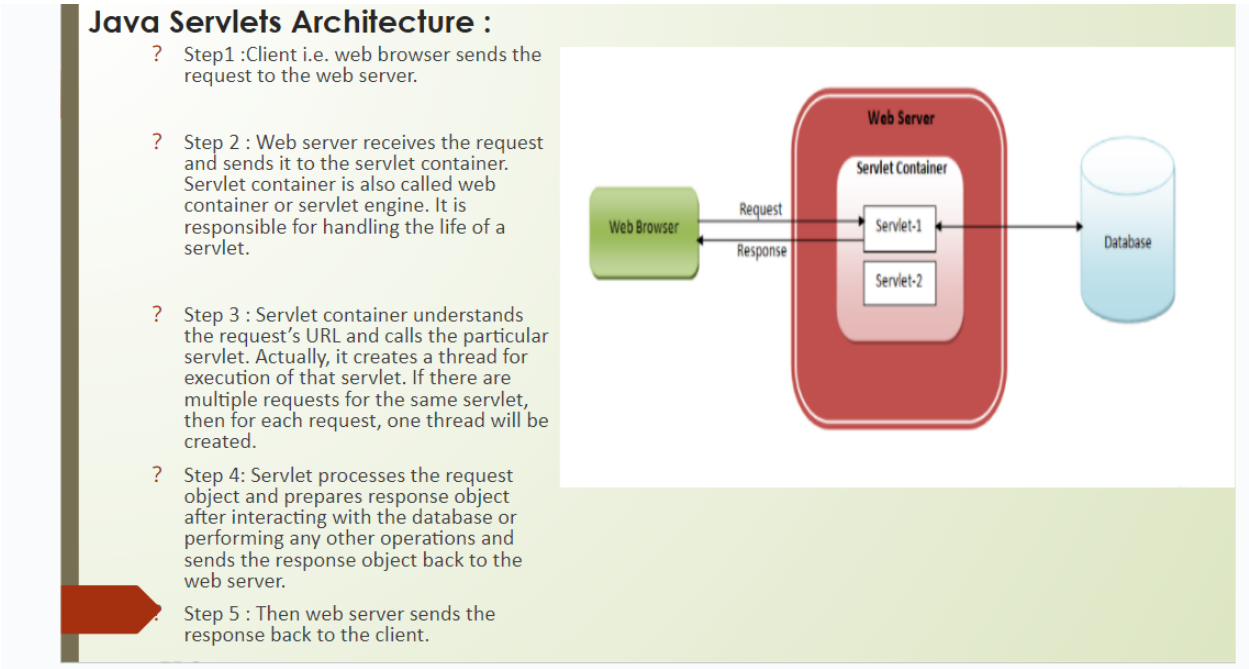
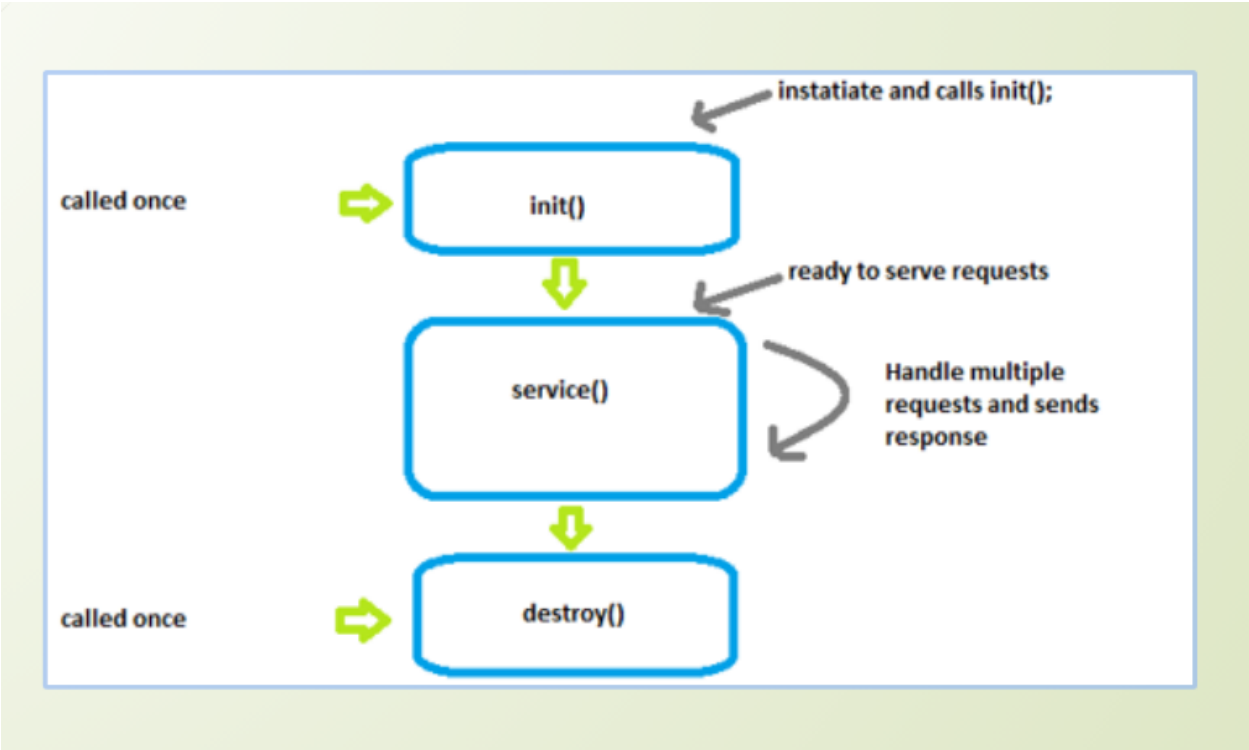
1. Explain Servlet architecture.



