War file

Equivalent to jar for a web server (tomcat) hast static files (html, css, js, jpg)

WEB-INF

Hidden files for the web app, cannot be seen by outside the app

Folder classes, classes of the project

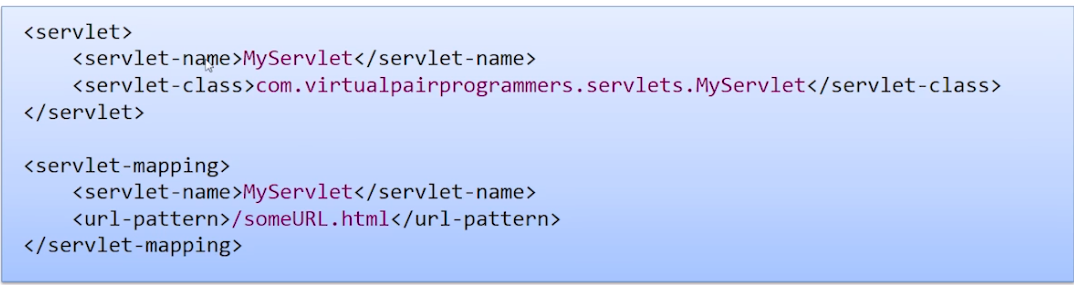
Lib folder, jar files needed on the project

META-INF

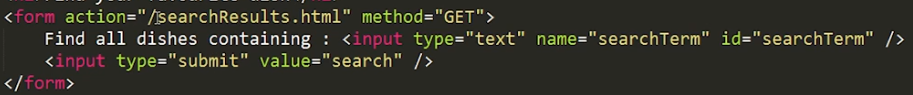
Any information about the war

Manifest.mf information about the war file

Servlet



Submit form

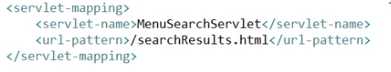


You can request any resource inside webapp from the server directly, contextroot/resource

Request.getParameter(“paramName”)

Allows to get the param submitted

Executing a servlet on a html path



doGet, doPost

methods on servlet

sendRedirect



Changes the url, redirection from browser, request values are lost

Website is stateless, after serving the request forgets everything about us

Ssh is statefull, server keeps information about us

Cookies

Only store text

Limited in size

User don’t accept them sometimes

Session, key value (string, object) value we can store and keep available

Returns object when getting the attribute





Abandoned sessions

We can decide how long a session will last before it is destroyed

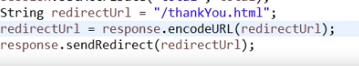
0 (default) never expire the session

 (in minutes)

Users with turned off cookies

Way to confirm user with cookies turned off ( set something to the session and try to retrieve it)

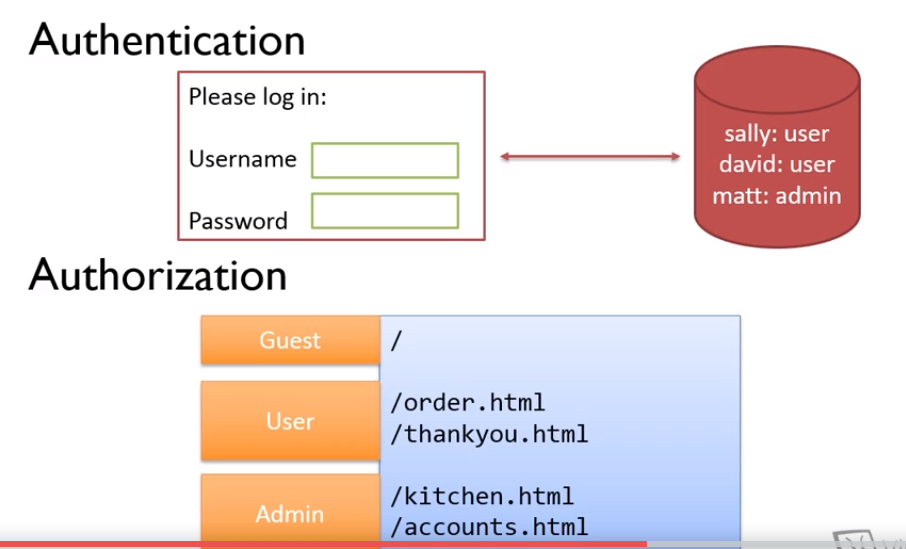
Also we can add the JsessionId (for the cookies turned off) on the url so the session will work even with the cookies disabled



