**Include:**

* RequestDispatcher is an interface which has following 2 abstract Methods

1. public abstract void forward(ServletRequest req, ServletResponse resp) throws ServletException, IOException
2. public abstract void include(ServletRequest req, servletResponse resp) throws ServletException , IOException

* An object of RequestDispatcher helps us to perform either Forward or Include Operation
* “forward() method” helps us to Forward the Request from One Servlet to any other Internal Resource(Static / Dynamic)
* Include() method “helps us to Include the Response of an another Internal Resource (Static/ Dynamic) into Servlet
* When we Include the content of one servlet into an another, it will include the response of “corresponding overridden version of doXXX() method “ in that servlet
* If corresponding overridden version of doXXX() methods does not exists, then Container “Does Not give any Complication Error/ Runtime Exception” In this case it just ignore the include statement.

Example:

RequestDispatcher dispatcher = null;

Out.print(“1111111111”);

///Internal Resource –Static

Dispatcher = req.getRequestDispatcher(“index.html”);

Dispatcher.include(req, resp);

Out.print(“2222222”);

//Internal Resource – Dynamic

String url = “currentDate?fname=Praveen&lname=D”;

Dispatcher = req.getRequestDispatcher(url);

Dispatcher.include(req, resp);

Out.print(“3333333”);

NOTE:

* We cannot include the response of “External Resource” into the Servlet
* Unlike Redirect & Forward , we can include more than one resource/URL at a time.
* Include helps us to reuse the internal Resource with that its “easy to maintain the Web Application”

Assignment 8:

* Create a HTML Form as shown below
* Advancesearch.html

Create a Servlet

* Which gets the request from this form,
* Takes the “keyword”
* Display the results for that keyword in ther correcponfing Website.

**Assignment : 9**

* Create a “Header Page “ as shown below

Header.html

* Provide the “href” for
* Creaate Profile
* Search
* Change Password
* Don’t Provide “href” for “Home” & “Logout”
* Create a “Footer Page” as shown below

Footer.html

* Include theseHeader.html & Footer.html in the responses of
* Login Page
* Create Profile
* Search
* Change Password
* Also, if Authentication failes durng Loogin, “Generate Login Page with Errot Message” as a Response as shown Below.

**Cookies**

* Cookies are little piece of information in the form of name-value string pair exchanged between browser & server
* Cookies are created by server and “Maintained by Browser” & hence Cookies are “Browser Dependent.
* Cookies are the one & Only way in Servlet API , if we want to store any information in “User Browser”.
* Browser before sending the request, it performs the below activities (automatically)

1. It takes “Domain Name” from request
2. Looks out for Cookies within it’s own memory
3. If present, it attaches cookies to Request
4. If NOT present it simply sends the Request to Server without Cookies

* Hence Cookies “optional” both in HTTP Request & Response
* There are two typed of cookies,

1. Non-Persistent Cookie
2. Persistent Cookie.
3. **Non-Persistance cookie (is default)**

* Non-Persistent Cookie lives as long as Browser is kept open. Once the Browser is closed cookie disappears
* Persistent Cookie:
* Persistent Cookie will be present in Browser even after Browser is Closed.

**public Cookie Constructor(String name, String value)**

public Cookie(String name, String value)

* It’s the ONE & ONLY constructor available in “javax.servlet.http.Cookie” Concrete Class
* It creates the Cookie Object with specified name & value
* Name can contain only “Alphanumeric Character” but should not contain comma, white space, semicolons and should not begin with “$”
* Cookies name “Cannot be changed “ after creation
* Value can be anything & it “can be changed” after creation.

1. **Void HttpServletResponse.addCookie(Coookie cookieObj)**

* This method add the specified Cookie to the Respnse
* This method can be called “multiple Time” to set more than one Cookie to the Response

1. **Cookie[] HttpServletRequest.getCookies()**

* This method returns
* Nan “Array containing all the Cookie Objects”

Or

* Return “NULL” if request does not have Cookies

1. **String Cookie.getName()**

* This method returns the Name of the Cookie

1. **String Cookie.getValue()**

* This method return the value of cookie

1. **Void Cookie.setMaxAge(int expiry)**

* Set’s the max age of Cookie in “Seconds”
* +ve value makes Cookie “Persistent”
* Any –ve Value makes Cookie “Non-Persistent
* 0 Value make Cookie to be “Deleted Immediately”

**Example:**

1. Create Coookie

//Non-Persistent Cookie

Cookie myNameCookie = new Cookie(“myName”,”Praveen D”);

//Persistent Cookie

Cookie myLocationCookie = new Cookie(“myLocation”,”Bangalore”);

//Time in Seconds

myLocationCookie.setMaxAge(7\*24\*60\*60);

//send the above Cookies in the Response

resp.addCookie (myNameCookie);

resp.addCookie (myLocationCookie);

out.print(“Cookies Created…”);

1. **Read Cookie**

//Get Cookies From Request

Cookie[] receivedCookies = req.getCookies();

If(receivedCookies==null){

out.print(“Cookies are NOT Present”);

}else{

out.print(“Cookies are present”);

for(Cookie rcvdCookie : receivedCookies){

//Print the Cookie Info (Name & Value) in Browser

out.print(“Cookie Name:” +rcvdCookie.getName());

out.print(“Cookie Value: “+rcvdCookie.getValue());

}

}

}

1. **Delete Cookie**

Cookie[] receivedCookies = req.getCookies();

If(recievedCookie == null){

Out.print(“Cookies are NOT present”);

}else{

Out.print(“Cookies are present”);

for(Cookie rcvdCookie : recevedCookies){

String Name = rcvdCookie.getName();

//Delete ONLY ‘myLocation’ Cookie

If(name.equals(“myLocation”)){

rcvdCookie.setMaxAge(0);

resp.addCookie(rcvdCookie);

out.print(“Deleted ‘myLocation’ Cookie”);

**break;**

}

}

}