2. Explain servlet lifecycle



The init() method is called only once, when the servlet is first loaded into the web container. The service() method is called each time the servlet receives a request from a client. The destroy() method is called only once, when the servlet is being unloaded from the web container.

The image shows that the servlet lifecycle is a three-phase process: initialization, service, and destruction. The initialization phase is responsible for setting up the servlet and making it ready to serve requests. The service phase is responsible for processing client requests and generating responses. The destruction phase is responsible for cleaning up any resources used by the servlet.

The servlet lifecycle is managed by the web container. The web container is responsible for loading and unloading servlets, calling their lifecycle methods, and managing their threads.

Here is a more detailed explanation of each phase of the servlet lifecycle:

Initialization

The initialization phase is responsible for setting up the servlet and making it ready to serve requests. This includes performing tasks such as:

Loading the servlet class
Creating an instance of the servlet
Calling the servlet's init() method
The init() method is responsible for initializing the servlet's state and configuring it for service. For example, the init() method might be used to connect to a database or to create a pool of threads.

Service

The service phase is responsible for processing client requests and generating responses. This includes performing tasks such as:

Reading the request headers and body
Performing the requested operation
Generating the response headers and body
Sending the response back to the client
The service() method is called each time the servlet receives a request from a client. The service() method is responsible for handling the request and generating a response.

Destruction

The destruction phase is responsible for cleaning up any resources used by the servlet. This includes performing tasks such as:

Closing any open database connections
Shutting down any thread pools
Releasing any other resources that were allocated by the servlet

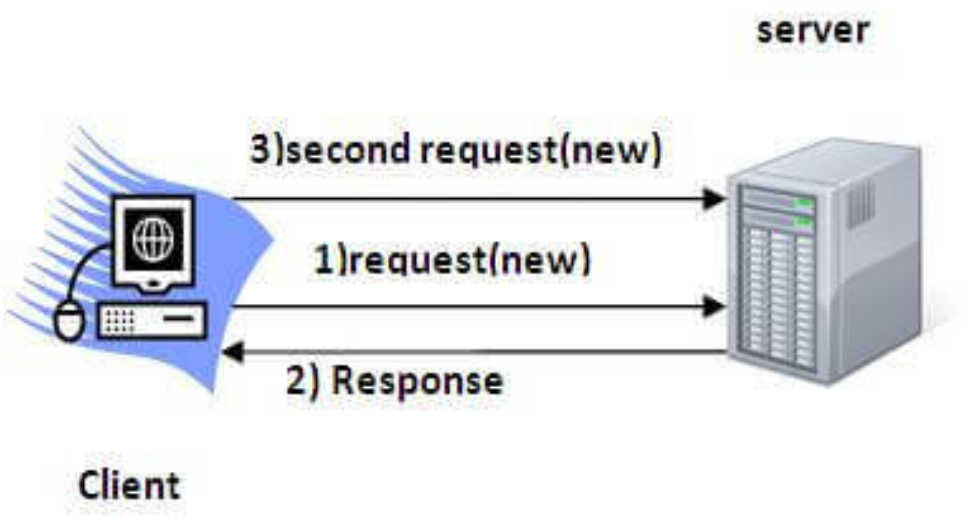
The destroy() method is called only once, when the servlet is being unloaded from the web container. The destroy() method is responsible for cleaning up any resources used by the servlet and ensuring that it is in a clean state before it is unloaded.

3. Explain Session Handling in Servlet

Session simply means a particular interval of time.

Session Tracking/Handling is a way to maintain state (data) of an user. It is also known as session management in servlet.

HTTP is stateless that means each request is considered as the new request. It is shown in the figure given below:



HTTP is a stateless protocol which means when a new request comes it can't keep any record or state of previous request of the user.
That's why we need session tracking for maintaining the state of the user.

