

BIG-7 methods

=============

What is the difference b/w GET and POST request type?

GET

if we want to get some information from the server then we need to for GET request.

eg: fetching the train ticket details by supplying source and destination information.

usually get request are read only request and at the server-side update operation won't be performed.

In case of GET request, end user provided information will be attached to the url in the form of "QueryString"

eg: http://localhost:9999/App-01/test?source=bengaluru&destination=shivamoghaa

In case of GET request, end user data is available inside url as query string, it is less secured.

In case of GET request, since the data is attached in the url as querystring, only small volume of data can be sent.

Bookmarking and caching is supported in case of GET request.

POST

we use POST request to send huge amount of data to the server.

eg: uploading resume

usually post request are write only request and at the server-side update/insert operation would be performed.

In case of POST request, end user provided information will be not be attached to the url in the form of "QueryString"

eg: http://localhost:9999/App-01/test

In case of POST request, end user data is not available inside url as querystring, so it is more secured.

In case of POST request, since the data is not attached in the url as querystring, large volume of data can be sent.

Bookmarking and caching is not supported in case of POST request.

Note: What is idempotent request?

By repeating the request multiple times, if there is no change in the response then such type of request we call as "idempotent" request.

eg: GET request is idempotent request, but POST is not Idempotent.

What is safe request?

By repeating the request multiple times,if there is no side effect at the server side then such type of request we call as "safe" request.

Eg: GET request is safe request, but POST is not safe request.

How to send GET request?

1. Type the address in the url bar and hit enter key (request is sent)

2. clicking the hyperlink (<a href=" ">CLICK HERE</a>)

3. submit the form with method attribute as "GET".

<form method ="GET">

<!--- -- >

</form>

4. submit the form without method attribute (default is GET only)

<form>

<!--- -- >

</form>

How to send POST request?

1. submit the form with method attribute as "POST".

eg: <form method ="POST">

<!--- -- >

</form>

Note:

When we send the request, automatically the HTTP Protocol create the HTTP Request Object and sends it to HTTP protocol.

The relevant information will be assigned in the respective sections of HTTP Request object.

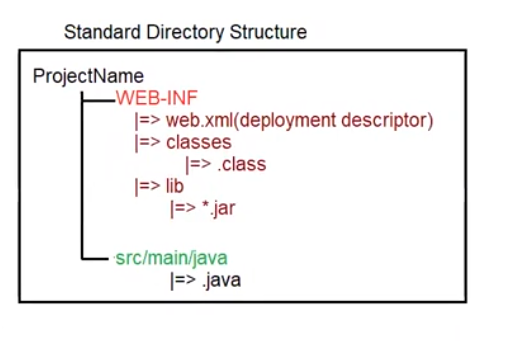
This HTTP Request object will be taken by HTTP Protocol and it will send it to the respective Server.

Upon sending the response, automatically the HTTP Protocol will create HTTP Response Object.

The relevant information will be assigned in the respective sections of HTTP Response Object.

This HTTP Response object will be taken by HTTP Protocol to the browser and browser uses this information for displaying the output.

To build web-application we need to follow standard directory structure given by Server vendor



How to create Servlet in Java?

To create Servlet in java there are 3 approaches

a. Servlet (I)

b. GenericServlet (AC)

c. HttpServlet (AC)

Servlet(I)

public interface javax.servlet.Servlet {

public abstract void init(javax.servlet.ServletConfig) throws javax.servlet.ServletException;

public abstract javax.servlet.ServletConfig getServletConfig();

public abstract void service (javax.servlet.ServletRequest, javax.servlet.ServletResponse) throws javax.servlet.ServletException,IOException;

public abstract java.lang.String getServletInfo();

public abstract void destroy();

}

Whenever we write Servlet, Automatically for the written servlet container will perform the following actions

1. Depending on the url pattern supplied by the user for a dynamic resource

a. Servlet Loading will happen=>static block will be executed.

b. Servlet Instantiation will happen=== >constructor will be called and object will be created.

c. Servlet Initialization will happen ====== >void init(ServletConfig config) throws ServletException

d. Serlvet Request Processing will happen == >void service(ServletRequest request,ServletResponse response)throws ServletException,IOException;

e. Servlet De-Instantiation will happen ==== >void destroy();

2. After creating a Servlet, for every servlet url-pattern matching should be provided and it should be informed to the container via XML, Annotation.



