Web Design and Development

DATE: _____ CS506 DAY: _____

Lec 23 Multithreading

Intro:

Multithreading is the ability to do multiple things at once within the same application.

A thread - sometimes called an execution context. Thread is single sequential flow of control within progra

Threads are light weight.

A thread is easy to creat & destroy.

Every program has at least one thread


```
//File Threeloop Test.java
public class ThreeloopTest {
public static void main (string args[]){
// First Loop
for (int i=1; i<=5; i++)
System.out.println (" first" +i);
// second Loop
for (int j=1; j<=5; j++)
System.out.println ("second" +j);
// third loop
for (int k=1; k<=5; k++)
System.out.println ("third" +k)
}}//end main
```

Note:

Yaha 3 loops h agr to ye 3no ask sath-aik time prchly gy to multithreading hogi. or agr ye sequce sy apna kam kry gi to sequential boldin gy.

DATE: DAY:

Java Threads: Java includes built-in support for threading.

Creating threads in Java: 2 name (imp) meq

- Using Interface
- Using Inheritance

create threads by using Interface. 4

① Create class ② Runable run() method

③ object ④ starts () method

create threads by using Inheritance: 4

① Inherit class from thread class ② run () method

③ object ④ start () method

Thread Priorities:

Threads with higher priority are executed in preference to threads with lower priority.

A threads default priority is same:

Thread.Max_Priority (typically 10)

Thread.NORM_Priority (" 5)

Thread.Min_Priority (" 1)

DATE:

L#24 More on Multithreading

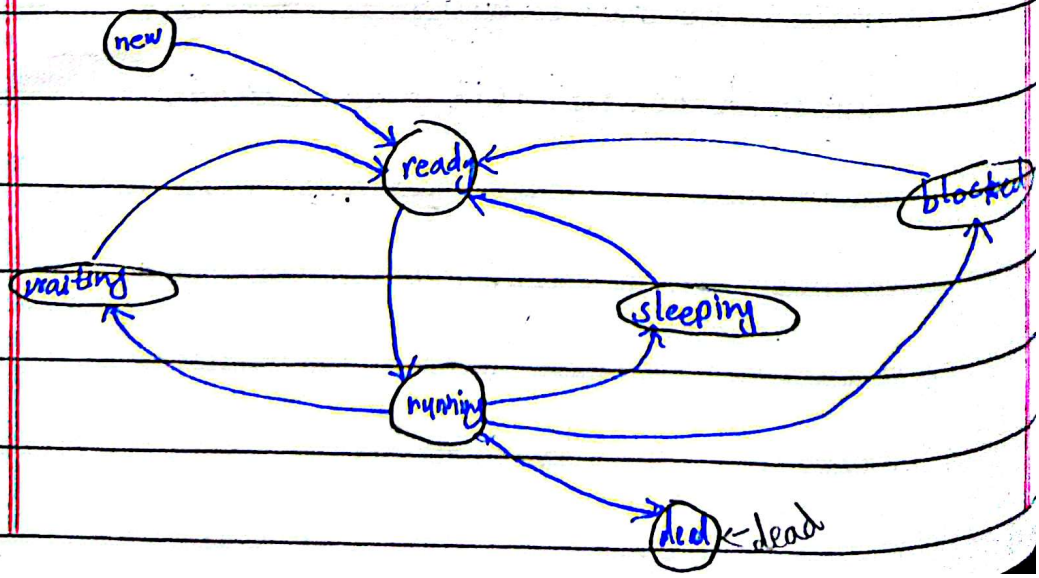
Useful Thread Methods:

imp

1- sleep method: (int time)

- sleep() method can be used for delay purpose. ie anyone can call Thread.sleep() method
- Causes the currently executing thread to wait for time specified.
- Threads come out of sleep when specified time interval expires or when intrupted by other thread.
- High priority thread should execute sleep method after some time to give low priority threads to run otherwise starvation may occurs.

2- Thread States: Life Cycle of Thread.



DATE: _____ DAY: _____

thread
new state mai "create hoga phr wo
start() method invoked
ready m chla jay ga usky bad
thread assigned processor and running
running m jay ga or complete ho k phr
thread completed
dead m chla jay ga.

ab jo baki states h un m
work k according jana h jesy
wait krwana h to waiting state
m jay ga running k bad or
phr udhr sy dobara ready m
jay ga or phr running m or agy
aisy his same sleeping or
blocked states m hoga.

Threads joining:

It is Used when a thread wants to
wait for another thread to complete
its run() method.

Lec 25: Web Application Development

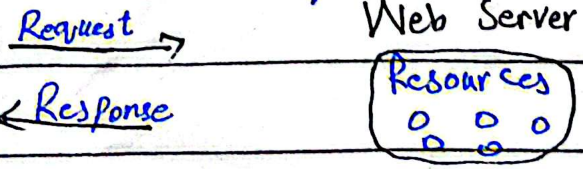
Web Application:

It is a piece of code running at server which facilitates a remote user connected to web server through HTTP protocol. Client sends a request to server which sends back respons or error msg.

HTTP Basics:

A protocol defines the method and way of communication b.w two parties. جیسے کہ بچہ اور دوست سے مختلف انداز میں بات کرتے ہیں۔ بچہ سے فارمل اور دوست سے Informally

HTTP protocol uses various types of request to response messages.



imp

Parts of an HTTP request: (6)

1. Request Method: It tells server type of (~~method~~) action that a client wants to perform.
2. URI: Uniform Resource Indicator specifies address of required document or resourse.

مطلب کہ جو client نے request بھیجی ہے اس کا address

DATE: DAY:

3. Header Files: Headers can be used by client to tell server extra information about request. اس میں وہ information لکھی جاتی ہے جو کہ URI میں نہیں ہوتی۔

4. Body: Contains data sent by client to the server. جو request بھیجی جاتی ہے اس سے related جتنی بھی info ہوتی ہے وہ Body میں آتی ہے۔

5. Other request headers like FROM (email of person responsible for request) FORM

6. Request Parameters: Request can also contain additional info in the form of request parameters. (additional information add krni hoti like Gmail, name)

Parts of HTTP response: (3)

1. Result Code: A numeric status code, its description اس میں status code اور اس کی description ہوتی ہے۔

2. Header Fields: Servers use these fields to tell client about server info like configurations. In اس میں server کی info موجود ہوتی ہے۔

3. Body: Data sent by sereserver as response اس میں data لکھا ہوتا ہے۔

HTTP Response Code: (5) اس

100-199 : 100: Continue with partial request. Means half request send hui h.

