Generating request to servlet component by typing URL in address bar of browser is complex process. So It is better to provide the GUI ‘s for end user like “Hyper links, form submit’s, other GUI elements to generate requests. Here we use “html” for generating such GUI components.

**3 Approaches to Html to Servlet communication:**

1. Hyper links ( Here request does not carry data to servlet)

Usecase: show trending jobs, show share prices , show live score … etc.

1. Form Submit( Request carry some data to servlet)

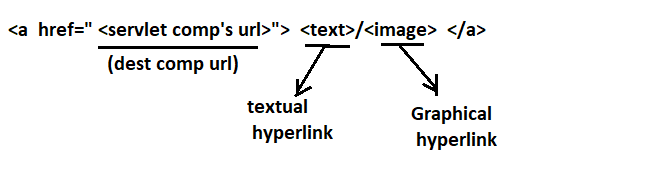
Usecases: Form login page, user registration page …etc.

1. Java Script Event:- ( when event raised on web page, some request goes to servlet )

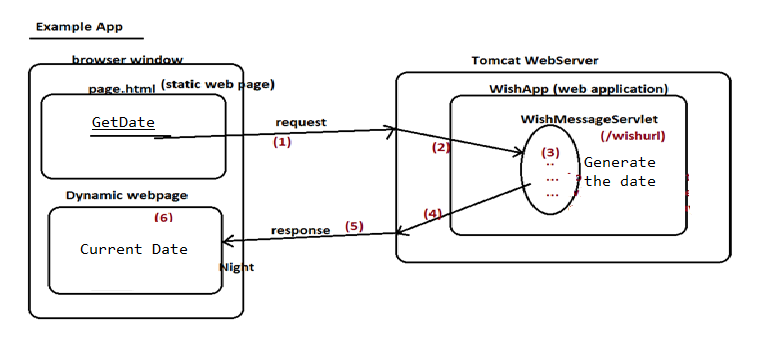
Usecases: (Loading countries Names, emp names in box, when web page is loaded).

**1.HTML to Servlet Communication Using HyperLinks:-**

For this we need to place the servlet component url as href url of <a> tag in .hrml page.



By default, the hyperlink generated request does not carry any data. If needed, we pass data along with request by appending “query string” to url placed in href attribute of <a> tag.

****

**Step1:- create a Dynamic web project in eclipse –IDE.**

**ProjectName:- wishMsgGenerator.**

**File 🡪 new 🡪 WishMsgGenerator**

**Stpe2:- create page.html(static web component ) in src/main/webapp folder**

<!DOCTYPE html>

<html>

<head>

<meta charset=*"ISO-8859-1"*>

<title>Insert title here</title>

</head>

<body>

<h2 align=*'center'*><u>Html To Servlet Communication Using HyperLink</u></h2>

<a href=*'http://locahost:7070/WishMsgGenerator/wishurl'*>Get Date</a>

</body>

</html>

Step3:- create package (com.nit.raos) in src/main/java

Create wish.msg (Dynmaic web component) in com.nit.raos package.

**package** com.nit.raos;

**import** java.io.IOException;

**import** java.io.PrintWriter;

**import** java.time.LocalDate;

**import** jakarta.servlet.ServletException;

**import** jakarta.servlet.annotation.WebServlet;

**import** jakarta.servlet.http.HttpServlet;

**import** jakarta.servlet.http.HttpServletRequest;

**import** jakarta.servlet.http.HttpServletResponse;

@WebServlet("/WishMsg")

**public** **class** WishMsg **extends** HttpServlet {

**public** WishMsg() {

// **TODO** Auto-generated constructor stub

}

**protected** **void** doGet(HttpServletRequest request, HttpServletResponse res) **throws** ServletException, IOException {

PrintWriter pw=res.getWriter();

res.setContentType("text/html");

LocalDate d=LocalDate.*now*();

