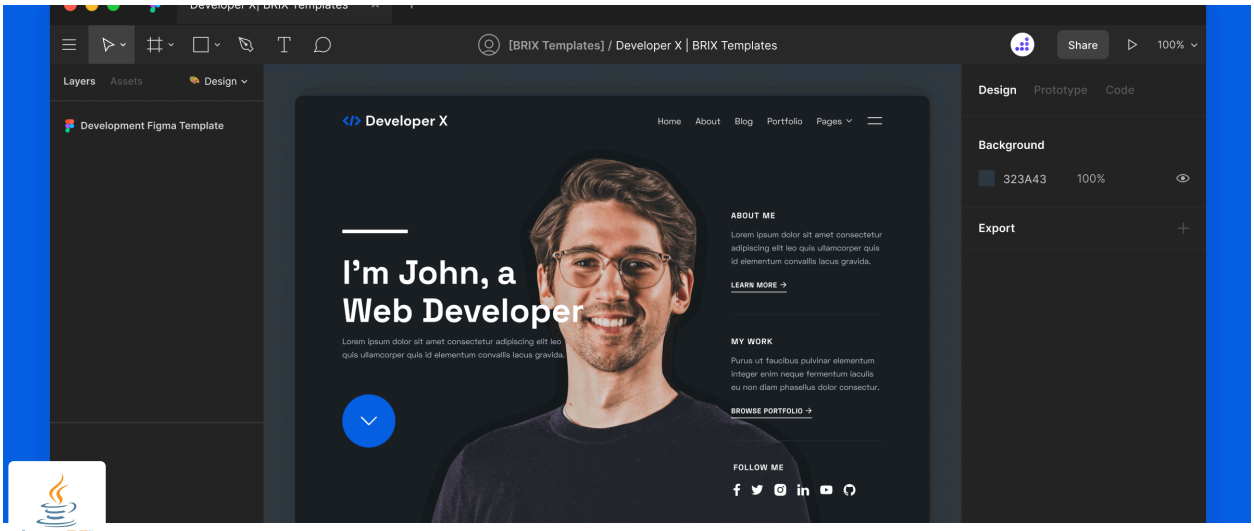


Java EE

- What is types of Application :

  Desktop Application

  Wep Application

  Mobile Application

by java EE can built the web application :

web application mean client send request to server and server comeback with the response

Why Java EE ?

- to solve Problems of Java SE How?
  - In java to built Desktop Application by Java SE
  - project in java SE combination in one file has Jar extension
  - to run project must setup this jar in each device
  - when there 100 device the previous step is wasted time
  - now you reach to Why Java EE

Web Programming Principles

- what is happen when you search by Facebook in Browser

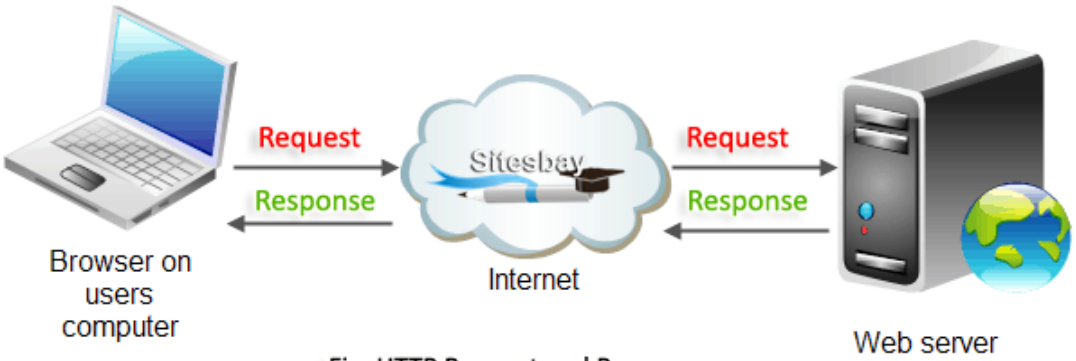


Fig: HTTP Request and Response

- Pages
  - ▼ Static web page
    1. for presentation only
    2. static form
    3. Use Html language
  - ▼ Dynamic web page
    1. interact with User
    2. dynamic form

Type of Dynamic web page

Client Site	Server Site
User's browser	Web server
JavaScript, AJAX, HTML, CSS	Server-side languages (PHP, Python, Java, etc.), databases,
Ex: website that displays a map	EX : product page on an online store.

- what is Mean Browser :

  a software program used to

  - Understand Html Only
  - locate and display information on the Internet or an intranet.
  - used to access Web pages.
  - Most can display graphics, photographs and text; multimedia information (e.g., sound and
    video)
- what is Mean HTTP :
  - HTTP (Hypertext Transfer Protocol) is a **set of rules that govern how information will be transferred between networked devices**, specifically web servers and client browsers.

