es easy-solutions

New Syllabus SPPU

easy-solutions

As per the New Credit System Syllabus (2019 course) of
Savitribai Phule Pune University w.e.f. academic year 2021-2022

WEB TECHNOLOGY

(Code : 310252)

"QUICK READ SERIES"

Semester VI
Computer Engineering

Chapterwise Solved University Paper Solution

For End Semester Examination

TechKnowledge™ Publications

easy – solutions

Savitribai Phule Pune University

As per New Credit System Syllabus(Rev. 2019) of Savitribai Phule Pune University with effective from Academic Year 2021-2022

Web Technology

(Code : 310252)

"Quick Read Series"

Semester VI - Computer Engineering

TechKnowledge™ Publications

EPE132A Price ₹ 90/-

Web Technology (Code : 310252)

(Semester VI - Computer Engineering) (SPPU)

Edition 2022

This edition is for sale in India, Bangladesh, Bhutan, Maldives, Nepal, Pakistan, Sri Lanka and designated countries in South-East Asia. Sale and purchase of this book outside of these countries is unauthorized by the publisher.

Published By

TECHKNOWLEDGE PUBLICATIONS

Printed @

37/2, Ashtavinayak Industrial Estate,
Near Pari Company,
Narhe, Pune, Maharashtra State, India.
Pune - 411041

Head Office

B/5, First floor, Maniratna Complex, Taware Colony,
Aranyeshwar Corner, Pune - 411 009.
Maharashtra State, India
Ph : 91-20-24221234, 91-20-24225678.
Email : info@techknowledgebooks.com,
Website : www.techknowledgebooks.com

Subject Code : 310252

Book code : EPE132A

SYLLABUS

In-Sem. Exam

Unit I : Web Essentials and Mark-up language- HTML **07 hrs**

The Internet, basic internet protocols, the World Wide Web, HTTP Request message, HTTP response message, web clients, web servers. HTML: Introduction, history and versions. HTML elements: headings, paragraphs, line break, colors and fonts, links, frames, lists, tables, images and forms, Difference between HTML and HTML5. CSS: Introduction to Style Sheet, CSS features, CSS core syntax, Style sheets and HTML, Style rule cascading and inheritance, text properties. Bootstrap.

Exemplar/Case Studies : Create a style sheet suitable for blogging application using HTML and using style sheet.

Unit II : Client Side Technologies: JavaScript and DOM **07 hrs**

JavaScript : Introduction to JavaScript, JavaScript in perspective, basic syntax, variables and data types, statements, operators, literals, functions, objects, arrays, built in objects, JavaScript debuggers. DOM: Introduction to Document Object Model, DOM history and levels, intrinsic event handling, modifying element style, the document tree, DOM event handling, jQuery, Overview of Angular JS.

Exemplar/Case Studies : Enhancement in created blogging application using JavaScript (Add Entry feature)

End-Sem. Exam

Unit III : Java Servlets and XML **07 hrs**

Servlet : Servlet architecture overview, A "Hello World" servlet, Servlets generating dynamic content, Servlet life cycle, parameter data, sessions, cookies, URL rewriting, other Servlet capabilities, data storage, Servlets concurrency, databases (MySQL) and Java Servlets. XML: XML documents and vocabularies, XML declaration, XML Namespaces, DOM based XML processing, transforming XML documents, DTD: Schema, elements, attributes. AJAX: Introduction, Working of AJAX.

Exemplar/Case Studies : Develop server-side code for blogging application

Unit IV : JSP and Web Services **07 hrs**

JSP : Introduction to Java Server Pages, JSP and Servlets, running JSP applications, Basic JSP, JavaBeans classes and JSP, Support for the Model-View-Controller paradigm, JSP related technologies. Web Services: Web Service concepts, Writing a Java Web Service, Writing a Java web service client, Describing Web Services: WSDL, Communicating Object data: SOAP. Struts: Overview, architecture, configuration, actions, interceptors, result types, validations, localization, exception handling, annotations.

Exemplar/Case Studies : Transform the blogging application from a loose collection of various resources (servlets, HTML documents, etc.) to an integrated web application that follows the MVC paradigm

Unit V : Server Side Scripting Languages **07 hrs**

PHP : Introduction to PHP, uses of PHP, general syntactic characteristics, Primitives, operations and expressions, output, control statements, arrays, functions, pattern matching, form handling, files, cookies, session tracking, using MySQL with PHP, WAP and WML. Introduction to ASP.NET: Overview of the .NET Framework, Overview of C#, Introduction to ASP.NET, ASP.NET Controls, Web Services. Overview of Node JS.

Exemplar/Case Studies : Use of PHP in developing blogging application.

Unit VI : Ruby and Rails **07 hrs**

Introduction to Ruby : Origins & uses of Ruby, scalar types and their operations, simple input and output, control statements, fundamentals of arrays, hashes, methods, classes, code blocks and iterators, pattern matching. Introduction to Rails: Overview of Rails, Document Requests, Processing Forms, Rails Applications and Databases, Layouts, Rails with Ajax. Introduction to EJB.

Exemplar/Case Studies : Study of dynamic web product development using ruby and rails

□□□

Table of Contents

□□□

Savitribai Phule Pune University
Semester VI (Computer Engineering)

Web Technology

Unit III : Java Servlets and XML

Q. 1 Write Advantages of Server Side Programs. **(3 Marks)**

Ans. :

1. All programs reside in one machine called the Server. Any number of remote machines (called clients) can access the server programs.
2. New functionalities to existing programs can be added at the server side which the clients' can advantage without having to change anything from their side.
3. Migrating to newer versions, architectures, design patterns, adding patches, switching to new databases can be done at the server side without having to bother about clients' hardware or software capabilities.
4. Issues relating to enterprise applications like resource management, concurrency, session management, security and performance are managed by service side applications.
5. They are portable and possess the capability to generate dynamic and user-based content (e.g. displaying transaction information of credit card or debit card depending on user's choice).

Q. 2 Write the differences between Client side scripting and server side scripting. **SPPU : Dec. 18. 4 Marks**

Ans. :

- Server-side scripting means all calculations are done by the server your website is hosted on. The script is interpreted by the supported language parser, like PHP or ASP. There are a number of types of programming things that can be done on the server. One of the primary functions you can do with code is to prepare the code that is to be sent to the Web browser.
- This includes such tasks as building pages customized for the type of browser that requested a page. It could also include doing tapping into a database to create information for a Web page. A very popular server side program is a Visitor Counter that keeps track of the number of people who have accessed a Web Site.
- The counter program would keep track of people who have come and store the information. It would also provide the actual number for any Web pages that are sent back to a browser.
- As all of the server side code is executed before the HTML is sent to the browser, your code is hidden. Server-side code is also the only choice if you want to access files and directories on the local machine. Server-side code is also browser independent. Because the HTML code returned by the server is simple HTML, you do not have to worry about the version of browser the client is using.
- The use of server-side scripting is often less visible to users. It is mostly used for content management, where the site's content is stored in a database, and presented to the user on request.

When to use server side scripts?

- Password protection.
- Browser sniffing/customization.

- Form processing.
- Building and displaying pages created from a database.

Advantages of server side scripting

- Server-side scripts are browser-independent.
- Maintaining your code becomes easier, since you don't have to modify it when new browser types or versions become available
- Global variables are available with server-side scripting.
- Web pages can be dynamically customized based on user input (e.g., browser identity, screen resolution, etc.)
- Server-side scripts provide live data through interfacing with server database.
- Server-side scripts are secured.
- You can have direct control over the security of your data. Sensitive information can be filtered out before sending the data to the client.
- For example (an extreme one), if we have an XML document that contains a list of users and passwords, there is a security risk involved in sending the whole XML document to the client to be transformed there. It would be better to filter the passwords out of the server side.

Disadvantages of server side scripting

- No debugging tools available for server-side scripting.
- No direct control over the user interface for server-side scripting.
- Server-side scripting is more difficult.
- It's hard to know when a session ends (i.e., the session variable cannot be cleared).

Q. 3 Write short note on Functionality of Servlet. **(5 Marks)**

Ans. :

A Servlets perform following functions :

1. **Read the explicit data sent by the client :** The end user normally enters this data in an HTML form on a Web page. Servlet is used to read data entered in html form. However, the data could also come from an applet.
2. **Read the implicit HTTP request data sent by the browser :** There are two varieties of data the explicit data and implicit data.

   Explicit data is data that the end user enters in a form and implicit data is additional information that is embedded in the HTTP request. The HTTP information includes cookies, information about media types and compression schemes the browser understands. Servlet reads explicit data and implicit data.
3. **Generate the results :** This process may require talking to a database, executing an RMI or EJB call, invoking a Web service, or computing the response directly. For most of application servlet talk to database and compute the result.
4. **Send the implicit HTTP response data :** There are really two varieties of data sent from the Web middle layer to the client, the document itself and HTTP response information. Sending HTTP response data involves telling the browser or other client what type of document is being returned (e.g., HTML), setting cookies and caching parameters, and other such tasks.
5. **Send the explicit data (i.e., the document) to the client :** The document can be sent in a variety of formats, including text (HTML or XML), binary (GIF images).But, HTML is the most common format.

Q. 4 How Does Servlet Works. **(3 Marks)**

Ans. :

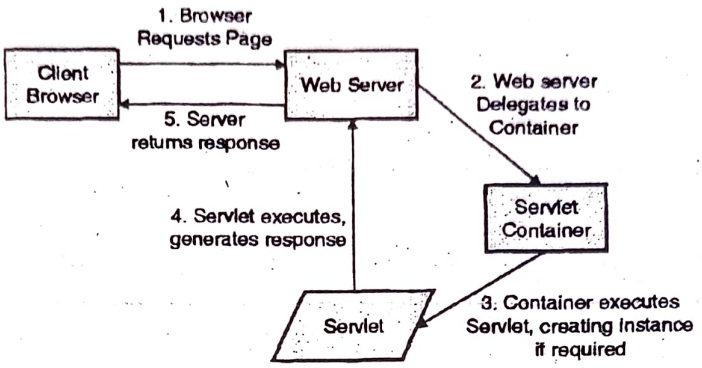


Fig. 3.1 : Working of servlet

- When a user (client browser) requests a page that is a Servlet, the request is first processed by the web server.
- The web server redirects the request to the Servlet container. The container will create an instance of the appropriate Servlet (if one doesn't already exist) and pass a request and a response object into the Servlet's service() method.
- The Servlet will then execute either the doGet() or doPost() method and create the HTML output. The container passes the output back to the web server through the response object, then on to the client browser.

Q. 5 Explain HttpServlet Request and HttpServlet Response with suitable examples. **SPPU : Dec.18, May 19, 5 Marks**

Ans. :

HttpServlet Request

- Servlet API provides two important interfaces **javax.servlet.ServletRequest** and **javax.servlet.http.HttpServletRequest** to encapsulate client request.
- Implementation of these interfaces provide important information about client request to a servlet **HttpServletRequest** interface adds the methods that relates to the **HTTP** protocol. The servlet container creates an HttpServletRequest object and passes it as an argument to the servlet's service methods (doGet, doPost, etc.).

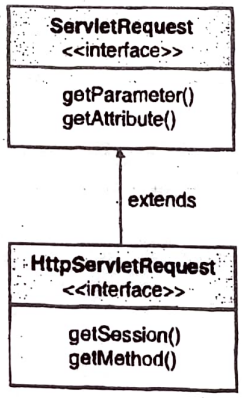


Fig. 3.2(a) : HttpServletRequest

- Some important methods of HttpServletRequest

Table 3.1

Methods	Description
String getContextPath()	Returns the portion of the request URI that indicates the context of the request
Cookies getCookies()	Returns an array containing all of the Cookie objects the client sent with this request
String getQueryString()	Returns the query string that is contained in the request URL after the path
HttpSession getSession()	Returns the current HttpSession associated with this request or, if there is no current session and create is true, returns a new session
String getMethod()	Returns the name of the HTTP method with which this request was made, for example, GET, POST, or PUT.
Part getPart(String name)	Gets the Part with the given name.
String getPathInfo()	Returns any extra path information associated with the URL the client sent when it made this request.
String getServletPath()	Returns the part of this request's URL that calls the servlet.

HttpServlet Response

- Servlet API provides two important interfaces **ServletResponse** and **HttpServletResponse** to assist in sending response to client. **HttpServletResponse** interface adds the methods that relates to the **HTTP** response.

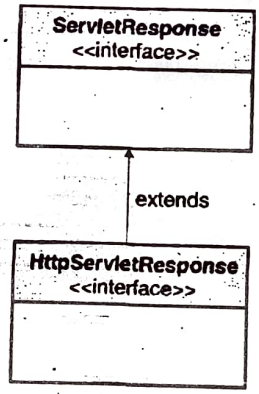


Fig. 3.2 (b)

- Some Important Methods of HttpServletResponse

Table 3.2

Methods	Description
void addCookie(Cookie cookie)	Adds the specified cookie to the response.
void sendRedirect(String location)	Sends a temporary redirect response to the client using the specified redirect location URL and clears the buffer.
int getStatus()	Gets the current status code of this response.
String getHeader(String name)	Gets the value of the response header with the given name.
void setHeader(String name, String value)	Sets a response header with the given name and value.
void setStatus(intsc)	Sets the status code for this response.
void sendError(intsc, String msg)	Sends an error response to the client using the specified status and clears the buffer.

Q. 6 Write a simple servlet to print "Hello,Good Day". **SPPU : Dec. 19, 5 Marks**

Ans. :

- A servlet is an instance of the class that implements the javax.servlet.Servlet interface.
- But almost all servlet extends javax.servlet.GenericServlet and javax.servlet.httpServlets implementation of the above interface.
- Servlet code that implements HttpServlet interface and return response in HTML format. Imported java.io class for PrintWriter javax.servlet class for HttpServlet, and javax.servlet.http class for HttpServletRequest and HttpServletResponse. Hello servlet extends the **HttpServlet** and overrides the doGet()method which it inherits from the HttpServlet class.
- The server invokes doGet() method whenever *web server* receives the GET request from the servlet. The doGet() method takes two arguments first is HttpServletRequest object and the second one is HttpServletResponse object and this method throws the ServletException.
- Whenever the user sends the request to the server then server generates two obects, first is HttpServletRequest object and the second one is HttpServletResponse object. HttpServletRequest object represents the client's request and the HttpServletResponse represents the servlet's response.
- Inside the doGet() method our servlet has first used the setContentType()method of the response object which sets the content type of the response to text/html. It is the standard MIME content type for the Html pages. After that it has used the method getWriter() of the response object to retrieve a **PrintWriter** object. To display the output on the browser we use the println() method of the **PrintWriter** class.

Q. 7 Explain the lifecycle of Servlet. **SPPU : May 16, Dec.16, Dec.19, 6 Marks**

Ans. : Lifecycle of Servlet

The servlet lifecycle consists of a series of events, which define how the servlet is loaded and instantiated, initialized, how it handles requests from clients, and how is it taken out of service.

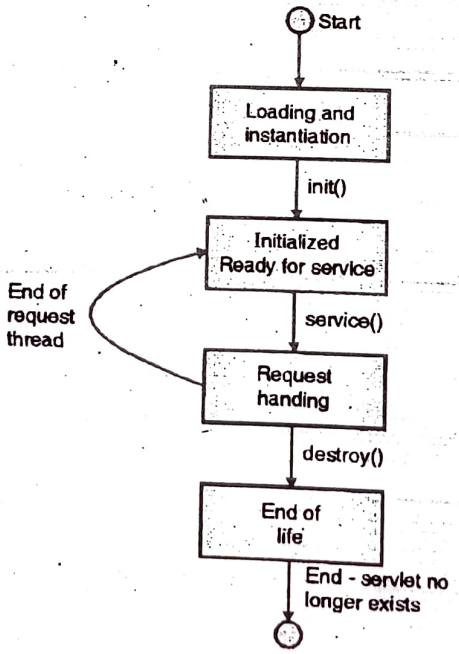


Fig. 3.3 : Servlet life cycle

1. Loading and instantiation

For each servlet defined in the deployment descriptor of the Web application, the servlet container locates and loads a class of the type of the servlet. The loading of the servlet depends on the attribute <load-on-startup> of web.xml file. If the attribute <load-on-startup> has a positive value then the servlet is load with loading of the container otherwise it load when the first request comes for service. After loading of the servlet, the container creates the instances of the servlet.

2. Initialization

After instantiation, the container initializes a servlet before it is ready to handle client requests. The container initializes the servlet by invoking its init() method, passing an object implementing the ServletConfig interface. In the init() method, the servlet can read configuration parameters from the deployment descriptor or perform any other one-time activities, so the init() method is invoked once and only once by the servlet container.

3. Request handling

After the servlet is initialized, the container may keep it ready for handling client requests. When client requests arrive, they are delegated to the servlet through the service()method, passing the request and response objects as parameters. Servlet creates separate threads for each request. The sevlet container calls the service() method for servicing any request . In the case of HTTP requests, the request and response objects are implementations of HttpServletRequest and HttpServletResponse respectively. In the HttpServlet class, the service() method invokes a different handler method for each type of HTTP request, doGet() method for GET requests, doPost() method for POST requests, and so on.

4. Removal from service

A servlet container may decide to remove a servlet from service for various reasons, such as to conserve memory resources. To do this, the servlet container calls the destroy() method on the servlet. Like the init() method this method is also called only once throughout the life cycle of the servlet.

Calling the destroy() method indicates to the servlet container not to sent the any request for service and the servlet releases all the resources associated with it. Java Virtual Machine claims for the memory associated with the resources for garbage collection. Once the destroy() method has been called, the servlet may not service any more client requests.

Q. 8 Write a servlet which will accept username and password in a form, which will compare both in the code to display successor failure. **SPPU : March 18, Dec. 19, March 19, 5 Marks**

Ans. :

To read a request (form) parameter, you simply call the get Parameter method of HttpServletRequest, supplying the case-sensitive parameter name as an argument. You supply the parameter name exactly as it appeared in the HTML source You can also call request.getParameterValues if the parameter appears more than once, or you can call request.getParameterNames if you want a complete list of all parameters in the current request.

Login.html

```
<html>
<head>
<title> form data </title>
</head>
<body>
<p>Please enter your username and password</p>
```

```
<form method="GET" action= "/htmlform/LoginServlet">
<p> Username <input type="text" name="username" size="20"> </p>
<p> Password <input type="text" name="password" size="20"></p>
<p><input type="submit" value="Submit" name="B1"></p>
</form>
</body></html>
```

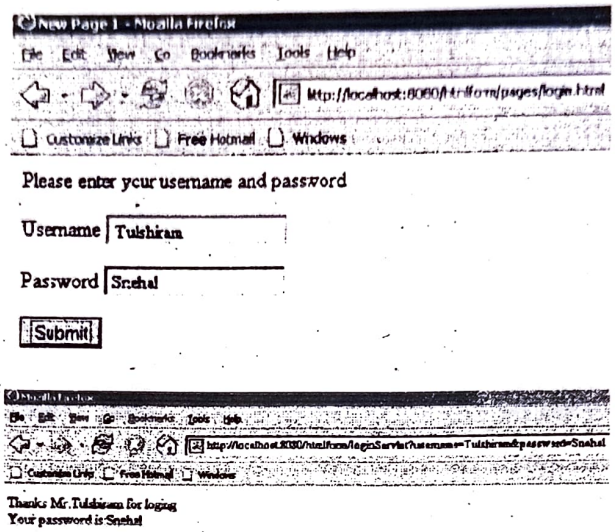
LoginServlet.java

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class LoginServlet extends HttpServlet
{
    public void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException
    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
String name = request.getParameter("username");
String pass = request.getParameter("password");
        out.println("<html>");
        out.println("<body>");
out.println("Thanks Mr." + " " + name + " " + "for loging <br>");
        out.println(" Your password is : " + " " + pass + "<br>");
        out.println("</body></html>");
    }
}
```

web.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!--<!DOCTYPE web-app
 PUBLIC "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
 "http://java.sun.com/dtd/web-app_2_3.dtd"> -->
 <web-app>
  <servlet>
  <servlet-name>Hello</servlet-name>
  <servlet-class>LoginServlet</servlet-class>
  </servlet>
  <servlet-mapping>
  <servlet-name>Hello</servlet-name>
  <url-pattern>/LoginServlet</url-pattern>
  </servlet-mapping>
 </web-app>
```

Output :



Q. 9 Write a Java Servlet which will display "welcome to Servlet" message. **SPPU - March 19, May 19, 5 Marks**

Ans. :

Writing Servlet to display welcome to Servlet :

