| **Email :** info@jump2learn.com|**Instagram :** www.instagram.com/jump2learn
Facebook : www.facebook.com/Jump2Learn | **Whatsapp :** +91-909-999-0960 | **YouTube :** Jump2Learn

WHAT IS ASP.NET?

ASP.NET is an entirely new technology for server-side scripting. It was written from the ground up and is not backward compatible with classic ASP.

ASP.NET is the major part of the Microsoft's .NET Framework.

"ASP.NET is a server side scripting technology that enables scripts (embedded in web pages) to be executed by an Internet server."

- ASP.NET is a Microsoft Technology
- ASP stands for Active Server Pages
- ASP.NET is a program that runs inside IIS
- IIS (Internet Information Services) is Microsoft's Internet server
- IIS comes as a free component with Windows servers
- IIS is also a part of Windows 2000 and XP Professional
- Files containing HTML and scripting code
- Access via HTTP requests
- Scripting code is interpreted on server side

WHAT CAN I DO WITH ASP.NET?

- Easily and quickly create simple Web applications
- Generate dynamic Web content
- Client-side scripting for validation
- Access COM components to extend functionality

WHAT IS AN ASP.NET FILE?

- An ASP.NET file is just the same as an HTML file
- An ASP.NET file can contain HTML, XML, and scripts
- Scripts in an ASP.NET file are executed on the server
- An ASP.NET file has the file extension ".aspx"

HOW DOES ASP.NET WORK?

- When a browser requests an HTML file, the server returns the file
- When a browser requests an ASP.NET file, IIS passes the request to the ASP.NET engine on the server
- The ASP.NET engine reads the file, line by line, and executes the scripts in the file
- Finally, the ASP.NET file is returned to the browser as plain HTML

SEVEN TOUCHSTONES(ADVANTAGE) OF ASP.NET DEVELOPMENT.

- ASP.NET Is Integrated with the .NET Framework

- ASP.NET Is Compiled,Not Interpreted
    - One of the major reasons for performance degradation in ASP scripts is that all ASP web-page code uses interpreted scripting languages.
    - ASP.NET applications actually go through two stages of compilation.
        - *In the first stage, the C# code you write is compiled into an intermediate language called Microsoft Intermediate Language (MSIL) code, or just IL.*
        - *This first compilation step may happen automatically when the page is first requested, or you can perform it in advance (a process known as* precompiling*)*
    - The compiled file with IL code is an *assembly*.
    - The second level of compilation happens just before the page is actually executed.
    - At this point, the IL code is compiled into low-level native machine code. This stage is known as *just-in time* (JIT) compilation

- ASP.NET Is Multilanguage
    - No matter what language you use, the code is compiled into IL.
    - IL is a stepping-stone for every managed application. (A *managed application* is any application that's written for .NET and executes inside the managed environment of the CLR.)
    - It's easy enough to look at the IL for any compiled .NET application. You simply need to run the IL Disassembler, which is installed with Visual Studio and the .NET SDK (software development kit).
    - Look for the file ildasm.exe in a directory like c:\Program Files\Visual Studio 2005\SDK\v2.0\Bin.
    - Once you've loaded the program, use the File -> Open command, and select any DLL or EXE that was created with .NET.

- ASP.NET Runs Inside the Common Language Runtime
    - *Automatic memory management and garbage collection:*
    - *Type safety:*
    - *Extensible metadata:*
    - *Structured error handling:*
    - *Multithreading*

- ASP.NET Is Object-Oriented

- ASP.NET Is Multidevice and Multibrowser

- ASP.NET Is Easy to Deploy and Configure

PAGE SYNTAX

- Directives
  - <%@ Page language=“C#“ […] %>

- Code Declaration Blocks
  - <script runat=“server“ […]>
  -
    [lines of code]
  -
    </script>

- Code Render Blocks
  - <%
    [inline code or expression]
  -
    %>

- HTML Control Syntax
  - <HTMLelement runat=“server“ [attribute(s)]>
  -
    </HTMLelement>

CUSTOM CONTROL SYNTAX

- Custom server controls
  - <ASP:TextBox id=“MyTb1“ runat=“server“>

- Server control property
  - <ASP:TextBox maxlength=“80“ runat=“server“>

- Subproperty
  - <ASP:Label font-size=“14“ runat=“server“>

- Server control event binding
  - <ASP:Button OnClick=“MyClick“ runat=“server“>

- Data Binding Expression
  - <asp:label text=‘<%# *databinding expression %>’*
  -
    runat=“server” />

- Server-side Object Tags
  - <object id="*id*" runat="server" *identifier*="*idName*" />

- Server-side Include Directives
  - <!-- #include *pathtype* = *filename* -->

- Server-side Comments
  - <%-- *comment block* --%>

PAGE DIRECTIVES

- Defines page-specific (.aspx file) attributes used by the ASP.NET page parser and compiler. <%@ Page *attribute*="*value*" [*attribute*="*value*"...] %> Attributes

- E.g <%@ Page language="C#" [...] %>

DIFFERENT ATTRIBUTES :

- **AutoEventWireup**
  - Indicates whether the page's events are autowired. true if event autowiring is enabled; otherwise, false. The default is true.
- **Buffer**
  - Determines whether HTTP response buffering is enabled. true if page buffering is enabled; otherwise, false. The default is true.
- **ClassName**
  - Specifies the class name for the page that will be dynamically compiled automatically when the page is requested. This value can be any valid class name but should not include a namespace.
- **ClientTarget**
  - Indicates the target user agent for which ASP.NET server controls should render content. This value can be any valid user agent or alias.
- **CodeBehind**
  - Specifies the name of the compiled file that contains the class associated with th page.
  - value of WebForm1.aspx.vb, for Visual Basic, or WebForm1.aspx.cs, for C#.

