**ASP.NET 2017**

**Q-1[A]ONE MARK:**

1]the .net framework provides a runtime enviroment called.. common language runtime

2]full form of FCL- Framework Class Library

3]full form of IIS- Internet Information Services

4]which is the property of button which is used for redirect to another webpage?

a. Use the JavaScript window. location Property You can simply use the JavaScript window. location property to make a page redirect

5]to set the image for the image control we can use the which property?

a. ImageUrl property

6]which control is used to give single select option to the user from multiple items?

a. dropdownlist

7]what is the default value of AutoPostBack property?

a. Default value of AutoPostBack on control is false.

8]extension of master page is

a. extension **.** master (for example, MySite. master)

9]define true or false:we can add multiple theme in website.

a.y ou cannot directly apply an ASP.NET theme to a master page. If you add a theme attribute to the @ Master directive, the page will raise an error when it runs.

10]validation control are checked at both side

a.These controls provides both server side and client side validation. ...

11]application state method is used to store some which for your web application.

a. Application state is a global storage mechanism that used to stored data on the server .

12]which namespace is imported to use SQL.server provider clasees?

a. SqlClient namespace is the . NET Data Provider for SQL Server.

13]what is the mappath()?

a. The MapPath method maps a specified path to a physical path.

MapPath Method When you need to know the exact location of a file, use the Server.MapPath method.

14]what is page directive?

a. Basically Page Directives are commands. These commands are used by the compiler when the page is compiled.

The Page directive defines the attributes specific to the page file for the page parser and the compiler.

15]which property of control is used to specify client side script that executes when the which control is used ?

a. server controls use client side scripting to provide response to the users without posting back to the server.t he Button control has a property OnClientClick, which allows executing client-side script, when the button is clicked.

16]which feature of ASP.NET able to log their applicaton by means to monitor how your application executes?

a. ASP.NET includes performance counters that you can use to track the execution of your application. You can also use the built-in ASP.NET tracing feature to track code execution for a page or an application.

17]which file has .asmx extension in asp.net?

a.

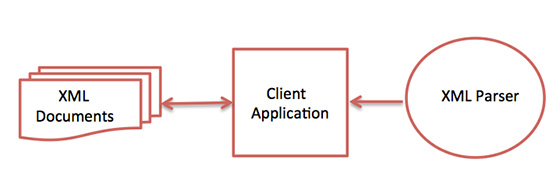
18]which namespaces should be imported in order to read any configuration information especially Webconfiguration Manager Class.

A. System.Web.Configuration

19]what is XML parser?

a. XML parser is a software library or a package that provides interface for client applications to work with XML documents. It checks for proper format of the XML document and may also validate the XML documents. ... The goal of a parser is to transform XML into a readable code.

Following diagram shows how XML parser interacts with XML document −



20]which method do you invoke on the DataAdapter control to load your generated datasetwith data?

a. Fill( )

**Q-2[A]TWO MARK:**

1]advantages of ASP.NET

1. a.**Less Coding and Increased Reuse of Code:**

This framework works on object-oriented programming which eliminates unnecessary codes and involves less coding for the developers. .NET consists of re-useable code and many re-useable components.  This translates into less time and consequently less cost to [develop applications](https://www.invensis.net/it-outsourcing-services/outsource-software-application-development-services).

1. **Deployment:**

With features such asno-impact applications, private components, controlled code sharing, side-by-side versioning and partially trusted code, the [.NET framework](https://www.invensis.net/it-outsourcing-services/outsource-dot-net-application-development-services) makes deployment easier post development.  The code execution environment supports safe code execution for reduced conflicts in software deployment and versioning, and minimized performance problems of scripted or interpreted environments.

1. **Reliability:**

Since its release in 2002, .NET has been used to develop thousands of applications.  Its performance on Microsoft® Windows Server™ 2003 and Windows 2000 Server is also very stable and reliable.

1. **Security:**

.NET offersenhanced application security as web applications developed using ASP. NET have Windows confirmation and configuration. Managed code and CLR offer safeguard features such as role-based security and code access security.

1. **Use across Platforms and Languages:**

. NET allows developers to develop applications for a desktop, a browser, a mobile browser (like on your cell phone), or an application running on PDA. .NET is promoted as a language-independent framework, which implies that development can take place in different compliant languages that include C#, managed C++, VB.NET, Visual COBOL, IronPython, IronRuby and more.

1. **Use for Service-Oriented Architecture:**

.NET is often used for Web Services, which are a solution for executing an SOA strategy. Through Web Services, applications which are designed in different programming languages or platforms, are able to communicate and transmit data utilizing standard Internet protocols.

1. **Integration with Legacy Systems:**

Thecapability of .NET to process all types of XML documents and write any format of file with swiftness and ease, provides multiple routes for integration.

2]explain application web servers

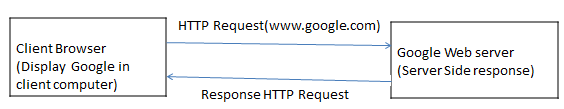
a. An application server is a server that hosts applications.[1]

Application server frameworks are software frameworks for building application servers. An application server framework provides both facilities to create web applications and a server environment to run them.

An application server framework contains a comprehensive service layer model. It includes a set of components accessible to the software developer through a standard API defined for the platform itself. For Web applications, these components usually run in the same environment as their web server(s), and their main job is to support the construction of dynamic pages. However, many application servers do more than generate web pages: they implement services such as clustering, fail-over, and load-balancing, so developers can focus on implementing the business logic.[2]

3]explain HTTP

A. Http is a stateless protocol. It is known as Hypertext Transfer protocol.It is used to send the client request to the web server.All the Browser such as [google chrome](https://www.google.com/chrome) ,[Mozilla Firefox](http://www.mozilla.org/en-US/firefox),[Netscape](http://netscape.aol.com/) and [Internet Explorer](http://windows.microsoft.com/en-us/internet-explorer/download-ie%E2%80%8E) are clients that helps to send a request to the web server.After verification of credentials,web server response the client's request through the HTTP protocol.

[](https://1.bp.blogspot.com/-Mi3zsjMvgy0/UlhIr8vvQeI/AAAAAAAACFQ/HuLiSh_Ix0s/s1600/http.PNG)

4]explain page output caching

a. Output Caching : Output cache stores a copy of the finally rendered HTML pages or part of pages sent to the client.

When the next client requests for this page, instead of regenerating the page, a cached copy of the page is sent, thus saving time.

