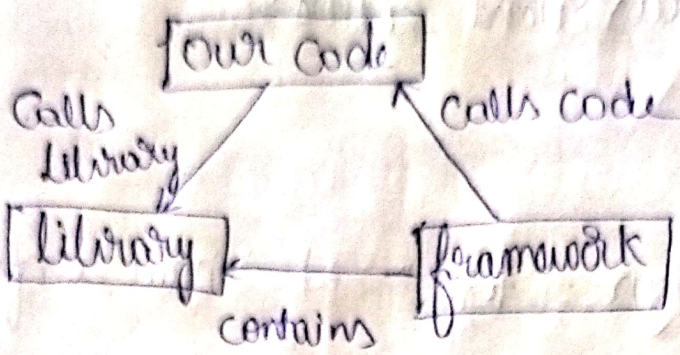
UNIT - III

Angular JS



1) Angular JS is a free and opensource Javascript based web framework for developing single page application.
2) It was maintained mainly by google.
3) It is mainly used for creating the webpages interactive, dynamic and efficient.

Angular JS features:-

MVC
Databinding
Expressive
Routing
Directions
one page application
less code
speed performance
maximum users
Productivity

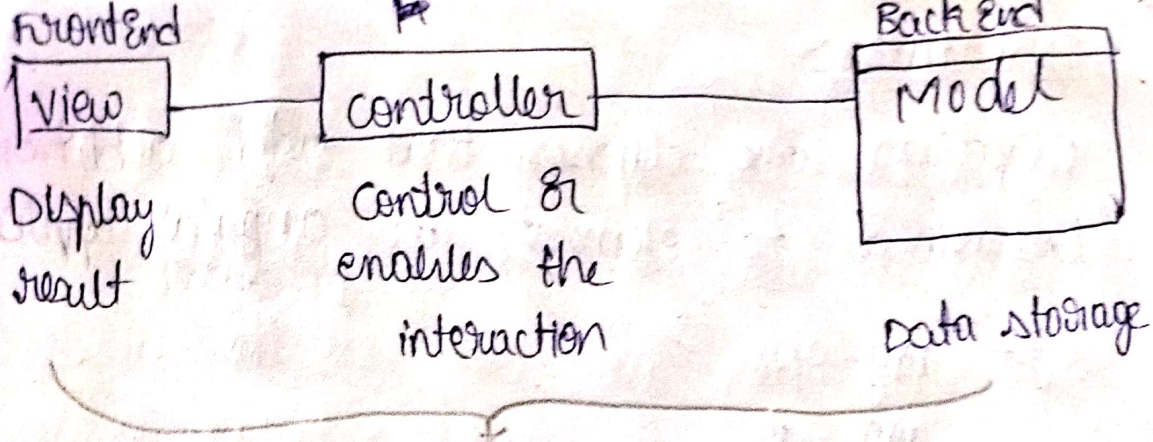


Fig: MVC

M - data and methods to work with model.

V - The interface

C - coordinates interactive between the view & model.

Data Binding :-

The data binding is used for transforming the data between model and view.

Expressive :-

Angular JS expressive can be written inside double braces ({{exp}})

Syntax:

{{ exp }}

Routing :-

Routing is used to navigate from one Component to another component.

Directives:-

Directives are classes that add additional behaviour to elements in angular applica

Eg:- ng-app
ng-init
ng-model

Data Binding:-

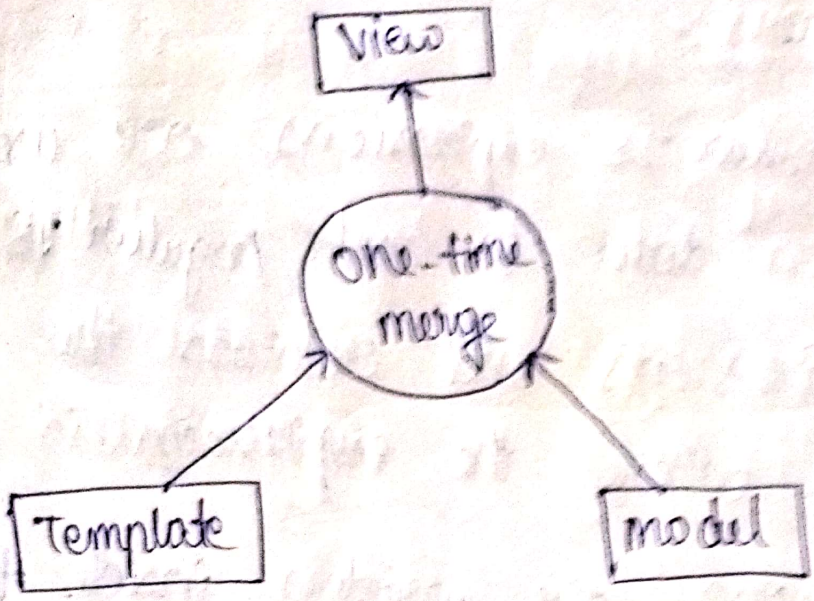
Types of Data Binding:-

1) one-way data binding
2) two-way data binding.

