Servlet :

How ot works :

Client -> req -> server -> web container(DD web.xml (url) ) -> class -> service -> doPost() or doGet() -> resp -> client

1. request dispatcher

Servlet 1 : (dopost())

K = i + j;

// pass value to another servlet

req.setAttribute("key", k);

// request dispatcher

RequestDispatcher rd = req.getRequestDispatcher("sq");

rd.forward(req, resp);

Servel 2 : (doget())

// get value from servlet

**int** d = (**int**) req.getAttribute("key");

d = d \* d;

PrintWriter out = resp.getWriter();

out.println("Square of No : " + d);

2 Url Redirecting:

Servelet 1: (doGet())

K = i + j ;

// url resp sending redirect

resp.sendRedirect("sq?k="+k);

Servlet 2 : (doget())

// get redirect request

**int** u = Integer.*parseInt*(req.getParameter("k"));

u = u \*u;

PrintWriter out = resp.getWriter();

out.println("Square of No by redirect : " + u);

3. Session :

Servlet 1: ((doGet())

// send data to other servlet

HttpSession session = req.getSession();

session.setAttribute("add", k);

resp.sendRedirect("sq");

Servlet 2 : (doGet())

// get data from first servlet

HttpSession session = req.getSession();

**int** u = (**int**)session.getAttribute("add");

u = u \*u;

PrintWriter out = resp.getWriter();

out.println("Square of No by session : " + u);

4. cookie

Servlet 1 : (doGet())

// send data through cookie

Cookie cookie = **new** Cookie("add",k+"");

resp.addCookie(cookie);

resp.sendRedirect("sq");

Servelt 2:

// get req through cookie

**int** u = 0;

Cookie cookies[] = req.getCookies();

**for** (Cookie c : cookies) {

**if**(c.getName().equals("add")) {

u = Integer.*parseInt*(c.getValue());

}

}

u = u \*u;

PrintWriter out = resp.getWriter();

out.println("Square of No by cookie : " + u);

ServletContext : if multiple server want same context/value

PrintWriter out = res.getWriter();

out.println("Hi ");

ServletContext ctx = getServletContext();

String str = ctx.getInitParameter("name");

out.println(str);

web.xml :

<context-param>

<param-name>name</param-name>

<param-value>Ajay Ingle</param-value>

</context-param>

ServeletConfig : if multiple servlet wants different context/value

ServletConfig cg = getServletConfig();

String str1 = cg.getInitParameter("name");

out.println(str1);

web.xml :

<servlet>

<servlet-name>xyz</servlet-name>

<servlet-class>com.ingle.MyServlet</servlet-class>

<init-param>

<param-name>name</param-name>

<param-value>Vijay Ingale</param-value>

</init-param>

</servlet>

ServelAnnotaion :

@WebServlet(“/cal”)

Servelt create method :

Servlet 🡨 Generic 🡨 HttpServlet

Using Servlet -> interface -> 5 methods write

Using GenericServlet -> abstract -> only 1 method service and rest is inbuild

* It extends abstract method from Servlet

Using HttpServlet -> it extends both genricservlet -> use for doget and dopost

JSP : Java Server Pages

Extend version of servlet

Jsp -> no need to create class like servlet it automatically convert into servlet.

Structure of jsp page:

1. directive -> used to import packages in servlet

<%@page import=”java.util.\* , ... ”; %>

🡪 taglib , page , include

2. declaration -> used for outside of service method -> <%! int k = 1; %>

3. scriptlet -> write inside service method - > <% … %>

4. Expression -> write out for service -> <%= k %>

JSP implicit objects:

request (HttpServelRequest),

response (HttpServletResponse) ,

pageContext (PageContext) ,

out (JSPWriter,PrintWriter),

session (HttpSession),

application (ServerletContext),

config (ServletConfig).

POJO 🡪 Plain old java object -> create simple one object for class

DAO 🡪 Data Access object

MVC Architecture :

Client 🡪 Controller(Servlet) 🡪 Model(POJO) 🡪 View(JSP)

|

``

Service class 🡪 DAO 🡪JDBC

* JSTL - > jsp standard tag library
* JSTl import in lib folder -> api and implementation
* Jstl tag

Flitering :

Init() , doFliter(), destroy()