Introduction

- Basically, a java program that runs on the server.
- Creates dynamic web pages.
- **Servlet** technology is used to create web application (resides at server side and generates dynamic web page).

Interfaces and Classes

- There are many interfaces and classes in the servlet API such as Servlet, GenericServlet, HttpServlet, ServletRequest, ServletResponse etc.

What is a Servlet?

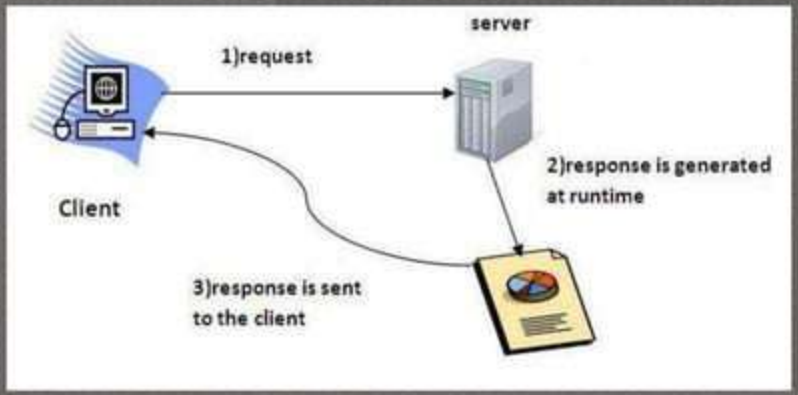
Servlet can be described in many ways, depending on the context.

- Servlet is a technology i.e. used to create web application.
- Servlet is an API that provides many interfaces and classes including documentations.
- Servlet is an interface that must be implemented for creating any servlet.

Contu..

- Servlet is a class that extend the capabilities of the servers and respond to the incoming request. It can respond to any type of requests.
- Servlet is a web component that is deployed on the server to create dynamic web page.

Request and Response



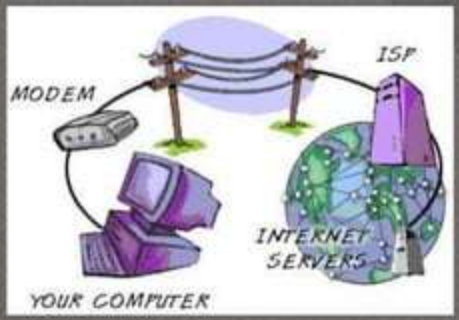
Request and response through URL

- Servlets are Java objects which respond to HTTP requests. Servlets may return data of any type but they often return HTML.
- Servlets are invoked through a URL which means that a servlet can be invoked from a browser.
- Servlets can be passed parameters via the HTTP request.

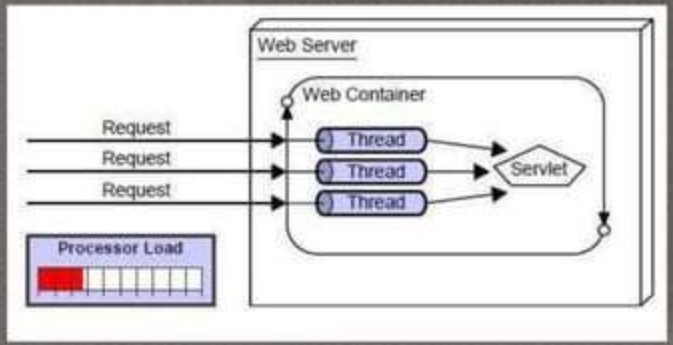
http://www.somehost.com/servlet/search?word=Java&language=english

What Servlets Do?

- Handle data/requests sent by users (clients)
- Create and format results
- Send results back to user



During Servlet Requests



Packages of Servlets

All servlets must import the following packages:

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

- The javax.servlet and javax.servlet.http packages represent interfaces and classes for servlet api.
- The **javax.servlet** package contains many interfaces and classes that are used by the servlet or web container. These are not specific to any protocol.
- The **javax.servlet.http** package contains interfaces and classes that are responsible for http requests only.

doPost and doGet

- Under HTTP, there are two methods available for sending parameters from the browser to the web server. They are POST and GET
- POST and GET are virtually the same except in terms of how parameter data is passed

- GET: Parameters are encoded within the URL. If data is passed via GET, it is limited in size to 2K
- POST: Parameters are sent AFTER the HTTP header in the request. There is no limit to the size of parameters sent via POST.

- The method of parameter passing is defined in the HTML loaded into the browser. Servlet authors can choose to implement the doGet method, doPost method or both.

The doGet() Method

Syntax:

public void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException { // Servlet code }

The doPost() Method

Syntax:

public void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException { // Servlet code }

Request and Response - GET v/s POST

- The HTTP request method determines whether doGet() or doPost() runs.

	GET (doGet())	POST (doPost())
HTTP Request	The request contains only the request line and HTTP header.	Along with request line and header it also contains HTTP body.
Parameter passing	The form elements are passed to the server by appending at the end of the URL.	The form elements are passed in the body of the HTTP request.
Size	The parameter data is limited (the limit depends on the container)	Can send huge amount of data to the server.
Idempotency	GET is Idempotent	POST is not idempotent
Usage	Generally used to fetch some information from the host.	Generally used to process the sent data.

