SIMATS

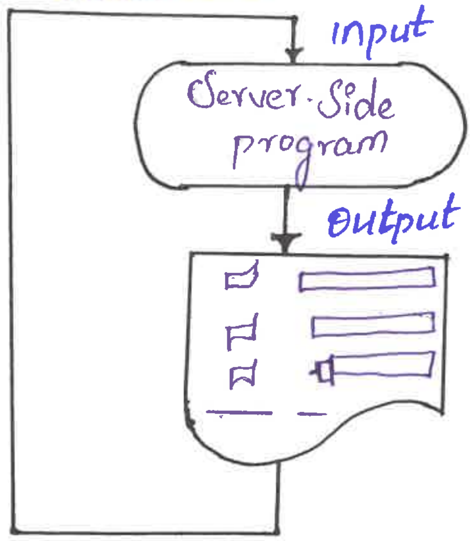
School of Engineering

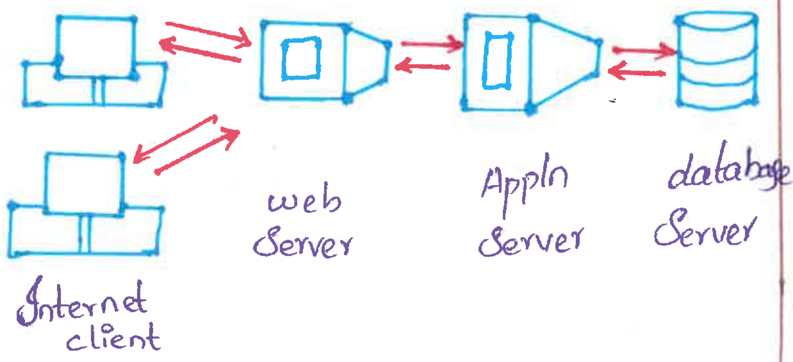
Internet Programming

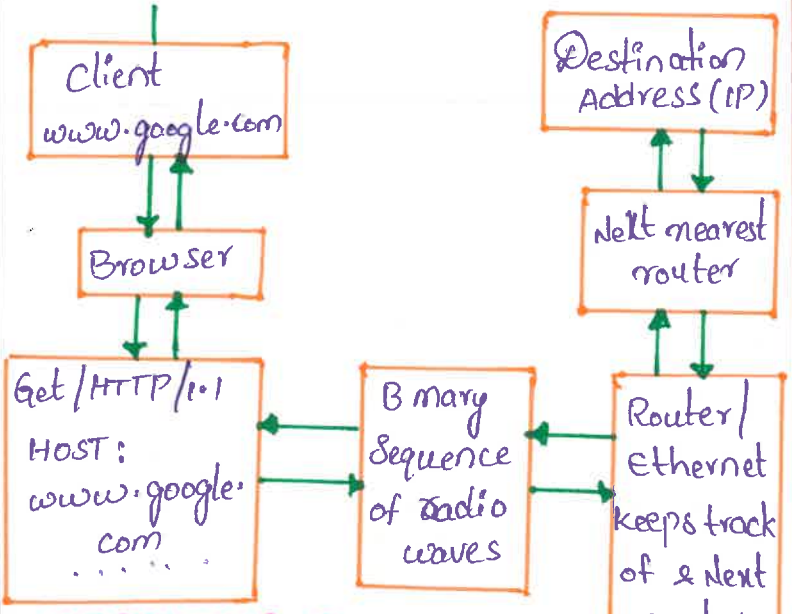
Computer Science and Engineering

Saveetha Institute of Medical And Technical Sciences,Chennai.

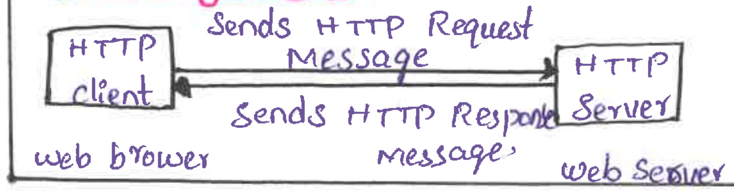
Web Essentials



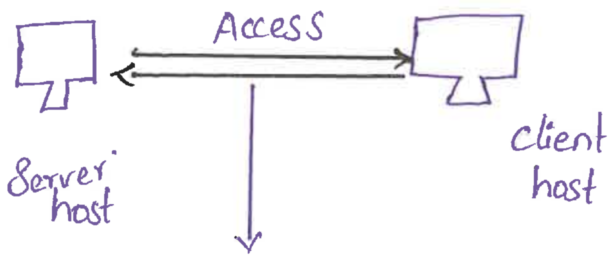




HTTP Protocol

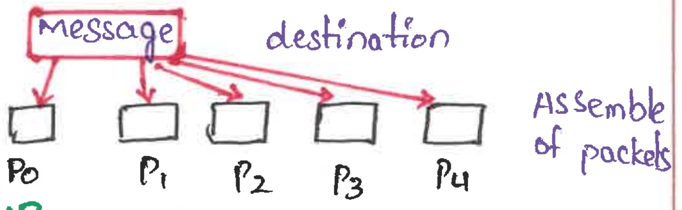


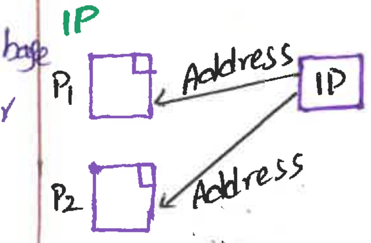
Internet and web programming



interaction via protocols

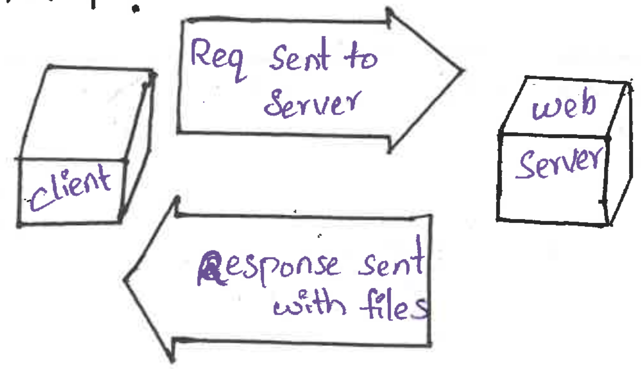
- Client side programming
- Server side programming

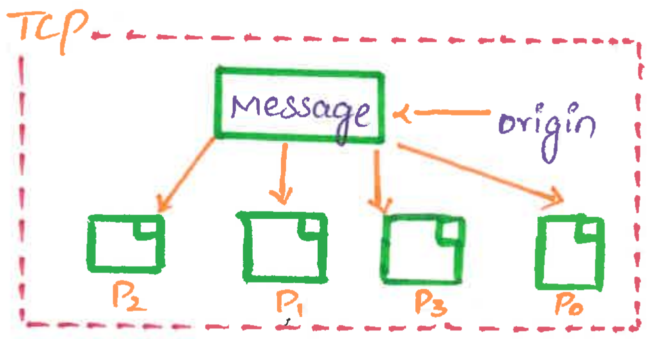




→ web browser
→ web Server
→ prolocols

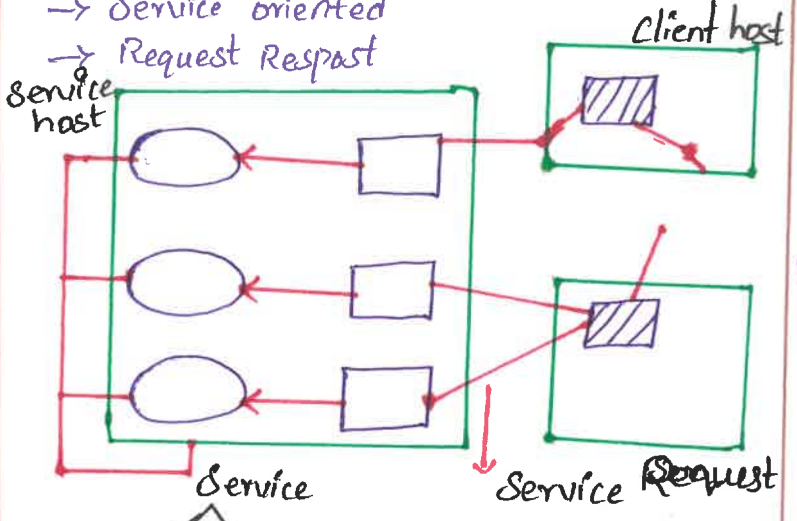
HTTP:

