**Servlet**

**Q-1: What is Servlet and What are the uses of Servlet?**

**A:**

1)Servlet is a technology i.e. used to create web application.

(2)Servlet is a web technology which is used to server side web componenet.

(3)server side web componenet performs the following

- Receives the incoming request data

- Process the requset

- Prepare the HTML Response.

(4)Servlet is a web component that is used to develop dynamic web Application

(5)Servlet is a server side programming language which is used for generating dynamic web pages. It generates web-page as a response of the request received from client(browser).

**Q-2: What is Web Server and what is the function of Server and what is the diff b/w “Web-Server” and “Application- Server”**

**A:** Web Server is an application which perform the following:

(1)Receives the Http Requset from Web Client

(2)processes that Requset with the help of Web Container

(3)send the Response to Web Client

ex-> Apache Tomcat Server ->java

A web server responsibility is to handler HTTP requests from client browsers and respond with HTML response. A web server understands HTTP language and runs on HTTP protocol.

Application Servers provide additional features such as Enterprise JavaBeans support, JMS Messaging support, Transaction Management etc. So we can say that Application server is a web server with additional functionalities to help developers with enterprise applications.

**Q-3: What is servlet container and What are the functions of Servlet container?**

**A:** The servlet container is a part of a Web server or application server

The main functions of Servlet container are:

a.) Lifecycle management - Managing the lifecycle events of a servlet like class loading, instantiation, initialization, service, and making servlet instances eligible for garbage collection.

b.) Communication support : Handling the communication between servlet and Web server.

c.) Multithreading support : Automatically creating a new thread for every servlet request and finishing it when the Servlet service() method is over.

d.) Declarative security : Managing the security inside the XML deployment descriptor file.

**Q-4: How many objects of a servlet is created?**

**A:** Only one object at the time of first request by servlet or web container.

**Q-5: How Servlet instance created ?**

**A:** An instance of servlet is created when the servlet is loaded for the first time in the container. And container call the default constructor .Init() method is used to configure this servlet instance. This method is called only once in the life time of a servlet,

**Q-6: What is the diff b/w ServeltConfig and ServletContext and how be defined in xml file write the code also.**

**1-ServletConfig ->**

(1)ServletConfig is an interface available in javax.servlet package . This interface is implemented by the servlet container to pass configuration information to a servlet.

(2)every servlet will have its own ServletConfig object and can not be shared.

(3)when you want to use any data which is common for all the users but specific to a particular Servlet,the data can be specified as Config Parameter or init Parameter.

(4)with servlet2.x:specify the config parameter in web.xml

<servlet>

<servlet-name>helloServ</servlet-name>

<servlet-class>com.cts.servlets.HelloServlet</servlet-class>

**<init-param>**

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

<param-value>subhag@gmail.com<param-value>

**</init>**

........

</servlet>

(5)As a Developer,you can collect that data from config object as follows.

String em = config.getInitParameter("email");

**2-ServletContext ->**

(1)ServletContext is an interface available in javax.sevlet package

(2)one web application will have only one ServletContext object i.e ServletContext object can be shared with all the Sevlets running in the Application.

(3)when you want to use any data which is common for all the users and common to all the Servlets then data can be specified as ContextParameters in the web.xml as follows

**<context-param>**

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

<param-value>www.yahoo.com</param-value>

**</context-param>**

(4)As a developer,you can collect that data from context object as follows:

String web = context.getInItParameter("website");

**Q-7: How many method in Servlet interface.**

**A:** There are 5 methods in Servlet interface. The init, service and destroy are the life cycle methods of servlet. These are invoked by the web container

**Method Description**

**public void init(ServletConfig config)->** initializes the servlet. It is the life cycle method of servlet and invoked by the web container only once.

**public void service(ServletRequest req,ServletResponse res)->**  provides response for the incoming request. It is invoked at each request by the web container.

