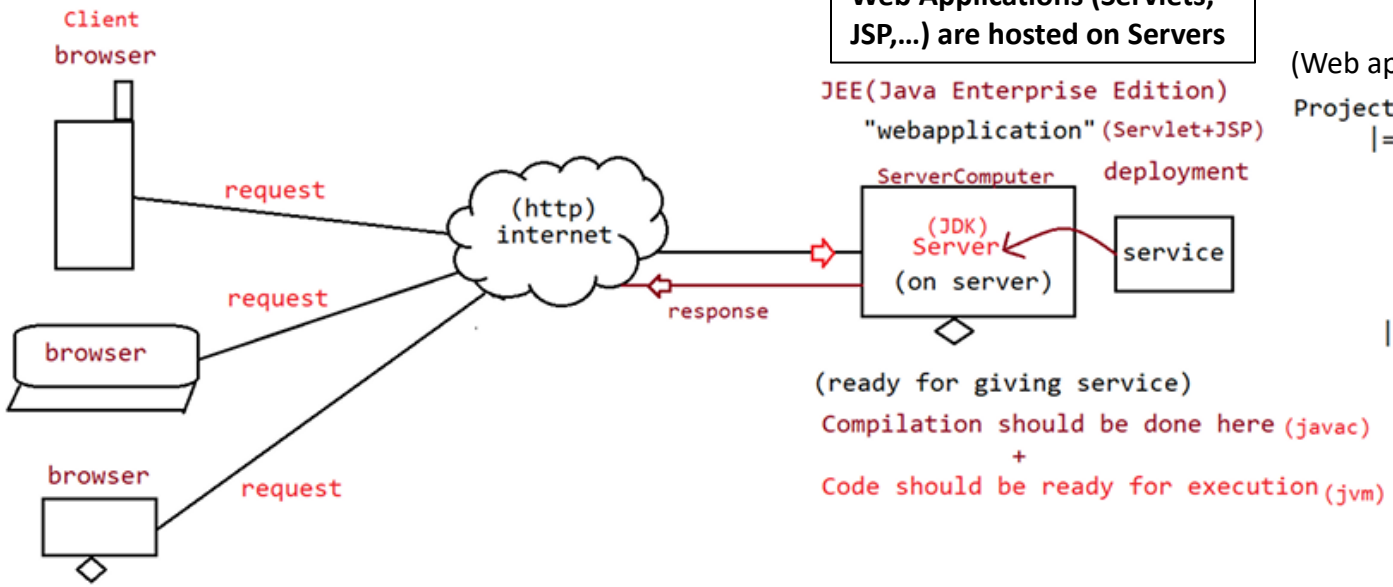


Web Applications (Servlets, JSP,…) are hosted on Servers

JEE(Java Enterprise Edition)

"webapplication" (Servlet+JSP)

(ready for giving service)

Compilation should be done here (javac)

+

Code should be ready for execution (jvm)

(Web application folder on server)

```
ProjectFolder (FirstApp)
    |=> WEB-INF
          |=>web.xml
          |=> classes
                  *.class (code is compiled and ready for execution)
          |=> lib (optional untill u use 3rd party api's)
                  *.jar
    |=> src
          |=>packages
                  |=> .java
```

Standalone(oneuser)

⇩ 1 Thread(JVM creates)

JSE => CoreJava

ExceptionHandling,String,MultiThreading

Collection,Wrapperclasses,FileHandling,JDBC

webapps(many user)

⇩ Server(ThreadPool)

JEE => AdvancedJava

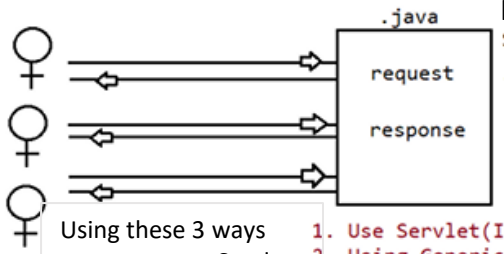
Servlet,JSP,JSTL,Thymleaf,....

Assign one thread for every client and give them the response

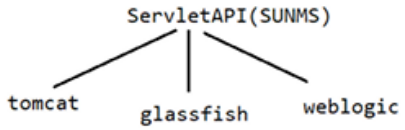
henceforth no main() in webapps

Structure of tomcat (server)

- bin ⇦ .exe files
- conf ⇦ configuration files
- lib ⇦ *.jar
- logs ⇦ (request,response information of clients)
- temp ⇦ (temporary files)
- webapps ⇦ (deployment folder)
- work ⇦ (backup files)



In order to server multiple users at a time
SUNMS community had given rules in the form of an api called "Servlet"



Implementation for this API is given by the vendors who are providing servers to host our web applications such as tomcat, glassfish, weblogic,…

Using these 3 ways we can create Servlets i.e web applications

1. Use Servlet(I)
2. Using GenericServlet(AC)
3. Using HttpServlet(AC)

1. XML
2. Annotation

This can be done in two ways:
1. Using web.xml file
2. @webservlet annotation

(takes the request) server
(process the request and generate the response) container(catalina)
(configure the container to execute the files based on the pattern search made by the user)

http://localhost:9999/FirstApp/test
AppName
url
/test
response

/test ===> DemoServlet
DemoServlet
FirstServlet(.class)

C:\
Tomcat 9.0
webapps
FirstApp
WEB-INF .web.xml
classes .class
FirstServlet.java

#1 Loading
#2 Instantiation
obj
"ZeroArgument constructor"
Life Cycle methods would be executed

#3
obj
config
.init(ServletConfig config)
{
;;;;;;;
}

#4
obj
ServletRequest
ServletResponse
.service(SR request, SR response)
{
;;;;;;;;;;
}

#5
obj
.destroy()
{
;;;;;
}

web.xml

```
<web-app>

   <servlet>                    #3
        <servlet-name>DemoServlet</servlet-name>
        <servlet-class>FirstServlet</servlet-class>
   </servlet>                   #4

   <servlet-mapping>            #2
        <servlet-name>DemoServlet</servlet-name>
        <url-pattern>/test</url-pattern>
   </servlet-mapping>           #1

</web-app>
```

```
Microsoft Windows [Version 10.0.22621.819]
(c) Microsoft Corporation. All rights reserved.

C:\Users\nitin>set path= C:\Program Files\Java\jdk1.8.0_202\bin;
```

```
C:\Users\nitin>set classpath=.;C:\Tomcat 9.0\lib\servlet-api.jar
```

```
C:\Tomcat 9.0\webapps\FirstApp>javap javax.servlet.Servlet
'javap' is not recognized as an internal or external command,
operable program or batch file.

C:\Tomcat 9.0\webapps\FirstApp>echo %path%
 C:\Program Files\Java\jdk1.8.0_202\bin;

C:\Tomcat 9.0\webapps\FirstApp>set path=C:\Program Files\Java\jdk1.8.0_202\bin;
```

How to load the .class file dynamically based on the runtime requirement?

```
Class c = Class.forName("FirstServlet");
```

How to instantiate the object for the dynamic loaded class?

```
FirstServlet obj=(FirstServlet)c.newInstance();
```

What does 'cls' command do in command prompt?

```
C:\Tomcat 9.0\webapps\FirstApp>javap javax.servlet.Servlet
Compiled from "Servlet.java"
public interface javax.servlet.Servlet {
  public abstract void init(javax.servlet.ServletConfig) throws javax.servlet.ServletException
;
  public abstract javax.servlet.ServletConfig getServletConfig();
  public abstract void service(javax.servlet.ServletRequest, javax.servlet.ServletResponse) th
rows javax.servlet.ServletException, java.io.IOException;
  public abstract java.lang.String getServletInfo();
  public abstract void destroy();
}

C:\Tomcat 9.0\webapps\FirstApp>
```

Note: load-on-startup

a. value should be 0 and any positive number
b. if 2 servlets has same value then we can't predict the order
3. less the load-on-startup for a value,that particular servlet loading,instantiation,initialziation will happen.

D:\servletpgms\.metadata\.plugins\org.eclipse.wst.server.core

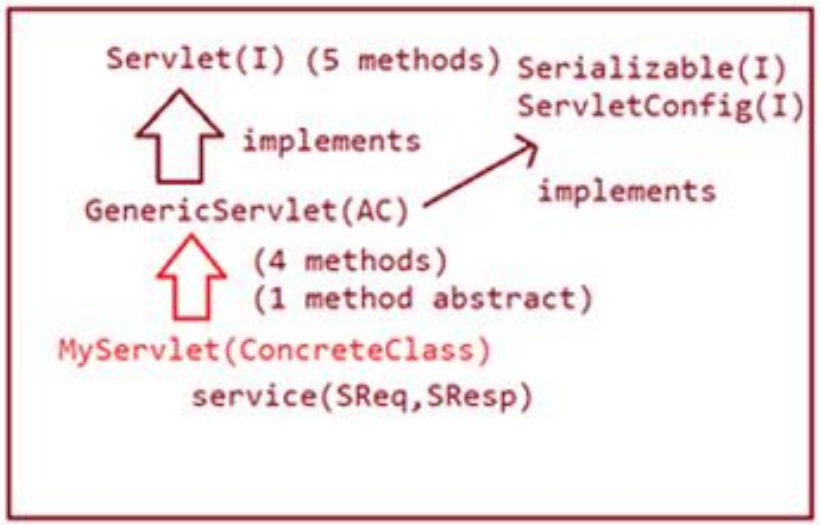
our project created location

inside this folder eclipse automatically
keeps the application.
this type of deployment is called "soft-deployment".

Servlet(I)
|=> 5 methods

init(SC)
service(SReq,SRes) <= request processing logic
destroy()

getServletConfig()
getServletInfo()



GenericServlet is an abstract class which implements Servlet interface.
It has implementations for four methods of Servlet but service() is abstract.

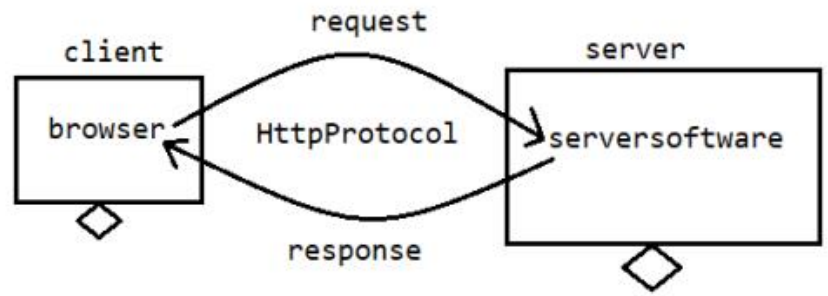
GenericServlet====> meant for any protocol interaction
like http, smtp,.....

For a webapplication, the protocol used in "Http",so to work with specific protocol SUNMS had come up with SRS called "HttpServlet".

SUNMS had come up with special package to promote
http protocol interaction

interface
=========
1.HttpServletRequest(I)
2.HttpServletResponse(I)
3.HttpSession(I)

class
=====
1.HttpServlet
2.Cookie



```
D:\>javap javax.servlet.GenericServlet
Compiled from "GenericServlet.java"
public abstract class javax.servlet.GenericServlet implements javax.servlet.Servlet,javax.servlet.ServletC
lizable {
  public javax.servlet.GenericServlet();
  public void destroy();
  public java.lang.String getInitParameter(java.lang.String);
  public java.util.Enumeration<java.lang.String> getInitParameterNames();
  public javax.servlet.ServletConfig getServletConfig();
  public javax.servlet.ServletContext getServletContext();
  public java.lang.String getServletInfo();
  public void init(javax.servlet.ServletConfig) throws javax.servlet.ServletException;
  public void init() throws javax.servlet.ServletException;
  public void log(java.lang.String);
  public void log(java.lang.String, java.lang.Throwable);
  public abstract void service(javax.servlet.ServletRequest, javax.servlet.ServletResponse) throws javax.servlet.ServletExcep
tion, java.io.IOException;
  public java.lang.String getServletName();
}

D:\>
```

GenericServlet is an abstract class which implements Servlet interface.
It has implementations for four methods of Servlet but service() is abstract.

