06/06/2024

Red Hat

Course Code: 22CS3004A

Course Title: Enterprise Programming (EP)

L-T-P-S : 3-0-4-4

Credits : 6

Syllabus:

CO:1 :- JDBC (Java Data Base Connectivity)

CO2 :- Servlets (Server Components)

JSP (Java Servlets)

Jakarta server page.

CO3:- JSF (Java Server Faces)

EJB (Enterprise Java Bean)

JPA (Java Persistence API) prominent.

ORM (Object Rational Mapping).

CO4:- JAX-RS (Java API for Restful web services)

↓

Application programmer Interface.

JAAS (Java Authentication Authorisation Service)

RBAC

JMS (Java Messenging Services)

Soap Rest API

CDI (Context and dependency Injection)

API → Communication blw two softwares

Post get put delete

Exam Name: EX183

EX183: Red Hat Enterprise Application Developer.

AD183 ⇒ Course.

1st project (1 tasks) 2nd project. (3 tasks)

max marks: 300

Pass mark : 210

Exam mode: online.

Pre Requisites:

1. Core Java
2. HTML, Java script, Css
3. DB Concepts (Oracle, mysql, Postgresql)

Structured Data (Schema Based)

Unstructured Data (NoSQL → not only SQL)

Semi structured Data (extra properties)

Software Required:

1. Java 8 (JDK 8) and min JDK 11
2. RedHat Code Ready studio IDE (Min JDK 11).
   IDE = Integrated Development Environment.
3. MySQL Server & work bench.
4. JBOSS EAP server.

Types of platforms / editions in Java.

1. Java SE (Software Edition) → Core Java
   → This edition is to develop desktop or console or standalone application. ~~or~~ window
   → All packages in Java SE will be used as
   ```
   import java.io.*
   import java.util.*
   import java.sql.*
   ```
2. Java EE (Enterprise Edition) → Advance Java.
   → This edition is used to develop web applications as well as enterprise edition/application.
   This applications will be developed by organisations based on business usecases.

→ All packages will be used as 'javax' or 'Jakarta'

3. Java ME (Micro Edition)

→ This edition of java is used to develop mobile applications and embedded systems

↓

Prog written to control a system.

eg:- setupboxes.

4. Java FX (Flex)

→ This edition of java is used to develop applications related to animations, drag and drop components etc.

Note:

In enterprise programming course we will be developing web applications followed by enterprise applications used in Java EE. Java EE contains Servlets, JSP, JSF, EJB, JBH.