5]explain textbox control box with its properties.

a. TextBox control is a rectangular box which is used to take user to input. In simple word the TextBox is a place where user can input some text on asp.net web form. To use TextBox on page we can write code or just drag and drop from toolbox

Syntax of ASP.Net TextBox

* <asp:TextBox ID=”TextBox1″ runat=”server”></asp:TextBox>

|  |  |
| --- | --- |
| **Properties** | **Description** |
| ID | Identification name of textbox control. |
| Text | It is used to display text in a control. |
| BackColor | It is used to set background color of textbox control. |
| ForColor | It is used to set text color of the control. |
| ToolTip | It displays a text on control when mouse over on it. |
| TabIndex | It is used manage tab order of control. |
| CssClass | It is used to apply style on control. |
| Enable | true/false – used to enable or disable control. |
| Enable Theming | true/false – It is used to enable or disable effect of theme on control. |
| CausesValidation | true/false – It is used to enable or disable validation effect on control |
| Visible | true/false – It is used to hide or visible control on web page. |
|  | **Important Properties of TextBox control** |
| MaxLenght | It is used to set maximum number of characters that can be input in TextBox. |
| TextMode | Single / Multiline / Password |
| ReadOnly | true/false – used to enable or disable control readonly. |

6]explain viewstate technique

a. The ASP.NET ViewState is a client side state management technique which enables web pages to persist their state during postbacks. In the life cycle of a page, the current state of the page is hashed to a string and is saved into a hidden field. When opening a page with the View Source operation you can find the ViewState's hidden field by searching \_\_VIEWSTATE keyword. An example of a ViewState on a web page can look like that:

<input type="hidden" name="\_\_VIEWSTATE" id="\_\_VIEWSTATE" value="/wEPDwUBMA9kFgJmD2QWAgIDD2QWBgI" />

**Q-2[B]THREE MARK:**

1]explain adroater control with example.

a. AdRotator is a control in ASP.NET that is all about ads in Web pages. It displays a number of ad images and rotates them or loads them when a page is refreshed. The ASP.NET AdRotator control uses an XML file to store the ads and their information. The XML file must begin and end with an <Advertisements> tag. Inside the <Advertisements> tag there may be several <Ad> tags and each tag defines an ad (we will see a sample XML in this article).

The basic syntax of adding an AdRotator is as follows:

<asp:AdRotator runat = "server" AdvertisementFile = "adfile.xml" Target = "\_blank" />

The Advertisement File

The advertisement file is an XML file, which contains the information about the advertisements to be displayed.

Properties and Events of the AdRotator Class

The AdRotator class is derived from the WebControl class and inherits its properties. Apart from those, the AdRotator class has the following properties:

Properties Description

AdvertisementFile The path to the advertisement file.

AlternateTextFeild The element name of the field where alternate text is provided. The default value is AlternateText.

DataMember The name of the specific list of data to be bound when advertisement file is not used.

DataSource Control from where it would retrieve data.

DataSourceID Id of the control from where it would retrieve data.

Font Specifies the font properties associated with the advertisement banner control.

ImageUrlField The element name of the field where the URL for the image is provided. The default value is ImageUrl.

KeywordFilter For displaying the keyword based ads only.

NavigateUrlField The element name of the field where the URL to navigate to is provided. The default value is NavigateUrl.

Target The browser window or frame that displays the content of the page linked.

UniqueID Obtains the unique, hierarchically qualified identifier for the AdRotator control.

Following are the important events of the AdRotator class:

Events Description

AdCreated It is raised once per round trip to the server after creation of the control, but before the page is rendered

DataBinding Occurs when the server control binds to a data source.

DataBound Occurs after the server control binds to a data source.

Disposed Occurs when a server control is released from memory, which is the last stage of the server control lifecycle when an ASP.NET page is requested

Init Occurs when the server control is initialized, which is the first step in its lifecycle.

Load Occurs when the server control is loaded into the Page object.

PreRender Occurs after the Control object is loaded but prior to rendering.

Unload Occurs when the server control is unloaded from memory.

* Following are the important events of the AdRotator class:

Events Description

AdCreated It is raised once per round trip to the server after creation of the control, but before the page is rendered

DataBinding Occurs when the server control binds to a data source.

DataBound Occurs after the server control binds to a data source.

Disposed Occurs when a server control is released from memory, which is the last stage of the server control lifecycle when an ASP.NET page is requested

Init Occurs when the server control is initialized, which is the first step in its lifecycle.

Load Occurs when the server control is loaded into the Page object.

PreRender Occurs after the Control object is loaded but prior to rendering.

Unload Occurs when the server control is unloaded from memory.

AdRotator Control Example in ASP.Net

**Step 1 –** Open Visual Studio –> Create a new empty web application.

**Step 2 –** Create New web page for display AdRotator control.

**step 3 –** Drag and drop AdRotator Control on web page from toolbox.

**step 4 –** Right click on Solution Explorer –> **Add New Item** –> Add New**XML File** in project for write advertisement detail.

**step 5 –** Assign XML File to **AdvertisementFile** Property of AdRotator control.

2]explain RegularExpressionValidator control with example

a. RegularExpressionValidator is one of the most useful validators, because it can be used to check the validity of any kind of string.

This validator is used to validate the value of an input control against the pattern defined by a regular expression.

It allows us to check and validate predictable sequences of characters like: e-mail address, telephone number etc.

The ValidationExpression property is used to specify the regular expression, this expression is used to validate input control.

## RegularExpression Properties

|  |  |
| --- | --- |
| **Property** | **Description** |
| AccessKey | It is used to set keyboard shortcut for the control. |
| BackColor | It is used to set background color of the control. |
| BorderColor | It is used to set border color of the control. |
| Font | It is used to set font for the control text. |
| ForeColor | It is used to set color of the control text. |
| Text | It is used to set text to be shown for the control. |
| ToolTip | It displays the text when mouse is over the control. |
| Visible | To set visibility of control on the form. |
| Height | It is used to set height of the control. |
| Width | It is used to set width of the control. |
| ErrorMessage | It is used to set error message that display when validation fails. |
| ControlToValidate | It takes ID of control to validate. |
| ValidationExpression | It is used to set regular expression to determine validity. |

## Example

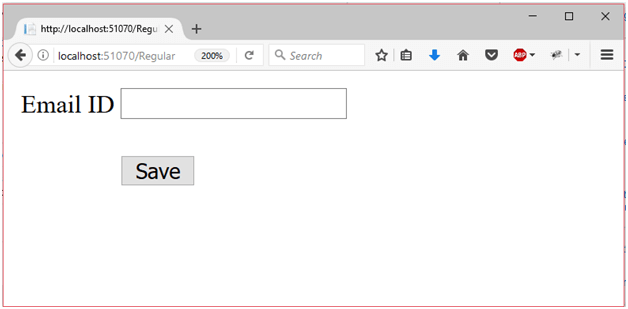
Here, in the following example, we are explaining how to use RegularExpressionValidator control to validate the user input against the given pattern.

**// RegularExpressionDemo.aspx**

1. **<**%@ Page Language="C#" AutoEventWireup="true" CodeBehind="RegularExpressionDemo.aspx.cs"
2. Inherits="asp.netexample.RegularExpressionDemo" %**>**
3. <!DOCTYPE html**>**
4. **<html** xmlns="http://www.w3.org/1999/xhtml"**>**
5. **<head** runat="server"**>**
6. **<title></title>**
7. **</head>**
8. **<body>**
9. **<form** id="form1" runat="server"**>**
10. **<div>**
11. **<table** class="auto-style1"**>**
12. **<tr>**
13. **<td** class="auto-style2"**>**Email ID**</td>**
14. **<td>**
15. **<asp:TextBox** ID="username" runat="server"**></asp:TextBox>**
16. **<asp:RegularExpressionValidator** ID="RegularExpressionValidator1" runat="server"ControlToValidate="username"
17. ErrorMessage="Please enter valid email" ForeColor="Red"ValidationExpression="\w+([-+.']\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*"**>**
18. **</asp:RegularExpressionValidator>**
19. **</td>**
20. **</tr>**
21. **<tr>**
22. **<td** class="auto-style2"**></td>**
23. **<td>**
24. **<br/>**
25. **<asp:Button** ID="Button1" runat="server" Text="Save"**/>**
26. **</td>**
27. **</tr>**
28. **</table>**
29. **</div>**
30. **</form>**
31. **</body>**
32. **</html>**

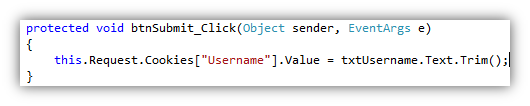
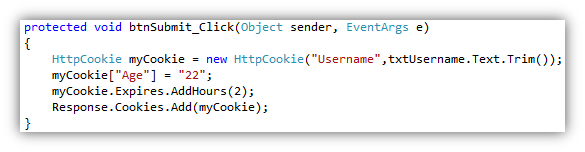
Output:

It produces the following output when view in the browser.