200-299 : 200: Everything is fine. Means k several cases a jaty to diff jaga sy data milta.

300-399 : 300: can be found several places.

400-499 : 400: values indicated an error by client.

500-599 : 500: codes in 500s signify error by server.

DATE: DAY:

imp SQ mcqs

Server Side Programming:

1- Static Pages 2- Dynamic Pages

1- Static Pages: Means simple page

But is m ye hota k is m jitny b user aik time m request krny gy unsbko aik response mily ga.

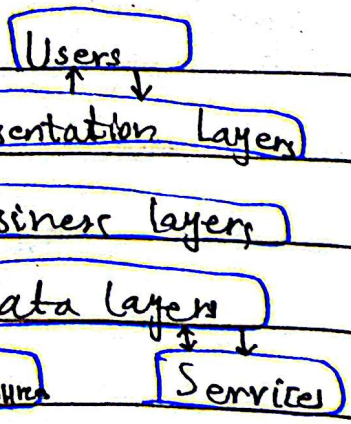
A static web page is a simple HTML (Hyper Text Markup Language) file. When a client requests an HTML page the server simple sends back response with required page.

2- Dynamic Pages:

But is m server wo response krta h jo client request krta h

In dynamic web pages server executes an application which generates HTML web page according to specific request coming from client. (Why)? Thats why we use dynamic pages because bcz server response according to our need.

- Layers and Web Applications:



DATE: DAY:

1- Presentation Layer: (User k liay interface provide kr dia jata h jis sy wo application sy interact krta.

Provides a user interface for client to interact with application. This is only part of application visible to clients.

2- Business Layers.. also service layer

The business or service layer implements the actual business logic or functionality of application. Example: In online shopping system it handle transaction method.

3 Data layer:

This layer consists of objects that represent real-world business objects such as an Order, products and so on.

Java-Web Application Technologies:

1- Java Sarvlets
2- JavaServer Pages
3- Java Server Faces

DATE: DAY:

L#26 Java Servlets

Servlets sy phly hm CGI Programming pr dynamic or static pages bnaty thy But ab Servlets sy bnaty.

Servlets are java technology's answer to CGI programming. BY use of servlets we create dynamic and static pages.

Advantages:

1- Convenient Easy to use
2- Efficient It handled by lightweight java threads
3- Powerful easily things to do
4- Portable portable across web servers
5- Inexpensive Not expensive

Software Requirements:

1- J2SE 2- Web engine

Jarkata is an Apache project and tomcat is subproject

L#27 Creating a Simple Web Application in Tomcat

Servlet Types:

Servlet related classes are included in two main packages:

1- javax.servlet 2- javax.servlet.http

DATE: jb ap apna servlet tyar DAY: krny gy to ye do hongy.

When you write your own servlet, you can subclass ~~(your)~~ from
1- GenericServlet 2- HttpServlet (imp)

1- GenericServlet class: (hm yaha own servlet m packages use kr rhy h.)
- It is available in javax.servlet package.
- Implements javax.servlet.Servlet.
- Extend your class from this class if you are interested in writing protocol independent servlets. (اگر آپ پروٹوکول انڈیپنڈنٹ سروولٹ لکھنا چاہیں تو آپ کو اس کلاس سے extend کرنا ہوگا)

2- HttpServlet class:
- Available in javax.servlet.http package.
- Extends from GenericServlet class.
- Extend your class from HttpServlet, if you want to write HTTP based servlets.

Types of HTTP requests: (6) (imp S.Q mcqs)
Most imp ones are get & post: (Request krni h server sy get yani leny k liay. {Method ko call krny k liay. doGet() method})
1. GET: Requests a page from the server. This is normal request used when browsing web pages.
2- POST: This request is used to pass info to the server. Its most common use is with HTML forms. doPOST() meth

DATE: DAY:

3-PUT : Used to put a new page on a server.

4-DELETE: Used to delete a web page from the server.

5-OPTIONS: Intended for use with web server, listing supported options.

6-TRACE: Used to trace servers.

L#28 Servlets Lifecycle

↳ Stages of Servlet lifecycle:

•1 Initialize •2 Service •3 Destroy

1 Initialize: In initialization stage the servlet is first created. The webserver invoked init() method of servlet. init() is only called once, not called for each request.

Aik bar servlet load ho jay phr wo client ki request ko bhi deky ga.

2- Service:

The service() method is the engine of servlet. It processes client's request. On every request from client, server creates new thread and calls service() method. It is efficient.

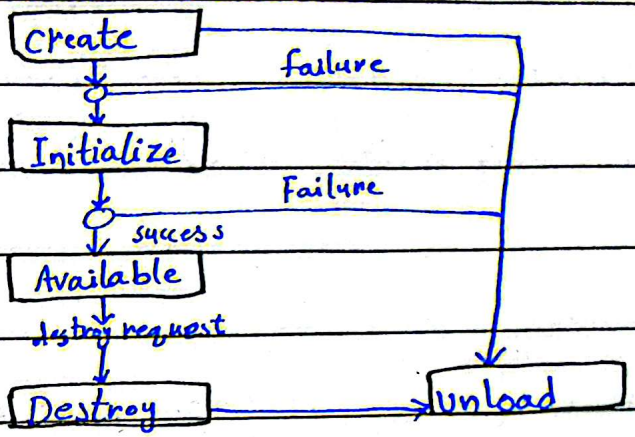
Ye thread rs liay create krta h taky kam asan ho jay thread s.y.

jb hm kam complete kr lety h to usy destroy kr dety h.

DATE: _____ DAY: _____

3- Destroy: The web server may decide to remove a previously loaded servlet instance.

Lifecycle



Note:
Agr kam thk hota gya to Seedha jata jay ga but in case of failure wo dusri side chl jay ga unload m.

Types of Data send to Web Server (imp) most

- Form Data
  ڈیٹا کو نارمل شکل میں سرور کو بھیجنا۔
  like. Registration Form.
- HTTP Request Header Data
  ڈیٹا کو HTTP ریکوسٹ کے ذریعے بھیجنا
  like cookies, browser type etc.