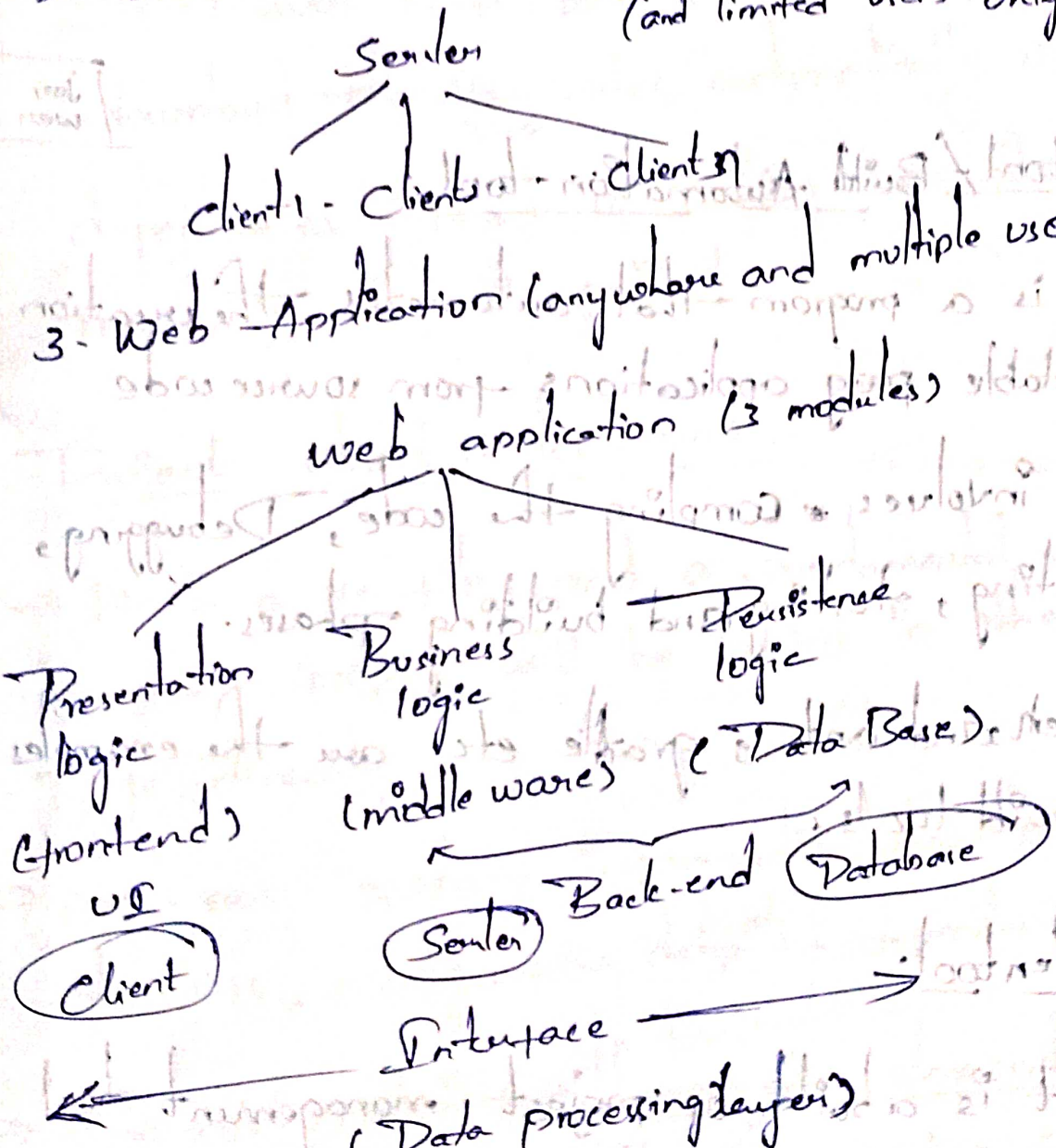
↓

Reusable components

oct 10th 2024

Types of Applications:

1. Desktop or window or console or standalone application. (one user at a time)
   GUI / CUI
2. Client Server application. (limited to organizations (and limited users only))



Client —request→ Server
(web browser) ←—— (web server)
Response

* Client side scripting language (frontend)
* Server side scripting language (middle ware)

jar
war

Build tool / Build Automation tool:

→ It is a program that automates the creation executable ~~prog~~ applications from source code

→ It involves * compiling the code, Debugging, executing, testing and building phases.

→ Maven and ant, gradle etc are the examples of build tools.

Maven tool:

→ It is a build or project management tool that is generally used in java framework

→ It was developed by "Apache" s/w foundation.

→ Maven will be controlled by a single XML file called as pom.xml.

Project object model — extendable markup language

Note: Maven build tool is used to download the JAR files automatically using dependencies instead of adding JAR files manually to the project.

Project object model (pom) :-

→ It is a XML file and a fundamental unit in maven project will be located in the page directly

→ pom.xml contains information about project, versions, configuration details, dependencies etc.

→ pom.xml contains three main elements and three mandatory fields, such as

1. groupId: Company domain name
2. artifactId: project name
3. Version :

eg:

group Id: com.ekey.ep

artifact Id: DemoProject

version: 1.0

The project notation is group

group Id : artifact Id : version

Note: 1. After project creation a default package will be created with the name group Id . artifact Id.

- com.ekey.ep.Demoproject.

2. In pom.xml there will be a root element called as project

→ There are mainly two types of maven projects

* maven-archetype-quickstart: (archetype = architecture type)

1. Using this archetype (architecture type) or Project template, we can create console applications.

2. After building these projects, jar files

will be created (jar archive)

* maven-archetype-webapp:

1. Using this, we can create web applications.
2. After building these projects, war file will be created (web archive)

Note:

→ 2 tier architecture means client and database [JAR FILE]

→ 3 tier architecture means client, server and database [WAR FILE]

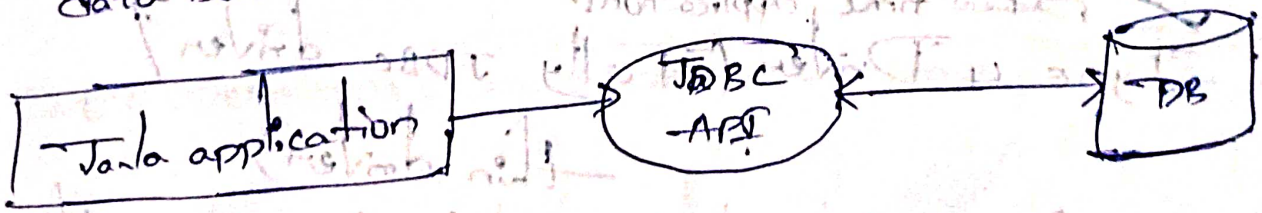
Official web site: https://maven.apache.org/

08/06/2024

JDBC (Java DataBase Connectivity):

↑ Application prog Interface

→ JDBC-API. It is a s/w component that enables java application to interact with data base



Note: JDBC-API acts as interface blw java and database

Types of JDBC Drivers

1. Type 1 Driver (JDBC, ODBC, bridge drivers)
   ↓
   Open DB connectivity

2. Type 2 Driver (Partial JDBC driver / Native No ODBC -API driver)

Native API: It means dedicated to specific platforms

3. Type 3 Driver (Network protocal driver / middle ware drivers)

→ Using this driver client will interact with database using middle ware.

