

RS=180/-

ADVANCE JAVA

By

Sateesh Gupta

Naresh Technology

SRI RAGHAVENDRA XEROX

Software Languages Material Available

Beside Bangalore Ayyangar Bakery, Opp. C DAC, Ameerpet, Hyderabad.

Cell: 9951596199

S.G.

21.09.15

Content of Adv. Java

Core Modules of Adv. Java is :

1. JDBC
2. Servlets
3. JSP.

Database Servers :

1. Oracle.
2. MySQL.
3. PostgreSQL
4. Ms Access.

Web Servers / Application Servers

1. Web Logic
2. Tomcat.
3. Glassfish.
4. JBOSS.

Tools :

1. ANT
2. MAVEN
3. LOG4J
4. JasperReports.

IDE's :

1. My Eclipse / Eclipse.
2. NetBeans.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Objective of Learning this Course:

→ Developing enterprise application which works on World Wide Web.

(OR)

→ Developing Internet or Intranet applications.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Java platforms

→ JSE (Java platform Standard Edition).
(core Java)

→ JEE (Java platform Enterprises Edition)

→ JME (Java platform Micro Edition)

JSE (Java platform Standard Edition)

→ This platform is used for developing standalone or desktop applications.

Q. What is Standalone application?

Ans:-(i) An application developed in context of one system or one user is called Standalone application.

(ii) A standalone Application can be,

- CUI (Command User Interface)
- GUI (Graphic User Interface)

JEE (Java platform Enterprise Edition)

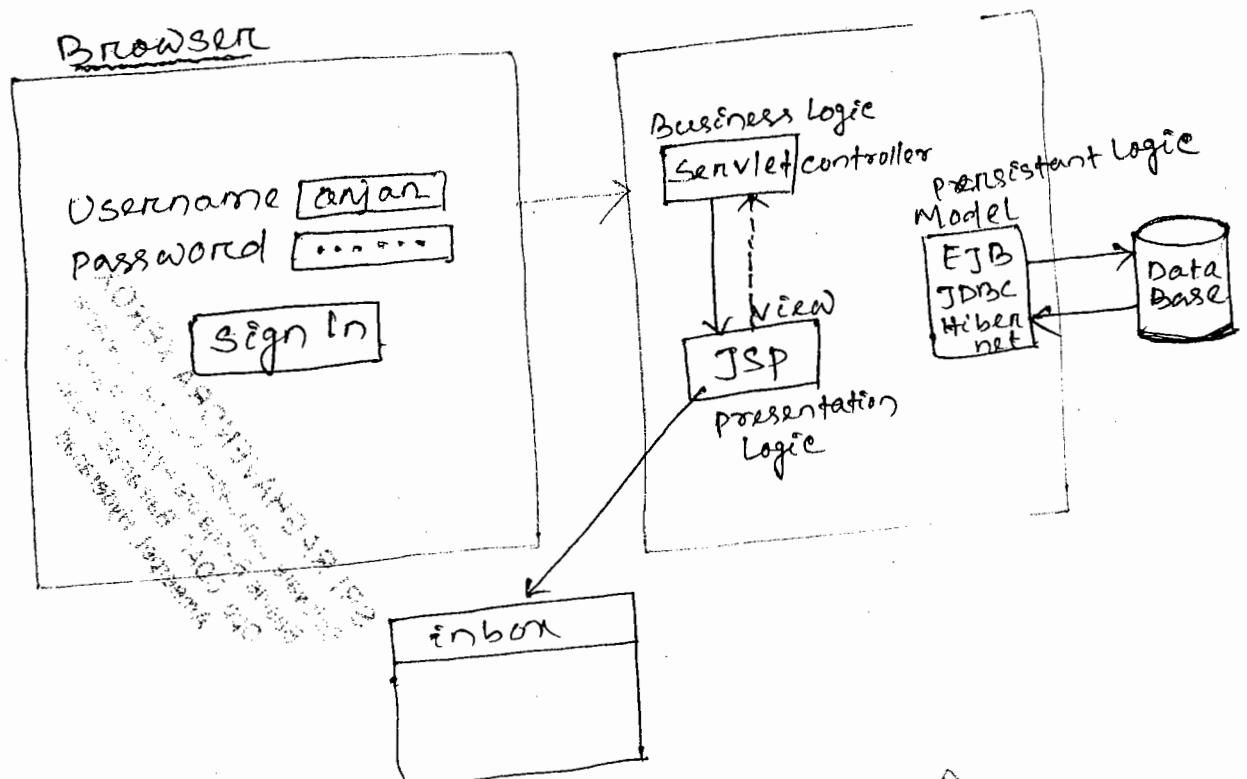
→ This platform is used for developing enterprise applications or distributed applications or Internet applications.

→ It consists

1. Servlet
2. JSP
3. EJB

→ Every Application has 3 layers

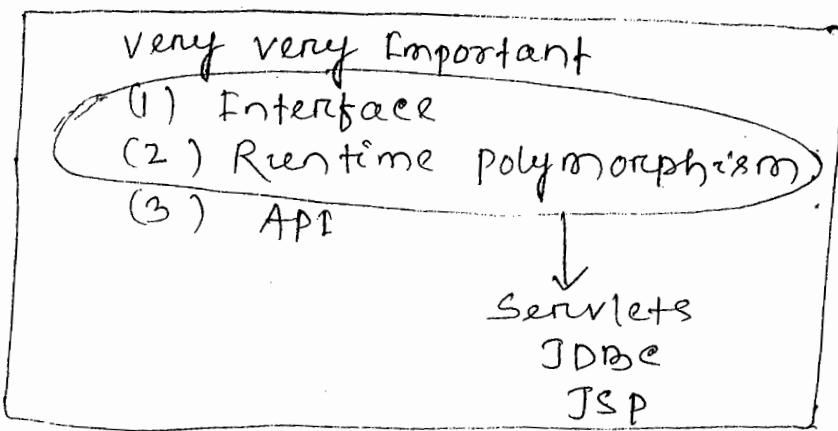
1. persistent layer / logic.
2. Business layer / logic.
3. presentation layer / logic.



JME (Java platform Micro Edition)

→ This platform is used for developing device applications.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.



Very very Important Questions ask in core Java Interview.

- (1) Inheritance.
- (2) polymorphism.
- (3) Strings
- (4) Wrapper class.
- (5) Exceptions
- (6) Multithreading.

Date:- 22.09.15 :-

Q: what is Interface?

- (i) Interface is a ^{reference} datatype.
- (ii) It is an abstract datatype.
- (iii) Interface is a collection of abstract methods.
- (iv) A method without body is called abstract method or A Method without implementation is called Abstract Method.
- (v) A Method with implementation is called concrete method.
- (vi) Interface define ^{public behaviour} of class.
- (vii) Interface define abstract or protocol or Standard.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

(VIII) Interface is inherited but not instantiated.

Syntax:-

interface < interface-name >

{

}

(IX) Interface is inherited but can't used for creating objects.

(X) The class which implements interface is called concrete class. The concrete class must provide the implement of each Abstract Method.

Advantage of Interface:

1. Runtime Polymorphism.
2. Loosely coupled applications.

Program for Example:

interface Animal

{

 void eat();

 void walk();

}

class Dog implements Animal

{

 public void eat()
 {
 ===== (logic)

}

 public void walk()

{

 ===== (logic)

}

}

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Example :

RBI → Service provider
interface Debitcard
{
 void withdraw();
}
SBI → Service implementor
implements
class SBIDebitcard : Debitcard
{
 public void withdraw()
 {
 sopn("2000");
 }
}
HDFC → Service Implementor
class HDFCDebitcard implements Debitcard
{
 public void withdraw()
 {
 sopn("4000");
 }
}

Service calling

ATM

class ATM
{
 void insert(Debitcard d)
 {
 d.withdraw();
 }
}

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Q. what is Runtime polymorphism?

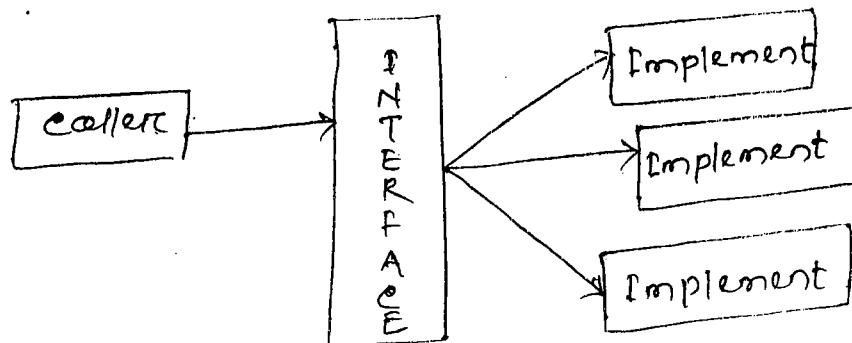
Ans:- An ability of reference variable change its behaviour based on the type object assigned is called Runtime Polymorphism.

Q. what is Loosely Coupled?

Ans:- The code which works irrespective of type is called Loosely coupled code.

→ Implementation can be changed without affecting the caller of the interface so maintaining the application becomes easy.

Example:



Example

```
interface A {  
    void m1();  
}
```

Service provider

```
class B implements A {  
    public void m1()  
    {  
        System.out.println("Inside m1 of B");  
    }  
}
```

Service
provider

class C implements B

```
{  
    public void m1()  
    {  
        System.out.println("Inside m1  
        of C");  
    }  
}
```

service calling.