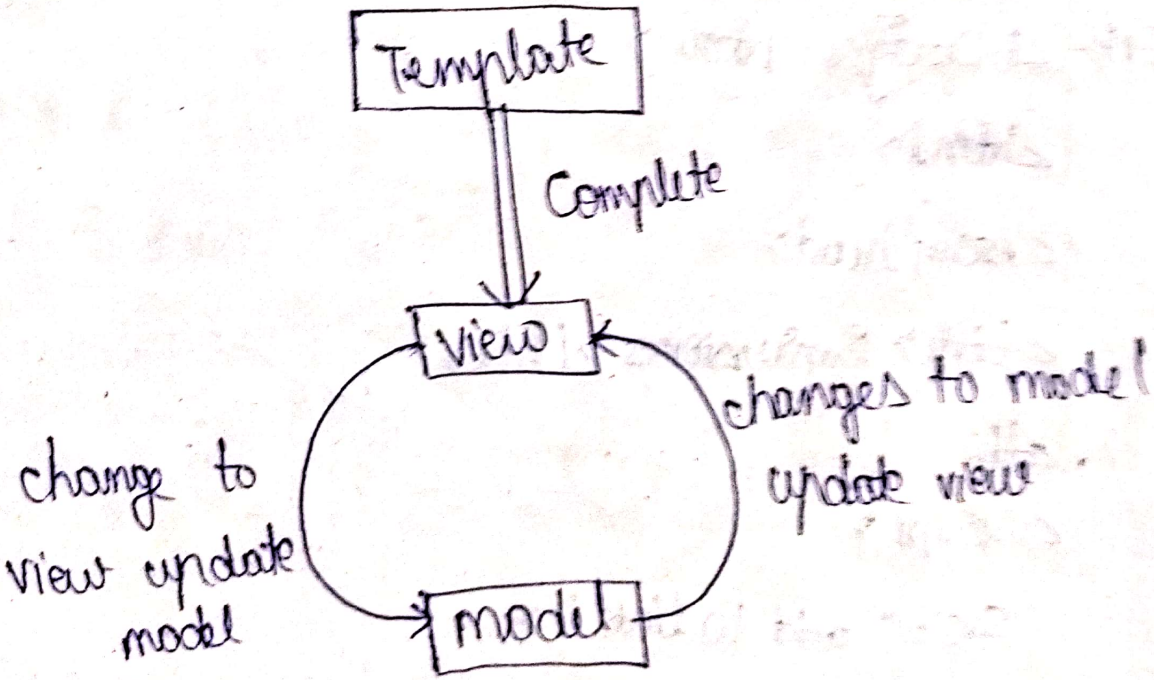
1) one-way data binding is a

Data binding is the process of automatic synchronization of data blw the view and model components.

1) one-way binding changes in the model are reflected on the view but changes in the view to data are not reflected on the model. This binding is called one-way data binding. The binding is from one-way from data to view.



2) Two-way binding is as the name itself suggested that the changes the model are reflected on the view as well as the view changes are reflected on the model.



Expressions :-

In angular JS expressions are used to bin application data to html. Angular JS resolves the expressions and returns the result exactly where the expression is written.

Expressions are written inside the double braces {{ exp }}

Angular JS expressions are very similar to Javascript expression they contain literals operators, strings etc.

Ex:-
```
<!DocType html>
<html>
<head>
<title> Expressions </title>
</th>
<script>
  src = " add to link " >

</script>
</head>
<body>
<div ng-app>
<p> The total value is : {{ 10+20 }} </p>
```

</body>
</html>

Note :- If you remove the directive ng-app html will display the expression without solving it.

Strings:-

```
<!DOCTYPE html>
<html>
<head>
<title> Expressions </title>
<script src = "add to link" ></script>
</head>
<body>
<div ng-app ng-init = " firstname = 'SCET';
        lastname = 'site' " >
<p> my full name is : {{ firstname + " " + lastname }} </p>

</body>
</html>
```

Diff b/w HTML & DHTML

HTML	DHTML
1) HTML is a hyper text markup language	1) DHTML is a dynamic hyper text markup lang
2) HTML stands for static webpage.	2) DHTML stands for Dynamic web page.
3) HTML doesnot have server side ~~too~~ Code	3) DHTML have a server side code.
4) client side technology slow in HTML	4) client side technolog very fast in DHTML
5) Database connectivity is not required in HTML	5) Database connecti required in DHTML
6) HTML file name extension .html, .xml	6) DHTML extension .DHTML

Angular JS Directives:-

ng-app directive
ng-model
ng-bind
ng-init
ng-controller
ng-repeat.

1)→ng-app directive defines root element of an angular JS application.

→ng-app directive will auto-bootstrap (automatically instialize the application when a webpage is loaded)

2) ng-init directive define initial value for angular JS application.

3) ng-model directive bind the value of html controls (input, select, textarea) to application data.

3) ng-repeat repeats the html elements for each element in a item in a collection.

4) ng-bind

Angular JS modules & Controller :-

Definition :-

Angular JS module defines an application. A module is a container for the different parts of an application.

The module is container for the application Controller

Creation :-

A module is created by using the function angular.module

```
<!Doctype HTML>
<html>
<head>
<title> Module </title>
</head> <script src="copy to link"></script>
<body>
<div ng-app="MyApp"></div>
<script>
var app = angular.module("myApp", []);

</script>
</body>
</html>
```

Controllers:-

Angular JS application are controlled by controllers the

The ng-controllers directive define the application controller.

A controller is a javascript object, created by a standard javascript object contrutter.

Constructor:

```
<!DocType HTML>
<html>
<script src=" https://....JS ">
</script>
<body>
 <div ng-app="myapp" ng-controller="mgctrl">
  {{ FirstName + " " + lastname }}
 </div>
<script>
 var app = angular.module ("myapp",[]);
 app.controller ("myctrl", function($scope){
 {
  $scope.Firstname = "SCET";
   $scope.Lastname = "SIET";
  });
```

copy to link

</script>
</body>
</html>
output :-
SCET SIET

** Filters :-

Angular JS filters are meant for formatting and transforming the data to specified format. Syntax :- {{ data | uppercase }}
{{ data | lowercase }}

Types of filters :-

1) lowercase
2) uppercase
3) filters
4) Limit to filter
5) currency
6) order by filter.