L#29: More on Servlets

1- Response Redirection: ریسپانس کو مزید آگے بھیجنا ہے۔

We can redirect the response of Servlet to another application resource (HTML page or JSP) (another servlets)

Two forms of response redirection: (imp)

- Sending a standard redirect.
- Sending a redirect to an error page.

DATE: _____ DAY: _____

1- Sending a Standard Redirect:

اس کا مطلب ہے کہ ایک page سے دوسرے page پر جا سکتے ہیں physically اور ایک page پر دوسرے page کا لنک دیں تو وہ redirect ہو جائے گا

Using response.sendRedirect("myHtml.html") method, a new request is generated which redirects the user to sepcified URL.

Sending a redirect to an error page.

اس میں ہم ایک page پر لنک ڈھونڈیں گے اور وہ نہیں ملے گا تو Error ہوگا

We can use response.sendError() to show user an error page.

An error code response.sendError(int msg) method

2- Request Dispatcher: (imp) mcqs

It provides a way to forward or include data from another source.

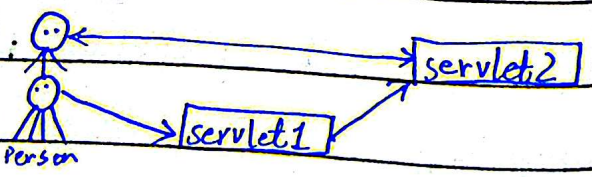
Two imp-- methods of RequestDispatcher:

اس کا مطلب ہے ایک page پر ہو اور دوسرے page پر بھیجنا ہو تو فارورڈ ہو request or response

- forward(ServletRequest req, ServletResponse resp)
- include(,)

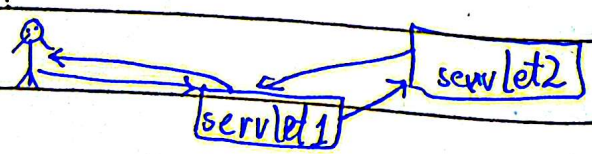
اس کا مطلب ہے کہ ایک page پر کھڑے ہو اور سارا ڈیٹا دوسرے page پر شو کروا دے

Forward method:



جیسے اب فارورڈ میں ایک سرور سے دوسرے کے پاس request گئی ہے پھر اوپر سے Person کے پاس

Include method:



اب اس میں Person ریکویسٹ بھیجتا ہے servlet1 کو پھر وہ servlet2 کو بھیجتا ہے اور پھر سارا ڈیٹا اکٹھا کر کے واپس servlet1 پر آجاتا اپنے include page پر تاکہ show ہو سکے اپنا data بھی اور دوسرے کا بھی۔

DATE: DAY:

L#30 Dispatching Requests

1- HttpServletRequest Methods: 7 imp mcqs

name and work learn.

1. setAttribute (String, Object)
Is m ap maya attribute set krty h, setattribute k method. call krty ho
2. getAttribute (String)
Ab jo attribute set kia usy lena ho get krna ho
3. getMethod() : is m ap koi bmethod easily get krskty
4. getRequestURL (): request ko get krny k liay
5. getProtocol (): It returns the name and version of protocol.
6. getHeaderNames(): header names ko get krna
7. getHeaderName(): header name ko get krna or na milny pr Null value dy deta h.

2- HttpServletResponse Methods: (4)

1. SetContentType(): Hr content type servlet k pas hoti to contentType k liay ye method hoga.
2. setContentLength(): for content length in integers. Parameter.
3. addCookie(). cookies ko add krny k liay.
4. sendRequest(): Ap ny aik page sy dusry pr jana hota to URL dy dety

L# 31 Session Tracking *imp*

jb koi system ap k sath (server k sath) attach hota to uska login logout time ap ka session kehlata h, ab knsa person ap k system sy attach (connect) hua uski sari details ~~tk~~ or information ko track

↳ krna "session tracking" h.

session tracking k 3 triky h.

(imp) **Three Typical Solutions:** mostly just name puchy jaty

1. • Cookies: cookies is a piece of text that a web server can store on client's hard disk.
2. • URL Rewriting
3. • Hidden Fields

↳ **Sending Cookies to Browser:** 3 steps — How to send code to browser?

1- Create a Cookie Object: Cookie constructor

```
Cookie c = new Cookie("name", "value");
```

2- Setting Cookie Attributes:

```
c.setMaxAge(60); //expired after one hour
```

3- Place Cookie into HTTP response:

```
response.addCookie(c);
```

Cookies Attribute کو چینج کرنے
بعد فوراً cookie ریلوڈ نہیں کو
browser کو cookie نہیں ملے گی -

DATE: DAY:

L7 Reading a Cookie from Client. 2steps ~~into HTTP Response.~~

1 Reading incoming cookies

```
Cookie cookies[] = request.getCookies();
```

2. Looping down Cookies Array:

when have array, you can iterate over it

L# 32 Session Tracking 2

2- URL Rewriting:

It is also for session tracking

We add extra information, add parameters.

This extra info — shows who's user

Disadvantages:

- Less secure then cookies
- It generates more network traffic.
- It is expensive.
- It works only links

3- Hidden form fields: It is also session tracking method. It does not effect appearance of HTML Page. It contain info. that needed to send to server. It also store information.

DAY:

DATE:

imp ↳ Working with HttpSession: step by step

must practice and learn.

1- Getting the user's session object:
(for creating object)

getSession() ← method

HttpSession sess = request.getSession(true);

2- Storing information in a Session:
(for getting object)

setAttribute() method

sess.setAttribute("sessionid", "123");

You can store no. of key and their values in pair.

3- Looking up info... associated with a Session:
(session ko info... get krny ke liy)

getAttribute() method

String sid = (String) sess.getAttribute("sessionid");

4- Terminating a Session:
(session ko khtm krny ke liay)

invalidate() method

sess.invalidate()

(Most imp)

↳ Some Methods of HttpSession:

- setAttribute(String, Object)
- getAttribute(String) get value from session if not found shows null
- removeAttribute(String)
- getId() for unique identifier

DATE: DAY:

- getCreationTime() : Returns time which session first created.
- getMaxInactiveInterval(), setMaxInactiveInterval(int)

L#33 : Address Book Case Study Using Servlets

Package: work on small project All java files into single directory.

but work on bigger project the number of files increasing, putting also all files into same directory. In java we can avoid this sort of problem by using Packages. (یعنی ساری فائلز کو ایک directory میں رکھ سکتے ہیں بلکہ چھوٹے چھوٹے Packages میں)

"A set of java classes organized for convenience in the same directory to avoid the name collisions."