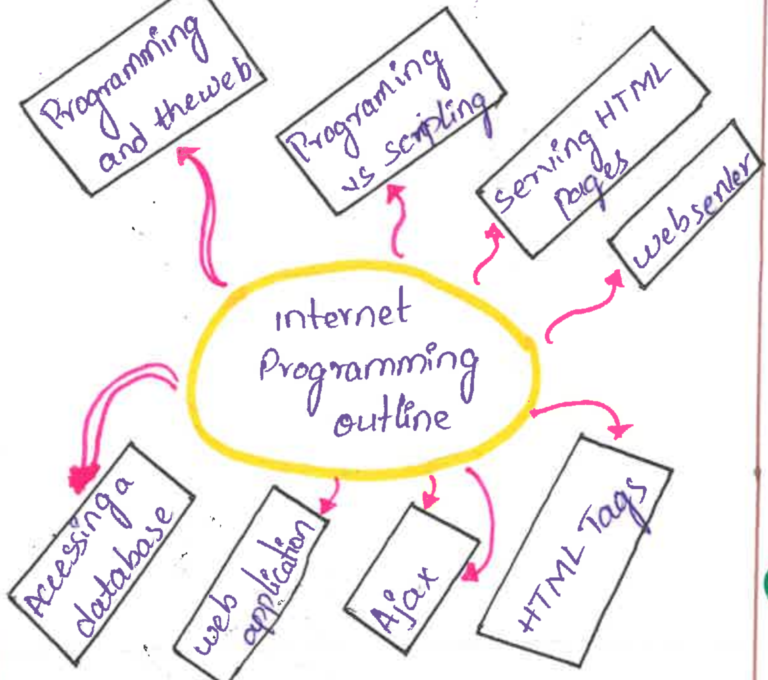




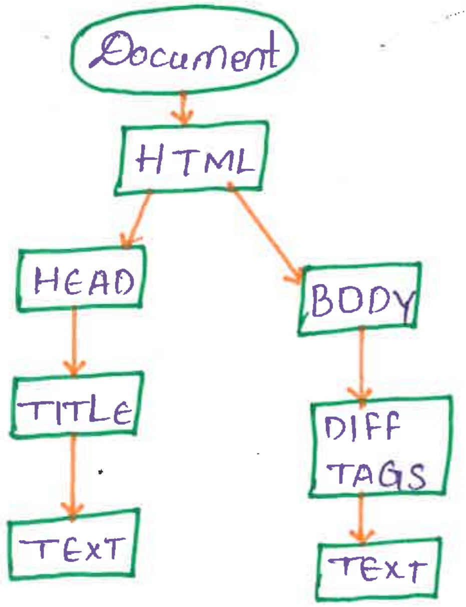
Dissamble of packets

→ Distributed Computing
→ Service oriented
→ Request Respast

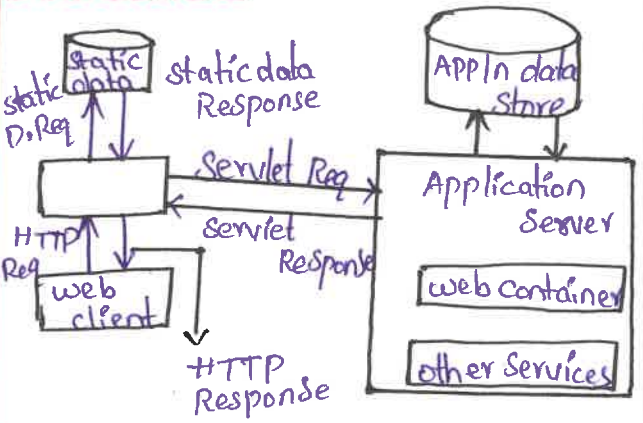




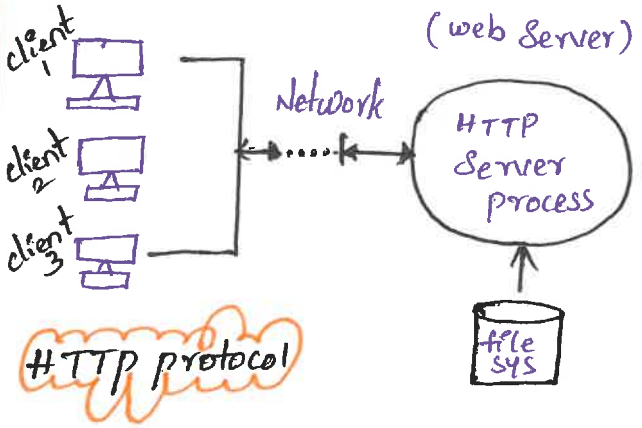
internet and web programming



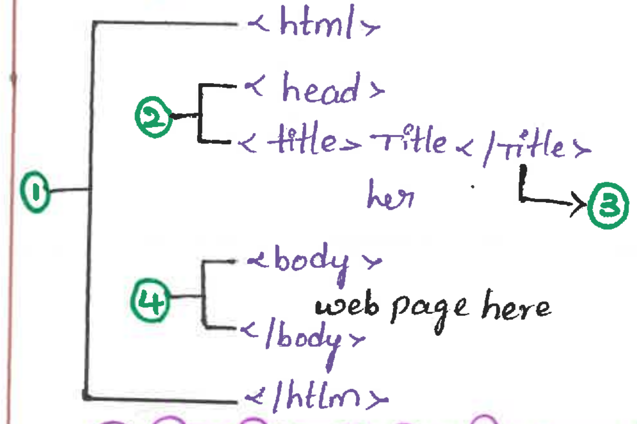
web Browser



web Server

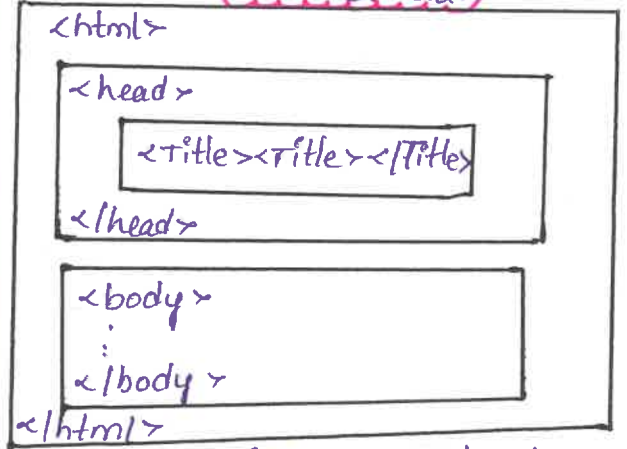


HTTP protocol



Fundamentals of internet programming

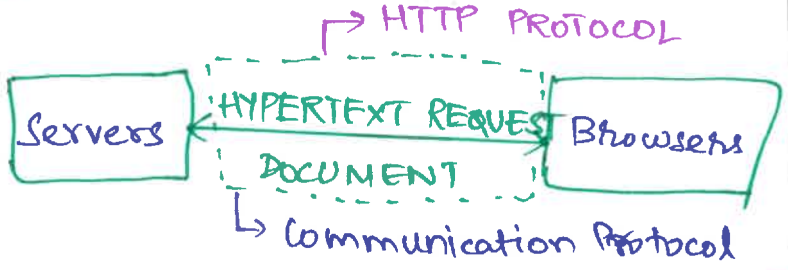
HTML Structure

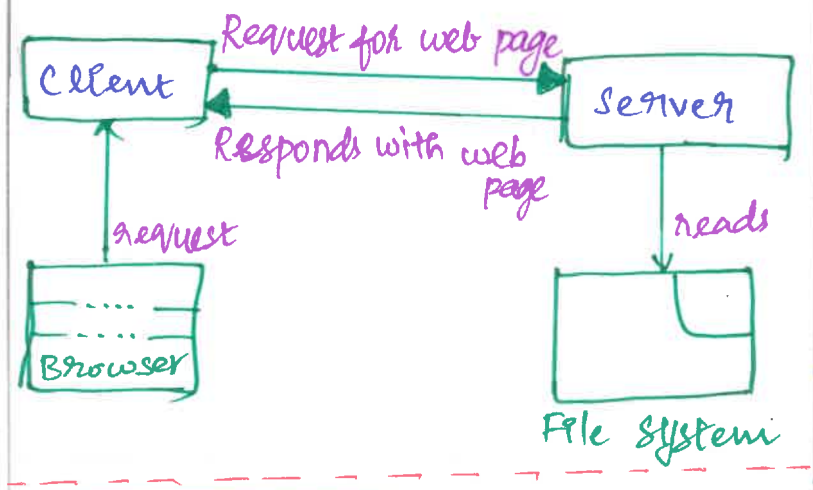


internet promming HTML Structure

HTTP REQUEST/RESPONSE

HYPERTEXT TRANSFER PROTOCOL





HTTP TRANSACTIONS:

Client

Server

Initiates → accepts connection

Request Message → recieves reqvest

Generates response → closes

Recieves response → Repeat

HTTP MESSAGES

- Request
- Response

Request Line

Header

Request / Response Header

Entity header

Body

REQUEST HEADER

- Header - Description
- From - Email address
- User Agent - Client S/W
- Accept File - File Types
- Accept Encoding - Compression
- Accept Language - Language
- If Modified
- Content Length

RESPONSE HEADER

- Header
- Server
- Date
- Last-Modified
- Expires
- Location
- Pragma
- MIME-version
- Link
- Content-Length
- Allowed

HTTP METHOD

HTTP METHOD

OPTIONS	GET	HEAD	POST	PUT	MOVE	DELETE

Message Body

- content type
- Length
- content

HTTP

Security
- Personal
- Information
- DNS Spoofing
- Location Header
- Proxies
- Caching

Caching

client and server

caching

Parameters

Versions
- HTTP 1.0
- HTTP 1.1

Status Codes

1xx - Informational

2xx - Success

3xx - Redirection

4xx - Client Error

5xx - Server Error

About - Appln Level
- Connections
- Stateless

Header
- General
- Request
- Response
- Entry

Request
- Request line
- <n> header
- CRLF
- Message body

Request Method
- Get
- Head
- Post
- Put
- Delete
- Connect
- Options
- Trace

Responses - Status line
- 0-<n>
- header fields
- CRLF
- Optional
- Message body
- CRLF - Indicating end of header fields

HTML

- → Overview
- → Basic tags
- → Elements
- → Attributes
- → formatting
- → phrase tag
- → Meta tag
- → Comments
- → images

→ Features

Documents structure

<html>

<head>

head related data tags

</head>

<body>

Document body related tags

</body>

</html>

DECLARATION

<! DOC TYPE html>

Basic tags

Heading tag → six levels

<h6>	<h5>	<h4>	<h3>	<h2>	<h1>

Paragraph tag	<P>
Line break	<br>
Centering	<center>
horizontal line	<he>
Preserving format	<pre>
abbreviation	<abbr>
address element	<address>
Code	<code>
resource reference	<link>
Non breaking Space	<nbsp>

ATTRIBUTES

id	Title	class	style

↓

Core attributes

Internationalized Attribute

dir	lang	XML lang

Generic Attribute

background (URL)	bgcolor	class	
align	valign	id	width height

Formatting Tags

Bold	<b>------</b>
Italic	<i>------</i>
underlined	<u>----</u>
Strike	<strike>------</strike>
Superscript	<Sup>-----</Sub>
Subscript	<Sub>------<Sub>
Delet	<del>------</del>
Grouping	<div>------</div>
Content	<span>------</span>

Phrase Tag

Emphasized	<em>----</em>
Marked text	<mark>-----</mark>
Strong text	<strong>----</strong>

Links + formatting

Anchor text <ahref=" ">... </a>

user → Specific document

<a name="name">---- </a>

user → div element

<a href ="# name>.... </a>

Outgoing mail → e-mail

<a href =" mail to :">----- </a>

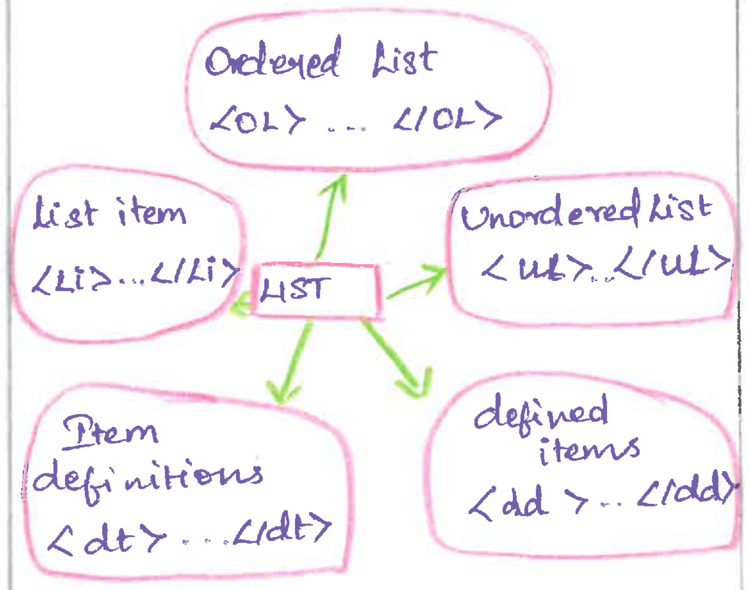
Phone Number clickable

<a href = "tel ://####">---- </a>

image (+ formatting)

image	<img/>
URL	src = "url"
image content	alt = "text"
image height	height=" "
image width	width = " "
alignment	align = " "
Spacing	[hspace = " " vspace = " "
Border style	Border = " "

HTML 5



Form (Formatting & attributes)

Create form

<form> ... </form>

attributes

- → action = "URL"
- → Method = " " GET / POST
- → autocomplete = " " ON/OFF
- → accept charset = " "
- → target

_blank	_self	_Parent	_top

Identifies group of fields

<fieldset> ... </fieldset>

< label > .... </ label>

< legend > ... </legend>

⇒ Fieldset tag :

★ Makes a group of related elements in the form .

★ Creates box over the elements.

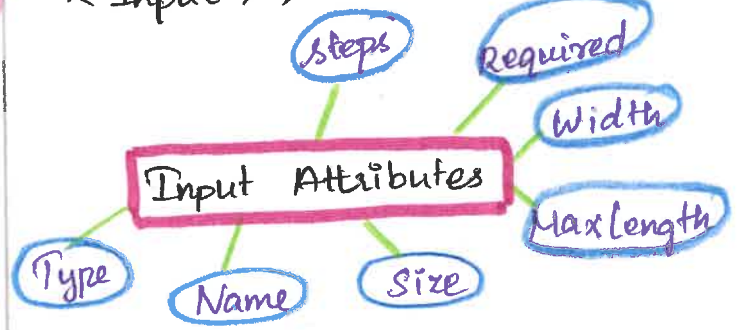
Legend tag :

★ The legend elements are parent elements .

★ Used to define the title for the child's contents.

INPUT - FORM

< Input />

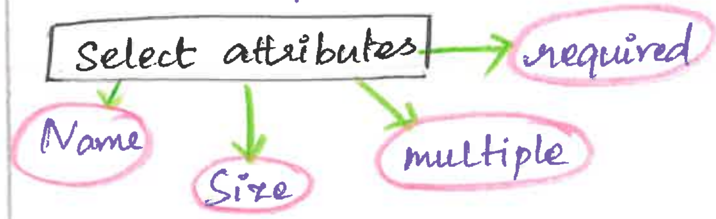


< text area >... </ textarea>

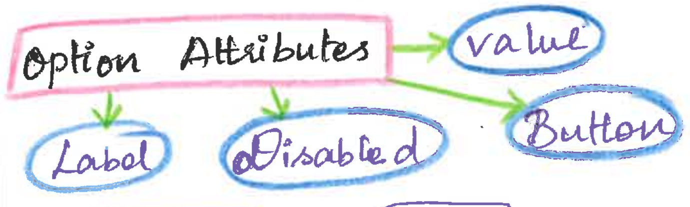
Text input

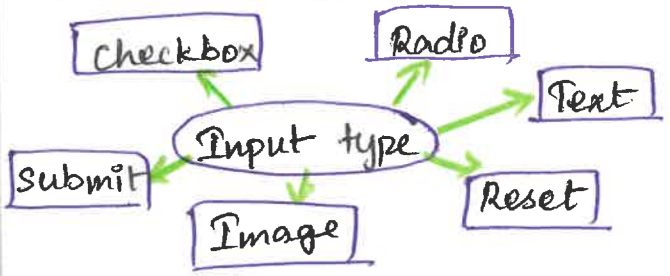
< select > ... </select>

Dropdown box



Grouping options :





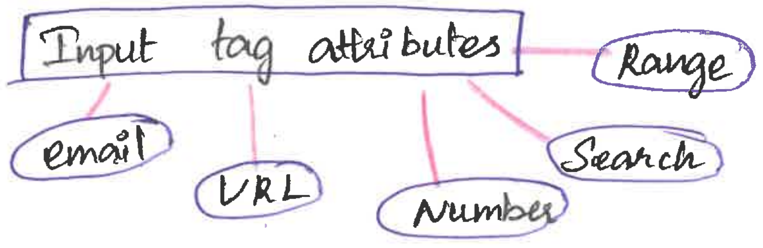


TABLE :

Create table :

< table > ... </ table>

Set each row ⇒ <tr> ... </ tr>

Set each cell ⇒ <td> ... </td>

Set table heading ⇒ <th> ... </th>

Table attributes :

<table border = ? > ⇒ cell width

<table cellspacing = ? > ⇒ space b/w cells

<table cell pading = ? > ⇒ space b/w borders

<table width = ? > ⇒ in pixels

< tr align = ? > ⇒ alignment (left/ right/ center)

<tr valign = ? > ⇒ vertical alignment (top / middle / bottom)

<td rowspan = ? > ⇒ no.of rows merge

<td colspan = ? > ⇒ colgumn merge

<td rowspan = ? > ⇒ prevent line within cells

<caption > ... </caption> ⇒ Description of tables

<thead > ... </ thead>
⇒ header labels the column

< tfoot > ... </tfoot>
⇒ footer content

<colgroup> ... </colgroup>
⇒ Group columns .

<object > ... </object>

file type - embed (audio , video , pdf)

width = " " width of object

usemap = " " client side image

height = " " height of objects

type = " " type of media .

FRAMES :

<iframe> ... </iframe>

an inline frame document .

Attributes :

src = " " URL of the object

width = " " width of frame

name = " " name of frame

srcdoc = " " html content within the frame .

<embed > ... </embed>

Container - another - external application .

<param /> adds extra parameter

<header > ... </header >

header block for document

<main > ... </main>

main content of a document

<aside> ... </aside>

content contained in sidebar

<article> ... </article>

identifiers .