**public void destroy() - >**is invoked only once and indicates that servlet is being destroyed.

**public ServletConfig getServletConfig()->** returns the object of ServletConfig.

**public String getServletInfo() ->** returns information about servlet such as writer, copyright, version etc.

**Q-8: What is Servlet Life Cycle .**

**A**:Servlet is loaded

servlet is instantiated

servlet is initialized

service the request

servlet is destroyed

**Q-9: What are the Life Cycle method.**

**A:**

**1-public void init(ServletConfig config:->** It is invoked only once when first request comes for the servlet. It is used to initialize the servlet

**2-public void service(ServletRequest request,ServletResponseQ:throws ServletException,IOException**->It is invoked at each request.The service(): method is used to service the request.

**3-public void destroy():-**>It is invoked only once when servlet is unloaded.

**Q-10: How many init() method in Generic Servlet**

**A:** There are two init()method in Generic Servlet**.**

**public void init(ServletConfig config) ->**is used to initialize the servlet

**public void init() ->**it is a convenient method for the servlet programmers, now there is no need to call super.init(config)

**Q-11: How many method in HttpServlet.**

**A:**

**protected void service(HttpServletRequest req, HttpServletResponse resp)-**Receives standard HTTP requests from the public service method and dispatches them to the doXXX methods defined in this class.

**public void service(ServletRequest req, ServletResponse res)->**Dispatches client requests to the protected service method.

protected void **doGet**(HttpServletRequest req, HttpServletResponse res)

protected void **doPost**(HttpServletRequest req, HttpServletResponse res)

protected void **doHead**(HttpServletRequest req, HttpServletResponse res)

protected void **doOptions**(HttpServletRequest req, HttpServletResponse res)

protected void **doPut**(HttpServletRequest req, HttpServletResponse res)

protected void **doTrace**(HttpServletRequest req, HttpServletResponse res)

protected void **doDelete**(HttpServletRequest req, HttpServletResponse res)

**10-**protected long **getLastModified**(HttpServletRequest req)

**HTTP Request -> Description**

**GET -**  Asks to get the resource at the requested URL.

**POST -** Asks the server to accept the body info attached.

It is like GET request with extra info sent with the request.

**HEAD -** Asks for only the header part of whatever a GET would return.

Just like GET but with no body.

**TRACE -** Asks for the loopback of the request message, for testing or

troubleshooting.

**PUT -** Says to put the enclosed info (the body) at the requested URL.

**DELETE -** Says to delete the resource at the requested URL.

**OPTIONS-** Asks for a list of the HTTP methods to which the thing at the request

URL can respond

**Q-12: Who is responsible to create the object of servlet**

**A:** The web container or servlet container.

**Q-13: When servlet object is created?**

**A:** At the time of first request.

**Q-14: What is the default HTTP method in the servlet?**

**A**:Default method is GET method for HTTPservlet.

**Q-15: What is difference between Get and Post method? And Where you will use Get and Where Post**

**A:** 1) In case of Get request, only limited amount of data can be sent because data is sent in header.

In case of post request, large amount of data can be sent because data is sent in body.

2)when you sent the request with Http method Get,the data will be attached to url as Query String

when you send the request with Http method Post,the data will be placed in the Request body.

3) Get request is not secured because data is exposed in URL bar.

Post request is secured because data is not exposed in URL bar.

(5)doGet() is faster if we set the response content length since the same connection is used. Thus increasing the performance

doPost() is generally used to update or post some information to the server.doPost is slower compared to doGet since doPost does not write the content length

**Note:** Always prefer to use GET (As because GET is faster than POST), except mentioned in the following reason:

1->If data is sensitive.

2->Data is greater than 1024 characters.

3->If your application don't need bookmarks.

**Q-16: What is RequestDispatcher Interface.How many method are their.**

**A:** (1)it is an interface available in javax.servlet package.

(2) The RequestDispacher interface provides the facility of dispatching the request to another resource it may be html, servlet or jsp.

(3)it has two methods as follows:

public void forward(ServletRequest req,ServletResponse res)

public void include(ServletRequest req,ServletResponse res)

**1->forward() method()->**

(1)forward() method is used to forward the request form:

a- Servlet to HTML

b- Servlet to JSP

c- Servlet to Servlet

d- JSP to HTML

e- JSP to JSP

f- JSP to Servlet

(2)you can place many forward() methods in one servlet conditionally to forward the control to other resorce but only one forward() will be executed.

