**What are page directives?**  
The first line of an ASP.NET page is the page directive; you will find it on all ASP.NET pages. These directives are instructions for the page. It begins with the @Page directive and continues with the various attributes available to this directive.  
   
It’s unreasonable to expect a candidate to know all of these attributes, but a few popular ones include the following.  
   
AutoEventWireup: Indicates whether page events are autowired.  
CodeBehind: The name of the compiled class associated with the page.  
Debug: Indicates whether the page is compiled in debug mode (includes debug symbols).  
EnableTheming: Indicates whether themes are used on the page.  
EnableViewState: Indicates whether view state is maintained across pages.  
ErrorPage: Specifies a target URL to be used when unhandled exceptions occur.  
Language: Indicates the language used when compiling inline code on the page.  
Trace: Signals whether tracing is enabled on the page.

Example:

<%@ Page Title="" Language="C#" MasterPageFile="~/MasterPage.master" AutoEventWireup="true" CodeFile="LeaveDetail.aspx.cs" Inherits="pages\_LeaveDetail" %>

**What is a master page?**  
   
A master page is a template for one or more Web Forms. The master page defines how the page will be laid out when presented to the user, with placeholders for content. The MasterPageFile Page Directive in a content page’s header is one way to assign a master page. The content pages rely solely on content and leave layout to the master page. ASP.NET merges the content with the master page layout when the content page is requested by a user.

**What is the code behind feature of ASP.NET?**  
   
The code behind feature divides ASP.NET page files into two files where one defines the user interface (.aspx), while the other contains all of the logic or code (.aspx.cs for C# and .aspx.vb for VB.NET). These two files are glued together with page directives like the following line, which ties the page to the specific code behind class.  
   
<![CDATA[<%@ Page language="c#" Codebehind="UICode.cs" Inherits="Library.UICode" %>  
What are ASHX files?  
   
ASP.NET Web handler files have an .ashx file extension. Web handlers work just like .aspx files except you don’t have to deal with the browser interface, thus no worrying about presentation. Web handlers are generally used to generate content dynamically like returning XML or an image. Web handlers use the IHttpHandler interface with the ProcessRequest() method invoked when the handler is requested. Web handlers are simpler than pages (fewer events and wiring), so they are ideal for performance-critical applications.  
   
**How does PostBack work?**  
   
PostBack is basically the ASP.NET submitting a form to it — it posts back to the current URL. The JavaScript \_\_doPostBack function is placed on the page (look at the source within the browser) to facilitate. PostBack uses ViewState to remember controls and data. The IsPostBack property of the ASP.NET page allows you to determine if the loading of the page is the result of a postback event; this is done in the Page\_Load event.  
   
**How can you pass values between ASP.NET pages?**  
   
There are numerous ways to accomplish this task; the option you choose depends on the environment. The oldest way to approach it is via the QueryString (i.e., passing values via URL); this is also one of the least secure ways because users can easily see the data and could possibly hack the site/page by changing parameter values. Next, you can use HTTP POST to pass values; these are available via a collection within the ASP.NET page. More specific to ASP.NET is the use of Session state, which makes information available to all pages within the ASP.NET application. Another approach is using public properties on the source page and accessing these properties on the target page. Also, you can use the PreviousPage property of the current page to access control information on the referring page. The last two require the source, and target pages are within the same ASP.NET application.  
   
**What are ASP.NET Server controls?**  
   
ASP.NET includes a number of built-in server controls that are the foundation of its Web programming model. They have various properties to control their behavior and appearance. These controls provide an event model where events are handled on the server (whereas HTML controls are handled in the client). Server controls have the ability to maintain state (via ViewState) across requests, and they can automatically detect the browser. With these controls, you will see the RunAt attribute (RunAt=”Server”) that signals its processing will be done on the server. A good example of these controls is the basic TextBox control (<asp:textbox runat="”Server”" xmlns:asp="#unknown">.  
   
