JEE

Monday:

Lifecycle of Servlet

Redirection, dispatch and how can we use servlet and jsp

Tuesday:

Basics of MVC architecture tomorrow.

JSP in details

Inter servlet communication and session management

Wed:

End to end case study.

Thursday:

Advanced concepts – filters, listener, connection pool,

Friday:

Implementations of advanced concepts

Prepare for examination

* Container are JVM which has specialized tasks
* Application server has 2 containers:
* Web Containers and EJB containers
* API for Web Containers: Hosting environment is contained in web containers(web server)
  + Servlets
  + JSP- Java server pages
  + JSF – Java Server Faces
  + JTA – Java Transaction API
  + JAAS – Java authentication and authorization services
* Web container will have JRE
* WODA – Write Once deploy anywhere

EJB Container

* Session Bean
* JPA
* MDB

1. Any processing required for Servlets/JSP that would be done by Servlet Engine. It will process and generate HTML response for the Web Browser
2. Browser only understands HTML

Request(x.html)

Web Browser

User HTML Rendering Response Web Server It has web pages

* Web Server has web.xml which will know what type of request it is and then will process accordingly
* Web server has HTML, CSS, JQuery, JS, Servlet, JSP

Tomcat Server

* Bin folder – Start.bat
* Conf -\*.xml, katlina (servlet engine)
* Additional jar files which is relevant to our application is stored in lib folder apart from servlet.api and jsp.api
* Work – This contains all the converted files

Folder Structure

* Webapplication
  + \*.jsp, .html, css, js files
  + Web INF
    - Business logic classes
    - Lib – all the jar files
    - Web.xml

Flow

* Create a folder in Tomcat/webapps folder for the project
* Create the folder WEB\_INF
* When the request is sent it first come to web.xml

Steps to load the class from web.xml

* It will look for the url pattern in servlet-mapping
* It will then see the servlet name in servlet mapping tag in web.xml
* It will look for the same servlet name in servlet tag
* It will then see the servlet class and then load it

Connection Metadata – Read all DBs in the server

DatabaseMetaData – Read All tables in a DB

REsultSetMetaData- read All Columns in a table

Servers

* Http Server – only http static web page. You request a web page and sends a web page.
* Web Server – inside which servlets of jsp can run. A special container (env) is needed. Jsp and servlet has different container.
  + Static pages – they cannot be altered at runtime.
  + Servlet, jsp – dynamic web page designing
  + Service layer – consists of business logic, database logic
  + Web layer – jsp, servlets
* Application server – enterprise arrangements and web arrangements. EJB, servlet and static pages all runs in application server.
* Difference between servlet and jsp – servlet is pure java and jsp abides the markup language. Jsp is for web designer. It looks like html
* Super interfaces from generic servlet

Lifecycle of servlets

Servlet ServletConfig Serializable

Init(SerConfig)

Service(req, res)

destroy

init() – This is implemented for overriding

In generic servlet.This is for overriding

GenericServlet(A)

Deployment

Constructor

Init()

Init(parameter)

Service

HttpServlet extended from genericservlet

Undeployment

Destroy()

Finalize()

Custom Servlet (user designed)

Servlet must be serializable because serializable is used for transferring objects from one heap to another. Hence for load balancing when the servlet has to be transferred from one machine to another then it should implement serializable

Servlet interface has the following methods:

* Init(ServConfig) – This is executed only once. But after the constructor. This will open the connection, resources
* Service(req, res) – When there are many request is coming from client(browser) this method is called
* Destroy()- the connection, resources which are opened in init are closed in this method. It is called exactly before the servlet is detached from the server. Then the garbage collector takes the servlet object and calls the finalize method to destroy it completely.

Http servlet

* Extends generic servlet class
* Service method is already implemented
* http defines a request type. In order to provide the support to http standard this class is used.

Deployment Cycle for Servlet:

Maven, ANT – they deploy the .class file into the webapp folder

Deployment

Warm deployment

Lifecycle

Refer configuration

EDITOR Ant, Maven Web Server

.java .class Copy in webapp

Server Server Server

.jsp .java .class jsp instance

Warm Deployment - monitoring the project and if there is any change in the .java /.class file then the old object is destroyed(undeployment) and new object is created automatically. And deploying the project

The project is deployed in the below location

C:\Users\Administrator\Documents\GitHub\New\_Java\_Repository\_Swati\JEE\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\wtpwebapps

The server (tomcat) is continuously monitoring this folder and any change is there the project is re deployed

\*\*\*\* New Servlet class is created in src folder in the project structure (not package structure)

\*\*\*\* New JSP file is created in Webcontent folder. Outside webinf

Redirection and dispatch

Server Side Client

a

Redirect b a

Status code 30X, location of b(url)

Browser creates a new request for b

b

browser hit the request to server and then b webpage is displayed. For the status code 30X Browser does the redirection

a will not directly jump to b because the status code (30x) is only understood by the browser.

