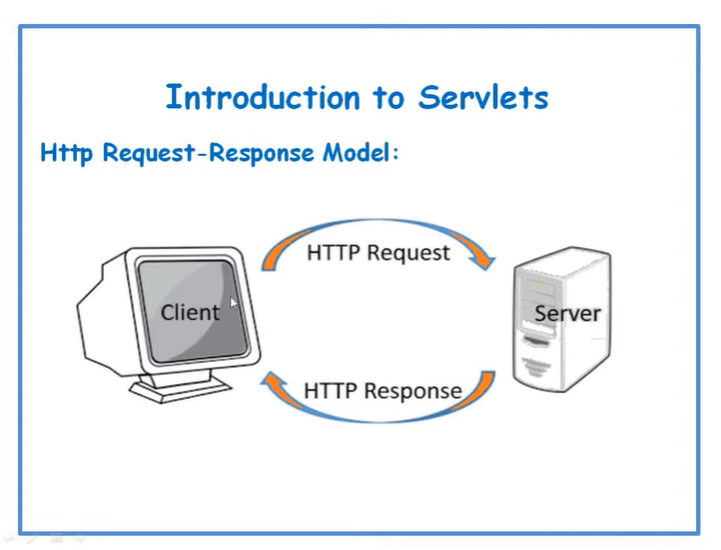
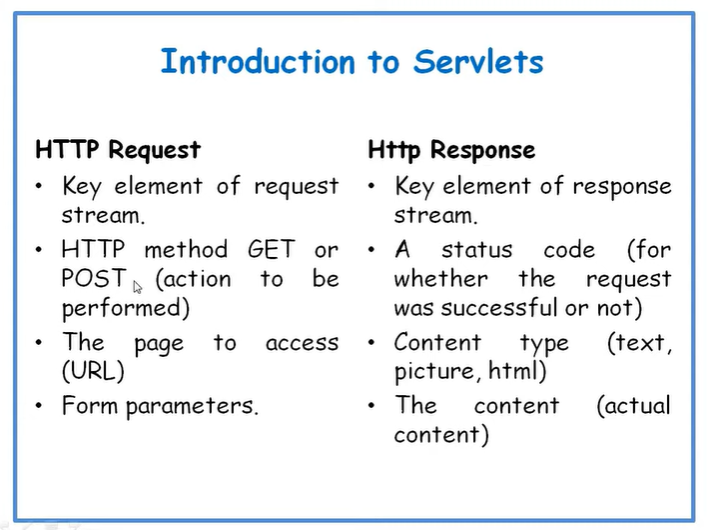


Stateless: It does not remember the previously held communication b/w client and server



Here server only handles static web pages, it doesn’t handle dynamic web pages.