3]explain cookie state managemnet technique with example

a. ASP.Net provides another way of state management, which is by using Cookies. Cookies are one of the best ways of storing information. It is nothing but a text file which is stored on the client's machine.  
  
When the user sends a request to the server, the server creates a cookie and attaches a header and sends it back to the user along with the response. The browser accepts the cookie and stores it at a specific location on the client's machine. Even large sites like Gmail, Facebook, Yahoo use cookies.  
  
There are 2 ways of assigning / storing values in cookies.

1. **Using the Request Object**  
   **Information**   
     
   Here I have made use of the Request object. The Cookies property of the HTTPResponse Object and the HTTPRequest Object can be used to assign values to the Cookies Collection and get values back from the Collection. We can also store multiple values in a cookie.
2. **Using the HTTPCookies Object**  
   **Information**   
     
   Another way of adding values to a cookie is using the "HTTPCookie" class. Its constructor takes either 1 or 2 parameters. If you want your cookie to expire after a specified time then you can even set the expiration date for that cookie. And the last line "Response.Cookies.Add(myCookie)" will add that cookie to the Cookies Collection.

4]explain SOAP

a. SOAP is an acronym for Simple Object Access Protocol. It is an XML-based messaging protocol for exchanging information among computers. SOAP is an application of the XML specification.

Points to Note

* SOAP is a communication protocol designed to communicate via Internet.
* SOAP can extend HTTP for XML messaging.
* SOAP provides data transport for Web services.
* SOAP can exchange complete documents or call a remote procedure.
* SOAP can be used for broadcasting a message.
* SOAP is platform- and language-independent.
* SOAP is the XML way of defining what information is sent and how.
* SOAP enables client applications to easily connect to remote services and invoke remote methods.

A SOAP message is an ordinary XML document containing the following elements −

* **Envelope** − Defines the start and the end of the message. It is a mandatory element.
* **Header** − Contains any optional attributes of the message used in processing the message, either at an intermediary point or at the ultimate end-point. It is an optional element.
* **Body** − Contains the XML data comprising the message being sent. It is a mandatory element.
* **Fault** − An optional Fault element that provides information about errors that occur while processing the message.

5]datareader class

a. A data reader provides an easy way for the programmer to read data from a database as if it were coming from a stream.

This class is used to read data from SQL Server database. It reads data in forward-only stream of rows from a SQL Server database. it is sealed class so that cannot be inherited. It inherits DbDataReader class and implements IDisposable interface.

The DataReader properties

|  |  |
| --- | --- |
| PROPERTY | DESCRIPTION |
| Depth | Indicates the depth of nesting for row |
| FieldCount | Returns number of columns in a row |
| IsClosed | Indicates whether a data reader is closed |
| Item | Gets the value of a column in native format |
| RecordsAffected | Number of row affected after a transaction |

**Table 5-27.**The DataReader methods

|  |  |
| --- | --- |
| METHOD | DESCRIPTION |
| Close | Closes a DataRaeder object. |
| Read | Reads next record in the data reader. |
| NextResult | Advances the data reader to the next result during batch transactions. |
| Getxxx | There are dozens of Getxxx methods. These methods read a specific data type value from a column. For example. GetChar will return a column value as a character and GetString as a string. |

6]what is XML?discuss reading dataset from XML wIth example.

a. **XML** is a cross-platform, hardware and software independent, text based markup language, which enables you to store data in a structured format by using meaningful tags.

DO.NET provides simple methods for working with XML data. In this walkthrough, you create a Windows application that loads XML data into a dataset. The dataset is then displayed in a [DataGridView](https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.datagridview) control. Finally, an XML schema based on the contents of the XML file is displayed in a text box.

## Create a new project

Create a new **Windows Forms App** project for either C# or Visual Basic. Name the project **ReadingXML**.

## Generate the XML file to be read into the dataset

Because this walkthrough focuses on reading XML data into a dataset, the contents of an XML file is provided.

1. On the **Project** menu, select **Add New Item**.
2. Select **XML File**, name the file **authors.xml**, and then select **Add**.

The XML file loads into the designer and is ready for edit.

1. Paste the following XML data into the editor below the XML declaration:

XMLCopy

<Authors\_Table>

<authors>

<au\_id>172-32-1176</au\_id>

<au\_lname>White</au\_lname>

<au\_fname>Johnson</au\_fname>

<phone>408 496-7223</phone>

<address>10932 Bigge Rd.</address>

<city>Menlo Park</city>

<state>CA</state>

<zip>94025</zip>

<contract>true</contract>

</authors>

<authors>

<au\_id>213-46-8915</au\_id>

<au\_lname>Green</au\_lname>

<au\_fname>Margie</au\_fname>

<phone>415 986-7020</phone>

<address>309 63rd St. #411</address>

<city>Oakland</city>

<state>CA</state>

<zip>94618</zip>

<contract>true</contract>

</authors>

<authors>

<au\_id>238-95-7766</au\_id>

<au\_lname>Carson</au\_lname>

<au\_fname>Cheryl</au\_fname>

<phone>415 548-7723</phone>

<address>589 Darwin Ln.</address>

<city>Berkeley</city>

<state>CA</state>

<zip>94705</zip>

<contract>true</contract>

</authors>

<authors>

<au\_id>267-41-2394</au\_id>

<au\_lname>Hunter</au\_lname>

<au\_fname>Anne</au\_fname>

<phone>408 286-2428</phone>

<address>22 Cleveland Av. #14</address>

<city>San Jose</city>

<state>CA</state>

<zip>95128</zip>

<contract>true</contract>

</authors>

<authors>

<au\_id>274-80-9391</au\_id>

<au\_lname>Straight</au\_lname>

<au\_fname>Dean</au\_fname>

<phone>415 834-2919</phone>

<address>5420 College Av.</address>

<city>Oakland</city>

<state>CA</state>

<zip>94609</zip>

<contract>true</contract>

</authors>

</Authors\_Table>

1. On the **File** menu, select **Save authors.xml**.

## Create the user interface

The user interface for this application consists of the following:

* A [DataGridView](https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.datagridview) control that displays the contents of the XML file as data.
* A [TextBox](https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.textbox) control that displays the XML schema for the XML file.
* Two [Button](https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.button) controls.
  + One button reads the XML file into the dataset and displays it in the [DataGridView](https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.datagridview) control.
  + A second button extracts the schema from the dataset, and through a [StringWriter](https://docs.microsoft.com/en-us/dotnet/api/system.io.stringwriter) displays it in the [TextBox](https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.textbox) control.