<web-app>

<servlet>

<servlet-name>DemoServlet</servlet-name>

<servlet-class>FirstServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>DemoServlet</servlet-mapping>

<url-pattern>/test</url-pattern>

</servlet-mapping>

</web-app>

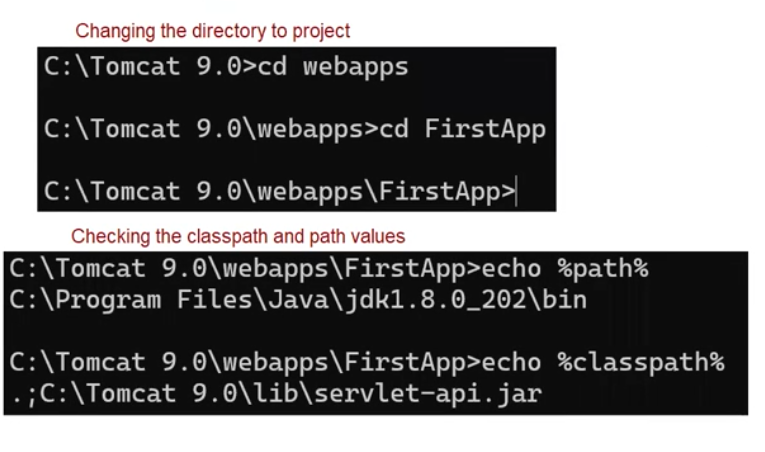
3. Before compilation set path and classpath environmental variable.

set path= C:\programfiles\jdk1.8\bin

set classpath =;.; C:\Tomcat 9.0\lib\servlet-api.jar

After compilation of FirstServlet.java copy the .class file to classes folder present under WEB-INF/classes.

Eg: Firstapp



Any Servlet will be executed by the container with its life cycle actions

a. Loading

b. Instantiation

c. Initialization

d. RequestProcessing phase

e. De-Instantiation

What is the meaning of webapps (deployment folder)?

Once we start the server, server would automatically go to webapp's folder and scans all the project present inside that folder and deploys the project in the execution area. (meaning ready for providing the service)

Since it scans for all the projects, we say webapp's folder as "Deployment folder".

Once the Tomcat engine loads all the project into execution area, It will create separate object for every project called "ServletContext" object.

Tomcat engine also scans web.xml file given by the user w.r.t every project.

It read the url-pattern for all the dynamic resource for future usage.

How to send the request to any application?

using the url pattern

eg: http://localhost:9999/[NameOfTheApplication/ContextRoot]/[url\_pattern of the resource]

Assume the request is sent for a particular url\_pattern, then what actions will be taken care by tomcat engine?

eg: <http://localhost:9999/SecondApp/demo>

Step1: Browser will send the request with the follwing url pattern

http://localhost:9999/SecondApp/demo

Step2: HttpProtocol will create a HttpRequest object depending upon the Request\_TYPE to carry the request data from client to server.

Step3: Once the Protocol hands over the HttpRequest Object to the tomcat server, server will pick up the RequestLine from the HttpRequestObject and take only ContextName/urlpattern for further processing.

Step4: Depending upon the dynamic resource identified by urlpattern, tomcat engine will hand over the control to container for execution.

Container will scan for web.xml file and identifies for a particular url\_pattern which servlet should be executed, based on that the servlet will be executed by the container.

Step5: As per the container life cycle actions, the requested url-pattern servlet will be executed.