- **Strict**
  - Indicates that the page should be compiled using the Visual Basic Option Strict mode. true if Option Strict is enabled; otherwise, false. The default is false.
- **Trace**
  - Indicates whether tracing is enabled. true if tracing is enabled; otherwise, false. The default is false.

- **Transaction**
  - Indicates whether transactions are supported on the page. Possible values are Disabled, NotSupported, Supported, Required, and RequiresNew.
  - The default is Disabled.

- **ValidateRequest**
  - Indicates whether request validation should occur. If true, request validation checks all input data against a hard-coded list of potentially dangerous values. If a match occurs, an HttpRequestValidationException Class is thrown. The default is true.
  - This feature is enabled in the machine configuration file (Machine.config). You can disable it in your application configuration file (Web.config) or on the page by setting this attribute to false.

❖ THE .NET FRAMEWORK:

Microsoft .NET is a software component that runs on the Windows operating system.

The .NET Framework is designed to fulfill the following objectives:

- To provide a consistent object-oriented programming environment whether object code is stored and executed locally, executed locally but Internet-distributed, or executed remotely.
- To provide a code-execution environment that minimizes software deployment and versioning conflicts.
- To provide a code-execution environment that promotes safe execution of code, including code created by an unknown or semi-trusted third party.
- To provide a code-execution environment that eliminates the performance problems of scripted or interpreted environments.
- To make the developer experience consistent across widely varying types of applications, such as Windows-based applications and Web-based applications.
- To build all communication on industry standards to ensure that code based on the .NET Framework can integrate with any other code.

The .NET Framework has two main components: the common language runtime (CLR) and the .NET Framework Base class library (BCL).

The Microsoft .NET Framework was developed to solve this problem.

- Easier and quicker programming
- Reduced amount of code
- Declarative programming model
- Richer server control hierarchy with events
- Larger class library
- Better support for development tools

THE .NET FRAMEWORK CONSISTS OF 3 MAIN PARTS:

- Programming languages:
  - C# (Pronounced C sharp)
  - Visual Basic (VB .NET)
  - J# (Pronounced J sharp)
- Server technologies and client technologies:
  - ASP .NET (Active Server Pages)
  - Windows Forms (Windows desktop solutions)
  - Compact Framework (PDA / Mobile solutions)
- Development environments:
  - Visual Studio .NET (VS .NET)
  - Visual Web Developer

❖ COMPILE CODE

- Two models for coding web pages and web services:
  - Inline code:
    - *This model is the closest to traditional ASP.*
    - *All the code and HTML is stored in a single .aspx file.*
  - Code-behind:
    - *This model separates each ASP.NET web page into two files: an .aspx markup file with the HTML and control tags, and a .cs or .vb code file with the source code for the page.*

❖ COMMON LANGUAGE RUNTIME

The common language runtime is the foundation of the .NET Framework. You can think of the runtime as an agent that manages code at execution time, providing core services such as memory management, thread management, and remoting, while also enforcing strict type safety and other forms of code accuracy that promote security and robustness. It is considered as the heart of the .NET framework.

The goals of the CLR are as follows:

- Secure and robust execution environment
- Simplified development process
- Multilanguage support
- Simplified management and simplified deployment

