

This PDF Created by

**JPG To PDF Converter for Mac
(Unregistered Version)**

Servlets

I Introduction

① Java appy

CGI vs EJB

webapp vs Best App

② static response vs dynamic response

③ Server side Tech.

① CGI vs Servlets

② JSP vs JSF

④ Client-Server architecture

① client

② server

③ protocol

⑤ protocol

① what is protocol

② why we have to use the HTTP protocol

③ how HTTP protocol is stateless?

④ HTTP methods

① GET ② POST ③ HEAD ④ OPTIONS ⑤ PUT

⑥ TRACE ⑦ DELETE

⑧ status code

⑨ Servers

① web server vs app. server

② server vs container

③ Container types

① Standalone Container

② In process Container

③ Out-of-process Container

II. Steps to design web application

- ① web application directory structure.
- ② deployment descriptor.
- ③ servlet's design.
 - ① servlet
 - ② generic servlet
 - ③ HttpServlet
- ④ Approaches to start server.
- ⑤ Access web application.

III. Servlet lifecycle

- ① life cycle stages.
- ② single thread model.
- ③ load on startup.

IV. Form design

- ① static form design
- ② dynamic form design.

— API —

V. Servlet config & servlet context

VI. Servlet Communication

- ① Browser to servlet Comm.

① Request-Response Comm

② sending error message.

③ request redirection.

① by using hreflink.

② by setting response header

③ by using sendRedirect mechanism.

② web component communication.

- ① include & request dispatching mechanism.
- ② forward & mechanism.

③ Applet to servlet communication.

— Applet —

VII. Session tracking mechanism

- ① HTTP session tracking
- ② cookies
- ③ URL rewriting
- ④ Hidden form field

— Applet —

VIII. Filters

— Applet —

IX. wrapper classes

① Request wrappers

- ① ServletRequestWrapper
- ② HttpServletRequestWrapper

② Response wrappers

- ① ServletResponseWrapper
- ② HttpServletResponseWrapper

X Listener

① Request listeners

- ① ServletRequestListener

- ② ServletRequestAttributeListener

② Context listeners

- ① ServletContextListener

- ② ServletContextAttributeListener

③ Session Listener:

- ① HttpSessionListener
- ② HttpSessionAttributeListener
- ③ HttpSessionBindingListener
- ④ HttpSessionActivationListener

Version: Servlet 3.0V

<u>Servers</u>	<u>Ide's</u>	<u>Database</u>
① Tomcat	① Eclipse	① Oracle
② weblogic	② Myeclipse	② MySQL
③ GlassFish	③ NetBeans	
④ JBoss	④ JDeveloper opt.	

Servlets (3.0v)

Introduction

Tomcat 7.0/

- Types of Java applications
CORBA vs EJB
Web application vs distributed application.
- static response and dynamic response
CGI vs Servlets
Servlets vs JSP's
- Client-Server architecture
Client (Uniform Resource Identifier vs URL vs URN)
Protocol
= http methods
- Sender (Web Server vs Application Server)
- Steps to design your first web application:
 - i deployment description (web.xml)
 - ii servlet design.
 - iii servlet
 - a generic servlet
 - b http servlet

Servlet life cycle

- ① Lifecycle stages
- ② Thread Safe (Single thread model)
- ③ load-on startup

Form design

- ① static and dynamic form design.

→ Servlet Config vs Servlet Context

→ Servlet Communication:

- i Binder-servlet Communication (bind and read)
- ii Web Component Communication.
- iii Applet-servlet Communication.

Session Tracking mechanism:

- HttpSession
- Cookies SPH
- Hidden-form field STH

JPG to PDF Converter For Mac - Unregis

Filters

— wrapper classes — (i) request wrapper
(ii) response wrapper.

Listeners:

- (i) request listeners
- (ii) context listeners
- (iii) session listeners

DBS

oracle 11xe, mysql 5.0x, html, JavaScript
eclipselink-3v, weblogic-10.3v, JBoss-7.1v

Introduction: In general by using Java technology we are able to design the following types of

- (i) standalone applications
- (ii) enterprise applications

Standalone application: These applications are Java applications which will be designed without using client-server architecture.

There are two types of standalone applications

- (i) CUI applications
- (ii) GUI applications

CUI applications: These applications are standalone applications which will take input and provide output by using command prompt, where command prompt acting as an interface and it able to support upto characters data so that command prompt is also called as CUI.

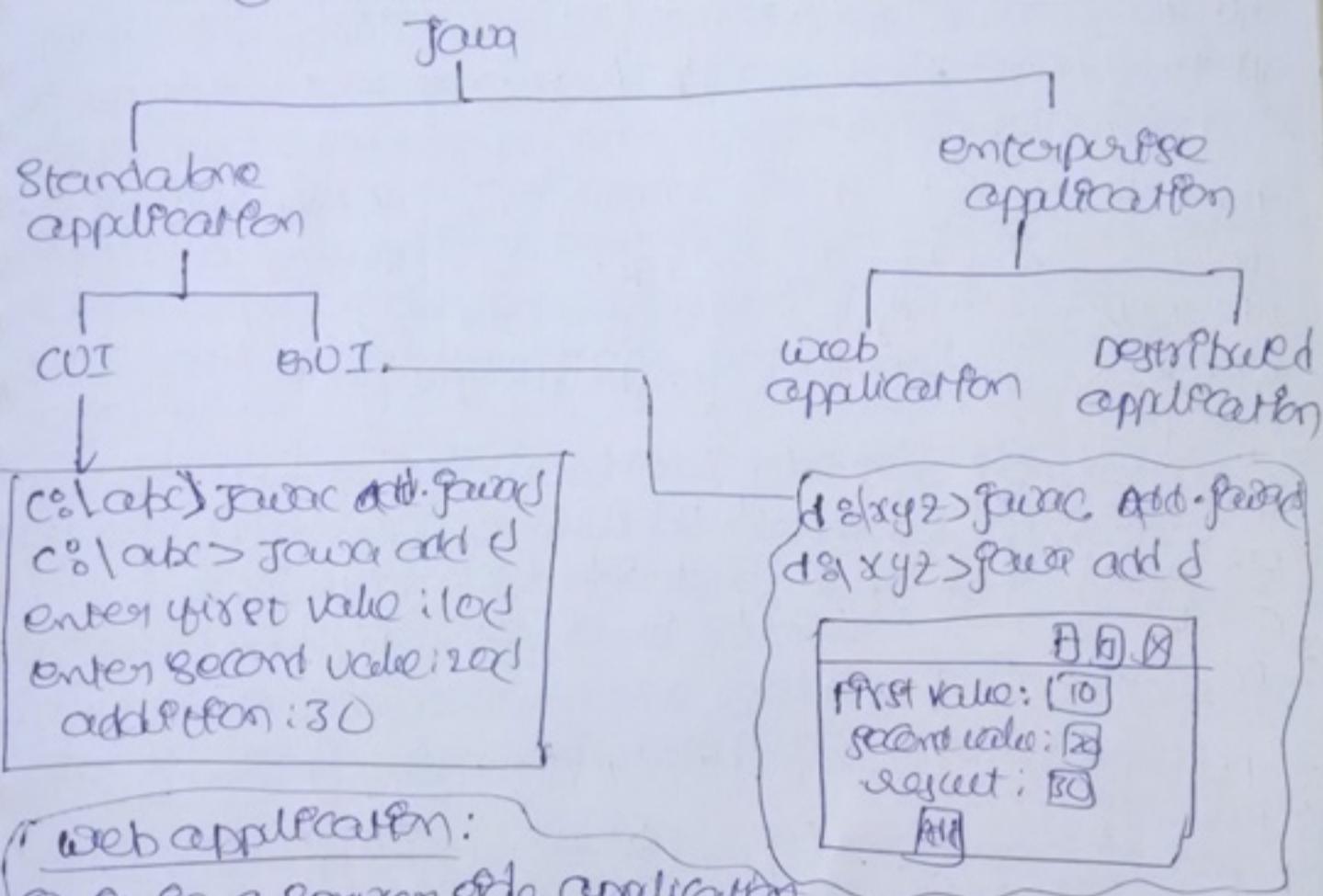
GUI applications: These applications are standalone applications which will take input and output by using a collection of GUI components, where the collection of GUI components is casting as an user interface and it able to support the graphic component so that collection of graphic component is also called GUI.

Enterprise applications (client - server)

The enterprise applications are the server applications which will be designed based on client - server architecture.

There are two types of enterprise applications

- ① web application.
- ② distributed application ✓



Web application:

- ① It is a server side application which will be designed without distributing whose application logic over multiple number of files.
- ② Web applications are server side applications which will be designed by using web technologies like CGI (Common Gateway Interface), Servlets, perl etc.
- ③ The main purpose of web application is to generate dynamic response from server machine.
- ④ In general web applications will be executed by both web servers and application servers but distributed applications will be executed not.

- ④ In case of web applications client is offered. That is web client (browser). Web applications are able to provide services only for web clients.
- ⑤ Web application is a collection of web components which will be executed by using only web container as part of servers like Servlet Container to execute Servlets, JSP Container to execute JSP's.

- Different web application & distributed application
 - A web application is a server side app which will be designed using distributed whose application logic over multiple nodes runs.
 - Distributed Application logic over multiple nodes runs.
- Distributed application is a client server application whose application logic will be distributed over multiple no. of JVM's.
- Note: In case of web applications there is no separate program for client. But in case of distributed applications we have to provide a separate java program as client.
- Web applications are the server side app which will be designed by using web technologies like CGI, Servlets, JSP's, PHP, Perl and so on..
- Distributed applications are the serverside applications, which will be designed by using the distributed technologies like RMI, EJB's, CORBA, Web services and so on..
- The main purpose of the web application is to generate dynamic response from server machine.
- The main purpose of distributed app is how to access remote services available at remote machine from local machine by providing distributed communication b/w local machine & remote machine.

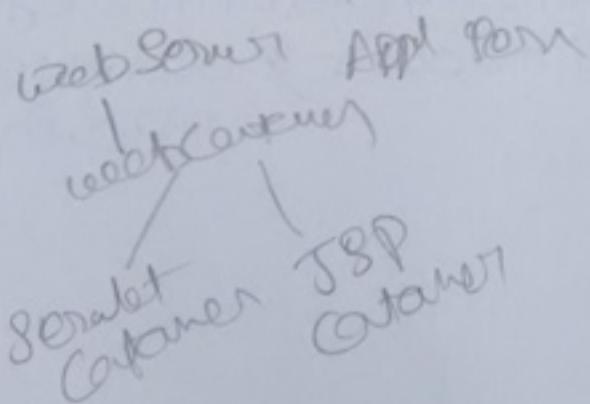
In general, web applications will be executed by both web servers and application servers but distributed applications will be executed by using only application servers.

- In case of web applications, client is fixed, that is webclient [Browser]. But in case of distributed applications client is not fixed, it may be an applet, a frame, a GUI application, it may be a serverside program upto a Servlet, a JSP page and soon--

Therefore web applications are able to provide services for only web applications but distributed applications are able to provide services for any type of client.

- web application is a collection of web components, which will be executed by using only web container, as part of the server software like Servlet container to execute Servlets, JSP Container to execute JSP's.

- distributed application is a collection of distributed components, which will be executed by using distributed containers like EJB container to execute EJB Component.

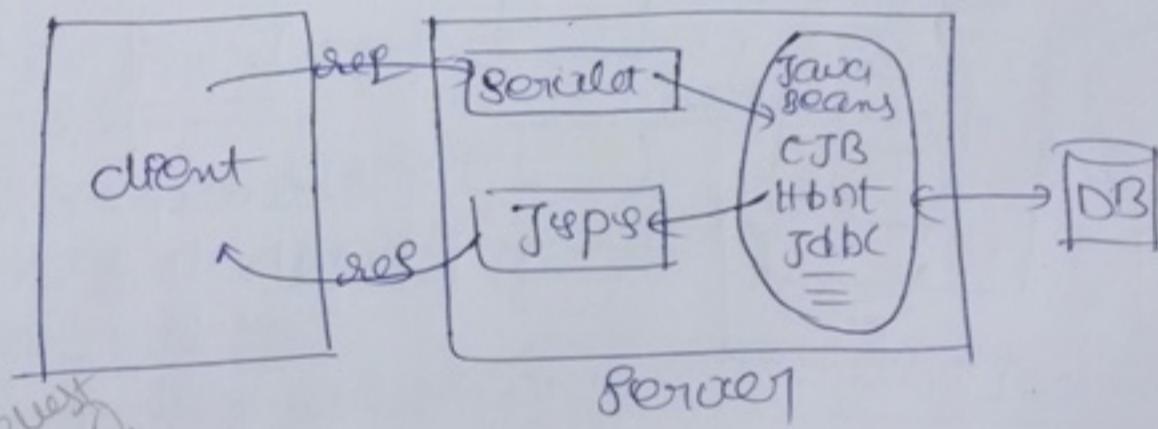


Client Server architecture.

- At the beginning stage of the computers there is client server architecture, where the main purpose of server machine is to hold up all the resources in order to provide shareability.

In the above context, if any client require any of the serverside resource then client may send a request to the server, where server will identify the requested resource and send the copy of the server side resource as response to client without performing any action at server machine. In this context whatever the response that was provided by server without executing any serverside resource then that response is called as static response.

- ~~Ans~~



HTTP
request
response
resource
server
client

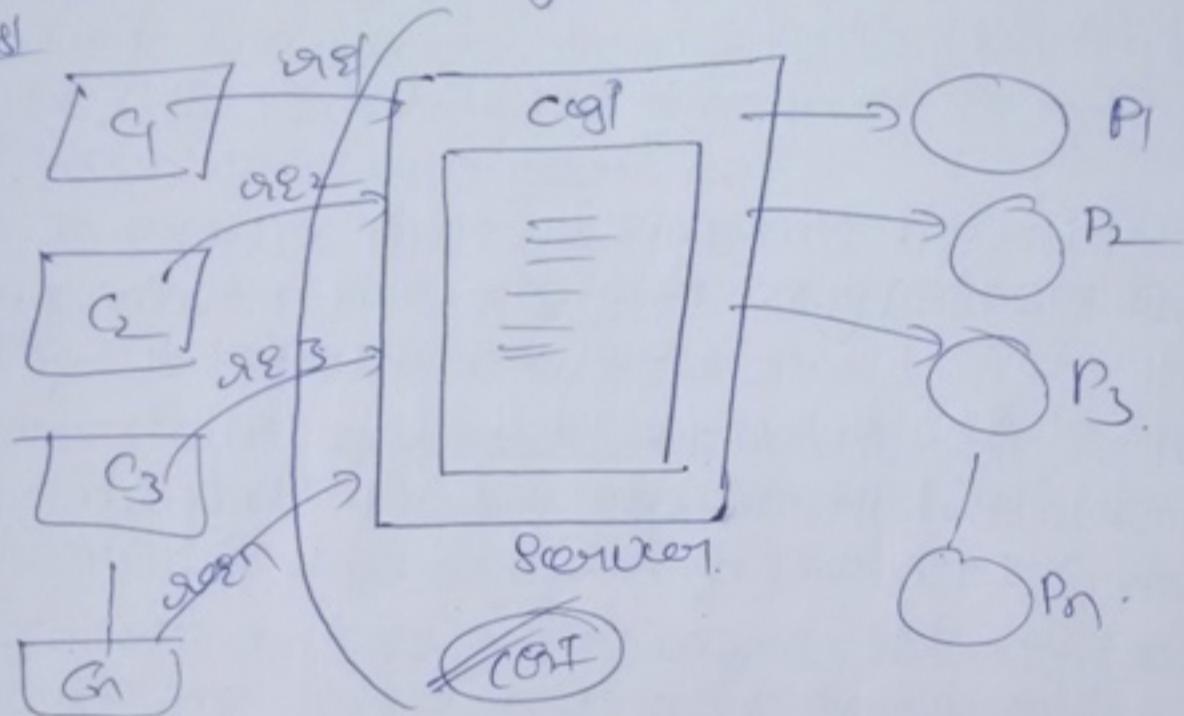
- As per the application requirement, we need dynamic response from server machine.
- To provide dynamic response, we have to provide a separate server side application called as web application.
- To design web applications, In order to generate dynamic response we require a set of server side technologies called as web technologies like CGI, Servlets, JSP's, PHP, perl.

≡

Interview question :

Q1 To generate dynamic response, we have already CGI technology then what is the requirement to go for servlets?

Ans

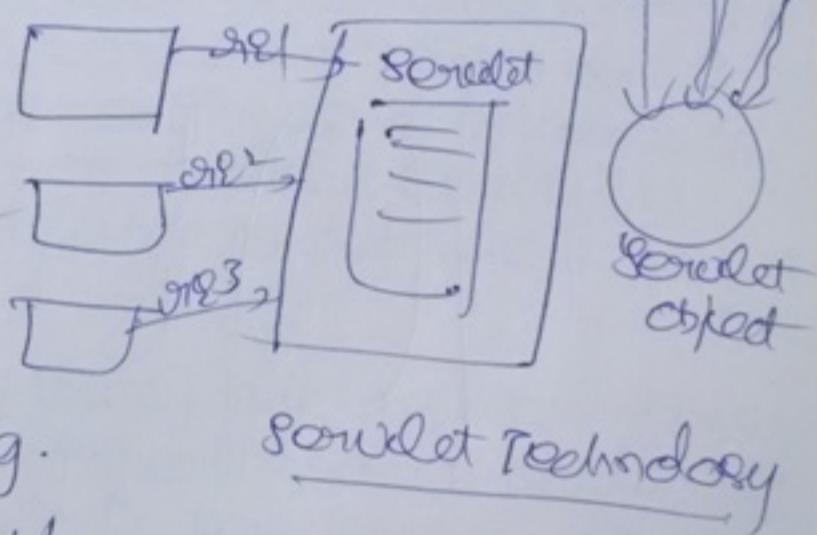


- CORBA vs. Servlet

- CORBA is a ~~server side~~ technology designed on the basis of C technology & scripting.

It is a procedural based technology. It will make CORBA a process based technology.

- If we deploy CORBA application at server side & when every new request a separate process will be created.



Date: 28th March, 2012

In EJB applications, we are unable to execute Entity Beans.

But it is possible to execute hibernate application with or without the application server. Only

— Due to the above reasons EJB's (entity beans) are less portable but hibernate is more portable in enterprise applications.

— To execute EJB's entity beans we must require application server so that entity beans are heavy weight data persistency solution.

— To execute hibernate application, it is not at all mandatory to use application server, so that hibernate is light weight data persistency solution.

— In case of EJB's entity beans, it is convention to provide one-to-one association between an entity class and the respective database table. But in case of hibernate, a single entity class is able to represent multiple no. of database tables, a single database table is able to represent many no. of entity classes.

— In case of EJB's (entity beans) all the bean classes will not participate in inheritance directly, but in case of hibernate, it is possible to provide inheritance relation b/w pop classes.

— EJB's (entity beans) are more API dependent and heavy weight data persistency solution so that entity beans are slower to provide data persistency. It will reduce the performance of the enterprise applications.

hibernate is light weight and doesn't depend so that hibernate is faster data persistency solution, it will improve the performance of enterprise applications.

Note: hibernate is faster data persistency solution but, it will take lot of time to setup the entire hibernate environment.

— what are the differences b/w hibernate and JPA?
Ans JPA is an API, a set of rules and regulations, which describe how to implement ORM rules and regulations in order to provide data persistency.

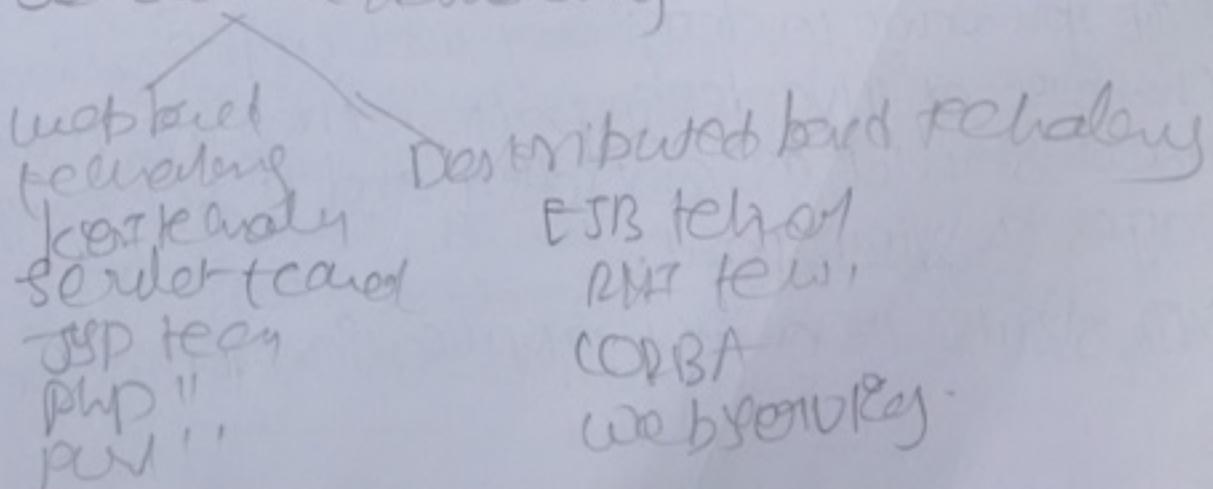
Sun micro systems has provided JPA implementation in the form of javax.persistence package. In general, in almost all ORM tools JPA rules and regulations are utilized.

hibernate is a tool, which uses implemented ORM rules and regulations as per the guideline provided by JPA.

Green
Yellow
Blue
Red
Yellow
Green
Blue

- [Section 7]
- In case of ~~cost~~ technology If we increase no. of ~~HTTP~~ requests to some cost application than that no. of processes will be created at server side.
 - Basically process is a heavy weight to handle multiple no. of processes at server machine, Server machine may get burden it will cause reduce in performance of the serverside application.
In the above context, to improve the performance of serverside application we have to use an alternative to ~~cost~~ technology.
 - ~~Servlet~~ is a server side technology designed on the basis of Java technology. Basically Java technology like ~~Java~~ based technology, It will make ~~Servlet~~ as serverside thread based technology. If we deploy any ~~Servlet~~ application at server machine then for every new request container will prepare a separate thread instead of creating process.
 - Even though if we increase number of requests from client, Container will create no. of threads only.
 - In this context, server machine may not have any burden while processing multiple no. of requests; It will improve the performance of serverside application.

Server side technology



Q: What are the differences b/w Servlets and JSP's

To design web applications if we use servlets then we should require very good Java knowledge.

To design the same web application if we use JSP technology then it is not required to have Java knowledge, it is sufficient to have presentation skills or web design.

Note: The main intention of JSP technology is to reduce Java code as much as possible in web applications.

① In web applications, Servlets are very good at the time of pick up the request and process the request [controlling the web application] But JSP's are very good at the time of generating dynamic response to the client with very good look and feel. [But JSP's are not very good at pick up the request and process the request]

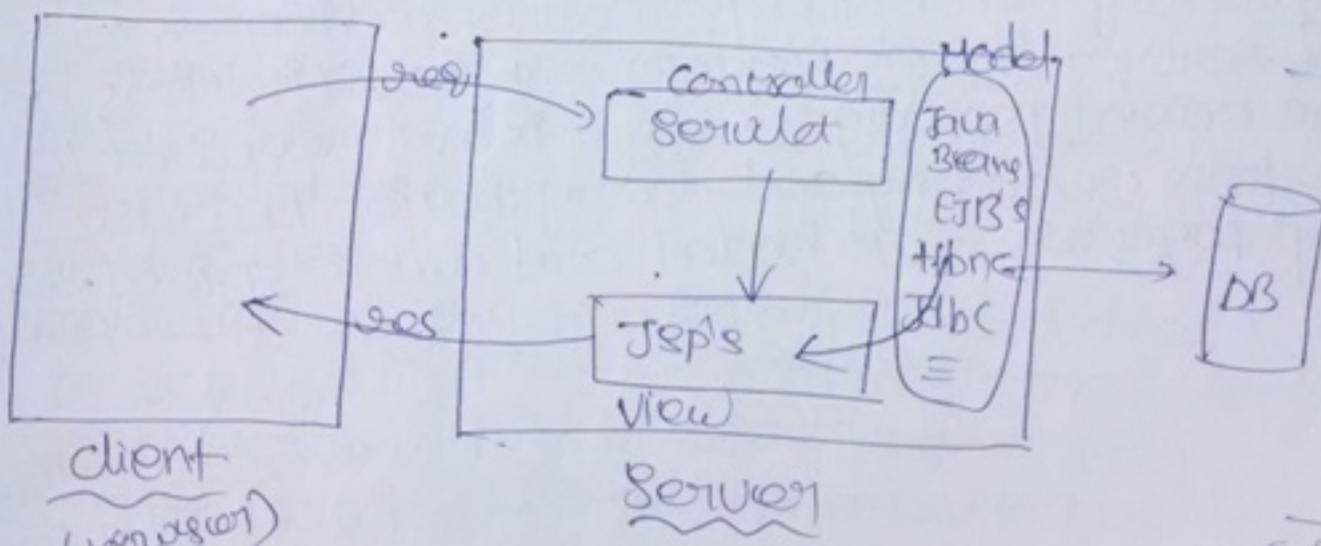
② In case of the Servlets we are unable to separate both presentation logic and business logic but in case of JSP's, we are able to separate both presentation logic and business logic because in JSP pages we are using HTML tags to prepare presentation logic and we will use JSP tag library to prepare business logic.

④ If you want to design any web application on the basis of MVC design pattern then we have to use a Servlet as Controller and a set of JSP pages as View part.

Q: Struts is a web application framework

designed on the basis of MVC architecture, where it will use ActionServlet, a type of servlet as controller and a set of JSP pages as view part.

Q2: JSF (Java Server Faces) is a web application framework, designed on the basis of MVC design pattern, it was provided facesServlet, a type of servlet as controller and a set of JSP pages as view part.



— Here, why we use servlet only ~~as~~ as a controller and JSP as a view part?

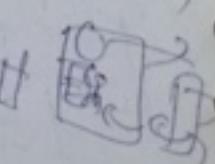
A1 Why because, servlet is very good at pick up the request and process the request. That's why servlet can be acts as a role of controller. But it shows poor performance when generating a dynamic response to the client after processing the request. That's why servlet cannot be act as a role of view. So JSP came into the picture to take this responsibility. why because JSP's are good at generating dynamic response to the client. That's why JSP act as a role of view part.

Why
use
we
re
for
der
as
our
obj
JSP
Gay
view
In
cu
the
app
obj
in
the
obj

→ This is the best combination suitable for any ^{But}
framework. If you observe the above discussed ^{the}
frameworks, In all the cases Servlets are acting as a
Controller and JSP's are acting as a view part. Of course
the model part will be taken care by other technologies
JDBC, Hibernate or Java Beans or may be Entity Beans

In web applications, If we perform any modification ^{on}
on the existed Servlets then we have to perform ^{copy}
explicitly decompilation and recompiling and ^{on}
reloading on to the server. If we perform any modifications on
the existed JSP Pages then it is not required to
perform reloading and decompilation because
JSP pages are auto loaded and auto compiled ^{copy}

↳ what is the internal
process for "how JSP's
are auto compiled and
auto loaded".



client - server architecture:-

Date :- 2nd Apr, 2012

1. To design any enterprise application, we have to define the degree of height for the enterprise application on the basis of enterprise application level.

- To define height of the enterprise application, we have to use system architectures.

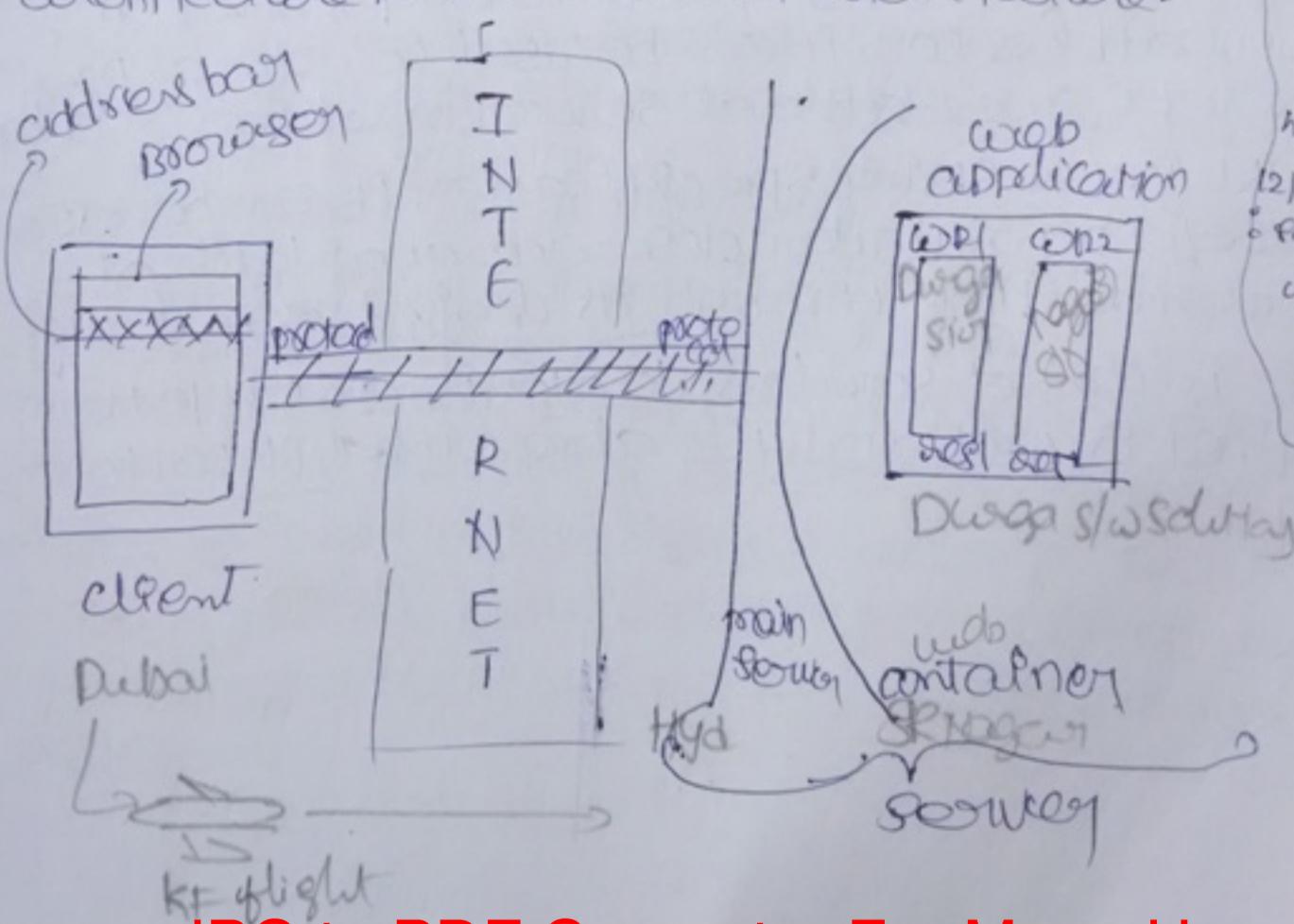
one-tier architecture

2-tier architecture

n-tier architecture

- From the above tiered architectures, one-tier architecture is highly recommended for standalone applications, not for enterprise application.

To design any enterprise application that is web application we need multi-tier architecture. The best example for 2-tier architecture is client-server architecture.



In the above client-server there are 3 main components

1. Client
2. Protocol
3. Server

Client: The main job of the client in client-server architecture is to send request to the server and to get responses from the server.

- At client machine to send request and to get response we have to use a tool called as browser where browser is acting as real client.
- To access a particular server side resource from client browser we have to use a string at client address bar called as URI.

There are two types of URI's

1. URL

2. ORN.

Ques:

- What are the differences b/w URI, URL, and ORN?
- Ans: URI is the String specification provided at client address bar, it can be used to refer a particular resource available at server machine.
- URL is the String specification, it can be used to refer a particular resource available at server machine through its designated network location.

Note: In case of Servlets, location is an URL pattern defined in web.xml file along with URL-pattern tag

What is this
location?
not in clear way

URN: URN is the string specification provided at client address bar, it can be used to refer a particular resource available at server machine through which it is logical name.

Note: In case of Servlets, logical name is a name specified along with <Servlet-name> tag in web.xml file.

To refer a particular server-side resource, we have to include protocol name, server ip address, server port number, application context, resource name in URL.

Protocol name: //Server : Server / application
ip address port no / context
resource name [? query string]

Ex: http://121.122.92.98:8080/JavaWebApp/

login?username=durga&password=123

If the server is available at the same client machine then it is possible to specify "local host" or 127.0.0.1 in place of server ip address.

Ques: what is the difference between ipaddress and port number?
Ip address is unique identification to each and every machine over the network which will be provided by the network manager at the time of network configuration.

port no is a unique identification to each & every process being executed at a single machine which will be provided by one local operating system.

URN: URN is the string specification provided at client address bar, it can be used to refer a particular resource available at server machine through using its logical name.

Note: In case of servlets, logical name is a name specified along with <Servlet-name> tag in web.xml file.

To refer a particular server side resource, we have to include protocol name, server ip address, server port number, application context, resource name in URL.

Protocol name: //Server : Server / application
ip address port no / context
resource name [?Query String]

Ex: http://121.122.92.98:8080/loginapp/
login ?username=durga&password=123

If the server is available at the same client machine then it is possible to specify "local host" or 127.0.0.1 in place of server ip address.

Ques: what is the difference between Ip address and port number?

Ip address is an unique identification to each and every machine over the network which will be provided by the network manager at the time of network configuration.

port no is an unique identification to each & every process being executed at a single machine, which will be provided by one local operating system.

What is query string and what is the purpose of query string in web applications?

All query string is a collection of name, values pairs (request parameters) appended to the URL to provide input data to a particular server side resource in order to perform the respective server side action.

Real scenario

Suppose for example, a software engineer located at Dubai want to meet Durga sir at S.R.Nagar. First s/w engineer Vijay will travel on a flight to reach hyderabad from Dubai. From airport, he will catch a car travelled to SR Nagar. And at SR Nagar in hyd there are many other places located other than SR Nagar. At SR Nagar, he will find durga s/w solutions. At durga s/w solution, we may find different number of faculties are available. But Vijay want to meet only durga sir. Finally Vijay met durga sir.

What the point get from the above discussion is that from client machine we will send a request. Assume this client is Vijay. At client side, before giving request protocol will be there. This protocol will establish a connection over the network, by taking the Server IP address and port no. from ORL. In the above context the flight KF acts as a protocol. Now this request will be passed through this network.

Now, once think on the Internet, different computers may exists. Then how this request can identify its server?

Every server on the Internet can be identified

through its IP address. This is 121.120.92.98. So request by using this it will reach server. Now on server, different servers may exist. There may be Tomcat or Weblogic or Glassfish. To which of these servers the client request needs to pass?

For this purpose each server will have a port number given by operating system. For tomcat it is by default 8080.

But before installing Tomcat server check whether there is Oracle ~~was~~ installed on your machine or not. Why because Oracle also by default uses this 8080 port no. (remember Oracle is also one type of Server). To resolve this, at the time of installing Tomcat, give some any other no ~~8081~~ Port no. Specifically why I need to change port no for Tomcat only? ^{on} Why I have to compromise? Because there is no alternative. We don't have any right to change the port no. Oracle. This is the only automated way.

OK coming to our discussion, by using this port no. request may identify which of the server. Now at server different types of web applications were available. For example vipay1, vipay2, vipay will be available.

How can one request identify its web application? By using the name specified at request in address bar ^{web application}

So after entering into web application for example vipay1 web application. There are different web resources may be available. For example ^{it} it may be a Servlet, JSP, Java service file - etc.

by using the web resource name specified in address bar, request search fits web resource.

So in this way the request will be reached to the server. On receiving request server process that request and executing that demanded web resource and finally it will send a dynamic response to the client in the form of response. The protocol at serverside take this as form. Form and send it through the same network connection back to the client.

In the example diagram, "After entering John, pass and click submit then "URlauthenticaed" will be displayed on the screen".

Now, which type of protocol is suitable? Tell me first what is the protocol?

Take a bicycle. It having its own design. A man can apply force on pedals so that cycle will be moved. [There will be no stops for any bicycles]. This is one type of rules and regulations followed by bicycle.

Now, take a bus. The main purpose of bus is to carry many passengers. There will be a engine for bus. In order to move and there will be bus stops for alighting into and out of buses. What I am saying here is bus will also having its own rules and regulations.

Like that flights will be there, flying, ships will be there. It is the desire of the passenger to take which type of travel he want. Suppose he want to go to softbank from Andhra then he will choose ship. Suppose he want to go from airport to softbank, he will choose a cycle.

That depends on the implementation

Date: 3rd April 2012

Protocol: protocol is a set of rules and regulations, which can be used to carry the data from one machine to another machine over the network.

Ex: TCP/IP, FTP, HTTP, UDP, SMTP and so on.

- The main job of the protocol in client-server applications is to carry request data from client to server and to carry response data from server to client.

- In general in web applications, we will use http protocol.

Int. que: Why?

In general in any web application, we require a protocol, it should be connectionless protocol, a stateless protocol and a compatible protocol to carry hypertext data between client and server. Where connectionless protocol is a protocol, which should not require any physical connection, It should require a logical connection.

Where stateless protocol is a protocol, It should not remember previous request data at the time of processing data request.

In general, in web applications, we will carry the request data from client to server in the form of hypertext data and we will carry the response data from server to client in the form of hypertext data.

In this context, to carry the hypertext data between client and server, we should .

require a compatible protocol.

Conclusion:

- To satisfy all the above three characteristics we will use http protocol.

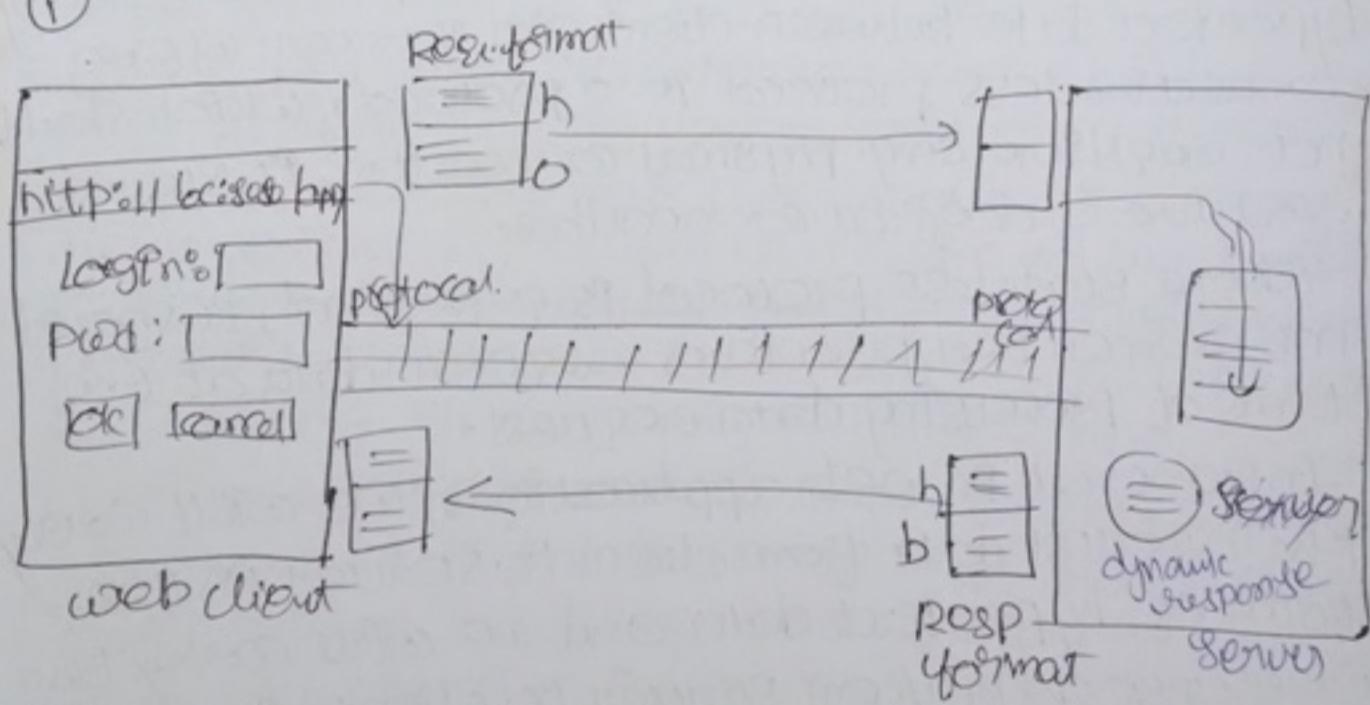
Protocol:

How http protocol is able to manage its stateless nature?

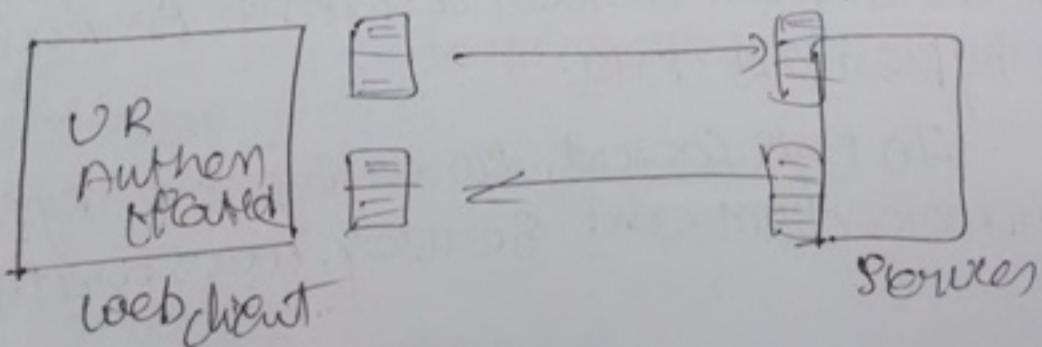
- In case of http protocol, when we send a request from client to server, protocol will pickup the request and perform the following actions.

① Protocol will establish a virtual socket connection b/w client and Server as per server IP address and the port number which will be provided in URL

①



②



Protocol will prepare a request format having header part and body part, where header part will manage the request headers that is meta data about client. Body part will manage request parameters which are provided by the user at client browser.

③ After getting request format protocol will carry that request format to server.

Note: When request is addressed to server, server is able to identify requested resource, execute it and generate dynamic response to the client. When protocol receives the response from server, then protocol will perform the following activity.

- 1. Protocol will prepare a response format having header part and body part, where header part will manage the response headers, meta data about the response and the body part will manage the actual dynamic response from the server.
- After getting response format protocol will carry that response format to client.
- When client receives the response from protocol, client is able to display the response and after that browser, with this protocol will terminate the virtual socket connection.
- In the above context, protocol is able to manage the present request data how long the connection is existed; once if the connection is terminated then protocol will remove the present request data from its memory.

- In the above situation at the time of processing later requests protocol is unable to provide previous request data.
Therefore http protocol is stateless protocol.
- In general in web applications, we will use http protocol that is a stateless protocol, it unable to provide previous request data at the time of processing later requests, but as per the application requirements, we need previous request data at the time of processing later requests.
- To achieve this requirement we have to use a set of mechanisms explicitly provided by servlet-api called as session tracking mechanism

- In web applications, http protocol will provide flexibility for the developers to specify the different types of requests at client browser.
- The above flexibility is possible for http protocol due to the availability of http methods [implied completely in http protocol]
- http protocol has provided the following 7 number of http methods.

http 1.0 version has introduced the http methods like get, post and head.

http 1.1 version has introduced the http methods like OPTIONS, PUT, TRACE, DELETE

http 1.0
get
post
head

int que.

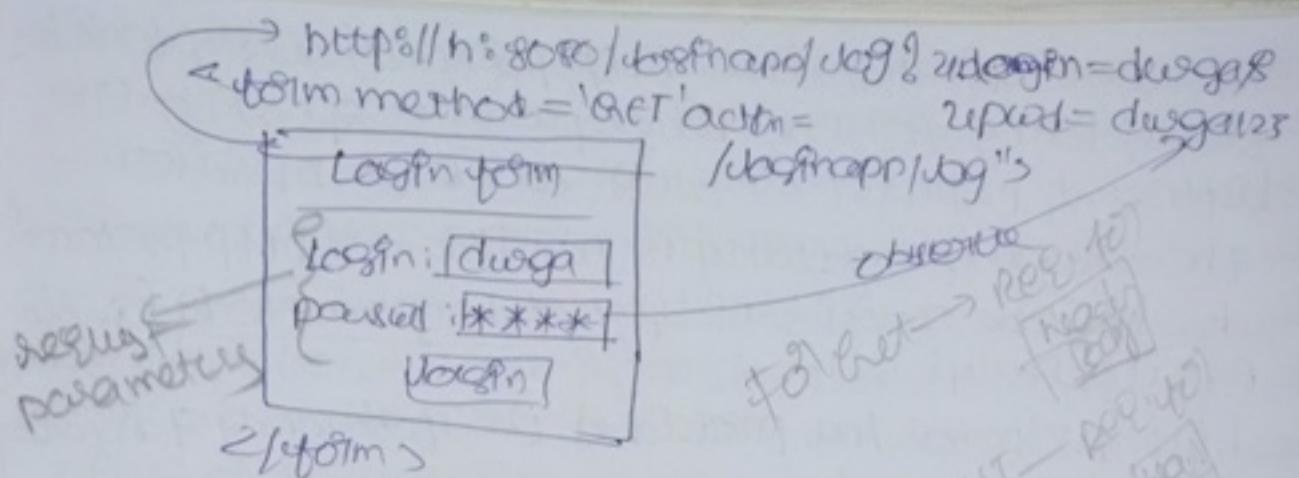
Date: 29th Apr, 2012

1. what are the differences b/w get request and post request.
- ① 'get' request is default request type whereas 'post' request is not default request type.

→ `http://localhost:8080/registernapp/login?uname=abc&age=20`

`<form method='get' action='/registernapp/login'>`

Reg form	
Name:	1980
Age:	20
<input type="checkbox"/> Registered	
</form>	

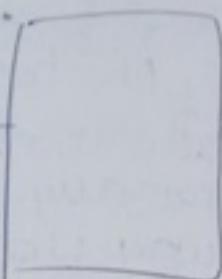
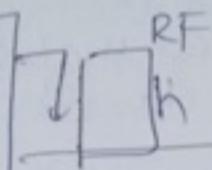


- GET request should not have body part in the request format but POST request should have body part in the request format.

- In case of 'GET' request if we provide request parameters along with 'GET' request than 'GET' request type will pickup all the request parameters and provide as query string along with url on address bar.
- In case of the 'POST' request, if we provide any secure data like password data, pin numbers and so on along with request then 'GET' request will display that secure data on client address bar as query string. Therefore 'GET' request will provide less security for the client data.
- If we provide request parameters along with 'POST' request then 'POST' request will not be displayed the request parameters on client address bar as query string. Therefore 'POST' request will provide very good security for the client data.

got

Request Format	
Name:	<input type="text"/>
Age:	<input type="text"/>
<input type="button" value="Request"/>	



Server

- In case of 'GET' request, request format should have only header part, If we provide request parameters along with 'GET' request then 'GET' request will carry that request parameters through request format header part from client to Server.

In general request format header part will have some memory demarcation so that it able to carry less data.

In case of 'POST' request, request format will have body part, If we provide request parameters then 'POST' request will carry that request parameters through request format body part, here the request format body part will not have memory demarcation so that it able to carry huge data from client to server.

— Think once what is the meaning of get "to get some data". yes this is meaning of get indirectly. 'get' will be used to 'get some data from Server'.

for example: suppose there are 10 books available at Server. out of those 10 books if need only one book to download into my system. So first I need to type that book name in the Search button available on the user interface. Now

Server displays the list of books in the form of hyperlinks one by one. The book you needed is at first position. Now if you click on that hyperlink, the book will be downloaded into your computer.

Internally that user interface was designed in such a way that 'get' request is written as a method to that hyperlink attachment. whenever you click on that link means "I requested server to give me that file", by giving you through get request.

In general in web application, we will utilize 'get' request to download the data from server. But post request can be utilized to upload the data from client to server.

Ques.

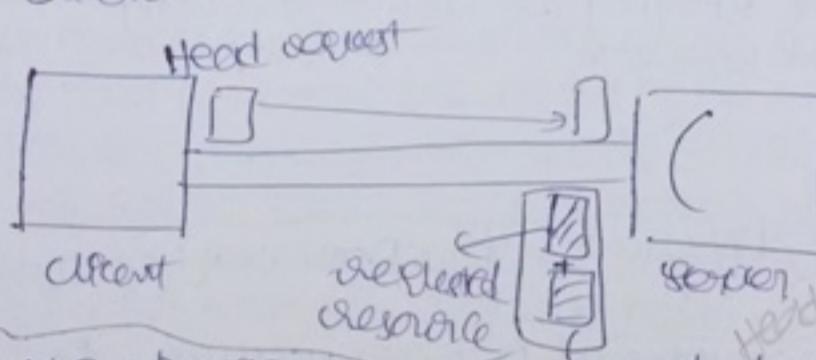
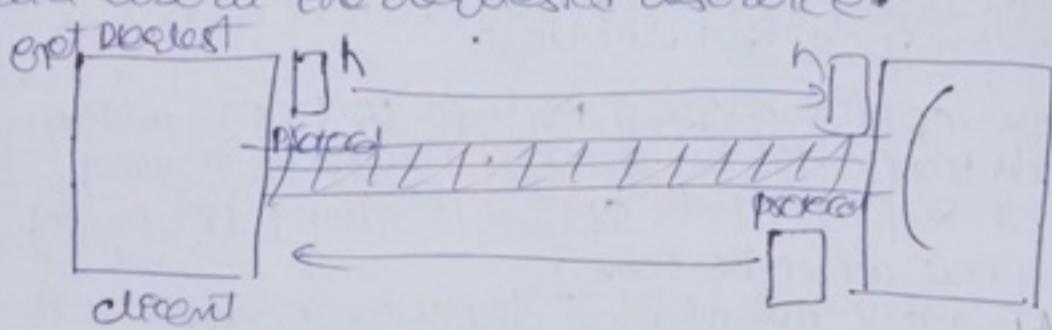
① What is the difference between GET request and HEAD request?

Ans. Here the 'get' request is used as a request to download the book. In 'get' method we will only specify a small amount of data, that is sufficient to request the server for downloading. That much of data is sufficient to put in the header part of the GET request.

But 'post' request is able to carry huge amount of data from client to server, why because it is taking body as a extra part apart from header part. So in any case if I want to upload any 100mb data to the server then ~~we need~~ to send the request to server by using post request.

If we send 'GET' request for a particular resource available at server machine then server will send only the requested resource as a response to client.

If we send 'HEAD' request for a particular resource available at server machine then server will send the requested resource as well as the meta data about the requested resource.



All the above methods are developed by protocol http only, not by the server. Server was developed on the basis of API given by Sun Micro Systems. And these methods are not implemented on the server. But server supports (knows) about these methods. observe the diff b/w supporting &

So, in most of all the servers, ~~they~~ ~~supporting~~ (Amplam) servers will support only two methods ~~they~~ (ontain) ~~supporting~~ GET and POST requests. But it is not mandatory to support remaining requests by all the servers.

In case of Tomcat server, it was supported only two requests one GET and another one POST.