### To add controls to the form

1. Open Form1 in design view.
2. From the **Toolbox**, drag the following controls onto the form:
   * One [DataGridView](https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.datagridview) control
   * One [TextBox](https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.textbox) control
   * Two [Button](https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.button) controls
3. Set the following properties:

| **TABLE 1** | | |
| --- | --- | --- |
| **Control** | **Property** | **Setting** |
| TextBox1 | **Multiline** | true |
|  | **ScrollBars** | **Vertical** |
| Button1 | **Name** | ReadXmlButton |
|  | **Text** | Read XML |
| Button2 | **Name** | ShowSchemaButton |
|  | **Text** | Show Schema |

## Create the dataset that receives the XML data

In this step, you create a new dataset named authors. For more information about datasets, see [Dataset tools in Visual Studio](https://docs.microsoft.com/en-us/visualstudio/data-tools/dataset-tools-in-visual-studio?view=vs-2019).

1. In **Solution Explorer**, select the source file for **Form1**, and then select the **View Designer** button on the **Solution Explorer** toolbar.
2. From the [Toolbox, Data tab](https://docs.microsoft.com/en-us/visualstudio/ide/reference/toolbox-data-tab?view=vs-2019), drag a **DataSet** onto **Form1**.
3. In the **Add Dataset** dialog box, select **Untyped dataset**, and then select **OK**.

**DataSet1** is added to the component tray.

1. In the **Properties** window, set the **Name** and [DataSetName](https://docs.microsoft.com/en-us/dotnet/api/system.data.dataset.datasetname) properties forAuthorsDataSet.

## Create the event handler to read the XML file into the dataset

The **Read XML** button reads the XML file into the dataset. It then sets properties on the [DataGridView](https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.datagridview) control that bind it to the dataset.

1. In **Solution Explorer**, select **Form1**, and then select the **View Designer** button on the **Solution Explorer** toolbar.
2. Select the **Read XML** button.

The **Code Editor** opens at the ReadXmlButton\_Click event handler.

1. Type the following code into the ReadXmlButton\_Click event handler:

C#Copy

private void ReadXmlButton\_Click(object sender, EventArgs e)

{

string filePath = "Complete path where you saved the XML file";

AuthorsDataSet.ReadXml(filePath);

dataGridView1.DataSource = AuthorsDataSet;

dataGridView1.DataMember = "authors";

}

1. In the ReadXMLButton\_Click event handler code, change the filepath = entry to the correct path.

## Create the event handler to display the schema in the textbox

The **Show Schema** button creates a [StringWriter](https://docs.microsoft.com/en-us/dotnet/api/system.io.stringwriter) object that's filled with the schema and is displayed in the [TextBox](https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.textbox)control.

1. In **Solution Explorer**, select **Form1**, and then select the **View Designer** button.
2. Select the **Show Schema** button.

The **Code Editor** opens at the ShowSchemaButton\_Click event handler.

1. Paste the following code into the ShowSchemaButton\_Click event handler.

C#Copy

private void ShowSchemaButton\_Click(object sender, EventArgs e)

{

System.IO.StringWriter swXML = new System.IO.StringWriter();

AuthorsDataSet.WriteXmlSchema(swXML);

textBox1.Text = swXML.ToString();

}

## Test the form

You can now test the form to make sure it behaves as expected.

1. Select **F5** to run the application.
2. Select the **Read XML** button.

The DataGridView displays the contents of the XML file.

1. Select the **Show Schema** button.

The text box displays the XML schema for the XML file.

## Next steps

This walkthrough teaches you the basics of reading an XML file into a dataset, as well as creating a schema based on the contents of the XML file. Here are some tasks that you might do next:

* Edit the data in the dataset and write it back out as XML. For more information, see [WriteXml](https://docs.microsoft.com/en-us/dotnet/api/system.data.dataset.writexml).
* Edit the data in the dataset and write it out to a database.

**Q-2[C]FIVE MARK:**

1]write note:session state

a. Use ASP.NET session state to store and retrieve values for a user.

ASP.NET session state enables you to store and retrieve values for a user as the user navigates ASP.NET pages in a Web application. HTTP is a stateless protocol. This means that a Web server treats each HTTP request for a page as an independent request.

**Session Identifiers**

Sessions are identified by a unique identifier that can be read by using the [SessionID](https://msdn.microsoft.com/en-us/library/856tc015(v=vs.100)) property. When session state is enabled for an ASP.NET application, each request for a page in the application is examined for a [SessionID](https://msdn.microsoft.com/en-us/library/856tc015(v=vs.100)) value sent from the browser. If no [SessionID](https://msdn.microsoft.com/en-us/library/856tc015(v=vs.100)) value is supplied, ASP.NET starts a new session and the [SessionID](https://msdn.microsoft.com/en-us/library/856tc015(v=vs.100)) value for that session is sent to the browser with the response.

By default, [SessionID](https://msdn.microsoft.com/en-us/library/856tc015(v=vs.100)) values are stored in a cookie. However, you can also configure the application to store [SessionID](https://msdn.microsoft.com/en-us/library/856tc015(v=vs.100)) values in the URL for a "cookieless" session.

A session is considered active as long as requests continue to be made with the same [SessionID](https://msdn.microsoft.com/en-us/library/856tc015(v=vs.100)) value. If the time between requests for a particular session exceeds the specified time-out value in minutes, the session is considered expired. Requests made with an expired [SessionID](https://msdn.microsoft.com/en-us/library/856tc015(v=vs.100)) value result in a new session.

The HttpSessionState class has the following properties:

|  |  |
| --- | --- |
| **Properties** | **Description** |
| SessionID | The unique session identifier. |
| Item(name) | The value of the session state item with the specified name. This is the default property of the HttpSessionState class. |
| Count | The number of items in the session state collection. |
| TimeOut | Gets and sets the amount of time, in minutes, allowed between requests before the session-state provider terminates the session. |

The HttpSessionState class has the following methods:

|  |  |
| --- | --- |
| **Methods** | **Description** |
| Add(name, value) | Adds an item to the session state collection. |
| Clear | Removes all the items from session state collection. |
| Remove(name) | Removes the specified item from the session state collection. |
| RemoveAll | Removes all keys and values from the session-state collection. |
| RemoveAt | Deletes an item at a specified index from the session-state collection. |

2]asp.net framework in detail

a. .NET is a framework to develop software applications. It is designed and developed by Microsoft and the first beta version released in 2000.

It is used to develop applications for web, Windows, phone. Moreover, it provides a broad range of functionalities and support.

This framework contains a large number of class libraries known as Framework Class Library (FCL). The software programs written in .NET are executed in the execution environment, which is called CLR (Common Language Runtime). These are the core and essential parts of the .NET framework.