Why two init methods in GenericServlet (AC)?
The one with the parameters is called by Web Container
The Parameterles init() is used by developers.

1. loading
2. instantiation
3. initialization

```
init(ServletConfig config)
{
    init()
}
```

⇧ local variable

4. service

```
public void service(Sreq,Sresp)throws SE,IOE
   {
       ServletConfig config =getServletConfig();
       System.out.println(config); // null
   }
```

5. deInstantiation

If you override init(ServeltConfig config) method recklessly you may loose the local config variable. Then, getSerlveltConfig() gives null; which is not all good.

If you need init method, then always override init() method because the init(ServletConfig config) method internally calls init() method

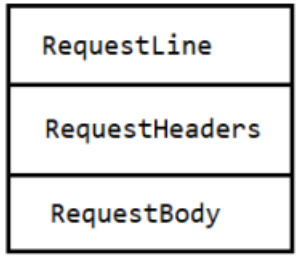
1. loading 2. instantiation
3. initialization

```
init(SerlvetConfig config)
init()
```

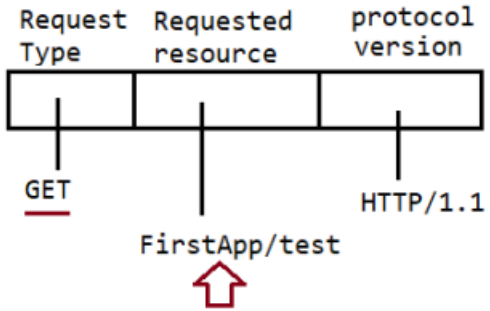
4. service
5. destroy

Structurue of HttpProtocol Request

Used in Spring REST API



RequestLine



configured in the server env

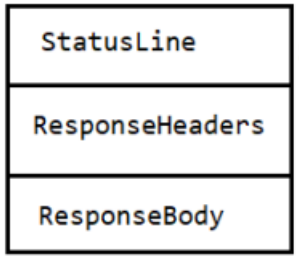
RequestHeaders ⇦ internally used to give a proper response

Browser details and its supported language format and so on.....

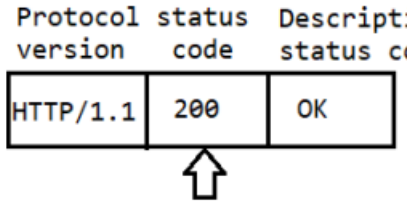
RequestBody ⇦ "user supplied inputs will be available"

It contains user information

Structure of HttpProtocol Response



StatusLine



1XX => informational
2XX => Succesfull
3XX => Redirection
4XX => ClientError
5XX => ServerError

ResponseHeader ⇦ MIME(MultiPurpose Internet Mail Exchange)

response information from the server
a. content-type
b. content-length
c. modified date etc.....

ResponseBody ⇦ "Acutal information"

Actual information sent by the server,this information will be displayed to the user.

Type of HttpRequest Methods

1. GET
2. POST
3. HEAD

introduced in HTTP/1.0V

4. PUT
5. DELETE
6. OPTIONS
7. TRACE

HTTP/1.1V

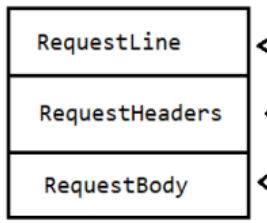
GET (it is also called as "idempotent request/safe request")
=> It is used to get the information from server
=> This request data will be visible in the address bar when we send the request, so it is not secured.
=> It can be book marked,and it also supports caching of data
=> since the data is visible in address bar, only limitied data can be sent.

Note: data would be sent in RequestLine

eg:: http://localhost:9999/FirstApp/test ?username = sachin & passsword = tendulkar

QueryString

1. Type url and hit enter key
2. clicking on hyperlink
3. submitting html form with method attribute as "GET"



GET(Query String is sent)

browser details would be available

empty

POST (It is not a safe request/idempotent request)
 => It is used for putting the data sent from client to server
 => This request data would not be visible in the address bar when we send the request, so it is secured.
 => It can't be book marked,and it won't supports caching of data
 => since the data is not visible in address bar, large volumen of data can be sent.

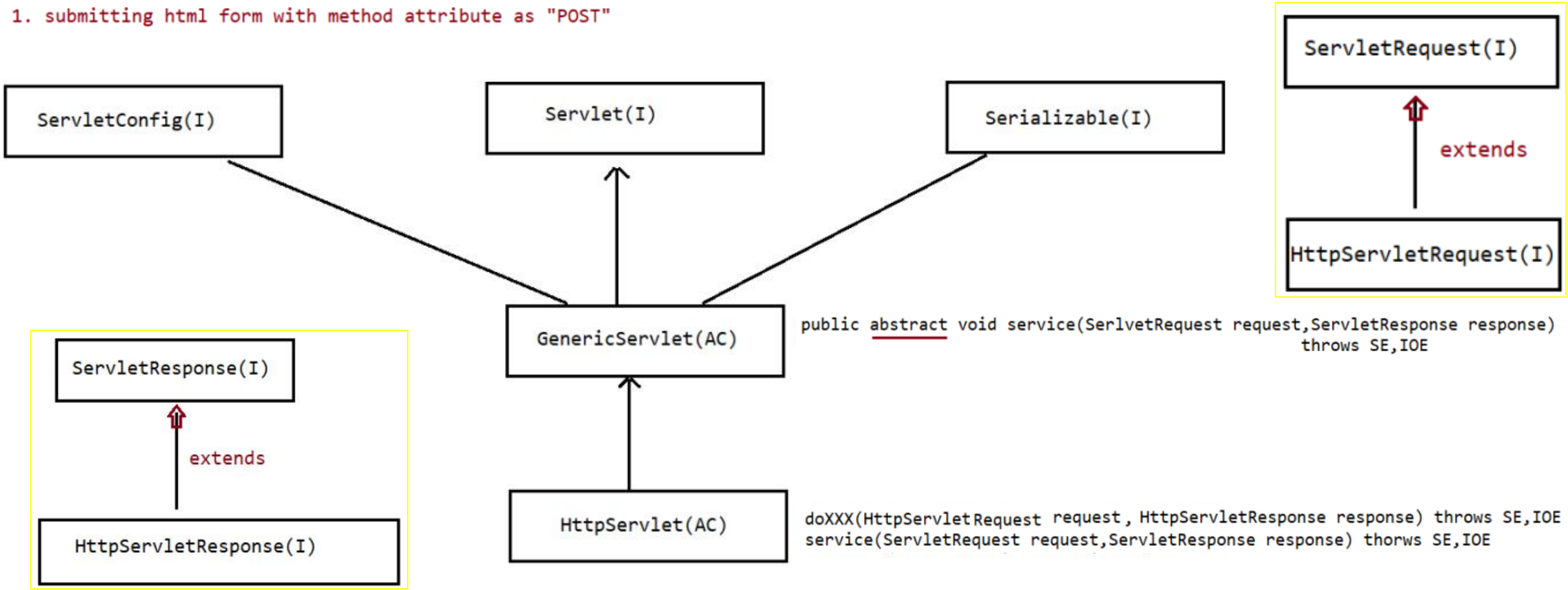
eg: http://localhost:9999/FirstApp/test ~~? username = sachin & password = tendulkar~~

QueryString

Note: data would be sent in RequestBody

RequestLine	POST
RequestHeaders	browser details would be available
RequestBody	(Query String is sent)

1. submitting html form with method attribute as "POST"



Why HttpServlet class is declared as abstract when no method in that class is abstract?

```
C:\Toncat 9.0\webapps\FirstApp>javac FirstServlet.java

C:\Toncat 9.0\webapps\FirstApp>javap javax.servlet.http.HttpServlet
Compiled from "HttpServlet.java"
public abstract class javax.servlet.http.HttpServlet extends javax.servlet.GenericServlet {
  public javax.servlet.http.HttpServlet();
  protected void doGet(javax.servlet.http.HttpServletRequest, javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletExce
ption, java.io.IOException;
  protected long getLastModified(javax.servlet.http.HttpServletRequest);
  protected void doHead(javax.servlet.http.HttpServletRequest, javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletExc
eption, java.io.IOException;
  protected void doPost(javax.servlet.http.HttpServletRequest, javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletExc
eption, java.io.IOException;
  protected void doPut(javax.servlet.http.HttpServletRequest, javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletExce
ption, java.io.IOException;
  protected void doDelete(javax.servlet.http.HttpServletRequest, javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletE
xception, java.io.IOException;
  protected void doOptions(javax.servlet.http.HttpServletRequest, javax.servlet.http.HttpServletResponse) throws javax.servlet.Servlet
Exception, java.io.IOException;
  protected void doTrace(javax.servlet.http.HttpServletRequest, javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletEx
ception, java.io.IOException;
  protected void service(javax.servlet.http.HttpServletRequest, javax.servlet.http.HttpServletResponse) throws javax.servlet.ServletEx
ception, java.io.IOException;
  public void service(javax.servlet.ServletRequest, javax.servlet.ServletResponse) throws javax.servlet.ServletException, java.io.IOEx
ception;
  static {};
}
```

```
protected void service(HSR req,HSR resp) throws SE,IOE
{
        String method = req.getMethod();

        if(method.equalsIgnoreCase("POST"))
           doPost(req,resp)

        else if(method.equalsIgnoreCase("GET"))
           doGet(req,resp);

        ;;;;;
        ;;;;;
        ;;;;;
        else
            return 501 status code saying http method not implemented

}
```

```
public void service(SReq req, SRes resp) throws SE,IOE
{
        HttpServletRequest request = (HSR) req; HttpServletResponse response = (HSR) resp;
        service(request,response);

}
```

SReq – ServletRequest
SRes – ServletResponse
HSR – HttpServletRequest/Response

HttpServlet(AC)

```
protected void doPost(HSR req, HSR resp) throws SE,IOE
{
    return 405|400 status code saying HTTP Method POST not s upported by this URL
}

protected void  doGet (HSR req, HSR resp) throws SE,IOE
{
    return 405|400 status code saying HTTP Method GET not s upported by this URL
}
```

1. doPost() and doGet() methods of HttpServlet class are not abstract. Don't you think they must be abstract like service() method of GenericServlet class so that we can define our own custom service functionality of servlet.
2. If you observe the definition of doPost() and doGet() methods, they serve only one purpose which is returning a code message when there is a client side error. So these instance methods of HttpServlet class are not at helpful for providing services to client request
3. To make HttpServlet more functional we have to override and give custom definition for doPost() and doGet(), and not use the instance methods of Abstract Class for serving request.

