UNIT 5 - AJAX & WEB SERVICES

Part-A

1. What is Ajax?
   - AJAX is an acronym for asynchronous JavaScript and XML
   - It is a set of web development techniques using many web technologies on the client-side to create asynchronous Web applications.
   - With Ajax, web applications can send data to and retrieve from a server asynchronously (in the background) without interfering with the display and behavior of the existing page.
   - Ajax is not a technology, but a group of technologies.
   - HTML and CSS can be used in combination to mark up and style information.
   - The DOM is accessed with JavaScript to dynamically display and allow the user to interact with the information presented.
   - JavaScript and the XMLHttpRequest object provide a method for exchanging data asynchronously between browser and server to avoid full page reloads.

2. Mention the open standards of Ajax.
   - Browser-based presentation using HTML and Cascading Style Sheets (CSS).
   - Data is stored in XML format and fetched from the server.
   - Behind-the-scenes data fetches using XMLHttpRequest objects in the browser.
   - JavaScript to make everything happen.

3. Brief about asynchronous nature of AJAX.
   - Asynchronous means that the script will send a request to the server, & continue it's execution without waiting for reply.
   - As soon as reply is received a browser event is fired, which in turn allows the script to execute associated actions.
   - Ajax knows when to pull data from server, because you tell it when to do it.

4. What is XHR?
   - XMLHttpRequest (XHR) is an API that can be used by JavaScript, JScript, VBScript, and other web browser scripting languages to transfer and manipulate XML data to and from a webserver using HTTP, establishing an independent connection channel between a webpage's Client-Side and Server-Side.
     - Update a web page without reloading the page
     - Request data from a server - after the page has loaded
     - Receive data from a server - after the page has loaded
     - Send data to a server - in the background

5. What is a web service?
   - Web services are open standard (XML, SOAP, HTTP etc.) based Web applications that interact with other web applications for the purpose of exchanging data.
   - It is OS and language independent
   - Web Services can convert your existing applications into Web-applications.
   - A web service is a collection of open protocols and standards used for exchanging data between applications or systems.

6. Mention the characteristics of web service.
   - Machine-to-machine interactions
   - Loose coupling
   - Interoperability
   - Platform-independence
   - Operating system-independence
   - Language-independence
   - Leveraging the architecture of the World Wide Web

7. What are the components of Web Services?
   The basic web services platform is XML + HTTP.
   - SOAP (Simple Object Access Protocol)
   - UDDI (Universal Description, Discovery and Integration)
   - WSDL (Web Services Description Language)

8. What are the advantages of web service?
   - Exposing the Existing Function on the network
   - Interoperability
   - Standardized protocol
   - Low Cost of Communication

9. What are RESTful web services?
   - RESTful Web Services are REST architecture based web services.
   - In REST Architecture everything is a resource, RESTful web services are light weight, highly scalable and maintainable.
   - It is very commonly used to create APIs for web based applications.
   - REST stands for Representational State Transfer.
   - REST is web standards based architecture and uses HTTP Protocol for data communication.

10. Define WSDL
    - WSDL stands for Web Services Description Language.
    - It is the standard format for describing a web service.
    - WSDL was developed jointly by Microsoft and IBM.
    - To exchange information in a distributed environment.
    - WSDL is used to describe services.
    - WSDL is written in XML
    - WSDL is a W3C recommendation from 26. June 2007

11. What are the elements of WSDL?
    - Types – a container for data type definitions using some type system (such as XSD).
    - Message – an abstract, typed definition of the data being communicated.
    - Operation – an abstract description of an action supported by the service.
    - Port Type – an abstract set of operations supported by one or more endpoints.
    - Binding – a concrete protocol and data format specification for a particular port type.
    - Port – a single endpoint defined as a combination of a binding and a network address.
    - Service – a collection of related endpoints.