OPTIONS request: The main purpose of OPTIONS request type is to get which http methods supported by present server, which we used for apple cotton.

If we use any other request other than get and post request then Tomcat server will treat that request as get request only. For example if we send post request to Tomcat server then it treated it as get request.

Here, option request is used to know which methods supported by server. In case if server does not support this options request [Not known] then what will be case?

Ans In that case simply we didn't get any output from server.

~ what is the difference b/w POST request and PUT request?

Ans Both POST request and PUT request can be used to upload the data on the server. To upload the data on server if we use POST request then it is not required to specify any server side location along with request.

To upload the data on server from client if we use PUT request then it is mandatory to specify a particular server side location along with request.

Trace request

The main purpose of Trace request type is to get the working status of a particular resource available at server machine.

Suppose I want to check whether a particular web resource is working or not. Then while designing of HTML page, I will specify this

Trace request. When server receives the request format, then it identifies the Trace request. Here server needs to inform to the client that particular web resource is available and healthy, simply server sends that web resource to client in response.

Here observe I was not send a request to get that particular webresource from server. But I only need to know the healthiness of that particular web resource. It's similar to checking the info system. Whenever I pronounce hello before the mouth then if the speaker is working properly then I will hear the sound hello from the indicating that speakers working properly speakers.

Delete request

This delete request can be used to delete a particular resource available at server machine.

Get Post Head options but Trace Delete

Note: http protocol has proposed the above seven number of http methods, where supporting or not supporting all the http methods by the server is completely depending on server implementation.

In general almost all the servers may support get and post request minimum. All most all the servers will not support put and delete requests.

- Tomcat server has provided implementation for get and post requests. If we specify any other request at client then tomcat server will treat that request types as the default request type. That is get request type.

Q1 Why delete req was not supported by server?

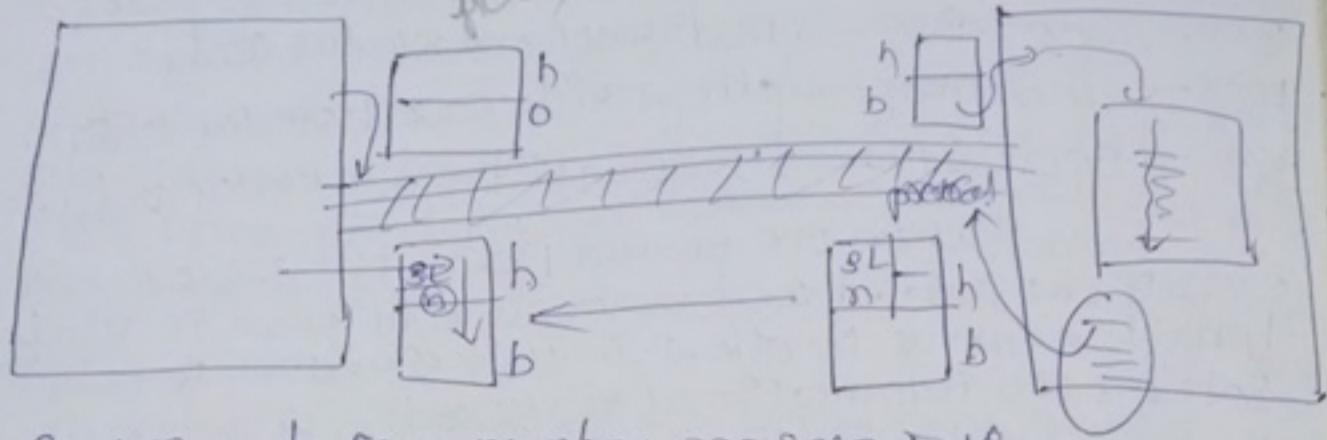
All suppose take the gmail website constructed from different servlets and JSP [Web resources]. Now, a client (a user) can't have any capability to delete a resource at server? No. why because if gmail give that option, then anyone control is in the hands of client. Suppose if any client want to destroy a particular website by deleting web resources then he will use delete req type which was not recommended in any case.

- Similarly if we want to use put req, we need to know the server location. If u know the server location, u attain the capability to destroy the applications available at server.

That's why many of the servers were not support these put and DELETE request.

Status Codes (Status line field)

[Date: 5th April, 2012]



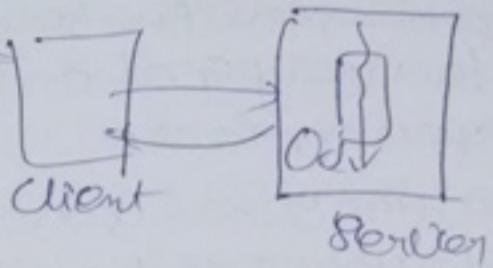
- Status code is a number representation provided by http protocol, it will describe the status of the request processing to the client.
When we generate some response from Server machine protocol will prepare response format with header part and body part, where header part there is a field like status line field to hold the respective status code.
- At the time of generating dynamic response Server will set the respective ~~out~~ ^{out} status code to the status line field, where ~~out~~ ^{out} protocol describes response format to client browser then client browser will pickup the status line field value and prepare itself to hold the respective response from response formats body part.
- http 1.1 version protocol has provided status codes. There are five types of status codes provided by http protocol to describe the status of the request processing.

- 1XX - 100 - 199 — informational status code.
2XX - 200 - 299 — success related status codes.
3XX - 300 - 399 → redirection status codes.
4XX - 400 - 499 — client side error status codes
5XX - 500 - 599 → server side error status codes

when the browser process the request and execute the related web resource and then it sends dynamic response to client. In this situation, server setups the informational status code.

- when the execution success and inorder to inform to the client that the execution was success then server setup the success related status code.
- suppose a client sends a request to a part web application, for any case if that web application not in a state to process that request then it will redirect that request to the another web application. now in this context, server setup the redirection status code inorder to informate to the client that "I got the response from another web application".
- server will set 4XX status code when it does not get the correct request format from the client. That means here the problem is at client side.
- server will setup 5XX status code into the status line of the response format when the server is not in a state to process that request. @ when server is shut down. @ If there will be an error occurred at serverside.

Server — The main job of the server in client-server applications is to pickup the request and identify the selected resource and execute the request resource and generate dynamic response to the client.



Note:

Servlet is a normal Java program, a .class file at server machine. If we create servlet some output will be generated, here server will send the generated output as dynamic response to client.

If we want to make any machine as Server machine then we have to install a piece of software called as Server Software.

Example Apache-Tomcat,

BIA-WebLogic

IBH-WebSphere

Sun Microsystems — SunOne, J2EE, Oracle8i

Macromedia — JRUN

Oracle — OracleWeb

There are two types of servers.

1. Web Servers

2. Application Servers

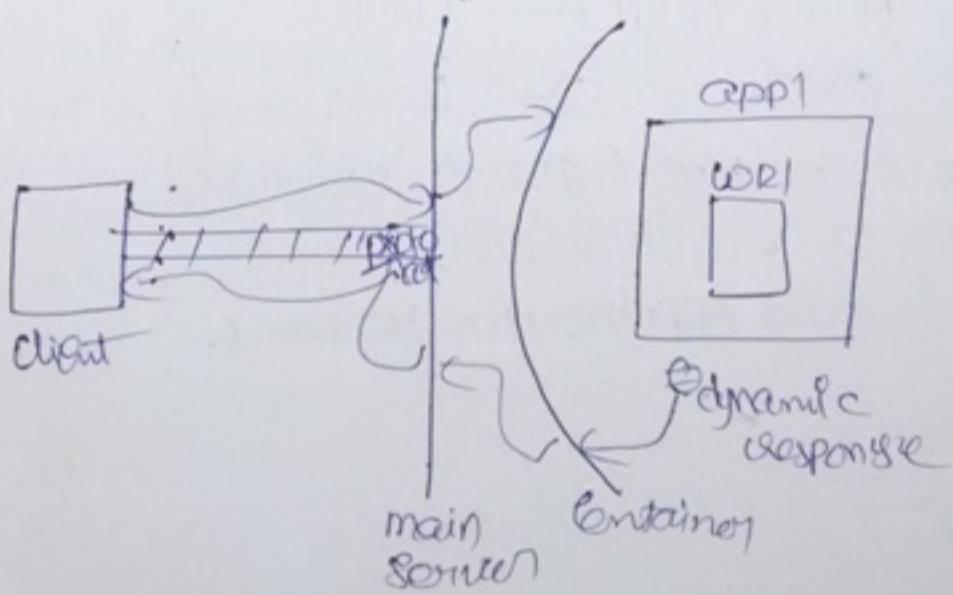
Entire

Q what is the diff b/w web servers and application servers?
Web servers are the servers, it wdy provide a very good environment to execute web applications.

Application servers are the servers, which wdy provide very good environment to execute any type of J2EE applications including web applications, distributed applications and so on.

- In general web servers will not provide middle ware services like JNDI, Datasource, Transaction support, security and so on--
- In general, application servers will provide all the middle ware services by default.
- When we install a server software on Server machine automatically the server software wdy be available in the form of the following two modules

1. main server
- & Container.



What is the diff b/w ^{main} Server and Container?

When we send a request from client to server, main server will process the request and check whether the request data is in well formed format or not. If the request data is in well formed format then main server will bypass request to container — upon receiving the request from main server, container will identify the requested resource, execute and generate dynamic response to main server; where main server will bypass that response to client, ~~to~~ protocol.

In general through
Containers will be classified into the following two ways

① As per the technology which we used to prepare server side components, we have some containers like

- ① Servlet-Container to execute Servlets,
- ② JSP Container to execute JSPs
- ③ EJB Container to execute EJBs
- ④ EAI Container to execute corba scripts

On the basis of the containers physical existing there are 3 types of Container

1. Standalone Container

↳ It is an

integration of main server and container as a single module.

Here, a piece of program can be act as both main server and container

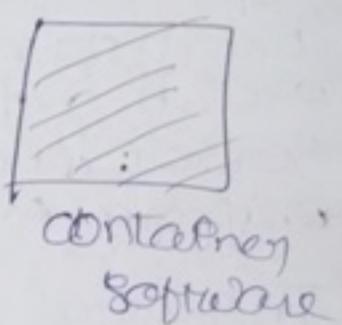
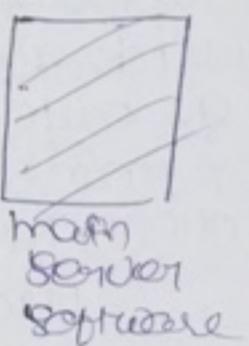
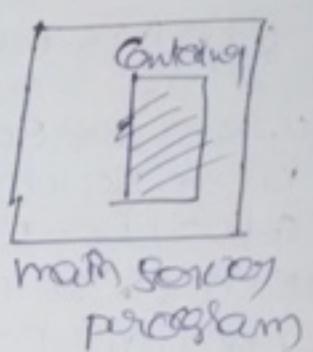


2. In-process Container:

It is a container existed inside the main server.

3. out-of process Container:

It is a container existed in outside of the main server.



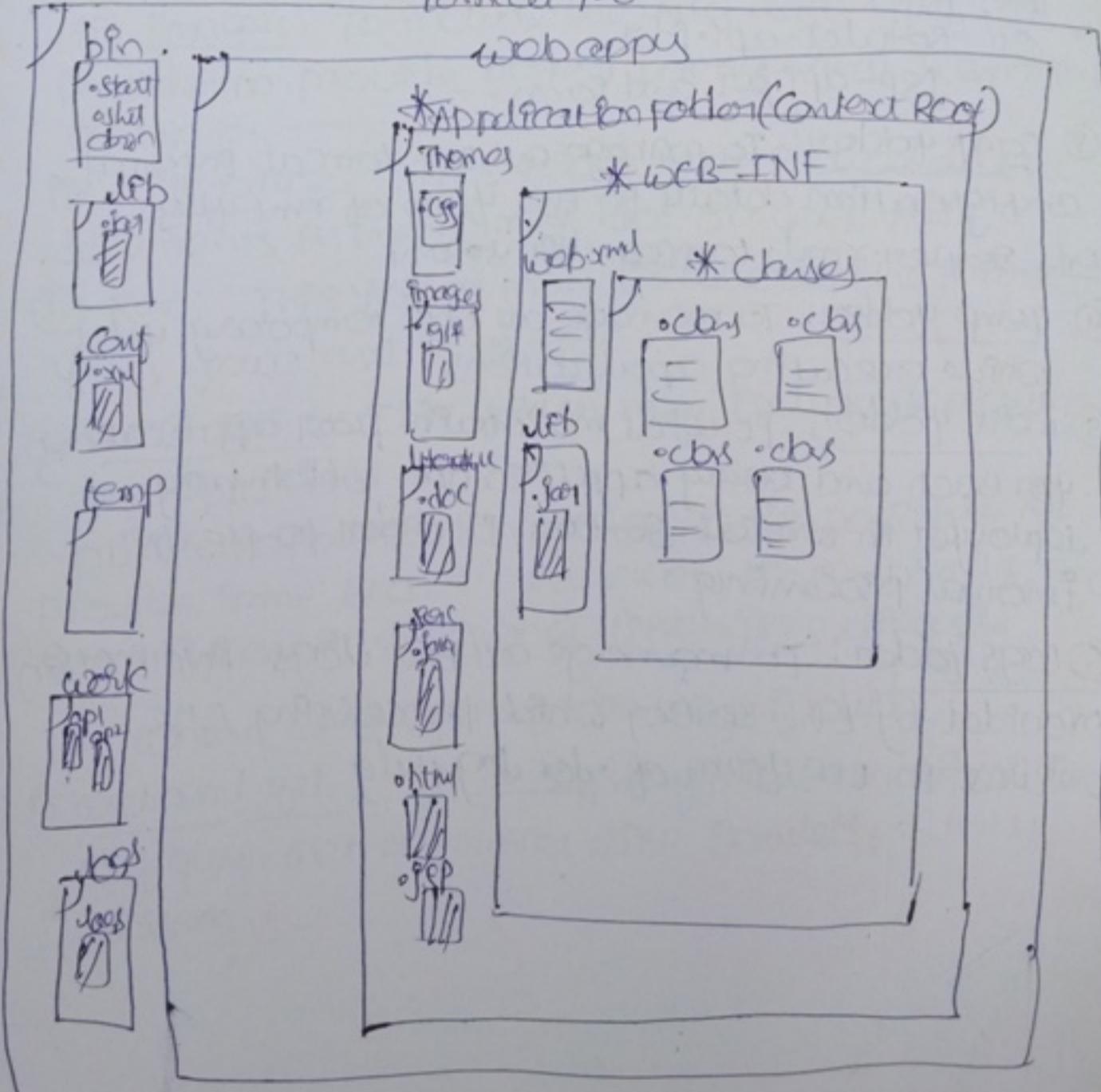
Digital

Steps to design first web application:

- ① prepare web application directory structure at server.
- ② prepare deployment descriptor (web.xml) file.
- ③ prepare web resources (Presenters, JSP's, HTML's and so on) as per the requirement.
- ④ Start the Server
- ⑤ Access the application from client ✓

web application directory structure:

Tomcat 7.0



JPG to PDF Converter For Mac - Unregis

To design any web application at Server side, we have to prepare the following directory structure at Server machine.

- When we install tomcat server on one Server machine automatically tomcat server will provide a base directory tomcat 7.0.
- Tomcat 7.0 include
 - ① bin folder: To manage server start up, shutdown batch files and .exe files.
 - ② lib folder: To manage all the API implemented jar files like
 - .jar servlet-api.jar
 - JSP-api.jar and soon -
 - ③ conf folder: To manage all the tomcat informal configuration details in the form of xml files.
 - .xml server.xml, tomcat-users.xml
 - ④ temp folder: To manage all the temporary files while executing applications.
 - ⑤ work folder: It will maintain peer application for each and every application which may deployed in tomcat server. In order to perform internal processing.
 - ⑥ logs folder: To manage all the logs information provided by the server while processing the request in the form of .dot log file.

⑦ webapps: This folder will be accessed by the server to manage all the custom web applications prepared by the developers.

— If we want to prepare any of our web applications on a tomcat server then we have to prepare a separate folder inside webapps folder called as application folder or Context root.

— Application folder may include

i) Themes: To include the files like .css --

ii) Images: To include the files like .jpg, .jif -- in order to provide logos, background scenario by and so on --

iii) Literature: This folder may include all the documents in the form of .doc & .docx file.

⑧ src: This folder may include all the source files .java and so on --

⑨ WEB-INF: This folder may include deployment descriptor and so on --

— Under application folder it is possible to provide some static resources like .html's and some dynamic resources like .jsp's directly where WEB-INF folder may include

i) web.xml file: It is deployment descriptor to configure web resources like Servlets, filters, listeners --

- ⑪ WEB folder:— This folder will include all the explicit files which are required in our web resources development like ojdbc6.jar file to interact with database, standard.jar & jstl.jar to get JSP's standard tag library support and
- ⑫ classes folder: This folder will include soon servlets.class files, writers.class files and soon

- The above web application directory structure was divided into the following two parts.
 - 1. public area ① client area
 - 2. private area ② server area.
- The area which come under application folder and outside of WEB-INF folder is called as public area.
- If we deploy any resource under public area then we are able to access that resource by using its name directly from client.
- The area which come under WEB-INF folder is called as private area. If we deploy any resource under private area, then we are unable to access that resource by using its name directly from client.

Note: If we deploy in general we will provide all the html pages under application folder. That is public area, where to access that html page we are able to use directly its name for URL at client browser.

`http://localhost:8080/loginapp/logininfo.html`

- In general we will provide all the Servlets.java file under classes folder. That is private area, where to access that respective Servlet, we have to define an URL pattern for web.xml file, through this URL pattern only we are able to access the respective Servlets from client

`http://localhost:8080/loginapp/` browser
where login is an URL login

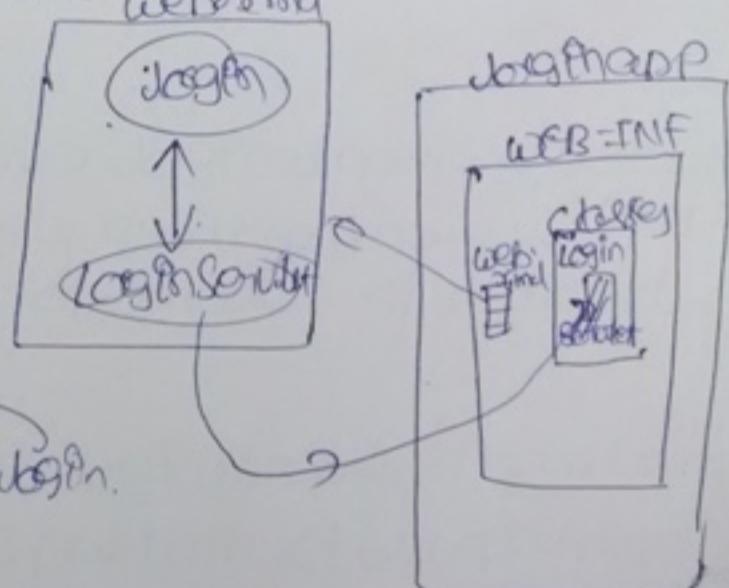
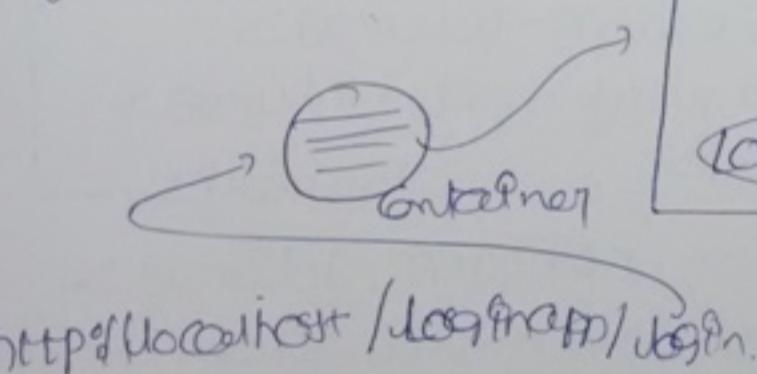
pattern defined in web.xml file to refer LoginServlet.class file available under

classes folder:

Deployment descriptor:

In deployment descriptor file, it will provide metadata or description about web application deployment required by the container to execute the application.

- In web applications, the name and location of deployment descriptor is agreed that is web.xml and WEB-INF folder.
- In web applications web.xml file is mandatory & optional is completely depending on the server which we used for our web application.
- In tomcat server web.xml file is optional, when we have not used servlets, filters, listeners and so on in our web application. If we use servlets, filters, listeners and so on in our web application in Tomcat server then web.xml file is mandatory.
- In case of web logic server web.xml file is mandatory irrespective of using servlets, filters and so on.



Note:

- upto servlets 2.5 version, web.xml file is mandatory if we use servlets, filters and listeners and so on, in any server.
- But in servlets 3.0 version, web.xml file is totally optional irrespective of whether we use servlets, filters, listeners and so on in our web application because servlets 3.0 version has provided an alternative to web.xml file in the form of annotations. This will applicable only if the servers support servlets 3.0 version only.

= In general, in web applications web.xml file, will include the following configurations.

1. welcome file configuration
2. display name configuration
3. context parameter configuration
4. servlet config & filter configuration
5. listeners configuration
6. initialization parameters configuration
7. load on startup configuration
8. session timeout configuration
9. error page configuration
10. tag library configuration
11. security constraints configuration

- To configure a particular servlet in web.xml file we have to use the following xml tags in web.xml file.

```
<web-app> → root tag. It includes the info about web application.  
      =  
<Servlet>  
      |<Servlet-name> logical name </Servlet-name>  
      |<Servlet-class> fully qualified name name of servlet </Servlet-class>  
      |</Servlet>  
<Servlet-mapping>  
      |<Servlet-names> logical name </Servlet-names>  
      |<URL-pattern> url pattern </URL-pattern>  
      |</Servlet-mapping>  
= </web-app>
```

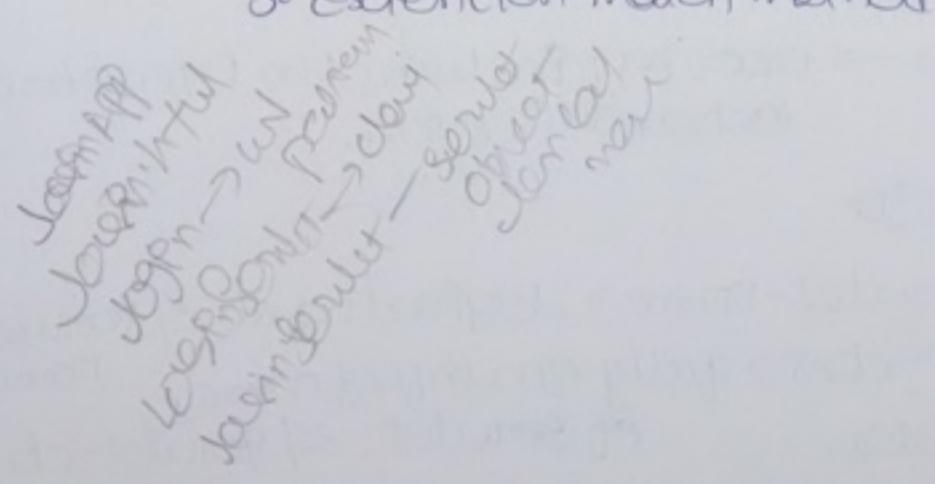
Example:

```
<web-app>
  <Servlet>
    <Servlet-name>LoginServlet </Servlet-name>
    <Servlet-class>com.bn>LoginServlet
  </Servlet>
</web-app>
```

```
<Servlet-mapping>
  <Servlet-name>LoginServlet </Servlet-
  name>
  <URL-pattern>/login </URL-pattern>
</Servlet-mapping>
```

~ In web applications, there are three ways to define URL patterns

1. Exact match method
2. Directory match method
3. Extension match method.



Exact match method;— In this approach, we will define url-pattern in web.xml file, It must be prefixed with '/' (forward slash) and it may be anything.

start pattern /abc/xyz <url-pattern> ✓

In this mechanism,

If we want to access the respective Servlet from client then we have to specify the same url pattern what we define in exactly web.xml file at client address bar.

Ex https://192.168.0.100/app1/abc/xyz ✓

~~exact match~~
~~forward slash~~
~~url-pattern~~
~~http://192.168.0.100/app1/xyz/abc~~ X exact match
~~http://192.168.0.100/app1/abc~~ X ~~don't consider~~
~~http://192.168.0.100/app1/abc/xyz/abc~~ X

Note: In web applications, we will use this mechanism when we have requirement to access the web application independently.

Forward
web browser address → http://192.168.0.100/app1/app1 forward slash
in written form <url-pattern>/app1<url-pattern>

for each query to one service

Directory match method:-

In directory match method, we will define our patterns in web.xml file, it must be prefixed with forward slash '/' and it must be terminated with '*'.

or <url-pattern>/abc/*</url-pattern>
* - any thing ✓

In this mechanism, If we want to access any specific server side resource (web.xml) then we have to provide url-pattern at client address bar, its prefix must be matched with the prefix of the url pattern defined in web.xml file.

Example:

http://localhost:8080/testapp/abc/xyz ✓

/abc/abc ✓

(xyz)/abc ✗

/abc ✓

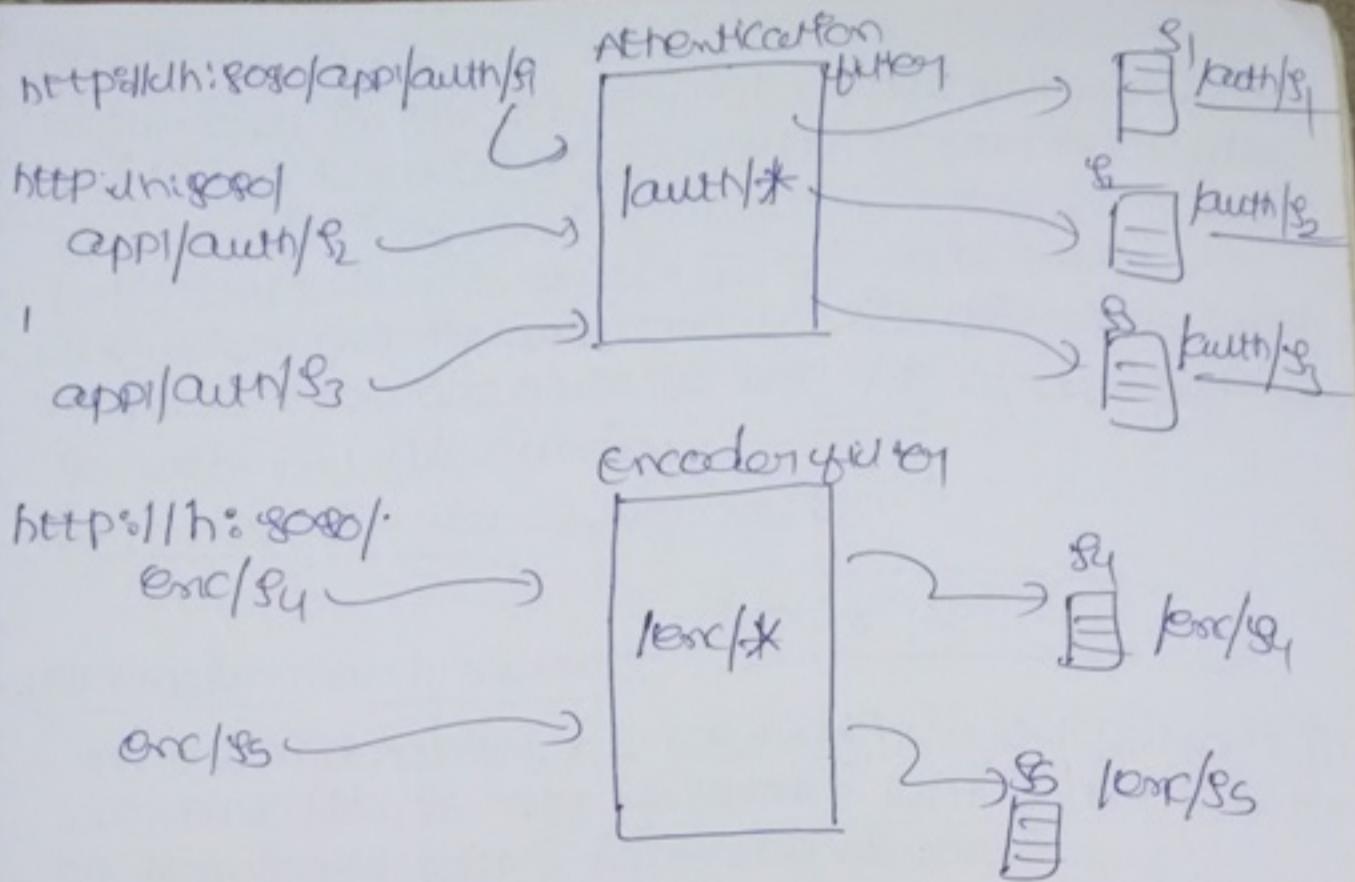
Notes

- In general
- In web applications we will define this mechanism, when we have a requirement to ~~allow~~ ^{serve} group's of request ~~serve~~ multiple request to a particular server side component

Note: In general in web applications we only write code to perform pre processing and post processing for the servlets. In web application it is required to perform Authentication, Authorization, date encoding and decoding and so on services as preprocessing for any request. In this context to implement authentication, authorization and so on services, we use the filters.

In web applications, a single filter may provide its preprocessing service for multiple number of servlets. That is if we send any request for any of these servlets then the respective filter has to be executed before the respective servlet execution.

In the above context, we need to define a URL-pattern for the filter component in such away to trap multiple no. of requests. To achieve this requirement, we have to use directory match method to define URL patterns.



Note:

In general in MVC based web applications, we only utilize a Servlet as Controller, as per MVC clearly and segregation all the requests must be mapped into controller Servlet. To achieve this requirement if we want to use directory match method then we have to use /* as wild-pattern in web.xml file under Controller Servlet Configuration.

Date: 8th April, 2012

Extension match method:

In this mechanism, we will define wild-pattern in web.xml file, it must be started with * and it must be terminated with a particular extension.

ex: <url-pattern> *.do </url-pattern>

In this mechanism, we will access the respective server side resource that providing an wild-pattern at client address bar, it will start with any thing but terminated with the extension which we specified in wild pattern defined at web.xml file.

expls:

http://127.0.0.1:8080/app1/reg.do ✓

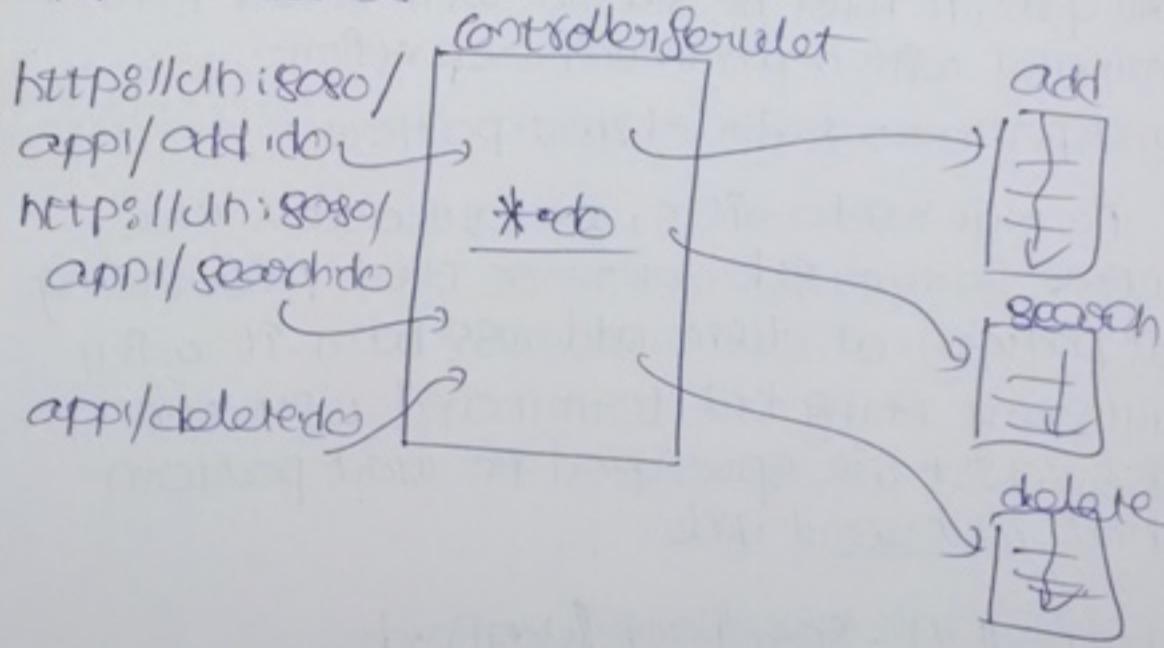
http://127.0.0.1:8080/app1/reg.do ✓

http://127.0.0.1:8080/app1/add.do ✓

http://127.0.0.1:8080/app1/search.xyz X

We will utilize this mechanism when we have a requirement to pass all the requests to a particular server side component and it will perform respective action as per the pattern name what we provided.

In MVC based web applications, we must send all the requests to the controller servlet only, since controller servlet has to pick up each and every request, pick up the pattern name and perform the respective action at server side. If we want to achieve the specific functionality in controller servlet then we have to define an exact pattern for the controller servlet by using exact match method.

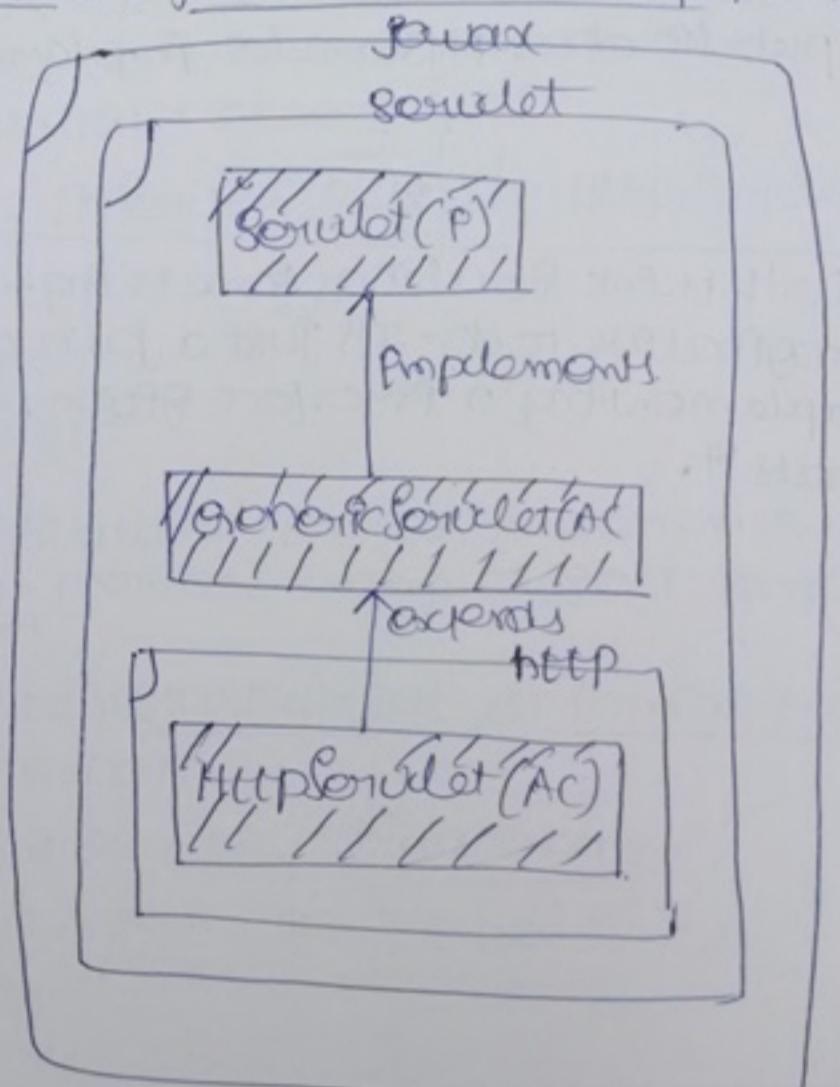


prepare the web resources as per the application requirement :-

In web applications, after preparing deployment descriptor, we have to prepare the web resources like HTML's, servlets, JSP's and so on.

how
to
make
a
thin
except
only
city

- To prepare servlet, Servlet API has provided the package required predefined library in the form of the following three components in `javax.servlet` and `javax.servlet.http` package



* What is servlet? and in how many ways we can prepare servlet?

- Servlet is an object available at server side, which must be implemented either directly or indirectly Servlet interface. As per the predefined library provided by Servlet API, there are 3 ways to prepare servlets.

1. Implementing Servlet Interface & In this approach, we will take an user defined class, it must be implemented from Servlet Interface.

public class MyServlet implements Servlet

```
{  
    ...  
}
```

Don't think Servlet as some thing. It is nothing programmatic to do. It's just a Java class which is implementing a interface given by Sun Microsystems. That's it.

2 Extending GenericServlet abstract class:-

If we want to create a servlet with this mechanism then we have to define an user defined class as Sub class to GenericServlet abstract class.

public class MyServlet extends GenericServlet

```
{  
}
```

3. Extending HttpServlet abstract class:- To define Servlets in this approach, we have to define an userdefined class, it must be a sub class to HttpServlet abstract class

public class MyServlet extends HttpServlet

```
{  
}
```

Start the Server:-

After installing Tomcat Server on our server machine there are 3 ways to start Tomcat Server.

- ① execute either startup.bat or tomcat7.exe files in the path,

Location
of
Tomcat
file

C:\Tomcat7.0\bin\tomcat7.exe

C:\Tomcat7.0\bin\startup.bat

② Use monitor service as system proxy ✓

Starts All programs > apache Tomcat 9.0. > monitor
tomcat

③ Use system service apache tomcat 7.0 ✓

goto Search option
Start menu

Type services.msc in search box

one dialogue box will open

Select apache tomcat 7.0 and select
the from start service.

— Access the web application; after starting

the server to access the respective web application

We have to open client browser, where we have
to type the complete application url like

http://localhost:8080/Megaphone/login ✓

It is possible to access the web application through
server administration console also. For this we
have to use the following url adminclient
browser.

http://localhost:8080/

If you provide the above url, tomcat admin
console will be open, stration

where we have to select managerapp on the
right side of screen in Tomcat 9.0 version. The
managerapp option was provided on the left hand
side of the console (screen) in Tomcat 6.0 version ✓

— If you select managerapp then list of applica-
tions will be displayed, where we have to select apply
the respective application ✓

First approach to design Servlets :-

If you want to design Servlets with this approach then we have to take an user defined class as an implementation class to Servlet interface.

public interface Servlet extends Serializable {

? public void init(ServletConfig) throws
ServletException;

public void service(ServletRequest req,
ServletResponse res) throws
ServletException, IOException;

public ServletConfig getServletConfig();

public String getServletInfo();

public void destroy();

}

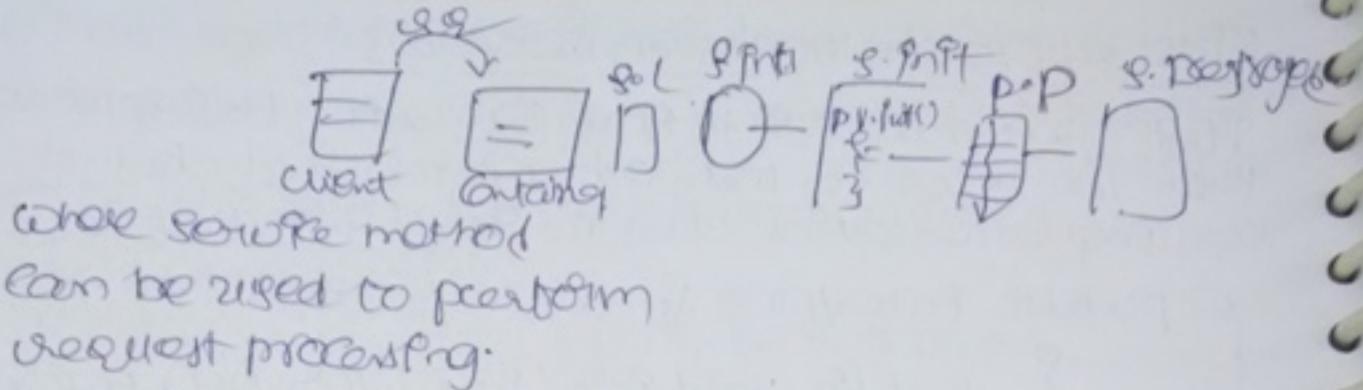
public class MyServlet implements Servlet

{
}

— where init() method can be used to perform Servlet initialisation.

Note: As part of Servlet initialisation, Container will access init method. For this Container has to create ServletConfig object.

ServletConfig is an object. It will provide all the configuration details of a particular Servlet, like the logical name of the Servlet, initialization parameters and the reference of ServletContext object.



- In Servlet applications, we will provide the required application logic inside service method because container will search and execute service method upon receiving request from client.
- Service method is almost equal to main method in GUI applications.
- Where getServletConfig() method can be used to return a generated ServletConfig object reference.
- Where getServletInfo() method can be used to return the generalized description about the respective servlet. Where destroy() method can be used to perform Servlet deinstantiation, that is ~~destroying~~ destroying Servlet object ✓

First client will give the request to container. Then container will look into web.xml. Before going to web.xml file container will create ServletContext object. After that when processing the web.xml, it will identify the servlet through url-pattern provided in web.xml. First container loads the servlet and then creates the object for servlet. This is called instantiation. After that container will start execution process of servlet. This is called Initialization. First it will execute init() method, but for executing init we require ServletConfig object. So container will create and we require ServletConfig object. So container will create and then execute init. After that go to service method. For this we require request & response. So it creates the two and then

- In the Servlet Interface init() method, service method, destroy method are called by Life cycle methods because which are participated directly in Servlet Life cycle.
- In Servlet Prototype getServletConfig, getServletInfo method are non-life cycle method because these methods are not participating in Servlet life cycle.
- While preparing Service method in Servlet Applications we have to use the following conventions:
 1. Specifying the response Content type.

2. In Service method, before generating dynamic response we have to give an information to the client about the type of response which we are going to generate from server, for this we have to use the following method of Servlet Response.

Ex:- `res.setContentType("text/html");`

text-

`public void setContentType(String MIME-Type)`

where MIME-Type can be text/html

text/xml

image/jpeg

image/gif

application/pdf

Now there objects of reference

to Service method. Now

execution starts in Service.

This is called Request processing.

After this container will prepare

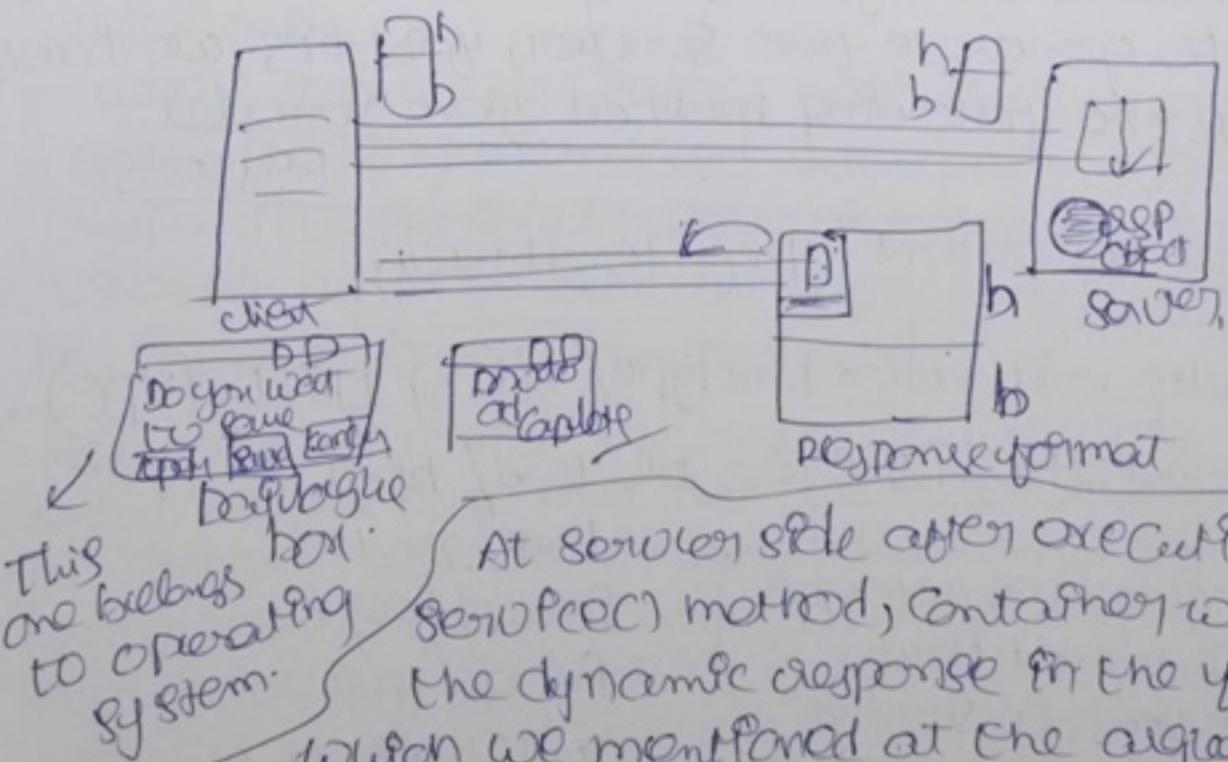
dynamic response and put it in response object. The content of this object will be displayed on browser as a response.

After that container will wait for some time for any other request from the browser. If not then destroy all the created objects.

when container execute the above instruction
then container will ~~will~~ set the specified
~~Content-Type~~ ^{Content-Type} to Content-Type response header in
response format.

Then response format will be encoded to client
browser, client will pickup Content-type response
header value and identify the response Content
type, prepare itself to hold the dynamic
response which is available in body part of
response format.

If the specified Content-type is normal text
oriented response then it able to display that
response directly otherwise container will
ask to the user where to save the generated
response.



At server side after executing the
service() method, container will prepare
the dynamic response in the form
which we mentioned at the argument in
setContentType() method. Here container
first get a particular option Content-type
in header of the ResponseFormat.

First the response format will be reached to the client. Client first reads the content-type header part of the response format. In this way, it recognizes the option Content-type. After recognizing this, browser will prepare itself to show the response.

In case the response is a pdf file, (Content in response object) then browser will generate a dialogue box saying whether you want to open the file or save the file or cancel. If you press enter, PC (browser) will start downloading. After this PC will generate download complete button.

In servlet applications if we want to generate dynamic response on response object then we require a writer object. Here the required writer object will be provided by servlet-api in the form of PrintWriter.

To get the predefined PrintWriter object we have to use the following method from Servlet Response.

```
public PrintWriter getWriter()
```

```
PrintWriter out = res.getWriter()
```

After getting PrintWriter object, we will generate dynamic by using println method of PrintWriter class in response object.

Here why we use another object to send if we use another response object, why can't we use already ServletResponse object.

```
graph LR; Response[Response] -- "getWriter creates" --> PrintWriter[PrintWriter]; Response -- putObject --> PrintWriter; Response -- printWriter --> PrintWriter; Response -- putObject --> SResponse[ServletResponse]; SResponse -- printWriter --> PrintWriter;
```

Q1 first with directory:

Open my computer. Go to D drive. Open folder Vipay
than open hello.txt file. How can you say your path

D:\Vipay\hello ↴

observe this is the path in Address bar
Here we use toolbar of Explorer

go to command prompt,

(not Internet
explorer)

go to the same location

D:\Vipay\hello → You can observe the same
thing in cmd also. We used
both the approaches back slash.

are same why because

both are operating systems one is windows ✓

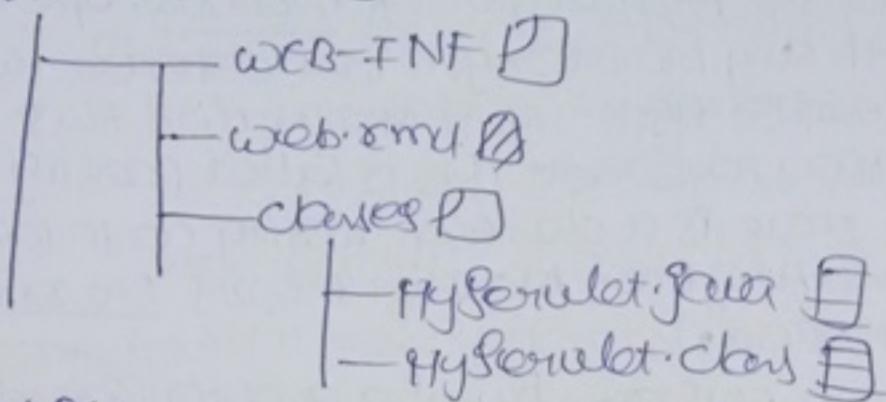
but remember,

2. DOS ✓

In browser, we will give request by using forward
http://localhost:8080/dogapp/dog slash ✓

out
post
http://localhost:8080/dogapp/dog

① first for webapp



PROCESSING

How to create folders: [At this time server was ~~was~~ be in off mode]

go to `catalina-7.0/webapps/`

Here you will create a folder and name it as firstforwebapp. Enter into this folder. Create sub folder and rename it as WEB-INF. Enter into this folder also. Create another sub folder classes. Now open notepad.

- prepare web.xml file

<web-app>

<Servlet>

<Servlet-name> FS </Servlet-name>

<Servlet-class> MyServlet </Servlet-class>

</Servlet>

<Servlet-mapping>

<Servlet-name> FS </Servlet-name>

<url-pattern> /first </url-pattern>

</Servlet-mapping>

</web-app>

give the above one as web.xml

under `catalina-7.0/webapps/firstforwebapp/WEB-INF/`

Now, I want to check whether

this file is written properly or not.

That means I want to check whether every file is created or not.

for this go to location - `firstServApp/WEB-INF`. Here you will see physical file of `web.xml`. Open that file with any browser. You will observe the o/p with `1` and `0` signs. If it shows like this then there way no error. This is called parsing. Suppose if there is a chance of missing close tag in above XML file. Even browser shows "The XML page cannot open!". We can check this one, by using option `Control-D` in notepad also.

Internally browser uses compiler to do this.

Preparing Servlet

Open Notepad:

```
import java.io.*;  
import javax.servlet.*;  
  
public class MyServlet implements Servlet  
{  
    public void init(ServletConfig config) throws  
        ServletException {  
        }  
    public void service(ServletRequest req,  
        ServletResponse res)  
        throws ServletException, IOException {  
        res.setContentType("text/html");  
        PrintWriter out = res.getWriter();  
    }  
}
```

```
out.println ("html");  
out.println ("<body>");  
out.println ("<center>");  
out.println ("<b><font size='7'>");  
out.println ("<br><br>");  
out.println ("Help from first Servlet Application");  
out.println ("</font></b>");  
out.println ("</center></body>");  
out.println ("</html>");  
public void getServletConfig() { return null; }  
public String getServletInfo() { return null; }  
public void destroy() { }
```

see the above program as MyServlet.java
In catalina\conf\webapps\firstServlet\WEB-INF\classes
Can we see the one
code for one
HTML code for one
out.print("Hello")
out.print("World")
out.print("<title>")
out.print("<h1>");

— To compile the Servlet file we have set the class path environment Variable to Servlet-api.jar file provided by Tomcat 7.0 version.