That is the reason HttpServlet class is declared as abstract so that one cannot create object for it as a result cannot use it's instance doPost() and doGet() methods. Instead, create a concrete class which extends HttpServlet abstract class and override doPost() and doGet() methods creating customized service

1. loading
   static block

2. instantiation
   constructor(public zero argument)

3. initialization
   init(ServletConfig config)
   init() ⇦ meant for devleoper

4. request processing

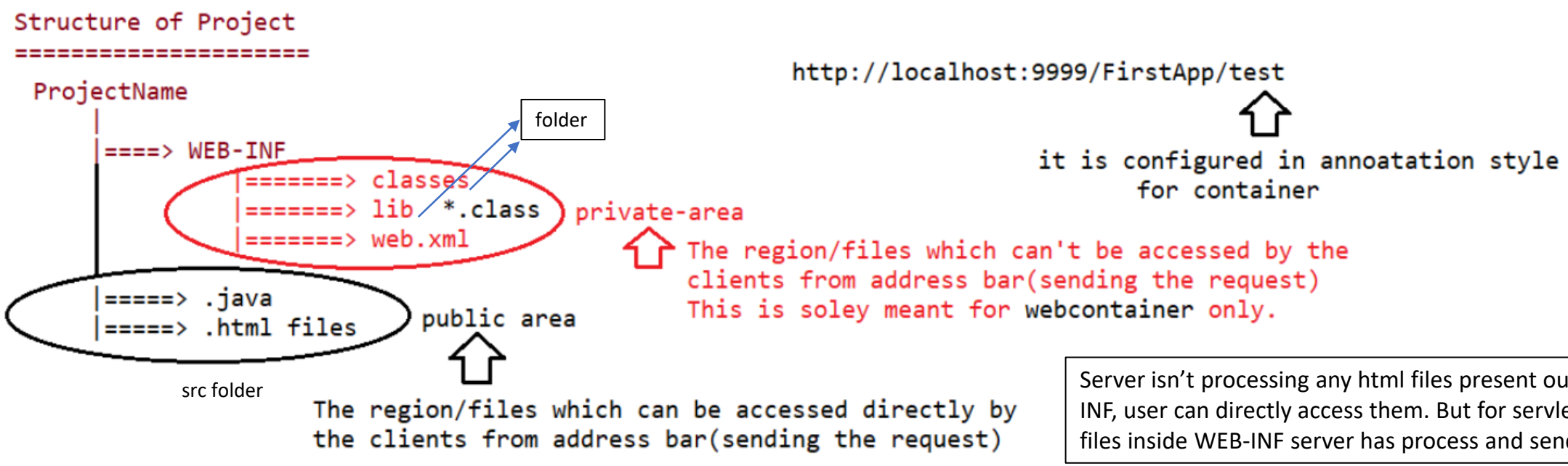
   public void service(SReq req,SRes) throws SE,IOE
   protected void service(HSR req,HSR res) throws SE,IOE

   ⇩

   user defined FirstServlet Object doXXX(HSR req,HSR res) throws SE,IOE should be called

If you didn't give any definition for doPost() or doGet() methods in your concrete class then Instance methods of HttpServlet Abstract Class are executed any way.

5.deinstantiation
   public void destroy()

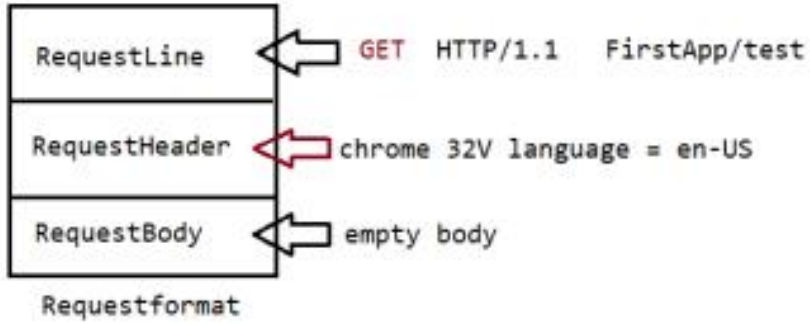


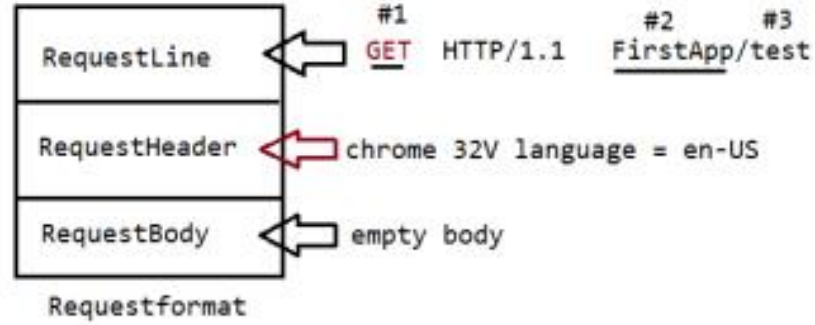
Server isn't processing any html files present outside WEB-INF, user can directly access them. But for servlet and other files inside WEB-INF server has process and send it to client.

1. Every Web Application must be present in a ProjectFolder/(WEB-INF,SRC)
2. Java Standard Edition – Core java(Until JDBC); only one thread i.e main thread (and from this thread we do multi threading)
3. Java Enterprise Edition – Servlets, JSP,JSTL, Thymleaf ……. (Server here maintains a thread pool to serve multiple users at a time using multiple threads instead of one single main thread)

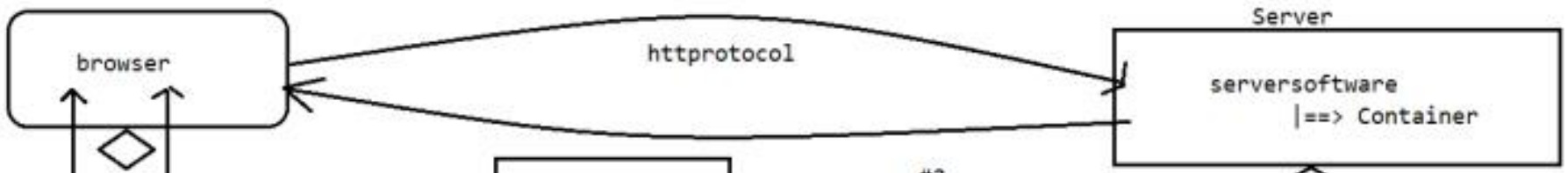
   So there is no main method in WEB APPS (Web Applications)

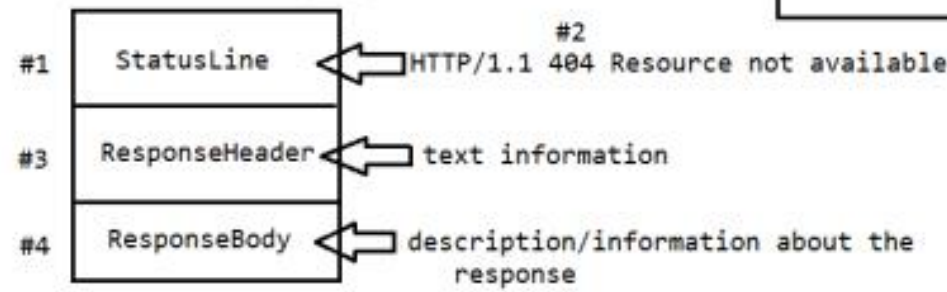
5. All the Web Applications must present in Web-Apps directory of the Tomcat 9.0 folder
6. What is tomcat ? – It is server provided by the third party vendor to host our Web application written Java.
7. What is present in lib folder of Tomcat directory and what is servlet-api.jar?
8. How many methods are there inside Servlet Interface? – 5 life cycle methods. – What are they?
9. What is the use of load-on-startup tag in web.xml or loadOnStartup attribute in WebServlet annotation?
10. Explain Servlet Interface, What is the need of GenericServlet and HttpServlet?
11. Who would give implementation for Servlet, ServletRequest, ServletResponse, ServletConfig interfaces?



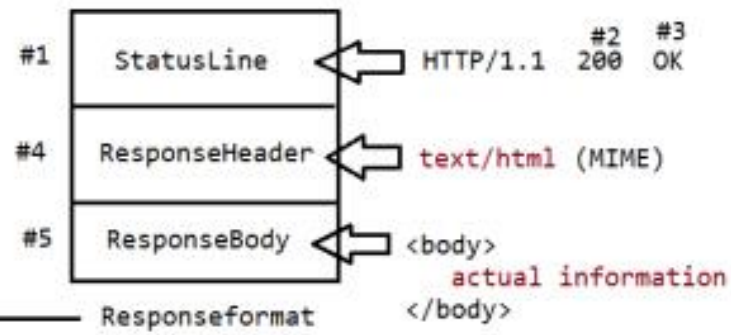


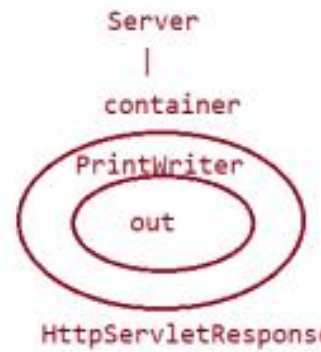
http://localhost:9999/FirstApp/test

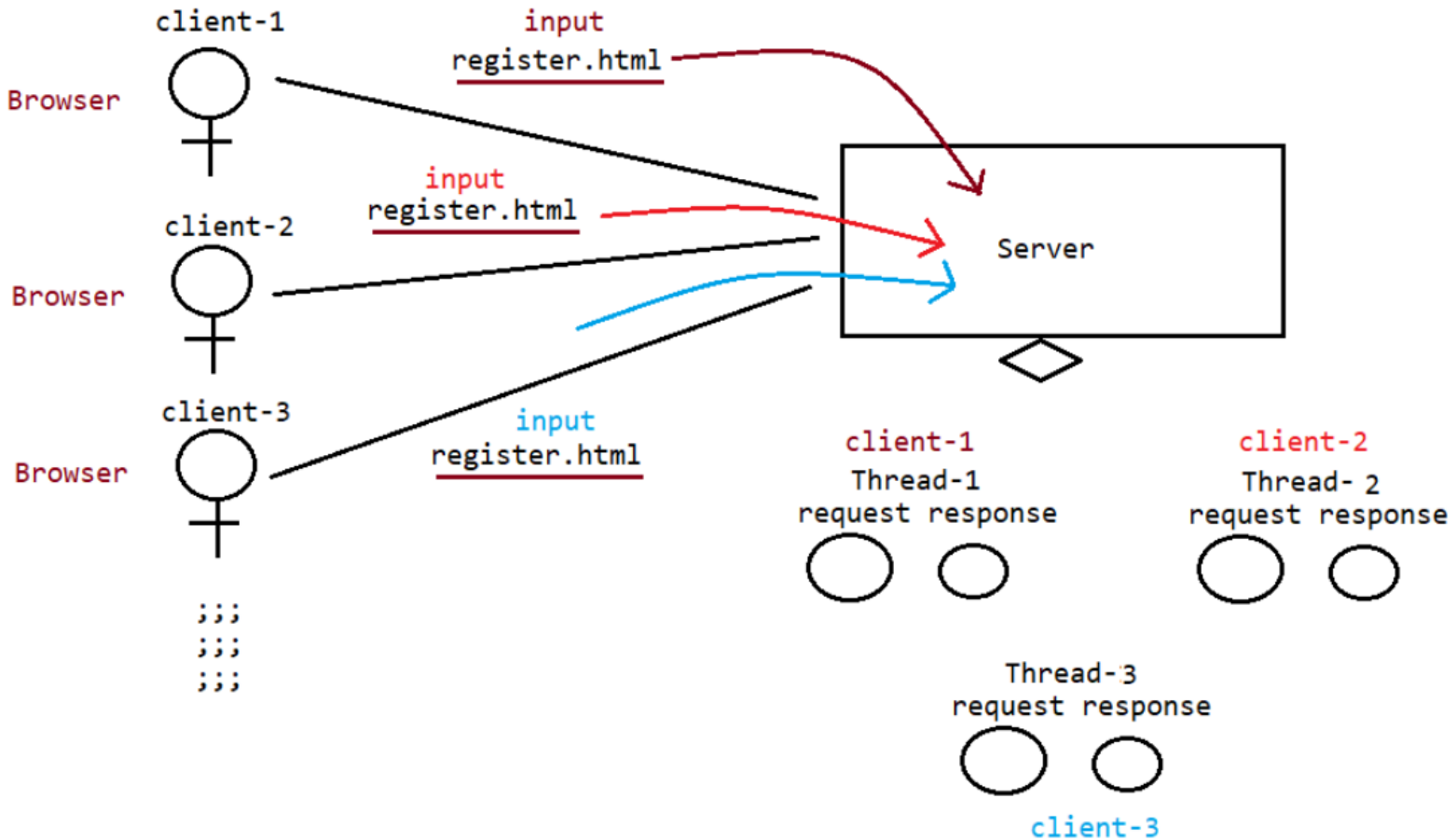




1. check the url pattern and find the resource
2. if resource not available
   => it generates clienterror and send it **to client**
3. if resource is available
   => loading
   => instantiation
   => initialization
   => request processing
      a. content-type
      b. use response entry
      c. write the response
      d. close the writer



