**ASP.NET Basics**

## What is ASP.NET?

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

* ASP.NET is a Microsoft Technology
* ASP stands for Active Server Pages
* ASP.NET is a program that runs inside IIS
* IIS (Internet Information Services) is Microsoft's Internet server
* IIS comes as a free component with Windows servers
* IIS is also a part of Windows 2000 and XP Professional

## What is an ASP.NET File?

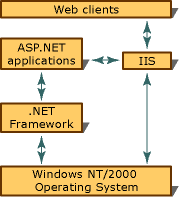
* An ASP.NET file is just the same as an HTML file
* An ASP.NET file can contain HTML, XML, and scripts
* Scripts in an ASP.NET file are executed on the server
* An ASP.NET file has the file extension ".aspx"

## How Does ASP.NET Work?

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

## Discuss the ASP.net architecture.

Below diagram provides an overview of the ASP.NET infrastructure and subsystem relationships, as they relate to the subject of security. The following illustration shows the relationships among the security systems in ASP.NET.



As the illustration shows, all Web clients communicate with ASP.NET applications through IIS. IIS deciphers and optionally authenticates the request. If **Allow Anonymous** is turned on, no authentication occurs. IIS also finds the requested resource (such as an ASP.NET application), and, if the client is authorized, returns the appropriate resource.

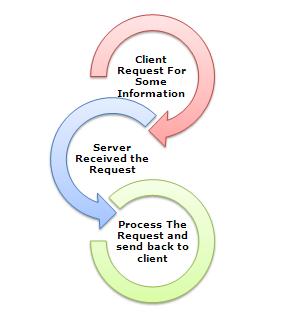
In addition to the built-in ASP.NET security features, an ASP.NET application can use the low-level security features of the .NET Framework.

**How IIS Process ASP.NET Request**

## What is Web Server?

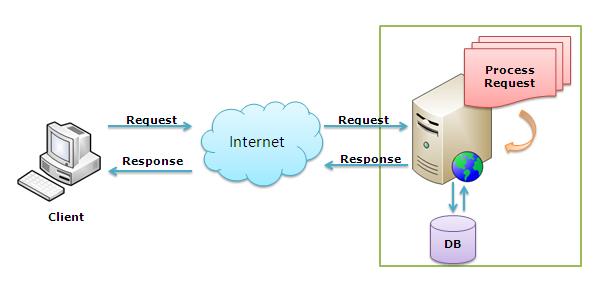
When we run our ASP.NET Web Application from visual studio IDE, VS Integrated ASP.NET Engine is responsible to execute all kind of asp.net requests and responses. The process name is **"WebDev.WebServer.Exe"** which actually takw care of all request and response of an web application which is running from Visual Studio IDE.

Now, the name *“Web Server”* come into picture when we want to host the application on a centralized location and wanted to access from many locations. Web server is responsible for handle all the requests that are coming from clients, process them and provide the responses.



## What is IIS?

*IIS (Internet Information Server)* is one of the most powerful web servers from Microsoft that is used to host your ASP.NET Web application. IIS has it's own ASP.NET Process Engine to handle the ASP.NET request. So, when a request comes from client to server, IIS takes that request and process it and send response back to clients.

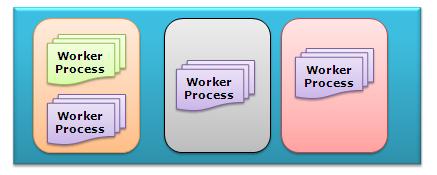


**Request Processing:**

Hope, till now it’s clear to you that what is Web server and IIS is and what is the use of them. Now let’s have a look how they do things internally. Before we move ahead, you have to know about two main concepts

1. *Worker Process*  
2. *Application Pool*

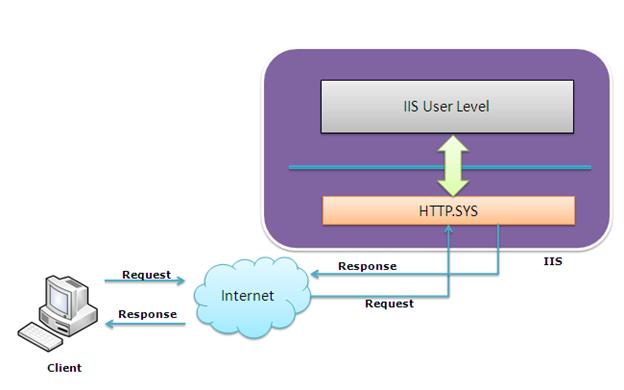
**Worker Process:** Worker Process (*w3wp.exe*) runs the ASP.Net application in IIS. This process is responsible to manage all the request and response that are coming from client system. All the ASP.Net functionality runs under the scope of worker process. When a request comes to the server from a client worker process is responsible to generate the request and response. In a single word we can say *worker process is the heart of ASP.NET Web Application which runs on IIS.*  
**Application Pool:** *Application pool is the container of worker process.* Application pools is used to separate sets of IIS worker processes that share the same configuration. Application pools enables a better *security, reliability, and availability* for any web application. The worker process serves as the process boundary that separates each application pool so that when one worker process or application is having an issue or recycles, other applications or worker processes are not affected. This makes sure that a particular web application doesn't not impact other web application as they they are configured into different application pools.



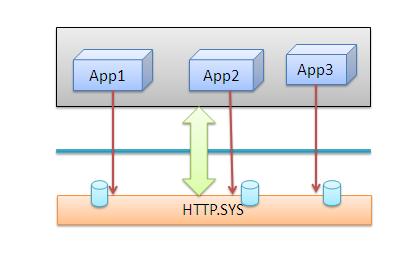
Application Pool with multiple worker process is called *“Web Garden”*.  
  
Now, I have covered all the basic stuff like Web server, Application Pool, Worker process. Now let’s have look how IIS process the request when a new request comes up from client.  
  
If we look into the IIS 6.0 Architecture, we can divide them into Two Layer

1. *Kernel Mode*  
2. *User Mode*

Now, ***Kernel mode*** is introduced with IIS 6.0, which contains the **HTTP.SYS**. So whenever a request comes from Client to Server, it will hit *HTTP.SYS* First.

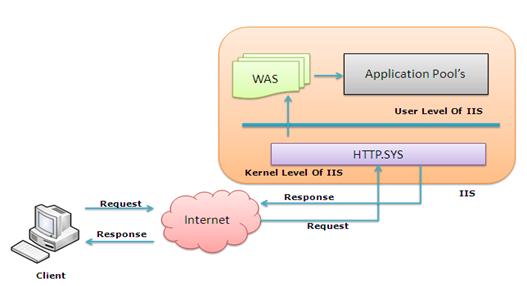


Now, HTTP.SYS is Responsible for pass the request to particular Application pool. *Now here is one question*, *How HTTP.SYS comes to know where to send the request?* This is not a random pickup. Whenever we creates a new Application Pool, the ID of the Application Pool is being generated and it’s registered with the HTTP.SYS. So whenever HTTP.SYS Received the request from any web application, it checks for the Application Pool and based on the application pool it send the request.

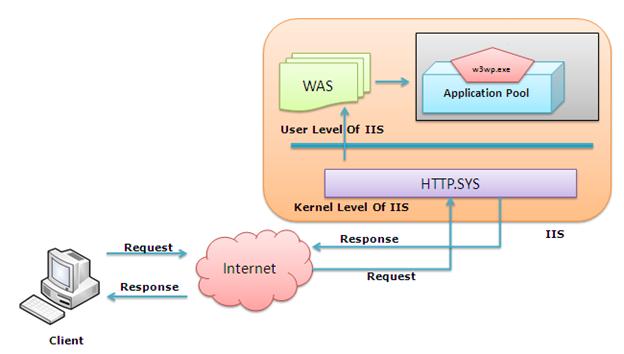


So, this was the first steps of IIS Request Processing.  
  
Till now, Client Requested for some information and request came to the Kernel level of IIS means at HTTP.SYS. HTTP.SYS has been identified the name of the application pool where to send. Now, let’s see how this request moves from HTTP.SYS to Application Pool.

In ***User Level*** of IIS, we have *Web Admin Services (WAS)* which takes the request from HTTP.SYS and pass it to the respective application pool.



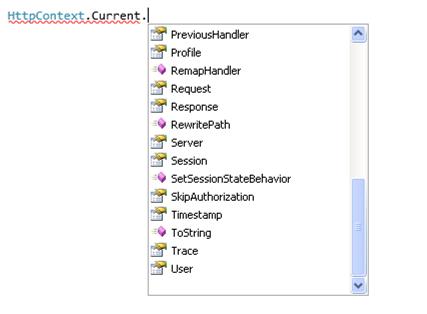
When Application pool receive the request, it simply pass the request to worker process (w3wp.exe) . The worker process *“w3wp.exe”* looks up the URL of the request in order to load the correct ISAPI extension. ISAPI extensions are the IIS way to handle requests for different resources. Once ASP.NET is installed, it installs its own ISAPI extension (aspnet\_isapi.dll) and adds the mapping into IIS.   
  
**Note :** Sometimes if we install IIS after installing asp.net, we need to register the extension with IIS using **aspnet\_regiis** command.



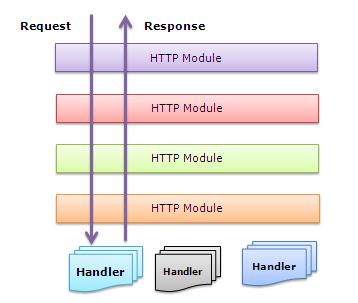
When Worker process loads the *aspnet\_isapi.dll*, it start an ***HTTPRuntime***, which is the entry point of an application. *HTTPRuntime* is a class which calls the *ProcessRequest* method to start Processing.



When this methods called, a new instance of ***HTTPContext*** is been created. Which is accessible using **HTTPContext.Current** Properties. This object still remains alive during life time of object request. Using HttpContext.Current we can access some other objects like *Request, Response, Session* etc.

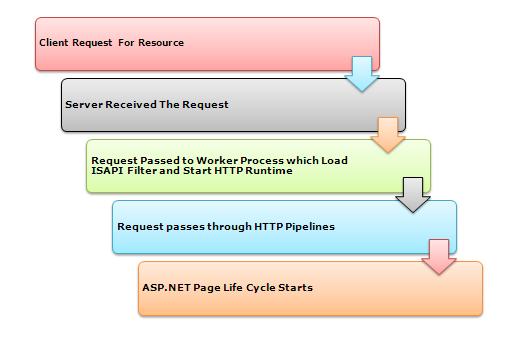


After that HttpRuntime load an *HttpApplication* object with the help of *HttpApplicationFactory* class.. Each and every request should pass through the corresponding HTTPModule to reach to HTTPHandler, this list of module are configured by the HTTPApplication.  
  
Now, the concept comes called “**HTTPPipeline**”. It is called a pipeline because it contains a set of HttpModules ( For Both Web.config and Machine.config level) that intercept the request on its way to the HttpHandler. HTTPModules are classes that have access to the incoming request. We can also create our own HTTPModule if we need to handle anything during upcoming request and response.



*HTTP Handlers are the endpoints in the HTTP pipeline.* All request that are passing through the HTTPModule should reached to HTTPHandler. Then HTTP Handler generates the output for the requested resource. So, when we are requesting for any aspx web pages, it returns the corresponding HTML output.

All the request now passes from httpModule to respective HTTPHandler then method and the ASP.NET Page life cycle starts. This ends the IIS Request processing and start the ASP.NET Page Lifecycle.



**Conclusion**

When client request for some information from a web server, request first reaches to HTTP.SYS of IIS. HTTP.SYS then send the request to respective Application Pool. Application Pool then forward the request to worker process to load the ISAPI Extension which will create an HTTPRuntime Object to Process the request via HTTPModule and HTTPHanlder. After that the ASP.NET Page LifeCycle events starts.

## What is the latest version of IIS released with Windows 7?

IIS 7.5

From where you can change the ASP.NET Version in IIS?

This can be change from Virtual Directory properties. First open Properties of Virtual Directory > GoTo ASP.NET Version Tab.   
  
There we can have change the ASP.NET Version.

## What are the different "Execution Permission" available for IIS for a virtual directory?

There are three Execution Permission available.   
1. None   
2. Scripts Only   
3. Scripts and Executable

## From where we can set the Session Time Out in IIS?

We can set the Session time out settings from the Virtual Directory for that site.   
  
Right Click on Virtual Directory > Properties > Click on "Configuration" Button   
Goto the "Option" Tab. There in **Enable Session State** Section you can configure the Session Timeout .

## How does IIS process an ASP.net request?

When client request for an aspx pages, request comes to kernel level off IIS means to **HTTP.SYS**. **HTTP.SYS** receives the request and based on the application pool name [ Which is already registred with the HTTP.SYS ] it send the request to worker process. *Windows Activation process* works as mediator of them. **w3wp.exe loads "aspnet\_isapi.dll"** files to start the **HTTPRuntime**. **HTTPRuntime** creates **HTTPApplication** objects and all request are passed through **HTTPModule** and finally reached to **HttpHandler**. This is the request pipeline. After end of Request pipeline ASP.NET Page lifecycle starts.   
  
For more Information:   
http://www.codeproject.com/KB/aspnet/aspnetrequestarchitecture.aspx

## What are the different ways that we can host site on IIS?

We can host site on IIS either creating Virtual Directory through IIS manager or Using Folder Web Sharing.   
Apart from that Visual studio provide some inbuilt features to host the site on IIS like using Publishing the web site , Using Copy web Tool or Creating Virtual directory during the creating the project by choosing Location as HTTP

## What are the main components of SVCHost.exe?

Main components for SVCHost.exe are **WWW Publishing Service** (W3SVC) and **Windows Activation Process** (WAP).   
  
W3SVC is the mediator of HTTP.SYS and Windows Activation Process. Windows Activation Process maintains the worker processes.

## Can we create one Application Pool from another Application Pool?

Yes. We can.   
While creating Application Application Pool from IIS, there should have two options available first one is for Default Setting and Another is for Existing Setting as template.   
We can select the second one and from the drop down listed below we can select any on the Application Pool as Template.

## Is there any alternative way to host site on IIS rather than opening IIS Manager?

Yes, We can directly host any site from the physical location of directory itself.   
  
Right Click on Physical Folder > Properties > Web Sharing   
  
There you need to select > "Share This Folder" Option Button. Then it will ask for alias name and other setting. Then Click on OK.   
  
To Validate : Run > Inetmgr > Check there should an virtual directory with the same "Alias" name that you have given.   
  
If there are already one Virtual directory exist it will showing you the error message while you providing the "Alias" name.

## How we can set the Idle Time out of a worker process?

We can set the Idle time out for an worker process from Application Pool Properties.   
  
In Performance Tab of Application pool, we can set the Idle Time out of the worker process. This means worker process will shut down after that given time period if it stay idle. And will again wake up again if a new request comes.

## How can we set the default page for any web application?

We can set the default page for a web site from the Virtual Directory Setting.   
How To:   
IIS Manager > Virtual Directory > Right Click > Properties > GoTo Document Tab.

## What is the default Identity of an Application Pool?

NetworkServices

## What is the Role of Windows Activation Process in IIS?

WAP is the Controller of Worker process under a Application Pool. Windows Activation Process which is managed by the worker process by starting, stopping and recycling the application pool. When to start, stop and Recycle should be defined on Application Pool Settings. Activation Process is also responsible for Health Monitor of Application Pool during runtime.   
  
FYI: Health monitoring setting can be easily found in Properties of Application Pool.

## What are the major innovation in IIS 7.0?

Below are the Major Innovation in IIS 7.0   
Components are designed as module and there are major change in administration settings.   
FYI : You can find out many of them, just go through Microsoft IIS web site.

## [What are the worker process for IIS 5.1 and IIS 6.0?](about:blank)

For IIS 5.1 > aspnet\_wp.exe   
For IIS 6.0 > w3wp.exe

## Name of the tool which is used for remote debugging of IIS

1. **msvsmon.exe**
2. iisremotedebugger.exe
3. remotehelper.exe
4. svcutil.exe

## What is name of default application pool in IIS?

1. dumyAppPool
2. **DefaultAppPool**
3. IISAppPool

## Which application pool having maximum privilege on the server?

1. **LocalSystem**
2. LocalServices
3. NetworkServices

LocalSystem is a built-in account that has administrative privileges on the server. It can access both local and remote resources.

## Which is not an Identity of Application Pool?

1. LocalSystem
2. LocalServices
3. **NetworkSystem**
4. NetworkServices

## What are the different types of application pool available in IIS 7.0?

IIS 7.0 having two types of application pool.   
  
1. DefaultAppPool (Integrated)   
2. ClassicAppPool

## What is ISAPI Filter?

Where is the default location for IIS Log files?

Its C:\WINDOWS\system32\LogFiles\W3SVC1

## Can we have multiple web sites on IIS?

Yes. IIS Can have multiple web sites and Each and every web sites can have multiple virtual Directory.   
  
Note: Here a web site means the Root Node.

## Why do we need to IIS Remote Debugging?

There are following reasons where we can use remote debugging   
1. Your development server does not have IIS installed.   
2. Development server and Build/Released/Hosting Server is different   
3. Multiple user want to debug simultaneously.

## What is the use of Enable Pinging Properties for Application Pool?

IIS should periodically monitor the health of a worker process [ Idle or not , Time for recycle or not, All Worker process are running properly or not ] .   
Pining means, Activation Process monitor Worker process performance, health, idle time etc.   
By default it sets to 30s .

## What is the use of aspnet\_regiis -i command?

This is used automatically register the .NET Framework with your IIS.   
  
For more information:   
http://msdn.microsoft.com/en-us/library/k6h9cz8h%28VS.80%29.aspx

## Does IIS allows multiple user to Remote debug simultaneously?

Yes. This is one of the great features of msvsmon.exe. Each instance of the remote debugger has a unique server name. We can give an instance of the remote debugger any server name. Now, multiple user can able to access the server instance.

## If there are multiple worker process running on IIS, then how can you attach a particular worker process for application?

Well, If there are multiple worker process running in IIS, it means I have to know the name of my application pool. Then I can run cscript iisapi.vbs script to find out the process ID and Application Pool name . Based on the process Id for particular application I have to attache the process from Visual studio IDE.

## How can we get the list of worker process running in IIS along with the Application pool name?

By running **iisapp.vbs** script from command Prompt.   
  
Below are the steps :   
1. *Start* > *Run* > *Cmd*   
2. Go To Windows > System32   
3. Run cscript iisapp.vbs

## What are the different authentication mode available for IIS Remote Debugging?

For IIS Remote Debugging msvsmon supported two authentication mode   
  
1. Windows Authentication   
2. No-Authentication

## Which Tool is used for Remote IIS Debugging?

Tools is: msvsmon.exe   
  
**This is located at :** Install path\Microsoft Visual Studio 8\Common7\IDE\Remote Debugger\x86

## How we can save an Application Pool Settings?

Application Pool Settings can be save as "XML" Format.   
  
Right Click on Application Pool > All Task > Save Configuration to a File .   
  
This will save all the settings of Application Pool as an XML file. We can make it password protected also.

## Which version of IIS is available in Windows Server 2008?

IIS 7.0 .   
  
Even Vista Home Premium and Ultimate edition is also having IIS 7.0

## Does One Web Application can have multiple Application Pool?

No. Every Web Application should have one Application Pool. Bydefault it is "DefaultAppPool ".

## What is the folder location for Virtual Directory?

<Drive>:\inetpub\wwwroot

## What is Application Pool in IIS?

Before Giving the Definition : you can say like this, Concept of Application pool has from IIS 6.0 .   
Application pools are used to separate sets of IIS worker processes that share the same configuration and application boundaries. Application pools used to **isolate** our web application for better security, reliability, and availability and performance and keep running with out impacting each other . The worker process serves as the process boundary that separates each application pool so that when one worker process or application is having an issue or recycles, other applications or worker processes are not affected.   
One Application Pool can have multiple worker process Also.   
  
Main Point to Remember:   
1. Isolation of Different Web Application   
2. Individual worker process for different web application   
3. More reliably web application   
4. Better Performance

## What is the Role of IIS?

Visual studio having its own ASP.NET Engine which is capable enough to run Asp.net web application from visual studio. So we just click on Run button to start the application.   
Now this is the scenarios of local environment. But If we want to host it on server from where all user can access the sites then IIS comes into the picture.   
  
IIS provides a redesigned WWW architecture that can help you achieve better performance, reliability, scalability, and security for our Web sites. IIS can support following Protocol *HTTP/HTTPS, FTP, FTPS, SMTP* Etc. We need to host the site on IIS, when request comes from client it first hits the IIS Server, then the server passed it to ASP.NET worker process to execute. Then the response also passes to client via IIS itself.   
Note only Hosting of Site we can create our FTP Server, SMTP Server using IIS itself.   
There are different version of IIS available like 5.1, 6.0, 7.0 etc.

## Name of default Identity of IIS6.0

Default Identity of IIS 6.0 is **NetworkServices**.   
Which is having very minimum rights on your system. The user can only have the read access of the site.

## What are the different types of Identity available in IIS 6.0?

IIS having three different Identity.   
1. Local System   
2. Local Services   
3. NetworkServices

## What is the Name of Default Application Pool in IIS?

Though we can create new application pool IIS with different settings, but IIS having its own default application pool named : **DefaultAppPool**

## What are the different version on IIS that you have worked on?

Before answering this question you need to know what are the different IIS version is available in different OS. Below is the list of IIS version with different Operating system.   
Windows Server 2008 - Windows Vista - Home Premium/ Ultimate - IIS 7.0   
Windows Server 2003 - IIS 6.0   
Windows XP Professional - IIS 5.1   
Now based on your working experience you can say that you have worked on IIS 5.1 and 6.0 or only IIS 7. Etc.   
Now, the next question that can asked after answering this question is “*what is the difference between them ?* ” – Well I will come with this later.

## How we can open IIS Configuration manager ?

Just simply **Run >inetmgr**   
Or we can open it from control panel > Administrative tools.

## What are the permission settings are available for Virtual Directory?

Below are the list of permission that can be set during virtaul directory creation   
1. Read   
2. Run Scripts   
3. Execute:   
4. Write:   
5. Browse

## How we can create a Virtual Directory on IIS?

Open IIS Configuration Manager   
First of all Right Click on Default web sites > New > Virtual Directory .   
Browse the Physical Path. Set the properites. Click on OK

## How we can debug a web application which is hosted on IIS?

We can easily debug any web application that are hosted on IIS by using Attaching of Worker Process.   
From Visual Studio IDE > Tools > Attach To Process   
Select the particular Process, then start debugging.   
  
For more information Read this article :   
http://www.codeproject.com/KB/aspnet/ProcessAttache.aspx

**How we can create an web garden?**

For creating web graden we need to go to Application Pool, then Right Click on Application Pool > Properties > Goto Performance Tab   
  
In Web Garden Section, increase the number of worker process. By default it is 1.

**Where session data stores in case of "In-Proc" Session mode?**

Session data store inside process memory of worker process [w3wp.exe].

**What is the default authentication settings for IIS?**

Anonymous authentication is the default authentication mode for any site that is hosted on IIS, and it runs under the **"IUSR\_[ServerName]"** account.

**What are the different security settings available in IIS?**

Below are the commonly used IIS Security settings   
  
1 Anonymous   
2 Integrated Windows Authentication   
3. Basic Authentication   
4. Digest Authentication   
5. Passport Authentication   
  
For Set security permission you need to go to Virtual Directory > Right Click > Properties > Directory Security   
Click on Edit Button .