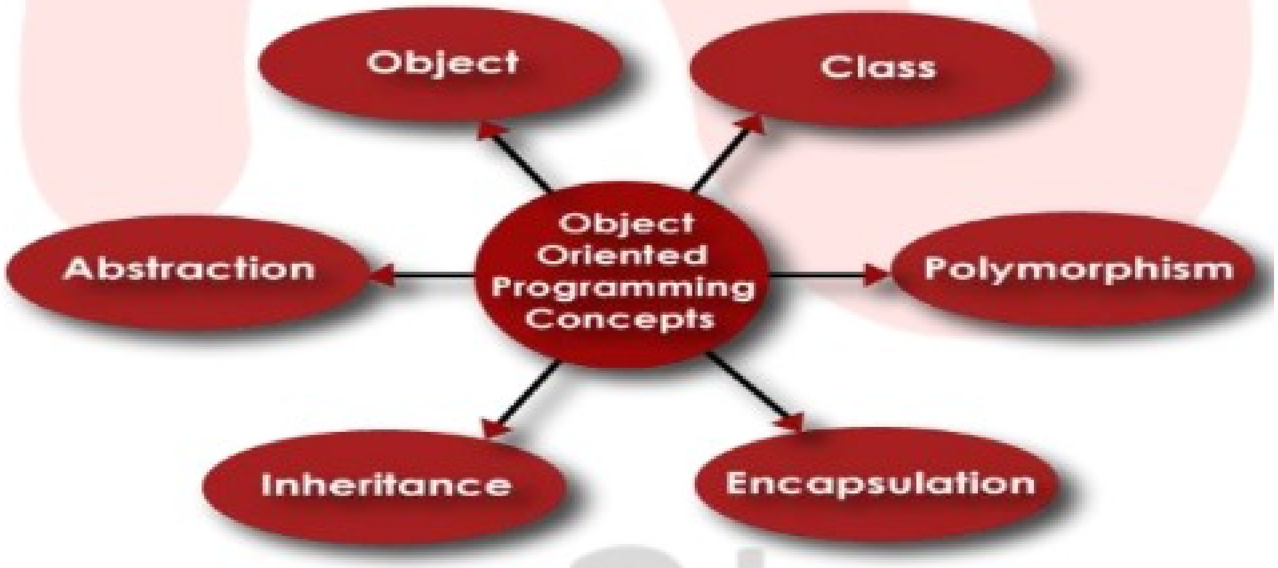
When you create any VB.NET application and if you compiled you code then it generate the CLR's intermediate code named as MSIL (Microsoft Intermediate Language) which is not directly executed by CPU so when we execute the code the MSIL code will be converted in to Native code which are executed by the CPU which is done by JIT compiler.

❖ OBJECT ORIENTED CONCEPTS

Object Oriented Programming (OOP) is one of the most popular programming languages.

In the class-based **object-oriented** programming paradigm, "**object**" refers to a particular instance of a class where the **object** can be a combination of variables, functions, and data structures. A good understanding of OOPs concepts can help in decision making when designing an application. How you should design an application and what language should be used.

Object Oriented programming is a programming style which is associated with the concepts like **class**, **object**, **Inheritance**, **Encapsulation**, **Abstraction**, **Polymorphism.**



OBJECT

Object is representative of the class and is responsible for memory allocation of its data members and member functions. An object is a real world entity having attributes (data type) and behaviors (functions).

It is a real time entity.

CLASS

Class is a data structure that contains data members (constants files, events), member function methods, properties, constructor, destructor, indexers and nested type. It is a collection of objects.

Basically

1) It is a user defined data type.
2) It is a reference type.
3) Infect class is a tag or template for object.

INHERITANCE

Inheritance is a process of deriving the new class from already existing class.

Inheritance is a feature of object-oriented programming that allows code reusability when a class includes property of another class. Considering Human Being a class, which has properties like hands, legs, eyes, mouth, etc, and functions like walk, talk, eat, see, etc.

ENCAPSULATION

Encapsulation means that we want to hide unnecessary details from the user. For example, when we call from our mobile phone, we select the number and press call button. But the entire process of calling or what happens from the moment we press or touch the call button to the moment we start having a phone conversation is hidden from us.

ABSTRACTION

Abstraction is "To represent the essential feature without representing the background details."

Abstraction lets you focus on what the object does instead of how it does it.

Abstraction provides you a generalized view of your classes or objects by providing relevant information.

The concept of abstraction focuses on what an object does, instated of how an object is represented or “how it works.” Thus, data abstraction is often used for managing large and complex programs.

REAL-WORLD EXAMPLE OF ABSTRACTION

A Laptop consists of many things such as processor, motherboard, RAM, keyboard, LCD screen, wireless antenna, web camera, usb ports, battery, speakers etc. To use it, you don't need to know how internally LCD screens, keyboard, web camera, battery, wireless antenna, speaker’s works. You just need to know how to operate the laptop by switching it on.

POLYMORPHISM

Polymorphism means one thing in many form. Basically polymorphism is capability of one object to behave in multiple ways. Example: A man role changes at home, college, and outside the home. There are following types of polymorphism:

1. ***Static polymorphism(compile time) :*** It is achieved using function overloading and operator overloading.
2. ***Dynamic polymorphism(runtime time) :*** It is achieved using function overriding means using virtual function.

❖ EVENT DRIVEN PROGRAMMING

In the conditions of event-based programming, objects (i.e., users) can initiate some events ('fire events') in the program, and the next thing happening in it is determined by those events. As a result, event-based programming fosters the dynamic interaction between users and computers.

After you design the look of your ASP.NET page using ASP.NET controls, you need to add the application logic. ASP.NET introduces the concept of an event-driven programming. It enables you to write code that executes in response to events raised by particular user actions. Events can be related to the page or the controls on the page. For example, you can write code to make your application do something when the page loads or a button is clicked.

HANDLING PAGE EVENTS

Whenever you request an ASP.NET page, a particular set of events is raised in a particular sequence. This sequence of events is called the page execution lifecycle.

Following is the sequence of events raised whenever you request a page:

1. PreInit
2. Init
3. InitComplete
4. PreLoad
5. Load
6. LoadComplete
7. PreRender
8. PreRenderComplete
9. SaveStateComplete
10. Unload