Class Test

```
p.s.v.main(String args[])  
{  
    A obj;  
    obj = new B();  
    obj.m1();  
    obj = new C();  
    obj.m1();  
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

API (Application programming Interface)

- API stands for Application programming Interface. It is nothing but library.
- Java Library is Packages.

Java.lang
java.util
java.io
java.net
java.applet
java.awt
java.sql

} → Predefined packages
} → These all packages compressed in one file called jar file.

Q. What is Jar?

- Jar is a compressed file which format understand by Java compiler and JVM.
- Jar stands for Java archive file.

23.09.15

Q. What is persistence?

- The process of storing data to permanent place and retrieving data from permanent place is called as persistence and such data is called as persistence data.

Q. What are persistence operations?

- Add, remove, read and modify operations on persistence data are called as persistence operations and the logic which is used for it is called persistence logic.

Q What is persistence store?

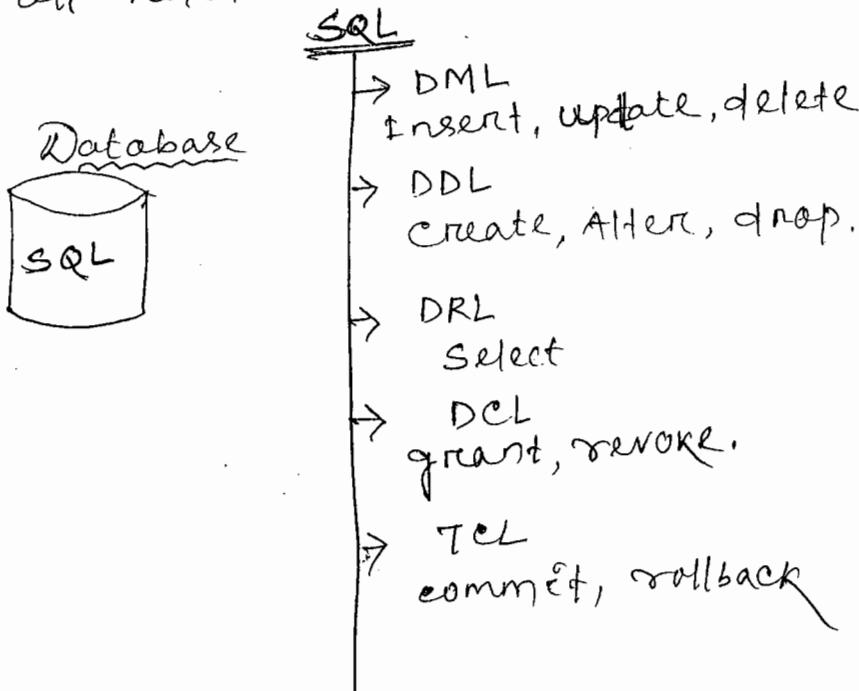
- The place where the data will store permanently is called as persistence store.
- As the data became permanent we can use it at any moment through programming.
- Persistence stores are 2:-
 - (1) Flat files.
 - (2) Database.

Limitation of Flatfile :-

- (i) No security for data.
- (ii) file may be deleted by mistakes.
- (iii) Don't support query language like SQL.
- (iv) Dealing with huge amount of data is quite complex.

SQL:-

- SQL stands for Structure Query Language.
- It is a language which provides persistence operation. This language is common for all relation database.



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

(Q) How frontend application communicate with backend (database)?

1. vendor specific library.

2. ODBC

3. JDBC.

SKI RAHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

1. Vendor Specific Library The person who developed DB.

→ Database Vendor provide a library which is used by for frontend application to communicate with backend (database).

Disadvantages:-

(i) Database Dependent.

(ii) platform Dependent Because these library are developed in C, C++.

2. ODBC :-

→ ODBC is a standard or specification or abstracts developed by Microsoft for developing ODBC drivers.

→ ODBC stands for open database connectivity.

→ ODBC API is written in C, C++ languages.

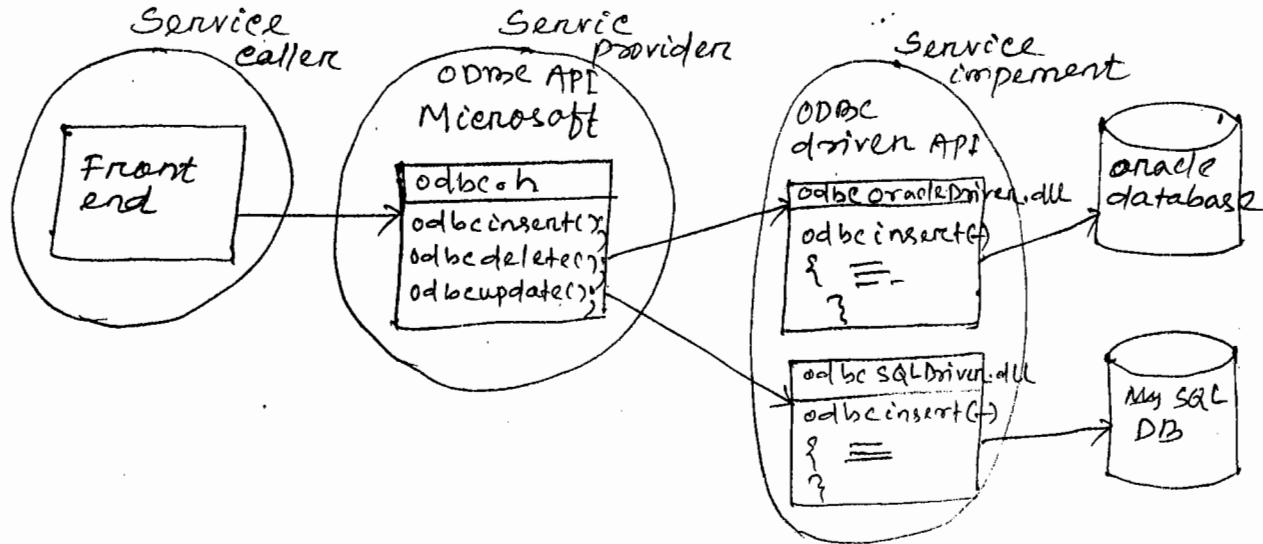
→ ODBC driver API is an implementation of ODBC API.

→ ODBC drivers are developed by,

(1) Microsoft

(2) Database Vendor.

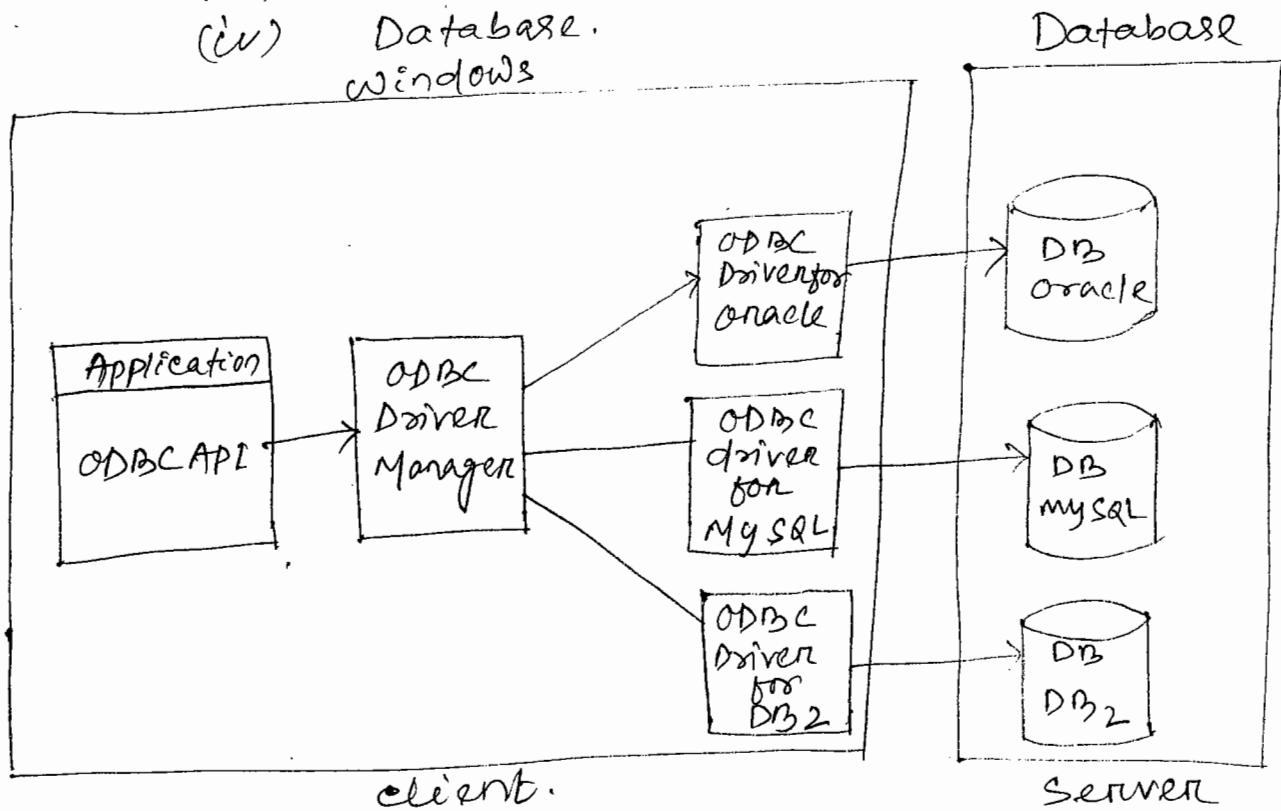
(3) Third party vendor.



ODBC Architecture :-

ODBC Architecture is having different layers
these are :-

- (i) Application
- (ii) ODBC DriverManager
- (iii) ODBC Driver.
- (iv) Database.



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

JDBC

JDBC:

- JDBC stands for Java Database connectivity.
- JDBC is not a language it is an abstract or standards or specifications given by Sun Micro System for developing database and platform independent Java applications.
- JDBC is an API or JDBC API is written or developed in Java Language.
- The current version JDBC API is 4.0 and that comes with Java 1.6 version.
- JDBC is a part of JSE (Java platform Standard Edition).

Q. what is difference between JDBC and ODBC?

ODBC	JDBC
→ ODBC stands for Open database Connectivity.	→ JDBC stands for Java database connectivity.
→ ODBC is developed by Microsoft.	→ JDBC is developed by sun Micro System.
→ This is written in C and C++.	→ JDBC is written in Java.
→ ODBC is platform Dependent.	→ JDBC is platform Independent

- JDBC API is given by Sun Micro System for developing JDBC Drivers.
- JDBC driver is an implementation of JDBC API.
- JDBC drivers are developed by 3 people.
 1. Sun Microsystem.
 2. Database vendor.
 3. Third Party vendor.

JDBC Architecture :-

JDBC Architecture is developed into 4

- | | |
|--------|---|
| Layers | <ol style="list-style-type: none"> (1) Application Layer. (2) JDBC Driver Manager layer. (3) JDBC Driver layer. (4) Database layer. |
|--------|---|

1. Application layer:-

Application uses JDBC API to send SQL statements.

- Application calls the functions of JDBC API
- Application calls the functions of JDBC API (Interface)

2. JDBC Driver Manager :-

JDBC Driver Manager manages JDBC drivers

→ Managing consists of two things

- (1) registering (installing)
- (2) deregistering (uninstalling)

→ Driver Manager acts as a interface between client and JDBC driver.

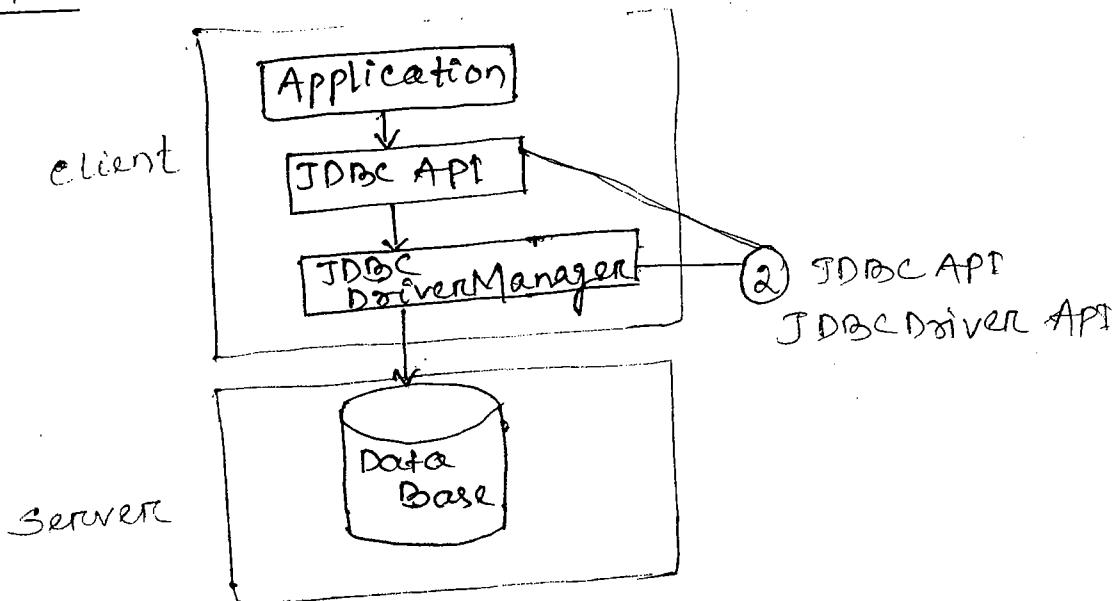
JDBC Driver :-

→ JDBC Driver receives SQL statements and submit those statements to database and return result to client.

Database :-

→ It process SQL statement and return result to driver.

25.09.15



Types of JDBC Drivers :-

JDBC Drivers divided into 4 categories

1. Type - 1 : JDBC - ODBC bridge Driver.
2. Type - 2 : Native API and partly Java Driver.
3. Type - 3 : Network API and All Java Driver.
4. Type - 4 : Native Protocol and pure Java Drivers.

SPI PAGHENVENDRA XEROX
Software Libraries Available
Dop, Amesphere, Hydrometer, Read,
BESIDE, Jdbc, Biographer, Reader,
Software Libraries Available
Material, Available

(i) JDBC-ODBC Bridge Driver : (Type-1)

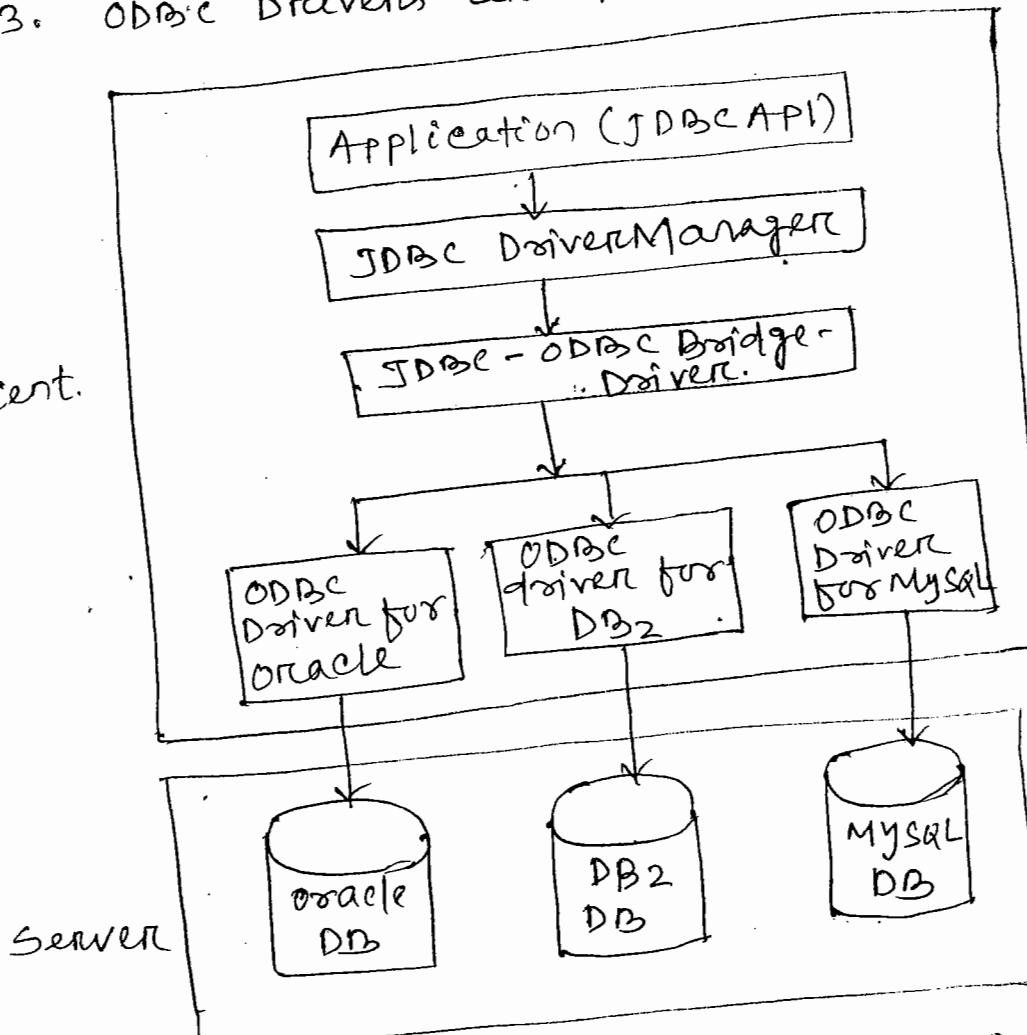
- JDBC-ODBC Bridge driver is developed by Sun Micro System.
- It is not open source.

→ This driver comes with JDK. It is a part of JDK upto 1.7.

→ JDBC-ODBC bridge Driver uses ODBC driver to communicate with database.
Disadvantages :-

1. Not efficient, Because required no. of translations.
2. Every client Machine required ODBC driver installed.
3. ODBC Drivers are platform dependent.

client.

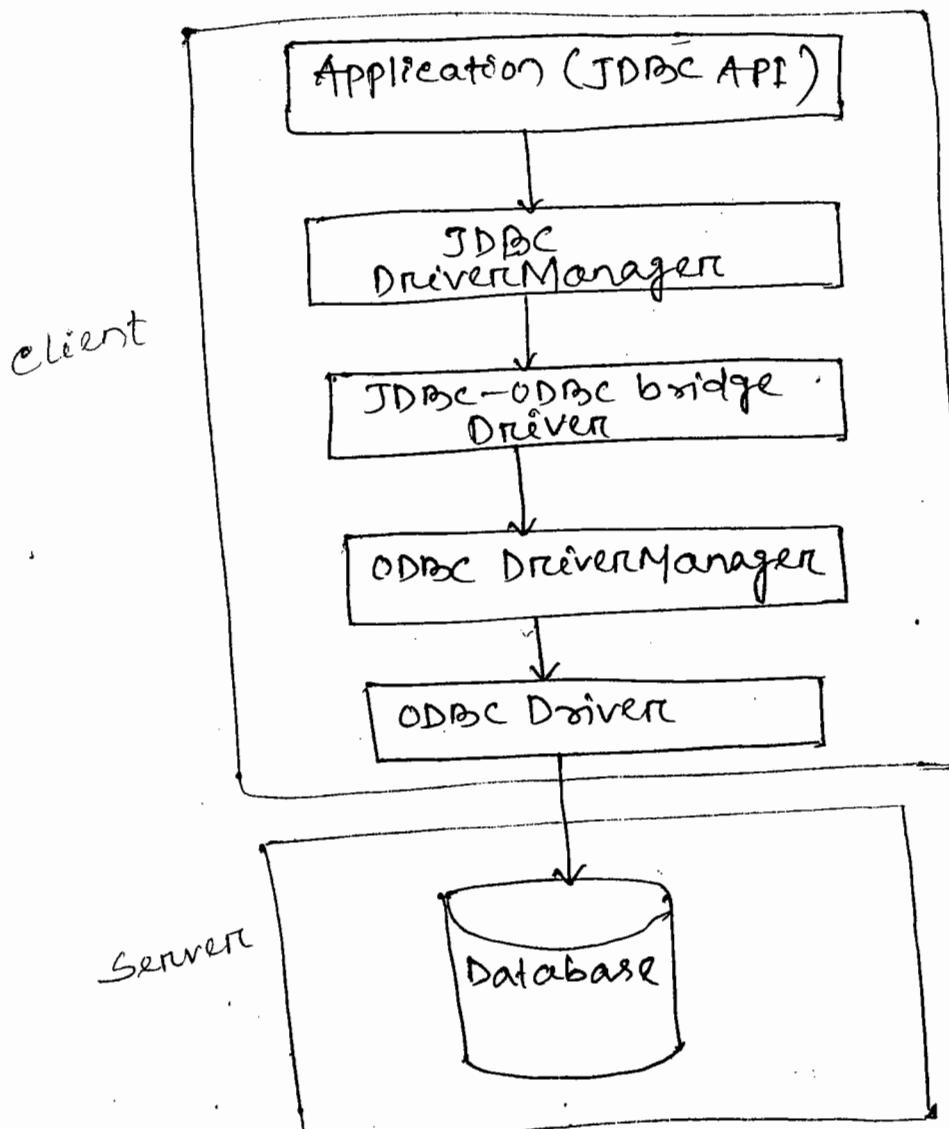


Type-1 JDBC-ODBC Bridge Driver

Advantages :-

→ It is only one driver communicate with various databases provided ODBC drivers.

More translation diagram for type-1 Driver

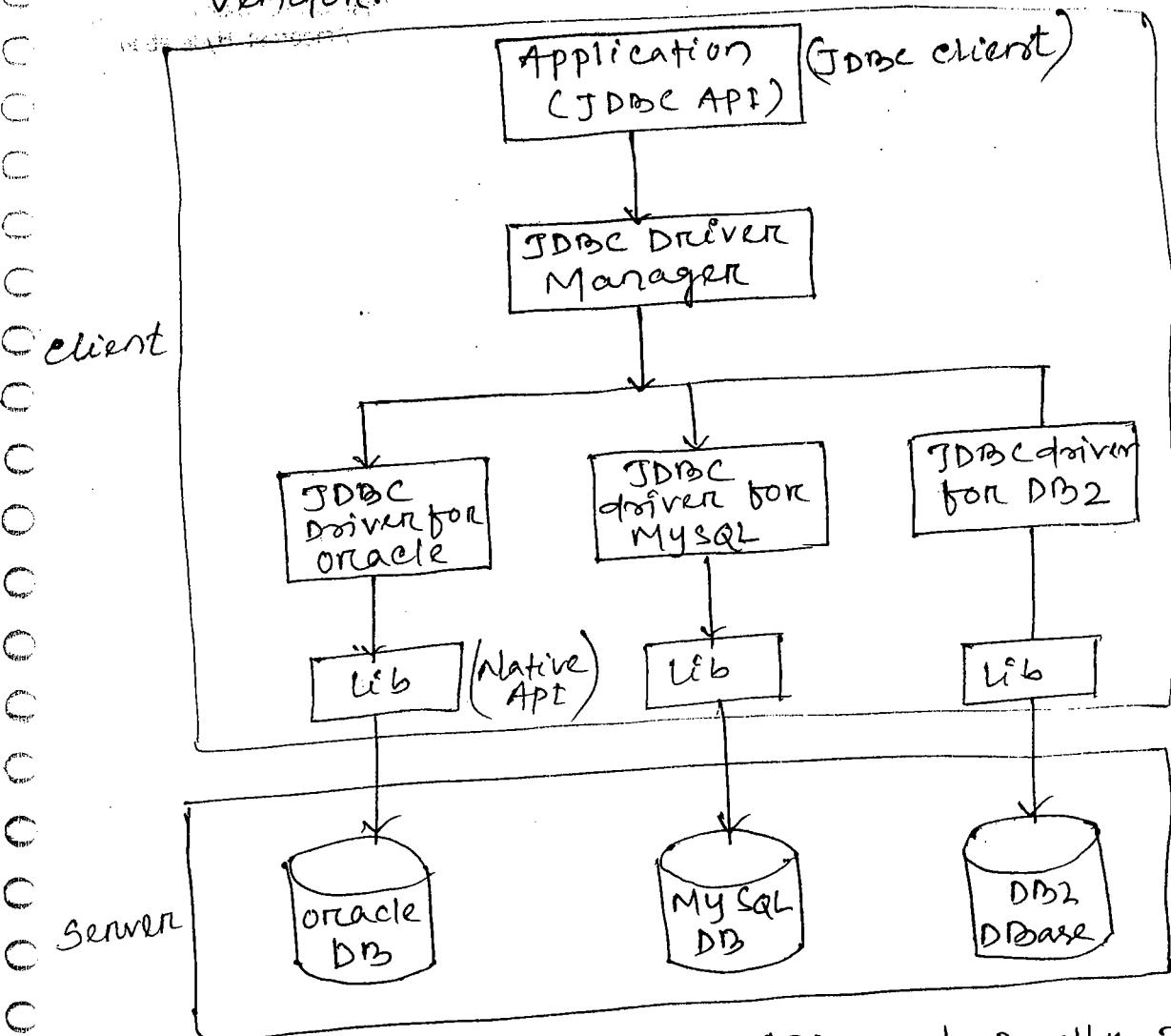


Type 1: JDBC - ODBC bridge Driver.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Type 2: Native API and partly Java Driver

- This type of driver uses a vendor specific database API to interact with the database.
- An example of such an API is Oracle OCI.
OCI → Oracle call Interface.
- These types of Drivers provided by database vendor.



Type 2: Native API and partly Java Driver.

Advantage :-

1. More efficient compare to JDBC-ODBC bridge drivers.

Disadvantage:-

1. Client Machine required native lib.
2. Native library are written in C, C++, these are platform dependent.

26.9.15

Type 3: Net Protocol All Java Driver

- This is not vendor specific and works by forwarding database requests to a remote database source using a net server component.
- How the net server component accesses the database is transparent to the client. The client driver communicates with the net server using a database independent protocol and the net server translates this protocol into database calls.
- This type of driver can access any database.
- Type-3 driver is developed based on 3-tier Architecture.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Advantage :-

Not required to install any driver at client side.

Dis-advantage

1. Not efficient, because Type-3 driver Submit SQL statements to middle Server, which again Submit these SQL statements to Database.

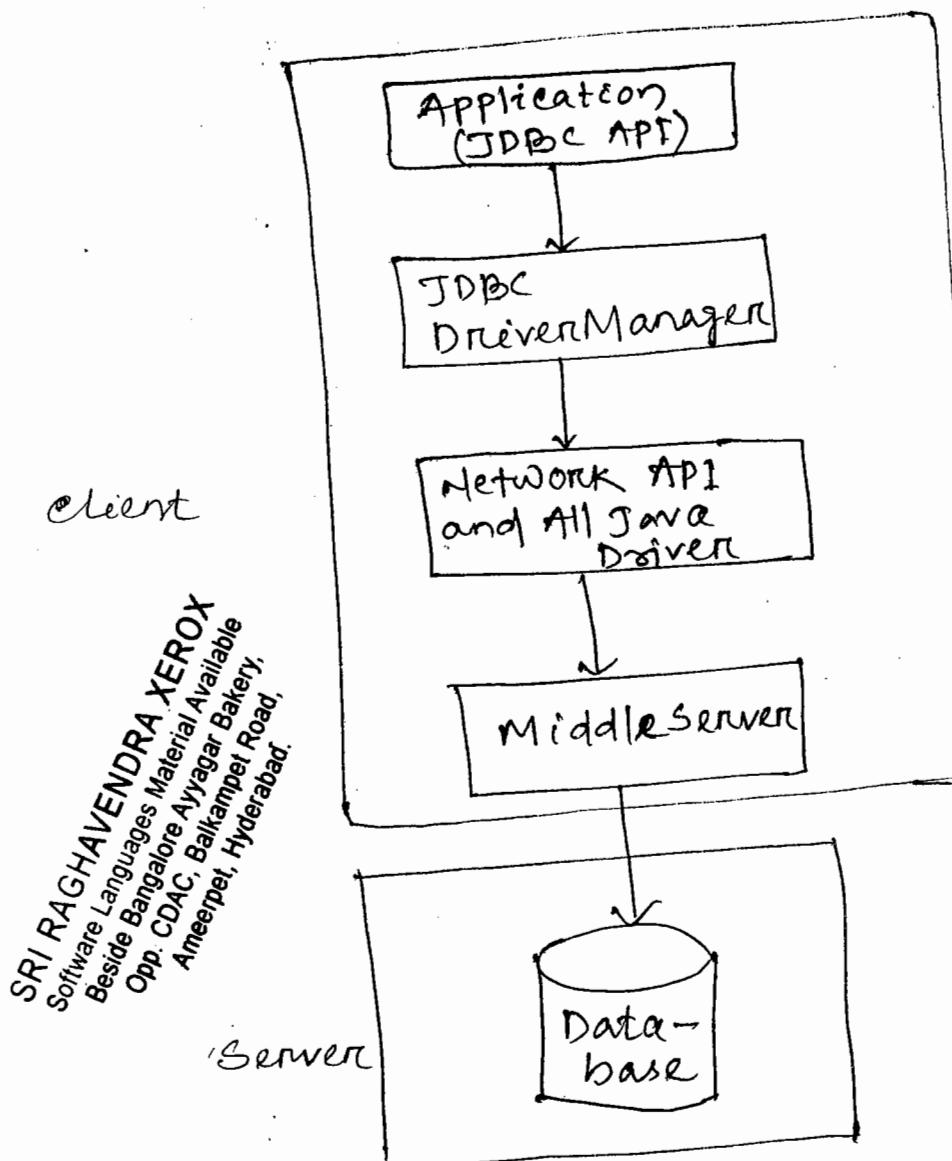
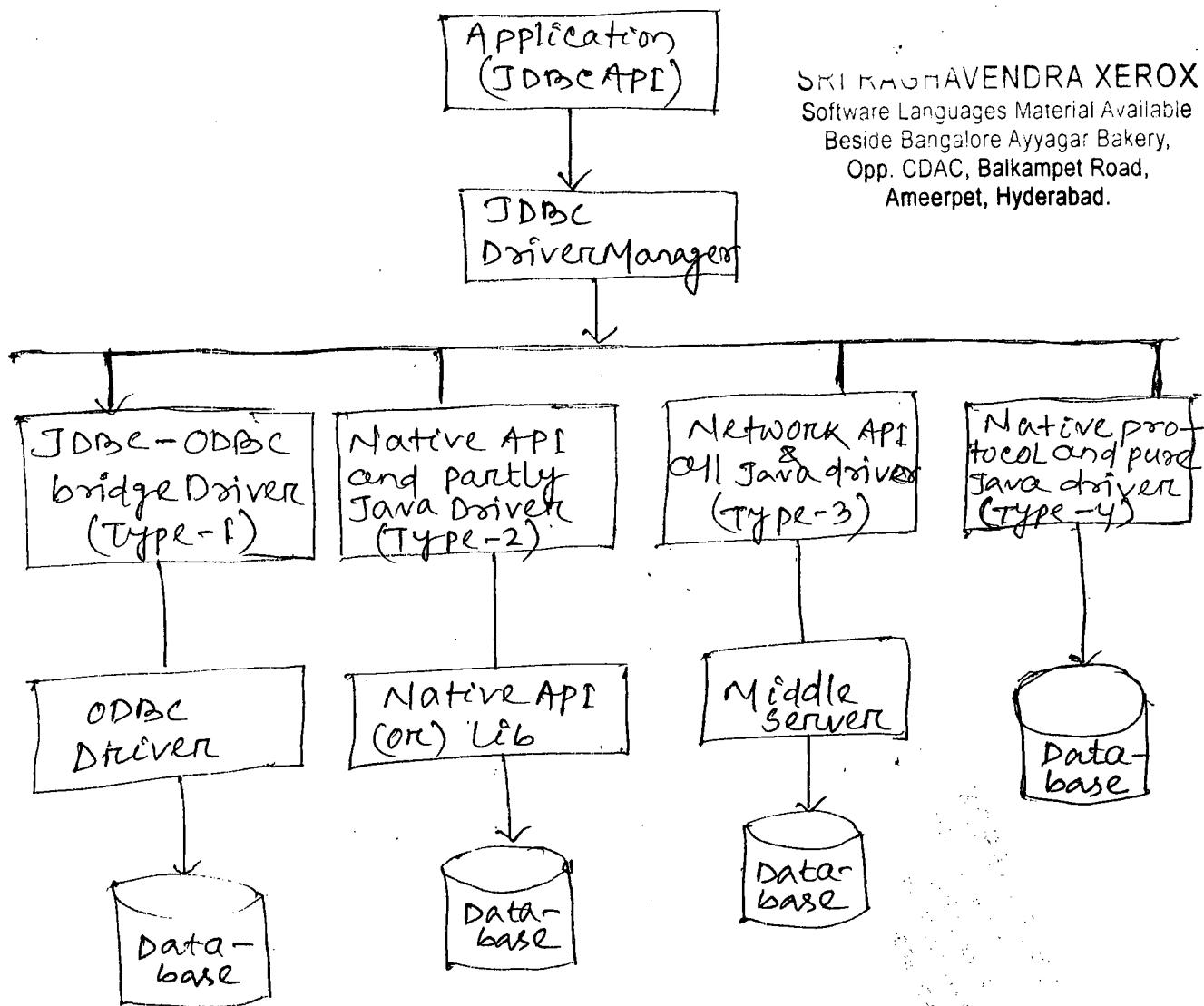


Diagram for All drivers



SRI RAHNAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

JDBC API

JDBC API :-

→ Current version of JDBC API is 4.0.

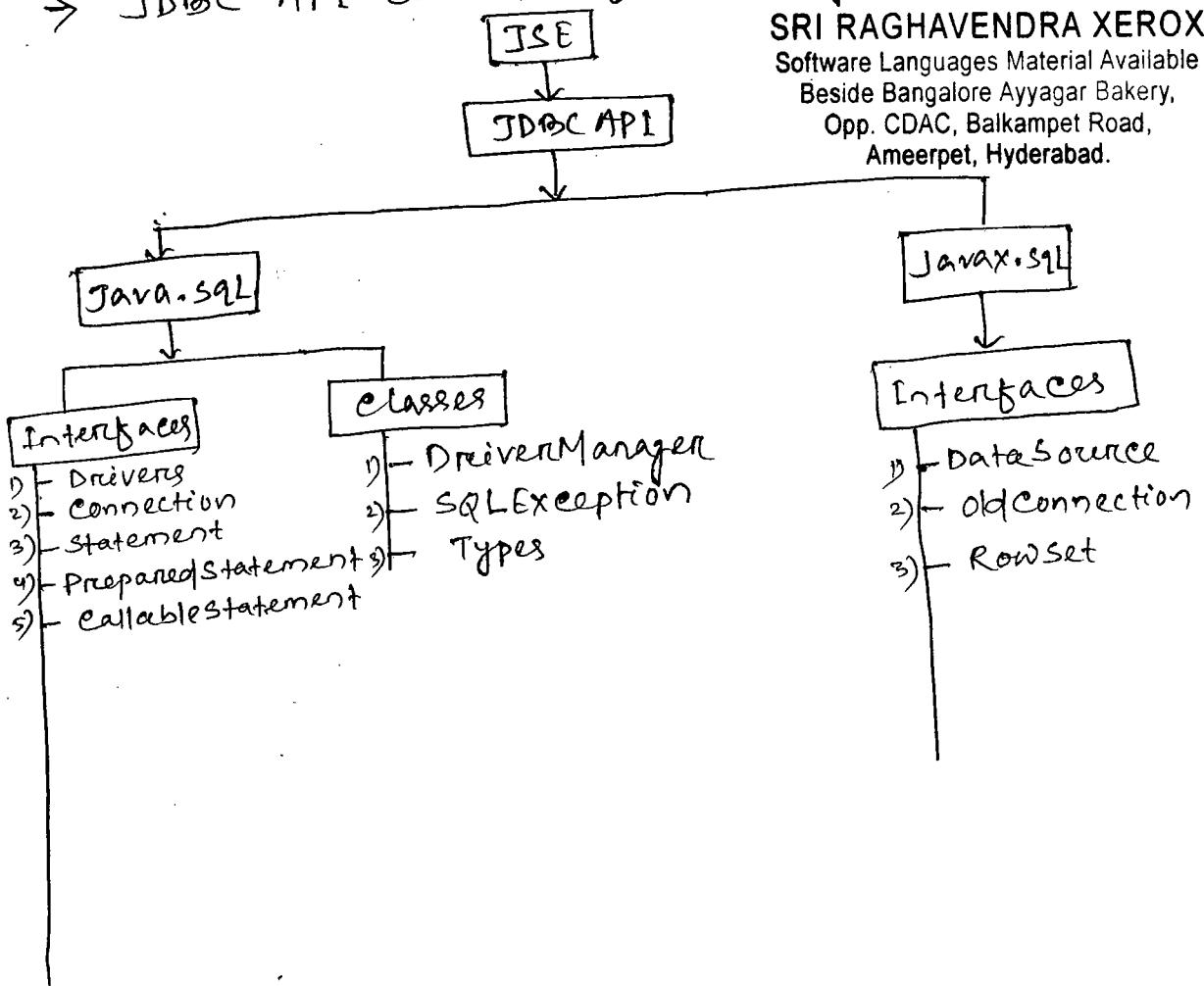
→ JDBC API is a part of JSE.

→ JDBC API 4.0 comes with JSE 6.0.

Q. What are the new features added to JDBC 4.0?

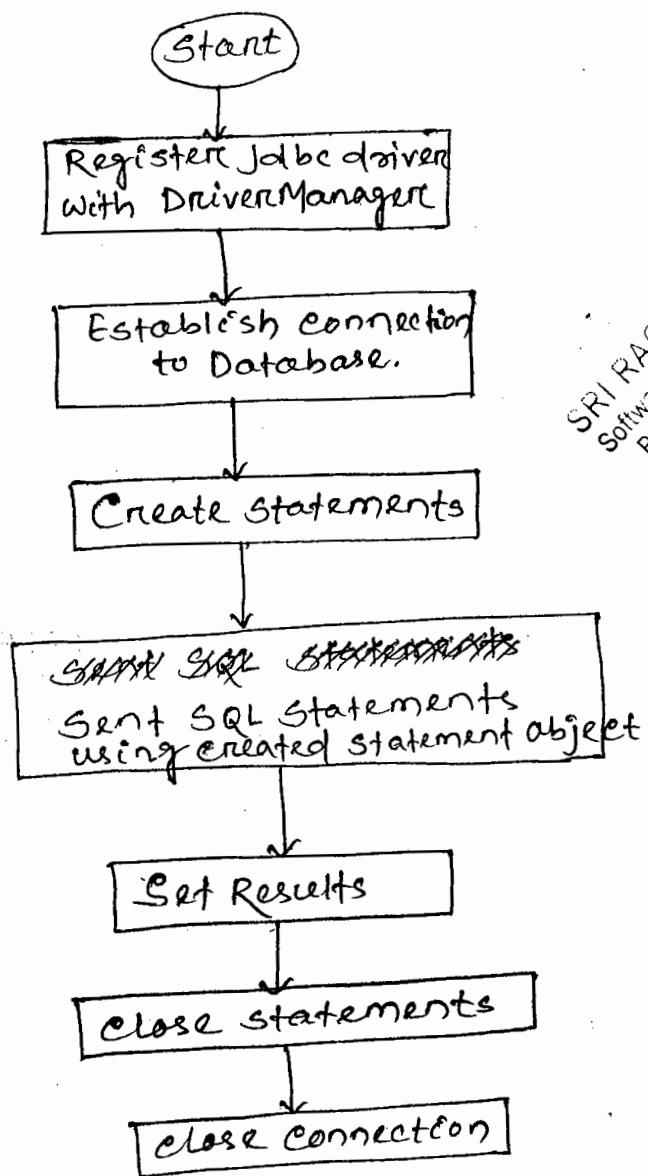
- (1) Auto loading of JDBC driver class.
- (2) Connection Management Enhancement.
- (3) Support for RowId SQL type.
- (4) Dataset implementation of SQL using Annotations.
- (5) SQL Exception handling enhancement
- (6) SQL XML Support.

→ JDBC API consists of 2 packages.



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Basic Steps for Developing JDBC Application :-



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Date:- 29.09.15.

Q. How to communicate with Oracle database using type-4 driver?

(c) Type-4 JDBC driver for Oracle database is provided by Oracle itself.

(ii) This driver comes with Oracle Software.

Ex: oracle 10g → ojdbc14.jar
oracle s/w driver

oracle 11g → ojdbc5.jar

> ojdbc6.jar

→ Objbe14.jar is developed using java 1.4.

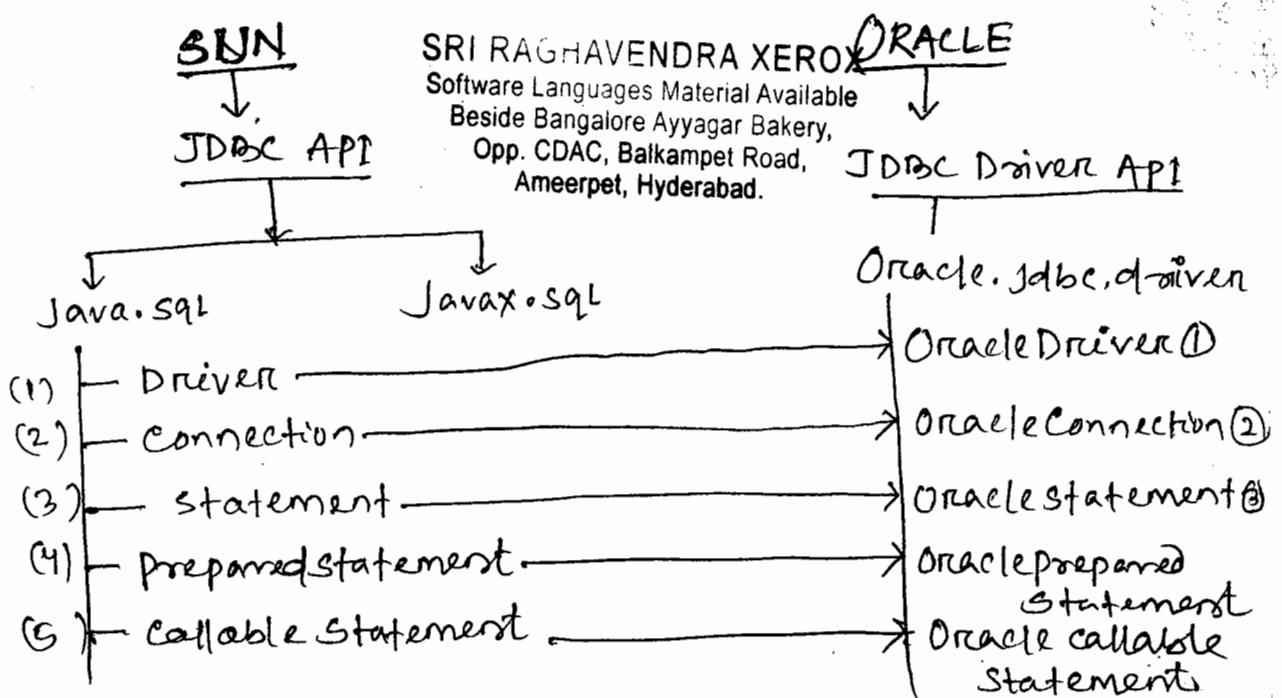
→ ojbc5.jar is developed by using java 5.0.

→ JDBC.jar is developed by using Java 6.0.

Q. What is name of driver class?

Ans:- The name of driver class is OracleDriver, this class is available in Oracle.jdbc.driver package.

→ Oracle.jdbc.driver package is available in ojdbc14.jar.



Step-1

Registering JDBC driver with driver Manager.

→ Creating JDBC driver object and giving to DriverManager is called Registering JDBC driver.

Q. What is DriverManager?

→ DriverManager is a class exist in java.sql package.

→ DriverManager who manages JDBC drivers.

→ DriverManager communicate with JDBC drivers.

→ Installing and uninstalling of JDBC driver are done by DriverManager.

JDBC Client \Rightarrow DriverManager \Rightarrow JDBC Driver \Rightarrow Database.

JDBC Client $\xrightarrow{\text{communicate}}$ JDBC DriverManager

DriverManager $\xrightarrow{\text{communicate}}$ JDBC Driver

JDBC Driver $\xrightarrow{\text{communicate}}$ Database

→ DriverManager is a class and we can't create object of this class because constructor of this class is private.

→ The Methods of this class are static.

Methods of DriverManager class are

1. Static void registerDriver(Driver driver)

→ Register the given driver with the DriverManager.

2. static void deregisterDriver (Driver driver)
→ Drops a driver from the DriverManager's list.

→ If Methods is having parameter of type Interface, it receive an object of concrete class.

→ If Methods is having return type as interface it returns object of concrete class.

Programme - 1:

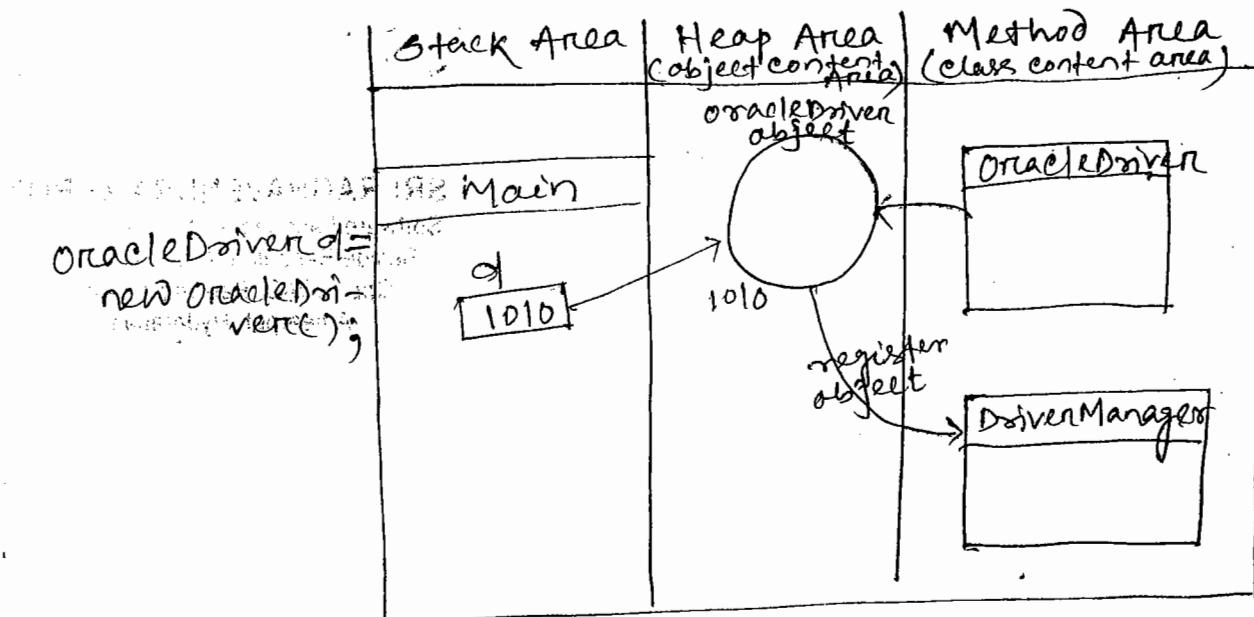
Programme to register driver with DriverManager.

```
import java.sql.*;  
import oracle.jdbc.driver.*;  
class JdbcTest1  
{  
    public static void main (String args [])  
    {  
        try  
        {  
            OracleDriver d = new OracleDriver ();  
            DriverManager.registerDriver (d);  
            System.out.println ("Driver is register with  
DriverManager");  
        }  
        catch (SQLException e)  
        {  
            System.out.println (e);  
        }  
    }  
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Set classpath = c:\oracle\product\10.2.0\db_1\jdbc\lib\ojdbc14.jar;

JVM Structure



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Battery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

30.9.15

Class :-

- class is a name of the class, which exist in Java.lang package.
- This class provides the methods for loading and creating object of given class at runtime.

The methods of this class are :-

- (i) static class forName (String className)
 - Returns the class object associated with the class or interface with the given string name.
 - This method load given class name and return the reference of loaded class.
- (ii) object newInstance()
 - create a new instance of the class represented by this class object.

→ forName method throws ClassNotFoundException if it's checked exception.

→ newInstance method throws

- (i) InstantiationException and
- (ii) IllegalAccessException.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Program-2:

Program to register driver with DriverManager
by defining driver class at run time.

```
import java.util.*;  
import java.sql.*;  
class JdbcTest2  
{  
    public static void main(String args[])  
    {  
        Scanner scan = new Scanner(System.in);  
        System.out.println("Input Driver class name");  
        String name = scan.nextLine();  
        try  
        {  
            Class c = Class.forName(name);  
            Driver d = (Driver)c.newInstance();  
            DriverManager.registerDriver(d);  
            System.out.println("driver is registered  
with DriverManager");  
        }  
        catch(ClassNotFoundException e1)  
        {  
            System.out.println("Invalid driver class name");  
        }  
        catch(InstantiationException e2)  
        {  
            System.out.println("error in creating driver object");  
        }  
    }  
}
```

SR. MR GHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
catch (IllegalAccessException e3)
{
    System.out.println ("Error in creating driver object");
}

catch (SQLException e4)
{
    System.out.println ("SQL error");
}

}
```

command prompt

```
JAVAC JdbcTest2.java
Java JdbcTest2
Error
Set classpath = _____
Input Driver class name!
oracle.jdbc.driver.OracleDriver
O/P :- Driver is register with DriverManager
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

- According to Sun Micro System every JDBC driver implementation class having static block, which contain code for creating JDBC driver object and sending to DriverManager.
- JDBC client/Application registers driver with DriverManager by loading JDBC driver class.

e.g: class OracleDriver implements Driver

```
{
```

```
static
```

```
{
```

```
OracleDriver d = new OracleDriver();
```

```
DriverManager.registerDriver(d);
```

```
}
```

SRI RAGHAVENDRA XEROX

Software Languages Material Available

Beside Bangalore Ayyagar Bakery,

Opp. CDAC, Balkampet Road,

Ameerpet, Hyderabad..

→ static block is executed for loading the class.