✓ Real time applications.

4. Type 4 Driver (Fully JDBC driver / thin driver)

→ using this driver client can directly access the database

9 steps JDBC programming:

ctrl+shift+O

1. Import sql package
   → import java.sql.*
   where '*' represents classes and interfaces.
2. Driver class Loading (oracle / mysql / postgresql)
3. Establishing a Connection with data base
4. Specifying a Query. (DDL / DML / DQL / DCL / TCL)
   - DDL → Data definition language → Create, Alter, Drops (CAD) (Entire data + schema data will be deleted)
   - DML → insert, update, delete, select, truncate (Entire data will be deleted)
   - DQL → select (or)

~~DML~~ ~~Sel~~

TCL: Commit, Savepoint, roll back

DCL: grant & revoke

CRUD operations (Create, Retrive, Update, Delete)

5. Creating a statements.

There are 3 statements.
   i) Statement
   ii) PreparedStatement
   iii) CallableStatement

6. Executing Query.

executeQuery() → This method will be used to execute select Query only.
Result set
→

executeUpdate() → except select Query
Integer value

execute() → except select Query
Boolean value.

7. Fetching information from ResultSet if it is select Query Only

8. Closing connection

9. Handle the Exception (SQLException)

Driver Management:

→ It will be acted as interface b/w user and the driver. It keeps track of drivers that are available and handles a connection establishment with appropriate driver.

The main job of driver manager is to load the driver class and establish connection with database

MYSQL Commands:

Create database <database-names>;

show databases;

use <database-names>; ⇒ to select specific db

desc

show-tables;

desc <table-names>; ⇒ to display schema (table description)

Constrains key:

→ Unique (allow null values)

→ Primary key (not null + unique)

→ not null

→ foreign key (allow null values)

→ check

→ default

decimal (p,s)

where p is precision (integer)

s is scale (decimal part)

→ p & s should be equal to p

10/06/2024

eg: decimal (4,2)

12.12 ✓

123.21 ✗

Note: decimal (p,s) where p is precision and it represents total no. of digits before and after decimal point and s is scale & it represents no. of digits after decimal point.

* Max size of int data type is 38.

Types of Statements:

1. Statement → This interface is used to execute i) string based sql Querys.

ii) Using this the Query will be compiled and executed every time.

iii) This interface wont accept input parameters at runtime.

iv) It is not recomendable. if you want to execute the same query for several no. of times due to low performance.

2. Prepare Statement ⇒ i) This interface is used to execute string based Parameterized SQL Queries.

ii) Using this the Query will be compailed only once and will be executed for different set of parameters.

iii) This interface will accept input parameters at runtime.

iv) It is recommendale if you want to execute the same query for several no of times dew to high performance.

3. Collable Statement ⇒ i) This interface is used to execute stored procedures and functions.

ii) It will accept input parameters at runtime.

Note:

1. getConnection() method contains 3 arguments.
   i) URL
   ii) User (root only)
   iii) Password

2. ResultSet - Object is used to read ~~the~~ ~~read~~ the data from records that are returned from select query.

* rs.next() → boolean - method is used to make the cursor from current row to next row (forword direction)

rs.previous() - method is used to move the cursor from current row to previous row (backward direction)

rs.getRow() → method is used to print the current row position (Initial Row Position is zero)

Cursor is temporary storage location (implicit cursor and explicit cursor)

11/06/2024

3. execute() method can be used in prepareStatement or CallableStudent in place of executeUpdate() method.

4. while retriving the data using ResultSet object you can use either column index or column na

The column index or column name depends on no. of columns you are retriving using Select Query. (*) represents all colums.

Stored procedures:

→ It is same as function or sub rotitine in order to perform DDL or DML statements

Advantage: Reusability

Subroutine / Stored procedures	Function
1. It will return zero or more values	1. It will return zero or one value.
2. Exception handling can be included	2. Exception handling can't be included.
3. We can call function by using stored procedures	3. We can't call function by using stored procedure.
4. It may or may not return the value	4. It must return the value

Syntax:

```
create or replace procedure <procedure_name>
                                (set of arguments)
IS ----> // DECLARE Declaration stmts;
BEGIN
   // Execution starts; [DDL/DML stmts]
EXCEPTION
   // Exception handling stmts;
END;
/
```

Note: / is used to end of procedure.