**What is the Role of Http.Sys in IIS?**

HTTP.SYS is the kernel level components of IIS. All client request comes from client hit the HTTP.Sys of Kernel level. HTTP.SYS then makes a queue for each and every request for each and individual application pool based on the request.   
Whenever we create any application pool IIS automatically registers the pool with HTTP.SYS to identify the particular during request processing.

**What are the main layers of IIS Architecture?**

IIS having mainly two layers: Kernel Mode and User Mode   
  
Below are the subsection of both of them.   
1. Kernel Mode   
o HTTP.SYS   
2. User Mode   
o Web Admin Service   
o Virtual Directory   
o Application Pool

**What is Recycling of Application Pool?**

Recycling Application pool means recycle the Worker process (**w3wp.exe**) and the memory used for the web application.   
*There are two types of recycling related with Application pool*   
  
1. Recycling Worker Process - Predefined Settings   
2. Recycling Worker Process - Based on Memory

**What are different IIS isolation levels?**

*Isolating* applications means configuring them to run in a process (memory space) that is separate from the Web server and other applications.IIS has three level of isolation:-

1.LOW (IIS process):- In this main IIS process and ASP.NET application run in same process.  
So if any one crashes the other is also affected.  
Example:  
I have hosted yahoo , hotmail .amazon and google on a single PC. So all application and  
the IIS process runs on the same process.In case any website crashes it affects every one.  
2.Medium (Pooled):- In Medium pooled scenario the IIS and web application run in different  
process. So in this case there are two processes process1 and process2. In process1 the IIS  
process is running and in process2 we have all Web application running.  
3.High (Isolated):-In high isolated scenario every process is running is there own process. In  
below figure there are five processes and every one handling individual application. This  
consumes heavy memory but has highest reliability.

**How would you get ASP.NET running in Apache web servers - why would you even do this?**

The mod\_mono Apache module is used to run ASP.NET applications within the Apache (http://httpd.apache.org) web server. Mod\_mono is available from (http://www.mono-project.com/Downloads).XSP is a standalone web server written in C# that can be used to run your ASP.NET applications. XSP works under both the Mono and Microsoft runtimes and the code is available from(<http://www.mono-project.com/Downloads>).

**Which dll handles the request of .aspx page?**

When the Internet Information Service process (inetinfo.exe) receives an HTTP request, it uses the filename extension of the requested resource to determine which Internet Server Application Programming Interface (ISAPI) program to run to process the request. When the request is for an ASP.NET page (.aspx file), IIS passes the request to the ISAPI DLL capable of handling the request for ASP.NET pages, which is ***aspnet\_isapi.dll.***

**Application life cycle and events processing**

A web application starts when a browser requests a page of the application first time. The request is received by the IIS which then starts ASP.NET worker process (aspnet\_wp.exe). The worker process then allocates a process space to the assembly and loads it. An application\_start event occurs followed by Session\_start. The request is then processed by the ASP.NET engine and sends back response in the form of HTML. The user receives the response in the form of page.    
  
The page can be submitted to the server for further processing. The page submitting triggers postback event that causes the browser to send the page data, also called as view state to the server. When server receives view state, it creates new instance of the web form. The data is then restored from the view state to the control of the web form in Page\_Init event.

The data in the control is then available in the Page\_load event of the web form. The cached event is then handled and finally the event that caused the postback is processed. The web form is then destroyed. When the user stops using the application, Session\_end event occurs and session ends. The default session time is 20 minutes. The application ends when no user accessing the application and this triggers Application\_End event. Finally all the resources of the application are reclaimed by the Garbage collector.

#### Explain the ASP.NET page lifecycle stages.

In general terms, the page goes through the stages outlined in the following table. In addition to the page life-cycle stages, there are application stages that occur before and after a request but are not specific to a page. For more information, see [Introduction to the ASP.NET Application Life Cycle](http://go.microsoft.com/fwlink/?LinkId=133108) and [ASP.NET Application Life Cycle Overview for IIS 7.0](http://msdn.microsoft.com/en-us/library/bb470252.aspx).

Some parts of the life cycle occur only when a page is processed as a postback. For postbacks, the page life cycle is the same during a partial-page postback (as when you use an [UpdatePanel](http://msdn.microsoft.com/en-us/library/system.web.ui.updatepanel.aspx) control) as it is during a full-page postback.

| **Stage** | **Description** |
| --- | --- |
| Page request | The page request occurs before the page life cycle begins. When the page is requested by a user, ASP.NET determines whether the page needs to be parsed and compiled (therefore beginning the life of a page), or whether a cached version of the page can be sent in response without running the page. |
| Start | In the start stage, page properties such as [Request](http://msdn.microsoft.com/en-us/library/system.web.ui.page.request.aspx) and [Response](http://msdn.microsoft.com/en-us/library/system.web.ui.page.response.aspx) are set. At this stage, the page also determines whether the request is a postback or a new request and sets the [IsPostBack](http://msdn.microsoft.com/en-us/library/system.web.ui.page.ispostback.aspx) property. The page also sets the [UICulture](http://msdn.microsoft.com/en-us/library/system.web.ui.page.uiculture.aspx) property. |
| Initialization | During page initialization, controls on the page are available and each control's [UniqueID](http://msdn.microsoft.com/en-us/library/system.web.ui.control.uniqueid.aspx) property is set. A master page and themes are also applied to the page if applicable. If the current request is a postback, the postback data has not yet been loaded and control property values have not been restored to the values from view state. |
| Load | During load, if the current request is a postback, control properties are loaded with information recovered from view state and control state. |
| Postback event handling | If the request is a postback, control event handlers are called. After that, the [Validate](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.basevalidator.validate.aspx) method of all validator controls is called, which sets the [IsValid](http://msdn.microsoft.com/en-us/library/system.web.ui.ivalidator.isvalid.aspx) property of individual validator controls and of the page. |
| Rendering | Before rendering, view state is saved for the page and all controls. During the rendering stage, the page calls the [Render](http://msdn.microsoft.com/en-us/library/system.web.ui.control.render.aspx) method for each control, providing a text writer that writes its output to the [OutputStream](http://msdn.microsoft.com/en-us/library/system.web.httpresponse.outputstream.aspx) object of the page's [Response](http://msdn.microsoft.com/en-us/library/system.web.ui.page.response.aspx) property. |
| Unload | The [Unload](http://msdn.microsoft.com/en-us/library/system.web.ui.control.unload.aspx) event is raised after the page has been fully rendered, sent to the client, and is ready to be discarded. At this point, page properties such as [Response](http://msdn.microsoft.com/en-us/library/system.web.ui.page.response.aspx) and [Request](http://msdn.microsoft.com/en-us/library/system.web.ui.page.request.aspx) are unloaded and cleanup is performed. |

#### Explain the ASP.NET page lifecycle events.

Within each stage of the life cycle of a page, the page raises events that you can handle to run your own code. For control events, you bind the event handler to the event, either declaratively using attributes such as onclick, or in code.

Pages also support automatic event wire-up, meaning that ASP.NET looks for methods with particular names and automatically runs those methods when certain events are raised. If the AutoEventWireup attribute of the [@ Page](http://msdn.microsoft.com/en-us/library/ydy4x04a.aspx) directive is set to true, page events are automatically bound to methods that use the naming convention of Page\_*event*, such as Page\_Load and Page\_Init. For more information on automatic event wire-up, see [ASP.NET Web Server Control Event Model](http://msdn.microsoft.com/en-us/library/y3bwdsh3.aspx).

The following table lists the page life-cycle events that you will use most frequently. There are more events than those listed; however, they are not used for most page-processing scenarios. Instead, they are primarily used by server controls on the ASP.NET Web page to initialize and render themselves. If you want to write custom ASP.NET server controls, you need to understand more about these events. For information about creating custom controls, see [Developing Custom ASP.NET Server Controls](http://msdn.microsoft.com/en-us/library/zt27tfhy.aspx).

| **Page Event** | **Typical Use** |
| --- | --- |
| [PreInit](http://msdn.microsoft.com/en-us/library/system.web.ui.page.preinit.aspx) | Raised after the start stage is complete and before the initialization stage begins.  Use this event for the following:   * Check the [IsPostBack](http://msdn.microsoft.com/en-us/library/system.web.ui.page.ispostback.aspx) property to determine whether this is the first time the page is being processed. The [IsCallback](http://msdn.microsoft.com/en-us/library/system.web.ui.page.iscallback.aspx) and [IsCrossPagePostBack](http://msdn.microsoft.com/en-us/library/system.web.ui.page.iscrosspagepostback.aspx) properties have also been set at this time. * Create or re-create dynamic controls. * Set a master page dynamically. * Set the [Theme](http://msdn.microsoft.com/en-us/library/system.web.ui.page.theme.aspx) property dynamically. * Read or set profile property values.   **Note**  If the request is a postback, the values of the controls have not yet been restored from view state. If you set a control property at this stage, its value might be overwritten in the next event. |
| [Init](http://msdn.microsoft.com/en-us/library/system.web.ui.control.init.aspx) | Raised after all controls have been initialized and any skin settings have been applied. The [Init](http://msdn.microsoft.com/en-us/library/system.web.ui.control.init.aspx) event of individual controls occurs before the [Init](http://msdn.microsoft.com/en-us/library/system.web.ui.control.init.aspx) event of the page.  Use this event to read or initialize control properties. |
| [InitComplete](http://msdn.microsoft.com/en-us/library/system.web.ui.page.initcomplete.aspx) | Raised at the end of the page's initialization stage. Only one operation takes place between the [Init](http://msdn.microsoft.com/en-us/library/system.web.ui.control.init.aspx) and [InitComplete](http://msdn.microsoft.com/en-us/library/system.web.ui.page.initcomplete.aspx) events: tracking of view state changes is turned on. View state tracking enables controls to persist any values that are programmatically added to the [ViewState](http://msdn.microsoft.com/en-us/library/system.web.ui.control.viewstate.aspx) collection. Until view state tracking is turned on, any values added to view state are lost across postbacks. Controls typically turn on view state tracking immediately after they raise their [Init](http://msdn.microsoft.com/en-us/library/system.web.ui.control.init.aspx) event.  Use this event to make changes to view state that you want to make sure are persisted after the next postback. |
| [PreLoad](http://msdn.microsoft.com/en-us/library/system.web.ui.page.preload.aspx) | Raised after the page loads view state for itself and all controls, and after it processes postback data that is included with the [Request](http://msdn.microsoft.com/en-us/library/system.web.ui.page.request.aspx) instance. |
| [Load](http://msdn.microsoft.com/en-us/library/system.web.ui.control.load.aspx) | The [Page](http://msdn.microsoft.com/en-us/library/system.web.ui.page.aspx) object calls the [OnLoad](http://msdn.microsoft.com/en-us/library/system.web.ui.control.onload.aspx) method on the [Page](http://msdn.microsoft.com/en-us/library/system.web.ui.page.aspx) object, and then recursively does the same for each child control until the page and all controls are loaded. The [Load](http://msdn.microsoft.com/en-us/library/system.web.ui.control.load.aspx) event of individual controls occurs after the [Load](http://msdn.microsoft.com/en-us/library/system.web.ui.control.load.aspx) event of the page.  Use the [OnLoad](http://msdn.microsoft.com/en-us/library/system.web.ui.control.onload.aspx) event method to set properties in controls and to establish database connections. |
| Control events | Use these events to handle specific control events, such as a [Button](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.button.aspx) control's [Click](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.button.click.aspx) event or a [TextBox](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.textbox.aspx) control's [TextChanged](http://msdn.microsoft.com/en-us/library/system.web.ui.mobilecontrols.textbox.textchanged.aspx) event.  **Note**  In a postback request, if the page contains validator controls, check the [IsValid](http://msdn.microsoft.com/en-us/library/system.web.ui.page.isvalid.aspx) property of the [Page](http://msdn.microsoft.com/en-us/library/system.web.ui.page.aspx) and of individual validation controls before performing any processing. |
| [LoadComplete](http://msdn.microsoft.com/en-us/library/system.web.ui.page.loadcomplete.aspx) | Raised at the end of the event-handling stage.  Use this event for tasks that require that all other controls on the page be loaded. |
| [PreRender](http://msdn.microsoft.com/en-us/library/system.web.ui.control.prerender.aspx) | Raised after the [Page](http://msdn.microsoft.com/en-us/library/system.web.ui.page.aspx) object has created all controls that are required in order to render the page, including child controls of composite controls. (To do this, the [Page](http://msdn.microsoft.com/en-us/library/system.web.ui.page.aspx) object calls [EnsureChildControls](http://msdn.microsoft.com/en-us/library/system.web.ui.control.ensurechildcontrols.aspx) for each control and for the page.)  The [Page](http://msdn.microsoft.com/en-us/library/system.web.ui.page.aspx) object raises the [PreRender](http://msdn.microsoft.com/en-us/library/system.web.ui.control.prerender.aspx) event on the [Page](http://msdn.microsoft.com/en-us/library/system.web.ui.page.aspx) object, and then recursively does the same for each child control. The [PreRender](http://msdn.microsoft.com/en-us/library/system.web.ui.control.prerender.aspx) event of individual controls occurs after the [PreRender](http://msdn.microsoft.com/en-us/library/system.web.ui.control.prerender.aspx) event of the page.  Use the event to make final changes to the contents of the page or its controls before the rendering stage begins. |
| [PreRenderComplete](http://msdn.microsoft.com/en-us/library/system.web.ui.page.prerendercomplete.aspx) | Raised after each data bound control whose [DataSourceID](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.databoundcontrol.datasourceid.aspx) property is set calls its [DataBind](http://msdn.microsoft.com/en-us/library/system.web.ui.control.databind.aspx) method. For more information, see [Data Binding Events for Data-Bound Controls](http://msdn.microsoft.com/en-us/library/ms178472.aspx#data_binding_events_for_databound_controls) later in this topic. |
| [SaveStateComplete](http://msdn.microsoft.com/en-us/library/system.web.ui.page.savestatecomplete.aspx) | Raised after view state and control state have been saved for the page and for all controls. Any changes to the page or controls at this point affect rendering, but the changes will not be retrieved on the next postback. |
| [Render](http://msdn.microsoft.com/en-us/library/system.web.ui.control.render.aspx) | This is not an event; instead, at this stage of processing, the [Page](http://msdn.microsoft.com/en-us/library/system.web.ui.page.aspx) object calls this method on each control. All ASP.NET Web server controls have a [Render](http://msdn.microsoft.com/en-us/library/system.web.ui.control.render.aspx) method that writes out the control's markup to send to the browser.  If you create a custom control, you typically override this method to output the control's markup. However, if your custom control incorporates only standard ASP.NET Web server controls and no custom markup, you do not need to override the [Render](http://msdn.microsoft.com/en-us/library/system.web.ui.control.render.aspx) method. For more information, see [Developing Custom ASP.NET Server Controls](http://msdn.microsoft.com/en-us/library/zt27tfhy.aspx).  A user control (an .ascx file) automatically incorporates rendering, so you do not need to explicitly render the control in code. |
| [Unload](http://msdn.microsoft.com/en-us/library/system.web.ui.control.unload.aspx) | Raised for each control and then for the page.  In controls, use this event to do final cleanup for specific controls, such as closing control-specific database connections.  For the page itself, use this event to do final cleanup work, such as closing open files and database connections, or finishing up logging or other request-specific tasks.  **Note**  During the unload stage, the page and its controls have been rendered, so you cannot make further changes to the response stream. If you attempt to call a method such as the Response.Write method, the page will throw an exception. |

**Server control events**

ASP.NET web form supports many server controls like Button, TextBox, etc. Each control has associated events. There are three types of server control events.

Postback event   
Cached event   
Validation event

#### Postback event

This event sends the page to server for processing. This causes the page a round-trip to the server.

#### Cached event

This event stores page data that gets processed when page is submitted to the server by postback event.

#### Validation event

This event is handled on the page just before the page is posted back to server.

#### The order of server control events on a Web form is below.

First validations Event occurs just before the page is submitted to the server.  
Postback Event occurs that cause the page to be submitted to the server.  
Page\_Init and Page\_Load events are handled.  
Cached events are handled.  
Lastly, the event that caused the postback is processed.

**Post Back**

#### What is IsPostback in asp.net?

*PostBack is the name given to the process of submitting an ASP.NET page to the server for processing.* PostBack is done if certain credentials of the page are to be checked against a database (such as verification of username and password). This is something that a client machine is not able to accomplish and thus these details have to be ‘posted back’ to the server.  
  
Usage of IsPostBack in ASP.NET-  
  
IsPostBack is a Boolean property of a page when is set (=true) when a page is first loaded. Thus, the first time that the page loads the IsPostBack flag is false and for subsequent PostBacks, it is true. Each time a PostBack occurs, the entire page including the Page\_Load is ‘posted back‘ and executed.

#### Explain the concepts and capabilities of cross page posting.

**Cross Page Posting**: It refers to the scenario where on event of some controls posts from one page to another instead of a normal postback. Normal postback is when for e.g. a button (or any control that postbacks) is clicked and web page is submits the page back to itself which means a return trip. In Cross page posting, on click of a button there would be no return trip.

Cross-page posting is done at the control level. It is possible to create a page that posts to different pages depending on what button the user clicks on. It is handled by done by changing the postbackurl property of the controls.

#### How to make Cross Page Postback in ASP.Net 2.0?

Button control has property **PostBackUrl** that can be set to URL of any page in our ASP.Net WebSite where we want to transfer all form values to.  
Along with that Asp.Net 2.0 **Page** class has a property **PreviousPage** that allows us to get reference to the Page object that initiated the postback (in other words to get the actual reference to the Page object of the aspx page on which user clicked the Submit button on a HTML form).  
  
So for example lets create two sample pages in our Web Application:

* SourcePage.aspx
* DestinationPage.aspx

In SourcePage in Html form we will put two TextBox controls (one for First Name and one for Last Name) and one Button component and set its PostBackUrl property to "~/DestinationPage.aspx".   
  
SourcePage.aspx:

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

<div>

First Name:&nbsp;<asp:TextBox ID="FirstName" runat="server"></asp:TextBox><br />

Last Name:&nbsp;<asp:TextBox ID="LastName" runat="server"></asp:TextBox><br /><br />

<asp:Button ID="Button1" runat="server" Text="Submit To Destination Page" PostBackUrl="~/CrossPagePostbacks/DestinationPage.aspx" />

</div>

</form>

When our user clicks the Submit button, all the values from the HTML Form on SourcePage.aspx will be transfered to the DestinationPage.aspx and we will also be able to get reference to the SourcePage.aspx in our DestinationPage with the **PreviousPage** property like this:  
  
So in our DestinationPage.aspx.cs code-behind we can easily access two TextBox controls on SourcePage.aspx and show them in two label controls like this:

protected void Page\_Load(object sender, EventArgs e)

{

// first check if we had a cross page postback

if ( (PreviousPage != null) && (PreviousPage.IsCrossPagePostBack) )

{

Page previousPage = PreviousPage;

TextBox firstName = (TextBox)previousPage.FindControl("FirstName");

TextBox lastName = (TextBox)previousPage.FindControl("LastName");

// we can now use the values from TextBoxes and display them in two Label controls..

labelFirstName.Text = firstName.Text;

labelLastName.Text = lastName.Text;

}

}

You probably noticed that we first checked if **PreviousPage** property of current page (DestinationPage.aspx) **is NOT NULL**, this is done to avoid running our code in case that user opens our DestinationPage.aspx directly, without running a cross page postback.  
  
Also here we checked the another PreviousPage property called **IsCrossPagePostBack** to see if we really had a CrossPagePostback.  
(If Server.Transfer is used to redirect to this page, **IsCrossPagePostBack** property will be set to FALSE.  
  
TIP: We can be completely sure that we have a real CrossPagePostback ONLY IF:

1. **Page.PreviousPage** is **NOT NULL**,
2. **PreviousPage.IsCrossPagePostback** is **true**

This important to check to avoid errors in code.  
  
Now this is very useful and i'm sure you are eager to use this in your next project. But wait, we are not over yet!  
  
Finding the controls on PreviousPage with FindControl method and type-casting them from **object** to their real **type** is a little messy.  
It feels like there must be a better solution for this!  
  
And here it is: We can use the **<%@ PreviousPageType %>** directive in the header of our DestinationPage.aspx like this

<%@ PreviousPageType VirtualPath="~/SourcePage.aspx" %>

to declare our previous page type, and then we can access Public properties of the PreviousPage without typecasting.  
Now all we need to do is to create some public properties on our SourcePage.aspx.cs to expose data/Controls we want to the destionation page:

public partial class SourcePage : System.Web.UI.Page

{

public string FormFirstName

{

get { return FirstName.Text; }

}

public string FormLastName

{

get { return LastName.Text; }

}

}

And then we can change the Page\_Load code in our DestinationPage.aspx to much cleaner code like this:

protected void Page\_Load(object sender, EventArgs e)

{

// first check if we had a cross page postback

if ( (PreviousPage != null) && (PreviousPage.IsCrossPagePostBack) )

{

SourcePage prevPage = PreviousPage;

// we can now use the values from textboxes and display them in two Label controls..

labelFirstName.Text = prevPage.FormFirstName;

labelLastName.Text = prevPage.FormLastName;

}

}

**SourcePage** type used in the code is offcourse name of the partial class defined is SourcePage.aspx.cs that inherits **System.Web.UI.Page** that is automatically created for us when we created new WebForm in VisualStudio.  
  
This code is much cleaner and easier to follow, there is no ugly typecasting, just simple property values to use to retrieve the data from previous page.  
  
Now isn't that nice?   
  
For the end, let us explain how all this actually works behind the scenes:  
  
When you set Forms Button PostbackUrl property to a page other than the current page, ASP.NET runtime adds new JavaScript **onclick** event handler with a call to JavaScript function called **WebForm\_DoPostBackWithOptions**, instead of the usual **\_\_doPostback**, that posts contents of current form to the specified target ASPX page.  
  
Also a hidden field is generated on the source page HTML code called **\_\_PREVIOUSPAGE** that contains info about the source page that is used on the target page to build a reference to the originating page (the PreviousPage property).

**What are the different techniques to send data from one web form to another web form?**  **1. Query strings :**  
Use these strings to pass information between requests and responses as part of the Web address. Query strings are visible to the user, so they should not contain secure information such as passwords.  
  
**2. Cookies :**  
Use cookies to store small amounts of information on a client. Clients might refuse cookies, so your code has to anticipate that possibility.

**3. Session state :**  
Use Session state variables to store items that you want keep local to the current session (single user).  
  
**4. Application state :**  
Use Application state variables to store items that you want be available to all users of the application.

**Web Farm and Web Garden**

**What does the term Scalability mean?**  
Web applications that serve a large number of users or that present large amounts of data need to able to add capacity as users’ demands increase. The ability to add capacity to an application is called scalability. ASP.[NET Web](http://venkataspinterview.blogspot.com/2008/10/aspnet-interview-questions-on-web-farm.html) applications support this concept through their ability to run in multiple processes and to have those processes distributed across multiple CPUs and/or multiple servers.

## What is the difference between Web Farm and Web Garden?

###### Web Farm

After developing our asp.net web application we host it on IIS Server. Now one standalone server is sufficient to process ASP.NET Request and response for a small web sites but when the site comes for big organization where there an millions of daily user hits then we need to host the sites on multiple Server. This is called web farms. Where *single site hosted on multiple IIS Server and they are running behind the Load Balancer.*

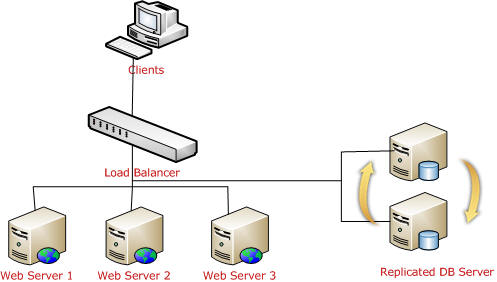
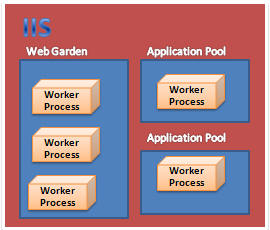
**

Fig: General Web Farm Architecture

This is the most common scenarios for any web based production environment. Where Client will hit an Virtual IP (vIP), Which is the IP address of Load Balancer. When Load balancer received the request based on the server load it will redirect the request to particular Server.