Program - 3:

- Q. How to load a driver class(oracleDriver) for register Driver to DriverManager.

```
import java.sql.*;
```

```
class JdbcTest3
```

```
{
```

```
    public static void main(String args[])
```

```
{
```

```
    try
```

```
{
```

```
        Class.forName("Oracle.jdbc.driver.OracleDriver")
```

```
        System.out.println("Driver register with  
        DriverManager");
```

```
}
```

```
    catch (ClassNotFoundException)
```

```
{
```

```
        System.out.println("Invalid driver class name");
```

```
}
```

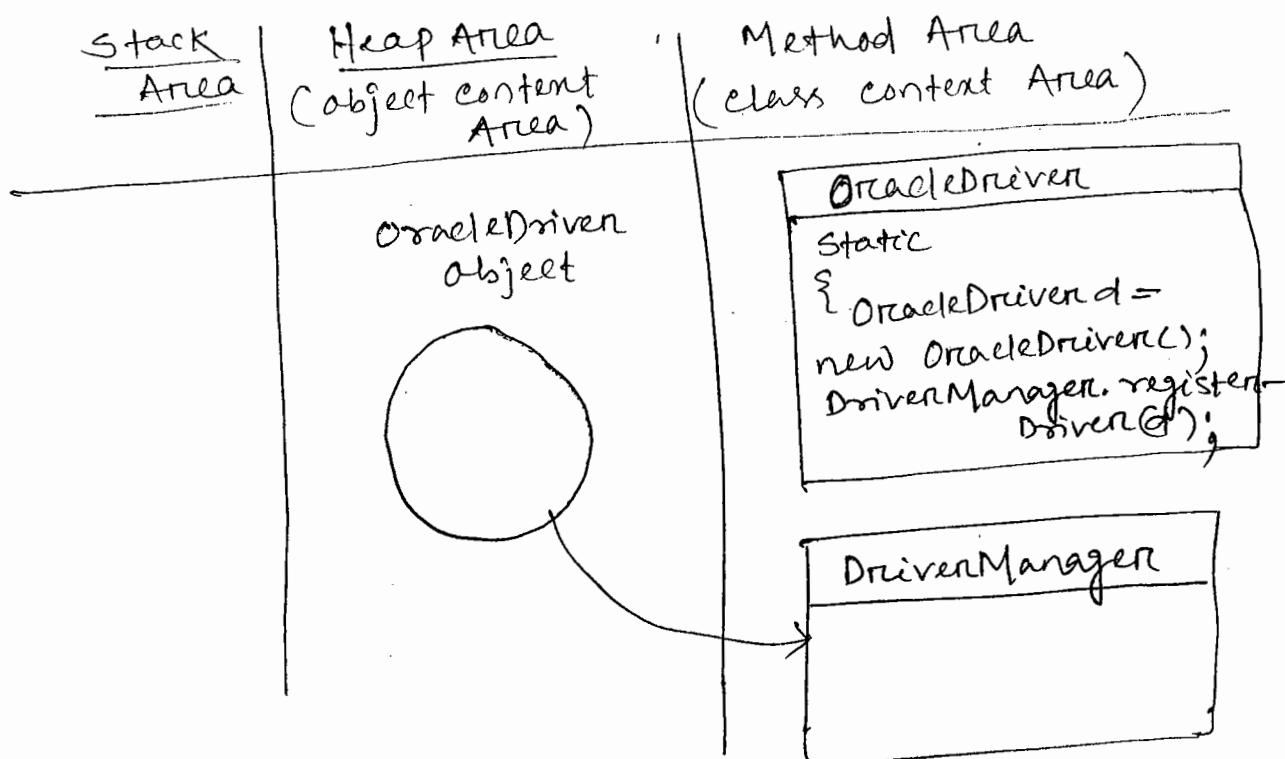
```
}}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

`Class.forName("oracle.jdbc.driver.OracleDriver");`

↓
ojdbc14.jar

JVM Structure for Loading class



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Step-2

Establishing Connection to Database :-

Q. What is URL?

→ Client establish connection to server using URL (Uniform Resource Location)

→ URL consists of

1. IP address
2. Port no.
3. Protocol.
4. Resource.

SRI RAGHAVENDRA XEROX

Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

IP Address:

→ IP Address is a system IP address where database server is running.

Port no.:-

→ Server is identify with a unique number called port no.

Protocol:-

→ protocol is used to exchange data between client and server.

Resource:-

→ Resource or programme or database where data is stored.

Syntax of URL:-

Mainprotocol : subprotocol : subprotocol : datasource

Example for JDBC URL

JDBC : Oracle

↑ Main protocol ↑ Subprotocol.

JDBC : db2

JDBC : odbc.

- Oracle ~~driver~~ provides 2 types of Driver
 - thin → it is type-4 driver.
 - oci → it is type-2 driver.

`jdbc:oracle:thin` → type4 driver

`jdbc:oracle:oci` → type 2 driver.

URL

`Jdbc: oracle: thin @ 127.0.0.1 : 1521 : XE`

↓
system IP address

↓ port no

servicename
database name

- Q. How to find port no. and database name of Oracle database.

Location:

`C:\oracle\product\10.2.0\db_1\network\ADMIN\tnsnames.ora`

- DriverManager provide the following method to establish connection to database.

(i) static connection

`get Connection (String url, String user,
String password)`

- Attempts to establish a connection to the given database URL.

- The DriverManager attempts to select an appropriate driver from the set of registered JDBC drivers based on URL.

Step 1: `Class.forName("Oracle.jdbc.driver.
OracleDriver");`

Step - 2: Reference variable

```

Connection con = DriverManager.getConnection
("jdbc:oracle:thin:@localhost:1521:Server",
"username", "password");
    
```

Driver Interface

getDriver(url)
connect
call getDriver method
call connect Method.

→ DriverManager.get-connection Method calls the connect Method of registered driver to establish connection to database.

connection:-

→ Connection is an Interface, this Interface is implemented by JDBC driver developer.

→ A connection(session) with a specific database, SQL statements are executed and results are returned within the context of a connection.

JDBC API (Interface)	JDBC OracleDriver Implements Driver.
<pre> interface Driver { Connection connect (String url); } interface Connection { Statement createStatement(); PreparedStatement prepareStatement (String sql); } </pre>	<pre> { Connection connect(String url) { { class OracleConnection implements Connection { Statement createStatement(); { { { { } } } } } } } } </pre>

SRIRAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Program :-

```
import java.sql.*;  
class JdbcTest4  
{  
    public static void main (String args[])  
    {  
        try  
        {  
            Class.forName ("oracle.jdbc.driver.OracleDriver");  
            System.out.println ("Driver register with DriverManager");  
        }
```

```
        Connection cn = DriverManager.getConnection  
        ("jdb:oracle:thin:@localhost:1521:server",  
         "SYSTEM", "Anjan",  
         "username", "password");  
    }
```

```
    System.out.println ("Established connection to database");  
    Statement st = cn.createStatement();  
    cn.close();
```

```
    Driver d = DriverManager.getDriver
```

```
    ("jdb:oracle:thin:@localhost:1521:server");  
    DriverManager.deregisterDriver(d);
```

```
    System.out.println ("driver is removed from DriverManager");  
}
```

```
    catch (ClassNotFoundException e)  
    {
```

```
        System.out.println ("Invalid driver class name");  
    }
```

```
        catch (SQLException e2)
    {
        sopln(e2);
    }
}
```

02.10.15
Q. Programme to read drivers information from
DriverManager?

Programme :-

```
import java.sql.*;
import java.util.*;
class JDBCTest5
{
    public static void main(String args[])
    {
        try
        {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            Enumeration<Driver> e = DriverManager.getDrivers();
            while (e.hasMoreElements())
            {
                Driver d = e.nextElement();
                System.out.println(d);
            }
        }
        catch (ClassNotFoundException e1)
        {
            sopln("Invalid driver class name");
        }
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

How to Send SQL statements to database? -

- JDBC provide 3 types of statement objects in order to send SQL statements / command to database.

KEY POINTS :- 1. Statement.

2. PreparedStatement.

3. CallableStatement.

Statement :

→ Statement is an Interface. This Interface is implemented by JDBC driver developer.

→ It is used to send static SQL statement to database.

Q. What is static SQL statement.

→ It is an SQL statement with value.

→ An SQL statement which compiled every time before execution.

How to create statement object? -

- Connection provides the following method which return statement object.

Statement createStatement()

- creates a Statement object for sending SQL statements to database.

programme to create statement.

```
import java.sql.*;
```

```
class JdbcTest6
```

```
{
```

```
    public static void main (String args [])
```

```
    {
        try
```

```
{
```

```
    Class.forName ("oracle.jdbc.driver.OracleDriver");
```

```
    Connection cn = DriverManager.getConnection
```

```
    ("jdbc:oracle:thin:@localhost:1521:XE", "system",  
     "Ayan") ;
```

```
    System.out.println ("connection Established");
```

```
    Statement st = cn.createStatement();
```

```
    System.out.println ("statement created");
```

```
    st.close();
```

```
    cn.close();
```

```
}
```

```
    catch (ClassNotFoundException e1)
```

```
{
```

```
    System.out.println ("Invalid driver class name");
```

```
}
```

```
    catch (SQLException e2)
```

```
{
```

```
    System.out.println (e2);
```

```
}
```

```
}}
```

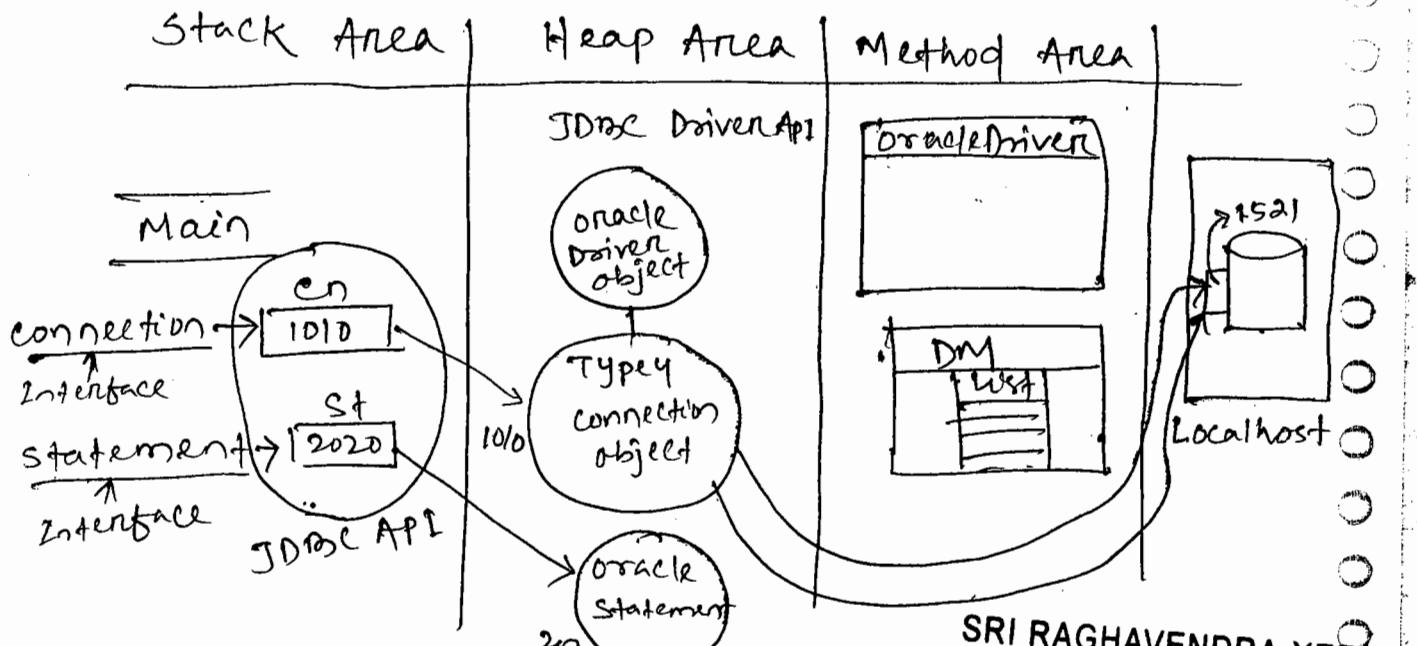
SRI RAGHAVENDRA XEROX

Software Languages Material Available

Beside Bangalore Ayyagar Bakery,

Opp. CDAC, Balkampet Road,

Ameerpet, Hyderabad.



Method of statement

1. `int executeUpdate(String SQL)`

→ Execute the given SQL statement which may be an INSERT, UPDATE or DELETE statement or an SQL statement that returns nothing such as an SQL DDL statement.

Program to register User (inserting user details into database table).

Table create

```
SQL> create table User.User_reg (name varchar2(20),
                                         uname varchar2(20),
                                         pwd varchar2(20),
```

Table created.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

C program:- import.sql.*;
C class UserRegister
C {
C     public main( String args[])
C     {
C         Connection cn = null;
C         Statement st = null;
C         try
C         {
C             Class.forName("oracle.jdbc.driver.OracleDriver");
C             cn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:XE","system","Anjan");
C             st = cn.createStatement();
C             int count = st.executeUpdate("insert into
C                 user_reg values('Anjan','Jena','123456')");
C             System.out.println(count);
C         }
C         catch (ClassNotFoundException e)
C         {
C             System.out.println("Invalid Driver class");
C         }
C         catch (SQLException s)
C         {
C             System.out.println(s);
C         }
C         finally
C         {
C             try
C             {
C                 if (st != null)
C                     st.close();
C                 if (cn != null)
C                     cn.close();
C             }
C             catch (SQLException e1)
C             {
C                 System.out.println(e1);
C             }
C         }
C     }
C }

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

03.10.15

programme to register user by reading values at run time.

```
import java.util.*;
import java.sql.*;
class UserRegistration
{
    public static void main(String args[])
    {
        connection cn = null;
        statement st = null;
        try
        {
            class.forName("oracle.jdbc.driver.
                           OracleDriver");
            cn=DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521:
                          XE", "system", "Anjan");
            st = cn.createStatement();
            Scanner scan = new Scanner(System.in);
            System.out.print("Input Name:");
            String name = scan.next();
            System.out.print("Input Username:");
            String uname = scan.next();
            System.out.print("Input Password:");
            String pwd = scan.next();
        }
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

String cmd = "Insert into user-reg values
              ('" + name + "','" + uname + "','" + t
               + pwdt + "');");
int count = st.executeUpdate(cmd);
System.out.println(count);

}

catch (ClassNotFoundException e1)
{
    System.out.println("Invalid driver class name");
}

catch (SQLException e2)
{
    System.out.println(e2);
}

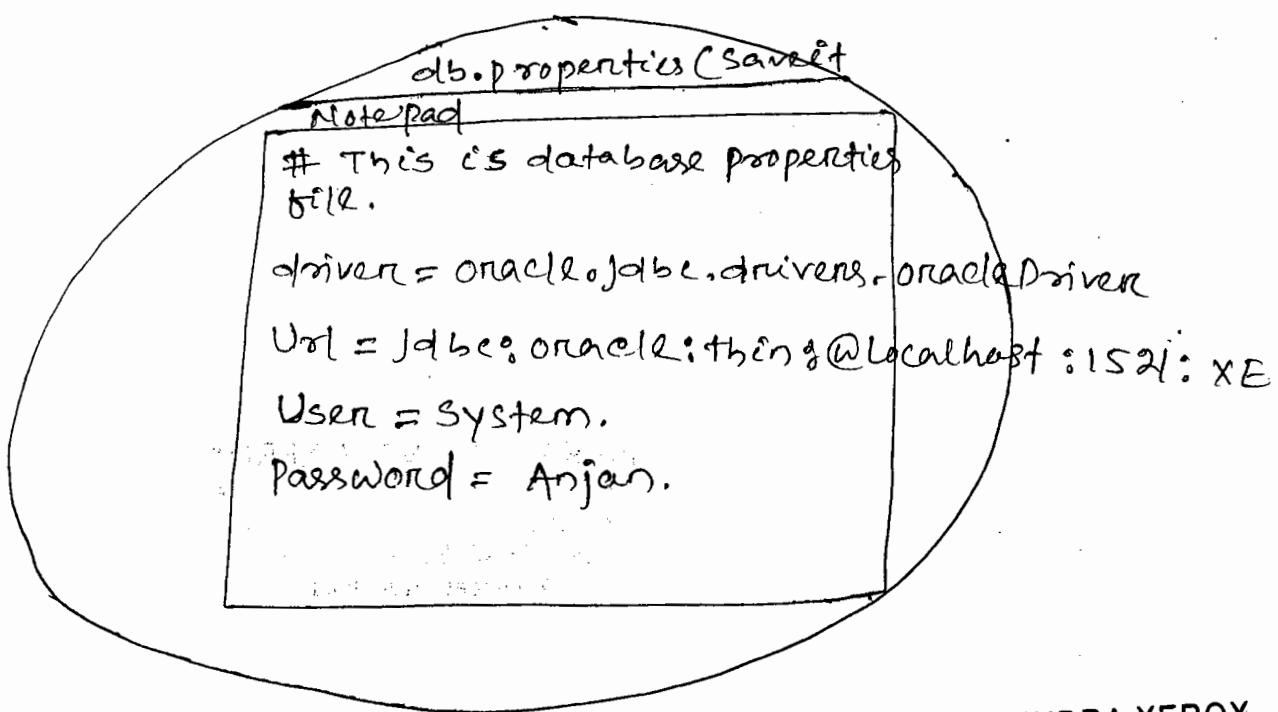
finally
{
    try
    {
        if (st != null)
            st.close();
        if (cn != null)
            cn.close();
    }
    catch (SQLException e3)
    {
        System.out.println(e3);
    }
}
}
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Creating Database properties File :-

- Database properties file is created to avoid hardcoding database details (drivername, url, username and password) inside program.



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

5-10-15

// programme to delete user.

For reading file content.

→ Properties File:-

```
driver = oracle.jdbc.driver.OracleDriver  
url = jdbc:oracle:thin:@localhost:1521:XE  
user = System  
password = Anjan
```

// program to update password or change password.

```
import java.sql.*;  
import java.util.*;  
import java.io.*;  
class changePassword  
{  
    public static void main(String args[]){  
        connection cn=null;  
        statement st=null;  
        fileInputStream fis=null;  
        try{  
            fis=new fileInputStream ("rdb.properties");  
            Properties p=new Properties();  
            p.load(fis);  
            class.forName(p.getProperty("driver"));  
            cn=DriverManager.getConnection(p.getProperty("url"),p);  
            st=cn.createStatement();  
        }  
    }  
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
Scanner scan = new Scanner(System.in);
System.out.println("Input username");
String u = scan.next();
System.out.println("Input password");
String ps = scan.next();
String SQLcmd = "update user_reg set";
set pwd = "'"+ps+"'";
where uname = "'"+u+"'";
int count = st.executeUpdate(SQL);
if(count==0)
    System.out.println("Invalid user name");
else
    System.out.println("password changed");
}
catch (ClassNotFoundException e1)
{
    System.out.println("Invalid driver name");
}
catch (SQLException e2)
{
    System.out.println(e2);
}
catch (IOException e3){{}
    System.out.println(e3);
}
finally
{
    try
    {
        if(st!=null)
            st.close();
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet, Hyderabad.
Ameerpet, Hyderabad.

```
if (cn != null)
    cn.close();
if (fis != null)
    fis.close();
}
catch (SQLException ey)
{
    sopin(ey);
}
catch (IOException es)
{
    sopin(es);
}
```

333

end prompt

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Developing database utility class (DBUtil class):-

→ This class is developed by programmer to avoid redundant code.

(i) Class.forName

(ii) DriverManager.getConnection

→ This class provide one static method which register driver and establish connection to database and return database connection object.

SQL constants :-

→ This class is developed avoid hard coding SQL statements.

→ This class is collection of constants.

// How to create DB util class:-

```
Package com.nit.db;
import java.sql.*;
public class DBUtil
{
    public static Connection getConnection() throws
        SQLException, ClassNotFoundException
    {
        Class.forName("oracle.jdbc.driver.OracleDriver");
        Connection cn = DriverManager.getConnection
            ("Jdbc:oracle:thin:@localhost:1521:XE", "system",
             "Anjan");
        return cn;
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

For compiler:- Java -> . is for current directory

Program

How to delete User.

```
import com. net. db. DBUtil;
import java. sql.*;
import java. util.*;
class DeleteUser
{
    public static void main (String args[])
    {
        Connection cn;
        Statement st;
        try
        {
            cn = DBUtil. getConnection();
            st = cn. createStatement();
            Scanner scan = new Scanner (System. in);
            System. out. println ("Input Username");
            String u = scan. next();
            System. out. println ("Input Password");
            String p = scan. next();
            String sql = "delete from userreg  

where uname = " + u + " and  

pwd = " + p;
            int count
            int count = st. executeUpdate(sql);
            if (count == 0)
                System. out. println ("Invalid Username or Pwd");
            else
                System. out. println ("User deleted");
        }
        catch (ClassNotFoundException e)
        {
            System. out. println ("Invalid Driver class name");
        }
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

    catch (SQLException e2)
    {
        sopln(e2);
    }
    finally
    {
        try
        {
            if (st != null)
                st.close();
            if (cn != null)
                cn.close();
        } catch (SQLException e3)
        {
            sopln(e3);
        }
    }
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

333

end prompt

```

java -d. DBUtil.java
java DeleteUser.java
Java DeleteUser.

```

06.10.15

Reading data from database table.

Statement provides 3 methods to send SQL statement.

(1) executeUpdate → DML, DDL

(2) executeQuery → DQL

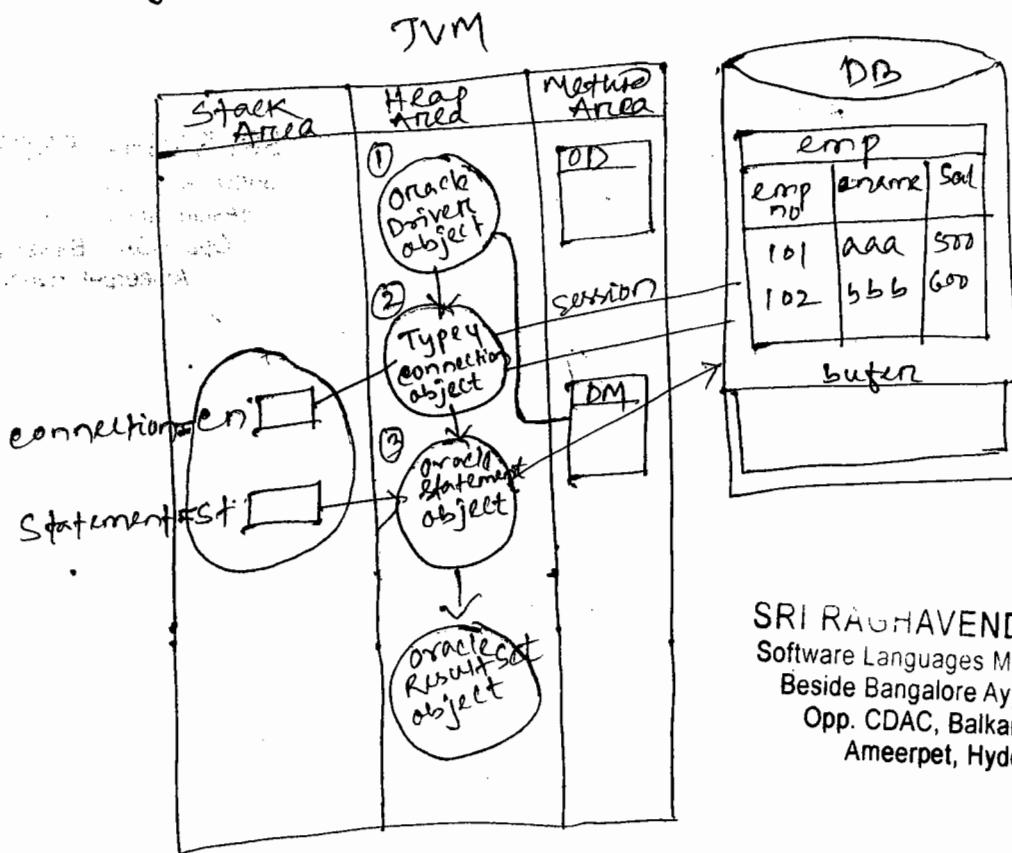
(3) execute → DML, DDL, DQL

Statement provide the following method to send SELECT statement to database

→ ResultSet executeQuery(String SQL)

→ SQL → an SQL statement to be sent to the database,
typically a static SQL SELECT Statement.

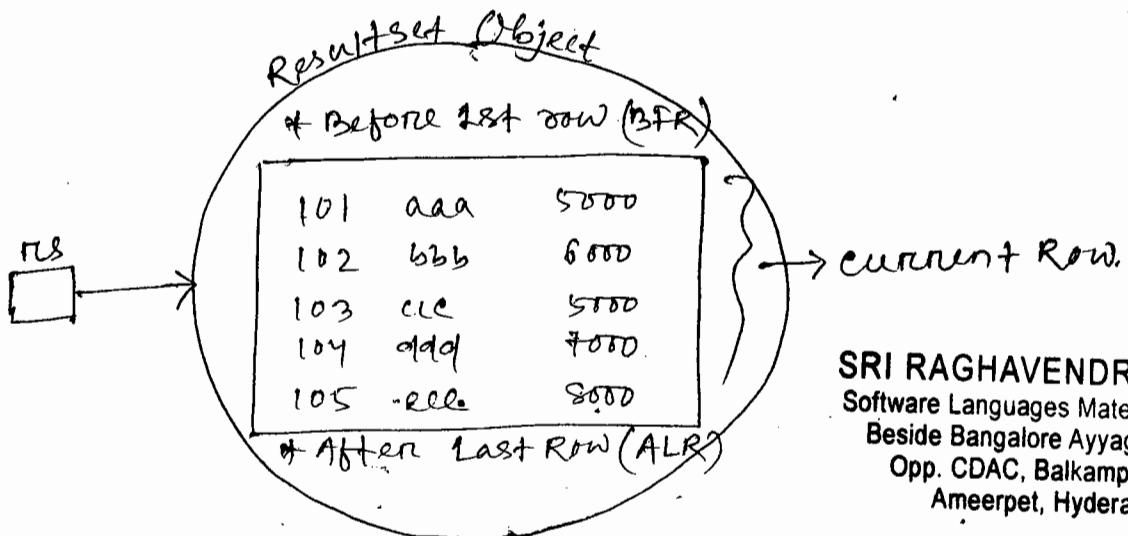
→ Execute the given SQL Statement which returns
a single ResultSet object.



SRI RAHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Resultset:-

- Resultset is an Interface.
- This Interface is implemented by JDBC driver developer.
- Resultset object contain result of executed Query.
- A table of data representing a database result set, which is usually generated by executing a statement that queries the database



SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

- A Resultset object maintains a cursor pointing to current row of data.
- Initially the cursor is position before the first.

Navigation Methods

Return type	Method Name
boolean	next()
boolean	previous()
boolean	first();
boolean	last();
boolean	absolute(int-row)
boolean	relative(int-row)
void	beforeFirstRow()
void	afterLastRow()

- A default Resultset object is not updatable and has a cursor that moves forward only. Thus, you can iterate through it only once and only from the first row to the last row.
- Resultset provide getter method to read data from resultset object.
- Resultset having 2 versions of getter method.
 - 1. getter method with column Index.
 - 2. getter method with column Name.
- int getInt (int columnIndex)
- int getInt (String columnName)
- Returns the value of designated column in the current row of this Resultset object as an int in the java programming language.
 - String getString (int columnIndex)
 - String getString (^{String} columnName)

Q. what is default fetch size of Oracle Resultset ?

Ans:- 10 Rows.

Q. How to find Fetch size ?

Ans:- getFetchSize(); → Method.

→ Statement provide following Method.

int getFetchSize()

→ Returns the number of result set rows that is the default fetch size

void setFetchSize(int rows)

→ Gives the JDBC driver a hint as to the number of rows that should be fetched from the database when more rows are needed.

Write a program to find Fetch size?

```
import java.sql.*;  
class FindFetchSize  
{  
    public static void main (String args[]) throws  
        Exception  
    {  
        Class.forName("oracle.jdbc.driver.Oracle-  
                        Driver");  
        Connection cn = DriverManager.getConnection  
            ("jdbc:oracle:thin:@localhost:1521:XE",  
                "System", "Anjan");  
        Statement st = createStatement();  
        // ResultSet rs = st.executeQuery("Select * from user_rg");  
        System.out.println(st.getFetchSize());  
        System.out.println(st);  
        cn.close();  
        st.close();  
    }  
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

// Programme for reading data from database.
// WAP to read data from database table.

```
import java.sql.*;  
class ResultSetTest1  
{  
    public void main(String args[]){  
        Connection cn=null;  
        Statement st=null;  
        ResultSet rs=null;  
        try{  
            cn=DriverManager.getConnection  
                ("jdbc:oracle:thin:@localhost:1521:XE","system",  
                 "Anjas");  
            st=cn.createStatement();  
            rs=st.executeQuery("select empno, ename,  
                               sal from emp");  
            while(rs.next())  
            {  
                int eno=rs.getInt(1);  
                String en=rs.getString(2);  
                float es=rs.getFloat(3);  
                System.out.printf("%d %s %.2f",eno,en,es);  
            }  
        }catch(SQLException e){}  
        finally{  
            try{  
                if(rs!=null)  
                    rs.close();  
            }  
            catch(Exception e){}  
        }  
    }  
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
    if (st != null)
        st.close();
    if (en != null)
        en.close();
}
catch (SQLException e2) {
}
}
```

program to verify Username and Password
or
program to login.

```
import java.sql.*;  
import java.util.*;  
  
class Login  
{  
    public static void main (String args[])  
{  
        connection cn = null;  
        Statement st = null;  
        ResultSet rs = null;  
  
        try  
        {  
            Class.forName ("oracle.jdbc.driver.Oracle-  
Driver");  
  
            cn = getconnection  
            cn = DriverManager.getConnection  
            "abc:oracle:thin:@localhost:1521:X.E", "System",  
            "Arijan");  
  
            st = cn.createStatement();  
        }  
    }  
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
Scanner scan = new Scanner(System.in);
System.out.println("Input username:");
String u = scan.next();
System.out.println("Input password:");
String p = scan.next();
String sql = "Select count(*) from user_xrg  
where uname = " + u + " and pwd = " + p;
System.out.println(sql);
Statement st = con.createStatement();
ResultSet rs = st.executeQuery(sql);
rs.next();
int c = rs.getInt(1);
if (c == 0)
    System.out.println("Invalid uname and pwd");
else
    System.out.println("Welcome to my Application");
}
catch (ClassNotFoundException e1)
{
    System.out.println("Invalid driver class name");
}
catch (SQLException e2)
{
    System.out.println(e2);
}
finally
{
    try
    {
        if (rs != null)
            rs.close();
        if (st != null)
            st.close();
        if (con != null)
            con.close();
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
        catch (SQLException e3)
    {
        System.out.println(r3);
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Result Set Types :-

→ There are 3 types of result sets.