Servlet Life Cycle

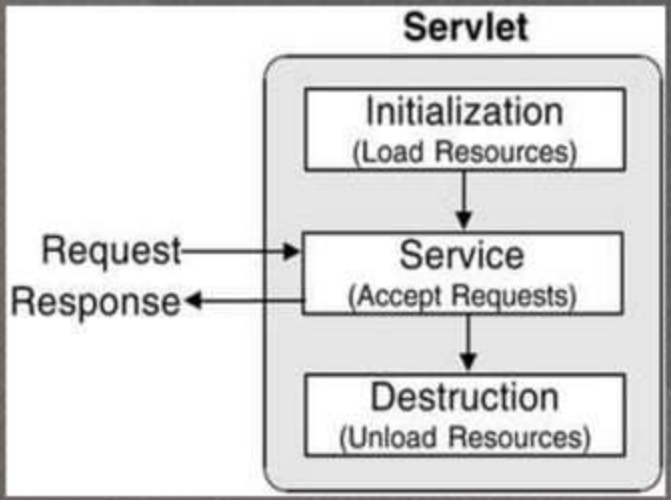
- *The servlet life cycle is the Java servlet processing event sequence that occurs from servlet instance creation to destruction. The servlet life cycle is controlled by the container that deploys the servlet.*

Life Cycle

The life cycle of servlet follows three steps:

- Initialize
- Service
- Destroy

Flow chart

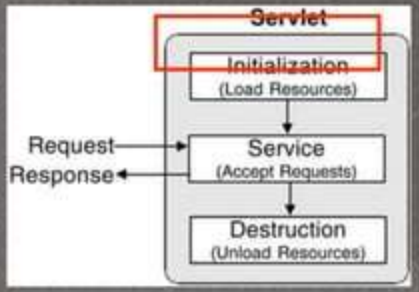

Servlet
Initialization
(Load Resources)
Request
Response
Service
(Accept Requests)
Destruction
(Unload Resources)

Life Cycle: Initialize

- Servlet is created when servlet container receives a request from the client

- Init() method is called only once

```
publiic void int ( ) throws serveletException {
// initialization code....
}
```



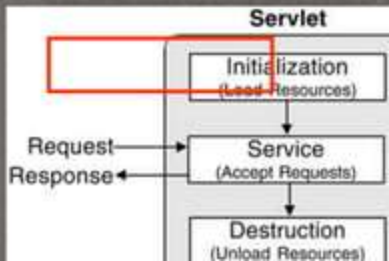
Life Cycle: Service

- Method service() is invoked every time a request comes it. It spawns off threads to perform doGet or doPost based on the method invoked

Public void service (servlet request)
(servlet response)

throws servlet Exception IOException

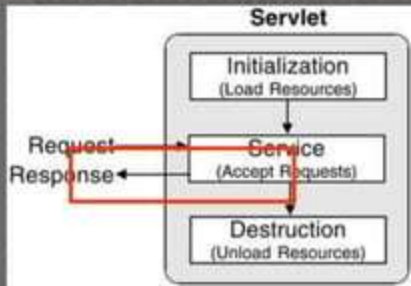
{ }



Life Cycle: Destroy

- destroy() method is called only once
- Occurs when
  - Application is stopped
  - Servlet container shuts down
- Allows resources to be freed

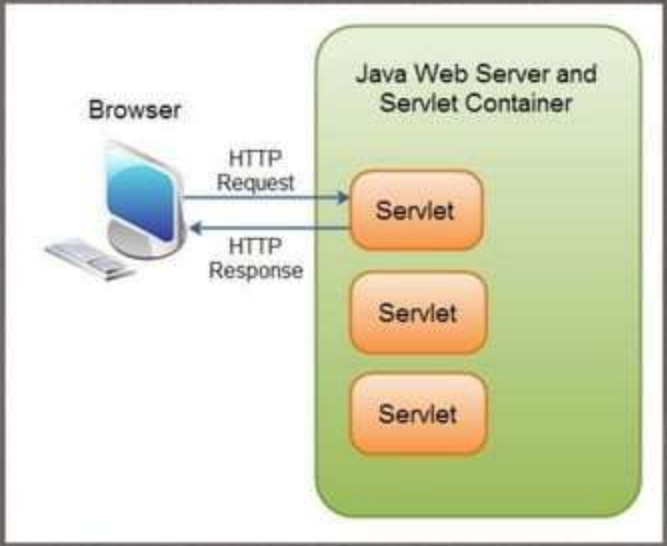
```
public void destroy ( )
{
  // destruction code....
}
```



Client Interaction

- Request
  - Client (browser) sends a request containing
    - Request line (method type, URL, protocol)
    - Header variables (optional)
    - Message body (optional)
- Response
  - Sent by server to client
    - response line (server protocol and status code)
    - header variables (server and response information)
    - message body (response, such as HTML)

WORKING


Java Web Server and
Servlet Container
Browser
HTTP
Request
HTTP
Response
Servlet
Servlet
Servlet