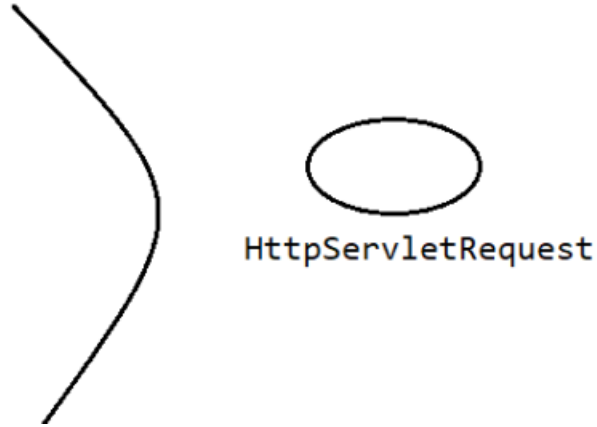





```
RequestApp
    |-> src
         |=> in.ineuron.controller
                    => RequestServlet.java

    |-> webapp
         |====> register.html

    |=> WEB-INF
          |=> classes
                 |=> RequestServlet.class
          |=> lib
                 |=> *.jar

          |=> web.xml
                 |=> mapping details in xml format
```

Servlet
=======

1. public String getParameter()
2. public String[] getParameterValues()
3. public Enumeration<String> getHeaderNames()
4. public Object getHeader(String headerName)
5. public String getRemoteHost()
6. public String getRemoteAddr()
7. public String getRemotePort()
8. public String getServerName()
9. public String getServerPort()

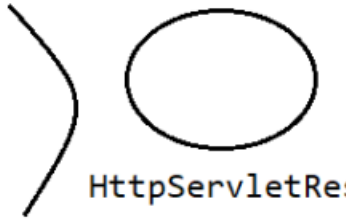


HttpServletRequest

Note: Writing images,pdf's,..... as a response

```
response.setContentType("image/jpg");
    ;;;;
    ;;;;
ServletOutptuStream os = response.getOutputStream();
```

1. public void setContentType(String name)
2. public PrintWriter getWriter()
3. public ServletOutputStream getOutputStream()



HttpServletResponse

location of image file
======================

D:\servletpgms\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps\ImageResponseApp\nitin.JPG

smooth deployment

ProjectName

Image

Note:

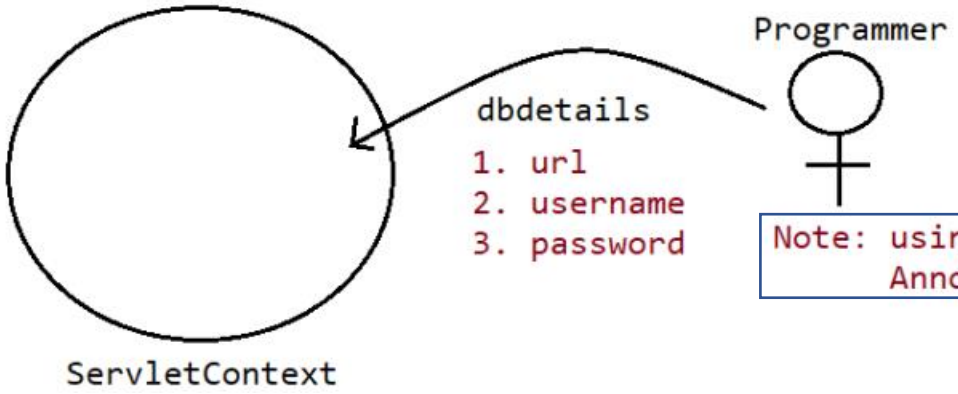
```
PrintWriter out = response.getWriter();
ServletOutputStream os = response.getOutputStream();
```

In Servlet Coding we can't open 2 streams for response, if we try to do it would result in java.lang.IllegalStateException

ServletContext(I)
==========

ServletContext is also an Interface

=> When we do deployment(manual), server will scan "webapps" folder and identifies the projects which is deployed.
=> All the identified Projects will be kept in Meta-space of server.
=> For Every Project which is deployed in Meta-Space automatically "ServletContext" object will be created.



Note: using XML only, no support of Annotation.

This ServletContext Object can be accessed by everyone in the entire project i.e all the servlets present in the project can access it.

ServletContext context = getServletContext();

context object can be accessed in the entire project, meaning in all the servlets of the project

1. public Enumeration getInitParameterNames()
2. public String getInitParameter(String name)
3. public Enumeration getAttributeNames()
4. public String getAttribute(String name)
5. public void setAttribute (String k,Object v)

Note: In ServletContext Object, we can add parameter data as well as attribute data

from xml

Parameter Data is Static

during execution

Attribute Data is Dynamic

getServletContext().setAttribute(String AttName, Obj AttValue)

These are also there.
Find about them -
request.setAttributeNames()
request.getAttribute()
request.setAttribute()

Context Object will be destroyed only when we undeploy the project.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<web-app>
  <display-name>ServletContextApp</display-name>

  <context-param>
    <param-name>jdbcURL</param-name>
    <param-value>jdbc:mysql:///enterprisejavabatch</param-value>
  </context-param>
  <context-param>
    <param-name>user</param-name>
    <param-value>root</param-value>
  </context-param>
  <context-param>
    <param-name>password</param-name>
    <param-value>root123</param-value>
  </context-param>

  <servlet>
    <servlet-name>TestServlet</servlet-name>
    <servlet-class>in.ineuron.controller.TestServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>TestServlet</servlet-name>
    <url-pattern>/test</url-pattern>
  </servlet-mapping>


  <servlet>
    <servlet-name>DemoServlet</servlet-name>
    <servlet-class>in.ineuron.controller.DemoServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>DemoServlet</servlet-name>
    <url-pattern>/demo</url-pattern>
  </servlet-mapping>
</web-app>
```

Only static parameters of servlet context are common for all servlets present in the web application. Dynamic attributes set by a servlet using setAttribute() cannot be accessed by other servlets.

@WebServlet() Annotation can be used only for servlet and url mapping but not for setting parameter values in case of ServletContext. Use only xml file for setting parameter values.

Here, both TestServlet and DemoServlet both can access the ServletContext Object

```
<servlet>
    <servlet-name>TestServlet</servlet-name>
    <servlet-class>in.ineuron.controller.TestServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>TestServlet</servlet-name>
    <url-pattern>/test</url-pattern>
</servlet-mapping>
```

```
<servlet>
  <servlet-name>DemoServlet</servlet-name>
  <servlet-class>in.ineuron.controller.DemoServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>DemoServlet</servlet-name>
  <url-pattern>/demo</url-pattern>
</servlet-mapping>
```

ServletContext(I)

```
<context-param>
   <param-name>jdbcURL</param-name>
   <param-value>
       jdbc:mysql:///enterprisejavabatch
   </param-value>
 </context-param>

 <context-param>
   <param-name>user</param-name>
   <param-value>root</param-value>
 </context-param>


 <context-param>
   <param-name>password</param-name>
   <param-value>root123</param-value>
 </context-param>
```

ServletConfig

=> This object is used to store the configuration details of a particular servlet like logical name of the servlet,initialization parameters, and so on...

=> Using ServletConfig we will get to know the complete view of a particular servlet.

=> Loading ===> static block

instantiation ===> public Zero argument constructor

Created automatically after Instantiation and injected while Initialisation

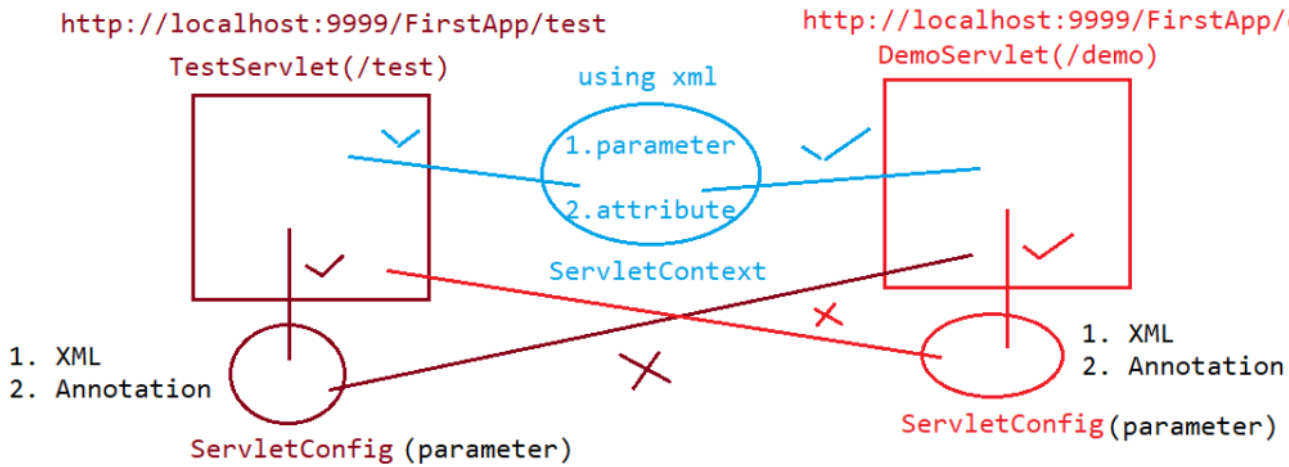
initialization ====> public void init(ServletConfig config) throws SE

RequestProcessing==> public void doXXXX(HSR request, HSR response) throws SE,IOE

DeInstantiation ==> public void destroy()

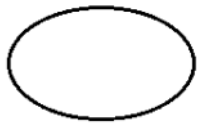
=> ServletConfig object is specific to a particular Servlet.

ServletConfig object will be destroyed just before ServletDeInstantiation



Note: ServletConfig object will be destroyed just before ServletDeInstantiation

1. public Enumeration getInitParameterNames()
2. public String getInitParameter(String name)

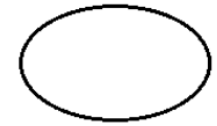


```xml
<servlet>
        <servlet-name>TestServlet</servlet-name>
        <servlet-class>in.ineuron.controller.TestServlet</servlet-class>
        <init-param>
            <param-name>jdbcurl</param-name>
            <param-value>jdbc:mysql:///enterprisejavabatch</param-value>
        </init-param>
        <init-param>
            <param-name>user</param-name>
            <param-value>root</param-value>
        </init-param>
        <init-param>
            <param-name>password</param-name>
            <param-value>root123</param-value>
        </init-param>
</servlet>
```

ServletConfig

jdbcurl
user
password