The four techniques used for session tracking are:

1. Cookies
2. Hidden form field
3. URL rewriting
4. Httpsession

Lets see how to do it with cookies:

Cookies are small pieces of data that a web server sends to a user's web browser while the user is browsing a website. These cookies are stored on the user's device and are sent back to the server with subsequent requests. Cookies are often used to track user sessions, store user preferences, and maintain state information across multiple HTTP requests. They can be both temporary (session cookies) and persistent (stored on the user's device for a specified duration).

Here are the small steps to use cookies for session tracking in a web application:

1.Cookie Creation: When a user first visits your website or logs in, generate a unique session identifier for them.

2. Sending a Cookie: Create an HTTP cookie and include it in the response sent from the server to the user's browser. This can be done using the `Set-Cookie` HTTP header. The cookie should typically include the following information:
-Name: A unique name for the cookie (e.g., "sessionId").
- Value: The unique session identifier generated in step 1.
-Domain: The domain for which the cookie is valid (usually the current domain).
- Path: The path within the domain for which the cookie is valid (usually "/").

Example in Java Servlet:

```
Cookie sessionCookie = new Cookie("sessionId", sessionID);
sessionCookie.setPath("/");
response.addCookie(sessionCookie);
```

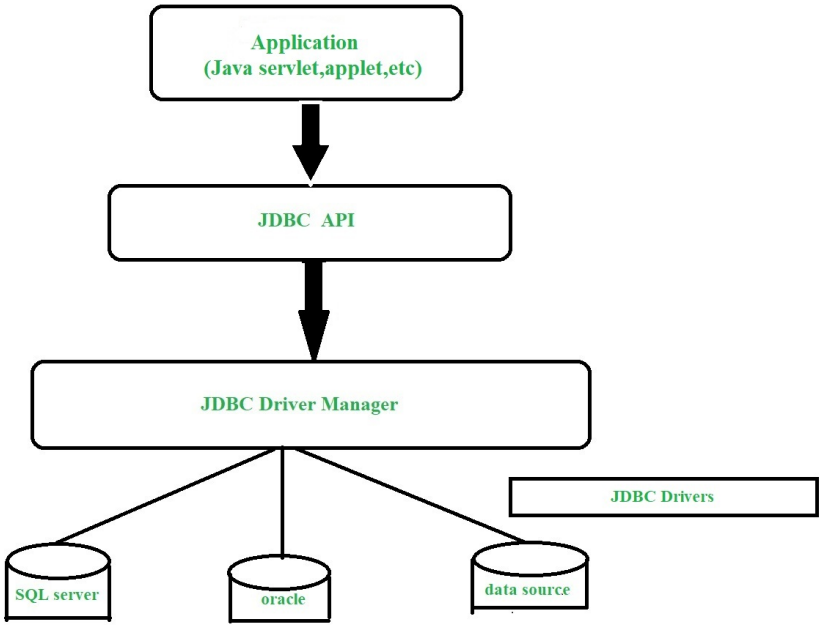
3. Receiving a Cookie: When the user makes subsequent requests to the server, their browser automatically includes the cookie in the request headers. On the server side, you can retrieve the cookie using the `Cookie` HTTP header in the request.

Example in Java Servlet:

```
Cookie[] cookies = request.getCookies();
String sessionID = null;
if (cookies != null) {
    for (Cookie cookie : cookies) {
        if (cookie.getName().equals("sessionId")) {
            sessionID = cookie.getValue();
            break;
        }
    }
}
```

4. Session Management: Use the session identifier (`sessionID`) to associate the user's request with their session data on the server. You can store session data in a server-side data structure.

4. Explain JDBC architecture



Description:

Application: It is a java applet or a servlet that communicates with a data source.
The JDBC API: The JDBC API allows Java programs to execute SQL statements and retrieve results. Some of the important classes and interfaces defined in JDBC API are as follows:
DriverManager: It plays an important role in the JDBC architecture. It uses some database-specific drivers to effectively connect enterprise applications to databases.
JDBC drivers: To communicate with a data source through JDBC, you need a JDBC driver that intelligently communicates with the respective data source.

JDBC provides a uniform way to access relational databases from Java applications. This makes it easier to develop database applications and to port applications to different database systems.
JDBC is supported by all major database vendors. This means that developers can choose the database system that is best for their needs.
JDBC is a mature and well-tested technology. It is used in a wide variety of applications, from small desktop applications to large enterprise applications.

Overall, JDBC is a powerful and flexible tool for accessing relational databases from Java applications. It is a good choice for developers who need to develop database-driven applications.