↳ How to create a package: imp

Keyword package with name of package we want to use on top of our source file.

~~int~~ world package

Note: like hm agr world k name sy likhy to "package world" likhy gy.

DATE: DAY:

↳ How to use package:

By using import keyword

//we can use any public classes inside world package

import world.*;

(imp)

↳ JavaServer Pages: (JSP)

JSP is a technology used to create dynamic pages. It allows developers to embed Java code within HTML pages. JSP enables development of web-based applications with the help of tags. It is easy method.

Java + HTML = JSP

Advantages:

- You can embed Java code directly within your HTML.
- Runs on any server that supports Java.
- JSP makes smart and fast websites.
- Full access to Java's extensive libraries, APIs.
- JSPs divide and conquer the problem of presentation and business logic.
- It is inexpensive. • It is portable.

DATE: DAY:

L#34 : Java Server Pages

JSP Ingredients: (imp)

Directive Elements
provides global controls of JSP <%@ %>
Scripting Elements
JSP comments <%--- %>
declarations <%! %>
expressions <%= %>
scriptlets <% %>
Action Elements.
Special JSP tags <jsp:---/>

L#35 Java Server Pages

Implicit Objects:

To simplify code in JSP expressions, you are supplied with eight automatically defined variables called implicit objects.
Three imp variables are: (imp)

1. request
2. response
3. out

// ye double slash comments likhny k liay use kr gaya.

DATE: DAY:

1- request:

Is ka mtlb h k aik page sy dusry pr request bhjy gy. mtlb ap isi page pr hngy or request dusry page pr bhjy gy.

This variable is type of HttpServletRequest, associated with the request. It gives you access to the request parameters.

2- response:

Is m ap ko response milta h.

This variable is type of HttpServletResponse, associated with the response to client.

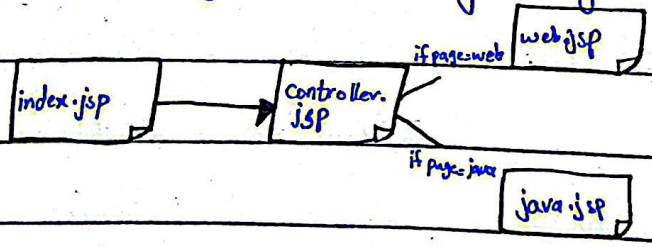
3- out:

This is thy object of JSPWriter used to send output to the client.

Code Example: Use of Implicit Objects:

"4" JSP pages.

user "java or web" m sy koi aik button choose kry ga jiski usy info chahiay hogi or request submit krwa dy ga.



index.jsp (must practice)

```
<html>
<body>
<h2>Select the page you want to visit</h2>
```

DATE: DAY:

```
<form name = "myForm" action="controller.jsp">
<h3>
<input type="radio" name="page" value="web" />
Web Design & Development
</h3>
<br>
<h3>
<input type="radio" name="page" value="java" />
Java
</h3>
<br>
<input type="submit" value="Submit" />
</form>
</body>
</html>
```

Ye index.jsp ka code h jismy hm ny web or java k do buttons lgaey jb jiski zrurt hegi usy use kr sky or select krty k kaun hm ny submit b krwana. h.

controller.jsp:

```
<html>
<body>
<!-- scriptlet -->
<%
// page ko read krna h or radio button ko use krna
```

DATE: DAY:

```
String pageName = request.getParameter("page");
//decide krna h k knsa page move krwana
// implicit requests k trriay web ya java
if(pageName.equals("web"))
{
  response.sendRedirect("web.jsp");
}
else if (pageName.equals("java"))
{
response.sendRedirect("java.jsp");
}
%>
</body> </html>
```

/ye output dikhany k liay

web.jsp

```
<html>
  <body>
// use of out implicit object, to generate HTML
<%
out.println("<h2>" + "Welcome to Web Design
& Development Page" + "</h2>");
%>
</body>
</html>
```

comment →

DAY:

java.jsp

```
<html>
  <body>
// use of out implicit object to generate HTML
<%
  out.println ("<h2>" + "Welcome to Java Page" + "</h2>");
%>
  </body>
</html>
```

Remaining 5 implicit objects:

is of Types:

1. Session : HttpSession ← work with session object
2. application: ServletContext ← allow to store values
3. config: ServletConfig ← Represents JSP configuration
4. pageContext: javax.servlet.jsp.PageContext ← give access
5. exception : java.lang.Throwable ← for exception

- Page : java.lang.Object

JSP Directives: (3)

Used to convey special processing info. about page to JSP container.

- Specify page settings
- To include content from other resources
- To specify custom-tag libraries

DATE: DAY:

(imp) Format of Directives:

1- Page: <%@ page {attribute = "val"}* %>
java ka jo khud package import kia wo is page m add krna

2- include:
<%@ include {attribute = "val"}* %>
jo page ap ny bnaya or ap usko jsp m include krna chahty h to include method

3- taglib:
<%@ taglib {attribute = "val"}* %>
is m hr trah ki libraries hoti.

(imp mcqs) The lists of attributes used with page directive.

- language: "java"
- extends: "package.class"
- import: "package.* Package.class"
- session: "true | false"
- info: "text"
- contentType: "mimeType"
- isThreadSafe: "true | false"
- errorPage: "relativeURL"
- isErrorPage: "true | false"

imp JSP Life Cycle Methods:

1- jspInit(), 2- jspService(), 3- jspDestroy()

DATE: DAY:

L#36

JavaBeans:

JavaBeans are reusable software components in java, designed to be easily integrated into applications.

JavaBeans Design Conventions:

- Bean class must have "zero argument contructor"
- Not have any public instance
- Private value should be accessed by "setters/getters"
- A bean class must be serializable.

L#37 JSP Action Element and Scope

- JSP Action Elements: 3

To work with javaBeans.

