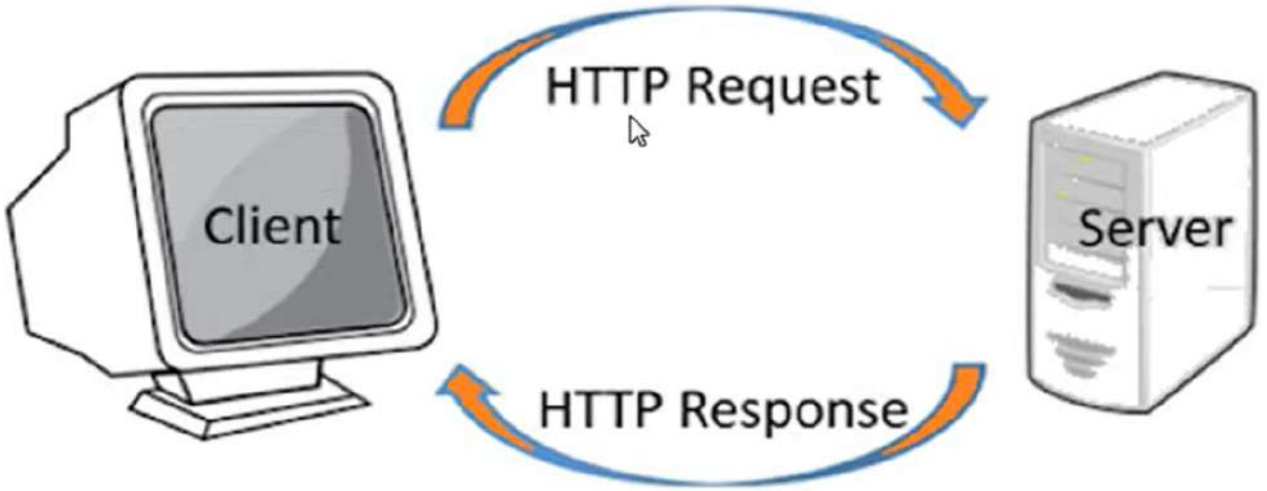
Introduction to Servlets

Http Request-Response Model:

- HTTP is a stateless request response based communication protocol.
- It is used to send and receive data on the web.
- It uses a reliable TCP connection either for the transfer data to and from clients which was web browsers.

Introduction to Servlets

Http Request-Response Model:



Introduction to Servlets

HTTP Request

- Key element of request stream.
- HTTP method GET or POST (action to be performed)
- The page to access (URL)
- Form parameters.

Http Response

- Key element of response stream.
- A status code (for whether the request was successful or not)
- Content type (text, picture, html)
- The content (actual content)

Introduction to Servlets

Need of servlet:

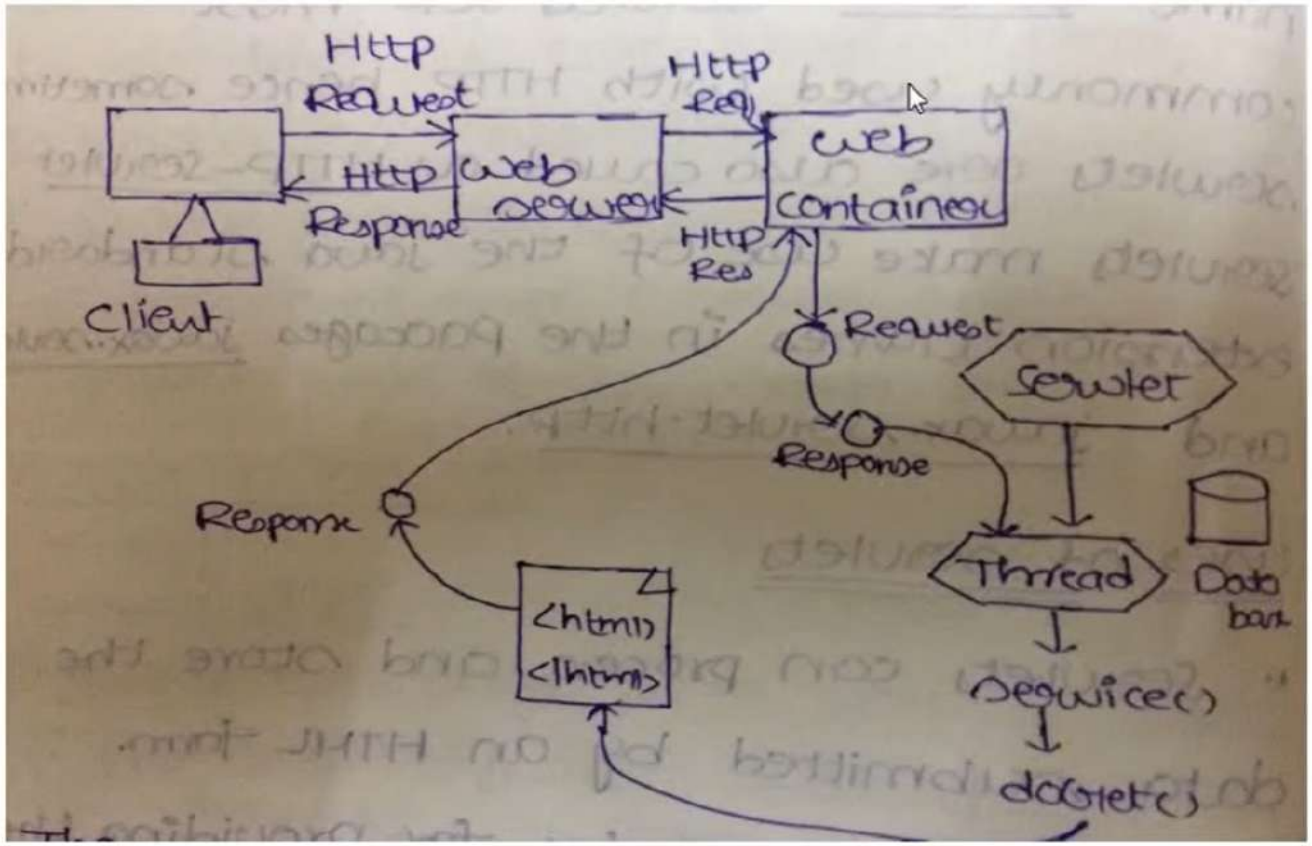
- Suppose the client needs a dynamic web page, but the server does not handle dynamic web pages. It can handle only static web pages.
- So in order to satisfies the client request for dynamic web pages, the servlet came into picture.