```xml
<servlet>
        <servlet-name>DemoServlet</servlet-name>
        <servlet-class>in.ineuron.controller.DemoServlet</servlet-class>
        <init-param>
            <param-name>Company</param-name>
            <param-value>iNeuron</param-value>
        </init-param>
        <init-param>
            <param-name>CourseName</param-name>
            <param-value>EnterpriseJava</param-value>
        </init-param>
        <init-param>
            <param-name>Mentor</param-name>
            <param-value>NavinReddy</param-value>
        </init-param>
</servlet>
```

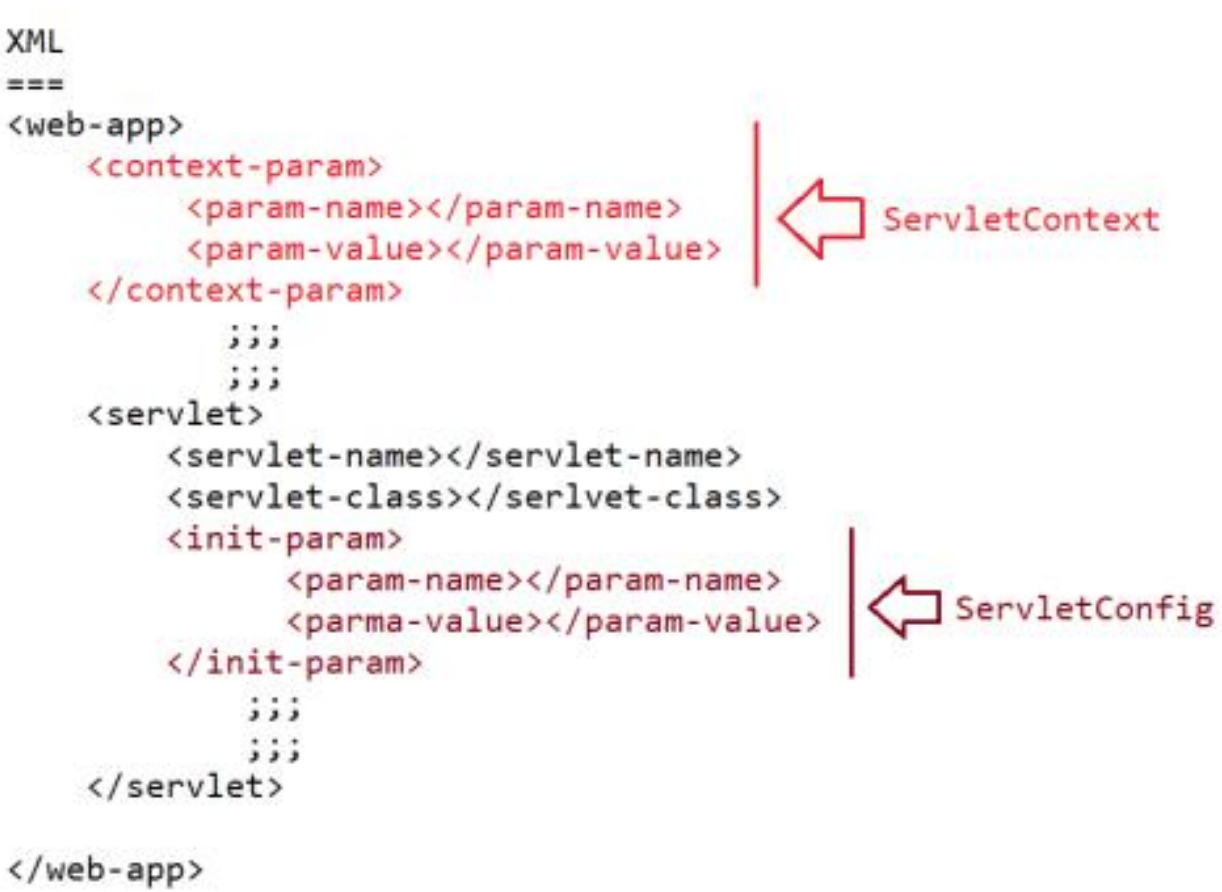


ServletConfig

Company
CourseName
Mentor

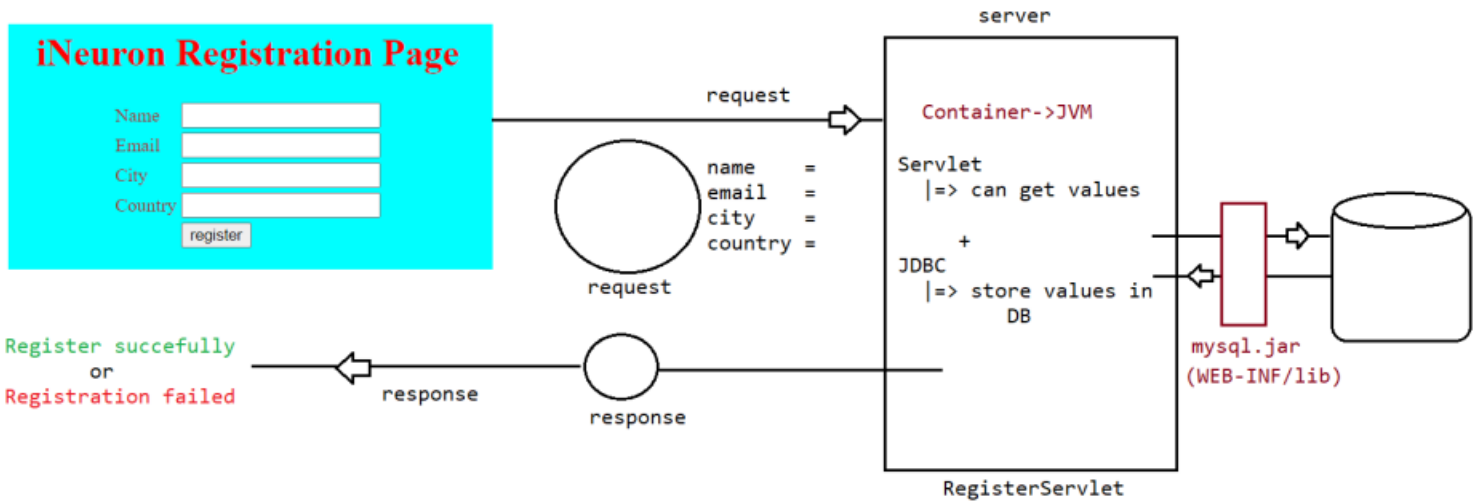
```java
@WebServlet
(
   urlPatterns = { "/test" },
   initParams = {
      @WebInitParam(name = "jdbcUrl", value = "jdbc:mysql:///enterprisejavabatch"),
      @WebInitParam(name = "user", value = "root"),
      @WebInitParam(name = "password", value = "root123")
   }

)
```

```java
@WebServlet
(
    urlPatterns = { "/demo" },
    initParams = {
            @WebInitParam(name = "Company", value = "iNeuron"),
            @WebInitParam(name = "course", value = "Enterprisejava"),
            @WebInitParam(name = "mentor", value = "NavinReddy")
    }


)
```



- Any changes in web.xml context-param doesn't reflect in the Application unless you restart the server and redeploy the Application. Because ServletContext Object is created when you deploy the Web Application.
- Similarly any changes in web.xml init-param doesn't reflect in the Application because ServletConfig object is created during initialization.
- Everything present in web.xml is static you cannot change during runtime. But you can see the change done to dynamic attributes of ServletContext during runtime.
- ServletConfig doesn't have attributes like ServletContext

DEVELOP AN APPLICATION USING SERVLET + JDBC



ServletCommunication

(request-response) Browser-Servlet Communication

Web-Component Mechanism or Request Dispatching Mechanism

include

forward

RequestResponse

SendindgError Messages

RequestReDirection

using Hyperlinks

SunMS --> 9999/sunms

OracleApp--> 9999/index.html

Setting Response Headers ✗

deprecated

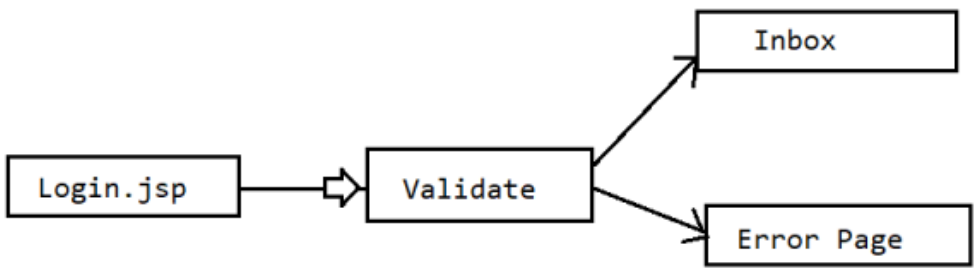
using RedirectMechanism ✓ response.sendRedirect()

Servlet

Login page
Validation
Inbox
ComposeMail
MailSending
ErrorPage
;;;;
;;;;
;;;;

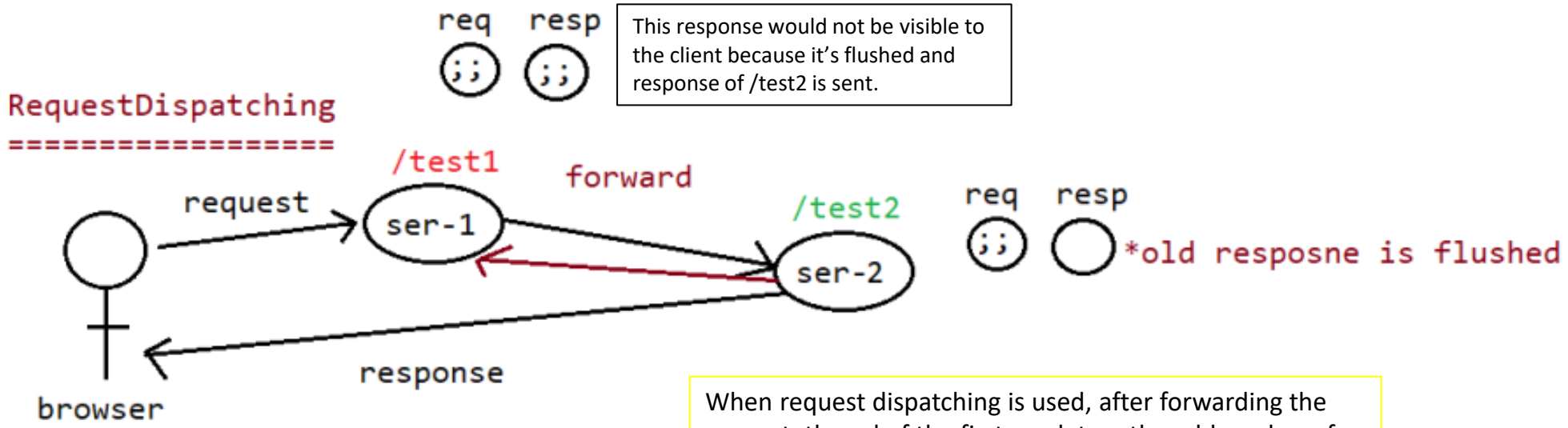
Maintainence is not easy
debuging becomes difficult
enhancement becomes difficult

Communication happening between webcomponent



Note: For server the default landing page of the application is "index.html".
meaning if we send the request then this page would be loaded automatiaclly.
eg: http://localhost:9999/FirstApp/index.html

not required to explicitly write



When request dispatching is used, after forwarding the request, the url of the first servlet on the address bar of browser is not changed to the second servlet is not.

RequestDispatcher Object can be obtained in 2 ways

a. ServletContext

```
SerlvetContext context = getServletContext();
RequestDispatcher rd   = context.getRequestDispatcher("/test2");//relative path only
```

b. ServletRequest

```
RequestDispatcher rd = request.getRequestDispatcher("/test2");// relative path
RequestDispatcher rd = request.getRequestDispatcher("test2");//  absolute path
```

/test1

```
RequestDispatcher rd = request.getRequestDispatcher("/test2");
rd.forward(request,response);
```

/test2

```
RequestDispatcher rd = request.getRequestDispatcher("/test1");
rd.forward(request,response);
```

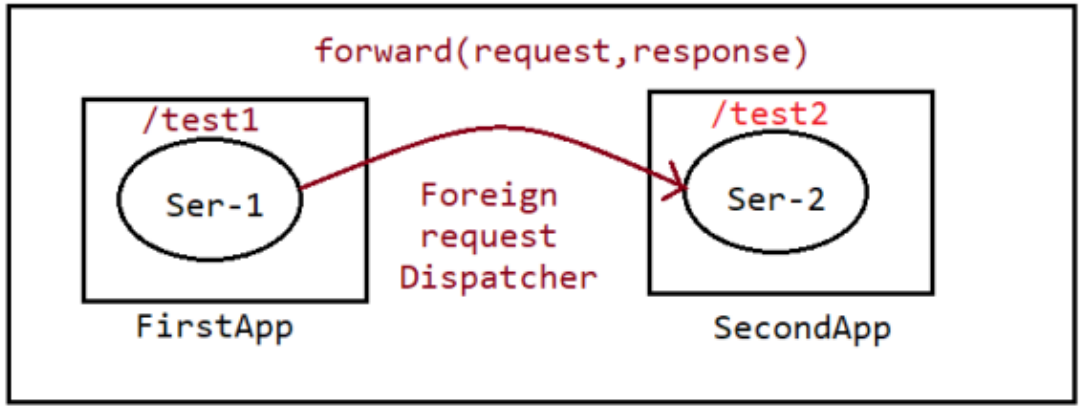
StackOverFlowError

Relative Path : It indicates we need to write path from the current directory(./ or /)

Absolute Path : It indicates we need to write path from the root.

Using absolute path is a good practice.

After commiting the response, we can't forward the request, if we try to do it would result in "IllegalStateExeption.



By default Foreign Request Dispatcher support is not available in eclipse

To see the effect just make the following changes

open tomcat/conf folder

a. open context.xml file

b. update as shown below