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class Hello extends HttpServlet
{
    public void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException,IOException
    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head><title>Hello </title></title>");
        out.println("<body>");
        out.println("<h1>welcome to Servlet </h1>");
        out.println("</body></html>");
    }
}
```

Q. 10 Explain Session and Cookies in Servlets? Write any one program to demonstrate session or cookies? **SPPU : Dec. 19, 5 Marks**

Ans. : Cookies

- Cookies are used a lot in web applications to personalize response based on your choice or to keep track of session.

- Cookies are the mostly used technology for session tracking. Cookie is a key value pair of information, sent by the server to the browser. This should be saved by the browser in its space in the client computer. Whenever the browser sends a request to that server it sends the cookie along with it. Then the server can identify the client using the cookie
- The method to program cookies is different for different languages.
- Most of the language provides some class that covers all the details of cookie creation and maintenance. For example in Java you have a *javax.servlet.http.Cookie* class that is used to work with cookies

Sending Cookies to the Client

To send cookies to the client, a servlet should use the Cookie constructor to create cookies with designated names and values and insert the cookies into the HTTP response headers with response.addCookie.

Creating a Cookie object

- You create a cookie by calling the Cookie constructor, which takes two strings : the cookie name and the cookie value.
- Cookie userCookie = new Cookie("tuls", "s1212");
- Here we have created a cookie named tuls with a value s1212

Setting the maximum age

Setting maximum age is optional attribute. By default cookies are session-level cookie i.e. a cookie that is stored in the browser's memory and deleted when the user quits the browser. If you want the browser to store the cookie on disk, use setMax-Age with a time in seconds.

userCookie.setMaxAge(60*60*24); //for one day

Place the Cookie into the HTTP response headers.

You use of the add Cookie method of HttpServletResponse response.addCookie(userCookie);

- To read incoming cookies, a servlet should call request.getCookies, which returns an array of Cookie objects associated with your site, then servlet should then loop down this array calling get Name on each cookie until it finds the one whose name matches the name it was searching for, then call get Value on that Cookie to see the value associated with the name.

Advantages

1. Cookies do not require any server resources since they are stored on the client.
2. Session tracking is easy to implement and maintain using the cookies
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).

Disadvantages

1. Users can delete a cookies.
2. The users can disable cookies using their browser preferences , In such case, the browser will not save the cookie at client computer and session tracking fails
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

Session management API

Session Management API is built on top of above methods for session tracking. Some of the major disadvantages of all the above methods are:

- Most of the time we don't want to only track the session, we have to store some data into the session that we can use in future requests. This will require a lot of effort if we try to implement this.
- All the above methods are not complete in themselves, all of them won't work in a particular scenario. So we need a solution that can utilize these methods of session tracking to provide session management in all cases.
- That's why we need **Session Management API** and J2EE Servlet technology comes with session management API that we can use.
- Servlet API provides Session management through HttpSession interface. We can get session from HttpServletRequest object using following methods. HttpSession allows us to set objects as attributes that can be retrieved in future requests.
- **HttpSessiongetSession()** – This method always returns a HttpSession object. It returns the session object attached with the request, if the request has no session attached, then it creates a new session and return it.
- **HttpSessiongetSession(boolean flag)** – This method returns HttpSession object if request has session else it returns null.
- When we use HttpServletRequestgetSession() method and it creates a new request, it creates the new HttpSession object and also add a Cookie to the response object with name *JSESSIONID* and value as session id.
- This cookie is used to identify the HttpSession object in further requests from client. If the cookies are disabled at client side and we are using URL rewriting then this method uses the jsessionid value from the request URL to find the corresponding session.
- *JSESSIONID* cookie is used for session tracking, so we should not use it for our application purposes to avoid any session related issues

---

Q. 11 Explain any two from the following JDBC API components :

i) Import the packages
ii) Open Database Connection
iii) Display Query result

SPPU : Dec. 19, 5 Marks

Ans. : Following steps are necessary to create and manage a database connection using JDBC.

1.	Import the Package	2.	Load the JDBC Driver
3.	Define Connection URL	4.	Open Database Connection
5.	Create Statement Object	6.	Execute Query
7.	Display Query result	8.	Close the connection

1. Import the packages

- In all Java applications you must import the packages that contain the classes you need for your Program.. JDBC application requires that you include the packages containing the JDBC classes needed for database programming.
- All the JDBC interfaces and classes exist in either the java.sql or javax.sql package. The java.sql package contains the JDBC core libraries where as javax.sql package contains the classes and interfaces to support enterprise level JDBC programming such as distributed transactions, connection pooling etc.

2. Establish (open) database the Connection

- Most database applications operate in a client-server environment. JDBC applications act as clients and to work with the server they must establish a physical connection to it. This is true regardless of whether the application resides on the database host or on a different host.

- To make the connection, pass the URL, database username, and database password to the getConnection method of the DriverManager class. For database that don't require explicit logins, the username and password string should be left blank .Once the connection is established, database queries can be performed until the connection is closed

```
String username = "sakshi";
String password = "sule44";
Connection conn =DriverManager.getConnection(oracleURL, username, password);
```

- An optional part of establishing the connection is to look up information about the database with the getMetaData method. This method returns a DatabaseMeta- Data.
- Object that has methods with which w can find out name and version of the database (getDatabaseProductName, getDatabaseProductVersion) or name and version of the JDBC driver (getDriverName, get DriverVersion).

3. Display the query results

- ResultSet object holds the data returned by an SQL SELECT statement. It stores the data as a table of rows and columns. The ResultSet object uses a cursor to point to rows in the result set. To access the result set data, you must move the cursor from row to row and retrieve the data from each row's columns. Moving through the result set is simple: Just call the ResultSet.next() method, which advances the cursor to the next row.
- Within a row, ResultSet provides various get*Xxx* methods that take a column name or column index as an argument and return the result in a variety of different Java types. For instance, use getInt if the value should be an integer, getString for a String, and so on for most other data types. However, if you use the version of get*Xxx* that takes a column index (rather than a column name), note that columns are indexed starting at 1 (following the SQL convention), not at 0 as with arrays.

```
while(rs.next()){
//Retrieveing data by using column name
int rno= rs.getInt("rno");
String name = rs.getString("name");

//Retrieving data by using column index
double fee = rs.getDouble(3);
//Display values
System.out.print("RNO: " + rno);
System.out.print(" Name: " + name);
System.out.print(" Fee: " + fee);
}
```

- In above example we use a while loop with the rs.next() method as the conditional statement. The method returns true when the cursor moves into a valid row. We retrieve data from the columns by calling the ResultSet.get*XXX*() method, where *XXX* maps to the Java data type of the variable i.e. we have used integer for roll number and string for name. Next we have retrieve fee by using column index then display Roll number, name and fee.

Q. 12 Write short note on Servlet Concurrency. **(5 Marks)**

Ans. : Servlet Concurrency

A Java servlet container / web server is typically multithreaded. That means, that multiple requests to the same servlet may be executed at the same time. Therefore, you need to take concurrency into consideration when you implement your servlet.By default servlets are not thread safe and it is a responsibility of a servlet developer to take care of it. there are a few basic rules of thumb you must follow:

1. Your servlet service() method should not access any member variables, unless these member variables are thread safe themselves.
2. Your servlet service() should not reassign member variables, as this may affect other threads executing inside the service() method. If you really, really need to reassign a member variable, make sure this is done inside a synchronized block.
3. Rule 1 and 2 also counts for static variables.
4. If you have a requirement which requires modification of instance variable then do it in a synchronized block.
5. Local variables are always thread safe. Keep in mind though, that the object a local variable points to, may not be so. If the object was instantiated inside the method, and never escapes, there will be no problem. On the other hand, a local variable pointing to some shared object, may still cause problems. Just because you assign a shared object to a local reference, does not mean that object automatically becomes thread safe.

A code example shows some of the rules

```
public class SimpleHttpServlet extends HttpServlet {

 // Not thread safe, static.
protected static List list = new ArrayList();

 // Not thread safe
protected Map map = new HashMap();

 // Thread safe to access object, not thread safe to reassign variable.
protected Map map = new ConcurrentHashMap();

 // Thread safe to access object (immutable), not thread safe to reassign variable.
protected String aString = "a string value";

 protected void doGet( HttpServletRequest request, HttpServletResponse response)
     throws ServletException, IOException {
  // Not thread safe, unless the singleton is 100% thread safe.
SomeClass.getSomeStaticSingleton();

  // Thread safe, locally instantiated, and never escapes method.
Set set = new HashSet();
 }
}
```

Q. 13 What is XML ? SPPU : May 17, Dec.19, 4 Marks

Ans. : XML

- XML stands for Extensible Markup Language. XML is text based markup language that enables you store data in a structured format by using meaningful tags. The term Extensible implies that you can extend your ability to describe document by defining meaningful tags for your application.
- XML was designed to carry data, not to display data. "XML is a cross-platform, software and hardware independent tool for transmitting information and therefore it is used to transfer structured documents (data) between heterogeneous systems. It is particularly useful for application over Internet.
- In XML, tags are not predefined. A user defines his own tags and XML document structure like Document Type Definition (DTD), XML Schema to describe the data. Hence it is self-descriptive too.

Q. 14 What are the strengths of XML technology ? Explain the need for XML. SPPU : May 17, Dec. 18, March 19, 5 Marks

Ans. : Strengths of XML technology

There are number of reasons that contributes to the XML's increasing acceptance :

1. **Plain Text :** As XML is plain text nature, no special editor application or document processor are required to create or maintain XML document.
2. **Scalability :** XML also provides scalability for anything from small configuration files to a company-wide data repository. i.e. XML is Useful for storing small amounts of data and also used to store large amounts of XML data through an XML front end to a database.
3. **Extensible :** HTML limits the document author to fixed set of tags. XML allows document author to describe data by creating new tags.
4. **Data Identification :** The markup tags in XML documents identify the information and break up the data into parts for example. a search program can look for messages sent to particular people from the rest of the message. Different parts of the information are identified and further they can be used in different ways by different applications.
5. **Universally Processed :** Apart from being valid, restrictions are imposed on a xml file to abide by a DTD or a Schema to make it well-formed .Otherwise, the XML parser won't be able to read the data. XML is a vendor-neutral standard, so a user can choose among several XML parsers to process XML data.
6. **Hierarchical Approach :** XML documents get benefited from their hierarchical structure. Hierarchical document structures are, faster to access. They are also easier to rearrange, because each piece is delimited. This makes xml files easy to modify and maintain.
7. **Platform independent:** XML is a cross-platform, software and hardware independent markup language and therefore it is used to transfer structured documents (data) between heterogeneous systems.

   It is used in application like data interchange, B2B E-commerce etc.
8. **Stylability:** When display matters, the stylesheet standard, XSL (an advance feature of XML), lets you dictate over the convectional designs (like using HTML) to portray the data. XML being style-free uses different stylesheets to produce output in postscript, TEX, PDF, or some new format that hasn't even been invented yet. A user can use a simple XML document to display data in diverse formats like,
   - A plain text file.
   - An XHTML file.
   - A WML (Wireless Markup Language) document suitable for display on a PDA.
   - An Adobe PDF document suitable for hard copy.
   - A VML (Voice Markup Language) dialog for a voicemail information system.

9. **XML is Used to Create New Internet Languages :** A lot of new Internet languages are created with XML. Here are some examples:
   - WSDL for describing available web services.
   - WAP and WML as markup languages for handheld devices.
   - RSS languages for news feeds.
   - RDF and OWL for describing resources and ontology.
   - SMIL for describing multimedia for the web XHTML the latest version of HTML.

Limitations of XML

1. XML syntax is verbose and redundant compared to other text-based data transmission formats such as JSON.
2. The redundancy in syntax of XML causes higher storage and transportation cost when the volume of data is large.
3. XML document is less readable compared to other text-based data transmission formats such as JSON.
4. XML file sizes are usually very large due to its verbose nature, it is totally dependant on who is writing it.

Q. 15 Write difference between HTML and XML. **(5 Marks)**

Ans. :

Sr. No.	HTML	XML
1.	It is used to display data and control how data is displayed (define data). HTML was designed to display data and to focus on how data looks.	XML is used to describe data and focus on what data is (describe data).
2.	HTML tags are predefined.	XML tags are not predefined.
3.	HTML tags are Case Sensitive.	XML tags are Case Sensitive.
4.	It is not mandatory to close tag.	It is mandatory to close tag.
5.	Stylesheet for HTML are optional.	Stylesheet for XML called XSL are compulsory for formatting of data.
6.	Some HTML elements can be improperly nested within each other.	XML Elements Must be Properly Nested.
7.	HTML is static in nature.	XML is dynamic in nature.
8.	HTML is presentation language.	XML is neither programming nor presentation language.

Q. 16 What are Components of XML Document. **(5 Marks)**

Ans. : Components of XML Document

An XML document is composed of a number of components that can be used for representing information in a hierarchical order. XML components are

1. XML declaration
2. Tags
3. Elements
4. Attributes
5. Comments

Develop a simple XML document :

<?xml version="1.0" encoding="ISO-8859-1"?><E-mail><To>Minal</To><From>Tulshiram</From><Subject>About holiday</Subject><Body>I have holiday tomorrow so i will catch u tonight</Body></E-mail>

1. XML declaration

- An XML document usually begins with XML document declaration statement. An XML declaration declares the version of XML document used to define the document. It may also indicate the character encoding used to store or transfer the document.
- It is written as

<?xml version="1.0" ?>

OR

<?xml version="1.0" encoding="UTF-8"?>

- XML declaration is optional but document without an XML declaration might be assumed to conform to latest version of XML.

2. Tags

- Tags are means of identifying data. Data is marked up using tags. Tags consist of opening and closing angular bracket (<>).These bracket enclose name of tags, usually tags occurs in pairs i.e. start tags and end tags.
- These are of three types :
  1. Start tag (Ex. <book-name>)
  2. End tag (Ex. </book-name>)
  3. Empty-element tag This is also known as bodyless tag. It has a start tag but does not have a matching end tag. E.g <book-image src="images/myxml.gif"/>
- Every start tag must have an end tag e.g tuls
- Empty tags must be closed using forward slash (/). In empty tags do not contain any information, however they can contain attribute. Value of these attribute is specified within opening and closing angular bracket.

  E.g. <image name= "tuls.jpg" />
- Here image is empty element and contain attribute name.

Q. 17 Write DOM vs SAX. **(5 Marks)**

Ans. :

SAX Parser	DOM Parser
It is called a Simple API for XML Parsing.	It is called as Document Object Model.
It's an event-based parser.	It stays in a tree structure.
SAX Parser is slower than DOM Parser.	DOM Parser is faster than SAX Parser.
Best for the larger sizes of files.	Best for the smaller size of files.
It is suitable for making XML files in Java.	It is not good at making XML files in low memory.
The internal structure can not be created by SAX Parser.	The internal structure can be created by DOM Parser.
It is read-only.	It can insert or delete nodes.
In the SAX parser backward navigation is not possible.	In DOM parser backward and forward search is possible
Suitable for efficient memory.	Suitable for large XML document.
A small part of the XML file is only loaded in memory.	It loads whole XML documents in memory.

Q. 18 Discuss the XSLT technology with an example. **SPPU : May 19, 5 Marks**

Ans. :

XSLT technology

- The greatest flexibility of XML is that it has no rules for display. It is all content oriented. However, this strength means that there needs to be style information included in an XML document. This can be HTML or style sheets, or XSL (eXtensible Stylesheet Language).
- When you create an HTML document, you use tags such as <table> and . Then, when that page is viewed in a browser, the browser knows how to display the content in those elements. But when you write an XML document, the browser doesn't necessarily know how to display those same elements.
- The <table> element might be defining an HTML table, or it might be defining a coffee table, the browser displaying the HTML has no idea how to display the elements. XSL Provides Formatting for XML.
- XSL is the intermediary between the XML elements and the browser. This is the language that tells the browser how to display the various elements, such as <table>.
- An XSL file is a style sheet that can be used to transform XML documents into other document types and to format the output. Since the XML language does not use predefined tags, it is necessary to provide the Web browser with information on how to interpret the XML document. The XSL document provides the browser with information on how to display an XML document.
- XSL is a family of recommendations for defining XML document transformation and presentation. It consists of three parts :
  1. **XSLT :** This language takes existing XML documents and transforms them into other XML documents.
     It allows an XML author to write content only once and put it in many different formats.
  2. **XPath :** The XPath language defines parts of an XML document. This allows XML authors to link to very specific locations within an XML document.
     It is an expression language used by XSLT to access or refer to parts of an XML document
  3. **XSL Formatting Objects:** Extensible Stylesheet Language Formatting Objects (XSL-FO) focuses on style that works for both screen and print, such as PDF files. XSL-FO is an XML vocabulary for specifying formatting semantics.
- XSL gives a developer the tools to describe exactly which data fields in an XML file to display and exactly where and how to display them. xml-stylesheet is a special declaration in XML for linking XML with stylesheets. Place this after your XML declaration to link your XML file to your XSLT code.
- xml-stylesheet has two attributes :
  1. **type :** The type of file being linked to. We will be using the value text/xsl to specify XSLT.
  2. **href :** The location of the file. If you saved your XSLT and XML file in the same directory, you can simply use the XSLT filename.

```
<?xml-stylesheet type="text/xsl" href="class.xsl"?>
```

- This line indicates that the this XML document should be transformed using class.xsl.

Q. 19 Write Differences between XML and XSLT. SPPU : Dec. 19, 5 Marks

Ans. :

Sr. No	XML	XSLT
1	XML is used for storing data in a structured format.	XSLT is used for transforming and also for formatting XML file to display the content or to process the content.
2	XML does not perform transformation of data.	XSLT is a specialized language that performs transformation of one XML document into a different XML document. XSLT can also perform transformation of XML document into HTML document and XHTML document.
3	XPath is a specialized language that is used to address portions of the XML file.	XSLT will be using XPath for transformation and formatting of XML file.

Q. 20 What are the DTDs? Explain how do they work. SPPU : May 18, 2 Marks

Ans. :

DTDs

- A DTD is a Document Type Definition. A DTD defines the structure and the legal elements and attributes of an XML document.
- DTDs check the validity of structure and vocabulary of an XML document against the grammatical rules of the appropriate XML language. With a DTD, independent groups of people can agree on a standard DTD for interchanging data. An application can use a DTD to verify that XML data is valid.
- An XML DTD can be either specified inside the document, or it can be kept in a separate document and then liked separately.
- DTD defines following three rules.
  1. Specifies the tags and attributes that can be used to creating XML document.
  2. How to tags combines and reuse.
  3. Specifies the entities which are represent the special characters.
- Basic syntax of a DTD is as follows :

```
<!DOCTYPE element DTD identifier
[
  declaration1
  declaration2
  .....
]>
```

In the above syntax,

- The DTD starts with <!DOCTYPE delimiter.
- An element tells the parser to parse the document from the specified root element.
- DTD identifier is an identifier for the document type definition, which may be the path to a file on the system or URL to a file on the internet. If the DTD is pointing to external path, it is called External Subset.
- The square brackets [ ] enclose an optional list of entity declarations called Internal Subset.

When to use a DTD?

- With a DTD, independent groups of people can agree to use a standard DTD for interchanging data.
- With a DTD, you can verify that the data you receive from the outside world is valid.
- You can also use a DTD to verify your own data.

Advantages of using DTD

- Documentation – You can define your own format for the XML files. Looking at this document a user/developer can understand the structure of the data.
- Validation – It gives a way to check the validity of XML files by checking whether the elements appear in the right order, mandatory elements and attributes are in place, the elements and attributes have not been inserted in an incorrect way, and so on.

Q. 21 Explain the differences between external and internal DTDs. **SPPU : March 18, 5 Marks**

Ans. :

Internal DTD	External DTD
A DTD is referred to as an internal DTD if elements are declared within the XML files.	In external DTD elements are declared outside the XML file.
To refer it as internal DTD, *standalone* attribute in XML declaration must be set to **yes.**	To refer it as external DTD, *standalone* attribute in the XML declaration must be set as **no**. This means, declaration includes information from the external source.
.You can write rules inside XML document using **<!DOCTYPE ... >** declaration. Scope of this DTD within this document. Advantages is document validated by itself without external reference.	You can write rules in a separate file (with .dtd extension). later this file linked to a XML document. This way you can linked several XML documents refer same DTD rules.
The syntax of internal DTD is as shown – <!DOCTYPE root-element [element-declarations]> where *root-element* is the name of root element and *element-declarations* is where you declare the elements.	Following is the syntax for external DTD – <!DOCTYPE root-element SYSTEM "file-name"> where *file-name* is the file with *.dtd* extension.

Q. 22 What are XML schemas? How are they better than DTDs ? **SPPU : March 18, 5 Marks**

Ans. : XML schemas

- XML Schema is commonly known as XML Schema Definition (XSD). It is used to describe and validate the structure and the content of XML data. XML schema defines the elements, attributes and data types. Schema element supports Namespaces. XML Schema is an XML-based (and more powerful) alternative to DTD.
- You need to declare a schema in your XML document as follows :

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
```

- One of the greatest strength of XML Schemas is the support for data types.
  - It is easier to describe allowable document content.
  - It is easier to validate the correctness of data.
  - It is easier to define data facets (restrictions on data).
  - It is easier to define data patterns (data formats).
  - It is easier to convert data between different data types.

Defination of XML schema elements in following ways :

1. **Simple Type :** Simple type element is used only in the context of the text. Some of predefined simple types are : xs:integer, xs:boolean, xs:string, xs:date. For example :

```
<xs:element name="phone_number" type="xs:int" />
```

2. **Complex Type :** A complex type is a container for other element definitions. This allows you to specify which child elements an element can contain and to provide some structure within your XML documents. For example :

```
<xs:element name="Address">
<xs:complexType>
<xs:sequence>
<xs:element name="name" type="xs:string" />
   <xs:element name="company" type="xs:string" />
<xs:element name="phone" type="xs:int" />
</xs:sequence>
</xs:complexType>
</xs:element>
```

In the above example, *Address* element consists of child elements. This is a container for other <xs:element> definitions, that allows to build a simple hierarchy of elements in the XML document.

3. **Global Types :** With the global type, you can define a single type in your document, which can be used by all other references.

For example, suppose you want to generalize the *person* and *company* for different addresses of the company. In such case, you can define a general type as follows –

```
<xs:element name = "AddressType">
<xs:complexType>
<xs:sequence>
<xs:element name = "name" type = "xs:string" />
<xs:element name = "company" type = "xs:string" />
</xs:sequence>
</xs:complexType>
</xs:element>
```

Use this type in our example as follows –

```
<xs:element name = "Address1">
<xs:complexType>
<xs:sequence>
<xs:element name = "address" type = "AddressType" />
<xs:element name = "phone1" type = "xs:int" />
</xs:sequence>
</xs:complexType>
</xs:element>
<xs:element name = "Address2">
<xs:complexType>
<xs:sequence>
<xs:element name = "address" type = "AddressType" />
```

```
<xs:element name = "phone2" type = "xs:int" />
</xs:sequence>
</xs:complexType>
</xs:element>
```

- Instead of having to define the name and the company twice (once for *Address1* and once for *Address2*), now have a single definition. This makes maintenance simpler, i.e., if you decide to add "Postcode" elements to the address, you need to add them at just one place.

