Q1 - What is AJAX?

A - **Ajax** stands for Asynchronous Javascript & XML. It is a web technology through which a postback from a client (browser) to the server goes partially, which means that instead of a complete postback, a partial postback is triggered by the Javascript XmlHttpRequest object. In such a scenario, web-application users won't be able to view the complete postback progress bar shown by the browser. In an AJAX environment, it is Javascript that starts the communication with the web server

.
Ajax technology in a website may be implemented by using plain Javascript and XML. Code in such a scenario may tend to look little complex, for which the AJAX Framework in .NET can be embedded in ASP.NET web applications.
In addition to XML & Javascript, AJAX is also based on DOM - the Document Object Model technology of browsers through which objects of the browser can be accessed through the memory heap using their address.
JSON - Javascript Object Notation is also one of the formats used in AJAX, besides XML.
So basically, in an AJAX-based web application, the complete page does not need to reload, and only the objects in context of ajaxification are reloaded.
Ajax technology avoids the browser flickering.

Q2 - Can Ajax be implemented in browsers that do not support the XmlHttpRequest object?

A - Yes. This is possible using remote scripts.

Q3 - Can AJAX technology work on web servers other than IIS?

A - Yes, AJAX is a technology independent of web server the web application is hosted on. Ajax is a client (browser) technology.

Q4 - Which browsers support the XmlHttpRequest object?

A - Internet Explorer 5.0+, Safari 1.2, Mozilla 1.0/Firefox, Opera 8.0 +, Netscape 7

Q5 - How to we create an XmlHttpRequest object for Internet Explorer? How is this different for other browsers?

A - For Internet Explorer, an ActiveXObject is used for declaring an XmlHttpRequest object in Javascript.
//Code as below for IE:

```
xmlHttpObject = new ActiveXObject("Msxml2.XMLHTTP");
```

//For Other browsers, code as below:

```
xmlHttpObject = new XMLHttpRequest();
```

Note that XmlHttpObject used above is simply a variable that holds the XmlHttpRequest object for the respective browsers.

Q6 - What are the properties of the XmlHttpRequest object? What are the different types of readyStates in Ajax?

A - i) onreadyStateChange - This function is used to process the reply from the web server.

ii) readyState - This property holds the response status of the web server. There are 5 states:

0 - request not yet initialized

1 - request now set

2 - request sent

3 - request processing

4 - request completes

iii) responseText - Has the data sent back by the web server

Code snippet below shows an example how these there properties are used to implement ajax :

```
xmlHttpObject.onreadystatechange=function()
{
if(xmlHttpObject.readyState==4)
{
document.Form1.time.value=xmlHttpObject.responseText;
}
}
```

Q7 - What is the ASP.NET Ajax Framework? What versions have been released so far?

A - ASP.NET AJAX is a free framework to implement Ajax in asp.net web applications, for quickly creating efficient and interactive Web applications that work across all popular browsers.

The Ajax Framework is powered with

1 - Reusable Ajax Controls

2 - Support for all modern browsers

3 - Access remote services and data from the browser without tons of complicated script.

Versions of Ajax release

1 - ASP.NET Ajax Framework 1.0 (earlier release to this was called the Atlas)
2 - ASP.NET Ajax Framework 1.0 was available as a separate download for ASP.NET 2.0

Q8 - What are Ajax Extensions?

A - The ASP.NET Ajax Extensions are set of Ajax-based controls that work in ASP.NET 2 (or above) based applications.

Ofcourse,they also need the Ajax runtime which is actually the Ajax Framework 1.0.

ASP.NET Ajax Extensions 1.0 have to be downloaded to run with ASP.NET 2.0

The new ASP.NET 3.5 Framework comes with the Ajax Library 3.5 (containing the Ajax Extensions 3.5). So in order

to use the latest Ajax, simply download .NET 3.5 Framework.

Summary :
ASP.NET Ajax Extensions 1.0 -> For ASP.NET 2.0
ASP.NET Ajax Extensions 3.5 -> For ASP.NET 3.5

Q9 - What is the ASP.NET Control Toolkit?

A - Besides the Ajax Framework (which is the Ajax engine) and Ajax Extensions (which contain the default Ajax controls), there is a toolkit called the Ajax Control Toolkit available for use & download (for free). This is a collection of rich featured, highly interactive controls, created as a joint venture between Microsoft & the Developer Community.

Q10 - What is Dojo?