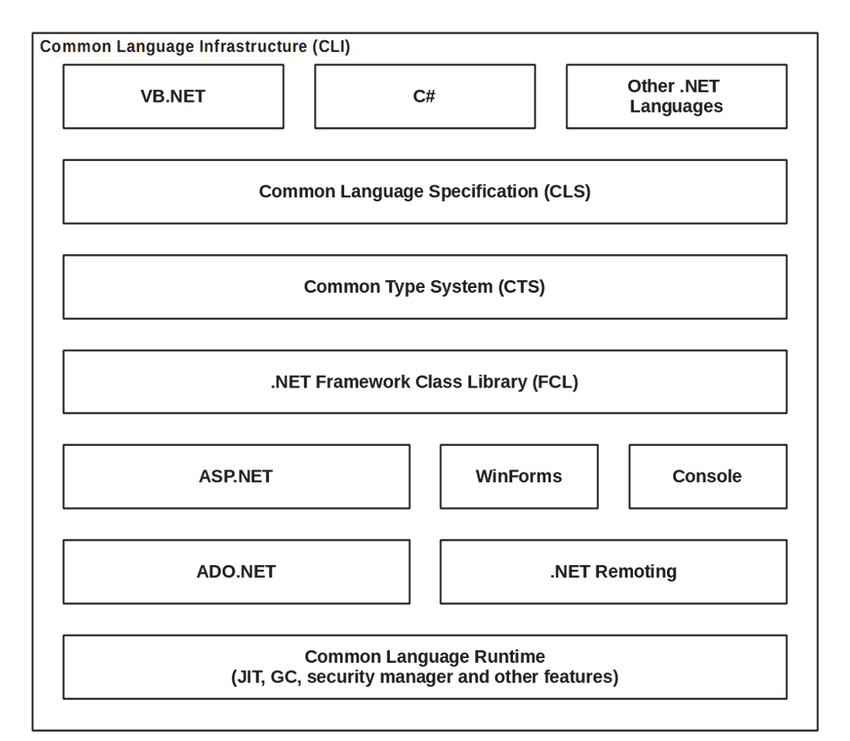
This framework provides various services like memory management, networking, security, memory management, and type-safety.

The .Net Framework supports more than 60 programming languages such as C#, F#, VB.NET, J#, VC++, JScript.NET, APL, COBOL, Perl, Oberon, ML, Pascal, Eiffel, Smalltalk, Python, Cobra, ADA, etc.

Following is the .NET framework Stack that shows the modules and components of the Framework.

The .NET Framework is composed of four main components:

1. Common Language Runtime (CLR)
2. Framework Class Library (FCL),
3. Core Languages (WinForms, ASP.NET, and ADO.NET), and
4. Other Modules (WCF, WPF, WF, Card Space, LINQ, Entity Framework, Parallel LINQ, Task Parallel Library, etc.)



The .NET Framework

## CLR (Common Language Runtime)

It is a program execution engine that loads and executes the program. It converts the program into native code. It acts as an interface between the framework and operating system. It does exception handling, memory management, and garbage collection. Moreover, it provides security, type-safety, interoperability, and portablility. A list of CLR components are given below:

## FCL (Framework Class Library)

It is a standard library that is a collection of thousands of classes and used to build an application. The BCL (Base Class Library) is the core of the FCL and provides basic functionalities.

## WinForms

Windows Forms is a smart client technology for the .NET Framework, a set of managed libraries that simplify common application tasks such as reading and writing to the file system.

## ASP.NET

ASP.NET is a web framework designed and developed by Microsoft. It is used to develop websites, web applications, and web services. It provides a fantastic integration of HTML, CSS, and JavaScript.

3]write note :authentication and authorization

a.

* *Authentication* is knowing the identity of the user. For example, Alice logs in with her username and password, and the server uses the password to authenticate Alice.
* *Authorization* is deciding whether a user is allowed to perform an action. For example, Alice has permission to get a resource but not create a resource.
* Windows Authentication
* Forms Authentication
* Passport Authentication
* Custom Authentication

The windows Authentication provider lets you authenticates users based on their windows accounts. This provider uses IIS to perform the authentication and then passes the authenticated identity to your code. This is the default provided for ASP.net.

The passport authentication provider uses Microsoft's passport service to authenticate users.

The forms authentication provider uses custom HTML forms to collect authentication information and lets you use your own logic to authenticate users. The user's credentials are stored in a cookie for use during the session.

**Windows authentication and IIS**

If you select windows authentication for your ASP.NET application, you also have to configure authentication within IIS. This is because IIS provides Windows authentication. IIS gives you a choice for four different authentication methods:

 Anonymous, basic digest, and windows integrated

 If you select anonymous authentication, IIS doesn't perform any authentication, Any one is allowed to access the ASP.NET application.

If you select basic authentication, users must provide a windows username and password to connect. How ever this information is sent over the network in clear text, which makes basic authentication very much insecure over the internet.

If you select digest authentication, users must still provide a windows user name and password to connect. However the password is hashed before it is sent across the network. Digest authentication requires that all users be running Internet Explorer 5 or later and that windows accounts to stored in active directory.

 Passport authentication

Passport authentication lets you to use Microsoft's passport service to authenticate users of your application. If your users have signed up with passport, and you configure the authentication mode of the application to the passport authentication, all authentication duties are offloaded to the passport servers.

Passport uses an encrypted cookie mechanism to indicate authenticated users. If users have already signed into passport when they visit your site, they'll be considered authenticated by ASP.NET. Otherwise they'll be redirected to the passport servers to log in. When they are successfully log in, they'll be redirected back to your site.

To use passport authentication you have to download the Passport Software Development Kit (SDK) and install it on your server. The SDK can be found at http://msdn.microdoft.com/library/default.asp?url=/downloads/list/websrvpass.aps. It includes full details of implementing passport authentication in your own applications.

Forms authentication

Forms authentication provides you with a way to handle authentication using your own custom logic with in an ASP.NET application. The following applies if you choose forms authentication.

1. When a user requests a page for the application, ASP.NET checks for the presence of a special session cookie. If the cookie is present, ASP.NET assumes the user is authenticated and processes the request.
2. If the cookie isn't present, ASP.NET redirects the user to a web form you provide
3. You can carry out whatever authentication, checks you like in your form. When the user is authenticated, you indicate this to ASP.NET by setting a property, which creates the special cookie to handle subsequent requests.

Configuring Authorization

After your application has authenticated users, you can proceed to authorize their access to resources. But there is a question to answer first: Just who is the user to whom your are grating access? It turns out that there are different answers to that question, depending on whether you implement impersonation. Impersonation is a technique that allows the ASP.NET process to act as the authenticated user, or as an arbitrary specified user.

ASP.NET impersonation is controlled by entries in the applications web.config file. The default setting is "no impersonation". You can explicitly specify that ASP.NET shouldn't use impersonation by including the following code in the file

*<identity impersonate="false"/>*

With this setting ASP.NET does not perform impersonation. It means that ASP.NET will runs with its own privileges. By default ASP.NET runs as an unprivileged account named ASPNET. You can change this by making a setting in the processModel section of the machine.config file. When you make this setting, it automatically applies to every site on the server. To user a high-privileged system account instead of a low-privileged, set the userName attribute of the processModel element to SYSTEM. Using this setting is a definite security risk, as it elevates the privileges of the ASP.NET process to a point where it can do bad things to the operating system.

When you disable impersonation, all the request will run in the context of the account running ASP.NET: either the ASPNET account or the system account. This is true when you are using anonymous access or authenticating users in some fashion. After the user has been authenticated, ASP.NET uses it own identity to request access to resources.

The second possible setting is to turn on impersonation.

*<identity impersonate="true"/>*

In this case, ASP.NET takes on the identity IIS passes to it. If you are allowing anonymous access in IIS, this means ASP.NET will impersonate the IUSR\_ComputerName account that IIS itself uses. If you aren't allowing anonymous access,ASP.NET will take on the credentials of the authenticated user and make requests for resources as if it were that user. Thus by turning impersonation on and using a non-anonymous method of authentication in IIS, you can let users log on and use their identities within your ASP.NET application.