Usage:

RequestDispatcher rd=null;

rd=request.getRequestDispatcher("test.html");

rd=request.getRequestDispatcher("test.jsp");

rd=request.getRequestDispatcher("test.kclink");

rd.forward(request,response);

(3)when you try to invoke two froward method then java.lang.IllegalStateException: can not forward after response has been committd.

(4)we can restrict the user from accessing the page directly by placing the page under WEB-INF directory

(5)when we have other statement after forward method then those statement will be executed

but we will not be able to write any content in the response stream.

**2-include() method->**

(1)include() method is used to include the response of servlet or jsp in another servlet or jsp.

(2)you can place many include() methods in one servlet to include many resources and all include()

**Q-17: What is the difference between forward() and include()?**

**A:** The RequestDispatcher include() method inserts the contents of the specified resource directly in the flow of the servlet response, as if it were part of the calling servlet.

The RequestDispatcher forward() method is used to show a different resource in place of the servlet that was originally called.

**Q-18: What is the Difference between forward() and sendRedirect()?**

**A:**

1) In forward() the same request is forwarded to the another resource. In sendRedirect() new request is send to the redirected resource.

2) forward() is taken care by the Servlet container while sendRedirect() is handled by the browser.

3) In forward() the URL(uniform resource locator) remains same on web browser. In sendRedirect() the URL changes in the web browser address bar.

4) forward() is faster compared to sendRedirect().

**Q-19: How many Servlet Scope is there and what is the diff b/w Request Scope and Session Scope.**

**A:** threre are 3 scopes available in servlet

(1) Request Scope

(2) Session Scope

(3) Context or Application Scope

**Request Scope->**

(1)when the data will be stored in HttpServletRequest object and the scope will be request scope

(2)the data from request scope can be accessed by that single user in that request only before sending the response to the client.

**Session Scope->**

(1)when the data will be stored in the HttpSession object then the scope will be session scope.

(2)the data from session scope can be accessed by single user across multiple request.

**Context or Application Scope->**

(1)when the data will be stored in ServletContext object then the scope will be context scope.

(2)the data from context or application scope can be accessed by multiple users across multiple requests.

**Q-20:What is the use of attribute in servlets?**

**A**:Attribute is a map object that can be used to set, get or remove in request, session or application scope. It is mainly used to share information between one servlet to another.

**Q-21: What is Session Tracking?**

**A:** Session simply means a particular interval of time.

Session Tracking is a way to maintain state of an user.Http protocol is a stateless protocol.Each time user requests to the server, server treats the request as the new request.So we need to maintain the state of an user to recognize to particular user**.**

Q-22: What is the need of session tracking in web application?

A: To identify the particular user

**Q-23: Why we use session Tracking.**

**A:** (1)HTTP protocol and web server are stateless i.e for web server every request is a new request to process and they can not identify whether it is coming from the same client or new client

(2)sometimes in web application,we need to identify the client and process the request as per the client.

ex->

in a shopping cart application,we should know

-> who is sending the request to add an item

-> in which cart the item has to be added

-> who is sending place order request so that it can charge the amount to correct client.

(3)"Session" simply means a particular interval of time.

(4)"Session" is a period of time where user sends multiple requests and receive multiple responces.

(5)in multiple requests user may send some data that should be stored and managed in the server.

(6)Session will be used to stored client specific data.

(7)with in the user session,you have to do 2 things.

-> identify the client

-> manage the client specific data.

(8)to identify the client as new or old,container uses Session id.

(9)to store clients state or data,you have to use HttpSession object.

**Q-24: What are the different ways of session tracking.**

**A:** There are four techniques used in Session tracking:

(1)HttpSession

(2)Cookies

(3)Url-Rewriting

(4)Hidden Fields

**Q-25: Explain the following things 1-HttpSession 2-Cookie 3-URL rewriting**

**A: 1-HttpSession->**

(1)HttpSession is an interface available in javax.servlet.http package.