This was located in

C:\Tomcat 7.0\webapps\firstServletApp\WEB-INF\classes>

set classpath = C:\Tomcat 7.0\lib\Servlet-api.jar;
press enter

javac MyServlet.java

Then think how to execute this Servlet program

not find HttpServlet

X

what will happen if we do java HttpServlet ?

There are two ways to execute that servlet class

① Open client browser and type in the address bar

`http://localhost:8080/firstServletApp/first`

Then a HTML page will be open showing message
Hello from first servlet Application.

② Go to the tomcat administration console by type the word

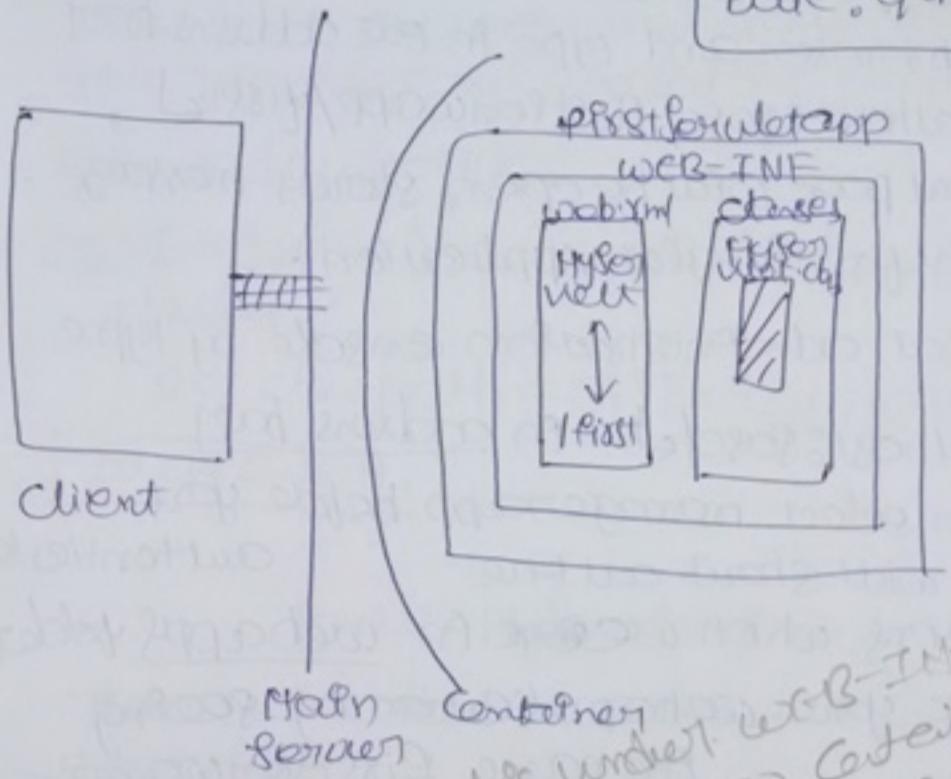
`http://localhost:8080/` in address bar

There you can select managerapp before you

After this it will show all the authenticated
web applications which were in webapps folder.

Go and click your webapplication by seeing
the name FirstServletApp.

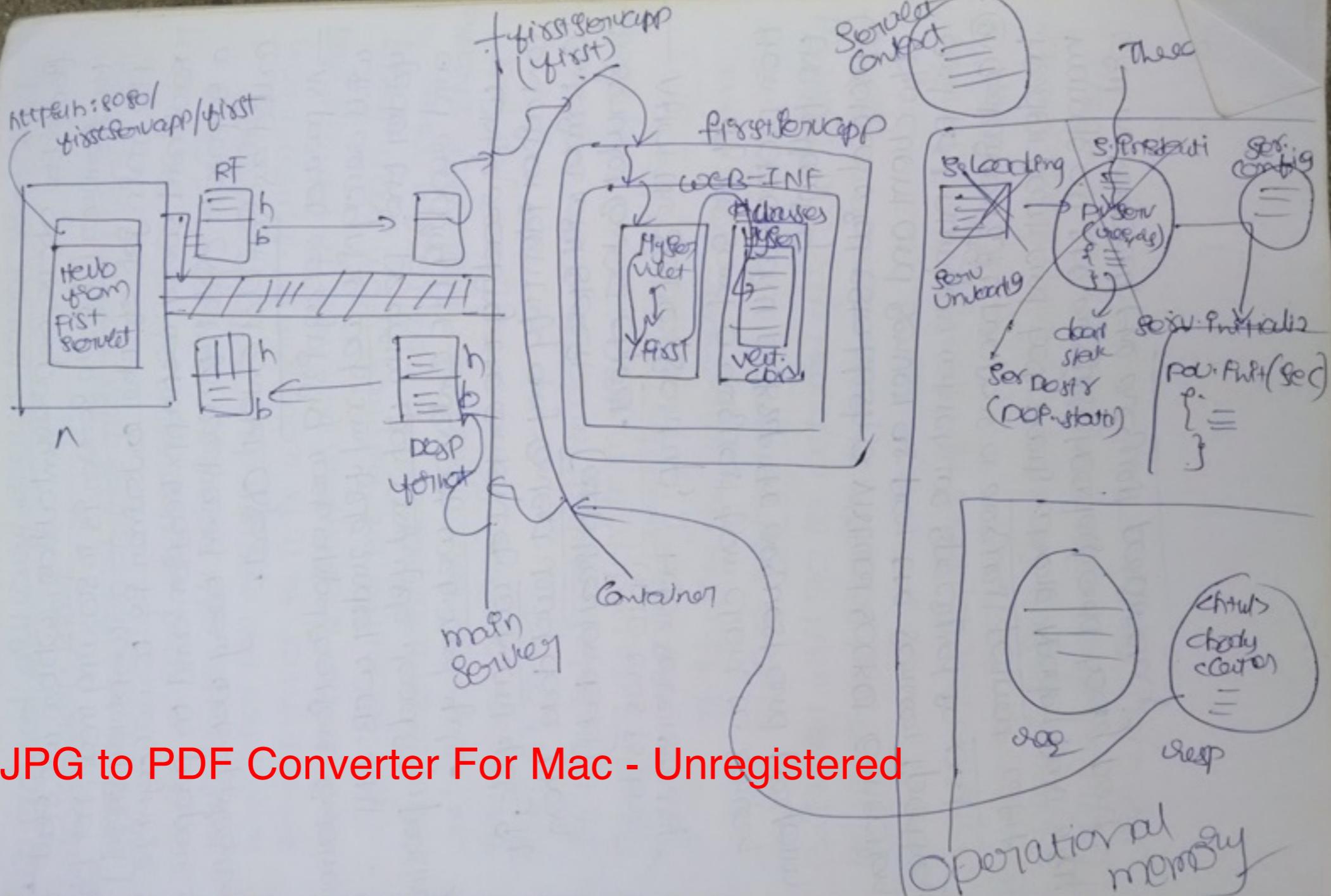
Date : 9th Apr, 2012



Actually web app is under WEB-INF folder
that is private area how Container accy
pt without knowing any information

there may be
more than one Container
in single server or
multiple servers exist
and security be assured to that
by service & is burden to bear all these
from different clients
the Container & is container owner &
very relieve of

JPG to PDF Converter For Mac - Unregis



JPG to PDF Converter For Mac - Unregistered

From the above representation, when we start the server (i.e. the server is in starting mode, not ~~completely started~~) the main job of the container is to recognise each and every web application and to prepare a separate object for each and every web application called as Servlet Context Object.

- As part of recognizing web applications, Container will recognise web.xml file under WEB-INF folder then perform web.xml file reading, parsing, and reading the content of web.xml file.
- While reading the content of web.xml file, If container identify any Context Level data then Container will store it (Application data) in Servlet Context object.
Up to this is done before Server startup ✓
- After the Server Startup,

If we send a request from client to server then protocol will pick up the request and perform the following

- ① protocol will establish a virtual socket connection b/w client and server as per the server ipaddress and port number which we specified in url
- ② ~~Container~~ protocol will prepare a request format with header part and body part, where header part will manage all the request headers and body part will manage all the request parameters

③ protocol will carry request format from client to server. This is the job of the protocol was completed.

- when we send request from client to the server, main server will pickup the request and check whether the request data is in well formed format or not, if it is well formed then main server will bypass request to the Container.
- upon receiving the request from main server, Container will pickup the application name and resource name, Container will check whether the resource name is an html file or jsp file resolvable or not.

If not Container will understand the resource name is an url-pattern of an particular servlet ^{it is available under class of folder.}

To identify the respective class name on the basis of url-pattern ^{servlet}

Container will go to web.xml file and identify the name and location of the respective servlet class.

After this, the process of identifying the servlet

when Container identify the requested servlet then Container will perform the following life cycle steps.

- ① Servlet Loading:- In this phase container will load servlet class byte code to the memory.
- ② Servlet Instantiation:- Here, container will create an object for the loaded servlet.
- ③ Servlet Initialization:-
The container will setup all the configuration details by providing servlet config object and by calling init method.
- ④ Creating request and response objects:-
After the servlet initialization container will create a thread to access service method, for this container has to create request and response objects.
- ⑤ Generating dynamic response:- After creating request and response objects container will access service(s) method, execute the application logic and generate dynamic response ~~and~~ on response object.
- ⑥ Dispatching dynamic response to client:-
When container generated thread reached to the ending point of service method then container will dispatch dynamic response to keep that thread in ~~dead~~ state, with this container will dispatch dynamic response to main server.

Note: when Container dispatch response to main server, main server will bypass that response to protocol where protocol will prepare response format with header part and body part, protocol will carry that response format to client.

Destroying ~~request~~ and ~~response~~ objects:—

when one dynamic response reached to client, protocol will terminate the virtual socket connection, by this container will destroy request and response objects.

Servlet deinstantiation:

After destroying request and response objects container will be in waiting state depending on the container implementation, when container identifies no further request for the same app then container will destroy servlet object.

Servlet unloading: After the servlet deinstantiation container will unload the servlet bytecode from operational memory.

Draw back w/ the first approach

In this approach to design a Servlet we have to take an user-defined class as an Implementation class to Servlet Interface. In this Context we must provide the implementation for all the methods declared in Servlet Interface or any Implementation class. But as per the application requirement, we need to provide only service method.

This approach will increase unnecessary methods in web applications and burden to the developers.

To overcome this problems we need to go for an alternative, where it is required to implement only service method. In the above Context the required alternative was provided by Servlet API in the form of GenericServlet.

Second approach to design Servlets

Date: 10th April, 2012

In this approach if we want to design servlets then we have to take an user defined class, it must be extended from GenericServlet abstract class ✓

— GenericServlet is an abstract class provided by Servlet API as an implementation class to Servlet interface.

```
public abstract class GenericServlet implements  
    {  
        private transient ServletConfig config;  
        public void init(ServletConfig config) throws  
            { this.config = config;  
            }  
        public void init() throws ServletException  
            {  
                /* Normally cycle method */  
            }  
        public abstract void service(ServletRequest req,  
            ServletResponse res) throws ServletException,  
            IOException;  
        public ServletConfig getServletConfig()  
            { return config;  
            }  
        public String getServletInfo()  
            { return null;  
            }  
        public void destroy()  
            {  
            }  
    }
```

public class Myclass extends GenericServlet

{
} = ↗ userdefined class ✓
}

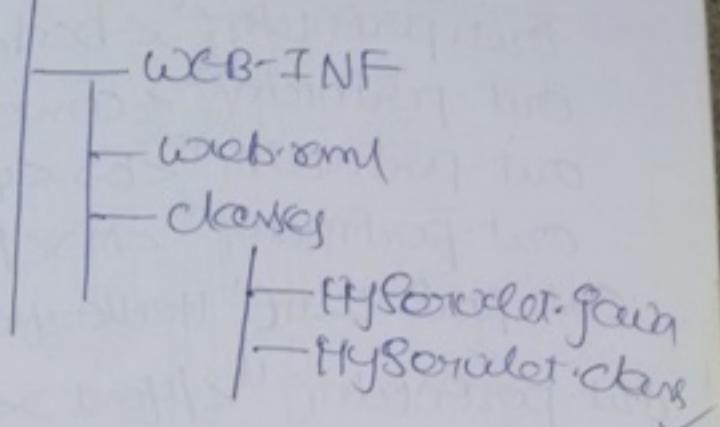
In Servlet API, GenericServlet is an idea provided by adaptor design pattern. predefined GenericServlet abstract class has implemented Serializable interface so that all the generic servlet objects [GenericServlet abstract class] are by default eligible for serialization and deserialization. But the predefined config variable will not be participated in serialization and deserialization processes because config variable was declared as transient. → why?

my analysis

If we have not declare configurable as transient variable in the generic servlet then it can participate in the configuration details that may leave us and other person of program through the network. we need to care this, why because if someone knows the configuration details he may potentially harm the application.

- In genericServlet init method it is overloaded method. In general in Servlet applications, it is required to ~~exist~~ override at the time of Servlet Initialization.
- In general in Servlet applications it is easier to interact with database from Servlets, where the complete JDBC environmental setup like driver loading, Connection establishment and Statement object creation has to be provided inside the init method, this approach will increase the performance of application.
- In the above context always it is suggested to override the second init method [init without parameter]. If JDBC environmental setup requires any data from ServletConfig object then it is suggested to override first init method that is init ServletConfig parameter.
- In genericServlet abstract class still service method is abstract method to improve sharability.

generic servlet app



example program

```
<web-app>
<servlet>
    <servlet-name> ms </servlet-name>
    <servlet-class> MyServlet </servlet-class>
</servlets>
<servlet-mapping>
    <servlet-name> ms </servlet-name>
    <servlet-pattern> /generic </servlet-pattern>
</servlet-mapping>
</web-app>
```

MyServlet.java

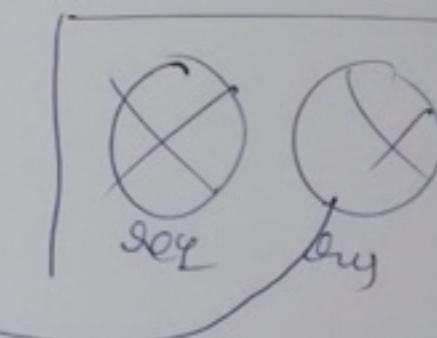
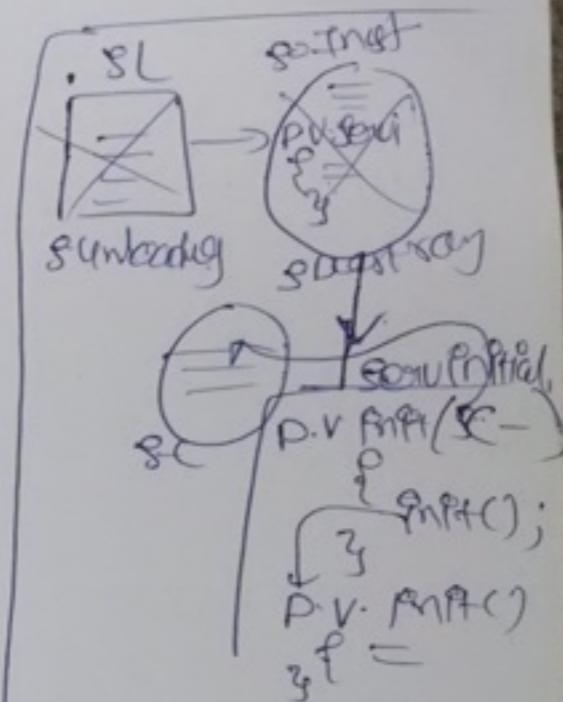
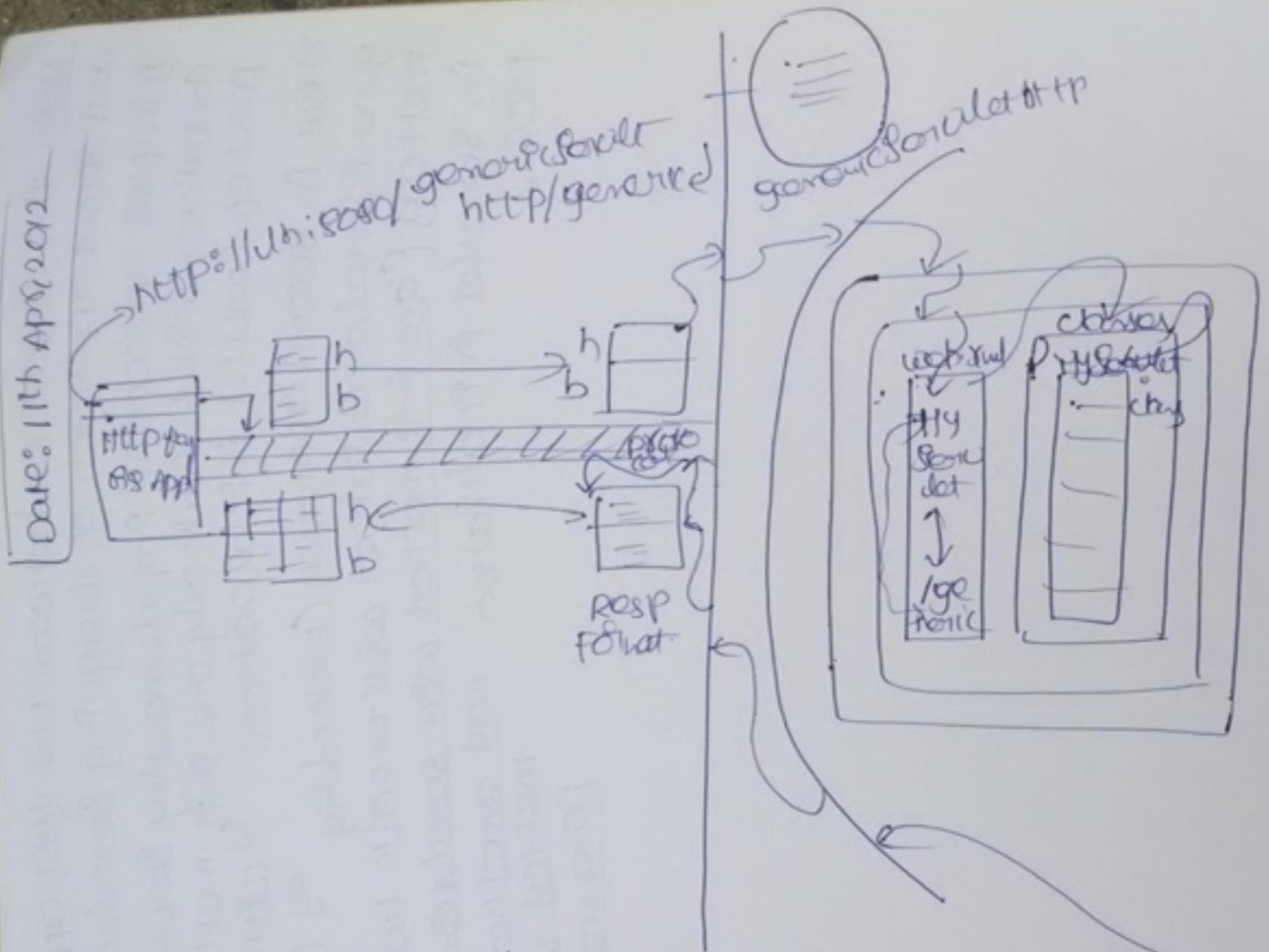
```
import javax.servlet.*;
import java.io.*;
public class MyServlet extends generic
{
    public void service(ServletRequest req,
                        ServletResponse res) throws
                                ServletException,
                                IOException
    {
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
    }
}
```

Annotations:

- getServletName(): returns the name of the servlet
- getServletClass(): returns the class of the servlet
- getServletMapping(): returns all the mappings of the servlet
- getServlet(): returns the servlet object
- destroy(): destroys the servlet

```
out.println("<html>");  
out.println("<body>");  
out.println("<center>");  
out.println("<b> <font style='font-size: 14pt;'>");  
out.println("<br><br>");  
out.println("Hello from GenericsForAll  
out.println(" </font></b>"); application);  
out.println("</center></body>");  
out.println("</html>");  
}  
}
```

Date: 11th April 2012



JPG to PDF Converter For Mac - Unregistered

Note: The difference between the first approach and second approach of designing Servlets is

- 1) In first approach (Implementing Servlet Interface) Container will execute only one init method as part of Servlet Initialization with ServletConfig parameter.
- 2) But in second approach (Extending genericServlet) Container will execute two init methods [one init method with ServletConfig object in Servlet Initialization and another init method without parameter]

But why:

what are the differences b/w genericServlet and

- ① GenericServlet is protocol is HTTPServlet protocol-independent (That means)
- But HttpServlet is HTTP protocol dependent.
- ② GenericServlet is able to process any type of protocol requests whereas HttpServlet is able to process only http protocol
- ③ GenericServlet is not very good compatible with protocols but HttpServlet is very good compatible with http protocol because GenericServlet will not provide any protocol specification at server side. But HttpServlet will provide http protocol specification at server side.
- ④ GenericServlet will not provide any flexibility for the developers to specify different types of requests but HttpServlet will provide a flexibility for the developers to specify different types of requests like get, post, put, head and so on -

Note: The main intention of introducing HttpServlet is to implement http protocol at server side.

If we want to design Servlets with this approach we have to take an user-defined class, which must be a subclass of ~~method~~ HttpServlet abstract class.

public abstract class HttpServlet extends ~~HttpServlet~~ {

 public void service(ServletRequest req, ServletResponse res) throws ServletException, IOException

 HttpServletRequest hreq = (HttpServletRequest)req;

 HttpServletResponse hres = (HttpServletResponse)res;

 service(hreq, hres);

 protected void service(HttpServletRequest hreq, HttpServletResponse hres)

 throws ServletException, IOException

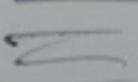
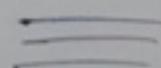
 String method = hreq.getMethod();

 if (method.equals("GET"))

 doGet(hreq, hres);

 if (method.equals("POST"))

 doPost(hreq, hres);



```
protected void (method.equals("DELETE"))  
    doDelete(huge, huge);  
} // service
```

```
protected void doGet(HttpServletRequest req,  
                      HttpServletResponse res)  
{  
    throws ServletException, IOException  
}  
}
```

```
protected void doPost(HttpServletRequest req,  
                      HttpServletResponse res)  
{  
    throws ServletException, IOException  
}  
}
```

```
public void doDelete(-, -)  
{  
    //  
}
```

On-life
code methods

public class MyServlet extends HttpServlet

```
{  
    public void doGet(HttpServletRequest req,  
                      HttpServletResponse res)  
    {  
        //  
    }  
}
```

In case of HttpServlet, we will provide our application logic by overriding doGet(), doPost(),
--- and so on, as per the request type what we have provided at client side.

Paul
for
VFC
written
using
Oracle
Database
In
order
to
use
in
TL
total
by
exp
& gp
only
but
to
do

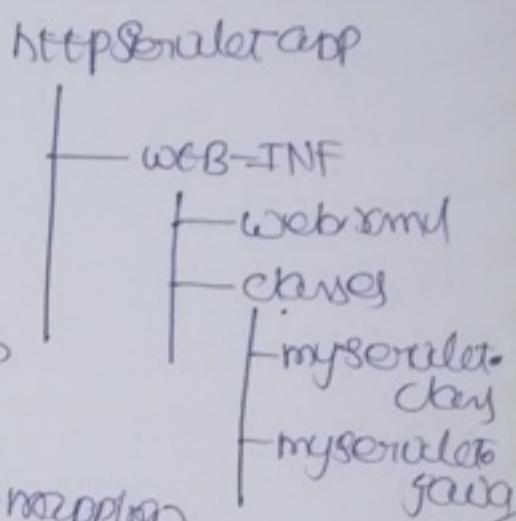
web.xml

```
<web-app>
  <Servlet>
    <Servlet-name>mys </Servlet-
    <Servlet-class>MyServlet <Servlet-class>
  </Servlet>
  <Servlet-mapping>
    <Servlet-name>mys </Servlet-mapping>
    <url-pattern>/http</url-pattern>
  </Servlet-mapping>
</web-app>
```

MyServlet.java

```
import javax.servlet.*; Here
import javax.servlet.http.*; 2 part = 2 part
import java.io.*; 3 part = 3 part
public class MyServlet extends HttpServlet
{
    public void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException
    {
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
        out.println("<html>");
        out.println("<body>");
        out.println("<center><b>");
        out.println("<font size=7 color='red'>");
        out.println("<br><br>");
    }
}
```

Date: 12th apr/ 2012



public void doGet(HttpServletRequest req, HttpServletResponse res)

Here service(HttpServletRequest req, HttpServletResponse res)

service(HttpServletRequest req, HttpServletResponse res) = 3 methods

public void destroy()

getServletConfig()

getServletName()

getExceptionType()

getExceptionName()

getExceptionMessage()

getExceptionStackTrace()

getExceptionType()

getExceptionName()

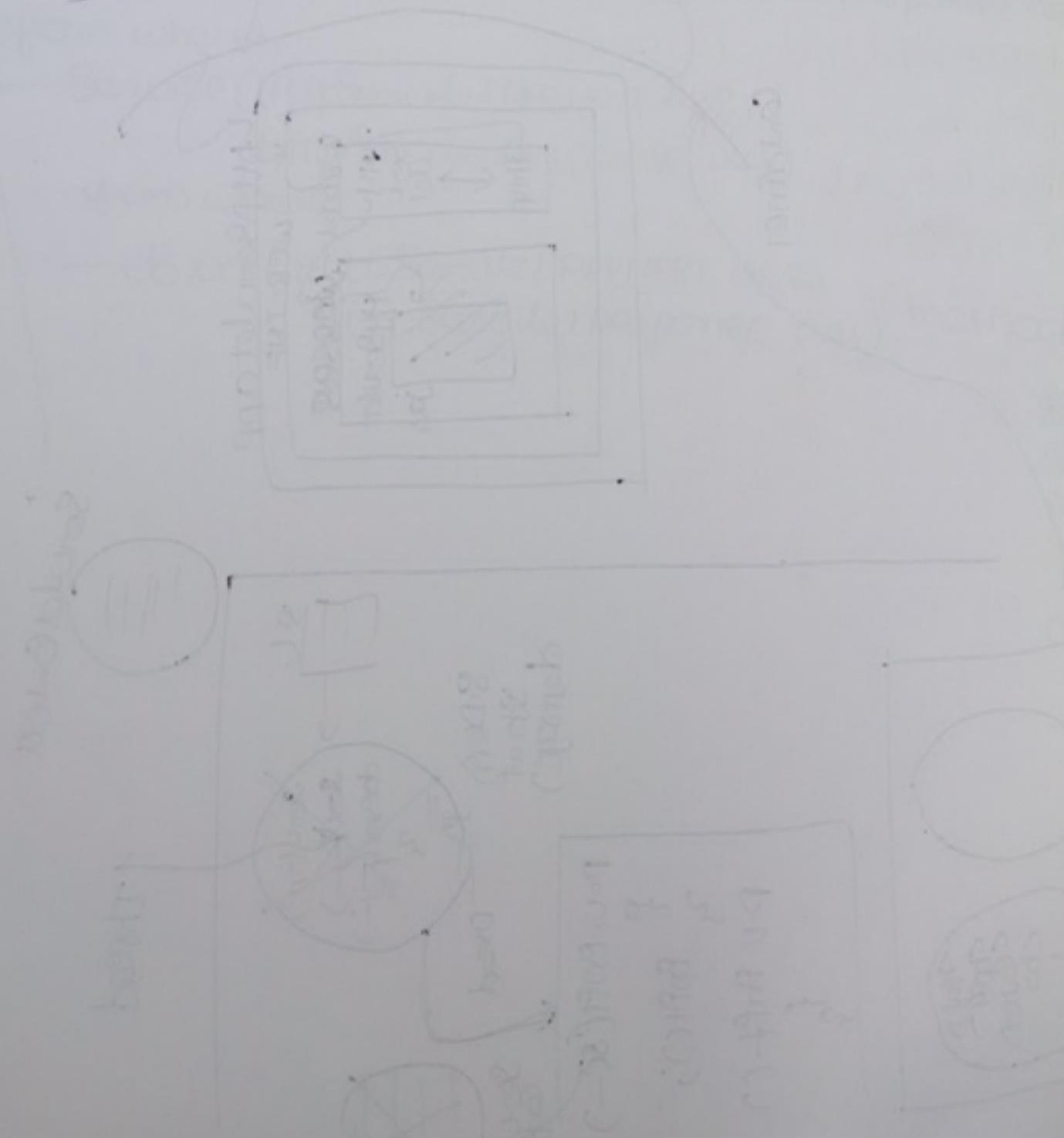
getExceptionMessage()

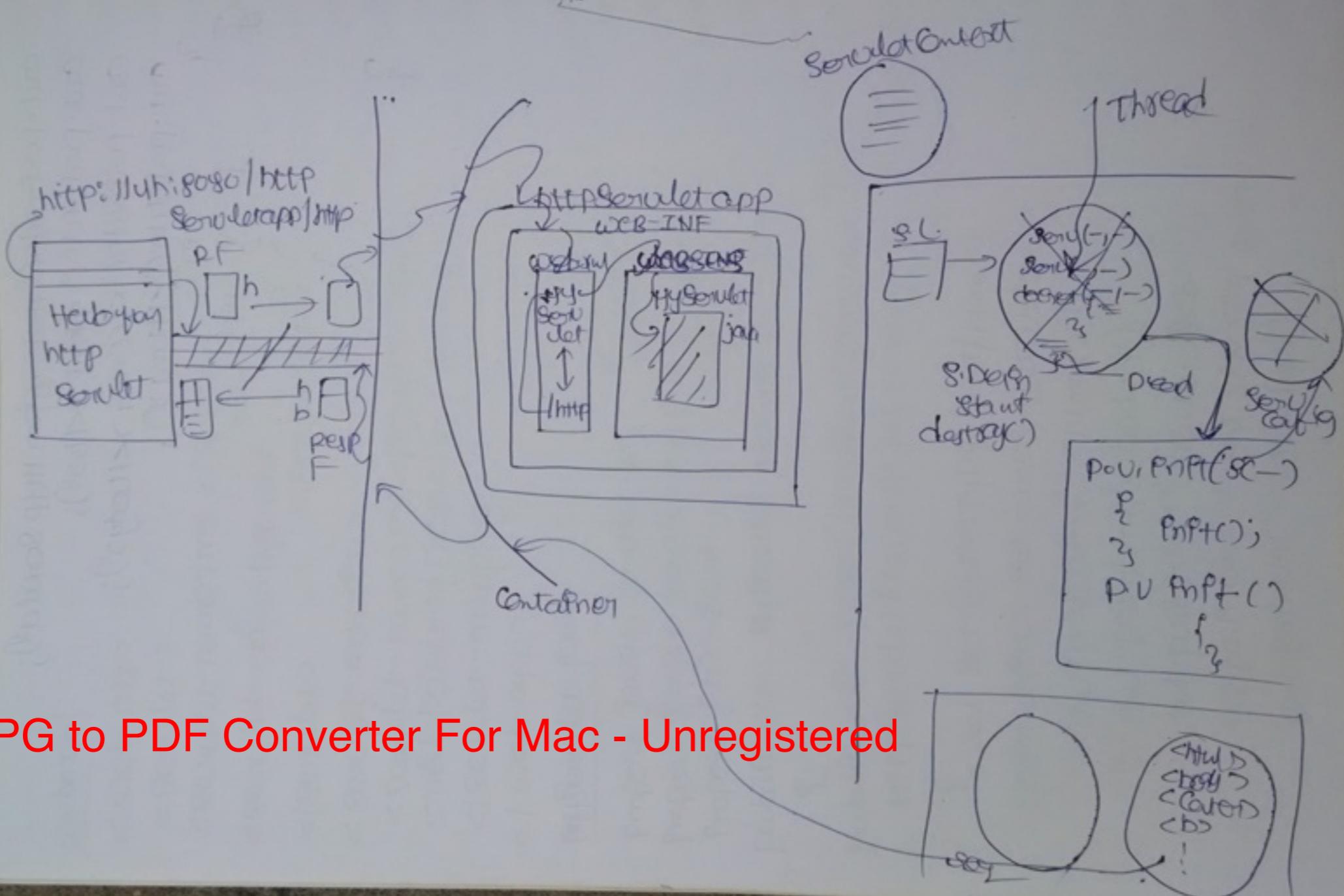
getExceptionStackTrace()

```
out.println("Hello from HTTP Server!");
out.println("<font><b>");  
out.println("</center></body>");
out.println("</html>");

? } P V m()
{ = }
```

? :





JPG to PDF Converter For Mac - Unregistered

The main difference between generic Servlet and HttpServlet flow of execution is

- ① In genericServlet execution to perform request processing container has to execute one service method [`service(ServletRequest req, ServletResponse res)`] only.

But in case of

- HttpServlet execution to perform request processing container will execute two service methods
 - `service(ServletRequest req, ServletResponse res)` from where,
 - `service(HttpServletRequest req, HttpServletResponse res)` from where,
 - `doXXX(HttpServletRequest req, HttpServletResponse res)` methods.

service
doXXX
Service
do

But
Container

Int. question

Is it possible to override service method in HttpServlet?

Ans Yes, it is possible to

override service method in HttpServlet. But it

is not suggestible. If we override service() method in HttpServlet then Container will execute only user defined service method, Container will not execute predefined service method. Due to this reason if we override service method in HttpServlet then it is not guarantee to reach upto ~~doGet, doPost, doPut, doDelete, doHead, doOptions, doTrace~~ methods.

N.B. If we want to override service method in Servlets always it is suggestible to use genericServlet, not suggestible to use HttpServlet. In HttpServlet it is always suggestible to override doXXX methods on the basis of request type.

pgm

Point Selection

static block

Is it possible to provide constructor and init method both at a time with in a single servlet?

Ans: In servlets it is possible to provide

both constructors and init method at a time.

If we provide a static block, a constructor and init method, doSet method, destroy method in a ~~Servlet~~ servlet, then container will execute static block at the time of performing servlet loading, constructor will be invoked at the time of performing servlet instantiation, init method will be executed at the time of servlet initialization, doSet method will be executed at the time of request processing and destroy method will be executed at the time of servlet destruction.

Note: In servlets if we want to provide initializations always it is suggested to use init method instead of constructor because servlet-config object support will be available in init method and it will not be available to constructors.

If we want to provide any constructors in a particular servlet class then the constructor should be public and zero argument because container will search and execute public and zero argument constructor while performing servlet instantiation.

— If we provide a parameterized constructor without zero argument constructor then container will raise an exception like java.lang.ServletException: cannot instantiating servlet class MyServlet, Root cause java.lang.InstantiationException

exception: MyServlet ✓

Ques

Date: 13th Apr, 2012

In general in Servlet application `destroy()` method can be used to perform Servlet deinstantiation. Where if we don't destroy method from `init() method` or `service() method` then what will be the response when we send a request?

A&1 In web applications Servlets will be executed by the container by following a particular life cycle which includes Servlet loading, Servlet instantiation, Servlet initialization, Request processing and Servlet deinstantiation.

To perform Servlet life cycle container software has included a separate module for each and every life cycle stage. At the time of executing Servlet life cycle respective modules container will access the respective methods which we provided inside Servlet class. For example to perform Servlet Initialization Container software has included hundreds of instruction. Among these instructions one instruction will access `init() method` that we provided inside the Servlet class. Therefore as part of Servlet initialization process Container will access `init() method` which is available in Servlet class, `init() method` is not performing the complete Servlet initialization.

— Similarly `destroy()` method

In this context even though we access `destroy()` method from `Print` method or service method of user-defined servlet class, Container will not perform servlet instantiation really. Container will access `destroy` method as a normal Java method.

Container

Servlet Loading

```
class c = clw.findName("Hyperlet")
```

Servlet Instantiation

```
Hyperlet ms = (Hyperlet) clw.createInstance();
```

Servlet Initialization

```
ms.init(sc);
```

Request process

```
ms.service(request, response);
```

Servlet destroy

```
ms.destroy();
```

Hyperlet Java

static

```
{ pop("sc"); }
```

```
Hyperlet()
```

```
{ pop("sInst"); }
```

```
P.U. Print(sc -)
```

```
{ destroy(); }
```

```
pop("sPrint");
```

```
P.U. serv(sReq, -)
```

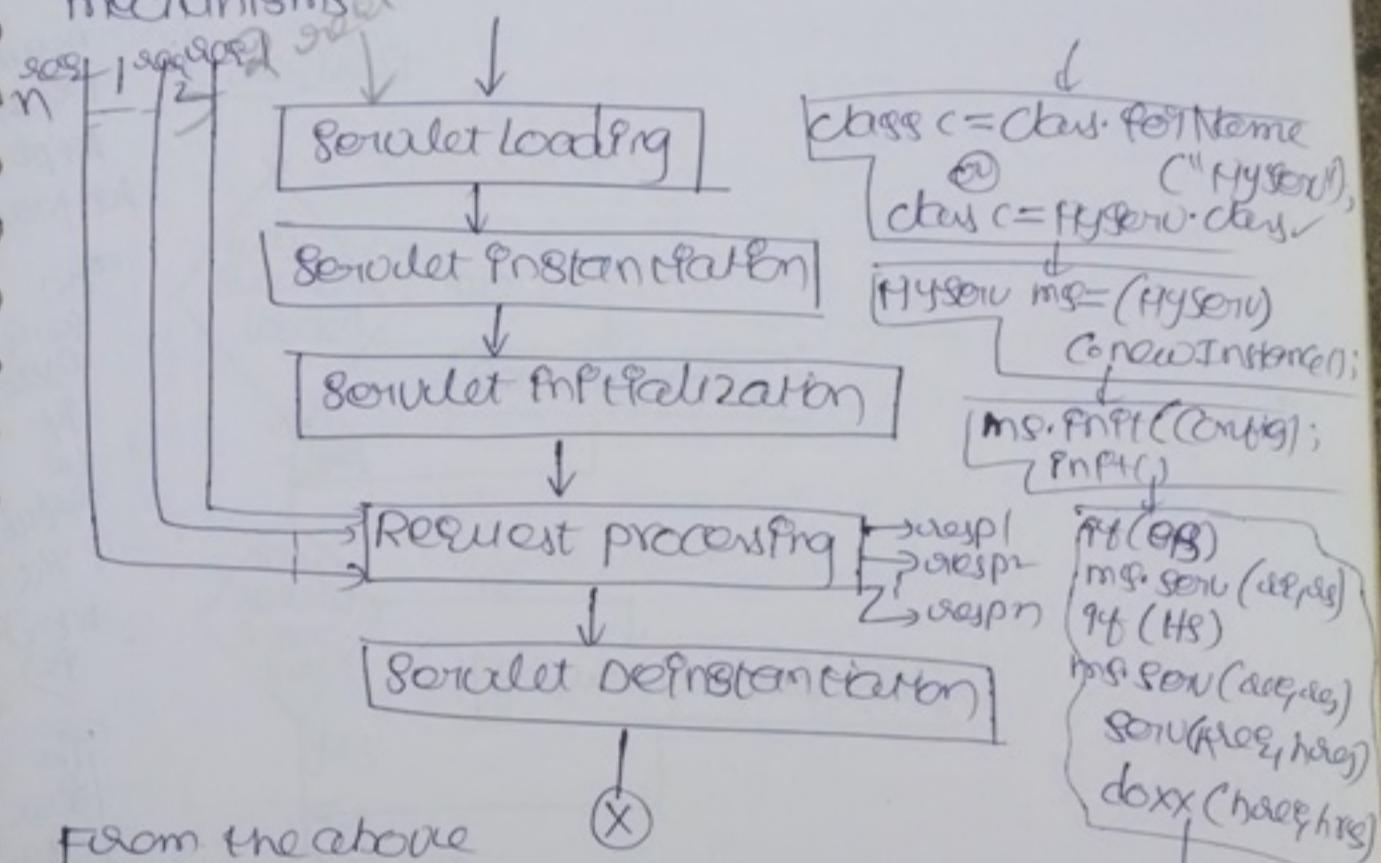
```
{ pop("RP"); }
```

```
P.U. destroy()
```

```
pop("sDestroy");
```

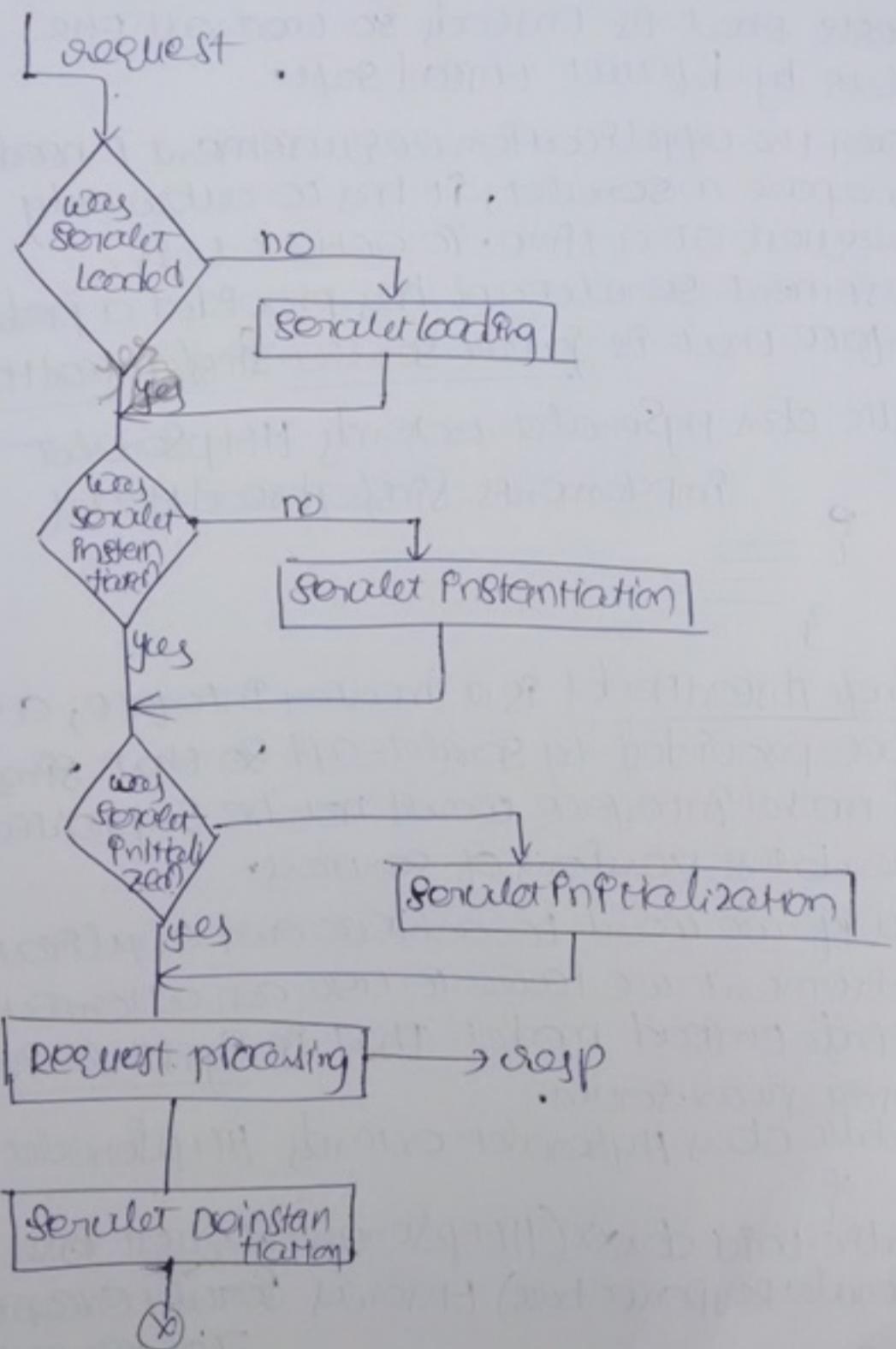
JPG to PDF Converter For Mac - Unregister

Servlet Life cycle:— when we send a request from client to server for a particular servlet then container will perform the following life cycle mechanisms



From the above representation,
 If we send multiple number of requests to a particular servlet then container will perform
 to servlet loading, servlet instantiation,
 servlet initialization life cycle stages for the
 first request only. From the second request onwards container will not perform servlet loading,
 servlet instantiation and servlet initialization.
 Container will bring all the requests directly to
 the request processing phase.

- If we send multiple numbers of requests at a time to a particular Servlet then Container will perform the following work for each and every request.



From the above servlet life cycle representation it is possible to send multiple number of requests at a time to a particular servlet. A single servlet object is able to process multiple number of requests that is already so that all the servlets by default thread safe.

As per the application requirement if need to prepare a servlet, it has to serve only one request at a time. To achieve this requirement Servlet API has provided a predefined interface that is javax.servlet.SingleThreadModel.

public class MyServlet extends HttpServlet implements SingleThreadModel

{
 ==
 }
}

SingleThreadModel is a marker interface, a dep deprecate interface provided by servlet-api so that single thread model interface would not be supported by all the latest versions of servers.

Still if we want to achieve our application requirement we have to use an alternative for single thread model. That is Synchronization.

import javax.servlet.
public class MyServlet extends HttpServlet

{
 ==
 }
}

 public void doxxx(HttpServletRequest req,
 HttpServletResponse res) throws ServletException,
 IOException

{
 synchronized(this)
 {
 ==
 }
}

— In servlet applications always it is suggested to avoid single thread model and synchronization because here we reduce the performance of the web application.

Load-on-Startup:-

Date: 14th April, 2012

- The main purpose of Load-on-Startup configuration is to perform a partial servlet loading, instantiation and Initialization at the time of Server Startup.
- In general in web applications container will perform servlet loading, instantiation and initialization for any servlet when it receive request from client after the server startup.
- If we perform servlet loading, servlet instantiation and servlet initialization at the time of server startup then container will directly move onto request processing phase when it receive request from the client after the server startup; this process will reduce response time, as a result the performance of the application will be improved.
- In general we will provide JDBC environment setup (driver loading, connection establishment and statement preparation) inside init method in the servlets if we want to interact with database from the respective servlet.
- In the above context if we have not provided load-on-startup configuration for the above servlet then container has to prepare JDBC environment as part of servlet Initialization when it receive request from client after the server startup.

This approach will reduce the performance of the applications.

In the above requirement if we provide load-on-startup configuration for the respective Servlet then container will prepare the complete JDBC environment at the time of Server startup as part of Servlet Initialization. In this context when we send request from client to server then the respective Servlet will enter into request processing without preparing JDBC environment.

- In general in MVC based web applications we will use a servlet as a controller. Here the controller servlet should have number of responsibilities to process a particular request. To achieve all these responsibilities controller servlet has to prepare some environment like preparing configuration files, preparing controller components and so on as part of Servlet Initialization.
- In the above context to improve the performance of MVC based web applications we ~~use~~ ^{can} ~~use~~ View

have to perform controller servlet initialization at the time of server startup. To achieve this we have to use load-on-startup configuration for controller servlet.

If we want to provide load-on-startup for any servlet we have to use the following code in web.xml tag.

<web-app>

 ≡

 <servlet>

 ≡
 <load-on-startup> value </load-on-startup>

 ≡
 </servlet>

 ≡
 </web-app>

- where load-on-startup value should be positive value including zero.
- where load-on-startup value should not be negative.
- If we provide load-on-startup configuration for more than one servlets then the order of servlets is completely depending on the load-on-startup values.
~~has~~
- If any servlet is keeping less load-on-startup value then that servlet will get more priority to load and if any servlet is having high load-on-startup value then that respective servlet will get less priority to load.
- If more than one servlet is having the same load-on-startup value then container will load both the servlets as per its own order. That is as per its internal implementation, container will not follow servlets configuration order from web.xml file.

```
start()
{ sys::cu("...") }
```

```
    }  
    HTTP()  
    { =
```

```
        }  
        init()  
        { =
```

```
            }  
            direct(HTTP, -)
```

```
                } =  
                }  
                destroy()  
                { =  
                }
```

double option in function.

```
quit()
{  
    destroy();  
}
```

Hyperactivities

```

state()
{
    SOP("GL");
}
HYPERACT()
{
    SOP("P-PH"));
}
P V Puit("gr-")
{
    destroy();
    SOP("gr-FnItal");
}
P-U Service("gr or destroy")
{
    SOP("gr Pn");
}
P-U destroy()
{
    SOP("gr Dest");
}
}

```

Game

close=close();

S. Post

HYPER-NE = (HYPER-new)

S. Initialized

obeying
the
diff
rules
P-P
except
the
P-P
large
case
BUT
is
not

→ local constants take two words
In external file take two words
Take the two files
 Local
quaternary
initialization

doubts?

PSP file area very
 Colors in normal -
 Greetings in contact ✓

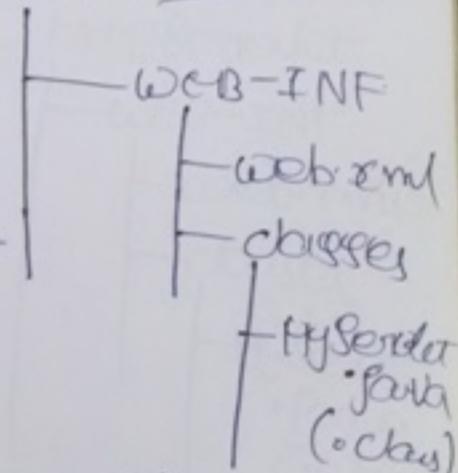
- Types of deployment:

If we prepare web applications directly under webapps folder then container will recognise and deploy all the applications at the time of server startup. This type of deployment is called as hard deployment, it is not suggested. As part of the web application development always it is suggested to use smooth deployment where we have to use the following steps.

1. prepare web application and ~~the war file at~~ some other location (not in webapps folder)

To prepare war file, we have to use the d:\apps\HTTPServer\ command prompt

```
D:\app\http\socket>jar -cvf  
                        app1.war **  
                        application  
                        Context
```



- ② Start the server and upload warfile to socket.

Start tomcat server by using either of the three approaches

To upload the warfile we have to open Tomcat server administration console.