Program :-

<html>
<head>
<script src="..."></script>
</head>
<body>


```
<li ng-repeat = "x in data"> {{x | uppercase}}
                                    </li>
</div>
<script>
 Var test = angular.module("test App", []);
 test.controller("test-ctrl, "function($scope)
 {
 $scope.data = ["ram", 'subha", "chinnu"]
 });
</script>
<body>
</html>
```

output :-

RAM
SOBHA
CHINNU

→ lower and uppercase filters are used to transform the data to lower and upprcase format.

3) Filters :-

Filter is also another type of filter which is meant for the filtering the data based on specific character & words matching.

Syntax:-

<input type="text" ng-model="pp" />

<li ng-repeat="x in data | filter:'a'">{{x}}

4) Limit to filter:-

Limit to filter, meant for limiting the data as per the specified length of the characte

Syntax:-

<input type="text" ng-model="pp" />

<li ng-repeat="x in data">{{x | limitTo:4}}

Services:-

1) Services is a functions | object that is availabl for and limited to your angular JS applica

2) Services is a javascript ~~obj~~ function which is ~~used~~ meant for performing specific tas.

Built in functions or services:-

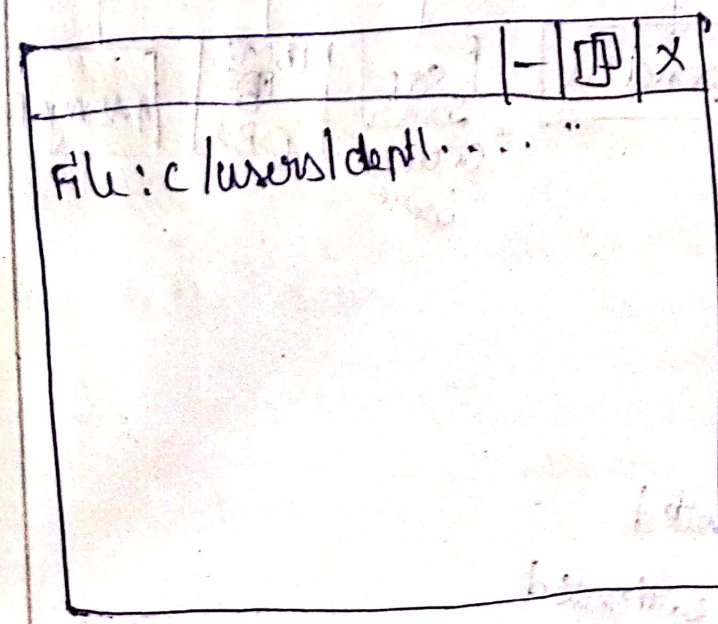
$ location

$ timeout

$ http

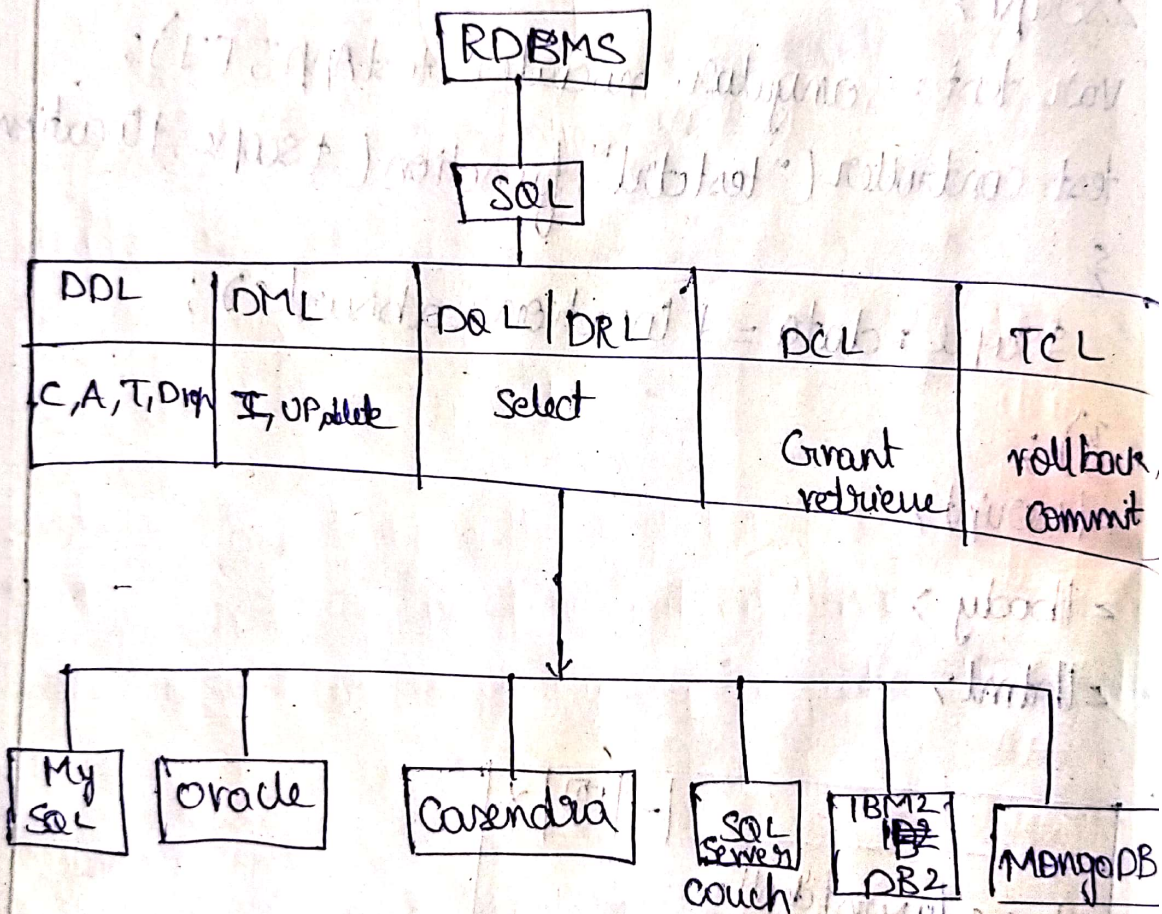
$ intervals.

```
<html>
<head>
 <script src="      " > </script>
<head> <body>
<div ng-app = "testapp" ng-controller = "testctrl">
   {{ data }}

</div>
<script>
var test = angular.module("testApp", []);
test.controller("testctrl", function( $scope, $location)
{
 $scope.data = $location.absurl();

});
</script>
</body>
</html>
```