1. TYPE_FORWARD_ONLY
 2. TYPE_SCROLL_SENSITIVE
 3. TYPE_SCROLL_INSENSITIVE

Q: what is a difference b/w TYPE_SCROLL_INSENSITIVE
and TYPE_SCROLL_SENSITIVE?

• TYPE-SCROLL-INSENSITIVE :-
 TYPE-SCROLL-SENSITIVE

~~SCROLL-INSENSITIVE~~ • TYPE-SCROLL-SENSITIVE :-

- TYPE - SCROLL

1. An insensitive resultset is like the snapshot of the data in the database when query was executed.

was executed.

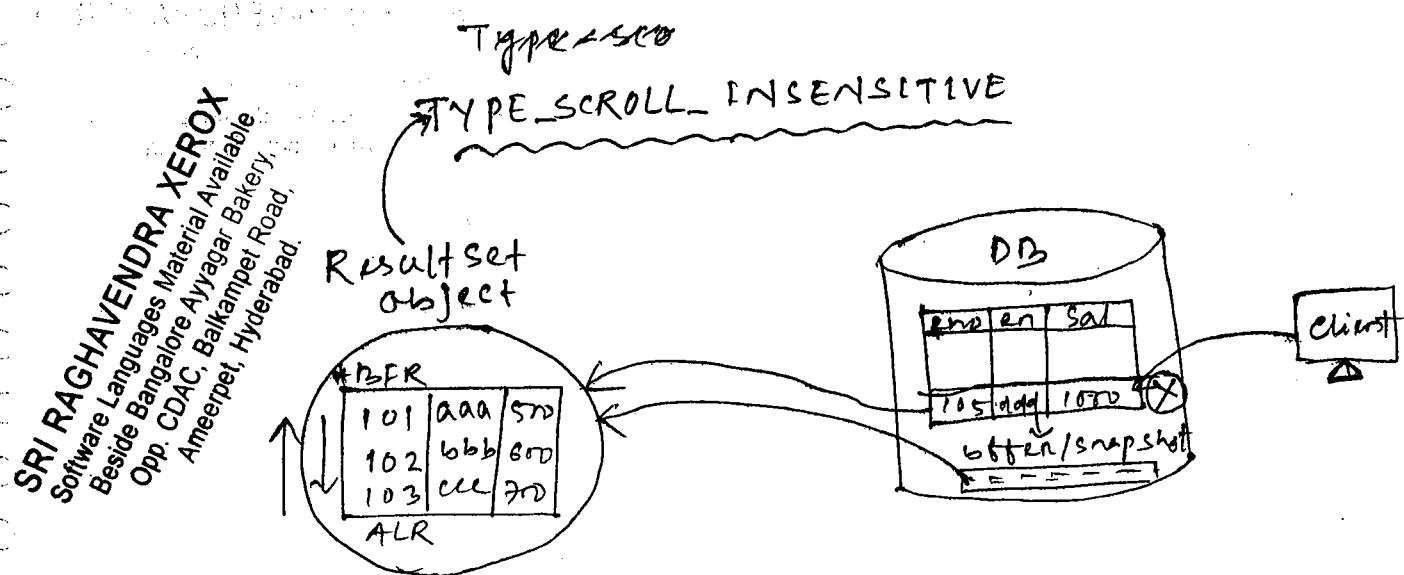
ANSWER : A sensitive result set does not represent a snapshot of data, rather it contains points to those rows which satisfy the query condition.

2. After we get the result set the changes made to data are not visible through the result set, and hence they are known as insensitive.

2. After we obtain the resultSet if the data is modified then such modifications are visible through resultSet.

3. performance not effected with insensitive

3. since a trip is made for every 'get' operation, the performance drastically get affected.



ResultSet Mode :-

- Resultset having two modes
 - 1. CONCUR_READ_ONLY (read only)
 - 2. CONCUR_UPDATABLE (read & do changes)
- How to change the ResultSet type and mode?
 - con.createStatement();
 - con.createStatement(type, mode);
- connection provid the following method to change ResultSet type and Mode.
 - Statement createStatement(int resultSetType, int resultSetConcurrency)

create a statement object that will generate ResultSet objects which with the given type and concurrency.

Example

Statement st = cn.createStatement(ResultSet.

TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_READ_ONLY);

08.10.15

Absolute (Method of result set)

boolean absolute (int row)

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

- Moves the cursor to the given row number in this ~~resultset~~ resultset object.
- If the row number is +ve the cursor moves to the given row number with respect to the beginning of the result set. The first row is row 1, the second row 2 and so on.
- If the row number is -ve, the cursor moves to an absolute row position with respect to the end of the result set. For example:—
calling the method absolute (-1) positions the cursor on the last row; calling the method absolute (-2) moves the cursor to the next to last row in backward direction.

program to read a row from ResultSet using
absolute method.

```
import java.sql.*;
import java.util.*;
class Absolute AbsoluteTest1
{
    public static void main (String args[])
    {
        Connection cn = null;
        Statement st = null;
        ResultSet rs = null;
        try
        {
            Class.forName ("oracle.jdbc.driver.Oracle-
                           Driver");
            cn = DriverManager.getConnection
                ("jdbc:oracle:thin:@localhost:1521:XE", "system",
                 "Anjan");
            st = cn.createStatement (ResultSet.TYPE_SCROLL-
                                   SENSITIVE, ResultSet.CONCUR_READ_ONLY);
            st.setFetchSize(20);
            rs = st.executeQuery ("select empno, ename,
                                  sal from emp");
            Scanner scan = new Scanner (System.in);
            System.out.println ("Input row number");
            int row = scan.nextInt();
            boolean b = rs.absolute (row);
            if (b == false)
                System.out.println ("Invalid row number");
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

else
{
    int eno = rs.getInt(1);
    String en = rs.getString(2);
    float s = rs.getFloat(3);
    System.out.printf("int%d.%s.%f", eno, en, s);
}
}

catch (ClassNotFoundException e1)
{
    sopln("invalid driver class name");
}

catch (SQLException e2)
{
    sopln(e2);
}

//relative method of fetch/get
//broken relative (X)X(X)
// moves cursor & relative position of row either
// forward or backward
// Finally
if (rs != null)
    rs.close();
if (st != null)
    st.close();
if (con != null)
    con.close();
}

catch (SQLException e3)
{
    sopln(e3);
}
}
}

```

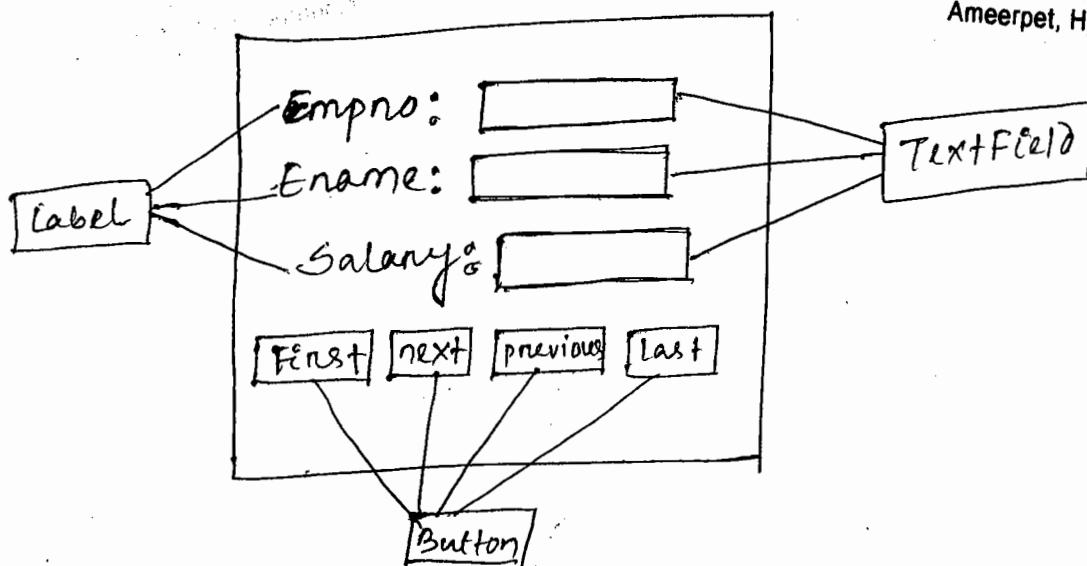
SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

'relative' (method of resultset)
boolean relative (int row)

- Moves cursor a relative number of row either positive (+ve) or negative (-ve).
- Attempting to move beyond the first/last row in the result set positions the cursor before/after the first/last row.
- calling relative method, if cursor is on before first row or after last row gives ^{SQL} exception.

Developing Navigation Window

AWT/Swing



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

import java.sql.*;
import java.awt.*;
import java.awt.event.*;
class NavigationWindow extends Frame implements
ActionListener
{
    connection cn;
    Statement st;
    ResultSet rs;
    Label l1,l2,l3;
    TextField t1,t2,t3;
    Button b1,b2,b3,b4;
}

```

```

NavigationWindow()
{
try {
    Class.forName("oracle.jdbc.driver.OracleDriver");
    cn = DriverManager.getConnection("jdbc:oracle:
    :thin:@localhost:1521:XE","system","Anjan");
    st = cn.createStatement
    (ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.
    CONCUR_READ_ONLY);
    rs = st.executeQuery("select empno, ename,
    sal from emp");
}
catch (ClassNotFoundException e1) {}
catch (SQLException e2) {}
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

```

l1 = new Label ("EmpNo");
l2 = new Label ("EName");
l3 = new Label ("Salary");
t1 = new TextField(10);
t2 = new TextField(10);
t3 = new TextField(10);
b1 = new Button ("First");
b2 = new Button ("Next");
b3 = new Button ("Previous");
b4 = new Button ("Last");
b1.addActionListener(this);
b2.addActionListener(this);
b3.addActionListener(this);
b4.addActionListener(this);
GridLayout g = new GridLayout(5, 2)
setLayout(g);
add(l1);
add(t1);
add(l2);
add(t2);
add(l3);
add(t3);
add(b1);
add(b2);
add(b3);
add(b4);
}
public void actionPerformed(ActionEvent e)
{
    String c = e.getActionCommand();
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

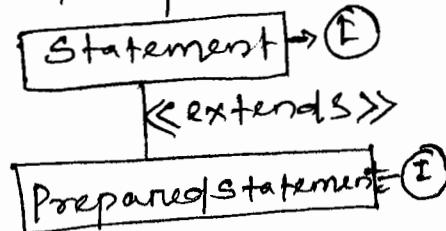
```
try
{
    if (c.equals("first"))
        rs.first();
    if (c.equals("next"))
        rs.next();
    if (c.equals("previous"))
        rs.previous();
    if (c.equals("last"))
        rs.last();
    t1.setText(String.valueOf(rs.getInt(1)));
    t2.setText(rs.getString(2));
    t3.setText(String.valueOf(rs.getFloat(3)));
}
catch (SQLException k) {}

public static void main (String args[])
{
    NavigationWindow w = new NavigationWindow();
    w.setSize (200, 200);
    w.setVisible (true);
}
```

9.10.15

PreparedStatement :-

- PreparedStatement is an Interface
- PreparedStatement interface is implemented by JDBC driver developer.



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

- An object that represents a precompiled SQL statement.
- A SQL Statement is precompiled and stored in a PreparedStatement object. This object can then be used to efficiently execute this statement multiple times.

Q) what is the difference b/w Statement and Prepared Statement?

<u>Statement</u>	<u>PreparedStatement</u>
(i) A standard statement is used to create Java representation of a literal SQL statement and execute it on the database.	(i) A preparedstatement is precompiled statement. This means that when the preparedstatement is executed, the RDBMS can just run the preparedstatement SQL statement without having to compile it first.
(ii) Statement has to verify its metadata against the database every time.	(ii) While a prepared has to verify its <u>Metadata</u> against the database only once.
(iii) If you want to execute the SQL statement ^{only} once go for STATEMENT.	

(v) Statement doesn't allow parameters

AVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

(vi) If you want to execute a single SQL Statement multiple number of time then go for PreparedStatement.

(vii) Prepared statement allows parameters.

DisAdvantage of Statement :-

1. Not efficient
→ (Not efficient means it needs to compile each time when it work to execute.)
2. SQL Injection problem.

How to Create PreparedStatement object :-

→ Connection provide the following method which return PreparedStatement object.

`PreparedStatement preparedStatement (String sql)`
→ create a prepared statement object for sending parameterized SQL statements to the database

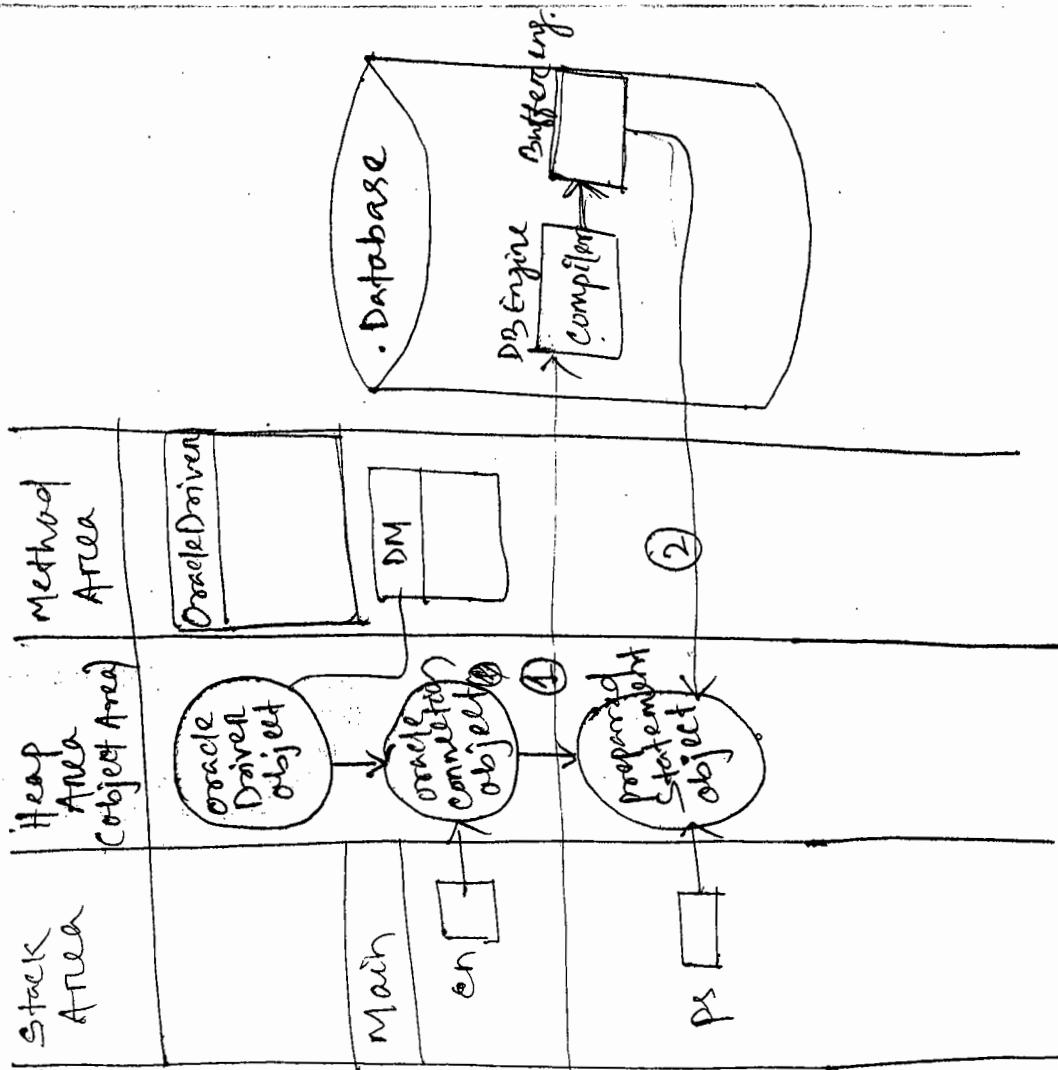
```

① Class.forName("oracle.jdbc.driver.
OracleDriver");

② connection en = DriverManager.
getConnection ("jdbc:oracle:thin:@local-
host:1521:XE", "system", "Anjan");

③ preparedStatement ps =
en.prepareStatement("select
* from emp");
ps.executeQuery();

```



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Opp. CDAC, Ayyagar Bakery,
Opp. CDAC, Ballampet Road,
Ameerpet, Hyderabad.

Q) What is SQL Injection?

- Sending SQL statement with comments or injecting comments in SQL statement and sending to database change the behaviour of SQL statement. These leads to logical errors.
- SQL comments are defined using "--"

II Program with SQL Injection.

```
import java.sql.*;  
import java.util.*;  
class SQLInjectionTest1  
{  
    public static void main (String args[]) throws  
        Exception  
{  
        Class.forName("oracle.jdbc.driver.OracleDriver");  
        Connection cn = DriverManager.getConnection  
            ("jdbc:oracle:thin:@localhost:1521:XE", "System",  
             "Aman");  
        Statement st = cn.createStatement();  
        Scanner scan = new Scanner(System.in);  
        System.out.println("Input username");  
        String u = scan.next();  
        System.out.println("Input password");  
        String p = scan.next();  
        String sql = "Select count(*) from user where  
            uname = " + " " + " " + " " + "  
            -- and pwd = " + " " + " " + " " + " " + "  
Test. // SQL = server(SQL); } ①
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

C ResultSet rs = st.executeQuery(sql);
C rs.next();
C int c = rs.getInt(1);
C if(c > 0)
C sopln("valid uname and pwd");
C else
C sopln("Invalid username username and pwd");
C place the box here

```

12.10.15

SQL Statement with parameters :-

→ PreparedStatement allows to represent an SQL statement with parameters.

→ Each parameter is rep. using a special symbol ?.

→ Each ? is called place holder, which receive

value during runtime.

→ Each Parameter is identified with unique number is called index number.

```

C → /* static String Server( String sql )
C {
C     {
C         sql = sql.replaceAll("-","");
C         return sql;
C     }
C }
C */

```

Add ① + ② After in the program SQL Injection
then check.

11 Write a program to register user using PreparedStatement.

Working with Eclipse

Install Eclipse software

Eclipse Luna (JEE)

Eclipse Kepler (JEE)

Eclipse Mars (JEE)

Open Eclipse

* workspace → it is a folder where all project get saved.

We can change the workspace.

d:\jdbc11am

* creating project

→ Select → File → New → select others → Java project
for creating Standalone application.

projectName: JDBeProject.

→ click on Next button.

→ click on Library.

click on External jar

→ Add jdbc4.jar.

→ finish.

(*) After creating object need to add external jar

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

package jdbcproject;
import java.sql.*;
import java.util.*;
public class UserReg
    public static void main (String args[])
    {
        Connection cn = null;
        PreparedStatement ps = null;
        try {
            Class.forName ("oracle.jdbc.driver.OracleDriver");
            cn = DriverManager.getConnection ("jdbc:oracle:thin:@localhost:1521:XE", "system", "Anjan");
            ps = cn.prepareStatement ("insert into user_reg values (?, ?, ?)");
            Scanner scan = new Scanner (System.in);
            System.out.println ("Input name");
            String u = scan.nextLine();
            System.out.println ("Input username");
            String n = scan.nextLine();
            System.out.println ("Input pwd");
            String p = scan.nextLine();
            ps.setString (1, n);
            ps.setString (2, u);
            ps.setString (3, p);
            int count = ps.executeUpdate();
            if (count > 0)
                System.out.println ("User register successfully");
            else
                System.out.println ("Error in register");
        }
    }
}

```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
        catch (ClassNotFoundException e1)
        {
            System.out.println("Invalid Driver class");
        }
        catch (SQLException e2)
        {
            System.out.println(e2);
        }
    Finally
    {
        try
        {
            if (ps != null)
                ps.close();
            if (cn != null)
                cn.close();
        }
        catch (SQLException e3)
        {
            System.out.println(e3);
        }
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

- Prepared statement provides ~~set~~ setter methods to define the value of each parameter.
- For each datatype it provides one setter method.

Set int (int parameterIndex, int x)

sets the designated parameter to the given Java int value. The driver converts this to an SQL INTEGER value when it sends it to the database.

public void setString (int parameterIndex, String x)

Sets the designated parameter to the given Java string value. The driver converts this to an SQL VARCHAR or LONGVARCHAR value when it sends it to the database.

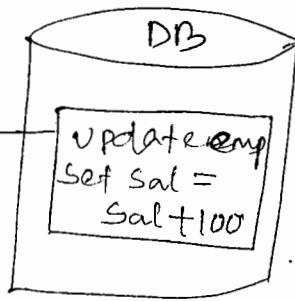
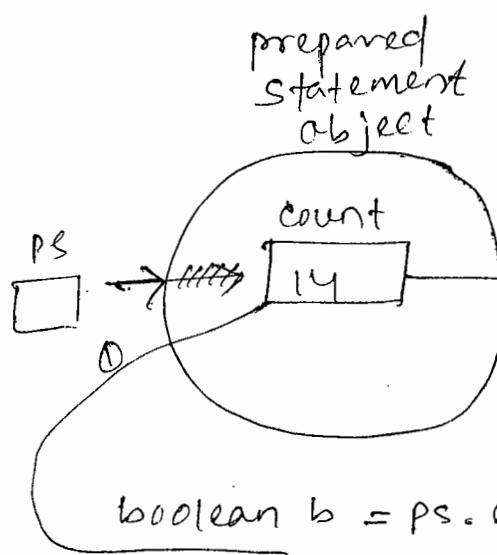
execute() Method :-

boolean execute()
Execute the SQL Statement in this PreparedStatement object. which may be any kind of SQL Statement.

- The execute method returns a boolean to indicate the form of the first result.
- true if first result is a Resultset object ;
false if the result first result is an update count or there is no result.

→ Prepared Statement `ps = cn.prepareStatement("Update emp set sal = sal + 100");`

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.



(return - False)
For DML execute

`boolean b = ps.execute();`

`int count = ps.getUpdateCount();`

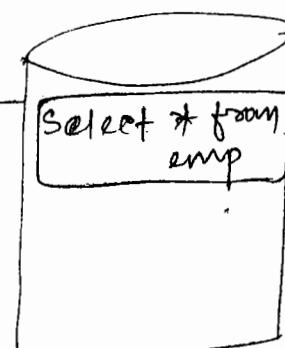
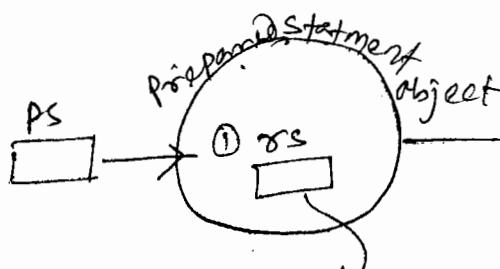
`sopln(b); → false`

`sopln(count); → 14`

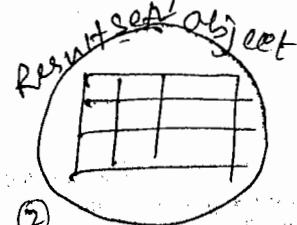
↳ To get return from
DML command for
execute() method.

Execute Method return type and count.

→ Prepared Statement `ps = cn.prepareStatement("select * from emp");`



(return - true)
For DQL execute



`boolean b = ps.executeQuery();
true`

`ResultSet rs = ps.getResultSet();`

`sopln(b); → true`

13.10.15

Write a program to insert and read values from department table ?

```
import java.sql.*;
import java.util.*;
class ExecuteTest
{
    public static void main (String args[])
    {
        Connection cn = null;
        PreparedStatement ps = null;
        ResultSet rs = null;
        try
        {
            Class.forName ("oracle.jdbc.driver.OracleDriver");
            cn = DriverManager.getConnection ("jdbc:oracle:thin:@localhost:1521:XE", "system", "Arijan");
            Scanner scan = new Scanner (System.in);
            System.out.println ("Input SQL Statement");
            String SQL = scan.nextLine();
            ps = cn.prepareStatement (SQL);
            boolean b = ps.executeUpdate();
            if (b == false)
            {
                int count = ps.getUpdateCount();
                System.out.println (count);
            }
            else
            {
                if (b == true)
                {
                    ResultSet rs = ps.getResultSet();
                    while (rs.next())
                    {
                        System.out.println (rs.getInt(1) + ":" + rs.getString(2) +
                                           ":" + rs.getString(3));
                    }
                }
            }
        }
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

        catch (ClassNotFoundException e)
    {
        sopln ("Invalid driver class name");
    }

    catch (SQLException e)
    {
        sopln (e);
    }

    finally
    {
        try
        {
            if (rs != null)
                rs.close();
            if (ps != null)
                ps.close();
            if (cn != null)
                cn.close();
        }
        catch (SQLException e)
        {
            sopln (e);
        }
    }
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Q: program to update salary of given employee.

Package Jdbcproject;

import java.sql.*;

import java.util.*;

public class UpdateSal

{ public static void main (String args[])

{

Connection cn = null;

PreparedStatement ps = null;

try

{

Class.forName ("oracle.jdbc.driver.OracleDriver");

cn = DriverManager.getConnection ("jdbc:oracle:

thin:@localhost:1521:XE", "system", "Aryan");

ps = cn.prepareStatement ("Update emp set sal =

("update emp set sal =
sal + ? where
empno = ? ") ;

Scanner scan = new Scanner (System.in);

System.out.println ("Input empno?");

int eno = scan.nextInt();

System.out.println ("Input Sal");

float s = scan.nextFloat();

PreparedStatement

ps.setFloat (1, s);

ps.setInt (2, eno);

int count = ps.executeUpdate();

System.out.println (count);

}

SRI RAGHAVENDRA XEROX

Software Languages Material Available

Beside Bangalore Ayyagar Bakery,

Opp. CDAC, Balkampet Road,

Ameerpet, Hyderabad.

```

        catch (ClassNotFoundException e)
        {
            System.out.println("Invalid driver class name");
        }
    catch (SQLException e)
    {
        System.out.println(e);
    }
    finally
    {
        try{
            if (ps!=null)
                ps.close();
            if (cn!=null)
                cn.close();
        }
        catch (SQLException e)
        {
            System.out.println(e);
        }
    } //Finally
} //main Method
} //class

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

CallableStatement :-

- callablestatement is an Interface.
- This interface is implemented by jdbc driver developer.
- It is a subtype of PreparedStatement.
- callablestatement is used for calling stored procedure and functions.

Q. what is difference b/w Statement and PreparedStatement and callablestatement.

Statement

- (i) Sending static SQL statements.
- (ii) SQL statement is compiled every time before execution.
- (iii) It is SQL statement without parameters.

PreparedStatement

- (i) Sending dynamic SQL Statement.
- (ii) SQL Statement is compiled only once and stored in prepared statement object.
- (iii) SQL statements with parameters.

callablestatement

- (i) For calling stored procedure.
- (ii) It also hold precompiled SQL Statement and call to procedure.
- (iii) SQL statements with parameters.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Q. How to get callablestatement object?

- connection provides the following method which returns callablestatement object.

CallableStatement prepareCall (String sql)

→ creates a callablestatement object for calling database stored procedures.

Syntax:-

{ call procedure-name ([parameters]) } → calling the procedure
 { ? = call function-name ([parameters]) } → calling the function

CallableStatement cs = cn. preparecall ("{ call updateSal }");

How to write procedure in Oracle database :-

create or replace procedure <procedure-name> ([parameters])
 is declare variables;

begin
 statements;

14-10-15

Procedure to update salary of Employee.

Procedure to update procedure -

1. create or replace procedure

update_emp is

2. begin

3. update emp set sal = sal + 100;

4. end;

5. /

// programme to call procedure by using callableStatement.

import java.sql.*;

public class CallablestmtTest1 {
 public static void main (String args[]) {
 connection cn = null;
 callablestatement cs = null;

try {
 Class.forName ("---");
 cn = DriverManager.getConnection ("", "", "");
 cs = cn. preparecall ("{ call update_emp }");
 cs. execute();

}

catch (SQLException e)

{ System.out.println (e); }

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

```

    catch(SQLE e1)
    {
        sopln(e1);
    }
    Finally{try
        if(@es!=null)
            es.close();
        if(en!=null)
            en.close();
    }
    catch(SQLE e2)
    {
        sopln(e2);
    }
}
}
}

```

// Procedure with parameter

⇒ procedure is define with two types of parameters

(1) IN parameter.

(2) OUT parameter.

(1) IN parameter receive value from caller (Java program)

(2) OUT parameter return value to caller (Java program)

// procedure to update salary of given employee.

create or replace procedure update_sal (teno number,

~~tsal~~ tsal number) is

begin

update emp set sal = sal + tsal where empno = teno;

end;

↳ paste this code into SQL*plus and enter.

SRI RAGHAVENDRA XEROX

Software Languages Material Available

Beside Bangalore Ayyagar Bakery,

Opp. CDAC, Balkampet Road,

Ameerpet, Hyderabad.