5. Explain syntax for JDBC program with one example (GPT se liye)

The Java Database Connectivity (JDBC) API allows you to connect and interact with databases from Java applications. To create a JDBC program, you need to follow a specific syntax and a series of steps. Here's a basic explanation of the JDBC syntax with an example:

JDBC Syntax:

1. **Import JDBC Packages**: Import the necessary JDBC packages at the beginning of your Java program.

```java
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.sql.ResultSet;
```

2. **Register the JDBC Driver**: Register the JDBC driver for the specific database you want to connect to. This step is required only for older versions of JDBC (prior to JDBC 4.0). In JDBC 4.0 and later, this step is optional as the drivers are automatically loaded.

```java
Class.forName("com.mysql.jdbc.Driver");
```

3. **Establish a Database Connection**: Create a connection to the database using the `DriverManager.getConnection` method. You need to provide the database URL, username, and password as parameters.

```java
String url = "jdbc:mysql://localhost:3306/your_database_name";
String username = "your_username";
String password = "your_password";
Connection connection = DriverManager.getConnection(url, username, password);
```

4. **Create SQL Statements**: Create SQL statements (e.g., SELECT, INSERT, UPDATE, DELETE) using the `Statement` or `PreparedStatement` interface.

```java
Statement statement = connection.createStatement();
String sqlQuery = "SELECT * FROM your_table_name";
```

5. **Execute SQL Statements**: Execute the SQL statements using the appropriate methods (`executeQuery`, `executeUpdate`, etc.) on the statement object.

```java
ResultSet resultSet = statement.executeQuery(sqlQuery);
```

6. **Process the Results**: If you executed a query that retrieves data (e.g., SELECT), process the results by iterating through the `ResultSet`.

```java
while (resultSet.next()) {
    // Process each row of data here
    String column1 = resultSet.getString("column1_name");
    int column2 = resultSet.getInt("column2_name");
    // ...
}
```

7. **Close Resources**: After you have finished using the database connection, statements, and result sets, make sure to close them to release resources and avoid memory leaks.

```java
resultSet.close();
statement.close();
connection.close();
```

8. **Handle Exceptions**: Handle exceptions that may be thrown during database operations. Wrap database-related code in try-catch blocks to handle `SQLException` appropriately.

Example:

Here's a simple example of a JDBC program that connects to a MySQL database, retrieves data from a table, and prints it:

```java
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;

public class JDBCTutorial {
    public static void main(String[] args) {
        String url = "jdbc:mysql://localhost:3306/mydb";
        String username = "root";
        String password = "your_password";

        try {
            Connection connection = DriverManager.getConnection(url, username, password);
            Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery("SELECT * FROM employees");

            while (resultSet.next()) {
                String name = resultSet.getString("name");
                int age = resultSet.getInt("age");
                System.out.println("Name: " + name + ", Age: " + age);
            }

            resultSet.close();
            statement.close();
            connection.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Remember to replace `"your_database_name"`, `"your_username"`, and `"your_password"` with your specific database information.

6. Write short note on characteristics of Rich Internet Architecture(RIA)

Desktop is a very powerful s/w experience when talked about look & feel.
To a user, it is easy to handle and work with any software that behaves in a manner similar to a users desktop.
New ways are being adopted to develop interfaces that provide users with better powerful experiences.

Applications developed to provide exciting rich interactivity features in software programs are known as Rich Internet Applications (RIA).

RIAs are developed as web applications behaving like desktop application in their look & feel.
RIA can perform computation on both the client side and sever side.
It is developed using various technologies that includes Java, Silverlight, JavaFX, Adobe Flash & Flex, AJAX, OpenLaszlo

Characterstics:
Enhanced Accessibility: RIAs prioritize accessibility features, making web applications more usable for individuals with disabilities, such as screen readers, keyboard navigation, and text-to-speech tools.

Direct Interaction: RIAs offer a more responsive and interactive user experience, allowing users to interact with content directly without frequent page reloads.

Improved Features: RIAs provide advanced features and functionality that were traditionally associated with desktop applications, enhancing user experience and productivity.

Partial Update of WebPages: Instead of refreshing entire web pages, RIAs update only the necessary parts, reducing bandwidth usage and improving speed.

Better Feedback to Users: RIAs provide real-time feedback to users, such as validation messages and progress indicators, enhancing the overall user experience.

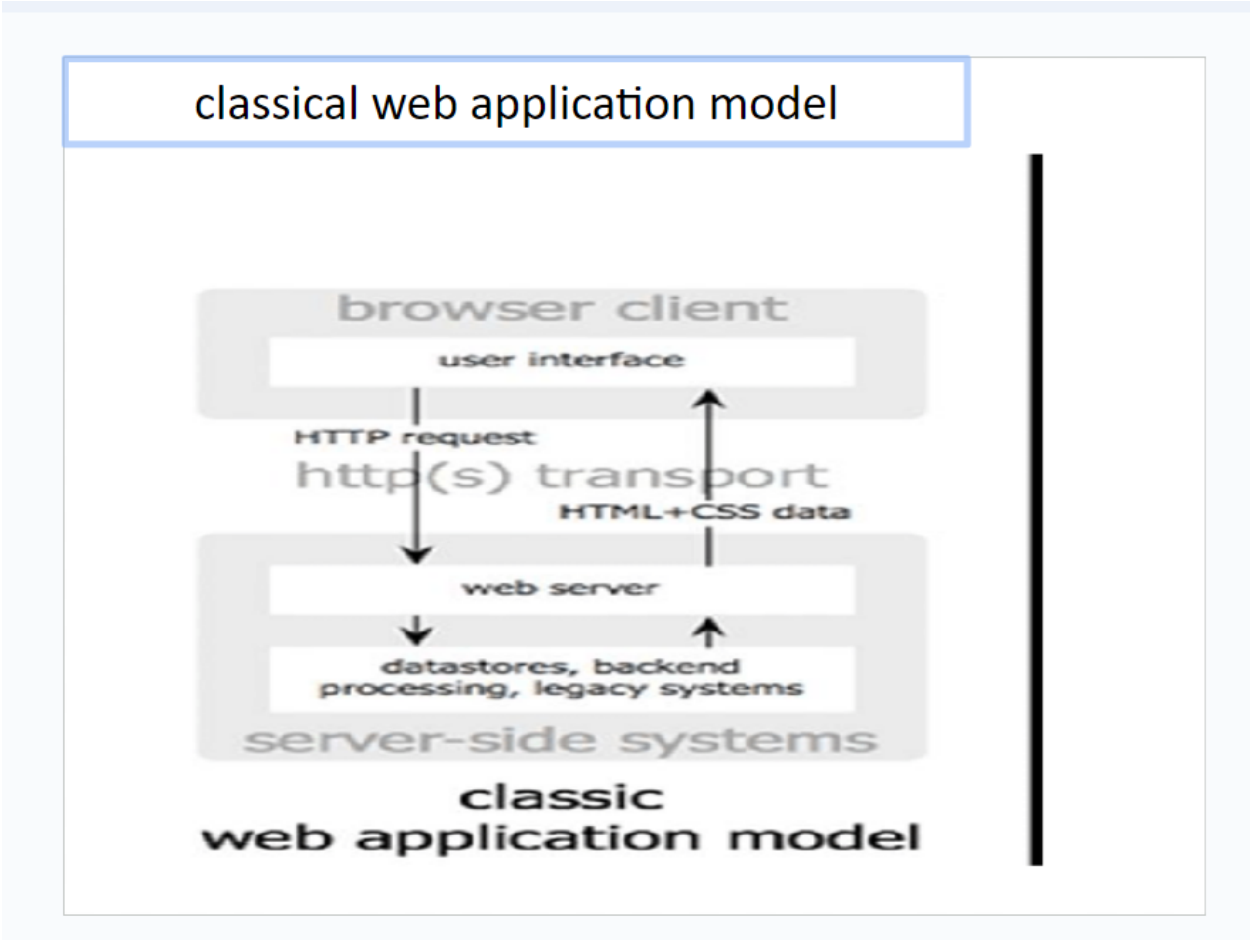
Collaborative Web Application Access: RIAs support real-time collaboration and communication, enabling users to work together on web-based applications simultaneously.