TOPIC:

CSS - Cascadina Style Sheet

- * Inline
- * Internal / Embedded
- * External

INLINE:

```
<P style = "color:
    #00900;
 font-size : 50px;">
    Hello </p>
```

INTERNAL:

```
<html> <head>
<style>
.main {
   text-align: center;
 }
.h {
font-style : bold;
font-size : 20px;
} </style> </head> </html>
```

External CSS

Sample css

```
.body {
   background-color : blue;
   }
.main {
   text-align : center ;}
```

```
<html>
<head>
<link rel = "stylesheet"
 href = "sample.css"/>
</head>
<body> Hello </body>
</html>
```

BACKGROUND

- → Background - Image
- → Background - Position
- → Background - Size
- → Background - repeat
- → Background - attachment
- → Background - origin
- → Background - clip
- → Background - color

BACKGROUND - IMAGE

Url / gradients / none

BACKGROUND - SIZE

auto / cover / contain

BAGRGROUND - REPEAT

repeat / repeat-x / repeat-Y / NO-repeat

BACKGROUND - ATTACHMENT

Scroll / fixed / local

BACKGROUND - ORIGIN

Border-box / Padding-box / content-box

BACKGROUND - CLIP

border-box / Paddling

BORDER

Border - width

thin / Medium / thick length

Border - style

- → none / hidden / dotted
- → dashed / solid / double
- → groove / ridge / inset / outset

Border - colour : color

Border - left : width

Border - Bottom : width

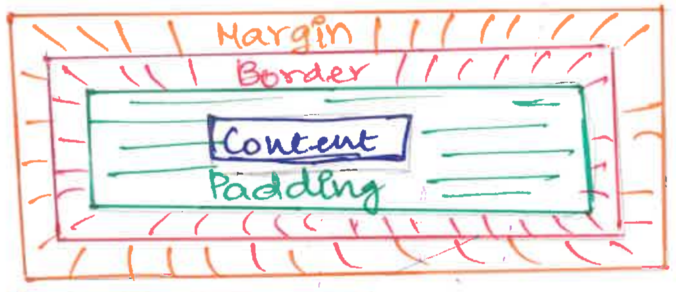
Border - Break : width

CSS - Margin :

Margin - top / Margin - bottom

Margin - right / Margin - left

CSS Box Model



CSS outline

- → outline - style
- → outline - color
- → outline - width
- → outline - offset

CSS Text :

- → Text Colour
- → Text allignment
- → Text decoration
- → Text transformation
- → Text spaing
- → Text shadon

CSS website layout

- → 1- column
- → 2- column
- → 3- column

CSS Math Functions

- → calc ()
- → max ()
- → min ()

JavaSCRIPT

- Istring
  - Property
    - Constructor
    - Length
    - Prototype
  - Method
    - Chart
    - Charcode
    - fromcharcode
    - repeat
    - to Locoer case
    - to strong
    - to uppercasc
  - String HTML Methods
    - Fixed
    - font colou
    - font Size
    - Italics
    - Links

- Number
  - Property
    - Constructor Max value
    - Negative ifinity
    - is finite
    - is Integer
  - Method
    - to expontail
    - to fixed
    - to String

Js Operators

- Arithmetic
  - + Addition
  - - subraction
  - / Division
  - x Multiplication
- Assegment - =,+=,-=+=,/=
  - %=
- String → +, +=
- Comparision ==,===,!=,!==,
  - >,<,>=,<=
- Conditional → Variable name
  - Condition.
  - Value
- Logical → && , ||, !
- Bitwise → &, |, ~, ^, <<, >>
- Type of
- delete
- in
- inslace of
- void.

Js Math

- Properties
  - E, LN2, LN10,
  - LOG2E, LOG10E.
  - SQRT - 2, SQRT2
- Methods
  - abs(x), acos (x)
  - asin(x), atan (x)
  - cos (x), exp(x)
  - random ( ), round
  - sqrt (x), tan (x)

Js BOOLEAN

- Boolean ( ) function
- Properties : constructor
- Methods : to Stringes, Value

JS ARRAY

Method: Concat( ) copywithro

- every( ) fill( ) filter( )
- find ( ), find index
- for each ( ), index of ( )
- Map ( ), pop ( ), push ( )
- reduce ( ), reduce Right( )
- sort ( ) splice ( )
- to string ( ) Unsift

JS Date

Method: get Date( ) get Day

- get full Year ( ) get Hours ( )
- get Minutes ( ) - get Month
- get Time( ) get UTc full year
- Set Hours ( ) - Set Milliseconds( )
- Set Minining( ) → set Time( )
- To Date string( ) → to GMTs string
- Isos string ( ) toJson ( )
- To UTo string( ) - UTc ( )

Js conversion

- Method
  - Data types
    - String
    - Number
    - Boolen
    - objects
    - function
  - Objects Types
    - Object
    - Date
    - Array
  - connot . contain vaves
    - Null
    - Undefina

Js ERROR

Property : Name, Massage

JavaSCRIPT Basic events

- click (onclic)
- Double click (ondclick)
- Right click (oncontext)
- Keyprocess
  - onkeypress
- key up
  - onckeyup
- Mouse Hover (onmouse onter)
- Mouse Out : onmouse Out
- mouse Down (onmousedown)
- Mouse up (onmouseUp)
- load : (onload)
- onload : (ounlod)
- Scroll (onscroll)

Eg:
Function add an event

MyMC.addEventListener (MouseEvent.CLICK, clickHandler X);

JS Reg Exp

- Syntax : /pattern/modifiers :
- Brackets [ ]
- Meta character /w \W \d \D \s \S
- Qualifiers : n+ n* n? N{x}
- Properties : Constructor global ignore close last index multi line source
- Method : * Compile( )
  - * Exe ( )
  - * text ( )
  - * to string ( )

DHTM : Dynamic Hyper Text Markup Language.

* Used to create interactive and animated web pages that are generated in real time.

* When Dynamic web pages are accessed, the code within the page is analyzed on the webserver

* Then, the resulting HTML is sent to the client's web browser.

Advantages :

⇒ Can make dynamic documents

⇒ Designer can control

⇒ Can position any element in window.

Components of DHTML :

⇒ Conventional HTML

⇒ Scripts - small programs - Javascript

⇒ DOM

⇒ CSS

⇒ HTML 4.0

Features of DHTML

⇒ Create web dynamically

⇒ Dynamic style

⇒ Feature of code reusability

⇒ Browsers for data binding

⇒ Easily change tags.

⇒ Create dynamic fonts for web-sites and web pages.

DHTML JavaScript :

Document.write() method.

Document.write() method of Javascript writes the output to a web page.

JavaScript and HTML event :

Use alert() method in which we type the data() function.

JavaScript and HTML DOM :

We type a code of javascript in the <body> tag.

CSS with Javascript in DHTML

Version 4 of HTML, Javascript code can change style i.e, colour, size, & face of an HTML document.

document.getElementById("demo")

Style.property = new.value

DHML CSS.

* CSS with DHTML ; Javascript and HTML DOM

* this.style.property = new.style

* document.getElementById (id)

* style.property = new_style .

eg: this.style.color = 'blue'

DHTML Events :

1. Click a button
2. Submitting a form
3. An image loading or a web page loading, etc..

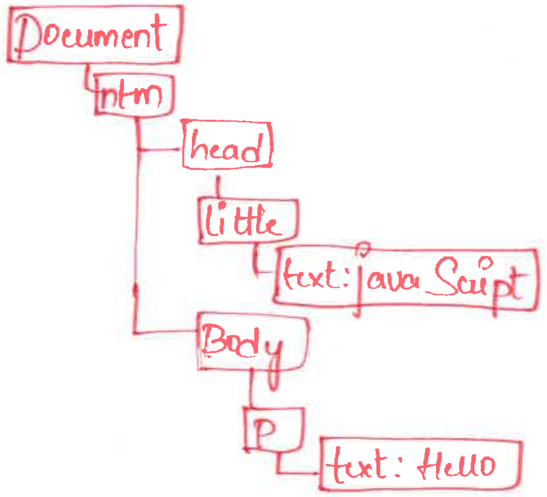
Events :

1. Onabort ←
2. Onblur ←
3. Onchange ←
4. Onclick ←
5. Ondblclick ←
6. On focus ←
7. onkeydown ←
8. Onkeypress ←
9. On key up ←
10. onload ←
11. on mouse down ←
12. on mouse move ←
13. on mouse over ←
14. on mouse out ←
15. on mouse up ←
16. on reset. ←

Java Script DOM

(DOM) Shows how to manipulate DOM elements effectively.

SECTION-1 (DOM is an apply. programming. interfeunce API for manipulating HTML Documents



SECTION-2 Selecting Elements

> get Element By Id() Select By an Element by I

> get Element By Name Select an Element by Name.

> get Element By Tag Name() Select an Element By Tag Name.

> Get Element By Class Name() By one. a more Classes

> Query Select a()

Section 3: Transvaing Elements

> Get the Parent Element - get the parent node

> Get Child Elements Children of the node.

> Get Sibling of an Element - Siblings of an Element.

Section: 4: Manipulating Elements

> Create Element()

> append child()

> Text Content:

> Inner HTML

> Inner HTML vs Create Element

> Document Dagment

> Insert Before()

> Insert After Helper function

> append()

> prepend()

> Insert adjacent HTML()

> Replace Child

> Clone node()

> Remove child() Removes child node.

Section 5: Waking with Attributes

> HTML Attributes & DOM Objects properties

> Set Attribute() Set Value for Attribute

> Get Attribute() Get Value of Specified Attributes

> Remove Attribute

> Has Attribute.

Section: 6 Manipulating Element's Styles

> Style property - Set inline Styles of an Element

> Get Computed Style - Return Computed Style

> Class Name property - List of Space Separted

> Class list property - CSS Class manipulate

> Elements width & Height - width & Height

Section: 7 Waking with Events

> Java Script Events

> Handling Events

> Page load Events

> DOM Content loaded.

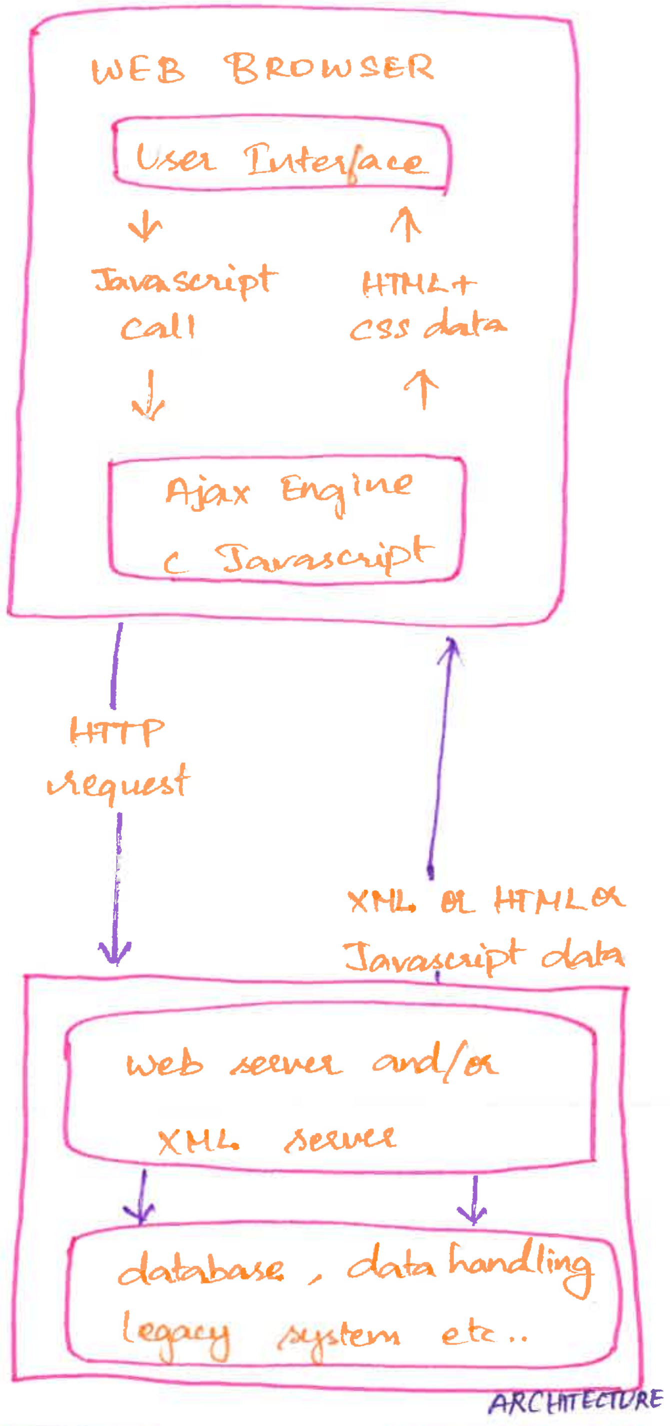
> Mouse Event

> Key Board Event.