- what is Mean Request:
- **a request made by a client, such as a web browser, to a server in order to retrieve a web page or other** information

▼ contains of :

- Header
  1. User-Agent (Browser)
  2. Ip
  3. Parameter
  4. Page
  5. Language
  6. What does he do?
- Body

Get Method	Post Method
Default	—-
Parameter store in Header	Parameter store in Body
Not Secure	Secure

Support Bookmarks to page	Not Support Bookmarks to page
	use when if there affects in Server as Database

- what is Mean Response :
  - **a answer of request made by a server, to return a web page or other** information into a Browser
  - ▼ contains of :
    - Header
      1. Content Type (Html as Browser Understand this page is Html Because response return to Browser , Mp3)
      2. Status Code (404 page not found, 200 page found)
    - Body
      1. Content

How Decide I want The Request or Response

Request	Response
when I want to receive information from Browser	when i want to send information from User

- what is Mean Server :
  - response to you when send request
    - ▼ Example Application server

      Tomcat

      Glassfish

      JBoss

      Web logic

    Server as Hardware Different Server Software

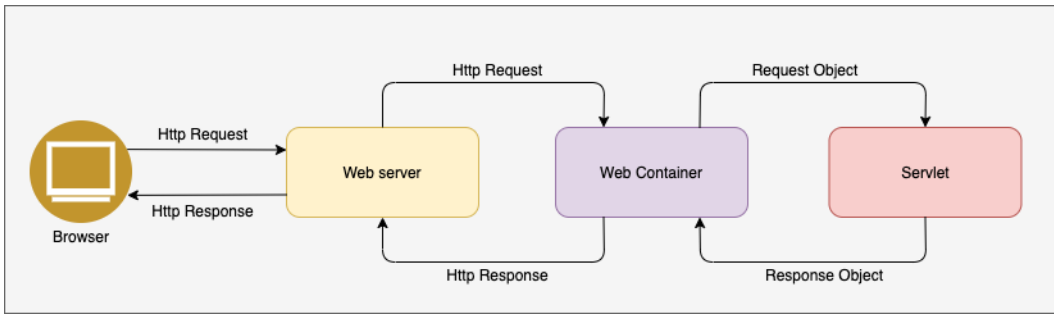
- Hardware server is a Device Hardware
- Software Server is a program

▼ contains of :

- `Web Server`
  1. the front part in server receive the pages from Browser
  2. can response to you when page is html
  3. when page is java send request to `Container`

- `Container`
  1. The back part in server receive the pages from `Web Server`
  2. response of the `Web Server` request to execute the java logic code

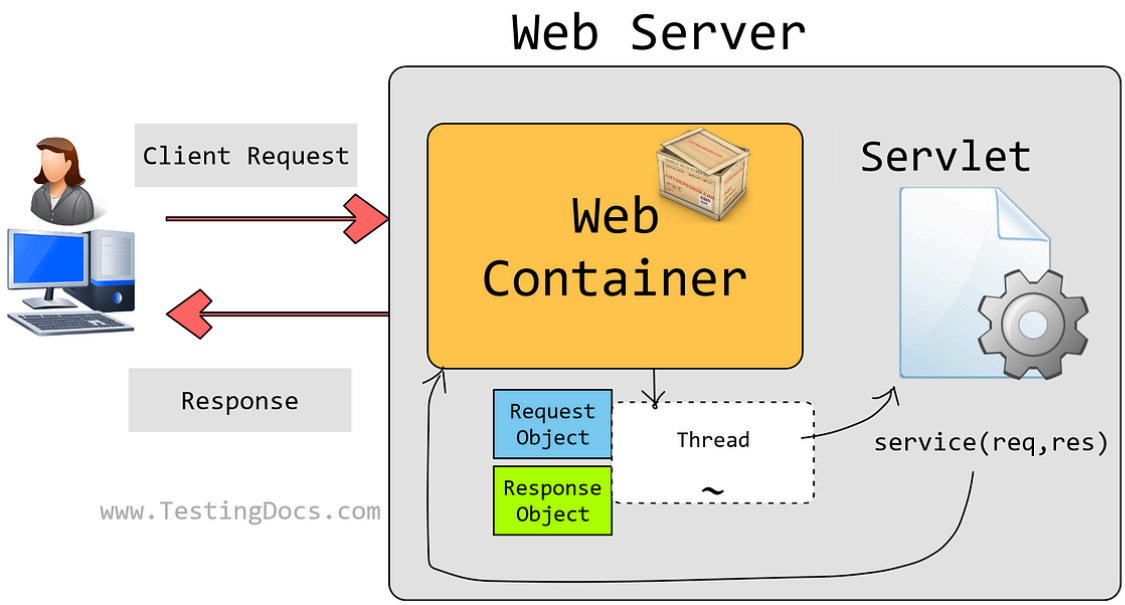
Cycle of request from Browser to Server :



