Servlet Lecture-2

Outline

❖ Lifecycle of a Servlet

❖ How Servlet works?

❖ How to execute Servlet?

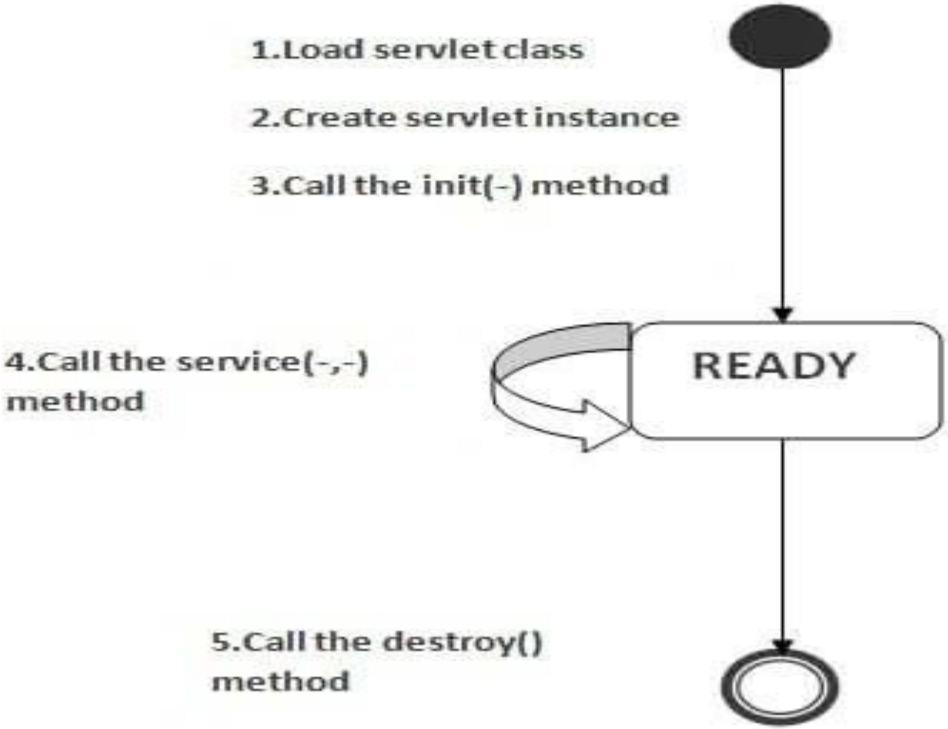
❖ How web container handles the servlet request?

❖ ServletRequest Interface

Lifecycle of a Servlet

- The web container maintains the life cycle of a servlet instance.

  - Servlet class is loaded.
  - Servlet instance is created.
  - init() method is invoked.
  - service() method is invoked.
  - destroy() method is invoked.



1.Load servlet class

2.Create servlet instance

3.Call the init(-) method

4.Call the service(-,-) method

5.Call the destroy() method

How Servlet Works?

- First the server checks if the servlet is requested for the first time.
  - If yes, web container does the following tasks:
    - Loads the servlet class.
    - Instantiates the servlet class.
    - Calls the init method passing the ServletConfig object.

- Else
  - Calls the service method passing request and response objects.

- The web container call the destroy method when it needs to remove the servlet such as at time of stopping server or undeploying the project.