<http://localhost:8080/>

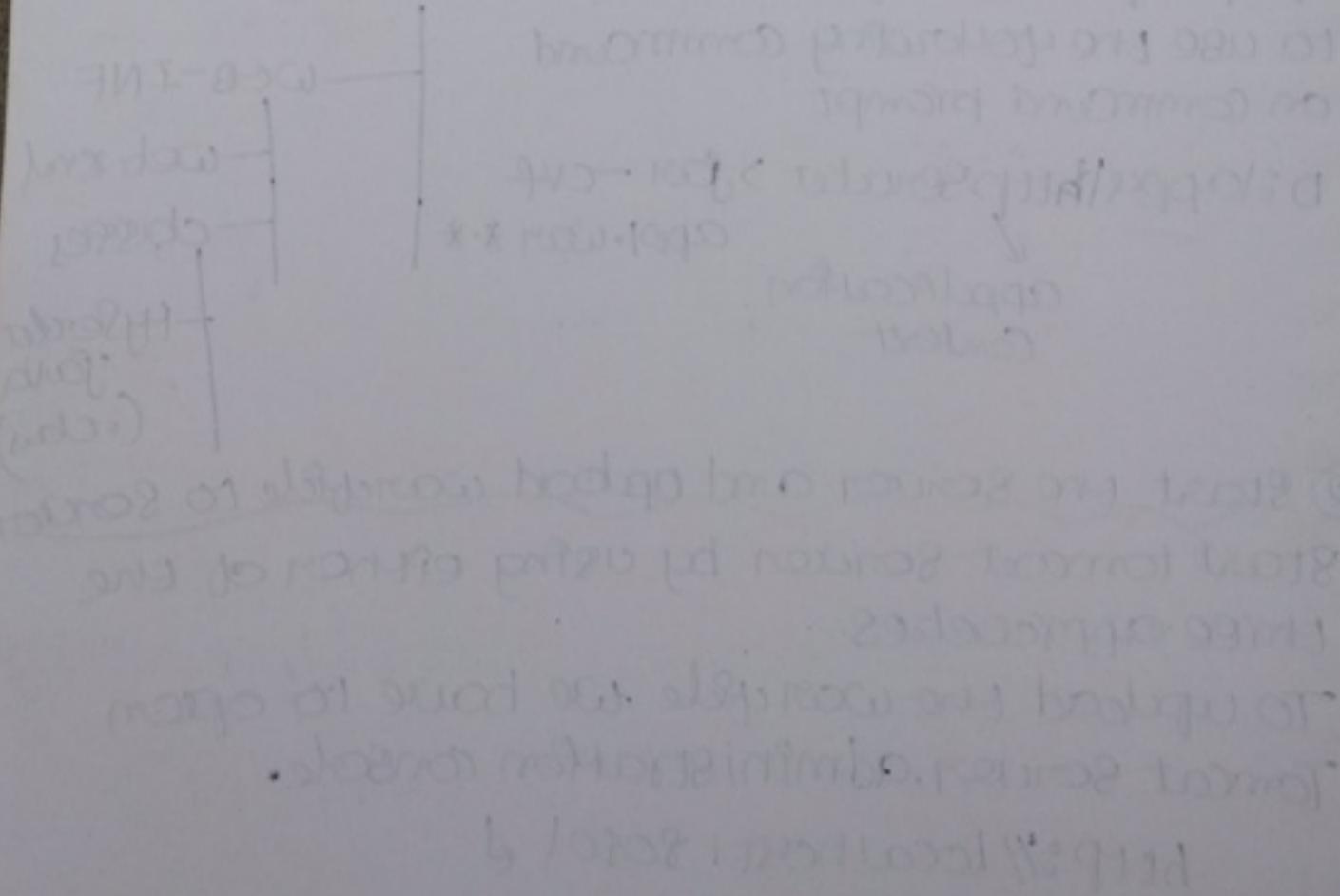
Select manager app

go to warfile to deployment section

Select warfile by click on browse button.

click on Deploy button.

2) Set weclper on deploy button automatically.
war file will be deployed to web application
and its context name will be available in
applications list then access the application.



JPG to PDF Converter For Mac - Unregis

Steps to deploy the web application into

weblogic 10.3 Server

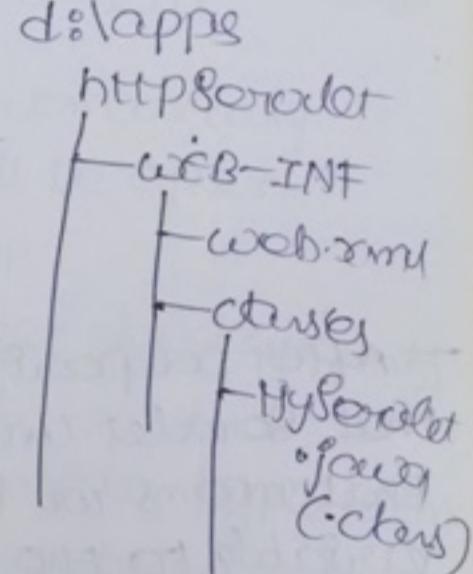
— weblogic Server is an application server, It will provide almost all the middleware services what application servers are providing like JNDI, JTA, JMS, JDBC and so on....

— ~~the~~ weblogic 10.3 Version is compatible with Java 6 version and it able to support Servlet 2.5, JSP 2.1 and so on web technologies.

Step①: prepare web application and its war file while preparing webxml in some any location. file we must provide doctype definition.

— we are able to get the doctype definition of web.xml file in web-app_2_3.dtd (This can be

↳ it may available in browser framework above.
available from internet etc)



<DOCTYPE web-app PUBLIC

"-//sun Microsystems, Inc//DTD web Application
2.3//EN"

"http://java.sun.com/dtd/web-app_2_3.dtd">

After understanding all methods & its implementation, we can move to the main question i.e. how to implement our own HttpServlet class.

As we have seen, HttpServlet class extends the Java's HttpServlet class. So, we have to implement the methods provided by HttpServlet class.

— After preparing Servlet.java, we have to compile that Servlet under that means we have to set class path environment variable to the following location.

```
c:\beal\wlservr-10.3\servers\dev1\weblogic\bin
```

Note: Where weblogic\bin file was provided by weblogic server, which includes the complete Servlet-API implementation provided by Servlet-API web logic server.

② Start weblogic application server.

Start all programs → double click weblogic

↓
weblogic server logs

when we start the weblogic WebLogicServer

server in the above approach

automatically weblogic will start and a browser will be open with weblogic server index page.

③ Start administration console.

— select start the administration console in endox

— provide user name and password both as weblogic

— click on login button then automatically weblogic server home page will be open.

④ Upload war file to the server.

Go to domain structure — select deployment on right side — choose install.

Select upload your files hyperlink

Select war file by click on browse button in deployment option

— then click next button.

— then click next button

next button

next button (now next button

Finish button

was disappear and)

— click on Testing button.

- click on <http://localhost:7001/webspi/>
- if you select one above hyperlink automatically another browser will be open, where specify word pattern and access that respective service.

Date: 15th April 2012

— Is it possible to design and execute any Servlet application without using web.xml file?

Ans: Yes. It is possible to design Servlet applications without using web.xml file but by using annotations. upto Servlet 2.5 version - to design and execute any Servlet it is mandatory to use web.xml file inorder to provide mapping b/w url pattern and the respective Servlet's class name. In the above context, to reduce xml dependency in web applications Servlet 3.0 version has provided annotation support as a replacement of web-

— Servlet 3.0 version has provided the xml file - compatible annotation support that is predefined library in the form of javaee-web-servlet. annotation package. Servlet 3.0 version has provided the following annotation to provide the mapping b/w url pattern and the respective Servlet class.

Syntax: @WebServlet(name = " -- ", value = " -- ", urlPattern = " -- ", initParams = " -- ",

where, applied small voidOnStartup = --)
name parameter will take the logical name of the servlet.
Here value parameter will take an url pattern
for the respective servlet.

Ex `@WebServlet(value = "/ann")` `@WebServlet("/ann")`

— In general in annotations the default param name should be value. If we want to ~~ever~~ provide only single parameter to the annotation then it is optional to specify value parameter name in annotations directly. It is possible to provide that value as a string.

Ex `@WebServlet("/ann")`

— where url pattern ~~slash~~ parameter will take one or array of url-patterns for the respective servlet.

Ex `@WebServlet(urlPatterns={"/ann1", "/ann2", "/ann3"})`

— where initparams parameter can be used to specify array of initialization parameters for the respective servlet, a specified initialization parameters will be stored in the respective servlet `ServletConfig` object.

— where loadOnStartup parameter will take a valid `loadOnStartup` value.

@WebServlet annotation is a class level annotation provided by `Servlet3.0` version, it cannot be used for variables and methods.

Ex `@WebServlet(name = "myServlet", urlPatterns = {"/first", "/second", "/third"}, loadOnStartup = 1)`

example program

public class MyServlet extends HttpServlet

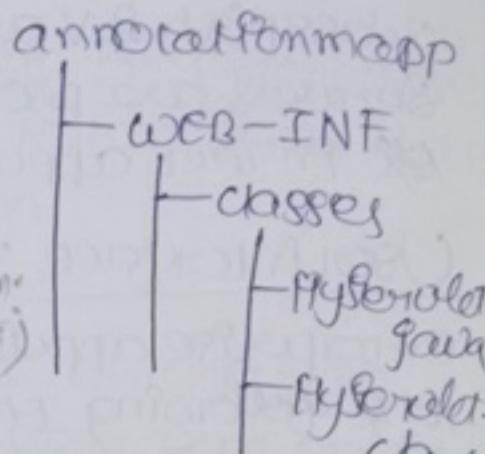
```
{  
}
```

file (pgm)

```
import javax.servlet.*;  
import javax.servlet.http.*;  
import java.io.*;  
import javax.servlet.annotation.*;  
@WebServlet("/ann")
```

public class MyServlet extends HttpServlet
{ public void doget(HttpServletRequest req,
HttpServletResponse res) throws ServletException

```
    {  
        res.setContentType("text/html");  
        PrintWriter out = res.getWriter();  
        out.println("<html>");  
        . . . ("<body>")  
        ("<center>")  
        ("<b><font size=7>")  
        ("<br><br>")  
        ("Annotation Servlet Test")  
        ("</font></b></center></body>  
        "</html>");  
    }  
}
```



ToException

- To execute the above servlet, underlying server has to provide annotation support.
That is, Servlet 3.0 version will be available along with Java 6 version.
- In general Tomcat 7.0, WebLogic 11 and above(b), GlassFish 2.1 and above, JBoss 7 and so on servers are providing annotations support ~~in~~ in web applications.

User Interface :- If we want to design any enterprise application then we have to provide the following three system logic layers.

1. User Interface layer → presentation logic
2. Business process layer → business logic
3. Data storage and access layer → persistence logic

User Interface layer : It is the topmost layer in enterprise application development. This layer is starting point for the user to interact with enterprise applications. This layer will provide a very good environment to accept data from user in order to execute enterprise application.

This layer will improve look and feel of the enterprise application.

This layer will provide a very good environment to perform client side validation ^{out} when javascript functionality

This layer will provide a flexibility for the developers to specify different types of requests get, post, head and so on.

- To prepare user interface layer in web applications, we will use a separate logic called as presentation logic.
- To prepare presentation logic in enterprise applications, we will use the technologies like awt, swing, html, Jsp and so on.

Business processing Layer:-

i. This layer is the heart of the enterprise applications, it will provide very good environment to define and execute all the business environment rules and regulations which are required by the client.

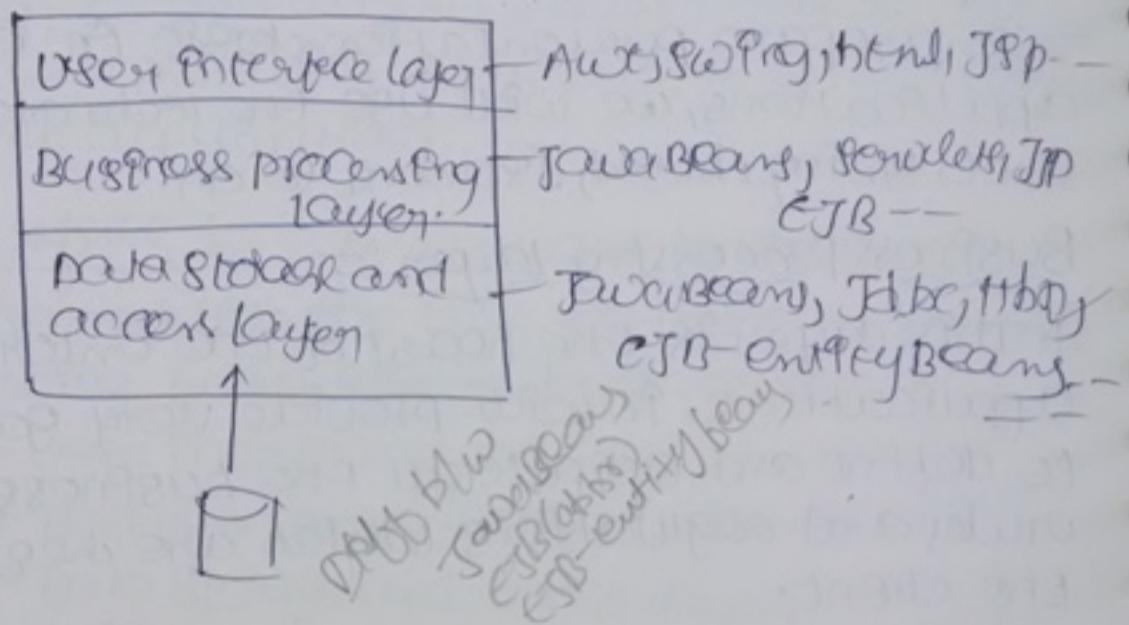
To prepare business processing layer in enterprise application development we will use a separate logic called as business logic.

- To prepare business logic in enterprise applications we will use the technologies like Java Beans, Servlets, Jsp and EJB and so on -

Data storage and access Layer:

It is the bottom most layer in enterprise application. It provides a very good environment to interact with the database in order to perform the basic database operations.

- To prepare data storage and access layer we will use a separate layer called as persistence layer.
- To provide persistence layer in enterprise application development we will use the technology like Java JDBC, Hibernate, EJB entity beans ^{ejb} Beans



2 → To prepare user interface in web application we will use html technology at basic level.

```

<html>
  <head>
    <Center><b><font style="font-size: 8px; color: red"> Registration
    Form
    </font>
    </b></Center>
  </head>
  <br> <br> <br>
  <br> <br>

```

Registration form

Name:	<input type="text" value="abc"/>
Password:	<input type="password" value="*****"/>
Qualification:	10th, 12th, Graduation
Gender:	Male Female
Technologies:	<input checked="" type="checkbox"/> Java <input type="checkbox"/> C <input type="checkbox"/> C++ <input type="checkbox"/> List
City:	<input type="text" value="Hyd"/>
Comments:	<input type="text" value=" "/>
<input type="button" value="Register"/>	

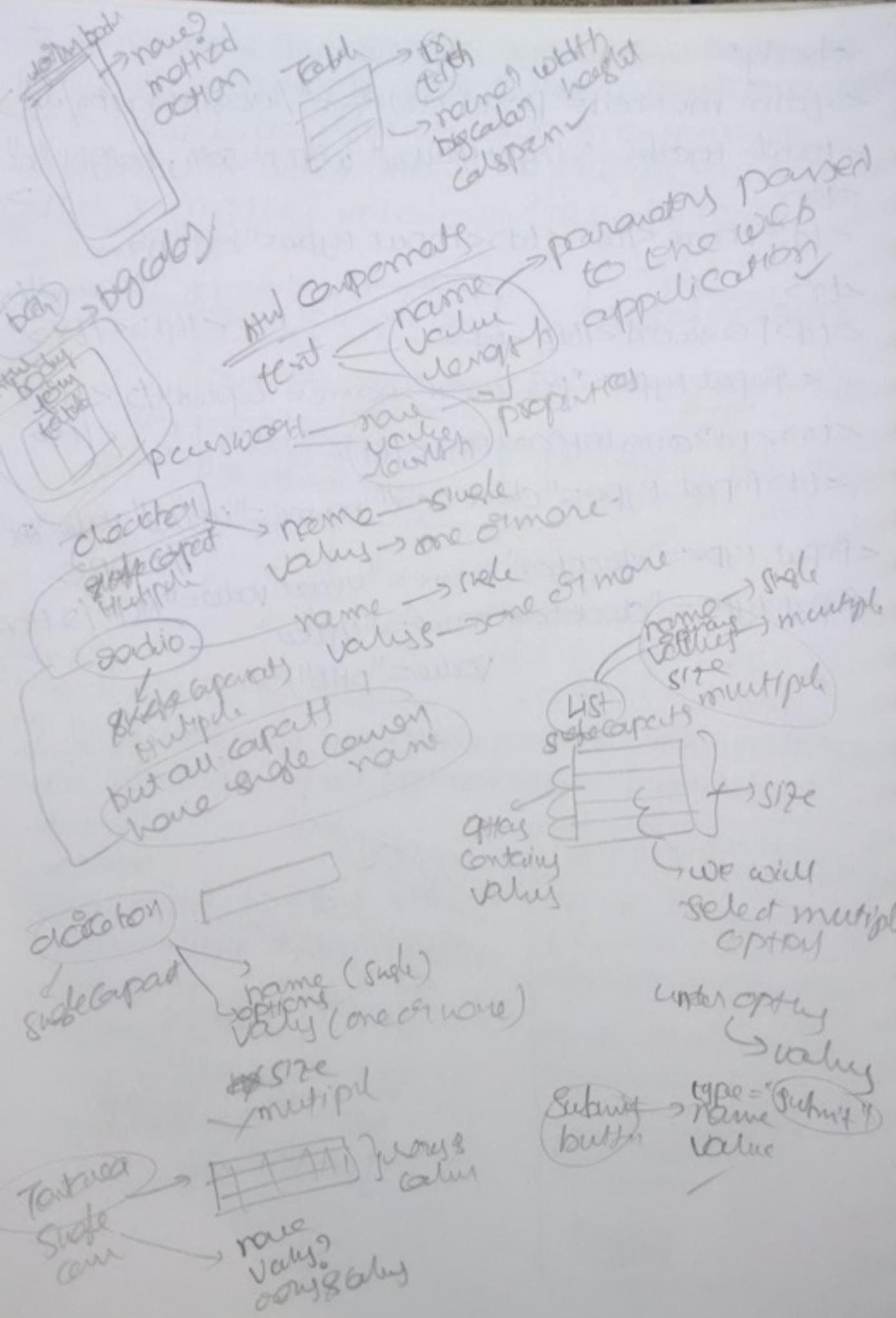
```

<body>
<form method="Post" action="/registration.cgi">
<table border="1" cellpadding="5" width="50%", height="50%">
<tr> colspan="2">
<td> Name <td> <input type="text" name="name" value="John Doe" length="10"/>
<tr> colspan="2">
<td> Password <td> <td>
<input type="password" name="repeat" value="Enter your password" /> <td>
<tr> <td> Qualification <td> <input type="checkbox" name="qual" value="BSC" checked="checked" /> BSC
<td> <input type="checkbox" name="qual" value="MCA" /> MCA
<td> <input type="checkbox" name="qual" value="PHD" /> PHD This was displayed actually on the screen
<td>
</tr>

```

upto now

- Servlet
- GenericServlet
- HttpServlet
- ServletConfig
- ServletRequest
- ServletResponse



- In web applications, we will prepare html forms under application folder. Once if we deploy html pages under application folder then it is possible to access that html file by using directly its name in the url.
`https://127.0.0.1:8080/registrationapp/registrationform.html`
- If we use the above url, then the respective html page file will open at client browser where we have (page) to fill all the fields then we have to submit that html form to server.
- When we send a request from user form to server for a particular servlet then the user provided data will be stored in request object in the form of key:value pairs that is request parameters.
- In general in web applications request object is able to accommodate the following three types of data
 1. Request headers
 2. Request parameters
 3. Request attributes
- The request headers are the key:value pairs, which will represent the metadata about client, like the content type which are supported by browser, the encoding mechanism (bundled mech supported by browser).

Specified-accepted language and so on.

- To get a particular request header value, we have to use the following method from Request
public String getHeader(String name)
- To get multiple values which are associated with a single request header, we will use the following method

public String[] getHeaderValues(String name)

- To get all the names of request headers, we will use the following method

public Enumeration getHeaderNames()

Request parameters:

Request parameters are the key-value pairs which will be stored on to the request object at the time of creation and which are provided by the user at user interface.

To get a particular request parameter value from request object then we have to use the following method from Request.

public String getParameter(String name)

- To get multiple values which are associated with the particular request parameter, we will use the following method

public String[] getParameterValues(String name)

- To get all the names of request parameters available in request object we have to use the following method

`public Enumeration getParameterNames()`

- Request Attribute:

are the key,value pairs, which will be stored onto the request object after executing the request object and while executing a particular servlet.

- In web applications, parameters data will provide static inclusion and attributes data will provide dynamic inclusion.

- To set an attribute on the request object, we will use the following method from `Servlet`

`public void setAttribute(String name, Object value)`

- To get a particular attribute value from request object, we will use the following method

`public Object getAttribute(String name)`

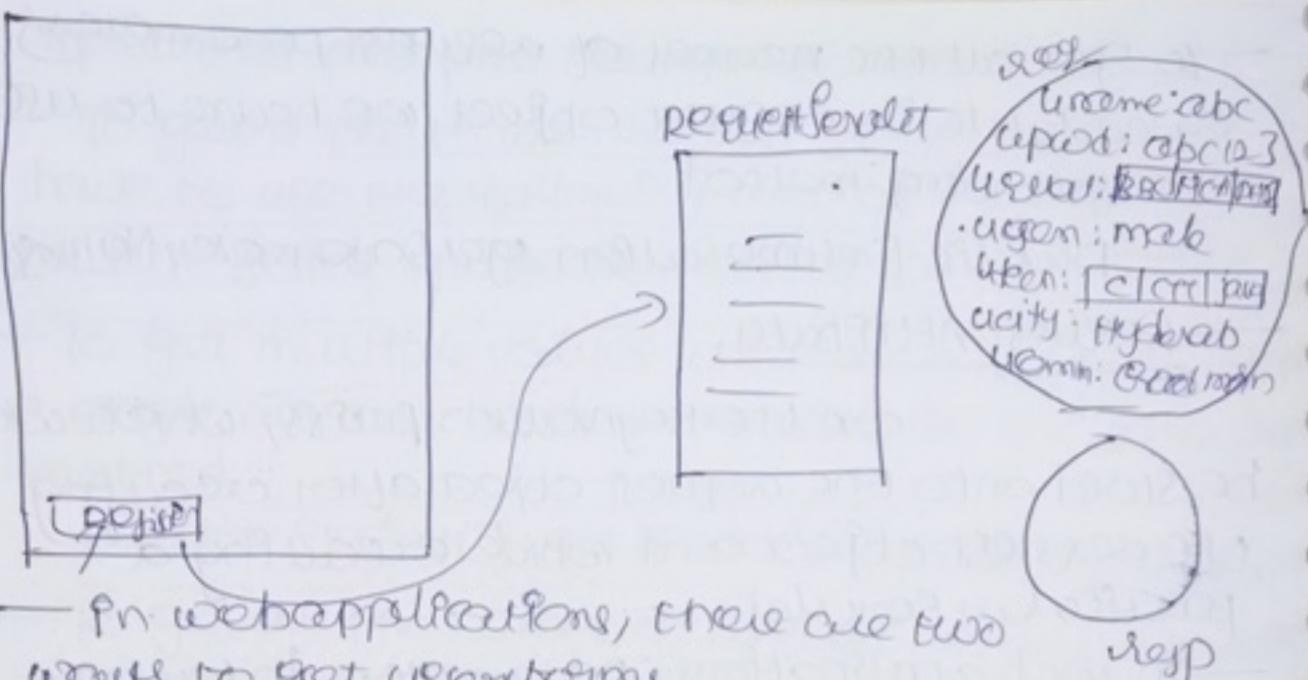
- To get all the names of the attributes from request object we will use the following method

`public Enumeration getAttributeNames()`

- To remove an attribute from request object we will use the following method

`public void removeAttribute(String name)`

`getAttributeValue(String name)?`



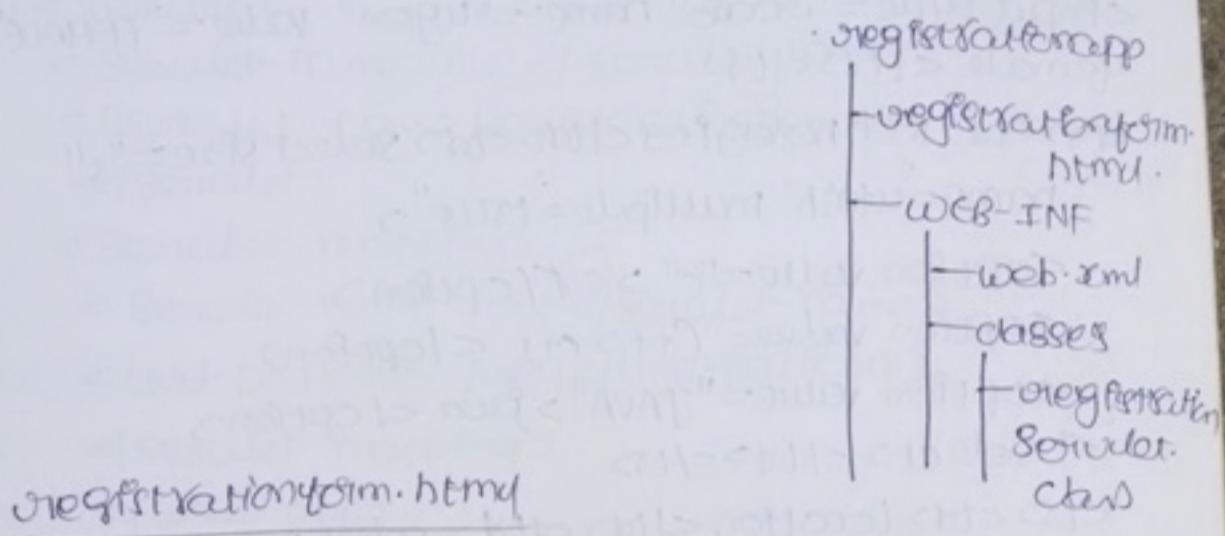
In web applications, there are two ways to get user form:

- ① Static form generation.
- ② Dynamic form generation.

① Static form generation: In case of static form generation we will deploy a separate html file under application folder and we will access it to get a form at client browser. In this case we are able to get user form by the execution of a particular html page that is start page.

② Dynamic form generation: In case of dynamic form generation we will provide the required html logic inside a particular servlet, we will access that servlet when we require a form at client.

In this case user form will be generated by the execution of a particular servlet that is dynamic resource.



registrationform.html

```

<html>
<body bgcolor="green">
<center>
<form method="post" action="/registrationapp/reg">
<table border="1" style="width: 100%; border-collapse: collapse; border: none; background-color: #ffffcc;">
    <tr><td colspan="2" style="text-align: center; font-size: 18px; font-weight: bold; color: black; padding-bottom: 10px;"><b>Registration Details</b></td></tr>
    <tr><td style="width: 15%;">User Name</td><td><font style="font-size: 14px; font-weight: bold; color: black; margin-top: 5px;"><input type="text" name="uname"/></font></td></tr>
    <tr><td style="width: 15%;">password</td><td><font style="font-size: 14px; font-weight: bold; color: black; margin-top: 5px;"><input type="password" name="upwd"/></font></td></tr>
    <tr><td style="width: 15%;">Qualifications</td><td><input type="checkbox" name="ugual" value="BSC"/> BSC
        <input type="checkbox" name="ugual" value="HCA"/> HCA
        <input type="checkbox" name="ugual" value="PHD"/> PHD</td></tr>

```

```
<tr><td> Gender </td><td> <input type="radio" name="ugen" value="MALE"/> male  
<input type="radio" name="ugen" value="FEMALE"/> female </td></tr>  
<tr><td> Technologies </td><td> <select size="3" name="utech" multiple="multiple">  
    <option value="C"> C </option>  
    <option value="C++"> C++ </option>  
    <option value="JAVA"> Java </option>  
</select> </td></tr>  
<tr><td> Location </td><td> <select name="uloc">  
    <option value="Hyderabad"> Hyderabad </option>  
    <option value="Chennai"> Chennai </option>  
    <option value="Pune"> Pune </option>  
</select> </td></tr>  
<tr><td> Comments </td><td> <textarea name="ucomment" rows="5" cols="30"> </textarea> </td></tr>  
<tr><td> <input type="submit" name="Submit" value="Registration!"/> </td></tr>  
</table></form></center></body></html>
```

web.xml

```
<web-app>
  <Servlet>
    <Servlet-name> dg </Servlet-name>
    <Servlet-class> RegistrationServlet </Servlet-
      class>
    </Servlet>
  <Servlet-mapping>
    <Servlet-name> dg </Servlet-name>
    <url-patterns> /reg </url-patterns>
  </Servlet-mapping>
</web-app>
```

RegistrationServlet.java

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

public class RegistrationServlet extends HttpServlet
{
  public void doPost(HttpServletRequest req,
                     HttpServletResponse res) throws ServletException
  {
    res.setContentType("text/html");
    PrintWriter out = res.getWriter();
    String uname = req.getParameter("uname");
    String[] qual = req.getParameterValues("qual");
    if(uname == null)
      out.println("User name not defined");
    else
      out.println("User name is " + uname);
    if(qual == null)
      out.println("Qualifications not defined");
    else
      out.println("Qualifications are " + qual[0]);
  }
}
```

```
String user1 = "";
for (int i=0; i<user1.length(); i++)
{
    user1 = user1 + " " + user1(i);
}

String user2 = user2.getparameter("ugen");
String[] tech = user2.getparametervalues("utech");
for (int i=0; i<tech.length; i++)
{
    user2 = user2 + " " + tech(i);
}

String user3 = user3.getparameter("ucfty");
String user4 = user4.getparameter("ucomment");
out.println("<html>");
out.println("<body background='light green'>");
out.println("  <center><br><br><br><br>");
out.println("  <table border='1'>");
out.println("    <tr><td>colspan=2</td><td>");
```

Registration Details

```
    <td>username</td><td>"+
```

```
        username+"</td></tr>);
```

```
out.println("  <tr><td>password</td><td>" + upwd
```

+ &pwd + "

```
        + "</td></tr>);
```

```
out.println("  <tr><td>Qualification
```

```
        <td><input type='text' value='1234567890'></td></tr>);
```

```
out.println("<tr><td>Gender</td><td>" +  
        agent + "</td></tr>");  
out.println("<tr><td>Technology</td><td>" +  
        uTech + "</td></tr>");  
out.println("<tr><td>Location</td><td>" + uCity  
        + "</td></tr>");  
out.println("<tr><td>Comments</td><td>" +  
        uComments + "</td></tr>");  
out.println("table></div></body></html>");  
    }  
}
```

Compilation :

C:\Tomcat7\webapps\oieRegistration

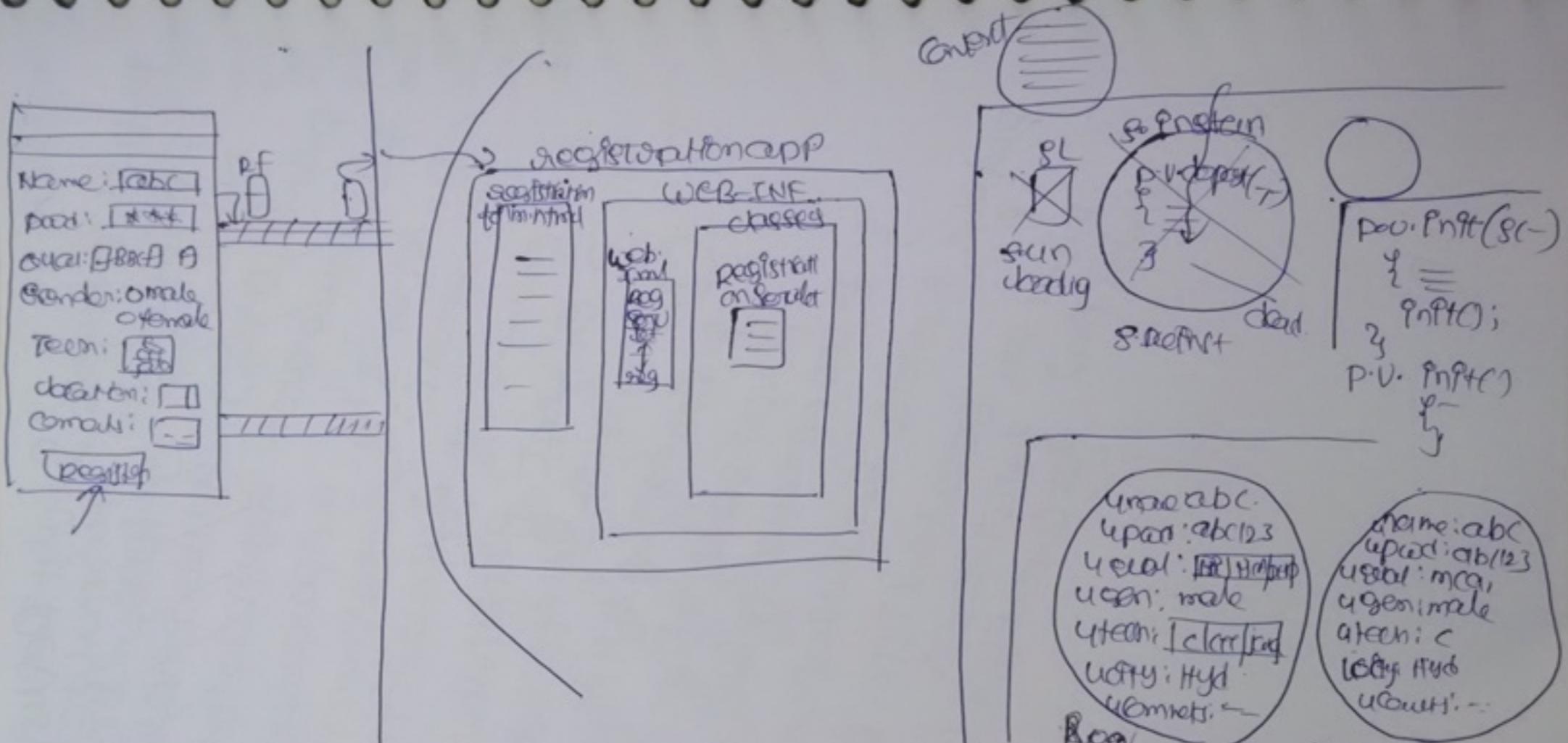
dp scenario

public private HttpWebRequest
get Headers (the new)
get HeaderNames
get HeaderValues

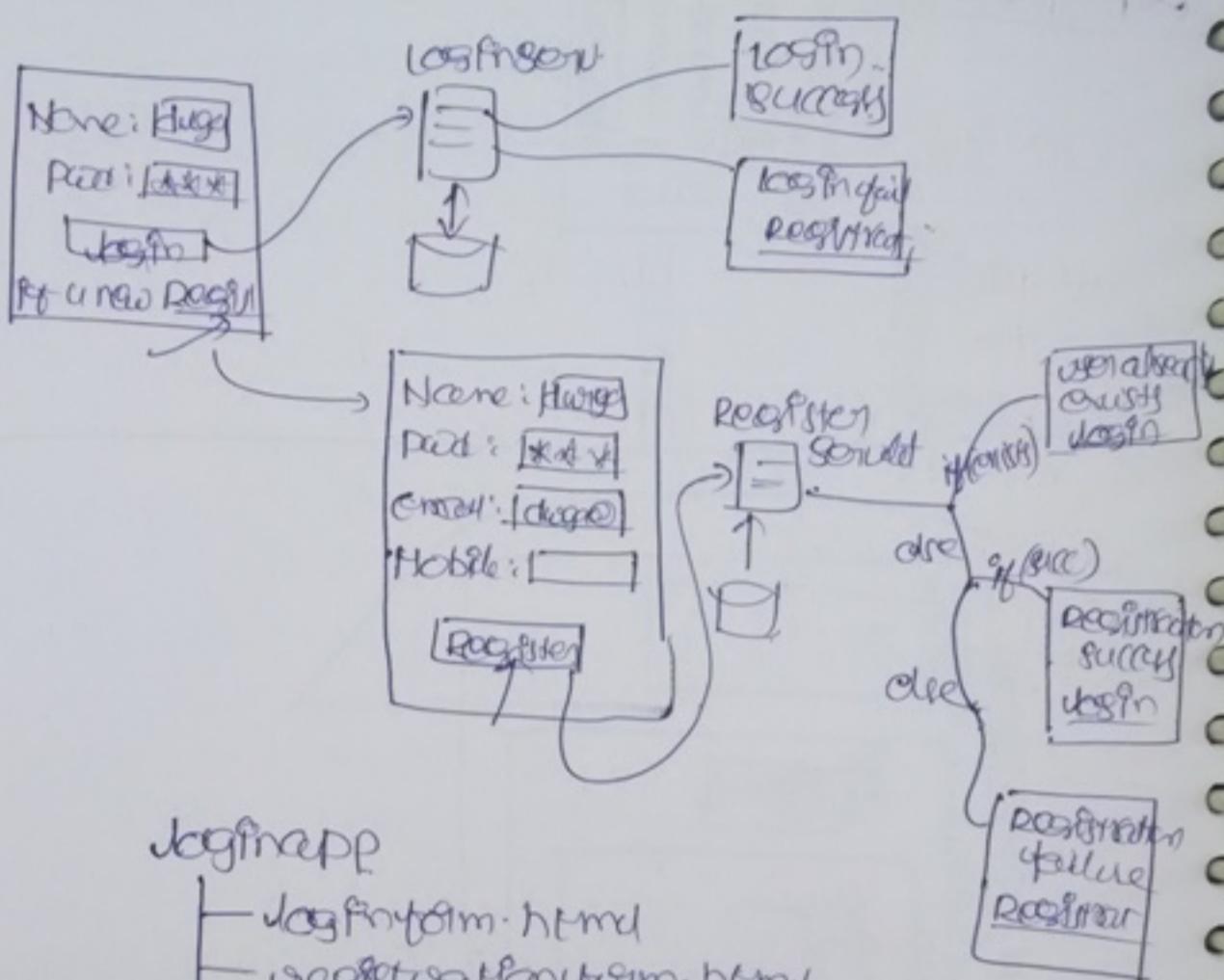
dh:
/design
html
how
why
and
the
details
and
see
what
are
the
details

~~public interface ServletRequest~~ (\rightarrow getReader(),
~~getServletContent()~~, getInputStream()); getPathInfo()
~~getHeaderNames(String name)~~
~~getHeaderValues(String name)~~
~~getHeaderNames();~~
~~getHeaderValues(String name);~~
~~getParameter(String name);~~
~~getParameterValue(String name);~~
~~getParameterNames();~~
P String
P Object
P Enumeration
P Object
P Object
P Object
P void setAttribute(String name, Object value);
P void removeAttribute(String name);
3 P void sendResponse(ServletResponse response);
public void setContentType(String type);
public PW getWriter();
public PW getContentType();

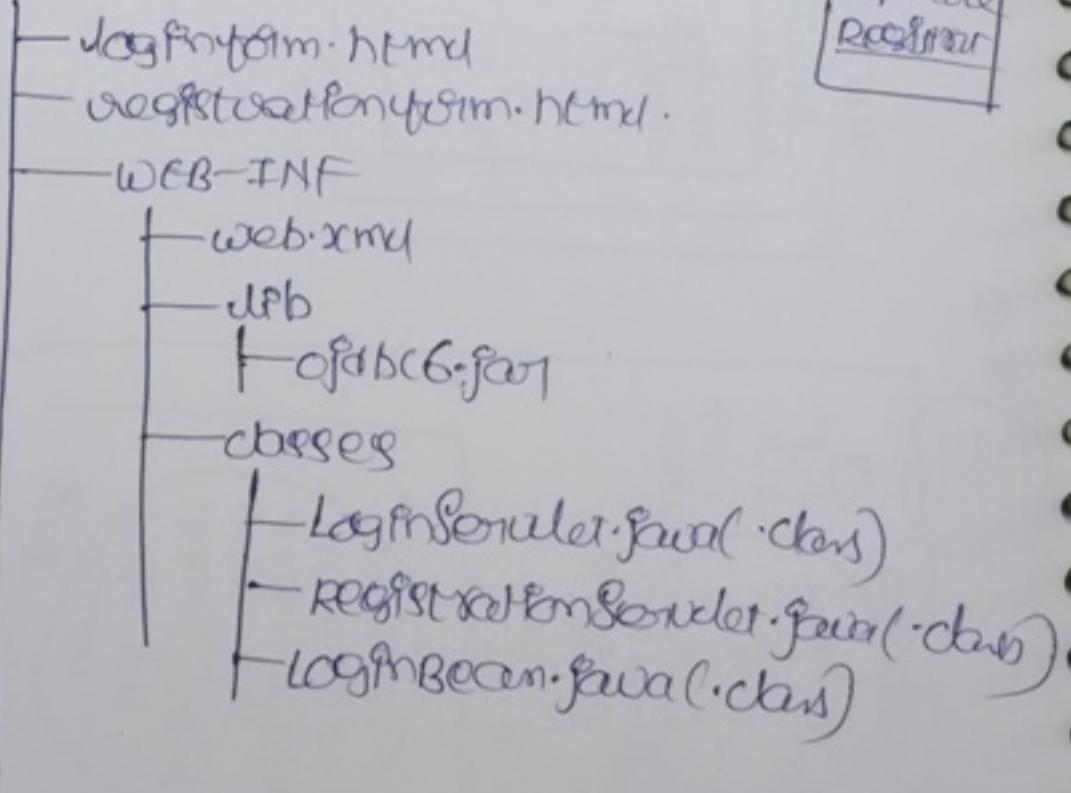
JPG to PDF Converter For Mac - Unregistered



On
DM



Loginapp



Loginform.html

```
<html>
  <head>
    <center><b><font size="7" color="red">
      Loginform
    </font>
    </b>
  </center></head>
  <br><br><br><br>
  <body bgcolor="lightpink">
    <b><font size="7">
    <form method="post" action="/loginapp/login">
      <p>
        Name <input type="text" name="uname"/>
        password <input type="password" name="upwd"/>
        <input type="submit" value="Login" />
      </p>
      If u new <a href="/loginapp/registrationform.html">
        registration.</a>
    </form>
    </body></html>
```

Registrationform.html

```
<html>
  <head>
    <center><b><font size="7" color="red">
      Registrationform
    </font>
    </b>
  </center></head>
  <br><br><br><br>
  <body bgcolor="lightpink">
    <b><font size="7">
    <form method="post" action="/loginapp/reg">
```

<pre>

Name: <input type="text" name="uname"/>
Password: <input type="password" name="pwd"/>
Email: <input type="text" name="email"/>
Mobile: <input type="text" name="mobile"/>
<input type="submit" value="Registration"/>
</pre> </form> </body> </html>

web.xml

<web-app>
< servlet>
< servlet-name>ds </servlet-name>
< servlet-class>LoginServlet </servlet-class>
</servlet>
< servlet-mapping>
< servlet-name>ds </servlet-name>
< url-pattern>/logIn </url-pattern>
</servlet-mapping>
< servlet>
< servlet-name>reg </servlet-name>
< servlet-class>RegistrationServlet </servlet-class>
</servlet>
< servlet-mapping>
< servlet-name>reg </servlet-name>
< url-pattern>/reg </url-pattern>
</servlet-mapping>
</web-app>

LoginServlet.java

```
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
public class LoginServlet extends HttpServlet
{
    public void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException
    {
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
        String uname = res.getParameter("uname");
        String upwd = res.getParameter("upwd");
        LoginBean ub = new LoginBean();
        String status = ub.CheckLogin(uname, upwd);
        out.println("<html>");
        out.println("<body background='dightyellow'>");
        out.println("<center><b>");
        out.println("<font size='7' color='red'>");
        out.println("<br><br>");
        if (status.equals("success"))
        {
            out.println("Login Success");
        }
        else
        {
            out.println("Login Failed");
        }
        out.println("</font></b></center></body>");
        out.println("</html>");
    }
}
```

RegistrationServlet.java

```
import javax.servlet.*;  
import javax.servlet.http.*;  
import java.io.*;  
public class RegistrationServlet extends HttpServlet  
{  
    public void doPost(HttpServletRequest req,  
                       HttpServletResponse res)  
        throws ServletException, IOException  
    {  
        res.setContentType("text/html");  
        PrintWriter out = res.getWriter();  
        String uname = req.getParameter("Uname");  
        String upwd = req.getParameter("Upwd");  
        String email = req.getParameter("Email");  
        String mobile = req.getParameter("Mobile");  
        LogInBean ub = new LogInBean();  
        String status = ub.registration(uname, upwd, email, mobile);  
        out.println("<html>" + status + "</html>");  
        if (status.equals("success"))  
        {  
            out.println("Registration Success");  
        }  
        if (status.equals("failure"))  
        {  
            out.println("Registration failed");  
        }  
    }  
}
```

```
if (status.equals("Exist"))  
{ out.println("User already exists");  
out.println("<html></body></html>");  
out.println("});  
}
```

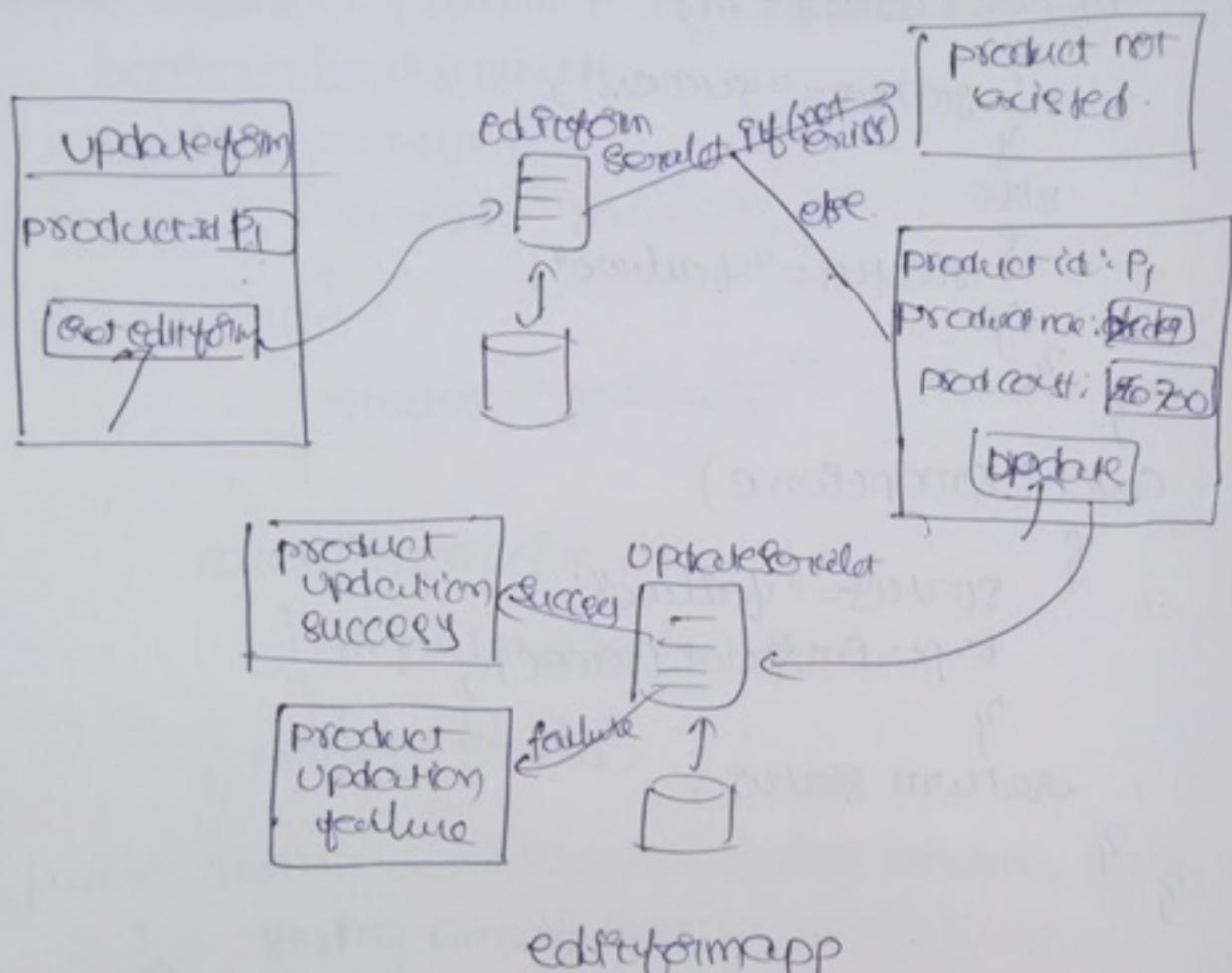
LoginBean.java

```
import java.sql.*;  
public class LoginBean  
{ Connection con;  
Statement st;  
ResultSet rs;  
String status="";  
public LoginBean()  
{ try  
{  
Class.forName("oracle.jdbc.driver.OracleDriver");  
con=DriverManager.getConnection("jdbc:oracle:  
thin:@localhost:1521:xe", "system", "durga");  
st=con.createStatement();  
}  
catch(Exception e)  
{ e.printStackTrace();  
}  
}  
public String checkLogin(String uname, String  
upwd)  
{ try  
{ rs=st.executeQuery("select * from userdetails");  
}
```

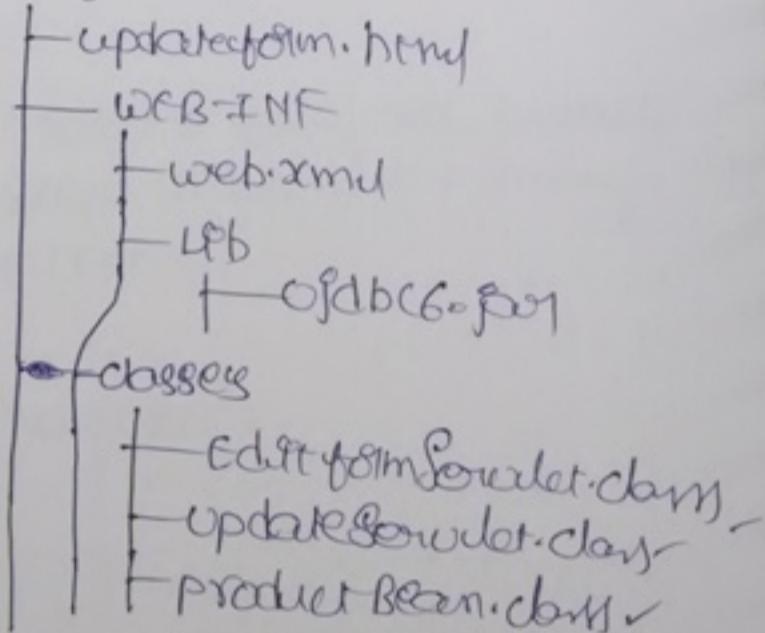
```
where uname = "' + uname + '" and updat = "' + updat + '");  
boolean b = rs.next();  
if (b == true)  
{  
    if (status == "success")  
    {  
    }  
    else  
    {  
        status = "failure";  
    }  
}  
catch (Exception e)  
{  
    e.printStackTrace();  
}  
return status;  
}  
public String registration(String uname, String updat,  
                           String email, String mobile)  
{  
    try  
    {  
        rs = st.executeQuery("Select * from reg_Detail  
                            where uname = '" + uname + "');  
        boolean b = rs.next();  
        if (b == true)  
        {  
            if (status == "existed")  
            {  
            }  
        }  
        else  
        {  
            int rowCount = st.executeUpdate("Insert INTO reg_Detail  
                                            values ('" + uname + "', '" + updat + "', '" + email + "',  
                                                '" + mobile + "')");  
        }  
    }  
}
```

```
if (zeroCount == 0)
    {
        status = "success";
    }
else
    {
        status = "failure";
    }
}

catch (Exception e)
{
    status = "failure";
    e.printStackTrace();
}
return status;
}
```



editformapp



updateform.html

Date: 18th April, 2012

```
<html>
<head>
<center><b><font size="7" color="red">
product update form
</font><b></b></center>
</head>
<body><br><br><br><br>
<body background="lightyellow">
<b><font size="7">
<form method="get" action=".//edit1">
<p>
product id <input type="text" name="pid"/>
<input type="submit" value="getEdtForm"/>
</p></form></font></b></body></html>
```

web.xml

```
<webapp>
< servlet >
< servlet-name > edit < /servlet-name >
< servlet-class > EditFormServlet < /servlet-
class >
< /servlet >
< servlet-mapping >
< servlet-name > edit < /servlet-name >
< url-pattern > /edit < /url-pattern >
< /servlet-mapping >
```

(Comps-D)
By
note
Dad
for
Visal