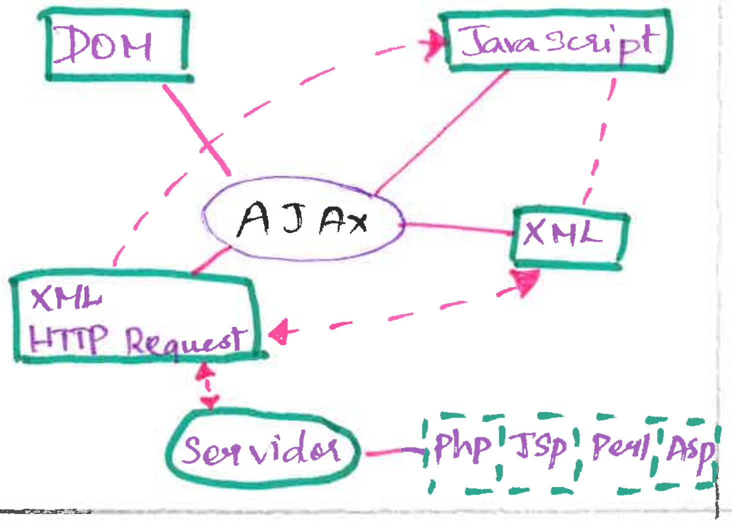
> Scroll Events

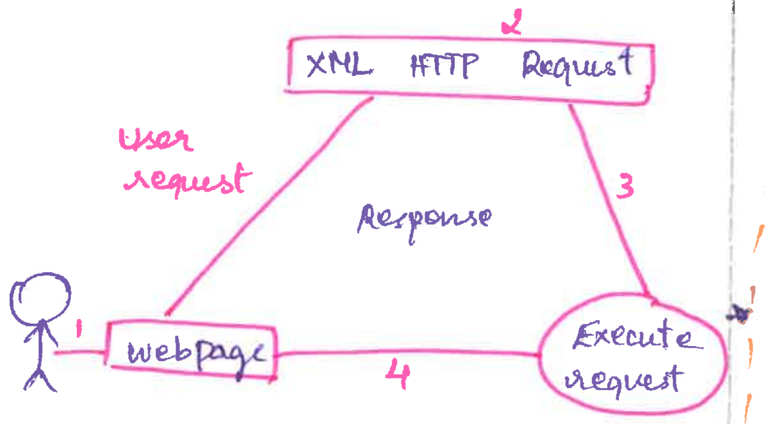
> Haschange Event

AJAX :

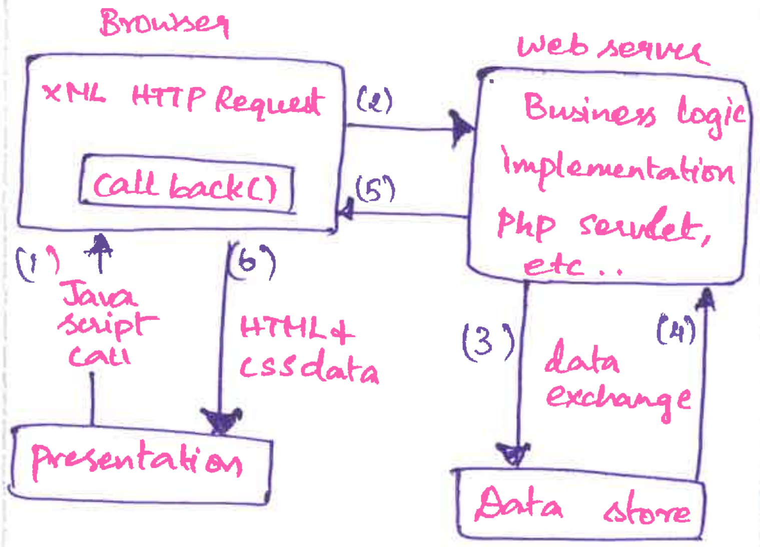

WEB BROWSER
User Interface
Javascript call
HTML+ css data
Ajax Engine c Javascript
HTTP request
XML or HTML or Javascript data
web server and/or XML server
database, data handling legacy system etc..

ARCHITECTURE

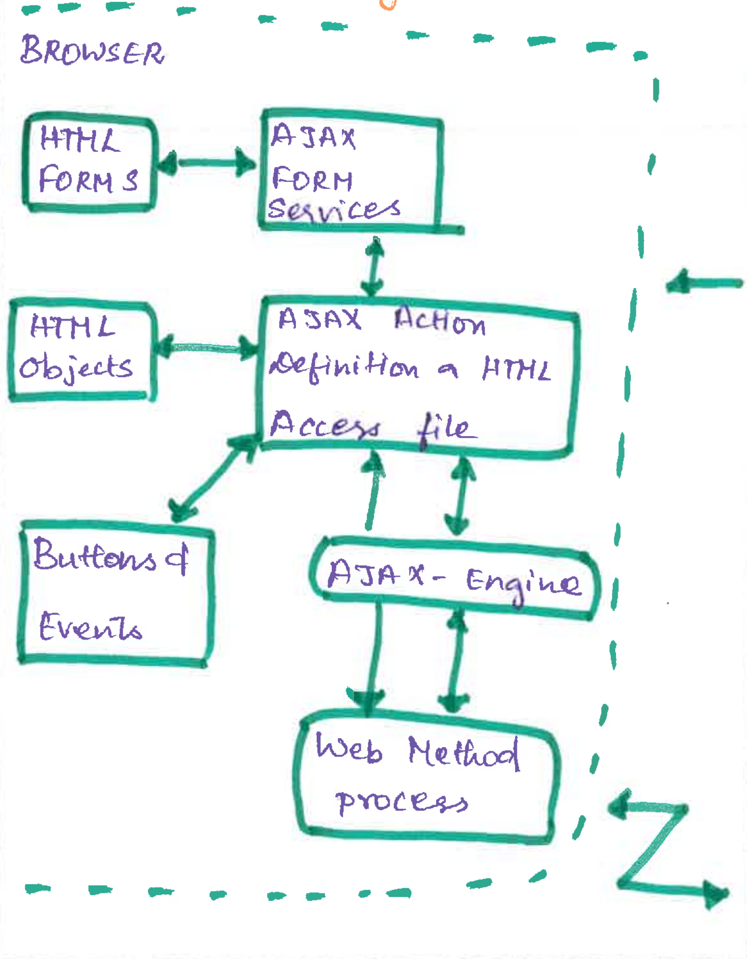




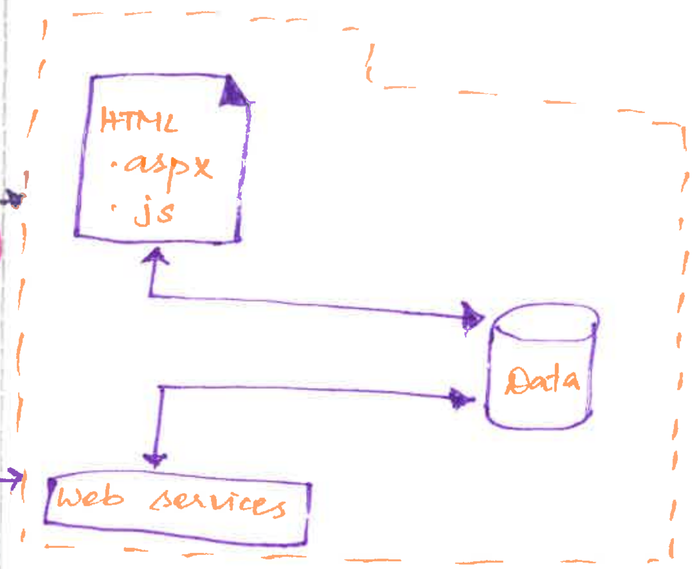
Ajax Request Response cycle

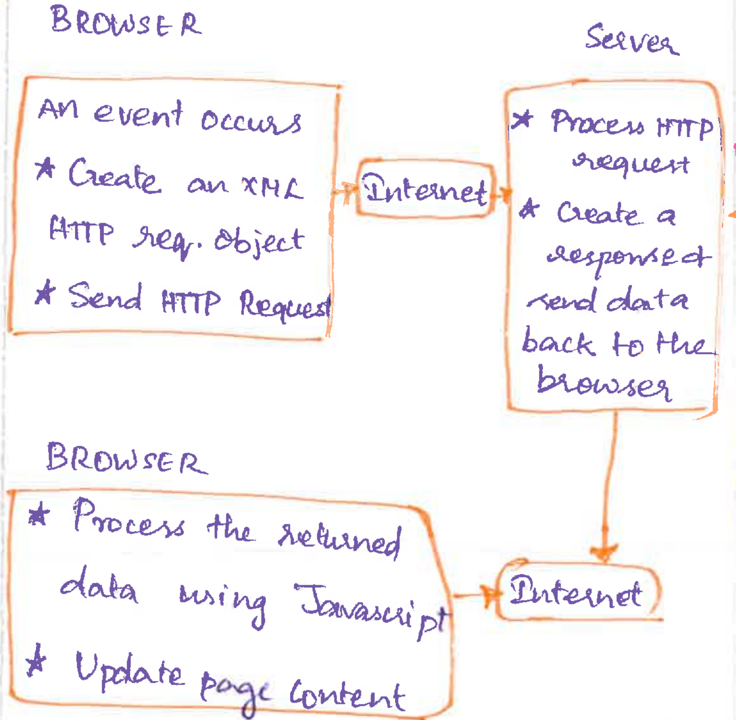


Ajax Security Issues :

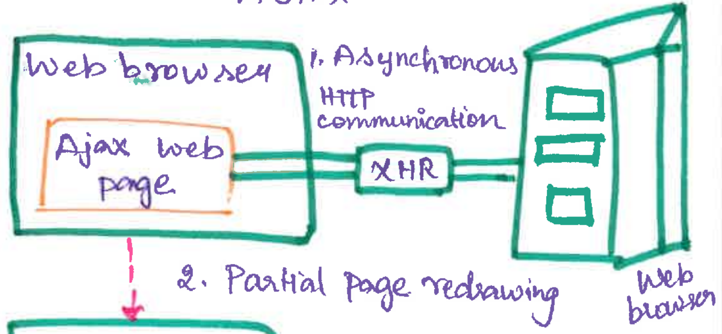


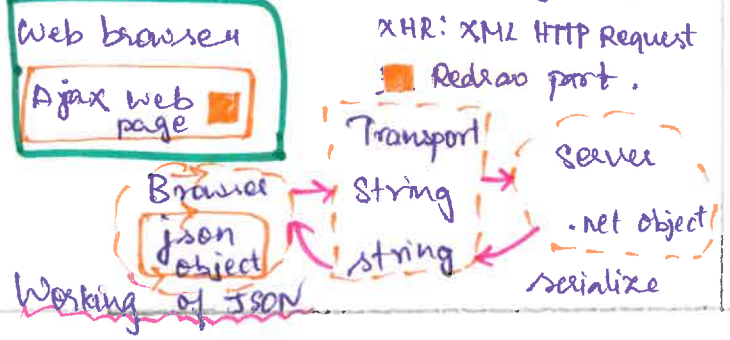
SERVER S



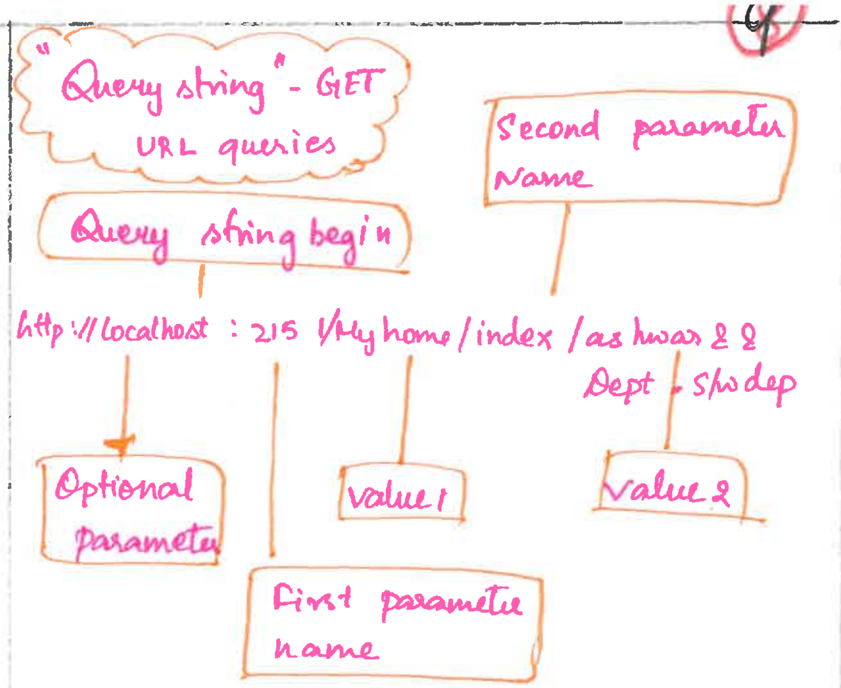

BROWSER
An event occurs
* Create an XML HTTP req. object
* Send HTTP Request
Internet
Server
* Process HTTP request
* Create a response & send data back to the browser
BROWSER
* Process the returned data using Javascript
* Update page content
Internet

PHP - AJAX .

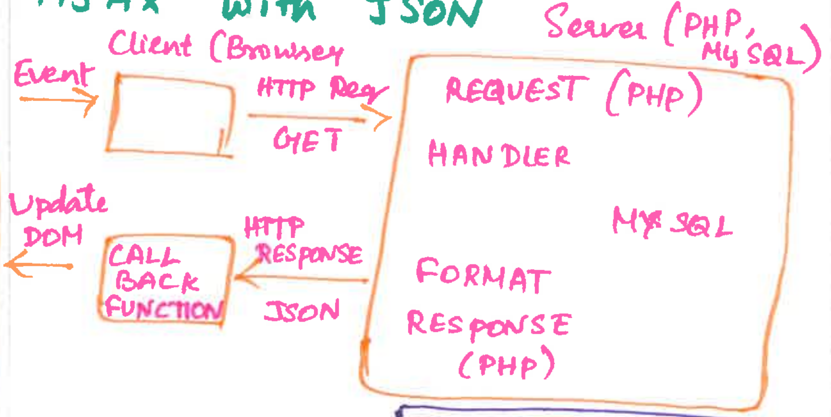


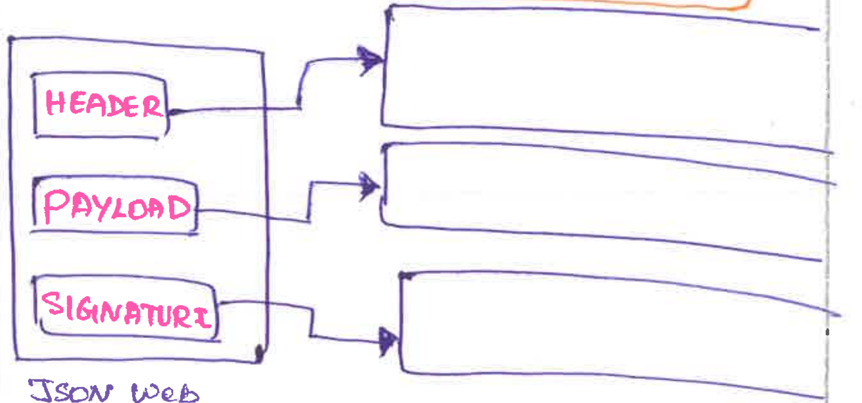


Working of JSON


"Query string" - GET URL queries
Second parameter Name
Query string begin
http://localhost : 215 1/My home/index /as hwas 2 & Dept : sh dep
Optional parameter
value 1
value 2
First parameter name