11 program to give runtime value to previous procedure by using CallableStatement.

```
class Callablestmt2 {
    public static void main ( String args[] ) throws Exception {
        Class.forName ("oracle.jdbc.driver.OracleDriver");
        Connection cn = DriverManager.getConnection
            ("---", "---", "---");
        CallableStatement cs = cn.prepareCall ("{call
            update_sal (?,?)}");
        Scanner scan = new Scanner (System.in);
        System.out.println ("Enter Employee number:");
        int n = scan.nextInt();
        System.out.println ("Enter Salary:");
        float s = scan.nextFloat();
        cs.setInt (1, n);
        cs.setInt (2, s);
        cs.execute();
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

// procedure with OUT parameter.
→ callableStatement provide the following method to define OUT parameter.
void registerOutParameter (int parameterIndex,
 int sqlType)

→ Register the OUT parameter in ordinal position parameterIndex to the JDBC type SqlType.

TYPES :-

→ Types is a class exist in java.sql package.
→ This class defines the constants that are used to identify generic SQL types, called JDBC types.

// create table in Database

create table account

(accno number(3),
balance number(10,2));

insert data into table

accno	balance
101	5000
102	6000

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

// procedure for Deposite.

→ write a procedure for Withdraw.

create or replace procedure withdraw (tacno number,
amt number, ~~bal~~ out number) is
begin
 b number(10,2);

 Select balance into b from account where ~~accno~~

 accno = tacno;

 if amt < b then

 update account set balance = balance - amt

 where accno = tacno;

 end if;

 Select balance into bal from account where

 accno = tacno;

 end;

SRI RAGHAVENDRA XEROX

Software Languages Material Available

Beside Bangalore Ayyagar Bakery,

Opp. CDAC, Balkampet Road,

Ameerpet, Hyderabad.

// program for previous procedure.

```
class Withdraw {
    public static void main (String args []) throws Exception {
        Class.forName ("com.mysql.jdbc.Driver");
        Connection cn = DriverManager.getConnection ("jdbc:mysql://localhost:3306/bank", "root", "root");
        CallableStatement cs = cn.prepareCall ("{call withdraw (?, ?, ?)}");
        cs.registerOutParameter (3, Types.NUMERIC);
        Scanner scan = new Scanner (System.in);
        System.out.print ("Enter Account number : ");
        int a = scan.nextInt ();
        System.out.print ("Enter withdraw amount : ");
        float t = scan.nextFloat ();
        cs.setInt (1, a);
        cs.setFloat (2, t);
        cs.execute ();
        float bal = cs.getFloat (3);
        System.out.println ("Balance available is : " + bal);
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

15.10.15

#. How to create Function in Oracle database

create or replace function <function-name>(<parameters>)

return <returntype> is
declaring local variables;

begin

statements;

return <value>;

end;

callableStatement cs = cn.prepareCall("{" + "call
function-name(<parameters>)" + "}")

// write a function which returns total salary
of employee.

SQL> create or replace function get_total(~~eno~~
(eno number) return number is

total number(10,2);

begin

select sal+nvl(comm,0) into total
from emp where empno = eno;

return total;

end;

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

11 Program to get total Salary of employee ?

```
import java.sql.*;  
import java.util.*;  
public class callableStatementTest4 {  
    public static void main ( String args [] ) {  
        connection cn = null ;  
        callablestatement cs = null ;  
        try {  
            Class.forName (" _____ ");  
            cn . DriverManager . ( _____ );  
            cs = cn . prepareCall (" {? = call get - total (?) } ");  
            cs . registerOutParameter (1 , Types . NUMERIC );  
            Scanner scan = new Scanner ( System . in );  
            System . out . print (" Input empno : ");  
            int e = scan . nextInt ();  
            cs . setInt (2 , e );  
            cs . execute ();  
            float t = cs . getFloat (1 );  
            System . out . print (" Total Salary " + t );  
        }  
        catch ( CNFE e )  
        {  
            System . out . print (" Invalid driver class " );  
        }  
        catch ( SLE e1 )  
        {  
            System . out . print ( e1 . getMessage () );  
        }  
        finally {  
            if ( cs != null )  
                cs . close ();  
            if ( cn != null )  
                cn . close ();  
        }  
    }  
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Material Available
Opp. CDAC, Balkampet Bakery,
Ameerpet, Hyderabad.

```
catch (SQLException k)
```

```
{  
    sopln(k);  
}
```

```
} } }
```

Working with SQL 99 Datatypes

1. BLOB
2. CLOB
3. Array
4. object.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

BLOB

- BLOB stands for Binary Large Object.
- If database table is created with column type as BLOB it hold binary objects like images and audio, video... files.
- SQL99 Datatypes supported by prepared statement.
- Prepared Statement provide the following method to work with BLOB.

```
void setBinaryStream(int parameterIndex,  
                     InputStream x, int Length)
```

- Sets the designated parameter to the given input stream, which will have the specified number of bytes.

→ The data will be read from the stream as needed → until end-of-file is reached.

Create one table

SQL> ~~create table member-reg (name varchar2(20),~~
SQL> create table member-reg (name varchar2(20),
 2 uname varchar2(20),
 3 pwd varchar2(20),
 4 image blob);

Table created

Program

```
import java.sql.*;  
import java.io.*;  
public class InsertImage {  
    public static void main (String [] args) throws Exception  
    {  
        Connection cn = DriverManager.getConnection  
            ("-----", "-----");  
        PreparedStatement ps = cn.prepareStatement  
            ("insert into member-reg values (?,?,?,?,?)");  
        File f = new File ("welcomescan.jpg");  
        FileInputStream fis = new FileInputStream (f);  
        ps.setString (1, "narush");  
        ps.setString (2, "nit");  
        ps.setString (3, "nit123");  
        ps.setBinaryStream (4, fis, (int)(f.length()));  
    }  
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

        int count = ps.executeUpdate();
        System.out.println(count);
    }
}

```

How to read image from database table?

→ ResultSet provides the following method to manipulate BLOB type.

⇒ `InputStream getBinaryStream (int columnIndex)`

→ Retrieves the value of the designated column in the current row of this ResultSet object as a binary stream of uninterpreted bytes.

Ques 10.15

// Program to read image from database table.

```

package jdbeproject;
import java.sql.*;
import java.io.*;
public class ReadImage{
    public static void main(String args[]) throws Exception
    {
        Class.forName("com.mysql.jdbc.Driver");
        Connection cn = DriverManager.getConnection(
            "jdbc:mysql://localhost:3306/test");
        PreparedStatement ps = cn.prepareStatement(
            "Select * from Member_Reg");
    }
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

```

C ResultSet rs = ps.executeQuery();
C   rs.next();
C   String str =
C     InputStream is = rs.getBinaryStream(4);
C     FileOutputStream fos = new FileOutputStream Column Index
C       ("d:\\x.jpg");
C     int b;
C     while ((b = is.read()) != -1)
C       fos.write(b);
C   }
C }

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

CLOB

- CLOB stands for character Large Object.
- If database table created with column type as CLOB it hold text files.
- PreparedStatement provide the following method to Manipulate CLOB type

<u>Method</u>	void setCharacterStream(int parameterIndex, Reader reader, int length)
---------------	--

- Sets the designated parameter to the given Reader object, which is the given number of characters long.

Step

- (i) create a text file and save it in D:drive.
- (ii) create a table in which ~~BLOB~~ CLOB column is there. (table name member-profile)

program

// program to insert text file in DB.

```

package Jdbcproject;
import java.sql.*;
import java.io.*;
public class storeblob {
    public static void main (String args[]) throws Exception {
        Class.forName ("-----");
        Connection cn = DriverManager.getConnection
            ("-----");
        PreparedStatement ps = cn.prepareStatement
            ("insert into Member-profile
             values (?,?)");
        ps.setString (1, "naresh");
        File f = new File ("d:\\p.txt");
        FileReader fr = new FileReader (f);
        ps.setCharacterStream (2, fr, (int) (f.length()));
        int count = ps.executeUpdate ();
        System.out.println (count);
    }
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

○ C # How to read CLOB type?

○ C ➔ ResultSet provide the following Method, for
C read CLOB type in database.

C Reader getCharacterStream(int columnIndex)

C ➔ Returns the value of the designated
C column in the current row of this
C ResultSet object as a java.io.Reader object.

SRI RAGHAVENDRA XEROX

Software Languages Material Available

Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

www.sriraghavendra.com
+91 98410 22222
+91 98410 22222
+91 98410 22222

Array :-

Q. How to create Array type in oracle database?

In oracle array is a user defined type.

Syntax :-

```
create type <type-name> is/AS VARRAY([size])  
    OF <type>
```

```
SQL> create type e-type As Varray(3) of varchar2(20);
```

```
SQL> create table Student (name, varchar2(20),  
                           e-type);
```

```
SQL> insert into Student values ('naresh',  
                                 e-type('Java',  
                                       'oracle'));
```

→ PreparedStatement provide the following method
for manipulating array type.

```
void setArray(int i, Array a)
```

→ sets the designated parameter
to the given Array object. The driver converts
this to an SQL ARRAY value when it
sends it to the database.

17.10.15

SQL> create type course-type as varray(3) of varchar2(20);
Type created.

SQL> create table student_info(rno number(3),
2. name varchar2(20),
3 course course-type);

Table created.

Array Interface :-

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

→ JDBC provide
→ The mapping in the Java programming Language
for the SQL type ARRAY.

program: program for Array.(Inserting array data).

```
package jdbcproject;
import java.sql.*;
import java.util.*;
public class ArrayTest {
    public static void main(String args[]) throws Exception {
        Class.forName("oracle.jdbc.driver.OracleDriver");
        Connection cn = DriverManager.getConnection
            ("-----");
        PreparedStatement ps = cn.prepareStatement
            ("insert into student_info values(?, ?, ?)");
        Scanner scan = new Scanner(System.in);
        System.out.print("Input rno:");
        int rno = scan.nextInt();
        ps.setInt(1, rno);
        ps.setString(2, "Ameerpet");
        ps.setString(3, "Hyderabad");
        ps.executeUpdate();
    }
}
```

```

        System.out.println("Input name");
        String name = scan.next();
        System.out.println("Input how many courses");
        int n = scan.nextInt();
        String s[] = new String[n];
        System.out.println("Input course name");
        for (int i=0; i<s.length; i++)
            s[i] = scan.next();
    }
}

```

ArrayDescriptor ad = ArrayDescriptor.createDescriptors -
 descriptors(cn, "COURSE_TYPE")

ARRAY a = new ARRAY(ad, cn, s);

```

ps.setInt(1, 80);
ps.setString(2, name);
ps.setArray(3, a);

```

int count = ps.executeUpdate();

System.out.println(count);

}

}

How to read array type column from database table :-

→ Resultset provide the following method which read array type.

Array getArray (int i)

→ Retrives the value of the designated column in the current row of this ResultSet object as an Array object in the Java programming language.

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

- an Array object representing the SQL ARRAY value ~~in~~ in the specified column.
- an Array object contains a logical pointer to the data in the SQL ARRAY value rather than containing the Array values data.
Array a = rs.getArray(3); // written in the programme.

Program

program to read Array list from database.

```
package JdbcProject;
```

```
package JdbcProject;
```

```
import java.sql.*;
```

```
import java.*;
```

```
class ArrayTest2 {
```

```
    public static void main(String args[]) throws Exception {
```

```
    {
```

```
        Class.forName("-----");
```

```
        Connection cn = DriverManager.getConnection("-----");
```

```
        PreparedStatement ps = cn.prepareStatement("select * from student_info");
```

```
        ResultSet rs = ps.executeQuery();
```

```
        rs.next();
```

```
        int sno = rs.getInt(1);
```

```
        String name = rs.getString(2);
```

```
        Array a = rs.getArray(3);
```

```
        SopIn(sno);
```

```
        SopIn(name);
```

```
        ResultSet rs = a.getResultSet();
```

SRI RAGHAVENDRA XEROX

Software Languages Material Available

Beside Bangalore Ayyagar Bakery,

Opp. CDAC, Balkampet Road,

Ameerpet, Hyderabad.

```

while ( r.next() )
{
    System.out.println (r.getString(2));
}
}
}

```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SQL Data :-

- The interface is used for the custom mapping of an SQL user-defined type (UDT) to a class in the Java programming language.
- The class object for a class implementing the SQLData interface will be entered in the appropriate connection objects type map along with the SQL name of the UDT for which it's a custom mapping.

Methods of SQL data Interface

① → String getSQLTypeName()

- Returns the fully-qualified name of the SQL user-defined type that this object represent.

② → void readSQL (SQLInput stream, String typename)

- populates this object with data read from the database.

③ → void writeSQL (SQLOutput stream)

- writes this object to the given SQL data stream, converting it back to its SQL value in the data store.

SQLOutput:-

- SQLOutput is an Interface, implemented by JDBC driver developer.
- The Output stream for writing the attributes of a user-defined type back to the database. This interface, used only for custom Mapping, is used by the drivers, and its methods are never directly invoked by a programmer.
- When an object of a class implementing the interface SQLData is passed as an argument to an SQL statement, the JDBC driver calls the method SQLData.getSQLType to determine the kind of SQL datum being passed to the database.
- The driver then creates an instance of SQLOutput and passes it to the method SQLData.writeSQL.
- The Method writeSQL in turn calls the appropriate SQLOutput writer methods writeBoolean, writeCharacterStream, and so on to write data from the SQL Data object to the SQL output SQLOutput output stream as the representation of an SQL user defined type.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

first we devlope a Address class (user defined datatype).

```
package Jdbcproject;
import java.sql.*;
public class Address implements SQLData
{
    private int hno;
    private String street;
    private String city;
    private String SQLType;
    public Address (int hno, String street,
                    String city, String SQLType)
    {
        this.hno = hno;
        this.street = street;
        this.city = city;
        this.SQLType = SQLType;
    }
    public void writeSQL (SQLOutput stream)
        throws SQLException
    {
        stream.writeInt(hno);
        stream.writeString(street);
        stream.writeString(city);
    }
    public *read
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
O
O     public void readSQL(CSQLInputStream stream,
C             String SQLType)
C
C             throws SQLException.
```

```
C     hno = stream.readInt();
C     street = stream.readString();
C     city = stream.readString();
C     this.SQLType = SQLType;
C }
```

```
C     public String getSQLTypeName() throws
C             SQLException.
```

```
C     {
C         return SQLType;
C     }
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
C }
```

```
C     Add this 3 method in Address class.
```

```
C     (1) int getHno() { return
C             return hno;
C         }
C     (2) String getStreet()
C         {
C             return street;
C         }
C     (3) String getCity()
C         {
C             return city;
C         }
```

// program for insert object into Database table.

```
package jdbcproject;
import java.sql.*;
public class InsertObject{
    public static void main(String args[]) throws Exception
    {
        Class.forName("-----");
        Connection cn = DriverManager.getConnection
        ("-----", "-----");
        PreparedStatement ps = cn.prepareStatement
        ("insert into person values (?,?)");
        ps.setString(1, "Naresh");
        Address a = new Address(101, "Ameerpet",
                               "Hyd", "ADD-TYPE");
        ps.setObject(2, a);
        int count = ps.executeUpdate();
        System.out.println(count);
    }
}
```

SRI RAHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

20.10.15

SQLInput:-

- SQLInput is an Interface. This interface is implemented by JDBC driver developer.
- An input stream that contains a stream of values representing an instance of an SQL structured type or an SQL distinct type.
- When the method getObject is called with an object of a class implementing the interface SQLData, the JDBC driver calls the method SQLData.getSQLType to determine the SQL type to determine the SQL type of the user defined type (UDT) being custom mapped.

// program to read object.

```
import java.sql.*;
class ReadObjectTest
{
    public static void main(String args[]) throws Exception
    {
        Class.forName("Oracle.jdbc.driver.OracleDriver");
        Connection cn = DriverManager.getConnection(
            "_____");
        PreparedStatement ps = cn.prepareStatement(
            "Select * from person");
        ResultSet rs = ps.executeQuery();
        rs.next();
        String name = rs.getString(1);
```

SRI RAGHAVENDRA XEROX

Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

struct s = (Struct) rs.getObject(2);
Object o[] = s.getAttributes();
System.out.println(o[0]);
    System.out.println(o[1]);
    System.out.println(o[2]);
}
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Structure

- Structure is an interface and implemented by JDBC driver developer
- Struct object contains a value for each attribute of the SQL structured type that it represents

→ Object[] getAttributes()

- ~~Result~~ produces the ordered values of the attributes of the SQL structure the type that this struct object represent.
- Resultset provides the following method which return related type of object.

→ Object getObject (int i, Map map)

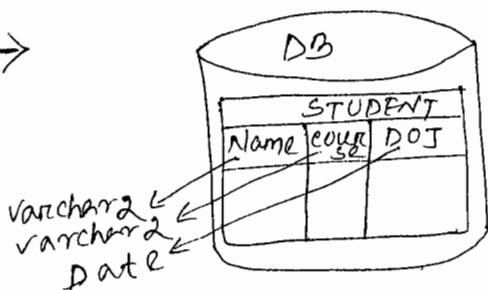
- Retrieves the value of the designated column in the current row of this ResultSet object as an object in the given programming language

// Read object from database by Mapping:

```
import java.sql.*;
import java.util.*;
class ReadObjectTest2
{
    public static void main(String args[]) throws Exception
    {
        Class.forName("oracle.jdbc.driver.OracleDriver");
        Connection cn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe", "scott", "tiger");
        PreparedStatement ps = cn.prepareStatement("select * from person");
        ResultSet rs = ps.executeQuery();
        rs.next();
        String name = rs.getString(1);
        Address a = (Address)rs.getObject(2, hm);
        System.out.println(name);
        System.out.println(a.getHno());
        System.out.println(a.getStreet());
        System.out.println(a.getCity());
```

Working with Date Datatype:-

DOJ = Date of Joining.



for Statement :-

```
statement st = cn.createStatement();
int count = st.executeUpdate("Insert into student
values ('Ram', 'Java', '20-oct-15')");
System.out.println(count);
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

7.11.15

Metadata:-

Q. what is Metadata?

→ Data about data is called Metadata.

→ In order to find information about data at runtime we use metadata.

→ JDBC provide the following interfaces in order to work with Metadata.

1. ResultSetMetaData

2. DatabaseMetaData

ResultSetMetaData :-

→ ResultSetMetaData is an Interface.

→ This interface is implemented by JDBC driver developer.

→ It is an object that can be used to get information about the types and properties of the columns in a ResultSet object.

Q. How to create ResultSetMetaData object?

→ ResultSet provide the following method which returns ResultSetMetaData object.

ResultSetMetaData getMetaData()

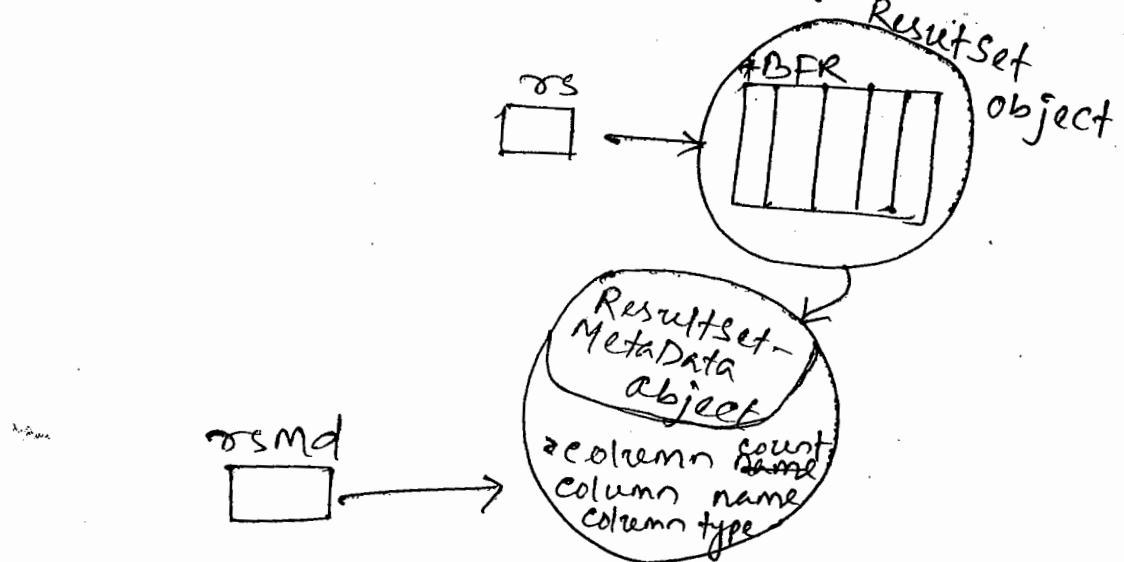
Method

→ Retrieves the number, types and properties of this ResultSet Object's columns.

```

Statement st = cn.createStatement();
ResultSet rs = executeQuery("Select * from
                           emp");
ResultSetMetaData rsmd = rs.getMetaData();

```



Methods of ResultSetMetaData

- `int getColumnCount()`
- Returns the number of columns ~~of~~ in this `ResultSet` object.
- `String getColumnName (int column)`
 - Get the designated column's name.
- `int getColumnType (int column)`
 - Returns the designated column's SQL type.
- `String getColumnTypeName (int column)`
 - Returns the designated column's database-specific type name.

// Program to display column count, column names
and column types exist in Resultset.

```
import java.util.*;
import java.sql.*;
class JdbcTest30
{
    public static void main (String args[])
    {
        Connection cn = null;
        Statement st = null;
        try
        {
            Class.forName ("_____");
            cn = DriverManager.getConnection ("_____");
            Scanner scan = new Scanner (System.in);
            System.out ("Input Query");
            String query = scan.nextLine();
            st = cn.createStatement();
            ResultSet rs = st.executeQuery (query);
            ResultSetMetaData rsm = rs.getMetaData();
            int count = rsm.getColumnCount();
            for (int i=1; i<=count; i++)
            {
                String cname = rsm.getColumnName(i);
                String ctype = rsm.getColumnTypeName(i);
                System.out.printf ("\n %s %s", cname, ctype);
            }
        }
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

catch ( CNFE e)
{
    sopln (*e);
}

catch (SQLE e1)
{
    sopln (e1);
}

finally
{
    try
    {
        if (st!=null)
            st.close();
        if (cn!=null)
            cn.close();
    }
    catch ( SQLE e3)
    {
        sopln (e3);
    }
}
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

// Write a program to display contents of any table?

```

import java.sql.*;
import java.util.*;
class JDBCtest31
{
    public static void main (String args[])
    {
        Connection cn = null;
        Statement st = null;
        ResultSet rs = null;
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            cn = DriverManager.getConnection("-----");
            st = cn.createStatement();
            Scanner scan = new Scanner(System.in);
            System.out.println("Input Query");
            String query = scan.nextLine();
            rs = st.executeQuery(query);
            ResultSetMetaData rsmd = rs.getMetaData();
            int count = rsmd.getColumnCount();
            while(rs.next()) {
                for (int i = 1; i <= count; i++) {
                    System.out.print(rs.getInt(i));
                    if (rs.getString(i).length() > 10) {
                        System.out.print("... ");
                    }
                }
                System.out.println();
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}

```

```

catch (CNFE e)
{
    sopin ("Invaled Driver class Name");
}

catch (SQLE e1)
{
    sopin (e1);
}

finally
{
    try
    {
        if (st != null)
            st.close();
        if (cn != null)
            cn.close();
    }
    catch (SQLE e3)
    {
        sopin (e3);
    }
}
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

DatabaseMetaData :-

- comprehensive information about the database as a whole.
- DatabaseMetaData is an interface this interface is implemented by driver vendors to let users know the capabilities of a Database Management System (DBMS) in combination with the driver based on JDBC technology ("JDBC driver") that is used with it.

How to get DatabaseMetaData objects?

- Connection provides the following method which returns DatabaseMetaData object.

`DatabaseMetaData getMetaData()`

- Retrieves a DatabaseMetaData object that contains metadata about the database to which the connection object represents a connection.

Methods of DatabaseMetaData

→ `int getJDBCMajorVersion()`

- Retrieves the major JDBC version number for this driver.

→ `int getJDBCMinorVersion()`

- Retrieves the ~~major~~ minor JDBC version number for this driver.

→ `String getDatabaseProductName()`

- Retrieves the name of this database product.

- String getDatabaseProductVersion()
→ Retrieves the version number of this database product.
- int getMaxConnections()
→ Retrieves the maximum number of concurrent connection to this database that are possible.

// Finding information about database.

```

import java.sql.*;
class JdbcTest32 {
    public static void main(String args[]) {
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            Connection cn = DriverManager.getConnection(
                "_____, _____, _____, _____");
            DatabaseMetaData dmd = cn.getMetaData();
            DatabaseMetaData dmd = cn.getMetaData();
            System.out.println(dmd.getDatabaseProductName());
            System.out.println(dmd.getDatabaseProductVersion());
            System.out.println(dmd.getMaxConnections());
        } catch (Exception k) {
            System.out.println(k);
        }
    }
}

```

→ `ResultSet getTables(String catalog, String schemaPattern, String tableNamePattern, String[] types)`

→ Retrieves a description of the tables available in the given catalog.

// Program to display tables and views exist in database.

```
import java.sql.*;
```

```
class JdbcTest33
```

```
{
```

```
    public void main( String args[] ) throws Exception
```

```
{
```

```
    Class.forName("oracle.jdbc.driver.OracleDriver");
```

```
    Connection cn = DriverManager.getConnection  
        ("_____, _____, _____");
```

```
    DatabaseMetaData dbmd = cn.getMetaData();
```

```
    String types[] = {"TABLE", "VIEW"};
```

```
    ResultSet rs = dbmd.getTables( null, null, "%", types);
```

test 1) SCOTT
test 2) ANJAN

```
    while( rs.next() )
```

```
    {  
        System.out.println( rs.getString(2) );
```

```
    }
```

```
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Batch updates :-

- Batch processing/updates allows you to group related SQL statements into a batch and submit them with one call to the database.
- JDBC drivers are not required to support this feature, you should use the DatabaseMetaData. supportsBatchUpdates() method to determine if the target database supports batch update processing. The method returns true if your JDBC driver supports this feature.

How to work with batch updates using statement :-

- Statement provide the following method for adding SQL statement to batch.
- **void addBatch(String sql)**
 - Adds the given SQL command to the current list of commands for this Statement object.
- **int[] executeBatch()**
 - Submits batch of commands to the database for execution and if all commands execute successfully returns an array of update counts.
- **void clearBatch()**
 - Empties this Statement object's current list of SQL commands.

```

C import java.sql.*;
C class JDBCtest34
C {
C     public static void main(String args[])
C     {
C         System.out.println("main() starts");
C         try
C         {
C             Class.forName("oracle.jdbc.driver.OracleDriver");
C             Connection cn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:XE", "system", "Anjan");
C             Statement st = cn.createStatement();
C             st.addBatch("update emp set sal = sal + 100 where deptno = 10");
C             st.addBatch("update emp set sal = sal + 200 where deptno = 20");
C             st.addBatch("update emp set sal = sal + 300 where deptno = 30");
C             int count[] = st.executeBatch();
C             for(int i=0; i<count.length; i++)
C                 System.out.println(count[i]);
C             st.close();
C             cn.close();
C         }
C     }
C }

```

How to work with batch updates using prepared statement:-

Void addBatch()

- Adds a set of parameters to this preparedStatement object's batch of commands.
- executeBatch method of preparedStatement does not return update count. It returns values indicates command executed or not.

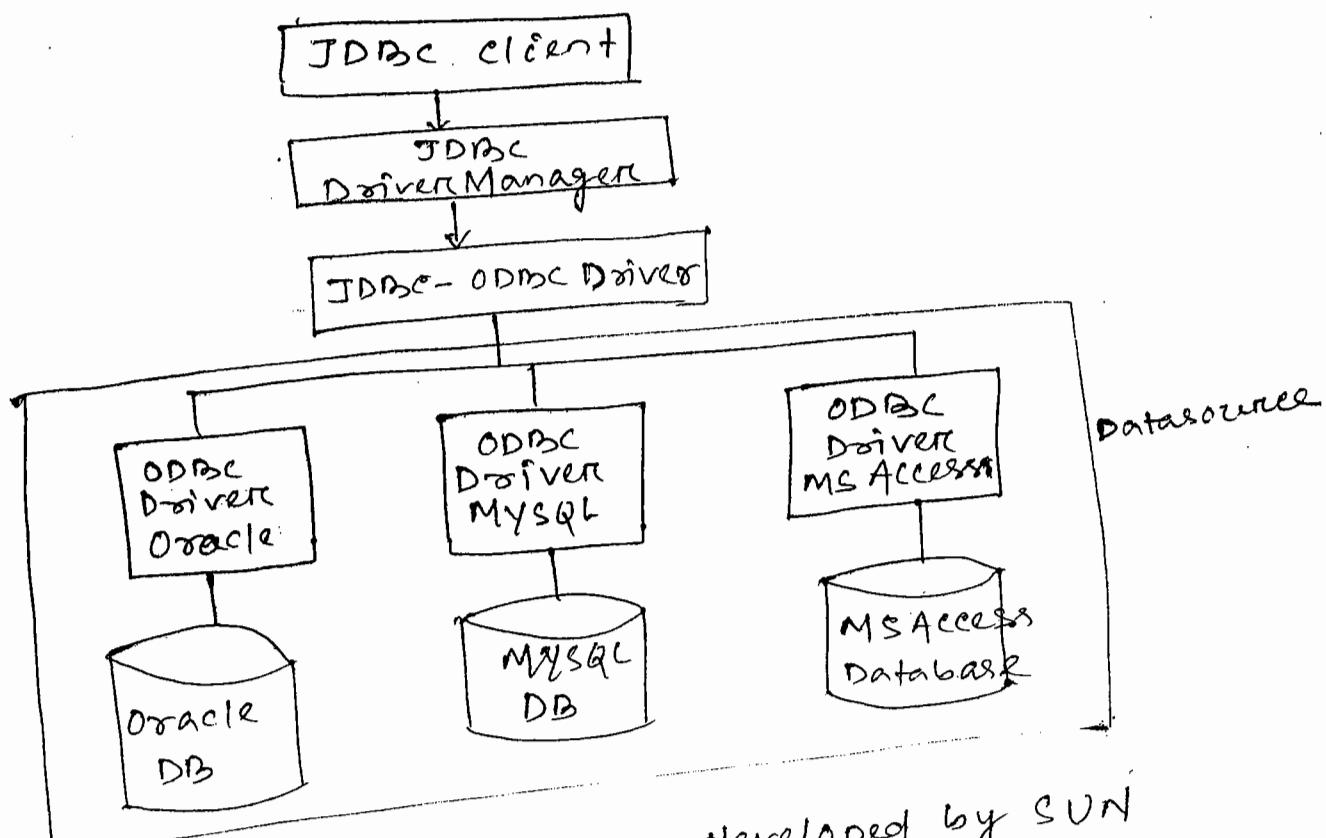
Example for prepared statement

```
import java.sql.*;
class JdbcTest35
{
    p.s. v. Main(String args[]) throws Exception
    {
        class.forName("oracle.jdbc.driver.OracleDriver");
        connection cn = DriverManager.getConnection
        ("jdbc:oracle:thin:@localhost:1521:XE", "system",
         "Anjan");
        preparedstatement ps = cn.prepareStatement
        ("update emp set sal = ? where deptno = ?");
        ps.setFloat(1, 100f);
        ps.setInt(2, 10);
        ps.addBatch();
        ps.setFloat(1, 200f);
        ps.setInt(2, 20);
        ps.addBatch();
        ps.executeBatch();
        int count[] = ps.getUpdateCount();
        for(int i=0; i<count.length; i++)
            System.out.println(count[i]);
        ps.close();
        cn.close();
    }
}
```

3 (continues)
O/P:- -2 } : A value of ~~success~~ SUCCESS_NO_INFO --
-2 } : indicates that the command was processed
successfully but that the no. of rows
affected is unknown.