```
< servlet >
< servlet-name > ers </ servlet-name >
< servlet-class > updateSearlet </ servlet-class >
</ servlet >
< servlet-mapping >
< servlet-name > us </ servlet-name >
< url-pattern > /update </ url-pattern >
</ servlet-mapping >
</ webapp >
```

EditFormSearlet.java

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

public class EditFormSearlet extends HttpServlet
{
    public void doGet(HttpServletRequest req,
                      HttpServletResponse res) throws ServletException,
                      IOException, SQLException, IOException
    {
        try
        {
            res.setContentType("text/html");
            PrintWriter out = res.getWriter();
            String pid = req.getParameter("pid");
            ProductBean pb = new ProductBean();
            ResultSet rs = pb.getproduct(pid);
            boolean b = rs.next();
```

```
if (b == true)
{
    out.println("<html>");
    out.println ("<head>");
    out.println ("<center><b><font style='font-size: 2em;
color: #red;'>");
    out.println ("producteditform");
    out.println ("</font></b></center>");
    out.println ("<br><br><br><br><br>");
    out.println ("<body background='lightyellow'>");
    out.println ("<b><font style='font-size: 1.5em;'>");
    out.println ("<form method='get' action='!/update'");
    out.println ("product pd $' + oe.getStr(1));
    out.println("<input type='hidden' name='pd'
value='!/ + pd + !!!' />\"");
    out.println("product Name <input type='text'
name='pname'
value='!!' + oe.getStr(2) + !!' />\"");
    out.println(");
    out.println("product Cost <input type='text'
name='pcost' value='!!' + oe.getInt(3) + !!'
out.println(); />\"");
    out.println(")<input type='submit' value='Update' />\"");
    ("</pre></form></body></html>");
    ("</body></html>");
```

{}

else

{

out.println("<html>");

out.println("<body bgcolor='lightyellow'>");

out.println("<center>");

</center></body></html>");

out.println("Product not existed");

out.println("</center>");

out.println("</body></html>");

}

catch (Exception e)

{

e.printStackTrace();

}

}

}

explain

updateServlet.java

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

public class updateServlet extends HttpServlet
{
    public void doGet(HttpServletRequest req,
                      HttpServletResponse res) throws ServletException,
                                                    IOException
    {
        try
        {
            res.setContentType("text/html");
            PrintWriter out = res.getWriter();
            String pid = req.getParameter("pid");
            String pname = req.getParameter("pname");
            int pcost = Integer.parseInt(req.getParameter(
                "pcost"));
            productBean pb = new productBean();
            String status = pb.update(pid, pname, pcost);
            out.println("<html>" +
                       "<body style='background-color:#ffffcc'>" +
                       "<center><table border='1'>" +
                       "<tr><td>Product ID:</td><td>" + pid +
                       "</td></tr>" +
                       "<tr><td>Product Name:</td><td>" + pname +
                       "</td></tr>" +
                       "<tr><td>Product Cost:</td><td>" + pcost +
                       "</td></tr>" +
                       "<tr><td colspan='2'>Status:</td></tr>" +
                       "<tr><td colspan='2'>" + status +
                       "</td></tr>" +
                       "</table></center>" +
                       "</body></html>");
        }
        catch (Exception e)
        {
            out.println("Error occurred while updating product");
        }
    }
}
```

we
Get
With
it
at
well
of
by
try
try
call
for
)

```
if (states.equals("failure"))  
    {  
        out.println("product update failed");  
        }  
    }  
    catch (Exception e)  
    {  
        e.printStackTrace();  
        }  
    }  
}  
  
explain
```

productBean.java

```
import java.sql.*;  
public class productBean  
{ Connection con;  
 Statement st;  
 ResultSet rs;  
 String status="";
```

```
public productBean() → constructor
```

```
{ try  
{  
 Class.forName("oracle.jdbc.driver.Oracle  
Driver");  
 con=DriverManager.  
 getConnection("jdbc:oracle:thin:@localhost:  
 1521:xe", "System", "duga");  
 st=con.createStatement();  
 }  
 catch(Exception e)  
 {  
 e.printStackTrace();  
 }
```

```
public ResultSet getproduct (String pid)
```

```
{ try  
{  
 rs=st.executeQuery("select * from  
 product  
 where pid ='" + pid + "'");  
 }  
 }
```

```
catch( Exception e)
```

```
{ e.printStackTrace(); }
```

```
return $;
```

```
}
```

```
public String update(String pid, String name,
```

```
{ try
```

```
{
```

```
int count = st.executeUpdate("update product
```

```
set name ='" + name + "'", pCost = " + pCost + "
```

```
where pid ='" + pid + "'");
```

```
If (count == 1)
```

```
{ status = "success"; }
```

```
else
```

```
{ status = "failed"; }
```

```
catch( Exception e)
```

```
{ status = "failed"; }
```

```
e.printStackTrace();
```

```
} return status;
```

```
}
```

Wk 10
+ 11

not
free,
not
to
date
will
after
the
prod
is
out
of
stock
or
not
available
at
server
we
have
already
placed
order

C:\> set classpath=%classpath%;C:\oracle---
10g\jre6\jar;

two
java
sql
jdb
at
clay
park

C:\> Tomcat701----\webinf\classes
fauc * forward

—config
bc
—scroll
—api

—Now start the server/
http://localhost:8080/
ManagerApp/
update

check whether the database is existed or not
SQL> select * from product;

PId	PName	PCost
P1	aaa	600.
P2	bbb	700
P3	ccc	800
P4	ddd	900

execution
(scenario)

HTTP
SQL
posting

Welcome-file: In general in web applications some html files or jsp files and so on are the first pages you'll each and every user upto now to access the respective web application we have to specify the respective loginform.html index.jsp and so on - first pages in the url's as a response name.

- once the above specified pages are first pages in web applications then we are expecting to execute the first pages by the container automatically when we access the respective web application.
- To achieve the above requirement we have to declare the respective file as welcome file.
- To make a jsp file or html file as welcome file then we have to configure it in web.xml file as welcome file. For this we have to use the following xml tags in web.xml file.

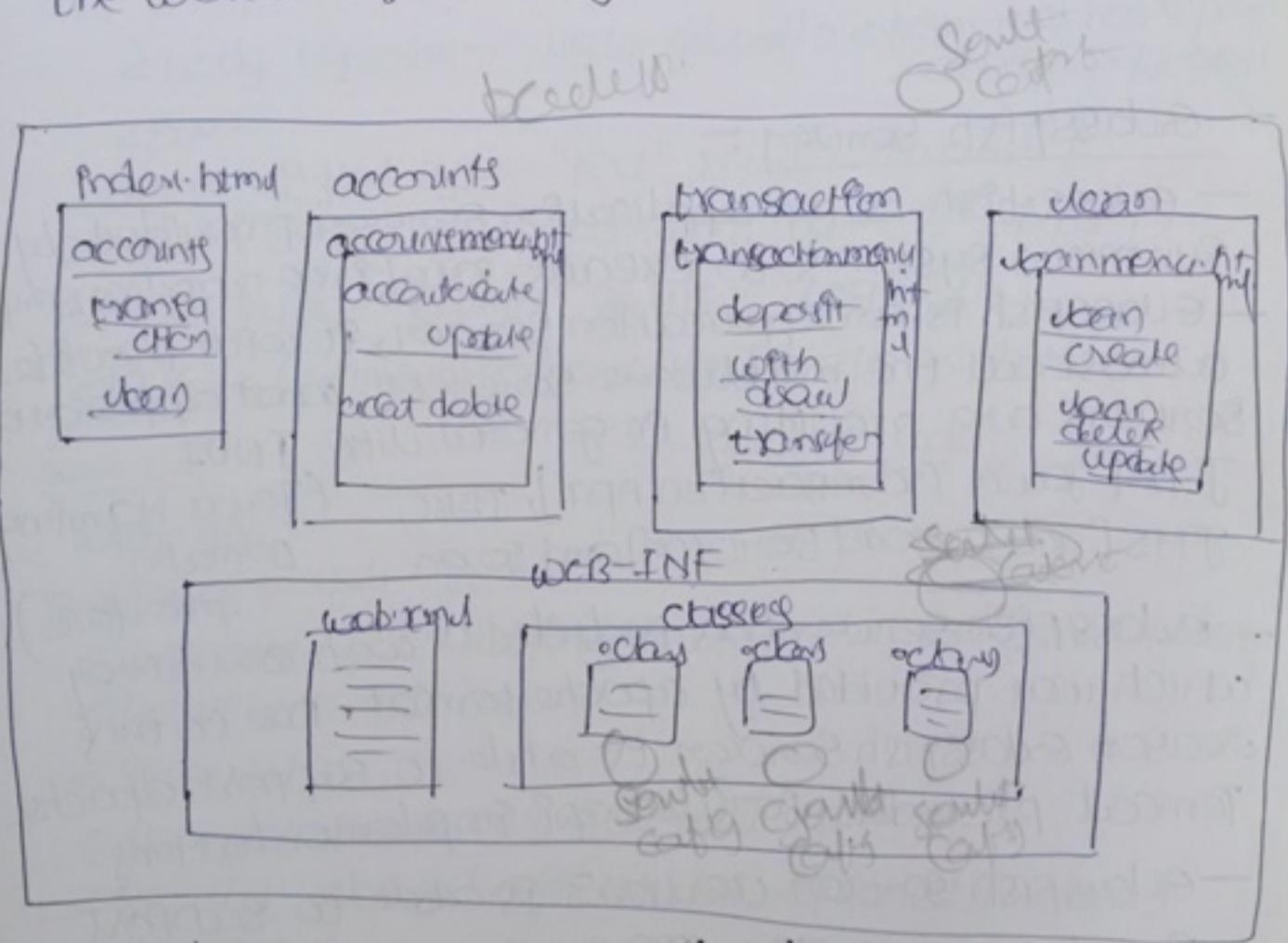
<web-app>

```
=  
<welcome-file-list>  
  <welcome-file>file1</welcome-file>  
  <welcome-file>file2</welcome-file>  
=  
</welcome-file-list>  
=  
</web-app>
```

As per the above welcome-files declaration, It is possible to provide more than one welcome file in a single web application but with respect to multiple number of modules.

Date: 19th April 2012

In the below context if we access a particular module then container will search for the welcome file in the module respective folder as per the order in which we provided by the welcome files configuration in web.xml file.



<web-app>

<welcome-file-list>

<welcome-files> index.html

transactionmenu.htm

accountsmenu.htm

loanmenu.htm

JPG to PDF Converter For Mac - Unregis

Welcome-file-UI.jsp

<web-app>

- `http://192.168.1.100:8080/backup/index.html`
- `http://192.168.1.100:8080/backup/accounts/index.html` transactionmenu.html ✓

Glassfish Server :

- Glassfish is an application server provided by Sun microsystems to execute java/free applications.
- Glassfish is an application server, It will provide almost all the middleware services what application servers are providing on general like JNDI, JTA [Java Transaction API], JDBC, JMS [Java Mail Service] and so on (Java Naming Domain Interface)
- Glassfish server has included a web container which was provided by apache tomcat. Due to this reason Glassfish server is able to support apache Tomcat provided servlet-api implementation.
- Glassfish server version 3 is able to support servlet 3.0 version, JSP 2.0 versions and so on.
- Glassfish server is able to support for annotations processing in Servlets application.
- To execute web applications with Glassfish server we have to use the following steps

Step①: prepare web application and its war file
in your work directory

exploded

loginform.html

```
<html><head><center><b><font style="font-size: 14pt; color: red">
    Login Form
</font></b></center></head>
<br><br><b><font style="font-size: 14pt; color: red">
    <form method="post" action="/login">
        Name <input type="text" name="username"/>
        password <input type="password" name="password"/>
        <input type="submit" value="Login"/>
    </form></body></center></b></html>
```

web.xml

↳ save this under

```
<web-app>
    <welcome-file-list>
        <welcome-files>loginform.html</welcome-files>
    </welcome-file-list>
    <servlet>
        <servlet-name>LogInServlet</servlet-name>
        <servlet-class>LogInServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>LogInServlet</servlet-name>
        <url-pattern>/logIn</url-pattern>
    </servlet-mapping>
</web-app>
```

LoginServlet.java

```
import javax.servlet.*;  
import javax.servlet.http.*;  
import java.io.*;  
public class LoginServlet extends HttpServlet  
{  
    public void doPost(HttpServletRequest req,  
                        HttpServletResponse res) throws ServletException,  
                                         IOException  
    {  
        res.setContentType("text/html");  
        PrintWriter out = res.getWriter();  
        String uname = req.getParameter("uname");  
        String upwd = req.getParameter("upwd");  
        out.println("<html>");  
        out.println("  <body bgcolor='red'>");  
        out.println("    <center><br> cent size=40 color=");  
        if(uname.equals("durga"))  
        {  
            if(upwd.equals("durga"))  
            {  
                out.println("      'yellow'>");  
                out.println("      login success");  
            }  
            else  
            {  
                out.println("      login failed");  
            }  
        }  
        out.println("    </center>");  
        out.println("  </body>");  
        out.println("</html>");  
    }  
}
```

↳ save this under ctesey

— To compile the above servlet, we have to set the class path environment variable either two
1. glassfish server provided servlet implementation
2. Tomcat provided servlet-api implementation
(servlet-api.jar)

— glassfish server has provided servlet-api implementation in the form of servlet-api.jar file at the following location:

C:\glassfish\1.3\glassfish\modules\servlet-api.jar

☞ set classpath=C:\glassfish\1.3\glassfish\1\modules\servlet-api.jar;

→ D:\app1\testapp1\WEB-INF\classes>javac *.java
↳ not in server environment.

D:\app1\testapp1\WEB-INF\classes>cd..

D:\app1\testapp1\WEB-INF>cd..

D:\app1\testapp1\WEB-INF\classes>javac *.java *

↳ war file creation completed

Step②: Start application server glassfish.

To start the glassfish server, we have to execute startserv.bat file provided at

C:\glassfish\1.3\glassfish\bin

Double click on the startserv.bat file

Step③ How to open server console

To open administration console, open web browser and use the following url.

<http://localhost:4848>

— If we use the above web administration console login page will be open, where we have to provide user name and password.

username: admin
password: admin
<input checked="" type="checkbox"/>

Step④: upload web application war file to glassfish server.

If we click on login button in administration console login page [Step no③] then glassfish server home page will be open, where we have to go for [Admin.jsf] deployment section in the middle frame of glassfish server.

— Click on List deployed applications either Deploy an application

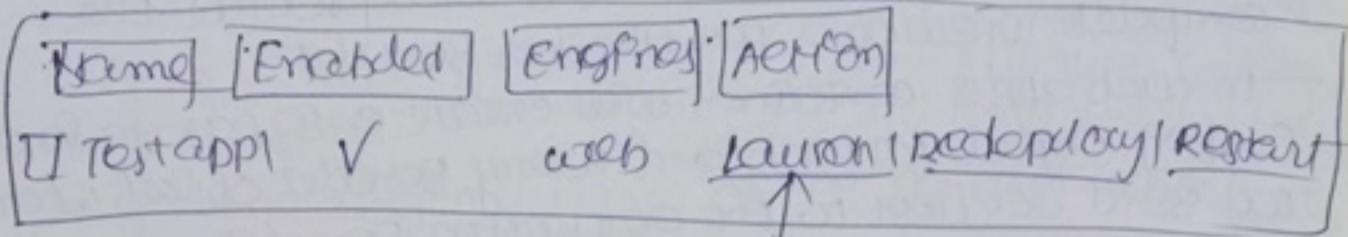
If we do the above (List Deployed Applications) then all the deployed web applications list will be available.

— To deploy a new web application click on Deploy button.

— If we click on Deploy button a page will be open to upload war file where click on Browse button

Location: C packaged file to be uploaded to the server

- In the above click browse button and select application war file. Then click on **OK** button on right side top corner.
- If we click on **Load** button, our web application war file will be uploaded to the server and listed in the displayed applications.



Step 6: Access the web application.

- To access the web application, we have to click on launch hyperlink in Action part or to the deployed web application.
- 2) if we click on Launch hyperlink automatically our application will be executed and open a http:// welcome file in a separate browser.

ServletConfig

Date: 21st April, 2012

- ① It is an object, it will provide all the configuration details of particular servlet like default name of the servlet, initialization parameters and so on.
- ② ServletConfig is an object, it will provide the complete view of a particular servlet.
- ③ In web app's container will create a separate ServletConfig object for each and every servlet object when we send request to the respective servlet.
- ④ If we keep any data inside ServletConfig object then that data will be shared up to the respective servlet due to this reason the scope of the ServletConfig object is upto the respective servlet object.
- ⑤ In web app's container will prepare ServletConfig object immediately after the servlet instantiation and just before calling init() in servlet initialization.
 - In web app, container will destroy ServletConfig object just before performing servlet deinstantiation.
 - Due to the above reason the life of the ServletConfig object is the life of the respective servlet object.
- In web app's there are 2 ways to get ServletConfig
 - 1. To override init() in the respective object
 - or class MyServlet extends HttpServlet
 - { ServletConfig config;
 - public void init(ServletConfig config) throws
 - { this.config = config; ServletException

3
—

Q. use getServletConfig() directly from Servlet
public class MyServlet extends HttpServlet {
 public void doGet(HttpServletRequest request, HttpServletResponse response)
 throws ServletException, IOException {
 ServletConfig config = getServletConfig();

To get the logical name of the servlet which we defined in web.xml then we have to use the following method from ServletConfig.

public String getServletName()

If we want to keep initialization parameters in servletConfig Object first we declare them in the respective web.xml under the respective Servlet Configuration for this we have to use the following XML tag in web.xml →

```
<web-app>  
  <servlet>  
    <init-param>  
      <param-name> Name </param-name>  
      <param-value> Value </param-value>  
    </init-param>  
  </servlet>  
</web-app>
```

logical view
initialization
parameters
servlet config

options
- setAttribute
- getAttribute
- createObject
- destroyObject
- getServletInitParameter
- getServletInitParameter
- creator of
req resp
object

- Request processing
 (access service)
- Servlet Delegation
 (call destroy()
method)
- Servlet Unloading

- When we send a request from client to specific Servlet then Container will pickup all the initialization parameters from web.xml at the time of getting mapping b/w url pattern and the servlet and stored onto Servlet Config at the time of creation as Initialization parameter.
 - To get a particular initialization parameter value we need to use the following method from Servlet Config.
`public String getInitParameter(String name);`
 - To get all the names of the initialization parameters from Servlet Config object then we have to use the following method
`public Enumeration getInitParameterNames();`
- ~~Ques~~ → What is the need to declare Initialization parameters for servicing?
- What are the different parameters?
Initialization parameters
Content parameters
Request headers
Request attributes

public interface ServletConfig

{ getServletName()

public String getServletName();

public String getParameterName(int n);

public Enumeration getParameterNames();

public byte[] getServletContent()

→ observe

Servlet { getServletInfo()
getServletConfig()
getServletName()

Servlet
Config

upto now

web-app

<welcome-file-list>
= welcome-file <welcome-file>
<welcome-file> index.html <welcome-file>

<welcome-file-list>
<welcome-file>

<context-param> <context-param>
<param-name> url <param-name>
<param-value> file:///C:/ <param-value>

<param-name>

<param-value>

<param-name> - <param-name>
<param-value> - <param-value>

<param-name> - <param-name>
<param-value> - <param-value>

```
<server-name> jatin </server-name>
<server-class> konferalit </server-
class>
</server>
<server-mapping>
  <server-name> tari </server-name>
  <server-class> /tarin </server-
class>
  <url-pattern>
    <url-pattern>
  </url-pattern>
</server-mapping>
</server>
<server>
  <location-startup> </location-startup>
  <full-param>
    <param-names> </param-names>
    <param-values> </param-values>
    </full-param>
  <full-param>
    <param-names> </param-names>
    <param-values> </param-values>
    </full-param>
  </full-param>
</server>
<server-mapping>
  <server-name> jatin </server-name>
  <server-class> konferalit </server-
class>
</server-mapping>
<server-mapping>
  <server-name> tari </server-name>
  <url-pattern> /tari </url-pattern>
</server-mapping>
```

example program

web.xml

```
<web-app>
< servlet >
    < init-param >
        < param-name > driverclass </param-name >
        < param-value > oracle.jdbc.driver.OracleDriver </param-value >
    < /init-param >
    < init-param >
        < param-name > url </param-name >
        < param-value > jdbc:oracle:thin:@localhost:1521:xe </param-value >
    < /init-param >
    < init-param >
        < param-name > user </param-name >
        < param-value > system </param-value >
    < /init-param >
    < init-param >
        < param-name > password </param-name >
        < param-value > durga </param-value >
    < /init-param >
    < servlet-name > ms </servlet-name >
    < servlet-class > MyServlet </servlet-class >
< /servlet >
< servlet-mapping >
    < servlet-name > ms </servlet-name >
    < url-pattern > /config </url-pattern >
< /servlet-mapping >
< /web-app >
```

Configapp

—WEB-INF
|—web.xml
|—classes
|—MyServlet.class

MyServlet.java

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

public class MyServlet extends HttpServlet
{
    public void doGet(HttpServletRequest req,
                      HttpServletResponse res) throws ServletException,
                      IOException
    {
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
        ServletConfig config = getServletConfig();
        String driverClass = config.getInitParameter("driverClass");
        String url = config.getInitParameter("url");
        String user = config.getInitParameter("user");
        String password = config.getInitParameter("password");
        java.util.Enumeration e = config.getInitParameterNames();
        out.println("<html>");
        out.println("  <b>font size='6'</b>");
        out.println("  <br><br> " + driverClass);
        out.println("  <br><br> " + url);
        out.println("  <br><br> " + user + " ->" + url);
        out.println("  <br><br> " + password);
        out.println("  <br><br> ");
    }
}
```

while (e.hasMoreElements())

```
{ out.println("out.nextElement());");
    out.println("<br><br>");
```

```
out.println("</pre></b></body></html>");
```

}

→ if good, we have all
the methods too and ready
to serve metrics

ditto
getsourceCode
new
FUD

→ To get ~~scout~~ → method
include

acceptHeaders → method
include

acceptAttributes → method
include

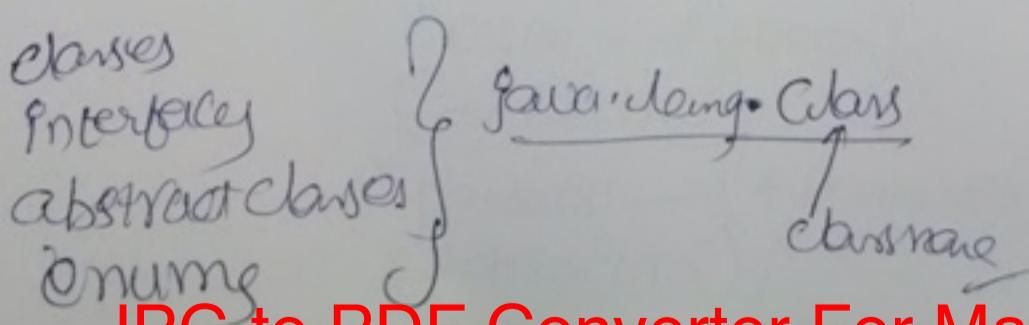
visualisationParameters → method
include

start
end
group

Any technology Reflection API

- Reflection API is a java api, it can be used to analyse all the capabilities of the programming constructs which we used in our java applications.
- As part of the application development we will provide number of instructions to represent all the real world entities, and their properties and behaviours. But reflection API has provided very good environment to analyse the programming constructs like classes, variables, methods, constructors, package and so on [constructors, interfaces, modifiers, etc] which we used to represent the real world entities. Due to the above reason, we will utilize the reflection api in products, tools, frameworks and soon, instead of application development.
- java technology has provided the complete object api in the form of the following two packages.
 - 1. java.lang package
 - 2. java.lang.reflect package.

- As part of product development to analyse all the java programming constructs first we need to represent them in our technology. For this reflection api has provided a separate class for each and every programming construct.



Tools
↓
Java, Javac, Javadoc, Javap.

method — java.lang.reflect.Method

variable — java.lang.reflect.Field

constructor — java.lang.reflect.Constructor

modifiers — java.lang.reflect.Modifiers

packages — java.lang.reflect.package

datatype — java.lang.reflect.Type

class
name

2) The main purpose of java.lang.Class is to store the analysis of a particular root class.

① Using forName factory method :- To create a particular class "Abc" object then if we use this approach then we have to use the following method from class "Class".

public static Class forName(String className)
throws ClassNotFoundException

2) obj Class c = Class.forName("Account");

To create then we have to use the following method from

Public class getAccount()

"Object" class

Obj Account a = new Account();
class c = a.getAccount();

object
(getAccount)

Account

③ Using class filename :- In Java technology all the class files will represents a class "Account" object

Object c = Account.class

Here
class
means
not the
Account.
?
?
class
object
↑
here we
created the
object for
the class
which every
the class
Account

→ To get the name of the class, the following method from class "class".

[public String getName()]

→ To get the modifiers list which we declare to the respective class, we will use the following method

[public int getModifiers()]

Note: To display the names of the modifiers, we need to pass an integer value to toString() method of Modifier class (in java.lang.reflect package) returned by getModifiers() method.

Print p = c.getModifiers();

System.out.println("modifier"+ Modifier.toString(p));

→ To get

Super class of the respective class, we have to use the following method from class "class".

[public class getSuperClass()]

→ To get all the interfaces list which are implemented in the respective class we will use the following method.

[public class[] getInterfaces()]

Ques 1:

```
import java.lang.*;  
import java.io.*;  
interface BankReles  
{  
    void withdraw();  
    void deposit();  
}  
  
class Account implements BankReles  
{  
    String name;  
    double balance;  
    public void withdraw(double amount)  
    {  
        if(amount > balance)  
            System.out.println("Insufficient Balance");  
        else  
            balance -= amount;  
    }  
    public void deposit(double amount)  
    {  
        balance += amount;  
    }  
    public String getName()  
    {  
        return name;  
    }  
    public void setName(String name)  
    {  
        this.name = name;  
    }  
    public double getBalance()  
    {  
        return balance;  
    }  
}  
  
final class SavingsAccount extends Account  
{  
    public void withdraw(double amount)  
    {  
        if(amount > balance)  
            System.out.println("Insufficient Balance");  
        else if(amount > 500)  
            System.out.println("Withdrawal amount must be less than or equal to 500");  
        else  
            balance -= amount;  
    }  
    public void deposit(double amount)  
    {  
        if(amount < 100)  
            System.out.println("Deposit amount must be greater than or equal to 100");  
        else  
            balance += amount;  
    }  
    public void withdraw(double amount, String reason)  
    {  
        if(amount > balance)  
            System.out.println("Insufficient Balance");  
        else if(amount > 500)  
            System.out.println("Withdrawal amount must be less than or equal to 500");  
        else if(reason.equals("Salaried"))  
            System.out.println("Withdrawal amount must be less than or equal to 500");  
        else  
            balance -= amount;  
    }  
    public void deposit(double amount, String reason)  
    {  
        if(amount < 100)  
            System.out.println("Deposit amount must be greater than or equal to 100");  
        else if(reason.equals("Salaried"))  
            System.out.println("Deposit amount must be greater than or equal to 100");  
        else  
            balance += amount;  
    }  
}  
  
class ClientTest  
{  
    public static void main(String[] args) throws IOException  
    {  
        Account a = new SavingsAccount();  
        System.out.println("Account Name - " + a.getName());  
        System.out.println("Modifiers - " + Modifier.toString(a));  
        System.out.println("Super class - " + a.getSuperclass());  
  
        Class[] interfaces = a.getInterfaces();  
        System.out.println("Interfaces - !.");  
  
        for (Class cl : interfaces)  
        {  
            System.out.println(cl.getName() + "");  
        }  
    }  
}
```

Note: In Java technology, we are able to declare all the annotations at various levels like class level, method level, variable level, constructor level and soon.

To get the class level annotations, we will use the following method

```
public Annotation[] getAnnotations()
```

Field:

The main purpose of Field class is to store the analysis of a particular field declared in the respective class.

In reflection applications if we want to get field class object first we need to prepare class "cls" Object.
To get field objects with respect to the properties declared in the respective class, we need to use the following methods from class "cls".

```
public Field[] getFields()
```

```
public Field[] getDeclaredFields()
```

where getFields() method can be used to get all the public fields which are declared in the both Super class and Sub class.

where getDeclaredFields() method can be used to get all the fields which are available in the respective class.

To get the name of the field, we will use the following method from Field class.

```
public String getName()
```

- To get modifiers list which we provided for the respective variable we need to use the following method
`[public int getModifiers()]`
- To get the data type of the respective variable, we will use the following method
`[public Type getType()]`
- To get the respective field value, we have to use the following methods
`[public XXX getField()]`

2

`[public Object getField()]`

Note:

To get all the annotations which are declared by the respective field we have to use the following method

`[public Annotation[] getAnnotations()]`

Ques.
Ans

import java.lang.reflect.*;

class A

```
{ public int p=10;
  float f = 22.22f;
}
```

class B extends A

```
{ public static final long l=100;
  static double d=33.3;
}
```

```
class FieldTest
{
    public static void main(String[] args)
    {
        class C = B.class;
        Field[] farr = C.getDeclaredFields();
        for(Field f : farr)
        {
            System.out.println("Name--" + f.getName());
            System.out.println("Modifiers--" + f.getModifiers());
            System.out.println("Data type--" + f.getType());
            System.out.println("Value---" + f.get(f));
        }
    }
}
```

methodology-constructor:— In reflection applications java.lang.reflect.Method can be used to store a particular method analysis part.

— In reflection applications java.lang.reflect can be used to store the analysis part Constructor of a particular constructor.

— In reflection applications, all the methods analysis part first we need to if we want to create class "class" object. After representing class analysis part to get all the methods in the form of method class objects we will use the following

public Method[] getMethods()

methods

public Method[] getDeclaredMethods()

— where getMethods() method can be used to return all the public methods which are declared in one ^{super} _{and sub} class; where getDeclaredMethods() can be used to get all the methods which are declared in the present class.

— To get name of the method we have to use the following method from Method class.

public String getName()

— To get all the modifiers which we declared in the respective method we will use the following method

public int getModifiers()

To get the return type which we specified in the respective method prototype we will use the following method

`public Type getReturnType()`

2 To get all the parameter types which we declared in the method prototypes we will use the following method

`public class[] getParameterTypes()`

To get all the exceptions which we specified along with throws keyword in the respective method.

We have to use the following prototype method

`public class[] getExceptionTypes()`

Note: To get all the method level annotations which are declared at the respective method then we have to use the following method.

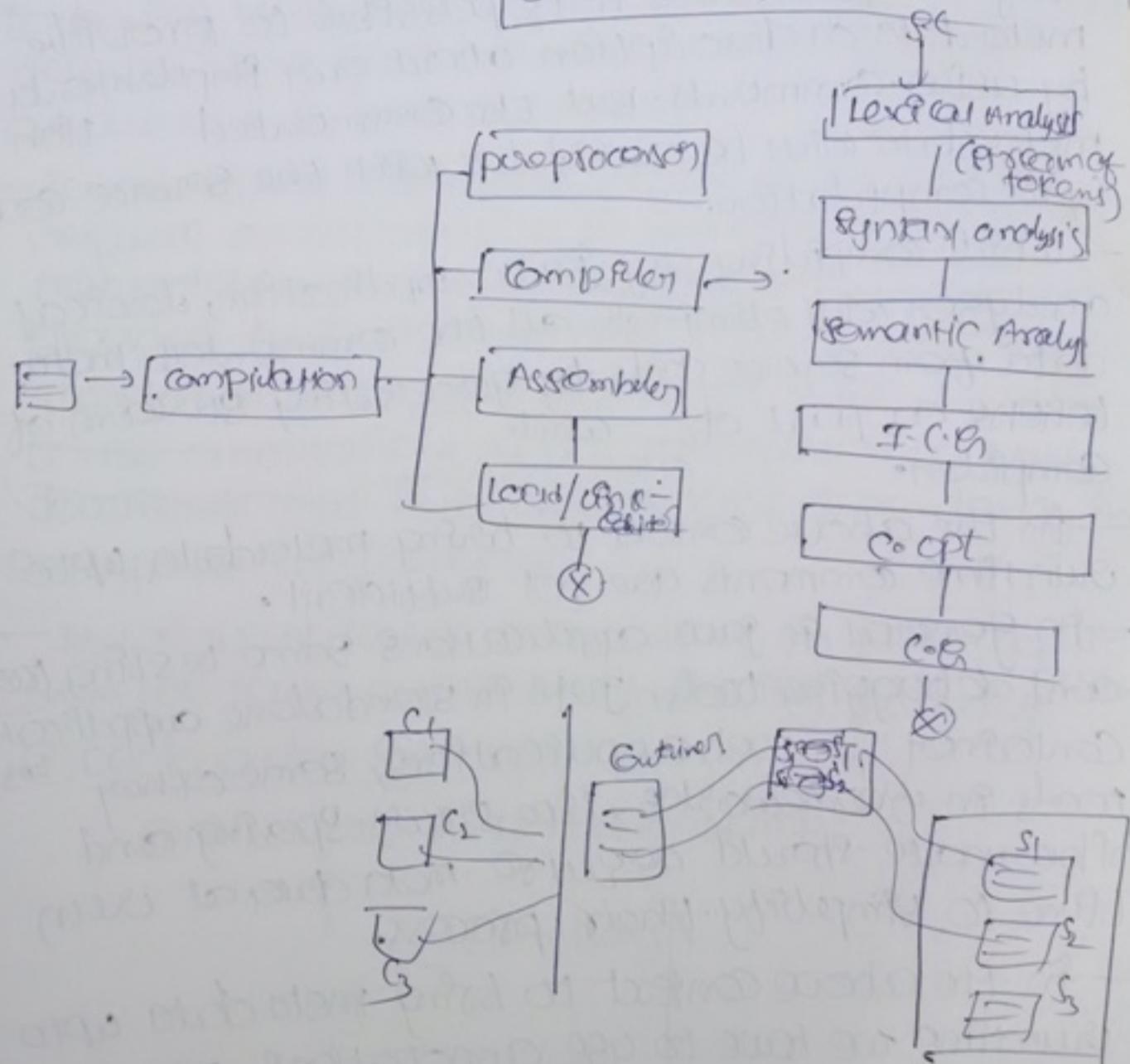
`public Annotation[] getAnnotations()`

```
BS  
PMT import java.lang.reflect.*;  
import java.io.*;  
class A  
{  
    public final static void m1 (int b, float t)  
        throws IOException, ArithmeticException  
    {  
    }  
    static int m2 (char c, long d) throws  
        NullPointerException  
    {  
        return 10;  
    }  
}
```

```
class MethodTest
{
    public static void main(String[] args)
    {
        Class c = A.class;
        Method[] m = c.getDeclaredMethods();
        for(Method mi : m)
        {
            System.out.println("Method name-->" + mi.getName());
            int p = mi.getModifiers();
            System.out.println("Modifiers-->" + Modifier.toString(p));
            System.out.println("Return type-->" + mi.getReturnType());
            Class[] clss = mi.getParameterTypes();
            System.out.println("parameter types-->");
            for(Class cl : clss)
            {
                System.out.println(cl.getName() + " ");
            }
            System.out.println();
            Class[] clss1 = mi.getExceptionTypes();
            System.out.println("Exceptions-->.");
            for(Class cl : clss1)
            {
                System.out.println(cl.getName() + " ");
            }
            System.out.println();
        }
    }
}
```

Notes To represent a particular constructor analysis part we have to use java.lang.reflect.Constructor class, which includes all the methods same as Method class methods, except getReturnType method. That means getReturnType method is not available in Constructor class.

Date: 28th April 2012



Annotation: Annotation is a feature you can use to add notes, highlights, and other annotations to your PDF files. It's to share your thoughts and ideas with others.

Annotations are typically added to PDF files using the following methods:

- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.

Annotations are typically added to PDF files using the following methods:

- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.
- For the above content to bring meta-data up to date.

- In general in enterprise applications, it is possible to provide some metadata at runtime in the form of XML documents like web.xml, struts-config.xml, hibernate mapping files (.hbm.xml) and so on.
- If we use XML document as an alternative to burying metadata at runtime then we should require very good knowledge or very good awareness on XML technology.
- To overcome the above problem, we should require an alternative in the form of Java technology, where the required alternative was provided by Java technology. That is annotation. Therefore in enterprise applications we are able to use annotations as an alternative to XML documents. That is to reduce XML dependency in enterprise applications.
- If we want to use annotations in Java applic then we have to use the following syntax:
 - ① Declaration Syntax:

② Interface Annotation Name
{
 ≡ Members
}

④ Utilization Syntax:

@AnnotationName([parameter-list])

In Java applications, to process the annotation we should require a separate tool. That is annotation processing tool (ATP).

On the basis of the above syntax, there are 3 types of annotations.

- ① Marker Annotation: If we declare any annotation without the members then that annotation is called as marker annotation.

ex @Entity

```
public class Employee  
{  
}
```

- ② single value Annotation: If we declare any annotation with exactly one member then that annotation is called as single valued Annotation.

```
public class Employee  
{  
    @Column(name = "eno")  
    private String eno;  
}
```

- ③ Multivalued Annotation: If we declare any annotation with multiple no. of members then that annotation is called as Multivalued Annotation.

```
@WebServlet  
(name = "myServlet", urlPatterns = {"/first/second"},  
loadOnStartup = 1)  
public class MyServlet extends HttpServlet  
{  
}
```

— Java technology has provided complete annotation support in the form of two packages

1. java.lang

2. java.lang.annotation

— In Java technology, all the annotations are interfaces; the common super type for all the annotations interface is java.lang.annotation.Annotation.

few
dele
they
are
int
fles

— Java technology has provided the following two types of annotations.

① predefined annotation

② userdefined annotations

predefined annotations :- predefined annotations are the annotations defined by the Java technology. predefined annotations can be divided into the following two types.

① general purpose annotation :-

- ① @Override annotation
- ② @Deprecated annotation
- ③ @SuppressWarnings

- ④ Meta annotation → which describe the metadata about the annotation.
- ① @Inherited
 - ② @Documented
 - ③ @Retention
 - ④ @Target
- That means an annotation which describes another annotation.

→ Java technology has provided all the general purpose annotations in `java.lang` package. Java technology has provided all the meta annotations in `java.lang.annotation` package where general purpose annotations are simple annotations. They will be utilized frequently in Java applications where meta annotations are the annotations about the annotations, which will be utilized to define another annotations.

1. @Override annotation:

```

class DBDriver {
    public void getDriver() {
        System.out.println("Type-1 driver");
    }
}

class NewDBDriver extends DBDriver {
    @Override
    public void getDriver() {
        System.out.println("Type-4 driver");
    }
}

```

class overrideTest

```
{ public static void main (String s[])
{
    DBDriver d = new NewDBDriver();
    d.getDriver();
}}
```

- The main purpose of this annotation is to check whether the sub class method is available or declared in the super class or not as part of method overriding process.
- As per the Java technology method overriding rules and regulations to perform method overriding we must maintain the same method prototype in both the methods. But if we have any single letter mistake in sub class method name, we are unable to get any message from compiler and JVM.

In the above context, to get a message when sub class method name is not matched with super class method name, we have to use @Override annotation.

Output ②

— If we compile the above program from command line it will provide the following message.

(error)

Only four commands found

{ } d->getDir()

{ } p & m(s)(c)

class Counterfeit

{ }

{ } type("Type-1 document")

public void getDir()

@Override

class NewDir extends DBDir

{ } type("Type-1 document")

public void getDir()

JPG to PDF Converter For Mac - Unregis

application on the screen
a message will

open your document just

} a.m(1);
{ A= new a();
P g U m(g+((a))

close document file

} b;

{ PPC,"m1c")-A11);

VBf R m()
@ document

close A

automation.

— To outcome—the above problem, you learned already
has provided an alternative that \rightarrow @ document

as documented ~~with~~ by
provided any support to make user-defined with
methods only documented, your rechability has not
in general, for your rechability some problems

definitely for your application

message when we access user defined document
as documented ~~with~~ and to get documentation
automatically PS to make any user defined up
@@ Documented :— the main purpose of this,

JPG to PDF Converter For Mac - Unregis

If we compile the above application then compiler will generate the following deprecation message.

Note: Deprecated test, four uses or override a deprecated API.

Note: Recompile with \rightarrow `-Xlint:deprecation` for details.

↳ Skipper warnings:

@SuppressWarnings :- In Java technology compiler may raise warning messages whenever it identifies unsafe operations. @unchecked operations and @on - In the above context to eliminate the warning messages which are going to generated by the compiler we need to use @SuppressWarnings annotation.

Qn

```
import java.util.*;
```

class A

```
{ @SuppressWarnings("unchecked")
```

```
public void getList()
```

```
{ ArrayList al = new ArrayList();
```

```
al.add('A');
```

```
('B');
```

```
('C');
```

```
System.out.println(al);
```

```
}
```

```
class WarningsTest
```

```
{ public static void main(String args)
```

```
{ A a = new A();
```

```
} a.getList();
```

Op Java.WarningsTest.java

Java WarningsTest

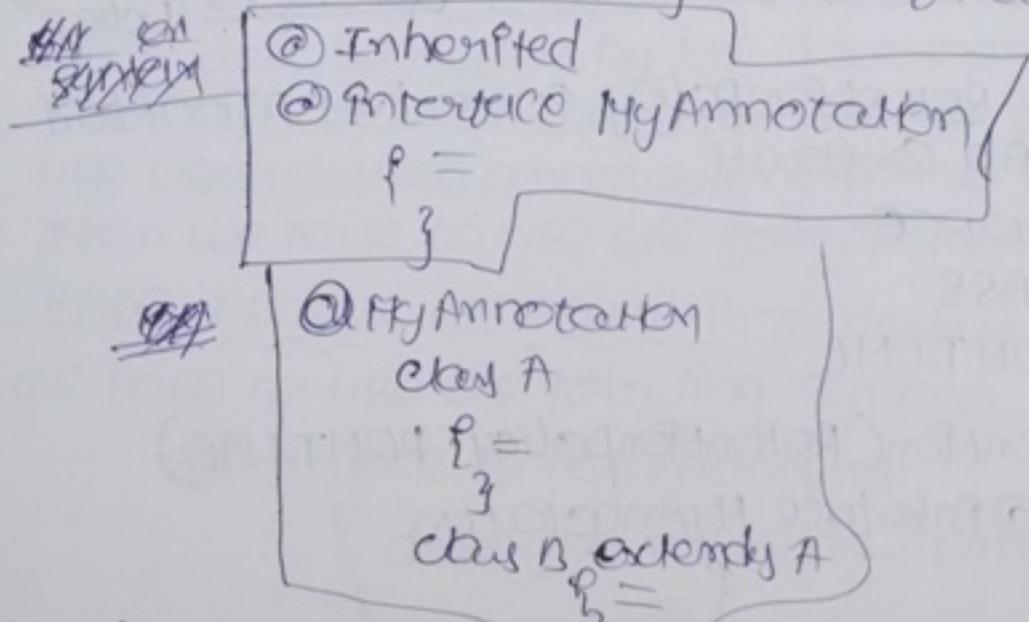
(A, B, C)

- In Java application if we add any elements to the collection without defining any generic type then compiler will provide warning messages.
- In the above application, we are able to stop the warning messages, we will use @SupressWarnings Annotations.
- If we compile the above piece of code, then compiler will not generate any warning messages.

Meta Annotation

@Inherited Annotation: In general in Java technology all the variables, methods, and so on are inheritable from super class to sub class by default. In Java technology, Annotations are not inheritable from super class to sub class by default.

In the above context if we want to make any user defined annotation as inheritable annotation then we have to use @Inherited annotation at the time of declaring user defined annotation.



In the above example, both class A and class B are getting the effect from @MyAnnotation.

@Documented annotation:

In general in Java technology, if we prepare documentation for any Java class then all the variables, methods, constructors and so on will be available in the documentation by default.

But as per the basic nature of the annotation, annotation will not be visible available.

In the respective documentation. In the above context, if we want to make any user-defined annotations as documentable annotation then we have to define that user-defined annotation with @Documented annotation.

- @Retention annotation :- the main purpose of @Retention annotation is to define scope or life of another annotation at the time of design.
- This annotation will take a property as parameter it should be the constants from RetentionPolicy enum.

Here the RetentionPolicy enum has provided the following constants

1. SOURCE
2. CLASS
3. RUNTIME

Ex @Retention(RetentionPolicy.RUNTIME)

 @Interface MyAnnotation

{
 =;
}

— @Target annotation:- This annotation can be used to specify the target elements at the time of declaration.

— If we want to make any annotation as a field level annotation or method level annotation and so on then we have to use @Target annotation at the time of declaring user-defined annotation.

This annotation will take parameters from Element-type enum like FIELD, METHOD, PARAMETER, CONSTRUCTOR, LOCAL-VARIABLE, ANNOTATION-TYPES PACKAGE.

ex) `@Target(ElementType.METHOD)
@Interface MyAnnotation
{
}`

User-defined Annotations: These are the annotations which were be define by the developers as per their application one requirement. If we want to use user-defined annotations in our Java application then we have to use the following steps.

Step 1: Declare Annotation: To declare an annotation we have to use the following syntax.

`@Interface AnnotationName
{
datatype member-name default value;
}`

Step 2: Utilize annotation in Java application as per the requirement.

To utilize annotations in Java applications, we will use the following syntax.

`@AnnotationName(member1=value1,
member2=value2,...)
=`

Step③: Access the data from annotation :-

To access the data from annotations first we need to get annotation object by using reflection API. That is class, Field, Method and so on -

If we use Field level annotation then we need to prepare class object, Field class object then we have to get Annotation object.

Example program

Institute.java

import java.lang.annotation.

 annotation;
 @Inherited
 @Documented
 @Target(ElementType.METHOD)
 @Retention(RetentionPolicy.RUNTIME)
 @Interface Institute

 String name() default "duga software solutions";
 String location() default "S.R. Nagar";
 String website() default "www.dugasoft.com".
}

Student.java

public class Student
{
 String id;
 String name;
 int marks;

Student(StudentId, StudentName, StudentMarks)

{ this.StudentId = StudentId;

this.StudentName = StudentName;

this.StudentMarks = StudentMarks;

@Institute(name = "Dwarka Engineering Academy",
location = "Mumbai")

public void getStudent()

{ System.out.println("Student details ..");

System.out.println("-----");

System.out.println("Student Id - " + StudentId);

System.out.println("Student Name - " + StudentName);

System.out.println("Student Marks - " + StudentMarks);

}

Annotation Test Java

import java.lang.reflect.*;

import java.lang.annotation.*;

public class AnnotationTest

{ presum(s) throws Exception

Student std = new Student("S1", "AAA", 78);

Class c = std.getClass();

Method m = c.getMethod("getStudent");

Institute t = (Institute) m.invoke(new AnnotationTest(),
(Institute.class));

```
std::getStudent());  
pop("Institute Details");  
pop("—");  
pop("Name—" + p.name());  
pop("Location—" + p.location());  
pop("website—" + p.website());
```

3
y
y

Difference between Servlet Config and Servlet Context

- ① Servlet Config is an object which will provide all the configuration details of a particular servlet like logical name of servlet, Initialization parameters.

Servlet context is an object, it will provide all the context details about a particular web application. It includes logical name of web application, context parameters of a web application and so on.

- ② Servlet Config object will provide the complete view of a particular servlet whereas Servlet context object will provide the complete view of a particular web application.
- ③ In web applications for each and every servlet a separate Servlet Config object will be prepared by the container when it receives request from client for the respective servlet. In general in the server container will prepare a separate Servlet Context object for each and every web application.
- ④ In web applications Container will create Servlet Config object immediately after servlet instantiation and just before calling init() in servlet Initialization.

Servlet Context is an object, it will be created by the container the moment when we startup the server i.e. at the time of web application development deployment.

⑤ Servlet config is an object it will be destroyed by the container just before servlet deinitialization. Servlet Context is an object it will be destroyed by the container if the movement when we undeploy the web application i.e. the movement when we shut down the server.

⑥ Due to the above reason the life of Servlet Config Object is almost equal to the life of respective Servlet

The life of the Servlet Context object is almost as equal to the life of the web application respectively

public interface ServletConfig
↳ public String getServletName();
↳ public String getInitParam();
↳ public Enviro... getInitParamValues();
↳ public ServletContent getServletContent()

↳ Public Interface ServletContent
↳ public String getServletContentName();
↳ getInitParameterNames();
↳ getInitParameterValues();
↳ getAttributeNames();
↳ getAttributeValues();
↳ setAttribute();
↳ removeAttribute();
↳ getAttribute();
↳ removeAttribute();

The life of ServletContext object is almost equal to the life of the respective web application.

2. If we keep any data inside ServletConfig Object then that data will be shared upto the respective servlet.

If we keep any data in ServletContext Object then that data will be shared upto all the ~~number~~ resources which are available in the respective web application. That is throughout the web.

Due to the above reason, the application scope of ServletConfig Object is upto a particular servlet. But the scope of ServletContext Object is upto a particular web application. Due to the above reason ServletConfig Object will provide less sharability but ServletContext Object will provide more sharability.

In web applications ServletConfig Object will allow only parameters data that is static inclusion of the data but ServletContext Object will allow both parameters data and attributes data.

That is both static inclusion and dynamic inclu-

In web applications container will prepare ^{8pm} ServletConfig Object only when it receives request from client except in load-on-startup case. But Container will prepare ServletContext Object at the time of server startup for perspective of getting request from client.

To get Serulet Context object in web application we have to use the following method from both Serulet Config and Serulet Request.

public SeruletContext getSeruletContext()

OR
SeruletContext context = config.getSeruletContext();
To specify logical name for the web application we have to use the following tags in web.xml file

<web-app>

<display-name> name </display-name>
</web-app>

To get the logical name of the web application from SeruletContext object we have to use the following method

public String getSeruletContextName()

If we want to keep Context parameters in SeruletContext object then first we have to declare them in web.xml file.

To declare Context parameters in web.xml file, we have to use the following tags in web.xml file.