Types of parameters in storedprocedures

1. IN → This type of parameter is used as input to the stored procedures
2. OUT → This kind of parameter is used as output to the storedprocedures
3. IN OUT → This kind of parameter is used as both input and output to the storedprocedure.

12/06/2024 → Holiday

13/06/2024

PL/SQL → procedural language / structured Query Langu

- Cursors
  - triggers
  - stored procedures
  - functions

Note: 1) The below command is used to execute stored procedure.

call <procedure_name> (set of arg values)

2) stored procedure may or may not have return value but function must return the value using return keyword.

3) In order to set variable value

set @eid = 101

In order to fetch out parameter value

select @ename

Transactions:

It is defined as set of read and write operations on any object (eg: Table)

ACID properties:

1. Atomicity (A) → It ensures that all operations with in a transactions are completed successfully. If any transaction is failed then the entire transaction must be aborted and the database is left unchanged.

2. Consistency (C) → It ensures that a transactions brings the database from one valid state to another valid state maintaing all constrains.

3. Isolations: → If multiple transactions are running simultaniously, one transaction must work independently without interference of other transactions.

4) Durability → It ensures that once a transaction is commited, the changes are permanent even in the case of system failure also.

. Commit: It makes all changes made in a transaction permanent in the DB.

Savepoint: It is a marker with in a transaction used to roll back to the part of the transaction without effecting whole.

Roll back: It undoes all changes made in the current transaction, reverting the db to the its previous state.

CO1 topics (JDBC)

1. Introduction to JDBC-API
2. Types of JDBC Drivers
3. 9 steps JDBC programming
4. JDBC program to perform CRUD operation using statement Interface.
5. JDBC program to perform CRUD operations using Prepared statement
6. JDBC program collable

7) PL/SQL stored procedures

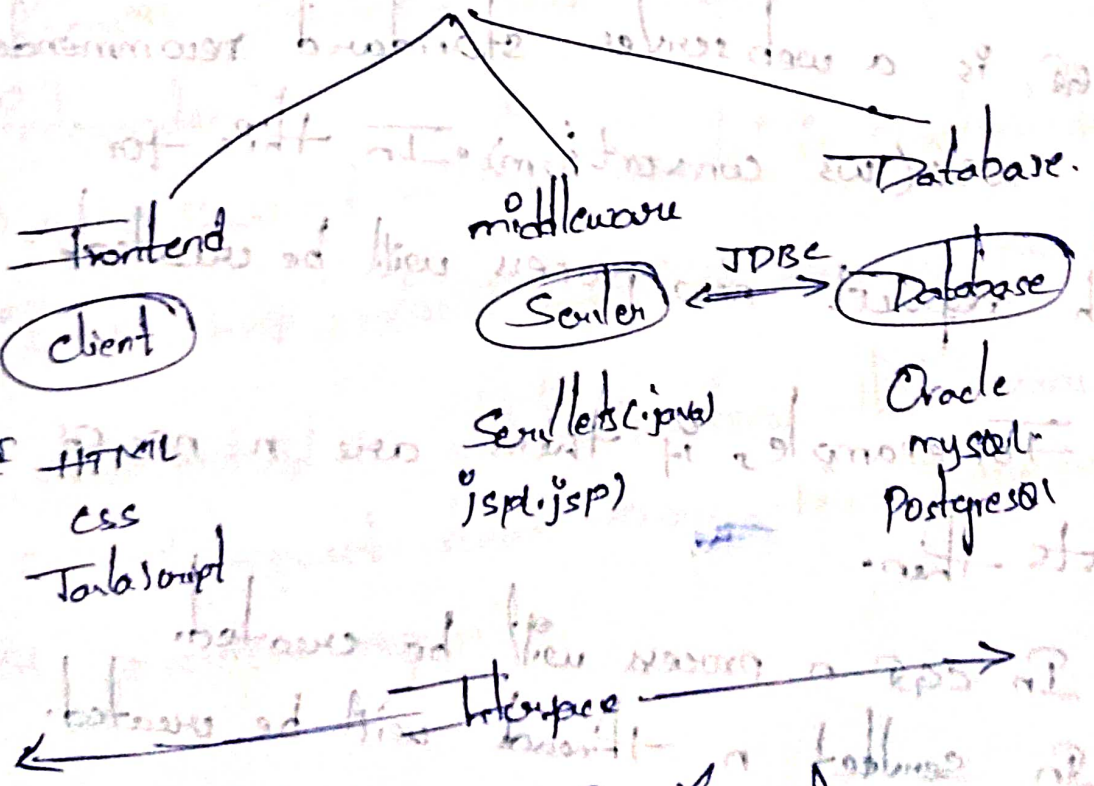
8) JDBC transaction management

10/06/2024

CO-2

Servlet [server component]

1. Servlet is a java program which is going to be compiled by web server.
2. Servlet is middleware technology. [server side scripting language] used to develop web applications in java.
3. The extension of servlet is .java.