XML schemas are better than DTDs

- There are many technical benefits of migrating older DTDs to XML Schema, including :
  - Support for primitive (built-in) data types (eg: xsd:integer, xsd:string, xsd:date, and so on), which facilitates using XML in conjunction with other typed-data, including relational data.
  - The ability to define custom data types, using object-oriented data modeling principles: encapsulation, inheritance, and substitution.
  - Compatibility other XML technologies, for example, Web services, XQuery, XSLT and other technologies can optionally be schema-aware.
  - DTD provides less control on XML structure whereas XSD (XML schema) provides more control.

Q. 23 Differentiate between DTD and XSD. **(5 Marks)**

Ans. :

Sr. No.	DTD	XSD
1.	DTD stands for Document Type Definition.	XSD stands for XML Schema Definition.
2.	DTDs are derived from SGML syntax.	XSDs are written in XML.
3.	DTD doesn't support datatypes.	XSD supports datatypes for elements and attributes.
4.	DTD doesn't support namespace.	XSD supports namespace.
5.	DTD doesn't define order for child elements.	XSD defines order for child elements.
6.	DTD is not extensible.	XSD is extensible.
7.	DTD is not simple to learn.	XSD is simple to learn because you don't need to learn new language.
8.	DTD provides less control on XML structure.	XSD provides more control on XML structure.

Q. 24 What is purpose of AJAX ? **SPPU : Dec. 19, 4 Marks**

Ans. :

Purpose of AJAX

- Asynchronous JavaScript and XML or Ajax for short is new web development technique used for the development of most interactive website.
- Ajax helps you in making your web application more interactive by retrieving small amount of data from web server and then showing it on your application. You can do all these things without refreshing your page.

- Usually in all the web applications, the user enters the data into the form and then clicks on the submit button to submit the request to the server. Server processes the request and returns the view in new page (by reloading the whole page).
- This process is inefficient, time consuming, and a little frustrating for you user if the only the small amount of data exchange is required. For example in an user registration form, this can be frustrating thing for the user, as whole page is reloaded only to check the availability of the user name.
- Ajax will help in making your application more interactive. With the help of Ajax you can tune your application to check the availability of the user name without refreshing the whole page.
- Examples of applications using AJAX : Google Maps, Gmail, Youtube, and Facebook tabs.

Q. 25 Find out and explain how XML and AJAX are related. **SPPU : Dec. 18, 4 Marks**

Ans. : Ajax is not a single technology, but it is a combination of many technologies. These technologies are supported by modern web browsers. Following are techniques used in the Ajax applications.

1. HTML/XHTML and CSS	2. DOM
3. XMLHttpRequest	4. XML or JSON
5. JavaScript	

1. **HTML/XHTML and CSS :** These technologies are used for displaying content and style. It is mainly used for presentation. CSS (Cascading Style Sheet) is used in web site for designing purpose.
2. **DOM :** The DOM (Document Object Model) is the object oriented representation of XML and HTML documents, and provides an API for changing the content, structure, and style. The DOM represents HTML and XML documents as object hierarchy, which is easy to parse by XML tools.
3. **XMLHttpRequest :** Most of the Ajax application used the XMLHttpRequest object to send the request to the web server. These calls are Asynchronous and there is no need to wait for the response to come back. User can do the normal work without any problem. XMLHttpRequest : Unlike other usual web pages, with AJAX, JavaScript communicates with server using JavaScript's XMLHttpRequest object. With the help of XMLHttpRequest a web page can send request and get a response from the server without refreshing the page. This object is supported by all the leading web browsers.
4. **XML or JSON :** XML may be used to receive the data returned from the web server. JavaScript can be used to process the XML data returned from the web server easily. JSON (Javascript Object Notation) is like XML but short and faster than XML.
5. **JavaScript :** JavaScript is used to make a request to the web server. Once the response is returned by the webserver, more JavaScript can be used to update the current page. DHTML and CSS is used to show the output to the user. JavaScript is used very heavily to provide the dynamic behavior to the application.

Benefits of Ajax

- Ajax can be used for creating rich, web-based applications that look and works like a desktop application Ajax is easy to learn. Ajax is based on JavaScript and existing technologies like XML, CSS, DHTML, etc. So, its very easy to learn Ajax.
- Ajax can be used to develop web applications that can update the page data continuously without refreshing the whole page.
- The biggest advantage of Ajax is its bandwidth usage because it generates HTML locally within the browser and does not require reloading the whole data after amending any portion of the programming.
- Its speedy response and tendency to share the client's system for accessing content reduces the bandwidth consumption for web applications and server load.

Q. 26 Draw and explain how AJAX works with the help of suitable example. **SPPU : May 18, Dec.19, 6 Marks**

Ans. :

- Ajax adds an extra layer of functionality in the communication model. Ajax engine acts as an intermediate between the user interaction to the browser and the server system. In case of using Ajax, the full page is loaded only once when it is requested first time.
- Ajax engine, as an intermediate, takes the request for small segment of the page, which then requests information from the web server asynchronously. Here, the word "asynchronously" means that the requested data is collected in the background without interfering with the whole display and behavior of the existing page.

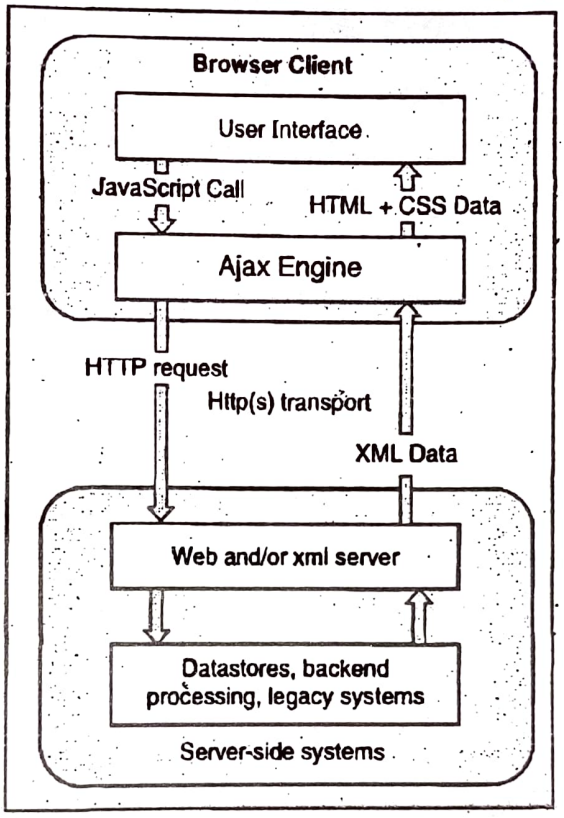


Fig. 3.4 : AJAX Web Application Model

- AJAX communicates with the server using XMLHttpRequest object. Flow of ajax or how ajax works by Fig. 3.4.

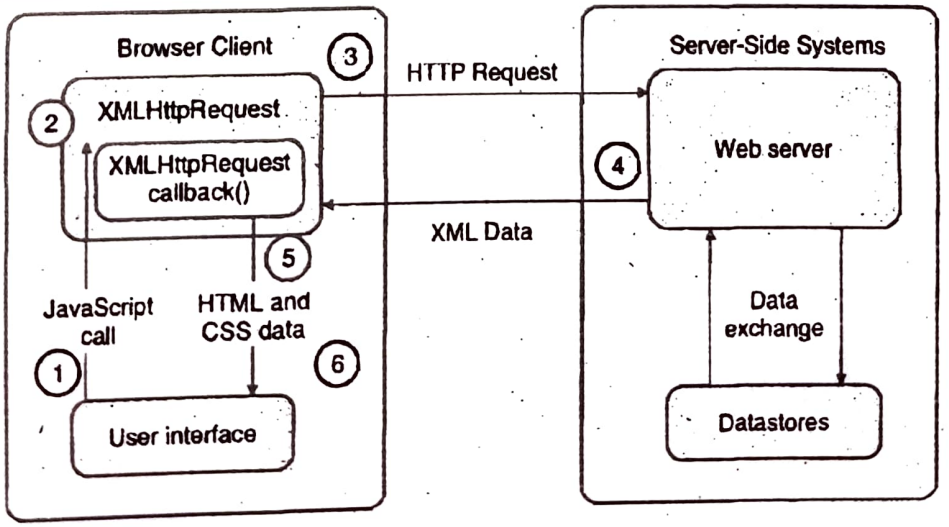


Fig. 3.5

- As per above example, XMLHttpRequest object plays a important role.
  1. User sends a request from the UI and a javascript call goes to XMLHttpRequest object.
  2. HTTP Request is sent to the server by XMLHttpRequest object.
  3. Server interacts with the database using JSP, PHP, Servlet, ASP.net etc.
  4. Data is retrieved.
  5. Server sends XML data or JSON data to the XMLHttpRequest callback function.
  6. HTML and CSS data is displayed on the browser.

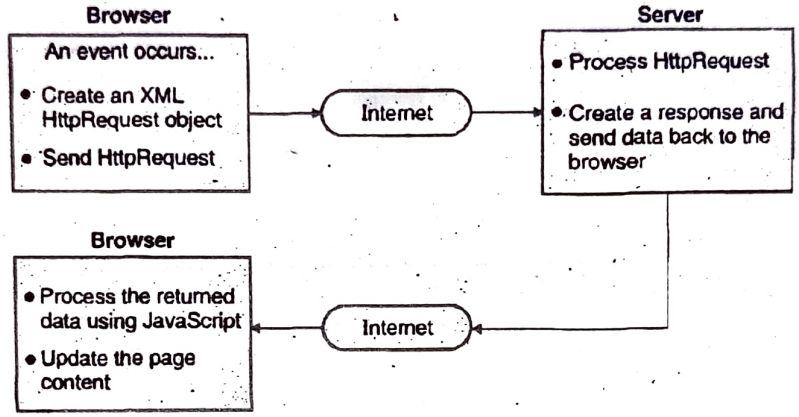


Fig. 3.6 : Working of AJAX with block digram

Q. 27 What are Steps of AJAX Operation. **(5 Marks)**

Ans. :

Steps of AJAX Operation

Following are the steps of AJAX operation.

1. A client event occurs.
2. An XMLHttpRequest object is created.
3. The XMLHttpRequest object is configured.
4. The XMLHttpRequest object makes an asynchronous request to the Webserver.
5. The Webserver returns the result containing XML document.
6. The XMLHttpRequest object calls the callback() function and processes the result.
7. The HTML DOM is updated.

Take these steps one by one.

1. **A Client Event Occurs**
   - A JavaScript function is called as the result of an event.
   - Example – *validateUserId()* JavaScript function is mapped as an event handler to an *onkeyup* event on input form field whose id is set to *"userid"*

```
<input type = "text" size = "20" id = "userid" name = "id" onkeyup = "validateUserId();">.
```

2. The XMLHttpRequest Object is Created

```
var ajaxRequest;  // The variable that makes Ajax possible!
function ajaxFunction() {
  try {
    // Opera 8.0+, Firefox, Safari
    ajaxRequest = new XMLHttpRequest();
  } catch (e) {
      // Internet Explorer Browsers
    try {
      ajaxRequest = new ActiveXObject("Msxml2.XMLHTTP");
    } catch (e) {

      try {
        ajaxRequest = new ActiveXObject("Microsoft.XMLHTTP");
      } catch (e) {

        // Something went wrong
        alert("Your browser broke!");
        return false;
      }
    }
  }
}
```

3. The XMLHttpRequest Object is Configured

In this step, we will write a function that will be triggered by the client event and a callback function processRequest() will be registered.

```
function validateUserId() {
  ajaxFunction();
   // Here processRequest() is the callback function.
  ajaxRequest.onreadystatechange = processRequest;

  if (!target) target = document.getElementById("userid");
  var url = "validate?id=" + escape(target.value);
  ajaxRequest.open("GET", url, true);
  ajaxRequest.send(null);
}
```

□□□

Unit IV : JSP and Web Services

Q. 1 What is JSP ? SPPU : Dec. 15, Dec.16, 2 Marks

Ans. :

JSP

- Java Server Pages (JSP) technology enables you to mix regular, static HTML with dynamically generated content. You simply write the regular HTML in the normal manner, using familiar Web-page-building tools.
- You then enclose the code for the dynamic parts in special tags, most of which start with <% and end with %>. JSP technology integrates numerous Java application technologies, such as Java servlet, JavaBeans, JDBC, and Enterprise JavaBeans. It also separates information presentation from application logic and fosters a reusable-component model of programming.
- JSP pages can be used in combination with servlets that handle the business logic, the model supported by Java servlet template engines.JSP is an integral part of Java EE, a complete platform for enterprise class applications. This means that JSP can play a part in the simplest applications to the most complex and demanding.

Q. 2 Why JSP is used ? (5 Marks)

Ans. :

Use of JSP

- In Java server pages JSP, the execution is much faster compared to other dynamic languages.
- It is much better than Common Gateway Interface (CGI).
- Java server pages JSP are always compiled before its processed by the server as it reduces the effort of the server to create process.
- Java server pages JSP are built over Java Servlets API. Hence, it has access to all Java APIs, even it has access to JNDI, JDBC EJB and other components of java.
- JSP are used in MVC architecture (which will be covered in MVC architecture topic) as view layer.
- The request is processed by a view layer which is JSP and then to servlet layer which is java servlet and then finally to a model layer class which interacts with the database.

Q. 3 Write advantages of JSP over Servlets? SPPU : March 18, 2 Marks

Ans. :

Following are advantages of JSP :

1. **Auto compilation (translation)**

   Whenever there is a change in the code we don't have to recompile the jsp.

   When the jsp changes, its corresponding servlet are automatically regenerated and jsp container automatically reloads them.

2. **Platform independence**

   - JSP Technology delivers "Write Once, Run Anywhere" capability.
   - Instead of being tied to single platform or vender, JSP technology can run on any web server And is supported by wide variety of tools from multiple vendors

3. ***Easy and Rapid Web Development***

- Java server pages simplify and speed up development process. Instead of writing a Java Programs, developer write page using HTML and then add Java code for dynamic content (add XML like tags or Scriplet).
- Also JSP provide reusable component (JavaBean, Enterprise JavaBeans, custom JSP tags) which speed up development process.

4. **Easy Maintenance**

JSP pages are easy to maintain because of separation of the application logic and page design/content.

Q. 4 Write Difference between Servlet and JSP. **(5 Marks)**

Ans. :

Servlet	JSP
Servlet is a java code.	JSP is a html based code.
Writing code for servlet is harder than JSP as it is html in java.	JSP is easy to code as it is java in html.
Servlet plays a controller role in MVC approach.	JSP is the view in MVC approach for showing output.
Servlet is faster than JSP.	JSP is slower than Servlet because the first step in JSP lifecycle is the translation of JSP to java code and then compile.
Servlet can accept all protocol requests.	JSP only accept http requests.
In Servlet, we can override the service() method.	In JSP, we cannot override its service() method.
In Servlet by default session management is not enabled, user have to enable it explicitly.	In JSP session management is automatically enabled.
In Servlet to implement everything like business logic and presentation logic in just one servlet file.	In JSP business logic is separated from presentation logic by using javaBeans.
Modification in Servlet is a time consuming task because it includes reloading, recompiling and restarting the server.	JSP modification is fast, just need to click the refresh button.

Q. 5 Why do we need to use JavaBeans in JSP ? **(5 Marks)**

Ans. :

1. It provides a default, no-argument constructor.
2. It should be serializable and implement the Serializable interface.
3. It may have a number of properties/variables which can be read or written.
4. It may have a number of "getter" and "setter" methods for the properties.

Working of JavaBeans in JSP

First, the browser sends the request for the JSP page. Then the JSP page accesses Java Bean and invokes the business logic. After invoking the business logic Java Bean connects to the database and gets/saves the data. At last, the response is sent to the browser which is generated by the JSP.

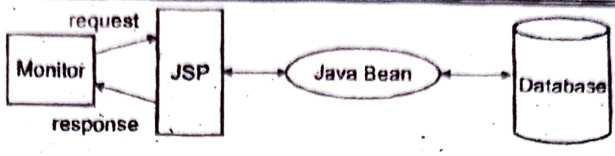


Fig. 4.1

Advantages of Java Beans

1. It is easy to reuse the software components.
2. The properties and methods of Java Bean can be exposed to another application.

Java Bean Properties

1. **getPropertyName() :** This method is used to read the property which is also called accessor.
2. **setPropertyName() :** This method is used to write the property which is also called mutator.

Q. 6 How to access JavaBeans in JSP Application ? **(5 Marks)**

Ans. :

<jsp:useBean>

- The jsp:useBean tag is utilized to start up an object of JavaBean or it can re-utilize the existing java bean object. The primary motivation behind jsp:useBean tag is to interface with bean objects from a specific JSP page.
- In this tag, it is consistently suggestible to give either application or meeting degree to the extension characteristic worth. At the point when the compartment experiences this label then the holder will get class characteristic worth for example completely qualified name of Bean class then the holder will perceive Bean .class record and perform Bean class stacking and launch.
- Subsequent to making the Bean object compartment will allot Bean object reference to the variable indicated as an incentive to id property.
- In the wake of getting Bean object reference holder will store Bean object in an extension indicated as an incentive to scope trait.

Syntax :

```
<jsp:useBean id="—" class"—" type"—" scope="—"/>
```

Attributes :

1. **Id:** It will take a variable to manage generated Bean object reference.
2. **Class:** This attribute will take the fully qualified name of the Bean class.
3. **Type:** It will take the fully qualified name of the Bean class to define the type of variable in order to manage the Bean object reference.
4. **Scope :** It will take either of the JSP scopes to the Bean object.

<jsp:getProperty>

The jsp:getProperty tag is utilized to get indicated property from the JavaBean object. The principle reason for <jsp:getProperty> tag is to execute a getter strategy to get an incentive from the Bean object.

Syntax :

```
<jsp:getProperty name="—" property="—"/>
```

Attributes :

1. **Name :** It will take a variable that is the same as the Id attribute value in <jsp:useBean> tag.
2. **Property :** It will take a particular property to execute the respective getter method.

Q. 7 Explain JSP Related Technologies. **(5 Marks)**

Ans. : JSP Related Technologies

JSP Related Technologies

JSP can be used to support the separation of data processing software from presentation software within a web application. JSP is by no means the only technology designed for this task. Here are few of these related technologies.

1. **ASP (Active Server Pages)** is a server-side scripting language created by Microsoft. An ASP file normally contains HTML tags, just like an HTML file. However, an ASP file can also contain server scripts, surrounded by the delimiters. Server scripts are executed on the server, and can contain any expressions, statements, procedures, or operators valid for the scripting language you prefer to use.
2. **ASP.NET** is part of Microsoft .NET platform. ASP.NET simplifies the transition from windows application development to web development and offers the ability to build web applications using controls similar to windows applications.
3. **Java Servlets** are server side component that provide a powerful mechanism for developing server side web application. Servlet API is used to write servlets which does not assume anything about server's environment or protocol and hence servlers can be embedded in many servers.
4. **ColdFusion** is a rapid development platform for building modern web applications. ColdFusion is designed to be expressive and powerful. The expressive characteristic allows you to perform programming tasks at a higher level than most other languages. The powerful characteristic gives you integration with functionality important to web applications like database access, MS Exchange access, PDF form creation. ColdFusion Markup Language, more commonly known as CFML, is a scripting language for web development that runs on the JVM, the .NET framework, and Google App Engine. ColdFusion (CFML) is an interpreted and dynamic ECMA Script like language that compiles to Java Bytecode directly, thus running in the Java Virtual Machine (JVM) and in almost every operating system.
5. **PHP** is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML. The main advantage of PHP is that it is very easy to understand by a beginner and offers many advanced features for a professional developer. Another advantage is that it can be used on all major operating systems: Windows, Linux, BSD, Mac etc. The most significant feature in PHP is its support for a wide range of databases.
6. **Ruby on Rails** is an open source Ruby framework for developing web-based, database-driven applications. Ruby on Rails divides itself into various packages, namely Active Record, Action Pack, Active Support, Action Mailer and Action Web Service. In addition, Developers can create plug-in to extend existing packages.
7. **EJB (Enterprise Java Beans)** are the Java EE server side components that run inside the EJB container and encapsulates the business logic of an enterprise application. An Enterprise application involves large number of data accessing concurrently by many users. Enterprise beans are used to perform various types of task like interacting with the client, maintaining session for the clients retrieving and holding data from the database and communicating with the server.