<web-app>

<

{ <context-param>

<param-name> name </param-name>

<param-value> value </param-value>

</context-param>

<

</web-app>

single
context
parameters

— when we deploy the web application (or when we start the server), the main job of the Container is to recognize the respective web application, where container will recognize web.xml file when performing loading, parsing and reading the content of web.xml file.

while reading the content of web.xml file, if container identify any context parameters then container will read them and stored on the respective ServletContext object at the time of creation. ~~of ServletContext~~ / all are from ServletContext

→ To get a particular Context parameter value from ServletContext object then we have to use the following method: ServletContext

public String getInitParameter(String name)

→ To get all the names of context parameters from ServletContext object we will use the following method

public Enumeration getInitParameterNames()

— To set an attribute onto the ServletContext object we will use the following method

public void setAttribute(String name, Object value)

— To get a particular attribute value from ServletContext object we will use the following method

public Object getAttribute(String name)

- To remove an attribute from Servlet Context object we will use the following method
`public void removeAttribute(String name)`
- To get all the attribute names from Servlet Context object, we will use the following method
`public Enumeration<String>getAttributeNames()`

* What is Foreign Context:

Foreign Context is the Servlet Context object of another web application being executed at the same server. To get ForeignContext object we will use the following method from day

`[public ServletContext getForeignContext(Parsing Depth)]`

In general almost all the servers will not support ForeignContext concept, because due to their security constraints. If the underlying server supports ForeignContext then getForeignContext() method will return ForeignContext object. otherwise getForeignContext() method will return null value.

web.xml

Date: 24th Apr, 2012

```
<web-app>
  <display-name>myContext
    <context-param>
      <param-name>DBURL
      <param-value>oracle.jdbc.driver.OracleDriver
    </context-param>
    <context-param>
      <param-name>url
      <param-value>jdbc:oracle:thin@localhost:1521:xe
    </context-param>
    <context-param>
      <param-name>user
      <param-value>system
    </context-param>
    <context-param>
      <param-name>password
      <param-value>durga
    </context-param>
  < servlet >
    < servlet-name >
      < servlet-class >
    </ servlet >
  < servlet-mapping >
    < servlet-name > ms < /servlet-name >
    < url-pattern > /context < /url-pattern >
  < servlet-mapping >
</web-app>
```

context app
| WEB-INF
| web.xml
| classes
+ MyServlet.class

MyServlet.java

```
import java.io.*;  
import javax.servlet.*;  
import javax.servlet.http.*;  
  
public class MyServlet extends HttpServlet  
{  
    public void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException,  
        IOException  
    {  
        ServletContext context = req.getServletContext();  
        // ServletContext context = getServletConfig().  
        // getServletContext();  
        String name = context.getServletName();  
        String driverClass = context.getInitParameter(  
            "driverClass");  
        String url = context.getInitParameter("url");  
        String user = context.getInitParameter("user");  
        String password = context.getInitParameter("password");  
        context.setAttribute("username", "Durga");  
        context.setAttribute("uage", new Integer(25));  
        res.setContentType("text/html");  
        PrintWriter out = res.getWriter();  
        out.println("<html>");  
        out.println("<body>");  
        out.println("<b><font size='3'>");  
        out.println("<br><br>");  
        out.println("<img alt='Unregister' border='1' src='http://www.google.com' />");  
    }  
}
```

```

out.println("<br><br>");  

out.println("prefixes--" + driverKey);  

out.println("<br><br>");  

out.println("DriverURL--" + url);  

out.println("<br><br>");  

out.println("User--" + user);  

out.println("<br><br>");  

out.println("password--" + password);  

out.println("<br><br>");

java.util.List<String> e = context.getInputs  

    .parameterNames();  

    while (e.hasMoreElements()) {  

        System.out.println(e.nextElement() + "<br><br>");  

        System.out.println("User name--" + context.getAttribute  

            ("uname"));  

        System.out.println("User Age--" + context.getAttribute("age"));  

        System.out.println("<br><br>");  

        e = context.getAttributeNames();  

        while (e.hasMoreElements()) {  

            System.out.println(e.nextElement() + "<br><br>");  

            System.out.println("What is the diff b/w  

                attributes & context parameters");  

            System.out.println("why we  

                are getting bying  

                attributes");  

            System.out.println("why we  

                are getting bying  

                attributes");
        }
    }
}

```

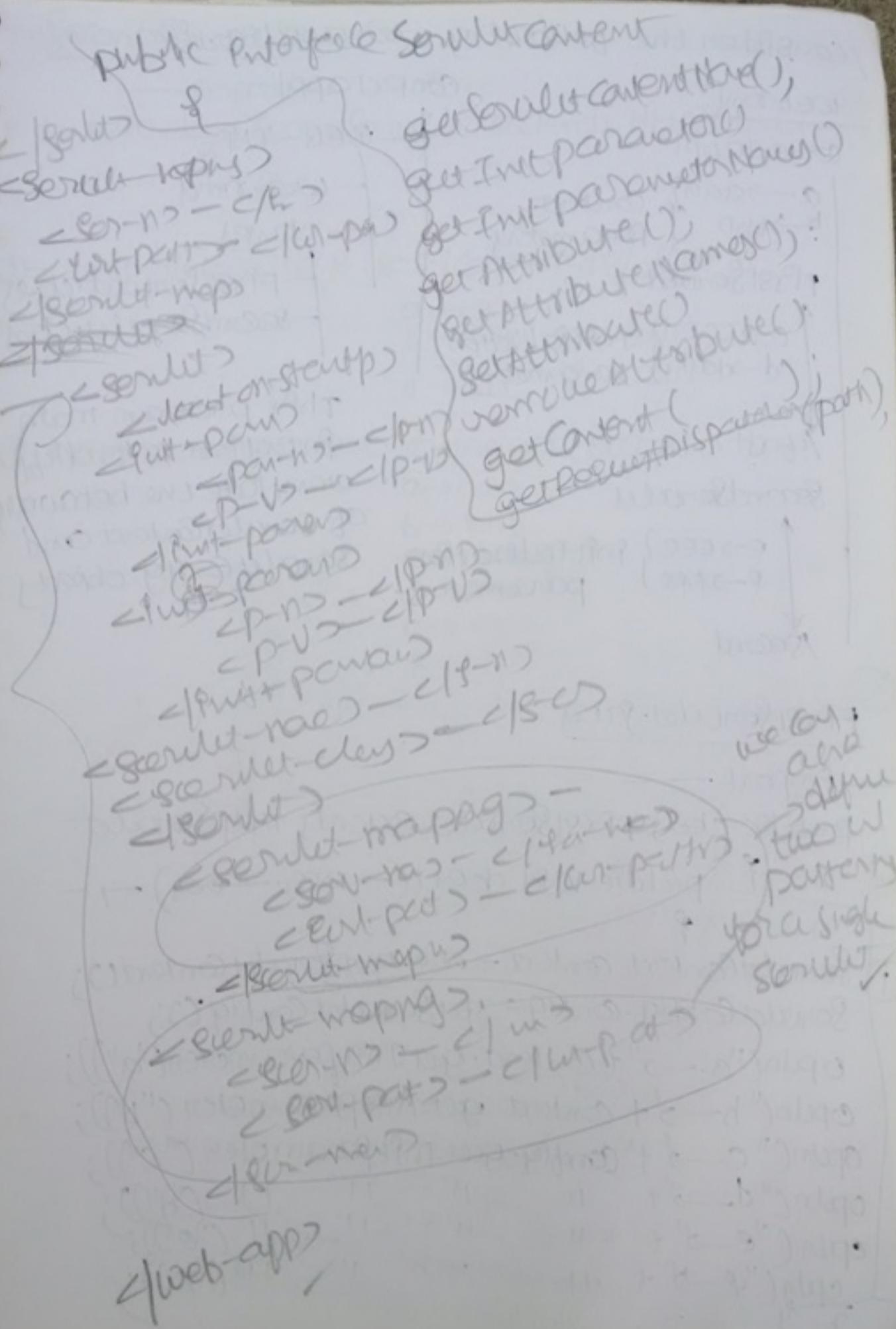
```
out.println("Foreign Context -- " + context);
context.getContext("/config/app"));
}
}
```

execution

upon web

```
<web-app>
  <display-name>larnepp</display-name>
  <content-param>
    <param-name> - <param-name>
    <param-value> - <param-value>
    <param-param?>
      <param-name> - <param-name>
      <param-value> - <param-value>
      <param-param?>
        <param-name> - <param-name>
        <param-value> - <param-value>
        <param-param?>
          <param-name> - <param-name>
          <param-value> - <param-value>
        </param-param?>
      </param-param?>
    </param-param?>
  </content-param>
  <welcome-file-list>
    <welcome-file> - <welcome-file>
    <welcome-file> - <welcome-file>
  </welcome-file-list>
  <serlet>
    <location-startup> / <location-startup>
    <init-param?>
      <param-name> - <param-name>
      <param-value> - <param-value>
    </init-param?>
    <init-param?>
      <param-name> - <param-name>
      <param-value> - <param-value>
    </init-param?>
    <init-param?>
      <param-name> - <param-name>
      <param-value> - <param-value>
    </init-param?>
  </serlet>
```

JPG to PDF Converter For Mac - Unregis



Consider the following web application

web.xml

ContextApp1

a → aaaa
b → bbbb } Context
parameters

FirstServlet

c → ccc
d → ddd } Initialization
parameters.
e → eee
f → fff } Initialization
parameters.
g → ggg
h → hhh } Initialization
parameters.

SecondServlet

FirstServlet.java

Import —

public class FirstServlet extends HttpServlet

{
 public void doGet(HttpServletRequest request, HttpServletResponse response) {
 //
 }
}

① ServletContext context = request.getServletContext();
ServletConfig config = getServletConfig();
System.out.println("a → " + context.getInitParameter("a"));
System.out.println("b → " + context.getInitParameter("b"));
System.out.println("c → " + config.getInitParameter("c"));
System.out.println("d → " + " " + " " + " " + config.getInitParameter("d"));
System.out.println("e → " + " " + " " + " " + config.getInitParameter("e"));
System.out.println("f → " + " " + " " + " " + config.getInitParameter("f"));
System.out.println("g → " + " " + " " + " " + config.getInitParameter("g"));
System.out.println("h → " + " " + " " + " " + config.getInitParameter("h"));

JPG to PDF Converter For Mac - Unregis

ContextApp1

WEB-INF

web.xml

classes

FirstServlet.class
SecondServlet.class

This program main intention is to differentiate the behavior of ServletContext and ServletConfig obviously

⑨ SecondServlet.java

— import —

public class SecondServlet extends HttpServlet
{
 }
}

① → http://localhost:8080/Contextapp/first

a → aaa
b → bbb e → null
c → ccc f → null
d → ddd

② → http://localhost:8080/Contextapp/second

a → aag
b → bbb
c → null
d → null
e → eee
f → fff

JBoss Server

[Date : 25th Aug 2012]

JBoss is an application server, it will provide very good environment to execute Java based enterprise applications. JBoss Server is an application server, it will provide almost all the middleware services which are provided by the application servers in general like JNDI, JDBC, Security, Transactions, JMS, and so on--

JBoss
as
7.1.0.
final
JBoss
- 6.0.0
final

- 2) JBoss 7 application server is compatible with Java 7 and it able to support JEE 6 version.
- 2) JBoss 7 application server is able to provide environment for Servlet 3.0 version, JSP 2.2 version and so on.
- 2) when we download JBoss Server from internet, we are able to get

(jboss-as-7.1.0-final-21.zip file)

If we extract it, JBoss Server distribution file will be available

- 2) By default JBoss Server was provided with the http port number 8080, with this if we run JBoss server then we are able to get an exception like JVM Bind exception.

2) To overcome this, we have to change JBoss Server port number. To change JBoss Server http port number we have to use the following way

- 2) open JBoss home directory jboss-as-7.1.0-final
 - ↓
↓ folder -
 - Standalone

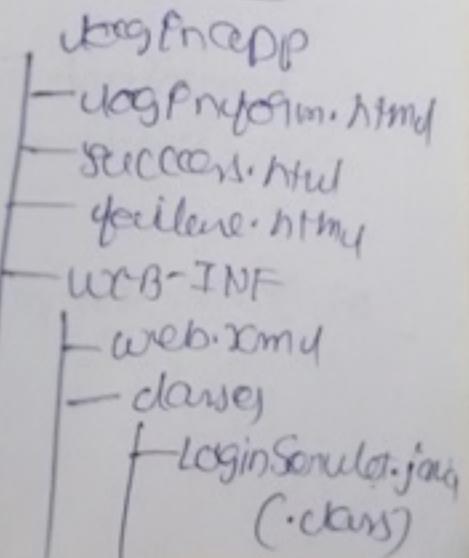
Standalone.xml ↗ open ↗ Configuration
 ↗ Configuration
 ↗ Configuration

- Search for 8080 in that XML file and replace with a new port number → [If we open in edit plus, at 291th line we can find this port number]
- To create an account in JBoss server, we have to use the following path
Open JBoss → bin → (double click on add-user.bat file)
Provide type of user that is Application User (Type this at cmd prompt)
- Enter the details of the new user to add :
 - Realm(Hanmanagement) : J (No need to provide information here just click on enter)
 - User name : vijay
 - password : vijaykumar
 - Repassword : vijaykumar
- Type yes and press enter

Step ①: prepare web application and RH user file separately.

Deployment War file

d:\pepsi\



To compile the servlets we have to use the following jar file in class path environment variable

Add Content → available at top right hand side

Create → D:\eclipse\logfinapp.war

Select war file.

browse → Save

If we click on save button, the ~~uploaded~~ specified war file will be uploaded to the server and listed in the displayed applications.

→ When we deploy a web application by default that application will be undeployed mode but we have to make it as enable.

Then click on enable button.

Confirm button.

→ If the above action over web application correctly be appear in the displayed applications list like

Name	RecentFinance	enable	Undeploy	Remove
logfinapp.war	logfinapp.war	✓	[enable]	[remove]

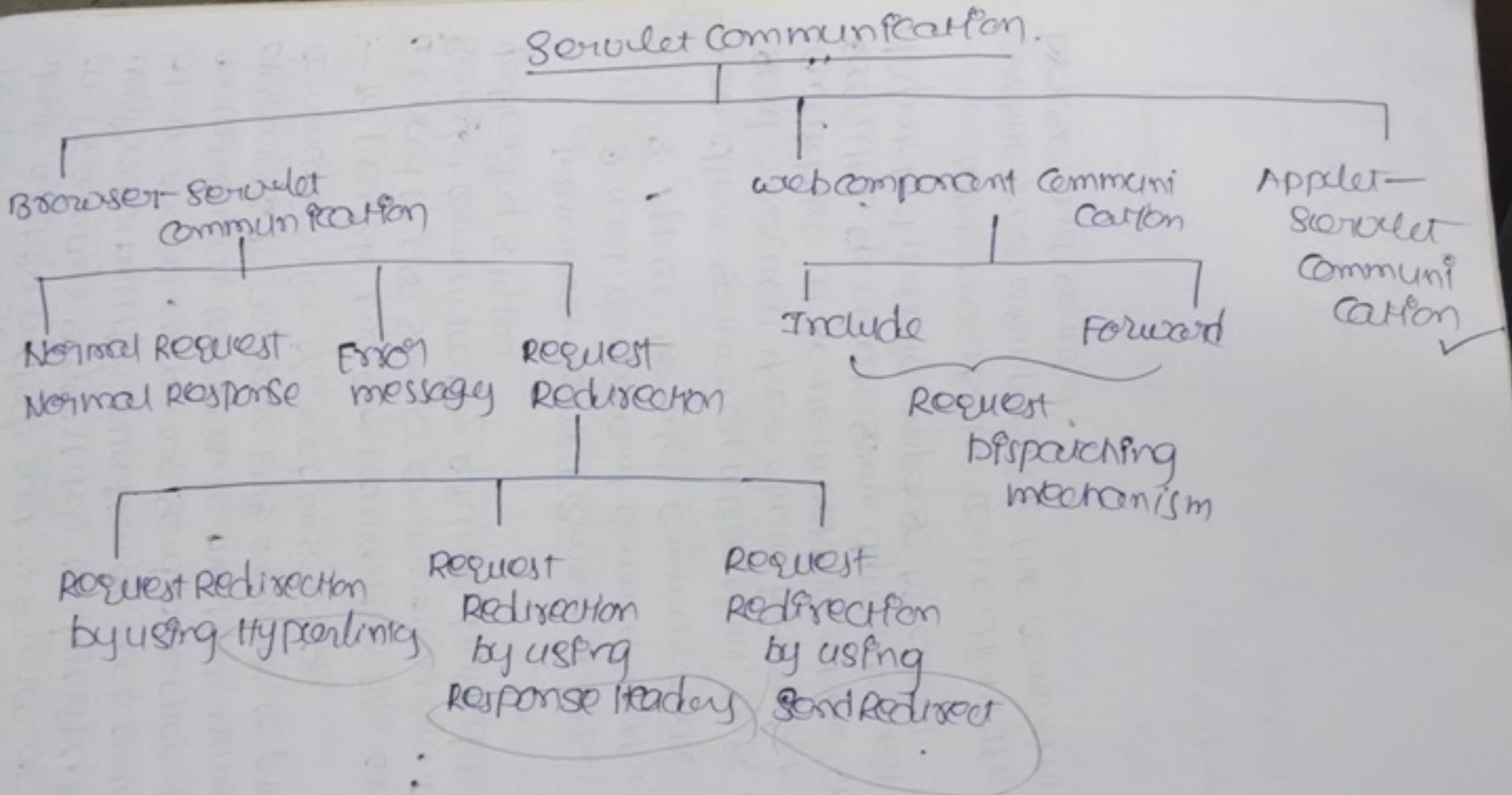
→ Access the web application

To access the web application,

open new window or tab where we have to use the following url:

https://localhost:8888/logfinapp

Servlet Communication:



JPG to PDF Converter For Mac - Unregistered

In general, in web application development, it is not always suggested to distribute the application logic within a single web resource at server side. It is always suggested to distribute application logic over multiple number of web resources. In the above context, when we send a request from client, the container has to execute multiple number of web resources. To achieve this we have to provide the communication b/w web resources.

To provide the communication b/w web resources we need to use Servlet Communication. In web application, Servlet communication will be divided into the following 3 types:

- 1. Browser-Servlet Communication
- 2. Web Component Communication
- 3. Applet-Servlet Communication

In client-server applications, it is convenient to send a request from client browser to a servlet available at server machine, where servlet will be executed, generate some dynamic response and dispatch dynamic response to client.

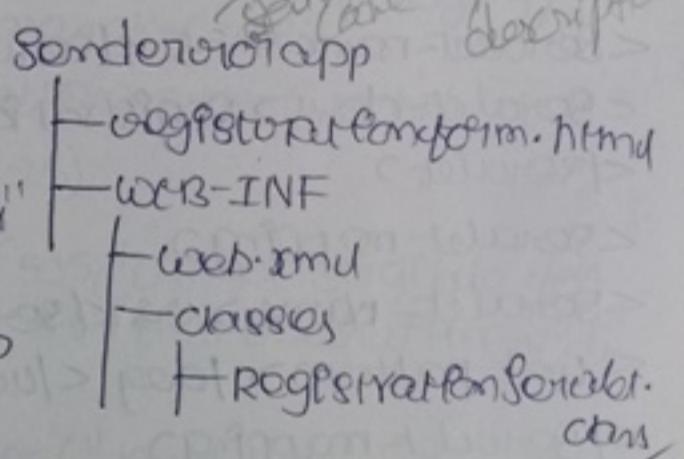
With the above convention client server architecture will provide the communication b/w browser and servlet.

② Sending Error message [Date : 26th April, 2012 Then]

In general when we send a request from client to server via a particular servlet then container will execute the respective servlet while executing the servlet if container encounter any problem or exception then container will send the respective error message in its own format to the client. Similarly as part of the application execution, as per the application requirement if we want to send application specific error message to the client in container defined format then we have to use the following method from Servlet Response.

public void sendError(int sc, String desc)

```
<html>    registration  
<head>        form.html  
<center><b>  
<font size="7" color="red">  
Registration form  
</font><b></center>  
<br><br><br>  
<body bgcolor="lightpink">  
<b><font size="7">  
<form method="get"  
action="/reg">
```



505 or not eligible

res.sendRedirect
(505, "OR not eligible
form");

<form>

 name:<input type="text" name="uname"/>
 Age:<input type="text" name="age"/>

 Address:<input type="text" name="address"/>

 Mobile:<input type="text" name="mobile"/>

<input type="submit" value="Submit"/>

</form></form></body></html>

web.xml

<web-app>

 <welcome-file-list>

 <welcome-file>registrationform.html</welcome-file>

 </welcome-file-list>

 <Servlet>

 <Servlet-name>rs</Servlet-name>

 <Servlet-class>RegistrationServlet</Servlet-class>

 </Servlet>

 <Servlet-mapping>

 <Servlet-name>rs</Servlet-name>

 <url-pattern>/reg</url-pattern>

 </Servlet-mapping>

</web-app>

RegistrationServlet.java

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

public class RegistrationServlet extends HttpServlet
{
    public void doGet(HttpServletRequest req,
                       HttpServletResponse res) throws ServletException,
    IOException
    {
        String uname = req.getParameter("uname");
        int uage = Integer.parseInt(req.getParameter("uage"));
        String uaddr = req.getParameter("uaddr");
        String mobfile = req.getParameter("mobile");
        if (uage < 18 || uage > 25)
        {
            res.sendError(505, "UR not eligible for  
this recruitment");
        }
        else
        {
            res.setContentType("text/html");
            PrintWriter out = res.getWriter();
            out.println("<html>");
            ("<body style='background-color: lightyellow;>");  

            ("<center><b>");  

            "<font size='6'><br><br>");  

            ("Name---" + uname);
```

```
out.println("<br><br>");  
out.println("Age---"+age);  
out.println("<br><br>");  
out.println("Address---"+address);  
out.println("<br><br>");  
out.println("Mobile---"+mobile);  
out.println("<br><br>");  
out.println("U'R registration success");  
out.println("</div></body></center></body></html>");  
}  
}  
catch(Exception e)  
{  
    e.printStackTrace();  
}  
}  
}  
} --| send over http://192.168.1.104:8080/ * found
```

why we use
as block
here.

Request Redirection:— The process of bypassing the request from one web application to another web application is called as Request Redirection.

In web applications,

there are three ways to achieve

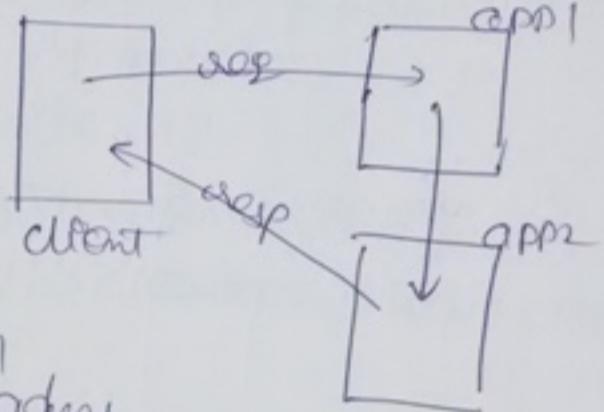
Request Redirection

1. Request Redirection by using hyperlink

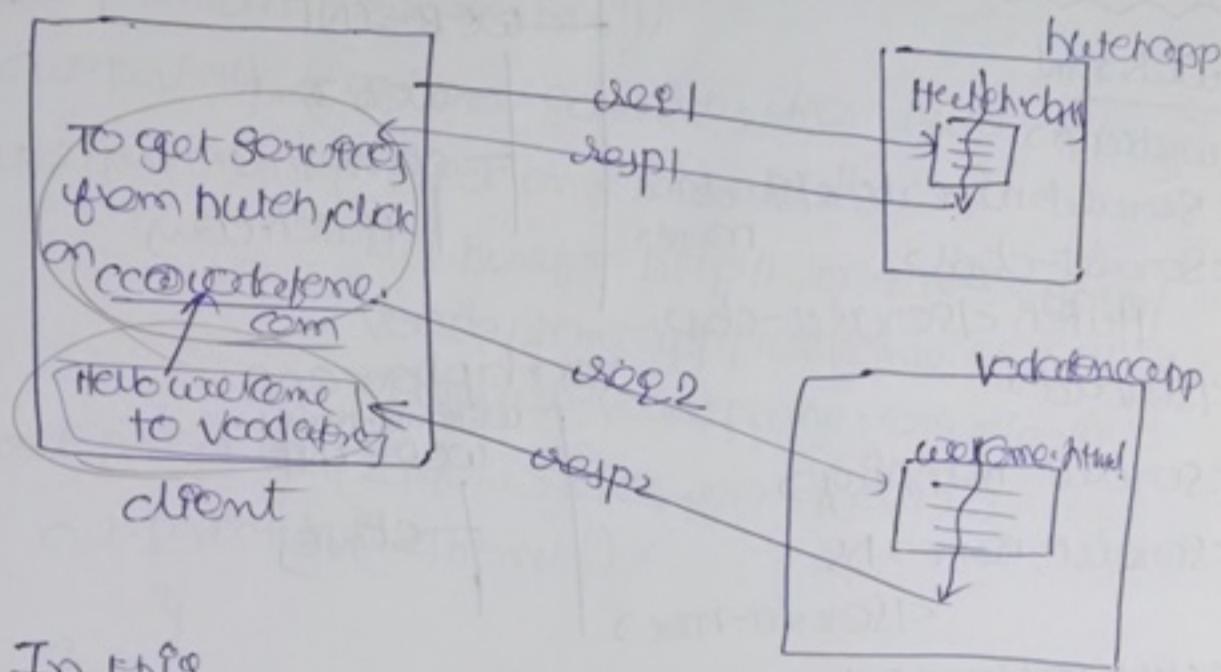
2. Request Redirection by using response header.

3. Request Redirection by using sendRedirect mechanism.

2



Request Redirection by using hyperlinks :-



In this

mechanism, when we send a request from client to first web application, where the first web application may generate dynamic response to the client with an hyperlink referring the new web application (second web application). In this context, If we click on generated hyperlink automatically another request will be sent to the new webapp, and it may provide the required response to the client.

hutchapp

web.xml

```

<web-app>
  <Servlet-name>Hutch </Servlet-name>
  <Servlet-class> Hutch </Servlet-class>
  <Servlet-mappings>
    <Servlet-name>Hutch </Servlet-name>
    <url-pattern>/Hutch </url-pattern>
  </Servlet-mappings>
</web-app>

```

Hutch.java

```

import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.IOException;
public class Hutch extends HttpServlet {
    public void doGet(HttpServletRequest req,
                      HttpServletResponse res) throws ServletException,
                      IOException {
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
        out.println("<html>");
        ("<body bgcolor='lightyellow'>");
```

hutchapp

WEB-INF

web.xml

classes

Hutch.class

Vodafoneapp

WEB-INF

classes

```
out.println("<center><b><font size=16's'>");  
out.println("<br><br>");  
out.println("To end surfaces from hatch <br><br>");  
out.println("click on <br><br>");  
out.println("a href='http://localhost:8080/  
voadaphoneapp/welcome.html'>  
@@www.Voadaphone.com </a>");  
out.println("</font></center></body>");  
out.println("</html>");  
}  
}  
compile
```

Vodafoneapp

Welcome.html:

<html>

<body bgcolor="lightgreen">

<center>

Welcome to vodafone.

</center></body></html>

executing:

Drawback

→ This mechanism is not guaranteed mechanism to perform Request Redirection because in this approach to achieve Request Redirection user has to click on hyperlink.

Request Redirection by setting Response Headers

In this mechanism, [Date : 27th April, 2012 FRI] when we send a request to first web application then first web application will set redirection status code that is 3xx and new web application will act as new web application and its local response header response format.

In the above case when the response is reached to client then client will pickup status line value that is 3xx that is redirection status code, with this client will pickup location response header value that is new web application url then client will send another request automatically as per the new web application url.

To set a status code to the response format we will use the following method from HTTP Servlet Response.

public void setStatus(int sc)

To represent request redirection HttpServlet Response has provided the following two static codes in the form of public static final constants ~~for all the possible values of the status code~~.
public static final int SC_MOVED_PERMANENTLY.
public static final int SC_MOVED_TEMPORARILY.

By ~~HTTP~~ in the form of ~~constants~~

To set a particular value to a particular Response Header, we have to use the following method.

public void setHeader(String name, String value)

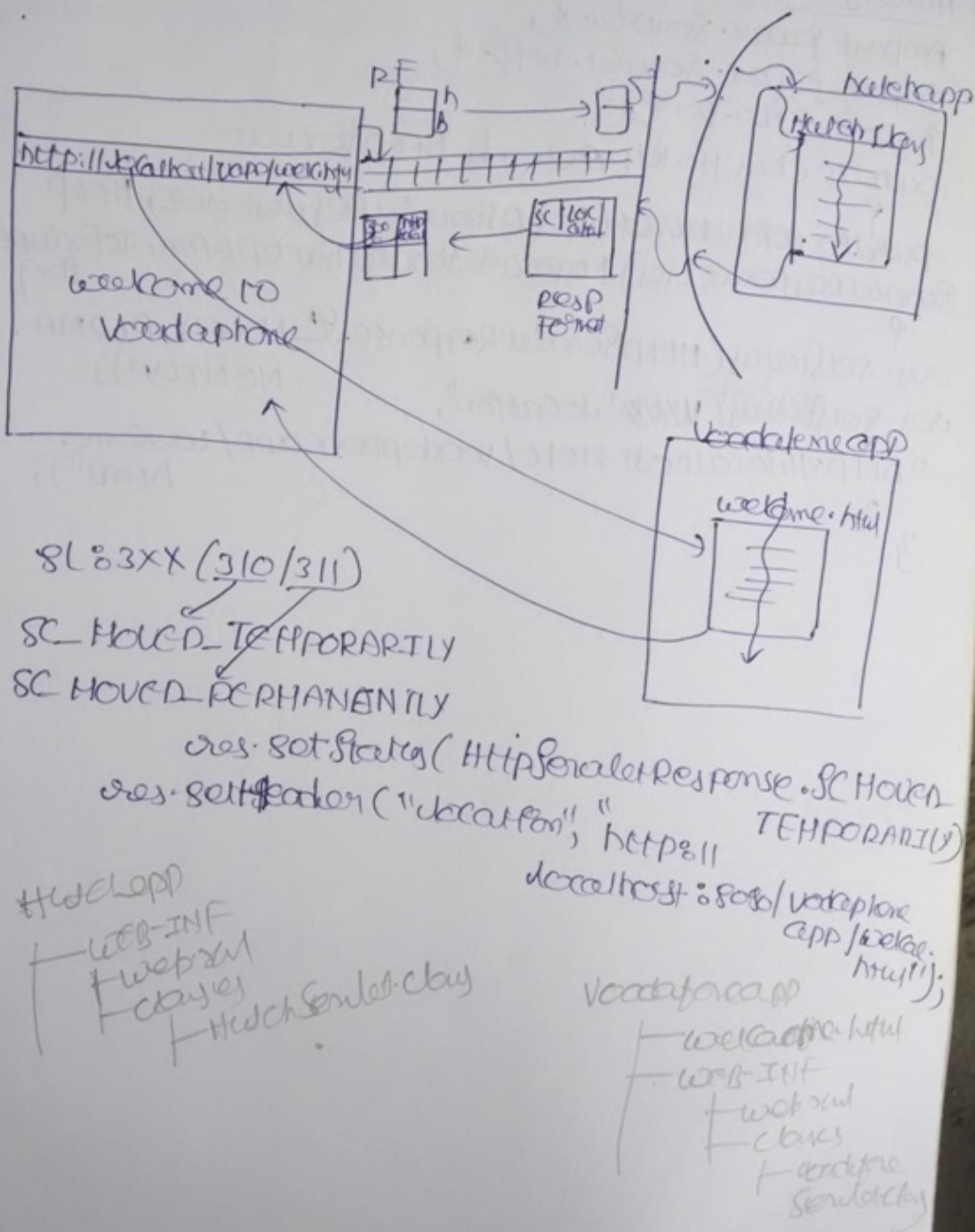
As part of Request Redirection we need to set new web application URL to location response header.

public interface HttpServletResponse
 {
 public void setStatus(int sc);
 public void setHeader(String name, String value);
 public static final int SC_MOVED_TEMPORARILY;

What are these
and their
meaning?

getstatus()
setHeader(- -)
getHeader(" ")
sendRedirect(" "))

this
value
will
be
set
for
not
permanently



JPG to PDF Converter For Mac - Unregis

- To set a status code to the response to we will use the following method from HTTPServlet
 - setStatus(int sc)
- To represent request redirection HTTPServlet we will use the following two static constants
 - SC_MOVED_TEMPORARILY — PERMANENTLY.
 - SC_MOVED_PERMANENTLY — TEMPORARILY.
- To represent request redirection HTTPServlet we need to set Response has provided the form of public static final int SC_MOVED_TEMPORARILY codes in the form of public static final int SC_MOVED_PERMANENTLY.
- To set a particular value to a particular response by HTTPServlet we have to use the following method
 - setHeader(String name, String value)
- To set a particular value to a particular response by HTTPServlet we have to use the following method
 - getHeader(String name)
- To set a particular value to a particular response by HTTPServlet we have to use the following method
 - getHeaders()
- To set a particular value to a particular response by HTTPServlet we have to use the following method
 - getHeader(-1)
- To set a particular value to a particular response by HTTPServlet we have to use the following method
 - getHeader(int index)
- To set a particular value to a particular response by HTTPServlet we have to use the following method
 - sendRedirect(String url)

Hutch.java

```
import java.io.*;  
import javax.servlet.*;  
import javax.servlet.http.*;  
import java.io.*;  
  
public class Hutch extends HttpServlet  
{  
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException, IOException  
    {  
        response.setStatus(HttpServletResponse.SC_MOVED_PERMANENTLY);  
        response.setHeader("HTTP", "location", "http://localhost:1010/voicophoneapp/welcome.html");  
    }  
}
```

Hutch
Anjou

Import Ruby

Import SubRequestor

Public class Box < ApplicationController

public class Hutch

public class Box < ApplicationController

req = Request.new("GET / HTTP/1.1", "127.0.0.1", "HTTP/1.1")

response = Response.new("HTTP/1.1 200 OK", "Content-Type: text/html", "Hello, world!")

request = Request.new("GET / HTTP/1.1", "127.0.0.1", "HTTP/1.1")

response = Response.new("HTTP/1.1 200 OK", "Content-Type: text/html", "Hello, world!")

request = Request.new("GET / HTTP/1.1", "127.0.0.1", "HTTP/1.1")

response = Response.new("HTTP/1.1 200 OK", "Content-Type: text/html", "Hello, world!")

request = Request.new("GET / HTTP/1.1", "127.0.0.1", "HTTP/1.1")

response = Response.new("HTTP/1.1 200 OK", "Content-Type: text/html", "Hello, world!")

request = Request.new("GET / HTTP/1.1", "127.0.0.1", "HTTP/1.1")

response = Response.new("HTTP/1.1 200 OK", "Content-Type: text/html", "Hello, world!")

request = Request.new("GET / HTTP/1.1", "127.0.0.1", "HTTP/1.1")

response = Response.new("HTTP/1.1 200 OK", "Content-Type: text/html", "Hello, world!")

request = Request.new("GET / HTTP/1.1", "127.0.0.1", "HTTP/1.1")

response = Response.new("HTTP/1.1 200 OK", "Content-Type: text/html", "Hello, world!")

request = Request.new("GET / HTTP/1.1", "127.0.0.1", "HTTP/1.1")

response = Response.new("HTTP/1.1 200 OK", "Content-Type: text/html", "Hello, world!")

JPG to PDF Converter For Mac - Unregistered

Drawback:

- To perform request redirection if we use this mechanism (achieved by using response headers) then everytime we have to set static value and this is not suggestible to set value to the response headers explicitly. Due to this reason to simplify Request Redirection we will go to an alternative mechanism that is Send Redirect mechanism.

Request Redirection by sendRedirect mechanism

In this mechanism, client is able to get the required response with a single request by performing Request Redirection. This mechanism is same as Request redirection by setting response headers mechanism, but in sendRedirect mechanism no need to set Redirecational status code to the Response Header explicitly.

To achieve Request Redirection with this mechanism we will use the following method from HttpServletResponse

public void sendRedirect(String url)

Detail

This method will set the
status code & automatically
close the response header
and to the response
automatically

Onpl HutchServerlet.java

Print -

public class HutchServerlet

{

public void doGet(-,-) throws -,-

{ =

res.sendRedirect("http://localhost:1016/
vodafoneapp/welcome.html");

}

(no print)

JPG to PDF Converter For Mac - Unregis

NOTE: The main advantage of Request Redirection is to provide the communication b/w two resources which may be available at same server and may be available at two different servers.

Simple
Program

JPG to PDF Converter For Mac - Unregistered

Date: 28th April, 2012 Sat

Web Component communication

- The process of providing communication between two server side resources is called as web component communication.
- In general in web applications, web component communication will be existed in the form of servlet-to-servlet, servlet-gsp, servlet-html, gsp-gsp, gsp-servlet, gsp-html.
- To achieve web component communication, in web application we can use the following two mechanisms:
 - ① include mechanism
 - ② forward mechanism.
- The above include and forward mechanisms are called as request dispatching mechanism. Before include and forward mechanisms should acquire request dispatcher object internally.
- To achieve web component communication in web applications we have to use the following two steps:
 - ① get request dispatcher object
 - ② access either include or forward on request dispatcher reference.

Request Dispatcher:— Request Dispatcher is an object, it will provide very good environment either to forward request from present web resource to target web resource or to include target web resource response to present web resource response.

— To represent request dispatcher object Server API has provided a predefined interface.

[Forward-Servlet Request Dispatcher]

— To get Request Dispatcher object we have to use either of the following methods

- ① from ServletContext, we can use the following methods
 - ① get Request Dispatcher(path);
 - ② getNamed Dispatcher(path);

- ② from Servlet Request

- ① get Request Dispatcher(path);

```
public Forward Result Dispatcher  
{  
    public void include(ServletRequest req, ServletResponse res) throws IOException;  
    public void forward(ServletRequest req, ServletResponse res) throws IOException;  
}
```

3

Date: 28th April, 2012 Sat

Using Communication

Once you have created an application, it can be used by other applications, which communicate with it.

- Inter-App, inter-Component
- Between different parts of the same application.
- Between different applications.

The mechanisms are

very similar to those used in Java.

Communication can be done through

Request Dispater
object, it wraps
request to you was
target to target
resource to target
resource with respect
— To response
App has provided a
factory. Some

Get Request Dispater
object of the following
① From Scroller Context
② From Request Dispater
③ From Scroller Request
④ Get Request Dispater

public pureVirtual Request

public void Provide(Request)
public void Forward(Request)

and on request

→ what is the difference between getRequestDispatcher() and getNamedDispatcher() methods

Ans: Both the methods from ServletContext Interface can be used to get RequestDispatcher object. To get RequestDispatcher object, if we use getRequestDispatcher() method then we have to pass the logical name of the target resource as parameter.

public RequestDispatcher getRequestDispatcher()

Note:

in case of Servlets logical name

(String path)

is an user pattern defined by Servlet in web.xml file.

To get RequestDispatcher object if we use getNamedDispatcher() then we have to pass the logical name of the target resource.

Note: in case of Servlets logical name is the name specified along with <Servlet-name> tag in web.xml file.

public RequestDispatcher

getNamedDispatcher(String name)

2) what is the diff b/w getRequestDispatcher from ServletContext and ServletRequest

Ans: Both the methods can be used to get the Request Dispatcher object. To get the RequestDispatcher object if we use getRequestDispatcher() from ServletContext then we have to pass the relative part of the target resource as a parameter.

To get RequestDispatcher object if we use getRequestDispatcher() from ServletRequest then we have to pass either relative part or the absolute path of the target resource as a parameter.

what is the difference between include and forward mechanism?
target resource @ observed path of the target resource.

Note: Relative path is a path it should be prefixed with (*) but observed path is a path which should not be prefixed with (*)

Step 2 After getting RequestDispatcher object to perform web-component communication, we have to access either of the following methods from RequestDispatcher

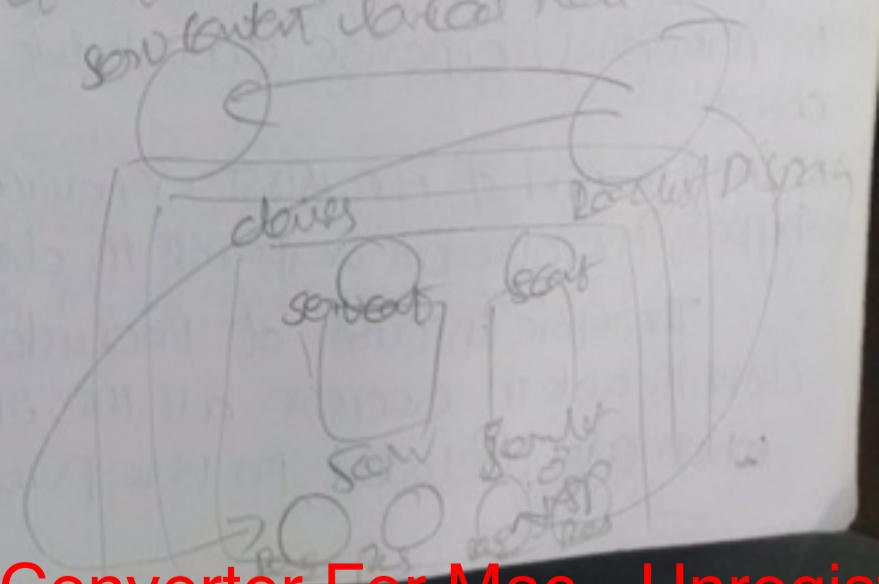
[Public void include(ServletRequest req, ServletResponse res) throws ServletException, IOException]

[Public void forward(ServletRequest req, ServletResponse res) throws ServletException, IOException]

Difference between include and forward mechanism

Include mechanism:

what is the benefit of this doAdd
so content is not new



To provide web component communication, if we use include mechanism then we are able to include the target resource response into present request response.

In case of include mechanism when we send a request from client to first resource then container will prepare request and response objects.

In first resource by the execution of some content some response will be generated from response object. When container encounters include method then container will bypass request and response objects to a target resource without refreshing or without eliminating previously response in the response object.

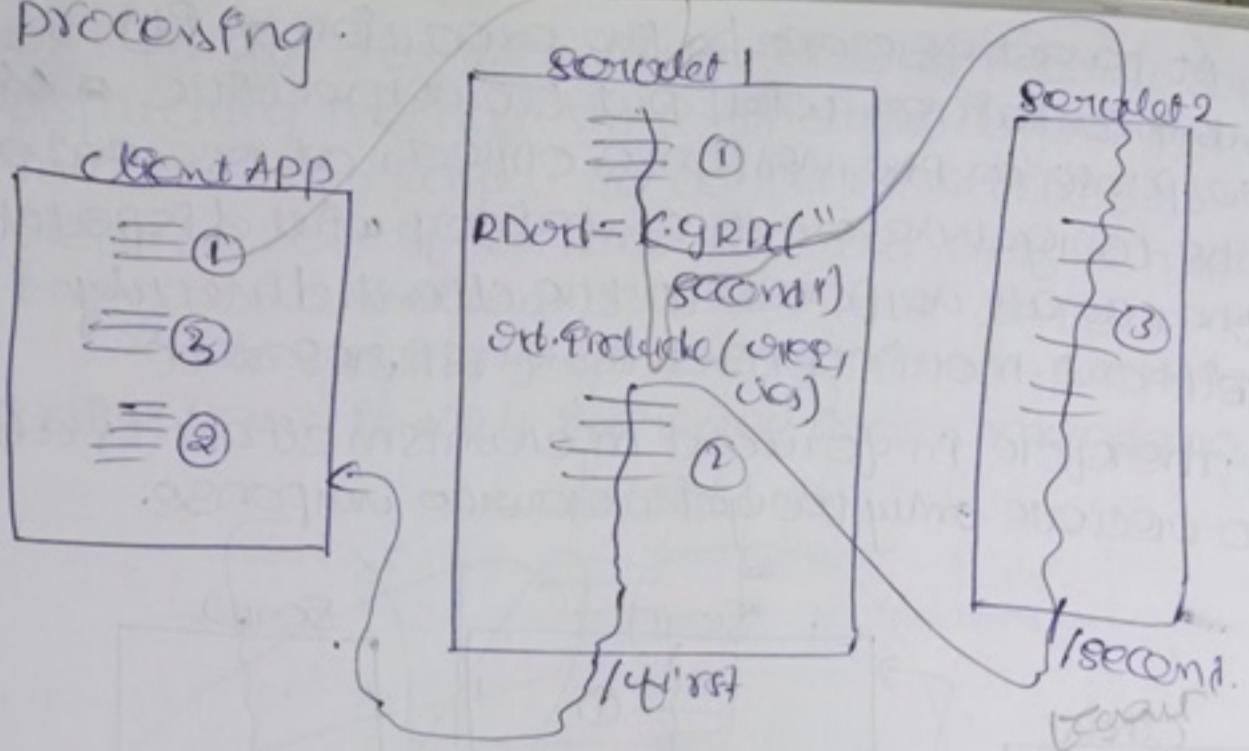
In this context, by the execution of target resource some other response will be added to previous response in the response object. At the end of the target resource container will bypass the request and response objects back to the first resource, without dispatching response to client directly.

By the execution of the remaining content in first resource some other response will be added to previous response available in the response object.

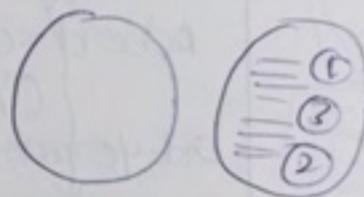
At the end of the first resource container copy bypass the overall response to client.

Therefore in case of include mechanism, client is able to receive all the resources response which are included in the present request.

processing.



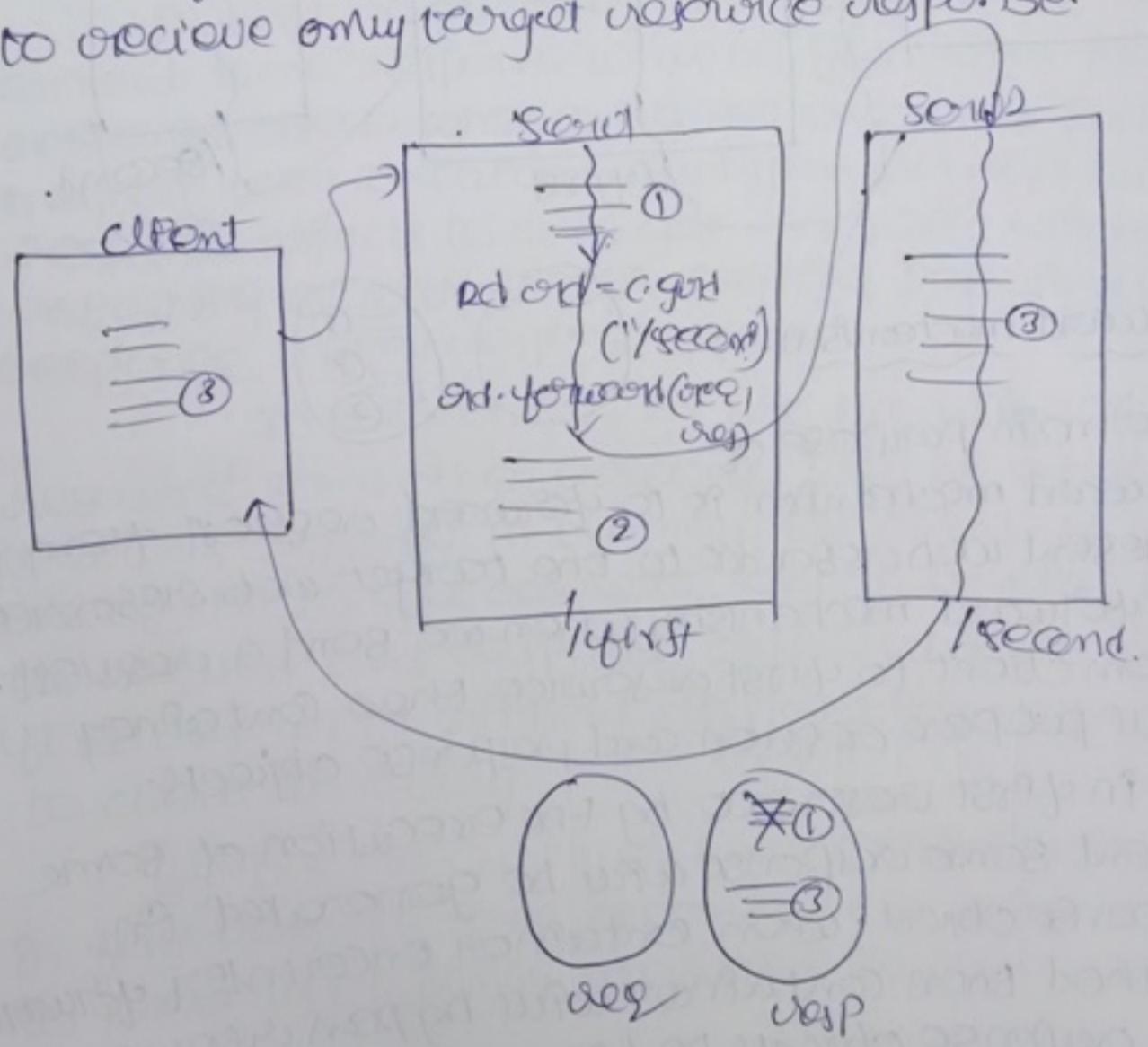
Forward mechanism:



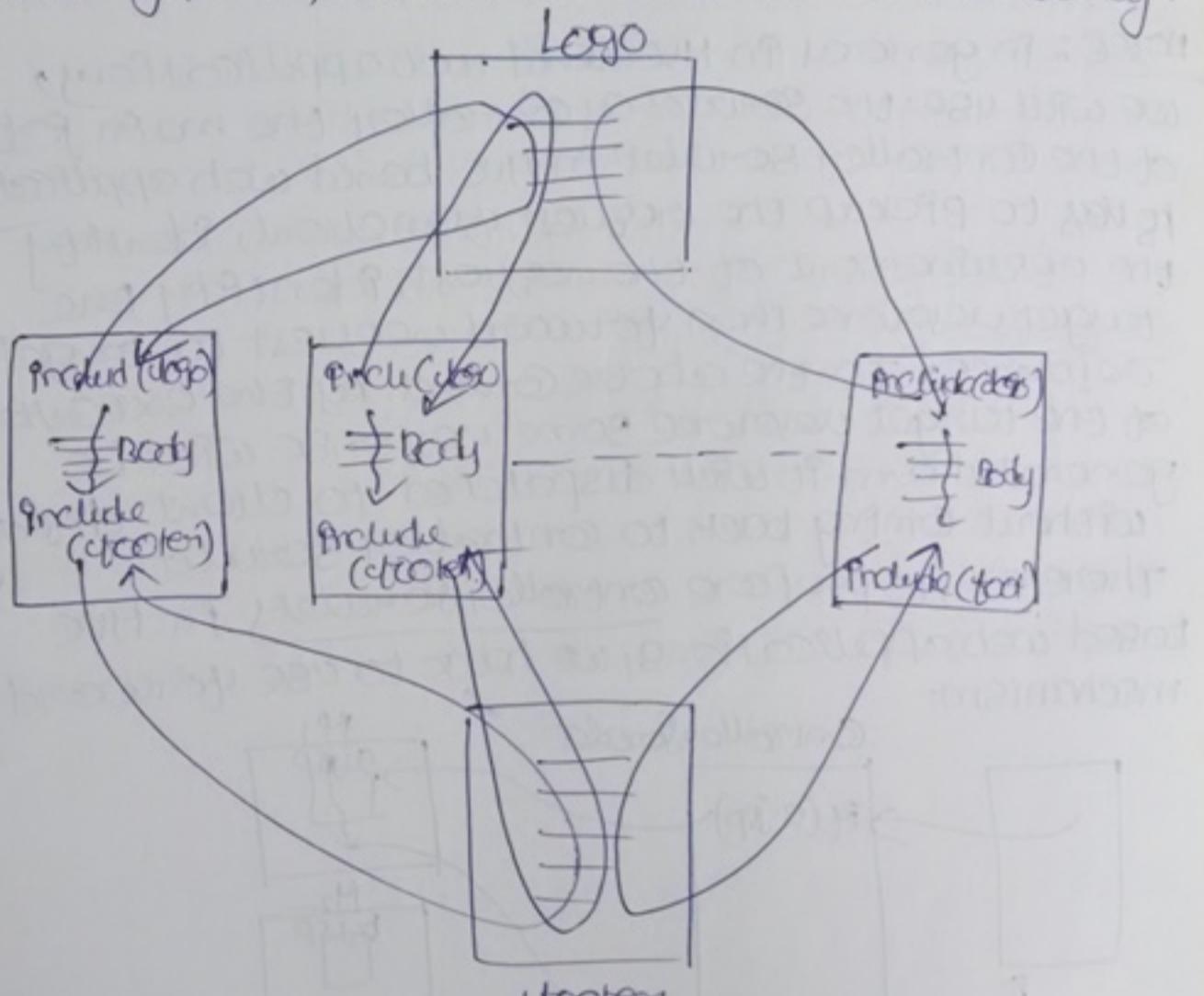
The main purpose of forward mechanism is to forward request from present web resource to the target web resource. In forward mechanism, when we send a request from client to first resource then container will prepare request and response objects.

In first resource by the execution of some content some response will be generated on response object. When container encounter forward method then container will pass request and response objects to the target resource by refreshing response object that is by eliminating previous response from response object.

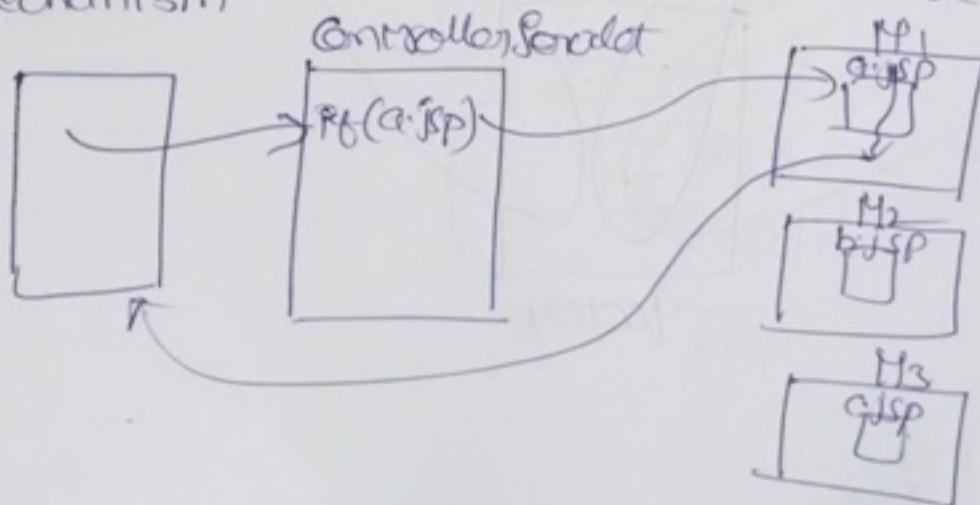
At target resource by the execution of `PT`
Content container will put the respective ~~area~~
response on the response object, at the end of
the target resource container will dispatch
the overall response to the client directly
without moving back to first resource.
Therefore in forward mechanism client is able
to receive only target resource response.



Note: In general we will utilize ~~the~~ include aspect dispatching mechanism to include logos, footers into the web pages. To achieve the above requirement if we have not used include mechanism then, each and every web resource we have to at to provide logo and footer respective coding part, it will increase code redundancy.



Note@: In general in MVC based web applications, we will use the Servlet as controller the main job of the Controller Servlet in MVC based web application is to pickup the request from client, identify the requirement of the request, identify the target resource then forward request to target resource. In the above context by the execution of the target resource some response will be generated and it will be dispatched to client directly without coming back to controller servlet. Therefore, to prepare controller Servlet in MVC based web applications, we have to use forward mechanism.



Q1 what is Servlet chaining?

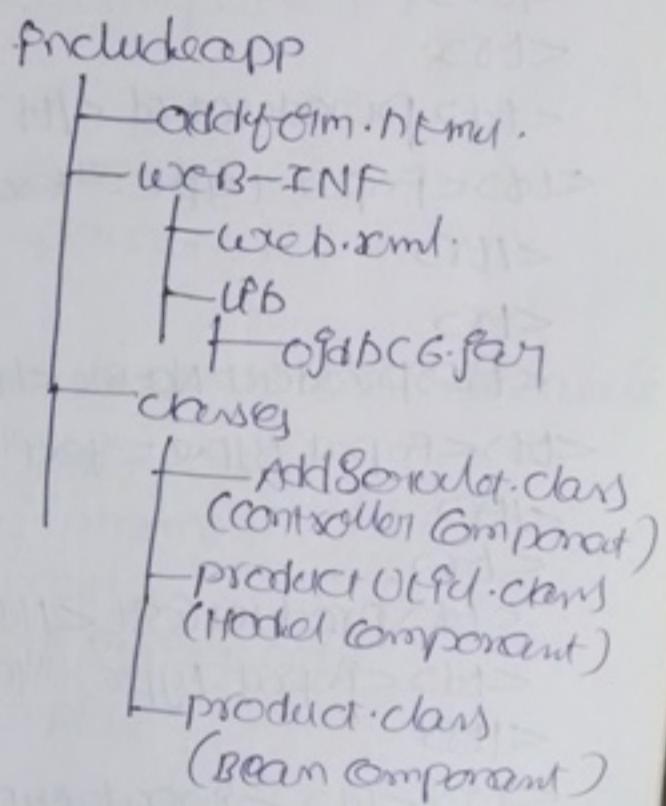
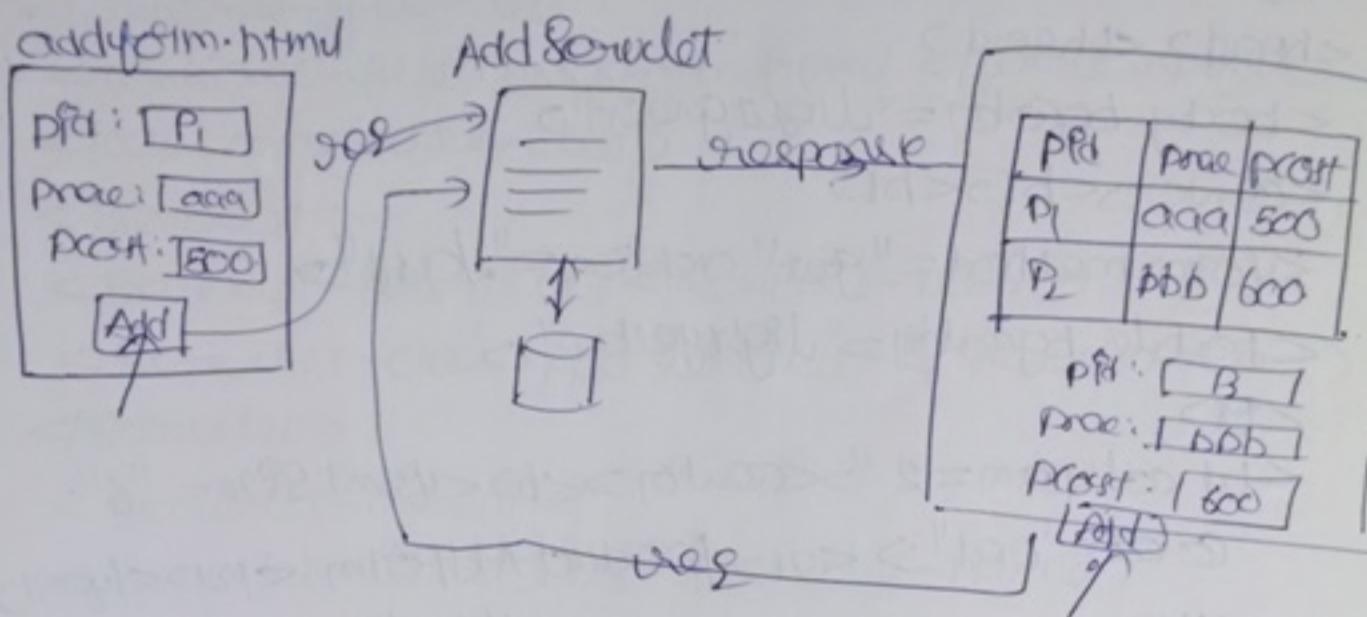
A1 The process of including more than one servlet to process a single request either by using include mechanism or by using forward mechanism is called as Servlet chaining.

Q2 what is the diff b/w forward mechanism and
Ans In web application, send Rediect mechanism?
forward request dispatcher mechanism can be used to provide the communication b/w two web resources which may be available at the same server.

In web applications, send Rediect mechanism can be used to provide the communication b/w two web resources which may be available at same server or may be available at two different servers.

To perform forward request dispatcher mechanism one request is sufficient from client. But to perform send redirect mechanism we must issue two request from client either directly or indirectly.

Date : 29th APR, 2012, Sun



addform.html

