Different ways of Creating a Servlet

1. Servlet(I) ==========> 5 abstract methods
2. GenericServlet(AC) ==> one abstract method (pvs(ServletRequest request, ServletResponse response)

Note:
When we build webapplications, internally httpprotocol is used and while sending the request , the request type can be
    a. POST
    b. GET

We can build Servlet in a easier way with the help of GenericServlet, then y need HttpServlet(AC)?
    Ans. In case of GenericServlet(AC), to process the request we have only one method(pvs(req,resp)) which is generic for
        any type of request like GET,POST,....
        Becoz there is only one method available which is generic for any type of request,Debugging the application becomes
        difficult.
        henceforth to deal with only Httpprotocol, we have a special approach to create a servlet called "HttpServlet".

Note:
```
C:\Users\nitin>javap javax.servlet.http.HttpServlet
Compiled from "HttpServlet.java"
public abstract class HttpServlet extends GenericServlet {

  protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException,IOException;
  protected void doPost(HttpServletRequest request,HttpServletResponse response) throws ServletException,IOException;

  protected void service(HttpServletRequest request, HttpServletResponse response) throws ServletException,IOException;
  public void service(ServletRequest request, ServletResponse response) throws ServletException,IOException;

}
```

Note:
request method type is GET ==========> doGet(request,response)
request method type is POST ==========> doPost(request,response)

Note:
  For a webapplication how to send GET request?
    a. type url in the address bar of browser and hit the request.
    b. by clicking the hyperlink in a webpage.
    c. using <form method="GET">
    d. default <form> method attribute is GET only.

  For a webapplication how to send POST request?
    a. using <form method="POST">

Life Cycle of HttpServlet

1. when we submit the form browser prepares HttpRequest and sends to server.

2. WebServer checks the request is for static/dynamic information.
3. If the request is for Static information then webserver provides required information(copy and paste)
     if available otherwise 404 Status Code(Saying the requested resource is not available).
4. If the request is for dynamic information, then webserver hands over the control to "catalina" container.
5. wecontainer identifies the request based on "web.xml" or through "annotation". (/test)
6. webcontainer will check whether the ServletObject(TestServlet) is available or not (/test====> TestServlet.class)
7. If the Servletobject(TestServlet.class) is not available, then it will perform the following action
            a. Loading ===> static block
            b. instantiation ===> constructor
            c. initialization ===> init()[same as GenericServlet lifecyle]
8.RequestProcessing phase
      a. webcontainer will create ServletRequest,ServletResponse object by invoking

            public void service(ServletRequest request, ServletResponse response) throws ServletException,IOException;
       container will check in our class service(ServletRequest,ServletResponse) is available or not, if it is avaialble it will
       execute our service() only and provides the response.
        if our servlet class does not contain service(), then container will execute HttpServlet
            public void service(ServletRequest request,ServletResponse response)

HttpServlet
=========
public void service(ServletRequest request,ServletResponse response)
{
             HttpServletRequest hreq = (HttpServletRequest)request;
             HttpServletResponse hresp = (HttpServletResponse)response;
            service(hreq,hresp);//protected void service(hreq,hresp)
}
webcontainer will call protected service(HttpServletRequest hreq,HttpServletResponse hresp) throws SE,IOE
if our class contains protected void service(HttpServletRequest hreq,HttpServletResponse hresp) throws SE,IOE
then container will call our class service() only.
if our class doesnot contains protected void service(HttpServletRequest hreq,HttpServletResponse hresp) throws SE,IOE
then container will HttpServlet class service().