Finally, you can specify a particular identity to use for all authenticated requests

1. <identity impersonate="true" username="DOMAIN\username" password="password"/>

With this setting, all the requests are made as the specified user (Assuming the password it correct in the configuration file). So, for example you could designate a user for a single application, and use that user's identity every time someone authenticates to the application. The drawback to this technique is that you must embed the user's password in the web.config file in plain text. Although ASP.NET won't allow anyone to download this file, this is still a security risk if anyone can get the file by other means.

**Authorization is** the process of allowing an authenticated user access to resources.

Authorization refers to the process that determines what a user is able to do. For example, an administrative user is allowed to create a document library, add documents, edit documents, and delete them. A non-administrative user working with the library is only authorized to read the documents.

Authorization is orthogonal and independent from authentication. However, authorization requires an authentication mechanism. Authentication is the process of ascertaining who a user is. Authentication may create one or more identities for the current user.

Types:7.45

4]note on master page

a. Master pages provide templates for other pages on your web site.

Master Pages

Master pages allow you to create a consistent look and behavior for all the pages (or group of pages) in your web application.

A master page provides a template for other pages, with shared layout and functionality. The master page defines placeholders for the content, which can be overridden by content pages. The output result is a combination of the master page and the content page.

The content pages contain the content you want to display.

When users request the content page, ASP.NET merges the pages to produce output that combines the layout of the master page with the content of the content page.

Master Page Example

<%@ Master %>  
  
<html>  
<body>  
<h1>Standard Header From Masterpage</h1>  
<asp:ContentPlaceHolder id="CPH1" runat="server">  
</asp:ContentPlaceHolder>  
</body>  
</html>

The master page above is a normal HTML page designed as a template for other pages.

The **@ Master** directive defines it as a master page.

The master page contains a placeholder tag **<asp:ContentPlaceHolder>** for individual content.

The **id="CPH1"** attribute identifies the placeholder, allowing many placeholders in the same master page.

This master page was saved with the name **"master1.master".**

5]repeater control with different support templates

a.   
The Repeater control is used to display a repeated list of items that are bound to the control. The Repeater control may be bound to a database table, an XML file, or another list of items.  
  
Repeater is a Data Bind Control. Data Bind Controls are container controls. Data Binding is the process of creating a link between the data source and the presentation UI to display the data. ASP .Net provides rich and wide variety of controls, which can be bound to the data.  
  
**Repeater has 5 inline template to format it:**  
1. <HeaderTemplate>  
2. <FooterTemplate>  
3. <ItemTemplate>  
4. <AlternatingItemTemplate>  
5. <SeperatorTemplate>  
6. <AlternatingItemTemplate>  
  
**HeaderTemplate:** This template is used for elements that you want to render once before your ItemTemplate section.  
  
**FooterTemplate:** - This template is used for elements that you want to render once after your ItemTemplate section.  
  
**ItemTemplate:**This template is used for elements that are rendered once per row of data. It is used to display records  
  
**AlternatingItemTemplate:** This template is used for elements that are rendered every second row of data. This allows you to alternate background colors. It works on even number of records only.  
  
**SeperatorTemplate:** It is used for elements to render between each row, such as line breaks.  
  
**Some point about Repeater Control**

* It is used to display backend result set. It is used to display multiple tuple.
* It is an unformatted control. The Repeater control is a basic templated data-bound list. It has no built-in layout or styles, so you must explicitly declare all layout, formatting, and style tags within the control's templates.
* The Repeater control is the only Web control that allows you to split markup tags across the templates. To create a table using templates, include the begin table tag (<table>) in the HeaderTemplate, a single table row tag (<tr>) in the ItemTemplate, and the end table tag (</table>) in the FooterTemplate.
* The Repeater control has no built-in selection capabilities or editing support. You can use the ItemCommand event to process control events that are raised from the templates to the control.

**Q-3[A]TWO MARK:**

1]connected architecture

a. in which connection must be opened to access the data retrieved from database is called as connected architecture. Connected architecture was built on the classes connection, command, datareader and transaction.

2]requirefieldvalidator control

a. This validator is used to make an input control required. It will throw an error if user leaves input control empty.

It is used to mandate form control required and restrict the user to provide data.

3]what are the use of web services in asp.net?

a. A Web Service is a software program that uses XML to exchange information with other software via common internet protocols. In a simple sense, Web Services are a way for interacting with objects over the Internet.

4]explsin absolute cache expiration.

a. Absolute Expiration allows us to specify the duration of the cache, starting from the time the cache is activated.

The cache is set to expire exactly two minutes after the user has retrieved the data.

5]dataadapter class

a. The DataAdapter works as a bridge between a DataSet and a data source to retrieve data. DataAdapter is a class that represents a set of SQL commands and a database connection. It can be used to fill the DataSet and update the data source.

6]client server architecture

a. Client-server architecture, architecture of a computer network in which many clients (remote processors) request and receive service from a centralized server (host computer). Client computers provide an interface to allow a computer user to request services of the server and to display the results the server returns.

**Q-3[B]THREE MARK:**

1]explain validationsummary control with property

a. This validator is used to display list of all validation errors in the web form.

It allows us to summarize the error messages at a single location.

We can set **DisplayMode** property to display error messages as a list, bullet list or single paragraph.

This control has following properties.

|  |  |
| --- | --- |
| **Property** | **Description** |
| AccessKey | It is used to set keyboard shortcut for the control. |
| BackColor | It is used to set background color of the control. |
| BorderColor | It is used to set border color of the control. |
| Font | It is used to set font for the control text. |
| ForeColor | It is used to set color of the control text. |
| Text | It is used to set text to be shown for the control. |
| ToolTip | It displays the text when mouse is over the control. |
| Visible | To set visibility of control on the form. |
| Height | It is used to set height of the control. |
| Width | It is used to set width of the control. |
| ShowMessageBox | It displays a message box on error in up-level browsers. |
| ShowSummary | It is used to show summary text on the form page. |
| ShowValidationErrors | It is used to set whether the validation summary should be shown or not. |

^complete che box.

2]disconnected architecture

a. The architecture in which data can be retrieved from the database even when the connection gets disconnected or closed is known as Disconnected Oriented Architecture.  
Following are the list of classes on which disconnected arch is built on:

**Connection :**Connection object is used to establish a connection to database and connectionit self will not transfer any data.  
  
**DataAdapter :**DataAdapter is used to transfer the data between database and dataset. It hascommands like select, insert, update and delete. Select command is used to retrieve data from database and insert, update and delete commands are used to send changes to the data in dataset to database. It needs a connection to transfer the data.

**CommandBuilder :**by default dataadapter contains only the select command and it doesn’tcontain insert, update and delete commands. To create insert, update and delete commands for the dataadapter, commandbuilder is used. It is used only to create these commands for the dataadapter and has no other purpose.

**DataSet :**Dataset is used to store the data retrieved from database by dataadapter and make itavailable for .net application.

**DataView :**DataView is a view of table available in DataSet. It is used to find a record, sort therecords and filter the records. By using dataview, you can also perform insert, update and delete as in case of a DataSet.