```
<html>
<head> </head>
<body bgcolor="lightgreen">
<center><br><br>
<form method="get" action=".add">
<table border="1" style="width: 100%; border-collapse: collapse; border: none; background-color: lightyellow; border: 1px solid black; border-radius: 10px; padding: 10px; margin: auto;">
<tr>
<td colspan="2" style="text-align: center; font-size: 1.2em; font-weight: bold; color: #0000ff; border: none; padding-bottom: 10px;">Product Add Form </td></tr>
<tr>
<td style="width: 15%;">Product ID </td>
<td style="width: 85%; text-align: center;">Product Name </td>
<td style="width: 85%; text-align: center;">Product Cost </td>
<td style="width: 85%; text-align: center;">
<input type="submit" value="ADD" style="width: 150px; height: 40px; border: 1px solid black; border-radius: 10px; background-color: #0000ff; color: white; font-weight: bold; font-size: 1em; cursor: pointer;"/>

```

web.xml

```
<web-app>
  <welcome-file-list>
    <welcome-file>indexform.html</welcome-file>
  </welcome-file-list>
  <servlet>
    <servlet-name>as</servlet-name>
    <servlet-class>AddServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>as</servlet-name>
    <url-pattern>/add</url-pattern>
  </servlet-mapping>
</web-app>
```

product.java (Bean component)

```
import java.io.*
```

```
public class product implements Serializable
{
    private String pid;
    private String pname;
    private int price;
    public void setPid(String pid)
    {
        this.pid=pid;
    }
    public String getPid()
    {
        return pid;
    }
}
```

```
public void setpname(String pname)
```

```
{ this.pname = pname; }
```

```
public String getpname()
```

```
{ return pname; }
```

```
public void setcost(String cost)
```

```
{ this.cost = cost; }
```

```
public int getcost()
```

```
{ return cost; }
```

```
}
```

AddServlet.java

```
import java.io.*;  
import java.util.*;  
import javax.servlet.*;  
import javax.servlet.http.*;  
public class AddServlet extends HttpServlet  
{  
    public void doget(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException  
    {  
        try  
        {  
            res.setContentType("text/html");  
            PrintWriter out = res.getWriter();  
            String pid = req.getParameter("pid");  
            String pname = req.getParameter("pname");  
            int pcost = Integer.parseInt(req.getParameter("pcost"));  
        }  
    }  
}
```

```
product() fd p4 = new product();  
ArrayList al = p4.add(pfd, pname, pcost);  
// preparing html environment  
out.println("<html>");  
out.println("<body border='1' background='yellow'>");  
out.println("<center>");  
out.println("<table border='1' border='1' background='white'>");  
out.println("<tr><td>PID </td><td> PNAME </td>");  
out.println("<td>PCOST </td></tr>");
```

23. for (Object o: al)

{ product p=(product)o;

out.println("
");

out.println("<td>" + p.getpid() + "</td><td>" +
p.getpname() + "</td><td>" + p.getpcost());

out.println("</tr>");

}

out.println("</table></td></tr></table>");
out.println("</body>");

RequestDispatcher rd = req.getRequestDispatcher()
rd.include(abc, abc);

}

catch (Exception e)

{ e.printStackTrace();

}

? JPG to PDF Converter For Mac - Unregis

productUtil class

```
import java.util.*;  
import java.sql.*;  
public class productUtil  
{ Connection con;  
  Statement st;  
  ResultSet rs; ArrayList al;  
  public productUtil()  
{  
    try {  
    }  
  }
```

```
Class.forName("oracle.jdbc.driver.OracleDriver");  
con=DriverManager.getConnection(  
"jdbc:oracle:thin:@localhost:1521:xe","system",  
"durga");  
st=con.createStatement();  
catch(Exception e)  
{  
  e.printStackTrace();  
}  
} // productUtil constructor
```

②1) public ArrayList add(String pid, String phone,
 int profit)

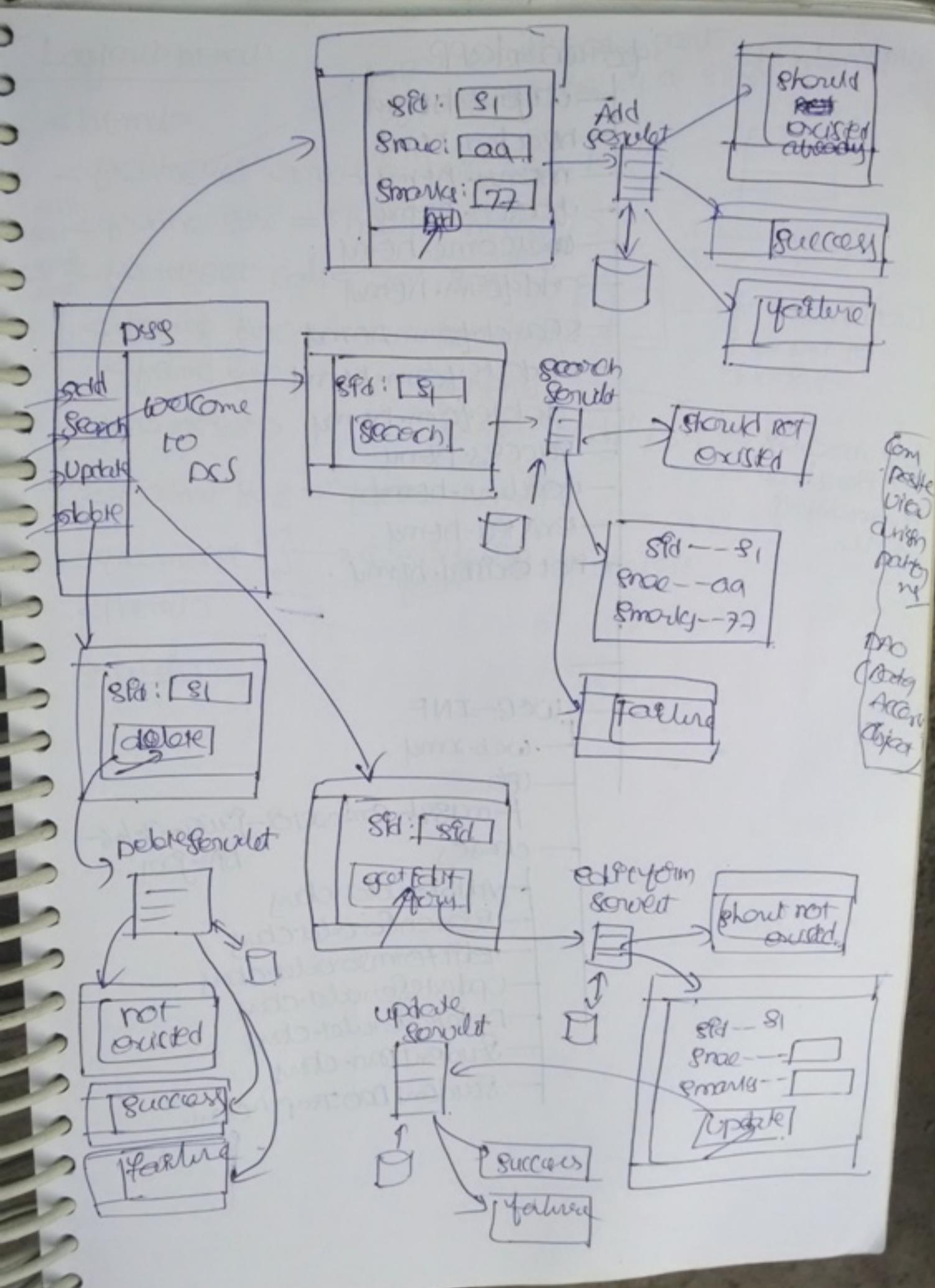
②2) st.executeUpdate("insert into product values
 (" + pid + " , " + phone + ", " + profit + ")");

②3) rs=st.executeQuery("select * from product");
 al=new ArrayList();

27

```
while (crs.next())  
    {  
        product p=new product();  
        p.setpid(crs.getInt(1));  
        p.setpname(crs.getString(2));  
        p.setpcost(crs.getInt(3));  
        al.add(p);  
    }  
    gallery  
    catch(exception e)  
    {  
        e.printStackTrace();  
        errorin al;  
    }  
}
```

C:\> set classpath=%classpath%;colorade\-----
C:\> Tomcat 7.0\--\bin\run.bat -log4j6.log
database *final
SQl create table product(
SQl > select * from product;
no rows selected
execution:
go to manager app (open browser)
click on



forwardapp

- layout.html
- header.html
- menu.html
- footer.html
- welcome.html
- addform.html
- searchform.html
- updateform.html
- deleteform.html
- success.html
- yorkine.html
- existed.html
- not existed.html

WEB-INF

- web.xml

- web

- mysql-connection-pool-3.0.6-
classes.bm.jar

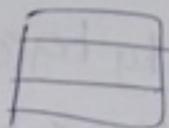
- AddServlet.class
- SearchServlet.class
- EditFormServlet.class
- UpdateServlet.class
- DeleteServlet.class
- StudentDAO.class
- StudentDOImpl.class

Layout.html:

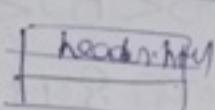
```
<html>
```

treat entire body part
of the webpage as a single block

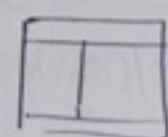
```
<frameSet rows = "20%, 65%, 15%">
```



```
  <frame src = "header.html" />
```



```
  <frameSet cols = "20%, 80%">
```



in second row
I want to
divide it.

```
    <frame src = "menu.html" />
```

```
    <frame src = "welcome.html" />
```

second row
ends with
implementation

```
  <frame src = "footer.html" />
```

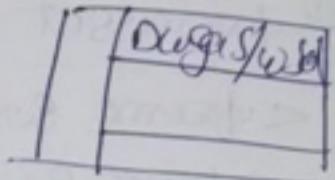
```
</frameSet> } → completing the entire  
form.
```

```
</html>
```

explain:

header.html

```
<html>
<head>
<body bgcolor="blue">
<center><b><font size="7" color="white">
Durga Software Solutions
</font></b></center></body>
</html>
```



Footer.html

```
<html>
<body bgcolor="blue">
<center><b><font size="7" color="white">
Copyright © 2010-2020 @ www.dss.com
</font></b></center></body></html>
```

menu.html

```
<html>
<body bgcolor="lightpink">
<center><b><font size="6">
<br>
<a href = "./addform.html" target = "body">Add </a>
<br><br>
<a href = "./searchform.html" target = "body">Search </a>
<br><br>
<a href = "./updateform.html" target = "body">Update </a>
<br><br>
<a href = "./deleteform.html" target = "body">Delete </a>
</font>
</b>
</center>
</body>
</html>
```

→ Where the desired form has to be displayed (which) on left/right, top, bottom, or at center.

welcome.html

```
<html> <body bgcolor="lightgreen">
<center>
<b><font size="7" color="red">
<br><br>
<marquee>
welcome To Durga Software Solutions
</marquee>
</font></b> </center> </body> </html>
```

addform.html

```
<html>
<body bgcolor="lightgreen">
<b><font size="7">
<form method="get" action="./add">
<p>
    studentID <input type="text" name="sId"/>
    studentName <input type="text" name="sName"/>
    studentMarks <input type="text" name="sMarks"/>
    <input type="submit" value="ADD"/>
</p>
</form></font></b></body>
</html>
```

searchform.html

```
<html>
<body bgcolor="lightgreen">
<b><font size="7">
<br><br>
<form method="get" action="./search">
<p>
```

add	

```
Student Id <input type="text" name="sId"/>
<input type="submit" value="Search"/>
</p>
</form></div></body></html>
```

updatedform.html

```
<html>
<body bgcolor="lightgreen">
<b><font size="7"><br><br>
<form method="get" action=".//edit">
<p>
Student Id <input type="text" name="sId"/>
<input type="submit" value="Edit" />
</p>
</form></font></b></body></html>
```

deleteform.html

```
<html>
<body bgcolor="lightgreen">
<b><font size="7"><br><br>
<form method="get" action=".//delete">
<p>
Student Id <input type="text" name="sId"/>
<input type="submit" value="Delete"/>
</p>
</form></font></b></body>
</html>
```

success.html

```
<html>
<body bgcolor = "lightyellow">
<center><b><font size="7" color="red">
success
</font>
</b></center></body></html>
```

failure.html

```
<html>
<body bgcolor = "lightyellow">
<center><b><font size="7" color="red">
failure
</font>
</b></center></body></html>
```

existed.html

```
<html>
<body bgcolor = "lightyellow">
<center><b><font size="7" color="red">
Student already existed
</font>
</b></center></body>
</html>
```

notesfigured.html

```
<html>
<body bgcolor="lightyellow">
<center><b><font size="2" color="red">
<br><br>
student not existed
</font></b></center>
</body></html>
```

web.xml

```
<web-app>
<welcome-file-list>
<welcome-file>layout.html </welcome-file>
</welcome-file-list>
<servlet>
① <servlet-name>obj </servlet-name>
<servlet-class>AkhilServlet </servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>obj </servlet-name>
<url-pattern>/add</url-pattern>
</servlet-mapping>
<servlet>
② <servlet-name>ss </servlet-name>
<servlet-class>SearchServlet </servlet-class>
</servlet>
```

③ <web-app>

 < servlet-mapping >

 < servlet-name > ss </servlet-name >

 < url-pattern > /search </url-pattern >

 </servlet-mapping >

 < servlet >

 < servlet-name > us </servlet-name >

 < servlet-class > UpdateServlet </servlet-class >

 </servlet >

 < servlet-mapping > us

 < servlet-name > us </servlet-name >

 < url-pattern > /update </url-pattern >

 </servlet-mapping >

 < servlet >

 < servlet-name > es </servlet-name >

 < servlet-class > EditFormServlet </servlet-class >

 </servlet >

 < servlet-mapping >

 < servlet-name > es </servlet-name >

 < url-pattern > /edit </url-pattern >

 </servlet-mapping >

 < servlet >

 < servlet-name > ds </servlet-name >

 < servlet-class > DeleteServlet </servlet-class >

 </servlet >

 < servlet-mapping >

 < servlet-name > ds </servlet-name >

 < url-patterns > /delete </url-patterns >

 </servlet-mapping >

</web-app>

AddServlet.java

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

public class AddServlet extends HttpServlet {
    public void doGet(HttpServletRequest req,
                      HttpServletResponse res) throws ServletException,
                      IOException {
        String sid = req.getParameter("sid");
        String sname = req.getParameter("sname");
        int smarks = Integer.parseInt(req.getParameter("smarks"));
        StudentDaoImpl dao = new StudentDaoImpl();
        String status = dao.add(sid, sname, smarks);
        if (status.equals("existed")) {
            res.getRequestDispatcher("/existed.html").forward(res, res);
        } else if (status.equals("success")) {
            res.getRequestDispatcher("/success.html").forward(res, res);
        } else {
            res.getRequestDispatcher("/failure.html").forward(res, res);
        }
    }
}
```

JPG to PDF Converter For Mac - Unregis

catch (Exception e)

{
e.printStackTrace();

}

explain:

```
SearchServlet.java
import java.sql.*;
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
public class SearchServlet extends HttpServlet {
    public void doGet(HttpServletRequest req,
                      HttpServletResponse res) throws ServletException,
                      IOException {
        try {
            String sid = req.getParameter("sid");
            StudentDaoImpl dao = new StudentDaoImpl();
            ResultSet rs = dao.search(sid);
            boolean b = rs.next();
            if (b == false)
                res.getRequestDispatcher("/notexist.html").forward(req, res);
            else {
                res.setContentType("text/html");
                PrintWriter out = res.getWriter();
                out.println('<html>');
                <body bgcolor="lightyellow">
                <b><u>Current SID = '612'</u>
                <br>
                </body>
            }
        }
    }
}
```

```
out.println("studentId" + rs.getString(1));  
out.println("studentName" + rs.getString(2));  
out.println("studentMarks" + rs.getInt(3));  
out.println("</pre></body></html>");
```

{

}

catch (Exception e)

{ e.printStackTrace(); }

{

{

explain

EditFormServlet.java

```
import java.io.*;  
import java.sql.*;  
import javax.servlet.*;  
import javax.servlet.http.*;  
  
public class EditFormServlet extends HttpServlet  
{  
    public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException  
    {  
        try  
        {  
            String sid = request.getParameter("sid");  
            Student DaoImpl dao = new Student DaoImpl();  
            ResultSet rs = dao.getStudent(sid);  
            boolean b = rs.next();  
            if (b == false)  
            {  
                request.getRequestDispatcher("/notexist.html").forward(request, response);  
            }  
            else  
            {  
                response.setContentType("text/html");  
                PrintWriter out = response.getWriter();  
                out.println("<html>");  
                out.println("  <body background='bluegreen.gif'>");  
                out.println("    <b><font size='2'>");  
                out.println("      <br>      <form method='get' action='!/update');  
                out.println("        <studentId ">" + rs.getString(1));  
                out.println("        <input type='text' value='");  
            }  
        }  
    }  
}
```

```
out.println("<input type='hidden' name='spid'  
value='"+cls.getString(1)+"'/>");  
out.println("student name <input type='text'  
name='sname' value='"+cls.getString(2)+"'/>");  
out.println();  
out.println(" student marks <input type='text'  
name='smarks' value='"+cls.getInt(3)+"'/>");  
out.println();  
out.println("<input type='submit' value='option'B');  
out.println("</p></div></form></body>  
          </html>");  
  
catch(Exception e)  
{  
    e.printStackTrace();  
}  
}  
  
explain
```

UpdateServlet.java

```
import java.io.*;  
import javax.servlet.*;  
import javax.servlet.http.*;  
public class UpdateServlet extends HttpServlet  
{  
    public void doGet(HttpServletRequest req,  
                      HttpServletResponse res) throws ServletException,  
                                              IOException  
    {  
        try  
        {  
            String sid = req.getParameter("sid");  
            String sname = req.getParameter("sname");  
            int smarky = Integer.parseInt(req.getParameter  
                                         ("smarky"));  
            StudentDAOImpl dao = new StudentDAOImpl();  
            String status = dao.update(sid, sname, smarky);  
            if (status.equals("success"))  
            {  
                res.getRequestDispatcher("/success.html").  
                    forward(req, res);  
            }  
            else if (status.equals("failure"))  
            {  
                res.getRequestDispatcher("/failure.html").  
                    forward(req, res);  
            }  
        }  
        catch (Exception e)  
        {  
            e.printStackTrace();  
        }  
    }  
}
```

DeleteServlet.java

```
import java.io.*;  
import javax.servlet.*;  
import javax.servlet.http.*;  
public class DeleteServlet extends HttpServlet  
{  
    public void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException  
    {  
        try  
        {  
            String id = req.getParameter("id");  
            StudentDAOImpl dao = new StudentDAOImpl();  
            String status = dao.delete(id);  
            if (status.equals("not existed"))  
            {  
                res.getRequestDispatcher("/notexisted.jsp").forward(res);  
            }  
            if (status.equals("success"))  
            {  
                res.getRequestDispatcher("success.html").forward(res);  
            }  
            if (status.equals("failure"))  
            {  
                res.getRequestDispatcher("failure.html").forward(res);  
            }  
        }  
        catch (Exception e)  
        {  
            e.printStackTrace();  
        }  
    }  
}
```

Student Dao.java

```
import java.sql.*;  
public interface Student Dao  
{  
    public String add(String sid, String sname, int  
    public ResultSet getStudent(String sid);  
    public String update(String sid, String sname, int  
    public String delete(String sid);  
    public ResultSet search(String sid);  
}
```

Student DaoImpl.java

```
import java.sql.*;  
public class Student DaoImpl implements Student Dao  
{  
    Connection con;  
    Statement st;  
    ResultSet rs;  
    String sqlkey = "";  
    public Student DaoImpl()  
    {  
        try  
        {  
            Class.forName("com.mysql.jdbc.Driver");  
            con = DriverManager.getConnection(  
                "jdbc:mysql://localhost:3306/dungadb", "root", "root");  
            st = con.createStatement();  
        } catch (Exception e)  
        {  
            e.printStackTrace();  
        }  
    }
```

```
public ResultSet getStudent(String sid)
```

```
{ try
```

```
{
```

```
rs=st.executeQuery("select * from student where  
sid ='" + sid +"');
```

```
} catch(Exception e)
```

```
{ e.printStackTrace();
```

```
}
```

```
} returning;
```

```
public String add(String sid, String sname, String smark)
```

```
{ try
```

```
{ rs=getStudent(sid);
```

```
boolean b=rs.next();
```

```
if(b==true)
```

```
{ status="exists";
```

```
else
```

```
int rowCount=st.executeUpdate("insert into  
student values ('" + sid + "', '" + sname + "', '" + smark + "')");
```

```
if(rowCount==1)
```

```
{ status="success";
```

```
}
```

```
else
```

```
{ status="failure";
```

```
}
```

```
} catch(Exception e)
```

```
{ status="failure";
```

```
e.printStackTrace();
```

```
}
```

```
return status;
```

```
public ResultSet search(String sid)
{
    return getStudent(sid);
}
```

```
public String update(String sid, String name,
                     int smarks)
{
    int rowCount = st.executeUpdate("update student
        set name = " + name + ", smarks = " + smarks +
        " where sid = " + sid + ")");
    if (rowCount == 1)
        status = "success";
    else
        status = "failure";
}
catch (Exception e)
{
    status = "failure";
    e.printStackTrace();
}
return status;
}
```

```
public String delete(String sid)
{
    ResultSet rs = getStudent(sid);
    boolean b = rs.next();
    if (b == true)
        rowCount = st.executeUpdate("delete from student
            where sid = " + sid + ")");
}
```

```
if (totalCount == 1)
    {
        status = "Success";
    }
else
    {
        status = "Failure";
    }
else
    {
        status = "not existed";
    }
}
catch (Exception e)
{
    status = "Failure";
    e.printStackTrace();
    return status;
}
```

c:\> set classpath=%classpath%; --mySQLconnector
java -jar hibernate.jar
c:\> Tomcat7\bin\cmd\bin> java *:9002
--check whether the database is ready or not
mysql> use db;
 > show tables;
 > create table student(id char(5), name char
 > commit;
 > select * from student;
 > desc student;

now open the browser

browser

click on forwardapp

JPG to PDF Converter For Mac - Unregis

Date: 30th April, 2012 Mon

Applet to servlet communication:- In general, in web appl, we will prepare web resources at server machine like servlets, JSP's and so on. In this we will access that web resources by sending a request from browser. i.e. by providing an url at browser address bar.

When we provide an url at browser address bar then browser will send a request to the server, where protocol will pickup that request, establish a virtual socket connection & carry that request data to the server.

After executing the web application at serverside the respective response will be generated and protocol will carry that response to the client (i.e. browser).

In general we will prefer to use applets to define templates and to design front pages and so on in web applications inorder to improve look and feel.

In the above context, if we use an applet in place of browser and if we send a request from an applet to the Servlet available at server machine then it is called as applet to servlet communication.

In case of applet to servlet communication, after designing the serverside application, we need to own client side applet, where is GUI part is created with the GUI components like text fields, buttons, check boxes and so on,--

where if we click on any button then automatically we have to send a request to respect the server side servlet. By executing servlet the generated response has to be send to the same applet from where we are getting request.

If we want to provide applet to servlet communication in our web applications we have to use the following steps:

Step①: prepare URL:

In applet to servlet communication, we need to prepare an URL to represent a server side resource, for this we have to create java.net.URL class object. To prepare URL class object, we have to use the following constructor:

public URL(String url)

Ex: String url = "http://jh:8080/JogithApp/Jogith?
username=" + tf.getText() + "uprod=" + tf.getText();
URL u = new URL(url);

Step②: establish the connection b/w client applet and servlet.

To establish the connection b/w client applet and the respective servlet available at server machine then we have to use the following method from URL class:

public URLConnection openConnection()

Ex: URLConnection con = u.openConnection();

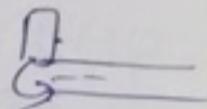
Step③ : Send request to the Server

After establishing the connection we have to send a request to the server as per the url which we prepared, for this we have to use the following method

(public void setDoInput(boolean b))

where 'b' must be true

~~obj: con.setDoInput(true);~~



Note: With the above steps automatically, a request will be sent to server from applet, where Servlet will process the request, identify the respective ^{new} Servlet, execute it and generate the dynamic response to the applet. Now, the response is available with URLConnection object.

Step④ : getInputStream from URLConnection:

To get InputStream from URLConnection object, we have to use the following ~~obj.get~~ method

(public InputStream getInputStream())

InputStream is = con.getInputStream();

Step⑤ : convert InputStream into BufferedReader and read the dynamic response from BufferedReader

BufferedReader br = new BufferedReader

(new InputStreamReader(is));

String response = br.readLine();

Why Project Stale com Reader?
Why Buffered Reader?

At Server

logfinapp

— WEB-INF

— web.xml

— classes

— LoginServlet.java

At Client Side

display

— LoginApplet.java

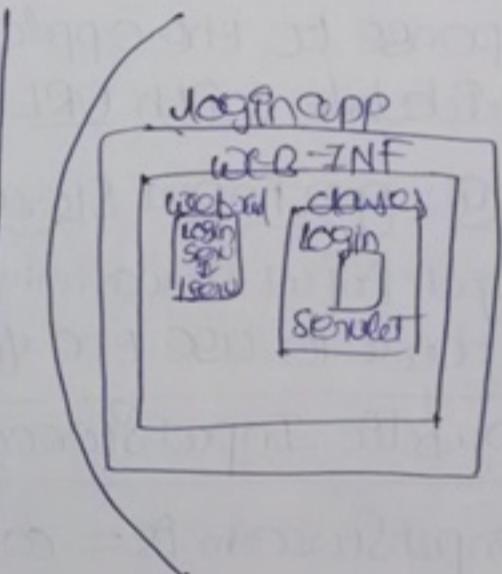
— LoginApplet.class

— LoginApplet.html

Diagram:

• Login Applet

Name:	<input type="text"/>
Pass:	<input type="password"/>
<input type="button" value="Login"/>	
(Login Success)	



At Server:

LoginApp.java

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

public class LoginServlet extends HttpServlet
{
    public void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException
    {
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
        String uname = req.getParameter("uname");
        ("upwd");
        if(uname.equals("durga") && upwd.equals("durga"))
        {
            out.println("Success");
        }
        else
        {
            out.println("failure");
        }
    }
}
```

Compilation:

web.xml

```
<web-app>
  <servlet>
    <servlet-name>
      <servlet-class>
        <!-->
    </servlet>
    <servlet-mapping>
      <servlet-name>
        <url-pattern>
          <!-->
        </url-pattern>
      </servlet-mapping>
  </web-app>
```

At client side:

LoginApplet.java

```
import java.awt.*; import java.net.*;
import java.awt.event.*; import java.io.*;
import java.applet.*;

public class LoginApplet extends Applet {
  JTextField tf1, tf2;
  Button b1;
  Label l1, l2;
  String strkey = "";
  public void init()
  {
    this.setBackground(Color.pink);
    this.setVisible(true);
    l1 = new Label("UserName");
    l2 = new Label("password");
```

COMPONENT

```
tf1 = new JTextField(20);
tf2 = new JTextField(20);
tf2.setEchoChar('*');
b = new JButton("Login");
b.addActionListener(this);
this.add(c1); this.add(tf1);
this.add(c2); this.add(tf2);
this.add(b);
}
```

```
public void actionPerformed(ActionEvent ae)
{
    try
    {
        String user = "http://localhost:10101/loginapp/login?
username=" + tf1.getText() + "&password=" + tf2.getText();
        URL u = new URL(user);
        URLConnection uc = u.openConnection();
        uc.setDoInput(true);
        InputStream ps = uc.getInputStream();
        BufferedReader br = new BufferedReader(new
InputStreamReader(ps));
        String status = br.readLine(); repaint();
    }
    catch(Exception e)
    {
        e.printStackTrace();
    }
}
```

```
public void paint(Graphics g)
{
    Font f = new Font("arial", Font.BOLD, 30);
    g.setFont(f);
}
```

JPG to PDF Converter For Mac - Unregister

compilation:

(a) compilation command -
 (b) buildroot command -
 (c) runroot command -
 (d) logon command -
 (e) runroot command -
 (f) buildroot command -

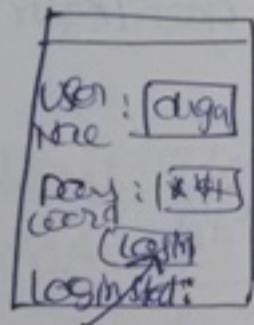
LoginApplet.html :-