Servlet

java class can handle with Browser

Life Cycle of Servlet :



Notes :

1. Container read web.xml file (or uses annotations) to map incoming requests to specific Servlets. to Know what is Servlet can Handle
2. container send request to Servlet by calls Init() Method from servlet is called *only once* in the Servlet's lifecycle, when the Servlet is first loaded and initialized by the container, not after every request.
3. Container created Two object (HttpServletRequest, HttpServletResponse)
4. Container send Two Objects to Servlet
5. Servlet receive Two Objects from Container in Service Method(req, res)
6. Servlet Execute The logic java code
7. Servlet Resend the modifications Response to Container
8. Container Destroy The Servlet By Destroy() Method is called *only once* when the Servlet is being unloaded by the container, not after every request.

Sharing data

- Container is created the object form servletConfig , servletContext because he can write the web.xml file
- The HttpSession object is created *when a client (user) first interacts with the web application*

	Request	Servlet Config	Servlet Context	Session
Scope of sharing data	Per request	data Sharing to The specific servlet	data Sharing to the whole application, across Servlets and Users	**One per user/machine, Every request object has a reference to the session object.**
Data Type	Request-specific data (parameters, headers,.)	String	Object	Object
method	request.getParameter()	getServletConfig()	getServletContext()	getSession()
Life Time	Request lifetime	per-servlet	Application-wide	Exists for the duration of a user's session (until timeout, invalidation, or browser closure)
Creation Time	With each incoming client request	When the servlet is loaded	When the application is loaded	When user Send request
Destruction Time	After the server sends the response	Servlet Destroy	application closure	browser closure
Example	get username , password from Client	Initialization parameters specific to the Servlet (e.g., database connection URL, file paths)	Application-wide settings, shared resources (e.g., database connection pool), logging	User login information, shopping cart data, user preferences

Cookies

- are small files of information that a web server generates and sends to a web browser. Web browsers store the cookies they receive for a predetermined period of time, or for the length of a user's session on a website

▼ Examples of data stored in Cookies

Session ID

User Preferences

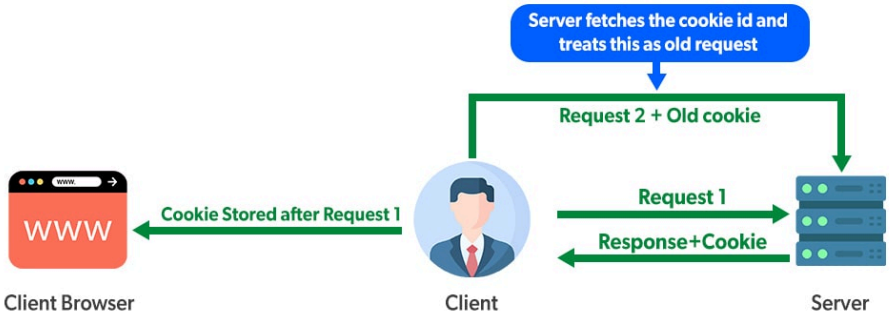
1. Location
2. Notification Settings
3. Language
4. Theme/Appearanc

Tracking and Analytics

1. Website Visits
2. Pages Visited
3. Time Spent on Site

Authentication

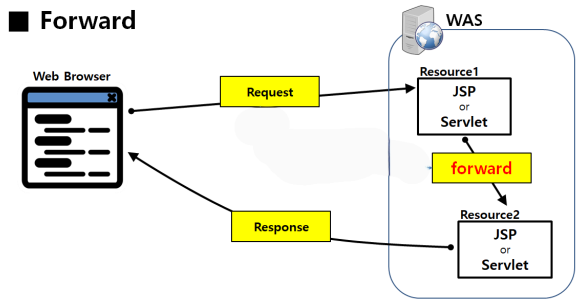
1. Login Status : Remembering that a user is logged in, so they don't have to re-enter their credentials on every page.
2. Authentication Tokens : Storing temporary tokens to verify a user's identity.



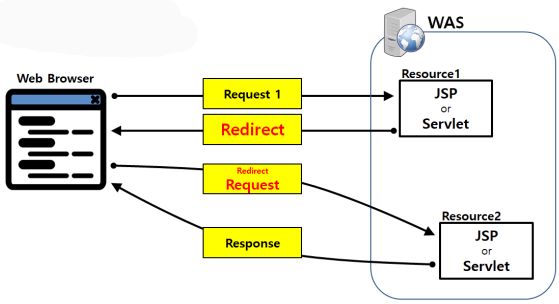
Request Dispatcher & Redirect Life Cycle

Request Dispatcher: an object that receives requests from the client and sends them to any resource (such as a servlet, HTML file, or JSP file) on the server.

