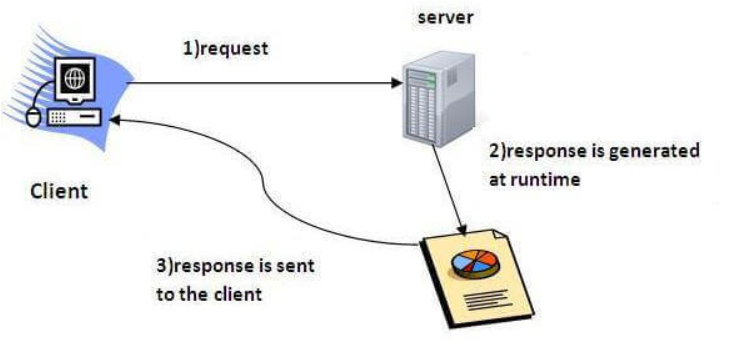
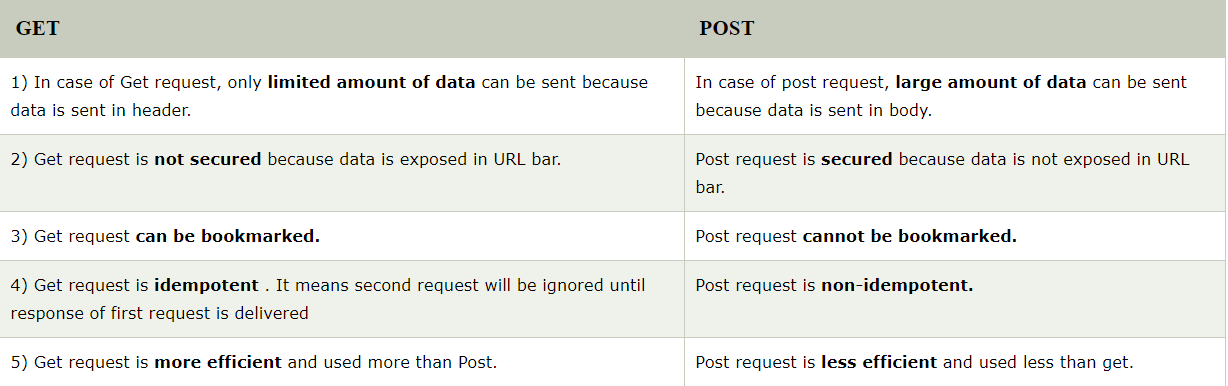
**Servlet**

* Servlet can be described in many ways depending on the context

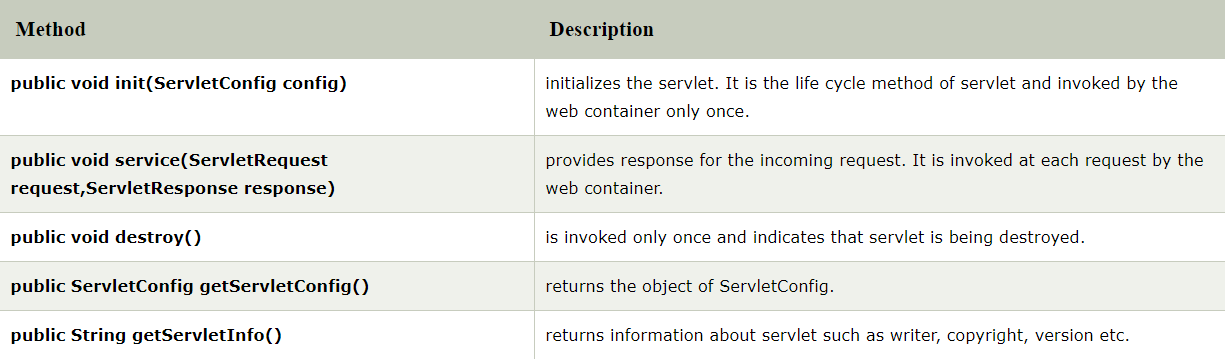
1. Servlet is a technology which is used to create a web application.
2. Servlet is an API that provides many interfaces and classes including documentation.
3. Servlet is an interface that must be implemented for creating any Servlet.
4. Servlet is used to extend the capabilities of the servers and respond to the incoming requests. It can respond to any requests.
5. Servlet is a web component that is deployed on the server to create a dynamic web page.



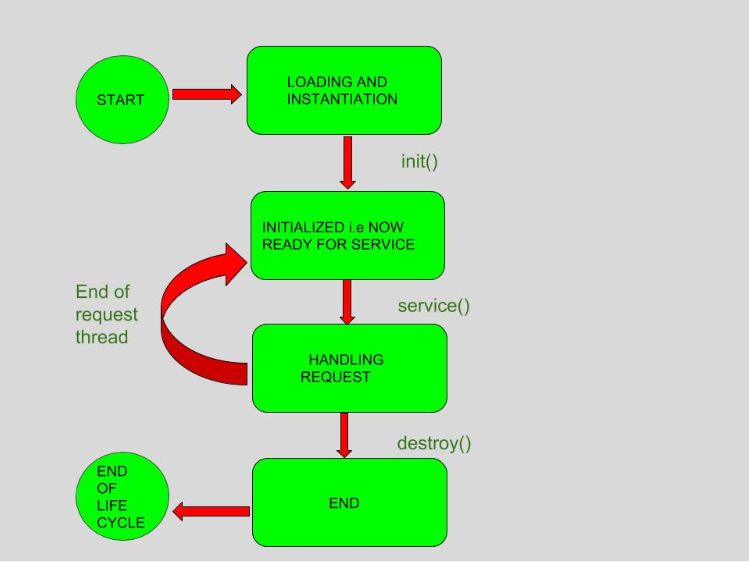
* Different HTTP request methods:

1. GET - Asks to get the resource at the requested URL. This only retrieves data.
2. Request has body- NO
3. Successful response has body- YES
4. Safe (does not alter the state of server)- YES
5. Idempotent (Identical request can be sent more than once with same effect) - YES
6. Allowed in HTML forms- YES
7. POST - The POST method is used to submit an entity to the specified resource, often causing a change in state or side effects on the server.
8. Request has body - YES
9. Successful response has body - YES
10. Safe - NO
11. Idempotent - NO
12. Allowed in HTML forms - YES
13. HEAD – Identical to GET just without any response body.
14. Request has body - NO
15. Successful response has body - NO
16. Safe - YES
17. Idempotent - YES
18. Allowed in HTML forms - NO
19. PUT - The PUT method replaces all current representations of the target resource with the request payload.
20. Request has body - YES
21. Successful response has body - NO
22. Safe - NO
23. Idempotent - YES
24. Allowed in HTML forms - NO
25. Delete - The DELETE method deletes the specified resource.
26. Request has body - MAY
27. Successful response has body - MAY
28. Safe - NO
29. Idempotent - YES
30. Allowed in HTML forms - NO
31. TRACE - The TRACE method performs a message loop-back test along the path to the target resource, providing a useful debugging mechanism.
32. Request has body - NO
33. Successful response has body - NO
34. Safe - NO
35. Idempotent - YES
36. Allowed in HTML forms - NO
37. OPTIONS - The OPTIONS method is used to describe the communication options for the target resource.
38. Request has body - NO
39. Successful response has body - YES
40. Safe - YES
41. Idempotent - YES
42. Allowed in HTML forms - NO

* Servlet Interface:

Servlet interface needs to be implemented for creating any servlet (either directly or indirectly). It provides 3 life cycle methods that are used to initialize the servlet, to service the requests, and to destroy the servlet and 2 non-life cycle methods.

* Life Cycle of a Servlet:



* GenericServlet Class:

1. GenericServlet implements Servlet, ServletConfig and Serializable interfaces. It provides the implementation of all the methods of these interfaces except the service method.
2. GenericServlet class can handle any type of request so it is protocol-independent.

* HttpServlet Class: The HttpServlet class extends the GenericServlet class and implements Serializable interface. It provides http specific methods such as doGet, doPost, doHead, doTrace etc.

Methods:

1. **public void service(ServletRequest req,ServletResponse res)** dispatches the request to the protected service method by converting the request and response object into http type.
2. **protected void service(HttpServletRequest req, HttpServletResponse res)** receives the request from the service method, and dispatches the request to the doXXX() method depending on the incoming http request type.
3. **protected void doGet(HttpServletRequest req, HttpServletResponse res)** handles the GET request. It is invoked by the web container.
4. **protected void doPost(HttpServletRequest req, HttpServletResponse res)** handles the POST request. It is invoked by the web container.
5. **protected void doHead(HttpServletRequest req, HttpServletResponse res)** handles the HEAD request. It is invoked by the web container.
6. **protected void doOptions(HttpServletRequest req, HttpServletResponse res)** handles the OPTIONS request. It is invoked by the web container.
7. **protected void doPut(HttpServletRequest req, HttpServletResponse res)** handles the PUT request. It is invoked by the web container.
8. **protected void doTrace(HttpServletRequest req, HttpServletResponse res)** handles the TRACE request. It is invoked by the web container.
9. **protected void doDelete(HttpServletRequest req, HttpServletResponse res)** handles the DELETE request. It is invoked by the web container.

* Servlet and request mapping are done in deployment descriptor file i.e. “web.xml” file.
* Calling a servlet from another servlet can be done by RequestDispatcher or sendRedirect.

1. The RequestDispatcher interface allows you to do a server side forward/include whereas sendRedirect() does a client side redirect.
2. SendRedirect() will search the content between the servers. it is slow because it has to intimate the browser by sending the URL of the content. then browser will create a new request for the content within the same server or in another one.
3. Sending data by requestDispatcher can be done by request.setAttribute(String, Object)
4. To send data by sendRedirect we can use “session” or “cookies”.

* Any information can also be shared by web.xml file, for this we have servletConfig and servletContext.
* ServletContext: Reads information stored in <context-param> element. Its value is same for all servlets.

1. getServletContext() method is used to return the object of servletContext.
2. getInitParameter(String name) gives the value of the parameter name specified.
3. getInitParameterNames() gives the names of stored context, its return type is Enumeration.

* ServletConfig: Reads information stored in <init-param> element. It has different value of the parameter for different servlets.

1. getServletConfig() method is used to return the object of ServletConfig.