CGI Vs Servlet

CGI (Common Gateway Interface) :-

1. It is also called as webserver gateway Interface which is going to implement request and response model based on specific protocal

2. webserver gateway Interface [WSGI] will be acted as interface b/w webserver and web application (Synronus and Asynronaw)

3. CGI is a webserver standard recommended by W3C (W3 consortium). In this for each request new process will be created.

~~eg:~~ For example, if there are n no. of requests then.

1. In CGI n process will be created.
2. In servlet n thread will be created.

Thread is a light weight process.

Note: CGI is not a programming language, which is a script can be used in any programming language

Advantages of Servlets:

1. Better performence.
2. Portability.
3. Secured
4. Robust - Able to handle errors.

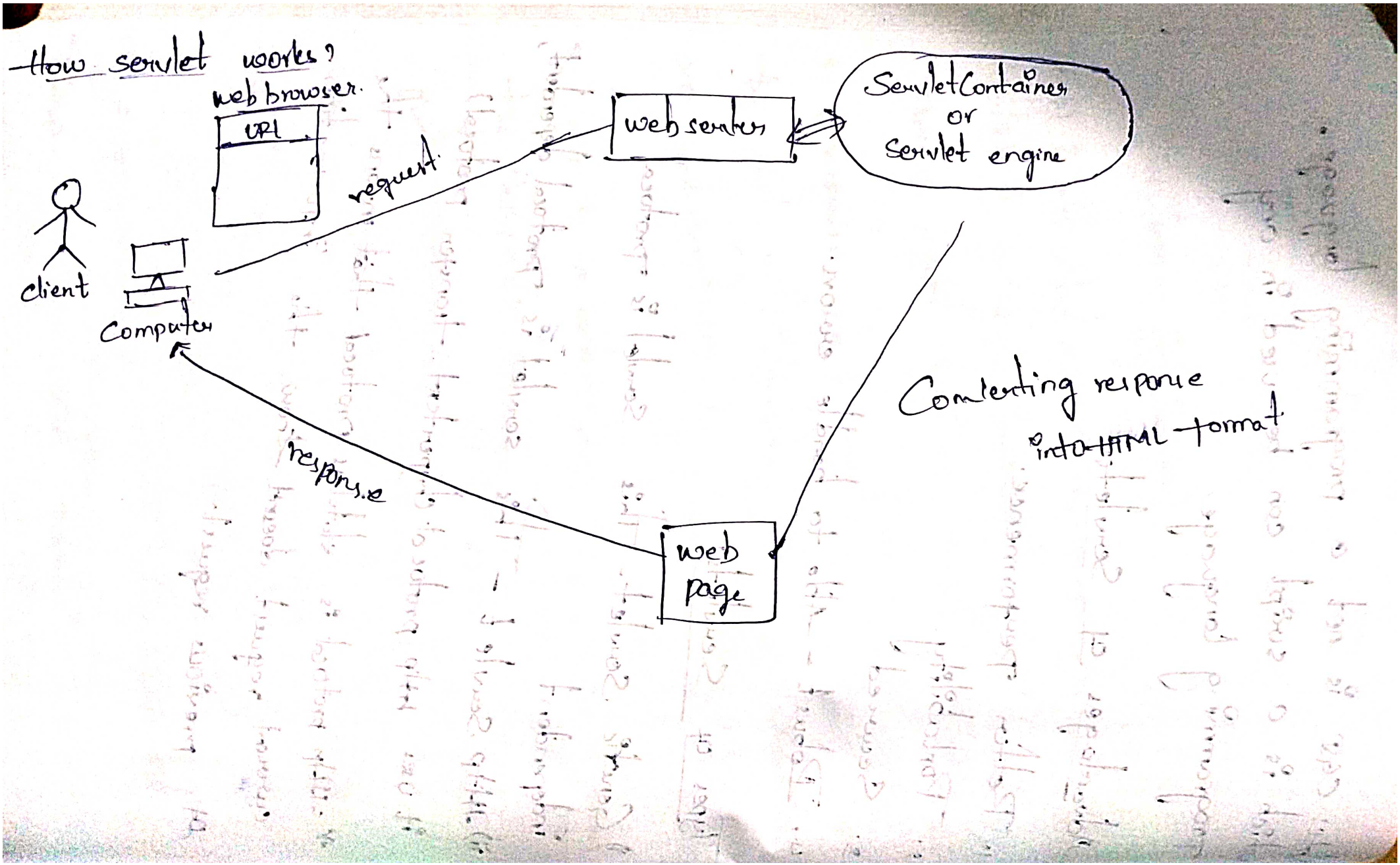
Types of Servlets:

i) Generic Servlet - This Servlet is protocal independent

ii) Http Servlet - This servlet is protocal dependent, (Hypertext transfer protocal)

it uses http protocol.