**What is View State?**  
   
Basically, View State is how ASP.NET Web pages persists data across requests. It handles data that must be preserved between postbacks, and you can use it to store page-specific data. By default, View State is enabled on a page and its controls. This can be a problem as the amount of data and controls on a page increases resulting in more data for ASP.NET to maintain. This is accomplished via the hidden \_\_VIEWSTATE field on a form (look at the page source in a browser), so more data in this field means a slower load time and slower overall processing, as it has to be posted to the server each time. You can limit the size of the data in View State by disabling controls that do not need to be persisted via the EnableViewState property. View State can be encrypted to address security concerns.  
   
**What is the global.asax file?**  
   
  
The global.asax file is an optional piece of an ASP.NET application. It is located in the root of the application directory structure. It cannot be directly loaded or requested by users. It provides a place to define application- and session-wide events. You can define your own events, but it does contain default Application events like when the application starts Application\_Start and ends with Application\_End. The same is true for Session events (Session\_Start and Session\_End).  
   
**How can you loop through all controls on an ASP.NET Web form?**  
   
You can easily traverse all controls on a form via the Web Form’s Controls collection. The GetType method can be used on each control to determine its type and how to work with it. Now, it gets tricky because the form contains a tree of controls; that is, some controls are contained within others (think of a Table). You would have to recursively loop through the controls to make sure everything is processed.  
   
**What is a web.config file? Machine.config?**  
   
The web.config is the basic configuration file for ASP.NET applications. It utilizes an XML format. It is used to define application settings, connection strings, and much more. These files can appear in multiple directories, and they are applied in a top-down approach; that is, configuration files apply to their container directory as well as all directories below it, but the configuration files in lower directories can override those in parent directories. This provides a way to granularly apply settings. The machine.config file contains ASP.NET settings for all of the applications on the server — it is at the top of the configuration file hierarchy, thus web.configs can override it.

ASP.Net Interview Question and Answer for 2+ or 3+ years experience

**1.) What is asp.net life cycle ?**   
Life Cycle Events   
  
**PreInit**  
  
The properties like IsPostBack have been set at this time.  
  
This event will be used when we want to: 

1. Set master page dynamically.
2. Set theme dynamically.
3. Read or set profile property values.
4. This event is also preferred if want to create any dynamic controls.

**Init**

1. Raised after all the controls have been initialized with their default values and any skin settings have been applied.
2. Fired for individual controls first and then for page.

**LoadViewState**

1. Fires only if IsPostBack is true.
2. Values stored in HiddenField with id as \_ViewState decoded and stored into corresponding controls.

**LoadPostData**  
  
Some controls like:

1. Fires only if IsPostBack is true.
2. Some controls like Textbox are implemented from IPostBackDataHandler and this fires only for such controls.
3. In this event page processes postback data included in the request object pass it to the respective controls.

**PreLoad**

* Used only if want to inject logic before actual page load starts.

**Load**

* Used normally to perform tasks which are common to all requests, such as setting up a database query.

**Control events**

1. This event is fired when IsPostBack is true.
2. Use these events to handle specific control events, such as a Button control's Click event or a TextBox control's TextChanged event.

**PreRender**Raised after the page object has created all the controls that are required for rendering which includes child controls and composite controls.

1. Use the event to make final changes to the contents of the page or its controls before the values are stored into the viewstate and the rendering stage begins.
2. Mainly used when we want to inject custom JavaScript logic.

**SaveViewState**

* All the control values that support viewstate are encoded and stored into the viewstate.

**Render**Generates output (HTML) to be rendered at the client side.

* We can add custom HTML to the output if we want here.

**Unload**

1. Fired for individual controls first and then for page.

   2. Used to perform cleanup work like closing open files and database connections.    
  
  
**2.) How the request is handled by IIS ?**  
  
We give an URL to an aspx page in the browser address bar and press enter. What happens next? We get the response in terms of rendered HTML but how?

1. We are requesting something from the browser, which means indirectly we are requesting something from the Web Server, that means IIS. IIS, based on the file extension, decides which ISAPI extension can serve the request.  
     