1-<jsp:useBean/>

format:

```
<jsp:useBean id="name"
scope="page|request|session|application"
class="package.Class"
/>
```

As it is formats ko yad krna or practice krni

DATE: DAY:

2- <jsp:setProperty />

format

<jsp:setProperty name="m" property="name" value="ali" />

3- <jsp:getProperty />

format

~~<jsp:getProperty name="m" property="name" value="ali" />~~

<jsp:getProperty name="beanName or id" property="name" />

L#38 JSP Custom Tags

L> Custom Tags?

(user k bnay hy tags.)

"A user defined component that is used to perform certain action."

Why build Custom Tags?

With custom tags, it is possible for web page designer to use complex functionality without knowing any java.

DATE: DAY:

Advantages: (imp) S.Q

- It is more better than JavaBeans tags.
- Have access to all JSP implicit objects like out, request etc. (child k attributes to bna skty)
- Can be customized by specifying attributes.

L> Types of Tags: (imp)

1. Simple Tag
2. Tag with Attribute
3. Tag with body

1. Simple Tag:

- It has start and End of tag.
- No body
- No attributes
  - example: <mytag:hello /> (Tag library → Tag name)

2. Tag with Attributes:

- Start and End of tag
- Attributes within tag
- No body

example: <mytag:hello attribute="value"/>

DATE: DAY:

3- Tag with body:

- Start and End of tag
- May be attributes
- Body enclosed within tags

example:

```
<mytag:hello optional_attributes....>
        some body
</mytag:hello>
```

(Phy built in tags hoty thy but ab khudsy bnany h.)

↳ Steps for Building Custom Tags:

1- Develop the Tag Handler class
2- Write Tag library Descriptor (tld) file
3- Deployment

1- example

```
public class MyTagHandler extends
    simpleTagSupport {
        ------ }
```

2- ◦ Tag library version
◦ JSP version
◦ Tag name
◦ Tag Handler class name
◦ Attribute names etc.

DATE: DAY:

3- Deployment:

place class → myapp/WEB-INF/classes folder of web appli...

L#39 MVC

Model View Controller (MVC)

'M' k andr database hoti yani jara ki sari files.

'V' front ka page hota is m like HTML files.

'C' sara logic jesy kis student ko login krna etc is m hota.

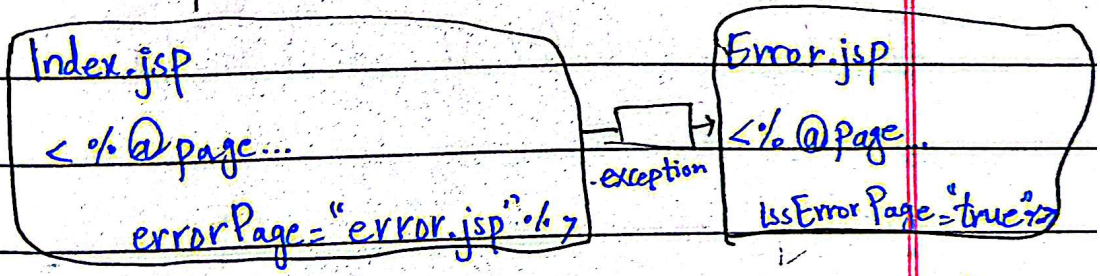
Error Page: (imp)

Error pages enables you to customize error message.

{Error page ko hm alg sy bnay gy or jsp k pages mai lga dain gy to jb b koi error ay ga to ye page ap ko bta dy ga yani ye handler bn jay ga.

Note:

(Build error page:)



Note:

Index waly page m hm errorPage ko likhy gy or error page alg sy bnay gy taky jo iss ye ho foran us page sy hmy pta chl jay k ye error h, or phly Page m link h to is sy pta chly g.

DATE: DAY:

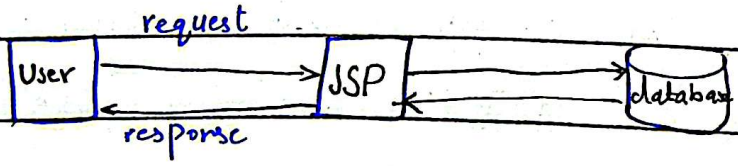
Ingredients of Address Book:
JavaBeans, Java Server Pages and Error Page.

↳ Evolution of MVC Architecture:
* MVC Model 1, * MVC Model 2 architecture
Standard web based application framework is JavaServer Faces (JSF).

is ki diagram mbl k ss m h jo imp h.

↳ MVC Model 1:
A Model 1 architecture consists of a Web browser directly accessing Web-tier JSP pages.
In MVC Model 1 architecture, the JSP page alone is responsible for processing the incoming request and replying back to the client.

L#40: Model 2 Architecture MVC
in Model 2
Page ~~with~~ - centric approach: 3 steps



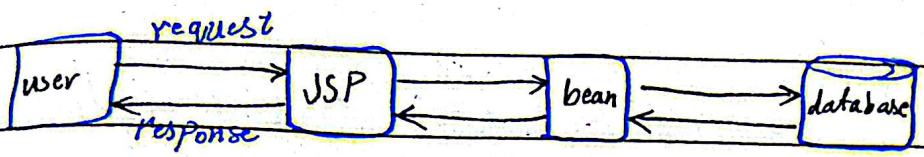
Note:
{ is m request jay gi JSP ko udhr sy phr database ko or phr udhr sy response wapis ay ga.

DATE: DAY:

4-steps

Page-with-Bean Approach:

Ye MVC Model 1 k liay h.



Note

ismy bean is liay use hua bcz java ki jo files h or tags sb ismy h to us sy wo connection dy k output dy ga.

MVC Model 2 Architecture:

Iski Diagram b mbl m h udhr sy krni imp h.

This archi:- introduces a controller. It is based on:

- The Request URL
- Input Parameters
- Application state

1- JSP is Right choice as Controller?

Ans: No, use Servlets as controller.

2- Reason for introducing JSPs?

Ans. JavaServerPages are built for presentation (view) only so JSP is really not good place. JSP performing the job of controller

DATE: DAY:

L # 41 Layers and Tiers

imp mcqs

Layers - represents logical view of application.

Tiers: - represents physical view . . .