Q.8 Explain Localization. **(5 Marks)**

Ans. : Localization

- Struts2 framework supports internationalization and we can create resource bundle property files to be used by the framework. Struts2 i18n is used a lot in creating labels based on the locale in result pages using UI tags.
- Internationalization (i18n) is the process of planning and implementing products and services so that they can easily be adapted to specific local languages and cultures, a process called localization.
- Struts2 framework supports i18n through I18NInterceptor interceptor and we can pass locale in request with parameter*request_locale*. This interceptor is part of default interceptor stack, so we don't need to do anything for localization.
- The I18NInterceptor simply waits for the *request_locale* parameter. Once it receives, the I18NInterceptor triggers and checks the resource bundles for the corresponding language code e.g. 'en' for English, 'es' for Espanol and change the language code.
- Now a Resource bundles (Properties in Java)contain locale-specific objects. When your program needs a locale-specific resource, a String or example, your program can load it from the resource bundle that is appropriate for the current user's locale.
- This example tries to show in a simple and straight forward way of creating a web application with Internationalization or I18N capability.
- In this example, we are creating following pages:
  1. HelloWorld.java
  2. HelloWorld_en.properties and
     HelloWorld _hi.properties
  3. HelloWorld.jsp
  4. struts.xml

1. Create the Action class

```
package com.tuls.sule;
import com.opensymphony.xwork2.ActionSupport;
public class HelloWorld extends ActionSupport {
   public String execute() throws Exception {
      setMessage(getText(MESSAGE));
      return SUCCESS;
   }
   public static final String MESSAGE = "HelloWorld.message";
   private String message;
   public String getMessage() {
      return message;
   }
   public void setMessage(String message) {
      this.message = message;
   }
}
```

2. Create the Properties files

1. HelloWorld_en.properties

```
HelloWorld.message=Good Morning!
```

2. HelloWorld_hi.properties

```
HelloWorld.message=Suprabhat!
```

3. Create HelloWorld.jsp for input

```
<%@ page contentType="text/html; charset=UTF-8" %>
<%@ taglib prefix="s" uri="/struts-tags" %>
<html>
<head>
<title><s:text name="HelloWorld.message"/></title>
</head>
<body>
<h2><s:property value="message"/></h2>
<h3>Languages</h3>
<ul>
<li>
<s:url id="url" action="HelloWorld">
<s:param name="request_locale">en</s:param>
</s:url>
        Click <s:a href="%{url}">here</s:a> to greet in English.
</li>

<li>
<s:url id="url" action="HelloWorld">
<s:param name="request_locale">in</s:param>
</s:url>

       Click <s:a href="%{url}" >here</s:a> to greet in Hindi.

</li>
</ul>
</body>
</html>
```

4. Define action in struts.xml

```
<struts>
<package name="com.tuls.sule " namespace="/com.tuls.sule "
extends="struts-default">
```

```
<action name="HelloWorld" class="com. tuls.sule.HelloWorld">
<result>/com.tuls.sule/HelloWorld.jsp</result>
</action>
</package>
</struts>
```

No need to specify 'method = "execute" ' as action tag attribute because Struts by default checks the 'execute' method if no method is specified. The 'method' attribute of action class is used only when we are using a method name other than 'execute' who's result we need to map in struts.xml file.

Q. 9 Explain various JSP directives. SPPU : Dec.15. May 18, March 19, May 18, 4 Marks

Ans. :

1. Directives
2. Actions
3. JSP Scripting
4. Implicit Objects
5. Comment

1. Directives

i. Directives are used to convey special processing information about the page to the JSP container i.e. Directives are messages to the JSP container containing information on how the JSP container must translate a JSP page into a corresponding servlet.

ii. Directives do not directly produce any output that is visible to end users when the page is requested instead, they generate side effects that change the way the JSP container processes the page.

iii. Directives have the following syntax :

```
<%@ directive attribute1 ="value1" attribute2="value2" ... %>
```

iv. Three types of directives are as follows:

a. Page directives
b. Include directives
c. Tag library directives

a. Page directives

- Page directive gives high-level information about the servlet that will result from the page. Page directive has optional attributes that provide the jsp engine with special processing information.
- Page directive can control Which classes are imported, What class the servlet extends, What MIME type is generated, How multithreading is handled, Is the servlet participates in sessions, The size and behavior of the output buffer, What page handles unexpected errors.

Syntax :

```
<%@ page attribute="value" attribute2="value2"...attributeN="valueN" %>
```

Attributes of Page directive are listed below :

i. The language Attribute

- The language attribute specifies the scripting language used in the JSP page. By default, the value is "java" and all JSP containers must support Java as the scripting language.
- With Tomcat, this is the only accepted language. Other JSP containers might accept different languages.

For example <%@ page language= "java" %>

ii. The contentType Attribute

The contentType attribute defines the Multipurpose Internet Mail Extension (MIME) type of the HTTP response sent to the client. The default value is "text/html; For example,

<%@ page contentType = "application/vnd.ms-excel" %>

iii. The import Attribute

- The import attribute is similar to the import keyword in a Java class or interface. The attribute is used to import a class or an interface or all members of a package.
- Whatever you specify in the import attribute of a page directive will be translated into an import statement in the generated servlet class.
- By default, servlet imports :

```
import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.jsp.*;
import java.lang. *;
```

- Use of the import attribute takes one of the following two forms.

<%@ page import="*package.class*" %>

<%@ page import="*package.class1,...,package.classN*" %>

- For example, the following directive signifies that all classes in the java.util package should be available to use without explicit package identifiers.

<%@ page import="java.util.*" %>

- The import attribute is the only page attribute that is allowed to appear multiple times within the same document.
- Although page directives can appear anywhere within the document, it is traditional to place import statements either near the top of the document or just before the first place that the referenced package is used.

iv. The buffer Attribute

- The buffer attribute specifies the size of the buffer used by the out variable, which is of type JspWriter. Use of this attribute takes one of two forms:

<%@ page buffer="*size*kb" %>

<%@ page buffer="none" %>

- By default, a JSP page's content is buffered to increase performance. The default size of the buffer is 8Kb or 8192 characters. Consider an example that specifies the buffer attribute with a size of 16Kb:

  <%@ page buffer="16kb" %>

- Change the size of the buffer by assigning a number to the attribute. The attribute value represents the number in kilobytes. Therefore, "16" means 16Kb alias 16384 characters.
- You can either write the number only, or the number plus "Kb". For example, "16" is the same as "16Kb".

v. The session Attribute

- The session attributes controls whether the page participates in HTTP sessions.
- Use of this attribute takes one of the following two forms.

<%@ page session="true" %> <%-- Default --%>

<%@ page session="false" %>

- A value of true (the default) signifies that the predefined variable session (of type HttpSession) should be bound to the existing session if one exists; otherwise, a new session should be created and bound to session.
- A value of false means that no sessions will be automatically created and that attempts to access the variable session will result in errors at the time the JSP page is translated into a servlet

vi. The isELIgnored Attribute

- The isELIgnored attribute controls whether the JSP 2.0 Expression Language (EL) is ignored (true) or evaluated normally (false).
- Use of this attribute takes one of the following two forms :

```
<%@ page isELIgnored="false" %>
<%@ page isELIgnored="true" %>
```

vii. autoFlush attribute :

- The autoFlush attribute controls whether the output buffer should be automatically flushed when it is full (the default) or whether an exception should be raised when the buffer overflows (autoFlush="false").
- Use of this attribute takes one of the following two forms.

```
<%@ page autoFlush="true" %><%-- Default --%>
<%@ page autoFlush="false" %>
```

viii. The extends Attribute

The extends attribute designates the superclass of the servlet that will be generated for the JSP page. It takes the following form.

```
<%@ page extends="package.class" %>
```

ix. The isThreadSafe Attribute

- The isThreadSafe attribute controls whether the servlet that results from the JSP page will allow concurrent access (the default) or will guarantee that no servlet instance processes more than one request at a time (isThreadSafe="false").
- Use of the isThreadSafe attribute takes one of the following two forms.

```
<%@ page isThreadSafe="true" %><%-- Default --%>
<%@ page isThreadSafe="false" %>
```

x. The info Attribute

- The info attribute defines a string that can be retrieved from the servlet by means of the getServletInfo method. Info attribute allows author to add documentations string to the page that summarizes functionality such as author, version, copyright information .
- Use of info takes the following form.

```
<%@ page info="Some Message" %>
```

xi. The errorPage and isErrorPage Attributes :

The errorPage attribute specifies a JSP page that should process any exceptions (i.e., something of type Throwable) thrown but not caught in the current page. It isused as follows:

```
<%@ page errorPage="Relative URL" %>
```

The exception thrown will automatically be available to the designated error page by means of the exception variable.

b. Include Directive

- The include directive is the second type of the JSP directive elements. This directive enables JSP page authors to include the contents of other files in the current JSP page.

- The include directive is useful if you have a common source that will be used by more than one JSP page. Instead of repeating the same code in every JSP page, thus creating a maintenance problem, you can place the common code in a separate file and use an include directive from each JSP page.
- The included page itself can be static, such as an HTML file, or dynamic, such as another JSP page. If you are including a JSP page, the included JSP page itself can include another file. Therefore, nesting include directives is permitted in JSP.
- The syntax for the include directive is as follows:

```
<%@ include file="relativeURL" %>
```

The following is an example of how to include HTML files:

```
<%@ include file="includes/header.html" %>
```

c. Taglib Directive

- Jsp specification lets you to define your own tags, build java classes to implement them then use the tags in your jsp.
- Defining custom tags in a Tag Library Descriptor (TLD) file. The TLD file is an XML file that describes the custom tags in a tag library and includes tag information, such as the tag names, type of content, attributes, and associated tag handler class.
- Build java classes to implement defined custom tag.
- To use a custom tag within a JSP page, you must first identify where the TLD file is located and identify a prefix to be used when any of the custom tags in the library are included in a JSP page. This is accomplished through the use of a taglib directive
- You can specify taglib directives as follows :

```
<%@ taglib uri="/ yeartags" prefix="year" %>
<%@ taglib uri="http:www.tuls/ monthtags
" prefix="month" %>
```

- A custom tag library is a set of custom tags that invoke custom actions in a JavaServer Pages (JSP) file. Tag libraries reduce the task of embedding excessive amounts of Java™ code in JSP pages by moving the functionality provided by the tags into tag implementation classes.
- Tag libraries are usually created by developers who are proficient in the Java programming language and can be used by Web designers who may not know Java, but would like to enhance their Web site by taking advantage of Java encoded tag libraries.

Tag libraries :

- Help separate presentation from implementation.
- Are easy to maintain and reuse.
- Simplify complex actions.

Q.10 What is use of is ThreadSafe in JSP? Also Explain Single Thread Model in JSP. SPPU : May 18, March 19, 5 Marks

Ans. :

- JSP Thread Safe is used to send only one client request for processing. It is used to implement Single Thread Model interface.
- The page directive defines an attribute 'isThreadSafe' whose value is either true or false. If the value is set to true which is a default value, the JSP container can send multiple concurrent client requests to the JSP page by starting a new thread.

- If the value of this attribute is set to false, then the JSP container sends client requests only one at a time to the JSP page which makes the jsp page safe.
- When you implement Single Thread Model interface, jsp engine creates new instance of that jsp page for each user request. But implementing Single Thread Model interface leads to some performance issue that may be suitable for low traffic sites. So it is better to avoid this if possible.
- Even if the isThreadSafe attribute is false, we must ensure that accesses to any shared objects using the ServletContext or the HttpSession are properly synchronized.

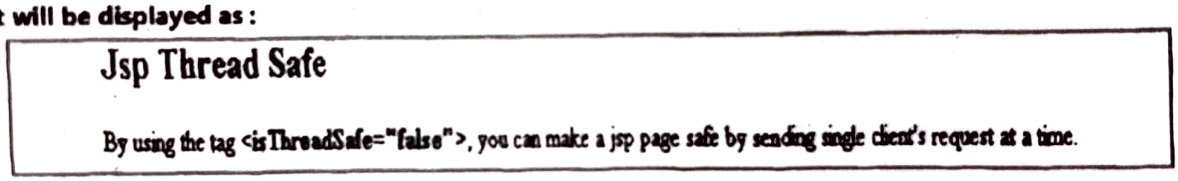
Syntax of isThreadSafe attribute is :

```
<%@ page isThreadSafe="true|false" %>
```

Here is the code of threadSafe.jsp.

```
<%@page isThreadSafe="false" %>
<html>
<head><title>Jsp Thread Safe</title></head>
<body>
<h2>Jsp Thread Safe </h2>
By using the tag <<b>isThreadSafe="false"</b>>, you can make a jsp page safe by sending single clients request at a time
</body>
</html>
```

Output will be displayed as :



1. **isThreadSafe="true"**, creates multiple objects for the same JSP file when requested by multiple clients. Each client is served with a separate **_jspService()** method (with only one JSP file loaded).
2. **isThreadSafe="false"**, allows the container to create one Servlet object for each client requesting the same JSP. Multiple clients will have multiple Servlet objects created by the container to honour all the clients.

Q. 11 Discuss in details jsp : useBean action tag. SPPU : May 18, 3 Marks

Ans. :

- The jsp:useBean action tag is used to locate or instantiate a bean class. If bean object of the Bean class is already created, it doesn't create the bean depending on the scope.
- But if object of bean is not created, it instantiates the bean.

Syntax of jsp:useBean action tag

```
<jsp:useBean id= "instanceName" scope= "page | request | session | application"
class= "packageName.className" type= "packageName.className"
beanName="packageName.className | <%= expression >" >
</jsp:useBean>
```

Attributes and Usage of jsp:useBean action tag

1. **id :** is used to identify the bean in the specified scope.

2. **scope :** represents the scope of the bean. It may be page, request, session or application. The default scope is page.
   (a) **page:** specifies that you can use this bean within the JSP page. The default scope is page.
   (b) **request :** specifies that you can use this bean from any JSP page that processes the same request. It has wider scope than page.
   (c) **session:** specifies that you can use this bean from any JSP page in the same session whether processes the same request or not. It has wider scope than request.
   (d) **Application :** specifies that you can use this bean from any JSP page in the same application. It has wider scope than session.
3. **Class :** instantiates the specified bean class (i.e. creates an object of the bean class) but it must have no-arg or no constructor and must not be abstract.
4. **type :** provides the bean a data type if the bean already exists in the scope. It is mainly used with class or beanName attribute. If you use it without class or beanName, no bean is instantiated.
5. **beanName:** instantiates the bean using the java.beans.Beans.instantiate() method.

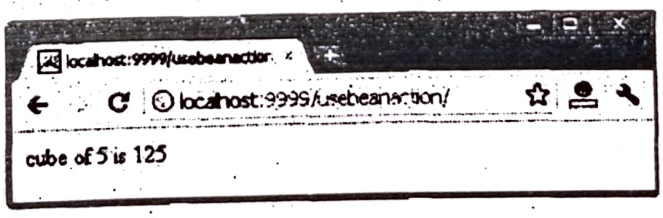
In this example, we are simply invoking the method of the Bean class.

Calculator.java (a simple Bean class)

```
package com.tuls;
public class Calculator{
public int cube(int n){return n*n*n;}
}
```

index.jsp file

```
<jsp:useBean id="obj" class="com.tuls.Calculator"/>
<%
int m=obj.cube(5);
out.print("cube of 5 is "+m);
%>
```



Q. 12 Explain life cycle of a JSP. SPPU : Dec.15,March 18, 3 Marks

Ans. :

- When a request is mapped to a JSP page for the first time, it translates the JSP page into a servlet class and compiles the class. It is this servlet that services the client requests.
- A JSP page has seven phases in its lifecycle, as listed below in the sequence of occurrence:

1. Translation	2. Compilation
3. Loading the class	4. Instantiating the class
5. jspInit() invocation	6. _jspService() invocation
7. jspDestroy() invocation	

1. **Translation**
   - The process of JSP page translation is determined by the semantics of a JSP page.
   - These semantics includes directive, action, and custom action in JSP page. In this phase, the JSP page is read, parsed, and validated. If there are no errors, a Java file containing the servlet class is created.
2. **Compilation**

   The Java file created in the translation phase is compiled into a class file. All the Java code is validated and syntax errors are reported in this phase.
3. **Loading and instantiating**

   The servlet class is loaded into memory and after successful loading JSP container create instance of the Servlet class.
4. **jspInit()**

   The jspInit() method is called only once in the life of the servlet. It is this method that we perform any initializations required for the servlet.
5. **_jspService**

   The request and response objects are passed to this method when each client request is received for the JSP page. JSP scriptlets and expressions are processed and included in this method.
6. **jspDestroy()**

   The jspDestroy() method is called when the servlet instance is taken out of service by the JSP engine. Any cleanup operation, such as releasing resources, can be performed in this method. After this method is called, the servlet is unable to serve any client requests.

Q. 13 Draw and discuss MVC architecture in details. **SPPU : Dec. 19, 5 Marks**

Ans.:

- The **Model-View-Controller (MVC)** is an architectural pattern that separates an application into three main logical components: the **model**, the view, and the controller. Each of these components are built to handle specific development aspects of an application. MVC is one of the most frequently used industry-standard web development framework to create scalable and extensible projects.
- MVC stands for Model, View and Controller. MVC separates application into three components - Model, View and Controller.
- **Model:** Model represents shape of the data and business logic. It maintains the data of the application. Model objects retrieve and store model state in a database. Model is a data and business logic.
- **View:** View is a user interface. View display data using model to the user and also enables them to modify the data. View is a User Interface.
- **Controller:** Controller handles the user request. Typically, user interact with View, which in-turn raises appropriate URL request, this request will be handled by a controller. The controller renders the appropriate view with the model data as a response. Controller is a request handler.
- The following figure illustrates the interaction between Model, View and Controller.
- MVC is popular as it isolates the application logic from the user interface layer and supports separation of concerns. Here the Controller receives all requests for the application and then works with the Model to prepare any data needed by the View. The View then uses the data prepared by the Controller to generate a final presentable response.

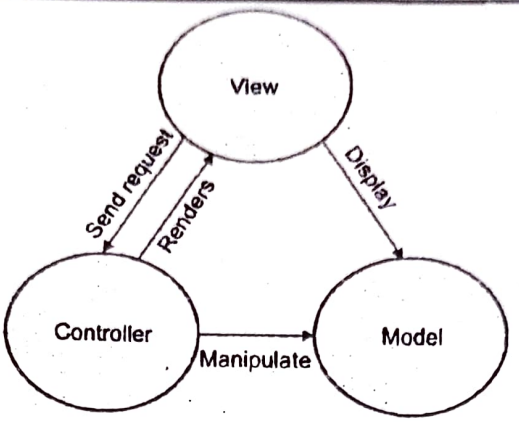


Fig. 4.2 : MVC Architecture

Q. 14 What are the web services ?

SPPU : Dec. 18, May 19, 4 Marks

Ans.

- A Web service is a service, which lives on the Web. Web service posses both the characteristics of a Web and a service. We know what a Web is; let's look at what a service is?
- A Web is a scalable information space with interconnected resources. A We interconnects resources like Web pages, images, an application, word document, e-mail etc.
- A service is an application that exposes its functionality through an API (Application Programming Interface). So what is a component you may ask? A service is a component that can be used remotely through a remote interface either synchronously or asynchronously.
- The term service also implies something special about the application design, which is called a Service-Oriented Architecture (SOA).
- One of the most important features of SOA is the separation of interface from implementation. A service exposes its functionality through interface and interface hides the inner workings of the implementation.
- The client application (i.e user of the service) only needs to know how to use the interface. The client does not have to understand actually how the service does its work.
- For example: There are so many different models of cars like MAZDA, HONDA, TOYOTA etc using different types of engines, motors etc but as a user or driver of the car you do not have to be concerned about the internals. You only need to know how to start the car, use the steering wheel etc, which is the interface to you.
- Usually a service runs on a server, waiting for the client application to call it and ask to do some work? These services are often run on application servers which manage scalability, availability, reliability, multi-threading, transactions, security etc.
- Web Services support loosely coupled connections. The interface of the Web service provides a layer of abstraction between the client and the server. The loosely coupled applications reduce the cost of maintenance and increases re-usability.
- Web Services present a new form of middleware based on XML and Web. Web services are language and platform independent. You can develop a Web service using any language and deploy it on to any platform, from small device to the largest supercomputer.
- Web service uses language neutral protocols such as HTTP and communicates between disparate applications by passing XML or JSON messages to each other via a Web API.

- A web service is a generic term for an interoperable machine-to-machine software function that is hosted at a network addressable location.
- A web service has an interface, which hides the implementation details so that it can be used independently of the hardware or software platform on which it is implemented, and independently of the programming language in which it is written.
- This independence encourages web service based applications to be loosely coupled, component-oriented, cross-technology implementations. Web services can be used alone or with other web services to carry out a complex aggregation or a business transaction.
- Web services offer is the standards based and platform-independent service via HTTP, XML, SOAP, WSDL and UDDI, thus allowing interoperability between heterogeneous technologies such as J2EE and .NET.
- A web service is a collection of open protocols and standards used for exchanging data between applications or systems.
- Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. This interoperability (e.g., between Java and Python, or Windows and Linux applications) is due to the use of open standards.
- To summarize, a complete web service is, therefore, any service that :
  - Is available over the Internet or private (intranet) networks.
  - Uses a standardized XML messaging system.
  - Is not tied to any one operating system or programming language.
  - Is self-describing via a common XML grammar.
  - Is discoverable via a simple find mechanism.

Q. 15 What are general features of web services ? **SPPU : Aug. 15, May 18. 4 Marks**

Ans. :

Web services have the following special behavioural characteristics :

1. **They are XML-Based :** Web Services uses XML to represent the data at the representation and data transportation layers. Using XML eliminates any networking, operating system, or platform sort of dependency since XML is the common language understood by all.
2. **Loosely Coupled :** Loosely coupled means that the client and the web service are not bound to each other, which means that even if the web service changes over time, it should not change the way the client calls the web service. Adopting a loosely coupled architecture tends to make software systems more manageable and allows simpler integration between different systems.
3. **Synchronous or Asynchronous Functionality :** Synchronicity refers to the binding of the client to the execution of the service. In synchronous operations, the client will actually wait for the web service to complete an operation. An example of this is probably a scenario wherein a database read and write operation are being performed.

   If data is read from one database and subsequently written to another, then the operations have to be done in a sequential manner. Asynchronous operations allow a client to invoke a service and then execute other functions in parallel.

   This is one of the common and probably the most preferred techniques for ensuring that other services are not stopped when a particular operation is being carried out.

4. **Ability to support Remote Procedure Calls (RPCs) :** Web services enable clients to invoke procedures, functions, and methods on remote objects using an XML-based protocol. Remote procedures expose input and output parameters that a web service must support.
5. **Supports Document Exchange :** One of the key benefits of XML is its generic way of representing not only data but also complex documents. These documents can be as simple as representing a current address, or they can be as complex as representing an entire book.
6. **Coarse-Grained :** In the coarse-grained operation, a few objects hold a lot of related data. It provides broader functionality in comparison to fine-grained service. It wraps one or more fine-grained services together into a coarse-grained service. It is fine to have more coarse-grained service operations.