Session will be able to read the jsessionid from the url instead from the cookie

Would need to add it on all pages (not recommended) jsession can be send to other person which will be a mass\

Servlet Security

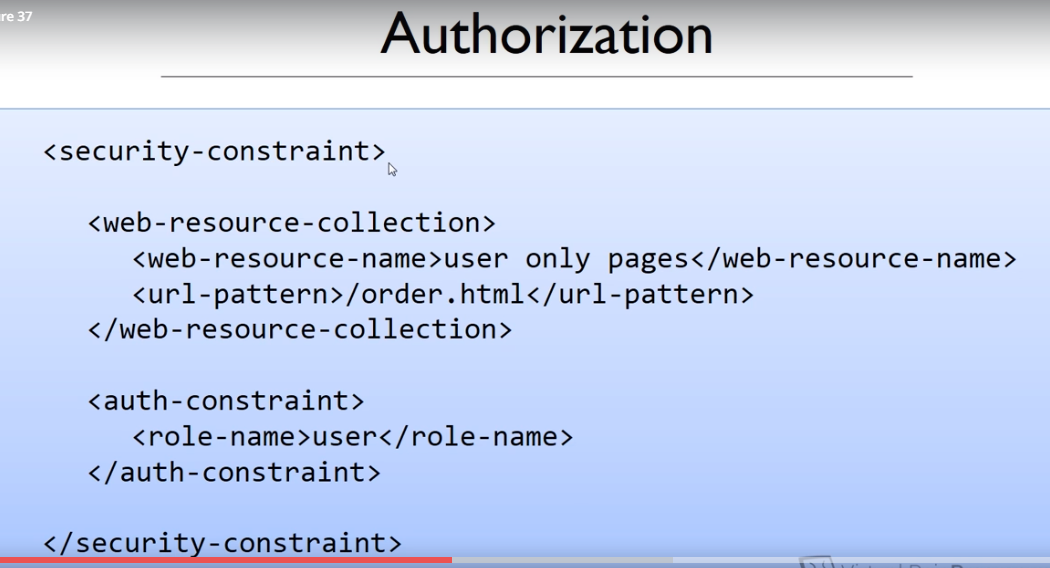


You can specify roles to urls to control where they can access and where not

Pretty basic on servlets, for more sophisticated you should use a framework

Authorization

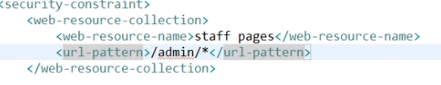
On web xml provide a set of roles



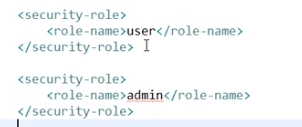
You can add multiple url-pattern for the role to have access to them



You can use wild cards as well (any resource in admin or its subfolders)

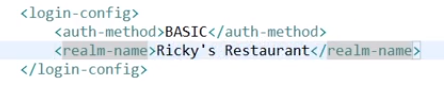


Specify roles available

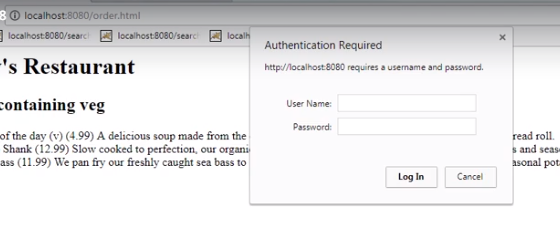


<login-config> Way for the users to be logging in

Basic authentication



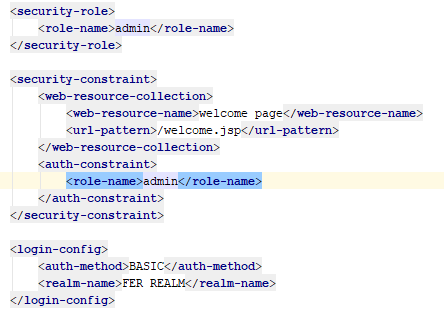
In basic authentication the browser expects the application to ask for the user and password



