1. Explain various State management policies in ASP.Net

- You can preserve the state of the application either at the server or the client end.
- The state of a web application helps you to store the runtime changes that have been made to the web application.
- For example, a user selects and saves some products in the shopping cart of an online shopping websites.
- For that you have to store it to state, otherwise the changes are discarded.
- There are various methods for storing state information.
  - Hidden fields
  - Cookies
  - Query Strings
  - Application state
  - Session State
  - Profile Properties

- **Page-Level State [View State]**

  - In Asp.Net, the ViewState property is used to store the page-level state of a web page, the ViewState property the state information of a single user, as long as the user is working with the page.
  - The ViewState property is used when a form is submitted and presented second time to a user, as it retains the information entered in the form's controls first time.
  - The VIEWSTATE property at each page is initialized and stored in ViewState.
  - The string is assigned to the Value attribute of VIEWSTATE field and is sent as a part of the web page.

```
ViewState["User-definedKey"] = user-definedValue;

Response. Write(ViewState["user-DefinedKey"]);
```

- **Cookies**
  - Cookie is a small amount of data that is stored as a text file on client system or in client browser's session.
  - It stores site specific information of a user, such as user name and inputs.
  - Cookies can be created temporarily with the specific expiration time.
  - Cookies can be created permanent, called as persistent cookies.
  - When a page is requested, the client sends information in form of cookies along request information.
  - The server reads cookie and extract value.

```
HttpCookie cookie = new HttpCookie("userinfo");
    cookie["nm"] = TextBox1.Text;
    cookie["rn"] = TextBox2.Text;
    cookie.Expires = DateTime.Now.AddDays(30);
Response.Cookies.Add(cookie);
```

ASP.Net Session State

- Session is defined as the period of time, in which a user interacts with a web application.
- Session state is a collection of data, which are related to a session and stored on a server.
- Session state acts as memory in the form of a hash table with key-value pair.
- Each user is given unique session ID when the session begins.
- Session state can be implemented in the following Modes:

In-Process

- In this mode, session state memory is kept within the Asp.Net process. This mode applies to the web applications that are hosted on a single server.

Out-of-Process

- In this mode, you get performance of reading from memory and the reliability of a separate process that manages the state for all servers.

SQL Server

- Here reliability of data within a web application with help of database.

Configuring Session state

- Following parameters with example coding:

```
<configuration>
<sessionState
   Mode = “inproc”
   Cookieless=”false”
   Timeout=”20”
   Sqlconnectionstring= “data source=127.0.0.1;user id = <userid>; password =
   <Password>” Server=”127.0.0.1”
   Port=”42424”/>
  </configuration>
```

2. AJAX controls with Update Panel

- AJAX is asynchronous Java Script and XML
- AJAX is not a programming language for developing websites.
- It is a collection of technologies to refresh the highly dynamic page according to page update approach not according to page replacement approach.
- AJAX controls are very useful because in a highly loaded graphical web page to refresh the content of small control, we have to reload the whole page, it took too much time.
- As well as tedious for user to use such web sites.
- But AJAX controls are capable that they only update the data of required contents only. Not the whole page.
- So page is quick updated.

- And easy to use graphically loaded web pages for highly dynamic interactions.

Script manager Control

- It helps to implement AJAX functionalities in Asp.Net website.
- On which page you want AJAX functionalities you have to add Script manager control.
- It is responsible for managing client scripts for AJAX enabled website.
- Without Script Manager Control AJAX functionalities are useless.

Update Panel Control

```
<form id="form1" runat="server">
   <asp:ScriptManager ID="ScriptManager1" runat="server" />
   <asp:UpdatePanel runat="server" id="UpdatePanel"
   updatemode="Conditional"> <Triggers>
   <asp:AsyncPostBackTrigger controlid="UpdateButton2" eventname="Click"
   /> </Triggers>
   <ContentTemplate>
   <asp:Label runat="server" id="DateTimeLabel1" />
    <asp:Button runat="server" id="UpdateButton1" onclick="UpdateButton_Click"
text="Update" />
       </ContentTemplate>
    </asp:UpdatePanel>
    <asp:UpdatePanel runat="server" id="UpdatePanel1" updatemode="Conditional">
     <ContentTemplate>
     <asp:Label runat="server" id="DateTimeLabel2" />
     <asp:Button runat="server" id="UpdateButton2"
onclick="UpdateButton_Click" text="Update" />
       </ContentTemplate>
     </asp:UpdatePanel>
  </form>
```

