FINALS

SCOPE

**SERVER-SIDE WEB SCRIPTING**

* JAVA (SERVLET AND JSP)
* PHP
* Node.js
* ASP.NET (if there’s still time)

**Java Programming Language Platforms:**

Java Platform, Standard Edition (Java SE)

Java Platform, Micro Edition (Java ME)

Java Platform, Enterprise Edition (Java EE)

* It used in larger scale
* Used in Development of Server
* Used for Java Web

Java EE versions:

Java EE 7 Web

Java EE 6 Web

Java EE 8 Web

**Web server/ Application Server:**

Java Web Application

In Netbeans in creating a web services you can either

* Apache Tomcat 8.0.27.0
* Glassfish server 4.1.1

Web Pages (folder)

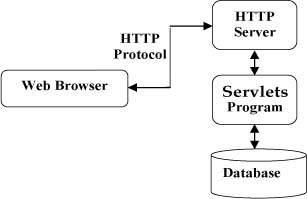
* META-INF
* WEB-INF
* images

**JAVA Servlet**

**What is Java Servlet?**

It is used for building Web-based application which provides a component-based and platform-based method without the performance limitations of CGI programs. It has access to all Java API including the JDBC API to access enterprise database.

It acts as a middle layer between a request coming from a Web browser or other HTTP client and databases or even an application on the HTTP server.

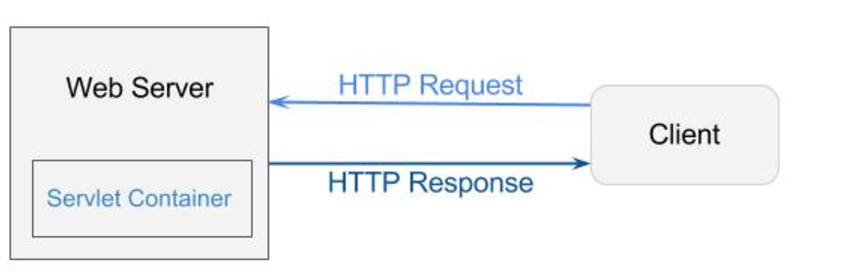
**SERVLET ARCHITECTURE:**

T. (n.d.). Servlets - Environment Setup. Retrieved May 20, 2017, from

https://www.tutorialspoint.com/servlets/servlets-environment-setup.htm

**SERVLET CONTAINER**

It provides the runtime environment for JavaEE application. It is used for dynamically generate web pages in server side. It part of the server side that interacts with the servlet for handling the dynamic web pages form the client.



**-** Serving Binary Content

- Handling Request Parameters (GET)

- Request Dispatching

**Java Servlet - JSP**

@WebServelet container

@WebServlet(name = “Hello Servlet”, urlPatterns = (“/HelloServlet”)

getInputStream() – Servlet Input Stream

getOutputStream() - Servlet Output Stream

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException { }

Portimage = request.getPart(“image”) – get the name

**Request Dispatching**

* **God Class –** a class that does a lot of things
* **Long Method –**
* **Divide & Conquer**

set/ get Atribute

request.setAttribute(“productlist”, productList);

request.setAttribute(“productlist”);

**include component** – either can be **static**, or **dynamic**

**Session Handling**

- “Stateless Http”

- extending Features : **Cookies**

-State of correlated transaction

**Extended Features**

**Cookies**

–small textual information, generated from the web application

- stored by the client in the browser (Cookie Jar)

- identifier

set.cookie.ck1

- non-persistent cookies

(as long as the browser is open)

- can set the time

- can set by HTTPS

HTTP – only can be access by javascript

Scoped objects

- context scopes

**PHP**

- does have many framework

- striplets to JSP

- there is a PHP API on Web services

- Switch in and out blocks in PHP

-Object-Oriented (close with Java)

echo- PHP print statement

Array – is an associative structured

<form action = “x.php” method = GET>

<input type = “text” name = “user”

<input type = “submit”>

</form>

isset function

isset ($\_GET[‘user’]

**Variable – SUPERGLOBALS:**

$GLOBALS

$\_SERVER

$\_GET

$\_POST

$\_FILES

$\_COOKIE

$\_SESSION

$\_REQUEST

$\_ENV

Example:

<?php fpreach ($\_SERVER as $Key = $values) {

echo “<p> $key = $ values </p>”;

} ?>

**HTTP Request Headers**

HTTP\_HOST = Localhost

HTTP\_CONNECTION = Keep-alive

HTTP\_CACHE\_CONTROL =max-age = 0

HTTP\_UPGRADE\_INSECURE\_ReQUESTS = 2

HTTP\_USER\_AGENT= Mozilla/5.0

HTTP\_ACCEPT

HTTP\_OUT

REMOTE\_PORT = 50625

GATEWAY\_INTERFACE = CGI 1.1

SERVER\_PROTOCOL = HTTP 11.1

REQUEST\_METHOD\_GET

QUERY\_STRING=

REQUEST\_URI =

SCRIPT\_NAME

PHP\_SELF

**NODE.JS Serverside**

**MEAN**

**M –** Monggo DB

**E –** Express JS

**A –** Angular framework

**N –** Node.JS

**BJON** – Binary JSON Format

API Node.JS Features

index.js(simple webserver)

var http require (‘http’)

var server = http.createServer();

server.on(‘request handler’);

function handle(request, response) {

response end(hello, nodeJs!!);

}

function handler (request, response) {

var method = request.method;

var url = request.url;

var headers = request.headers;

console.log(‘Request Method: ${method}’);

console.log(‘Request URL: ${url}’);

console.log(‘Request Headers: \n ${JSON.String}’);

* **asynchronous handler –** callback handler
* **synchronous handler**

**SECURITY**

**OWASP Top Ten Project**

**Top Ten Security Vulnerability in the Web.**

**(2017)**

1. Injection
2. Broken Authentication and Session Management
3. Cross-Site Scripting (XSS)
4. Broken Access Control
5. Security Misconfiguration
6. Sensitive Data Exposure
7. Insufficient Attack Protection (IAP)
8. Cross-site Request Forgery (CSRF)
9. Using components with known vulnerabilities
10. Under protected API

**Digital Certificate**

- encrypting of the Website

letsencrypt.org – website for free Digital certificate

**HASH Algorithm**

- function

- One-way function

-intractable

ex.

MD5

SHA1

SHA2

\_SALTED

(‘pwd’ + salt) = HASH