AJAX with JSON



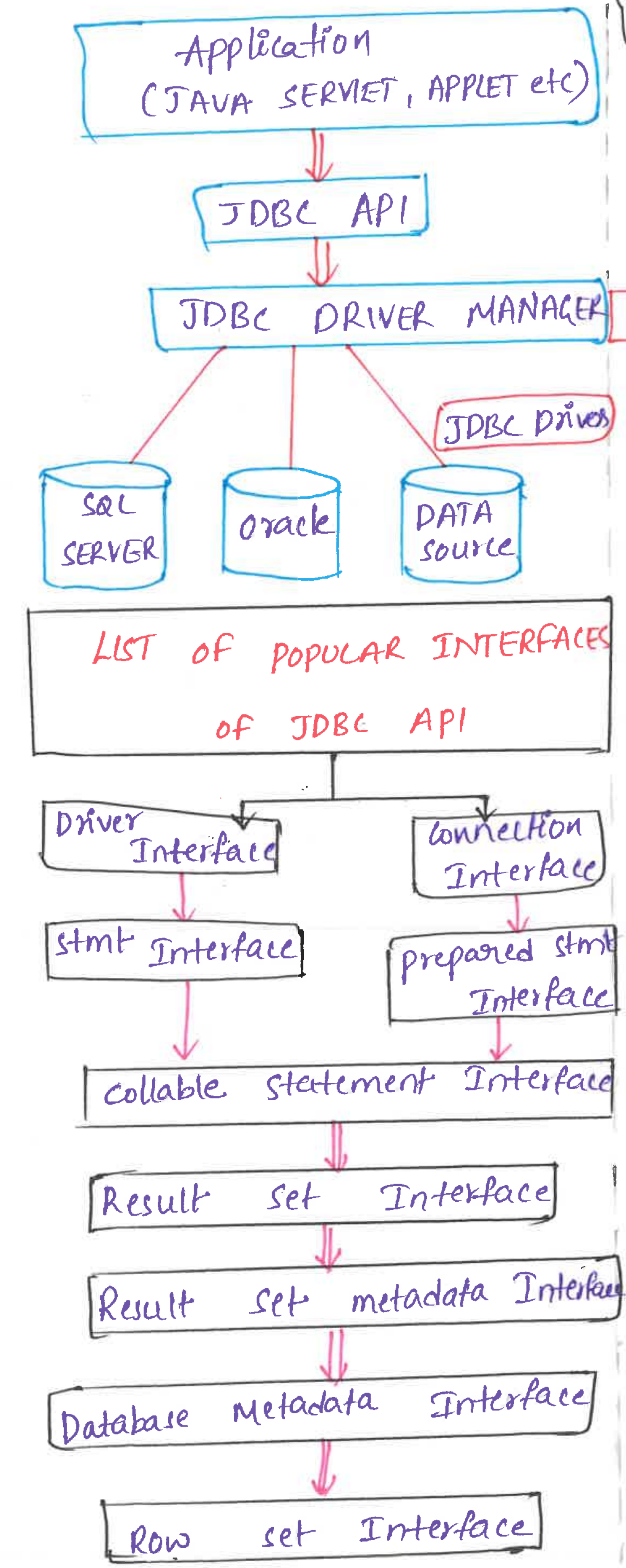


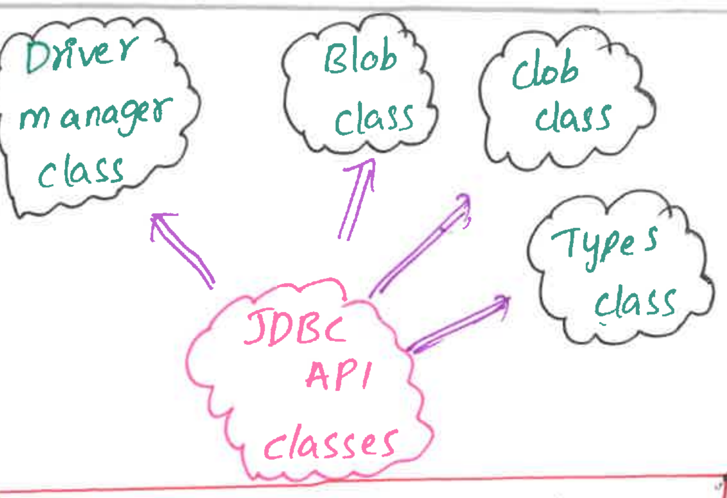
JSON Web token.

AJAX XML HTTP Request

```
$ ajax({
type : "POST",
url : "Default.aspx/Method Name";
data : "{param : "value"}";
Content type "application/json charset-utf8"
data type "json",
success : on success
});
```