Advantages of web services

Some of the advantages of web services are:

1. **Interoperability:** Web services are accessible over network and runs on HTTP/SOAP protocol and uses XML/JSON to transport data, hence it can be developed in any programming language. Web service can be written in Java programming and client can be PHP and vice versa.
2. **Reusability:** One web service can be used by many client applications at the same time.
3. **Loose Coupling :** Web services client code is totally independent with server code, so we have achieved loose coupling in our application.
4. Easy to deploy and integrate, just like web applications.
5. Multiple service versions can be running at same time.

Q. 16 List and explain layers in protocol stack of web serivce architecture. **SPPU : May 19, 4 Marks**

Ans. :

There are two types of web services.

1. **SOAP** : SOAP stands for Simple Object Access Protocol. SOAP is an XML based industry standard protocol for designing and developing web services. Since it's XML based, it's platform and language independent. So our server can be based on JAVA and client can be on .NET, PHP etc. and vice versa.
2. **REST** : REST is an architectural style for developing web services. It's getting popularity recently because it has small learning curve when compared to SOAP. Resources are core concepts of Restful web services and they are uniquely identified by their URIs.

Java Web Services

Java provides its own API to create both SOAP as well as REST web services.

i. **JAX-WS :** JAX-WS stands for Java API for XML Web Services. JAX-WS is XML based Java API to build web services server and client application.

ii. **JAX-RS :** Java API for RESTful Web Services (JAX-RS) is the Java API for creating REST web services. JAX-RS uses annotations to simplify the development and deployment of web services.

1. SOAP Web Services

- SOAP stands for Simple Object Access Protocol. It is an XML based lightweight protocol, which allows software components and application components to communicate, mostly using HTTP (can use SMTP etc). SOAP sits on top of the HTTP protocol.
- SOAP is nothing but XML message based document with predefined format. SOAP is designed to communicate via the Internet in a platform and language neutral manner and allows you to get around firewalls as well.

SOAP Protocol Stack

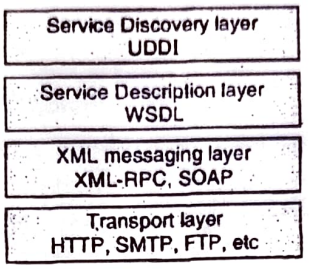


Fig. 4.3

SOAP protocol stack is (SOAP web service components) comprised of

- **Service transport** is the lowest layer in the stack, and is responsible for transporting messages between applications. Currently, this layer includes Hypertext Transfer Protocol (HTTP), Simple Mail Transfer Protocol (SMTP), File Transfer Protocol (FTP), and newer protocols, such as Blocks Extensible Exchange Protocol (BEEP).
- **XML messaging layer** is responsible for encoding messages in a common XML format so that messages can be understood at either end. Currently, this layer includes XML-RPC and SOAP.
- **Service description layer** responsible for describing the public interface to a specific web service. Currently, service description is handled via the Web Service Description Language (WSDL or WADL[for RESTful]).
- **Service discovery layer** is responsible for centralizing services into a common registry, and providing easy publish/find functionality. Currently, service discovery is handled via Universal Description, Discovery and Integration (UDDI).
- There are 2 approaches for generating Web services.
  i. **The contract-first approach**, where you define the contract first with XSD and WSDL and generate the Java classes from the contract.
  ii. **The contract-last approach** where you define the Java classes first and then generate the contract, which is the WSDL file from the Java classes.

A. Contract-first Web Service

Pros

a. Clients are decoupled from the server, hence the implementation logic can be revised on the server without affecting the clients.
b. Developers can work simultaneously on client and server side based on the contract both agreed on.
c. You have full control over how the request and response messages are constructed — for example, should "status" go as an element or as an attribute? The contract clearly defines it. You can change OXM (i.e. Object to XML Mapping) libraries without having to worry if the "status" would be generated as "attribute" instead of an element. Potentially, even Web service frameworks and tool kits can be changed as well from say Apache Axis to Apache CXF, etc.

Cons

a. More upfront work is involved in setting up the XSDs and WSDLs. There are tools like XML Spy,Oxygen XML, etc to make things easier. The object models need to be written as well.
b. Developers need to learn XSDs and WSDLs in addition to just knowing Java.

B. Contract-last Web Service

Pros

a. Developers don't have to learn anything related to XSDs, WSDLs, and SOAP. The services are created quickly by exposing the existing service logic with frameworks/tool sets. For example, via IDE based wizards, etc.

b. The learning curve and development time can be smaller compared to the Contract-first Web service.

Cons

a. The development time can be shorter to initially develop it, but what about the ongoing maintenance and extension time if the contract changes or new elements need to be added? In this approach, since the clients and servers are more tightly coupled, the future changes may break the client contract and affect all clients or require the services to be properly versioned and managed.

b. In this approach, The XML payloads cannot be controlled. This means changing your OXM libraries could cause something that used to be an element to become an attribute with the change of the OXM.

The best practice is to use "contract-first" as the contract last can be more fragile. You will have to decide what is most appropriate based on your requirements, tool sets you use, etc.

Q. 17 Write various features of struts framework. **SPPU : Dec. 18, 4 Marks**

Ans. :

- Struts is a Java-based open-sourced framework that helps in developing web application in J2EE. It extends the Java Servlet API and promotes the Model, View, Controller (MVC) architecture.
- This makes the web applications developed in standard technologies like JSP, JavaBeans, and XML, more maintainable, extensible, and flexible.
- The Struts Framework was initially developed by Craig McClanahan and was handed over to Apache Foundation in May 2000. Gradually, it captures the position of a top-level Apache project in 2005 and later on February 2007, it was replaced by **Struts 2**.
- Struts 2 is based on **Open Symphony WebWork framework**. Struts 2 is very flexible in terms of development and configurations. Struts 2 is the combination of **webwork framework** of open symphony and **struts 1.**
- The Struts Framework was extensively based on the MVC(Model-View-Controller) design paradigm. Its main aim was to separate the Model from the View and the Controller in the application to reduce dependency and promote Separation of Concerns (SoC).

Features of Struts 2

1. **POJO Based forms and actions** : Action classes in Struts are treated as the controller in the application. They are responsible for responding to a user action, executing business logic, and returning a result with the view that has to be rendered. It acts as Model class as well.
2. **Improved Tags and Customization :** Various types of tags have been introduced in Struts 2 like UI tags, control tags, Data tags, etc which aids in application development.
3. **AJAX Functionality :** Struts 2 supports ajax technology which is typically used for generating an asynchronous request. It makes the enhances the performance of the application by sending only the required field data to the server.
4. **Easy Integration :** It provides easy integration with other Web frameworks such as Spring, DWR, SiteMesh and Tiles.
5. **Minimal Configurations :** While using Struts 2 application, there are no overhead configurations required. It works with minimal configurations where most of the settings take the default values unless there is any deviation.

6. **Integrate View Technologies :** With Struts 2, you can easily integrate with various view technologies such as XSLT, JSP, Freemarker, velocity, etc.
7. **Theme and Templates:** Struts 2 provides support to 3 types of themes:

   a. Xhtml b. Simple c. Css_xhtml

   Here XHTML is the default theme for Struts 2 and is mainly used for common look and feel.It would be easy for the developers to change the over all look of a website by changing the theme files.

Q. 18 Draw and explain neat diagram which depicts MVC to the struts architecure. **SPPU : May 18, May 19, 6/8 Marks**

Ans. : Fig. 4.4 shows the architecture of Struts 2 Framework.

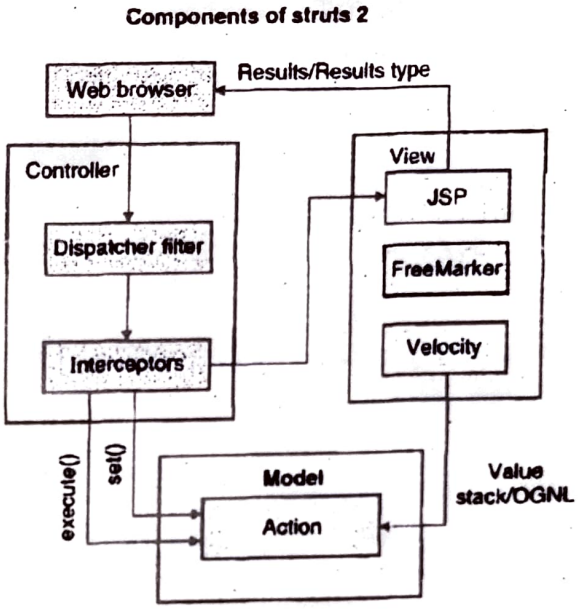


Fig. 4.4 : Struts 2 Architecture

- The **Model-View-Controller** framework in Struts 2 has five core components:

  1. Interceptors in Struts 2
  2. Actions
  3. Value Stack / OGNL
  4. Results
  5. View

- Controller is implemented with a Dispatch Filter and Interceptors. Model is implemented with Actions. View is implemented with a combination of result types and results. Value Stack and OGNL provide linking between the other components.

1. Struts 2 Interceptors

Interceptors can be configured according to every action. They can execute a code before and after an Action is called. Interceptors help in implementing double-submit guards, type conversion, object population, validation, file upload, page preparation, etc. Interceptors are defined in a stack, which in turn decide the execution order of the Interceptor.

2. Action

- Actions are the core basic unit of work in Struts2 framework. Each action provides the processing logic for a specific URL with which it is linked. Actions are mostly associated with a HTTP request of User.
- The action class contains business logic, retrieve resource bundle, hold the data, validation, and select the view result page that should send back to the user. Action class transfer data from request to the View. It also determines which result should render the view.

```
class MyAction
{
public String execute() throws Exception { return "success"; }
}
```

- Action class does not extend another class and nor it implements any interfaces."execute" method returns a String result code, which actions configuration matches with a specific result that will be rendered to the user.

3. **Value Stack / OGNL**

- Object Graph Navigational Language (OGNL) is a open source framework used to get properties from Java Beans. In Struts 2 framework, OGNL provides way to access objects within the value stack. OGNL is a fully featured expression language.
- In Struts 2 framework, it works along with Value Stack to handle requests.OGNL uses dot notation to navigate object graphs.
  - OGNL also supports following features:
  - type conversion
  - calling methods
  - collection manipulation and generation
  - projection across collections
  - expression evaluation
  - lambda expressions
- **Value Stack** is a stack of objects. Whenever a user sends a request a Value Stack object is created. Value Stack also gets the references to request attributes, Session attributes and Application context.

  Value Stack has following objects:

  1. Temporary Objects 2. Model Object 3. Action Object 4. Named Objects
- Control Tags along with expression are used to access properties of objects in value stack.

Q. 19 What are the different configuration files are require to develop any struts application? Explain each configuration file.

SPPU : Dec. 18, March 20, 6 Marks

Ans. : Table 4.1 shows the list of files where we can do configuration for struts2 Application. It also states relative location of file w.r.t webapp and purpose of it.

Table 4.1

File	Location (relative to webapp)	Purpose
web.xml	/WEB-INF/	Web deployment descriptor to include all necessary framework components.
struts.xml	/WEB-INF/classes/	Main configuration, contains result/view types, action mappings, interceptors, and so forth .
default.properties	/WEB-INF/classes/	Framework properties.
struts-plugin.xml	At the root of a plugin JAR	Optional configuration files for Plugins in the same format as struts.xml.
velocity.properties	/WEB-INF/classes/	Override the default *Velocity* configuration.

- You can even place struts-plugin.xml file in the JAR, and it will be automatically plugged into the application. This helps the programmers to develop self-configured components.
- If you want to use the frameworks such as Freemaker and Velocity modules, then the templates can also be loaded from classpath. Velocity.properties files is read from class path. This enables the developer to package entire module just in single JAR file.

1. **Web.xml**
   - This file provides an entry point for any web application. The entry point of Struts2 application will be a filter defined in deployment descriptor (web.xml). Hence we will define an entry of *FilterDispatcher* class in web.xml. FilterDispatcher was provided by Struts 2 for handling all request which needs to be controlled by struts framework.
   - Following is the content of web.xml file which we used in our last example.

```
<?xml version = "1.0" Encoding = "UTF-8"?>
<web-app xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
 xmlns = "http://java.sun.com/xml/ns/javaee"
 xmlns:web = "http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
 xsi:schemaLocation = "http://java.sun.com/xml/ns/javaee
 http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
 id = "WebApp_ID" version = "3.0">
<display-name>Struts 2</display-name>
<welcome-file-list>
<welcome-file>index.jsp</welcome-file>
</welcome-file-list>
<filter>
<filter-name>struts2</filter-name>
<filter-class>
    org.apache.struts2.dispatcher.FilterDispatcher
</filter-class>
</filter>

<filter-mapping>
<filter-name>struts2</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>
</web-app>
```

Q. 20 How to use interceptors in struts 2? List and describe important interceptors provided by struts 2 framework.

SPPU : May 18, 6 Marks

Ans. :

- **Struts Interceptors** are like Servlet Filters that executes before and after the request is being processed. They are used to perform common operations for different actions. For example logging, session validation, adding common headers to response etc.
- Interceptors are basically objects that intercept the Action invocations dynamically and provide the developers with an opportunity specific codes that can be executed before and/or after the execution of an action.

- Most of the functions of frameworks based on Struts 2 are implemented as Interceptors, which includes functions like double-submit guards, type conversion, object population, validation, file upload, page preparation and much more.
- Struts 2 provides a bunch of interceptors and most of them are defined in *struts-default package* and used in defaultStack interceptor stack.
- Interceptors are the power of Struts 2 framework that plays a crucial role in achieving high level of separation of concerns.
- Intercepter are "pluggable" and hence the developer or user can decide exactly which features an Action needs to support. various interceptors come configured by default with struts 2 but depending upon your need custom interceptors can also be created and integrated to the framework. In addition to that, you can also mix and match the interceptors already bundled with the framework.
- Once the request is made by the client for an action, the framework calls the action object but before executing the action, interceptors intercepts the invocation and once the action executes the invocation is intercepted again by the interceptors.
- However, in some cases, Interceptors may prohibit the execution of action as a result of double submits or validation error. Moreover, Interceptors can also change the state of an Action before executing it.

How does it works and how to apply?

- Interceptors carry out most of the work in Struts 2 framework. Interceptors can invoke logic before processing and after processing an action. At first the object of ActionInvocation is created.
- It encapsulates all the action and the interceptors, invoking interceptors before calling the action. After action is called, a result is rendered. If there are any interceptors, they are called again in reverse order before the result is generated and shown to the user.
- Interceptors can modify the action and can even prevent their execution.

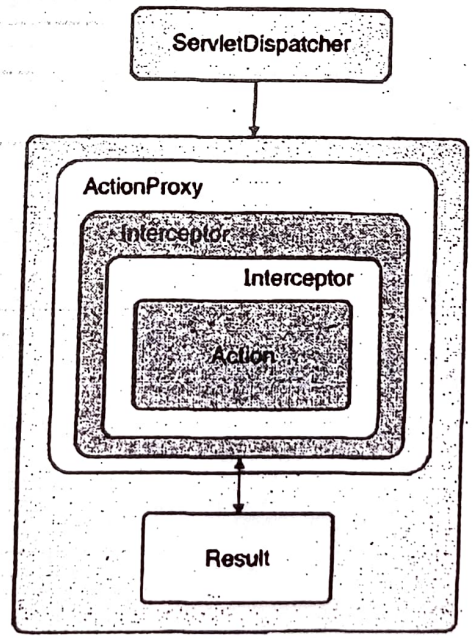


Fig. 4.5

- Action is executed using the interceptor by invocation.invoke() call. Whenever invoke() method is called, ActionInvocationexecutes interceptors (if any). After all interceptors have been invoked, invoke() will execute the action.

How interceptors work:

1. The ServletDispatcher initiates The ActionProxy to invoke the execute() method to which the interceptor intercepts the request before action is executed.
2. The request is intercepted by the Interceptor before (preprocessing) and/or after the action (postprocessing) is executed and completed.
3. After the action is executed, the request is sent to the Result page to render the result and show to the user.

Here we have configured param and timer interceptor

```
<struts>
<package name = "helloworld" extends = "struts-default">
   <action name = "hello"
         class = "com.tuls.struts2.HelloWorldAction"
         method = "execute">
   <interceptor-ref name = "params"/>
   <interceptor-ref name = "timer" />
<result name = "success">/HelloWorld.jsp</result>
   </action>
   </package>
</struts>
```

Built-in Interceptor

The Struts2 framework provides a comprehensive set of pre-defined interceptors and common interceptor stacks which I have listed in the below table:

1. **alias :** It converts similar parameters that have different names between requests.
2. **checkbox :** It is used to handle the check boxes in the form. By this, we can detect the unchecked checkboxes.
3. **cookie :** It adds a cookie to the current action.
4. **conversionError :** It adds conversion errors to the action's field errors.
5. **createSession :** It creates and HttpSession object if it doesn't exists.
6. **clearSession :** It unbinds the HttpSession object.
7. **exception :** It maps exception to a result.
8. **fileUpload :** It provides support to file upload in struts 2.
9. **i18n :** It provides support to internationalization and localization.
10. **logger :** It outputs the action name.
11. **store :** It stores and retrieves action messages, action errors or field errors for action that implements ValidationAware interface.
12. **modelDriven :** It makes other model object as the default object of valuestack.
13. **params :** It populates the action properties with the request parameters.
14. **scope :** It is used to store the action state in the session or application scope.
15. **servletConfig :** It provides access to maps representing HttpServletRequest and HttpServletResponse.
16. **tokenSession :** It prevents duplication submission of request.
17. **validation :** It provides support to input validation.

□□□

Unit V : Server Side Scripting Languages

Q. 1 Classify data type of PHP and describe various data types in each type. SPPU : May 18, May 19, 4/8 Marks

Ans. : A data type is a set of values, and the allowable operations on those values. PHP has eight data types which are Boolean, integer, float, string, array, object, resources and null. These data types are categorized as Scalar types, Compound types and Special types.

i. **Scalar types :** Scalar data types are used to represent a single value.
- Boolean
- Integer
- Float
- String

ii. **Compound types**
- Array
- Object

iii. **Special types**
- Resources
- Null

1. Integer

Integer is stored as signed integers with 32 bits, with a range of -2,147,483,648 and 2,147,483,647.

The PHP var_dump() function returns the data type and value of variables.

Rules for integers are :
- An integer must have at least one digit (0-9).
- An integer cannot contain comma or blanks.
- An integer must not have a decimal point.
- An integer can be either positive or negative.
- Integers can be specified in three different notations in PHP. Decimal, hexadecimal and octal. Octal values are preceded by 0, hexadecimal by 0x.

Example :

```
<?php
$var1 = 31;
$var2 = 031;
$var3 = 0x31;
echo "$var1\n";
echo "$var2\n";
echo "$var3\n";
$x = 5985;
var_dump($x);
?>
```

Output :

31

25

49

Int(5985)

2. Float

A floating point number is a number with a decimal point. It is stored as IEEE floating point number with 64 bits.

Example :

```
<!DOCTYPE html>
<html>
<body>
<?php
$x = 10.365;
var_dump($x);
echo "<br>";
$x = 2.4e3;
var_dump($x);
echo "<br>";
$x = 8E-5;
var_dump($x);
?>

</body>
</html>
```

Output :

float(10.365)

float(2400)

float(8.0E-5)

3. String

String is a data type representing textual data in computer programs. It is a sequence of 8-bit characters. A string literal can be specified in three different ways.

1. single quoted (' ')
2. double quoted(" ")
3. here doc syntax (<<<.)

Example :

```
<!DOCTYPE html>
<html>
<body>
<?php
$x = "Tulshiram Sule!";
echo $x;
echo "<br>";
$x = 'Tulshiram Sule!!';
echo $x;
?>
</body>
</html>
```

Output :

Tulshiram Sule!

Tulshiram Sule!!

Q. 2 Explain server side include in PHP with sample code. **SPPU : Dec. 18. 6 Marks**

Ans. :

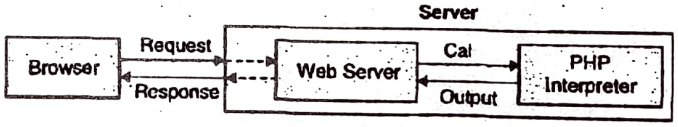


Fig. 5.1

- Always start with a browser making a request for a web page. This request is going to hit the web server. The web server will then analyze it and determine what to do with it.
- If the web server determines that the request is for a PHP file (often index.php), it'll pass that file to the PHP interpreter.
- The PHP interpreter will read the PHP file, parse it (and other included files) and then execute it. Once the PHP interpreter finishes executing the PHP file, it'll return an output. The web server will take that output and send it back as a response to the browser.
- Create a file named hello.php and put it in your web server's root directory (DOCUMENT_ROOT) with the following content:

hello.php

```
<html>
 <head>
  <title>PHP Test</title>
 </head>
 <body>
 <?php echo '<p>Hello World</p>'; ?>
```

```
</body>
</html>
```

- Use your browser to access the file with your web server's URL, ending with the /hello.php file reference. When developing locally this URL will be something like http://localhost/hello.php or

  http://127.0.0.1/hello.php.
- Only the line between <?php and ?> is PHP code. <?php marks the start of an embedded PHP script and ?> marks its end.
- The web server is asked to interpret everything between these two delimiters and convert it to regular HTML code before it sends the web page to the requesting browser.
- If everything is configured correctly, this file will be parsed by PHP and the following output will be sent to your browser.

```
<html>
 <head>
  <title>PHP Test</title>
 </head>
 <body>
 <p>Hello World</p>
 </body>
</html>
```

Q. 3 Explain the basic structure of PHP program with an example. SPPU : Dec. 15, 8 Marks

Ans. :

- PHP basic syntax is nothing it just how to start writing PHP script. Basically, it gives the basic structure of PHP.
- The PHP parsing engine needs a way to differentiate PHP code from other elements in the page. The mechanism for doing so is known as 'escaping to PHP'. There are three ways to do this.