The layerd architecture based on three {layers:

mcqs ors bi just names and few details like what is in it

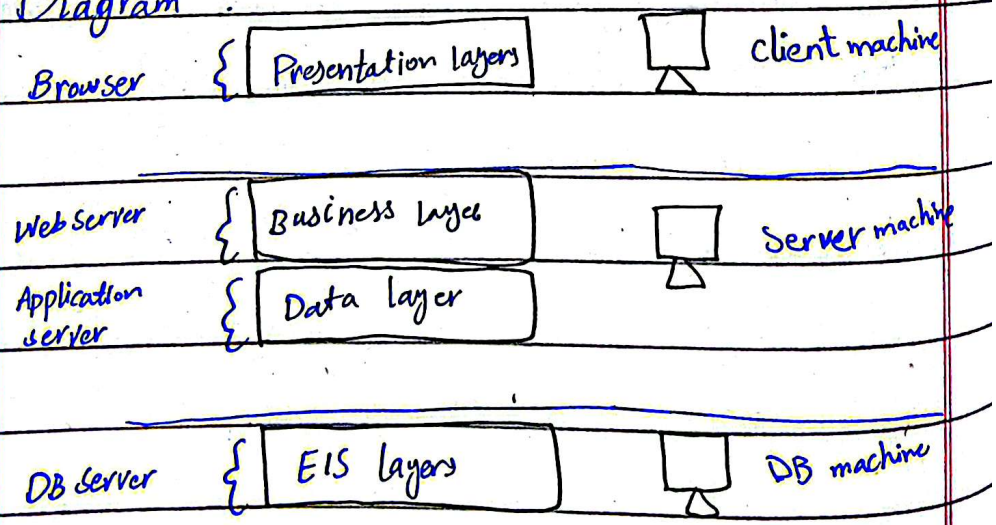
1. Presentation Layer: jo apky samny hota wo presentation layer h. (is m login hona sari info isi mein hoti) presentation sy any k bad business layer
2. Business Layer: pury data ko accept krti h phr next obj
3. Data Layer: phr business sy any k bad final data is layer. m ajata

imp mcqs

Tiers: An application can be based on single-tier, Two-tier, Three-tier or N-Tier.

Diagram:

Three-Tier diagram.



Note:

client machine k pas presentation layer hoti

Server k pas Business or data layers hoti or

DB machine k pas EIS layers bs ye yad rkhna

DATE: DAY:

imp diagram yad krain.

Layers Support in Java:

Layers	Java/J2EE Technology
Client Presentation	HTML/Applets (for view)
Server Presentation	JSP / Servlets Frameworks (Struts, JSF etc)
Business	JavaBeans/EJB
Data	DAO/Connectors

L#42 Expression Language (EL)

JavaServer Pages Standard Tag Library (JSTL) (Abbriviation) a skte

JSP before and after EL: iski dono diagrams mobile m h udhr sy read krni.

Expression Language Nuggets:

important pieces of EL:

- Syntax of EL: ${validExpression} format ${jwt $ ye liya k URL dena sb.
- Expression and Identifiers: all identifiers -> to all implicit objects and all other to -> scoped variables.
- Arithmetic, logical & relational operators:
- Automatic type conversion:
- Access to beans, arrays, lists, maps.
- Access to set of implicit objects:

DATE: DAY:

imp ↳ Adding 2 numbers using EL: {Ye add krna itna sy easy ho gya.}

Result is: ${param.num1 + param.num2}

↳ EL Accessors:

The dot (.) and bracket [] operators.

Note: variable single object ko hasil kr skty object property ko easily hasil kr skty is m.

Dot (.) operator:

${person.name}

Note: → (identifier, Property)

Bracket ([]) operator: is map array and collections hasil krskty

${myMap["id"]}

key bracket k andr specify ki jaty gi is m.

imp ↳ Robust Features: characteristics

- Multiple expressions can be combined in EL.
  e.g: ${"Hello" ${user.firstName}${user.lastName}}
- EL also supports automatic type conversion.
  Note: begin int data type h but btana hi pra bcz EL uska
  e.g: begin = "${student.marks}"
- if object/identifier is null, no NullPointerException would be thrown.

DATE: DAY:

L#43 JSTL JavaServer Pages Standard Tag Library

JSTL is a collection of custom tags libraries, that implement general-purpose functionality, iteration, conditionalization, formating and manipulation of XML.

Four tag Libraries: these are all contains tags for specific purpose like

Core (Predix C): for conditions, control flow

XML (X) manipulation: for XML parsing, processing

SQL (sql): for accessing and working with database

Internationalization & formatting (fmt): to support msgs, text number

L#44 Client Side Validation & JavaServer Faces (JSF)

Client Side Validation: jesa ap ko kam krny k liay kaha jay or ap nhi krty to apko validate kr dia jata

Means: Uni m form fill krny ko dair or ap w trah sy ma krain to client side validation hi

Why its Cood? Acha is liay h k fori bata dety k kaha mistake.

It's fast form validation

You can safely display only one error at a time.

DATE: DAY:

JavaServer Faces (JSF):

JSF technology simplifies building user interface for web application.

→ Framework:

- Struts
- Trapestry

→ JSF Validators:

Validators make input validation simple and save developers hours of programing.

built-in validators:

- DoubleRangeValidator
- LongRange Validator
- LengthValidator

L#45 JavaServer Faces

Web Services:

Note: Data ko exchange krny k liay Protocol ko mad-e-nzr rkh a gya h

Web services are Web-based enterprise applications that use open, XML-based standards and transport protocols to exchange data with calling clients.

DATE: DAY:

Web service definition by W3C:

According to W3C, "A Web service is a software application identified by a URL."

Why Web Services? (Advantages)

- Interoperable ← connects web-based standards
- Economical ← Recycle components
- Automatic ← No human
- Accessible ← accessible on web
- Available ← services on any device, any time
- Scalable ← No limits

Types: (imp) name

Data Providers: service providing stock quotes

Business-to-business process integration → purchase orders

Enterprise application integ — Diff applications work together

Difference (imp) b.w Web page & Web Service:

Web page	Web Service
1. Has a UI.	No GUI.
2. Interacts with user.	Interacts with application.
3. Works with web browser client.	Works with any type of client.

Figure 4: HTTP Response Example

25.3.3 HTTP Response Codes

- An HTTP Response code tell the client about the state of the response i.e. whether it's a valid response or some error has occurred etc. HTTP Response codes fall into five general categories

  - **100-199**

    Codes in the 100s are informational, indicating that the client should respond with some other action.
    100: Continue with partial request.