Working with type-1 drivers:-

JDBC - ODBC bridge drivers



- JDBC-ODBC bridge driver developed by SUN MICRO SYSTEM. It is a part of JDK.
- JDBC-ODBC bridge driver required ODBC driver to communicate with database.

How to create Datasource?

Step 1: Open control panel.

Step 2: Open administrative tools.

Step 3: Open odbc datasource.

There are 3 types of DSN (Data Source Name)

(i) User DSN:- An user datasource is only visible to you and can be used only on this computer.

(ii) System DSN :- System DSN is visible to all users in the Machine including network service.

(iii) File DSN :- File DSN can be shared by other users who have the same drivers installed.

How to Create Database in Microsoft Access

→ Microsoft Access comes with Microsoft Office.
→ File → New for creating new database.
Type Database name: db1
click on create.

→ click right mouse button on table and select
design view for adding fields.

empno	number
ename	text
sal	number

Create datasource :-

1. control panel
2. Administrative tools.
3. odbc datasource

4. select userDSN → click on Add for creating new datasource.

5. Select driver → Microsoft Access Driver (*.mdb, *.accdb)

6. click on Finish.

Asking datasource name:- dsn1

Select database

OK → OK → OK.

```

// communicating with MS Access database using
// type1 driver.

import java.sql.*;
class JdbcTest36
{
    public static void main (String args[]) throws Exception
    {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection cn = DriverManager.getConnection
        ("jdbc:odbc:dsn1", "", "");
        Statement st = cn.createStatement();
        int count = st.executeUpdate("insert into
        emp values (101, 'Naresh', 5000)");
        System.out.println(count);
        cn.close();
        st.close();
    }
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

14.11.15

updateable resultset :-

- Resultset is having two modes
 - 1. read only mode.
 - 2. updateable Mode.
- Resultset which allows to read data but doesn't allow to modify is called read only resultset. Default resultset mode is read only.
- Resultset which allows to update data is called updateable resultset. This mode allows to insert, update and delete data from resultset and from database.

en. `createStatement(type, mode)`
en. `prepareStatement(SQL, type, mode)`

Resultset Interface

<u>types</u>	<u>Modes</u>
<code>TYPE_FORWARD_ONLY</code>	<code>CONCUR_READ_ONLY</code>
<code>TYPE_SCROLL_SENSITIVE</code>	<code>CONCUR_UPDATABLE</code>
<code>TYPE_SCROLL_INSENSITIVE</code>	

Inserting data/row into Resultset and into database :-

- Resultset provides following Methods

1. moveToInsertRow()

→ Move the cursor to the insert row. The current cursor position is remembered while the cursor is positioned on the insert row.

→ The insert row is a special row associated with an updateable result set.

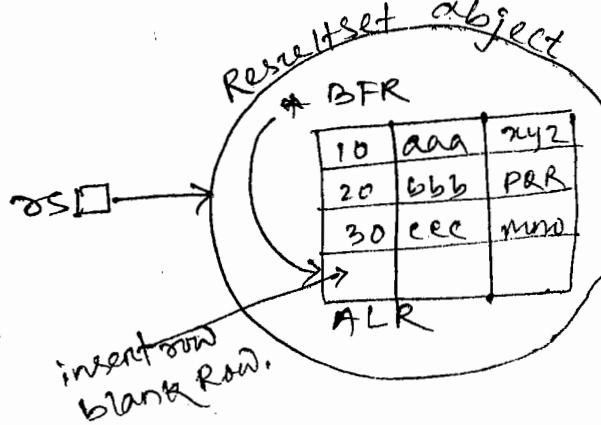
→ It is essentially a buffer where a new row may be constructed by calling the update methods prior to inserting the row into the result set.

Statement st = cn.createStatement(ResultSet.

TYPE_SCROLL_SENSITIVE,

ResultSet.CONCUR_UPDATABLE);

ResultSet rs = st.executeQuery("select * from dept");



2. Resultset provide update methods to replace values into resultset object. For each datatype it provides one update Method.

updateInt(columnIndex, int value)

updateFloat(columnIndex, float value)

updateDouble(columnIndex, double value)

updateString(columnIndex, String value)

3. void insertRow()

→ Inserts the contents of the insertRow into this Resultset object and into the database.

// Inserting row using ResultSet,

```
import java.sql.*;
class JdbcTest37
{
    public static void main (String args[])
    {
        Connection cn = null;
        Statement st = null;
        ResultSet rs = null;
        try
        {
            Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
            cn = DriverManager.getConnection ("jdbc:odbc:dsn1","","","");
            st = cn.createStatement
            (ResultSet.TYPE_SCROLL_SENSITIVE, ResultSet.CONCUR_UPDATABLE);
            rs = st.executeQuery ("select * from emp");
            rs.moveToFirst
            rs.moveToInsertRow ();
            rs.updateInt (1, 102);
            rs.updateString (2, "Sita");
            rs.updateFloat (3, 9000f);
            rs.insert
            rs.insertRow ();
        }
        catch (ClassNotFoundException e)
        {
            System.out.println (e);
        }
    }
}
```

```

        catch (SQLException s) {
            sopln(s);
        }
    Finally
    {
        try {
            if (rs != null)
                rs.close();
            if (st != null)
                st.close();
            if (en != null)
                en.close();
        }
        catch (SQLException k)
        {
            sopln(k);
        }
    }
}

```

How to delete row from Resultset and from database:-

Step 1:- Move the cursor to given row.

Step 2:- To call following Method to delete row

void deleteRow()

→ Deletes the current row from this ResultSet object and from the underlying database. This method can't be called when the cursor is on the insert row.

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

// Deleting row from Resultset object and from database.

```
import java.sql.*;
import java.util.*;
class JdbcTest38
{
    public static void main(String args[]) throws Exception
    {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection cn = DriverManager.getConnection
            ("jdbc:odbc:dsn1","","","");
        PreparedStatement ps = cn.prepareStatement
            ("select * from emp", ResultSet.TYPE_SCROLL_SENSITIVE,
             ResultSet.CONCUR_UPDATABLE);
        ResultSet rs = ps.executeQuery();
        Scanner scan = new Scanner(System.in)
        System.out.println("Input row");
        int row = scan.nextInt();
        boolean b = rs.absolute(row);
        if (b == false)
            System.out.println("invalid row number");
        else
            rs.deleteRow();
        rs.beforeFirst();
        while (rs.next())
            System.out.println(rs.getInt(1) + " " +
                rs.getString(2) + " "
                + rs.getFloat(3));
    }
}
```

Update row of ResultSet and Database :-

Steps

1. Move cursor to current row.
2. update values using update Methods.
3. void updateRow()

→ Updates the underlying database with the new contents of the current row of this ResultSet object.

// Program to update sal of Employee using ResultSet object.

```
import java.util.*;  
import java.sql.*;  
class JdbcTest39
```

```
{
```

```
    p.s.v.M(String args[]) throws Exception
```

```
{
```

```
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
```

```
    Connection cn = DriverManager.getConnection
```

```
( "jdbc:odbc: dsn1", "", "" ));
```

```
    PreparedStatement ps = cn.prepareStatement
```

```
( "select * from emp", ResultSet.TYPE_SCROLL_SENSITIVE,  
    ResultSet.CONCUR_UPDATABLE );
```

```
    ResultSet rs = ps.executeQuery();
```

```
    Scanner scan = new Scanner( System.in )
```

```
    System.out.println("Input row");  
    int row = scan.nextInt();
```

```
    boolean b = rs.absolute(row);
```

```
    if (b == false)
```

```
        System.out.println("Invalid row");
```

```

        else
    {
        System.out.println("Input update salary");
        float s = scan.nextFloat();
        rs.updateFloat(3, s);
        rs.updateRow();
        rs.beforeFirst();
        while (rs.next())
            System.out.println(rs.getInt(1) + " " + rs.getString(2) +
                               " " + rs.getFloat(3));
    }
}
}

```

TYPE_SCROLL_SENSITIVE :-

→ If ResultSet type is TYPE_SCROLL_SENSITIVE
 it allows to move cursor in forward and
 backward direction. It is sensitive to
 changes done in database does
 not effect resultset object.

TYPE_SCROLL_SENSITIVE

II. Example for TYPE_SCROLL_SENSITIVE

```

import java.util.*;
import java.sql.*;
class JdbcTest40
{
    public static void main(String args[])
    throws Exception
    {
        Class.forName("oracle.jdbc.driver.OracleDriver");
        Connection con = DriverManager.getConnection
        ("jdbc:oracle:thin:@localhost:1521:orcl", "Anjan",
         "Anjan");
    }
}

```

```

Statement st = cn.createStatement
(Resultset. TYPE_SCROLL_SENSITIVE, Resultset.
CONCUR_UPDATABLE);

Resultset rs = st.executeQuery("select empno,
ename, sal from emp");

Scanner scan = new Scanner(System.in)
sopln ("continue?");

String s = scan.next();
while (rs.next())
{
    sopln (rs.getInt(1) + " " + rs.getString(2) + " "
    + rs.getFloat(3));
}
}
}
}

```

Transaction Management :-

Q. What is Transaction?

Ans. An operation performed on database within

session is called transaction.

→ Group of statements executed within session

on database is called transaction.

→ In transaction all statements executed

on database or no statement executed on
database.

Connection provides the following methods to
Manage transactions:-

1 boolean getAutoCommit()

→ Retrieves the current auto-commit mode
for this connection object.

2. void setAutoCommit(boolean autoCommit)

→ Sets this connection's auto-commit mode to the given state.

→ Default auto commit mode is true.

// program to find auto commit mode.

```
import java.sql.*;
class JdbcTest41
{
    public static void main(String args[])
        throws Exception
    {
        Class.forName("oracle.jdbc.driver.OracleDriver");
        Connection cn = DriverManager.getConnection(
            "jdbc:oracle:thin:@localhost:1521:XE", "System",
            "Anjanika");
        boolean b = cn.getAutoCommit();
        System.out.println(b); // by default true
    }
}
```

// program to change Auto commit Mode :

```
import java.sql.*;
class JdbcTest42
{
    public static void main(String args[])
        throws Exception
    {
        Class.forName("oracle.jdbc.driver.OracleDriver");
        Connection cn = DriverManager.getConnection(
            "-----", -----, -----);
        boolean b = cn.getAutoCommit();
        if (b == true)
            cn.setAutoCommit(false);
    }
}
```

```

System.out.println(cn.getAutoCommit());
}
}

// Program to insert row.
import java.sql.*;
class JdbcTest43
{
    public static void main(String args[]) throws Exception
    {
        Class.forName("oracle.jdbc.driver.OracleDriver");
        Connection cn = DriverManager.getConnection(
            "jdbc:oracle:thin:@localhost:1521:XE",
            "system", "Anjan");
        cn.setAutoCommit(false);
        Statement st = cn.createStatement();
        int count = st.executeUpdate("insert into
            dept values (90, 'HR', 'Hyd')");
        System.out.println(count);
    }
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

→ The row is inserted in the session but not visible in the database or changes are not done in database.

How to commit changes on database.?

- changes can be done on database in two ways
 - 1. By calling commit() method.
 - 2. before closing connection.

void commit()

- Makes all changes made since the previous commit/rollback permanent and releases any database locks currently held by this connection object.

void close()

- Releases this connection object's database and JDBC resources immediately instead of waiting for them to be automatically released.
Before closing this method calls commit method.
- Before closing this method calls ~~then close~~.

// Example for commit.

```
class JdbcTest44
{
    public static void main(String args[])
    {
        Connection cn = null;
        Statement st = null;
        try
        {
            Class.forName("oracle.jdbc.driver.
                           OracleDriver");
            cn = DriverManager.getConnection("jdbc:
                oracle:thin:@localhost:1521:XE", "system", "Anjan");
        }
    }
}
```

```

        cn. setAutoCommit(false);
        st = cn.createStatement();
        int count = st.executeUpdate("insert into dept
                                     values (100, 'SALES', 'Hyd')");
        System.out.println(count);
        cn.commit();
        count = st.executeUpdate("delete from dept
                                 where deptno=90");
    }
}

catch (ClassNotFoundException e)
{
    System.out.println("Invalid driver class");
}
catch (SQLException e)
{
    System.out.println(e);
}
finally
{
    try {
        if (st != null)
            st.close();
        if (cn != null)
            cn.close();
    }
    catch (SQLException k)
    {
        System.out.println(k);
    }
}
}
}

```

Here the row is inserted, and ~~one~~ one row which deptno=90 is deleted, bcz of cn.close

rollback :-

void rollback()

- void rollback()

→ Undoes all changes made in the current transaction and releases any database locks currently held by this connection object.

void rollback(Savepoint savepoint)

- undoes all changes made after the given savepoint object was set.

Example: —

```
import java.sql.*;
```

class JdbcTest45

۳

{ p.s.v.m (String args[]) throws Exception }

3

```
class.forName("oracle.jdbc.driver.OracleDriver");
```

```
Connection cn = DriverManager.getConnection("...");
```

```
("jdbc:oracle:thin:@localhost:1521:XE", "system",  
 "Anjan"),
```

```
cn.setAutoCommit(false);
```

```
    st = cn.createStatement();
```

```
Statement st = cn.createStatement();
int count = st.executeUpdate("insert into
dept values(88,'aaa','hyd'));
```

gsm(count);

```
        saptm (contn),  
ResultSet rs = st.executeQuery("select *  
from dept");
```

```
while (re.next())
```

```
sopIn (rs.getInt(1) + " " + rs.getString(2) +  
" " + rs.getString(3));
```

```

    cn.rollback();
    rs1 = st.executeQuery("select * from dept");
    while (rs1.next())
        System.out.println(rs1.getInt(1) + " " + rs1.getString(2) + " "
                           + rs1.getString(3));
}
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

13.11.15 Savepoint :-

- Savepoint is an interface.
- It is implemented by JDBC driver developer.
- The representation of a savepoint, which is a point with the current transaction that can be referenced from the Connection.rollback method.
- When a transaction is rolled back to a savepoint all changes made after that savepoint are undone.

How to create Save-point ?

- Connection provide the following method.

Savepoint setSavepoint()

creates an unnamed savepoint in the current transaction and returns the new savepoint object that represents it

Savepoint setSavepoint(String name)

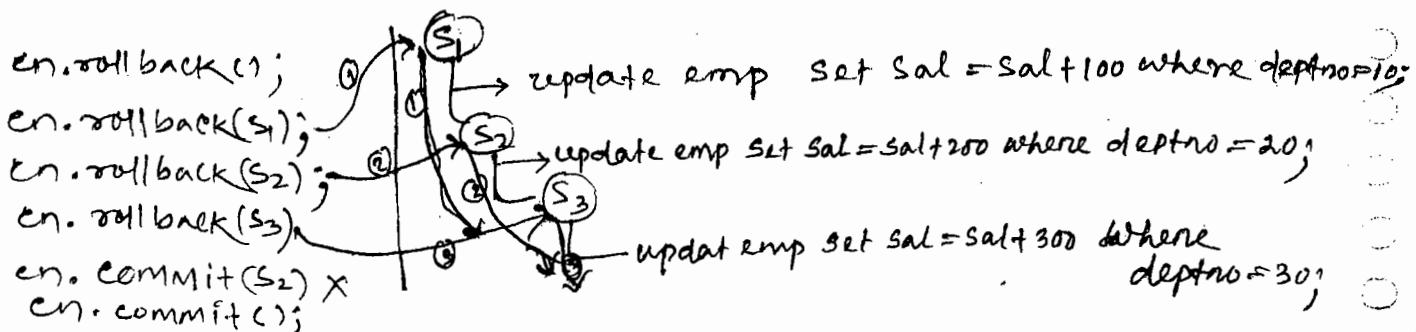
creates a savepoint with the given name in the current transaction and returns the new savepoint object that represent it.

// savepoint.

```
import java.sql.*;  
class JdbcTest46
```

```
{  
    public void main(String args[]) throws Exception,  
    {  
        Class.forName("oracle.jdbc.driver.OracleDriver");  
        Connection cn = DriverManager.getConnection  
        ("jdbc:oracle:thin:@localhost:1521:XE", "system",  
         "anjan");  
        cn.setAutoCommit(false);  
        Savepoint s1 = cn.setSavepoint();  
        Statement st = cn.createStatement();  
        int count = st.executeUpdate("update emp set  
        set sal=sal+100 where deptno=10");  
        System.out.println(count);  
        Savepoint s2 = cn.setSavepoint();  
        count = st.executeUpdate("update emp set sal=  
        sal+200 where deptno=20");  
        System.out.println(count);  
        Savepoint s3 = cn.setSavepoint();  
        count = st.executeUpdate("update emp set sal=  
        sal+300 where deptno=30");  
        System.out.println(count);  
        cn.rollback(s2);  
        cn.close();  
    }  
}
```

Session/connection



Type-1 Driver (CSV → comma separated value)

CSV File :-

→ CSV is text file.

→ It is a data file which contains data separated with , (comma).

How to read data from CSV file?

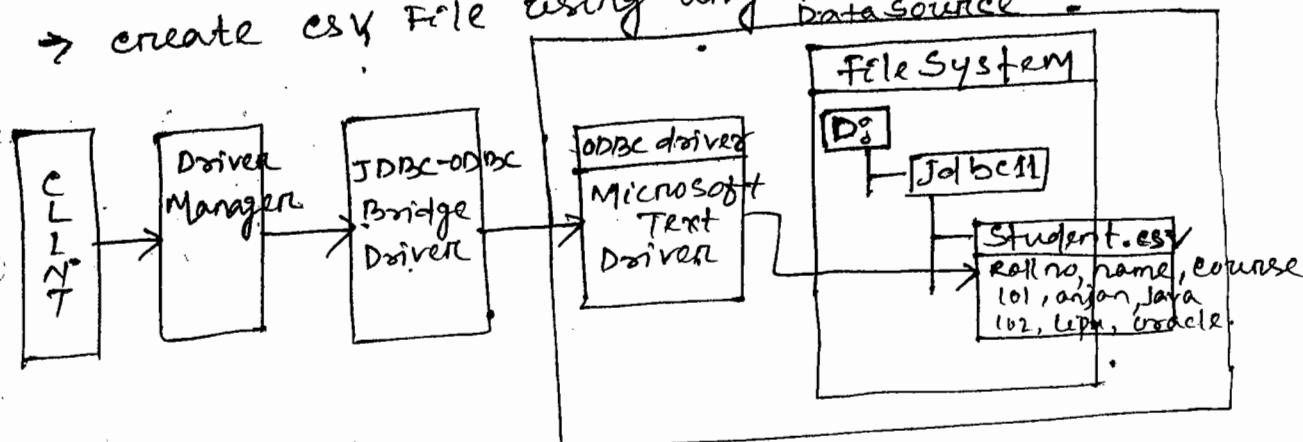
→ Type-1 driver is used to read data from CSV file.

→ Microsoft provides ODBC driver to manipulate contents of CSV file.

Microsoft Text Driver (.txt, CSV)

→ Create CSV file using any text editor.

DataSource



Create first data source

↳ `nitdsn1 (dataSource name)`

Program

```
import java.sql.*;  
class JdbcTest47  
{ p. s. v. m. (String args[]) throws Exception  
{  
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");  
    Connection cn = DriverManager.getConnection  
        ("jdbc:odbc:nitdsn1", "", "");  
    Statement st = cn.createStatement();  
    ResultSet rs = st.executeQuery ("Select * from  
        [Student.csv]");
```

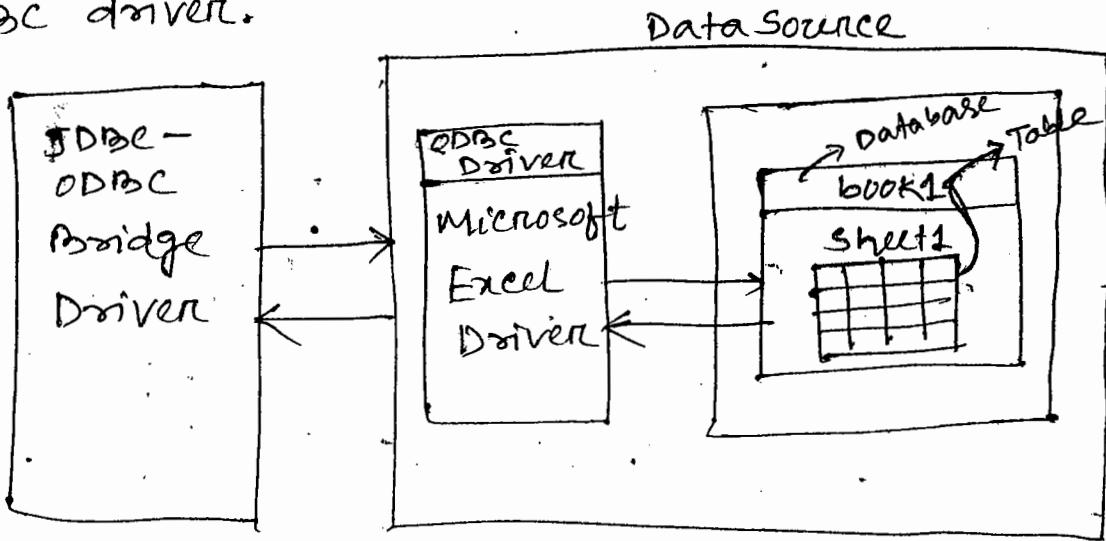
```

while (rs.next())
    System.out.println(rs.getInt(1) + ":" + rs.getString(2) + ":" + rs.getString(3));
    st.close();
    cn.close();
}
}

```

How to Manipulate data exist in Microsoft Excel :-

→ Microsoft provide Microsoft Excel Driver which is ODBC driver.



Program

```

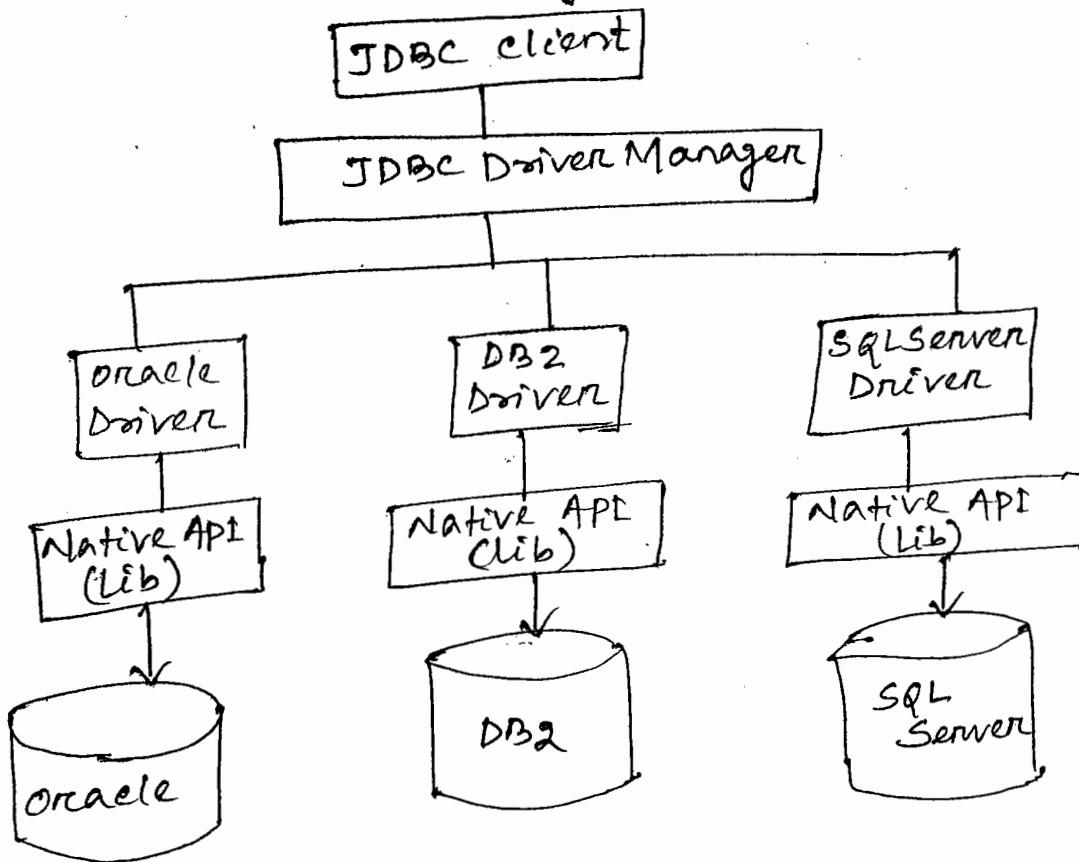
import java.sql.*;
class JdbcTest48
{
    public static void main (String args[]) throws Exception
    {
        Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection cn = DM.getConnection ("jdbc:odbc:nitdex2",
                                         "", "");
        Statement st = cn.createStatement ();
        ResultSet rs = st.executeQuery ("select * from
                                         [sheet1]");
        while (rs.next())
            System.out.println (rs.getInt(1) + ":" + rs.getString(2) + ":" + rs.getString(3));
    }
}

```

```
    st.close();
    cn.close();
}
}
```

Type-2 Driver:-

→ Native API and partly java drivers.



Oracle Type-2 Driver

~~Driver Class Name~~: Oracle.jdbc.driver. OracleDriver

URL: jdbc:oracle:thin → type-4

URL: jdbc: oracle:oci → type-2

Path:- C:\oracle\product\10.2.0\db_1\bin

ocijdbc10.dll (Native API)

oci → Oracle Call Interface

Set path = C:\oracle\product\10.2.0\db_1\bin
ocijdbc10.dll;%path%;

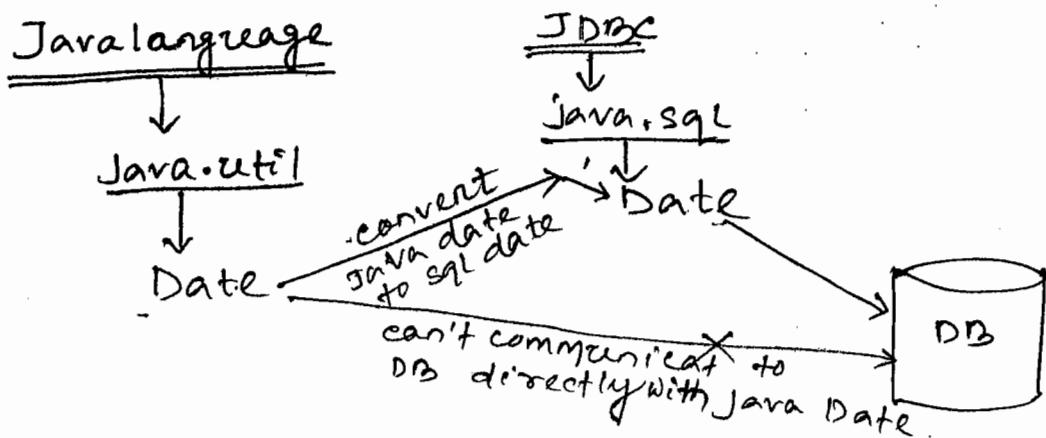
Using type-2 driver for oracle

```
import java.sql.*;
class JDBCTest49
{
    public static void main (String args[])
        throws Exception
    {
        Class.forName ("oracle.jdbc.driver.OracleDriver");
        Connection cn = DriverManager.getConnection
            ("jdbc:oracle:oci:@serveror system", "anjan");
        System.out.println ("connection established");
        Statement st = cn.createStatement();
        ResultSet rs = st.executeQuery ("select empno,
            ename, sal from emp");
        while (rs.next())
            System.out.println (rs.getInt(1) + ":" + rs.getString(2) + ":" + rs.
                getFloat(3));
        st.close();
        cn.close();
    }
}
```

before executing ~~setpath~~ set path.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Working with date type :-



java.util.Date

- The class Date represents a specific instant in time with millisecond precision.
- It allowed the interpretation of data as year, month, day, minute and second value.

long getTime() → Method

- Returns the number of milliseconds since January 1, 1970, 00:00:00 GMT represented by this Date object.

```
Date d = new Date();
long l = d.getTime();
```

java.sql.Date program :-

program :-

```
import java.util.*;
class Date Test
{
    p. s. v. m (String args[])
    {
        Date d1 = new Date();
        sopln(d1);
        long l = d1.getTime();
    }
}
```

SRI RAJENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

Sopln(1);
Date d2 = new Date(1);
Sopln(d2);
}
}
}

```

Java.sql.Date

→ A thin wrapper around a millisecond value that allows JDBC to identify this as an SQL DATE value.