1. Canonical PHP tags

The most universally effective PHP tag style is :

```
<?php...?>
```

If you use this style, you can be positive that your tags will always be correctly interpreted.

2. Short-open (SGML-style) tags

Short or short-open tags look like this –

```
<?...?>
```

- Short tags are, as one might expect, the shortest option You must do one of two things to enable PHP to recognize the tags –
- Choose the - enable-short-tags configuration option when you're building PHP.
- Set the short_open_tag setting in your php.ini file to on. This option must be disabled to parse XML with PHP because the same syntax is used for XML tags.

3. HTML script tags

HTML script tags look like this –

```
<script language = "PHP">...</script>
```

a. PHP is whitespace insensitive

- Whitespace is the stuff you type that is typically invisible on the screen, including spaces, tabs, and carriage returns (end-of-line characters).
- PHP whitespace insensitive means that it almost never matters how many whitespace characters you have in a row.one whitespace character is the same as many such characters.
- For example, each of the following PHP statements that assigns the sum of 2 + 2 to the variable $four is equivalent –

```
$four = 2 + 2; // single spaces
$four <tab>=<tab2<tab>+<tab>2 ; // spaces and tabs
$four =
2+
2; // multiple lines
```

b. PHP is case sensitive :

PHP is a case sensitive language.

```
<html>
  <body>
    <?php
     $capital = 67;
     print("Variable capital is $capital<br>");
     print("Variable CaPiTaL is $CaPiTaL<br>");
    ?>
  </body>
</html>
```

This will produce the following result –

```
Variable capital is 67
Variable CaPiTaL is
```

c. Statements are expressions terminated by semicolons

The semicolon (;) signifies the end of a PHP statement. There should be a semicolon after each line. If you did not terminate any PHP statement then error message will occur.

Q. 4 Discuss about various control structures used in PHP. Give suitable example for each.

SPPU : Dec. 15, May 16. 8 Marks

Ans. :

In control structure we should consider Conditional statements and Looping statements

Conditional Statements

Conditional statements are used to perform different actions based on different conditions. Following Conditional statements are used in PHP.

1. if statement
2. if...else Statement
3. if...elseif....else Statement
4. switch statement

1. if statement

The if statement has the following general form :

```
if (expression){
                statement
              }
```

The if keyword is used to check if an expression is true. If it is true, a statement is then executed. The example below will output "Good Morning!" if the current time (HOUR) is less than 11:

```
<!DOCTYPE html>
<html>
<body>
<?php
$t=date("H");
if ($t<"11")
  {
  echo " Good Morning!";
  }
?>
</body>
</html>
```

2. if...else Statement

Use the if....else statement to execute some code if a condition is true and another code if the condition is false.

Syntax

```
if (condition)
{
   code to be executed if condition is true;
}
else
{
   code to be executed if condition is false;
}
```

The example below will output " I am a boy ,Om " if sex is male, and " I am a girl ,sakshi " otherwise.

```
<?php
$sex = "male";

if ($sex == "male") {
  echo "I am  a boy ,Om\n";
}
else {
echo "I am a girl ,sakshi \n";
}
?>
```

3. if...elseif....else Statement

if....elseif..else statement to select one of several blocks of code to be executed.

Syntax

```
if (condition)
 {
 code to be executed if condition is true;
 }
elseif (condition)
 {
 code to be executed if condition is true;
 }
else
 {
 code to be executed if condition is false;
 }
```

The example below will output "Have a good morning!" if the current time is less than 10, and "Have a good day!" if the current time is less than 20. Otherwise it will output "Have a good night!":

```
<?php
$t=date("H");
if ($t<"10")
 {
 echo "Have a good morning!";
 }
elseif ($t<"20")
 {
 echo "Have a good day!";
 }
else
 {
 echo "Have a good night!";
 }
?>
```

Q. 5 Explain creation of array in PHP. **SPPU : May 17, 8 Marks**

Ans. :

- In PHP array is declared using constructor array (). Arguments of constructor array () are comma separated.
- Argument here is data contain in the array. PHP allows mixed data (i.e. different types of data) in an array. With array( ), you specify a comma-delimited list of key/value pairs. The key and the value are separated by =>.

- Following are few example of how arrays are declared.

```
<?php
$varTulshiArray=array(); //Declaration of an empty array.
$varTulshiArray=array(14,15,19,12,15); //Declaration of an array containing integer data. Data are separated by comma.
$varTulshiArray=array(14,15,'three',12,15); //Declaration of mixed variable array.
// Not all other languages but PHP allows different types of data in an array.
$varTulshiArray=array('color'=>'pink','size'=>'small','price'=>111);//Declaration of a mixed data array with index key.
?>
```

Consider the array declared by the following statement.

```
$varTulshiArray=array(14,15,'three',12,15);
```

Here $varTulshiArray contain five data which are $varTulshiArray[0], $varTulshiArray[1], ...,$varTulshiArray[4].

The integers in the square bracket are the index number of an array. Normally index numbers are started from 0 and ended with (elements count - 1).

Creating array examples :

```
// An array called $vegetables with string keys
$vegetables['corn'] = 'yellow';
$vegetables['beet'] = 'red';
$vegetables['carrot'] = 'orange';
// An array called $dinner with numeric keys
$dinner[0] = 'Sweet Corn and Asparagus';
$dinner[1] = 'Lemon Chicken';
$dinner[2] = 'Braised Bamboo Fungus';
// An array called $computers with numeric and string keys
$computers['trs-80'] = 'Radio Shack';
$computers[2600] = 'Atari';
$computers['Adam'] = 'Coleco';
```

In PHP, there are three types of arrays :

1. Indexed arrays : Arrays with numeric index.
2. Associative arrays : Arrays with named keys.
3. Multidimensional arrays : Arrays containing one or more arrays.

Q. 6 What is Associative arrays in PHP ? Explain it with simple PHP code. **SPPU : May 18, May 19, 6/8 Marks**

Ans. :

- Associative arrays are arrays that use named keys that you assign to them. Associative array will have their index as string so that you can establish a strong association between key and values.
- Both the keys and values can be of any data type in the same array. => operator is used to combine the key and the value of an element of an associative array.

- here are two ways to create an associative array:

```
$salaries = array( "tulshitam" => 2000, "sakshi" => 1000, "om" => 500 );
```

or:

```
$salaries['tulshitam'] = "high";
$salaries['sakshi'] = "medium";
$salaries['om'] = "low";
```

In following example we will the employees names as the keys in our associative array, and the value would be their respective salary

Example

```
<html>
<body>
<?php
/* First method to associate create array. */
$salaries = array(
        "tulshitam" => 2000,
        "sakshi" => 1000,
        "om" => 500
       );
echo "Salary of tulshitam is ". $salaries['tulshitam'] . "<br />";
echo "Salary of sakshi is ". $salaries['sakshi']. "<br />";
echo "Salary of om is ". $salaries['om']. "<br />";
/* Second method to create array. */
$salaries['tulshitam'] = "high";
$salaries['sakshi'] = "medium";
$salaries['om'] = "low";
echo "Salary of tulshitam is ". $salaries['tulshitam'] . "<br />";
echo "Salary of sakshi is ". $salaries['sakshi']. "<br />";
echo "Salary of om is ". $salaries['om']. "<br />";
?>
</body>
</html>
```

Output :

```
Salary of tulshitam is 2000
Salary of sakshi is 1000
Salary of om is 500
Salary of tulshitam is high
Salary of sakshi is medium
Salary of om is low
```

Q. 7 What is Multi-dimensional arrays in PHP? Explain it with simple PHP code.

SPPU : May 15, Dec. 18,March 20, 8 Marks

Ans. :

- A multi-dimensional array each element in the main array can also be an array. And which in turn can hold other arrays as well, and so on.
- In such a way we can create multi dimensional arrays. Values in the multi-dimensional array are accessed using multiple index.

Example

In following example we are storing mark of three student in three subject, so we have created two dimensional array.

```
<<html>
<body>
<?php
 $marks = array(
      "tulshiram" => array
      (
      "physics" => 35,
      "maths" => 30,
      "chemistry" => 39
      ),
      "sakshi" => array
         (
         "physics" => 30,
         "maths" => 32,
         "chemistry" => 29
         ),
         "om" => array
         (
         "physics" => 31,
         "maths" => 22,
         "chemistry" => 39
         )
      );
 /* Accessing multi-dimensional array values */
 echo "Marks for tulshiram in physics : " ;
 echo $marks['tulshiram']['physics'] . "<br />";
 echo "Marks for sakshi in maths : ";
 echo $marks['sakshi']['maths'] . "<br />";
 echo "Marks for om in chemistry : " ;
```

```
echo $marks['om']['chemistry'] . "<br />";
?>
</body>
</html>
```

Output :

Marks for tulshiram in physics : 35

Marks for sakshi in maths : 32

Marks for om in chemistry : 39

Delete data in PHP array

Use PHP unset() function to delete a data from PHP array. Consider the following example.

Example

```
<<html>
<body>
<?php
$array=array(); //Declaration of an empty array.
$array[]='name';
$array[]='address';
$array['age']=26;
foreach($myvariable as $element){echo $element.'<br />';}
unset $array[1];
echo 'After delete 2nd element<br />';
foreach($myvariable as $element){echo $element.'<br />';}
</body>
</html>
```

Output :

name

address

26

After delete 2nd element

Name

26

Q. 8 What are cookies ? Explain cookies in PHP. SPPU : May 15, Dec. 16, 8 Marks

Ans. :

- Cookies are small bits of textual information that a Web server sends to a browser and that the browser later returns unchanged when visiting the same Web site or domain.
- When server send cookie, browser store stores it in its memory or on the hard disk of the computer. By default it is a session-level cookie: a cookie that is stored in the browser's memory and deleted when the user quits the browser.

Cookie work as follows :

When we interact with a web site for first time, server create unique id for you.

- Server stores this unique id along with other information (which link/page you have visited or what information you have downloaded) in database and also sends cookie back to user along with response.
- The next time we interact with server, our browser would automatically send our id along with the HTTP request for particular page to the server.
- The server now takes this id and tries to find a match in its database. After having found a match, it read the cookie and accordingly it sends response back to client along with updated cookie.

Advantages of cookies

1. Remembering Usernames and Passwords
2. Focusing Advertising
3. Customizing Sites
4. Identifying a user during an E-commerce session
5. Cookies are not a serious *security* threat, as cookies are never interpreted or executed in any way and thus cannot be used to insert viruses or attack your system.

Q. 9 Explain Session management techniques in PHP. **SPPU : May 15, May 16, March 20, 8 Marks**

Ans. :

Session management techniques in PHP

- Session is much like the conversation over telephone, whenever we start talking to someone, session starts, and after disconnecting the line session expires.
- Our computer knows its user and it may keep track of who logs-in, time duration etc., but in the case of web servers, it can not keep the tracks of all these things.
- Session is capable of keep the record of the user name, time duration etc.
- Session does not store the data permanently, whenever the user close the website every data is lost. Session creates an UID(Unique ID), which is unique for every user. The UID is generally store in the cookie or it displays in the URL.
- Sessions are a means to store and track data for a user while they travel through a series of pages, or page iterations, on your site. The most significant differences between the two are that cookies are stored on the client, while the session data is stored on the server.
- As a result, sessions are more secure than cookies and sessions work even when the user has disabled cookies in their browser.
- A session creates a file in a temporary directory on the server where registered session variables and their values are stored. This data will be available to all pages on the site during that visit.
- The location of the temporary file is determined by a setting in the **php.ini** file called **session.save_path**. Bore using any session variable make sure you have setup this path.
- When a session is started following things happen :
- PHP first creates a unique identifier for that particular session which is a random string of 32 hexadecimal numbers such as 3c7foj34c3jj973hjkop2fc937e3443.
- A cookie called PHPSESSID is automatically sent to the user's computer to store unique session identification string.

- A file is automatically created on the server in the designated temporary directory and bears the name of the unique identifier prefixed by sess_ i.e. sess_3c7foj34c3jj973hjkop2fc937e3443.
- When a PHP script wants to retrieve the value from a session variable, PHP automatically gets the unique session identifier string from the PHPSESSID cookie and then looks in its temporary directory for the file bearing that name and a validation can be done by comparing both values.
- A session ends when the user loses the browser or after leaving the site, the server will terminate the session after a predetermined period of time, commonly 30 minutes duration.

A. How Sessions Works ?

- Session_start function creates a session or resumes the current one based on the current session id that is being passed via a request like GET, POST, or a cookie. For using named session, first you will have to call session_name() before calling session_start(). The session_start function returns TRUE if the session starts successfully otherwise it returns FALSE.

Example

```
<?php
session_start();
echo "your session ID is ".session_id();
?>
```

- The session_start( ) function generates a random Session Id and stores it in a cookie on the user's computer. The default name for the cookie is PHPSESSID, although this can be changed in the PHP configuration files on the server. To reference the session Id in you PHP code, you would therefore reference the variable $PHPSESSID.

B. Using Session Data

- Having established a session, you can now create, store and retrieve information pertaining to that session. You might want, for example, to keep track of items in your visitor's shopping cart.
- Information for sessions is stored in a special directory on the server; the path of that directory is specified in the server's PHP configuration files.
- Information to be stored for a session is kept in session variables. Session variables are created by registering them for the session, using the session_register( ) function.
- To use that information you simply reference the variable just like you would any other variable. Here's an example:

```
<?php
session_start( );
?>
<html>
<head>
<title>Using a session variable</title>
</head>
<body>
<?php
print "Welcome to session number: ";
```

```
print $PHPSESSID;
?>
<br />
<?php
session_register("username");
$username = "Goody";
print "Your name is: ";
print $username;
?>
</body>
</html>
```

C. Destroying Session

- Session_destroy() function is used for destroying all of the data associated with the current session. Neither it does not intervene any of the global variables nor the session cookie.
- For completely removing the session, it is usually used with session_unset() function. It returns TRUE on success or FALSE on failure.

Example

```
<?php
session_start();
$_SESSION = array();
if (isset($_COOKIE[session_name()])) {
echo setcookie(session_name(), '', time()-42000, '/');
}
session_destroy();
?>
```

Output

As the time cross over 42000 seconds, the session would getexpired and all the related data will be deleted automatically.

Q. 10 List and Explain steps involved in connecting to MySQL with PHP? SPPU : May 18, March 20. 6 Marks

Ans. :

Following are the five steps involved to connect to Mysql from PHP.

1. Connect to the DBMS (MySQL)
2. Select the database required
3. Run the query
4. Retrieve and process result
5. Close the DBMS connection

1. Connect to the DBMS (MySQL)

First step is open a connection to the MySQL server. The function used to connect to MySQL is called mysql_connect(),has following syntax.

```
mysql_connect(hostname,username,password);
```

Where

a. **hostname :** The hostname of the DBMS server to use.

b. **username :** The username of a user having access to the database.

c. **password :** The password of the user

This function returns a resource which is a pointer to the database connection (A connection handle to the DBMS). If MySQL is installed on the same server as the scripting engine, we can use local host as the hostname.

```
$connection= mysql_connect("localhost", "tulshiram ", "password");
 <?php
$username = "your_name";
$password = "your_password";
$hostname = "localhost";
//connection to the database$dbhandle = mysql_connect($hostname, $username, $password)
 or die("Unable to connect to MySQL");
echo "Connected to MySQL<br>";
?>
```

2. Select a Database

Once you've connected to database, select a database to work with. Select a database using mysql select db() fuction. Function has two parameters:

1. The name of the database required.
2. The connection handle to the DBMS (obtained in step 1).

Here we assume that ness is the name of database created

```
<?php
//select a database to work with
$selected = mysql_select_db("ness",$dbhandle)
 or die("Could not select ness");
?>
```

3. Run a Query

Run a query on the database using mysql query() function. This function takes two parameter ,an SQL query string and the connection handle to the DBMS (obtained in step 1) and return a result set handle - to retrieve the output (result set) of the query.

Following functions are used in running query

1. mysql_query(SQL,conn) : Executes the Query with the dbserver & returns the Query Handler.
2. mysql_affected_rows : Returns the num of row affected for the last insert, Update or delete Query.
3. mysql_num_rows(Query handler) : Returns the no of records in the resultant for the select Query .

1. Connect to the DBMS (MySQL)

First step is open a connection to the MySQL server. The function used to connect to MySQL is called mysql_connect(),has following syntax.

```
mysql_connect(hostname,username,password);
```

Where

a. **hostname** : The hostname of the DBMS server to use.

b. **username** : The username of a user having access to the database.

c. **password** : The password of the user

This function returns a resource which is a pointer to the database connection (A connection handle to the DBMS). If MySQL is installed on the same server as the scripting engine, we can use local host as the hostname.

```
$connection = mysql_connect("localhost", "tulshiram ", "password");
<?php
$username = "your_name";
$password = "your_password";
$hostname = "localhost";
//connection to the database$dbhandle = mysql_connect($hostname, $username, $password)
 or die("Unable to connect to MySQL");
echo "Connected to MySQL<br>";
?>
```

2. Select a Database

Once you've connected to database, select a database to work with. Select a database using mysql select db() fuction. Function has two parameters:

1. The name of the database required.
2. The connection handle to the DBMS (obtained in step 1).

Here we assume that ness is the name of database created

```
<?php
//select a database to work with
$selected = mysql_select_db("ness",$dbhandle)
 or die("Could not select ness");
?>
```

3. Run a Query

Run a query on the database using mysql query() function. This function takes two parameter ,an SQL query string and the connection handle to the DBMS (obtained in step 1) and return a result set handle - to retrieve the output (result set) of the query.

Following functions are used in running query

1. mysql_query(SQL,conn) : Executes the Query with the dbserver & returns the Query Handler.
2. mysql_affected_rows : Returns the num of row affected for the last insert, Update or delete Query.
3. mysql_num_rows(Query handler) : Returns the no of records in the resultant for the select Query .

1. Connect to the DBMS (MySQL)

First step is open a connection to the MySQL server. The function used to connect to MySQL is called mysql_connect(),has following syntax.

```
mysql_connect(hostname,username,password);
```

Where

a. **hostname :** The hostname of the DBMS server to use.

b. **username :** The username of a user having access to the database.

c. **password :** The password of the user

This function returns a resource which is a pointer to the database connection (A connection handle to the DBMS). If MySQL is installed on the same server as the scripting engine, we can use local host as the hostname.

```
$connection = mysql_connect("localhost", "tulshiram ", "password");
 <?php
$username = "your_name";
$password = "your_password";
$hostname = "localhost";
//connection to the database$dbhandle = mysql_connect($hostname, $username, $password)
 or die("Unable to connect to MySQL");
echo "Connected to MySQL<br>";
?>
```

2. Select a Database

Once you've connected to database, select a database to work with. Select a database using mysql select db() fuction. Function has two parameters:

1. The name of the database required.
2. The connection handle to the DBMS (obtained in step 1).

Here we assume that ness is the name of database created

```
<?php
//select a database to work with
$selected = mysql_select_db("ness",$dbhandle)
 or die("Could not select ness");
?>
```

3. Run a Query

Run a query on the database using mysql query() function. This function takes two parameter ,an SQL query string and the connection handle to the DBMS (obtained in step 1) and return a result set handle - to retrieve the output (result set) of the query.

Following functions are used in running query

1. mysql_query(SQL,conn) : Executes the Query with the dbserver & returns the Query Handler.
2. mysql_affected_rows : Returns the num of row affected for the last insert, Update or delete Query.
3. mysql_num_rows(Query handler) : Returns the no of records in the resultant for the select Query .

```
<?php
//run the SQL query
$result = mysql_query("SELECT * FROM  car", $connection);
or
$query = "SELECT * FROM car";
$result = mysql_query($query, $connection);
}
?>
```

Q. 11 Write short note on Node JS. SPPU : May 18. 4 Marks

Ans. :

- Node JS is a runtime library and environment which is cross-platform and used for creating running JavaScript applications outside the browser.
- It is free and open-source and utilized for creating server-side JS applications.
- Node JS allows developers to execute their code on the server-side. It provides a faster way to write scripts that are scalable and light. Developers can write real-time applications, and at the same time, it provides scope for mobile application development.
- One can easily utilize Node JS for the front end as well as for back-end development as it allows the use of the same JavaScript.
- Server-side capabilities are provided extensively in Node JS, a developer can listen to and reply to HTTP requests on the computer, listen to traffic network and at the same time can access the database from a computer directly.
- Node JS uses an event-based model to address scalability, and allow rich JavaScript libraries for JavaScript modules which helps in simplify the coding.
- There are plenty of frameworks based on Node JS such as Express JS, Partial JS, etc. Basically, Node JS gives JavaScript the ability to interact with I/O (input/output) devices through its APIs, and connect with other external libraries written in various other languages.

Features of Node.js

Key features of Node.js.

1. **Asynchronous event-driven IO helps concurrent request handling** : This is probably the most significant selling point of Node.js. This feature basically means that if a request is received by Node for some Input/Output operation, it will execute the operation in the background and continue with processing other requests.
2. Node uses the V8 JavaScript Runtime engine, the one which is used by Google Chrome. Node has a wrapper over the JavaScript engine which makes the runtime engine much faster and hence the processing of requests within Node also become faster.
3. **Handling of concurrent requests** : Another key functionality of Node is the ability to handle concurrent connections with a very minimal overhead on a single process.
4. **The Node.js library uses JavaScript** : This is another important aspect of development in Node.js. A major part of the development community is already well versed in javascript, and hence, development in Node.js becomes easier for a developer who knows javascript.
5. There is an active and vibrant community for the Node.js framework. Because of the active community, there are always keys updates made available to the framework.

This helps to keep the framework always up-to-date with the latest trends in web development.

Who uses Node.js

Node.js is used by many large companies. Below is a list of a few of them.