* Http protocal is stateless protocal that means the current request doesn't know the information of previous request.


How servlet works?
web browser
URL
request
web server
Servlet Container
or
servlet engine
client
Computer
response
Converting response
into HTML format
web
page

1. When client makes a request for a specific servlet using web browser in which the request will be taken in the form of URL.

http://localhost:2014/servletproject/demo

where, demo is URL pattern which will be linked to a specific servlet class. This servlet mapping will be done in web.xml [deployment descriptor]

2. the web browser will send the request to the webserver then it will first find the requested servlet container in servlet container or engine.

Servlet container or engine is defined as a component that will reside inside the webserver. It is responsible for managing servlet life cycle, URL mapping and sending the response to the client.

3. The obtained servlet will convert the response into HTML format by using servlet container.

4. The HTML webpage will be sent to the client as a response using webbrowser.

5. If servlet is not found in servlet container or engine then client will see a response message as "404 not found".

Servlet lifecycle:

1. load the servlet class.
2. Create servlet instance.

→ Servlet container will create servlet instance only once during entire lifecycle. This servlet instance is used to manage servlet request and response.

3. init() method - This method will be called by the servlet container to initilize the server. This will be invoke only one time when you run the servlet for the 1st time.

4. service() method - This method will be invoke for every request made by the client. This method will process the request and send response to the client

5. destroy() method - this method will be called before removing the servlet instance. Whenever you do modifications in your project, you need the project will be redeploy and new WAR file will be created.

Types of Servlet lifecycle programs:

1. Servlet lifecycle by extending generic servlet
2. Servlet lifecycle by extending http servlet
3. Servlet lifecycle by implementing servlet interface.

17/06/2024

Note:

1) During servlet lifecycle servlet class will be

loaded by using Class.forName("class name")

2) You need to stop and start the server for every modification you do in the servlet project. Then only previous servlet instance will be removed and new one will be created.

web.xml

XML stands for Extensible Markup language.

There are 2 types of XMLs.

i) DTD (Document type Definition)

ii) XSD (XML schima Definition)

web.xml is also called as deployment descripter, which will be resided in WEB-INF folder of webapp folder. This file is used to map the servlet using <servlet> tag and <servlet-mapping> tag

The root tag of web.xml is <web-app>

```
<servlet>
<servlet-name>DemoServlet</servlet-name>
<servlet-class>packagename.classname</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>Demoservlet</servlet-name>
<url-pattern>/demo</url-pattern>
</servlet-mapping>
```

Note: There are 3 lifecycle methods.

i) init() ii) service() iii) destroy()

There are 2 non lifecycle methods.

i) getServletConfig() ii) getServletInfo()

getServletConfig() method is used to return config object which will be created during servlet initialization

getServletInfo() method is used to return string value which contains author name, version, copy right information etc.

ServletConfig & ServletContext:

» Servlet object will be created servlet initilization process. Servlet context will be created at the time of web application deployment (during war file creation).

» Servlet config object is servlet scope, where as servlet context is application scope.

init & context parameters:

init parameters:

→ These parameters will be specified with in a servlet and will be applicable to that specific servlet only. we can't access init parameters of one servlet in another servlet.

→ ServletConfig object is used to initilize the init parameters and or context startup parameters

→ init parameters will be specified within a <servlet> tag and inside the <init-param> tag in webxml file.

Context parameters:

⇒ These parameters will be specified for entire application and can be accessed in any servlet with in the application.

→ Servlet context object is used to initilize context parameters (during webapp deployment)

⇒ Context parameters will be specified <context-param> tag and outside the <servlet> tag with in <web-app> tag.

Servlet Chaining (or) Servlet Collaboration:

It is defined as communication between multiple servlets. Using this concept, we can establish communication from one servlet to another servlet and we can pass the data also.

Request Dispatcher(rd):

i) forward(request, response);

ii) include(request, response);

response.sendRedirect()

18/06/24

Request Dispatcher	Send Redirect
1. 2 methods forward(req, res) include(req, res)	1. 1 method response.sendRedirect()
2. Resource must be available on same server. Resource can be servlet (URL, html, JSP)	2. Resource can be available on same server or different server.
3. Visually we will not be able to see the forwarded address. So it is transparent.	3. In address bar we will be able to see the redirected address. so it is not transparent.
4. It will take minimum one request	4. It will take minimum 2 request
5. Request Dispatcher is faster when compared to send Redirect	5. Send Redirect is slower than request dispatcher.

Note:

Request Dispatcher is an interface which is used to transfer or pass the client request from one resource to another resource.