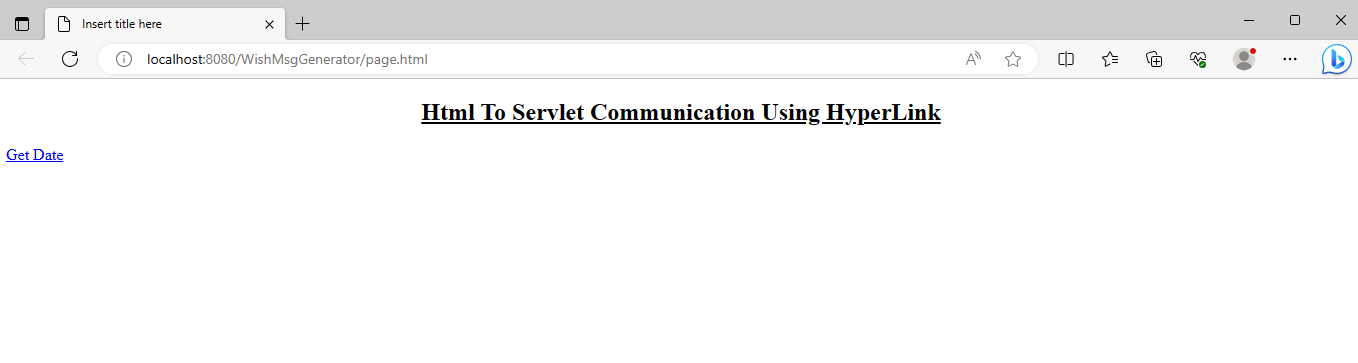
pw.println(d);

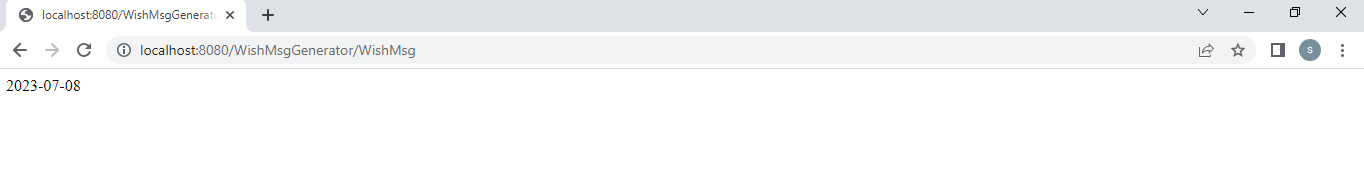
}

}

Stpe4: start web server

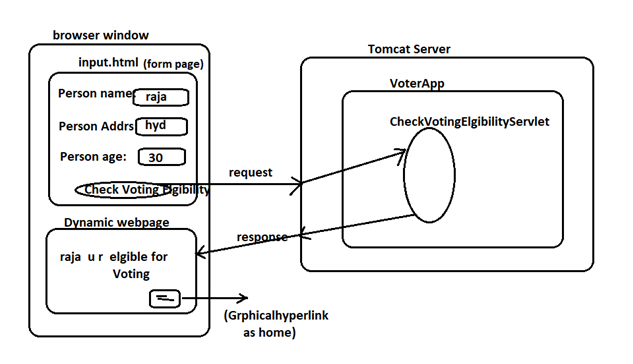
**Step5:- test the application**

****

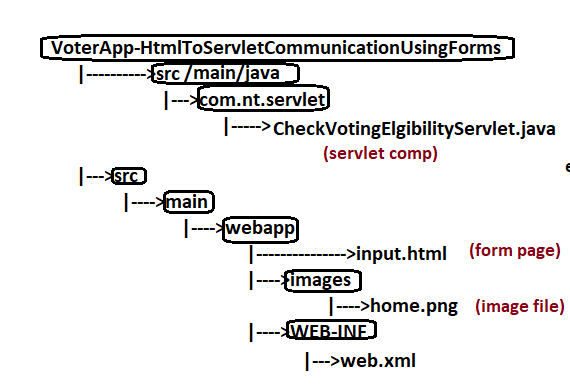


**2.HTML to Servlet Communication Using Form Submission:-**

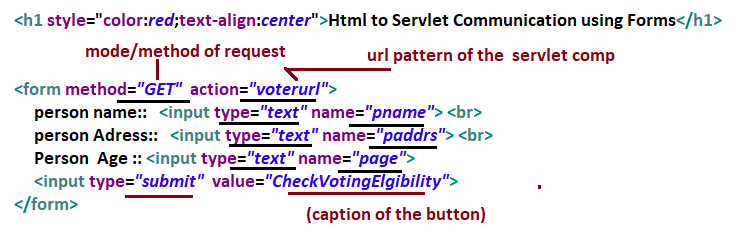
If end user want to send data along the request from webpage to servlet component then go for HTML to servlet communication using form submission.