- **Paypal** : A lot of sites within Paypal have also started the transition onto Node.js.
- **LinkedIn** : LinkedIn is using Node.js to power their Mobile Servers, which powers the iPhone, Android, and Mobile Web products.
- **Mozilla** has implemented Node.js to support browser APIs which has half a billion installs.
- eBay hosts their HTTP API service in Node.js

When to Use Node.js

Node.js is best for usage in streaming or event-based real-time applications like

1. **Chat applications**
2. **Game servers :** Fast and high-performance servers that need to processes thousands of requests at a time, then this is an ideal framework.
3. **Good for collaborative environment :** This is good for environments which manage documents. In a document management environment, you will have multiple people who post their documents and do constant changes by checking out and checking in documents. So Node.js is good for these environments because the event loop in Node.js can be triggered whenever documents are changed in a document managed environment.
4. **Advertisement servers :** Again here you could have thousands of request to pull advertisements from the central server and Node.js can be an ideal framework to handle this.
5. **Streaming servers :** Another ideal scenario to use Node is for multimedia streaming servers wherein clients have request's to pull different multimedia contents from this server.

- Node.js is good when you need high levels of concurrency but less amount of dedicated CPU time.

□□□

Unit VI : Ruby and Rails

Q. 1 What are the usage of Rubby. **(5 Marks)**

Ans. : There are many advantages to using Ruby.

Uses of Rubby are

1. **Object :** As Ruby is an object oriented programming language ,everything is an object. That means every object has unique and having its own methods and properties at the time of the object of the class has created. It is also called as singleton. Ruby is simple executing the code with self pointing at the class so it can very easily accessible from any location.
2. **Code Development :** The another reason to using ruby is the development of code is much faster than any other programming languages. In past some studies shows that the Ruby is not the fastest programming language for running and processing requests but developing the software products in ruby is the way faster than other languages.
3. **Modules :** The modules, which allows the dynamic addition of new elements of the class hierarchy at runtime .The modules are making much easier to extend the required functionality. And the modules are easily evaluated which are dynamically added at runtime.
4. **Dynamic Typing :** Ruby is a Dynamic Typed language which means interpreter tries to infer data type of variables during runtime also it can change the object properties as well at runtime. This generally results to programs being more simple, faster and dynamic to code, also the compiler/interpreter loading the code more faster.
5. **Duck Typing :** The ruby is a dynamically typed language that means typing is defined a trun time. There is no implicit type checking in Ruby. It gives maximum control to the developers, so the developer is responsible of the code that it produces. Going forward, the core team proposes a set of principles that each developer decides to adopt or not. One of them is Duck Typing principle:

   It is based on the well known Duck Test.

   When I see a bird that walks like a duck and swims like a duck and quacks like a duck, I call that bird a duck.

   That we can translate In Ruby as following

   If an object quacks like a duck (or acts like an array), just go ahead and treat it as a duck (or an array)
6. **Maintability :** Ruby makes the developers for easy to maintain and understand. It makes the code to run faster and quicker. As the Ruby code is very easy and super simpler ,it is more helpful to track down the bugs and fix them in less time.

   To maintain the large piece of code and already written code can be used again means same code need not be writing again and again this leads to minimizing the developers effort.
7. **Code Quality :** The quality of code is mainly depend s on best practices followed and compliance with common standards. Ruby provides a good quality of code to the application and it is very easy and simple to read and write as well.
8. **Security and Performance :** Ruby ensures the high performance and secured deliverable of application for better customer experience which makes it the first choice of developers to develop the application in ruby. It has a clean code so while developing applications which makes it smooth and great performance without throwing any issues. The applications runs faster because of this and it also ensures the security of the application.

9. **other features :** Ruby is a case-sensitive language which means lowercase and uppercase letters are completely different. It also supports free format means we can start writing of program from any line and column. In Ruby we can use of "#" for writing comment, the interpreter wont take it into consideration. In ruby, keywords are mainly referred to as reserve keywords. Multiple statements on one line must be separated with help of semicolon but it's not required at the end of the line.
10. **Community :** The community helps in every way out to help the developers. It also maintains the applications of ruby and provides the latest information regarding the latest frameworks, libraries, and tools which are getting developed to make ruby better and because of which we can use ruby and develop the new applications for user or customer. It supports the new developers to learn with help of providing the material, books, courses, and other discussion platforms Ruby has given the popular framework Ruby on Rails which is widely used among the developers for the development of applications.

Q. 2 Explain Code Blocks. **(5 Marks)**

Ans. : Code Blocks

A *block* is the same thing as a method, but it does not belong to an object. Blocks are called *closures* in other programming languages. There are some important points about Blocks in Ruby :

- Block can accept arguments and returns a value.
- Block does not have their own name.
- Block consist of chunks of code.
- A block is always invoked with a function or can say passed to a method call.
- To call a block within a method with a value, yield statement is used.

Syntax

```
block_name {
  statement1
  statement2
  .....
}
```

Yield

The *yield statement* is used to call a block inside the method using the *yield keyword* with a value.

```
def test
  puts "You are in the method"
yield
  puts "You are again back to the method"
yield
end
test {puts "You are in the block"}
```

This will produce the following result –

```
You are in the method
You are in the block
You are again back to the method
You are in the block
```

You have seen how a block and a method can be associated with each other. You normally invoke a block by using the yield statement from a method that has the same name as that of the block.

BEGIN and END Block

- Ruby source file has a feature to declare the block of code which can run as the file is being loaded i.e the BEGIN block. After the complete execution of the program END block will execute.
- A program can contain more than 1 BEGIN and END block. BEGIN blocks will always execute in a order but END blocks will execute in reverse order.

```
BEGIN{
# BEGIN block code
  puts "BEGIN code block"
}

END{
# END block code
  puts "END code block"
}
# MAIN block code
puts "MAIN code block"
```

- A program may include multiple BEGIN and END blocks. BEGIN blocks are executed in the order they are encountered.
- END blocks are executed in reverse order. When executed, the above program produces the following result :

```
BEGIN code block
MAIN code block
END code block
```

Q. 3 Explain in details Iterators. (5 Marks)

Ans. : Iterators

- Iterators are nothing but methods supported by *collections*. Objects that store a group of data members are called collections.
- In Ruby, arrays and hashes can be termed collections. Iterators return all the elements of a collection, one after the other Iterator Types

1. Each Iterator	2. Collect Iterator
3. Times Iterator	4. Upto Iterator
5. Downto Iterator	6. Step Iterator
7. Each_Line Iterator	

1. Each Iterator :

This iterator returns all the elements of an array or a hash. Each iterator returns each value one by one. In the below syntax, the collection can be the range, array or hash.

Syntax :

```
collection.each do |variable_name|
  # code to be iterate
end
```

E.g

```
(0..9).eachdo|i|
  puts i
end
```

2. Collect Iterator :

This iterator returns all the elements of a collection. The collect iterator returns an entire collection, regardless of whether it is an array or hash.

Syntax :

```
Collection = collection.collect
```

Example : using collect iterator printing table of 5

```
a = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
b = a.collect{ |y| (5* y) }
puts b
```

Q. 4 What is Pattern Matching. (5 Marks)

Ans. : Pattern Matching

- A regular expression is a pattern that can be matched against a string. It can be a simple pattern such as the string must contain the sequence of letters "cat", or the pattern can be complex, such as the string must start with a protocol identifier, followed by two literal forward slashes, followed by..., and so on.
- Two uses of ruby regex are Validation and Parsing. Ruby regex can be used to validate an email address and an IP address too. Ruby regex expressions are declared between two forward slashes.

```
"Hi there, i am using gfg" =~ /hi/
```

- This will return the index of first occurrence of the word 'hi' if present, or else will return ' nil '. In Ruby pattern matching you can use literals: Booleans, nil, Numbers, Strings, Symbols, Arrays, Hashes, Ranges, Regular Expressions, Procs.

Ruby Pattern Matching Syntax

- Pattern matching in Ruby is done through a case statement. However, instead of using the usual when, the keyword in is used instead. It also supports the use of if or unless statements:

```
case [variable or expression]
in [pattern]
  ...
in [pattern] if [expression]
  ...
else
  ...
end
```

- The patterns are run in the order, like with normal case until we find the first match. In case there will be no pattern found, else clause will be executed. If there is no pattern and no else clause, we will get NoMatchingPatternError.

Pattern matching with Array

```
case[1,2]
in[2,a]
:no_match
in[1,a]
:match
end
=>:match

irb>a
=>2
```

- First pattern didn't match to the array [1, 2], so we went to second one where a match was found. We also get assignment a = 2.

Q. 5 Benefits of Using Ruby on Rails. **(5 Marks)**

Ans. :

Benefits of Using Ruby on Rails :

1. **Cost-effective :** Ruby on Rails is a 100% free and open-source framework. From a developer's point of view, it is easy to use, and with the add on advantage of tons of gems, the developer can save plenty of time and effort.
2. **Secure :** The framework is by default, installed and enabled with some security measures. When you use Ruby in Rails, you are following a secure development process.
3. **Flexibility :** While creating web applications in Rails, it uses frontend and backend abilities; these are simpler and easier to create. A single-page web application uses Rails at the backend and maybe any other framework like Angular for the front end. This makes the application depend on Rails for the best qualities.
4. **Productivity :** Employing Ruby to create your web application permits you to develop features extremely fast. This happens because Ruby effortlessly combines the 3rd party software libraries. It is amongst the most productive programming languages.
5. **Consistent :** Ruby on Rails helps keep a project organized and decipherable as the developers have to follow standardized file storage and programming conventions. Plus, it saves a lot of time.

Q. 6 Explain Rails Request-Response Lifecycle. **(5 Marks)**

Ans. : Following are the steps generally followed

1) The browser issues a request for the "/users" URL.
2) Rails routes "/users" to the index action in the Users controller.
3) The index action asks the User model to retrieve all users (User.all).
4) The User model pulls all the users from the database.
5) The User model returns the list of users to the controller
6) The controller captures the users in the @users variable, which is passed to the index view

7) The view uses embedded Ruby to render the "/users" page as HTML.

8) The controller passes the HTML back to the browser.

These steps follow this path through the Rails file structure :

```
request: GET /users  (http://.../users)                  #URL request (1)
  ↓                                                      #is sent to router...
config/routes.rb
resources :users                                         (#helper includes abstraction of..)
#get /users => 'users#index'                             #Request matches route and
  ↓                                                      #is sent to controller (2)...
app/controllers/users_controller.rb
def index                                                #The 'index' action is run, which
  @users = User.all                                      #makes a request
end                                                      #for all of the user instances
  ↓                                                      #from the model (3)...
app/models/user.rb
class User < ActiveRecord::Base                          #Gets all of the users
end                                                      #from the database (4)
  ↓                                                      #and returns to controller (5)...
app/controllers/users_controller.rb
def index                                                #Assigns all of the users to
  @users = User.all                                      #an instance variable, and
end                                                      #sends them to view (6)...
  ↓
index.html.erb
...                                                      #View uses @users (7)
<% @users.each do |user| %>                              #to display a list
  <tr>                                                   #of all of the users'
    <td><%= user.name %></td>                            #names and emails at
    <td><%= user.email %></td>
...                                                      #"http://.../users" (8)
  ↓
```

Q. 7 Explain the Rails with database. **(5 Marks)**

Ans. :

Rails is database agnostic, meaning it can be used with a variety of different databases. SQLite is supported by Ruby on Rails by default as a highly compatible database, but it's quite easy to use with Postgres instead

Configuring a Database

Just about every Rails application will interact with a database. The database to use is specified in a configuration file, config/database.yml. If you open this file in a new Rails application, you'll see a default database configuration using SQLite. The file contains sections for three different environments in which Rails can run by default :

- The development environment is used on your development computer as you interact manually with the application
- The test environment is used to run automated tests
- The production environment is used when you deploy your application for the world to use.

Configuring a SQLite Database

- Rails comes with built-in support for SQLite, which is a lightweight serverless database application. While a busy production environment may overload SQLite, it works well for development and testing.

- Rails defaults to using a SQLite database when creating a new project, but you can always change it later. Here's the section of the default configuration file with connection information for the development environment:

```
development:
adapter: sqlite3
database: db/development.sqlite3
pool: 5
timeout: 5000
```

Configuring a MySQL Database

If you choose to use MySQL, your config / database.yml will look a little different. Here's the development section:

```
development:
adapter: mysql
encoding: utf8
database: blog_development
pool: 5
username: root
password:
socket: /tmp/mysql.sock
```

Configuring a PostgreSQL Database

If you choose to use PostgreSQL, your config / database.yml will be customized to use PostgreSQL databases:

```
development:
adapter: postgresql
encoding: unicode
database: blog_development
pool: 5
username: blog
password:
```

Run this command on your terminal to create the development database

```
> rake db:setup
```

Database Setup for PostgreSQL

By default, PostgreSQL does not provide any users. We have to create new users. Use the following command to create a user with the name **rubyuser**.

```
>sudo -u postgrescreateuserrbuser -s
```

If you want to create a password for the new user, then use the following command.

```
>sudo -u postgrespsql

postgres=# \password rbuser
```

Use the following command for creating a database **siya_dev.**

```
postgres=# CREATE DATABASE siya_dev OWNER rbuser;

CREATE DATABASE
```

Creating a database table

We create a database table and fill it with data.

create_table.rb

```
require 'pg'

begin

  con = PG.connect :dbname => 'siya_dev', :user => 'rbuser'

con.exec "DROP TABLE IF EXISTS Cars"
con.exec "CREATE TABLE Cars(Id INTEGER PRIMARY KEY,
    Name VARCHAR(20), Price INT)"
con.exec "INSERT INTO Cars VALUES(1,'Audi',52642)"
con.exec "INSERT INTO Cars VALUES(2,'Mercedes',57127)"

rescue PG::Error => e

  puts e.message
ensure

con.close if con

end
```

Q. 8 Executes a query that returns multiple rows of data. (5 Marks)

Ans. :

multiple_rows.rb

```
require 'pg'

begin

con = PG.connect :dbname => 'siya_dev', :user => 'rbuser'

rs = con.exec "SELECT * FROM Cars LIMIT 5"
```

```
rs.each do |row|
    puts "%s %s %s" % [ row['id'], row['name'], row['price'] ]
  end

rescue PG::Error => e

  puts e.message

ensure

rs.clear if rs
con.close if con
end
```

Q. 9 How Layouts and Templates are Stitched Together. **(5 Marks)**

Ans. :

At its simplest level, this is what happens when a request is made to your Rails application :

1. Rails finds the template for the corresponding action based either on convention or any other options passed to the render method in your controller action.
2. Similarly, it then finds the correct layout to use, either through naming/directory conventions or from specific options that you provided.
3. Rails uses the action template to generate the content specific to the action. (Note that the template might be composed of partial views, which you'll learn about a bit later.)
4. It then looks for the layout's yield statement and inserts the action's template there.

For every request handled by Rails, at most one layout and action template will be used. The action template can call out to other templates, called partials, to render itself.

Q. 10 Explain rails with AJAX. **(5 Marks)**

Ans. : Rails with Ajax

- Rails has a simple, reliable model for how it implements Ajax operations. Once the browser has rendered and displayed the initial web page, different user actions cause it to display a new web page (like any traditional web application) or trigger an Ajax operation
  - **Some trigger fires** – This trigger could be the user clicking on a button or link, the user making changes to the data on a form or in a field, or just a periodic trigger (based on a timer).
  - **The web client calls the server** – A JavaScript method, XMLHttpRequest, sends data associated with the trigger to an action handler on the server. The data might be the ID of a checkbox, the text in an entry field, or a whole form.
  - **The server does processing** – The server-side action handler does something with the data and returns an HTML fragment to the web client.
  - **The client receives the response** – The client-side JavaScript, which Rails creates automatically, receives the HTML fragment and uses it to update a specified part of the current page's HTML, often the content of a tag.

- These steps are the simplest way to use Ajax in a Rails application, but with a little extra work, you can have the server return any kind of data in response to an Ajax request, and you can create custom JavaScript in the browser to perform more involved interactions.

Example: Rails with Ajax

An example of performing Ajax on delete action.

Step 1 : Create an application named item. rails new item

Step 2 : Write the following command. rails generate scaffold Itemm name:string surname:string

Step 3 : Write migrate command. rake db:migrate

Step 4 : Update your destroy action in aap/views/itemms/index.html.erb file by writing the following code:

```
:remote => true, :class => 'delete_itemm'
<tbody>
<% @itemms.each do |itemm| %>
<tr>
<td><%= itemm.name %></td>
<td><%= itemm.surname %></td>
<td><%= link_to 'Show', itemm %></td>
<td><%= link_to 'Destroy', itemm, method: :delete, data: { confirm: 'Are you sure?' }, :remote => true, :class =>
'delete_itemm' %></td>
</tr>
<% end %>
</tbody>
```

Step 5 : Create app/views/itemms/destroy.js.erb file.

```
$('.delete_itemm').bind('ajax:success', function() {
  $(this).closest('tr').fadeOut();
});
```

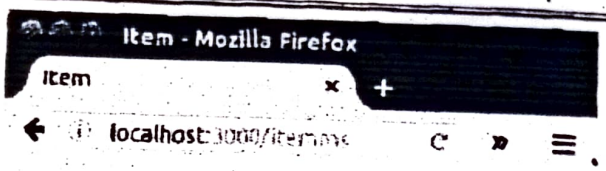
Step 6 : Go to controller file at app/controllers/ itemms_controller.rb and write the following code.

```
def destroy
  @itemm = Itemm.find(params[:id])
  @itemm.destroy
  respond_to do |format|
    format.html { redirect_to item_url }
    format.json { head :no_content }
    format.js   { render :layout => false }
  end

end
```

Step 7 : Start the Rails server. rails s

Step 8 : Run it on localhost.

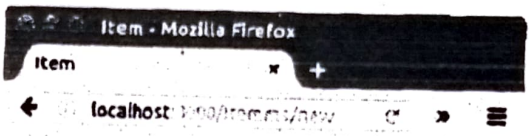


Items

Name Surname

New Item

Create an item record as shown in the following snapshot.



New Item

Name

Tulshiram

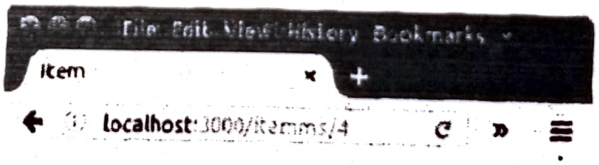
Surname

Sule

Create Itemm

Back

It will create the item as shown below.



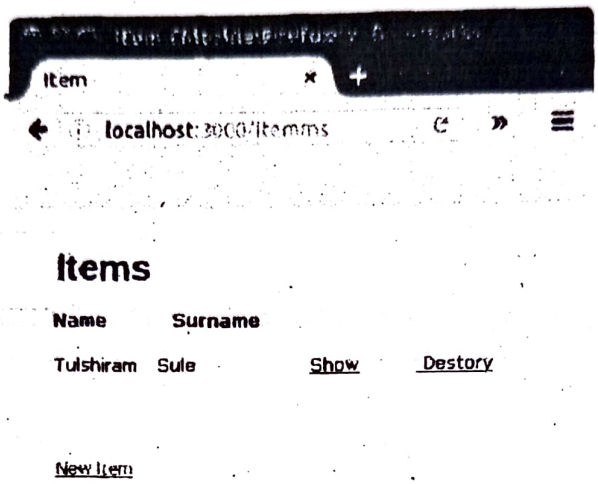
Item was successfully created

Name: Tulshiram

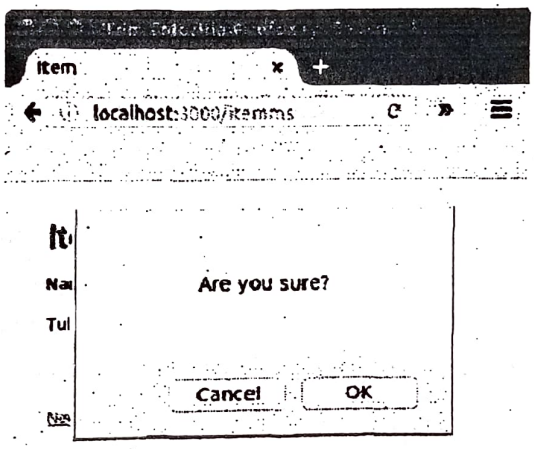
Surname: Sule

Edit | Back

Click on Back button.



If you will click on Destroy link, a popup will be shown through AJAX. It will destroy this item from the list.



Click OK to delete the item finally.

Q.11 What are Advantages / Benefits of Enterprise Java Beans. **(5 Marks)**

Ans. :

EJB technology enables rapid and simplified the process of distributed, transactional, secure and portable java desktop applications development and Java EE web applications development.

1. **EJB provide developers architectural independence :** EJB insulates developers from the underlying middleware, since the only environment an EJB developer sees is the Java environment. It also helps the EJB server/container vendor to change and make improvements on the underlying middleware layer without affecting a user's existing enterprise applications.
2. **WORA for server side components :** Since EJB is based on Java technology, both the developer and the user are guaranteed that their components are Write Once, Run Anywhere (WORA). As long as an EJB Server faithfully conforms to the EJB specification, any EJB application should run within that server.

3. **EJB establishes Roles for Application Development :** The EJB specification assigns specific roles for project participants charged with enterprise application development utilizing EJB. The server vendor can take care of providing support for complex system services and make available an organized framework for a Bean to execute in, without assistance from Bean developers.
4. **EJB provides Distributed Transaction support :** EJB provides transparency for distributed transactions. This means that a client can begin a transaction and then invoke methods on Beans present within two different servers, running on different machines, platforms or JVM.
5. **It helps create Portable and Scalable solutions :** Beans conforming to the EJB API will install and run in a portable fashion on any EJB server.
6. **It provides of vendor specific enhancements :** Since the EJB specification provides a lot of flexibility for the vendors to create their own enhancements, the EJB environment may end being feature rich.