Explain about structural directives :-

1) ng-if
2) ng-for
3) ng-switch



Types of DBMS :-

1) DBMS
2) RDBMS
3) DDBMS - Distributed
4) OODBMS - Object oriented
5) NDBMS - Networking

Servlet :-

Life cycle of servlet.

Define servlet :-

1) Servlet is a technology which is used to create a web applications.

2) servlet is an API (Application Program Interface) that provides many interfaces and classes including documentation.

3) servlet is a class that extends the capabilities of the servers and respond to the incoming request it can respond to any request.

4) It is used to generate the dynamic content

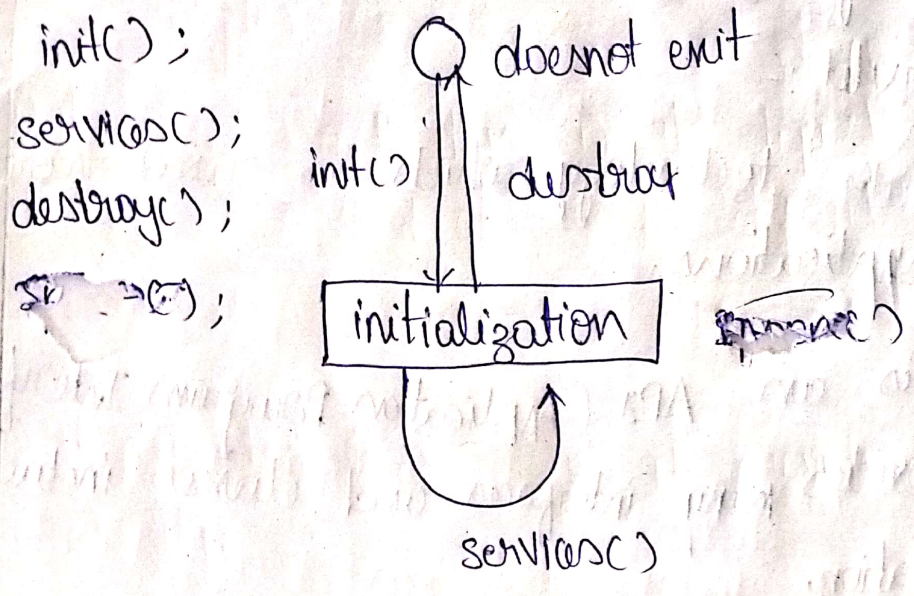
5) servlets are most commonly used with http, hence sometimes servlets are called as http servlets

6) servlet make a use of the java standards, extentions classes in the package

E.g:- Javax.servlet

Javax.servlet.http

Life cycle of servlet:-



1) init():-

init() is designed to be call only once in the life cycle of a servlet.

- It is called when the servlet is first created.
- It is used for one time initialization.
- The servlet is normally created when the user first invoke a URL corresponding to the servlet.

Syntax:-

```
Public void init() throws Exception
{ initialization code
}
```

2) Service():-

The service() is the main method to perform the actual task.

- the servlet container call the services method to handle requests coming from the client and to write the formatted response back to the client.

- The services method check the HTTP request type (get(), post(), call(), doget(), dopost()).

Syntax:-

```
Public void service(ServletRequest request, ServletResponse response)
{
  Actual business logic for providing services to client
}
```

3) destroy():-

1) It is called only once at the end of the life cycle of the servlet.

2) It gives to servlet a chance to close database connections and perform other cleanup activities.