###### Web Garden

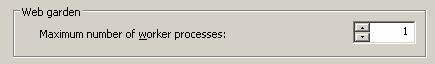
All IIS Request process by worker process ( w3wp.exe). By default each and every application pool contain single worker process. But an application pool with multiple worker process is called Web Garden. Many worker processes with same Application Pool can sometimes provide better throughput performance and application response time. And Each Worker Process Should have there own Thread and Own Memory space.



There is some Certain Restriction to use Web Garden with your web application. If we use Session Mode to "in proc", our application will not work correctly because session will be handled by different Worker Process. For Avoid this Type of problem we should have to use Session Mode "out proc" and we can use "Session State Server" or "SQL-Server Session State".

## How to Configure Web Garden?

Right Click on Application Pool > Properties > GoTo Performance Tab   
In bottom Group Section Increase the Worker Process Count.



**If your web server has multiple processors, how can you specify that ASP.NET runs on all or some of the CPUs?**  
If your server has multiple processors, you can specify that ASP.NET runs on all or some of the CPUs by setting the **webGarden** attribute of the processModel element in the server’s Machine.config file.

**What are the implications on Application and** [**Session state**](http://venkataspinterview.blogspot.com/2008/10/aspnet-interview-questions-on-web-farm.html) **variables in a web farm or a web garden?**  
In both a **Web garden** and a **Web farm**, client requests are directed to the ASP.NET process that is currently least busy. That means that a single client can interact with different CPUs or servers over the course of his or her session. This has the following implications for Application and Session state variables:  
  
**1.Application state variables are unique to each separate instance of the Web application.**  
Clients can share information through Application state if the Web application is running on a Web garden or a Web farm.  
  
**2. Session state variables are stored in-process by default.**  
To enable Session state in a Web garden or Web farm, you need to specify a Session state provider.  
  
**How can you share Application State in a web farm or a web garden?**  
To share data across multiple sessions in a **Web garden** or **Web farm**, you must save and restore the information using a resource that is available to all the processes. This can be done through an XML file, a database, or some other resource using the standard file or database access methods.  
  
**What are the two built-in ways provided by ASP.NET to share Session state information across a Web garden or Web farm?**  
ASP.NET provides two built-in ways to share Session state information across a Web garden or Web farm. You can share Session state using:  
  
**A state server, as specified by a network location**  
This technique is simple to implement and doesn’t require you to install Microsoft SQL Server.  
**A SQL database, as specified by a SQL connection**  
This technique provides the best performance for storing and retrieving state information.  
  
**What are the steps to follow to share Session state information using a state server?**  
To share Session state information using a state server, follow these steps:  
**1.** In the Web application’s Web.config file, set the sessionState element’s mode and stateConnectionString attributes.  
**2.** Run the aspnet\_state.exe utility on the Session state server. The aspnet\_state.exe utility is installed in the \WINDOWS\Microsoft.NET \Framework\version folder when you install Visual Studio .NET Professional or Visual Studio .NET Enterprise Architect editions.  
  
**What are the steps to follow to share Session state information using a SQL database?**  
To share Session state information using a SQL database, follow these steps:  
**1.** In the Web application’s Web.config file, set the sessionState element’s mode and sqlConnectionString attributes.  
**2.** Run the InstallSqlState.sql utility on the Session state server. This utility installs the SQL database that shares Session state information across processes. The InstallSqlState.sql utility is installed in the \WINDOWS\Microsoft.NET \Framework\version folder when you install Visual Studio .NET Professional, Visual Studio .NET Enterprise Developer, or Visual Studio .NET Enterprise Architect editions.

**State Management**

#### What is state management in ASP.NET?

State management is implemented in order to retain information about the user requests. Web pages are stateless. Each request creates new page without retaining any previous information about the user requests. ASP.NET supports several State management techniques to maintain state information.

State management in ASP.NET can be classified into   
Client-side state management   
Server-side state management

#### Define Client-side state management and Server-side state management.

***1.Client-side state management***This maintains information on the client's machine using Cookies, View State, and Query Strings.

*Cookies*A cookie is a small text file on the client machine either in the client's file system or memory of client browser session. Cookies are not good for sensitive data. Moreover, Cookies can be disabled on the browser. Thus, you can't rely on cookies for state management.

*View State*   
Each page and each control on the page has View State property. This property allows automatic retention of page and controls state between each trip to server. This means control value is maintained between page postbacks. Viewstate is implemented using \_VIEWSTATE, a hidden form field which gets created automatically on each page. You can't transmit data to other page using view state.

*Querystring*Querystring can maintain limited state information. Data can be passed from one page to another with the URL but you can send limited size of data with the URL. Most browsers allow a limit of 255 characters on URL length.

***2. Server-side state management***  
This kind of mechanism retains state in the server.

*Application State*The data stored in an application object can be shared by all the sessions of the application. The application object stores data in the key value pair.

*Session State*Session state stores session-specific information and the information is visible within the session only. ASP.NET creates unique sessionId for each session of the application. SessionIDs are maintained either by an HTTP cookie or a modified URL, as set in the application's configuration settings. By default, SessionID values are stored in a cookie.

*Database* Database can be used to store large state information. Database support is used in combination with cookies or session state.

#### What are the ways to retain variables between requests?

Following are the ways to retain variables between requests:

**Context.Handler**This object can be used to retrieve public members of the web form from a subsequent web page.

**Querystring**Querystring is used to pass information between requests as part of the web address. Since it is visible to the use, we can't use it to send any secured data.

**Cookies**Cookies are stored small amount of information on a client. But we can't rely on cookies since many clients can refuse cookies.

**View state**View state stores items added to a page’s ViewState property as hidden fields on the page.

**Session state**Session state stores items that are local to the current session.

**Application state**Application state stores items that are available to all users of the application.

#### Cookies

#### What are cookies in ASP.NET? Cookies are small pieces of information stored on the client computer. Use cookies to store small amounts of information on the client’s machine. Web sites often use cookies to store user preferences or other information that is client-specific. Because cookies can be refused, it is important to check whether the browser allows them before you try to create them. They are limited to storing only character data and they are limited to 4K in size. Explain the types of cookies in ASP.NET.

There are two types of cookies in ASP.NET

Single valued cookies

Example:  
request.cookies(”UserName”).value=”myCookies”

Multivalued cookies

Example   
request.cookies(”CookiName”)(”UserName”)=”Lucky”  
request.cookies(”CookiName”)(”UserID”)=”1234?

**How do you create a permanent cookie?**

Setting a permanent cookie is similar to Session cookie, except give the cookie an expiration date too. It is very common that you don't specify any arbitrary expiration date, but instead expire the cookie relative to the current date, using the DateAdd() function.  
  
Response.Cookies("Name") = "myCookie"  
Response.Cookies("Name").Expires = DateAdd("m", 1, Now())

#### Describe the Cookies collection in ASP.NET.

* Cookies are text files that store information about the user.
* A user is differentiated from the other by the web server with the help of the cookies. It can also determine where the user had been before with them.
* A cookie can store information only when the user sends it.
* Cookies are stored on client’s machine. The maximum number of cookies allowed is 300 and each of them can carry at the most 4 kb of data.

**What are different types of Cookies?**Session Cookies  
Persistent Cookies  
  
**What are Session Cookies?**   
Session cookies are stored in-memory during the client browser session. When the browser is closed the session cookies are lost.  
  
**How can you create Session Cookies?**You can create session cookies by calling the Add method of the Cookies collection on the Response object. The Cookies collection contains individual cookie objects of type HttpCookie.  
  
//Code to create a UserName cookie containing the name David.  
HttpCookie CookieObject = new HttpCookie("UserName", "David");  
Response.Cookies.Add(CookieObject);   
  
//Code to read the Cookie created above  
Request.Cookies["UserName"].Value;

**What is the difference between Session Cookies and Persistent Cookies?**Persistent Cookies are same as Session Cookies except that, persistent cookies have an expiration date. The expiration date indicates to the browser that it should write the cookie to the client's hard drive. Keep in mind that because a user can delete cookies from their machine that there is no guarantee that a cookie you "drop" on a user machine will be there the next time they visit your site.  
  
**What are Persistent Cookies used for?**   
Persistent cookies are generally used to store information that identifies a returning user to a Web site. Typical information found in Persistent Cookies includes user names or user IDs.  
  
**How do you create a Persistent Cookie?**You create a persistent cookie the same way as session cookies except that you set the Expires property to a Date in the future which will store the Cookie to the client computer hard drive.  
  
//Code to create a UserName Persistent Cookie that lives for 10 days  
HttpCookie CookieObject = new HttpCookie("UserName", "David");  
CookieObject.Expires = DateTime.Now.AddDays(10);  
Response.Cookies.Add(CookieObject);   
  
//Code to read the Cookie created above  
Request.Cookies["UserName"].Value;   
  
**What is Cookie Dictionary?**   
A cookie dictionary is a single cookie object that stores multiple pieces of information. You use the Values property to access and assign new values to the cookie dictionary.  
  
**Give an example using Cookie Dictionary?**//Code to create a Cookie Dictionary  
HttpCookie CookieObject = new HttpCookie("UserPreference");   
  
//Use the Values property to assign new values to the cookie dictionary  
CookieObject.Values.Add("UserName", "David");  
CookieObject.Values.Add("Country", "USA");  
CookieObject.Values.Add("PreviousVisit", DateTime.Now.ToString());  
CookieObject.Expires = DateTime.MaxValue;   
  
//Add the Cookie to the client machine using the Response object  
Response.Cookies.Add(CookieObject);  
  
//Code to read the Cookie created above  
HttpCookie ObjectCookie = Request.Cookies["UserPreference"];  
string UserName = ObjectCookie.Values["UserName"];  
string Country = ObjectCookie.Values["Country"];  
string PreviousVisit = ObjectCookie.Values["PreviousVisit"];  
  
**What are the advantages of Using Cookies?**   
**1.** Cookies do not require any server resources since they are stored on the client.  
**2.** Cookies are easy to implement.  
**3.** You can configure cookies to expire when the browser session ends (session cookies) or they can exist for a specified length of time on the client computer (persistent cookies).  
  
**What are the disadvantages of Using Cookies?  
1.** Users can delete a cookies.  
**2.** Users browser can refuse cookies,so your code has to anticipate that possibility.  
**3.** Cookies exist as plain text on the client machine and they may pose a possible security risk as anyone can open and tamper with cookies.  
  
**How do you create a Cookie that never expires?**To create a Cookie that never expires set the Expires property of the Cookie object to DateTime.MaxValue.  
  
**Are Cookies secure?**   
No, Cookies are not secure. You must pay attention to the type of data you store in cookies.  
**1.** Cookies are not designed to store critical information so storing passwords in a cookie is a bad idea.  
**2.** Keep the lifetime of a cookie as short as practically possible.  
**3.** Encrypt cookie data to help protect the values stored in the cookie.

#### ViewState

#### What is ViewState? Web forms have very short lifetimes.In ASP.NET, the data that is entered in controls is encoded and stored in a hidden field. This encoded data is then sent with each request and restored to controls in Page\_Init. The data in these controls is then available in the Page\_Load event. The data that ASP.NET preserves between requests is called the Web form’s [view state](http://venkataspinterview.blogspot.com/2008/07/aspnet-viewstate-related-interview.html).

Viewstate is the mechanism that automatically saves the values of the page's items just before rendering the page. It stores items added to a page’s ViewState property as hidden fields on the page.

Viewstate is used to maintain or retain values on postback. It helps in preserving a page. Viewstate is internally maintained as a hidden field in encrypted form along with a key.

#### Explain its benefits and limitations.

*Advantages:*

i) No server resources.  
ii) Viewstate ensures security because it stores the data in encrypted format.  
iii) View states are simple. They are used by enabling or disabling the viewstate properties.  
iv) It is based on the wish of developer that they want to implement it at the page level or at control level.

*Disadvantages:*

i) If large amount of data is stored on the page, then page load might cause a problem.   
ii) Does not track across pages. Viewstate information does not automatically transfer from page to page.

**What are the disadvantages of using ViewState?**1. On all page visits, during the save view state stage the Page class gathers the collective view state for all of the controls in its control hierarchy and serializes the state to a base-64 encoded string. (This is the string that is emitted in the hidden \_\_ViewState form filed.) Similarly, on postbacks, the load view state stage needs to deserialize the persisted view state data, and update the pertinent controls in the control hierarchy.  
  
2. The \_\_ViewState hidden form field adds extra size to the Web page that the client must download. For some view state-heavy pages, this can be tens of kilobytes of data, which can require several extra seconds (or minutes!) for modem users to download. Also, when posting back, the \_\_ViewState form field must be sent back to the Web server in the HTTP POST headers, thereby increasing the postback request time.

**How do you enable or disable a ViewState for a control on the page?**   
Every ASP.NET control has a property called EnableViewState. If EnableViewState is set to true ViewState is enabled for the control. If EnableViewState is set to false ViewState is disabled for the control.  
  
**How do you enable or disable a ViewState at the** [**page level**](http://venkataspinterview.blogspot.com/2008/07/aspnet-viewstate-related-interview.html)**?**At the page level you can enable or disable ViewState using EnableViewState property of the page.  
  
**What is the name of the hidden form field in which ViewState of the page is saved?**   
\_\_ViewState  
  
**What are the performance implications of ViewState?**ViewState is usually good to retain the state of the controls on the webform across postbacks. If you have a huge DataGrid with tons of data being loaded on every page load. It is a good idea to disable the ViewState of the DataGrid for the page to load faster. If the ViewState of a large DataGrid is not disabled, ViewState can easily get very large, on the order of tens of kilobytes. Not only does the \_\_ViewState form field cause slower downloads, but, whenever the user posts back the Web page, the contents of this hidden form field must be posted back in the HTTP request, thereby lengthening the request time, as well.  
  
**When does ViewState restoration happens?**   
During the Page\_Init event

**Is ViewState encoded?**   
Yes, ViewState is base-64 encoded.  
  
**Can you encrypt ViewState of Page?**Yes, we can use the LosFormatter class to encrypt ViewState of Page  
  
**Can the HTML controls retian State accross postbacks?**   
No, by default HTML controls donot retain state accross postbacks.  
  
**Can you make HTML controls retain State accross postbacks?**Yes, HTML controls can retain State accross postbacks, if you convert HTML controls to Server Controls. There are 2 ways to convert HTML control to Server Controls.  
  
1. Right click on the HTML Control and then click "Run As [Server Control](http://venkataspinterview.blogspot.com/2008/07/aspnet-viewstate-related-interview.html)"  
Or  
2. Set runat="server" attribute for the Control.  
  
**Is ViewState supported in** [**classic ASP**](http://venkataspinterview.blogspot.com/2008/07/aspnet-viewstate-related-interview.html)**?**   
No,ViewState is introduced in asp.net, it was not in classic asp.  
  
When a form is submitted in classic ASP, all form values are cleared. Suppose you have submitted a form with a lot of information and the server comes back with an error. You will have to go back to the form and correct the information. You click the back button, and what happens.......ALL form values are CLEARED, and you will have to start all over again! The site did not maintain your ViewState.  
  
When a form is submitted in ASP .NET, the form reappears in the browser window together with all form values. How come? This is because ASP .NET maintains your ViewState. The ViewState indicates the status of the page when submitted to the server.  
  
**Is ViewState of one page available to another page?**No, ViewState of a Page is available only in that page. You cannot access ViewState of one page from another page.  
  
**Can you programatically store and retrieve data from ViewState?**   
Yes. In ASP.NET you can programatically store and retrieve data from ViewState. See the example below  
  
//Save the value in ViewState object  
ViewState("SomeVar") = txtFirstName.text;   
  
//Retrieve the value from ViewState object  
String strFirstName = ViewState("SomeVar").ToString();  
  
**Can someone view the Page HTML source and read ViewState?**   
No. ViewState is base-64 encoded. Hence you cannot read ViewState. If you right click on the Page and View Source you will find \_\_ViewState is base-64 encoded.  
  
**What happens during the Page\_Init event?**The server controls are loaded and initialized from the Web form’s view state. This is the first step in a Web form’s life cycle.

#### What are the characteristics of ViewState in ASP.NET?

It allows the state of objects (serializable) to be stored in a hidden field on the page.  
It is transported to the client and back to the server, and is not stored on the server or any other external source.   
It is used to retain the state of server-side objects between postbacks.

#### Explain how to access ViewState value of this page in the next page.

PreviousPage property is set to the page property of the nest page to access the viewstate value of the page in the next page. Page poster = this.PreviousPage;  
Once that is done, a control can be found from the previous page and its state can be read.   
Label posterLabel = poster.findControl("myLabel");  
string lbl = posterLabel.Text;

#### What is the lifespan for items stored in ViewState?

Items stored in the ViewState exist for the life of the current page.   
This includes postbacks (to the same page).

#### Explain the purpose of EnableViewState property

It allows the page to save the users input on a form across postbacks.   
It saves the server-side values for a given control into ViewState, which is stored as a hidden value on the page before  sending the page to the clients browser.   
When the page is posted back to the server, the server control is recreated with the state stored in viewstate.

#### What is a View State?

* If a site happens to not maintain a ViewState, then if a user has entered some information in a large form with many input fields and the page is refreshes, then the values filled up in the form are lost.
* The same situation can also occur on submitting the form. If the validations return an error, the user has to refill the form.
* Thus, submitting a form clears up all form values as the site does not maintain any state called ViewState.
* In ASP .NET, the ViewState of a form is maintained with a built-in state management technique keeps the state of the controls during subsequent postbacks by a particular user.
* The ViewState indicates the status of the page when submitted to the server. The status is defined through a hidden field placed on each page with a <form runat="server"> control.  
  <input type="hidden" name="\_\_VIEWSTATE" value="CareerRide">
* The ViewState option can be disabled by including the directive <%@ Page EnableViewState="false"%> at the top of an .aspx page
* If a ViewState of a certain control has to be disabled, then set EnableViewState="false".

#### Query String

**Give an example of using querystrings to send data from one page to another?**  
*Query strings are a very simple and popular technique to pass data from one Web page to the next. You send data as part of the URL.* In the below example FName and LName are sent as part of the URL. In the page load of QueryStrings2.[aspx](http://venkataspinterview.blogspot.com/2008/07/interview-questions-on-query-strings-in.html) we use Request.QueryString to read the values. As we are sending more than one query string we use the & symbol to seperate query strings.  
  
**//Code to send query strings FName and LName as part of the URL**QueryStrings2.aspx?FName=David&LName=Boon   
  
protected void Page\_Load(object sender, EventArgs e)  
{  
//Code to read Query String values  
string FirstName = Request.QueryString["FName"];  
string LastName = Request.QueryString["LName"];  
Response.Write("Data from QueryStrings1.aspx : " + FirstName + ", " + LastName);  
}

**Give an example to send Query Strings from code?**   
You can send query strings from server side code using the Response.Redirect() method as shown below.  
Response.Redirect("QueryStrings2.aspx?FName=David&LName=Boon");  
  
**What are the advantages of using Query Strings?  
1.** Query strings are easy to implement.  
**2.** Browser support for passing values in a query string is nearly universal.  
**3.** Query strings are contained in the HTTP request for a specific URL and do not require server resources.  
  
**What are the disadvantages of using querystrings to send data from one page to another?  
1.** Query strings are insecure because the information in the query string is directly visible to the user on the address line in the browser.  
**2.** Many browsers impose a 255 URL character limit which can limit their flexibility.

#### Application and Session

**What is a Session?**A Session is a unique instance of the browser. A single user can have multiple instances of the browser running on his or her machine. If each instance visits your Web application, each instance has a unique session. A session starts when a user accesses a page on a Web site for the first time, at which time they are assigned a unique session ID. The server stores the user's session ID in the Session.SessionID property.

**Define Session, SessionId and Session State in ASP.NET.**

A session is the duration of connectivity between a client and a server application.

SessionId is used to identify request from the browser. By default, value of SessionId is stored in a cookie. You can configure the application to store SessionId in the URL for a "cookieless" session.

**What is Session Identifier?**

Session Identifier is used to identify session. It has SessionID property. When a page is requested, browser sends a cookie with a session identifier. This identifier is used by the web server to determine if it belongs to an existing session. If not, a Session ID (120 - bit string) is generated by the web server and sent along with the response.

**What is the default session timeout period?**   
20 minutes.  
  
**Where do you generally specify the Session Timeout?**You specify the Session Timeout setting in the web.config file.  
  
**Can you specify Session Timeout in a code behind file?**   
Yes, can specify the Session.Timeout property as shown below in a code behind file.  
Session.Timeout = 10;  
  
**How do you end a user session?**You can call the Session.Abandon() method to end a user session. If a user then tries to access a page the server will assign them a new session ID and it will clear all the previous session variables. You'll typically use Session.Abandon() on log-out pages.  
  
**What type of data can you store in Application State and** [**Session State**](http://venkataspinterview.blogspot.com/2008/07/aspnet-session-state-and-application.html) **variables?**   
Application State and Session State variables are used to store data that you want to keep for the lifetime of an application or for the lifetime of a session. You can store any type of data in the Application or Session state, including objects.  
  
**Are Application State or Session State variables type safe?**No, Application and Session state variables are created on the fly, without variable name or type checking.  
  
**Do maintaining Session state affects performance?**   
Yes  
  
**Can you turn off Session state?**Yes, Session state can be turned off at the application and [page levels](http://venkataspinterview.blogspot.com/2008/07/aspnet-session-state-and-application.html).  
  
**How do you disable Session state for a Web form?**To turn Session state off for a Web form set EnableSessionState property of the Page to False.  
  
**How do you turn Session state off for an entire web application?**   
In the Web.config file, set the sessionstate tag to False.

**Advantages and disadvantages of using Session State.**

The advantages of using session state are as follows:   
It is easy to implement.  
It ensures data durability, since session state retains data even if ASP.NET work process restarts as data in Session State is stored in other process space.   
It works in the multi-process configuration, thus ensures platform scalability.

The disadvantages of using session state are:   
Since data in session state is stored in server memory, it is not advisable to use session state when working with large sum of data. Session state variable stays in memory until you destroy it, so too many variables in the memory effect performance.

#### What are the different types of Session state management options available with ASP.NET? Explain them.

ASP.NET provides In-Process and Out-of-Process state management.

* In-Process stores the session in memory on the web server.
* Out-of-Process Session state management stores data in an external data source. The external data source may be either a SQL Server or a State Server service.  
  Out-of-Process state management requires that all objects stored in session are serializable.

***What are the Session State Modes? Define each Session State mode supported by ASP.NET.***

ASP.NET supports three Session State modes.

* InProc
* State Server
* SQL Server

**InProc Mode**This mode stores the session data in the ASP.NET worker process.  
This is the fastest among all of the storage modes.  
This mode effects performance if the amount of data to be stored is large.  
If ASP.NET worker process recycles or application domain restarts, the session state will be lost.

**State Server mode**   
In this mode, the session state is serialized and stored in memory in a separate process.  
State Server can be maintained on a different system.   
State Server mode involves overhead since it requires serialization and de-serialization of objects.   
State Server mode is slower than InProc mode as this stores data in an external process.