   And in case of ASP.Net (.aspx) it will be aspnet\_isapi\_dll so the request is passed to it for processing.
2. When the first request comes to the website,  
     
   an application domain is created by the ApplicationManager class where exactly the website runs, and which creates an isolation between 2 web applications.  
   Within the application domain an instance of the HostingEnvironment class is created which provides access information about the application such as the name of the folder where the application is stored.
3. Next ASP.Net creates core objects like HttpContext, HttpRequest,HttpResponse.
4. Finally the application starts by creating an instance of the HttpApplication Class (which can be reused for multiple requests to maximize performance).

**3.) What is Difference between Session and Cookies  ?**  
  
The basic and main difference between cookie and session is that cookies are stored in the user's browser but sessions can't store in user's browser. This specifies which is best used for.   
  
• A cookie can keep all the information in the client's browser until deleted. If a person has a login and password, this can be set as a cookie in their browser so they do not have to re-login to your website every time they visit. You can store almost anything in a browser cookie.   
  
• Sessions are not reliant on the user allowing a cookie. They work like a token in the browser which allowing access and passing information while the user has opened his browser. The problem in sessions is when you close the browser the session will automatically lost. So, if you had a site requiring a login, this couldn't be saved as a session but it can be saved as a cookie, and the user has to re-login every time they visit.  
cookies are nothing but a small piece of information on the client machine. before we create a cookies we should check whether the cookies are allowed at the browser side. They are limited in a size 4k.(they are 2 types of cookies peristant cookie , and session cookies)   
  
Sessions cookies are stored in a server memory during the client browser session.When the browser is closed the session cookies are lost.   
  
  
**4.) Advantages and disadvantages of Session?**  
  
Following are the basic advantages and disadvantages of using session. I have describe in details with each type of session at later point of time.

**Advantages:**

* It helps maintain user state and data all over the application.
* It is easy to implement and we can store any kind of object.
* Stores client data separately.
* Session is secure and transparent from the user.

**Disadvantages:**

* Performance overhead in case of large volumes of data/user, because session data is stored in server memory.
* Overhead involved in serializing and de-serializing session data, because in the case of StateServerand SQLServer session modes, we need to serialize the objects before storing them.

Besides these, there are many advantages and disadvantages of session that are based on the session type. I have discussed all of them in the respective sections below.

**5). What is state management?**  
Ans: State management is the process by which you maintain state and page information over multiple requests for the same or different pages.   
  
**6). Http is stateless, What does this mean?**  
Ans: Stateless protocol is a communications protocol that treats each request as an independent transaction that is unrelated to any previous request so that the communication consists of independent pairs of requests and responses.  
  
**7). What is Session?**  
Ans: We know that Http is stateless, means when we open a webpage and fill some information and then move to next page then the data which we have entered will lost.  
It happed do to Http protocol stateless nature. So here session come into existence, Session provide us the way of storing data in server memory. So you can store your page data into server  
memory and retrieve it back during page postbacks.  
  
**8).  What are the Advantage and disadvantage of Session?**  
Ans: Advantages:  
Session provide us the way of maintain user state/data.  
It is very easy to implement.  
One big advantage of session is that we can store any kind of object in it. :eg, datatabe, dataset.. etc  
By using session we don't need to worry about data collesp, because it store every client data separately.  
Session is secure and transparent from the user.  
Disadvantages:  
Performance overhead in case of large volumes of data/user, because session data is stored in server memory.  
Overhead involved in serializing and de-serializing session data, because in the case of StateServer and SQLServer session modes, we need to serialize the objects before storing them.  
  
**9). What is Session ID in Asp.net?**  
Ans: Asp.Net use 120 bit identifier to track each session. This is secure enough and can't be reverse engineered. When client communicate with server, only session id is transmitted, between them. When client request for data, ASP.NET looks on to session ID and retrieves corresponding data.  
  