JDBC - JAVA DATABASE CONNECTIVITY





JAVA DATABASE CONNECTIVITY

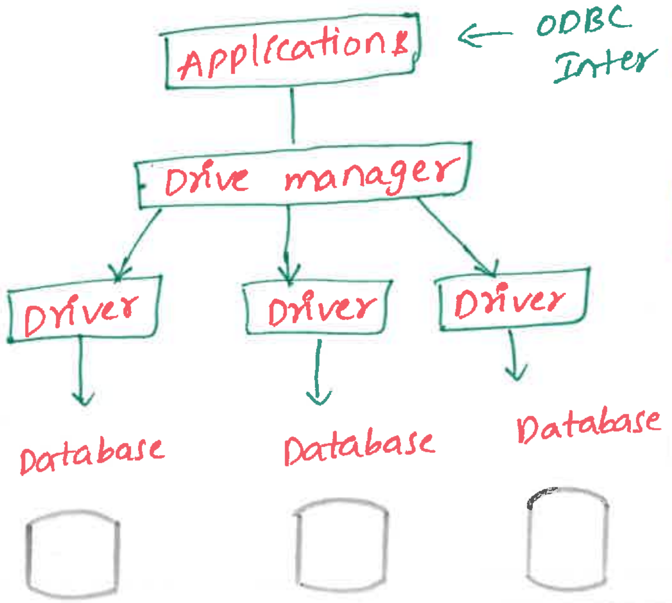
1 ⟺ Register Driver

2 ⟺ Get connection

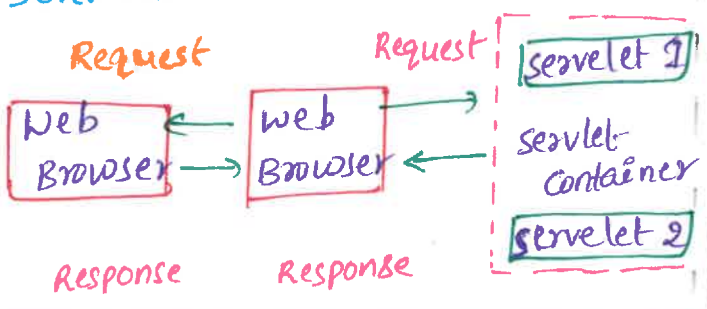
3 ⟺ create statement

4 ⟺ Execute query

5 ⟺ close connection



SERVLETS



⟹ Creating a Directory structure

⟹ Creating a seavlet

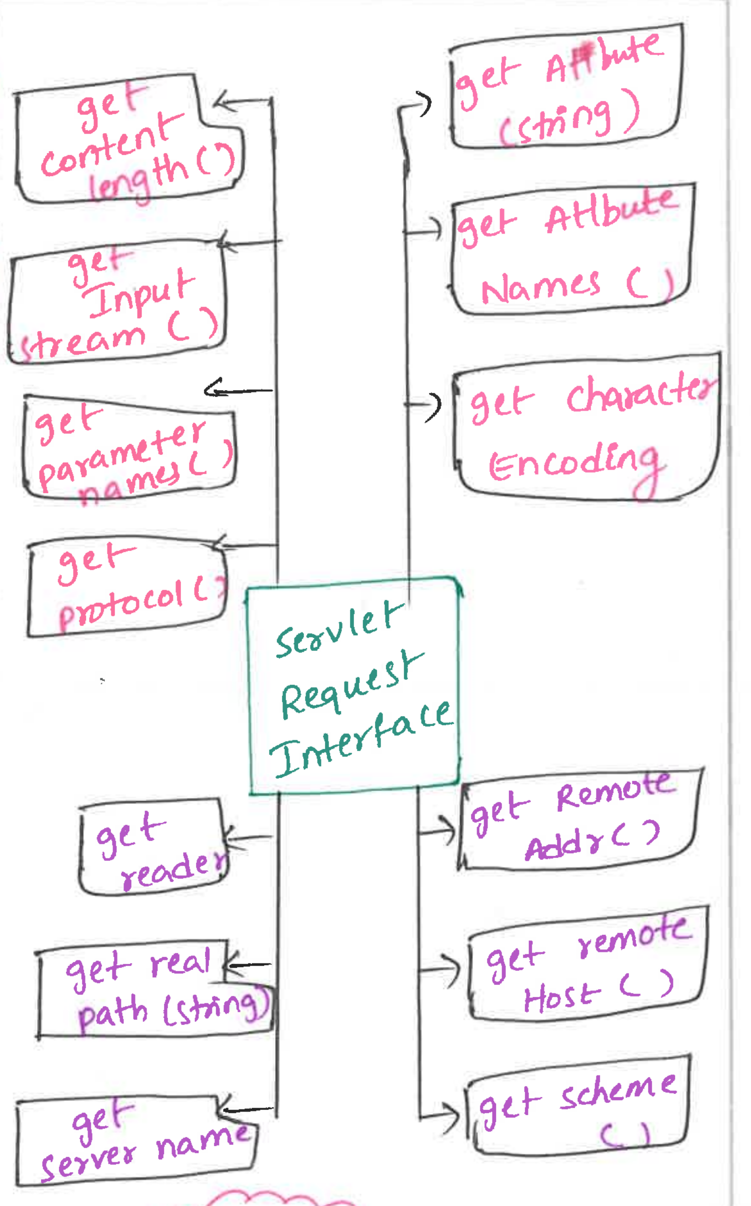
⟹ compling the seavelet

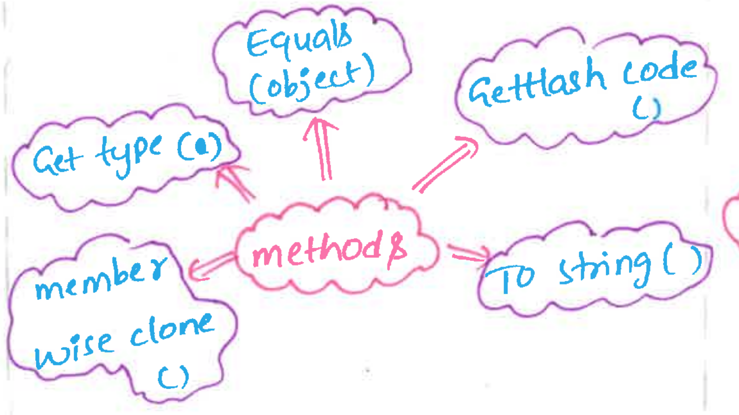
⟹ create Development Descriplor

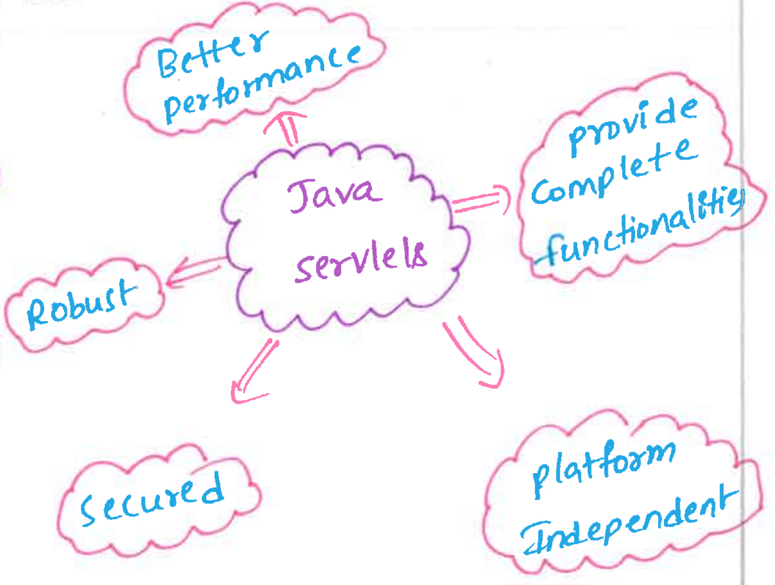
⟹ start the server

⟹ starting tomcat server for the first tyme

⟹ Running the Servelet Appln



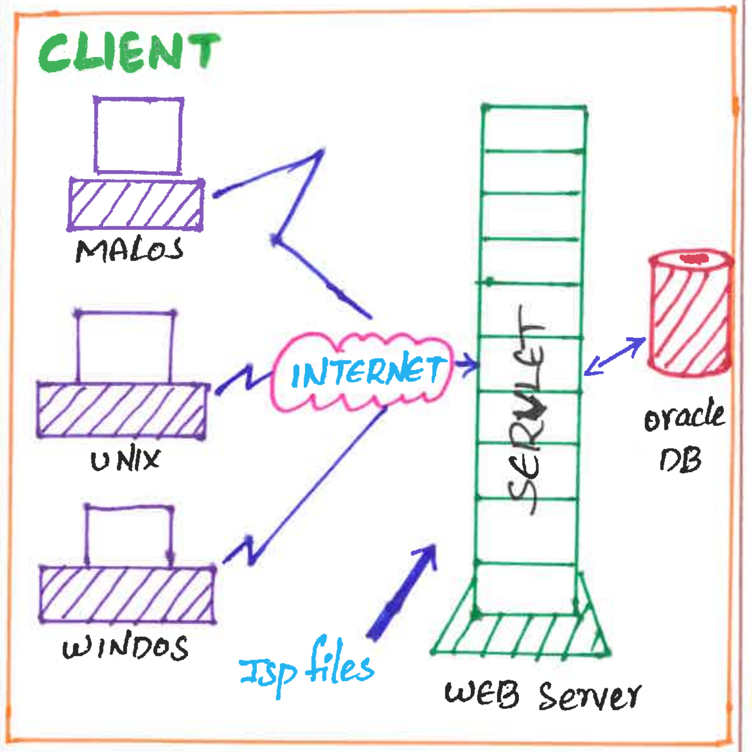




- Generic servlets
- HTML Ale
- Java class file
- XML file

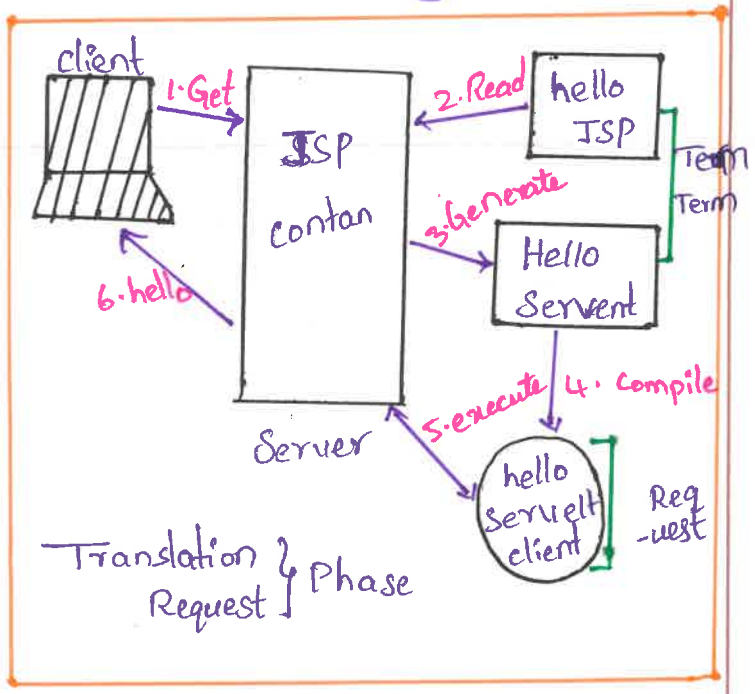
JSP - Java Server Page

JSP - Java Server page

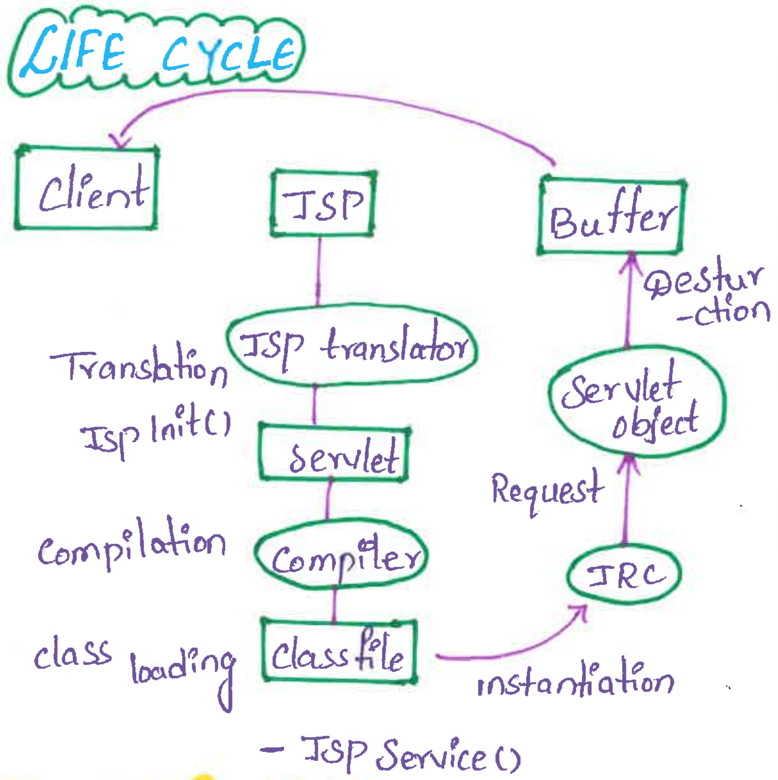


ARCHITECTURE

JSP Processing



LIFE CYCLE



JSP fundamentals

<%= Java Statement %>

⇓ JSP Expression

<% unnonymous Java blocks %>

⇓ JSP Scriplit

<%@ page import= Comma Seperated Java import %>

⇓ JSP import

<% |named Java blocks %>

⇓ JSP declaration

JSP Build-in objects

- request
- response
- out
- Session
- App

JSP_ SYNTAX

```
<Jsp:scriplet>
    code fragment
</Jsp:scriplet>
```

JSP Declaration

<%! declaration: [declaration;]+..%>

```
<JSP: declaration>
    code fragment
<JSP: declaration>
```

JSP Expressions

<% expression %>

JSP DIRECTIVES

<%@ directive attribute = "value" %>

Page-dependent => <%@page..... %>

include file => <%@ include .... %>

Taglibraly => <%@taglib..... %>

JSP Actions

<JSP: actions_name attribute = "value"/>

Action elements

- JSP:include - include file
- JSP: use bean - initials
- JSP: set property - Javabean
- JSP: get property - Javabean
- JSP: forward - new page
- JSP: plugin - Generate new user
- JSP: element - XML element
- JSP: attribute - defines XML element
- JSP: body - define XML body
- JSP: text - JSP page

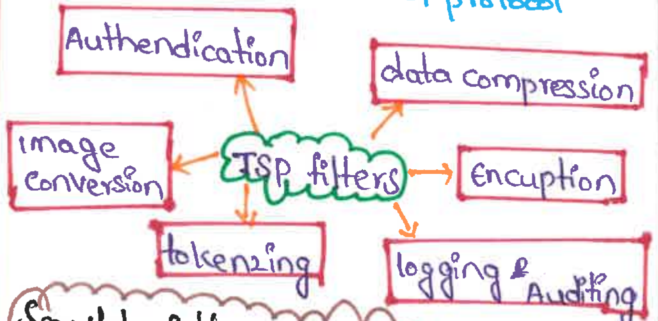
JSP client Request

- Accept - HIME type
- Accept_charset. : character Set
- Accept-encoding : Compress
- Accept-language : Perfered lang
- Authorisation : client identity
- Connection : HTTP connection
- Cookie : Server

HTTP Servlet - Request object

Javax. servlet. http. HttpServlet Request

- Cookie[] get cookie() - returns array
- Enumeration get Attribute Names() - Enumeration_Names
- Enumeration get parameter Names() - Enumeration_string
- Http Session get session() - Current session
- Http session get session (boolean create) - current Http Session
- String get Method() - Http method (Put, Get, Post)
- string get protocol() - Name & version of protocol



Servlet filter Methods

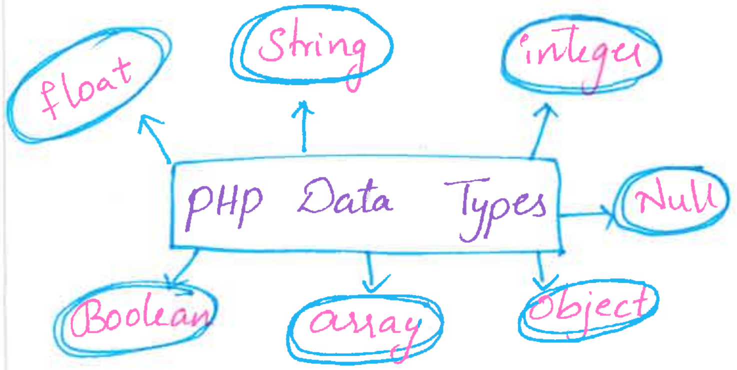
Javax. Servlet. filter

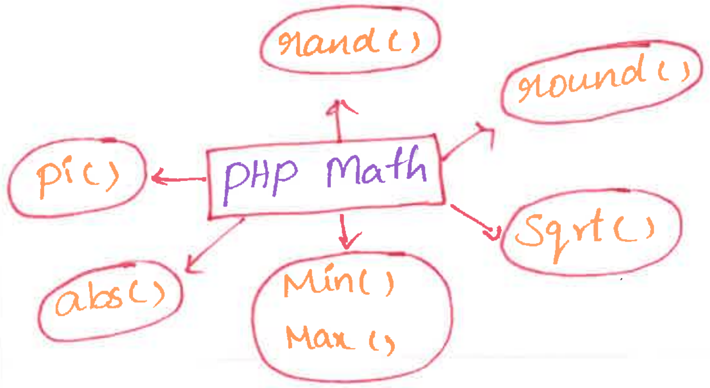
PHP - Hyper Text Prepeocessor

<?PhP opening php tag

function echo "Syntax" ; end state
String

?> Closing php tag



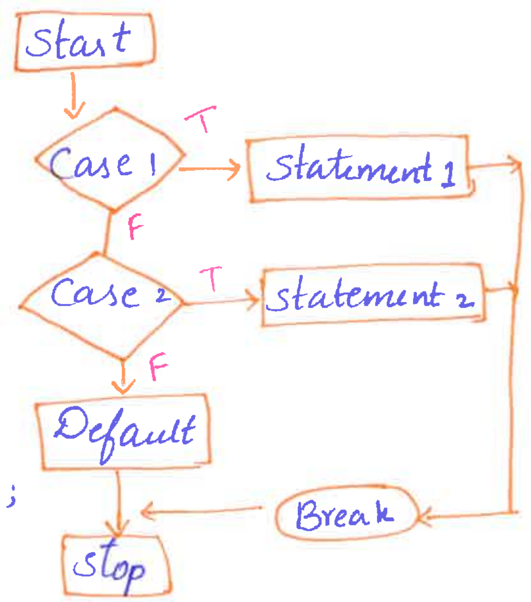


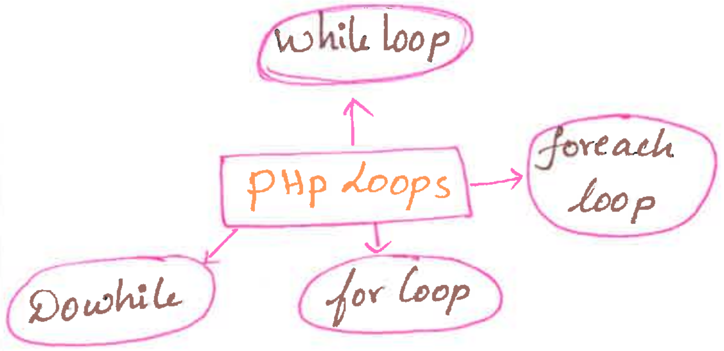
PHp Operators

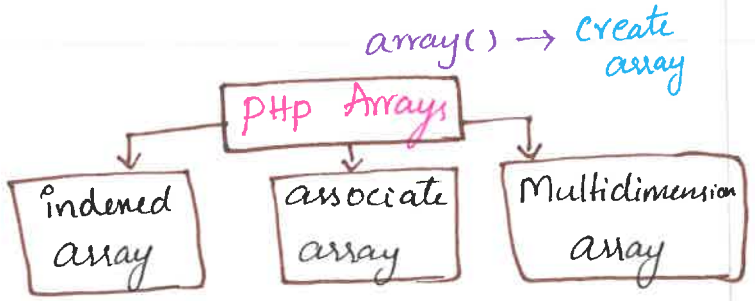
- Arithmetic
- Assignment
- Comparison
- Increment / decrement
- logical
- String
- Array
- Conditional

PHp Switch

```
Switch
 (choice)
{
 Case 1:
  Statement1;
  break;
 Case 2:
 statement 2;
   break;
Default:
    Default statement;
}
```







php array function

- array - diff (arr1, arr2, ...)
- array - filter (arr, function)
- array - flip (arr)
- array - pop (arr)

php array functions

- array-push (arr)
- array - reverse (arr)
- array - walk (arr, function)
- Count (Count)
- in-array (needle, haystack)
- array_merge (arr1, arr2, ...)

php Date and time function

- Check date (month, day, year)
- date (format, Timestamp)
- get date (timestamp)
- mktime (hr, min, sec, month, day, yr)
- Strftime (format string, timestamp)
- strtotime (str)
- time ()

php string function

- Crypt (str, salt)
- explode (sep, str)
- Implode (glue, arr)
- nl2br (str)
- str_replace (search, replace str)
- strip_tags (str, allowed - tags)

php filesystem function

- Clearstatcache ()
- copy (source, dest)
- fclose (handle)
- fget (handle, get)
- file (file)
- filesize (file)
- fopen (file, mode)
- fread (handle, len)
- fwrite (handle, str)
- readfile (file)

Regular Expression Syntax

- ^ start of string
- $ End of string
- - Single character
- (a|b) a or b
- ( ) Group section
- [abc] In range (a, b or c)
- [^abc] not in range
- \s whitespace
- a? zero or one of a
- a* zero or more of a
- a(3) Exactly 3 of a
- a[3] 3 or more of a

File handling in PHP:

⇒ Data Storage

↳ through slower than a database.

⇒ Manipulating uploaded files:

↳ from forms.

⇒ Creating files for download

Open / close a File

* A file is opened with fopen() as a "stream", and PHP returns a "handle" to the file that can be used to reference the open file in other functions.

* Each file is opened in a particular mode.

* A file is closed with fclose() or when your script ends.

File Open Modes

⇒ Open for read only. Starts at beginning of file ⇒ 'r'

'r+' Open for reading & writing

'w' Open only for writing

'a' Open writing, but start at END of current content

'a+' Open for reading & writing

Example :

```
<?php
// open file to read
$toread = fopen('some/file.ext',
        'r');
fclose($toread);
?>
```

Reading Data.