**SQL Server Mode**In this storage mode, the Session data is serialized and stored in a database table in the SQL Server database.   
This is reliable and secures storage of a session state.   
This mode can be used in the web farms.   
It involves overhead in serialization and de-serialization of the objects.  
SQL Server is more secure than the InProc or the State server mode.   
  
**What are Application State variables?**   
Application State variables are global variables that are available from anywhere in the application. All Sessions can access Application State variables.  
  
**How to add and remove data to Application State Variables?**   
//Code to add data to Application State  
Application.Add("AppName", "Sample");  
  
//Code to remove data from Application State  
Application.Remove("AppName");   
  
**How do you remove all Application State Variables data?**   
//Code to remove all Application State Variables data  
Application.RemoveAll();

**Are Application state variables available throughout the current process?**   
Yes, Application state variables are available throughout the current process, but not across processes. If an application is scaled to run on multiple servers or on multiple processors within a server, each process has its own Application state.

#### What are application and session state variables? Why do we need them?

By default, ASP.NET maintains page data between the requests using mechanism called View state. Every web form has view state property to retain data. The form's view state can retain data only in that form. So, when you need one form data in other form, you can't rely on view state. ASP.NET provides state variables in the Application or Session objects that helps in maintaining state of the form which can be accessed in other form.

#### Application state variables

The data stored in these variables is available to all the users i.e. all the active sessions.

#### Session state variables

These are available to the single session who has created the variables.

#### Point to be noted about Application and Session state variables

These variable can store any type of data.  
Maintaining Session state affects performance.   
Session state can be turned off at the application and page levels.  
Application state variables are available throughout the current process, but not across processes.

**What are the advantages of client and server side State Management?**

**Advantages of Client – Side State Management:**   
  
1. Better Scalability: With server-side state management, each client that connects to the Web server consumes memory on the Web server. If a Web site has hundreds or thousands of simultaneous users, the memory consumed by storing state management information can become a limiting factor. Pushing this burden to the clients removes that potential bottleneck.   
  
2. Supports multiple Web servers: With client-side state management, you can distribute incoming requests across multiple Web servers with no changes to your application because the client provides all the information the Web server needs to process the request. With server-side state management, if a client switches servers in the middle of the session, the new server does not necessarily have access to the client’s state information. You can use multiple servers with server-side state management, but you need either intelligent load-balancing (to always forward requests from a client to the same server) or centralized state management (where state is stored in a central database that all Web servers access).   
  
**Advantages of Server – Side State Management:**   
  
1. Better security: Client-side state management information can be captured (either in transit or while it is stored on the client) or maliciously modified. Therefore, you should never use client-side state management to store confidential information, such as a password, authorization level, or authentication status.   
  
2. Reduced bandwidth: If you store large amounts of state management information, sending that information back and forth to the client can increase bandwidth utilization and page load times, potentially increasing your costs and reducing scalability. The increased bandwidth usage affects mobile clients most of all, because they often have very slow connections. Instead, you should store large amounts of state management data (say, more than 1 KB) on the server.

**I don’t know about Application and Session State management. Can anyone explain me with simple example?**

Application state allows you to store global objects that can be accessed by any client. Application

state is based on the System.Web.HttpApplicationState class, which is provided in all web

pages through the built-in Application object.

Application state is similar to session state but the difference is that session state is user specific. It supports the same type of objects, retains

information on the server, and uses the same dictionary-based syntax. A common example

with application state is a global counter that tracks how many times an operation has been

performed by all the web application’s clients.

For example, you could create a Global.asax event handler that tracks how many sessions

have been created or how many requests have been received into the application. Or you can

use similar logic in the Page.Load event handler to track how many times a given page has

been requested by various clients. Here’s an example of the latter:

protected void Page\_Load(Object sender, EventArgs e)

{

// Retrieve the current counter value.

int count = 0;

if (Application["HitCounterForOrderPage"] != null)

{

count = (int)Application["HitCounterForOrderPage"];

}

// Increment the counter.

count++;

// Store the current counter value.

Application["HitCounterForOrderPage"] = count;

lblCounter.Text = count.ToString();

}

**Caching technique**

#### Define Caching in ASP.NET.

Caching technique allows to store/cache page output or application data on the client. The cached information is used to serve subsequent requests that avoid the overhead of recreating the same information. This enhances performance when same information is requested many times by the user.

*Caching is the process of storing frequently used data, usually data that is costly to generate, for reuse.* Typically this data is stored in memory since retrieving data from memory is much more efficient than retrieving the data from other locations, such as a database.

ASP.NET has several facilities for supporting caching: a ***Cache API*** for storing arbitrary data and **an Output Cache** used to store frequently requested Pages.

Caching is the most effective technique you can use to improve the performance of your ASP.NET web application. Designing your application with caching in mind, improves both the performance and the scalability of that application.

Caching is about storing data in memory the first time it is requested and then re-using it for the following requests for a specified period of time.

#### How caching is useful? Explain with an example.

A catalog used on an e-commerce site might only change once a week. An ASP.NET Web application could be built that provides a front-end interface for that catalog, allowing customers to purchase products. When a customer is simply browsing the catalog, the system is making (in most cases) network calls to a back-end database server. The database server is also doing calculations on the data, such as a join query, and returning results.

This type of configuration is quite common, be it a catalog or some other type of commonly requested data from a database. However, the design in the above example can be improved upon. We know that the data in the database only changes once a week, and we know that there are several performance costs associated with retrieving the data:

1. Executing of the ASP.NET code to make the database request.
2. Use of the network for the Web server to communicate with the database server.
3. Work done on the database server to compile and execute the query (or simply execute stored procedure).

Caching allows us to eliminate much of the above work and improve the performance and scalability of our application. We can improve performance by caching the results and serving them statically (versus dynamically on each request) and our scalability increases since we're using fewer resources to service each request.

#### What are the types of caching in ASP.NET?

ASP.NET provides two types of caching that you can use to create high-performance Web applications.

* The first is called ***output caching***, which allows you to store dynamic page and user control responses on any HTTP 1.1 cache-capable device in the output stream, from the originating server to the requesting browser. On subsequent requests, the page or user control code is not executed; the cached output is used to satisfy the request.
* The second type of caching is traditional application ***data caching***, which you can use to programmatically store arbitrary objects, such as data sets, to server memory so that your application can save the time and resources it takes to recreate them.

#### What are the advantages of Caching? Is caching good? Justify by giving reasons.

It increases performance of the application by serving user with cached output.   
It decreases server round trips for fetching data from database by persisting data in the memory.  
It greatly reduces overhead from server resources.

#### What is the difference between caching and pooling? How to implement caching and pooling?

The difference is simple. Pooling is appropriate if you don't care about the internal state of a particular class / type. Pooling is mainly motivated by technical reasons like performance, latency or memory optimizations.   
Caching is all about state. You want to retain the state for performance reasons and store (cache) it in objects between calls. So, instead of loading the state from the database in every request, you could cache the data between requests in valid objects.

Pooled objects are in an undefined state, but the state of cached objects is always well defined.

A *pool* is a collection of *stateless* objects. Eample – database connection pools, thread pools, and Servlet pools.

A *cache* is a collection of *stateful* objects.

#### What are the types of Caching in ASP.NET?

Caching in ASP.NET can be of the following types  
Page Output Caching  
Page Fragment Caching  
Data Caching

#### Explain in brief each kind of caching in ASP.NET.

**Page Output Caching**  
This type of caching is implemented by placing OutputCache directive at the top of the .aspx page at design time.   
For example:  
<%@OutputCache Duration= "30" VaryByParam= "DepartmentId"%>  
  
The duration parameter specifies for how long the page would be in cache and the VaryByParam parameter is used to cache different version of the page.   
The VaryByParam parameter is useful when we require caching a page based on certain criteria.   
  
**Page Fragment Caching**   
This technique is used to store part of a Web form response in memory by caching a user control.   
  
**Data Caching**Data Caching is implemented by using Cache object to store and quick retrieval of application data.   
Cache object is just like application object which can be access anywhere in the application.   
The lifetime of the cache is equivalent to the lifetime of the application.

#### What is Fragment Caching in ASP.NET?

* Fragment caching refers to the caching of individual user controls within a Web Form.
* Each user control can have independent cache durations and implementations of how the caching behavior is to be applied.
* Fragment caching is useful when you need to cache only a subset of a page.
* Navigation bars, header, and footers are good candidates for fragment caching.

Fragment caching allows to cache specific portions of the page rather than the whole page. It is done by implementing the page in different parts by creating everything in form of user controls and caching each user control individually.

**What is fragment caching?**  
Caching parts of web form is called as **fragment caching**. Sometimes you want to cache only part of a Web form response. For instance, a Web form might contain many pieces of variable information plus a single large table that almost never changes. In this case, you might place that table in a Web [user control](http://venkataspinterview.blogspot.com/2008/10/aspnet-interview-questions-on-fragment.html) and store the response for that control in cache. This technique is called **fragment caching.**  
  
**What are the steps to follow to cache parts of web form?**  
To cache part of a Web form, follow these steps:  
**1.** Place the controls and content that you want to cache in a Web user control.  
**2.** Set the caching attributes for that Web user control.  
**3.** Create an instance of the Web user control on the Web form.  
  
**What is PartialCaching attribute used for?**  
You can include the **PartialCaching** attribute in the control’s class declaration to enable **fragment caching.**  
  
**What are the OutputCache directive attributes that apply only to user controls?**  
**Shared**  
Cache a single response from a user control for use on multiple Web forms. By default, ASP.NET caches a separate response for each Web form that uses a cached user control. This attribute is only available in the .NET Framework version 1.1 or later.  
  
**VaryByControl**  
Cache multiple responses for a single user **control based** on the value of one or more controls contained in the user control. Can you cache multiple versions of a user control?Yes, You can cache multiple versions of a user control based on the value of controls contained in a user control (VaryByControl) or based on a custom string (VaryByCustom).

**If a user control is read from the cache, can you access its members from code?**  
No, In general, cached controls are used to present data such as queries from a database, rather than as interactive components. However, if you do need to access a cached control from code, you must first check that the control exists. If the control is read from the cache, you can’t access its members from code. Control members are available only when the control is not read from the cache, such as when the control is first instantiated and when it is reloaded after its cache duration has expired.  
  
**When caching is set at both the Web form and user control levels, How does the cache settings interact?**   
The cache location is determined by the Web form setting. Location settings on a user control have no affect.  
If the Web form’s cache duration is longer than the user control’s, both the Web form response and the user control response will expire using the Web form setting.

#### Explain the difference between cache object and application object.

**Application Object:** Application variable/object stores an Object with a scope of availability of the entire Application unless explicitly destroyed.

**Caching Object:** The lifetime of cache is throughout the lifetime of an application or is based upon timeouts. Unlike **Application** or **Session**, items within the Cache can expire.

The Cache API is similar to **Application** and **Session**, a simple key/value pair dictionary.

#### What is Cache Callback in Cache?

The cache object has dependencies e.g. relationships to the file it stores. Cache items remove the object when these dependencies change. As a work around we would need to simply execute a callback method whenever items are removed from the cache to add the items back in cache.

**e.g.: We have a string variable string var="hello" and store it in cache. This item would be removed from the cache if value is changed from "hello" to "bye". In such a case we need to simply right a callback function which would add updated var back in cache as it’s been removed as a result of its dependencies changing.**

#### What is Scavenging?

A process where items are removed from cache in order to free the memory based on their priority. A property called "CacheItemPriority" is used to figure out the priority of each item inside the cache. This priority property is set when an item is added to the cache.

The cache automatically removes the least used items from memory, when system memory becomes low. This process is called ***scavenging***. We can specify priority values for items we add to the cache so that some items are given more priority than others:

Cache.Insert("Name", strName,

new CacheDependency(Server.MapPath("name.txt"),

DateTime.Now.AddMinutes(2), TimeSpan.Zero,

CacheItemPriority.High, null);

The *CacheItemPriority* enumeration has members to set various priority values. The *CacheItemPriority.High* assigns a priority level to an item so that the item is least likely to be deleted from the cache.

#### Show with an example how to Cache different version of same page using ASP.NET Cache object.

The ways to cache different versions on the same page using ASP.NET cache object is using OutputCache object.

Three parameters of OutputCache, has 3 parameters:

* **VarByParam:** based on input parameters passed through HTTP POST/GET.
* **VarByHeader:** based on the page header content.
* **VayByCustom:** based on declaration of an attribute and overriding GetVaryByCustomString handler.
* **VarByControl:** varies with the value of properties of ASP objects.

#### Explain how to implement Fragment Cache

Fragment cache is to store user controls individually within a web form in cache instead of the whole webform as such. The idea is to simply have different cache parameters for different user controls.

control1: <%@ OutputCache Duration="40" VaryByParam="none"%>

control2: <%@ OutputCache Duration="60" VaryByParam="none"%>

If control1 and control2 exist on a single webform together, the values of control1 and control2 would be cached for different durations based on @OutputCache directive.

#### What is Absolute and Sliding expiration in .NET?

Absolute and sliding expiration are two Time based expiration strategies.   
Absolute Expiration: Cache in this case expires at a fixed specified date or time.  
Example: Cache. Insert("ABC", ds, null, DateTime.Now.AddMinutes(1), Cache.NoSlidingExpiration);  
The cache is set to expire exactly two minutes after the user has retrieved the data.

Sliding Expiration: the cache duration increases in this case by the specified sliding expiration value every time the page is requested. More a page is requested it will remain in cache, whereas a less requested page will not remain in cache.

**Example: Cache.Insert("ABC", ds, null, Cache.NoAbsoluteExpiration, TimeSpan.FromMinutes(1));**

#### What is SQL Cache Dependency in ASP.NET?

SQL Cache Dependency in ASP.NET: It is the mechanism where the cache object gets invalidated when the related data or the related resource is modified. It is a feature in SQL Server 2005 and SQL Server 2000.

3 types of SQL-cache dependencies exist:

a. Other cache items.  
b. Files/folders.  
c. Dependencies on a database query.

#### Explain the concepts of Post Cache Substitution in .NET

It works opposite to fragment caching. The entire page is cached, except what is to be kept dynamic. When [OutputCache] attribute is used, the page is cached on both the server and the client. Whereas, you use in case of post cache substitution the page is cached on the server only. HttpResponse.WriteSubstitution method is a way to implement it.

**Which object can be used to store frequently used items in the server’s memory for quick retrieval?**  
Cache object can be used to store frequently used items in the server’s memory for quick retrieval.  
  
**Is the cache object available for all web forms with in a web application?**  
Yes, the Cache object is global, that is, data stored in the Cache object is available anywhere within a Web application. In this way, the Cache object is very similar to the intrinsic Application object.  
  
**What are the 3 different ways to store data in the Cache object?**  
**Use assignment.**  
Assigning a value to an unused key in the Cache object automatically creates that key and assigns the value to that key. Assigning a value to a key that already exists replaces the cached value with the assigned value.  
**Use the Insert method.**The Insert method uses parameters rather than assignment to create or change cached data. Insert optionally accepts parameters to establish dependencies and set expiration policy.  
**Use the Add method.**The Add method is similar to Insert; however, it requires all parameters and returns an object reference to the cached data.  
  
For example, the following Cache statements all add the same item to the cache:  
  
using System.Web.Caching;  
private void Page\_Load(object sender, System.EventArgs e)  
{   
if(!IsPostBack)   
{   
Cache["NewItem"] = "Some string data";   
Cache.Add("NewItem", "Some string data", null, Cache.NoAbsoluteExpiration, System.TimeSpan.FromMinutes(1), CacheItemPriority.Default, null);   
Cache.Insert("NewItem", "Some string data");   
}  
}

**What are absoluteExpiration and slidingExpiration parameters of the Insert and Add methods?**  
**absoluteExpiration**A DateTime object that identifies when the data should be removed from the cache. If you’re using sliding expiration, specify Cache.NoAbsoluteExpiration for this parameter.  
**slidingExpiration**A TimeSpan object that identifies how long the data should remain in the cache after the data was last accessed. If you’re using absolute expiration, specify Cache.NoSlidingExpiration for this parameter.  
  
**Which delegate can be used to notify the application when items are removed from the cache?**  
onRemoveCallback is used to notify the application when items are removed from the cache.  
  
**How do you retrieve the value of a cache item stored in the server’s memory?**  
You can retrieve the value of a cache item stored in the servers memory through the item’s key, just as you do with the Application and Session objects. Because cached items might be removed from memory, you should always check for their existence before attempting to retrieve their value, as shown in the following code:  
  
private void Button1\_Click(object sender, EventArgs e)  
{   
if (Cache["ChachedItem"] == null)   
{   
Lable1.Text = "Cached Item not found.";   
}   
else   
{   
Lable1.Text = Cache["ChachedItem"].ToString();   
}  
}  
  
**Which method can be used to remove data from the cache?**  
Cache object’s Remove method can be used to remove data from the cache as shown in the following code example / sample.  
  
private void RemoveButton\_Click(object sender, System.EventArgs e)  
{   
Cache.Remove("CachedItem");  
}  
  
**How do you control how long data is cached?**  
The Cache object’s Add and Insert method parameters allow you to control how long an item is stored in the server’s memory. In practice, these parameter settings provide only indirect control of how long data remains in memory. If your server runs low on available memory, ASP.NET recovers as much memory as possible from expired cache items. If that’s not enough, ASP.NET will unload unexpired items from the cache based on their priority and when they were last accessed.  
  
**What is CacheItemPriority enumeration used for?**  
CacheItemPriority enumeration is used to set the relative importance of cached items. CacheItemPriority.NotRemoveable has the highest priority and CacheItemPriority.Low has the lowest priority.  
  
**Which is the only "event” provided by Cache object?**  
CacheItemRemoved "event” is the only "event” provided by Cache object.   
  
**How do you update the Cache object when data changes?**  
Items stored in the cache are often copies of data that is stored and maintained elsewhere, such as records in a database. Use the Add and Insert methods’ dependency parameter to establish a relationship between a cached data item and an external source, such as a file, a folder, or a group of files.  
  
The dependency parameter accepts a CacheDependency object, which in turn identifies the file, folder, or set of files to watch for changes. ASP.NET checks the time stamp of the items in the CacheDependency object, if one of those time stamps is later than the DateTime entered for the cached item, ASP.NET unloads that item from the cache.

**Authentication and Authorization**

##### Define Authentication and Authorization.

Authentication is the process of verifying user's identity. Authorization is the process of granting privilege to authenticated user.

The user is validated using authenticated process and then the authorization process identifies if the user has access to a given resource. In ASP.NET, you can authenticate user in code or allow the user to be authenticated by other party such as MS Passport. You have two layer of authentication in ASP.NET i.e. IIS layer and ASP.net authentication process layer. IIS performs authentication if it is configured to do so. By default, IIS allows anonymous access which means all the users are authenticated. All the requests pass through IIS layer and then to ASP.NET authentication process. If any user requests IIS layer for anonymous access, the user is treated as authenticated and pass to ASP.NET process. ASP.NET checks if impersonation is enabled in the web configuaration file i.e. web.config file. If impersonation is enabled, ASP.net acts as though it were the authenticated user otherwise it process with its own configured account.   
To enable the application to authenticate users,   
you need to add <deny users = "?"> element in the authorization section of Web.config.

##### [What is the authentication mode available in ASP.NET?](http://www.careerride.com/ASP-authentication-mode.aspx)

ASP.NET supports three authentication modes through the System.Web.Security namespace.

Windows Authentication  
The windows authentication authenticates users based on their windows accounts. In short, it uses windows network security. It uses IIS to perform authentication.

Passport authentication  
The Passport authentication uses Microsoft's passport service to authenticate users. The new user is directed to the Microsoft site where he can register his identity. This facilitates user to access multiple sites using single user name and password. You need to install the Passport SDK to enable the Passport classes in the System.Web.Security namespace.

Form authentication  
The Form authentication collects user's credential and lets the application use own logic to authenticate users. The collected user's credential is validated using the list maintained by the application. The application maintains its own user list either using <credential> element in the web.config file or using database. The advantage of using form authentication is that the users don't need to be the member of windows network to have access to the application.

##### [How do you set authentication mode in the ASP.NET application?](http://www.careerride.com/ASP-how-to-authentication-mode.aspx)

You can set authentication mode using web.config file.  
<authentication mode="windows">  
<authentication mode="passport">  
<authentication mode="forms">

##### [List out the difference between windows authentication and form authentication.](http://www.careerride.com/ASP-difference-windows-form-authentication.aspx)

Windows authentication uses windows account whereas form authentication maintains its own user list. Windows authentication is best suited for the application which is meant for a corporate users whereas form authentication is preferable for the applications which have diversified users from several places.  
User lists for windows authentication are found in <authorization> element whereas in case of form authentication, lists are there in <credential> element of web.config file.

#### What is the difference between login controls and Forms authentication?

* Forms authentication can be easily implemented using login controls without writing any code.
* Login control performs functions like prompting for user credentials, validating them and issuing authentication just as the FormsAuthentication class.
* However, all that’s needs to be dne is to drag and drop the use control from the tool box to have these checks performed implicitly.
* The FormsAuthentication class is used in the background for the authentication ticket and ASP.NET membership is used to validate the user credentials.

##### *ASP.NET - Login controls and Forms authentication - June 04, 2009 at 15:00 PM by Shuchi Gauri*

#### What is the difference between login controls and Forms authentication?

Login control provides form authentication. If we implement for authentication through form authentication then we do it through code. On the other hand, login control allows the easy implementation on the basis of form authentication without writing much of code. Underneath the control, the class used for login control is also FormAuthentication class. So instead of creating your own set of user credential validations and issuing of authentication ticket, it is simpler to use a normal login control.

##### [How do you impersonate the authenticated user in ASP.NET?](http://www.careerride.com/ASP-how-to-impersonate-authenticated-user.aspx)

Impersonation means delegating one user identity to another user. In ASP.NET, the anonymous users impersonate the ASPNET user account by default. You can use <identity> element of web.config file to impersonate user. E.g. <identity impersonate="true"/>

#### What is impersonation in ASP.NET?

Impersonation is when a user accesses a resource without revealing his identity.

The two types of accounts that are set up using IIS make the task of being identifiable very difficult. These are IUSR\_machinename and IWAM\_machinename and they get added on a web server automatically. When IIS receives a request for a web page or other resource that has permission for anonymous access, IIS treats the IUSR\_machinename/ IWAM\_machinename account (depending upon the type of the resource) as the user's account, to access the resources. This obviates the need to authenticate a user.

**Navigation methods**

#### What are the various navigation methods?

ASP.NET supports following ways to navigate between pages in your application.

### *Hyperlink control*

This is server control used for navigation to another page specified in the NavigateURL property. Hyperlink control doesn’t expose any server side event.

### *Response.Redirect method*

This method is used to navigate to another page from code. You can use this method to navigate from a Linkbutton or ImageButton control.  
For example

Private Sub LinkButton1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles LinkButton1.Click  
               Response.Redirect("Page2.aspx")  
End Sub

### *Server.Transfer method*

This method can be used only with .aspx file. It allows to retain some information between the requests when its preserveForm argument is set to true.

### *Server.Execute method*

Like Server.Transfer, this method is also used with .aspx file only. This method enables new page execution while still displaying the current web form.

### *Window.Open method*

Display a page in a new browser window on the client.

#### Explain the difference between Server.Transfer and Response.Redirect.

Redirect and Transfer both cause a new page to be processed. The difference lies in the way the interaction between the client and the server occurs.

Response.Redirect messages the client browser asking it to request for another page.   
e.g. if a browser is on page A which has a Response.Redirect, then it asked to request for another page B by the server. When the client browser requests for it, then it is provided with the requested page B.