(2)you can get the HttpSession object with the following methods of HttpServletRequest

-> HttpSession getSession()

-> HttpSession getSession(boolean)

-> String getId()

(3)you can store and access the client specific data in HttpSession object as an attribute

with the following methods:

-> void setAttribute(String aname,Object val)

-> Object getValue(String)

-> void removeAttribute(String aname)

-> Enumeration getAttributeNames()

**Cookies->**

(1)Cookie is a class available in javax.servlet.http package.

(2)Cookie is a simple information with name and value.

(3)name and value of the cookie will be of String type.

(4)Normally Cookies will be created at server m/c and will be stored at client m/c.

(5)Cookies created at server m/c will come to client m/c along with HttpResponse.

(6)Cookies stored at client m/c will go to server m/c along with HttpRequest.

**Creating Cookie->**

Cookie ck = new Cookie("email","gupta.subhag@gmail.com");

**Adding Cookie to Response->**

Cookie ck[]=requset.getCookies();

for(Cookie c:ck)

{

String cn = c.getName();

String cv = c.getValue();

System.out.println(cn+" : "+cv);

}

**Web Contanier Tasks->**

by default,Web Container will do the following regarding session managemant:

(1) creates one special Cookie with

-> JSESSIONID as Cookie name

-> sessionid as Cookie value

Cookie ck = new Cookie("JSESSIONID",ses.getId());

(2)Adds that cookie to response.

-> response.addCookie(ck);

(3)Collects the special cookie and identifies the Session object based on Session Id collected

from special Cookie.

Cookie ck[]=requset.getCookies();

for(Cookie c:ck)

{

if(cn.equals("JSESSIONID"))

{

String sid=c.getValue();

HttpSession hs= (HttpSession)sessionMap.get(sid);

}

}

Assume that sessionMap is the Map object which contains key and value.

key will be SESSIONID

value will be SessionObject.

(4)use the following options to remove the Cookies stored in the client m/c:

->you need to specify the max age of Cookie as 0.

->you need to add the same cookie to the response.

Cookie c=..........;

c.setMaxAge(0);

res.addCookie(c);

String bnm=request.getParameter("bname");

Cookie ck[]=request.getCookies();

for(Cookie c:ck)

{

String cn=c.getName();

if(cn.equals(bnm))

{

c.setMaxAge(0);

response.addCookie(c);

}

}

**Q-26: How to create Login and Logout in servlet by using HttpSession and cookie**

**A:** **By HttpSession**

**Login ->** HttpSession session=request.getSession();

session.setAttribute("name",name);

**Logout->**  HttpSession session=request.getSession();

             session.invalidate();

**By Cookie**

**Login->**  Cookie ck=**new** Cookie("name",name);

             response.addCookie(ck);

**Logout ->**  Cookie ck=**new** Cookie("name","");

        ck.setMaxAge(0);

         response.addCookie(ck);

**Q-27: What is the diff b/w session.setAttribute() and request.setAttribute()**

**A:** The scope, session attribute live all the session and the request attribute

only in a request

**Q-28:** **What is the diff b/w cookie.setMaxAge(0) , cookie.setMaxAge(-1) ,**

**cookie.setMaxAge(3500)**

**A: A positive value** indicates that the cookie will expire after that many seconds have passed. Note that the value is the *maximum* age when the cookie will expire, not the cookie's current age.

**A negative value** means that the cookie is not stored persistently and will be deleted when the Web browser exits.

**A zero value** causes the cookie to be deleted

**Q-29:What is difference between Cookies and HttpSession?**

**A**:Cookie works at client side whereas HttpSession works at server side.

**Q-30: What is the disadvantage of cookies?**

**A**:It will not work if cookie is disabled from the browser.

**Q-31: How can be destroy session by using HttpSession.**

**A:** session.invalidate();

**Q-32: How can be remove Cookie from client m/c.**

**A:** cookie.setMaxAge(0)

**Q-33: What is the use of “JSESSIONID”**

**A:** **JSESSIONID** is a cookie in J2EE web application that is created by web container and send along with response to client. which is used in session tracking. Since HTTP is a stateless protocol, we need to use any session to remember state.