12. Define SOAP
    - SOAP is an acronym for Simple Object Access Protocol.
    - It is an XML-based messaging protocol for exchanging information among computers.
    - SOAP is an application of the XML specification.
    - SOAP is an application communication protocol
    - SOAP is a format for sending and receiving messages
    - SOAP is platform independent
    - SOAP is based on XML
    - SOAP is a W3C recommendation

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

- An **Envelope** element that identifies the XML document as a SOAP message
- A **Header** element that contains header information
- A **Body** element that contains call and response information
- A **Fault** element containing errors and status information

15. Mention the advantages of SOAP.

<table>
<thead>
<tr>
<th>Advantage</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simplicity</td>
<td>Universal acceptance</td>
</tr>
<tr>
<td>Portability</td>
<td>Versatile</td>
</tr>
<tr>
<td>Firewall</td>
<td>Flexible</td>
</tr>
<tr>
<td>Scalability</td>
<td>Interoperability</td>
</tr>
<tr>
<td>Friendliness</td>
<td>No bi-directional HTTP communication</td>
</tr>
<tr>
<td>Use of open</td>
<td>No distributed garbage collection</td>
</tr>
<tr>
<td>No Object</td>
<td>No Object by reference.</td>
</tr>
</tbody>
</table>

Steps of Processing in Ajax:-

1. An event occurs in a web page (the page is loaded, a button is clicked)
2. An XMLHttpRequest object is created by JavaScript
3. The XMLHttpRequest object sends a request to a web server
4. The server processes the request
5. The server sends a response back to the web page
6. The response is read by JavaScript
7. Proper action (like page update) is performed by JavaScript

**AJAX is based on the following open standards:-**

- Browser-based presentation using HTML and Cascading Style Sheets (CSS).
- Data is stored in XML format and fetched from the server.
- Behind-the-scenes data fetches using XMLHttpRequest objects in the browser.
- JavaScript to make everything happen.

**Components of Ajax:-**

**JavaScript**
- Loosely typed scripting language.
- JavaScript function is called when an event occurs in a page.
- Glue for the whole AJAX operation.

**DOM**
- API for accessing and manipulating structured documents.
- Represents the structure of XML and HTML documents.

**CSS**
- Allows for a clear separation of the presentation style from the content and may be changed programmatically by JavaScript.

**XMLHttpRequest**
- JavaScript object that performs asynchronous interaction with the server.

**Example for AJAX:** Google maps, Gmail, cricket update websites, stock markets websites, etc

**Example:-**

```html
<!DOCTYPE html>
<html>
<body>
  <div id="demo">
    <h1>The XMLHttpRequest Object</h1>
    <button type="button" onclick="loadDoc()">Change Content</button>
  </div>
  <script>
    function loadDoc() {
      var xhttp = new XMLHttpRequest();
      xhttp.onreadystatechange = function() {
        if (this.readyState == 4 && this.status == 200) {
          document.getElementById("demo").innerHTML = this.responseText;
        }
      };
      xhttp.open("GET", "ajax_info.txt", true);
      xhttp.send();
    }
  </script>
</body>
</html>
```
2. Explain XMLHttpRequest object with example.

- The XMLHttpRequest object can be used to request data from a web server.
- The XMLHttpRequest object is a developers dream, because you can:
  - Update a web page without reloading the page
  - Request data from a server - after the page has loaded
  - Receive data from a server - after the page has loaded
  - Send data to a server - in the background
- The XMLHttpRequest object is the key to AJAX. It has been available ever since Internet Explorer 5.5 was released in July 2000, but was not fully discovered until AJAX and Web 2.0 in 2005 became popular.
- XMLHttpRequest (XHR) is an API that can be used by JavaScript, JScript, VBScript, and other web browser scripting languages to transfer and manipulate XML data to and from a webserver using HTTP, establishing an independent connection channel between a webpage's Client-Side and Server-Side.
- The data returned from XMLHttpRequest calls will often be provided by back-end databases. Besides XML, XMLHttpRequest can be used to fetch data in other formats, e.g. JSON or even plain text.

Example:
```html
<!DOCTYPE html>
<html>
<body>
<h2>Using the XMLHttpRequest Object</h2>
<div id="demo">
<button type="button" onclick="loadXMLDoc()">Change Content</button>
</div>
<script>
function loadXMLDoc() {
    var XMLHttpRequest = window.XMLHttpRequest;
    var request = new XMLHttpRequest;
    request.open('GET', 'data.xml', true);
    request.onreadystatechange = function()
    {
        if (this.readyState == 4)
        {
            if (this.status == 200) {
                var xmlDoc = this.responseXML;
                // Process the XML data
            } else {
                console.log('Error: ' + this.status);
            }
        }
    }
    request.send(null);
}
</script>
</body>
</html>
```
WSDL is pronounced as 'wiz-dull' and spelled out as 'W-S-D-L'.