With Server.Transfer, the browser is not requested to ask for another page. Instead it is directly provided with the page B. In this scenario, the browser address bar continues to show the address of the previous URL.

common misconception is the difference between Server.Transfer and Response.Redirect in ASP.NET applications. Redirect and Transfer both cause a new page to be processed, but the interaction between the client (web browser) and server (ASP.NET) is different in each situation.   
  
Redirect: A redirect is just a suggestion – it’s like saying to the client “Hey, you might want to look at this”. All you tell the client is the new URL to look at, and if they comply, they do a second request for the new URL.   
  
If you want to pass state from the source page to the new page, you have to pass it either on the URL (such as a database key, or message string), or you can store it in the Session object (caveat: there may be more than one browser window, and they’ll all use the same session object).   
  
e.g. Redirect to the new.aspx page, passing an ID on the query string. "true" stops processing the current page:

| Response.Redirect("new.aspx?id=34", true); |
| --- |

Transfer: A transfer happens without the client knowing – it’s the equivalent of a client requesting one page, but being given another. As far as the client knows, they are still visiting the original URL.   
  
Sharing state between pages is much easier using Server.Transfer – you can put values into the Context.Items dictionary, which is similar to Session and Application, except that it lasts only for the current request. (search for HttpContext in MSDN). The page receiving postback can process data, store values in the Context, and then Transfer to a page that uses the values.   
  
e.g. Store a message in the context dictionary, and transfer to the default.aspx page (which can then display the message):

| Context.Items["Message"] = "Your password was changed successfully";  Server.Transfer("default.aspx"); |
| --- |

Caveats:

* Response.Redirect is more user-friendly, as the site visitor can bookmark the page that they are redirected to.
* Transferred pages appear to the client as a different url than they really are. This means that things like relative links / image paths may not work if you transfer to a page from a different directory.
* Server.Transfer has an optional parameter to pass the form data to the new page.
* Since the release version, this no longer works, because the Viewstate now has more security by default (The EnableViewStateMac defaults to true), so the new page isn’t able to access the form data. You can still access the values of the original page in the new page, by requesting the original handler:

| Page originalPage = (Page)Context.Handler;  TextBox textBox1 = (TextBox)originalPage.FindControl("textBox1"); |
| --- |

#### When should we use Server.Transfer and Response.Redirect?

**Response.Redirect** should be used when:

* we want to redirect the request to some plain HTML pages on our server or to some **other** web server
* we don't care about causing additional roundtrips to the server on each request
* we do not need to preserve Query String and Form Variables from the original request
* we want our users to be able to see the new redirected URL where he is redirected in his browser (and be able to bookmark it if its necessary)

**Server.Transfer** should be used when:

* we want to transfer current page request to another .aspx page on the same server
* we want to preserve server resources and avoid the unnecessary roundtrips to the server
* we want to preserve Query String and Form Variables (optionally)
* we don't need to show the real URL where we redirected the request in the users Web Browser

**What are different page navigation techniques in ASP.NET?**   
**Hyperlink control :** Navigate to another page.  
**Response.Redirect :** Navigate to another page from code. This is equivalent to clicking a hyperlink.  
**Server.Transfer :** End the current Web form and begin executing a new Web form. This method works only when navigating to a Web Forms page (.[aspx](http://venkataspinterview.blogspot.com/2008/07/interview-questions-on-aspnet-page.html)).  
**Server.Execute :** Begin executing a new Web form while still displaying the current Web form. The contents of both forms are combined. This method works only when navigating to a Web Forms page (.aspx).  
**Window.Open script method :** Display a page in a new browser window on the client.  
  
**What is the difference between Response.Redirect and Server.Transfer?  
1.** When we use Server.Transfer the redirection happens on the server where as when we use Response.Redirect the redirection happens from the browser.  
**2.** Server.Transfer is faster as there is no round trip involved while navigating from one webform to another webform. Response.Redirect is slower than Server.Transfer as there is round trip from the server to the client browser.  
**3.** Server.Transfer works only with .aspx files where as Response.Redirect works with .aspx and .Htm pages.  
**4.** Server.Transfer will work with files on the same web server. You can't use Server.Transfer to send the user to an external site where as Response.Redirect can do that.  
**5.** Server.Transfer does not update the URL in the browser. For example when you navigate from WebForm1.aspx to WebForm2.aspx using Server.Transfer the URL in the browser still shows WebForm1.aspx while you are actually looking at WebForm2.aspx. Response.Redirect updates the URL in the browser.  
  
**What is the use of Server.Execute method?**Server.Execute method is used to process a second Web form without leaving the first Web form. This technique lets you direct the results from a Web form to a region on the current page.  
  
**Is it possible to send a webform's QueryString, ViewState, and event procedure information to another webform?**   
Yes, we can use Server.Transfer or Server.Execute to send a webform's QueryString, ViewState, and event procedure information to another webform.  
  
For this to work you have to set the preserveForm argument to True.To be able to read one Web form’s ViewState from another, you must first set the EnableViewStateMac attribute in the Web form’s Page directive to False. By default, ASP.NET hashes ViewState information, and setting this attribute to False disables that hashing so that the information can be read on the subsequent Web form.

**Tracing**

#### What is tracing? Why is it required?

Tracing information can help you investigate errors or unwanted results while ASP.NET processes a page request.

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.

* [Visual Studio 2008](http://msdn.microsoft.com/en-us/library/bb386420(v=VS.90).aspx)

#### How to enable Tracing for an ASP.NET Application?

### To enable tracing for an application

1. Open your Web site's Web.config file. If no Web.config file exists, create a new file in the root folder and copy the following into it:

<?xml version="1.0"?>

<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">

<system.web>

</system.web>

</configuration>

1. Add a trace element as a child of the system.web element.
2. In the trace element, set the enabled attribute to true.
3. If you want trace information to appear at the end of the page that it is associated with, set the trace element's pageOutput attribute to true. If you want tracing information to be displayed only in the trace viewer, set the pageOutput attribute to false.

For example, the following application trace configuration collects trace information for up to 40 requests and allows browsers on computers other than the server of origin to display the trace viewer. Trace information is not displayed in individual pages.

<configuration>

<system.web>

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

</system.web>

</configuration>

#### When you enable tracing for the whole application in the Web.config file, trace information is gathered and processed for each page in that application. To override the application-wide settings, set the Trace attribute in that page's [@ Page](http://msdn.microsoft.com/en-us/library/ydy4x04a.aspx) directive to false. Any [Write](http://msdn.microsoft.com/en-us/library/system.web.tracecontext.write.aspx) or [Warn](http://msdn.microsoft.com/en-us/library/system.web.tracecontext.warn.aspx) statements that you include in a page's code are stored and returned to the trace viewer only.

#### How do you view trace information?

#### You can view trace information at the bottom of individual pages. Alternatively, you can use the trace viewer (Trace.axd) to view trace information that is collected and cached by ASP.NET when tracing is enabled.

#### How to enable Tracing for an ASP.NET Page?

You 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.

### To enable tracing for a page

1. Include an [@ Page](http://msdn.microsoft.com/en-us/library/ydy4x04a.aspx) directive at the top of your .aspx file.
2. Add a Trace attribute and set its value to true, as shown in the following example:

<%@ Page Trace="true" %>

| **Security Note** |
| --- |
| When tracing is enabled for a page, trace information is displayed in any browser requests that page. Tracing displays sensitive information, such as the values of server variables, and can therefore represent a security threat. Be sure to disable page tracing before porting your application to a production server. You can do this by setting the Trace attribute to false or by removing it. You can also configure tracing in the Web.config file by setting the enabled, localOnly, and pageOutput attributes of the [trace Element (ASP.NET Settings Schema)](http://msdn.microsoft.com/en-us/library/6915t83k.aspx). The Trace attribute in the [@ Page](http://msdn.microsoft.com/en-us/library/ydy4x04a.aspx) directive takes precedence over attributes set in the [trace](http://msdn.microsoft.com/en-us/library/6915t83k.aspx) element in the Web.config file. Therefore, even if you disable tracing in the Web.config file by setting the enabled attribute to false, the page might still show tracing information if the Trace attribute in its [@ Page](http://msdn.microsoft.com/en-us/library/ydy4x04a.aspx) directive is set to true. |

1. Optionally, include the TraceMode attribute to specify the order in which you want your trace messages to appear:
   * Set TraceMode to SortByTime to sort trace messages in the order in which they are processed.
   * Set TraceMode to SortByCategory to sort trace messages by the categories that you specified in the [TraceContext.Warn](http://msdn.microsoft.com/en-us/library/system.web.tracecontext.warn.aspx) and [TraceContext.Write](http://msdn.microsoft.com/en-us/library/system.web.tracecontext.write.aspx) method calls in your page or server control code.

The following example shows how to enable tracing in a page and to sort trace messages by category.

<%@ Page Language="VB" Trace="True" TraceMode="SortByCategory" %>

#### What would you do to if you want trace information to appear at the end of the page that it is associated with?

If you want trace information to appear at the end of the page that it is associated with, you can set the trace element's [PageOutput](http://msdn.microsoft.com/en-us/library/system.web.configuration.tracesection.pageoutput.aspx) attribute to true. If you enable application-level tracing, but you do not want trace information displayed for some pages, you can set the Trace attribute in those pages' [@ Page](http://msdn.microsoft.com/en-us/library/ydy4x04a.aspx) directive to false. For more information about how to configure an ASP.NET application, see [ASP.NET Configuration Overview](http://msdn.microsoft.com/en-us/library/ms178683.aspx).

By default, application-level tracing can be viewed only on the local Web server computer. To make application-level trace information visible from remote computers, you can set the trace element's [LocalOnly](http://msdn.microsoft.com/en-us/library/system.web.configuration.tracesection.localonly.aspx) attribute to false.

**Validation control**

#### Define Validation Control in ASP.NET.

The validation control is used to implement page level validity of data entered in the server controls. This check is done before the page is posted back to the server thus avoid round trip to the server. If data does not pass validation, it will display an error message to the user.

#### What are the validation types supported by ASP.NET?

Following are the validation types in ASP.NET:

* The RequiredFieldValidator control forces the input control a required field.
* The CompareValidator control is used to compare values of the two input fields.
* The CustomValidator control allows you to write a method to handle the validation of the value entered.
* The RangeValidator control is used to check that the user enters an input value that falls between two values. It is possible to check ranges within numbers, dates, and characters.
* The RegularExpressionValidator control ensures an input value matches a specified pattern.
* The ValidationSummary control is used to display a summary of all validation errors occurred in a Web page.

#### Describe the steps to use Validation Control.

* Draw a validation control on a Web form
* Set the ControlToValidate property to the control you want to validate.
* Specify the ControlToCompare property with CompareValidator control.
* Specify error message to the validation control’s ErrorMessage property.
* Specify error message to the validation control’s Text property if you want to display a message other than the message in the ErrorMessage property.
* Draw a ValidationSummary control on the Web form to display the error messages from the validation controls in one place.

#### How many types of validation controls are provided by ASP.NET? Explain them.

5 Types of validation control exists:

* **RequiredFieldValidator:** To enforce a user to fill a particular field before submitting the form. This is done when the field is a mandatory input .e.g.: username & password
* **RangeValidator:** To validate if the user input fits within a range of lower and upper limits.
* **CompareValidator:** This is to compare the user input against other existing value.
* **RegularExpressionValidator:** This is to ensure that the use input is as per the expected pattern. e.g.: Phone number, emails
* **CustomValidator:** Custom validator allows developers to create their own validators based on their logics. This could be a mix of other validators.

**Exception handling technique**

##### [Define Exception handling in ASP.NET.](http://www.careerride.com/ASP-exception-handling.aspx)

Exceptions or errors are unusual occurrences that happen within the logic of an application. The CLR has provided structured way to deal with exceptions using Try/Catch block. ASP.NET also supports exception handling through server events such as Page\_Error and Application\_Error events.

##### Exception handling correct unusual occurrences and prevent application from getting terminated. You can use Try(try) block and Error event procedures to handle exceptions.

##### [What are the ways of handling exceptions in ASP.NET?](http://www.careerride.com/ASP-ways-of-handling-exceptions.aspx)

There are three ways to handle exceptions in ASP.NET  
Try/catch/finally block.  
Using Events like Page\_Error and Application\_Error  
Using Custom error page

##### [Explain Try/catch block of exception handling.](http://www.careerride.com/ASP-try-catch-block-method.aspx)

You can enclose code in Try/Catch/Finally block. You can catch all exceptions in the catch block. The third part of this block is finally. It is executed irrespective of the fact that an exception has been raised.

##### [Define Error Events.](http://www.careerride.com/ASP-error-events-method.aspx)

ASP.NET supports events that occur when any unhandled exception occurs in an application. These events are called as Error Events.  
  
ASP.NET provides two such events to handle exceptions

* Page\_Error : This is page event and is raised when any unhandled exception occur in the page.
* Application\_Error: This is application event and is raised for all unhandled exceptions in the ASP.NET application and is implemented in global.asax

The Error events have two methods to handle the exception:

* GetLastError: Returns the last exception that occurred on the server.
* ClearError: This method clear error and thus stop the error to trigger subsequent error event.

##### Define Custom Error Pages.

##### Why is exception handling important for an application?

Exception handling is used to prevent application from being stuck due to unusual occurrences. If the exceptions are handled properly, the application will never get terminated abruptly.

##### When can you use tracing with exception handling?

You can use tracing with exception handling to log unanticipated exception to the trace log. The log file can be used to diagnose unanticipated problems and thus can be corrected.

**Master Pages**

#### What is Master Page in ASP.NET?

A Master page offers a template for one or more web forms. It defines placeholders for the content, which can be overridden by the content pages. The content pages contains only content. When users request the content page, ASP.NET merges the layout of the master page with the content of the content page and produce output.

**What are Master Pages in ASP.NET? or What is a Master Page?**  
ASP.NET master pages allow you to create a consistent layout for the pages in your application. A single master page defines the look and feel and standard behavior that you want for all of the pages (or a group of pages) in your application. You can then create individual content pages that contain the content you want to display. When users request the content pages, they merge with the master page to produce output that combines the layout of the master page with the content from the content page.

#### Advantages of using Master Page in ASP.NET

Master pages enables consistent and standardized layout to the website.   
You can make layout changes of the site in the master page instead of making change in the pages.   
It is very easy to implement.

#### Define Multiple Master Page.

In ASP.NET, you can have multiple master pages each for a different purpose. You can provide users several layout options using Multiple Master Page. You can define Master Page at multiple places in the web application.   
  
You can specify page-level using the @Page dierective.  
You can specify using the Web.config.   
  
Remember that the definition closest to the user wins that means page-level definition supersedes site-level definition.

**What are the 2 important parts of a master page?**  
The following are the 2 important parts of a master page  
1. The Master Page itself  
2. One or more Content Pages  
  
**Can Master Pages be nested?**  
Yes, Master Pages be nested.  
  
**What is the file extension for a Master Page?**  
.master  
  
**How do you identify a Master Page?**  
The master page is identified by a special @ Master directive that replaces the @ Page directive that is used for ordinary .aspx pages.  
  
**Can a Master Page have more than one ContentPlaceHolder?**  
Yes, a Master Page can have more than one ContentPlaceHolder

#### How do you create Master Page?

Steps to create Master Page  
  
Right-click your web project in the Solution Explorer window  
Select Add New Item.  
Select the Master Page item type from the Add New Item dialog.

Following is an example of Master Page and a Content Page.

<%@ Master %>  
<html>  
<body>  
<Title>Title of the Pages</Title>  
  
<asp:ContentPlaceHolder id="Content1" runat="server">  
</asp:ContentPlaceHolder>  
</body>  
</html>

This is simple HTML page that serves template for other pages.  
The @ Master directive at the top of the page defines that the page is Master Page.   
A placeholder tag <asp:ContentPlaceHolder> is overridden by the content page to place the content.  
You can have many placeholders tag each of them with different id.

Content Page Example   
%@ Page MasterPageFile="master1.master" %>  
<asp:Content ContentPlaceHolderId="Content1" runat="server">  
<h1>ASP Interview Questions</h1>  
</asp:Content>  
  
When you request this page, ASP.NET merges layout specified in the Master Page and content of content page to produce page result.

**Application file**

#### What is the Global.asax file?

The Global.asax file, also known as the ASP.NET application 

file, is an optional file that contains code for responding to application-level events raised by ASP.NET. The Global.asax file is parsed and dynamically compiled by ASP.NET into a .NET Framework class the first time any resource or URL within its application namespace is activated or requested. Whenever the application is requested for the first time, the Global.asax file is parsed and compiled to a class that extends from the HttpApplication base class. When the Global.asax file changes, the framework reboots the application and the Application\_OnStart event is fired once again when the next request comes in. Note that the Global.asax file does not need recompilation if no changes have been made to it. *There can be only one Global.asax file per application and it should be located in the application's root directory only.*

This file exposes the application and session level events in ASP.NET and provides a gateway



to all the application and the session level events in ASP.NET. This file can be used to implement the important application and session level events such as Application\_Start, Application\_End, Session\_Start, Session\_End, etc.

***Why the Global.asax file is used in ASP.NET?***

The Global.asax file is used in ASP.NET to specify the global objects and the application and the session level events that would be used by the application. It contains all the application and session level events that are used by the application.

#### What are the events in global.asax file?

| **Events in the Global.asax file** |
| --- |
| The following are some of the important events in the Global.asax file.           Application\_Init           Application\_Start           Session\_Start           Application\_BeginRequest           Application\_EndRequest           Application\_AuthenticateRequest           Application\_Error           Session\_End           Application\_End  The purpose of these event handlers is discussed in this section below.  **Application\_Init**  The Application\_Init event is fired when an application initializes the first time.  **Application\_Start**  The Application\_Start event is fired the first time when an application starts.  **Session\_Start**  The Session\_Start event is fired the first time when a user’s session is started. This typically contains for session initialization logic code.  **Application\_BeginRequest**  The Application\_BeginRequest event is fired each time a new request comes in.  **Application\_EndRequest**  The Application\_EndRequest event is fired when the application terminates.  **Application\_AuthenticateRequest**  The Application\_AuthenticateRequest event indicates that a request is ready to be authenticated. If you are using Forms Authentication, this event can be used to check for the user's roles and rights.  **Application\_Error**  The Application\_Error event is fired when an unhandled error occurs within the application.  **Session\_End**  The Session\_End Event is fired whenever a single user Session ends or times out.  **Application\_End**  The Application\_End event is last event of its kind that is fired when the application ends or times out. It typically contains application cleanup logic. |
| **Show how to use the Global.asax file** |
| The following code sample shows how we can use the events in the Global.asax file to store values in the Application state and then retrieve them when necessary. The program stores an Application and a Session counter in the Application state to determine the number of times the application has been accessed and the number of users currently accessing the application.  **Listing 1**  using System;  using System.ComponentModel;  using System.Web;  using System.Web.SessionState;  public class Global : HttpApplication  {   protected void Application\_Start(Object sender, EventArgs e)   {      Application["appCtr"] = 0;      Application["noOfUsers"] = 0;   }   protected void Application\_BeginRequest(Object sender, EventArgs e)   {     Application.Lock();     Application["appCtr"] = (int) Application["appCtr"] + 1;     Application.UnLock();   }     protected void Session\_Start(Object sender, EventArgs e)   {    Application.Lock();    Application["noOfUsers"] = (int) Application["noOfUsers"] + 1;    Application.UnLock();   }   // Code for other handlers  }  After storing the values in the Application state, they can be retrieved using the statements given in the code sample below.  **Listing 2**  Response.Write("This application has been accessed "+Application["appCtr"] + " times");  Response.Write("There are "+ Application["noOfUsers"] + " users accessing this application"); |

**Configuration File**

## What are configuration files?

The configuration files in ASP.NET (web.config and machine.config) are XML based and comprise of elements and their attributes. The Machine.config file contains configuration information that apply to all .NET applications for a specific version of the framework installed on the machine, whereas a Web.config file contains configuration settings that apply to a specific ASP.NET application or resource (typically a directory or a file)

## What is Web.Config File?

Web.config file, as it sounds like is a configuration file for the Asp .net web application. An Asp .net application has one web.config file which keeps the configurations required for the corresponding application. Web.config file is written in XML with specific tags having specific meanings.

## What is Machine.config File?

As web.config file is used to configure one asp .net web application, same way Machine.config file is used to configure the application according to a particular machine. That is, configuration done in machine.config file is affected on any application that runs on a particular machine. Usually, this file is not altered and only web.config is used which configuring applications.

**Where does the web.config and machine.config reside?**

The Web.config is an XML based configuration file for the entire application and resides in the application root. It provides the application wide settings for the entire application. The Machine.config located at the "C:\WINDOWS\Microsoft.NET\Framework\vx.x.xxxx\CONFIG" directory, is used to apply configuration settings for any applications that are running in the entire system.

**How many web.config and machine.config file?**

## There can be multiple config files in your application (one per each folder in your application) but only one machine.config file. Note that the settings specified in the Web.config file actually override the settings that are specified in the Machine.config file.

**Discuss the layout of web.config file.**

Typically, the layout of the web.config file is as follows.

**Listing 1**

<configuration>

<system.web>

<!-- Specify your Compilation, Custom Error, Authentication, Authorization, etc sections here -->

</system.web>

<appSettings>

<!-- Specify the Database connection string, File path, Server Name and other Custom Settings here -->

</appSettings >

</configuration>

## How to enable or disable session and/or view state?

You can use the <pages> section of the <system.web> section group to specify whether session and/or view state would be enabled or disabled for the web pages of the application. Refer to the code snippet below.

**Listing 2**

<configuration>

<system.web>

<pages enableSessionState="true" />

</system.web>

</configuration>

## How to specify the authentication mode applicable for your application and the authorization information for the authenticated users?

You can specify the authentication mode as shown below.

**Listing 3**

<authentication mode="Windows" />

You can specify the authorization information for users as shown below.

**Listing 4**

<authorization>

<allow roles="Users" />

<deny users="\*" />

</authorization>

## How do we change the encoding or the culture of your application?

The globalization section is useful when you want to change the encoding or the culture of your application. Globalization is such an extensive subject that an entire article could be dedicated to the matter. In short, this section allows you to define which character set the server should use to send data to the client (for instance UTF-8, which is the default), and which settings the server should use to interpret and displaying culturally specific strings, such as numbers and dates.

**Listing 6**

<globalization requestEncoding="utf-8" responseEncoding="utf-8" culture="en-GB" />

Encoding is done through the attributes requestEncoding and responseEncoding. The values should be equal in all one-server environments. In this example, the application culture is set to Dutch. If you don't supply a culture, the application will use the server's regional settings.

## How to specify the application wide error information?

You can specify the application wide error information usingthe <customErrors> section. The following is an example of how you can use this section to specify the application wide error information.

**Listing 7**

<customErrors mode="On">

<error statusCode="404" redirect="FileUnavailable.aspx"/>

</customErrors>

## How to specify the session state storage mode (InProc, OutProc, StateServer, SqlServer)?

You can use the <sessionState> section of the web.config file to specify the session state storage mode (InProc, OutProc, StateServer, SqlServer) that would be used. Refer to the code snippet below that illustrates how you can specify the session state storage mode in the <sessionState> section of the web.config file.

**Listing 9**

<sessionState mode="InProc" />

## Discuss the uses of <appSettings> section in web.config file?

The <appSettings> section is the most widely used of all the sections in the web.config file and it typically stores the following.

* Database Connection Strings
* Server Name
* File Path
* Custom Key - Value Settings

You can also specify an external configuration file in the web.config file as shown below.