Way to define users, on Tomcat

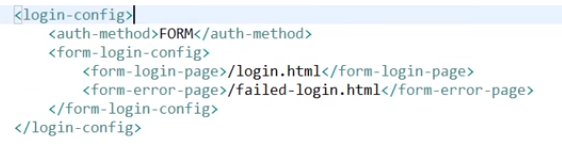


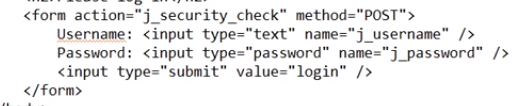
tomcat.addUser, tomcat.addRole creates the user and role on the server



lOGIN WITH A FORM

Allows to link a jsp into the authentication method





j\_SECURITY\_CHECK: STANDARD AUTHENTICATION MECHANISM (NEED TO FOLLOW THE CONVENTION J\_USERNAME, j\_pASSWORD

WITH THIS AUTHENTICATION MECHANISM, IF YOU TRY TO ACCESS A SECURED BY ROLE PAGE, IT WILL REDIRECT YOU TO THE LOGIN PAGE

Servlet With Annotations

@WebServlet

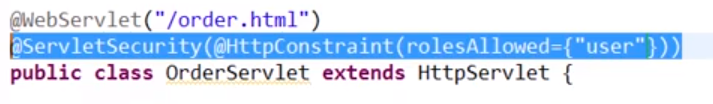
Servlet config which replaces the web xml configuration of the servlet



Value refers to the urlpattern map to this servlet

Authorization Annotations

You need to add the annotation @ServletSecurity on the servlets which you want to be secured by a role, it will replace the security constraint from the web xml



Note: security role and login config are not part of servlets specification, they need to stay on the web xml

Filters

Lets us intercept the request before the servlet executes

Inspect and change content of request or response objects

Ex: logging

Methods: destroy, init, doFilter

Init:

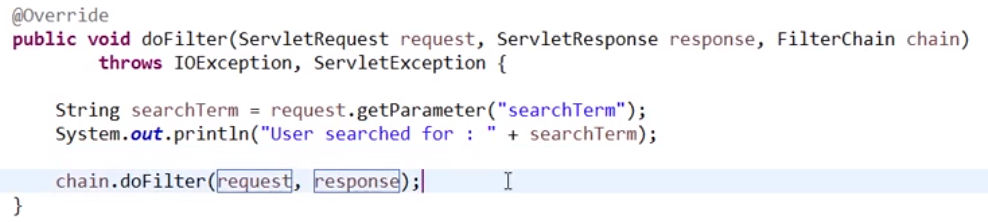
Called by the web container to indicate to a filter that it is being placed into service. The servlet container calls the init method exactly once after instantiating the filter.

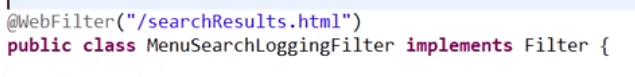
destroy():

When the servlet container determines that a servlet should be removed from service, it calls the destroy method of the Servlet interface to allow the servlet to release any resources it is using and save any persistent state. For example, the container may do this when it wants to conserve memory resources, or when it is being shut down

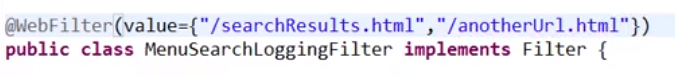
doFilter(req, res, filterChain):

filterChain, object which controls and says when we can go to continue to the next filter or the servlet calling the doFilter method.