In Eclipse Project Directory Structure:



1. **Input.html:-**

****

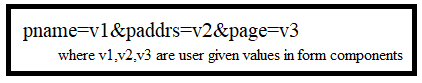
**Generally,** webpage send request to servlet component in one of two modes/methods.

1. **Get()**
2. Post()

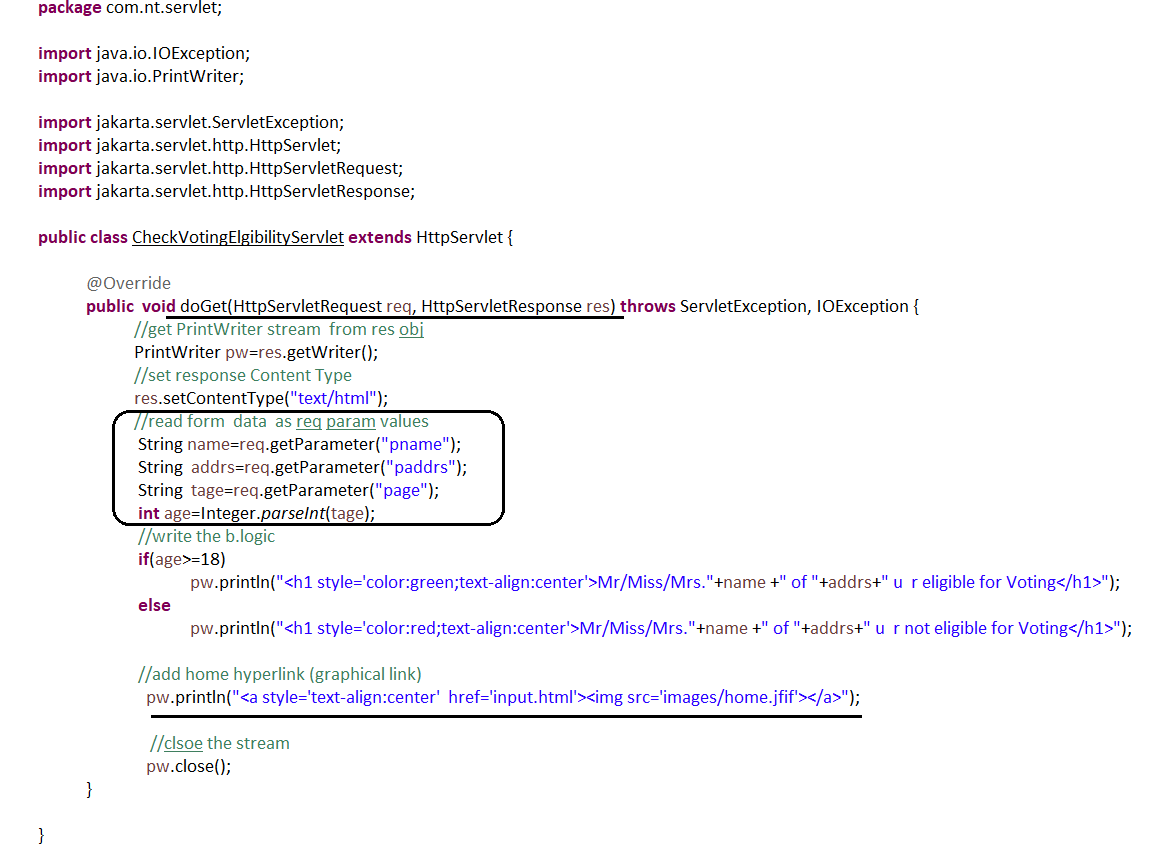
See front-end technologies notes for above two methods.

When submit button is pressed, form data will be formed as **query string** having form component as parameter and form component value as parameter value and the request from webpage with Query String will go to servlet component whose url is specified in the **action** attribute of <form> tag.