Date (long date) → constructor.

constructor a Date object using the given milliseconds. time value

// program to insert Date into DB

```

import java.util.*;
import java.sql.*;
class JdbcTest50
{
    p. s. v. m (String args[])
throws Exception.
    {
        class.forName ("oracle
connection cn = DM.getconnection

```

```

preparedstatement ps = cn.prepareStatement
("insert into sample values (?)")
java.util.Date jd = new java.util.Date();
long l = jd.getTime();
java.sql.Date sd = new java.sql.Date(l);

```

Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road
Ameerpet, Hyderabad.

```

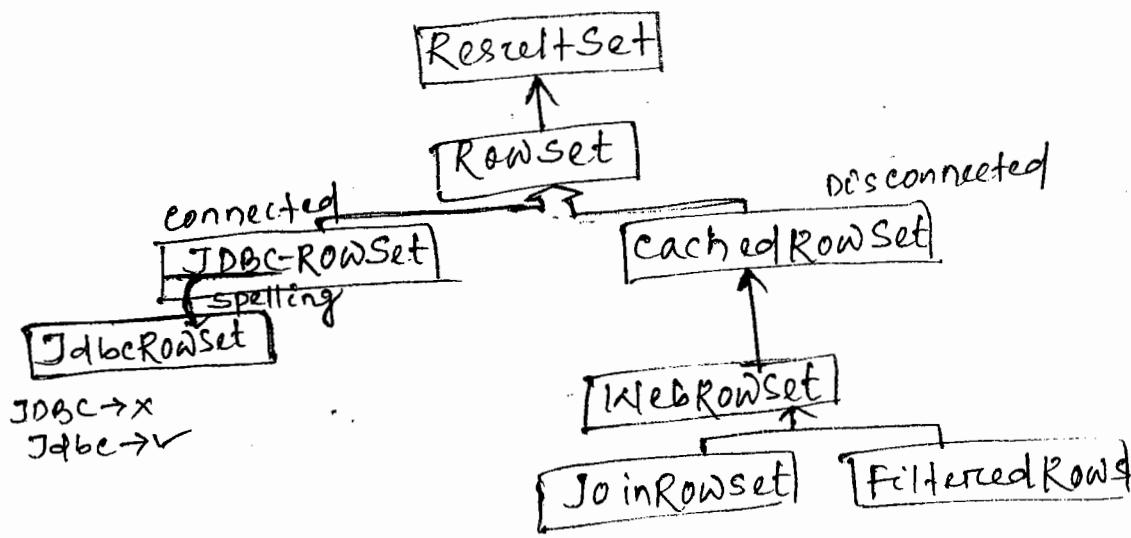
    ps.setDate(1, sd);
    int count = ps.executeUpdate();
    System.out.println(count);
}
}
}

```

19.12.15

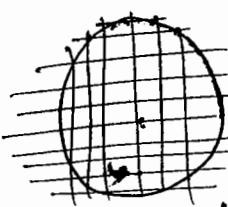
RowSet

- Rowset is an Interface.
- This interface exist in javax.sql package.
- The Interface that adds support to the JDBC API for the JavaBeans component Model.
- A rowset, which can be used as a JavaBeans component in a visual Beans development environment, can be created and configured at design time and executed at run time.
- The Rowset interface provides a set of JavaBeans properties that allow a Rowset instance to be configured to connect to a JDBC data source and read some data from the data source.
- There are 2 types of Rowset.
 1. Connected Rowsets
 2. Disconnected Rowsets.



- A Rowset object may make a connection with a datasource and maintain ~~with a data source~~ and that connection throughout its Lifecycle. in which case it's called a connected rowset.
- A rowset may also make a connection with a data source, get data from it, and then close the connection. Such a rowset is called a disconnected rowset.

Methods available in Rowset :-



① **void setCommand (String cmd)**

→ Sets this Rowset object's command property to the given SQL query.

② **void setUrl (String url)**

→ Sets the URL this Rowset object will use when it uses the DriverManager to create a connection.

③ **void setUsername (String name)**

→ Sets the username property for this Rowset object to the given string.

④ **void setPassword (String password)**

→ Set the database password for this Rowset object to the given string

⑤ **void execute()**

→ Fills the Rowset object with data.

JdbcRowset :-

- A wrapper of a ResultSet object that makes it possible to use the result set as a JavaBeans component.
- JdbcRowset is a connected rowset, that is, it continually maintains its connection to a database using a JDBC technology-enabled driver, it also effectively makes the driver a JavaBeans component.
- JdbcRowset is an interface and implemented by JDBC^{driver} developer.

// How to create a JdbcRowset.

```
import javax.sql.rowset.*;
import java.sql.*;
class JdbcTest
class JdbcTest51
{
    p. s. v. Main(String args[])
throws Exception
{
    JdbcRowSetImpl();
    JdbcRowSet rw = new OracleRowSet();
    rw.setURL("jdbc:oracle:thin:@localhost:1521:XE",
              "system", "anjan");
    rw.setUser("system");
    rw.setPassword("tiger");
    rw.setCommand("select empno, ename, sal from emp");
    while (rw.next())
        System.out.println(rw.getInt(1) + ":" + rw.getString(2) + ":" +
                           rw.getFloat(3));
}
```

(Not executed)

CachedRowSet :-

- A cachedRowset object is a container for rows of data that caches its rows in memory, which makes it possible to operate without always being connected to its data source.

Program

```
import javax.sql.rowset.*;
import java.sql.*;
import java.io.*;
import oracle.jdbc.rowset.*;

class JdbcTest52
    p.s.v.m (String args[]) throws Exception
{
    RowSet rw = new OracleCachedRowset();
    OracleCachedRowset rw = new OracleCachedRowset();
    rw.setURL ("jdbc:oracle:thin:@localhost:1521:XE");
    rw.setUsername ("system");
    rw.setPassword ("Anjan");
    rw.setCommand ("select empno, ename, sal
                    from emp");
    while (rw.next())
        System.out.println (rw.getInt(1) + ":" + rw.getString(2) + ":" +
                            rw.getFloat(3));
}

fileOutputStream fos = new FileOutputStream ("file1");
ObjectOutputStream oos = new ObjectOutputStream (fos);
oos.writeObject (rw);

}}
```

```

C // Read Rowset object from file.
C
C import javax.sql.*;
C import java.sql.rowset.*;
C import oracle.jdbc.rowset.*;
C import java.io.*;
C class JDBCtest53 {
C     public static void main (String args[]) throws Exception {
C         {
C             FileInputStream fis = new FileInputStream ("file1");
C             ObjectInputStream ois = new ObjectInputStream (fis);
C             OracleCachedRowSet row = (OracleCachedRowSet)
C                 ois.readObject();
C             row.execute();
C             while (row.next())
C                 System.out.println (row.getInt(1) + ":" + row.getString(2) + ":"
C                     + row.getFloat(3));
C         }
C     }
C }

```

WebRowSet:-

→ A WebRowSet is an extension to CachedRowSet. The WebRowSet interface provides support for the production and consumption of result sets and their synchronization with the data source, both in Extensible Markup Language (XML) format and in disconnected fashion. It allows result sets to be shipped across file and over internet protocol.

Program :- Writing data in xml format

```
import javax.sql.rowset.*;  
import javax.sql.*;  
import java.io.*;  
import java.sql.*;  
import java.jdbc.rowset.*;  
  
class Test54  
{  
    public static void main (String args []) throws Exception,  
    {  
        Class.forName ("oracle.jdbc.driver.OracleDriver");  
        OracleWebRowSet rw = new OracleWebRowSet();  
        Connection cn = DM.getConnection ("-", "-", "-");  
        Statement st = cn.createStatement();  
        ResultSet rs = st.executeQuery ("select empno,  
                                         ename, sal from emp");  
        rw.populate (rs);  
        FileWriter fw = new FileWriter ("file2.xml");  
        rw.writeXML (fw);  
        fw.close();  
    }  
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Servlet

Servlet - JEE

JSP - JEE

JDBC - JSE

→ Servlets, JSP are called web technologies. ~~These~~

Q. what is Web application?

→ Web application is a collection of programs, whose resources are accessed using Internet/intranet.

Advantages :-

1. sharing resources
2. providing 24/7 services.

Types of Web Applications :-

There are 2 types of Web Applications

- (1) Presentation Oriented Web Application
- (2) Service Oriented Web Application.

Presentation Oriented Web Application :-

- It is a collection of static resources.
- These resources can be
- (i) html files
 - (ii) images
 - (iii) audio files
 - (iv) video files

- These web application provide content.
- these are not interactive webapplications.
eg. tutorial.

SRI RAHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

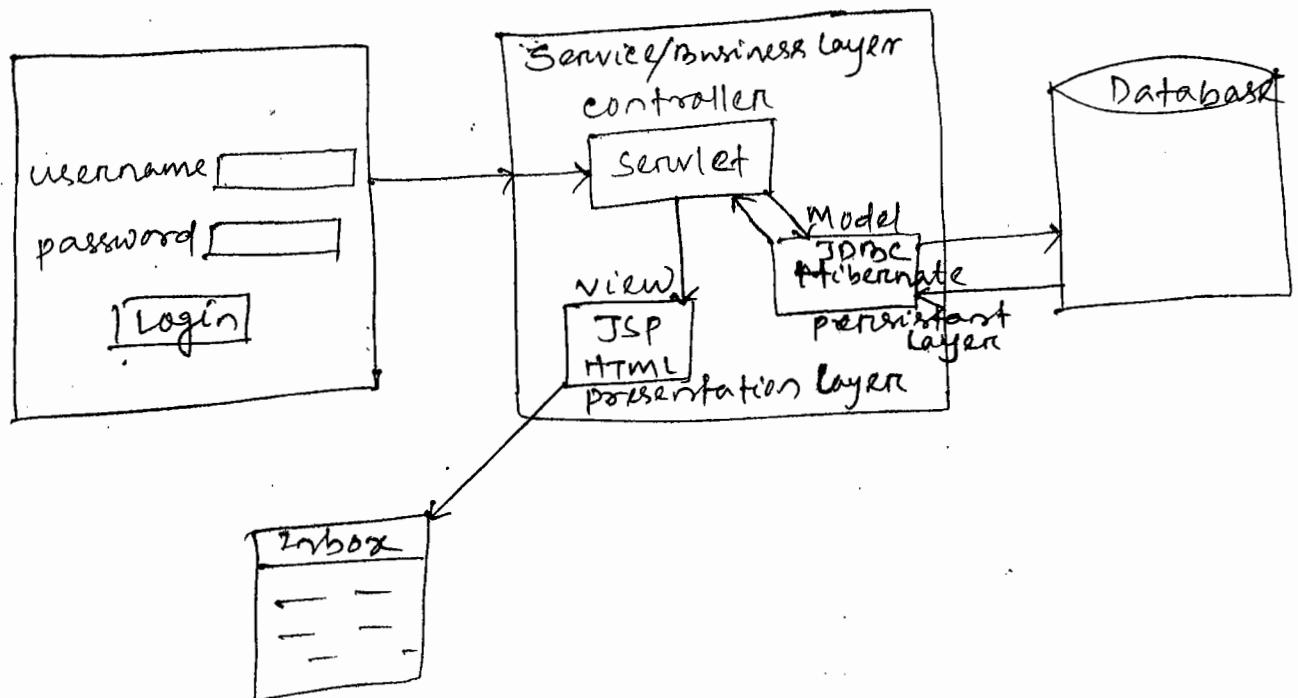
Service Oriented Web Application :-

- These are interactive web applications.
- It is a collection of static resources and dynamic resources.
- This application read input from client process it and generate output. This output changes from one ip to another.
- It consists of,
 - html files
 - images
 - audio file
 - video file
 - servlet
 - JSP
 - ejb
 - XML/webservices

UNIVERSITY
Software Languages Material Available
Beside Bangalore Material Available
Opp. CDAC, Ayyagar Bakery,
Ameerpet, Hyderabad.

Web Application Architecture:-

- Web Application is developed using MVC design
- Web application is developed using MVC design pattern.
 - MVC stands for Model-view-controller.
 - Web application is divided into 3 modules/layers.
 - (i) presentation logic/layer.
Business
 - (ii) service/ logic/ layer.
 - (iii) persistent logic/ layer.



- controller which receive all request for the application and is responsible for taking appropriate action in response to each request.
- Model uses JDBC in-order to communicate with database. It consists of persistant logic.
- HTML or JSP files are used to code the presentation.

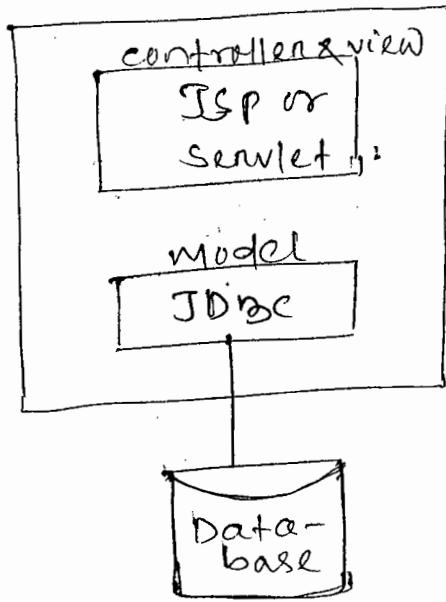
Types of MVC design pattern:-

1. MVC Model - 1
2. MVC Model - 2

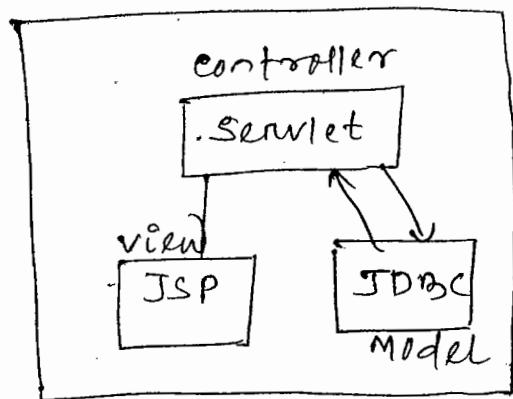
SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

MVC Model - 1

→ page centric



MVC Model - 2



Date:- 27.10.15

Client/Server :-

→ In networking we required two programs

1. Client.

2. Server.

→ Client is a program which send request to Server.

→ Server receive request and process it

generate response (output).

→ This client/server communication is done using ~~using~~ protocols.

Q. What is a protocol?

→ Protocol define set of rules and regulation, in order to communication between Client and Server.

Application Protocols :-

1. HTTP - Hyper Text Transfer Protocol
2. FTP - File Transfer Protocol.
3. SMTP - Simple Mail Transfer protocol.
4. POP - Post Office Protocol.

- In Internet client user browser in order to communicate with server.
- In Internet server is HTTP Server.
- By default HTTP server is having static resource handler which handle static resources.
- A resource which is downloaded browser is called static resource. These resources are not executed within server.

Q. What is need of server side technologies?

- In order to extend functionality of server.
- To develop dynamic Web site or service oriented webapplication or interactive webapplications.
- In order to read information send by client process it and generate output.

Server Side Technologies :-

1. ASP
2. PHP
3. JSP
4. Servlets
5. CGI

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

CGI

- Servlet technology was introduced to overcome the shortcomings of CGI technology.
- CGI is an abstracts or specifications used to develope programs at Server Side. This technology is developed using HTTP protocol.
- These programs are written in 'C' language
- CGI programs are process based.
- For each request send by client CGI container create a process to run the programme.
- These programs are not efficient.

Servlet :-

- Servlet is a server side Technology.
- Servlet is a program which receive request from client, process it and generate output.
- Servlet is a web technology.
- Servlet is an API given by sun Micro
- System for developing servlet and servlet container.
- Servlet is a specification or abstracts given by SUN for developing servlet.
- Servlet is used for generating dynamic content or dynamic Web pages.

what is Servlet?

- Servlets are protocol and platform independent server-side software components, written in Java.
- Servlet is a Java program used to generate dynamic components (audio, video)

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Servlet API History :-

Servlet API version	Released	Platform	Important changes
Servlet 3.0	May 2013	JavaEE 7	Non-blocking I/O, HTTP protocol upgrade mechanism.
Servlet 3.0	DEC 2009	JavaEE 6, JavaSE	pluggability, easy of development, Async servlet, security, file upload
Servlet 2.5	Sep 2005	JavaEE 5, JavaSE 5	Requires Java SES, supports annotation!
Servlet 2.4	Nov 2003	J2EE 1.4, J2SE 1.3	web.xml uses XML Schema
Servlet 2.2	Aug 1999	J2EE 1.3, J2SE 1.2	Addition of Filter
Servlet 2.1	NOV 1998	J2EE 1.2, J2SE 1.2	Become part of J2EE introduced in dependent Web app in .war files
Servlet 2.0		Unspecified	First official specification, servlet context.
Servlet 1.0	June 1997	JDK 1.1	Part of Java Servlet Development Kit 2.0

Advantages of Servlets :-

Advantages Of Servlets :-

(i) Efficiency :-

→ More efficient uses lightweight Java threads

(ii) Persistence :-

Servlets remain in memory, servlets can maintain state between requests.

(iii) Portability :-

Since servlets are written in Java they are platform independent.

(iv) Robustness :-

Error handling, ~~and~~ garbage collector prevent problems with memory leaks.
are the

Q. what ~~is~~ Advantages of servlet over CGI ?

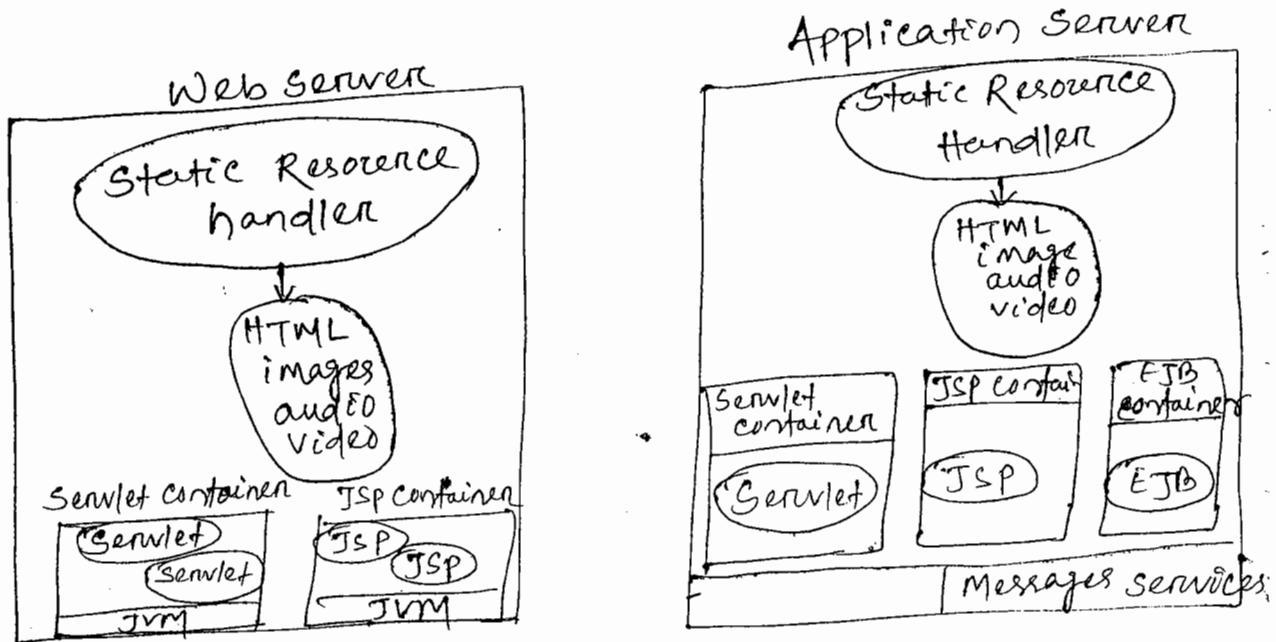
- A servlet doesn't run in a separate process. This removes the overhead of creating a new process for each request.
- A servlet stays in memory between requests. A CGI program (and probably also an extensive runtime system or interpreter) needs to be loaded and started for each CGI request.

- There is only a single instance which answers all requests concurrently. This saves memory and allows a servlet to easily manage persistent data.

Servers:-

~~There are 2 types~~

- In Internet Server is an http server.
- Server's responsibility is serving application resources to one or more than one client.
- By default a server can serve only static resources.
- In order to serve dynamic resources it requires containers.
- These servers are 2 types.
 - Web Server.
 - Application Server.

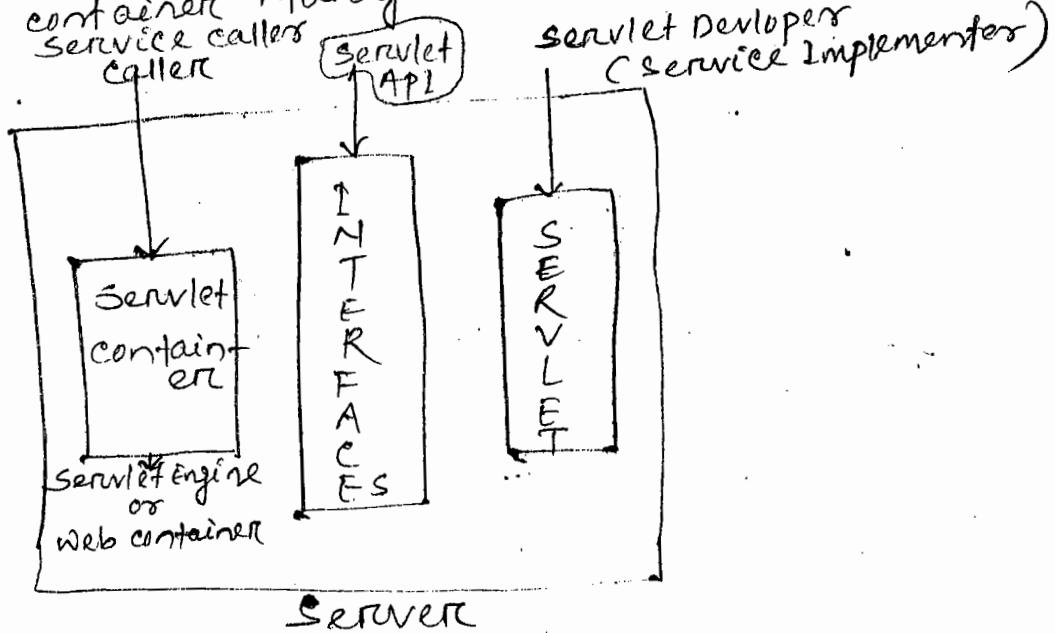


Q. What is difference b/w Web server and Application Server?

- A Web server responsibility is to handle HTTP requests from client browsers and respond with HTML response. A web server understands HTTP languages and runs on HTTP protocol.
- A Web server is having specific types of containers which can execute Servlets and JSP. (Servlet container and JSP container)
- Application server provides additional features such as enterprise JavaBeans support, JMS messaging support, Transaction Management etc. So we can say that application server is a web server with additional functionalities to help developers with enterprise applications.

Q. What is Servlet Container?

- Server uses a separate module for executing servlets called Servlet container.
- Server container manages execution of Servlet.



Types of Servlet containers :-

→ It is 3 types

- (i) Standalone Servlet container.
- (ii) in-process servlet container.
- (iii) out-process servlet container.

Standalone servlet container :-

→ In standalone servlet container server and container both are one program.

Ex:- Apache Tomcat Server.

In process servlet container :-

→ Servlet container and server are two different programs.

→ Servlet container runs within process of Server.

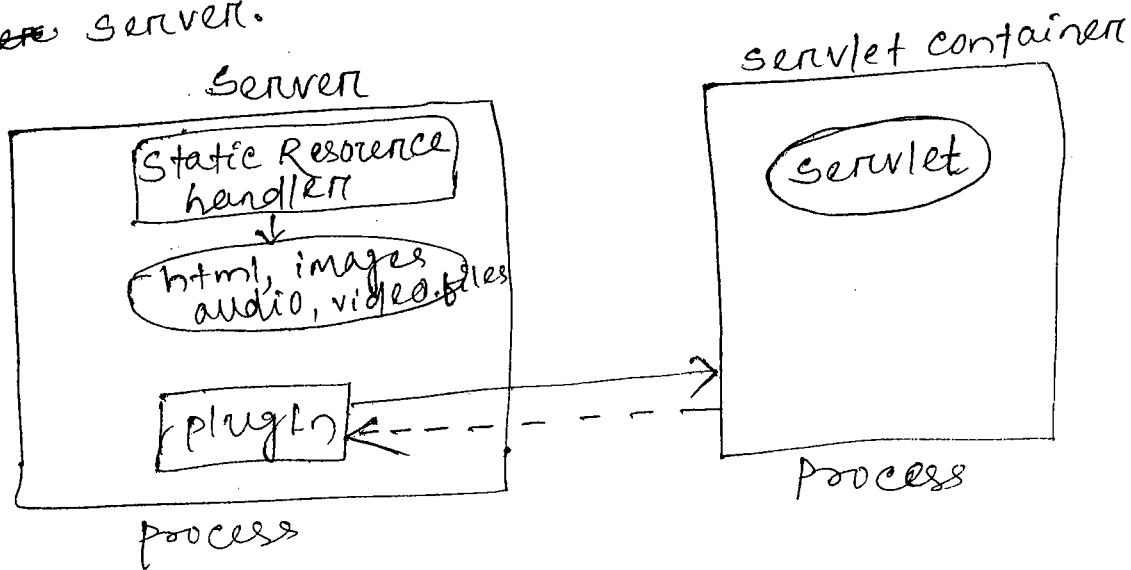
e.g:- Web logic, JBoss.

out process servlet container :-

→ Servlet container and server are two different programs.

→ Servlet container runs outside the process of Server.

→ Server uses plugins in order to communicate with servlet container which runs outside server.



List of Servers available in Market:-

1. apache tomcat - Webserver - Java based.
2. weblogic - application server - Java based.
3. JBOSS - application server - Java based.
4. GlassFish - Application server - Java based.
5. IWS (java web server) - web server - Java based.
6. Oracle 10g Application server - Application server - Java based.
7. IIS - Web server - non Java based.
8. pws - Web server - non Java based.
9. pramati - Web server - Java based
10. Web Sphere - Web server - Java based.

Q who provide Servlet API ?

→ Servlet API is provided by server.
→ It comes with Web server or application server.
 → tomcat - servlet-api.jar
 → weblogic - weblogic-api.jar.

→ Servlet API consists of two packages

1. javax.servlet
2. javax.servlet.http

→ Sun Micro System has given Servlet API for developing two types of Servlets.

1. generic Servlets (Protocol Independent Servlet)

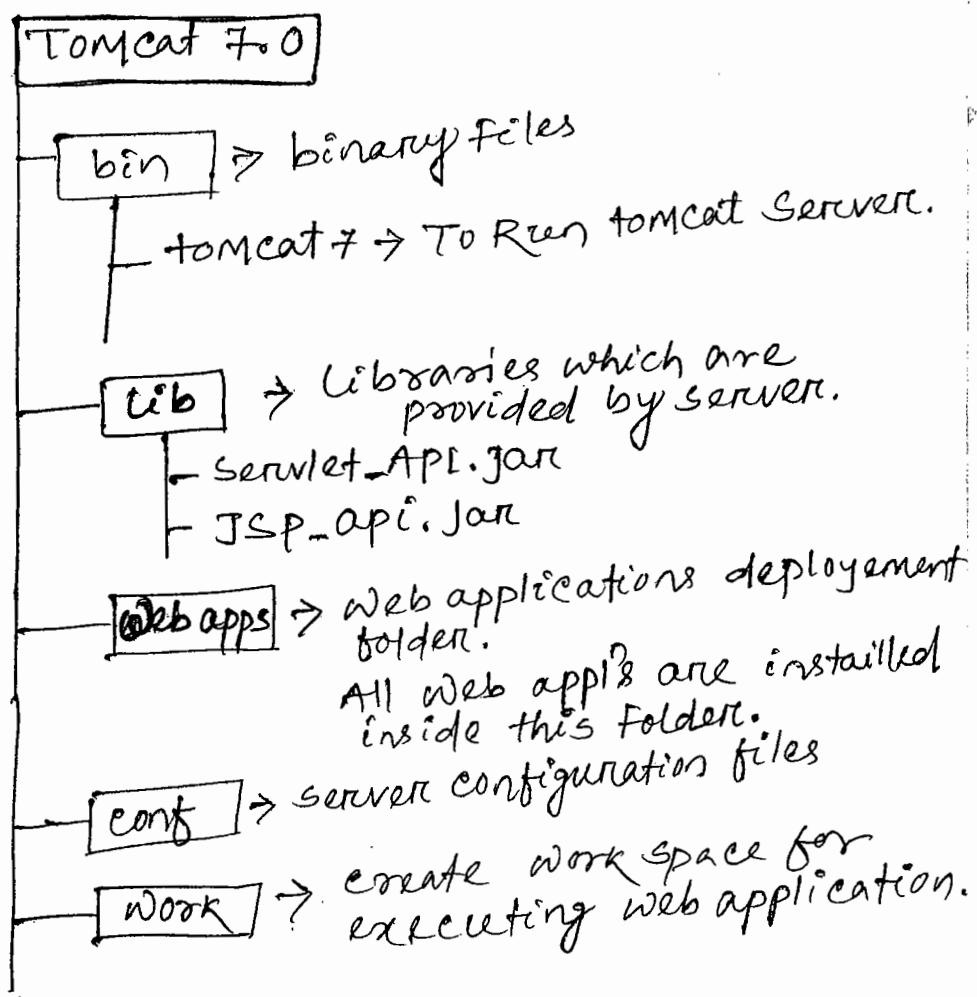
2. http Servlets.

- ⇒ Servlet API provided by Sun Micro System is extensible.

~~Q & A~~

Working with tomcat Server :-

- ⇒ Download tomcat web server from apache website
tomcat 8.0 - Jdk 1.7 or higher version.
tomcat 7.0 - Jdk 1.7 or higher version.
tomcat 6.0 - Jdk 1.5, Jdk 1.6
tomcat 5.0
- ⇒ Exploring tomcat folder or directory :-



JAVA_HOME :-

- JAVA_HOME is a Java environment variable.
- This variable is used by other application softwares to locate Java softwares.

JAVA_HOME = c:\program files\java\jdk1.7.0
(location)

JRE_HOME

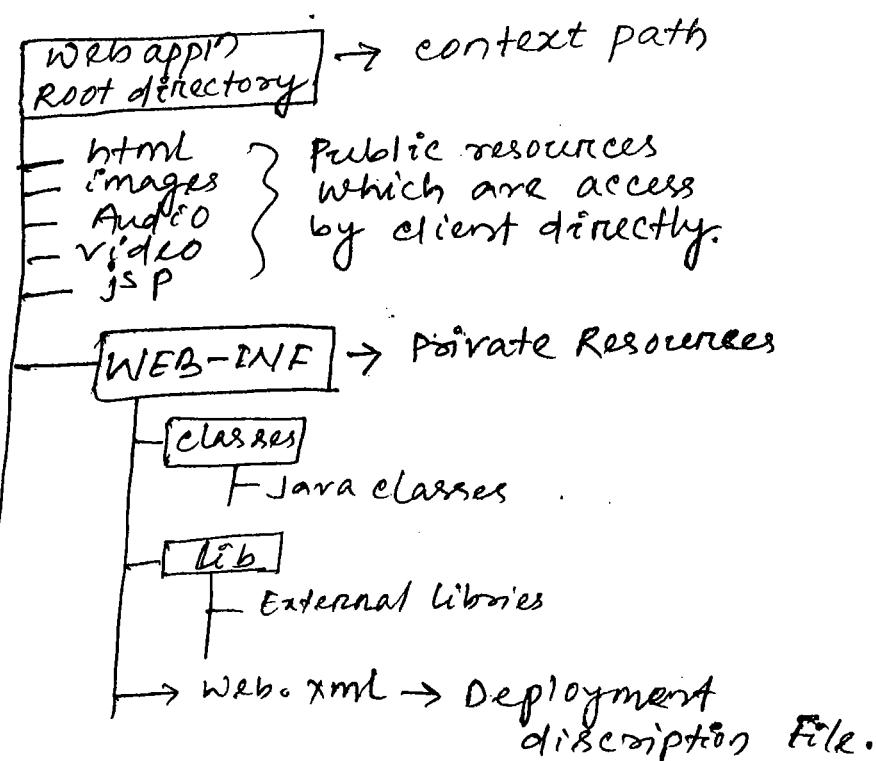
- It is a Java environment variable.
- It is used by other application softwares to locate Java runtime environment.

JRE_HOME = location

JRE_HOME = c:\Program files\java\jre7

Web Application directory structure :-

- Web application is a collection of programs and this programs are organized by following the specification given by Sun Microsystems.



→ A simple Web Application program.

→ Webapplication → root directory
(or)
content path
Welcome.html.

