

JEE Web context Structure -----

Tomcat / Weblogic / WildFly ,..... --- installations are different , folder structures are different

But

Folder structure of WAR file and the extracted WAR file = WEB **CONTEXT** - application ---- is **SAME FOR ALL JEE compliant web servers !!!**

At the time of deployment ----

The war file is copied to WEBAPPS

The tomcat container extracts the contents of war file ----

For example SWP.war -----> SWP folder --- Web **CONTEXT** ---web application

JEE structure of the WEB CONTEXT /APPLICATION / SWP folder in TOMCAT !!!

SWP

-----*.html

-----css

-----*.css

WEB-INF

----- web.xml

-----lib

-----JAR files that the tomcat/lib does not have BUT the application Using
mysql-connector.jar

-----classes

-----java packages with .class files

WAR file is the **PRODUCT** that can be deployed on any JEE compliant server !!!!

IDE ----- Eclipse , Netbeans , IntelliJ --- folder structures are different

But when WAR is created then the folder structure is as per above JEE context structure !!!

Container is a Java class(es) ---

It reads the annotation OR web.xml and accordingly manages life cycle of servlet components

Java class **Class** = this is used for **INTROSPECTION** /**REFLECTION** !!!

AT run time ---- my code can peep inside the .class file and do the following

1. Find the metadata = details of the class
 - i. What are the fields
 - ii. What are the methods
 - iii. What are the overloaded constructors
 - iv. What is the super class, what interfaces are implemented
 - v. Is annotated, which annotation is added
2. Invoking the methods at run time
 - i. method.invoke(.....)
3. Creating **object without using new** when class name is given at run time
 - i. clsobj.newInstance()

WE cannot create the class Class object !!!WHY ? No public constructor !!!

class Class object is AUTOMATICALLY CREATED by the JVM

WHEN ?? When the .class is LOADED into the JVM class area by class loader

How to get the reference of the class Class object ???

3 ways =

Class obj = Class.forName(classname) ;

Class obj = java.lang.String.class ;

Class obj = new Thread().getClass() ;

Servlet Chaining using Request Dispatcher ----

Chain -----

Browser(Request)

|

Http Request (HR)

|

FirstServlet --(HR)--->SecondServlet----(HR)----->ThirSelvet--(HR)--->NthServlet

|

|

Http Response (Hresp)

|

|

Browser

REQUEST does not KNOW that it will be sent to many servlets !!

CHAINING is transparent to browser !!!

The SAME **request object is SHARED** by many servlets during that request !!!

To implements chaining the servlet MUST talk to the container !!!

This can be done using an API --- `getServletContext()`

Browser -- <http://localhost:8080/chain/First>

|

|

Request object

|

|

FirstServlet

|

| `request.setAttribute("name", value)` //added name value pair to the request

|

SecondServlet

|

| `request.getAttribute("name")`

| `request.setAttribute("name", value)`

|

ThirdServlet

```

|
| request.getAttribute("name")
|
| response.getWriter().append()
|
Browser

```

This chaining is called as FORWARD chaining

We call `rd.forward()`

The response is generated by the FINAL SERVLET in the CHAIN

That response is sent to browser by FINAL SERVLET

The same request object is SHARED by all servlets !!!

HW -----

Write a program ----

1	User calls a LoginServlet GET	User gets login page	
2	User enters login details clicks login -- LoginServlet Post is called	LoginServlet Post will verify the login details	
		If correct forward chain to HomeServlet	HomeServlet POST will response as WELCOME Username
		If wrong then forward to ErrorServlet	ErrorServlet POST will response INCORRECT Login

Include Chaining

We call `rd.include()`

The response can be generated by ALL the SERVLETs in the CHAIN

That response is sent to browser by FIRST SERVLET

The same request object is SHARED by all servlets !!!

Forward Chaining

```

Browser -----R1 ----First----->R1 ----->Second-----R1 ---->Third
                                                                |
                                                                | Generate Response
                                                                |
<-----Response-----|

```

Include Chaining

```

Browser -----R1 ----First----->R1 ----->Second-----R1 ---->Third
<-----total resp----| <-----resp from2-----| <-----resp from 3rd

```

HW ---

Try the TestClassClass program done in class

Try the Include Chaining example done in class.

Start using embedded tomcat !!!

Running Tomcat From Eclipse ? WHY ?? To speed up development !!!

REAL deployment ----- WE create a WAR file and copy to the EXTERNAL tomcat's WEBAPP

Session Management in Servlets
JSP

HTTP Session -----

Http = STATELESS PROTOCOL

It has no provision to REMEMBER DATA after one request response cycle

Http Session Management -----

Container has to implement some extra logic to remember previous request data between some scope

That scope is called as a SESSION

Between LOGIN to LOGOUT !!!