Sample query string as per above index.html code



1. **CheckVotingEligibilityServlet.java:-**



The form page gives only two modes(do,get) of request, programmer should use doGet(-,-) or doPost(-,-) method to process the request in servlet class.

* To process doGet(-,-) request of form page, use doGet(-,-) method.



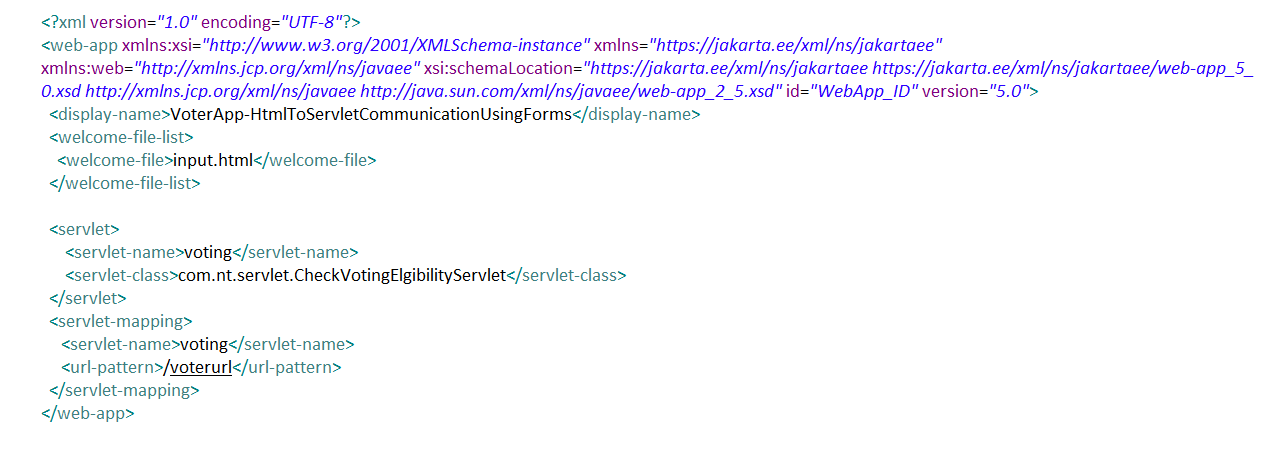
* To process doPost(-,-) request of form page, use doPost(-,-) method.



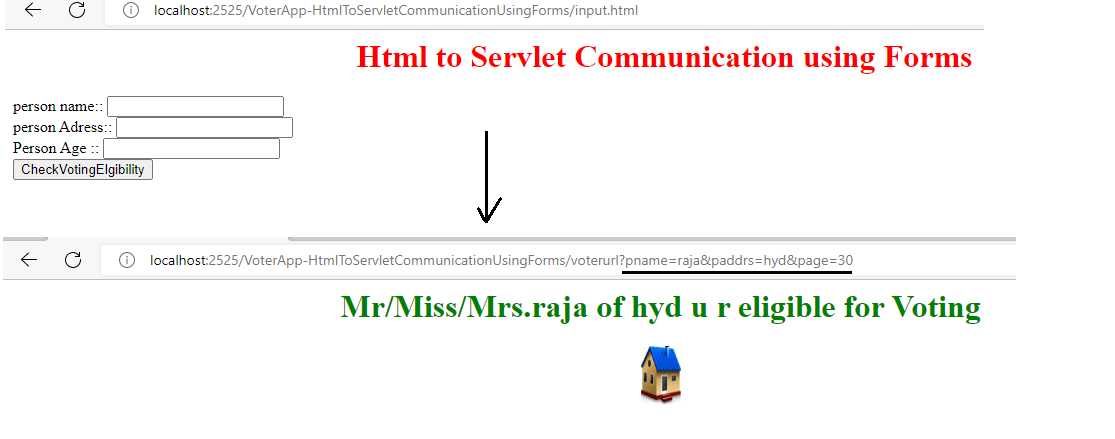
All Input values came from webpage are stored automatically in HttpServletRequest object. The Servlet component should use getParameter(-) method of HttpServletRequest class to get input values.

HttpServletRequestobject. getParameter( “html form element –name”);

1. **Web.xml :-**

****

1. **Test the application:-**

****