Consistency of Look & Feel: RIAs maintain a consistent and visually appealing user interface, improving brand identity and user recognition.

Offline Use of the Application: Some RIAs allow limited functionality when offline, enabling users to work without an internet connection and syncing data when online.

Impact on Performance: RIAs can have varying impacts on performance, depending on factors like complexity and optimization. Proper design and development can lead to improved performance.

7.Describe Traditional Approach of web communication. How AJAX works differently than it?



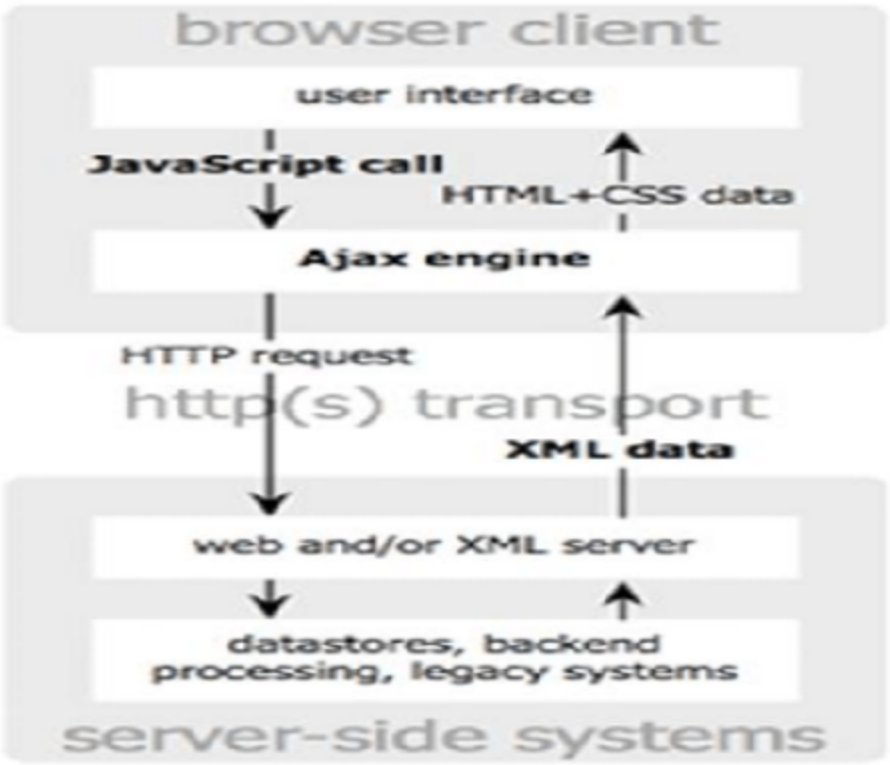
classical web application model

- When web server receives the request it processes it,by calling upon server-side scripts and databases required, then sends a response back to the browser via HTTP.

- When web browser receives the response it can then update its content accordingly.

- The round trip time taken to send a request and receive a response from a web server is known as “network latency”

AJAX web application model



Ajax
web application model

AJAX web application model

- AJAX engine is written in javascript,gets loaded when web page opens in web browser.
- Responses that do not require contact with web server such as data validation or editing of data in memory will get fulfilled internally AJAX engine.
- Responses that do require contact with web server such as submission of data for processing or retrieval of new data gets fulfilled asynchronously without interrupting the user's interaction with application.

8. Write short note on JSTL.

JSTL library

- The Java Server Pages Standard Tag Library (JSTL) is a collection of useful JSP tags which encapsulates the core functionality common to many JSP applications.

- JSTL has support for common, structural tasks such as iteration and conditionals, tags for manipulating XML documents, internationalization tags, and SQL tags.
- It also provides a framework for integrating the existing custom tags with the JSTL tags.
- To install JSTL library compatible with our servlet and JSP version

Link is:
http://www.java2s.com/Code/Jar/j/Downloadjavaxservletjspjstl30jar.htm

- To use any of the libraries, you must include a <taglib> directive at the top of each JSP that uses the library.

JSTL library

Classification of The JSTL Tags:

The JSTL tags can be classified, according to their functions, into the following JSTL tag library groups that can be used when creating a JSP page –

- Core Tags
- Formatting tags
- SQL tags
- XML tags
- JSTL Functions

Note :
That all the JSTL standard tags URL starts with https://java.sun.com/jsp/jstl/ and we can use any prefix we want but it's best practice to use the prefix defined above

JSTL library

Classification of The JSTL Tags-

2) Formatting Tags:

JSTL Formatting tags are provided for

1. formatting of Numbers,
2. Dates and i18n support through locales and resource bundles.

We can include these jstl tags in JSP with below syntax:

```
<%@ taglib uri="https://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>
```

JSTL library

Classification of The JSTL Tags-

3) JSTL SQL Tags:

1. JSTL SQL Tags provide support for interaction with relational databases such as Oracle, MySql etc.
2. Using JSTL SQL tags we can run database queries,

we include these JSTL tags in JSP with below syntax:

```
<%@ taglib uri="https://java.sun.com/jsp/jstl/sql" prefix="sql" %>
```

JSTL library

Classification of The JSTL Tags-

1) Core Tags:

JSTL Core tags provide support for

1. iteration,
2. conditional logic,
3. catch exception,
4. url,
5. forward or redirect response etc.

To use JSTL core tags, we should include it in the JSP page like below.

```
<%@ taglib uri="https://java.sun.com/jsp/jstl/core" prefix="c" %>
```

JSTL library

Classification of The JSTL Tags-

4) JSTL XML Tags:

1. JSTL XML tags are used to work with XML documents such as parsing XML, transforming XML data and XPath expressions evaluation.
2. Syntax to include JSTL XML tags in JSP page is:

```
<%@ taglib uri="https://java.sun.com/jsp/jstl/xml" prefix="x" %>
```

JSTL library

Classification of The JSTL Tags-

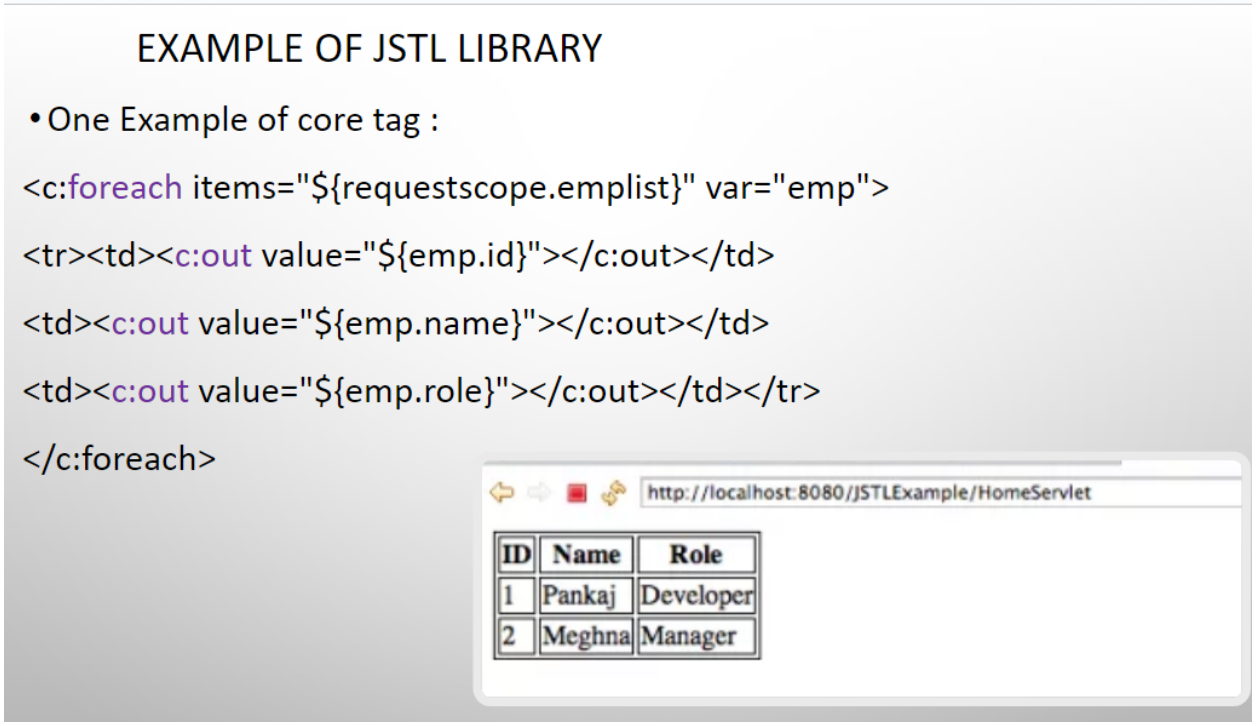
5) JSTL Functions Tags:

JSTL tags provide several functions that we can use,

1. to perform common operation,
2. most of them are for String manipulation such as String Concatenation, Split String etc.

- Syntax to include JSTL functions in JSP page is:

```
<%@ taglib uri="https://java.sun.com/jsp/jstl/functions" prefix="fn" %>
```



9. Explain user defined functions in PHP with suitable example.

Creating PHP Function:

```
<html>
<head>
<title>Writing PHP Function</title>
</head>
<body>
<?php
/* Defining a PHP Function */
function writeMessage()
{
echo "You are really a nice person, Have a nice time!";
}
/* Calling a PHP Function */
writeMessage();
?>
</body>
</html>
```

Output:
You are really a nice person, Have a nice time!

PHP Functions with Parameters:

```
<html>
<head>
<title>Writing PHP Function with Parameters</title>
</head>
<body>
<?php
function addFunction($num1, $num2)
{
$sum = $num1 + $num2;
echo "Sum of the two numbers is : $sum";
}
addFunction(10, 20);
?>
</body>
</html>
```

Sum of the two numbers is : 30

PHP Functions returning value:

```
<html>
<head>
<title>Writing PHP Function which returns value</title>
</head>
<body>
<?php
function addFunction($num1, $num2)
{
$sum = $num1 + $num2;
return $sum;
}
$return_value = addFunction(10, 20);
echo "Returned value from the function : $return_value
?>
</body>
</html>
```

Returned value from the function : 30

User-defined functions in PHP are custom code blocks created by developers to perform specific tasks or operations within a PHP script. These functions are defined using the function keyword, and they encapsulate a set of instructions that can be executed multiple times throughout a script.
These functions allow us to modularize code, reuse code, save our time and reduce redundant snippets in the code base.

10. Explain PHP MySQL connectivity with example.

PHP MySQL connectivity allows you to interact with MySQL databases from PHP scripts, making it possible to retrieve, insert, update, and manipulate data in your database. Here's an explanation of how to establish PHP MySQL connectivity with an example:

Step 1: Set Up Your MySQL Database

Before you can connect to a MySQL database from PHP, you need to create a MySQL database, a user with appropriate privileges, and a table to work with. You can use tools like phpMyAdmin or MySQL command-line tools to set up your database.

Step 2: PHP MySQL Connection Code

Here's an example of PHP code that establishes a connection to a MySQL database:

```
<?php
$servername = "localhost"; // Replace with your database server hostname or IP address
$username = "your_username"; // Replace with your MySQL username
$password = "your_password"; // Replace with your MySQL password
$dbname = "your_database_name"; // Replace with the name of your database