Introduction to Servlets

What is servlet?

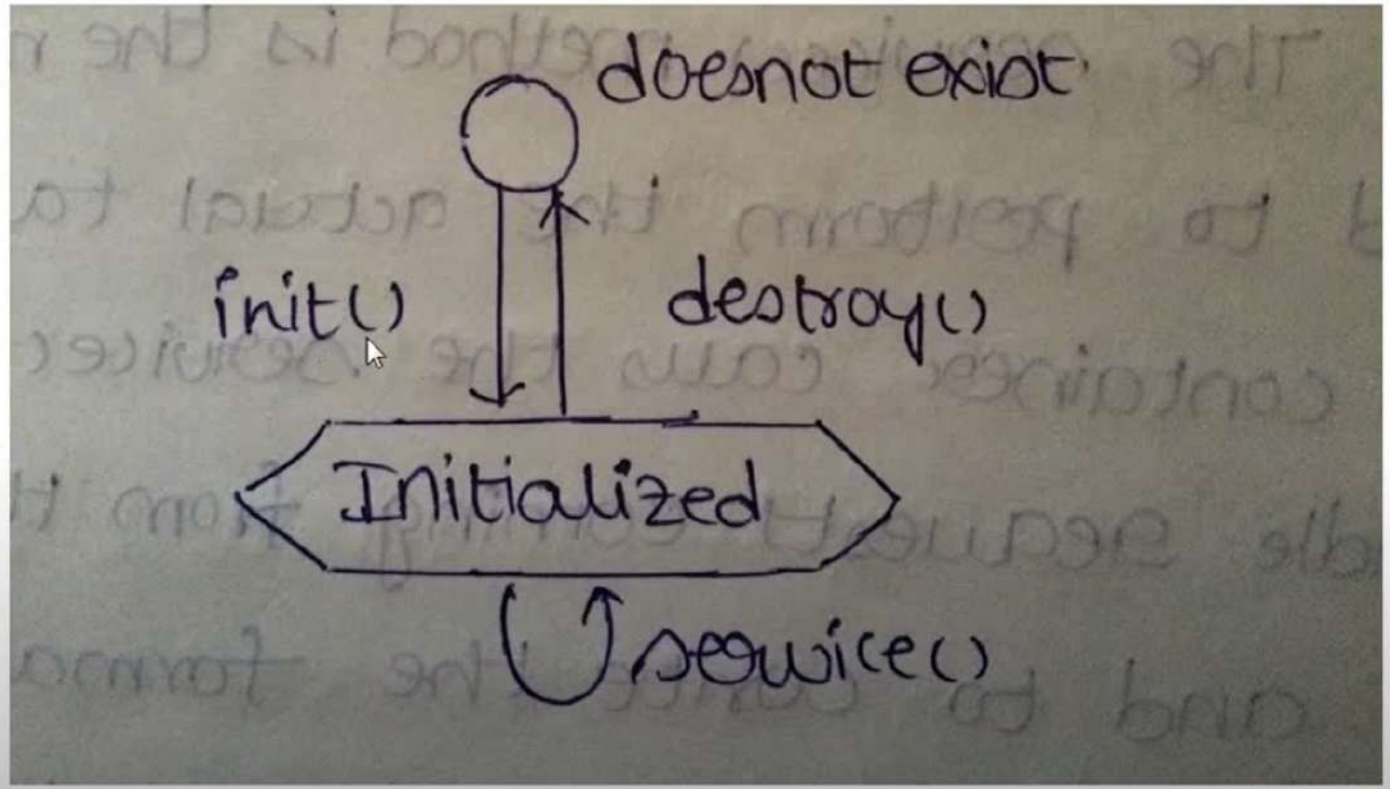
- Servlets are simple java programs that run on the servers. Hence the name servlets came into picture.
- Servlets are most commonly used with HTTP hence sometimes servlets are also called as HTTP-Servlet.
- Servlets make use of the java standard extension classes in the packages javax.servlet and javax.servlet.http