**10). By default where the sessions ID's are stored ?**  
Ans: By default, the unique identifier for a session is stored in a non-expiring session cookie in the browser. You can specify that session identifiers not be stored in a cookie by setting the cookieless attribute to true in the sessionState configuration element.  
We can also configure our application to store it in the url by specifying a "cookieless" session  
The ASP Session cookie has this format:-  
ASPSESSIONIDACSSDCCC=APHELKLDMNKNIOJONJACDHFN  
  
  
**11). Where does session stored if cookie is disabled on client’s machine?**  
Ans: If you want to disable the use of cookies in your ASP.NET application and still make use of session state, you can configure your application to store the session identifier in the URL instead of a cookie by setting the cookieless attribute of the sessionState configuration element to true, or to UseUri, in the Web.config file for your application.  
The following code example shows a Web.config file that configures session state to use cookieless session identifiers.

Code:

<configuration>  
  <system.web>  
    <sessionState   
      cookieless="true"  
      regenerateExpiredSessionId="true"  
      timeout="30" />  
  </system.web>  
</configuration>

**12). Can you describe all the property set in web.config under session state?**  
Ans:

Code:

<configuration>  
  <sessionstate   
      mode="inproc"  
      cookieless="false"   
      timeout="20"   
      sqlconnectionstring="data source=127.0.0.1;user id=<user id>;password=<password>"  
      server="127.0.0.1"   
      port="42424"   
  />  
</configuration>

Mode: The mode setting supports three options: inproc, sqlserver, and stateserver. As stated earlier, ASP.NET supports two modes: in process and out of process. There are also two options for out-of-process state management: memory based (stateserver), and SQL Server based (sqlserver). We'll discuss implementing these options shortly.  
Cookieless: The cookieless option for ASP.NET is configured with this simple Boolean setting.  
Timeout: This option controls the length of time a session is considered valid. The session timeout is a sliding value; on each request the timeout period is set to the current time plus the timeout value  
Sqlconnectionstring: The sqlconnectionstring identifies the database connection string that names the database used for mode sqlserver.  
Server: In the out-of-process mode stateserver, it names the server that is running the required Windows NT service: ASPState.  
Port: The port setting, which accompanies the server setting, identifies the port number that corresponds to the server setting for mode stateserver.  
  
**13). What are Session Events?**  
Ans: There are two types of session events available in ASP.NET:  
Session\_Start  
Session\_End  
You can handle both these events in the global.asax file of your web application. When a new session initiates, the session\_start event is raised, and the Session\_End event raised when a session is abandoned or expires.  
  
**14). How you can disable session?**  
Ans: If we set session Mode="off" in web.config, session will be disabled in the application. For this, we need to configure web.config the following way:

Code:

<configuration>  
  <sessionstate  Mode="off"/>  
</configuration>

**15). If I have more than one version of one assemblies, then how will I use old version (how/where to specify version number?) in my application?**  
  
Ans.The version number is stored in the following format: …. The assembly manifest can then contain a reference to which version number we want to use.  
  
**16). How do you create threading in.NET? What is the namespace for that?**  
Ans.  
System.Threading;  
  
//create new thread using the thread class’s constructor  
  
Thread myThread = new Thread(new ThreadStart (someFunction));  
  
**17). What do you mean by Serialize and MarshalByRef?**  
  
Serialization is the act of saving the state of an object so that it can be recreated (i.e deserialized) at a later date.  
The MarshalByRef class is part of the System.Runtime.Remoting namespace and enables us to access and use objects that reside in different application   
domains. It is the base class for objects that need to communicate across application domains. MarshalByRef objects are accessed directly within their own   
application domain by using a proxy to communicate. With MarshalByValue the a copy of the entire object is passed across the application domain  
  
 18). **What is the difference between Array and LinkedList?**  
An array is a collection of the same type. The size of the array is fixed in its declaration.  
A linked list is similar to an array but it doesn’t have a limited size.  
  
**19). What is Asynchronous call and how it can be implemented using delegates?**  
  
A synchronous call will wait for a method to complete before program flow is resumed. With an asynchronous call the program flow continues whilst the method executes.  
  