3. Describe WSDL with diagrams necessary.
   - WSDL stands for Web Services Description Language
   - WSDL is used to describe web services
   - WSDL is written in XML
   - WSDL is a W3C recommendation from 26 June 2007
   - It is the standard format for describing a web service.
   - WSDL was developed jointly by Microsoft and IBM.

Features of WSDL
- WSDL is an XML-based protocol for information exchange in decentralized and distributed environments.
- WSDL definitions describe how to access a web service and what operations it will perform.
- WSDL is a language for describing how to interface with XML-based services.
- WSDL is an integral part of Universal Description, Discovery, and Integration (UDDI), an XML-based worldwide business registry.
- WSDL is the language that UDDI uses.
- WSDL is pronounced as 'wiz-dull' and spelled out as 'W-S-D-L'.

WSDL Elements
- Types – a container for data type definitions using some type system (such as XSD).
- Message – an abstract, typed definition of data being communicated.
- Operation – an abstract description of an action supported by service.
- PortType – an abstract set of operations supported by one or more endpoints.
- Binding – a concrete protocol and data format specification for a particular port type.
- Port – a single endpoint defined as a combination of a binding and a network address.
- Service – a collection of related endpoints.

The WSDL Document Structure

<definitions>
  <types>
    <definition of types....>
    <messages>
      <definition of a message....>
      <portTypes>
        <definition of a binding....>
        <services>
          <definition of a service....>
</services>
</portTypes>
</messages>
</types>
</definitions>

Contents of HelloService.wsdl file:

  <message name="SayHelloRequest">
    <input message="tns:SayHelloRequest"/>
  </message>
  <message name="SayHelloResponse">
    <output message="tns:SayHelloResponse"/>
  </message>
  <portType name="Hello_PortType">
    <operation name="sayHello">
      <input message="tns:SayHelloRequest"/>
      <output message="tns:SayHelloResponse"/>
    </operation>
  </portType>
  <binding name="Hello_Binding" type="tns:Hello_PortType">
    <soap:binding style="rpc"
      transport="http://schemas.xmlsoap.org/soap/http"/>
    <operation name="sayHello">
      <input>
        <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/
        namespace="urn:examples:helloservice"
        use="encoded">
      </input>
      <output>
        <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/
        namespace="urn:examples:helloservice"
        use="encoded">
      </output>
    </operation>
  </binding>
</definitions>
Why SOAP?
- It is important for web applications to be able to communicate over the Internet.
- The best way to communicate between applications is over HTTP, because HTTP is supported by all Internet browsers and servers. SOAP was created to accomplish this.
- SOAP provides a way to communicate between applications running on different operating systems, with different technologies and programming languages.

SOAP Building Blocks
- An Envelope element that identifies the XML document as a SOAP message
- A Header element that contains header information
- A Body element that contains call and response information
- A Fault element containing errors and status information

Syntax Rules
- A SOAP message MUST be encoded using XML
- A SOAP message MUST use the SOAP Envelope namespace
- A SOAP message MUST use the SOAP Encoding namespace
- A SOAP message must NOT contain a DTD reference
- A SOAP message must NOT contain XML Processing Instructions

SOAP request
POST /Quotation HTTP/1.0
Host: www.xyz.org
Content-Type: text/xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
<SOAP-ENV:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope"
    soap:encodingStyle="http://www.w3.org/2003/05/soap-encoding">
    <soap:Header>
        ...
    </soap:Header>
    <soap:Body>
        ...
    </soap:Body>
    <soap:Fault>
        ...
    </soap:Fault>
</soap:Envelope>
SOAP response

HTTP/1.0 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: nnn

<?xml version="1.0"?>
  <SOAP-ENV:Body xmlns:m="http://www.xyz.org/quotation">
    <m:GetQuotationResponse>
      <m:Quotation>Here is the quotation</m:Quotation>
    </m:GetQuotationResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>