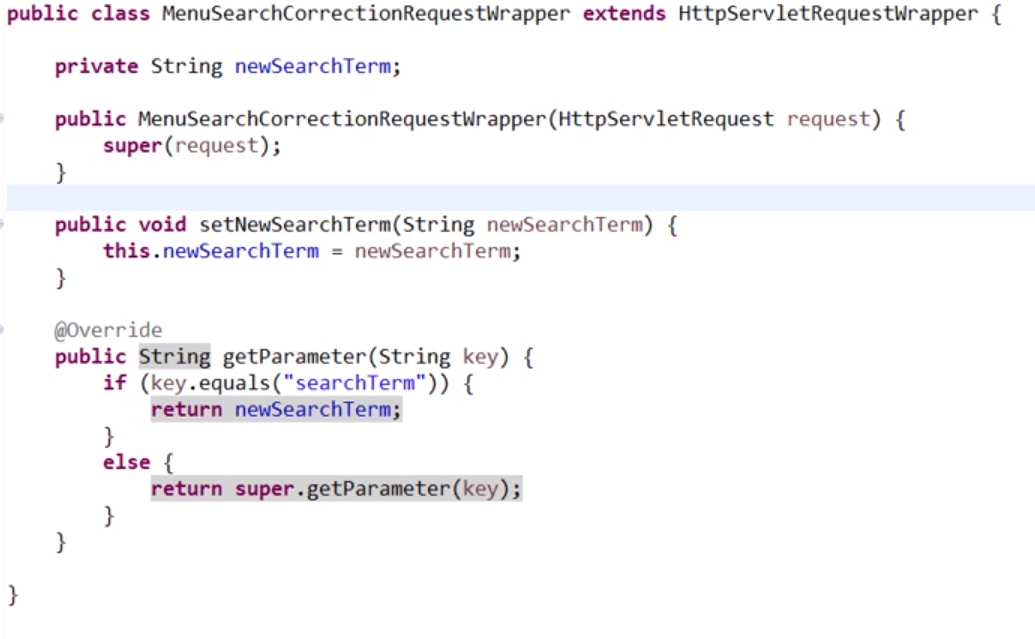


whenever the value (/searchResults.html) is called, first it will come to the filter and then will look for a servlet matching the url

filter to multiple servlets 

HttpServletRequestWrapper

Request wrapper which allows to overwrite parameters sent by the user



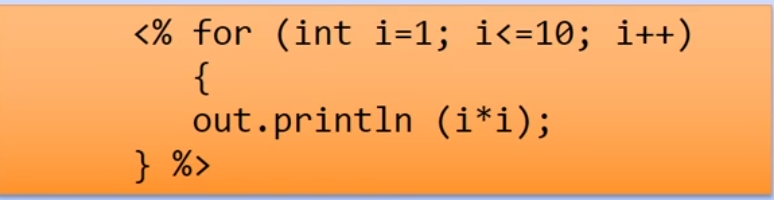


No way on choosing which filter will run first on equal url pattern if using @WebFilter

Using it on web xml will execute the one coming first in order of appearance

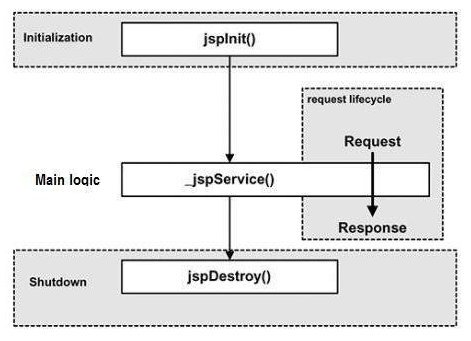
Scriptlet

Allows to add java code inside the jsp



Jsp

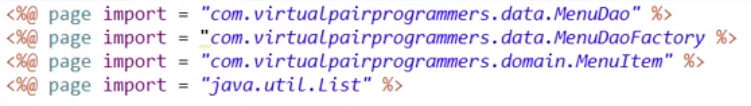
Translation, compilation, initialization, execution, cleanup.



Jsp is compiled into tomcat folder

Page Directive

Allows to import classes



Expression

Allows to print a value (void not allowed)



Declaration

Code inside is placed outside the service() method

**<**%! int data=50; %**>**

jsp:include

allows to include a jsp at runtime





Exists on the HttpServlet, gets you the servlet context of the app

Jstl



C:out

Prints something calling out.println



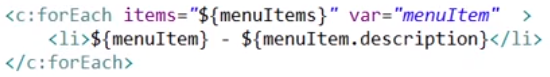


Any text is actually interpreted as c:out

c:forEach

items collection which we will loop through

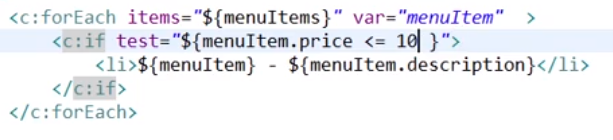
var local var we can use for each element inside the loop