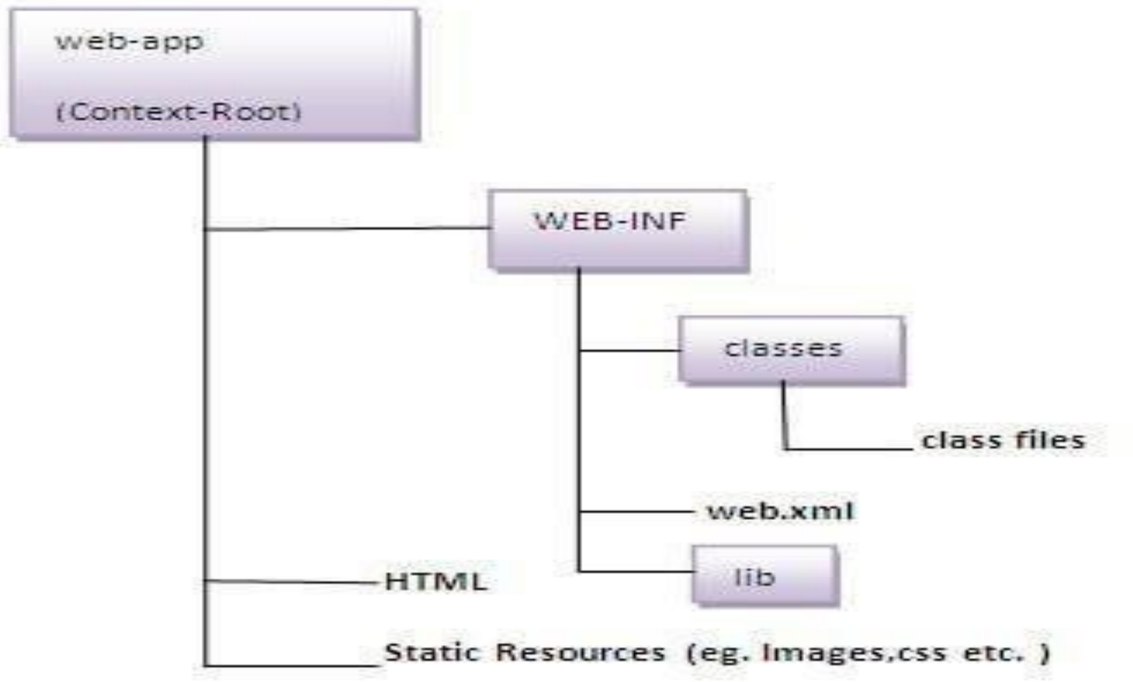
How to run Servlet

1. Create a directory structure.
2. Create a servlet.
3. Compile the servlet.
4. Create a deployment descriptor.
5. Start the server and deploy the project.
6. Access the servlet

Create a directory structure

- The directory structure defines that where to put the different types of files so that web container may get the information and respond to the client.

- The Sun Microsystem defines a unique standard to be followed by all the server vendors


web-app
(Context-Root)
WEB-INF
classes
class files
web.xml
lib
HTML
Static Resources (eg. Images,css etc.)

Create a servlet

- There are three ways to create the servlet.
  - By implementing the Servlet interface.
  - By inheriting the GenericServlet class.
  - By inheriting the HttpServlet class.

- The HttpServlet class is widely used to create the servlet because it provides methods to handle http requests such as doGet(), doPost(), doPost() etc.

Compile the Servlet

- For compiling the Servlet, jar file is required to be loaded. Different servers provide different .jar files.

Create the Deployment Descriptor(web.xml)

- The deployment descriptor is an xml file, from which Web Container gets the information about the servlet to be invoked.

- The web container uses the Parser to get the information from the web.xml file.

- There are many xml parsers such as SAX, DOM and Pull.

Example

```
<web-app>

        <servlet>
                <servlet-name>ABC</servlet-name>
                <servlet-class>HelloWorld</servlet-class>
        </servlet>

        <servlet-mapping>
                <servlet-name>ABC</servlet-name>
                <url-pattern>/Hello</url-pattern>
        </servlet-mapping>

</web-app>
```

Description of the elements of web.xml

- **<web-app>** represents the whole application.
- **<servlet>** is sub element of <web-app> and represents the servlet.
- **<servlet-name>** is sub element of <servlet> represents the name of the servlet.
- **<servlet-class>** is sub element of <servlet> represents the class name of the servlet.
- **<servlet-mapping>** is sub element of <web-app>. It is used to map the servlet.
- **<url-pattern>** is sub element of <servlet-mapping>. This pattern is used at client side to invoke the servlet.

How web container handles the servlet requests?

- The web container is responsible to handle the request.
  - Maps the request with the servlet in the web.xml file.
  - Creates request and response objects for each request.
  - Calls the service method on the thread.
  - The public service method internally calls the protected service method
  - The protected service method calls the doGet, doPost etc. depending on the type of request.
  - The doGet method generates the response and it is passed to the client.
  - After sending the response, the web container deletes the request and response objects.
  - The threads which is contained in the thread pool are deleted depends on the server implementation.

ServletRequest Interface

- An object of ServletRequest is used to provide the client request information to a servlet such as content type, content length, parameter names and values, header information, attributes etc.

- Methods of ServletRequest Interface
  - **public String getParameter(String name)**
    - is used to obtain the value of a parameter by name.
  - **public String[] getParameterValues(String name)**
    - returns an array of String containing all values of given parameter name. It is mainly used to obtain values of a Multi select list box.

Example -1(get parameter from index.html)

```
// index.html

<html>
    <body>
        <form method = "get" action = "Hello">
            <input type= "text" name = "name1">
            <button type = "submit" >Submit</button>
        </form>
    </body>
</html>
```

Continue...

```
//HelloWorld.java

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

public class HelloWorld extends HttpServlet{

    public void doGet(HttpServletRequest req, HttpServletResponse res)throws IOException, ServletException{

        res.setContentType("text/html");
        String name = req.getParameter("name1");
        PrintWriter out = res.getWriter();
        out.println("<html><body>");
        out.println("Hello " + name);
        out.println("</html></body>");}
}
```

Continue...

```
//web.xml

<web-app>

    <servlet>
        <servlet-name>s1</servlet-name>
        <servlet-class>HelloWorld</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>s1</servlet-name>
        <url-pattern>/Hello</url-pattern>
    </servlet-mapping>
</web-app>
```