Syntax:-

```
Public void destroy()
{
  closing database connections
}
```

Differences between doget(), dopost(), ~~dooptions()~~

doget()	dopost()
1) The doget() method is the default method of http servlet request.	1) The dopost() method is the default method of http servlet response.
2) It provides less security	2) It provides more secu
3) It can send only limited amount of data.	3) It can send unlim amount of data.
4) It is generally used to query & get some information from the server.	4) It is generally us to update, modify the information fo in the server.
5) In doget() method parameters are not encrypted	5) In dopost() method parameters are encry

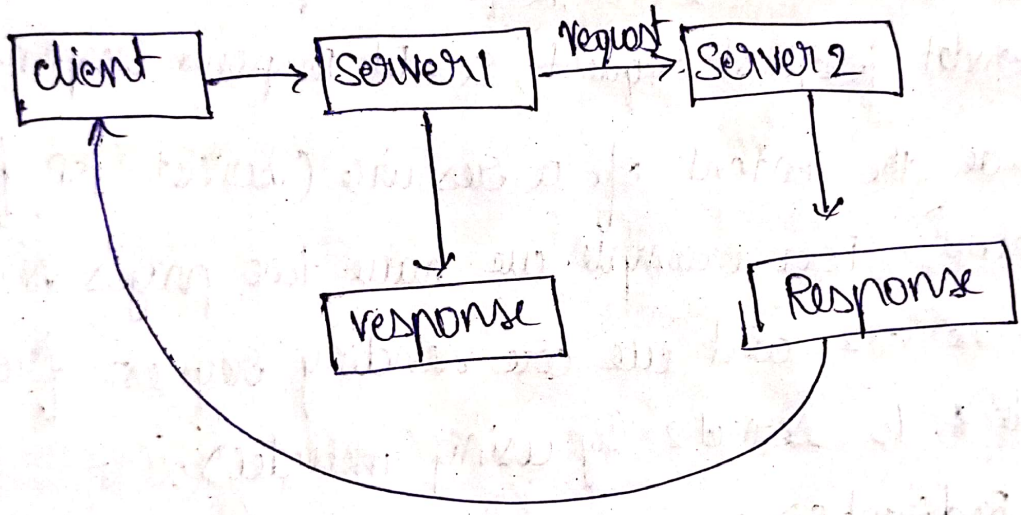
Request dispatchers in servlet :-

⇒ To send requests from one server to another server we use request dispatchers.

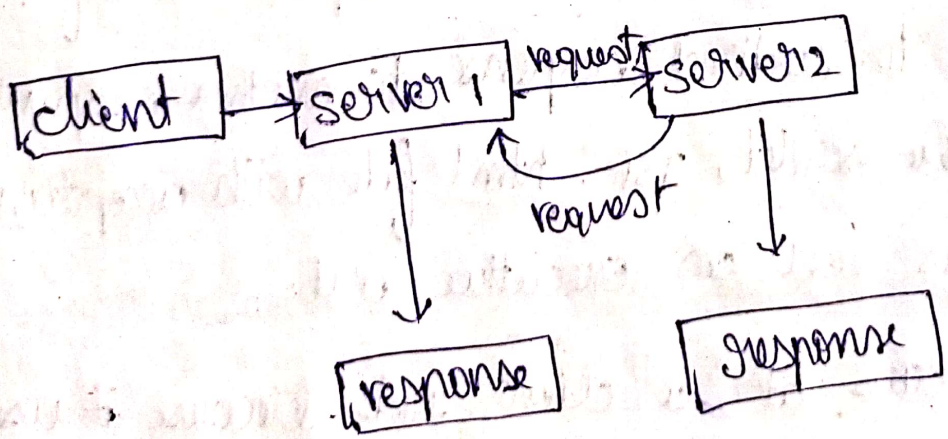
forward() } These two should present in same server
include() }

sendRedirect() } May present in different servers.

1) forward() :-



2) Include() :-



3) Send Redirect() :-

1) Forward():- (ServletRequest Response)

Forward request from a servlet to another resource (servlet, JSP file & html file or server). For example we have two pages servlet1 and servlet2 we are sending request from servlet1 to servlet2 by using forward(). The final response send to the client by using servlet to pages.

2) include():- ~~(ServletRequest, response~~

(ServletRequest request, ServletResponse response) include the content of a resource (servlet, jsp file & html) For example we have two pages server1 and server2 and we are sending request from servlet1 to servlet2 by using include().

SendRedirect():-

SendRedirect() of http servlet response interface can be used to redirect response to another response it may be servlet, JSP, html file with acceptance relative as well as executive url.

It works at a client side because it uses the URL bull of browser to make another request, it can work inside and outside server.

Http Request Response model :-

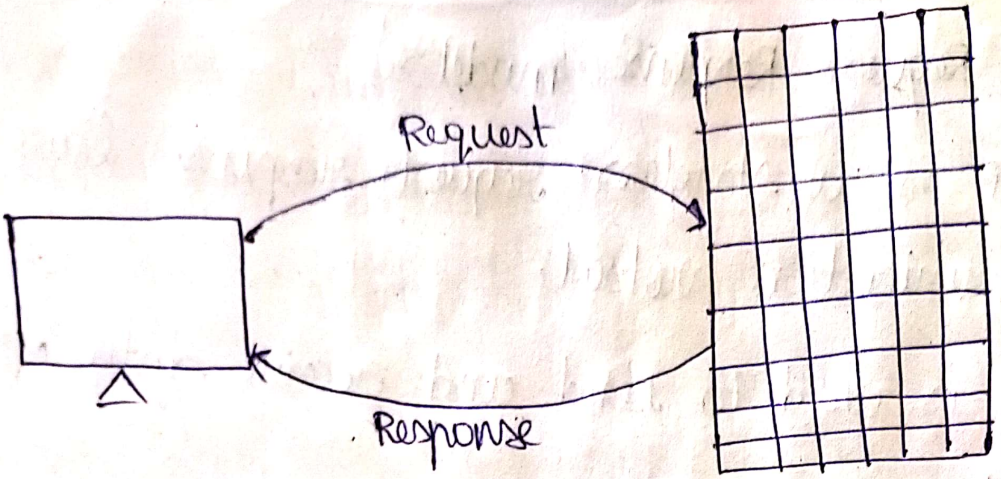
1) Http is a stateless request response based communication protocol.