Code Behind File :

```
protected void UpdateButton_Click(object sender, EventArgs e)
{
DateTimeLabel1.Text = DateTime.Now.ToString();
DateTimeLabel2.Text = DateTime.Now.ToString();
}
```

3. Silverlight in ASP.Net

- Silver light is a web based technology that allows web designers and developers to stretch the boundaries of web application development.
- It is an integration of the rich user interface of desktop applications and the transparency of other web languages, such as html and java script.
- Silver light allows you to develop web applications that contain high-fidelity multimedia content and eye catching visual effects.
- The primary components of silver light include HTML, Java script, XAML and .net Framework.

- While developing silver light applications you will be using the combination of varied features of these technologies.

  It has two main features:
  - .net framework for silver light
  - Core presentation framework

.Net framework for silver light

- Silver light comes with a smaller version of .net framework.
- The .net frame work in silver light component also enables you to easily access data and services at remote locations through WCF.
- In addition, you can use HTTP objects, cross-domain HTTP requests, Really simple Syndication (RSS) feeds, JSON and SOAP services in Silver light application.

Core Presentation Framework

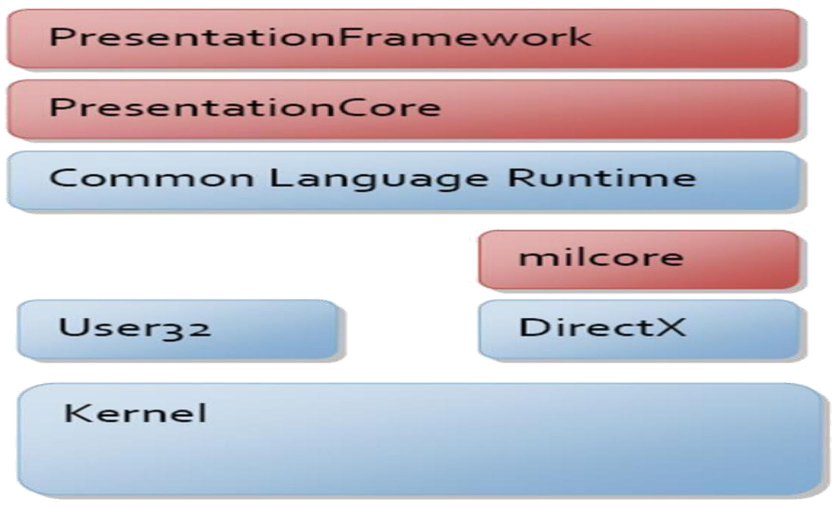
- It is a component in Silver light that offers the features and services for designing the UI of silver light applications through controls, styles, templates, 2-d vectors graphics, animations and multimedia.
- This component also allows you to incorporate user input and interactivity through various input Devices.
- Moreover you can also manage the digital rights of multimedia contents used in silver light application through this component.

Silver light controls

- The controls in silver light correspond to different XAML elements.
- E.g Border, button, Canvas, Content Control, Data Grid, data Picker, Image etc.

4. Explain WPF with its features.

- Windows Presentation Foundation (WPF) is a means for the programmer to create Windows applications possessing rich user interfaces and graphics which the classic .NET Windows applications lack. The initial version of WPF was released as a part of .NET 3.0 and it was really like a preview of WPF itself. The actual version of WPF was released as a part of .NET Framework 3.5.
  - High level architecture of WPF:



Features of WPF

- **Declarative Programming**
  - WPF application paves the way for the developers to define the thick client UI in a declarative way, which was never supported by the traditional .NET windows forms. Tasks like defining a template for a control ,creating a control hierarchy and similar work would be much easier if it is done in a declarative fashion.
  - In WPF declarative programming was made possible with the introduction of Extensible Application Markup Language (XAML). It can be compared to the HTML part in a web user interface.
- **Independent of screen resolution**
  - This is a neat feature of WPF. What I mean by independency of screen resolution is, the WPF user interface will look better even on a screen with low resolution.
  - It uses DirectX components whereas the WindowsForms applications make use of the **User 32** components of a machine.
  - WPF framework has the **Media Integration Layer (MIL)** in order to talk to the **DirectX** components.
  - The direct components impose a vector based graphics on the WPF user interface. The below screen shot will show you the difference between the Windows forms UI and WPF UI at lower resolutions of the screen.
- **Control Inside a control**
  - WPF allows you to provide not only the text but it also allows you to define a control as a content of another basic control like a Button.
  - This feature is truly astonishing fact for the developers and this lets the world know, what the power of WPF is when it comes to user interfaces.
  - You can have a Text Box inside a Button for example.
- **Control transforms**
  - WPF contains a handful of 2D transforms which will enable you to change the size, position, rotation angle and also allows skewing.
  - Control transforms can be performed in two ways LayoutTransform and RenderTransform.
  - Rotate transform
  - Scale transform