A - Dojo is a third-party javascript toolkit for creating rich featured applications. Dojo is an Open Source DHTML toolkit written in JavaScript. It builds on several contributed code bases (nWidgets, Burstlib, f(m)), which is why we refer to it sometimes as a "unified" toolkit. Dojo aims to solve some long-standing historical problems with DHTML which prevented mass adoption of dynamic web application development.

For more on Dojo, check this link: Click Here

Q11 - How to handle multiple or concurrent requests in Ajax?

A - For concurrent requests, declare separate XmlHttpRequest objects for each request. For example, for request to get data from an SQL table1, use something like this...

xmlHttpObject1.Onreadystatechange = functionfromTable1();

and to get data from another table (say table2) at the same time, use

xmlHttpObject2.Onreadystatechange = functionfromTable2();

Ofcourse, the XmlHttpObject needs to be opened & parameters passed too, like as shown below...

xmlHTTPObject1.open("GET","http://"localhost// " + "Website1/Default1.aspx" true);

Note that the last parameter "true" used above means that processing shall carry on without waiting for any response from the web server. If it is false, the function shall wait for a response.

Q12 - How to create an AJAX website using Visual Studio?

A - Using Visual Studio Web Developer

Express 2005 & versions above it, Ajax based applications may easily be created. Note that the Ajax Framework & Ajax Extensions should be installed (In case of VS 2005). If using Visual Studio 2008 Web Developer Express or above, Ajax comes along with it (so no need of a separate installation).

Steps: Start Visual Studio, Click on File -> New Website -> Under Visual Studio Installed templates -> Select ASP.NET Ajax-Enabled Site. Enter a location & select OK.

Q13 - What is the role of ScriptManager in Ajax?

A - ScriptManager class is the heart of ASP.NET Ajax. Before elaborating more on ScriptManager, note that ScriptManager is class and a control (both) in Ajax.

The ScriptManager class in ASP.NET manages Ajax Script Libraries, partial page rendering functionality and client proxy class generation for web applications and services. By saying client proxy class, this means an instance of the Ajax runtime is created on the browser.

This class is defined in the System.Web.Extensions.dll. You will find this DLL in your system's Global Assembly Cache at C:\Windows\Assembly (For XP)

The ScriptManager control (that we may drag on a web form) is actually an instance of the ScriptManager class that we put on a web page. The ScriptManager manages all the ASP.NET Ajax controls on a web page. Following tasks are taken care by the ScriptManager class:
1 - Managing all resources (all objects/controls) on a web page
2 - Managing partial page updates
3 - Download Ajax Script Library to the client (means to the browser). This needs to happen so that Ajax engine is accessible to the browsers javascript code.
4 - Interacting with UpdatePanel Control, UpdateProgress Control.
5 - Register script (using RegisterClientScriptBlock)
6 - Information whether Release OR Debug script is sent to the browser
7 - Providing access to Web service methods from the script by registering Web services with the ScriptManager control
8 - Providing access to ASP.NET authentication, role, and profile application services from client script after registering these services with the ScriptManager control
9 - Enable culture specific display of clientside script.
10 - Register server controls that implement IExtenderControl and IScriptControl interfaces.

ScriptManager class' **EnablePartialRendering** property is true by default.

Q14 - Can we override the EnablePartialRendering property of the ScriptManager class?

A - Yes. But this has to be done before the init event of the page (or during runtime after the page has already loaded). Otherwise an InvalidOperationException will be thrown.

Q15 - How to use multiple ScriptManager controls in a web page?

A - No. It is not possible to use multiple ScriptManager control in a web page. In fact, any such requirement never comes in because a single ScriptManager control is enough to handle the objects of a web page.

Q16 - Whats the difference between RegisterClientScriptBlock, RegisterClientScriptInclude and RegisterClientScriptResource?

A - For all three, a script element is rendered after the opening form tag. Following are the differences:
1 - RegisterClientScriptBlock - The script is specified as a string parameter.
2 - RegisterClientScriptInclude - The script content is specified by setting the src attribute to a URL that points to a script file.
3 - RegisterClientScriptResource - The script content is specified with a resource name in an assembly. The src attribute is automatically populated with a URL by a call to an HTTP handler that retrieves the named script from the assembly.

Q17 - What are type/key pairs in client script registration? Can there be 2 scripts with the same type/key pair name?

A - When a script is registered by the ScriptManager class, a type/key pair is created to uniquely identify the script.

For identification purposes, the type/key pair name is always unique for dentifying a script. Hence, there may be no duplication in type/key pair names.