```
<Context crossContext = "true">
    ;;;;;;;
</Context>
```

1. sendRedirect(request,response) Location of redirect
   different app and those applications can run in different server also.

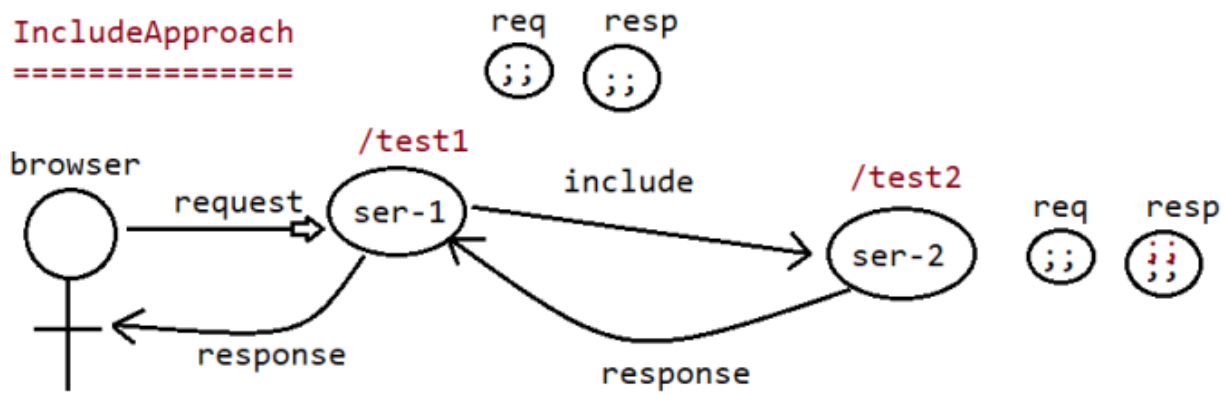
2. forward(request,response)
   different app and those application should run compulsorily in the same server.

For Dispatching request to servlet present in another web application use this:

getServletContext().getContext(/WebApplication Name).getRequestDispatcher("/urlpatternofservlet")

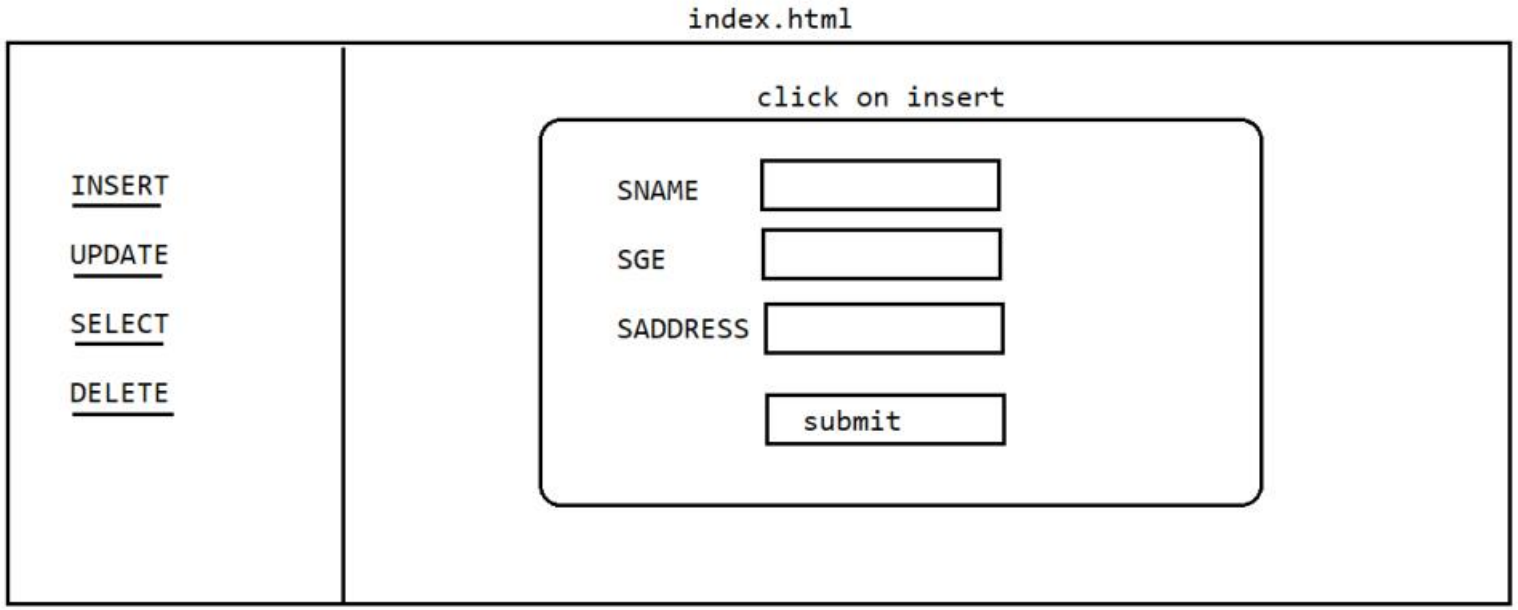
With the above dispatcher you can use forward/include

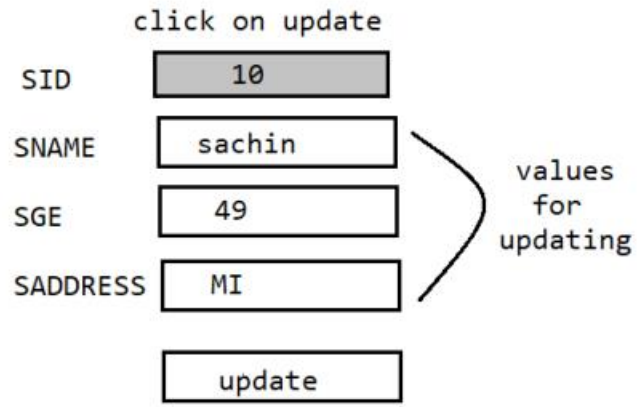
Note: different application means use Context object only.



Total response => response from ser-1 + response from ser-2

CRUD Application





select => display in table format

SID

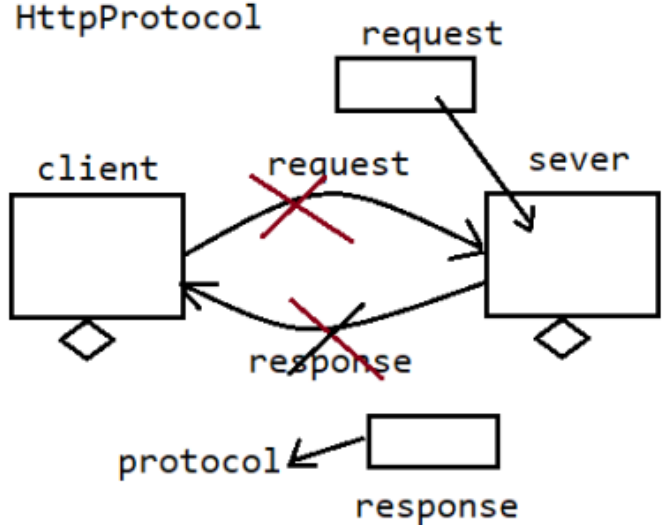
select

delete

SID

delete

Working with HttpProtocol



1. At the time of processing the later request, if we want to get the data of previous request, then we need to go for "Session Tracking"
2. Every server side technology should support session tracking because the protocol used is http and it is "stateless" in nature

To keep track of all the session objects at the server machine, we need a set of explicit mechanisms called as "Session Tracking Mechanism".

a. HttpSession Tracking mechanism
b. Cookies Session Tracking mechanism
c. URL-ReWriting Session Tracking mechanism
d. Hidden form field session Tracking mechanism

Servlet-api level

Used in real time applications

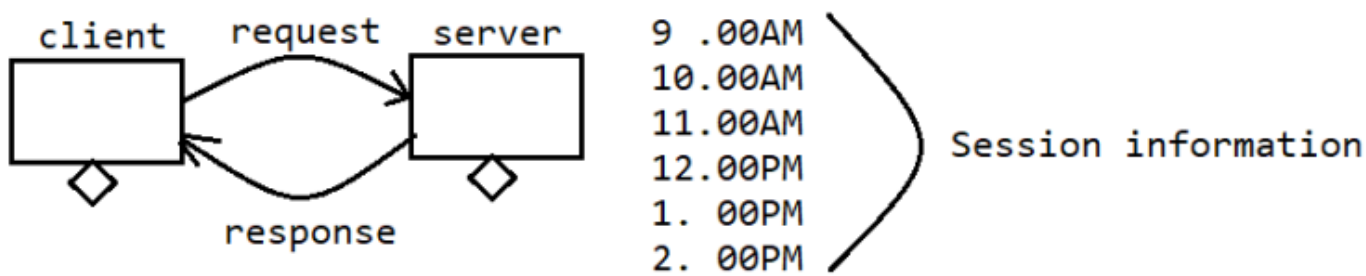
purely devloper level

once the server sends the response to the protocol, server will forget the client data because the Http protocol is "Stateless".

It would assume every request as the new request

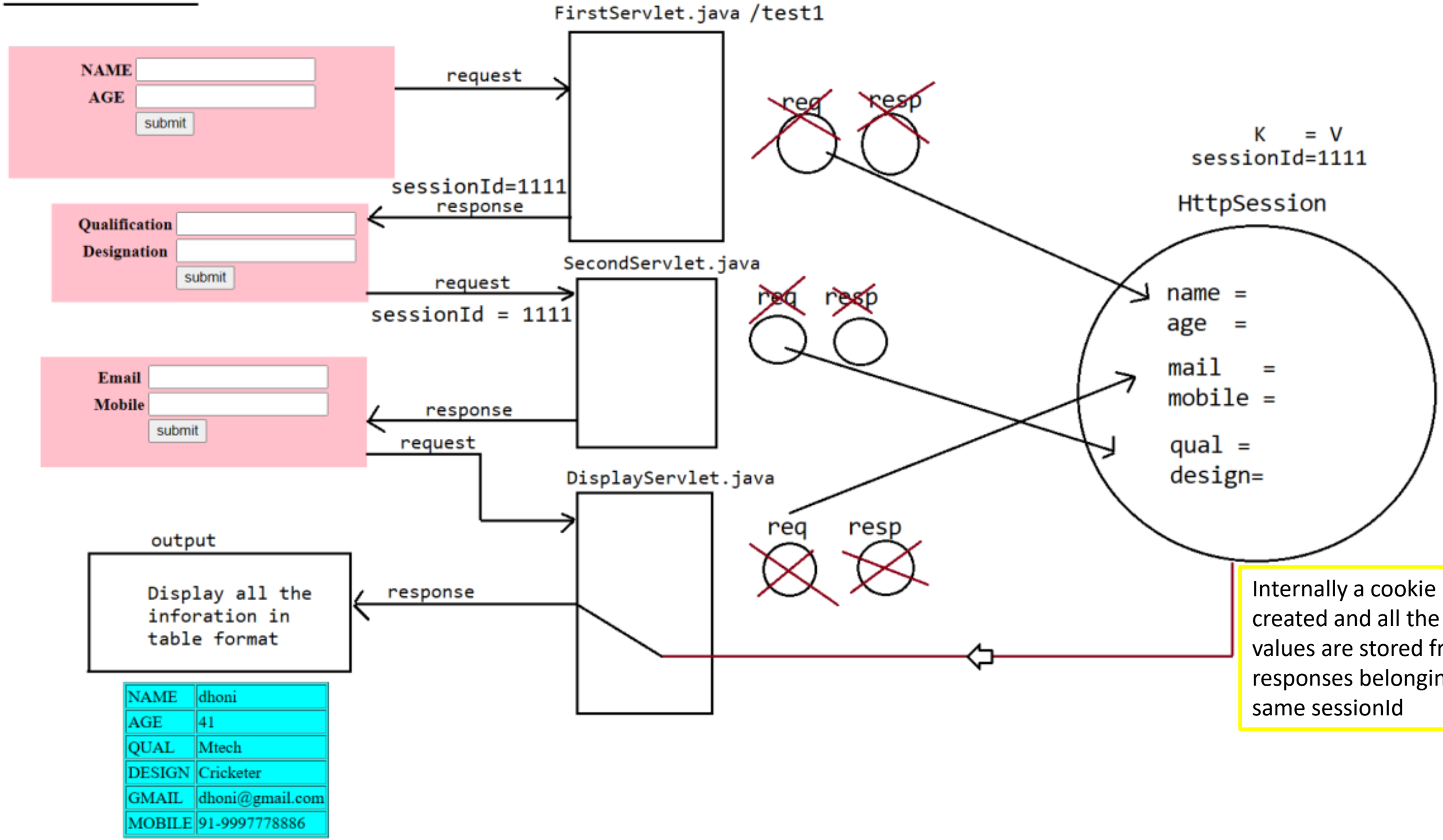
What is Session?

Sesison is a time duration, it will start from starting point of the client conversation with server and will terminate at the ending point of client conversation with the server



The data which is transferred b/w client to server through multiple request during a particular session then that data is called "State of a session".

HttpSessionTracking

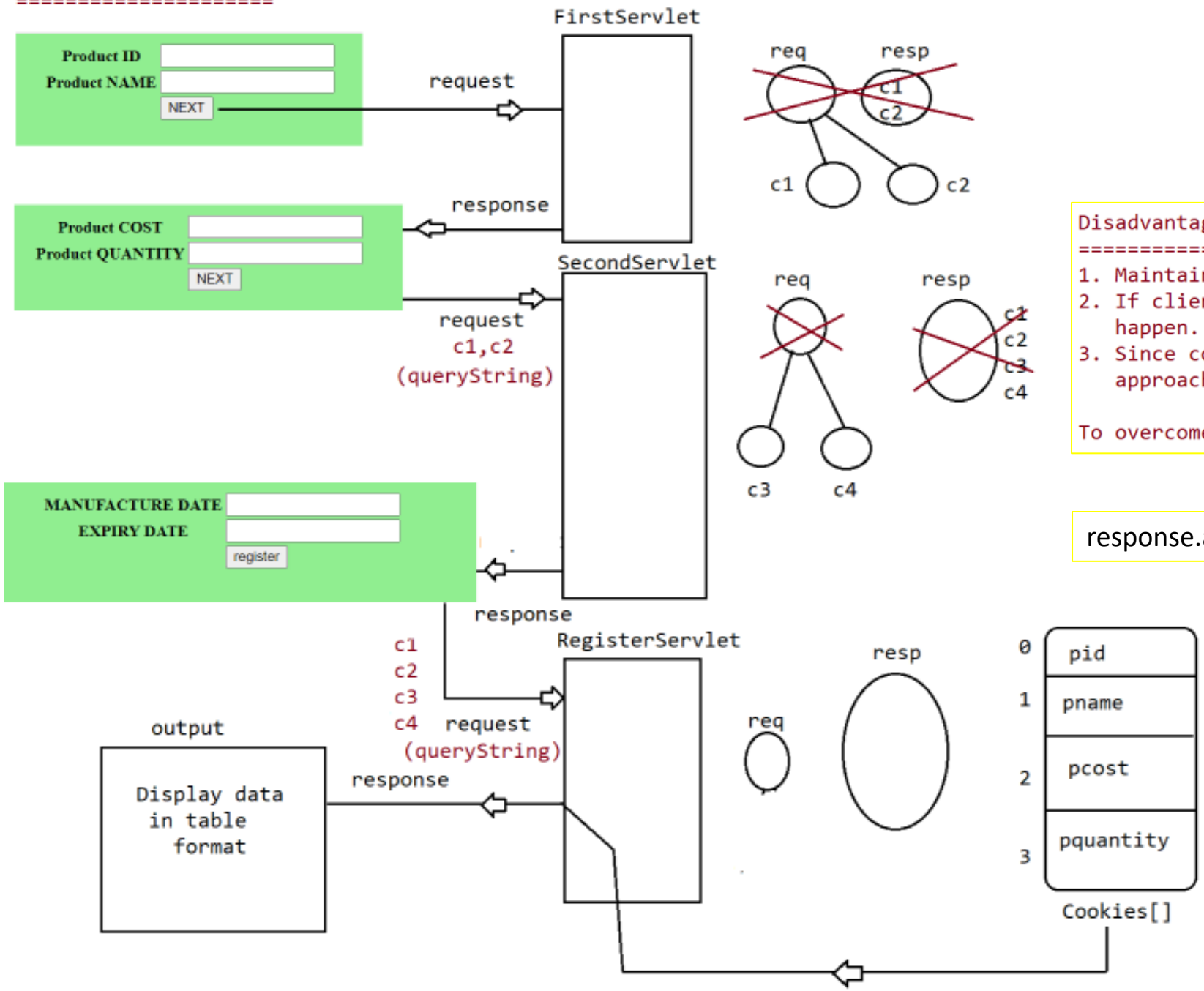


NAME	dhoni
AGE	41
QUAL	Mtech
DESIGN	Cricketer
GMAIL	dhoni@gmail.com
MOBILE	91-9997778886

Internally a cookie is created and all the values are stored from responses belonging to same sessionId