If a has to jump directly to b then it is possible only through dispatch. Where a will receive the request and then it is given to b and then b will publish the response in the form of web page.

Dispatching is done by forward and include methods

Forward method will send the control servlet or jsp

Once a project is done follow the below steps:

* Configuring the project – Right click on the project name and properties, check
  + Web Project settings
  + Targeted
* Add the project to tomcat. Go to server view-> right click Add and remove->

Dispatch

a request is sent to a

response is sent by b

b

//The response if altered by this serlet does not reach to next in chain because Ther forward dispathching wipes off the old response

//In the forward chain, the last entity to generate full and final response.

//The request if altered, is sent to the next in chain as altered.

//Query string : User entries passed through address bar when method type is GET seperated by ? from URI, seperated by & from another field

//User entries passed through seperate connection when method type is POST

//Form Data: User entries passed through seperate connection when method type is POST

// Form data is not exposed in address bar.

//Always send sensitive data/data of large size as Form Data.

//Prefer GET if data is small in size and not sensitive.

//REquest.getParameter() : Brings every data in String form. One method to bring Query string or form data

When you place any folder in WEB\_INF then the Tomcat server forbids that folder and cannot be run directly

We can divide the a single page in multiple JSPs. Here the response is:Normally done for the last component in the chain to collectively build the response.

Package structure =

Writing the servlet in web.xml for server to recognize

<servlet>

<servlet-name>P1</servlet-name>

<!-- Canonical Name which the xml uses to servlet mapping -->

<servlet-class>com.jp.config.servlets.Process1</servlet-class>

</servlet>

Configuring the servlet

<servlet-mapping>

<servlet-name>P1</servlet-name>

<url-pattern>/process1</url-pattern>

</servlet-mapping>

<servlet-mapping>

<servlet-name>P1</servlet-name>

<url-pattern>/process2</url-pattern>

</servlet-mapping>

Context parameters are accessible to all but config parameters are only for the servlets

|  |  |
| --- | --- |
| Context Type | Config Type |
| It can provide configuration | It is just for giving private data |
| It can provide access to server side services/resources |  |
| It can also provide public storage area |  |

Servlet object is created when request is send by the server

* Cookie is the storage in the user (client) machine
* When the user fills a form and doesn’t submit the form the information is stored in the client (user) machine. And when the user submits the form it is stored in the server machine
* Here cookies are created by servlets. And will send the cookies along with the response to the browser (client) and then browser will store the cookies. And from the next request cookies are added to the request and give it to the server to the servlets
* Cookie is a class given in servlet and we can put information in it.
* Cookies can hold text only.
* A cookie can hold 4 MB
* For one website max 20 cookies can be stored. When 21st cookie comes the oldest one is deleted.
* For a browser max 200 cookies can be stored.

There are 2 types of cookies:

* Persistent Cookies – Age is set and go to the disc.
* Session Cookie - Age is not set and does not go to the disc. They are held at the memory (RAM) of the browser.

Assignment

Product table

* Product ID
* Category
* Name
* Price

Insert the data in the table

JSP Basics

* It is java embedded in HTML. It is added for dynamic pages. It can be done by java or JavaScript. Client side is managed by JavaScript. Server side dynamism is handled by java.
* JSP is executed at the server side
* Can go with JavaScript, JQuery, Sprite( for image management)
* JSP Components:
  + HTML Code – HTML template
  + Scriptlet: <% Java Code %> Goes into the service method of Servlet in the order of their mention.
  + Directives : <%@ page, include, taglib. %>
  + Declarations: If we need to override the init and destroy methods in JSP declarations is used. <%! Instance fields, override init()/destroy(), define custom method %>
  + Expressions : <%= Math expression/method returning a value%> whatever the value it receives it echo to the method.
  + JSP elements : <jsp:xxx/>
* Directives:
  + Page
  + Include: pre compiled inclusion of JSP code
  + Taglib : To define URL for library
* Expression language : ${}
* Standard tag library: Pre written library of java code. We call these methods by markup syntax.

Cookies

Client Server

Theme

* Javascript/scripting language generate cookies.
* Java at server side generate cookies.
* Transferred Server to client through response, transferred from client to server thru next request
* When site is hit on the browser they send the cookies to the browser. When the browser is closed then the cookies gets deleted. They are non-persistent cookies.
* They don’t carry virus.
* Persistent – Disk(age)

Sessions

* When the request is sent through http protocol, then there are multiple connection opened to the server. Now the server doesn’t know the request is coming from which user. To make server identify the user, we need to create a session
* There will be a token added to each request. So that data if we need to preserve will go into a session space allocated to that request.
* To allot a token and space to a user on the request that is called session created.