Q18 - What is an UpdatePanel Control?

A - An UpdatePanel control is a holder for server side controls that need to be partial postbacked in an ajax cycle. All controls residing inside the UpdatePanel will be partial postbacked. Below is a small example of using an UpdatePanel.

```
<script runat="server">
protected void btn1_Click(object sender, EventArgs e)
{
 lb123.Text = "new";
}
</script>

<asp:ScriptManager ID="ScriptManager1" runat="server">
  </asp:ScriptManager>
<asp:UpdatePanel ID="UpdatePanel1" runat="server">
  <ContentTemplate>
 <asp:Button id="btn1" runat="server" text="click"/>
<br/>
 <asp:Label id="lb123" runat="server" text="Old"/>
</ContentTemplate>
</UpdatePanel>
```

As you see here after running the snippet above, there wont be a full postback exhibited by the web page. Upon clicking the button, the postback shall be partial. This means that contents outside the UpdatePanel wont be posted back to the web server. Only the contents within the UpdatePanel are refreshed.

Q19 - What are the modes of updation in an UpdatePanel? What are Triggers of an UpdatePanel?

A - An UpdatePanel has a property called **UpdateMode**. There are two possible values for this property: 1) Always 2) Conditional

If the UpdateMode property is set to "Always", the UpdatePanel control’s content is updated on each postback that starts from anywhere on the webpage. This also includes asynchronous postbacks from controls that are inside other UpdatePanel controls, and postbacks from controls which are not inside UpdatePanel controls.

If the UpdateMode property is set to Conditional, the UpdatePanel control’s content is updated when one of the following is true:
1 - When the postback is caused by a trigger for that UpdatePanel control.

2 - When you explicitly call the UpdatePanel control's Update() method.

3 - When the UpdatePanel control is nested inside another UpdatePanel control and the parent panel is updated.

When the ChildrenAsTriggers property is set to true and any child control of the UpdatePanel control causes a postback. Child controls of nested UpdatePanel controls do not cause an update to the outer UpdatePanel control unless they are explicitly defined as triggers for the parent panel.

Controls defined inside a <Triggers> node have the capability to update the contents of an UpdatePanel.

If the ChildrenAsTriggers property is set to false and the UpdateMode property is set to Always, an exception is thrown. The ChildrenAsTriggers property is intended to be used only when the UpdateMode property is set to Conditional.

Q20 - How to control how long an Ajax request may last?

A - Use the ScriptManager's AsyncPostBackTimeout Property.

For example, if you want to debug a web page but you get an error that the page request has timed out, you may set <asp:ScriptManager id="ScriptManager1" runat="server" AsyncPostBackTimeout="9000"/>

where the value specified is in seconds.

Q21 - What is ASP.NET Futures?

A -
ASP.NET AJAX Futures

The new release includes support for managing browser history (Back button support), selecting elements by CSS selectors or classes, and information on accessing “Astoria” Web data services. The Futures (July 2007) release adds:

History support for the Safari browser, inclusion of “titles”, encoding and encrypting of server-side history state and the ability to handle history in the client without a server requirement.

CSS Selectors APIs have been modified to be applicable to W3C recommendations.

A script resource extraction tool that allows you to create script files on disk that originate from embedded resources in assemblies. Important: this version of the browser history feature is now outdated and should not be used. Instead, please download the ASP.NET 3.5 Extensions Preview, which contains the new version.

For More: Click Here

Q22 - What are limitations of Ajax?

A - 1) An Ajax Web Application tends to confused end users if the network bandwidth is slow, because there is no full postback running. However, this confusion may be eliminated by using an UpdateProgress control in tandem.
2) Distributed applications running Ajax will need a central mechanism for communicating with each other

Q23 - How to make sure that contents of an UpdatePanel update only when a partial postback takes place (and not on a full postback)?

A - Make use of ScriptManager.IsInAsyncPostBack property (returns a boolean value)

Q24 - How to trigger a postback on an UpdatePanel from Javascript?

A - Call the __doPostBack function. ASP.NET runtime always creates a javascript function named __doPostBack(eventTarget, eventArgument) when the web page is rendered. A control ID may be passed here to specifically invoke updation of the UpdatePanel.

Q25 - Which request is better with AJAX, Get or Post?

A - AJAX requests should use an HTTP GET request while retrieving data where the data does not change for a given URL requested. An HTTP POST should be used when state is updated on the server. This is in line with HTTP idempotency recommendations and is highly recommended for a consistent web application architecture.