Here 4 Elements in HTTP request and response separately

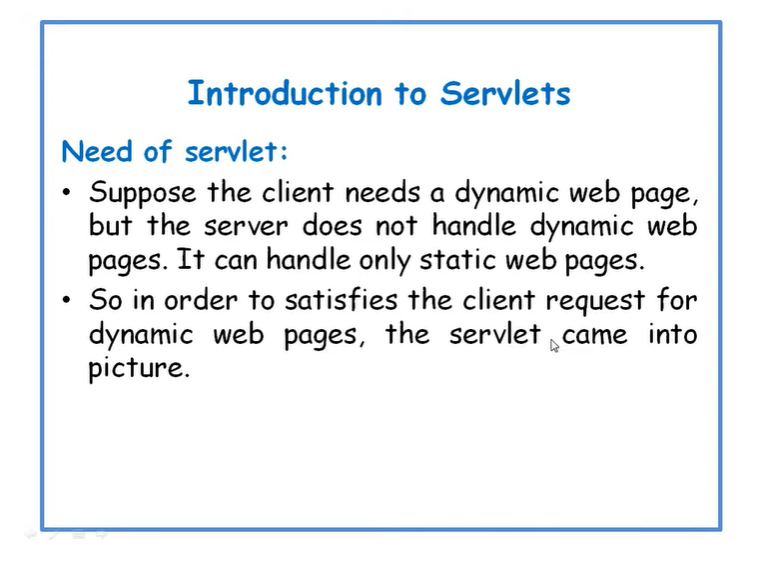
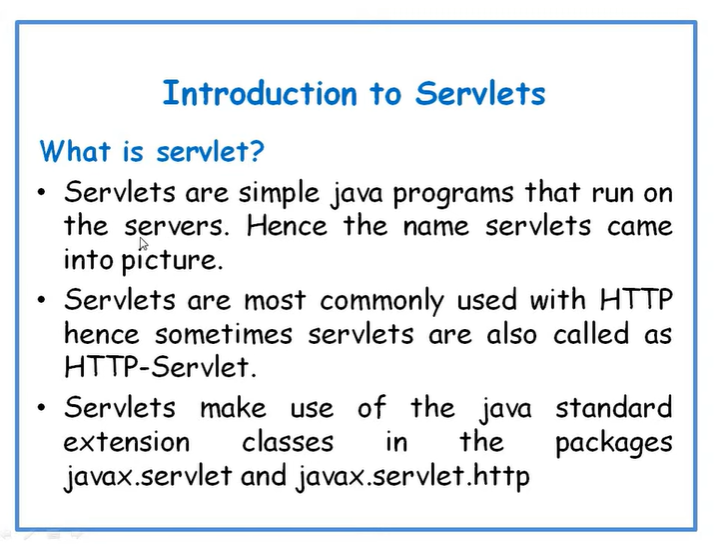


Post Method is secure than Get method,

http-404 request fail/ http-200 request success

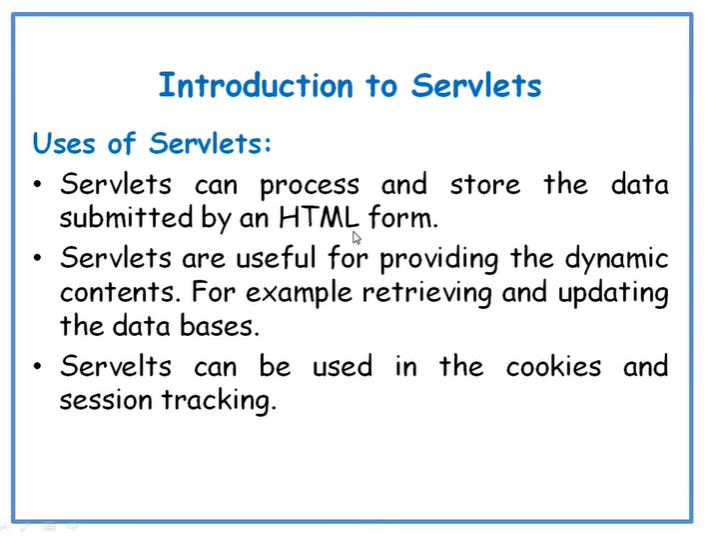
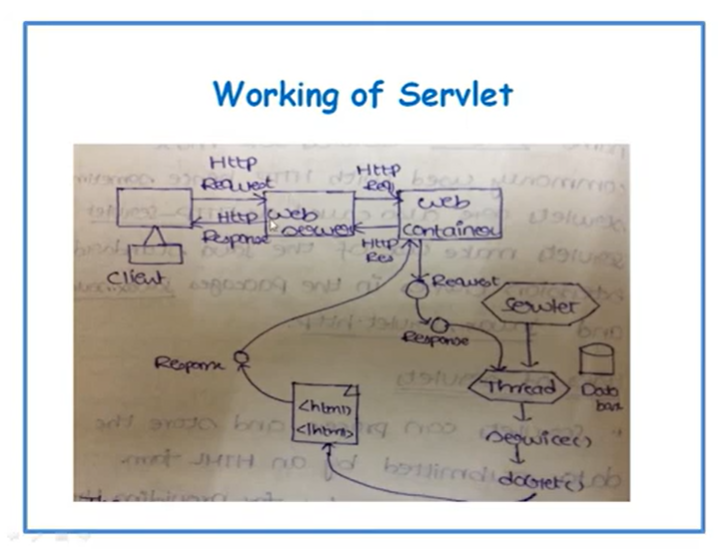
Here server only handles static web pages, it doesn’t handle dynamic web pages.