Working of Servlet



Life Cycle of Servlet|| Servlets in Telugu Lecture-3

Life cycle of Servlet



0:18 / 9:04

Life cycle of Servlet

init(): The init() method is designed to be called only once in the life cycle of servlet.

- It is called when the servlet is first created.
- It is used for one-time initializations.
- The servlet is normally created when a user first invokes a URL corresponding to the servlet.

Syntax:

```
public void init() throws ServletException
{
Initialization code
}
```

Life cycle of Servlet

service(): The service () method is the main method to perform the actual task.

- The servlet container calls the service() method to handle requests coming from the client and to write the formatted response back to the client.
- Each time the server receives a request for a servlet, the server creates a new thread and calls the service() method.
- The service method checks the Http request type(GET or POST) and calls doGet() or doPost() methods.

Syntax:

```
public void service(ServletRequest req, ServletResponse res)
{
Actual business logic for providing service to client
}
```

Life cycle of Servlet

service(): The service () method is the main method to perform the actual task.

- The servlet container calls the service() method to handle requests coming from the client and to write the formatted response back to the client.
- Each time the server receives a request for a servlet, the server creates a new thread and calls the service() method.
- The service method checks the Http request type(GET or POST) and calls doGet() or doPost() methods.

Syntax:

```
public void service(ServletRequest req, ServletResponse res)
{
Actual business logic for providing service to client
}
```

Life cycle of Servlet

destroy(): The destroy() method is called only once at the end of the life cycle of servlet.

- It gives our servlet a chance to close database connections and perform other cleanup activities.
- After the destroy() method is called, the servlet object is marked for garbage collection.

Syntax:

```
public void destroy()
{
Closing database connections
}
```

Differences between doGet() & doPost()

doGet()	doPost()
The doGet() is the default method of HttpServletRequest	The doPost() is the default method of HttpServletResponse
In this user entered data is appended to URL as query string	In this user entered data is not appended to URL
It provides less security	It provides more security
It can send only limited amount of data	It can send any amount of data
It is generally used to query or get some information from the server	It is used to update or post some information to the server
In doGet() method parameters are not encrypted	In doPost() method parameters are encrypted
It is faster	It is slower

Session Tracking

- Basically there are two types of protocols.
- Stateful protocol
- Stateless protocol

Session Tracking

- Stateful protocol: In this protocol, part of data is exchanged between client and server and these protocols always keep tack of communication sessions.

Example: TCP

- Stateless protocol: In this protocol, part of data is exchanged between client and server and these protocols can not remember previously held communications.

Example: HTTP

- But some times there is a need to keep track of previous communication sessions. In such situations we go for session tracking.

Session Tracking

- The session tracking is a mechanism by which we can keep track of previous sessions between server and the browser.
- For creating the sessions getSession() method can be used.
- This method returns the object which stores the bindings with the names that use this object.
- These bindings can be managed using getAttribute() and setAttribute() methods.

Session Tracking

- In session tracking two things are playing an important role.
- One is HttpServletRequest interface which supports getSession() method.
- The another class is HttpSession class which supports the binding managing methods such as getAttribute() and setAttribute().

Cookies

- Cookies are small programs which can make use of information submitted on currently accessed web pages.
- A cookie is a key-value pair created by the server and is installed on the client's browser when the client makes a request for the first time.

Cookies

- Browsers also maintained a list of cookies installed in them and send them to the server as a part of sub sequent HTTP Requests.
- The server can then easily identify that this request is a part of a sequence of related requests.
- That is why cookies provide an elegant solution to session tracking.
- Cookie is represented using the class Cookie.

Syntax:

```
Cookie(String key, String value);
```

Cookies

- The cookie is added by the addCookie() method of the HttpServletResponse interface.
- The server can get all cookies sent by the web browser using getCookie() method of the HttpServletRequest interface.