```
<applet code = "LoginApplet" width="500" height="500">
```

— We must start the browser

CD D:\inetpub\wwwroot\Startup.html

— Double > appletviewer LoginApplet.html



when we click on the button [Login]
some security exception will be
raise on the command prompt. why?
this because the [Java] some systems
are not supporting this communication.

— But some systems may support this communication.

Security is not clear yet
demo ati apn room

Date: 1st May, 2012 Tue

Steps to design J2SE application with eclipse IDE:-

Step 1: Create Java Project

File → New → Java Project → Next
File → New → Other → Java Project → Next
If we click on Next button, a window will be open to take project details.

Create a Java Project

Project name: app1

use default location.

JRE:

Choose an execution environment:

use project specific J2SE 1.7

use default JRE (currently J2SE)

Project Layout:

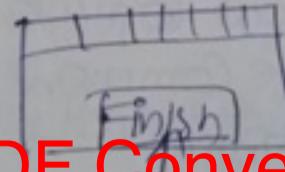
use project as root for source and folder class files

create separate folder for source and class files

Working Sets class files

Add project to working sets.

Next Finish



If we do the above step automatically a project with the name app2 will be created in package (that is left side frame) explorer

Step②: create a Java class with main method

Right click on Superject(app) → new → class
→ if we do the following the subdirectory above automatically should be unindexed. A window will be open to take class details.

Source folder: [app1/src] [browse]

Package: [com.deg] [browse]

enclosing type.

Name: [First]

Modifiers: public default

abstract final

Superclass: [java.lang.Object] [browse]

Interfaces:

[] [Add]

which method stubs would you like to create?

public static void main(String[] args)

Constructors and super class.

Inherited abstract methods.

Do you want to add comments?

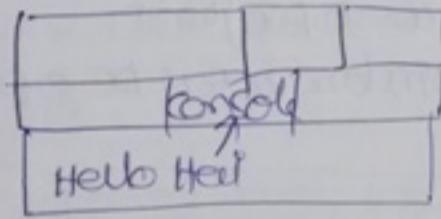
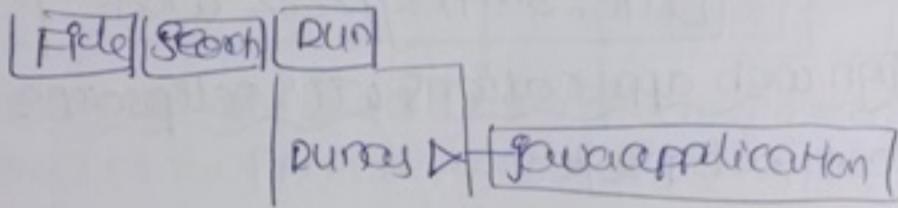
Generate Comments

[Finish]

DAPPL	class first	
DCmds.java	i =	
DFirst	y	

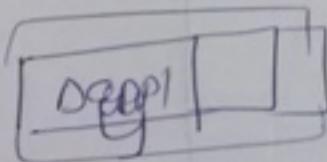
→ If we do the above first Java file will be created in the specified package under src folder and it will be open in work area. That is the middle frame.

Step③ : Run Java application



If we do the above application will be executed and the required output will be generated on the console.

Note: If we want to design any JDBC application then we have to use the above steps as it is, but we need to add driver specific jar file to our project specific library, for this we have to use the following path.



right click on project

Properties

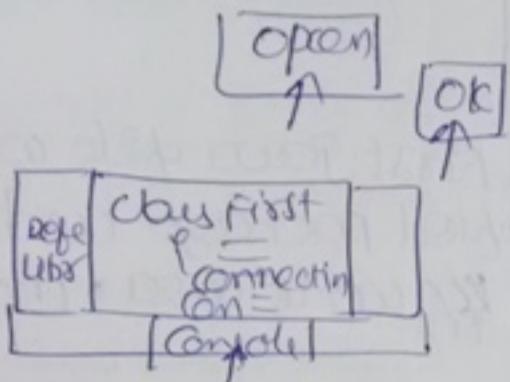
Java build path

Libraries

Add external jars

JPG to PDF Converter For Mac - Unregis

and select required gear file ojdbc14.jar @
ojdbc5.jar @ ojdbc6.jar



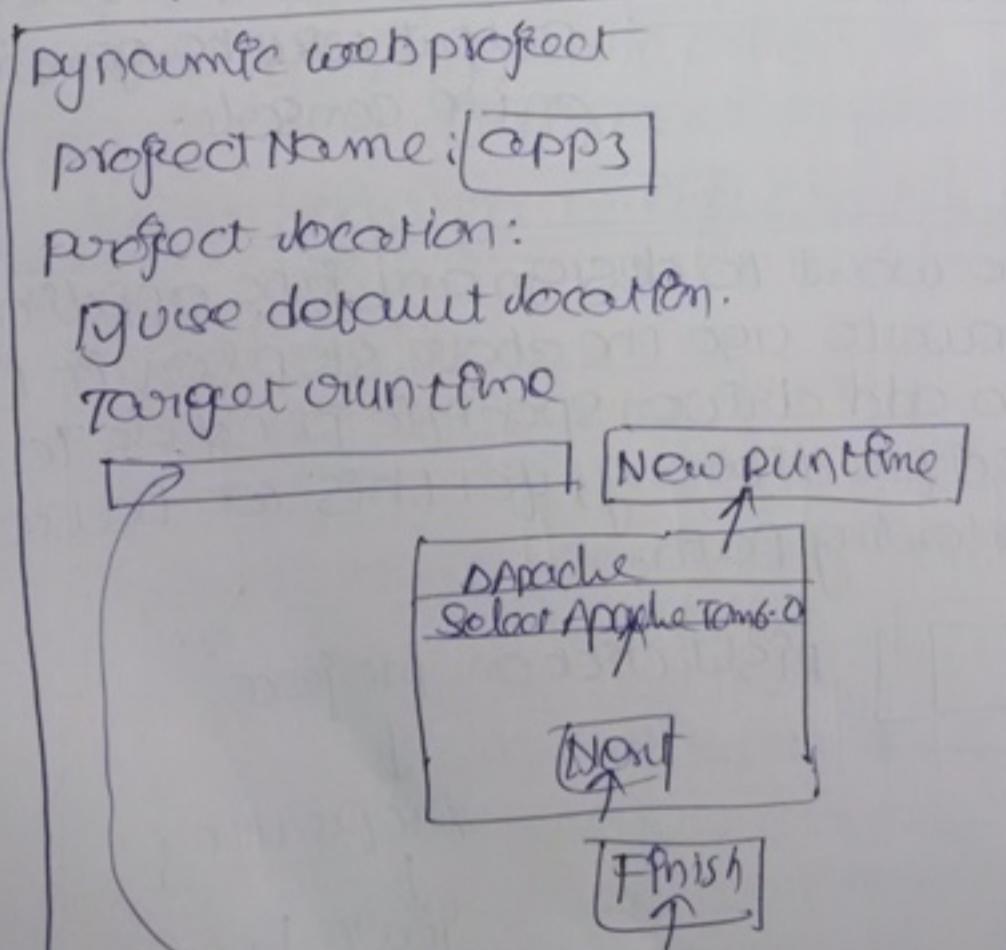
Date: 2nd May, 2012 wed

Steps to design web applications with eclipse IDE.

Step 1: prepare web project

file → New → Dynamic web project

If we do the above a window will be open to set web project details.



JPG to PDF Converter For Mac - Unregis

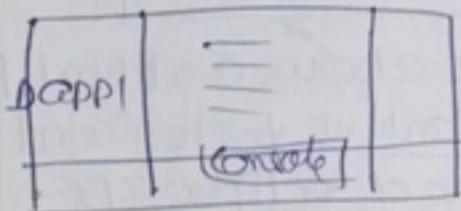
Dynamic web module creation - [2.5]
Configuration:

Default configuration for apache
Tomcat 6.0

[Host]

[NOMINATE]

web module
[Finish]



If we do the above automatically web project will be created with the name app3 in package explore

Step②: prepare web resources like Servlets, html, JSPs

> app3

To prepare html page, we have to use the following path Right click on project (app3) → new → html file

app3
[WebContent]
File Name: DogPerformance.html
[Next] [Finish]

If we do the above a window will be open to take the name and location of html page.

where provide file name

Select HTML templates

Use HTML templates

Name

Description

New HTML(S)

HTML(S)

Finish

Destination

app1

src/main/webapp

index.html

! DoctyBe htdc

<html>

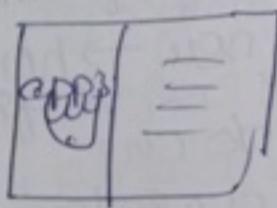
<head>

=

<form method="post" action="index.htm"

If we do the above an HTML page will be created under webContent folder and it will be open in the middle frame of Eclipse IDE. Type the page template logic in the HTML page what you want to design. And save the pgm by pressing ctrl+s.
(Don't forget to press ctrl+s after writing the logic)

To create a Servlet, we will use the following path



Right click on project(APP1) → New

If we do the above, a window will be created to take servlet details.

↓
servlet

Create Servlet

Project: APP1

Source folder: APP1/SRC

base

Java package: com.dss

Class name: LoginServlet

Super class: javax.servlet.http.HttpServlet

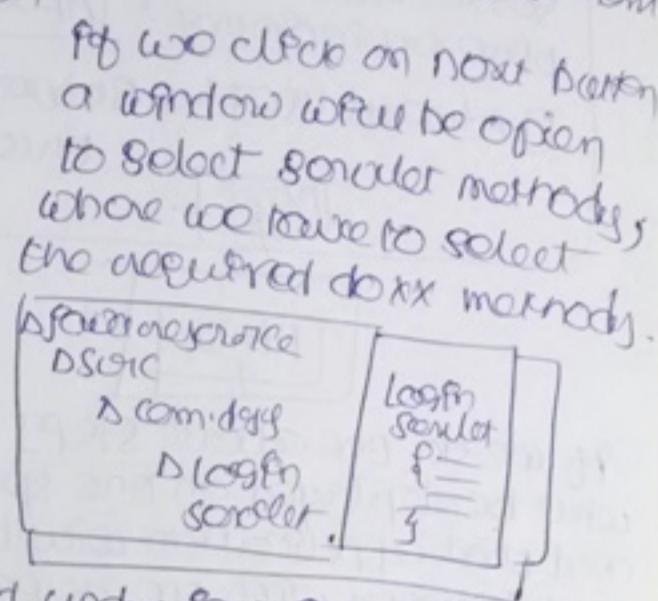
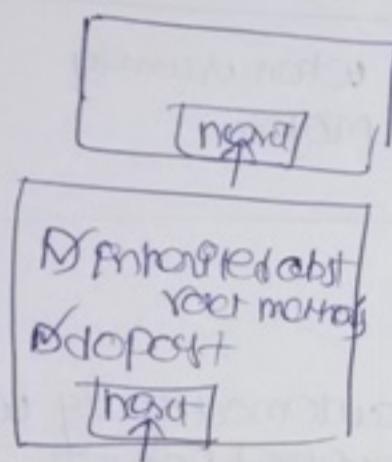
Use existing Servlet class or JSP -

next

finish

JPG to PDF Converter For Mac - Unregis

If we click on next button, a window will be created to take Initialization parameters and url mappings. Where edit the url pattern as per the requirement by click on edit button.



If we do the above step a server will be created under Java Sources / src / com/dsfd and it will be open in the middle frame of IDE. Here provide the implementation class and after that do Ctrl + S.

2 Step 3 : Run the web application

Right click on project (app) → Run → Run on If do above automatically a window Server will be open to specify server details.

Run on Server

How do you want to select the Server?

- Select an existing server
- Manually define a new server

Select the server type.

Apache

Tomcat 7.0 Server

Server's host name: Local host

Server name: Tomcat V6.0 at C:\localhost\

Server run

OS environment: Apache Tomcat V6.0

- Always use this server when running this project

Next
↓

Finish
↓

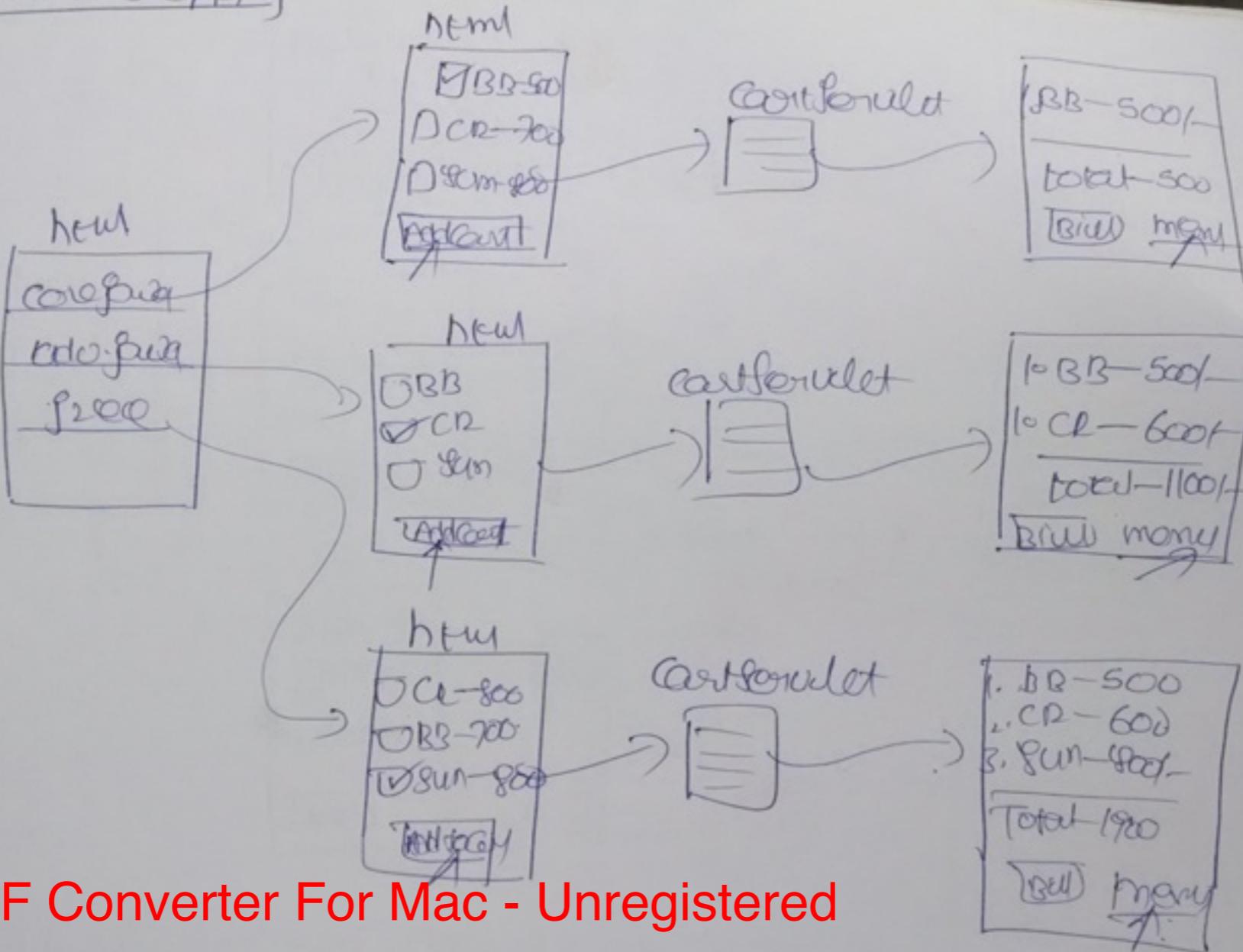
If we do the above steps automatically web app will be deployed on the specified server and that application will be accessed by open web browser with the respective url automatically.

	<u>http://127.0.0.1:8080/APP1/dashplatform.html</u>
<u>red cobr</u>	<u>username: Hung</u>
<u>≡</u>	<u>password: 123456</u>
	<u>Login</u>
	<u>connect</u>

Session tracking mechanism

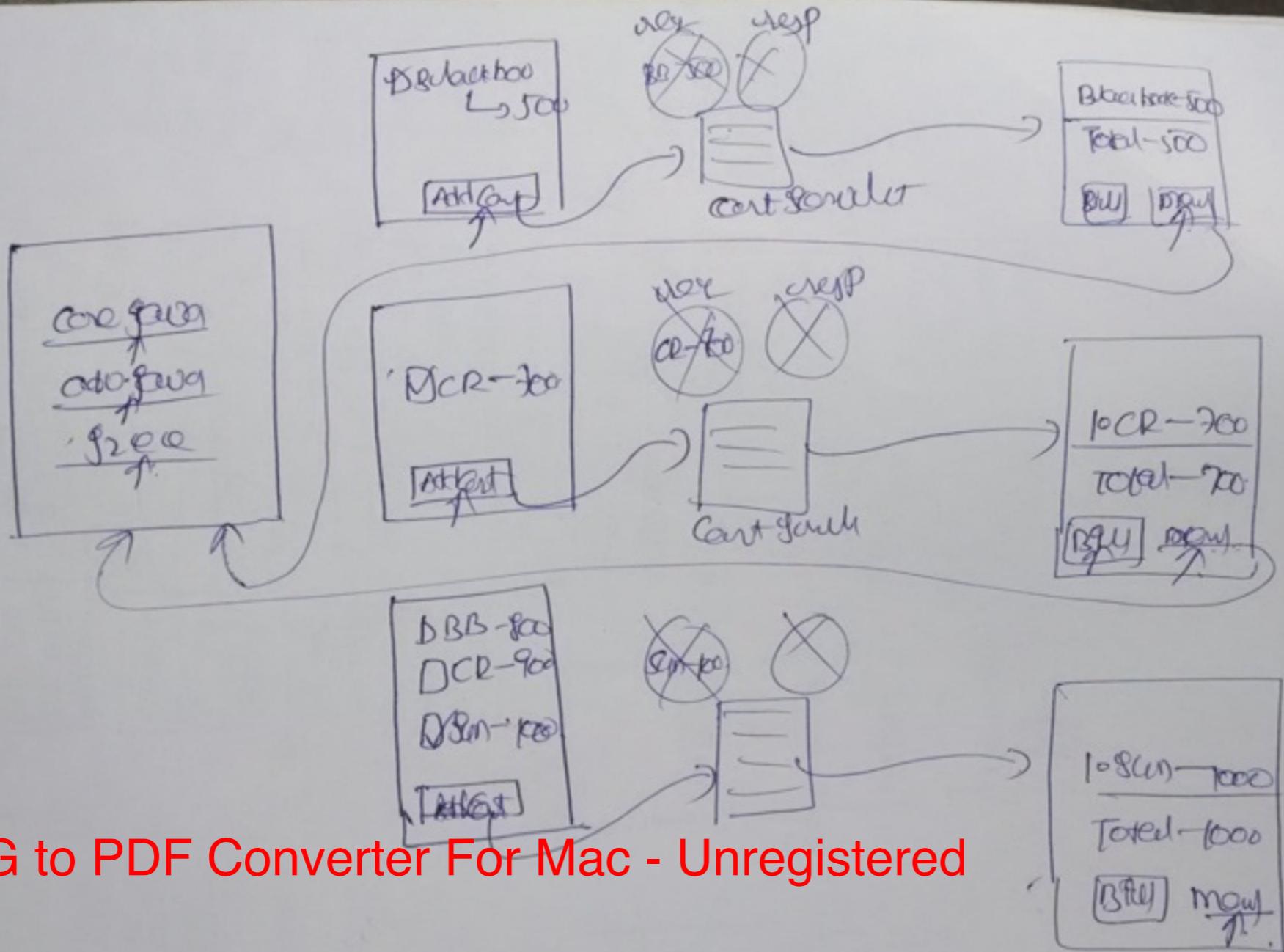
Date: 6th Apr, 2012

online shopping



JPG to PDF Converter For Mac - Unregistered

Project objects

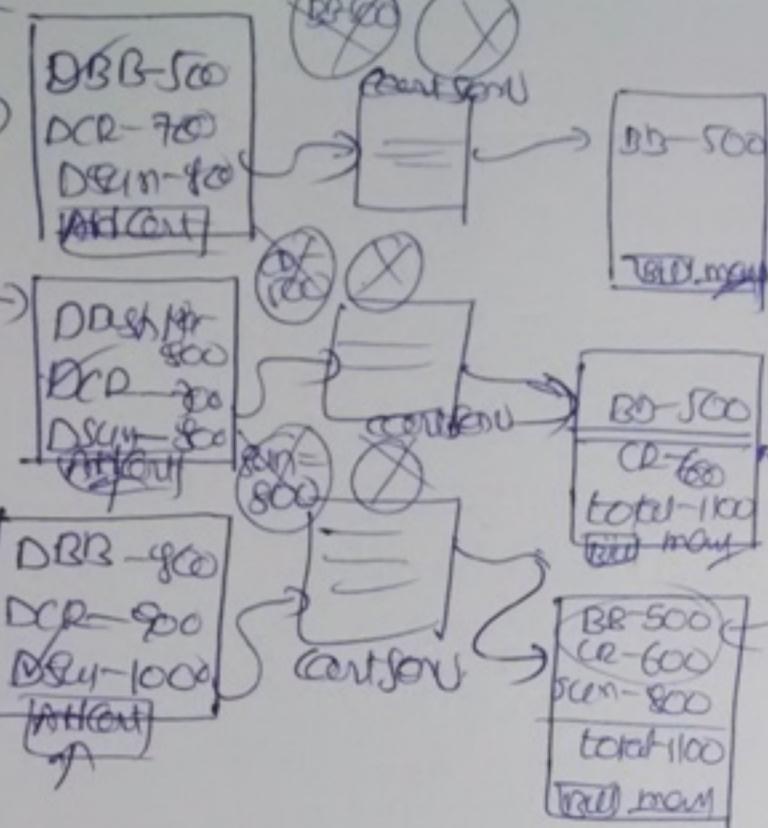


JPG to PDF Converter For Mac - Unregistered

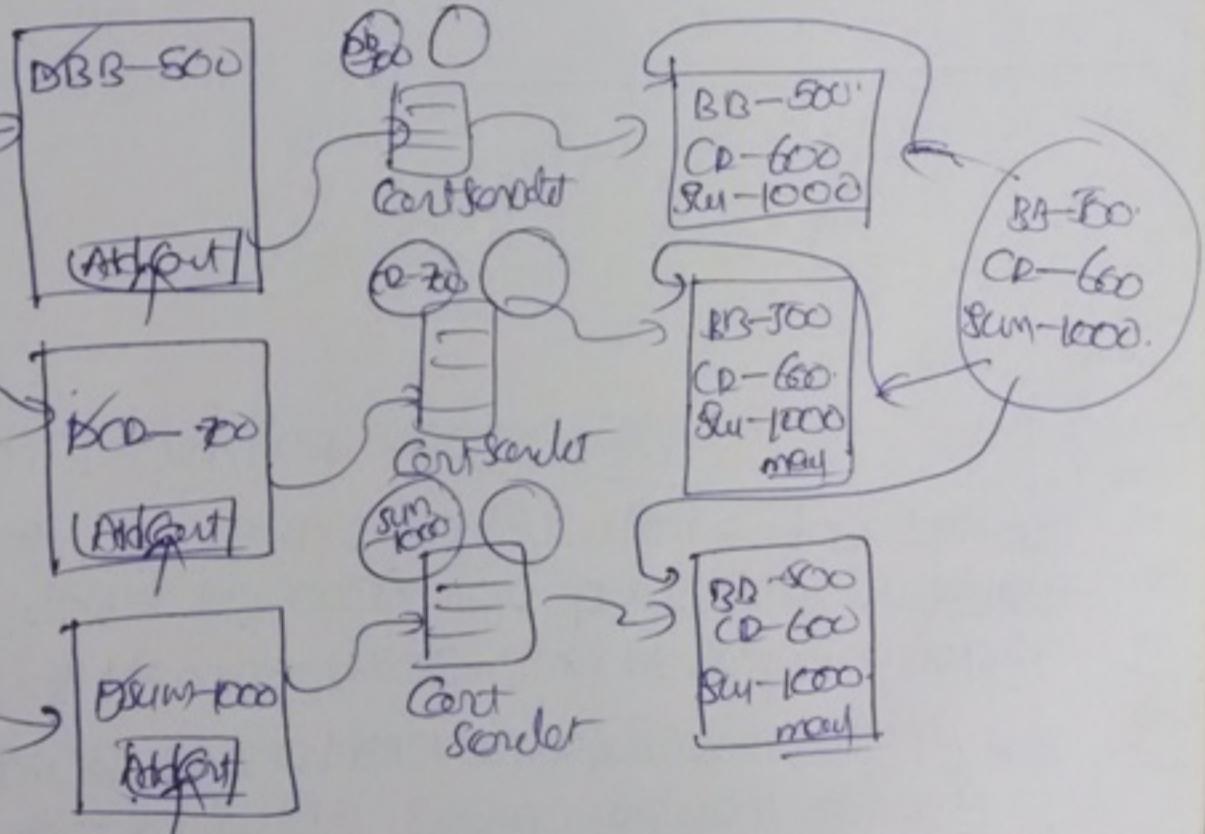
UPS is the
prime delivery mode

Carry
Trade
Business

Free



more
dijpu
free



Session

Session
using
cookies
not
better
centralized
with
clients
Pop
out
User
a
get
ab
me
and
can
Pr
S/
more
for
ex
ye
un
on
ok

Session Tracking mechanisms:

As part of the web application development it is required to manage previous request data at the time of processing later request.

- To achieve the above requirement If we use request objects then request objects will be created and destroyed when we send a request and when we get the response.
- Due to above reason request objects won't be available upto the present request processing, which are not available in the later request processing.

To achieve the application requirement if we use ServletContext object then we are able to manage previous request data at the time of processing later requests, because ServletContext object will share its data through any number of requests.

— ServletContext object is able to achieve our application requirement but it is unable to provide clear cut separation b/w multiple number of users; because ServletContext Object will share its data to any number of users and any number of requests.

In the above context to manage client's previous request data at the time of processing later request and to provide clear cut separation b/w multiple users, we have to use a set of mechanisms explicitly called as Session Tracking mechanisms.

Session: Session is a time duration, it will start at the starting point of the client conversation with the server and it will terminate at the ending point of client conversation with the server.

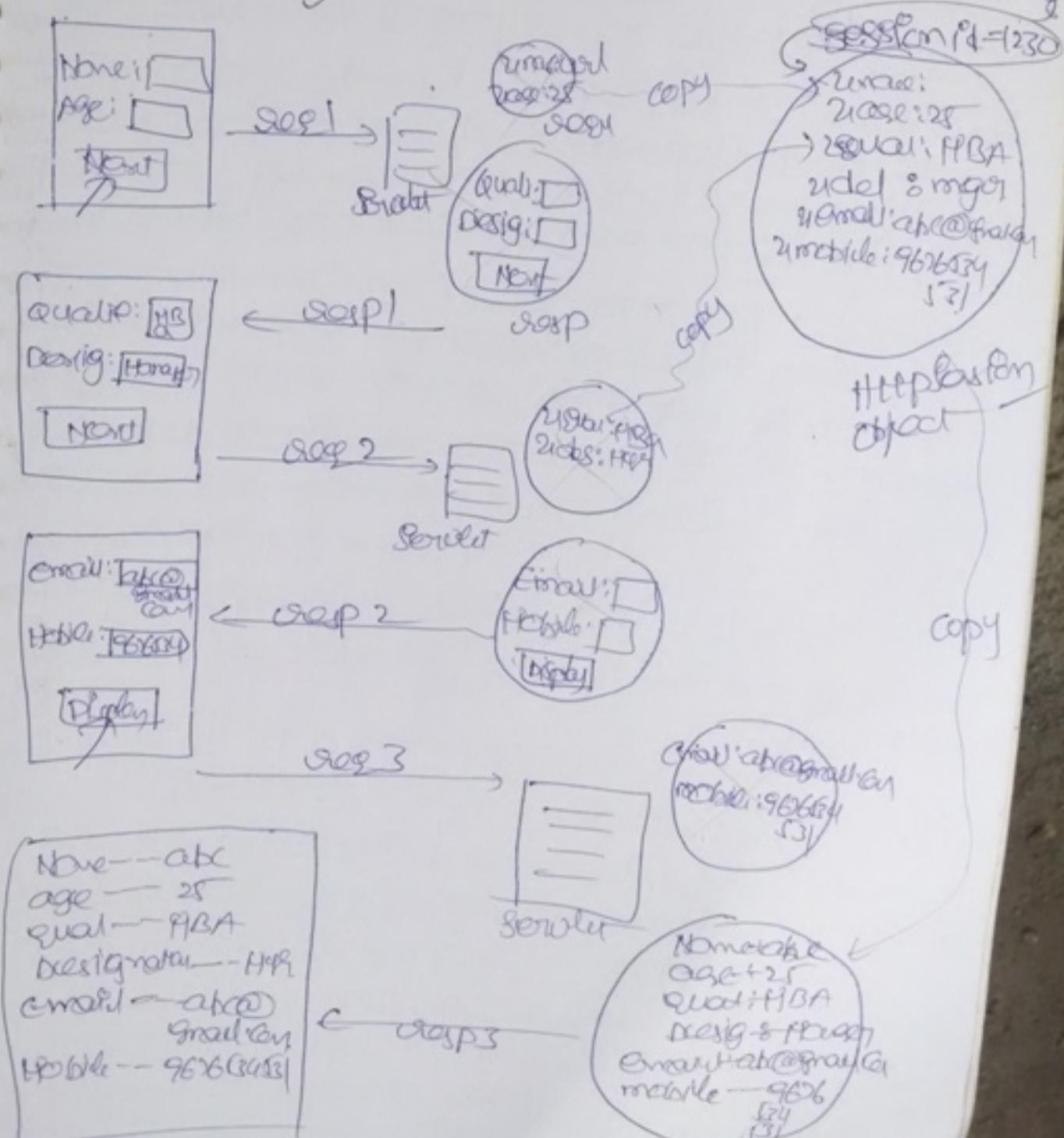
The data which we send through one or more number of request from a single client during a particular session is called state of the session.

— In web application for each and every user a separate session will be created, where to keep track all the sessions we have to use a set of mechanisms at server side.

JPG to PDF Converter For Mac - Unregis

- In web applications, there are four types of session tracking mechanisms.

- official
mechanism
- ① HttpSession Session tracking mechanism
 - ② Cookies Session tracking mechanism
 - from sur ③ URL Rewriting Session tracking mechanism
 - ④ Hidden form fields Session tracking mechanism
developers creation



In the above mechanisms, the first three mechanisms are provided by Seroulet API. But the last mechanism is developer creation.

In http Session session tracking mechanism, we will create a separate HttpSession object for each and every user.

- In case of HttpSession session tracking mechanism at each and every request processing we will get the data from request object and we will send that data to HttpSession object as attribute.
- If we repeat the above process for each and every request then it is possible to collect previous request data at the time of processing later requests directly from HttpSession object.
- In web applications, to represent HttpSession object servlet API has provided a predefined interface in the form of java.servlet.http.HttpSession.
To get HttpSession object, we HttpSession. We will use the following two methods.
 - ① getSession() method
 - ② getSession(true) method ✓

What is the diff b/w getSession() and getSession(true)?

Ans: HttpSession ~~ses = req.getSession();~~ ^{method (true);}
HttpSession ~~ses = req.getSession();~~ ^{method (true);}

In web applications, both the methods can be used to get HttpSession object. To get HttpSession object, if we use getSession() method then JSP container will search whether any HttpSession object is existed or not, with respect to the user.

If Any HttpSession object is existed with respect to the user then Container will return that object reference. If no HttpSession object is existed with respect to the user then Container will prepare a new HttpSession object and return its reference.

public HttpSession getSession()

out HttpSession ses = req.getSession();

→ To get HttpSession object, By we use getSession() method. Then Container will check whether any HttpSession object is existed or not with respect to the user, If any HttpSession object is existed then Container will return that object reference. If no HttpSession object is existed with respect to the user, then Container will return null value from getSession() method without creating new HttpSession object.

public HttpSession getSession(boolean b)

out HttpSession ses = req.getSession(value);

Note: getSession(boolean) method functionality is almost all same as getSession() method.

How to set data to a HttpSession object:-

To set an attribute onto the HttpSession object we will use the following method

public void setAttribute(String name, Object value)

- To get a particular attribute value from HttpSession object we have to use the following method

public Object getAttribute(String name)

- To remove an attribute from HttpSession object we will use the following method

public void removeAttribute(String name)

- To get all attribute names, from HttpSession object we will use the following method

public Enumeration getAttributeNames()

- In web applications, to destroy HttpSession object, we will use the following method from HttpSession. This approach contains

public void invalidate() ~~to immediately~~ ~~destroy the HttpSession object~~

- If we want to destroy HttpSession object after the specified time duration, we have to use the following methods. (Idle Time)

public void setMaxInactiveInterval(^{in seconds} int time)

- To get the specified Max Inactive Interval time ^{the above method} we have to use the following method. (Idle time)

public int getMaxInactiveInterval()

But See:

In web applications, In HttpSession session tracking mechanism for every user we will create a separate HttpSession object. In this context how Container will decompose user specific HttpSession object among multiple number of HttpSession objects when it receives request from client?

Ans In case of HttpSession session tracking mechanism, when we create a HttpSession object then Container will create an unique identification called as SessionId.

— Container will prepare SessionId in the form of a cookie. Cookie is a small object able to store one name-value pair.

Only

The default nature of the Cookie is to transfer cookie from server to client along with the response automatically and to transfer the cookie from client to server along with request automatically when we send the further request from the same client.

— Due to the above specific nature of the cookies, SessionId cookie will carry the SessionId value from server to client along with the response automatically and it will carry the same SessionId value from client to server along with the request automatically when we send further request from the same client.

In the above context, if we access getSession() method or getSession(name) method contains, we'll get SessionId cookie from HttpServletRequest object and on the basis of SessionId value, container will identify user specific HttpSession object among multiple no. of HttpSession objects.

- To get the generated SessionId value, we have to use the following method from HttpSession.

public String getId()

Cookie
function
HttpSession
getId()

the
client
from
server
at a
time
all
the
users
give
order
to
him
he put
one
no
on
each
table
How
he
was
give
this
?.

JPG to PDF Converter For Mac - Unregistered

form1.html

name: abc
age: 25
button: Next

client

Name — abc
Age — 25
Qualification: MBA
Designation: Mgr
Email: abc@gmail.com
Mobile: 9676134531

user1
age=25

firstForward.java

```
String surname = req.get("surname");
int age = Integer.parseInt(req.getParameter("age"));
HttpSession hs = req.getSession();
hs.setAttribute("name", surname);
hs.setAttribute("age", age);
RequestDispatcher rd = req.getRequestDispatcher("/form2.html");
rd.forward(req, res);
```

form2.html

Qualification:	<input type="text"/>
Designation:	<input type="text"/>
Ruby	

client

user1, HSB
rides:HGS1

secondForward.java

```
String surname = req.getAttribute("surname");
String rides = req.getAttribute("rides");
HttpSession hs = req.getSession();
hs.setAttribute("age", age);
ts.setAttribute("rides", rides);
RequestDispatcher rd = req.getRequestDispatcher("/form3.html");
rd.forward(req, res);
```

form3.html

Email: abc@gmail.com
Mobile: 9676134531
Output

client

displayForward.java

```
String email = req.getAttribute("email");
String mobile = req.getAttribute("mobile");
HttpSession hs = req.getSession();
String name = hs.getAttribute("name");
String age = hs.getAttribute("age");
String qualification = "visual";
String designation = "opx";
op("Email—" + email);
op("Mobile—" + mobile);
```

```
op("Email—" + email);
op("Mobile—" + mobile);
```

To specify idle time for the HttpSession object, it is possible to use in declarative approach also. In declarative approach, we will use the following XML tag in web.xml file.

<web-app>

≡

<session-config>

<session-timeout>time in minutes

/session-config>

</session-timeout>

≡

</web-app>

The
sessi
on w
ill b
e cle
ared
after
some
time
when
object
is
not
used
for
some
time
now
it
will
be
cle
ared
by
the
objec
t
in
old
one)

form1.html

```
<html>
<body bgcolor="#lightgreen">
<center><br><br>
<form method="post"
      action=".\\first">
<table border="1" bgcolor="#lightyellow">
<tr><td colspan="2"><center><br><b>
<font size="6" color="red"><u>Registration Form</u>
</u> </font><|b></center></td></tr>
<tr>
<td>User Name</td>
<td><input type="text" name="username"/></td>
<tr>
<td>User Password </td>
<td><input type="password" name="password"/></td>
<tr>
<td><input type="submit" value="NEXT"/></td>
</tr>
</table></form></center></body></html>
```

http://asfonapp/

```
   + form1.html
   + form2.html, form3.html
   + WEB-INF
     + web.xml
     + classes
       + FirstServlet.class
       + SecondServlet.class
       + DisplayServlet.class
```

form2.html:

```
<html>
<body bgcolor="#lightgreen">
<center><br><br>
<form method = "get" action = ". / second">
<table border="1" bgcolor="#lightyellow">
<tr>
<td colspan="2" style="text-align: center; font-size: 16px; color: red"><u> Registration Form (cont) </u></td>
<tr>
<td colspan="2" style="text-align: center; font-size: 14px;">Qualification
<td><input type="text" name="qual"/></td>
<tr>
<td colspan="2" style="text-align: center;">Designation
<td><input type="text" name="udel1"/></td>
<tr>
<td colspan="2" style="text-align: center;"><input type="submit" value="NEXT"/>
</tr>
</table></form></center></body></html>
```

form3.html:-

```
<html>
<body bgcolor="#ffff00">
<center><b><b>
<form method="get" action="display">
<table border="1" style="width:100%; border-collapse: collapse; text-align: center;">
|  |  |
| --- | --- |
| Registration Form (Continue) | |
| Email ID </td>         <td> <input type="text" name="email"/> </td>         <br>         <td> <input type="text" name="mobile"/> </td>         <br>         <td> <input type="submit" value="Display" /> </td> | |

</table>
</form>
</center>
</body>
</html>
```

Web.xml

```
<web-app>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>
  <servlet>
    <servlet-name>SS</servlet-name>
    <servlet-class>
      <servlet-mapping>
        <servlet-name>SS</servlet-name>
        <url-pattern>/first</url-pattern>
      </servlet-mapping>
    </servlet>
  <servlet>
    <servlet-name>SS</servlet-name>
    <servlet-class>SecondServlet</servlet-class>
    <servlet-mapping>
      <servlet-name>SS</servlet-name>
      <url-pattern>/second</url-pattern>
    </servlet-mapping>
  <servlet>
    <servlet-name>DS</servlet-name>
    <servlet-class>DisplayServlet</servlet-class>
    <servlet-mapping>
      <servlet-name>DS</servlet-name>
      <servlet-class><del>DisplayServlet</del></servlet-class>
    </servlet-mapping>
  </servlet>
</web-app>
```

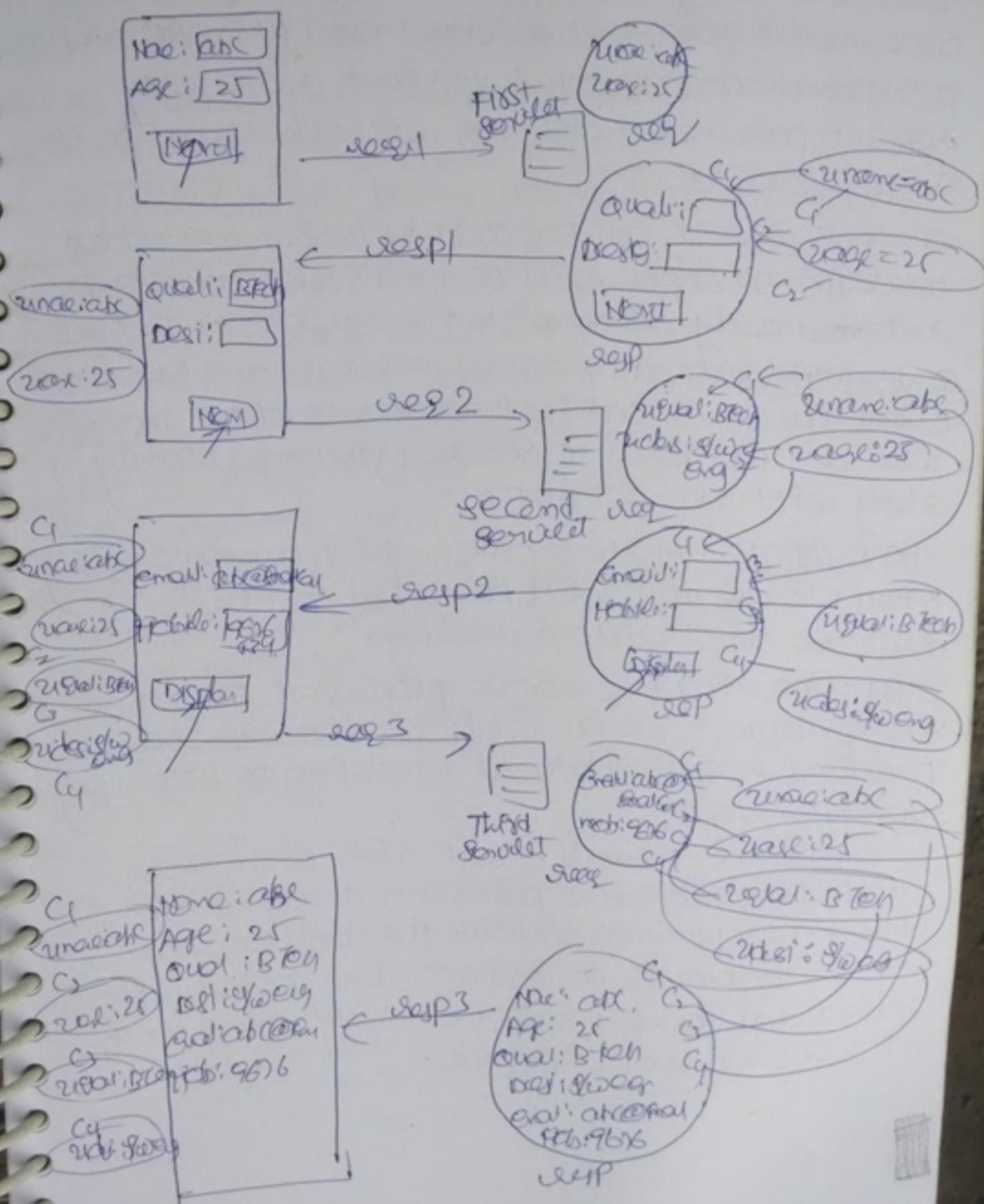
Drawbacks with the HttpSession Session tracking mechanism:-

- ① In HttpSession session tracking mechanism for each and every user a separate HttpSession object will be created at server machine, here if we increase number of users then number of HttpSession objects has to be prepared at server machine, this approach will increase burden to the server machine, it may ~~will~~ reduce the web application performance.
— To overcome this problem we have to use CookieSession tracking mechanism.
- ② In HttpSession session tracking mechanism when we create HttpSession object a SessionId will be created in the form of a cookie, it may be transport b/w server and client automatically in order to identify user specified HttpSession object.

In this context if we disable cookies at client browser then HttpSession session tracking mechanism will not be executed.

To overcome the above problem, we have to use URLRewriting session tracking mechanism.

Cookies Session Tracking mechanism:



JPG to PDF Converter For Mac - Unregis

— Process of cookie session tracking mechanism at each and every request we will prepare request parameters from request object and we will prepare a separate cookie object for each and every request parameter and we will add them to the response object.

In this context, all the cookies which we added to response object will be transferred to client automatically along with the response. In this context if we send another request from the same client all the available cookies at client machine will be transferred to server automatically along with the request.

Here whatever the cookies which are come along with the request will be added to the response object automatically.

— By repeating the above process at each and every request we are able to manage clients previous request data at the time of processing later request.

— cookie is a small object able to manage a single name-value pair and stored at client hard disk permanently. The default nature of the cookies is to transfer the cookies b/w client and server automatically along with the request and response.

— In Servlet applications, to represent a cookie object, we will use a predefined class provided by Servlet API `[javax.servlet.http.Cookie]`

— To prepare cookie object we will use the following constructor from `Cookie` class:

`[public Cookie(String name, String value)]`

Ex: `Cookie c = new Cookie("username", "abcd");`

To add a cookie to the response object explicitly we have to use the following method from `Cookie` class ~~HttpServlet~~ `HttpServletResponse`

`[public void addCookie(Cookie c)]`

Ex: `res.addCookie(c);`

— To get all the cookies from request object we have to use the following method from `HttpServletRequest`.

`[public Cookie[] getCookies()]`

Ex: `Cookie[] c = res.getCookies();`

— To get the name of name-value pair available in cookie, we have to use the following method
`public String getName()`

Ex:

— To get the value of the name-value pair from cookie we have to use the following method
`public String getValue()`

To set a name and value to the cookie explicitly we have to use the following methods

```
public void setName(string name)
```

```
public void setValue(string value)
```

In cookies session Tracking mechanism it is possible to set version numbers to the cookie and it is possible to get the present Version number of the cookie for this we will use the following methods

```
public void setVersion(int no);
```

```
public int getVersion();
```

In cookies session Tracking mechanism, it is possible to set a path to store the respective cookie on a particular location on client and it is possible to get a specified path, for this we have to use the following methods from

```
public void setPath(string path) Cookies class
```

```
public string getPath();
```

In cookies session Tracking mechanism, it is possible to set domain name from which website the cookies are coming and it is possible to get the specified domain name, for this we have to use the following methods

```
public void setDomain(string name)
```

```
public string getDomain()
```

— In cookies Session tracking mechanism. It is possible to set age to the cookie and it is possible to get the specified age from the cookie. For this we have to use the following methods:

```
public void setMaxAge(int time)
public int getMaxAge()
```

From Scott Copy

— In web applications, it is possible to provide comments to the cookie. To set a comment to the cookie and to get the comment from cookie, we have to use the following methods from Cookie class:

```
public void setComment(String comment)
public String getComment()
```

— In web applications it is possible to provide version numbers to the cookies.

What is this version number?

A version number which specifies the number of times updates will be performed on the respective cookie object.

Suppose in my web application, I use the cookies for the first time. So I will set the version number as 1 to the cookie. Some other time when I modified the cookie values then I will modify the cookie version number as 2.

what is cookie path?

Ans: Cookies are stored on client machine. now if we want to specify my own path on the client machine where the respective cookies have to be stored. for this we have to use getpath()

what is website domain name? method

Ans: Each and every website on the Internet is associated and identified by a name. By using this name only we can visit the website. This name is called website domain name. we can set explicitly a website domain name to the cookies [as a programmer] (in servlet program). Client will know from which website these cookies are coming, by see this domain name added to the cookies.

(+) 1

FirstServlet.java

```
import javax.servlet.*;  
import javax.servlet.http.*;  
import java.io.*;  
  
public class FirstServlet extends HttpServlet  
{  
    public void doPost(HttpServletRequest req,  
    HttpServletResponse res) throws ServletException,  
    IOException  
    {  
        String uname = req.getParameter("uname");  
        String upwd = req.getParameter("upwd");  
        HttpSession hs = req.getSession();  
        hs.setAttribute("uname", uname);  
        hs.setAttribute("upwd", upwd);  
        RequestDispatcher rd = req.getRequestDispatcher(  
            "/form2.html");  
        rd.forward(req, res);  
    }  
}
```

SecondServlet.java

```
import javax.servlet.*;  
import javax.servlet.http.*;  
import java.io.*;  
  
public class SecondServlet extends HttpServlet  
{  
    public void doPost(HttpServletRequest req,  
    HttpServletResponse res) throws ServletException,  
    IOException  
    {  
    }
```

```
String uequal = ureq.getParameter("uequal");
String udes = ureq.getParameter("udes");
HTTPSession hs = ureq.getSession(false);
hs.setAttribute('uequal', uequal);
hs.setAttribute('udes', udes);

RequestDispatcher ord = ureq.getRequestDispatcher(
    "/form3.html");
ord.forward(ureq, ureq);
```

{}

3

DisplayServlet.java

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

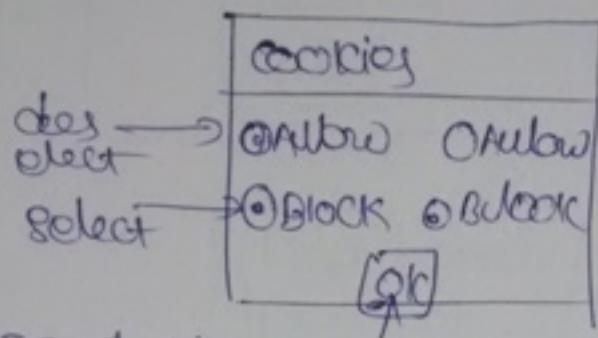
public class DisplayServlet extends HttpServlet
{
    public void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException
    {
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
        String email = req.getParameter("email");
        String mobile = req.getParameter("mobile");
        HttpSession hs = req.getSession();
        out.println("<html>");
        out.println("  <body background='lightgreen'>");
        out.println("    <center><br><br>");
        out.println("    <table border='1'>");
        out.println("      <tr><td><center><b><font");
        out.println("          size='7'>100</font></b></td><td><input");
        out.println("          type='button' value='Register' style='width:100px;'></td></tr>");
        out.println("    </table>");
        out.println("  </center></body>");
```

```
pt.println("<tr><td>Designation </td><td>" +  
    h.getAttribut("uDes") + "</td></tr>");  
pt.println("<tr><td>email <(td><td>" + email + "</td></tr>");  
pt.println("<tr><td>mobile " + NO + "<td><td>" + mobile + "</td></tr>");  
out.println("<table><tr><td></td></tr>");
```

3

3

Execution: In the above program,
open the browser → Tools → properties,

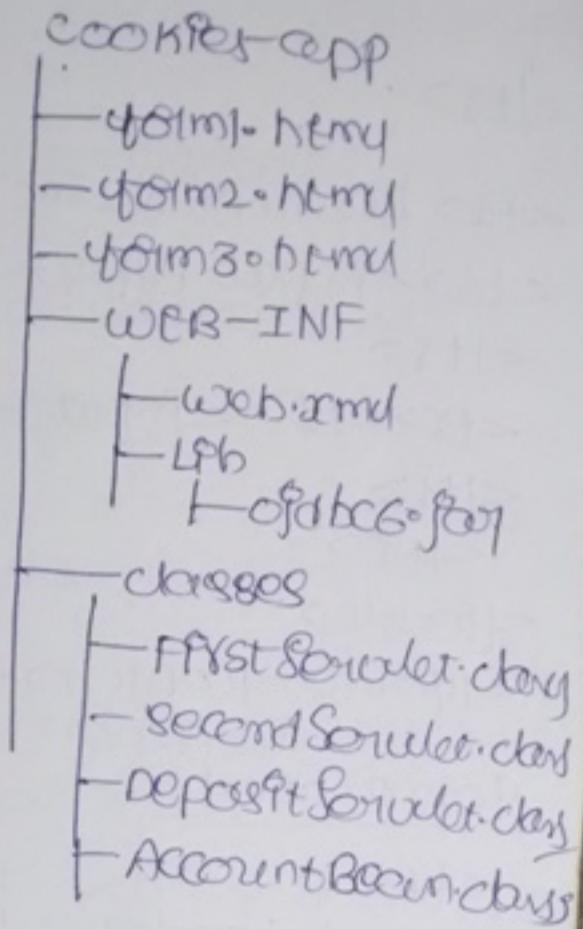
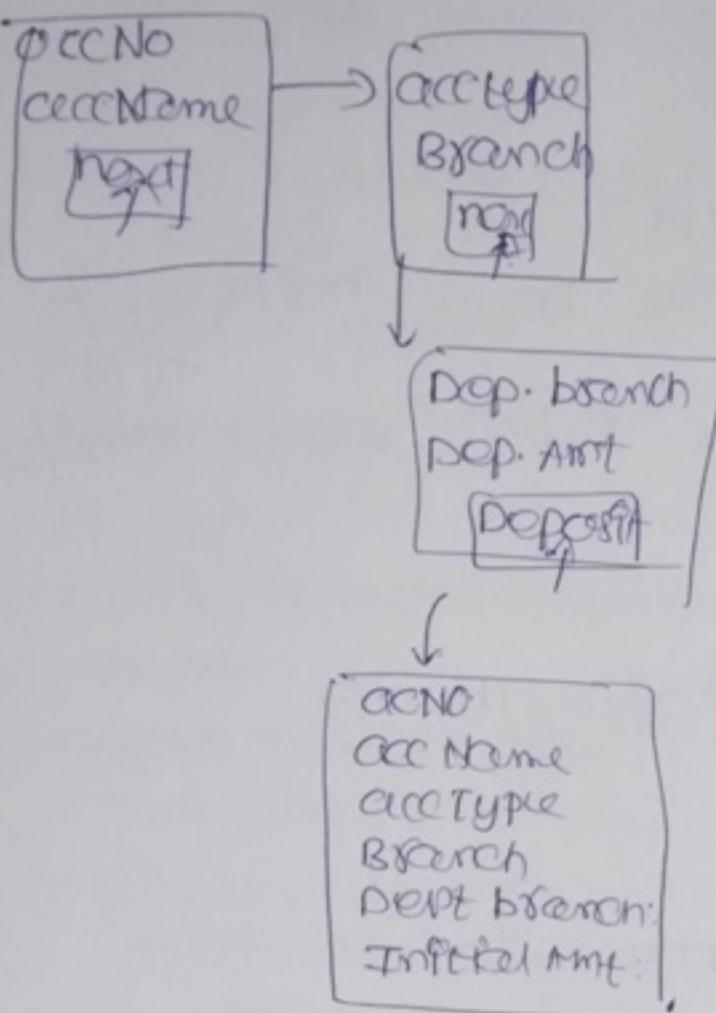


Now, send the
request from browser,

↓
privacy

↓
Advanced privacy
settings.

Here in this step, we
block the cookie in the
browser to show what
will happen



f1.m1.html

```

<html>
<body bgcolor="#lightgreen">
<center>
<form method="get" action=".//f1.m2">
<table border="1" style="width: 100%; border-collapse: collapse; background-color: #ffff00;">
<tr><td colspan="2" style="text-align: center; padding: 5px; font-size: 14pt; color: red; background-color: #ffff00;">
Account Details </td></tr>
<tr><td style="width: 15%; padding: 5px; text-align: right; vertical-align: top;">
<label style="font-size: 10pt; margin-bottom: 5px;">Account NumberInitial Amount

```

<tr>

<td> Account Name </td>

<td> <input type="text" name="accName"/>

<td>

<td>

<table>

<form> <center> <body> </body>

Form2.html

<html>

<body background="lightgreen">

<center>

<form method="get" action="/second">

<table background="lightyellow">

<tr> <td colspan="2">

<center> </center>

Account Details(Cont)

</tr> </table> <tr> <td> </td> <td>

 <td> Account Type </td>

<td> <input type="text" name="acctype"/>

</td>

<td> Account Branch </td>

<td> <input type="text" name="accBranch"/>

JPG to PDF Converter For Mac - Unregis

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

```
<html><td>
```

```
<input type="submit" value="Next"/>
```

```
<td></td>
```

```
<table><form><center><body></body></form>
```

form3.html

```
<html>
```

```
<body bgcolor="lightgreen">
```

```
<center>
```

```
<form method="get" action="/deposit/">
```

```
<table border="1">
```

```
<tr><td colspan="2" style="text-align: center; border: 1px solid red;">
```

Account Details (Cont--)

```
<tr></tr><tr></tr>
```

```
<tr>
```

```
<td> Dept. Branch </td>
```

```
<td><input type="text" name="deptBranch"/>
```

```
</td></tr>
```

```
<tr><td> Dept. Amount </td>
```

```
<td><input type="text" name="deptAmount"/>
```

```
</td></tr>
```

```
<td><input type="submit" value="Deposit"/>
```

```
</td></tr>
```

```
</table></form></center></body></html>
```

web.xml

```
<web-app>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
  </welcome-file-list>
  <servlets>
    <servlet>
      <servlet-name>fs</servlet-name>
      <servlet-class>FirstServlet</servlet-class>
    </servlet>
    <servlet-mapping>
      <servlet-name>fs</servlet-name>
      <url-pattern>/first</url-pattern>
    </servlet-mapping>
    <servlet>
      <servlet-name>ss</servlet-name>
      <servlet-class>SecondServlet</servlet-class>
    </servlet>
    <servlet-mapping>
      <servlet-name>ss</servlet-name>
      <url-pattern>/second</url-pattern>
    </servlet-mapping>
    <servlet>
      <servlet-name>ds</servlet-name>
      <servlet-class>DepositServlet</servlet-class>
    </servlet>
    <servlet-mapping>
      <servlet-name>ds</servlet-name>
      <url-pattern>/deposit</url-pattern>
    </servlet-mapping>
  </servlets>
</web-app>
```

FirstServlet.java

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

public class FirstServlet extends HttpServlet
{
    public void doGet(HttpServletRequest req,
                      HttpServletResponse res) throws ServletException
    {
        String accNo = req.getParameter("accNo");
        String accName = req.getParameter("accName");
        Cookie c1 = new Cookie("accNo", accNo);
        Cookie c2 = new Cookie("accName", accName);
        res.addCookie(c1);
        res.addCookie(c2);
        RequestDispatcher rd = req.getRequestDispatcher("form2.html");
        rd.forward(req, res);
    }
}
```

SecondServlet.java

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

public class SecondServlet extends HttpServlet
{
    public void doGet(HttpServletRequest req,
    HttpServletResponse res) throws ServletException,
    IOException
    {
        String accType = req.getParameter("accType");
        String accBranch = req.getParameter("accBranch");
        Cookie c3 = new Cookie("accType", accType);
        Cookie c4 = new Cookie("accBranch", accBranch);
        res.addCookie(c3);
        res.addCookie(c4);
        RequestDispatcher rd = res.getRequestDispatcher();
        rd.forward(req, res); ("form3.html");
    }
}
```

output:

DepositServlet.java

Date: 8th May, 2012 Tue

```
res.setContentType("text/html");
pw = res.getWriter();
Cookie c = res.getCookie("c");
out.println("Cookie");
<tag type="checkbox" checked="checked">;
<checkbox checked="checked" size="8">;
for (int i = 0; i < c.getLength(); i++) {
    out.println(c[i].getName());
    out.println(c[i].getValue());
    out.println("<br><br>");
    out.println("if (true) <br> " + c[i]);
}
```

```
out.println("<tr><td>AccountNumber </td><td>" +  
    accNO + "</td></tr>");  
out.println("<tr><td>AccountName </td><td>" +  
    accName + "</td></tr>");  
out.println("<tr><td>AccountType </td><td>" +  
    accType + "</td></tr>");  
out.println("<tr><td>TransactionType </td>  
    <td>Deposit </td></tr>");  
out.println("<tr><td>" + accBranch + "</td></tr>");  
out.println("<tr><td>Deposit Branch </td><td>" +  
    depBranch + "</td></tr>");  
out.println("<tr><td>Initial amount </td><td>" +  
    (total_Amount - depAmount) + "</td></tr>");  
out.println("<tr><td>Deposit Amount </td><td>" +  
    depAmount + "</td></tr>");  
out.println("<tr><td>Total Amount </td><td>" +  
    total_Amount + "</td></tr>");  
out.println("<tr><td>Transaction Status </td>  
    <td>SUCCESS</td></tr>");  
out.println("<tr><td colspan=2><center>  
    color='red'> Thank You & Visit Again </p><br/>  
    </center></td></tr>");
```

columns > select

SQL > select * from account;

ACCNO	ACCTNAME	ACCTTYPE	BALANCE
a1	AAA	savings	10000

Account Bean class

import java.sql.*;

public class AccountBean

{ Connection con; int initialAmount;
Statement st; int totalAmount;
ResultSet rs;

AccountBean()

{ try

{

Class.forName("oracle.jdbc.driver.OracleDriver");
con = DriverManager.getConnection("jdbc:oracle:
thin:@localhost:1521:xe", "system", "durga");

st = con.createStatement();

} catch (Exception e)

{ e.printStackTrace();

}

public int deposit(String accNo, int depAmount)

{ try

{

rs = st.executeQuery("select * from account
where accNo='?accNo ?'");
rs.next();

```
if(Initial_Amount = rs.getInt(4);  
total_Amount = Initial_Amount + dept_Amount;  
int rowCount = st.executeUpdate("update  
account set balance = " + total_Amount + "  
where accNo = " + accNo + "");  
System.out.println total  
Amount;  
} catch (Exception e)  
{  
    System.out.println initial_Amount;  
    e.printStackTrace();  
}  
}  
  
Execution.
```

ince the data is available at client machine then
that will be open to every user of that machine
Therefore cookies session tracking mechanism
do not provide security for the client data.

-In cookie session tracking mechanism the client
converstation will be maintained by the server
until user logs out.

Drawbacks with the cookie session tracking mechanism

- In cookie session tracking mechanism client conversation will be maintained at the same client machine in the form of cookies, where the client data in the form of cookies is open to every user of that client machine so that (cookies) session tracking mechanism will not provide security for the client data.
 - In cookie session tracking mechanism if we disable cookies at client browser then automatically cookie session tracking mechanism will not be executed.
- To overcome the above problems we have to use URL rewriting session tracking mechanism

URL Rewriting Session tracking mechanism

In session tracking mechanism, if we disable the cookies at client browser then HttpSession session tracking mechanism and cookie session tracking mechanisms will not be executed. To overcome this problem only, we have to use URL rewriting session tracking mechanism.

— In URL rewriting session tracking mechanism is almost all same as HttpSession session tracking mechanism, but in case of HttpSession session tracking mechanism, we will consider the SessionID value from SessionID cookie.

But in case of URL

we will get SessionID value explicitly by calling getID() method, we will append that SessionID value to the URL in the next form generation by rewriting URL.

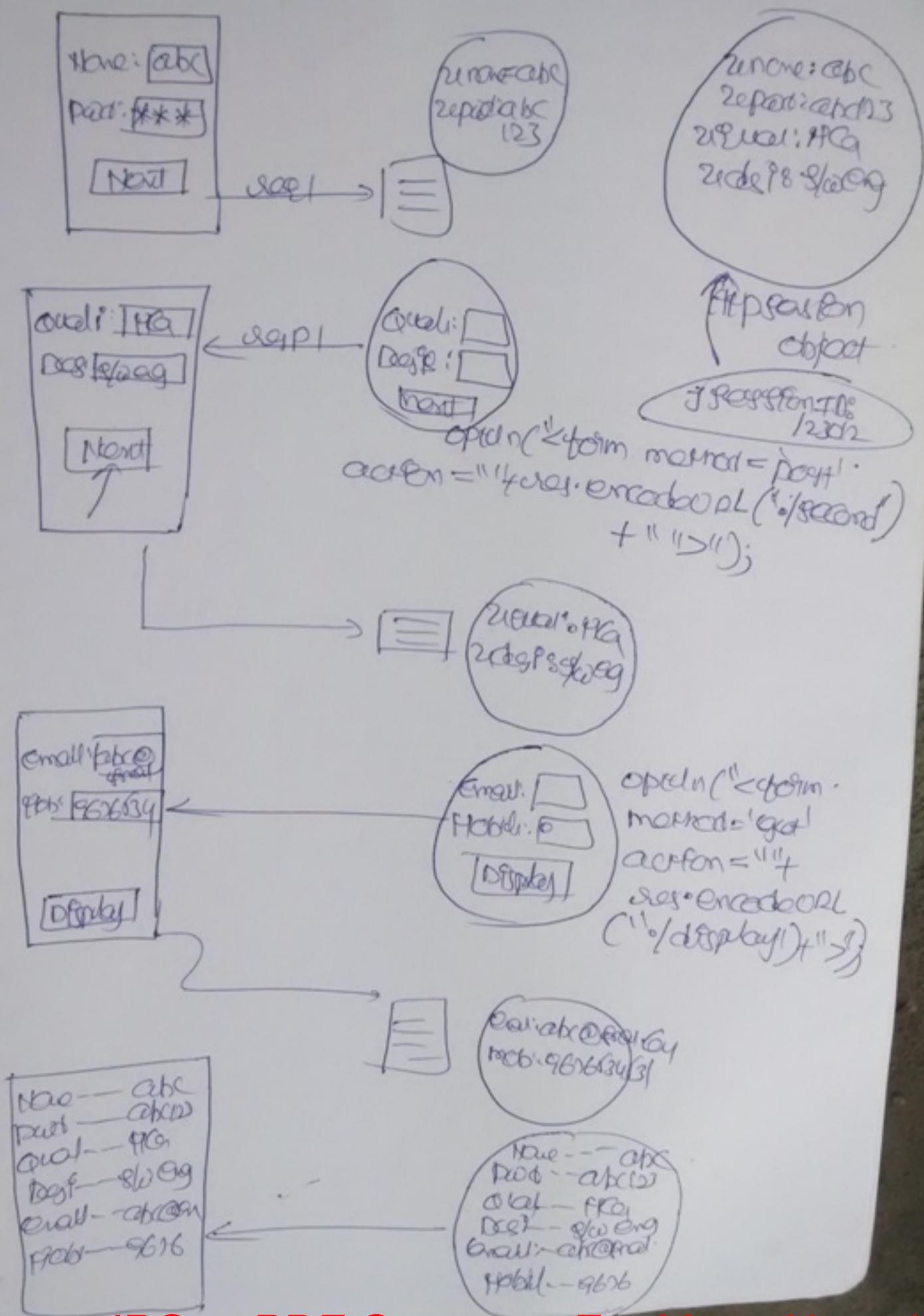
```
cpIn("<form method='get' action='!/Second.jsp'>  
    JSessionID=" + hs.getId() + "</>");
```

— In the above instruction, we instead of getting SessionID value and appending SessionID value to the URL explicitly we will use an alternative method provided by HttpServletResponse. That is

```
public String encodeURL(String url)
```

```
out.println("<form method='post'  
action ='" + res.encodeURL("./re  
+ "')");
```

by repeating the above
process at each and every element we can easily
manage the previous request data at one time
of processing later requests.



JPG to PDF Converter For Mac - Unregis