To satisfy the client, when it gone through dynamic pages then we Need the concept of dynamic web pages.

Here We use two pakages: javax.servlet and javax.servlet.http

Here ‘X’ means extension

WebContainer maintains the web.xml file in order to know which/what request is raised by the client.

It Convers http request into Request Object, since servelets are part of java programmig, java is a object orientet programming language, everything is in the form of object.

Servelet doesn’t understand the direct http request, it accepts the valid reqest object from web container.

WebContainer also creates a thread for eery client request. It creates multiple threads for multiple client requests, So it employs multi-threading concept.

Servlet Recognize the requet received from the client through webcontainer it gives appropriate response by using call back methods. It calls the method i.e; service( )

If the request method type from the client is get(), then service() call the do\_get( ) method,

If the request method type from the client is post(), then service() call the do\_post( ) method.

In the form parameters if we use get(), then call method invokes do\_get( ) method,

if we use post(), then call method invokes do\_post( ) method.

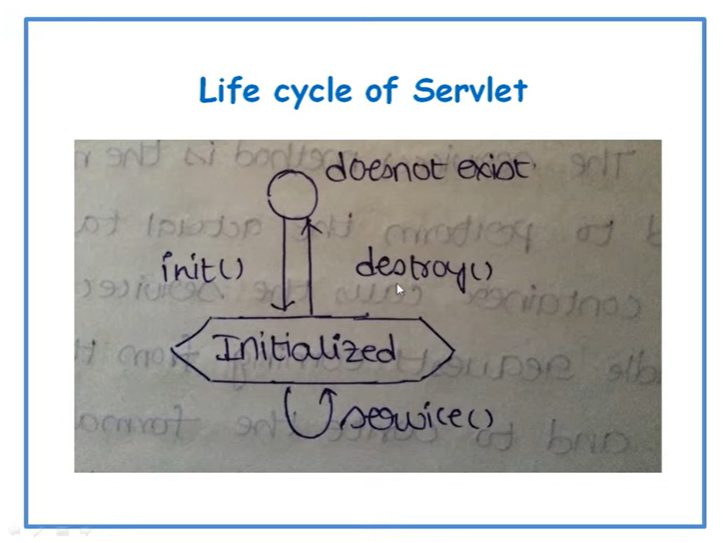
Here the method i.e; what ever it may be , the main job is to process the client request and get the response from the server.

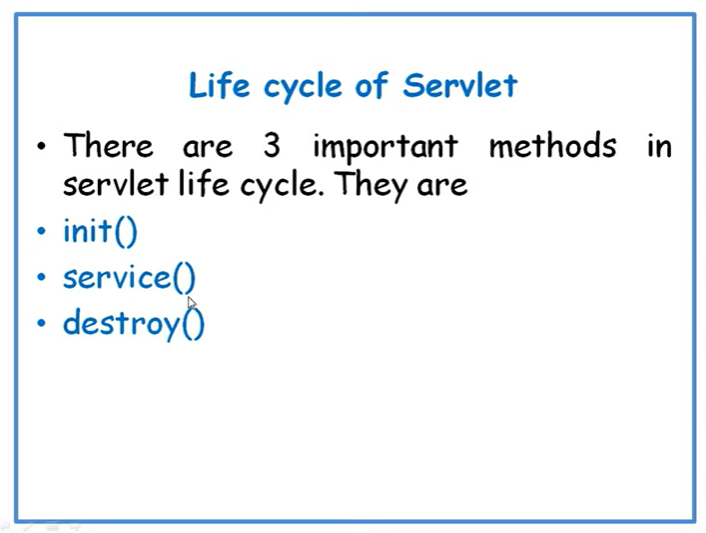
Valid business logic would be written in the do get ( ) or do post( ) methods only.