Form Handling:

19/06/2024

Form Handling:

i) method (get/post) by default get method

method = "get"

method = "post"

ii) action = "resource"

resource can be Servlet url or JSP file or HTML file.

The current page will be redirected to the resource you having as specified under action attribute after form submission.

Note:

When you submit the form data using get method, the data will be visible in the address bar & the data will be hided if you use post method.

So, post method is secure.

If the method is post then service() method will be doPost() method.

If the method is get then service() method

Note:

By default, you can use service as method name in Servlet irrespective of any time of HTTP request (POST, GET, PUT, DELETE).

How to fetch the form data from HTML file to servlet?

Enter Name <input type="text" name="name" />

request.getParameter("name"); // this method returns string value.

where name is field name in the HTML form.

Note: After fetching the data from HTML form (front end), to Servlet (middleware), you can use JDBC code to perform any CRUD operation.

@WebServlet Annotation

Annotation is used to reduce xml code and it will be placed at class level, interface level, method and field level.

@webServlet annotation is used to reduce web.xml code for servlet mapping. This should be placed at class level.

example:

```
@webservlet ("/demo")
Public class DemoServlet extends HttpServlet
```

where demo is url pattern

The below code in web.xml is used to set one of the pages as welcome page or landing page.

```
<welcome-file-list>
<welcome-file>demo.html</welcome-files>
</welcome-file-list>
```

Session Management or Session Tracking Techniques.

Session is defined as conversation between client and server. It involves multiple requests & responses

One session - Duration between logout and login.

- As HTTP protocol is stateless protocol, it doesn't maintain any seperate memory, so for this reason we need to track the user using tracking technique and remember

i) HTTPSession [creating session attribute, retriving session attribute]
ii) Cookies [storing data in web browser]
iii) URL Rewriting [Passing Parameters to URL]
iv) Hidden Form Fields
=> <input type = "hidden" name = "demo"/>

Servlet DB Operations:

1. Add Course

addcourse.html
AddCourseServlet.java

2. View Courses

ViewCoursesServlet.java

JSP (Java Server Page)

JSP = Java code + HTML code
(template code)

1. JSP is a middleware technology (server side scripting language) used to develop web applications in java.
2. JSP is extension to Servlet.
3. The extension of JSP file is .jsp and will be complied by web server.

Advantages of JSP

* Extension to Servlet
* Faster deployment.
* Easy to maintain.
* less code than servlet.

JSP Architecture (How JSP page is compiled?)

- In this architecture there are two phases
  1. Translation phase.

In this phase if the given file is JSP then it will be translated to servlet. There is no translation phase if the file is servlet Directory.

2. Compilation phase:

In this phase servlet will be compiled by webserver.

JSP lifeCycle methods:

i) jspInit()

ii) jspService()

iii) jspDestroy()

1. Translation of jsp into servlet.
2. Compilation of servlet file into class (bytecode)
3. Loading Servlet class loading.
4. Create Servlet instance
5. Call jspInit() method for initilization.

6. Call ~~jsp~~ _jspService() method for processing request and sending response to web server.

7. Destroy by jspDestroy() method.

JSP Scripting elements / elements:

1. Declarations:

Using this element we can declare variables and methods.

syntax → <%! start
// variables
// methods
%> end

Note: When we declare variable or method in jsp file inside the declaration tag it means that the decleration is made inside the Servlet class and outside the _jspService() method.

2. Scriptlet:

This element is used to implement the functionality of _jspService() method. This element is used to include any java code

except methods.

Syntax: <% start
// Java code
%> end

3. Expressions:

This element is used to print a variable on web browser using HTML tags.

Syntax: <%= start
//variable
%> end.

eg: <%!
int a=20; } Declarations
%>

<%
System.out.println(a); // console
out.println(a); } Scriptlet
%>

<%= a %> } expression

4. Comments:

JSP Comments

Syntax: <%-- start

--%> end

// Single line comments

/* */ multiline comments

5. Directives:

There are 3 types of directives

i) Page Directive.

ii) Include Directive.

iii) Taglib Directive (JSTL)
Java server Tag lib

20/06/2024

Note:

1. You can use Scriplet tag directly if you want to include java code in jsp file except methods

```
<%
// Any java code except methods
%>
```

2. If you want to write Html code (template code) in jsp file you can write directly

21/06/2021

Implicit objects:

These objects are created by web container that will be available to jsp page. There are Predefined objects where we can use them directly without creating explicitly.

1. request

Type: HttpServletRequest.

request.getParameter("name"); ⇒ To get the data from Html formfiled to Jsp.

Return type: String

request.setAttribute(name,value);

request.getAttribute(name)

request.removeAttribute(name)

} You can use request object to pass the data from one to another file

2. response - HttpServletResponse

response.sendRedirect ("resource"); ⇒ Resource can be servlet url, html file, jsp file or url.