//create object  
SomeFunction objFunc = new SomeFunction();  
  
//create delegate  
SomeDelegate objDel = new SomeDelegate(objFunc.FunctionA);  
  
//invoke the method asynchronously (use interface IAsyncResult)  
IAsyncResult asynchCall = SomeDelegate.Invoke();  
  
**20). How to create events for a control? What is custom events? How to create it?**  
Ans.  
An event is a mechanism used in a class that can be used to provide a notification when something interesting happens. (typical evens in a windows application   
include: change text in textbox, double click or click a button, select an item in dropdown box).   
A custom event is an event created by the user that other developers can use. For example assuming that we have a CashTransaction class and we have a bank   
balance property in that class. We may want to set-up an event that provides a notification when the bank balance drops below a certain amount. In order to   
produce an event the process would be roughly as follows:   
Create the class for the event derived from EventArgs.  
Create a delegate with a return type of void.  
Create a class containing the method that will activate the event.  
Create a class with methods to handle the event.  
  
**21). If you want to write your own dot net language, what steps you will you take care?**  
  
We will need to ensure that the high level code is compiled to MSIL (Microsoft intermediate language) so that it can be interpreted by the CLR.  
  
**22). Describe the diffeerence between inline and code behind - which is best in a loosely coupled solution?**  
  
The term ‘code behind’ refers to application code that is not embedded within the ASPX page and is separated out into a separate file which is then referenced   
from the ASPX page. Inline code is the traditional ASP architectural model where business logic code was embedded within the ASP page. Separating the business   
logic code from the presentation layer offers several advantages:   
1) It allows graphic designers and web developers to work on the presentation layer whilst the application developers concentrate on the business logic.  
2) The codebehind file is compiled as a single dll increasing the efficiency of the application,  
3) The codebehind model offers a true OO development platform,  
4) It speeds up development time as it allows developers to fully maximise the features of the .NET framework such as Cahing, ViewState, Session, Smart Navigation etc.  
5) Code is much easier to maintain and susceptible for change.  
6) The compiler and VS.NET provides much better support for error checking, intellisense and debugging when using the code behind model.  
  
**23). How dot net compiled code will become platform independent?**  
  
The raison d’etre for .NET was to cater for multiples languages on a single windows platform whereas the aim of Java was to be a single language on multiple   
platforms. The only way that .NET can be platform independent is if there is a version of the .NET framework installed on the target machine.  
 **24). Without modifying source code if we compile again, will it be generated MSIL again?**  
Ans.No.  
 **25). How does you handle this COM components developed in other programming languages in.NET?**  
  
Ans.  
use TlbImp.exe to import the COM types into your .NET project. If no type library for the COM component then use System.Runtime.InteropServices  
use RegAsm.exe to call a .NET developed component in a COM application.  
  
**26). How CCW (Com Callable Wrapper) and RCW (Runtime Callable Wrappers) works?**  
  
CCW: When a COM application calls a NET object the CLR creates the CCW as a proxy since the COM application is unable to directly access the .NET object.  
RCW: When a .NET application calls a COM object the CLR creates the RCW as a proxy since the .NET application is unable to directly access the .COM object.  
  
**27). What are the new thee features of COM+ services, which are not there in COM (MTS)?**  
  
Role based security.  
Neutral apartment threading.  
New environment called context which defines the execution environment  
  
  **28). What are the differences between COM architecture and.NET architecture?**  
  
.Net architecture has superseded the old COM architecture providing a flexible rapid application development environment which can be used to create windows,   
web and console applications and web services. .NET provides a powerful development environment that can be used to create objects in any .NET compliant language.  
.NET addresses the previous problems of dll hell with COM by providing strongly named assemblies and side-by-side execution where two assemblies with the same name can run on the same box.  
 **29). Can we copy a COM dll to GAC folder?**  
  
Ans.  
No. It only stores .NET assemblies.  
 **30). What is Shared and Repeatable Inheritance?**  
  
Shared Inheritance-: Shared Inheritance-: ITt is multiple times using same class. The mechanism of deriving a new class from an existing class is called inheritance. Shared inheritance introduces a new opportunity of ambiguity and additional implementation complexity. Assume D inherits from B and C, both of which inherits from A. Here A in shared. Single copy made from both derived classes is called shared inheritance.  
  