After processing the clent request and while giving the response to the client, the response also In the form of object only.

Again web container converts the response object into valid http-response since web-server doesn’t know the response object.

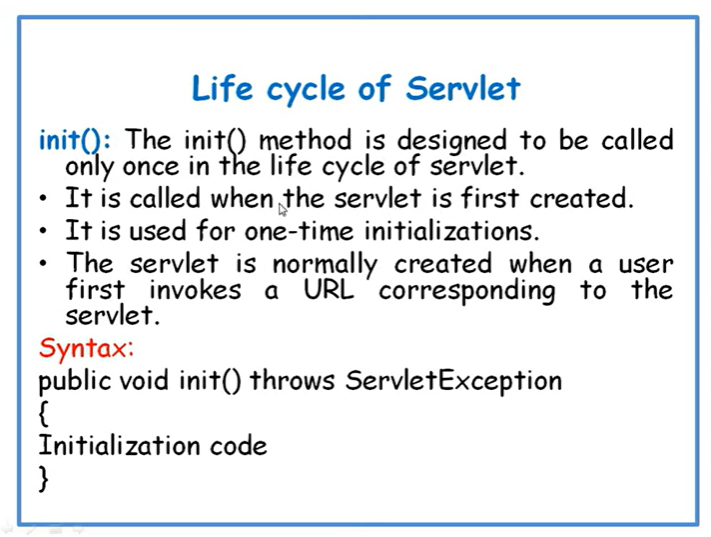
Finally web server sends the valid http-response to the client.





The primary stage/state of servlet is does not exist state.

When user gives the URL as a request, the servlet is intitialized by init() method.



Public -------- is a access modifier, we can access this method anywhere

Void ----------is a return type

Init() ---------is a method

Throws----------- is a keyword followed by exception name

servletException---------- is a exception name

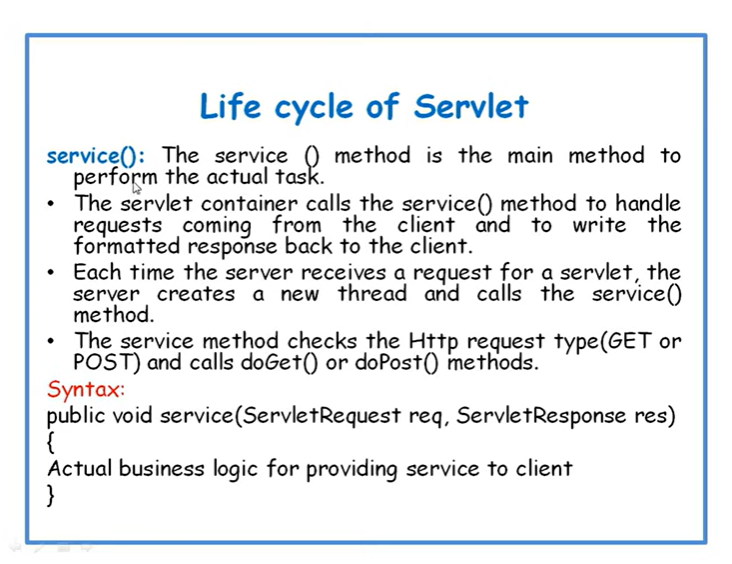
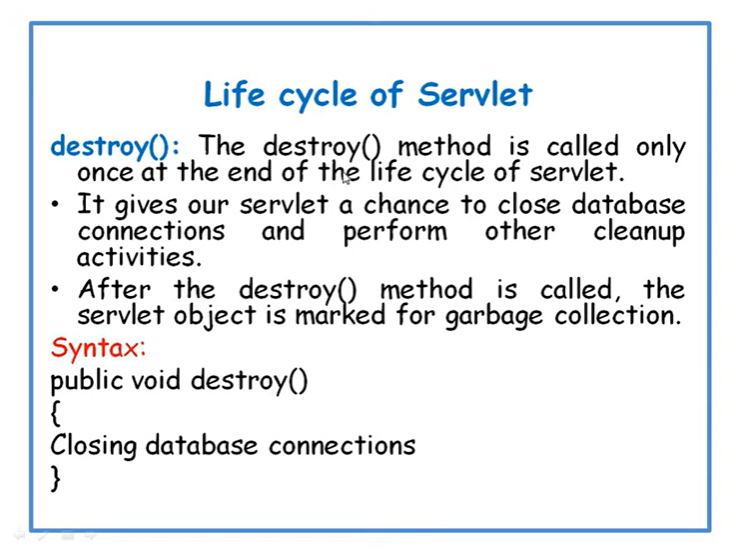
intialization code--------- here we use