**Listing 10**

<appSettings file="externalSettings.config"/>

The database connection string is specified in the <appSettings> section as shown below.

**Listing 11**

<add key="databaseConnectionString" value="server=(localhost);database=test;uid=sa;pwd=sa"/>

However, with ASP.NET 2.0, you can use the new <connectionStrings> section to store your connection strings. Refer to the code s nippet below.

**Listing 12**

<connectionStrings>

<add name ="Test"

connectionString ="server=(localhost);database=test;uid=sa;pwd=sa"/>

</connectionStrings>

You can specify custom application - wide settings using the <appSettings> section of the web.config file as shown below.

**Listing 13**

<appSettings>

<add key="key" value ="value"/>

</appSettings >

You can also add a new configuration section using the <configSections> section. This is illustrated below.

**Listing 14**

<configuration>

<configSections>

<section name="TestSection" type="Test.TestSettings" />

</configSections>

</configuration>

## What can be stored in Web.config file?

There are number of important settings that can be stored in the configuration file. Here are some of the most frequently used configurations, stored conveniently inside Web.config file..

1. Database connections
2. Session States
3. Error Handling
4. Security

## Database Connections:

The most important configuration data that can be stored inside the web.config file is the database connection string. Storing the connection string in the web.config file makes sense, since any modifications to the database configurations can be maintained at a single location. As otherwise we'll have to keep it either as a class level variable in all the associated source files or probably keep it in another class as a public static variable.

   But it this is stored in the Web.config file, it can be read and used anywhere in the program. This will certainly save us a lot of alteration in different files where we used the old connection.

Let’s see a small example of the connection string which is stored in the web.config file.

| <configuration>  <appSettings>  <add key="ConnectionString"  value="server=localhost;uid=sa;pwd=;database=DBPerson" />  </appSettings>  </configuration> |
| --- |

As you can see it is really simple to store the connection string in the web.config file. The connection string is referenced by a key which in this case is "ConnectionString". The value attribute of the configuration file denotes the information about the database. Here we can see that if has database name, userid and password. You can define more options if you want.

There is a very good website that deals with all sorts of connection strings. Its called [www.connectionstrings.com](http://www.connectionstrings.com) , in the website you will find the connection strings for most of the databases.

Let’s see how we access the connection string from our Asp .net web application.

| using System.Configuration;  string connectionString = (string )ConfigurationSettings.AppSettings["ConnectionString"]; |
| --- |

The small code snippet above is all that is needed to access the value stored inside the Web.config file.

## Session States:

   Session in Asp .net web application is very important. As we know that HTTP is a stateless protocol and we need session to keep the state alive. Asp .net stores the sessions in different ways. By default the session is stored in the asp .net process. You can always configure the application so that the session will be stored in one of the following ways.

**1) Session State Service**

   There are two main advantages of using the State Service. First the state service is not running in the same process as the asp .net application. So even if the asp .net application crashes the sessions will not be destroyed. Any advantage is sharing the state information across a Web garden (Multiple processors for the same computer).

Let’s see a small example of the Session State Service.

| <sessionState mode="StateServer" stateConnectionString="tcpip=127.0.0.1:55455" sqlConnectionString="data source=127.0.0.1;user id=sa;password='' cookieless="false" timeout="20"/> |
| --- |

The attributes are self explanatory but I will go over them.

**mode:** This can be StateServer or SqlServer. Since we are using StateServer we set the mode to StateServer.

**stateConnectionString:** connectionString that is used to locate the State Service.

**sqlConnectionString:** The connection String of the sql server database.

**cookieless:** Cookieless equal to false means that we will be using cookies to store the session on the client side.

**2) SQL Server**

The final choice to save the session information is using the Sql Server 2000 database. To use Sql Server for storing session state you need to do the following:

1) Run the InstallSqlState.sql script on the Microsoft SQL Server where you intend to store the session.

You web.config settings will look something like this:

| <sessionState mode = "SqlServer" stateConnectionString="tcpip=127.0.0.1:45565" sqlConnectionString="data source="SERVERNAME;user id=sa;password='' cookiesless="false" timeout="20"/> |
| --- |

SQL Server lets you share session state among the processors in a Web garden or the servers in a Web farm. Apart from that you also get additional space to store the session. And after that you can take various actions on the session stored.

The downside is SQL Server is slow as compared to storing session in the state in process. And also SQL Server cost too much for a small company.

### 3) InProc:

  This is another Session State. This one is mostly used for development purposes. The biggest advantage of using this approach is the applications will run faster when compared to other Session state types. But the disadvantage is Sessions are not stored when there is any problem that occurs with the application, when there is a small change in the files etc., Also there could be frequent loss of session data experienced..

## Error Handling:

Error handling is one of the most important part of any web application. Each error has to be caught and suitable action has to be taken to resolve that problem. Asp.net web.config file lets us configure, what to do when an error occurs in our application.

Check the following xml tag in the web.config file that deals with errors:

| <customErrors mode = "On">  <error statusCode = "404" redirect = "errorPage.aspx" />  </customErrors> |
| --- |

This tells the Asp.net to display custom errors from a remote client or a local client and to display a page named errorPage.aspx. Error "404" is "Page not found" error.

If custom error mode is turned "off" than you will see Asp.net default error message. This error messages are good for debugging purposes but should never be exposed to the users. The users should always be presented with friendly errors if any.

## Security:

The most critical aspect of any application is the security. Asp.net offers many different types of security method which can be used depending upon the condition and type of security you need.

**1) No Authentication:**

No Authentication means "No Authentication" :) , meaning that Asp.net will not implement any type of security.

**2) Windows Authentication:**

The Windows authentication allows us to use the windows user accounts. This provider uses IIS to perform the actual authentication, and then passes the authenticated identity to your code. If you like to see that what windows user is using the Asp.net application you can use:

**User.Identity.Name;**

This returns the *DOMAIN\UserName* of the current user of the local machine.

**3) Passport Authentication:**

Passport Authentication provider uses Microsoft's Passport service to authenticate users. You need to purchase this service in order to use it.

**4) Forms Authentication:**

Forms Authentication uses HTML forms to collect the user information and than it takes required actions on those HTML collected values.

In order to use Forms Authentication you must set the Anonymous Access checkbox checked. Now we need that whenever user tries to run the application he/she will be redirected to the login page.

| <authentication mode="Forms">  <forms loginUrl = "frmLogin.aspx" name="3345C" timeout="1"/>  </authentication>  <authorization>  <deny users="?" />  </authorization> |
| --- |

As you can see we set the Authentication mode to "Forms". The forms loginUrl is the first page being displayed when the application is run by any user.

The authorization tags has the deny users element which contains "?", this means that full access will be given to the authenticated users and none access will be given to the unauthenticated users. You can replace "?" with "\*" meaning that all access is given to all the users no matter what.

#### Why is the Machine.config file?

The Machine.config file controls issue like process recycling, number of request queue limits, and what interval to check if user is connected.

### What is AppSetting Section in “Web.Config” file?

Web.config file defines configuration for a webproject.

AppSetting section is used to set the user defined values. For e.g.: The ConnectionString which is used through out the project for database connection.

<configuration>  
<appSettings><BR><addkey="ConnectionString"value="server=xyz;pwd=www;database=testing" />  
</appSettings>

##### *AppSetting Section in ASP.NET - May 07, 2009 at 15:30 PM by Shuchi Gauri*

#### What is AppSetting Section in “Web.Config” file?

AppSetting section in the configuration file is a section that allows us to keep configurable and application wide settings (for e.g.: ConnectionString) that an application requires in order to perform the tasks properly. This helps in easy maintenance and deployment of the application.

**Web.confg:**<appsettings>  
       <add key="ConnectionString" value="(your connection string)" />   
</appsettings>

**Code behind:**   
string strConnection = ConfigurationSettings.AppSettings["ConnectionString"];

**If we remove web.config or machine.config from the application  
then, Is this application will works?**

If we remove the web.config file from the application it will work.  
  
unless it doesnt have the things like connectionstring etc.

**How to modify user settings in asp.net?**

by using appsettings tag in web.config

Globalization and Localization

##### [What is Globalization and Localization in ASP.NET?](http://www.careerride.com/ASP-globalization-and-localization.aspx)

Localization is the process of adapting a software application for a specific locale. Globalization is the process of identifying the localizable resources of the application. You can provide support for Localization and Globalization to the application using System.Globalization, System.Resources and System.Threading namespaces.  
The developer can define culture specific information using the System.Globalization namespace. The System.Resources namespace contains ResourceManager class that allows access to resources either from the main assembly or from the Satellite Assemblies. The System.Threading namespace supports for multithreaded programming.   
A web form has two culture values, Culture and UICulture. Culture value is used for date and number formatting and UICulture values are used to determine culture specific resources.   
You can set culture and UICulture values in the application as follows.   
  
Using <globalization> element of Web.Config.    
  
Using @Page directive in the Page.  
  
In Code-Behind Page e.g.  
Thread.CurrentThread.CurrentCulture = CultureInfo.CreateSpecificCulture ("en-GB");  
Thread.CurrentThread.CurrentUICulture=new CultureInfo("en-GB");

##### [What are the Globalization approaches possible in ASP.NET?](http://www.careerride.com/ASP-globalization-approaches.aspx)

You can follow many approaches to have globalized application.  
You can create separate web application for each culture.  
You can create an application that can detect the user’s culture and adjusts output at run time using format specifiers and other tools.  
You can store culture-dependent strings in resource files that are compiled into satellite assemblies.

##### Implementing ASP.NET Globalization.

Create resource files and compile them into a binary resource file.   
Create satellite assembly for each of the resource file for each culture.   
Store them in separate folders for easy access and replacement.   
Read the resources from the satellite assembly that is stored in different folders based on the locale and culture.

##### [Define Resource Files and Satellite Assemblies.](http://www.careerride.com/ASP-resource-files-and-satellite-assemblies.aspx)

**Resource Files:**A resource file contains non-executable data that are used by the application and deployed along with it. Bitmaps, Icons etc are the examples of resource files. In ASP.NET, resource files are used to make application to support multiple cultures. You can create resource files each of them correspond to specific locale or culture of the application. You can use resgen utility to compile resource file into an assembly. You can create a satellite assembly from a compiled resource file using the AL utility provided with Microsoft .NET SDK.

Advantages of resource files are as follows.  
It supports Globalization features in ASP.NET.  
You can have culture based information separate from the content and logic of the application.   
You can change resource content without effecting application's code.

Satellite Assemblies   
Satellite Assemblies are the special kinds of assemblies that exist as DLL and contain culture-specific resources in a binary format. They store compiled localized application resources. They can be created using the AL utility and can be deployed even after deployment of the application.   
Satellite Assemblies encapsulate resources into binary format and thus make resources lighter and consume lesser space on the disk.  
  
Note: Resource-only assemblies can contain any resource, such as images and text. Satellite assemblies contain only culture-specific resources.

#### Explain the use of localization and Globalization.

Users of different countries, use different languages and others settings like currency, and dates. Therefore, applications are needed to be configurable as per the required settings based on cultures, regions, countries.

Globalization: Is a way of writing the executables in a way that they are culture or language neutral.  
The culture and language details specific details should be kept configurable. The main part of Globalization is to find out the locale-sensitive resources that are needed to be kept configurable and to segregate these resources from the executable code.

**Localization: Is the task to customize the application. The main part of localization is the translation of resources found in globalization.**

#### Explain the concepts of CODE Page approach. What are the disadvantages of this approach?

Code Page was used before Unicode came into existence. It was a technique to represent characters in different languages. It was a different interpretation of ASCII set which means it kept 128 characters for English and rest 128 were specifically tailored for other languages.

**Disadvantages:**

a. Languages like Japanese and Chinese having more than 5000 characters were difficult to represent n 128 character set.

b. Clients also need to have a support for code page.

c. It‘s representation is done on the basis of operating system.

#### What are resource files and explain how do we generate resource files?

Resource files are files in XML format. They contain all the resources needed by an application. These files can be used to store string, bitmaps, icons, fonts.

Steps to generate a resource file:

a. Open the web page in the design view.  
b. Click Tools  
c. Select generate local resource  
d. .resx file generated in the solution explorer  
e. Type in the resources. The file contains the key and value pairs.   
f. Save the file.

#### What are Satellite assemblies and how to generate Satellite assemblies?

To support the feature of multiple languages, we need to create different modules that are customized on the basis of localization. These assemblies created on the basis of different modules are knows as satellite assemblies.

Steps to generate satellite assemblies:

a. Set the paths for resgen and al.exe:  
b. Create a .resources file.  
c. Create the satellite assembly.  
d. The assembly should have the naming convention for .NET to be able to search for it.   
e. Specify the settings for culture.  
f. Put the satellite assembly in the appropriate folder.  
g. Once the satellite assembly is created, physically copy it to the appropriate directory.  
h. Repeat the process for each language in which you are creating an assembly.

##### *ASP.NET - What are Satellite assemblies? - May 20, 2009 at 10:10 AM*

#### What is Satellite Assembly?

Satellite Assemblies are language-specific assemblies and are used for language-specific resources for an application. Each assembly of this kind has a separate language specific ID and is installed in a language-specific subdirectory for each language.

#### Define AL.EXE and RESGEN.EXE.

Al.exe: It embeds the resources into a satellite assembly. It takes the resources in .resources binary format.   
resgen.exe:The input for Al.exe is generally plain text or XML-based resource files i.e. in .resx format. resgen.exe is used to convert such resources to the .resource in binary format which makes then compatible with al.exe.

#### Explain the concepts of resource manager class.

ResourceManager class: It provides convenient access to resources that are culture-correct. The access is provided at run time.

This class provides resource withdrawal when a localized resource does not exist. It helps in resource serialization, and provides access to resources that are not packaged in your assembly. It is very much possible to derive classes from ResourceManager class and create our own custom resource solutions.

Deployment

**Define deployment.**

Deployment refers to the process of copying an asp.net web application from the development system to the server on which the application will be run. There are several way we can deploy our web application

**What are the different ways to deploy ASP.net application on server?**

We can deploy ASP.NET Application in 3 different ways

* xCopy Deployment
* Precompiled Deployment
* Web Setup Project

**How do you decide which deployment technique best suits you application need?**

The choice of best deployment alternative depends upon particular need of each application. Xcopy deployment is the most easiest, and it is often used during development to create copies of an application n different servers for testing purpose. For small application xcopy deployment may be the best choice.  
  
Precompiled deployment has several advantages over XCopy deployment. Eg. Precompiled deployment is always gives better performance for the first users of the site at the same time it is more secure as we don’t need to copy our source code files on to server. If our application deployed on one or few servers then precompiled deployment is usually best choice.  
  
When we are going to deploy our application on number of servers then creating a setup program is a very handy tool. Although creating this setup program is much tedious and involves considerable working, the deployment from this setup program becomes very easier.

**Discuss each deployment technique?**

## xCopy Deployment

To manually copy the files of an asp.net web site to a server. We can use the xcopy command from a command prompt. Then we can use IIS’s (Internet Information Serve management console t o create a virtual directory that’s mapped to the directory that you copied the web site to.  
  
It is easier to create a batch file for the xcopy command. Then after we can run that batch file at given time we make changes to the application and want to deploy the updated code.  
  
We can also perform xcopy deployment from visual studio by using copy website command.

To perform xcopy we use copy web site command. This command lets us to copy website to file system, local IIS, FTP or remote IIS website. At the same time we can copy all or selected files.  
  
*How to use this command*

1. In visual studio open the website to be deployed and choose the website copy web site command.
2. Then click the connect button to display an open website dialog box that lets to choose the destination to copy the web site to.
3. Click the arrow buttons to copy the files from the source web site to remote web site.

## Publish Web Site

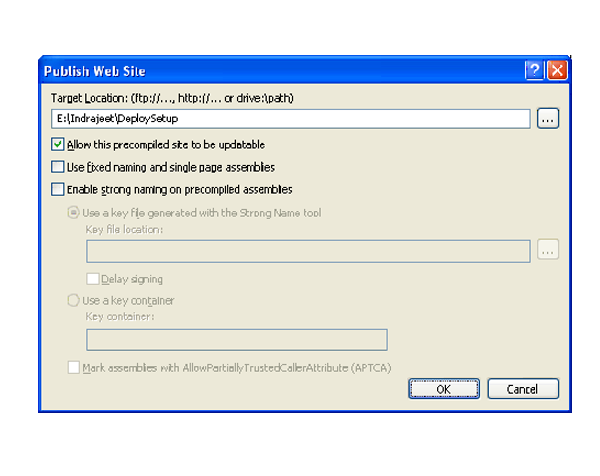
The publish web site command lets us to precompile an asp.net application and copy the precompiled assemblies to a target server. This is the easiest way to use the precompiled deployment feature.

* Deploys precompiled assemblies to the specific server
* Lets us to deploy the web site without source code files
* It is done either from publish web site command or from command prompt using the aspnet\_compiler command

**Advantages**

* Avoids delays caused by compiling web pages when they are first accessed by a user.
* Finds compile errors before the site is deployed.
* Can copy just executable files and not the source files to the server.

The build -> publish web site command compiles all of the files makes up an asp.net application, and then deploys the compiled assemblies to the location we specify.



**Publish Web Site**

If we check allow precompiled site to be updatable box, the source files are deployed along with the executable files. If we uncheck then source code files aren’t deployed.

**The syntax of aspnet\_compiler command**  
The aspnet\_compiler command is located in the asp.net framework directory,   
**c:\WINDOWS\Microsoft.NET\Framework\v2.0.50727**  
  
**aspnet\_compiler –v virtual-directory [-u] [-d] [-f] [target-directory]**  
  
**where**

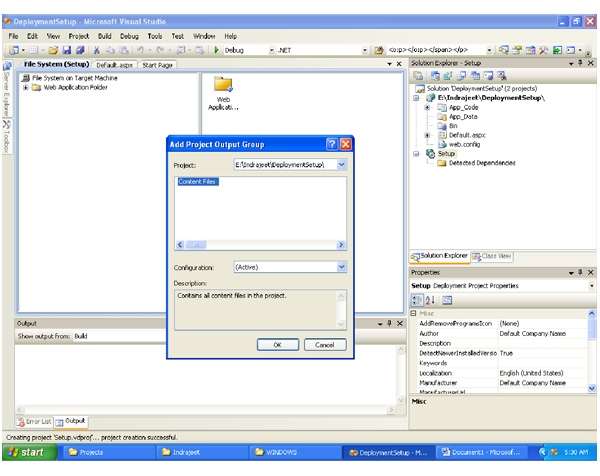
* -v name of virtual directory of existing web site
* -u precompiled website will be updated
* -d debug information is included in compiled assemblies
* -f overwrite target directory if exist

Eg.  
Precomplies existing website  
Aspnet\_compiler –v test d:\test  
  
In same place  
Aspnet\_complier –v test  
  
With debugging information  
Aspnet\_compiler –v test –u –d d:\test  
  
If we specify target directory the precompiled web site is stored in the specified directory else the website is precompiled in place

## Setup Deployment

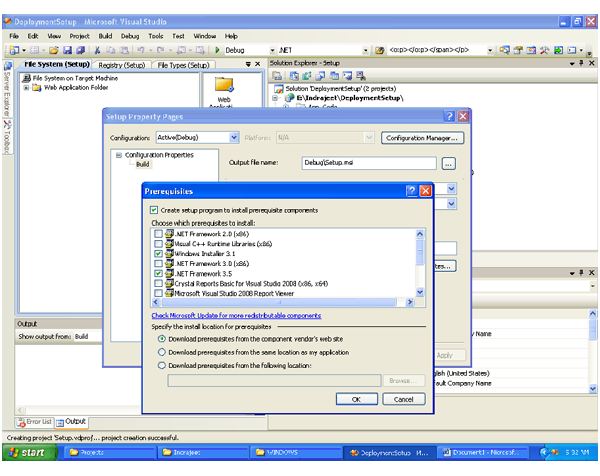
* It uses a web setup project to build a windows setup program used to deploy website onto server.
* Useful for deployment on multiple servers.
* Can be used to deploy precompiled assemblies and can be configured to include or omit the source code.
* The installed application can be removed by using add or remove programs dialog box from control panel

**How to create a setup**   
Choose the file -> add -> new project command to display the add new project dialog box. Then, choose setup and deployment. Select web setup project as template, enter a name for websetup project click ok.



**Add project to Out put Group**

In the solution explorer right click the web setup project and choose the add-> project output command to display the add project output group dialog box, then click ok to add the content files from you website to the web setup project   
  
Use the buttons that are displayed at the top of the solution explorer when web setup project is selected to access setup editors that lets us to customize various aspects of the web setup project.



**Add setup prerequisites**

Visual studio creates files named setup.exe and setup.msi. setup.exe is the file we will run to install the application and setup.msi contains all of the files to be installed.  
  
The setup.exe and setup.msi files are stored in the web setup project’s debug or release folder.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#### Explain how to pass a querystring from an .asp page to aspx page.

Consider the following URL:

http:// localhost/form.aspx?param1=career&param2=ride

This html addresses use QueryString property to pass values between pages.

From the URL above the information obtained is:

form.aspx: which is the destination page for your browser.  
Param1 is the first parameter, the value of which is set to career  
Param2 is the first parameter, the value of which is set to ride

The ‘?’ marks the beginning of the QueryString  
‘&’ is used as a separator between parameters.

private void formButtonSubmit\_Click(object sender, System.EventArgs e)  
{  
Response.Redirect("form.aspx?Param1=" +  
this.formTextfieldParam1.Text + "&Param2=" +  
this. formTextfieldParam2.Text);  
}

The above code is a submit button event handler and it sends the values of the query string to the second page.

The following code demonstrates how to retrieve these valus on the second page:  
private void Page\_Load(object sender, System.EventArgs e)  
{  
      this.form2TextField1.Text = Request.QueryString["Param1"];  
      this. form2TextField2.Text = Request.QueryString["Param2"];  
}   
  
You can also use the following method to retrieve the parameters in the string:  
  
for (int i =0;i < Request.QueryString.Count;i++)  
{  
        Response.Write(Request.QueryString[i]);  
}

##### *ASP.NET - How to pass a querystring from an .asp page to aspx page - June 04, 2009 at 15:00 PM by Shuchi Gauri*

From HTML in asp page: <a href="abc.aspx?qstring1=test">Test Query String</a>  
From server side code: <%response.redirect "webform1.aspx?id=11"%>

#### Difference between src and Code-Behind

With the ‘src’ attribute, the source code files are deployed and are compiled by the JIT as needed.   
Though the code is available to everyone with an access to the server (NOT anyone on the web), this method is preferred as it does away with the compilation of the DLLs.

‘CodeBehind’ attribute just has the VS.NET associate the code file with the aspx file. This is necessary since VS.NET automates the pre-compiling that is harder by hand.

Due to this the ‘Src’ attribute is done away with having only a DLL to be deployed enhancing the protection level even though it can be decompiled.

##### *ASP.NET - Difference between src and Code-Behind. - June 04, 2009 at 15:00 PM by Shuchi Gauri*

Src: is a way mention the name of the code-behind class to dynamically compile on the request for a page.

**Code-behind: is the logic written behind the UI design file. It specifies the name of the compiled file that contains the class. Code-behind attribute is only used for.Net.**

#### What is the difference between URL and URI?

A URL (Uniform Resource Locator) is the address of some resource on the Web. A resource is nothing but a page of a site. There are other type of resources than Web pages, but that's the easiest conceptually.

A URI is a unique identifier to usually a namespace.   
Though it looks like a URL but it doesn’t have to necessarily locate any resource on the web.

URI is a generic term. URL is a type of URI.