There are two main functions to read data :

⊙ fgets ($handle, $bytes)

⊙ fread ($handle, $bytes)

Example :

```
$handle = fopen ('people.txt',
        'r');
while (!feof($handle)) {
   echo fgets ($handle, 1024);
   echo '<br />';
}
```

File Open Shortcuts :

$lines = file($filename)

→ Reads entire file into an array with each line a separate entry in the array.

fwrite ($handle, $data)

→ write $data to the file

File operations :

* Delete ⇒ unlink ('filename');

* Rename ⇒ rename ('old name', 'new name')

* Copy file ⇒ copy ('source', 'destination');

PHP Connection to Database :

⇒ In php, we can connect to DB using XAMPP web server.

Requirements :

XAMPP web server.

How to connect ?

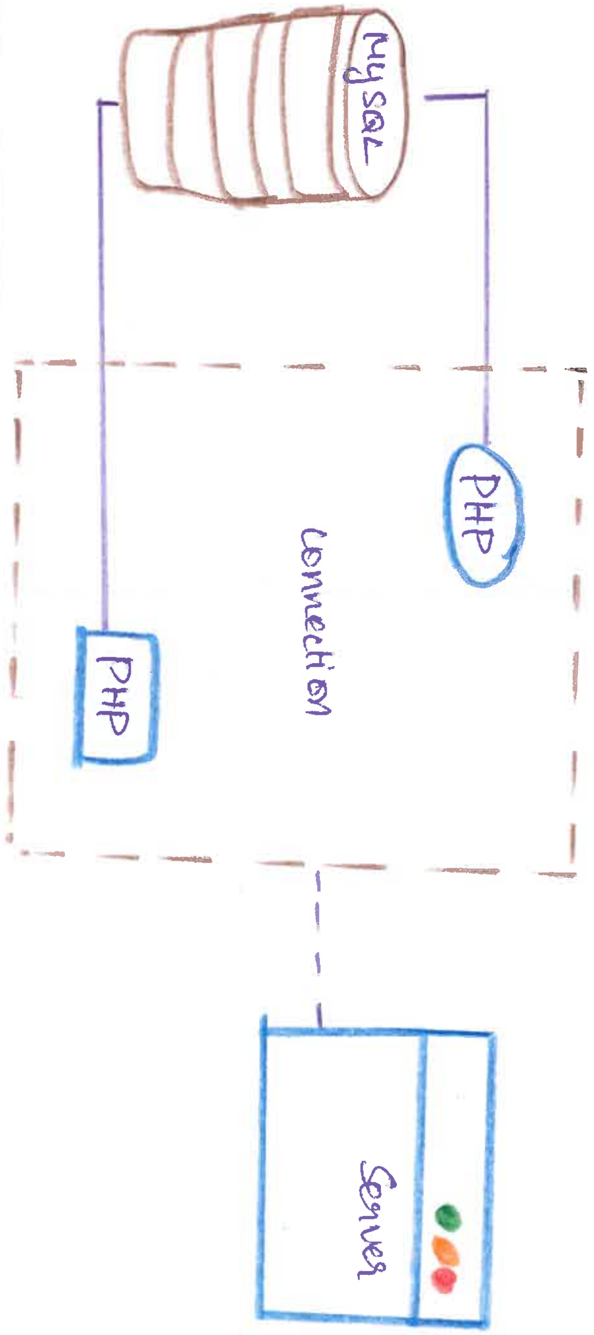
① Connect PHP application with MySQL

② Retrieve database server information.

③ Manage error generated from db calls.

④ Work with db records using the Create, Read, update and Delete function

PHP Database connection :



Structure of XML

Declaration

<?XML Version = "1.0"
encoding = "UTF-8"?>

Document

<message>
<Warning>
Hello World
</Warning>
</message>

Element Declaration

<!ELEMENT name (content model)

<!ELEMENT → Key word

name → type/name

(content model) → format definition

Attribute Declaration

<!ATT LIST element name

<!ATT LIST → Key word

element → name associated

name → attribute name

declaration → type of value

default → default value.

Keywords

ID

CDATA

IDREF

ENTITY

NMTOKEN

IDREFS

ENTITIES

Grouping

<start >End

Occurrence Indicator
- ?optional
- +required repeatable
- optional repeatable

Enumerated Value
- Notation (x/y)
- (a/b/c) attribute list

Default attributes
- "Values"
- #Required
- # implied

Reserved attributes
- Xml Space
- Xml lang

Predefined General entities.

&amp

&lt

&quot;

&gt

&spos

Document Type Declaration

<!DOCTYPE name External-ID

[Declaration] >

<!DOCTYPE → Keyword

name → docname

External-ID → Pointer.

Parameter Entity

→ INTERNAL

<!ENTITY % name "text">

<!ENTITY → Keyword

% → Parameter entity

name → entity name

"text" → entity Value

→ EXTERNAL

<!ENTITY % name External ID>

<!ENTITY → Keyword

% → Parameter entity

name → entity name

External ID → Pointer

General Entity

- Internal
- External

<!ENTITY name External ID

<!ENTITY → Keyword

name → entity name

External ID → Public Indicator

NDATA name → Keyword NDATA

Processing Instruction

<?target xxx text xxx ?>

Notation Declaration

<!NOTATION name External-ID

<!NOTATION → Keyword

name → name of identity

External-ID → Public Identifier

Comment

<! - - - xxx - - ->

↳ any text content

Start Tag:

<tag attribute name = "attribute Value">

tag → element name

"attribute Value" → One more Values.

Empty Element

<name/>

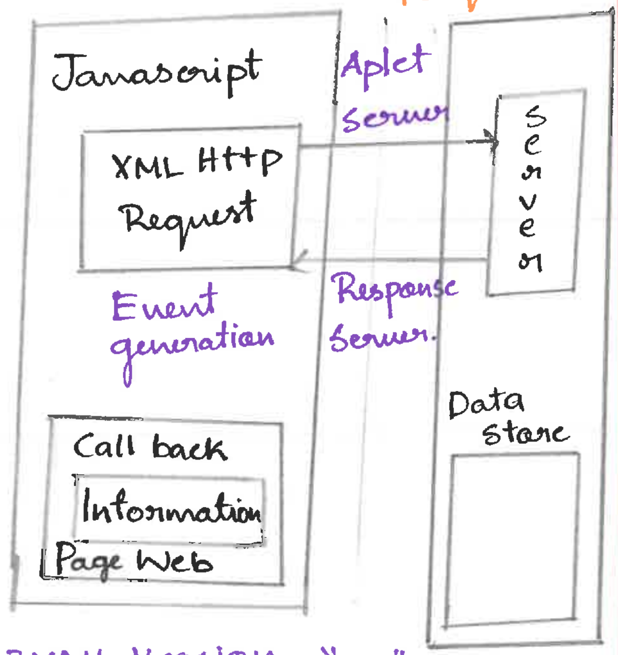
<name> </name>

DTD - Document type Declaration

XML HTTP Request

Create

Var http = new XMLhttp Request()