**Q-34: What is the diff b/w 1- req.getSession()**

**2- req.getSession(true)**

**3- req.getSession(false)**

**A:**

**1- req.getSession()->**checks whether session object is available for this user or not.

if session object is available then returns that object.

if session object is not available then creates the new Session object and returns that object.

**2-req.getSession(true)->** same as getSession()

**3- req.getSession(false)->** checks whether session object is available for this user or not.

if session object is available then returns that object.

if session object is not available then return null.

**Q-35: What is filter and what is the important functions of filters and how we create Filter.**

**A:**

(1)Filter is a web component like servlet.

(2)Web Container is responsible for managing completer lifecycle of Filter.

(3)Servlet is responsible for core Request Proc which is called as Processing.

(4)Before core Request Processing by Servlet i.e before calling the service() method,you may want to perform some tasks is called as PRE PROCESSING tasks.

(5)After Core Request Processing by Servlet i.e after completing the service() method,you may want to perform some tasks which is called as POST PROCESSING tasks.

(6)Following are various PRE PROCESSING tasks which you can perform on incoming request

before core request processing.

->Logging

->Security checks includes Authentication and Authorization

->Varifying Session Validity

etc..

(7)Following are various POST PROCESSING tasks which you can perform on incoming request

before core request processing.

->Data Compression

->Data Encription or Encoding

->URL Rewriting

etc..

(8)if you write the code for PRE PROCESSING tasks and POST PROCESSING tasks across all the

Servlets then code gets duplicated and gives the maintainance problem when you try to change that code.

(9)to avoid the code duplication problem and maintainance problem,you need to write the PRE PROCESSING tasks and POST PROCESSING tasks code in a Centralized place called **"Filter".**

**Steps to Develop the Filter with Servlet->**

(1)write your own filter class by implementing javax.servlet.Filter interface.

(2)your filter class has to override the following 3 lifecycle methods.

->public void init(FilterConfig fc)

->public void doFilter(ServletRequest req,ServletResponse res,FilterChain fc)

->public void destroy()

**(3)Configure the filter in web.xml as follows**

<filter>

<filter-name>DemoFilter</filter-name>

<filter-class>DemoFilter</filter-class>

<init-param>

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

<param-value>Banglore</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>DemoFilter</filter-name>

<url-pattern>/DemoFilter</url-pattern>

</filter-mapping>

**Q-36: How we achieve Filter-Chaining.**

**A:** chain.doFilter(req,res);

**Q-37: What is Listener. Why we use the Listener.**

**A:** if We want to send the Request from HTML then how Container identified ,Request is created . So Container create the “Listener” to capture the event.

**Listener Interface Event Class**

ServletContextListener ServletContextEvent

ServletRequstListener ServletRequstEvent

HttpSessionListener HttpSession

**………..etc**

**ServletContextListener ->** it handle

Creating ServletContext object

Destroying ServletContext object

**ServletRequstListener->** it handle

1- Creating HttpServletRequest object

2- Destroying HttpServletRequest object

**HttpSessionListener->**it handle

Creating HttpSession object

2- Destroying HttpSession object

**Steps to write the Listener with Servlet->**

Write your own Listener class by implementing the required Listener interface.

Override the Corresponding methods of the Listener interface which you are

implementing.

Class MyListener implements ServletContextListener

{

//implement the overridden methods

}

Register the listener by writing the following in web.xml

<listener>

<listener-class>com.jlc.MyListener</listener-class>

</listener>

**Q-38: Which event is fired at the time of project deployment & undeployment?**

**A**:ServletContextEvent.

**Q-39: Which event is fired at the time of session creation and destroy?**

**A**:HttpSessionEvent.

**Q-40:Which event is fired at the time of setting, getting or removing attribute from application scope?**

**A**:ServletContextAttributeEvent.

**Q-41: What is the use of welcome-file-list?**

**A**:It is used to specify the welcome file for the project. If we have multiple Html file in our project.and we want when we deployed our project then specific html page should be first page. By default index.html is the default html file

**<web-app>**

 ....