```
HttpSession session = request.getSession()

getSession() -> The container will check whether any HttpSession object existed for particular user or not.
                if any httpsession exists then the container will return the existed HttpSession object reference.
                if no httpSession exists is existed for particular user then container will create a new HttpSession
                object and returns the reference.

getSession( false )
             -> The container will check whether any HttpSession object existed for particular user or not.
                if any httpsession exists then the container will return the existed HttpSession object reference.
                if no httpSession exists is existed for particular user then it would return null.


                Disadvantages
                =============
                1. More the no of users, more would be the session object
                2. More the session objects, those object will be in the server
                   memory.
                3. More the session objects, maintainence would be difficult at the
                   server side.

                To resolve this problem we use "CookieSessionTracking" mechanism.


                In case of HttpSessionTracking mechanism, if the client
                disables cookies then HttpSessionTracking mechanism won't work.
```

CookieSessionTracking
==========



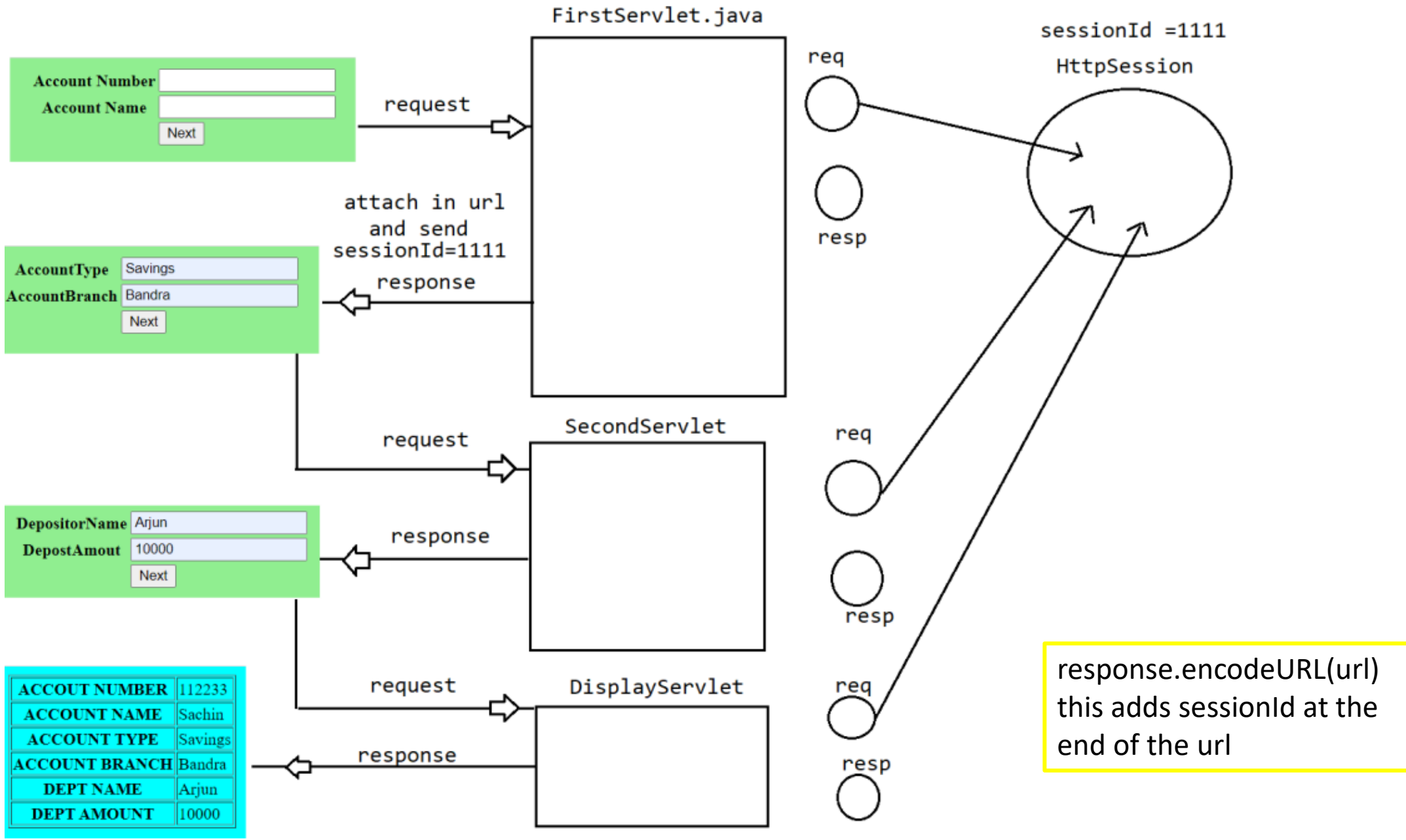
Disadvantage
==========
1. Maintainece of client data is happening at the broswer
2. If client disables cookies, then SessionTracking won't happen.
3. Since cookie is exposed to the client, it is not a safe approach.

To overcome this limitation we need to use "URLReWriting".

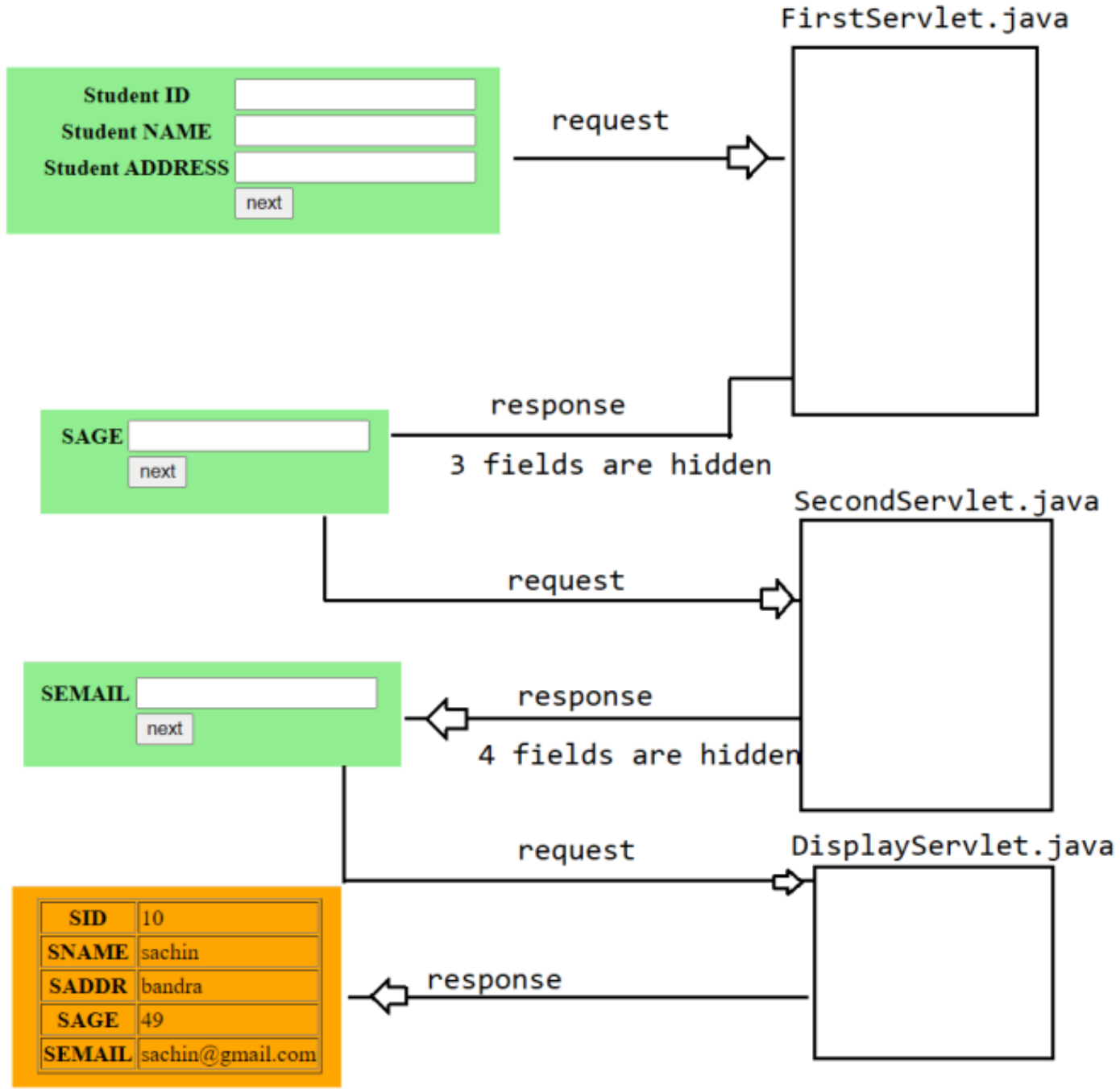
response.addCookie(new Cookie(param-name, value))

Both httpsession tracking and cookiesession tracking mechanisms are based on cookie cooncept

URLReWriting Mechanism



Hidden Form Field

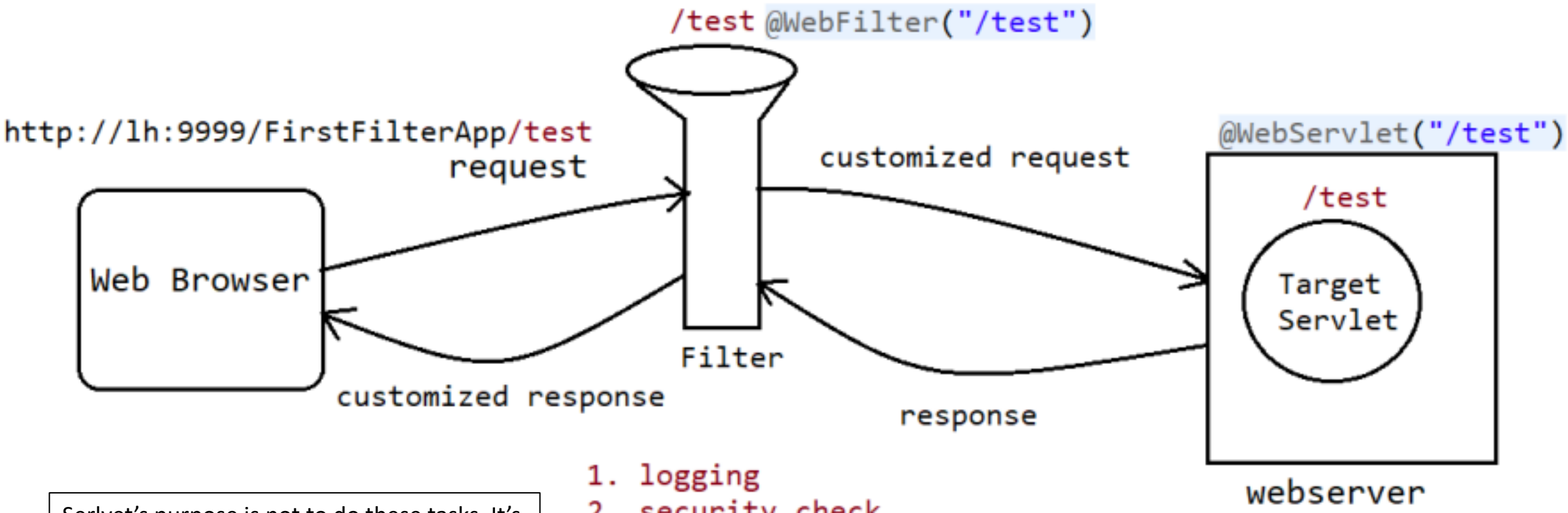


<input type='hidden' name='' value=''/>

Filter

IMPORTANT FOR SPRING AND MVC

It can be used for PreProcessing the request and Postprocessing of request before they reach the target resource of the webapplication.



1. logging
2. security check
3. altering the request information
4. compressing the response
5. encryption of response
6. authentication..

Serlvet's purpose is not to do these tasks. It's purpose is to be as controller. That's why we need one more component between client and server that is Filter. It is present inside Web Container.