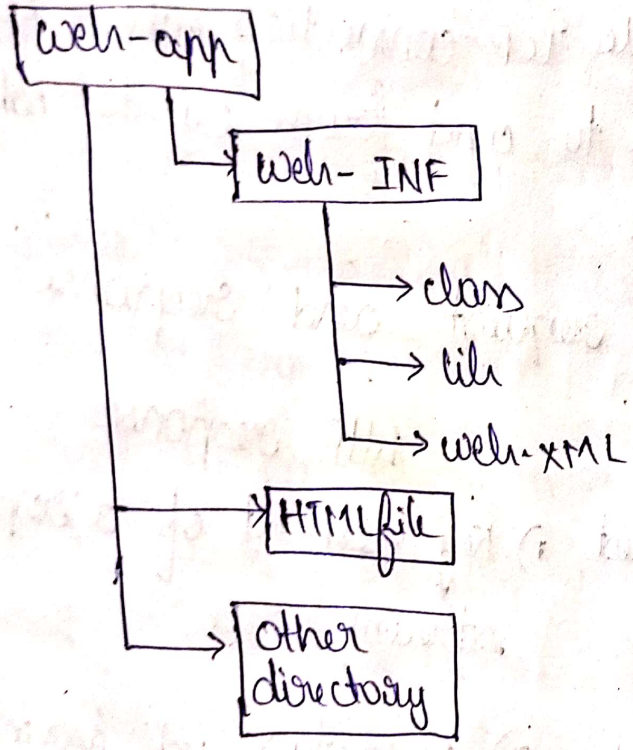
2) It is used to send and receive data on the web.

3) It uses a reliable TCP connection either for the transform data to and from clients which was web browser.

Difference blw Http request and response

Http request	Http response
1) Key element of request stream	1) Key element of a response stream
2) Http method get & post (action to be performed)	2) A status code for whether the request was successful & not.
3) the page to access (URL)	3) the content type
4) From parameters	4) the content (actual content)






```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
Public class <<servlet name>> extends HttpServlet
{
 Public void doget(HttpServlet request, Http Response
                   response) throws servletException IoException
    // Code for Business logic here
}
```

use request objective reads clients requests.
// user response objective through output back to the client.
3// close doget()
3// close dopost()

Types of servlets:-

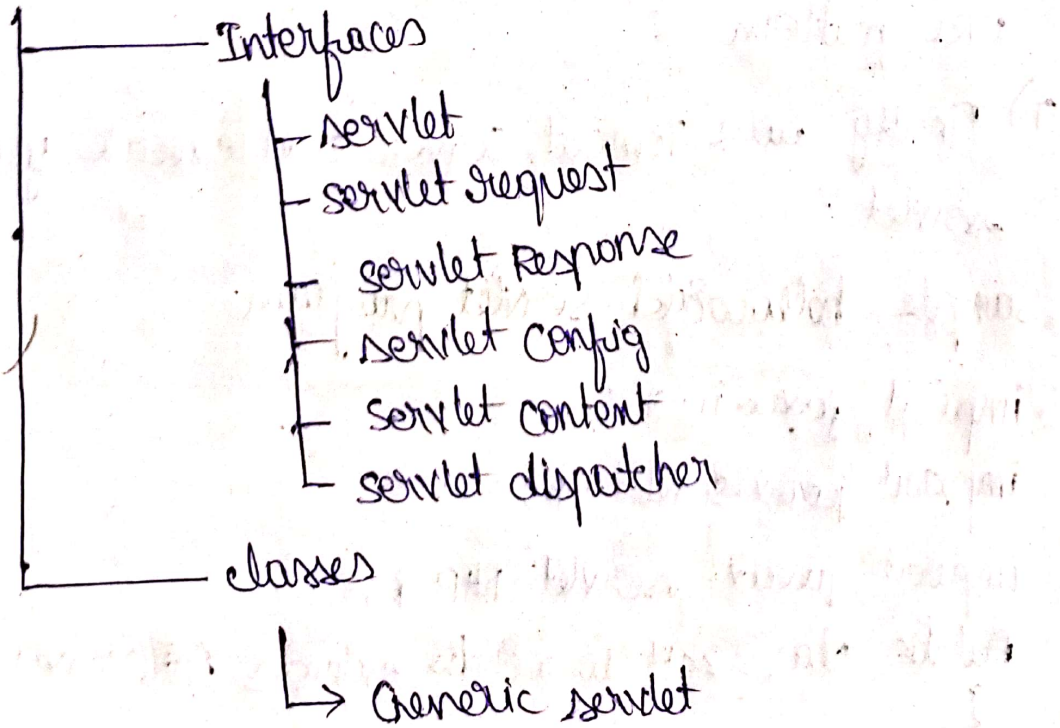
A servlet program developed by 3 ways.

1) A servlet program developed by implementing servlet interfaces.

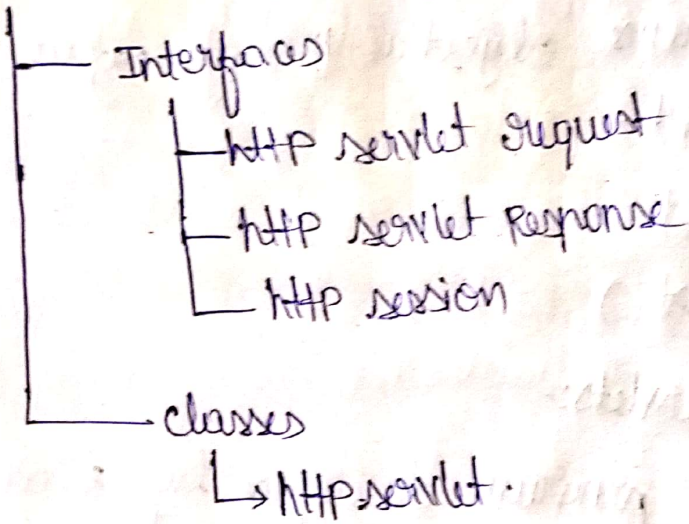
2) A servlet program developed by implementing generic servlet.

