QUESTIONS : https://javarevisited.blogspot.com/2011/09/servlet-interview-questions-answers.html

https://www.journaldev.com/2015/servlet-interview-questions-and-answers

1. What is initialization order of servlet ? What are lifecycle method and who calls them?

2. How Servlet and JSP are related to each other ?

3. What is Servlet Container, What are responsibilities of Servlet Container?

4. What is difference between ServletContext and ServletConfig?

5. How to control initialization of Servlet? How will you ensure pre initialization of Servlet?

6. What is difference between GenericServlet and HttpServlet?

7. What is difference between doGET() and doPOST() method?

8. Who calls doGET() and doPOST() method ?

9. Who creates object of HttpServletRequest and HttpServletResponse?

10. How do you make a Servlet thread safe ?

11. What is difference between Servlet and Filter?

12. What is lifecycle of filter?

13. How do you make your application to run using SSL?

14. How do you ensure that a particular servlet can only be accessed after authetincation?

15. How to implement authentication and authorization in Servlet application?

16. What is advantage of Servlet over CGI?

1. **what is load on startup?**

Load-on-startup is a tag element which appear inside <servlet> tag in web.xml. load-on-startup tells the web container about loading of a particular servlet. if you don't specify load-on-startup then container will load a particular servlet when it feels necessary most likely when first request for that servlet will come, this may lead to longer response time for that query if Servlet is making [database connections](http://javarevisited.blogspot.com/2011/11/database-transaction-tutorial-example.html) or performing [ldap authentication](http://javarevisited.blogspot.com/2011/09/spring-interview-questions-answers-j2ee.html) which contribute network latency or any other time consuming job, to avoid this, web container provides you a mean to specify certain servlet to be loaded during deployment time of application by using load-on-startup parameter.

If you specify load-on-startup parameter inside a servlet than based upon its value, Container will load it.you can specify any value to this element but in case of load-on-startup greater than 0 ,servlet with less number will be loaded first. For example in below web.xml **AuthenticationServlet** will be loaded before **AuthorizationServlet** because load-on-startup value for AuthenticationServlet is less (2) while for AuthorizationServlet is 4.



Important points on load-on-startup element

1. If <load-on-startup> value is same for two servlet than they will be loaded in an order on which they are declared inside web.xml file.

2. if <load-on-startup> is 0 or negative integer than Servlet will be loaded when Container feels to load them.

3. <load-on-startup> guarantees loading, initialization and call to init() method of servlet by web container.

4. If there is no <load-on-startup> element for any servlet than they will be loaded when web container decides to load them.

**When to use <load-on-startup> in web.xml**

<load-on-startup> is suitable for those servlet which performs time consuming jobs e.g. Creating Database Connection pool, downloading files or data from network or prepare environment ready for servicing client in terms of initializing cache , clearing pipelines and loading important data in memory. If any of your servlet performs these jobs then declare them using <load-on-startup> element and specify order as per your business logic or what suites your application. **Remember lower the value of <load-on-startup>, servlet will be loaded first.** You can also check your web container documentation on how exactly load on start-up is supported.

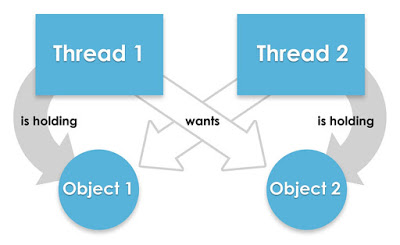
**2: How can we create deadlock condition on our servlet?**

One simple way to call doPost() method inside doGet() and doPost() method inside doGet() it will create deadlock situation for a servlet.

**What is deadlock?**  
when two or more threads are waiting for each other to release lock and get stuck for infinite time, situation is called deadlock . It will only happen in case of multitasking.

**How do you detect deadlock in Java ?**

Look at the code if you see nested synchronized block or calling one synchronized method from other or trying to get lock on different object then there is good chance of deadlock if developer is not very careful.



See Programs **https://javarevisited.blogspot.com/2010/10/what-is-deadlock-in-java-how-to-fix-it.html**

3: For initializing a servlet can we use a constructor in place of init()?

No, we can not use constructor for initializing a servlet because for initialization we need an object of servletConfig using this object we get all the parameter which are defined in deployment descriptor for initializing a servlet and in servlet class we have only default constructor according to older version of java so if we want to pass a Config object we don’t have parametrized constructor and apart from this servlet is loaded and initialized by container so it's a job of container to call the method according to servlet specification they have lifecycle method so init() method is called firstly.

# Can You Declare Constructor inside Servlet Class?

Yes, *Servlet can have Constructor*, it's perfectly legal but it's *not the right way to initialize* your Servlet. You should use the [init()](http://java67.blogspot.com/2012/09/difference-between-servletconfig-and-servletcontext-j2ee-jsp.html) method provided by the Servlet interface to initialize the Servlet. If you remember, Servlet's are special in a sense that they are instantiated by the container and managed by the container**. A servlet container like Tomcat creates a pool of multiple Servlets to serve multiple clients at the same time. They instantiate Servlet by calling the** [**default no-argument constructor**](http://javarevisited.blogspot.com/2014/01/why-default-or-no-argument-constructor-java-class.html) **and suppose you have declared another constructor which takes a parameter e.g. HelloServlet(String name) than Java compiler will not add the default no-argument constructor and Servlet container will not able to initialize the Servlet**. That's why it's important not to provide a constructor in Servlet, but if you do, make sure you also add a [default constructor](http://java67.blogspot.com/2014/09/Why-constructor-is-important-in-java-example.html) there for Servlet container.