// Create a connection
$conn = new mysqli($servername, $username, $password, $dbname);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
} else {
    echo "Connected successfully";
}

// Close the connection when done
$conn->close();
?>
```

We use the mysqli extension to create a MySQL database connection.
We check if the connection is successful and display a message accordingly.
Finally, we close the connection using $conn->close() when we're done with it.

11. Diff between HTML and XML

HTML	XML

1.	It was written in 1993.	It was released in 1996.
2.	HTML stands for Hyper Text Markup Language.	XML stands for Extensible Markup Language.
3.	HTML is static in nature.	XML is dynamic in nature.
4.	It was developed by WHATWG.	It was developed by Worldwide Web Consortium.
5.	It is termed as a presentation language.	It is neither termed as a presentation nor a programming language.
6.	HTML is a markup language.	XML provides a framework to define markup languages.
7.	HTML can ignore small errors.	XML does not allow errors.

8.	It has an extension of .html and .htm	It has an extension of .xml
9.	HTML is not Case sensitive.	XML is Case sensitive.
10.	HTML tags are predefined tags.	XML tags are user-defined tags.
11.	There are limited number of tags in HTML.	XML tags are extensible.
12.	HTML does not preserve white spaces.	White space can be preserved in XML.
13.	HTML tags are used for displaying the data.	XML tags are used for describing the data not for displaying.

14.	In HTML, closing tags are not necessary.	In XML, closing tags are necessary.
15.	HTML is used to display the data.	XML is used to store data.
16.	HTML does not carry data it just displays it.	XML carries the data to and from the database.
17.	HTML offers native object support.	IN XML, the objects are expressed by conventions using attributes.
18.	HTML document size is relatively small.	XML document size is relatively large as the approach of formatting and the codes both are lengthy.
19.	An additional application is not required for parsing of	DOM(Document Object Model) is required for parsing JavaScript codes and mapping of text.

	JavaScript code into the HTML document.	
20.	Some of the tools used for HTML are: • Visual Studio Code • Atom • Notepad++ • Sublime Text and many more.	Some of the tools used for XML are: • Oxygen XML • XML Notepad • Liquid Studio and many more.

14. Write short note on JQuery

Introduction

- jQuery is a fast, lightweight, and feature-rich JavaScript library that is based on the principle *"write less, do more"*.
- We can do things using Jquery like

  1.HTML document traversal and manipulation,

  2.Event handling,

  3. Adding animation effects to a web page etc.

Advantages of Jquery

- **Save lots of time**
- **Simplify common JavaScript tasks**
- **Easy to use**
- **Compatible with browsers**
- **Absolutely Free**

jQuery is a popular and widely-used JavaScript library that simplifies web development by providing a fast, concise, and versatile way to manipulate HTML documents, handle events, create animations, and perform AJAX (Asynchronous JavaScript and XML) requests. Here's a short note on jQuery:

Key Features of jQuery:

Simplified DOM Manipulation: jQuery makes it easy to traverse and manipulate the Document Object Model (DOM) of a web page. It provides a simple and consistent API for selecting, modifying, and updating HTML elements.

Event Handling: jQuery simplifies event handling by providing methods to attach event listeners to elements. This makes it effortless to respond to user interactions such as clicks, keyboard events, and mouse movements.

Animation and Effects: jQuery offers built-in functions for creating smooth animations and adding effects to web elements. Developers can easily create dynamic and interactive user interfaces without writing complex animation code from scratch.

AJAX Requests: jQuery simplifies making asynchronous HTTP requests to a server, allowing web applications to retrieve data and update content without refreshing the entire page. This is crucial for building responsive and dynamic web applications.

Cross-Browser Compatibility: jQuery abstracts many browser-specific quirks and inconsistencies, ensuring that code behaves consistently across various web browsers, including older versions.

Extensibility: jQuery is highly extensible. Developers can create and use plugins to add custom functionality or leverage existing plugins from the vast jQuery plugin ecosystem.

Performance: jQuery is designed with performance in mind. It optimizes DOM traversal and manipulation operations to ensure efficient execution of scripts.

Advantages of Using jQuery:

Productivity: jQuery simplifies common tasks, reducing the amount of code required and speeding up development.

Widespread Adoption: jQuery is one of the most widely adopted JavaScript libraries, which means a wealth of documentation, tutorials, and community support is available.

Cross-Platform Compatibility: It helps ensure that web applications work consistently across different browsers and platforms.

Enhanced User Experience: jQuery's animation and effects capabilities enable the creation of visually appealing and interactive web interfaces.

Scalability: jQuery is suitable for both small-scale projects and large, complex web applications.

In summary, jQuery is a versatile and efficient JavaScript library that streamlines web development by simplifying common tasks, enhancing cross-browser compatibility, and improving user interface interactivity. It continues to be a valuable tool for web developers worldwide.