3]discuss theme concept

a. ASP.NET themes are a collection of properties that define the appearance of pages and controls in your Web site. A theme can include skin files, which define property settings for ASP.NET Web server controls, and can also include cascading style sheet files

**To create a page theme**

1. In Solution Explorer, right-click the name of the Web site for which you want to create a page theme, and then click **Add ASP.NET Folder**.
2. Click **Theme**.

If the App\_Themes folder does not already exist, Visual Web Developer creates it. Visual Web Developer then creates a new folder for the theme as a child folder of the App\_Themes folder.

1. Type a name for the new folder.

The name of this folder is also the name of the page theme. For example, if you create a folder named \App\_Themes\FirstTheme, the name of your theme is FirstTheme.

1. Add files to your new folder for control skins, style sheets, and images that make up the theme.

**To apply a theme to a Web site**

1. In the application's Web.config file, set the [<pages>](https://msdn.microsoft.com/en-us/library/950xf363(v=vs.100)) element to the name of the theme, either a global theme or a page theme, as shown in the following example:

Copy

<configuration>

<system.web>

<pages theme="ThemeName" />

</system.web>

</configuration>

**Note**

If an application theme has the same name as a global application theme, the page theme takes precedence.

1. To set a theme as a style sheet theme and be subordinated to local control settings), set the styleSheetTheme attribute instead:

Copy

<configuration>

<system.web>

<pages styleSheetTheme="Themename" />

</system.web>

</configuration>

4]note:data caching.

a. Caching is a technique of storing frequently used data/information in memory, so that, when the same data/information is needed next time, it could be directly retrieved from the memory instead of being generated by the application.

Caching is extremely important for performance boosting in ASP.NET, as the pages and controls are dynamically generated here. It is especially important for data related transactions, as these are expensive in terms of response time.

ASP.NET provides the following different types of caching:

* **Output Caching** : Output cache stores a copy of the finally rendered HTML pages or part of pages sent to the client. When the next client requests for this page, instead of regenerating the page, a cached copy of the page is sent, thus saving time.
* **Data Caching** : Data caching means caching data from a data source. As long as the cache is not expired, a request for the data will be fulfilled from the cache. When the cache is expired, fresh data is obtained by the data source and the cache is refilled.
* **Object Caching** : Object caching is caching the objects on a page, such as data-bound controls. The cached data is stored in server memory.
* **Class Caching** : Web pages or web services are compiled into a page class in the assembly, when run for the first time. Then the assembly is cached in the server. Next time when a request is made for the page or service, the cached assembly is referred to. When the source code is changed, the CLR recompiles the assembly.
* **Configuration Caching** : Application wide configuration information is stored in a configuration file. Configuration caching stores the configuration information in the server memory.

5]explain gridview control in detail

a. The GridView control displays the values of a data source in a table. Each column represents a field, while each row represents a record. The GridView control supports the following features:

* Binding to data source controls, such as SqlDataSource.
* Built-in sort capabilities.
* Built-in update and delete capabilities.
* Built-in paging capabilities.
* Built-in row selection capabilities.
* Programmatic access to the GridView object model to dynamically set properties, handle events, and so on.
* Multiple key fields.
* Multiple data fields for the hyperlink columns.
* Customizable appearance through themes and styles.

Creating a GridView

1. <asp:GridView ID="gridService" runat="server">
2. </asp:GridView>

6]explain web.config file

a. ASP.NET *Web.config* allows you to define or revise the configuration settings at the time of developing the application or at the time of deployment or even after deployment. The following are brief points that can be understood about the *Web.config* file:

* *Web.config* files are stored in XML format which makes us easier to work with.
* You can have any number of *Web.config* files for an application. Each *Web.config* applies settings to its own directory and all the child directories below it.
* All the *Web.config* files inherit the root *Web.config* file available at the following location *systemroot\Microsoft.NET\Framework\versionNumber\CONFIG\Web.config* location
* IIS is configured in such a way that it prevents the *Web.config* file access from the browser.
* The changes in *Web.config* don’t require the reboot of the web server.

Web.config Settings

Before we start working with configuration settings of ASP.NET, we see the hierarchy of the *Web.config* file.

Hide   Copy Code

<configuration>

<configSections>

<sectionGroup>

</sectionGroup>

</configSections>

<system.web>

</system.web>

<connectionStrings>

</connectionStrings>

<appSettings>

</appSettings>

…………………………………………………………………………………………………………

…………………………………………………………………………………………………………

…………………………………………………………………………………………………………

…………………………………………………………………………………………………………

…………………………………………………………………………………………………………

</configuration>

**Q-3[C]FIVE MARK:**

1]different types of files in asp.net

a. There are some file types managed by ASP.NET as given below:-

1.) .asax --> It refers to the Global.asax file containing code that drives from the HttpApplication class.It resides Application root directory.

2.) .ascx --> It refers to a web user control file. It resides Application root directory or a subdirectory.

3.) .aspx --> It refers to a ASP.NET Web forms.It resides Application root directory or a subdirectory.

4.) .asmx --> It refers to an xml web services file that contains classes and methods. It resides Application root directory or a subdirectory.

.5.) .axd --> It refers to a handler file that is used to website administration requests.It resides Application root directory.

6.) .cd --> It refers to a class diagram file. It resides Application root directory or a subdirectory.

7.) .compile --> It refers to a precompiled stub file that point to an assembly representing a compiled website file.Fir example .aspx,.ascx ,.master file are precompiled. It resides Bin subdirectory.

8.) .browser --> It refers to a browser definition file used to identify the features of client browsers. It redies App\_Browsers Subdirectory.

9.) .dll --> It refers to a compiled class library files (assembly file).It resides Bin subdirectory.

10.) .cs,.vb,.jsl --> It refers to a Source -code file that contains application logic.It resides App\_Code Subdirectory or same directory as web page.

11.) .csproj,.vbproj,.vjsproj --> It refers to a project file for a visual studio client application project.It resides Project Directory of visual studio.

12.) .master --> It refers to a master page that defines the layout of web page in a web application.It resides Application root or subdirectory.

13.) .mdf,.sdf --> It refers to a SQL database file. It resides App\_Data subdirectory.

14.) .mdb,ldb --> It refers to a Access database file. It resides App\_Data subdirectory.

15.) .msgx,svc --> It refers to an indigo messaging framework (MFx) service file.It resides Application root or a subdirectory.

16.) .soap --> It refers to a SOAP extension file .It resides Application root or a subdirectory.

17.) .sin --> It refers to a Solution file for visual web developer project. it resides Visual web Developer project directory.

18.) .skin --> It refers to a skin file.It is used for consistent formatting in web controls. It resides App\_Themes subdirectory.

19.) .sitemap --> It refers to a site-map file that containing the structure of the website. It resides Application root directory.

20) .rem --> It refers to a handler file which implements remoting concepts in web application.It resides Application root or a subdirectory.

2]explain tracing with its different level

a. ASP.NET tracing enables you to view diagnostic information about a single request for an ASP.NET page. ASP.NET tracing enables you to follow a page's execution path, display diagnostic information at run time, and debug your application. ASP.NET tracing can be integrated with system-level tracing to provide multiple levels of tracing output in distributed and multi-tier applications.

 There are two ways:

 (i) Page Level Tracing

(ii) Application Level Tracing

**(i) Page Level Tracing**

We can control whether tracing is enabled or disabled for individual pages. If tracing is enabled, when the page is requested, ASP.NET appends to the page a series of tables containing execution details about the page request. Tracing is disabled by default in an ASP.NET application.