- Skew transform
- Translate transform
- Matrix transformation

- **Control Templates**
  - What if the user wants to change the shape of a button, this will definitely sound weird for .NET developers.
  - Now it can be done in WPF by defining the control template. For example you can declare a Button on your WPF window and can change its shape to elliptical.
  - **2D, 3D, graphics, animation and Media**

5. What is web Service in .Net? Write a Program to create Web Service and another program to consume same web service. (Take simple webmethod to add two integer numbers).

- A web service is an entity that you can develop to provide particular functionality over the internet.
- For example weather information, live score of cricket, simple and compound interest formula.
- For such purposes you make a web service for each. It will useful not only for one website but also for many other websites those are providing same facilities.
- Developer just needs to consume the web services and then its required function is working.
- So, it is better to develop a web service common to all instead of writing the same business logic for all websites.
- And moreover web service provides common functionalities and results for all websites.
- And web service irrespective from any programming language as it is based on XML so can be useful to all language based applications.
- Web Service messages are formatted as XML, a standard way for communication between two incompatible system. And this message is sent via HTTP, so that they can reach to any machine on the internet without being blocked by firewall.
- **Take an example to demonstrate web service**

- Right click on project → Add new item → select web services → OK

```
public class WebService1 : System.Web.Services.WebService
 {
 [WebMethod]
 public string HelloWorld() { return "Hello World"; }
 [WebMethod]
        public int sum(int a, int b)
        {
        return a+b;
        }
 }
```

- Now you have to **add web reference** to achieve web services working for any application.
  - Web services for solution only
  - Web services for local machine
  - Web services over internet

Consume web service

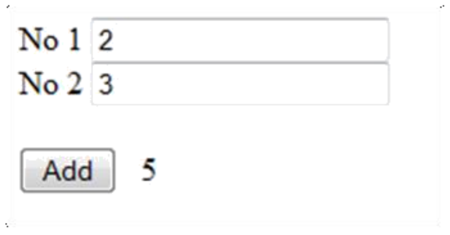
- Make an object of web service and you can call any method within that web service.

```
public partial class counter : System.Web.UI.Page
{
protected void Page_Load(object sender, EventArgs e){}
protected void btnAdd_Click(object sender, EventArgs e)
{
WebService1 intsum = new WebService1();
int a = Convert.ToInt32(txt1.Text);
int b = Convert.ToInt32(txt2.Text);
lblSum.Text=intsum.sum(a,b).ToString();
}
```

Designer Page:

```
No 1<asp:TextBox ID="txt1" runat="server"></asp:TextBox><br />
No 2<asp:TextBox ID="txt2" runat="server"></asp:TextBox><br /><br />
<asp:Button ID="btnAdd" runat="server" Text="Add" onclick="btnAdd_Click" /> &nbsp;
<asp:Label ID="lblSum" runat="server"></asp:Label>
```

Output Would be:



6. Explain WCF with Example.

- Windows Communication Foundation (WCF) is a framework for building service-oriented applications.
- Using WCF, you can send data as asynchronous messages from one service endpoint to another.
- A service endpoint can be part of a continuously available service hosted by IIS, or it can be a service hosted in an application.
- An endpoint can be a client of a service that requests data from a service endpoint.
- The messages can be as simple as a single character or word sent as XML, or as complex as a stream of binary data.

A few sample scenarios include:

- A secure service to process business transactions.
- A service that supplies current data to others, such as a traffic report or other monitoring
- service.
- A chat service that allows two people to communicate or exchange data in real time.
- A dashboard application that polls one or more services for data and presents it in a logical
- presentation.
- Exposing a workflow implemented using Windows Workflow Foundation as a WCF service.
- A Silver light application to poll a service for the latest data feeds.
- In summary, WCF is designed to offer a manageable approach to creating Web services and
- Web service clients.