3) A servlet program developed by implementing http servlet.

javax.servlet package



Javax.servlet.http package



Compile and Execute your servlets:-

1) Create and compile your servlet program.
2) Create your web application folder.
3) Create WEB-INF folder
4) Create the web.xml file and the classfolder.
5) Copy the servlet classes to classfolder.
6) Edit web.xml to include servlets name and URL pattern
7) Finally run Tomcat server and execute your servlet.

Sample helloworld servlet program

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;
Public class <<Hello world>> extends HttpServer
{
Public void doget(HttpServlet Request request, HttpServlet
```

Response response) throws ServletException IOException

```
{
  response.setContentType("text/html");
  PrintWriter out = response.getWriter();
  out.println("<html>");
  out.println("<head>");
  out.println("<title>HelloWorld!</title>");
  out.println("</head>");
  out.println("<body>");
  out.println("<h1>HelloWorld!</h1>");
  out.println("</body>");
  out.println("</html>");
}
```

Web.XML :-

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<web-app>
  <servlet>
    <servlet-name>Hello</servlet-name>
    <servlet-class>HelloWorld</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>Hello</servlet-name>
    <url-pattern>/HelloWorld</url-pattern>
  </servlet-mapping>
</web-app>
```

Session Tracking:-

1) The session tracking is a mechanism by which you can keep track of previous sessions between server and the browser.

2) For creating the session getSession() method can be used.

3) The method returns the object which stores the binding with the names that use this object.

4) This binding can be managed using getAttribute() and setAttribute() methods.

Types of session Tracking:-

1) stateful protocol (TCP)

2) stateless protocol (HTTP)

1) stateful protocol (TCP):-

In this protocol part of data is exchanged between client and server and this protocol always keep the communication session.

For Example - TCP

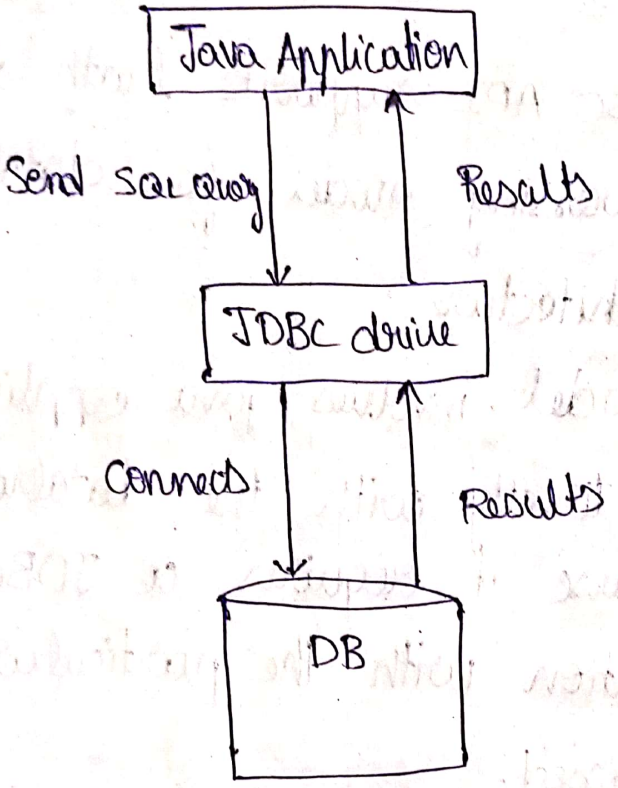
2) stateless protocol (HTTP):-

In this protocol part of data is exchanged between client and server and this protocol did not keep the communication session.

For Example :- HTTP.

* Not remember the previous communication *

Java DataBase Connectivity (JDBC) :-



JDBC stands for Java Database connectivity. It is a Java API to connect and execute the query ~~to~~ with the database.

~~Java~~. import java.sql.*;

import javax.sql.*;

Features of JDBC :-

1) JDBC is an API using which you can communicate with any database without retrieving our java application. the most of the JDBC drivers are implemented in java. JDBC

is known as a platform independent te

2) Using JDBC API you can able to perfo
all basic data operations easily.

3) The JDBC API supports both 2-tier an
3-tier processing model for database acce

2-tier Architecture:-

1) 2-tier model involves java application comm
-nication directly with the database. For
this purpose it requires a JDBC drive fo
communication with the particular databa
being accessed.

2) After that the user send the query to th
database.

3) The database proces the query and it
sends the response ~~to the~~ back to the user.

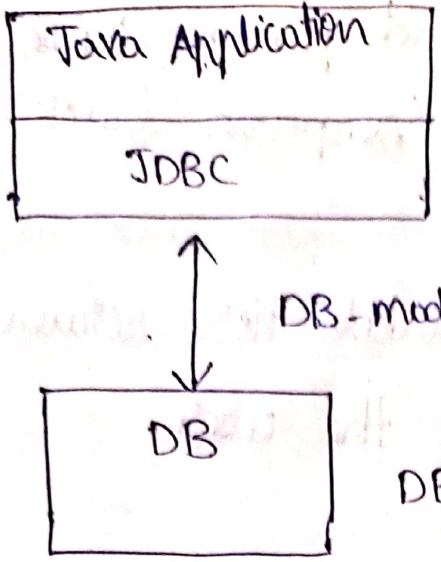
Advantages:-

1) It is easy for
maintainence and modifi-
-cations

2) Communication is fast.