3. Out - JspWriter

out.println ("message"); // or Browser.

Servlet : PrintWriter

Jsp : JspWriter

4. Session - HttpSession

session.setAttribute (name, value)

session.getAttribute (name) :- object as returntype.

session.removeAttribute (name)

session.invalidate () → Remove all attributes.

session.setMaxInactiveInterval (n) → seconds

5. Config - ServletConfig

To initilize init parameters.

6. Application - ServletContext

To initilize config parameters.

7. Page – this object of Servlet class.

8. pageContext – using this object we can set or get or remove attributes using following scopes. There are 3 scopes in Servlet.

1) ~~Req~~ request
2) session
3) application

There are 4 scopes in jsp

1) request
2) session
3) application
4) Page

9. exception – Throwable

request → It can be accessed from any page that serves the request more than one page can serve the single request

session → It can store objects throughout the lifetime of user session.

application → It can be accessed from any page throughout the application.

Page → The jsp object can be accessed only from the same page where it was created.

Session Tracking Techniques in jsp:

Note: Refer the same heading in servlets

1) Session:

Note - Refer session implicit object in JSP

2) Cookies: The data will be stored in clients web browser.

There are two types:

i) Non persistent cookies :- Cookies will be destroyed when browser is closed

ii) Persistent cookies :- Cookies will be destroyed after certain period (or) age in seconds/minutes/days etc

3) URL Redirecting

Passing parameters to the URLs.

If you want to pass one parameter using param name then

demo.jsp?id = 10%

request.getParameter("id").

If you want to pass more than one parameter using param name then

demo.jsp?id=10&name=klu

request.getParameter("id")

request.getParameter("name")

If you want to pass one parameter without Param name then

demo.jsp?101

request.getQueryString()

4) Hidden form field

<input type="hidden" name="username".

request.getParameter("username")

~~JSP Action~~ Tag

JSP Action Tages:

It is used to control the flow b/w JSP pages.

i) <jsp:forward>
ii) <jsp:include>
iii) <jsp:Params>
iv) <jsp:useBeans>
v) <jsp:set Properties>
vi) <jsp:get Properties

mvc (model view control)
JSP
pojo
servlet
Bean-reusable components
(pojo class)
plain old java object
[setter & getter methods]

JSP Action Tags:

exception implicit object demo:

```
<%@page errorPage="displayerror.jsp"%>
<%@page isErrorPage="true"%>
```

errorPage is used to specify the page name in which exception will be specified

isErrorPage is used to set the Boolean value (by default false).

JSP CRUD Operations:

index.html (main-Home Page)

Userregistration.html
Insertuser.~~html~~jsp } user registration.

usersuccess.html

~~html~~
userlogin.~~html~~jsp } user login
loginfail.html
Checkuserlogin.jsp

viewusers.jsp
deleteuser.jsp } delete user
userdeletion.jsp

updateuserform.jsp } user update
updateuser.jsp

user home.jsp
user logout.jsp
session expiry.html

24/06/2024

JSP MVC Architecture:

M - Model
V - View
C - Controller

Model:-

Model means Bean where Bean is Pojoclass

plain old Java class

Pojoclass contains properties, setter methods, getter methods

Bean is reusable component.

front controller

View :-

View is Jsp file

Controller

Controller means servlet it will be acted as interface between model and view

Controller will handle → Request processing and data validation

View will handle → Response generation

Model will handle → Business logic and data manipulation.

eg: write servlet code to insert employee using model using model and display the response like "employee added successfully" using jsp file.

Directives:

1. Page directive → It defines attributes that will be applied to entire JSP page and the main purpose of Page directive is to import the package.

Syntax:

```
<%@ page (start)
- - - - -
%>
```

The following are the main attributes in page directives.

1. Language
2. Content type
3. Charset
4. Page encoding
5. Import
6. Session (True/False)
7. IsErrorPage
8. ErrorPage

2. Include directive →

Syntax: <%@ include file = "resource" %>

where resource is JSP file, servlet and HTML

Note:

<jsp:include> action tag - will include the resource at request time where as include directive will include the resource at translation time.

Servlet topics:

1. CGI vs Servlet
2. Types of servlets
3. Servlet Architecture / How servlet works?
4. Servlet life cycle. (3 programs)
5. Init() and Context() parameters.
6. Servlet chaining and collaboration.
7. Session tracking techniques
8. Servlet CRUD operations.

JSP Topics:

1. Introduction to JSP
2. JSP life cycle.
3. JSP Architecture
4. JSP Scripting elements.
5. JSP implicit & explicit
6. JSP CRUD Operations
7. JSP Action tags
8. JSP Directives
9. MVC architecture.