Features of WCF

- Service Orientation
- Interoperability
- Multi message patterns
- Service Metadata and data contracts
- Security
- Durable messages
- Transactions
- AJAX and REST support
- Extensibility

7. Explain various Rich server Controls in ASP.Net.

- Rich server controls are used to deploy rich applications in very easy manner.
- Such as to rotate advertisements in webpage.
- Rich customized online calendar control.
- **Ad rotator**

- Ad rotator controls are used to rotate advertisements on web screen using XML.
- XML file make the control more effective.

Example Ad rotator :

```
 <asp:AdRotator ID="AdRotator1" runat="server" DataSourceID="XmlDataSource1"
  Height="100px" Width="200px" />
<asp:XmlDataSource ID="XmlDataSource1" runat="server" DataFile="~/Data.xml">
</asp:XmlDataSource>
```

<u>Data.xml file</u>

```
____<?xml version="1.0" encoding="utf-8"
          ?> <Advertisements>
           <Ad>
             <ImageUrl>123.jpg</ImageUrl>
             <NavigateUrl>http://www.google.com</NavigateUrl>
             <AlternateText>go to google</AlternateText>
             <Impressions>80</Impressions>
             <keyword>search</keyword>
           </Ad>
           <Ad>
             <ImageUrl>234.jpg</ImageUrl>
             <NavigateUrl>http://www.youtube.com</NavigateUrl>
             <AlternateText>go to youtube</AlternateText>
             <Impressions>30</Impressions>
             <keyword>search</keyword>
           </Ad>
           <Ad>
             <ImageUrl>789.jpg</ImageUrl>
             <NavigateUrl>http://www.amazon.in</NavigateUrl>
             <AlternateText>go to amazon</AlternateText>
             <Impressions>70</Impressions>
             <keyword>Shopping</keyword>
           </Ad>
          </Advertisements>
```

▪ **Calendar Control**

- Calendar controls are used to maintain online calendar.
- Customized calendar is developed with reminders, birthdays etc.

Example Calendar Control:

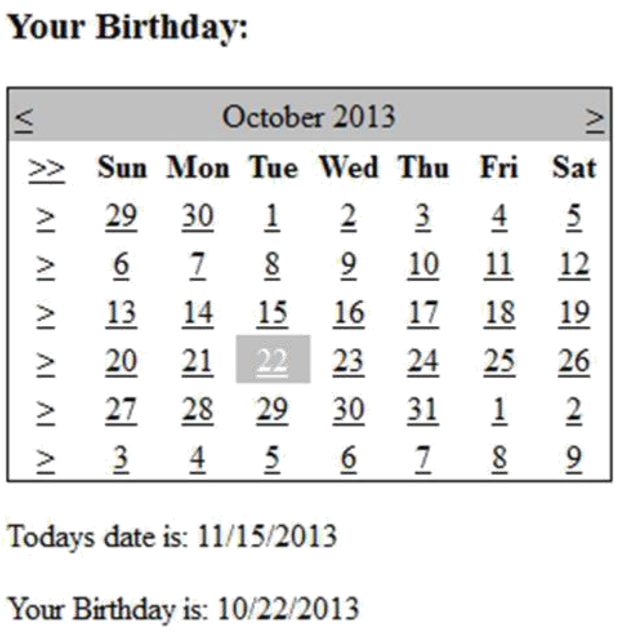
<u>Designer File :</u>

```
<h3> Your Birthday:</h3>
<asp:Calendar ID="Cal1" runat="server" SelectionMode="DayWeekMonth"
        onselectionchanged="Calendar1_SelectionChanged">
</asp:Calendar>
</div>
<p>Todays date is: <asp:Label ID="lblday" runat="server"></asp:Label></p>
<p>Your Birthday is: <asp:Label ID="lblbday" runat="server"></asp:Label> </p>
```

Code Behind file :

```
 protected void Calendar1_SelectionChanged(object sender, EventArgs e)
{
   lblday.Text = Calendar1.TodaysDate.ToShortDateString();
   lblbday.Text = Calendar1.SelectedDate.ToShortDateString();
}
```

Output is :


Your Birthday:

<			October 2013				>
>>	Sun	Mon	Tue	Wed	Thu	Fri	Sat
>	29	30	1	2	3	4	5
>	6	7	8	9	10	11	12
>	13	14	15	16	17	18	19
>	20	21	22	23	24	25	26
>	27	28	29	30	31	1	2
>	3	4	5	6	7	8	9