  **31). Can you explain what inheritance is and an example of when you might use it?**  
  
Ans.  
Inheritance is a fundamental feature of any OO language. It allows us to inherit the members and attributes from a base class to a new derived class. This   
leads to increased code reusability and also makes applications easier to develop, maintain and extend as the new derived class can contain new features not   
available in the base class whilst at the same time preserving the attributes inherited from the base class.  
  
**32). How can you write a class to restrict that only one object of this class can be created (Singleton class)?**  
  
Ans.  
Use the singleton design pattern.

 public sealed class Singleton    
 {    
   static readonly Singleton Instance=new Singleton();    
      static Singleton()    
      {    
      }    
      Singleton()    
      {    
      }    
      public static Singleton Instance    
      {    
           get    
           {    
                return Instance;    
           }    
      }    
 }

**33). What are virtual destructures?**  
  
  
A constructor can not be virtual but a destructor may. Use virtual destructors when you want to implement polymorphic tearing down of an object.  
  
 **34). What is close method? How its different from Finalize and Dispose?**  
    
finalise is the process that allows the garbage collector to clean up any unmanaged resources before it is destroyed.  
The finalise method can not be called directly; it is automatically called by the CLR. In order to allow more control over the release of unmanaged resources  
the .NET framework provides a dispose method which unlike finalise can be called directly by code.  
Close method is same as dispose. It was added as a convenience.  
  
**35). What is Boxing and UnBoxing?**  
  
Ans.  
Boxing is the process of converting a value type to a reference type. More specifically it involves encapsulating a copy of the object and moving it from  
stack to heap. Unboxing is the reverse process.  
 **36). What is check/uncheck?**  
  
Ans.  
checked: used to enable overflow checking for arithmetic and conversion functions.  
unchecked: used to disable overflow checking for arithmetic and conversion functions   
  
 **37). What is the use of base keyword? Tell me a practical example for base keyword’s usage?**  
  
Ans.  
The base keyword is used to access members of the base class from within a derived class:  
\* Call a method on the base class that has been overridden by another method.  
\* Specify which base-class constructor should be called when creating instances of the derived class.  
  
A base class access is permitted only in a constructor, an instance method, or an instance property accessor.  
It is an error to use the base keyword from within a static method.  
Example:In this example, both the base class, Person, and the derived class, Employee, have a method named Getinfo. By using the base keyword,  
it is possible to call the Getinfo method on the base class, from within the derived class.  
// keywords\_base.cs  
// Accessing base class members

 using System;    
 public class Person    
 {    
 protected string ssn = "444-55-6666";    
 protected string name = "John L. Malgraine";    
 public virtual void GetInfo()    
 {    
 Console.WriteLine("Name: {0}", name);    
 Console.WriteLine("SSN: {0}", ssn);    
 }    
 }    
 class Employee: Person    
 {    
 public string id = "ABC567EFG";    
 public override void GetInfo()    
 {    
 // Calling the base class GetInfo method:    
 base.GetInfo();    
 Console.WriteLine("Employee ID: {0}", id);    
 }    
 }    
 class TestClass {    
 public static void Main()    
 {    
 Employee E = new Employee();    
 E.GetInfo();    
 }    
 }  

38). Difference Between Query String and Session

|  |  |
| --- | --- |
| **Querystring** | **Session** |
| Querystring is client side state management technique. | Session is server side state management technique. |
| Querystring data is page specific i.e. can be accessed in that page only. | Session data can be accessed throughout the session. |
| Querystring data is visible to user and can be seen in browser url. | Session data is not visible to user. |
| Data is not secured and can be altered hence insensitive data is stored in querystring. | Data is secured hence sensitive data such as user information is stored. |
| Querystring has constraint of Maxlength. | Session does not have such constraint. |

**39). Difference between Query string and Cookies**

**cookies** is a text file stored on client machine when we surf ant thing on internet by the server automatically we dont have to create it   
  
**query string** is used to transfer data from 1 page to anothe but this is not safe s it shows in url what data we r sending   
pen any site and see url after question mark tht is url  
  
**Cookies:**- Cookies are little pieces of information that a server stores on a browser. They are of two types   
1. Temporary cookie   
2. Persistent cookie   
  
Temporary cookie: - They are also known as session cookies. These are volatile in nature. When the browser is shutdown they are erased.   
  