**<welcome-file-list>**

**<welcome-file>**home.html**</welcome-file>**

**<welcome-file>**default.html**</welcome-file>**

**</welcome-file-list>**

**</web-app>**

**Q-42: Why do we need constructor in servlet even though we have a init() method?**

**A**: init() method is used for initializing the servlet however constructor is required in order to instantiate the Servlet class. Servlet container instantiate the Servlet class.

**Q-43: Can servlet have a constructor?**

**A**: Yes

**Q-44: Can we use the constructor, instead of init(), to initialize servlet?**

**A**: Yes , of course you can use the constructor instead of init(). There's nothing to stop you. But you shouldn't. The original reason for init() was that ancient versions of Java couldn't dynamically invoke constructors with arguments, so there was no way to give the constructur a ServletConfig. That no longer applies, but servlet containers still will only call your no-arg constructor. So you won't have access to a ServletConfig or ServletContext.

**Q-45: Who is responsible for writing a constructor?**

**A:**Container is responsible for writing constructor without arguments in servlet.

**Q-46: When we don't write any constructor for the servlet, how does container create an instance of servlet?**

**A**: Container creates instance of servlet by calling Class.forName(className).newInstance().

**Q-47: Can we override init() and destroy() method in servlet.**

**A:** We can override the init() and destory() methods in servelets.  
The init() method called by the container only once ,when the servlet has been loaded into the memory. We can override init() method for connecting to the database or some other resources.We can override the destroy method to release the resources.

**Q-48:** **Can we override service() method in my servlet class..?? if**

**yes or no why ??**

**A :1-** **we can override service method in my servlet class** if the request is not protocol dependent.Bcoz we exdend HttpSrevlet class to handle Http GET and POST requests and we have to override some methods like doGet,doPost,doHead,doPut,doDelete.  
But if u want to override service method then u have to extend genricServlet Class.Then only u can implement Service() method

**A:2-**. **yes, we can override service() in our servlet, but not preferable.** if we want use these service from any protocol like ftp,telnet, then we have to override.

**Q-49: What happens if you call destroy() from init() in java servlet?**

**A:** Nothing strange! The destroy() method will simply be called/executed and the control will return back to the next instruction in the init() method.

in java servlet, destroy() is not supposed to be called by the programmer. But, if it is invoked, it gets executed. The implicit question is, will the servlet get destroyed? No, it will not. destroy() method is not supposed to and will not destroy a java servlet.

**Q-50: What happens if you call destroy() from service() in java servlet?**

**A:** destory() is part of servlet life cycle methods, it is used to kill the servlet instance. Servlet Engine is used to call destory(). In case, if you call destory method from service(), it just execute the code written in the destory(), but it wont kill the servlet instance. destroy() will be called before killing the servlet instance by servlet engine.

**Q-51:** [**Can we call service() method from destroy() method in Servlet?**](http://stackoverflow.com/questions/16745857/can-we-call-service-method-from-destroy-method-in-servlet)

A: : You can, but it doesn't make any sense. service() requires a request and a response parameter that are usually provided by the container when the Servlet is called. If you are calling service() by yourself, how are you gonna provide those parameters?

**Q-52: What is Generic Servlet class?**

A: Generic servlet is the super class of all servlets .GenericServlet class implements Servlet, ServletConfig andSerializable interfaces. It provides the implementaion of all the methods of these interfaces except the service method.GenericServlet class can handle any type of request so it is protocol-independent.You may create a generic servlet by inheriting the GenericServlet class and providing the implementation of the service method.

Q-53: Difference between GenericServlet and HTTPServlet?

A:

1) GenericServlet is an abstract class which implements Servlet interface while HTTPServlet abstract class extends the GenericServlet class. In short: GenericServlet class is a parent class for HTTPServlet.

2) GenericServlet does not support any protocol. HTTPSeervlet support HTTP and HTTPS protocol.