Q. 12 What are the different types of EJB? Identify and explain situations about when to use session beans.

Ans. : SPPU : Dec. 18, 6 Marks

There are several types of enterprise Java beans. The list can be :

1. Session beans
2. Entity beans
3. Message-driven beans

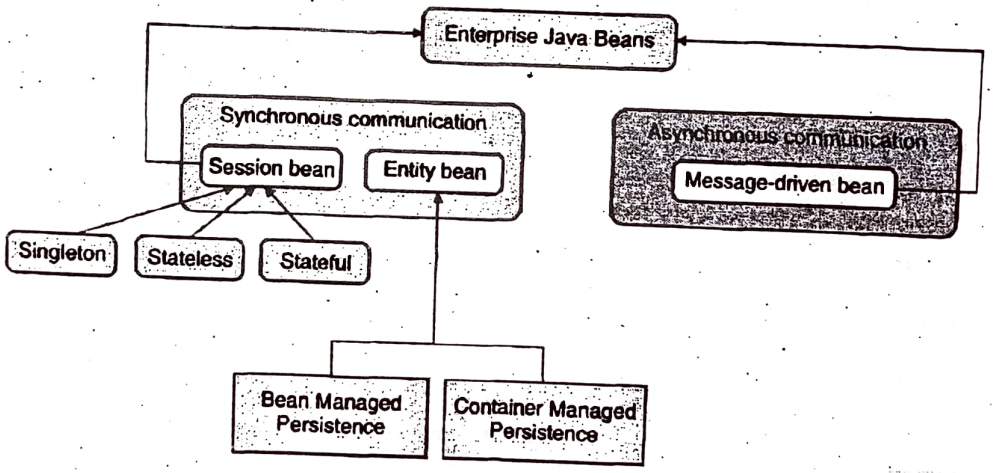


Fig. 6.1 : EJB types

1. Session beans

- Session beans are non-persistent enterprise beans. Session bean encapsulates business logic only, it can be invoked by local, remote and webservice client. It can be used for calculations, database access etc. The life cycle of session bean is maintained by the application server (EJB Container).
- For example, whenever a client wants to perform any of these actions such as making a reservation or validating a credit card, a session bean should be used. The session bean decides what data is to be modified. Typically, the session bean uses an entity bean to access or modify data.
- They implement business logic, business rules, algorithms, and work flows. Session beans are relatively short-lived components. The EJB container may destroy a session bean if its client times out.

There are three kinds of session beans :

a. **Stateful session bean**

- A stateful session bean is a type of enterprise bean, which preserve the conversational state with client. A stateful session bean as per its name keeps associated client state in its instance variables.
- EJB Container creates a separate stateful session bean to process client's each request. As soon as request scope is over, statelful session bean is destroyed.
- A stateful session Bean maintains client-specific session information across several transactions. It exists for the duration of a single client/server session. Thus Stateful session beans has the extra overhead for the server to maintain the state than the stateless session bean.
- For example, consider a customer using a debit card at an **ATM** machine. The ATM could perform various operations like checking an account balance, transferring funds, or making a withdrawal. These operations could be performed one by one, by the same customer. So the bean needs to keep track its state for each of these operations to the same client.

b. **Stateless session bean**

- A stateless session bean is a type of enterprise bean, which is normally used to perform independent operations. A stateless session bean as per its name does not have any associated client state, but it may preserve its instance state.
- EJB Container normally creates a pool of few stateless bean's objects and use these objects to process client's request. Because of pool, instance variable values are not guaranteed to be same across lookups/method calls.
- Example of stateless session bean is sum service which return sum of two number

c. **Singleton session bean**

- A Singleton session bean is instantiated once per application and exists for the lifecycle of the application. Singleton session beans are designed for circumstances in which state must be shared across all clients. Similar to Stateless beans, developers must ensure that singletons thread safe.
- **Example :** Loading a global daily price list that will be the same for every user might be done with a singleton session bean, since this will prevent the application having to do the same query to a database over and over again..

When to use Session Beans ?

- In general, you should use a session bean if the following circumstances hold:
  1. At any given time, only one client has access to the bean instance.
  2. The state of the bean is not persistent, existing only for a short period (perhaps a few hours).
  3. The bean implements a web service.
- Use stateful session beans, when bean is associated with and contains data about a specific client during method calls. Use stateless session beans, when methods of the bean do not depend on the specific client and your application is accessed by many clients.
- Stateful session beans are appropriate if any of the following conditions are true:
  1. The bean's state represents the interaction between the bean and a specific client.
  2. The bean needs to hold information about the client across method invocations.

3. The bean mediates between the client and the other components of the application, presenting a simplified view to the client.
4. Behind the scenes, the bean manages the work flow of several enterprise beans.

- You might choose a stateless session bean if it has any of these traits.
  - The bean's state has no data for a specific client.
  - In a single method invocation, the bean performs a generic task for all clients.
  - For example, you might use a stateless session bean to send an email that confirms an online order.
  - The bean implements a web service.
- Singleton session beans are appropriate in the following circumstances.
  - State needs to be shared across the application.
  - A single enterprise bean needs to be accessed by multiple threads concurrently.
  - The application needs an enterprise bean to perform tasks upon application startup and shutdown.
  - The bean implements a web service.

2. Entity beans

Entity beans contain persistent data and it can be saved in the data source.

There are two types:

1. **Container managed persistence** : these entity beans assign their persistence to the EJB container
2. **Bean managed persistence** : these entity beans manage their own persistence

- An entity bean is an object representation of persistent data maintained in a permanent data store such as a database. A primary key identifies each instance of an entity bean. Entity beans are transactional and are recoverable in the event of a system crash.
- Entity beans are representations of explicit data or collections of data, such as a row in a relational database. Entity bean methods provide procedures for acting on the data representation of the bean. An entity bean is persistent and survives as long as its data remains in the database.
- An entity bean can be created in two ways: by direct action of the client in which a *create()* method is called on the bean's home interface, or by some other action that adds data to the database that the bean type represents. In fact, in an environment with legacy data, entity objects may exist before an EJB is even deployed.
- An entity bean can implement either bean-managed or container-managed persistence. In the case of bean-managed persistence, the implementer of an entity bean stores and retrieves the information managed by the bean through direct database calls.
- The bean may utilize either Java Database Connectivity (JDBC) or SQL-Java (SQLJ) for this method. (Session beans may also access the data they manage using JDBC or SQLJ). A disadvantage to this approach is that it makes it more difficult to adapt bean-managed persistence to alternative data sources.
- In the case of container-managed persistence, the container provider may implement access to the database using standard APIs. The container provider can offer tools to map instance variables of an entity bean to calls to an underlying database. This approach makes it easier to use entity beans with different databases.

Q. 14 Draw and explain the role of EJB container in Enterprise applications. SPPU : May 18, 6 Marks

Ans. :

1. EJB Container

- An EJB container is nothing but the program that runs on the server and implements the EJB specifications. EJB container provides special type of the environment suitable for running the enterprise components. Enterprise beans are used in distributed applications that typically contains the business logic.
- All of the entity objects live in container during its creation to removal. We can deploy more than one entity beans in a container.
- When the entity bean is deployed in the container the work of the container is to provide a home interface for the entity bean. This home interface allows a client in removing, creating and finding entity objects.
- It also helps in executing home interface business methods that are not specific to a particular entity bean object. EJBs use the proxy design pattern to make remote invocation (i.e. remote proxy) and to add container managed services like security and transaction demarcation.
- The container performs the various tasks few of them are :
  - **Lifecycle Management** : Individual enterprise beans do not need to explicitly manage process allocation, thread management, object activation, or object destruction. The EJB container automatically manages the object lifecycle on behalf of the enterprise bean.
  - **State Management** : Individual enterprise beans do not need to explicitly save or restore conversational object state between method calls. The EJB container automatically manages object state on behalf of the enterprise bean.
  - **Security** : Individual enterprise beans do not need to explicitly authenticate users or check authorization levels. The EJB container automatically performs all security checking on behalf of the enterprise bean.
  - **Transactions** : Individual enterprise beans do not need to explicitly specify transaction demarcation code to participate in distributed transactions. The EJB container can automatically manage the start, enrolment, commitment, and rollback of transactions on behalf of the enterprise bean.
  - **Persistence** : Individual enterprise beans do not need to explicitly retrieve or store persistent object data from a database. The EJB container can automatically manage persistent data on behalf of the enterprise bean.

Container hosts following components :

EJB Object : A client invokes beans instance through EJB Object. It is called Request Interceptor. EJB have generally 4 interfaces. These are as follows.

i. **Remote Interface** : Remote interface are the interface that has the methods that relate to a particular bean instance. In the Remote interface we have all get methods as given below in the program. This is the interface where all of the business method go.**javax.ejb.Remote** package is used for creating Remote interface.

ii. **Local Interface** : Local interface are the type of interface that are used for making local connections to EJB.**@Local** annotation is used for declaring interface as Local. **javax.ejb.Local** package is used for creating Local interface.

iii. **Home Interface** : Home interface is the interface that has methods that relate to all EJB of a certain class as a whole. The methods which are defined in this interface are create() methods and find() methods . The create() method allows us to create beans. find() method is used in Entity beans.

iv. **Localhome Interface :** The local interfaces extend the following interfaces. These interfaces are generally for use by clients javax.ejb.EJBLocalObject - for the Object interface javax.ejb.EJBLocalHome - for the Home interface

Q. 15 Draw and explain each tier of three tiers architecture using EJB. **SPPU : May 18, May 19, 8 Marks**

Ans. :

- EJB architecture is made up of a client application, an application server, and an EJB container in which the different types of beans live.
- Following are component of EJB Architecture.

1. EJB Container
2. EJB Server
3. Enterprise Beans
4. EJB Client
5. Deployment Descriptor

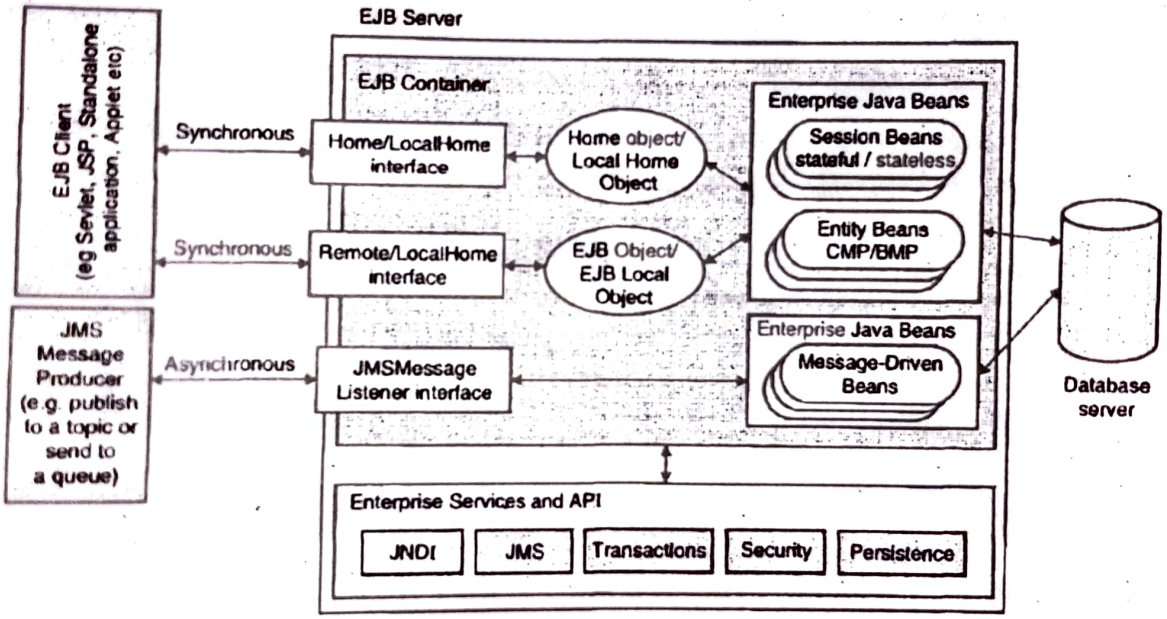


Fig. 6.2 : EJB Architecture

2. Deployment descriptor

- Deployment descriptor is the file which tells the EJB server that which classes make up the bean implementation, the home interface and the remote interface. it also indicates the behavior of one EJB with other. The deployment descriptor is generally called as ejb-jar.xml and is in the directory META-INF of the client application.
- A deployment descriptor is an XML file packaged with the enterprise beans in an EJB JAR file or an EAR file.
- It contains metadata describing the contents and structure of the enterprise beans, and runtime transaction and security information for the EJB container.
- It lists the bean properties and elements like : JNDI Name for Bean,Home and Remote Interface,Bean Implementation Class,Environment Variables,Access Rules or Rights,Beans Management, Etc.

Q. 16 What is Enterprise Java Bean?

SPPU : May 18, 8 Marks

Ans. : Enterprise Beans

Enterprise beans are the **Java EE server side** components that are installed in the EJB containers through the process of deployment and then **Java Virtual Machines (JVM)** at the request of a client program.

A. **Session Bean :** is a non-persistent object that implements some business logic running on the server. Session beans do not survive system shut down. There are two types of session beans

1. **Stateless session beans** (i.e. each session bean can be reused by multiple EJB clients).
2. **Stateful session beans** (i.e. each session bean is associated with one EJB client).

B. **Entity Bean :** is a persistent object that represents object views of the data, usually a row in a database. They have the primary key as a unique identifier. Multiple EJB clients can share each entity bean. Entity beans can survive system shutdowns. Entity beans can have two types of persistence

C. **Container-Managed Persistence (CMP) :** The container is responsible for saving the bean's state.

D. **Bean-Managed Persistence (BMP) :** The entity bean is responsible for saving its own state.

E. **Message-driven Bean :** is integrated with the Java Message Service (JMS) to provide the ability to act as a message consumer and perform asynchronous processing between the server and the message producer.

Q. 17 Draw and explain scenerio of client accesing remote EJB. List some of the EJB clients.

SPPU : Dec. 18, May 19, 8 Marks

Ans. :

An EJB client accesses the business logic contained in the EJBs, in this general way :

1. The EJB client uses a naming service to locate the EJB's home interface.
2. The naming service (usually the Java Naming and Directory Interface, JNDI) returns a reference to an object that implements the EJB's home interface .
3. The EJB client makes a call on the EJB's home interface, to gain access to the EJB's remote interface .
4. The EJB client make calls to the EJB's business methods against the remote interface.

- However, the actual code you use to access an EJB from an EJB client will vary according to the type of Web Application Server you are using.

Remote Clients

- A remote client is a client that can run on the same or a separate machine and a Java virtual machine (JVM) than the enterprise bean it accesses.
- A remote client can be a Web component, an application client, or another enterprise bean. The location of the enterprise bean is transparent to the remote client.
- To access Enterprise JavaBeans deployed in an application you use the JNDI API. JNDI Registry Service is the standard way to associate names with objects and to find objects by their names.
- Remote clients use the JNDI to look up objects. To connect to JNDI you have to create a javax.naming. InitialContext object. In this way you access the root of the naming tree and then you look up the desired object.
- You access beans from remote clients using the EJB lookup schemes.

EJB Lookup Scheme

- The EJB lookup schemes are a new format of the lookup strings for nonstandard Java EE components. The EJB lookup schemes provide the needed abstraction between the client lookup strings and the real bindings in the naming tree.

- Thus, the EJB container has the freedom to bind its objects to different locations in the naming tree and, at the same time, you do not have to change the lookup strings, they remain compatible.

1. You create the InitialContext object by specifying the JNDI properties.

```
InitialContextctx = new InitialContext ();
```

You look up the bean using the specific lookup string in the root context `ctx`.

```
Object obj = ctx.lookup(<lookup-string>);
```

You have to replace the <lookup-string> with one of the formats listed below.

Extended Format

ejb:/key1 =value1, key2=value2..., where the keys can be the following:

Table 6.2 : Allowed Keys Table

Key	Presence	Description
Interface Name	Mandatory	The fully-qualified class name of the desired bean interface.
appName	Optional	The name of the application, which contains the desired bean.
jarName	Optional	The name of the JAR, which contains the desired bean.
beanName	Optional	The name of the desired bean. the value of either the <ejb-name> element in the ejb-jar.xml or the name attribute of the @Stateless or @Stateful annotation).
Interface Type	Optional	The type of the desired bean interface (either local, remote, home or local-home).

Q. 18 Decide and explain various aspects while deciding between local and remote interface can be considered.

SPPU : May 18, 8 Marks

Ans. :

When you design a Java EE application, one of the first decisions you make is the type of client access allowed by the enterprise beans: remote, local, or web service.

Remote Access

A remote client of an enterprise bean has the following traits:

- It may run on a different machine and a different Java Virtual Machine (JVM) than the enterprise bean it accesses. (It is not required to run on a different JVM.)
- It can be a Web component, a J2EE application client, or another enterprise bean.
- To a remote client, the location of the enterprise bean is transparent.
- To create an enterprise bean with remote access, you must code a remote interface and a home interface.
- The *remote interface* defines the business methods that are specific to the bean. For example, the remote interface of a bean named BankAccountEJB might have business methods named debit and credit.
- The *home interface* defines the bean's life cycle methods—create and remove. For entity beans, the home interface also defines finder methods and home methods. Finder methods are used to locate entity beans. Home methods are business methods that are invoked on all instances of an entity bean class.

Local Access

- A local client has these characteristics:
  - It must run in the same JVM as the enterprise bean it accesses.
  - It may be a Web component or another enterprise bean.
  - To the local client, the location of the enterprise bean it accesses is not transparent.
  - It is often an entity bean that has a container-managed relationship with another entity bean.
- To build an enterprise bean that allows local access, you must code the local interface and the local home interface. The local interface defines the bean's business methods, and the local home interface defines its life cycle and finder methods.
- Whether to allow local or remote access depends on the following factors.

1. **Tight or loose coupling of related beans** : Tightly coupled beans depend on one another. For example, if a session bean that processes sales orders calls a session bean that emails a confirmation message to the customer, these beans are tightly coupled. Tightly coupled beans are good candidates for local access. Because they fit together as a logical unit, they typically call each other often and would benefit from the increased performance that is possible with local access.
2. **Container-managed relationships:** If an entity bean is the target of a container-managed relationship, it must use local access.
3. **Type of client** : If an enterprise bean is accessed by application clients, it should allow remote access. In a production environment, these clients almost always run on machines other than those on which the GlassFish Server is running. If an enterprise bean's clients are web components or other enterprise beans, the type of access depends on how you want to distribute your components.
4. **Component distribution** : Java EE applications are scalable because their server-side components can be distributed across multiple machines. In a distributed application, for example, the server that the web components run on may not be the one on which the enterprise beans they access are deployed. In this distributed scenario, the enterprise beans should allow remote access.
5. **Performance** : Owing to such factors as network latency, remote calls may be slower than local calls. On the other hand, if you distribute components among different servers, you may improve the application's overall performance. Both of these statements are generalizations; performance can vary in different operational environments. Nevertheless, you should keep in mind how your application design might affect performance.

- If you aren't sure which type of access an enterprise bean should have, choose remote access. This decision gives you more flexibility. In the future, you can distribute your components to accommodate the growing demands on your application.
- Although it is uncommon, it is possible for an enterprise bean to allow both remote and local access.
- If this is the case, either the business interface of the bean must be explicitly designated as a business interface by being decorated with the @Remote or @Local annotations, or the bean class must explicitly designate the business interfaces by using the @Remote and @Local annotations. The same business interface cannot be both a local and a remote business interface.

Q. 19 Write EJB code to aquire JNDI conext through
(i) default JNDI properties and
(ii) Seperate JNDI properties.

SPPU : Dec. 18, May 19, 8 Marks

Ans. :

- The client of an enterprise bean obtains a reference to an instance of an enterprise bean through either dependency injection, using Java programming language annotations, or JNDI lookup, using the Java Naming and Directory Interface syntax to find the enterprise bean instance.
- Applications that run outside a Java EE server-managed environment, such as Java SE applications, must perform an explicit lookup. JNDI supports a global syntax for identifying Java EE components to simplify this explicit lookup.
- JNDI or **Java Naming Directory Interface** is a directory service which allows lookup of resources. Every resource like an EJB, a Datasource or a JMS Queue running on an application server is given a JNDI name which will be used to locate the resource.
- All servers have a default scheme of assigning JNDI names but it can be overridden to provide custom names. The general convention is *{resourceType}/{resourceName}*. For example, a DataSource's JNDI name can be *jdbc/TestDatabase* and a JMS queue can have *jms/TestQueue* as JNDI name.

Portable JNDI Syntax

Three JNDI namespaces are used for portable JNDI lookups: java:global, java:module, and java:app.

- The java:global JNDI namespace is the portable way of finding remote enterprise beans using JNDI lookups. JNDI addresses are of the following form:

java:global[/application name]/module name /enterprise bean name[/interface name]

- Application name and module name default to the name of the application and module minus the file extension. Application names are required only if the application is packaged within an EAR. The interface name is required only if the enterprise bean implements more than one business interface.
- The java:module namespace is used to look up local enterprise beans within the same module. JNDI addresses using the java:module namespace are of the following form:
- java:module/enterprise bean name/[interface name]
- The interface name is required only if the enterprise bean implements more than one business interface.
- The java:app namespace is used to look up local enterprise beans packaged within the same application. That is, the enterprise bean is packaged within an EAR file containing multiple Java EE modules. JNDI addresses using the java:app namespace are of the following form:

java:app[/module name]/enterprise bean name [/interface name]

- The module name is optional. The interface name is required only if the enterprise bean implements more than one business interface.
- For example, if an enterprise bean, MyBean, is packaged within the web application archive myApp.war, the module name is myApp. The portable JNDI name is java:module/MyBean. An equivalent JNDI name using the java:global namespace is java:global/myApp/MyBean.

□□□

Note