Persistent cookie:- These may be called as permanent cookies. These are especially saved in files. It may remain for a month or year.

**Properties of cookies**  
Some properties of cookie   
Name: - represent the name of cookie.   
Name value: - represent a collection of key values of cookie   
Domain: - represent the domain associated with a specific cookie.   
Path: - the path associated with a cookie.   
Expires: - expired time of cookie.   
Hashkey: - identifies whether the cookie is a cookie dictionary.   
Secure: - specifies whether the cookie is to be sent in an encrypted connection or not  
**Query string** is the limited way to pass information to the web server while Transferring from one page to another page. This information is passed in url of the request. see below the code sample   
  
  
Code Sample   
  
//Retrieving values from query string   
String name;   
//Retrieving from query string   
name = Request.Param["umar"].ToString();   
  
But remember that many browsers impose a limit of 255 characters in query strings. You need to use HTTP-Get method to post a page to server otherwise query string values will not be available.

**40). How can we identify that the Page is Post Back?**

Page object has an "IsPostBack" property, which can be checked to know that is the page posted back.

**41). What is the lifespan for items stored in ViewState?**

PThe items stored in ViewState live until the lifetime of the current page expires including the postbacks to the same page

**42). What is AutoPostBack?**

If you want a control to postback automatically when an event is raised, you need to set the AutoPostBack property of the control to True

**43). Why do you use the App\_Code folder in ASP.NET?**

The App\_Code folder is automatically present in the project. It stores the files, such as classes, typed data set, text files, and reports. If this folder is not available in the application, you can add this folder. One of the important features of the App\_Code folder is that only one dll is created for the complete folder, irrespective of how many files it contains.

**44). In which event of page cycle is the ViewState available?**

After the Init() and before the Page\_Load().

**45). How long the items in ViewState exists?**

They exist for the life of the current page.

**46). Where the viewstate is stored after the page postback?**

ViewState is stored in a hidden field on the page at client side. ViewState is transported to the client and back to the server, and is not stored on the server or any other external source.

**47). What are the different Session state management options available in ASP.NET?**

1. In-Process
2. Out-of-Process.

**In-Process** stores the session in memory on the web server.

**Out-of-Process** Session state management stores data in an external server.  The external server may be either a SQL Server or a State Server.  All objects stored in session are required to be serializable for Out-of-Process state management.

**48). What is the difference between web config and machine config?**

Web config file is specific to a web application where as machine config is specific to a machine or server. There can be multiple web config files into an application where as we can have only one machine config file on a server.

**49). What are the different types of cookies in ASP.NET?**

**Session Cookie** – Resides on the client machine for a single session until the user does not log out.  
**Persistent Cookie** – Resides on a user’s machine for a period specified for its expiry, such as 10 days, one month, and never.

**50). What is the difference between ExecuteScalar and ExecuteNonQuery?**

ExecuteScalar returns output value where as ExecuteNonQuery does not return any value but the number of rows affected by the query. ExecuteScalar used for fetching a single value and ExecuteNonQuery used to execute Insert and Update statements.

51. difference between session and cookies

The basic and main difference between cookie and session is that cookies are stored in the user's browser but sessions can't store in user's browser. This specifies which is best used for.   
  
• A cookie can keep all the information in the client's browser until deleted. If a person has a login and password, this can be set as a cookie in their browser so they do not have to re-login to your website every time they visit. You can store almost anything in a browser cookie.   
  
• Sessions are not reliant on the user allowing a cookie. They work like a token in the browser which allowing access and passing information while the user has opened his browser. The problem in sessions is when you close the browser the session will automatically lost. So, if you had a site requiring a login, this couldn't be saved as a session but it can be saved as a cookie, and the user has to re-login every time they visit.