try {

}

Catch{

}

**WebServer Installation:**

<https://tomcat.apache.org/download-9.cgi>

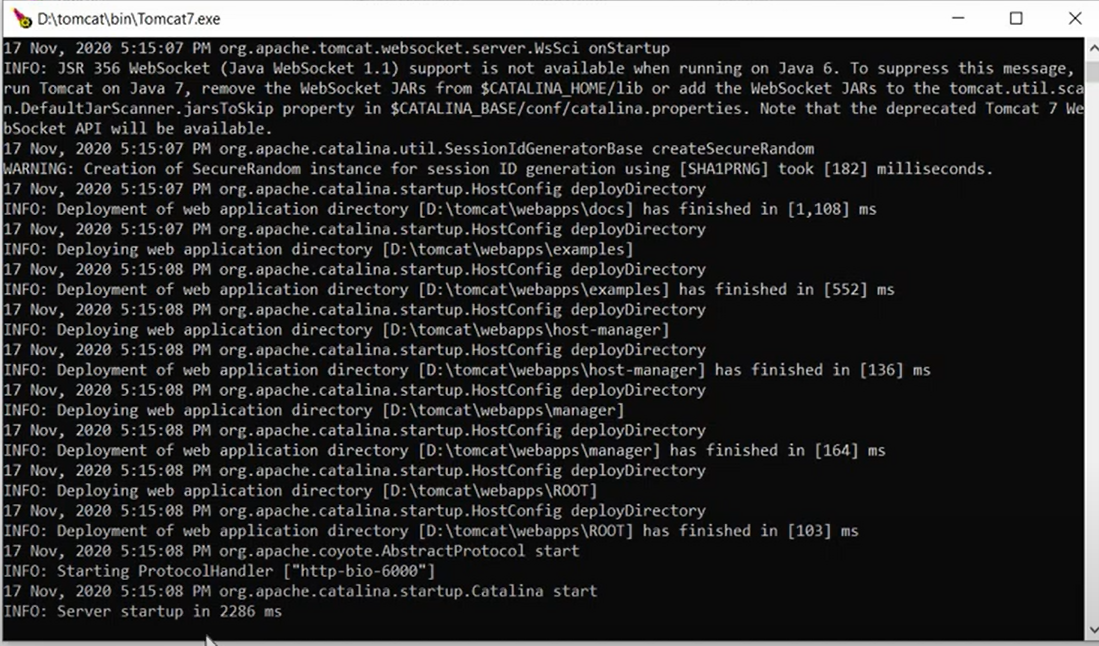
Go to the location where you installed(port number :8081 given by me)