Todays date is: 11/15/2013

Your Birthday is: 10/22/2013

8. Explain various Data Controls of ASP.Net

- Data controls are used to display records from database in a repeated manner.
- The results can be customized according to the query style and user preferences.
- Custom edit and paging and select record type functionalities are ready made available.
- Just drag and drop data controls and display the records according to your need.
- Some Controls are mentioned and explained below:

Grid View, Data List, Repeater Control

Grid View :

```
<asp:GridView ID="GridView1" runat="server" AllowPaging="True" AllowSorting="True"
AutoGenerateColumns="False" DataKeyNames="Id" DataSourceID="SqlDataSource1">
      <Columns>
        <asp:BoundField DataField="Id" HeaderText="Id" InsertVisible="False"
          ReadOnly="True" SortExpression="Id" />
        <asp:BoundField DataField="Name" HeaderText="Name" SortExpression="Name"
        /> <asp:BoundField DataField="College" HeaderText="College"
          SortExpression="College" />
      </Columns>
    </asp:GridView>
    <asp:SqlDataSource ID="SqlDataSource1" runat="server"
      ConnectionString="<%$ ConnectionStrings:ConnectionString %>"
     SelectCommand="SELECT * FROM [Student]"></asp:SqlDataSource><br />
```

The Output :

Id	Name	College
1	abc	AITS
2	xyz	vvp
3	efg	AITS

Repeater :

```
  <asp:Repeater ID="Repeater1" runat="server" DataSourceID="SqlDataSource2">
    <HeaderTemplate>
      <table border=1 width="150px">
        <tr><td>Id</td>
        <td>Name</td>
        <td>College</td></tr> </table>
    </HeaderTemplate>
    <ItemTemplate>
      <table border=1 width="150px">
        <tr><td><asp:Label ID="id" Text='<%#Eval("Id") %>'
runat="server"> </asp:Label></td>
        <td><asp:Label ID="Name" Text='<%#Eval("Name") %>'
runat="server"> </asp:Label></td>
        <td><asp:Label ID="College" Text='<%#Eval("College") %>' runat="server">
</asp:Label></td></tr>
      </table>
    </ItemTemplate>
</asp:Repeater>
    <asp:SqlDataSource ID="SqlDataSource2" runat="server"
      ConnectionString="<%$ ConnectionStrings:ConnectionString %>"
      SelectCommand="SELECT * FROM [Student]"></asp:SqlDataSource>
```

The Output :

<u>Id</u>	<u>Name</u>	<u>College</u>
1	abc	AITS
2	xyz	vvp
3	efg	AITS

<u>**DataList :**</u>

```
<asp:DataList ID="DataList1" runat="server" DataKeyField="Id"
        DataSourceID="SqlDataSource3" RepeatDirection="Horizontal">
<ItemTemplate>
Id:<asp:Label ID="Id" runat="server" Text='<%# Eval("Id") %>' /><br />
Name:<asp:Label ID="Name" runat="server" Text='<%# Eval("Name") %>' />
College:<asp:Label ID="College" runat="server" Text='<%# Eval("College") %>'
/> </ItemTemplate>
</asp:DataList>
    <asp:SqlDataSource ID="SqlDataSource3" runat="server"
     ConnectionString="<%$ ConnectionStrings:ConnectionString %>"
      SelectCommand="SELECT * FROM [Student]"></asp:SqlDataSource>
```

The Output :

```
Id : 1          Id : 1         Id : 1
Name :abc       Name :xyz      Name :efg
College:AITS    College:vvp    College:AITS
```

9. Explain various Server Controls of ASP.Net.

- Asp.net provides many built in server controls that can be easily drag and drop to the web pages.
- **There are various Controls like** Text Box,Check Box , Radio Button ,Label **,** Button ,Hyper Link ,Image Etc.
- **Some Common Properties for all controls like** ID ,Runat ,Text ,CssClass ,Height , Width ,Auto post back ,Visible etc.
- All the controls are available in System.Web.UI.WebControls namespace.
- Server side Controls are useless without runat="server" Property.

Text Box:

- This control is used to take input from user into a default string format.

```
<asp:TextBox      ID="TextBox1"      runat="server"      AutoPostBack="true"
height="10px" width="10px" visible="true">
</asp:TextBox>
```

Example:

Enter your Name….