##### *ASP.NET - What is the difference between URL and URI? - June 04, 2009 at 15:00 PM by Shuchi Gauri*

**URI** - Uniform Resource Identifier: it’s a string and its responsibility is to identify a resource by meta-information. It gives information about only one resource.

**URL** - Uniform Resource Locator: identifies the resource on the net and tells it is obtainable using what protocols.

#### What is the Pre-Compilation feature of ASP.NET 2.0?

Previously, in ASP.NET, the pages and the code used to be compiled dynamically and then cached so as to make the requests to access the page extremely efficient. In ASP.NET 2.0, the pre-compilation feature is used with which an entire site is precompiled before it is made available to users.

 There is a pre-defined folder structure for enabling the pre-compilation feature:

* App\_Code: stores classes
* App\_Themes: stores CSS files, Images, etc.
* App\_Data: stores XML files, Text Files, etc.
* App\_GlobalResources: stores all the resources at global level E.g. resx files, etc
* App\_LocalResources: stores all the resources at local/Page level

##### *ASP.NET - What is the Pre-Compilation feature of ASP.NET 2.0? - June 04, 2009 at 15:00 PM by Shuchi Gauri*

It is a process where things that can be handled before compilation are prepared in order to reduce the deployment time, response time, increase safety. It’s main aim to boost performance.

It also helps in informing about the compilation failures.

During development, it allows you to make changes to the web pages and reuse it using the same web browser to validate the changes without compiling the entire website.

During deployment, it generates the entire website folder structure in the destination. All the static files are copied to the folder and bin directory would later on contain the compiled dll.

#### Explain the various modes of storing ASP.NET session.

Types of sessions:

* InProc: The default way to use sessions. InProc is the fastest way to store and access sessions.
* OutProc:   
  i. State server: 15% slower than InProc. Session is serialized and stored in aspnet\_state.exe process. Stateserver sessions can be stored on a separate machine too.   
  ii. SQL Server: 25% slower than InProc. Used when data is to be serialized and stored in SQL Server database.

##### *ASP.NET Session state management options- March 13, 2009 at 15:20 PM by Amit Satpute*

#### What are the different types of Session state management options available with ASP.NET?

**In-Process state management:**

* The In-Process type of Session state management stores the session in memory on the web server.
* In order to have a user always reconnect to the same web-server, a sticky server is needed.

**Out-of-Process state management:**

* Out-of-Process Session state management stores data in an external data source.
* The external data source may be a SQL Server or a State Server service.
* Out-of-Process state management requires the objects in a session to be serializable..

#### What are the benefits and limitations of using hidden fields?

**Advantages:**

a. Easy to implement  
b. Hidden fields are supported by all browsers   
c. Enables faster access of information because data is stored on client side

**Disadvantages:**

a. Not secure because the data is stored on Client side.   
b. Decreases page performance if too many hidden fields  
c. Only support single value.

#### What are the benefits and limitations of using Hidden Frames?

**Advantages:**

a. Hidden frames allow you to cache more than one data field.  
b. The ability to cache and access data items stored in different hidden forms.

**Disadvantages:**

a. Not supported by all browsers  
b. Stored data is not secured as it is stored on client's machine in the form of hidden fields

#### What are benefits and limitations of using Cookies?

**Advantages:**

a. They are simple to use.  
b. Light in size, thus occupy less memory.   
c. Stores server information on client side  
d. Data need not to be sent back to server.   
e. Cookies persist for much longer period of time than a session.

**Disadvantages:**

a. Since they are stored on the client side in plain text, they are not secure.   
b. Number of cookies that can be stored and their size is limited.   
c. They don't work if security level is set too high in browser.   
d. Some people might just disable cookies on their browsers.

#### What is QueryString? Benefits and limitations of using querystring.

Querystring is way to transfer information from one page to another through the URL. QueryString is attached to the URL with "?".

**Advantages:**

a. Supported by all the browsers  
b. No extra effort is needed to code.   
c. Easy to use.

**Disadvantages:**

**a. All the attributes and values are visible to the end user. Therefore, they are not secure.   
b. There is a limit to URL length of 255 characters.**

#### What are Volatile and Dead letter queues?

**Volatile Queues:** There are scenarios in the project when you want the message to deliver in proper time. The timely delivery of message is very more important and to ensure they are not lost is important too. Volatile queues are used for such purposes.

**Dead-Letter Queues:** Queues are needed so that client and server are not required to run at the same time when the message is sent. When a message lies inside a queue for a long time on the server and becomes invalid after some time period, then such messages are not to be sent to the client. All these messages are sent to dead-letter queue. Therefore, dead-letter queues are responsible for storing messages that fail timely delivery.

#### How to make an application offline in ASP.NET 2.0.

Microsoft's Internet Information Services web server software is used to make an application offline. The IIS is instructed to route all incoming requests for the web site to another URL automatically.

Steps for application configuration:

* In the properties of IIS manager, choose Virtual Directory tab and indicate that the content for this application should come from "A redirection to a URL."
* Then enter the same URL to redirect.

##### *How to make an application offline in ASP.NET 2.0 - June 04, 2009 at 15:00 PM by Shuchi Gauri*

Ways to take an asp.net application offline:

* Stop IIS server.
* Redirect your website to an ErrorPage.htm file which displays appropriate information/feedback to users requesting the website.
* In the web.config look for and set it to false. This will return a default ASP.Net 4040 status page to the requesting end users.  
  <httpRuntime enable="false" />
* Add to App\_Offline.htm page to the root of the website. To bring the application back up, simply delete this file.

#### What are script injection attacks?

Script injection attacks called Cross-site scripting (XSS) attacks exploit vulnerabilities in Web page validation by injecting client-side script code.

This code executes on the user’s browser after the browser downloads the script code from a trusted site and then the browser has no way of determining the legitimacy code.

##### *ASP.NET - What are script injection attacks? - June 04, 2009 at 15:00 PM by Shuchi Gauri*

Script injection attacks occur when an end user tries to fill in malicious code in the form or input fields of a form to access database or change it or destroy it. The malicious code tries to fool the application, that it was just another end user. The technique involves submitting contents wrapped in <script>, <object>, <applet>, <embed>, <frame>, <link> etc tags.

Request validation and validating the input provided by the end user are a solution to such attacks. One can use the following in the web.config for validating requests for all pages in the application.

<configuration>  
     <appSettings/>  
     <connectionStrings/>  
     <system.web>  
           <pages validateRequest="false"/>  
     </system.web>  
</configuration>

#### What is Authentication in ASP.NET?

Authentication is the process of verifying user’s details and find if the user is a valid user to the system or not. This process of authentication is needed to provide authority to the user. ASP.NET implements authentication through authentication providers. Each provider has OnAuthenticate event. It allows building a custom authorization scheme.

#### What is Authorization in ASP.NET?

Authorization is a process that takes place based on the authentication of the user. Once authenticated, based on user’s credentials, it is determined what rights des a user have. In ASP.NET, there are two ways to authorize access to a given resource:

a. File authorization   
b. URL authorization

#### Name the namespace for Web page in ASP.NET

System.Web.UI.Page

#### Describe the concepts of Monitor class, the interlocked class and the ReaderWriterLock class.

Access to objects by is controlled by the Monitor class. It grants a lock for an object to a single thread.  
Critical sections can be access restricted by using object locks.  
While a thread owns the lock for an object, no other thread can acquire that lock.   
Interlocked class provides atomic operations for variables that are shared by multiple threads.  
When two threads are executing concurrently on separate processors error protection is done.  
The methods of this class help protect against errors that can occur when the scheduler switches contexts while a thread is updating a variable that can be accessed by other threads.  
ReaderWriterLock synchronizes access to a resource. It allows either concurrent read access for multiple threads, or write access for a single thread.   
A ReaderWriterLock provides better throughput than a simple one lock a time like Monitor.

#### What are the methods fired during the page load? Explain each of them

Init() - when the page is instantiated.  
Load() - when the page is loaded into server memory.  
PreRender() - the brief moment before the page is displayed to the user as HTML  
Unload() - when page finishes loading.

#### Define Server-side and Client-side code in ASP.NET.

Server-side code runs on the server.  
Client-side code runs on the client's browser.

#### What tag do you use to add a hyperlink column to the DataGrid?

<asp:HyperLinkColumn>

#### Where does VS.NET store Web application projects?

Web application projects create a virtual folder for each project where all the files of the projects are stored. The virtual folder can be viewed in IIS and property of that folder determines where the files are physically stored.

#### Describe Web application’s life cycle.

A Web application starts with the first request for a resource.   
On request, Web forms are instantiated and processed in the server.   
Server sends its response to the client.   
Application ends once all client sessions end.

#### Steps to execute a stored procedure from Web Application.

Create a command object.   
Set the object’s CommandText property to the name of the stored procedure.   
Set the CommandType property to stored Procedure.   
Execute stored procedure using command object’s method.

#### Describe exception handling in ASP.NET.

Exception handling correct unusual occurrences and prevent application from getting terminated. You can use Try(try) block and Error event procedures to handle exceptions.

#### Describe use of error pages in ASP.NET.

Error pages are used when exceptions are outside the scope of the application and the application can’t respond directly to these exceptions. These types of exceptions are identified by HTTP response codes, which IIS can respond to by displaying custom error pages listed in your application’s Web.config file.

#### Explain tracing in ASP.NET.

Tracing records unusual events while an application is running. It helps in observing problems during testing and after deployment. An application can write a message in case of unusual events to the trace log using Trace class.

#### What is Secure Sockets Layer (SSL) security?

SSL protects data exchanged between a client and an ASP.NET application by encrypting the data before it is sent across the internet.

#### Explain how to distribute shared components as part of an installation program

Shared components should be included as a merge module within the setup project. Merge modules can’t be installed by themselves. They can be installed only as part of an application installation.

#### Define HTML and XML

HTML: It has predefined elements names '<h1>', '<b>' etc  
XML: You can create your own elements with syntax much stricter than HTML.

#### Explain the concept of Automatic Memory Management in ASP.NET.

The .NET framework has introduced a concept called Garbage collector. This mechanism keeps track of the allocated memory references and releases the memory when it is not in reference. Since it is automatic, it relieves the programmers from putting extra energy to manage memory.

#### What is Finalizer in .NET?

Finalizer in .NET are the methods that help in cleanup the code that is executed just before the object is garbage collected. The methods of Finalizer are Dispose and Finalize. When we call Dispose method it cleans managed as well as unmanaged resources and when we call Finalize method, it clears only the unmanged resources.

**User control and Custom control**

**How does a user controls differs from an ASP.NET Web page?**

A user controls differs from an ASP.NET Web page in these ways:

* The file name extension for the user control is .ascx.
* Instead of an [@ Page](http://msdn.microsoft.com/en-us/library/ydy4x04a.aspx) directive, the user control contains an [@ Control](http://msdn.microsoft.com/en-us/library/d19c0t4b.aspx) directive that defines configuration and other properties.
* User controls cannot run as stand-alone files. Instead, you must add them to ASP.NET pages, as you would any control.
* The user control does not have html, body, or form elements in it. These elements must be in the hosting page.

You can use the same HTML elements (except the html, body, or form elements) and Web controls on a user control that you do on an ASP.NET Web page. For example, if you are creating a user control to use as a toolbar, you can put a series of [Button](http://msdn.microsoft.com/en-us/library/system.web.ui.webcontrols.button.aspx) Web server controls onto the control and create event handlers for the buttons.

#### How can we create custom controls in ASP.NET?

Custom controls are user defined controls. They can be created by grouping existing controls, by deriving the control from System.Web.UI.WebControls.WebControl or by enhancing the functionality of any other custom control. Custom controls are complied into DLL’s and thus can be referenced by as any other web server control.

Basic steps to create a Custom control:

1. Create Control Library  
2. Write the appropriate code  
3. Compile the control library  
4. Copy to the DLL of the control library to the project where this control needs to be used  
5. The custom control can then be registered on the webpage as any user control through the @Register tag.

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

**Specify the best ways to store variables so that we can access them in various pages of ASP.NET application?**

**What are the XML files that are important in developing an ASP.NET application?**

**How many objects are there in ASP?**

Answer1  
8 objects, they are request,response, server,application,session,file, dictionary, textstream.   
  
Answer2  
There are 6 objects in ASP.net  
a) Server  
b) Session  
c) Application  
d) ObjectContext  
e) Response  
f) Request

**How many types of cookies are there?**

Answer1  
Two type of cookeies.  
a) single valued eg request.cookies(”UserName”).value=”Mahesh”  
b)Multivalued cookies. These are used in the way collections are used.  
e.g.  
request.cookies(”CookiName”)(”UserName”)=”Mahesh”  
request.cookies(”CookiName”)(”UserID”)=”ABC003?  
  
rember no value method in multivalued cooki   
  
Answer2  
There are two types of cookies:  
Session cookies  
Persistent cookies  
  
Answer3  
2 types, persistant and impersistant.

**Tell few steps for optimizing (for speed and resource) ASP page/application.**

Avoid mixing html code with asp code

If there is a calendar control to be included in each page of your application, and we do not intend to use the Microsoft-provided calendar control, how do you develop it? Do you copy and paste the code into each and very page of your application?

**Explain the differences between Server-side and Client-side code?**

Server side code is executed at the server and written in VB Code whereas client side code is executed at the client side. Mostly client side coding is done in Javascrit.  
  
The reason why we do client side validations is to save the to and fro time taken for the request to go to the server and the response to get back to the browser.

**What r the ASP.NET list controls and diff. between them?**

ASP.NET List controls ==> There are 3 . 1. DropDownList, 2. ListBox and 3.HTMLSelect.1. System.Web.UI.WebControls.DropDownListthis control renders a drop-down list in the page at runtime. Only the selected item is visible when the user is not interacting with the list, and the other items become visible when the user clicks the control to see the items that can be selected. Only one item can be selected in this control. This control must be inserted into a server side form (runat="server") applied.2. System.Web.UI.WebControls.ListBoxthis control renders a list of items within a scrolling box, multiple items can be selected in this control if required. This control must be inserted into a server side form (runat="server") applied.3. System.Web.UI.HTMLControls.HTMLSelectthis control can be used to render a drop-down list or a scrolling list of items. This control has less built in functionality when compared with controls above (It lacks many of the properties that can be used to influence the display style such as ForeColor and BackColor to name a couple) ,but it is still capable of performing most common uses of a list control. This control can be inserted anywhere, and does not require a server side form.

**How to delete images from the folder?**

string[] files = System.IO.Directory.GetFiles(Path of Images Folder);  
foreach (string file in files)  
{  
System.IO.File.Delete(file);  
}

**Can a .NET web application consume Java web service ?**

Yeah Offcourse.Actually Webservices are independent to language. it depends on WSDL and SOAP. so any one can develope the Webservices anddisclose the wsdl and users can cosume the webservices.wsdl and soap both are xml based.. and all languages having xml parsing capability and access to http protocol will be able to work with Webservices

**Distinguish the differences between client-side and server-side data validations in web pages.**

Client side validation and verification involves with javascript or vbscripts  
  
Client side validation has done before the client hit the server.  
  
But in server side validation validation occurs on server.

**How to connect usercontrols to database?**

using <SqlDataSource> control.

**How to create dynamic Gridview?**

Many times we have the requirement where we have to create columns dynamically.  
This article describes you about the dynamic loading of data using the DataTable as the datasource.  
  
  
Details of the Grid  
  
  
Let?s have a look at the code to understand better.  
  
  
  
Create a gridview in the page,  
  
Drag and drop the GridView on to the page  
Or  
  
Manually type GridView definition in the page.  
public partial class \_Default : System.Web.UI.Page  
  
{  
  
#region constants  
  
const string NAME = "NAME";  
  
const string ID = "ID";  
  
#endregion  
  
  
  
protected void Page\_Load(object sender, EventArgs e)  
  
{  
  
loadDynamicGrid();  
  
}  
  
  
  
private void loadDynamicGrid()  
  
{  
  
#region Code for preparing the DataTable  
  
  
  
//Create an instance of DataTable  
  
DataTable dt = new DataTable();  
  
  
  
//Create an ID column for adding to the Datatable  
  
DataColumn dcol = new DataColumn(ID ,typeof(System.Int32));  
  
dcol.AutoIncrement = true;  
  
dt.Columns.Add(dcol);  
  
  
  
//Create an ID column for adding to the Datatable  
  
dcol = new DataColumn(NAME, typeof(System.String));  
  
dt.Columns.Add(dcol);  
  
  
  
//Now add data for dynamic columns  
  
//As the first column is auto-increment, we do not have to add any thing.  
  
//Let's add some data to the second column.  
  
for (int nIndex = 0; nIndex < 10; nIndex++)  
  
{  
  
//Create a new row  
  
DataRow drow = dt.NewRow();  
  
  
  
//Initialize the row data.  
  
drow[NAME] = "Row-" + Convert.ToString((nIndex + 1));  
  
  
  
//Add the row to the datatable.  
  
dt.Rows.Add(drow);  
  
}  
  
#endregion  
  
  
  
//Iterate through the columns of the datatable to set the data bound field dynamically.  
  
foreach (DataColumn col in dt.Columns)  
  
{  
  
//Declare the bound field and allocate memory for the bound field.  
  
BoundField bfield = new BoundField();  
  
  
  
//Initalize the DataField value.  
  
bfield.DataField = col.ColumnName;  
  
  
  
//Initialize the HeaderText field value.  
  
bfield.HeaderText = col.ColumnName;  
  
  
  
//Add the newly created bound field to the GridView.  
  
GrdDynamic.Columns.Add(bfield);  
  
}  
  
  
  
//Initialize the DataSource  
  
GrdDynamic.DataSource = dt;  
  
  
  
//Bind the datatable with the GridView.  
  
GrdDynamic.DataBind();  
  
}  
  
}

**Which dll handles the request of .aspx page?**

When the Internet Information Service process (inetinfo.exe) receives an HTTP request, it uses the filename extension of the requested resource to determine which Internet Server Application Programming Interface (ISAPI) program to run to process the request. When the request is for an ASP.NET page (.aspx file), IIS passes the request to the ISAPI DLL capable of handling the request for ASP.NET pages, which is aspnet\_isapi.dll.

**Where would you use an iHTTPModule, and what are the limitations of any**

IHttpModule is used when u want to add u r own module or extra module so that when u request u r module also gets called.First and foremost httpmodules are used when u make request for a page.While processing the page some modules gets called.Some example modules are security module etc.If u want add a module to be used when processing request u add/implement ihhtp moduleok "Happy Programming"

**How do you create a permanent cookie?**

Setting a permanent cookie is similar to Session cookie, except give the cookie an expiration date too. It is very common that you don't specify any arbitrary expiration date, but instead expire the cookie relative to the current date, using the DateAdd() function.  
  
Response.Cookies("Name") = "myCookie"  
Response.Cookies("Name").Expires = DateAdd("m", 1, Now())

permanent cookies are available until a specified expiration date and are stored on the hard disk.so set 'expires' property any value greater than DateTime.MinVlue withrespect to the currect datetime.if u want the cookie which never expires set its expires property equal to DateTime.maxValue.

**What are client activated objects and server activated objects?**

1. Basically for a SAO, the lifetime of the object is controlled by server, whereas for a CAO, the lifetime is controlled by the client.Below are definitions from MSDN. Server activated objects includes Single Call & Singleton.Single Call objects service one and only one request coming in. Single Callobjects are useful in scenarios where the objects are required to do afinite amount of work. Single Call objects are usually not required tostore state information, and they cannot hold state information betweenmethod calls. However, Single Call objects can be configured in aload-balanced fashion.Singleton objects are those objects that service multiple clients and henceshare data by storing state information between client invocations. Theyare useful in cases in which data needs to be shared explicitly betweenclients and also in which the overhead of creating and maintaining objectsis substantial.Client-activated objects (CAO) are server-side objects that are activatedupon request from the client. This way of activating server objects is verysimilar to the classic COM coclass activation. When the client submits arequest for a server object using "new" operator, an activation requestmessage is sent to the remote application. The server then creates aninstance of the requested class and returns an ObjRef back to the clientapplication that invoked it. A proxy is then created on the client sideusing the ObjRef. The client's method calls will be executed on the proxy.Client-activated objects can store state information between method callsfor its specific client and not across different client objects. Eachinvocation of "new" returns a proxy to an independent instance of theserver type.

**What is IPostBack? How to use it?**

Gets a value indicating whether the page is being loaded in response to a client postback, or if it is being loaded and accessed for the first time.  
Property Value  
  
true if the page is being loaded in response to a client postback; otherwise, false.  
  
void Page\_Load() {  
if (!IsPostBack) {  
// Validate initially to force asterisks  
// to appear before the first roundtrip.  
Validate();  
}  
}

The question is about IpostBack, it might be IsPostBack or IPostBackevent handler.  
As they both are different, the answer provided above can't be assumed to be correct.

**Where would you use an iHTTPModule, and what are the limitations of any?**

IHttpModule is used when u want to add u r own module or extra module so that when u request u r module also gets called.First and foremost httpmodules are used when u make request for a page.While processing the page some modules gets called.Some example modules are security module etc.If u want add a module to be used when processing request u add/implement ihhtp moduleok "Happy Programming"

**How would you get ASP.NET running in Apache web servers - why would you even do this?**

The mod\_mono Apache module is used to run ASP.NET applications within the Apache (http://httpd.apache.org) web server. Mod\_mono is available from (http://www.mono-project.com/Downloads).XSP is a standalone web server written in C# that can be used to run your ASP.NET applications. XSP works under both the Mono and Microsoft runtimes and the code is available from(http://www.mono-project.com/Downloads).

**What is the transport protocol you use to call a Web service SOAP**

Web service use only three types of protocal. first HTTP-GET,HTTP-POST and SOAP.

**Name some of the Microsoft Application Blocks. Have you used any? Which ones?**

Exception Management Logging Data Access User Interface Caching Application Block for .NET Asynchronous Invocation Application Block for .NET Configuration Management Application Block for .NET (there are others) We use Exception and Data Access

**What is the standard you use to wrap up a call to a Web service**

We use Soap standard to wrap calls to web service.Soap stands for Simple Object Access Protocol

**Web service support  
  
a) Data set   
b) dataReader  
c) both of above  
d) none of above**

Ans : a) Data Set   
  
Web service support Data Set and not support data reader

**What is Viewstate?**

The web is a stateless medium - state is not maintained between client requests by default. Technologies must be utilized to provide some form of state management if this is what is required of your application, which will be the case for all but the simplest of web applications. ASP.NET provides several mechanisms to manage state in a more powerful and easier to utilize way than classic ASP.   
  
Page level state is information maintained when an element on the web form page causes a subsequent request to the server for the same page - referred to as 'postback'. This is appropriately called ViewState as the data involved is usually, though not necessarily, shown to the user directly within the page output.   
  
The Control.ViewState property is associated with each server control in your web form and provides a dictionary object for retaining values between such multiple requests for the same page. This is the method that the page uses to preserve page and control property values between round trips.   
  
When the page is processed, the current state of the page and controls is hashed into a string and saved in the page as a hidden field. When the page is posted back to the server, the page parses the view state string at page initialization and restores property information in the page.   
  
ViewState is enabled by default so if you view a web form page in your browser you will see a line similar to the following near the form definition in your rendered HTML:   
  
ViewState offers a substantial improvement over the two competing techniques for state management via the client: standard hidden fields and cookies, in that ViewState is not limited to the storage of simple values. You can use ViewState to store any object as long as it is serializable, and the standard VB.NET types are. Serialization is the process of storing an object's data and other information necessary to reconstruct the object later.   
  
There is a further complication: a type that either is serializable or has a TypeConverter defined for it can be persisted in ViewState. However, types that are only serializable are slower and generate a much larger ViewState than those that have a TypeConverter. The TypeConverter class provides a unified way of converting types of values to other types, as well as for accessing standard values and subproperties.   
  