Open tomcat

Open bin folder

You will find tomcat9 and tomcat9 w file, double click on tomcat9 file

Click on YES, Allow



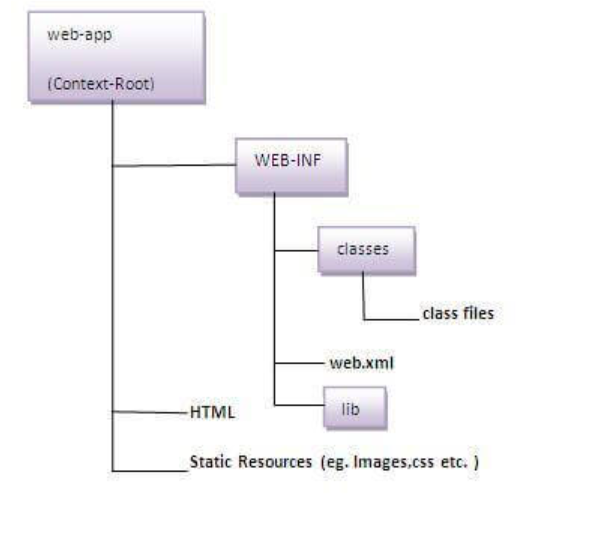
You will get above

INFO : Server startup in 2286(varried value) ms indicates successful installion of tomcat

Goto webapps folder (C:\Program Files\Apache Software Foundation\Tomcat 11.0\bin \webapps)

In webapps create a new folder i.e; WEB-INF

In WEB-INF folder create two folders 1.classes 2.lib



ServletDemo.java

import javax.servlet.http.\*;

import javax.servlet.\*;

import java.io.\*;

public class ServletDemo extends HttpServlet {

    String msg;

    public void init() {

        msg = "Hello World";

    }

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

        res.setContentType("text/html"); // setting the content type

        PrintWriter out = res.getWriter(); // get the stream to write the data

        try {

            // writing html in the stream

            out.println("<html><body>");

            out.println("<h1>" + msg + "</h1>");

            out.println("</body></html>");

        } catch (Exception e) {

            out.close(); // closing the stream

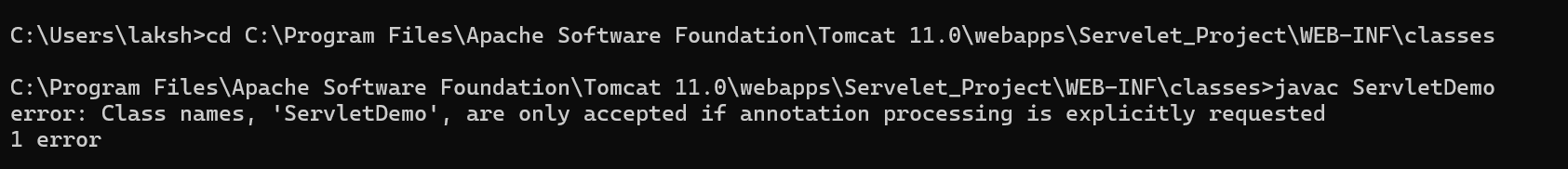
        }

    }

}

Save the file with extension .java

cd C:\Program Files\Apache Software Foundation\Tomcat 11.0\webapps\Servlet\_Project\WEB-INF\classes



You will get the error

Add the jar file while compling, You can find the jar file from below location

(C:\Program Files\Apache Software Foundation\Tomcat 11.0\lib)

C:\Program Files\Apache Software Foundation\Tomcat 11.0\webapps\Servlet\_Project\WEB-INF\classes>javac ServletDemo.java -classpath "C:\Program Files\Apache Software Foundation\Tomcat 9.0\lib\servlet-api.jar"

C:\Program Files\Apache Software Foundation\Tomcat 9.0\lib

It works temporarly unitl closing this window(its life time)

Create web.xml

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee

http://xmlns.jcp.org/xml/ns/javaee/web-app\_4\_0.xsd"

version="4.0">

<servlet>

<servlet-name>ServletDemo</servlet-name>

<servlet-class>ServletDemo</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>ServletDemo</servlet-name>

<url-pattern>/Vivek</url-pattern>

</servlet-mapping>

</web-app>

Save above file in WEB-INF folder

Goto browser

Type: <http://localhost:8081/Servlet_Project/Vivek>

Then press enter

<https://www.eclipse.org/downloads/packages/release/kepler/sr2/eclipse-ide-java-ee-developers>