<?XML Version = "1.0" encoding = UTF = 8?>

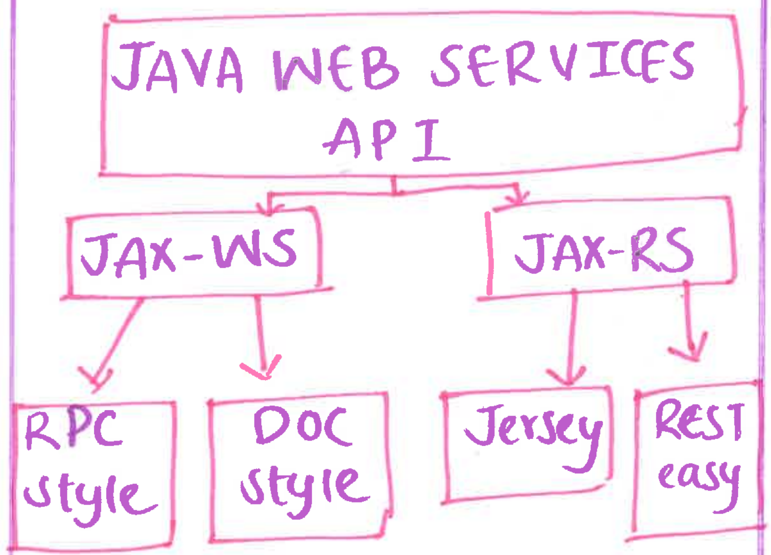
<?Resource Xmls = "URL"?>

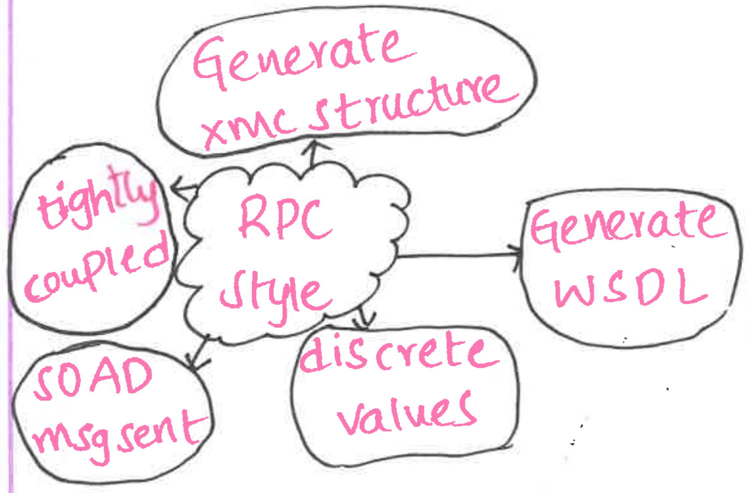
<Request method = "Get">

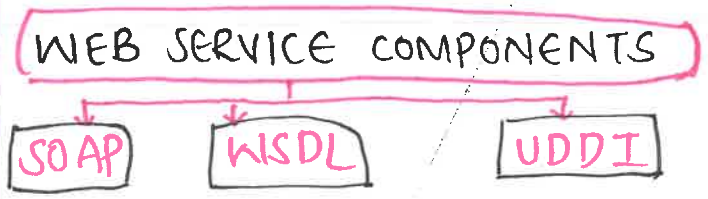
<Desc> xxx </Desc>

<text-ID> aa </text-ID>

JAVA WEB SERVICES



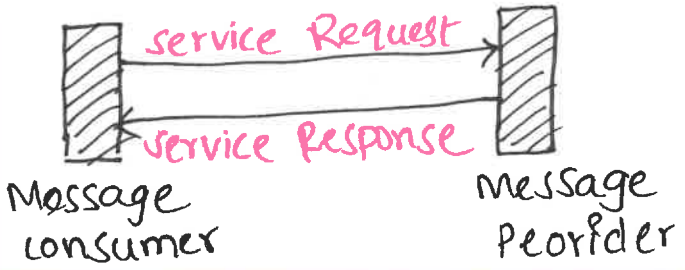




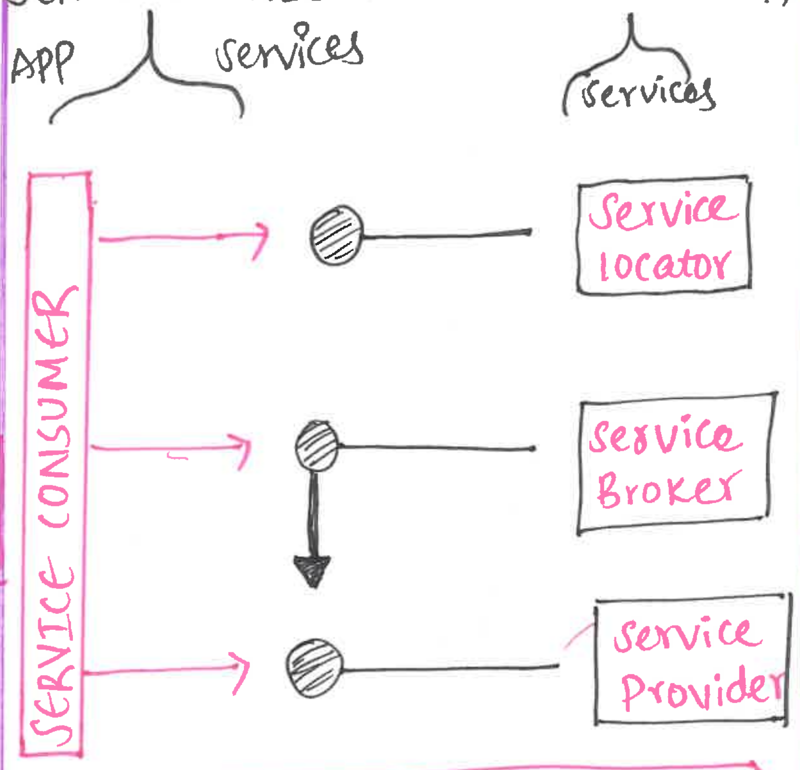
SOAP – SIMPLE OBJECT ACESS PROTOCOL

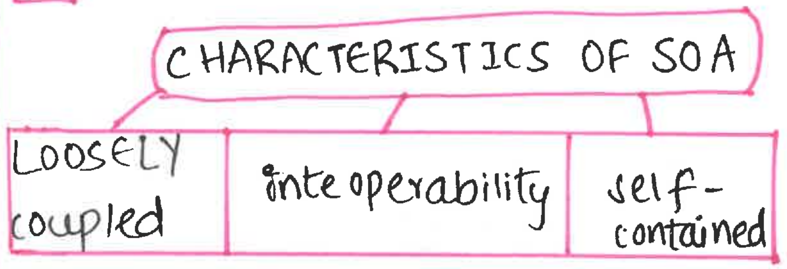
REST – REPRESENTATIONAL STATE TRANSFER

SOA service orientad Architecture

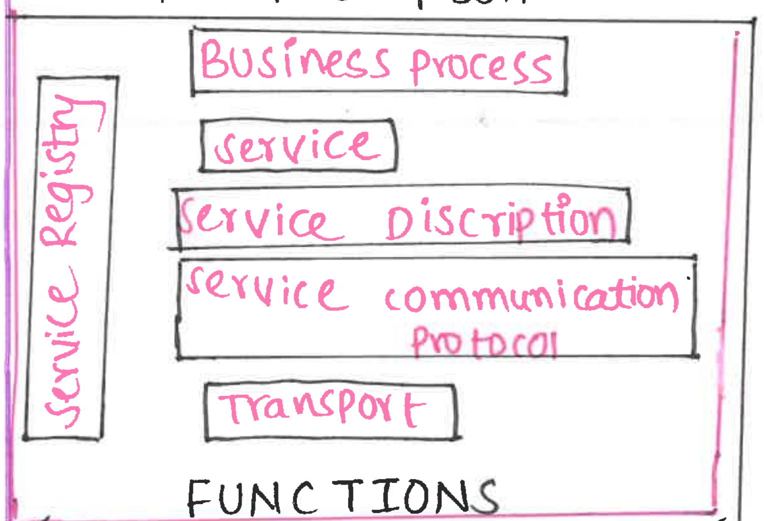


SERVICE ORIENTED TERMINOLOGY

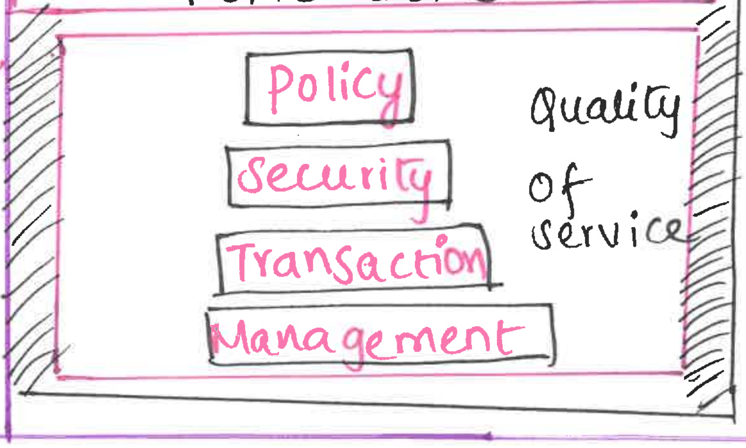




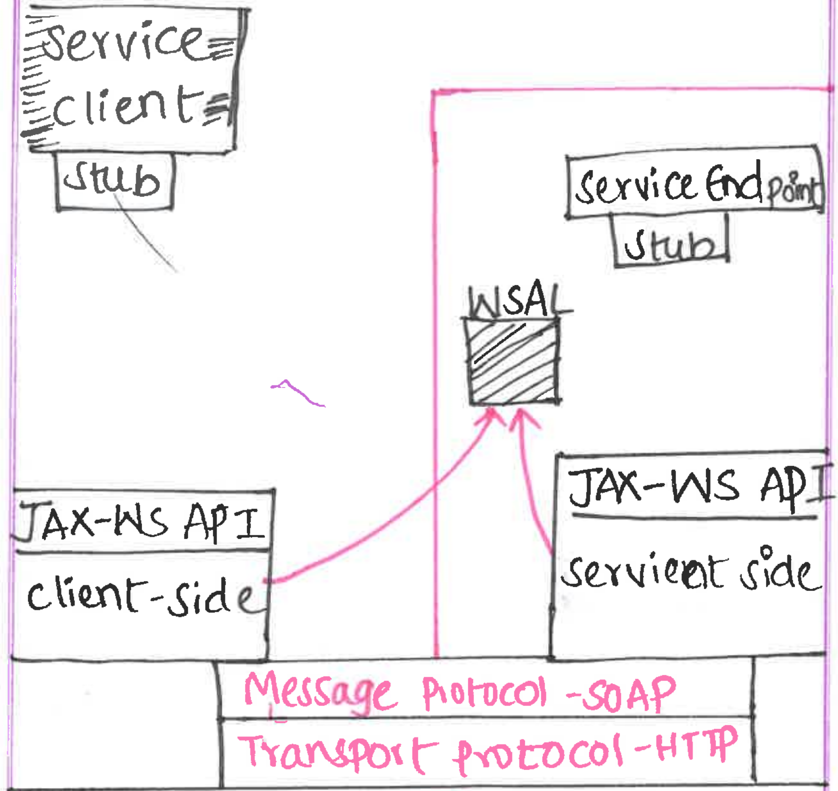
COMPONENTS OF SOA

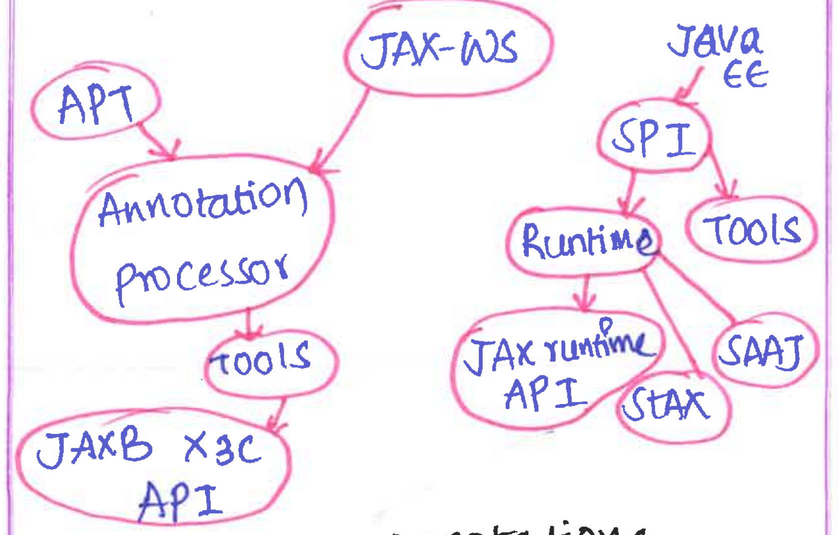


FUNCTIONS

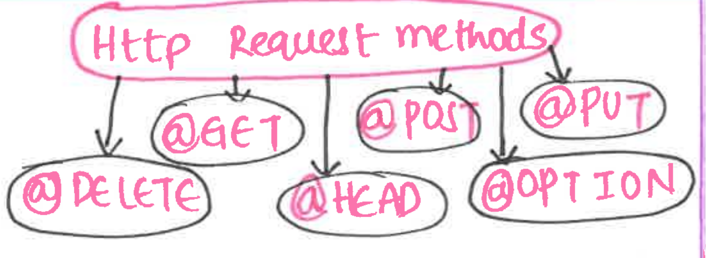


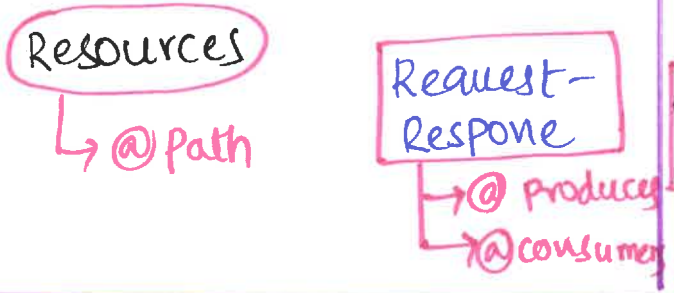
JAX-WS

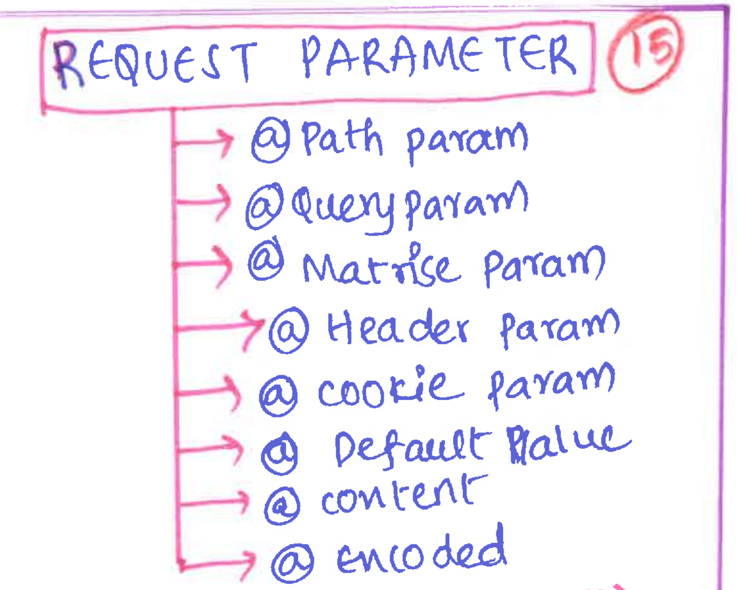




JAX-RS Annotations

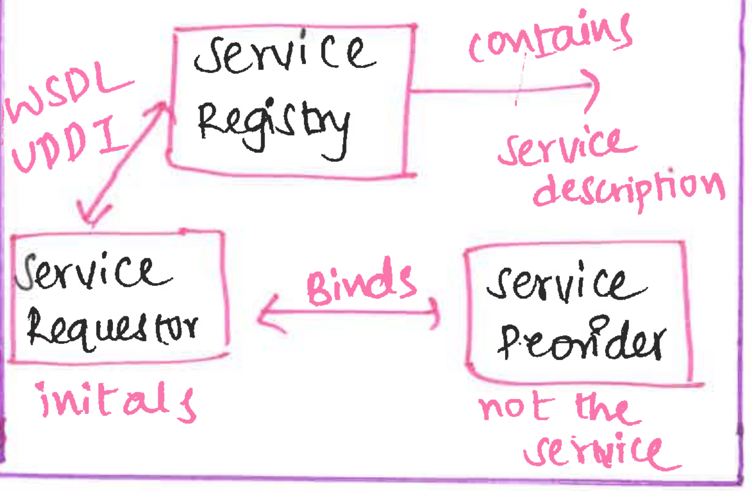






@Path ("/atm/{cardid}")
resources
public class Atmservice {
@GET @Path ("/balance")
@Produces ("text/plain")
Builed in sevilization
Public string balance
{ (@ Pathparam ("cardid")
@ Query Param ("pin")
URI parameter string pin) {
return double string
(getbalance (card,pin)); }

WEB SERVICES ROLES



UDP. User Defined Protocol

> UDP Protocol allows the computer application to send the messages in the form of datagrams. over IP

> Also known as TCP/IP and UDP/IP

> Reliable transport medium.

Features of UDP Protocol

> Transport layer Portocol min amount of communication mechanisms.

> Connectionless

Does not create virtual Path.

> ordered Delivery of Data is not Gwaranted.

Datagrams are not numbered.

> Ports:

Port numbers are defined b/w 0 and 1023.

> faster Transmission connectionless Protocol, transfers through Packets.

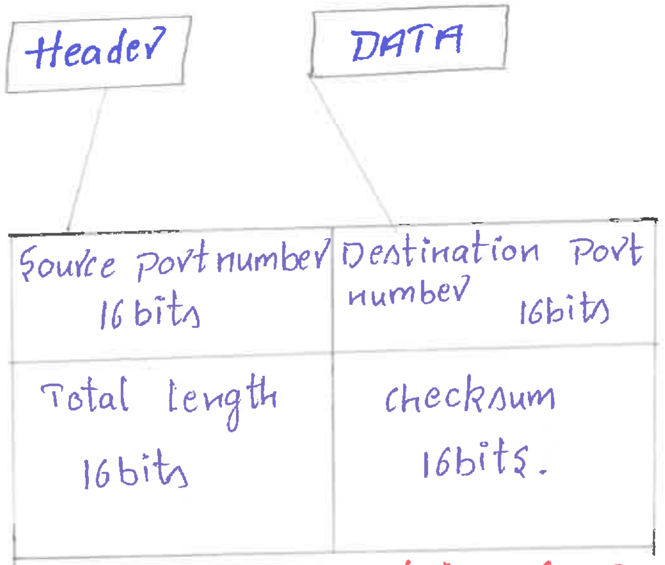
> Segments are handled independents set as segments.

> Stateless

Sender does not get acknowledged UDP is unreliable, deployed with large amount of band width along with actual data.

UDP header format

8 bytes



Source Port number 16 bits	Destination Port number 16bits
Total length 16bits	checksum 16bits.

The UDP header contains four fields

> Source Port number: 16-bit information that identifies packet.

> Destination Port number:
-to identify application level service.

> Length: 16-bit field entire length of UDP.

> Checksum:
- optional field, to check, acuracy.

Concept of Queuing in UDP Protocol

> Distinguish different Processes.

> Two queus for each Process.

> first queus receives msg.

> second one sends the msg.

Input queus:

The UDP Packet uses a set of queues.

Input Module:

- user datagram from the IR
- It enqueues the data.

> Control Block Module:

- It manages the control block table.

> control Block table:

- contains the entry of open Ports.

> Output module:

- creates and sends the user datagram.

Limitations:

> Process-to-Process communication.

> Provides unrediable connection delivery service.

> The UDP message can be lost, delayed, duplicated or can be out of order.

> It does not Provide a reliable transport delivery service.

> It cloes Provide error control to some extent.

Advantage's:

> It Produces a minimal number of overheads.

Differences:

1. TCP connection oriented where as UDP connectionless overted.
2. UDP is an unreliable Protocol, TCP is reliable.
3. UDP is faster than TCP, as it does not guarantee the delivery Packets.
4. Size of UDP 3 bytes, TCP 20 bytes.
5. UDP does not waits for acknowledgement just sends duties.

SIMATS

SCHOOL OF ENGINEERING

Engineer to Excel

Approved by AICTE | IET-UK Accreditation

Saveetha Nagar, Thandalam, Chennai - 602 105, TamilNadu, India