1. Servlet Loading (.class file loading)

Class c = Class.forName ("SecondServlet");

2. Servlet Instantiation (for loaded class create an Object)

SecondServlet obj = (SecondServlet)c.newInstance();

3. Servlet Initalization (for the created object inject the required values)

obj.init (ServletConfig config);

4. Request Processing phase (for client request this method will be called)

obj.service (ServletRequest request, ServletResponse response);

Note:

If we use System.out.println() in the request processing logic, then output will be available on the console of the tomcat engine.

To get the inputs supplied by the user to the Servlet, we need to use "ServletRequest" Object.

To write the output from the application to the browser, we need to use "ServletResponse" Object.

Inside the ServletResponse object we have empty "PrintWriter" object, using which we need to write the Output to browser window.

If we use System.out.println() in the request processing logic, then output will be available on the console of the tomcat engine.

5. Once the server is stopped(undeployment)/for the same resource if the request wont' come for some period of time automatically container will execute "De-Instantiation" event.

obj.destory()

Before sending the request, we need to keep the compiled code in WEB-INF/classes folder

set path=C:\Program Files\Java\jdk1.8.0\_202\bin

set classpath=C:\Tomcat 9.0\lib\servlet-api.jar

Once the compiled code available then, we need to start the server

1. c:\tomcat9.0\bin\tomcat9.exe

Eg: SecondServlet

Output For First Request

=====================

SecondServlet .class file is loading ...

SecondServlet Object is instantiated ...

Servlet initialziation ...

Servlet Request Processing ...

Output for Second Request

==========================

Servlet Request Processing ...

As we noticed above, the processing time for second request is less when compared to first request becoz only requestprocessing logic is executed.

How to maintain the uniformity of response time for all the request?

Ans.To uniform response time for all the request we need to configure the tomcat engine.

Tomcat engine can be configured in 2 ways

a. XML <load-on-startup> 10 </load-on-startup> // any int value can be given

b. Annotation

To maintain the uniform response time we need to use <load-on-startup> tag.

<web-app>

<servlet>

<servlet-name>DemoServlet</servlet-name>

<servlet-class>SecondServlet</servlet-class>

<load-on-startup>10</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>DemoServlet</servlet-name>

<url-pattern>/demo</url-pattern>

</servlet-mapping>

</web-app>

Output for Firstrequest

======================

Servlet Request Processing ...

Output for Second Request

========================

Servlet Request Processing ...

Servlet Code w.r.t Annotation

============================

Use the following annotations on the top of Servlet class as shown below.

@WebServlet(urlPatterns="/test",loadOnStartup = 10)

Eg: ThirdApp

Limitation of implementing Servlet interface:

========================================

If we create a servlet using Servlet(I), then it is mandatory for us to give the implementation for all the methods of the interface whether it is required or not.

Becoz of this length of the code would increase and it decreases the readability.

To Overcome this problem we need to use "GenericServlet".

GenericServlet has already implemented Servlet Interface and it gives body for all the methods of Servlet interface except service().

if we use GenericServlet to create a Servlet, then we need to give body only for service() as a result of which the lines of code would be less, which increase the readablity of the application.

javap javax.servlet.GenericServlet

Compiled from "GenericServlet.java"

public abstract class javax.servlet.GenericServlet implements javax.servlet.Servlet,javax.servlet.ServletConfig,java.io.Serializable {

public javax.servlet.GenericServlet();

public void destroy();

public java.lang.String getInitParameter(java.lang.String);

public java.util.Enumeration<java.lang.String> getInitParameterNames();

public javax.servlet.ServletConfig getServletConfig();

public javax.servlet.ServletContext getServletContext();

public java.lang.String getServletInfo();

public void init(javax.servlet.ServletConfig) throws javax.servlet.ServletException;

public void init() throws javax.servlet.ServletException;

public void log(java.lang.String);

public void log(java.lang.String, java.lang.Throwable);

public abstract void service(javax.servlet.ServletRequest, javax.servlet.ServletResponse) throws javax.servlet.ServletException, java.io.IOException;

public java.lang.String getServletName();

}

Note:

GenericSevlet is an best example for "Adapter class design pattern".

init() is overloaded in GenericServlet.