${menuItems} call the toString of the menuItem class

c:if

allows to print what is inside it if the test is true



fmt:

allows to add format to the values



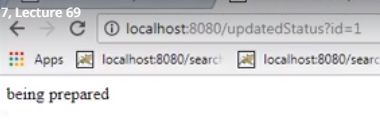


Ajax

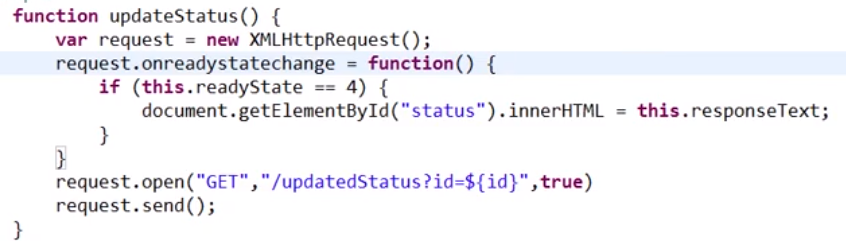
Allows to update a piece of html instead of the whole page



Resource ajax will request, with the out we can “return” the value which will be received by ajax (this can be improved, better to send json than the whole page)

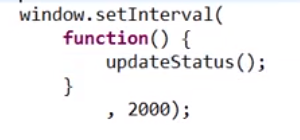


Ajax call which will be executed on click:



Window.setInterval

Allows to execute a function on the interval (ms)



Receiving json instead of a web page:





Comet/Reverse Ajax

Communication between service and clients, the server and client keep a http conexion for a long period which can be configured

Streaming

Server can send info to client whenever they want, permanent conexion

Long polling

Client send request (like ajax) client keep conexion open until the server respond

Asynchronous Servlets

We have a thread pool with http worker thread, when there is a request one of this threads is assigned to it, then it assign it to the background worker thread to continue its processing and the worker thread is freed .

Good on known limited number of cliets

Asynchronous servlet, connection always open

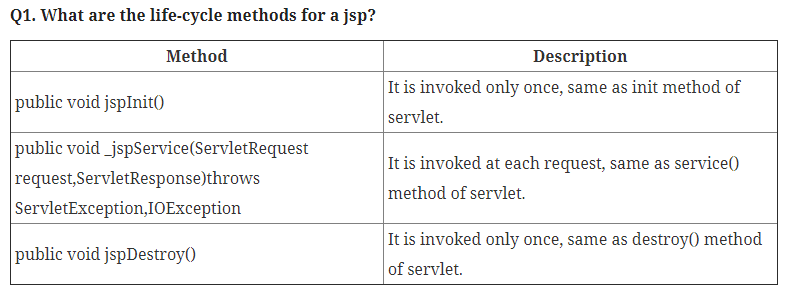


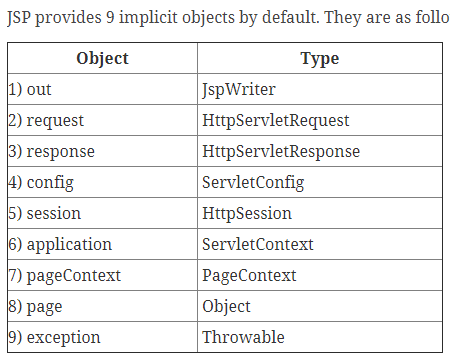
Websockets

Provide a direct connection between client and server using tcp.

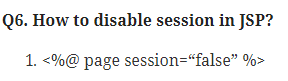
It reduces the size of the transmitted information making it more efficient

JavaServer Faces (JSF) is a user interface (UI) designing framework for Java web applications.



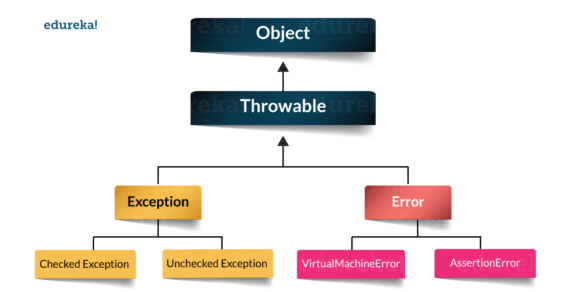












Java Servlet is server side technologies to extend the capability of web servers by providing support for dynamic response

javax.servlet.ServletConfig is used to pass configuration information to Servlet.