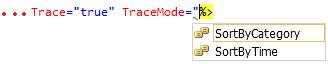
To enable Page Level Tracing follow the steps:

i) Include Trace="true" in <%@ Page Title="" Language="C#"...%> directive, for example:

<%@ Page Title="" Language="C#" MasterPageFile="~/MasterPage.master" AutoEventWireup="true" CodeFile="Default.aspx.cs"Inherits="chat\_Default" **Trace="true"**%>

 Look at the above code, I'll be using Trace=true at the end.

 ii) Optionally, we can use TraceMode attribute in above <%@ Page Title="" Language="C#"...%> directive, for example:



SortByCategory: Set TraceMode to SortByTime to sort trace messages in the order in which they are processed.

SortByTime: Set TraceMode to SortByCategory to sort trace messages by the categories.

 iii) Now press F5 to run the application, you will see the immediate trace record on an existing page that you have set Trace and TraceMode.

 You can see the traced record in the Trace Viewer as well; you will learn this in 'Application Level Tracing'.

**Application Level Tracing**

 Instead of enabling tracing for individual pages, you can enable it for your entire application. In that case, every page in your application displays trace information. Application tracing is useful when you are developing an application because you can easily enable it and disable it without editing individual pages. When your application is complete, you can turn off tracing for all pages at once.

 When you enable tracing for an application, ASP.NET collects trace information for each request to the application, up to the maximum number of requests you specify. The default number of requests is 10. You can view trace information with the trace viewer.

 By default, when the trace viewer reaches its request limit, the application stops storing trace requests. However, you can configure application-level tracing to always store the most recent tracing data, discarding the oldest data when the maximum number of requests is reached.

 To enable Application Level Tracing follow the steps:

 i) Delete your Page Level Tracking for better result.

 ii) Open the Web.config file and add the following information to it; if there is not a Web.config file available then add a new one in the root.

  :::::::::::::::::

  <system.web>

  <trace enabled="true" pageOutput="true" requestLimit="40" localOnly="false"/>

  </system.web>

</configuration>

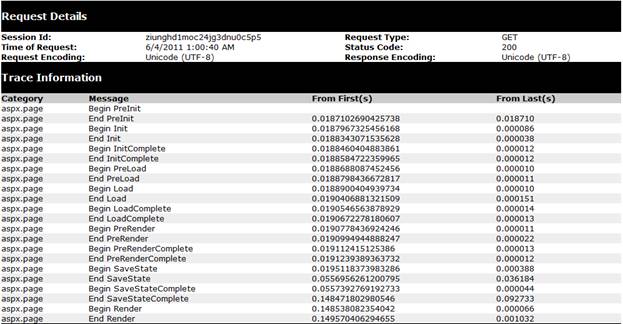
 iii) The above code has many attributes used, find the detailed information about them below.

 Enabled: Set it true to enable tracing for the application; otherwise, false. The default is false. You can override this setting for individual pages by setting the Trace attribute in the @ Page directive of a page to true or false.

 PageOutput: Set it true to display trace both in pages and in the trace viewer (trace.axd); otherwise, false. The default is false.

 RequestLimit: The number of trace requests to store on the server. The default is 10.

 LocalOnly: Set it true to make the trace viewer (trace.axd) available only on the host Web server; otherwise, false. The default is true.



3]fileupload control with example

a. The FileUpload control allows the user to browse for and select the file to be uploaded, providing a browse button and a text box for entering the filename.

Once, the user has entered the filename in the text box by typing the name or browsing, the SaveAs method of the FileUpload control can be called to save the file to the disk.

The basic syntax of FileUpload is:

<asp:FileUpload ID= "Uploader" runat = "server" />

The FileUpload class is derived from the WebControl class, and inherits all its members. Apart from those, the FileUpload class has the following read-only properties:

|  |  |
| --- | --- |
| **Properties** | **Description** |
| FileBytes | Returns an array of the bytes in a file to be uploaded. |
| FileContent | Returns the stream object pointing to the file to be uploaded. |
| FileName | Returns the name of the file to be uploaded. |
| HasFile | Specifies whether the control has a file to upload. |
| PostedFile | Returns a reference to the uploaded file. |

The posted file is encapsulated in an object of type HttpPostedFile, which could be accessed through the PostedFile property of the FileUpload class.

The HttpPostedFile class has the following frequently used properties:

|  |  |
| --- | --- |
| **Properties** | **Description** |
| ContentLength | Returns the size of the uploaded file in bytes. |
| ContentType | Returns the MIME type of the uploaded file. |
| FileName | Returns the full filename. |
| InputStream | Returns a stream object pointing to the uploaded file. |

## Example

The following example demonstrates the FileUpload control and its properties. The form has a FileUpload control along with a save button and a label control for displaying the file name, file type, and file length.

In the design view, the form looks as follows:



The content file code is as given:

<body>

<form id="form1" runat="server">

<div>

<h3> File Upload:</h3>

<br />

<asp:FileUpload ID="FileUpload1" runat="server" />

<br /><br />

<asp:Button ID="btnsave" runat="server" onclick="btnsave\_Click" Text="Save" style="width:85px" />

<br /><br />

<asp:Label ID="lblmessage" runat="server" />

</div>

</form>

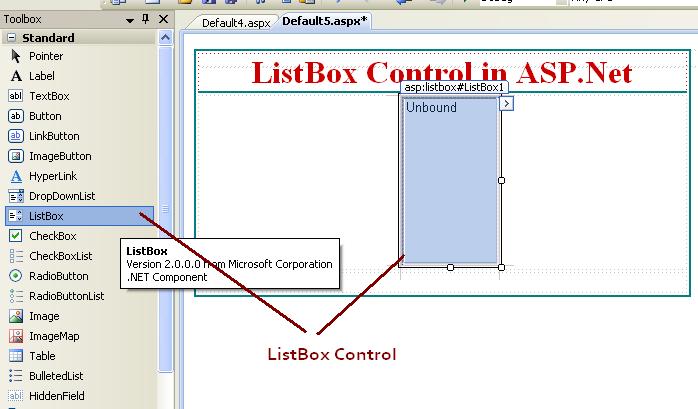
</body>

4]listbox control with property,metod&example

a. ListBox control is an asp.net web server control. ListBox control used to store the multiple items and allow user to select multiple item from listbox control. ListBox control is same as dropdownlist control.

In a ListBox control there is a **SelectionMode property** to change the mode of section from single to multiple. By default listbox control selection mode is single if you want to select multiple items from listbox, then just change the SelectionMode property to multiple.

ListBox Control Example in ASP.Net C#

[](https://meeraacademy.com/wp-content/uploads/2012/06/listbox.jpg)

**Some important Properties of ListBox Control.**

1. **Items.Count:** Returns the total number of items in the list box.
2. **Items.Clear:** Clears all the items from the list box.
3. **SelectedItem.Text:** Returns the text of the selected item.
4. **Items.Remove(“name”):** Removes the item with text “name”.
5. **Items.RemoveAt(int index):** Removes the item present at the given index.
6. **Items.Insert(int index, “text”):** Inserts the “text” at the given index.
7. **SelectedItem:** Returns the index of the selected item.
8. **SelectionMode:** Specifies the selection mode “single or multiple”.

5]programe