

J2EE structure

client: web browser	
server: JSP/JServletlet	Tomcat
Backend: database	
Eclipse	Server config
MVC: Struct, Spring, JSF	

JSP fundamentals

jsp expression	<%= java statement %>
jsp scriptlet	<% unnonymous java blocks %>
jsp import	<%@ page import= comma separated java import %>
jsp declaration	<%! named java blocks, ie methods %>
call java class from jsp: avoid large chunk of java code (scriptlets or declarations) in jsp	
<jsp:include file="header.jsp" />	

jsp built-in objects

request	http request header + form data
response	Http support for send reponse
out	jspWriter: include content in html
session	unique session for each user across different page
application	shared among users of web app

Read html form in JSP

```
<form id="test" action
="student-response.jsp"
method='post">
    name: <input type="text"
name="name" />
    <input type="radio"
name="isShortForm"
value=" Yes "> Yes
    <input type="radio"
name="isShortForm"
value=" No"> No
```

Read html form in JSP (cont)

```
> <input type="checkbox" name="language"
value="Javascript"> Javascript
<input type="checkbox" name="language"
value="Java"> Java
<input type="checkbox" name="language"
value="SQL"> SQL
<input type="checkbox" name="language"
value="python"> Python
<input type="submit" value="Submit"/>
</form>
<select name="country" form="test">
    <option value="China"> China </option>
    <option value="US"> US </option>
    <option value="Other"> Other </option>
</select>
in JSP
${param.name} ${param.country}
<%=request.getParameter("country")%>
<ul>
<%
    String [] language=request.getParameterValues("language");
    if (language !=null)
        for (String s:language)
            out.println("<li>"+s+"</li>");
%>
</ul>
```

Session -Cookies

```
Cookie api : javax.servlet.http;
imported for all jsp implicitly
only send to specific server
domain matched
<!-- -send cookies to web browser
-->
<%
```

Session -Cookies (cont)

```
> String favLang=request.getParameter("F-
avoriteLanguage");
Cookie theCookie=new Cookie ("myApp.f-
avoriteLang",favLang);
theCookie.setMaxAge(606024*365);
//default 30min
response.addCookie(theCookie);
%>
cookies and session:
A cookie is a bit of data stored by the
browser and sent to the server with every
request. Cookies with no expiration time will
be deleted after browser closed ( some
browser).facilitate the session.
A session is a collection of data stored on
the server and associated with a given user
(usually via a cookie containing an id code);
session deleted at reconnection to server,
give http a state.
<!--read cookies in web browser-->
<%
Cookie[] theCookies =request.getCookies();
if (theCookies !=null) {
    for (Cookie tempCookie:theCookies){
        if ("myApp.favoriteLang".equals(temp-
Cookie.getName())){
            favLang=tempCookie.getValue();
            break;
        }
    }
}
%>
```

Session (copy)

unique for a user, share data across pages, like a shopping cart for a user

Kept in memory, each user has a session id

session id (create, transfer) is handled by browser automatically

```
session.setAttribute("name", value)
setAttribute(String, Object)
```

```
(List<String> ) session.getAttribute("name")
Object getAttribute(String)
```

```
session.isNew()
session.getId()
```

```
session.invalidate()
session.setMaxInactiveInterval(ms)
```

1. create form 2. check set session attribute 3. read from session

to disable session: `%@ page session="false" %>`

PageContext

Cookie API in `javax.servlet.http`, imported implicitly for all JSP pages

```
Cookie(String name, String value);
```

```
Cookie theCookie = new Cookie("myApp.favoriteLang", favLang)
```

```
theCookie.setMaxAge(60*24*365);
//default 30min
```

```
response.addCookie(theCookie);
```

```
Cookie[] theCookies = request.getCookies();
```

```
if (theCookies != null) {
    for (Cookie tempCookie:
```

JDBC

install mysql, workbench + shell + demo data setup
 setup tomcat 7.0: download tomcat 7, unzip to c:\
 setup connection pool: multiple users (amazon) using the same db, need more connections, pool together to save resources (not setup connection every time, preconfigured connect in the pool)

JDBC (cont)

> download jdbc mysql connector jar ->

WebContent/WEB-INF/lib

define connection pool: WebContent/META-INF/context.xml

```
<Context>
```

```
<Resource name="jdbc/web_student_tracker"
```

```
auth="Container" type="javax.sql.DataSource"
```

```
maxActive="20" maxIdle="5" maxWait="10000"
```

```
username="webstudent" password="webstudent"
```

```
driverClassName="com.mysql.jdbc.Driver"
```

```
url="jdbc:mysql://localhost:3306/web_student_tracker?useSSL=false"/>
```

```
</Context>
```

resource injection: tomcat automatically set the connection pool/ datasource on your servlet.

```
--connector/J 8
```

```
alter user 'webstudent'@'localhost' identified with mysql_native_password by 'webstudent';
```

cache busting?

<https://curtistimson.co.uk/post/front-end-dev/what-is-cache-busting/>