**Web System and Technology (Finals Notes)**

Server-side Web Scripting

- eg Electronic Booking, Email

- more important than client-side

- focusing on:

1. Java (Servlets/JSP (Java Server Program)

2. PHP – developing typical website

3. Node.js

4. ASP. Net

- Java Script on a browser – Sandbox (environment)

- no access to relational database

- serve to protect the user from malicious things

- can’t access file system

- ask permission granted by you

Server-side Data (Technology)

- Java

- Java SE (Standard Edition)

- Java ME (Micro Edition)

- Java EE (Enterprise Edition)

- Server Centric Application

- Java Web Application

- Check the use dedicated folder in NetBeans

- Server:

- Apache Tomcat

- Apache Jerorimo

- GlassFish Server

- IVM = Websphere

- Oracle = Web Logic

- Java EE Version – Java EE 7 Web

- context path: /MyProject

- where going to be staying

- eg: ([www.google.org/MyProject](http://www.google.org/MyProject))

- acknowledge the application

- Frameworks

- Spring Web MVC

- Struts 1.3.10

- Java Server Faces – UI

- Hibernate 4.3.1 – OR

- Private Folder

- Web – INF; Meta – INF

- configuration files

- not accessible by the server

- Source Packages

- Libraries

- JDK Library, Application Server

- Web Archive

- MyProject.war (dist)

- deploy to stand alone application (tomcat)

- Servlets

- java program that sits to the server and run to the URL

- not responsible for html/read text

- read data, put it in array

- Class HttpServlet

- wait for request to be process

- Interface HttpServletResponse/Request

- Setup

- URL Pattern :/Hello-URL Mapping

- eg [www.hi.org/MyProject/Hello](http://www.hi.org/MyProject/Hello)

- Servlet Demo

- Basic Servlet Processing

- response.setHeader(“context-type”, “text/html”)

- Serving Binary Content

- Heading Request Parameters (GET)

- with inputs + get the output

- Handling Request Parameters (POST)

- Service Method

- do xxx method (eg. doGet)

- get, post

- like “public static void”

- HTTPServletRequest (get)

- what request is all about

- method use

- eg. getRequesr URI()

- getRemoteUser()

- getHeader(String name)

- getIntHeader()

- getReader() – return a buffered reader

- Reader – line oriented; String – byte oriented

- HTTPServletReponse (set)

- (text) – version, status code, headers, new line, tail

- eg. setStatus, setHeaders (post)

-

- Multipart config allocation

- gain access to the method (getPart)

- insert, delete, select

- update data

- Request Dispatching

- processing a request and have advantage to the other component to be divided between you and others

- Divide and Conquer Strategy

- Inclusion

- dispatching method: request dispatch

- create a servlet (E) – banner, servlet (F) – footer, servlet (N) – navigator

- call servlet to get the output and get encoded

- set it to the page # (consolidate) of concerns

- Separation of concerns

- dispatching method: forwarding

- query the database and get the data

- return the data

- get attribute: generic object

Query = create – request – execute

Microsoft

- Model (data and processing) View (presentation) Presenter/ Controller (controls)

JSP (Java Server Pages)

- html-static technology

- standard action

- EL Expression

- Jasper

- tomcat

- <%-- (info) --%> - JSP Comment

- <%@ page(attribute-value; attribute-value) %> - JSP Directives

- page directives; like <meta -> & import statement in java

- error page/ IsErrorPage(=”True”)

- data not accessible; exception to be thrown

- <% include (info) %> - include directives

- Template Text

- <html> </html> - html text

- JSON Text

- JSP Standard Actions

- <JSP: include page=”(info)”/>

- <JSP: forward page=”(info)”/>

- <JSP: element page=(info)> </JSP: element>

- <JSP: attribute page = (info)/>

- <JSP: body page=”(info)”> Go </JSP:body>

- Using JSP Document (XML)

- <JSP: scriple page=”(info)”/>

- <JSP: useBean page=”(info)”/>

- <JSP: plugins page=”(info)”/>

- <JSP: params page=”(info)”/>

- <JSP: param page=”(info)”/>

- <JSP: callback page=”(info)”/>

- Expression language (EL) Expression (starts with $)

- <p> hi $ (panam.name)! </p>

- <p> Your browser is ${header[‘User-Agent’]} </p>

- $ {requestScope.product.prodid}

- Scriplet Scriplet

- <% int->(100p)->%>

- Scriplet Expressions (<%=)

- <%= (info)%>

- Scriplet Declaration (<%!)

- <%! (int->void) %>

- JSTL (JavaServer Pages Standard Tag Library)

- <jsp: include = “(info)”/>

- <%@taglib uri = (info)%>

- uri (include) – prefix = “ “ < to access

- core – loops

- formatting – date, number

- simple functions – upper case, lower case, trim

- sql

- xml

- <sql: (info)/> - param, query, dateParam, setDataSource

- ${fn – function; fmt – format} <- always put this to become a function

Session Handling

- stateless communication – simple to implement

- wait for request then send the response then stop

- (apache server) administrative purpose – with access log

- solutions:

- modify http but standard

- extension called cookie

- small text file

- associate with url of the app

- to identify the client

Cookie (session handling)

- not persistent

- exist when browser is up; once close the cookie will be gone

- known for a time-out mechanism

Https – to be secure

Http only – cannot be access by javascript

Install Wamp

- php module (also installed)

- <? php (echo “hi, world!”;) ?> - markup for php script

- look run and whatever output will replace the code itself

- <? php if(true)?> (info) <?php}?>

- general purpose programming language

- can use html, css (read)

Past: Procedural Language

- class Product{ //declaration of a class

private $prodId, $price; //instance variable, starts with $

public function \_construct ($prodId, $price) { //constructor

- current instance <- $this -> prodId = $prodId;

public function getProductId(){ //get/set/add

return $this-> prodID;}

$this - > product = []; //empty array of product

Remove item($index){ // remove a specifi