Redirect : a type of response sent back to the browser to instruct it to fetch another page



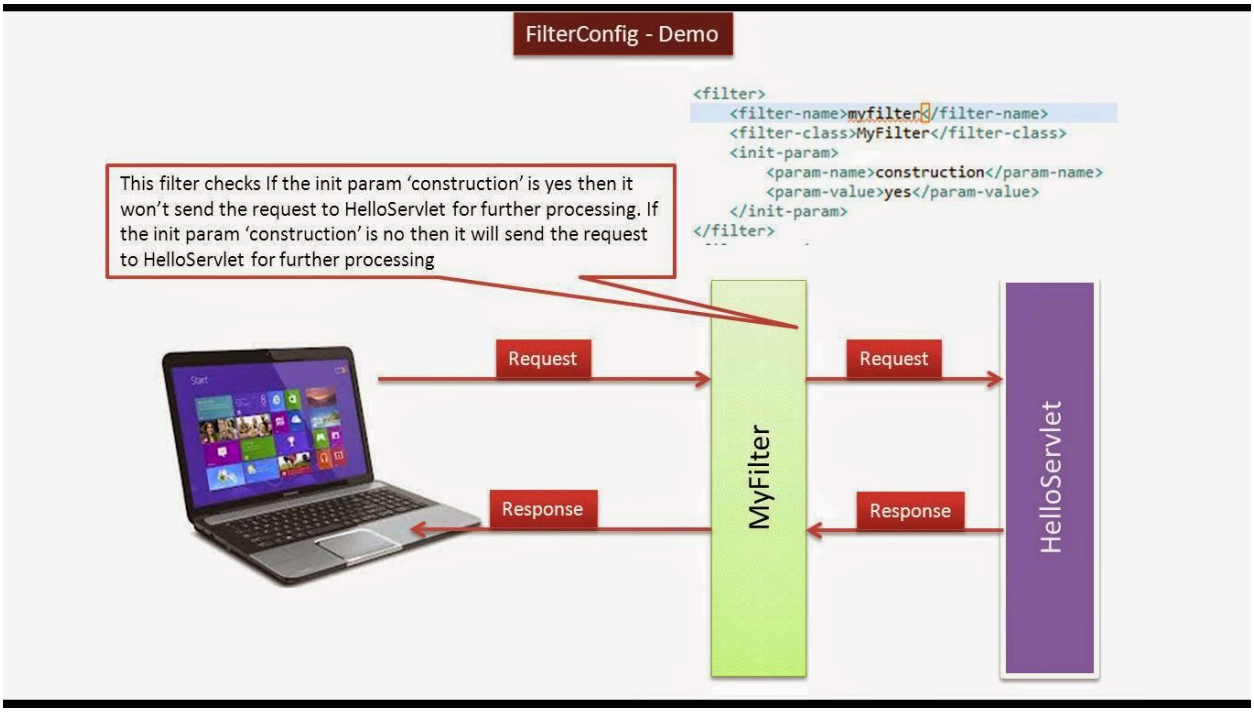
Request Dispatcher



Redirect

Filter :

is used to intercept the client request and response to do some pre-processing. before sending to the client or server



Listener

We know that using
`ServletContext`, we can create an attribute with application scope that all other servlets can access but we can initialize **ServletContext init parameters as String only** in deployment descriptor (web.xml). What if our application is database oriented and we want to set an attribute in ServletContext for Database Connection. If you application has a single entry point (user login), then you can do it in the first servlet request but if we have multiple entry points then doing it everywhere will result in a lot of code redundancy. Also if database is down or not configured properly, we won't know until first client request comes to server. To handle these scenario, servlet API provides Listener interfaces that we can implement and configure to listen to an event and do certain operations. **Event** is occurrence of something, in web application world an event can be
initialization of application, destroying an application, request from client, creating/destroying a session, attribute modification in session etc.
Servlet API provides different types of Listener
interfaces that we can implement and configure in web.xml to process something when a particular event occurs. For example, in above scenario we can create a Listener for the

application startup event to read context init parameters and create a database connection and set it to
context attribute for use by other resources.

Types of Listeners:

1. ServletContext : for events related to the `ServletContext` (the application's global scope).

- `ServletContextListener`: Listens for application startup and shutdown events.
- `ServletContextAttributeListener`: Listens for changes to attributes in the `ServletContext`

1. HttpServletRequest : for events related to `HttpSession` objects (user sessions).

- `HttpSessionListener`: Listens for session creation and destruction events.
- `HttpSessionAttributeListener`: Listens for changes to attributes in a session.
- `HttpSessionActivationListener`: Listens for session activation and passivation events (used in distributed session management).

1. ServletRequest : for events related to `ServletRequest` objects (client requests).

- `ServletRequestListener`: Listens for request start and end events.
- `ServletRequestAttributeListener`: Listens for changes to attributes in a request.