javax.servlet.Filter is an interface

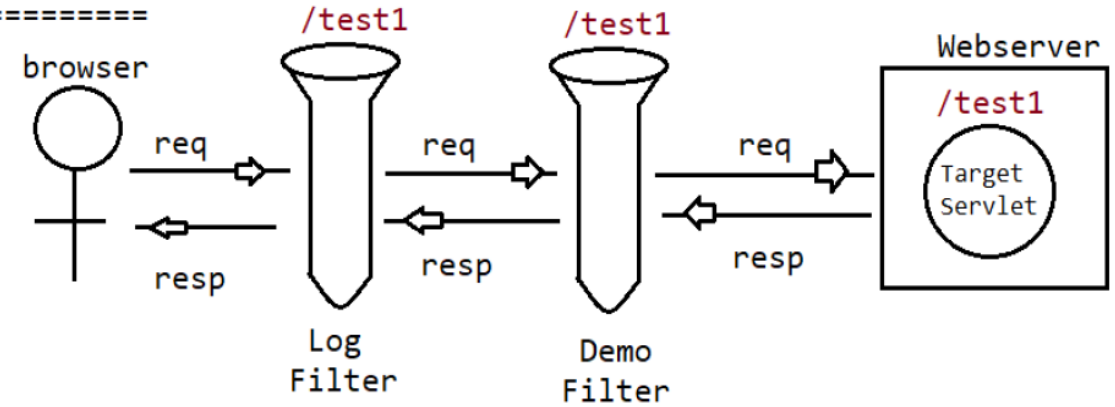
```
void init(FilterConfig config) {

}

void doFilter(ServletRequest request, ServletResponse response, FilterChain chain) throws ServletException{
//preprocessing logic (before servlet executed)

chain.doFilter(request,response);//pass the request along the filter chain

//postprocessing logic (after servlet executed)
}

void destroy(){

}
```

url pattern for both servlet and filter must be same for mapping those filter and servlet.
@WebServlet(x)
@WebFilter(x)

If there are multiple servlets and one filter to be mapped with all those servlets use asterisk(*) as url pattern for filter.
@WebFilter(*)

```
Whenever we are sending the request to Target Servlet, container will check whether
any filter is configured for this servlet or not.
if filter is configured then container will forward the request to filter instead of
servlet
      void doFilter(req,resp,chain) throws SE,IOE

After executing the above logic, filter forwards the request to TargetServlet.
After processing the request by TargetServlet the response will be forwarded to the
Filter instead of browser.
After executing the Filter logic, Filter forwards the total response to the browser.
```

FilterChain
==========



Order of Execution of Filters in XML
====================================

1. identify all the filters which are configured in url-pattern & execute all these filters from top to bottom
2. identify all the filters which are configured according to servlet-name and execute all these filters from top to bottom

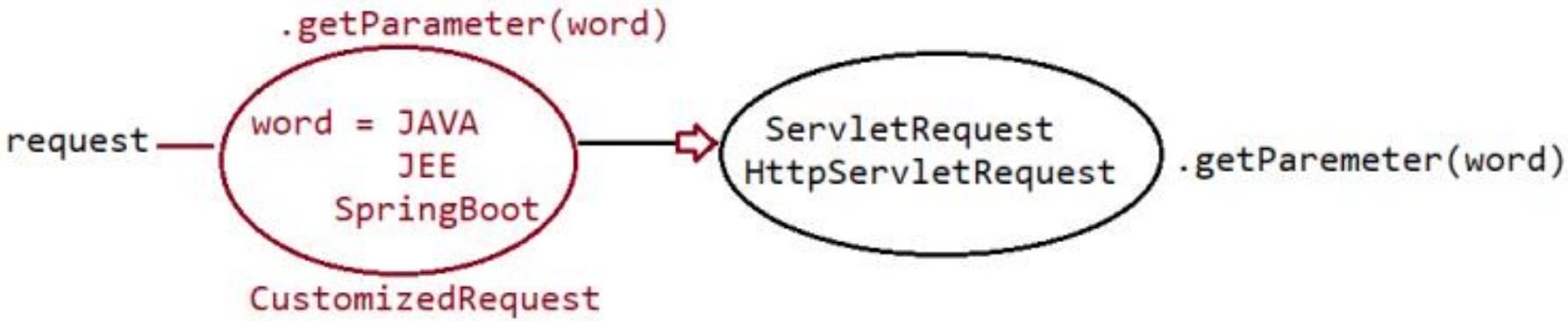
Order of Execution of Filters w.r.t Annotations
===

1. Filters are executed based on the Alphabetical names of the Filter names.

1. loading of filters is not at the user level, it is at the container level.
2. Filters will loaded automatically even with out load-on-startup.

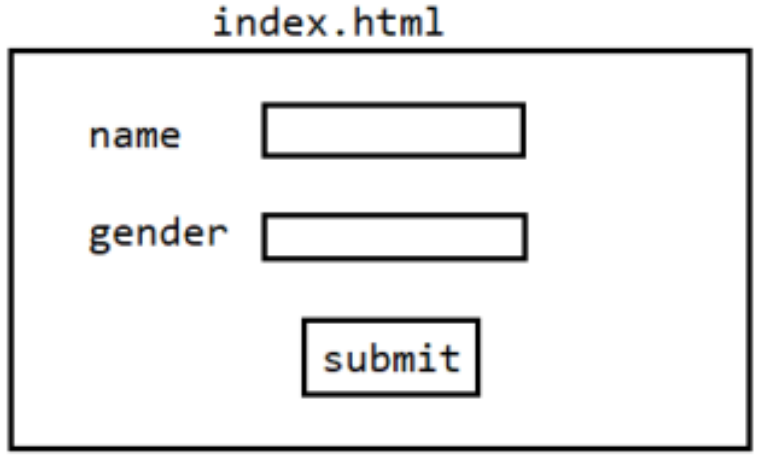
learn about FilterConfig.getServletContext()

HttpServletRequestWRapper



Listeners

1. Event



Something which happens on a webpage is called as "Event".
It can be mouseclick,typing some text inside textbox,hovering on the page,.....

1. RequestListener
2. ContextListener
3. SessionListener
4. ServletRequestAttributeListener

Event =========> class
Listeners======> Interface

request ===========> to particular servlet
SRE
any servlet u send request object should be created.

ServletReqeustListener

requestInitialized => At the time of request Object creation
requestDestory => At the time of request object destruction