3) GenericServlet cannot handle cookies and session while HTTPServlet can handle them.

**Q-54: Why HttpServlet class is declared abstract? Even it does not contain any abstract method.**

**A:** The HttpServlet class is declared abstract because the default implementations of the main service methods do nothing and must be overridden. This is a convenience implementation of the Servlet interface, which means that developers do not need to implement all service methods.

**Q-55: What is deployment descriptor?**

A: web.xml file of a web application is known as deployment descriptor. It is usually placed inside WEB-INF folder of application. It has the information like Servlet name, Servlet mapping etc. This file tells the Servlet container which Servlet class needs to be called for the given URL pattern.

Q-56: What are the advantages of cookies?

A: Cookies are used to store long term information that can be maintained without server interaction. Small and Medium size data are kept in a queue.

**Q-57: What is the difference between Server and Container?**

**A:** A server can provide service to the client and it contains one or more containers such as EJBs, Servlet, JSP containers. Containers hold set of objects.

**web server:** Is responsible for receiving request from client and sending

responses to client.  
**web container:** w.c contains Servlet engine and jsp engine, responsible for executing servlets(JSP) life cycle.once webserver receivs the request it sends the control to web container,after execution of servlet(jsp) response will   
be sent to web server ,web server will send the response to   
client.  
**Application server**: web container + EJB container(engine)

**Q-58: What is the use of <load-on-startup>?**

**A**: If more than one files are specified then the files will be loaded in the same order in which they have been specified in it at the Container start-up.

load-on-startup can specify an (optional) integer value. If the value is 0 or greater, it indicates an order for servlets to be loaded, servlets with higher numbers get loaded after servlets with lower numbers.

<servlet>

<servlet-name>MyServletNameHere</servlet-name>

<servlet-class>ServletClassHere-FullyQualified</servlet-class>

<load-on-startup>1</load-on-startup>

</servlet>

**Q-59:What if we pass negative value in load-on-startup?**

**A:**It will not affect the container, now servlet will be loaded at first request.

**Q-60: if you are not specifying <load-on-startup> then what will happen.**

**A:** file will be load when user send first request.

**Q-61: How can be get the ServletContext data and ServletConfig data in our java program.**

**Q-62: Servlet is having any connection with JVM or not**

**A:** Using servlets allows the JVM to handle each request within a separate Java thread, and this is one of the key advantage of Servlet container. Each servlet is a Java class with special elements responding to HTTP requests. The main function of Servlet contain is to forward requests to correct servlet for processing, and return the dynamically generated results to the correct location after the JVM has processed them. In most cases servlet container runs in a single JVM, but there are solutions when container need multiple JVMs.

**Q-63: How can be create servlet single threaded.**

**A:** By impleneting SingleThreadModel interface

**Q-64: if I make servlet Single – Threaded. Then how it is work.**

**A:** in this case , Container create new servlet instance for every instance.

**Q-65: Write the Tomcat Hierarchy ->what you do in conf-folder**

Q-66: can be configure EJB in apache Tomcat.

A: yes

**Q-67: What is servlet chaining? How to achieve servlet chaining.**

**A**: Servlet Chaining is a method where the output of one servlet is piped into a second servlet. The output of the second servlet could be piped into a third servlet, and so on. The last servlet in the chain returns the output to the Web browser We can achive Servlet Chaining by Request Dispatcher.

Public class ServletA extends HttpServlet

{

Protected void service(HttpServletRequest req, HttpServletResponse res)

{

System.out.println(“\*\*ServletA Service() started”);

RequestDispatcher rd = req.getRequestDispatcher(“b.jlc”); // b.jlc is url pattern of

ServletB

System.out.println(“\*\*ServletA Service() completed”);

}

}

Public class ServletB extends HttpServlet

{

Protected void service(HttpServletRequest req, HttpServletResponse res)

{

System.out.println(“\*\*ServletB Service() started”);

RequestDispatcher rd = req.getRequestDispatcher(“c.jlc”); // c.jlc is url pattern of

ServletC

System.out.println(“\*\*ServletB Service() completed”);

}

}