ViewState is serialized using a limited object serialization format that is optimized for primitive types, and for String, ArrayList, and HashTable types. You may want to consider these issues if page performance concerns are key

View state data is maintained as a hidden field and automatically returned to the server with every postback.  
  
You can store the information in viewstate, that will be accessed during only in pageload and the next time its returned to the server

Viewstate is use to provide state managemet at page level.

**How do we get only edited/deleted/inserted records from a Dataset?**

Data set maintains state of each row. For new entry row state is Inserted, For modified row it is Modifies and for deleted row , the row state is deleted.

**Describe the difference between inline and code behind - which is best in a**

Inline code written along side the html in a page. Code-behind is code written in a separate file and referenced by the .aspx page.

Inline Code:- Inline Code written along side the HTML in a page.  
  
Code Behind:- Code Behind is simply a separate code file linked to your web form or .aspx page. Code is stored in separate file which is identified by a .cs suffix or .vb suffix.  
  
Code Behind is the best.  
Because:-  
Separation of the content from the code. It is practical to have a designer working on a markup while a programmer writes code.  
  
Better security also.

The best example of inline coding is classic asp which is too messy because both server side and client side code is written on the same page while on the code behind (asp.net), server side code is separated from the client side code.

**What property do you have to set to tell the grid which page to go to when using the Pager object?**

ans:  
  
itemCommand

DataGrid1.CurrentPageIndex=e.NewPageIndex

**In order to get assembly info whcih namespace we should import?**

System.reflection

**How to connect ms-access database in asp.net2005 with c#..pls send the coding?**

Check the below Sample File to acess MS Access on ASP .net...  
<html>  
<head>  
<title>My First ASP Page</title>  
</head>  
<body bgcolor="white" text="black">  
<%   
'Dimension variables  
Dim adoCon 'Holds the Database Connection Object  
Dim rsGuestbook 'Holds the recordset for the records in the database  
Dim strSQL 'Holds the SQL query to query the database   
  
'Create an ADO connection object  
Set adoCon = Server.CreateObject("ADODB.Connection")  
  
'Set an active connection to the Connection object using a DSN-less connection  
adoCon.Open "Provider=Microsoft.Jet.OLEDB.4.0; Data Source=" & Server.MapPath("guestbook.mdb")  
  
'Set an active connection to the Connection object using DSN connection  
adoCon.Open "DSN=guestbook"  
  
'Create an ADO recordset object  
Set rsGuestbook = Server.CreateObject("ADODB.Recordset")   
  
'Initialise the strSQL variable with an SQL statement to query the database  
strSQL = "SELECT tblComments.Name, tblComments.Comments FROM tblComments;"  
  
  
'Open the recordset with the SQL query   
rsGuestbook.Open strSQL, adoCon  
  
  
'Loop through the recordset   
Do While not rsGuestbook.EOF   
  
'Write the HTML to display the current record in the recordset   
Response.Write ("<br>")   
Response.Write (rsGuestbook("Name"))   
Response.Write ("<br>")   
Response.Write (rsGuestbook("Comments"))   
Response.Write ("<br>")   
  
'Move to the next record in the recordset   
rsGuestbook.MoveNext   
Loop  
  
  
'Reset server objects  
rsGuestbook.Close  
Set rsGuestbook = Nothing  
Set adoCon = Nothing  
%>  
  
</body>  
</html>

**What are webservices?**

Web services are a core technology provided by the .NET Framework. By using web services, companies can more easily integrate internal applications, but they can also access services exposed by other businesses. By combining web services exposed on the Internet with internally built services, companies can create a wide variety of value-added applications. For example, a company could unify banking, electronic bill payment, stock trading, and insurance services into a single, seamless financial management portal. Another possibility is the integration of inventory control, fulfillment mechanisms and purchase-order tracking into a comprehensive supply chain management system.

First Thing Web Services is not property of .Net Framework. It can be built on Java also.  
  
In simple way to define Webservice, is this "Calling a method through web which is actallycreated by other web application" For example to check credit card authenticity by Online shopping website through calling a method (called webService method) which is actually created by bank for authenticity purpose. How it works \* how it recongize all question have one answer is Web service is formatted/created in XML format.

webservices are the applications that runs on the webserver and communicate with other applications.it uses a series of protocols to respond to different requests

**What does WSDL stand for?**

WSDL stands for web service description language.

**In A Page I have gridview with Options of select and delete using hyperlink when i am selecting any one of then it has to open another page how can it**

Ans 1: You can have template column for select and delete instead of the databound column. In which you can mention the destination page where you need to navigate.  
  
Ans 2: Using RowDataBound event, you can add attribute to the select and delete hyperlink like:  
  
e.Row.Cells(CellPosition).Controls(0).Attributes.Add("OnClick","return fnJavascriptFunction()")  
  
e.Row.Cells(CellPosition).Controls(0).Attributes.Add("OnClick","return fnJavascriptFunction('"& If any argument &"')").

**What are the disadvantages of viewstate/what are the benefits**

Viewstate has lots of advantages and as well as disadvantages, so you need to weigh carefully before making the decision to use it. As view state doesnt require any server resources for its operation. It is passed to the client during every postback as an hidden element. Since it is added with every page, it adds few Kbytes to the page. This effects the loading of the page in the client. Other main problem with Viewstate is, since it is passed as plain text to the client. Anybody can tamper this value, because of this you shouldnt store any important data in the viewstate. View state is one of the most important features of ASP.NET, not so much because of its technical relevance, but more because it makes the magic of the Web Forms model possible. However, if used carelessly, view state can easily become a burden. Although ViewState is freely accessible in a hidden field called \_\_VIEWSTATE, the view state information is not clear text. By default, a machine-specific authentication code is calculated on the data and appended to the view state string. The resulting text is then Base64 encoded only, but not encrypted. In order to make the view state more secure, the ASP.NET @Page directive supports an attribute called EnableViewStateMac whose only purpose is detecting any possible attempt at corrupting original data.

**Should validation (did the user enter a real date) occur server-side or client-side? Why?**

Its not that validation has to be done only on the client side, depending on the situation it varies, if it is not required to fetch data from the database and validate, client side validation can be done.  
  
server side validation is done whenever it is required since it utilizes the server side resources

**How to get the cell value on which we click on Data Grid and how can we high light the clicked area of data grid with some color?  
i.e. for example in Datagrid1 emp details records(empid,empname,empsal...) are dispalyed if we click on one record how to get that empid,empname values and how to highlight that clicked cell with some colour?**

protected void Page\_Load(object sender, EventArgs e)  
{  
string con = "Database=DataBaseName;Server=serverName;UserId=yourId;password=yourpwd;";  
SqlConnection sCon = new SqlConnection(con);  
SqlCommand cmd = new SqlCommand();  
cmd.Connection = sCon;  
cmd.CommandType = CommandType.StoredProcedure;  
cmd.CommandText = "[EmpDetails\_select]";  
SqlDataAdapter da = new SqlDataAdapter(cmd);  
DataSet ds = new DataSet();  
da.Fill(ds);  
GridView1.DataSource = ds;  
GridView1.DataBind();  
  
}  
protected void GridView1\_SelectedIndexChanged(object sender, EventArgs e)  
{  
int iIndex = GridView1.SelectedIndex;  
GridView1.Rows[iIndex].BackColor = System.Drawing.Color.Pink;  
string EmpId = GridView1.Rows[iIndex].Cells[1].Text.Trim();  
Response.Write("EMPID:" +EmpId+ ", ");  
string EmpName = GridView1.Rows[iIndex].Cells[2].Text.Trim();  
Response.Write("EMPName:" + EmpName + "  
");  
  
}

**What types of data validation events are commonly seen in the client-side form validation?**

Required Field Validator

requiredfield,range,regular expresion,compare,coustom validators

**Can you edit data in the Repeater control?**

A Repeater control is a light weight control which can be used for simple reporting purposes. It supports basic event-handling like Init, Load, Unload etc., This also supports some basic formatting of data and can be presented to the user. A Repeater control offers limited level of data editing or selecting capabilities. For such editing and updates ASP .Net offers DataList and DataGrid controls.

A Repeater control is a light weight control which can be used for simple reporting purposes. It supports basic event-handling like Init, Load, Unload etc., This also supports some basic formatting of data and can be presented to the user. A Repeater control offers limited level of data editing or selecting capabilities. For such editing and updates ASP .Net offers DataList and DataGrid controls.

**What is the life cycle of an asp.net page?**

The Life Cycle represents all the Events and Methods that are called from Initializing the Page to Unloading the Page. Stages and corresponding events in the life cycle of the ASP.NET page cycle: Stage Events/Method====================== ===========================Page Initialization Page\_InitView State Loading LoadViewStatePostback data processing LoadPostDataPage Loading Page\_LoadPostBack Change Notification RaisePostDataChangedEventPostBack Event Handling RaisePostBackEventPage Pre Rendering Phase Page\_PreRenderView State Saving SaveViewStatePage Rendering Page\_RenderPage Unloading Page\_UnLoad

**Which two properties are on every validation control?**

ControlToValidate property and Text property

1) ErrorMessage  
2) ControlToValidate

**Which property on a Combo Box do you set with a column name, prior to setting the DataSource, to display data in the combo box?**

DisplayMember and ValueMember Properties need to be set.

DataTextField="Column Name"  
  
DataValueField must be set

**Which control would you use if you needed to make sure the values in two different controls matched?**

Compare Validator Control

**How to reduce the width of textbox in EditCommandColumn of DataGrid?**

convert textbox column into a template then it lets you change its width

**What are ASP.NET Web Forms? How is this technology different than what is available though ASP (1.0-3.0)?**

ASP .NET Web form is a tool used to design a web page using the drag and drop feature in VS .NET and is similar to the 'Form' tool in Visual Studio 5.0/6.0. It functions like the Form tag in classic ASP to get or post data from or to the server.  
  
The difference between a classic ASP page and an ASP web Form is as follows:-  
  
A classic ASP has the extension .asp.  
  
An ASP.NET web form has the extension .aspx  
  
-------------  
  
A classic ASP page can be created using a Text editor or programs like DreamWeaver.  
  
An ASP .NET web form is created in a project in the VS.NET environment.  
  
------------  
  
The source of a classic ASP is in VBScript or JavaScript and is visible to the end user.  
  
The source of an ASP.NET web form is written in the code behind file ex: .cs for C# or .vb for VB.net and is invisible to the end user.  
  
-------------  
  
An ASP.NET Web form is Event driven.

**Whenever I am inserting an item in the listbox i.e. list1.items.add (item), I want to add an link button (eg. Remove) for each of the listitem entered.  
When I will click on the link button, the corrosponding listitem will be removed.  
Please help me in this matter.  
Substitute of the listbox can be any other control.**

Here i wirte the coding for removing the selcted item from the listbox when we click a button.  
In case if this is not u r expectation then send a mail to me....  
  
  
Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click  
Dim n As Integer  
n = ListBox1.Items.Count  
MsgBox(n)  
ListBox1.Items.Remove(ListBox1.SelectedItem)  
End Sub

**Diff between Web User Control and Web Custom Control.**

Web user controls:1. Easier to create 2. Limited support for consumers who use a visual design tool 3. A separate copy of the control is required in each application 4. Cannot be added to the Toolbox in Visual Studio 5. Good for static layout Web Custom Controls:1. Harder to create2. Full visual design tool support for consumers3. Only a single copy of the control is required, in the global assembly cache4. Can be added to the Toolbox in Visual Studio5. Good for dynamic layout

**How to create a DB connection at one place/page so that we can use that connection for all pages/forms/windows.what r the steps ned to be performed.  
if question not clear,let me know.**

<?xml version="1.0" encoding="utf-8" ?>  
<!-- Web.Config Configuration File -->  
<configuration>  
<appSettings>  
<add key="ConnectionString"  
value="server=localhost;database=Northwind;uid=sa;password=secret;" />  
</appSettings>  
<system.web>  
<customErrors mode="Off"/>  
</system.web>  
</configuration>  
  
Imports System  
Imports System.Web  
Imports System.Web.UI  
Imports System.Web.UI.WebControls  
Imports System.Data  
Imports System.Data.SqlClient  
Imports System.Configuration  
  
Public Class ConnString : Inherits Page  
  
Protected dataGrid As DataGrid  
  
Protected Sub Page\_Load(ByVal Sender As Object, ByVal E As EventArgs)  
Dim sqlConn As SqlConnection  
Dim sqlCmd As SqlCommand  
Dim strConnection As String  
  
Try  
'Get connection string from Web.Config  
strConnection = ConfigurationSettings.AppSettings("ConnectionString")  
sqlConn = New SqlConnection(strConnection)  
sqlCmd = New SqlCommand("SELECT \* FROM Customers WHERE " \_  
& "(CompanyName LIKE 'A%') OR (CompanyName LIKE 'B%')", sqlConn)  
sqlConn.Open()  
dataGrid.DataSource = sqlCmd.ExecuteReader()  
dataGrid.DataBind()  
Catch ex As Exception  
Response.Write(ex.ToString & "<br>")  
Finally  
sqlConn.Close()  
End Try  
End Sub  
  
End Class

**What is event bubbling?**

Event Bubbling is nothing but events raised by child controls is handled by the parent control. Example: Suppose consider datagrid as parent control in which there are several child controls.There can be a column of link buttons right.Each link button has click event.Instead of writing event routine for each link button write one routine for parent which will handlde the click events of the child link button events.Parent can know which child actaully triggered the event.That thru arguments passed to event routine. " Happy programming"

The DataGrid control, as you likely know, can easily be configured to add a column of buttons. By adding a ButtonColumn, whenever one of the DataGrid's buttons in the column is clicked, the Web page posts back and the DataGrid's ItemCommand event fires. In fact, the same behavior can be noted if you manually add a Button or LinkButton control into a DataGrid TemplateColumn, as the ButtonColumn class simply adds a Button (or LinkButton) Web control to each row of the DataGrid.  
  
When the Button (or LinkButton) is clicked, its Command event is raised. But how does the DataGrid know when this event has been raised so that it can raise its ItemCommand event in response? The answer is through a process referred to as event bubbling. Event bubbling is the process of moving an event up the control hierarchy, from a control low in the hierarchy - such as a Button within the row of a DataGrid - and percolating it up to an ancestor control - such as the DataGrid. Once the ancestor control has learned of the event it can respond however it sees fit; in the DataGrid's case, the DataGrid "swallows" the Button's Command event (that is, it stops the bubbling) and raises its own ItemCommand event in response.  
  
In this article we'll look at how, precisely, event bubbling works in ASP.NET. Event bubbling is a technique that all server control developers should be aware of. Additionally, it can be used as a means to pass event information from a User Control to its parent page, as discussed in Handle Events from Web User Controls (although personally I find it simpler to just use the technique of having the User Control raise its own events through the standard event firing syntax as discussed at An Extensive Examination of User Controls). Read on to learn more!

**Which method do you use to redirect the user to another page without performing a round trip to the client?**

Server.transfer( without roundtrip)  
Response.redirect(with roundtrip)

**How to implement a dynamic datagrid with dropdown list control in it?**

we can have template column facility to implement datagrid with controls.   
In grid view we can have column   
as   
<asp:TemplateField>  
<ItemTemplate>  
<select id="ddl" name = "ddl" runat="Server" class="abc"></select>  
</ItemTemplate>  
</asp:TemplateField>

**What base class do all Web Forms inherit from?**

System.web.UI.Page class

**What method do you use to explicitly kill a user s session?**

session.abandon-to kill all   
  
session.remove-to spec item

**I want to create a project named 'To run sql create command through asp.net with graphical view' how to work for that?**

No answer available currently. Be the first one to reply to this question by submitting your answer from the form below.

**What tags do you need to add within the asp:datagrid tags to bind columns manually?**

<asp:datagrid id="DataGrid1" runat="server" CssClass="content" AutoGenerateColumns="False" AllowSorting="False"  
AllowPaging="True" PageSize="25">  
<AlternatingItemStyle CssClass="GridAlternateItem"></AlternatingItemStyle>  
<ItemStyle CssClass="GridItem"></ItemStyle>  
<HeaderStyle CssClass="GridHeader"></HeaderStyle>  
<Columns>  
<asp:BoundColumn DataField="formid" SortExpression="formid" HeaderText="Form Id" Visible="False"></asp:BoundColumn>  
<asp:BoundColumn DataField="accounttitle" SortExpression="accounttitle" HeaderText="Customer Name            "></asp:BoundColumn>  
<asp:BoundColumn DataField="CreatedOn" SortExpression="CreatedOn" HeaderText="Step-1 Date              "></asp:BoundColumn>  
<asp:BoundColumn DataField="email" SortExpression="email" HeaderText="Email Address   "></asp:BoundColumn>  
<asp:TemplateColumn>  
<ItemTemplate>  
<asp:Button id="btnEmail" Runat="server" Text="Email" CommandName="email"></asp:Button>  
</ItemTemplate>  
</asp:TemplateColumn>  
</Columns>  
<PagerStyle Font-Size="XX-Small" Font-Names="Arial Black" HorizontalAlign="Left" ForeColor="#000066"  
BackColor="White" CssClass="abc" Mode="NumericPages"></PagerStyle>  
</asp:datagrid>

**Can you give an example of when it would be appropriate to use a web service as opposed to a non-serviced .NET component**

If you are developing an intranet app and you want to display the current stock price of your company, its best to consume a webservice the provide that service. IOW, when you want to get 'outside' data throuh a corporate firewall  
  
If, say, you want to display latest news from HR deartment in your portal, better to have a .NET DLL that connect to corporate HR Database and get the info.

**What does the "EnableViewState" property do? Why would I want it on or off?**

In Dot Net State of Objects/Controls is stored in hidden field named \_ViewState this is done automatically by dot net   
life time of this control is life of page (i.e till we are doing PostBack to same page)   
through EnableViewState=true /false we can define whether we have to maintain values of control in hidden fields when page is posted back again and again

As we know Web page is stateless, and every time takes round trip to server once we click any control. when it back, for example a textbox dose not have Value what you entered in it before clicking on Button. So overcome this problem .net provide faciltiy called viewstate which conatin value in hidden format and show in control in process of round trip to server.  
  
ViewState is bydefalut True. If u make it false in control then control dose not save value in round trip.

**How does u get record no from 5 to 15 from a dataset of 100 records?**

// ds-> daata set  
  
//dr as datarow   
  
for i=5 to 15   
  
{  
  
dr=ds.table.row(i)  
  
}

1. **sDescribe the role of *inetinfo.exe, aspnet\_isapi.dll* and*aspnet\_wp.exe* in the page loading process**.  
   inetinfo.exe is theMicrosoft IIS server running, handling ASP.NET requests among other things.When an ASP.NET request is received (usually a file with .aspx extension), the ISAPI filter aspnet\_isapi.dll takes care of it by passing the request to the actual worker process aspnet\_wp.exe.
2. **What’s the difference between Response.Write() andResponse.Output.Write()?**Response.Output.Write() allows you to write formatted output.
3. **What methods are fired during the page load?**Init() - when the page is instantiated  
   Load() - when the page is loaded into server memory  
   PreRender() - the brief moment before the page is displayed to the user as HTML  
   Unload() - when page finishes loading.
4. **When during the page processing cycle is ViewState available?**  
   After the Init() and before the Page\_Load(), or OnLoad() for a control.
5. **What namespace does the Web page belong in the .NET Framework class hierarchy?**System.Web.UI.Page
6. **Where do you store the information about the user’s locale?**System.Web.UI.Page.Culture
7. **What’s the difference between Codebehind="MyCode.aspx.cs" andSrc="MyCode.aspx.cs"?**CodeBehind is relevant to Visual Studio.NET only.
8. **What’s a bubbled event?**When you have a complex control, like DataGrid, writing an event processing routine for each object (cell, button, row, etc.) is quite tedious. The controls can bubble up their eventhandlers, allowing the main DataGrid event handler to take care of its constituents.
9. **Suppose you want a certain ASP.NET function executed on MouseOver for a certain button.  Where do you add an event handler?**Add an OnMouseOver attribute to the button.  Example: btnSubmit.Attributes.Add("onmouseover","someClientCodeHere();");
10. **What data types do the RangeValidator control support?**Integer, String, and Date.
11. **Explain the differences between Server-side and Client-side code?**Server-side code executes on the server.  Client-side code executes in the client's browser.
12. **What type of code (server or client) is found in a Code-Behind class?**The answer is server-side code since code-behind is executed on the server.  However, during the code-behind's execution on the server, it can render client-side code such as JavaScript to be processed in the clients browser.  But just to be clear, code-behind executes on the server, thus making it server-side code.
13. **Should user input data validation occur server-side or client-side?  Why?**All user input data validation should occur on the server at a minimum.  Additionally, client-side validation can be performed where deemed appropriate and feasable to provide a richer, more responsive experience for the user.
14. **What is the difference between Server.Transfer and Response.Redirect?  Why would I choose one over the other?**Server.Transfer transfers page processing from one page directly to the next page without making a round-trip back to the client's browser.  This provides a faster response with a little less overhead on the server.  Server.Transfer does not update the clients url history list or current url.  Response.Redirect is used to redirect the user's browser to another page or site.  This performas a trip back to the client where the client's browser is redirected to the new page.  The user's browser history list is updated to reflect the new address.
15. **Can you explain the difference between an ADO.NET Dataset and an ADO Recordset?**Valid answers are:  
    ·  A DataSet can represent an entire relational database in memory, complete with tables, relations, and views.  
    ·  A DataSet is designed to work without any continuing connection to the original data source.  
    ·  Data in a DataSet is bulk-loaded, rather than being loaded on demand.  
    ·  There's no concept of cursor types in a DataSet.  
    ·  DataSets have no current record pointer You can use For Each loops to move through the data.  
    ·  You can store many edits in a DataSet, and write them to the original data source in a single operation.  
    ·  Though the DataSet is universal, other objects in ADO.NET come in different versions for different data sources.
16. **What is the Global.asax used for?**The Global.asax (including the Global.asax.cs file) is used to implement application and session level events.
17. **What are the Application\_Start and Session\_Start subroutines used for?**This is where you can set the specific variables for the Application and Session objects.
18. **Describe the difference between inline and code behind.**Inline code written along side the html in a page. Code-behind is code written in a separate file and referenced by the .aspx page.
19. **Explain what a diffgram is, and a good use for one?**The DiffGram is one of the two XML formats that you can use to render DataSet object contents to XML.  A good use is reading database data to an XML file to be sent to a Web Service.
20. **Whats MSIL, and why should my developers need an appreciation of it if at all?**MSIL is the Microsoft Intermediate Language. All .NET compatible languages will get converted to MSIL.  MSIL also allows the .NET Framework to JIT compile the assembly on the installed computer.
21. **Which method do you invoke on the DataAdapter control to load your generated dataset with data?**The Fill() method.
22. **Can you edit data in the Repeater control?**No, it just reads the information from its data source**.**
23. **Which template must you provide, in order to display data in a Repeater control?**ItemTemplate.
24. **How can you provide an alternating color scheme in a Repeater control?**Use the AlternatingItemTemplate**.**
25. **What property must you set, and what method must you call in your code, in order to bind the data from a data source to the Repeater control?**You must set the DataSource property and call the DataBind method.
26. **What base class do all Web Forms inherit from?**The Page class.
27. **Name two properties common in every validation control?**ControlToValidate property and Text property.
28. **Which property on a Combo Box do you set with a column name, prior to setting the DataSource, to display data in the combo box?**DataTextField property.

**Which control would you use if you needed to make sure the values in two different controls matched?**CompareValidator control. 