Check Box:

- This control is used to select multiple values from a list.

```
<asp:CheckBox ID=”CheckBox1” runat=”server” AutoPostBack=”true” visible=”true”>
<asp:CheckBox>
```

Example:

Apple ☐ Banana ☐ Grapes ☐ Guava ☐

Radio Button:

- This control is used to select Single value from a list.

```
<asp:RadioButton ID=”RadioButton1” runat=”server” AutoPostBack=”True”
Visible=”True” Group=”one”>
</asp:RadioButton>
<asp:RadioButton ID=”RadioButton1” runat=”server” AutoPostBack=”True”
Visible=”True” Group=”one”>
</asp:RadioButton>
```

Example:

Male ○ Female ○

Label:

- This Control is used to display information on the web Page.

```
<asp:Label ID=”Label1” runat=”server” Visible=”True” Text=”Hello Good Morning ”>
<asp:Label>
```

Example:

Hello Good Morning

- **Button:**
    - This control provides firing of various events at a single click.
    - Button Provides page navigation as well as event firing.

```
<asp:Button ID=”Buttton1” runat=”server” Text=”Click Me” OnClick=”Button1_Click” />
```

Example:

Click me

- **Hyperlink:**
    - This control provides easy navigation between various Pages.
    - NavigateUrl = address of destination page

```
<asp:HyperLink ID="HyperLink1" runat="Server" NavigateUrl="Home.aspx" Text="Home
Page"></asp:HyperLink>
```

Example:

Home → navigate to Home page on single Click

- **Image:**
  - This Control is used to display images on the web page.
  - ImageUrl → Location of the image in application folder or in computer.
  - AlternateText= alternate text if image is not found

```
<asp:Image ID="Img" runat="server" ImageUrl="~/images/123.jpg" AlternateText="natural"/>
```

Example:

10. What is Exception in C#? Explain Exception handling in C# with Example.

- An exception is a problem that arises during unsafe code executes of a program.
- Exceptions provide a way to transfer control from one part of a program to another.
- C# exception handling is built upon four keywords: **try**, **catch**, **finally** and **throw**.
- **try:** A try block identifies a block of code for which particular exceptions will be activated. It's followed by one or more catch blocks.
- **catch:** A program catches an exception with an exception handler at the place in a program where you want to handle the problem. The catch keyword indicates the catching of an exception
- **finally:** The finally block is used to execute a given set of statements, whether an exception is thrown or not thrown. For example, if you open a file, it must be closed whether an exception is raised or not.
- **throw:** A program throws an exception when a problem shows up. This is done using a throw keyword.
- A try/catch block is placed around the code that might generate an exception.
- Code within a try/catch block is referred to as protected code.
- The exception classes in C# are mainly directly or indirectly derived from the **System.Exception** class.

- Some of the exception classes derived from the System.Exception class are the System.ApplicationException and System.SystemException classes.

For Example :

```
using System;
namespace ErrorHandlingApplication
{
  class DivNum
  {
    int result;
    DivNum()
    {
      result = 0;
    }
    public void division(int num1, int num2)
    {
      try
      {
        result = num1 / num2;
      }
      catch (DivideByZeroException e)
      {
        Console.WriteLine("Exception caught: {0}", e);
      }
      finally
      {
        Console.WriteLine("Result: {0}", result);
      }

    }
    static void Main(string[] args)
    {
      DivNum d = new DivNum();
      d.division(25, 0);
      Console.ReadLine();
    }
   }
 }
```

11. Explain C# Modifiers.

PUBLIC	The item is visible to any other code, no restrictions on the use of a class declared asPublic.
PRIVATE	The item is visible only inside the type to which it belongs, members in class A that are marked private are accessible only to methods of class A.
PROTECTED	The item is visible only to any derived type, members in class A that are marked protected are accessible to methods of class A and also to methods of classes derived from class A.
INTERNAL	The item is visible only within its containing assembly, members in class A that are marked internal are accessible to methods of any class in A’s assembly.
INTERNAL PROTECTED	The item is visible to any code within its containing assembly and also to any code inside a derived type, members in class A that are marked protected internal are accessible to methods of class A, to methods of classes derived from class A, and also to any class in A’s assembly.

Note : In VB .Net we are using FRIEND and PROTECTED FRIEND instead of INTERNAL and INTERNAL PROTECTED Respectively.