HttpServlet
=========
protected void service(HttpServletRequest request, HttpServletResponse response)
throws ServletException,IOException
{
      String requestType = request.getMethod();
      if(requestType.equals("GET")){
                  doGet(request,response);//protected/public void doGet(request,response);
      }
      else if (requestType.equals("POST")){
                  doPost(request,response);//protected/public doPost(request,response);
      }

container will call the public service method
then that public service method internally calls the protected service method
then that protected service method will call the doGet or doPost method requestType
in our code if we override the doGet or doPost Method then our logic will be executed otherwise 501 error

```
      ;;;;;
      ;;;;
      else
            return 501 status code saying Http method not implemented
}
```
webcontainer will check whether our servlet class contains doXXXX(req,resp).
if it contains doXXXX(req,resp) it will be executed and it provides the response.
if our servlet class does not contain doXXXX(req,resp) then HttpServlet class doXXXX(req,resp) will be called.

HttpServlet
=========
```
protected void doXXXX(HttpServlet request,HttpServletResponse response ) throws SE,IOE
{
      return 405|400 status code saying HttpMethod GET is not supported by this URL.
}
```

Note:
Hierachy of calling the methods

a. public service(SRreq,SResp)
b. protected service(HSReq,HSResp)
c. public void doXXXX(HSReq,HSResp)

case1:
If our servlet class contains public void service(SReq,SResp) then for every type of request(POST,GET)
same method will be executed.

case2:
If our servlet class contains public void service(SReq,SResp) and protected void servcie(HSReq,HSResp)
then for every type of request(POST,GET) public void service(SReq,SResp) same method will be executed.

case3:
If our servlet class contains protected void servcie(HSReq,HSResp) and doGet() then for every type of
request(POST,GET) protected void service(HSReq,HSResp) same method will be executed.

case4:
we are sending GET request,but our servlet doesnot contain doGet(),it contains doPost() then which method would be
called?
it calls HttpServlet doGet() which would send 405 status code.

case5:
we are sending POSt request,but our servlet doesnot contain doPost(),it contains doGet() then which method would be
called?
it calls HttpServlet doPost() which would send 405 status code.

case6:
Assume we need to give common response for both the request type, then how to code?

eg:

```
import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.annotation.*;
import java.io.*;

@WebServlet(urlPatterns="/test")
public class TestServlet extends HttpServlet
{
      @Override
      public void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException,IOException
      {
            doProcess(request,response);
      }
      @Override
      public void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException,IOException
      {
            doProcess(request,response);
      }
      public void doProcess(HttpServletRequest request,HttpServletResponse
response)throws ServletException,IOException
      {
            System.out.println("Request method is of type ::
"+request.getMethod());
            String userName = request.getParameter("username");
            System.out.println("username is :: "+userName);
      }
}
```

Playing with request Object

```
=======================
To retrieve only one value from request object
   public abstract java.lang.String getParameter(java.lang.String);

To retrive multiple values from request object
      public abstract java.lang.String[] getParameterValues(java.lang.String);

To know the type of request from request object
        public abstract java.lang.String getMethod();


import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.annotation.*;
import java.io.*;

@WebServlet(urlPatterns="/reg")
public class TestServlet extends HttpServlet
{

      @Override
      public void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException,IOException
      {

            response.setContentType("text/html");

            System.out.println("Request type is :: "+request.getMethod());
```

```
            String username  = request.getParameter("username");
            String useremail = request.getParameter("useremail");
            String useraddr  = request.getParameter("useraddr");
            String[] courses = request.getParameterValues("course");

            PrintWriter out = response.getWriter();
            out.println("<html><head><title>OUTPUT</title></head>");
            out.println("<body>");
            out.println("<center>");
            out.println("<h1>Student Registration details</h1>");
            out.println("<table border='1'>");
            out.println("<tr><th>NAME</th><td>"+username+"</td></tr>");
            out.println("<tr><th>EMAIL</th><td>"+useremail+"</td></tr>");
            out.println("<tr><th>ADDR</th><td>"+useraddr+"</td></tr>");
            out.println("<tr><th>COURSE</th>");
            String data = "";
            for(String course: courses)
                  data +=course+" ";
            out.println("<td>"+data+"</td>");
            out.println("</tr>");
            out.println("</table>");
            out.println("</center>");
            out.println("</body>");
            out.println("</html>");

            out.close();
      }
}
```