  - **200-299**

    Values in the 200s signify that the request was successful.
    200: Means everything is fine.

  - **300-399**

    Values in the 300s are used for files that have moved and usually include a Location header indicating the new address.

- **400-499**

  Values in the 400s indicate an error by the client.

  404: Indicates that the requested resource is not available.

  401: Indicates that the request requires HTTP authentication.

  403: Indicates that access to the requested resource has been denied.

- **500-599**

  Codes in the 500s signify an error by the server.
  503: Indicates that the HTTP server is temporarily overloaded and unable to handle the request.

404: Indicates That The Requested Resource Is Not Available

2 Initialization method

One way is to override init() method as shown in the code below. The ServletConfigobject can then be used to read initialization parameter.

```
public void init(ServletConfig config) throws ServletException {
String name = config.getInitParameter("paramName");
}
```

Another way to read initialization parameters out side the init () method is

- Call getServletConfig() to obtain the ServletConfig object
- Use getInitParameter() of ServletConfig to read initialization parameters

```
public void anyMethod() // defined inside servlet
{
ServletConfig config = getServletConfig();
String name = config.getInitParameter("param_name");
}
```

Example Code: Reading init parameters

MyServlet.javawill read the init parameter (log file name) defined inside web.xml. The code is given below:

In the later part of this handout, we'll also learn how to make a simple web application using servlet.

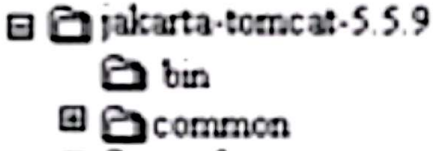
imp

27.1 Standard Directory Structure of a J2EE Web Application

A web application is defined as a hierarchy of directories and files in a standard layout. Such hierarchies can be used in two forms

- Unpack
  - Where each directory & file exists in the file system separately
  - Used mostly during development
- Pack
  - Known as Web Archive (WAR) file
  - Mostly used to deploy web applications

The *webapps* folder is the top-level Tomcat directory that contains all the web applications deployed on the server. Each application is deployed in a separate folder often referred as *"context"*.



```
import java.io.*;
import java.net.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class MyServlet extends HttpServlet {

/* this method is being called by both doGet() and doPost().We usually follow this practice, when we
are not sure about the type of incoming request to the servlet. So the actual processing is being done in
the processRequest().
*/
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
response.setContentType("text/html");
PrintWriter out = response.getWriter();

out.println("<html>");
out.println("<body>");
out.println("<h1>Start of include request </h1>");
out.flush();

// getting the object of ServletContext, that will be used to
// obtain the object of RequestDispacther
ServletContext context = getServletContext();
```

```
</servlet-mapping>
</web-app>
```

imp

Code Example: Request Dispatcher – forward

As discussed earlier, we can forward the request processing to another resource using forward method of request dispatcher. In this example, the user enters his/her name and salary on the index.htmland submits the form to FirstServlet,which calculates the tax on salary and forwards the request to another servlet for further processing i.e. SecondServlet.

index.html

```
<html>
<body>
<form method="POST" ACTION = "firstservlet" NAME="myForm">
<h2> Enter your name</h2>
<INPUT TYPE="text" name="name"/>
<br/>
<h2> Salary</h2>
<INPUT TYPE="text" name="salary"/>
```


© Copyright Virtual University of Pakistan


```
<BR/><BR/>
<INPUT type="submit" value="Submit"/>
</form>
</body>
</html>
```

FirstServlet.java

```
import java.io.*;
import java.net.*;
import javax.servlet.*;
import javax.servlet.http.*;
```

```
public class FirstServlet extends HttpServlet {
// this method is being called by both doGet() and doPost()
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

// getting value of salary text filed of the HTML form
String salary = request.getParameter("salary");

// converting it to the integer.
int sal = Integer.parseInt(salary);
// calculating 15% tax
int tax = (int)(sal * 0.15);
// converting tax into string
String taxValue = tax + "";
// request object can store values in key-value form, later it
// can be retrieved by using getAttribute() method
request.setAttribute("tax", taxValue);

// getting object of servletContext
ServletContext sContext = getServletContext();
// getting object of request dispatcher
RequestDispatcher rd = sContext.getRequestDispatcher("/secondservlet");

// calling forward method of request dispatcher
```

- **EL Literals**

The list of literals that can be used as an EL expression and their possible values are given in the tabular format below:

Literals	Literal Values
Boolean	true or false
Integer	Similar to Java e.g. 243, -9642
Floating Point	Similar to Java e.g. 54.67, 1.83
String	Any string delimited by single or double quote e.g. "hello" , 'hello'
Null	Null

Examples of using EL literals are:

```
${ false } <%-- evaluates to false --%>
${ 8*3 }  <%-- evaluates to 24 --%>
```


Difference b.w EL Literals & EL Operators?
imp S.Q

- **EL Operators**

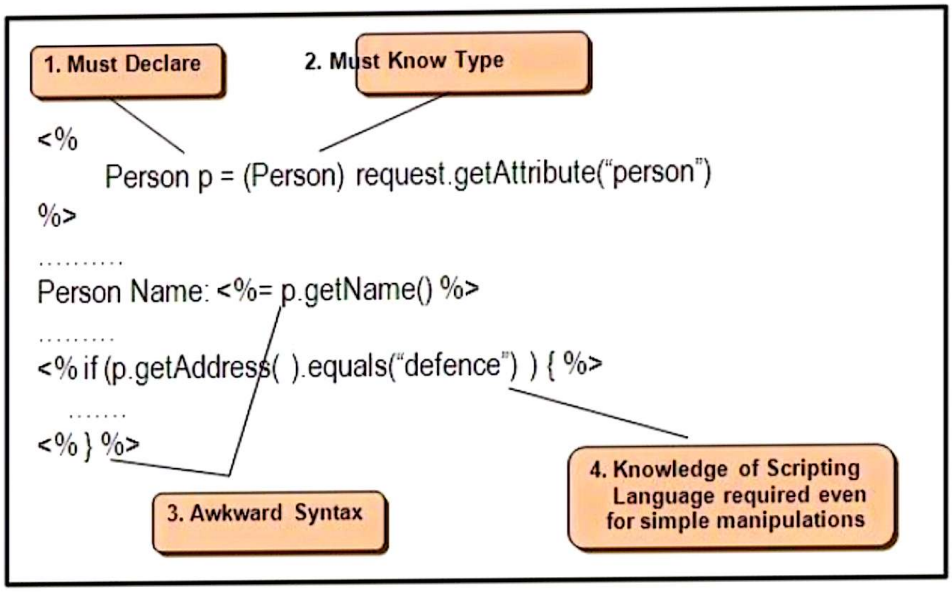
The lists of operators that can be used in EL expression are given below:

Type	Operator
Arithmetic	+ - * / (div) % (mod)
Grouping	()
Logical	&&(and) ||(or) !(not)
Relational	== (eq) != (ne) < (lt) > (gt) <= (le) >= (ge)
Empty	The empty operator is a prefix operation used to determine if a value is null or empty. It returns a Boolean value.
Conditional	?:

Let us look at some examples that use operators as valid expression:

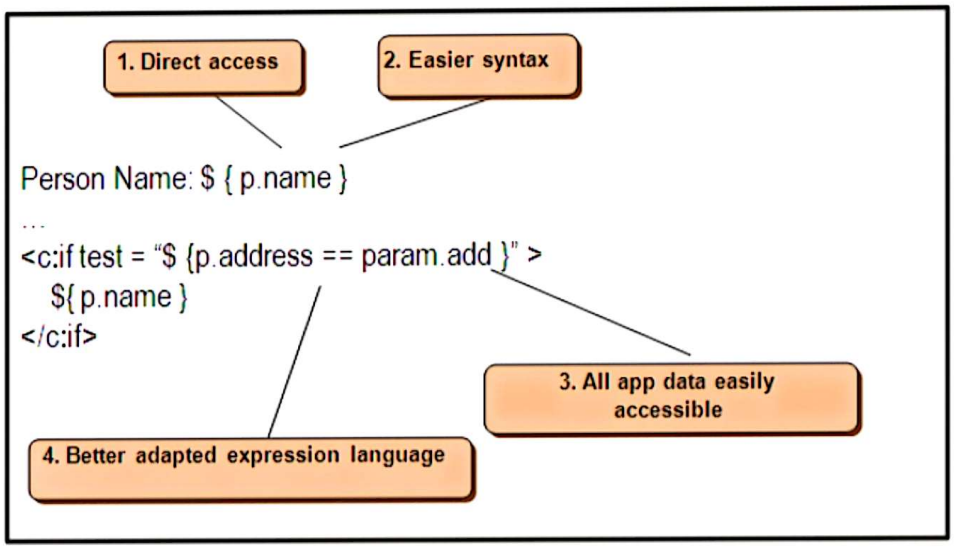
- ${ (6*5) + 5 } <%-- evaluate to 35 --%>
- ${ (x >= min) && (x <= max) }
- ${ empty name }
  - Returns *true* if name is
    - Empty string (""),
    - Null etc.
- **EL Identifiers**

Before & After using EL?



JSP before EL

Contrary to the above figure, have a look on the subsequent figure that gives you a hint how useful EL can be?



JSP After EL

Examples of using implicit objects are:

Examples of implicit objects...

- ${ pageContext.response }
  - Evaluates to `response` implicit object of JSP
- ${ param.name }
  - This expression is equivalent to calling `request.getParameter("name");`
- ${ cookie.name.value }
  - Returns the value of the first cookie with the given name
  - Equivalent to

```
if (cookie.getName().equals("name"){
String val = cookie.getValue();
  }
```

42.3.5 Using Expression Language

Expression Language can be used in following situations

- As attribute values in standard & custom actions. E.g.

```
<jsp:setProperty id = "person" value = ${....} />
```

- In template text – the value of the expression is inserted into the current output. E.g.

```
<h3> $ { ..... } </h3>
```

- With JSTL (discussed in the next handout)

43.4 Twin Tag Libraries

JSTL comes in two flavors to support various skill set personal

- **Expression Language (EL) version**
  - Dynamic attribute values of JSTL tags are specified using JSTL expression

imp learn

language (i.e. `${ expression }`)

- The EL based JSTL tag libraries along with URIs and preferred prefixes are given below in tabular format

Library	URI	Prefix
Core	http://java.sun.com/jsp/jstl/core	c
SQL	http://java.sun.com/jsp/jstl/sql	sql
Internationalization/	http://java.sun.com/jsp/jstl/fmt	fmt
XML	http://java.sun.com/jsp/jstl/xml	x

- **Request Time (RT) version**
  - Dynamic attribute values of JSTL tags are specified using JSP expression (i.e. `<%= expression %>`)
  - The RT based JSTL tag libraries along with URIs and preferred prefixes are given below in tabular format

Library	URI	Prefix
Core	http://java.sun.com/jsp/jstl/core_rt	c_rt
SQL	http://java.sun.com/jsp/jstl/sql_rt	sql_rt
Internationalization/	http://java.sun.com/jsp/jstl/fmt_rt	fmt_rt
XML	http://java.sun.com/jsp/jstl/xml_rt	x_rt

44.2.4 JSF Events Handling Code

A JSF application works by processing events triggered by the JSF component
These events are caused by user actions. For example, when the user clicks bu
triggers an event. You, the JSF programmer, decide what the JSF application
particular event is fired. You do this by writing event listeners.In other words, a
is event-driven.

For example, if you write a JSF code to create a button, you will write:

```
<h:commandButton value="Login"
actionListener="#{customer.loginActionListener}"
action="#{customer.login}" />
```

code imp for S.Q

44.2.6 JSF – Managed Bean-Intro

These are JavaBeans defined in the configuration file and are used to hold the data from JSF components. Managed beans represent the data model, and are passed between business logic and pages. Some other salient features are:

- Use the declarative model
- Entry point into the model and event handlers
- Can have beans with various states

Here is an example of a managed-bean element whose scope is *session*, meaning that an instance of this bean is created at the beginning of a user session.

```
<managed-bean>
      <managed-bean-name>myBean</managed-bean-name>
      <managed-bean-class>myPackage.MyBean</managed-bean-class>
      <managed-bean-scope>session</managed-bean-scope>
</managed-bean>
```