Disadvantages:-

1) In this architecture
application performs
will be based on increa
the user.

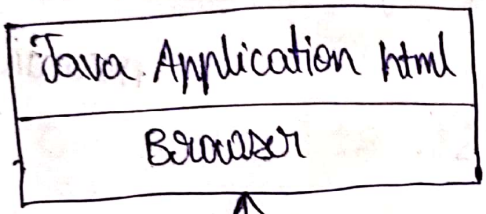


~~server~~ client machine
client

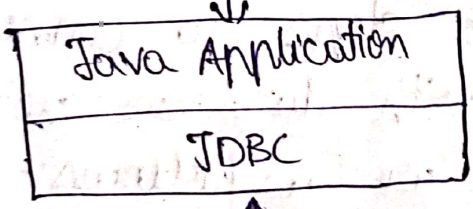
DB-model

DB server.

3-tier Architecture :-

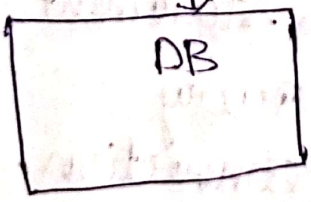


client machine
(GUI)



server machine

DB-modul



DB server.

1) 3-tier model the user queries are sent to the middle tier server.

2) Again from middle-tier serve the queries are again sent to the database.

3) The database takes the queries from middle-

tier server and process those queries and then sends the response back to the middle tier server.

4) After that middle tier server sends the result back to the user.

5) I

Advantages:-

1) It provides high performance
2) It improves security.

Disadvantages:-

1) It is more complex maintain.

There are 5 steps to connect any Java application with database using JDBC. These steps are as follows:-

1) Register the driver
2) creating the connection
3) creating the statement
4) Execute the query
5) closing the connection.

```
import java.sql.*;
Public class JDBCprogram{
    Public static void main(String[] args) throws
SQLException
{
  Driver d = new Oracle.jdbc.driver.OracleDriver();
DriverManager.register.Driver(d);
Connection con= DriverManager.getConnection("JDBC.oracle:
                    thin:@localhost:"root","root");

Statement st = con.createStatement();
st.executeUpdate("create table stu(SNO,number(3),Snam
                    varchar(10));

con.close();

}
}
```

Output:-

SNO	sname

JSP :- (Java Server page)

1) JSP stands for Java Server Page
2) It is a server side technology.
3) JSP is a combination of template text and JSP element.
4) The template text can be scripting code su as html, xml (web markup language), xml & a simple plane text.
5) JSP elements are responsible for generatir dynamic web pages.

Ex:-

```
<html>
<body>
<%
  out.println("JSP is better than servlet");
%>
</body>
</html>
```

Directive Elements:-

There are 3 types:-

* Page Directive
* Include Directive
* Taglib Directive

Page Directive:-

The page directive is used to provide the information about the page.

Ex:-

```
<html>
<head>
</head>
</body>
```

```
<%@ Page language="Java" contenttype="text/html %>
<%@ page import="Java.util.*" %>
<html>
<body>
<%
 out.println("Today date is " +newDate().toString());

 %>
</body>
</html>
```

Include Directive:-

the include directive is used to provide the name of the file that can be included in the JSP file.

Ex:-

```
<%@ include file="one.html" %>
```

Taglib directive:-

The taglib directive is used to specify the custom tag.

Implicity objects in JSP:-

1) JSP implicit objects are predefined objects th are accessible to all JSP pages

2) JSP implict objects are used to make the p dynamic.

3) The JSP container automatically this objects installation

4) these objects while coding the scripting conte in the scriptlet (<%) and expressions.

Types of implicit objects:-

* Request
* Response
* Applications
* session

Request:-

Request objects are used to pass as a param to the JSP services() method. when a client request is made.

Ex:-

request.getParameter("I/P Controlname")

Response :-

The response object is used to carry the response of a client request after services method is executed.

Ex :-

response.getparameter("Text/html")

Applications :-

This object provides the resources shared with a web application.

Ex :-

application.getServerInfo();

Session :-

This variable is used to access the current client session.

Ex :-

session.setAttribute();

session.getAttribute();

*** JSP Scripting Element :-

Scripting element provide the ability to insert the java code inside the JSP. There are 3 types

1) scriptlet tag
2) Expression tag
3) Declaration tag

→ Scriptlet tag is used Execute the java source code in JSP

Syntax :-

<% Java Source code %>

→ *JSP expression tag is written to the output stream of the response. So you don't write output statement to the write the data.

*It is mainly used to print value of variable and method.

Syntax :-

<% = statement %>

Ex :-

```
<html>
<body>
<% = "welcome to JSP" %>
</body>
</html>
```

→ A declaration tag in JSP is used to declare fields and methods.
The code written

Syntax :-

<% ! field & method declaration %>

Action Elements:-

1) There are many JSP action tags or elements
2) Each JSP action tag is used to perform some specific task.

Types of action tags:-

* forward
* include
* Javabean / usebean

Forward:-

Forward request and response the another element.

Syntax:-

<JSP : forward page = "relative URL | <%= Expression %>" />