```
<html>
<body bgcolor = cyan>
<font color = blue size = 6>
This is my first webapplication
</font>
</body></html>
```

deployment :-

→ Installing webapplication into server this process is called deployment.

Types of deployment

→ An appln can be deployed into server using three methods.

1. hot deployment
2. console deployment.
3. tool deployment.

hot deployment :-

copying web application and pasting inside deployment folder of server is called hot deployment.

ctrl + C → copy

ctrl + V → paste

console deployment :-

- deploying Web application into Server using console Manager.
- servers provide console Manager.

tool deployment

- deploying Web application into server using deployment tools.
 - ANT, MAVEN.

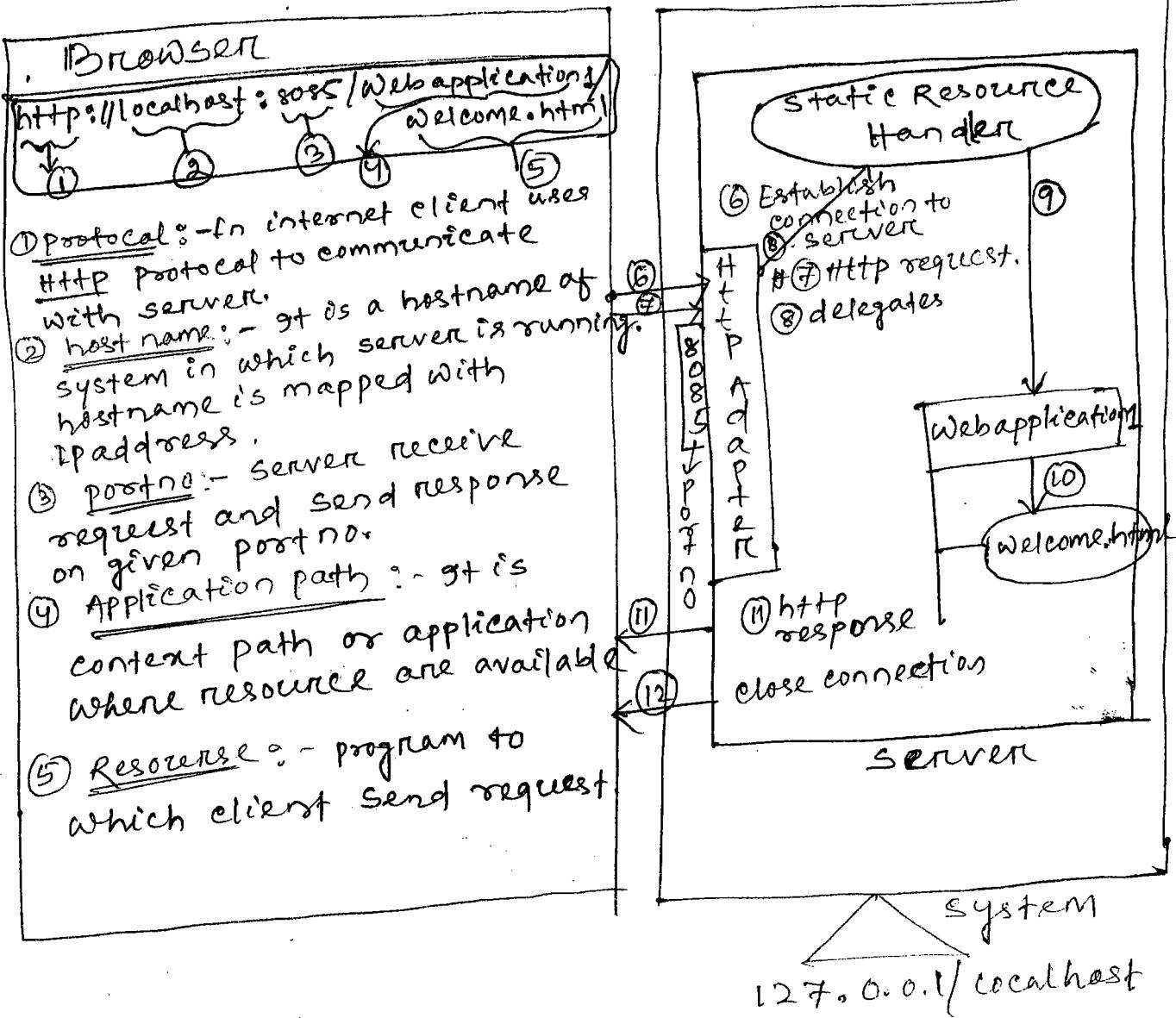
Modes of deployment :-

- Web application is deployed in two modes
 1. development Mode.
 2. production mode/compressed Mode.
- In development Mode web application directory structure is copied into server.
- In production mode web application is compressed and deployed into server.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Q/How to change server port no?

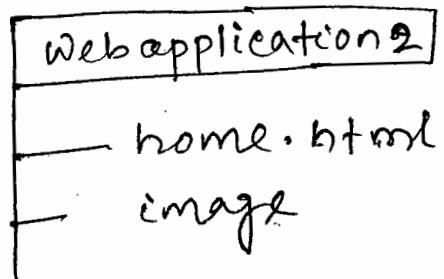
- Go to tomcat 7.0/conf/server.xml
- open server.xml
- change connector port no.
- connector port = "8085"



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Program 2

program for putting image and show it on browser.



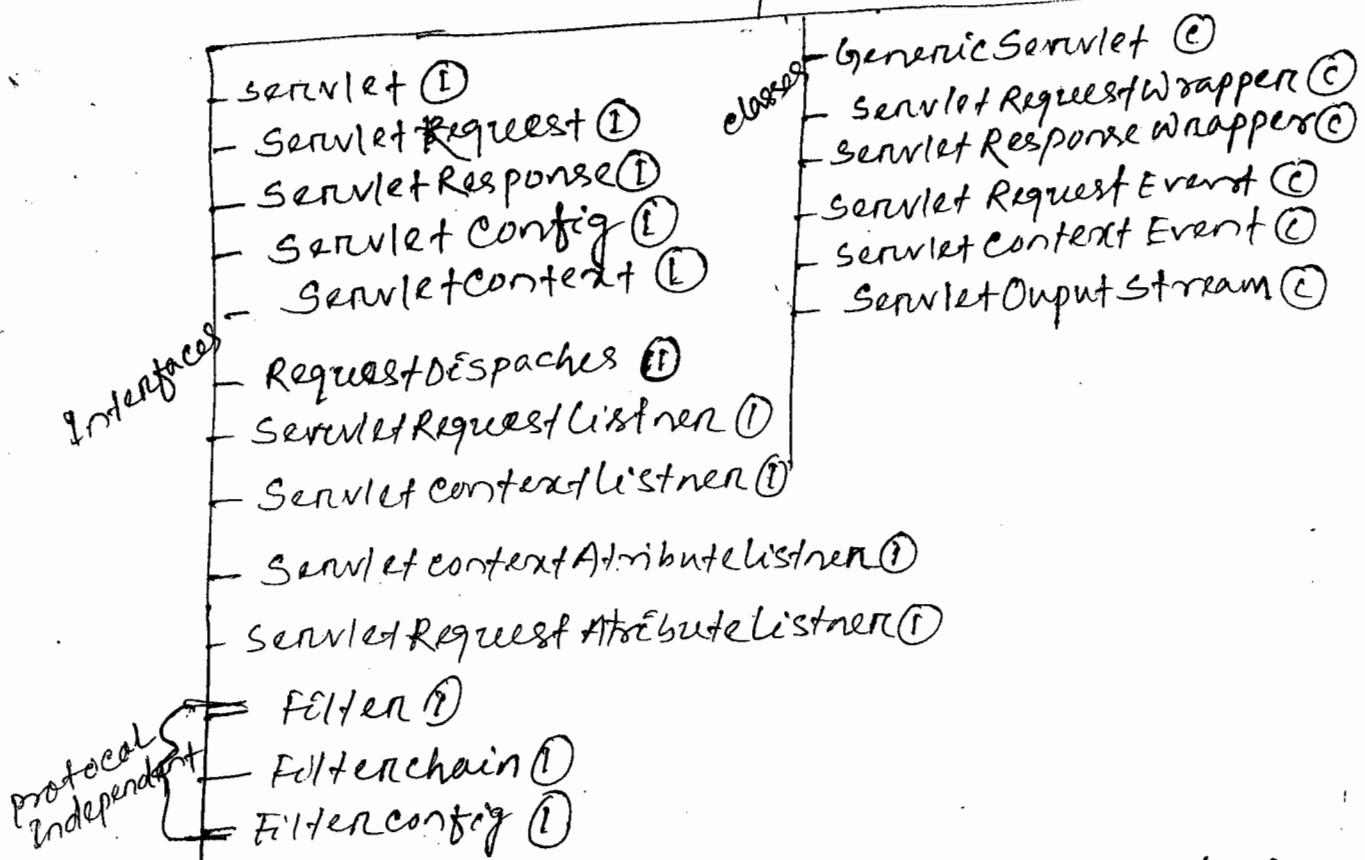
```
<html>
<body bgcolor = cyan>
<img src = "x.jpg" width = 200
      height = 300> </img>
</body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

How to develop Servlet :-

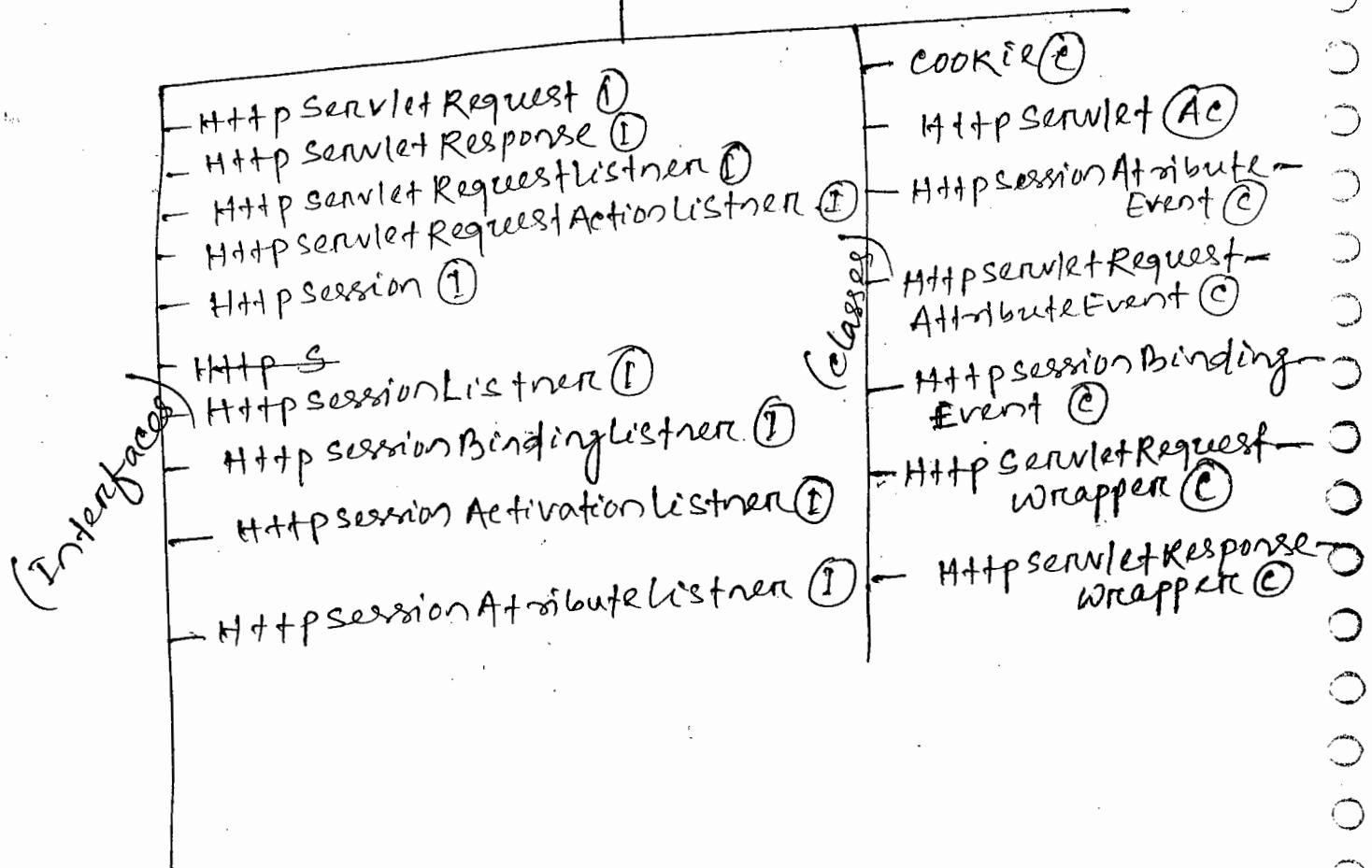
- In order to develop Servlet servlet API provide two packages
 - 1. javax.servlet package.
 - 2. javax.servlet.http.
- javax.servlet package used for developing protocol independent Servlets

Javax.servlet package



- javax.servlet.http package used for developing http Servlets or protocol dependent Servlets

Javax. Servlet. http



Servlet Interface

- It is a root interface for all servlets.
- Every servlet must be inherited from servlet interface.
- Servlet interface provides 5 methods
 - 3 Methods are called lifecycle Methods
 - 2 Methods are called non lifecycle Methods.

Q: What is lifecycle method?

- A Method which is executed by servlet container is called lifecycle Method.

- Q. what is non Lifecycle Method?
→ A method not executed by servlet container is called non Lifecycle Method.

Methods of Servlet Interface:-

(1) `public void init (ServletConfig config)`
throws ServletException.

- It is a lifecycle Method of servlet.
- called by the Servlet container to indicate to a servlet that the servlet is being placed into service.
- This method is called servlet container only once.
- The servlet container calls the init method exactly once after instantiating the servlet. The init method must complete successfully before the servlet can receive any requests.
- servlet is initialized by calling init method

(2) `public void service (ServletRequest req,
ServletResponse res)`
throws IOException.

- It is a lifecycle Method.
- This method is called by servlet container to process the request of client.
- called by the servlet container to allow the servlet to respond to a request.
- Operation performed by servlet is defined inside Service Method.

→ this method is called by servlet container by creating thread.

(3) public void destroy()

→ it is a lifecycle method this method is called by servlet container before servlet object is removed.

→ This method is called only once.

(4) public String getServletInfo()

→ Returns information about the servlet such as author, version and copyright.

→ It is a ~~non lifecycle~~ Method.

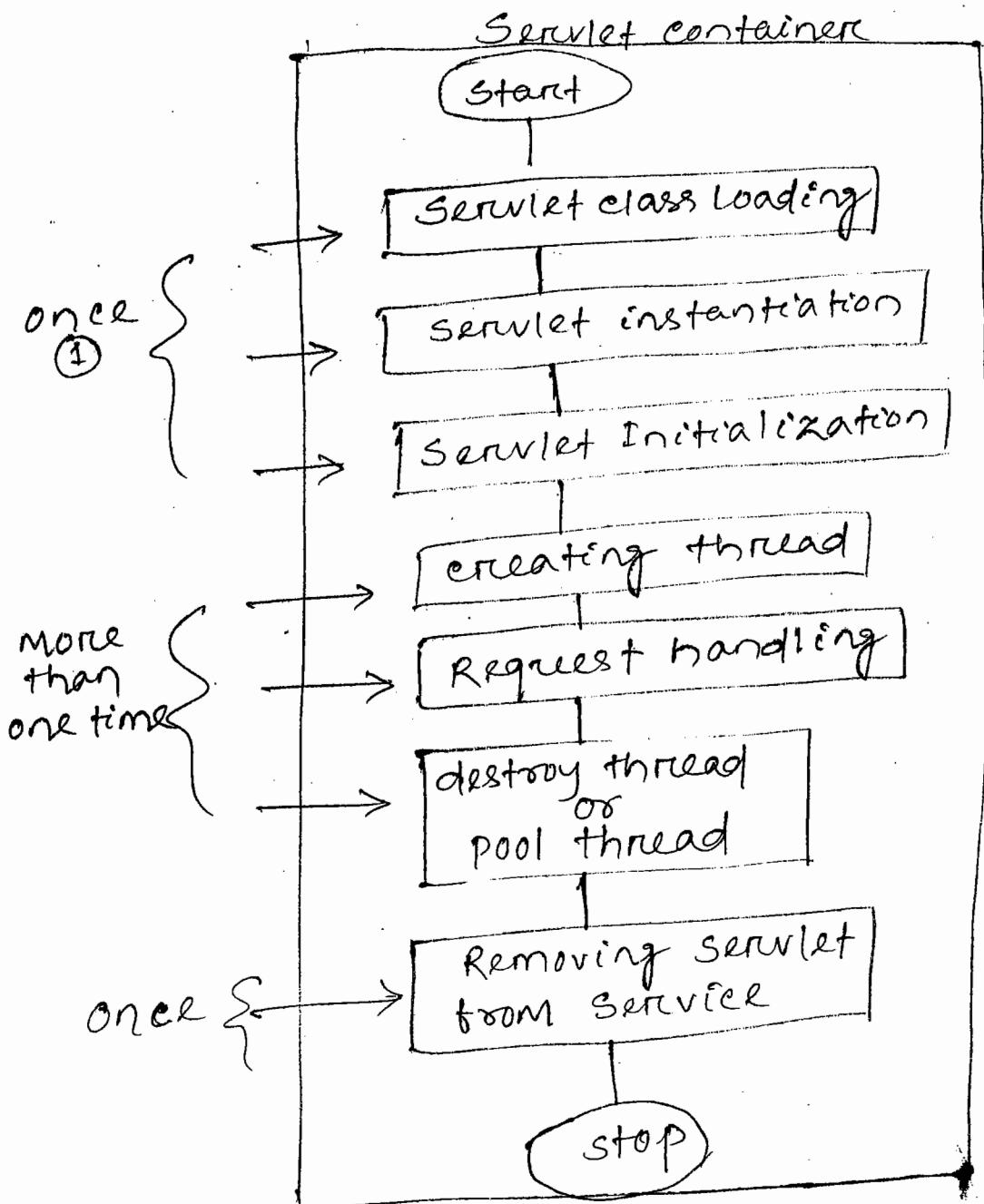
(5) public ServletConfig getServletConfig()

→ It is a non lifecycle Method.

→ Returns a servletconfig object, which contains initialization and startup parameters for this servlet.

Life cycle of servlet

→ servlet is a program executed by servlet container.



(Life cycle of servlet)

- ① servlet API (service provider SUN Microsystem)
- ② servlet developer (service implementer)
- ③ servlet container (service caller)

Servlet class loading

- When container receives request for a Servlet, it first loads the class into memory.
- Servlet class loading is done ~~into~~ only once.
- It is loaded by class loader of Servlet container.

Servlet Instantiating / Initialization

- After Loading Servlet container create an object of loaded Servlet.
- This servlet object ~~to~~ persist in Memory in order to process the request of multiple client.

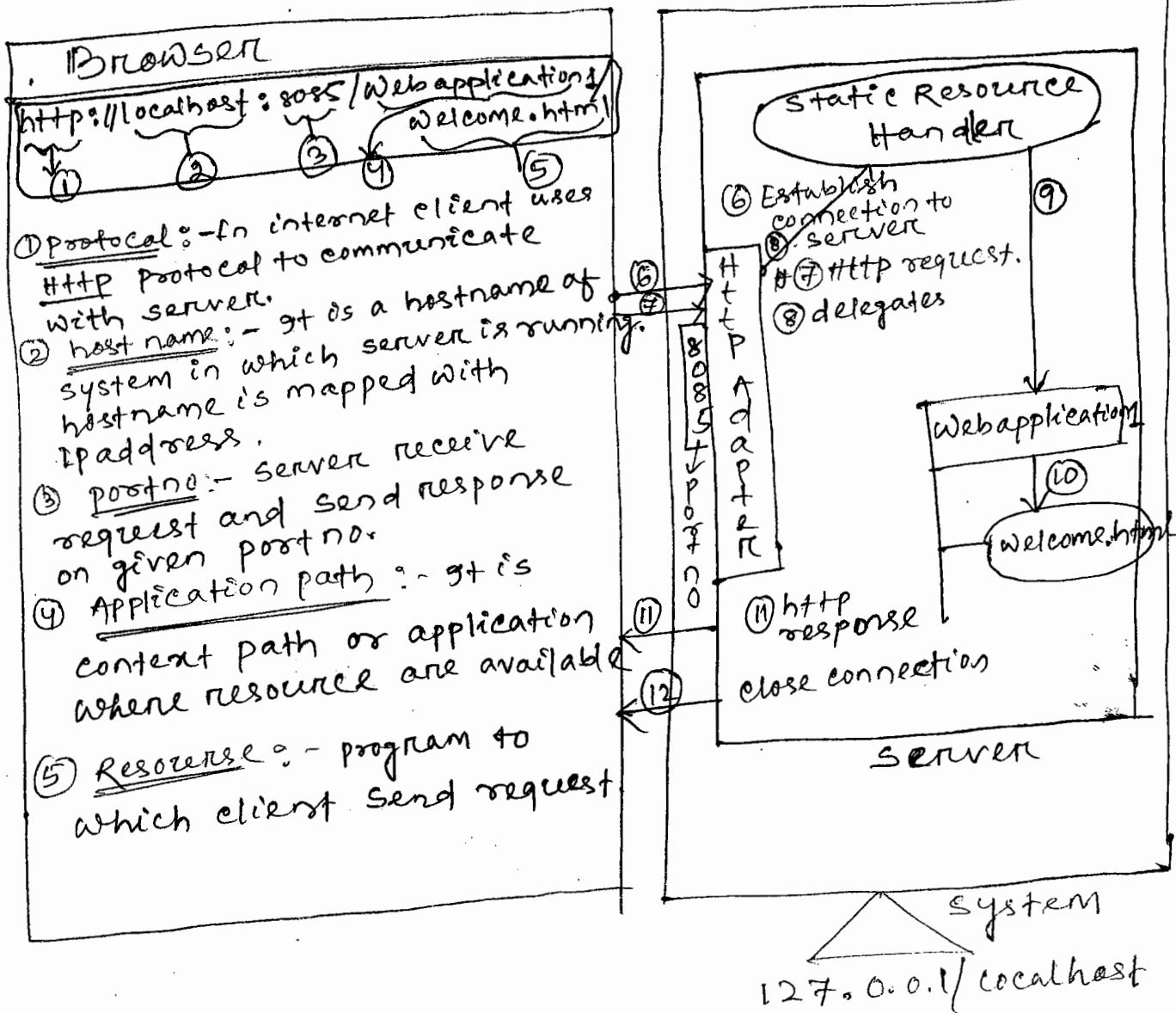
Servlet Initialization

- After instantiation, the container initialize Servlet before placing into service, container initialize Servlet by calling init method.

Request handling (call the service Method)

- After the servlet is initialized, the container may keep it ready for handling client requests. When client request arrives they are delegated to the servlet through the `service()` method, passing the request and response object as parameters.

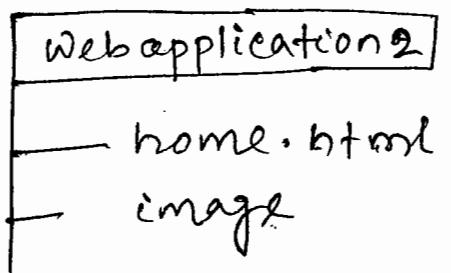
- Q/How to change server port no?
- Go to tomcat 7.0/conf/server.xml
 - open server.xml
 - change connector port no.
 - connector port = "8085"



SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Program 2

program for putting image and show it on browser.



```
<html>
<body bgcolor = cyan>
<img src = "x.jpg" width = 200
      height = 300> </img>
<h2> This is webpage with image </h2>
</body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

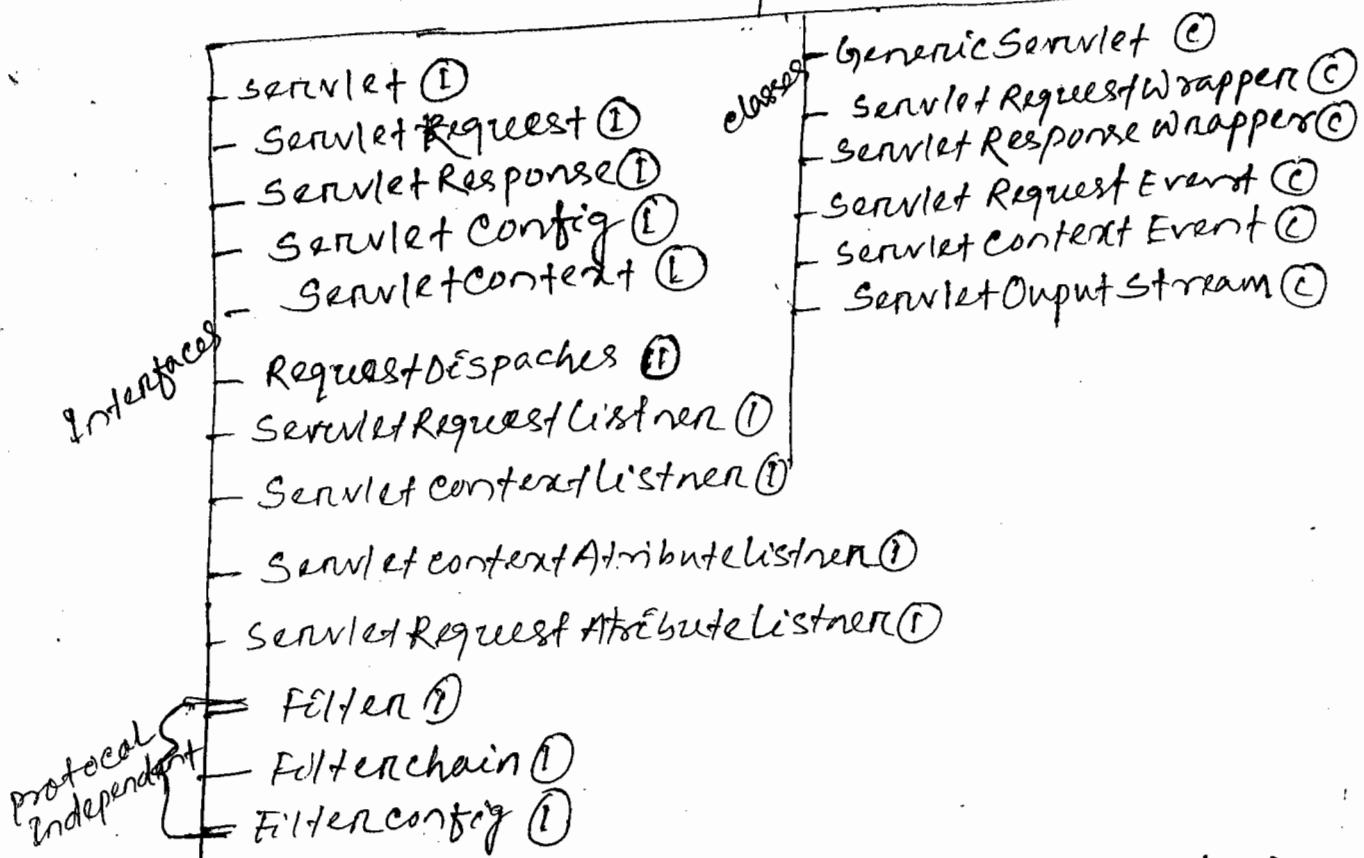
How to develop Servlet :-

→ In order to develop Servlet servlet API provide two packages

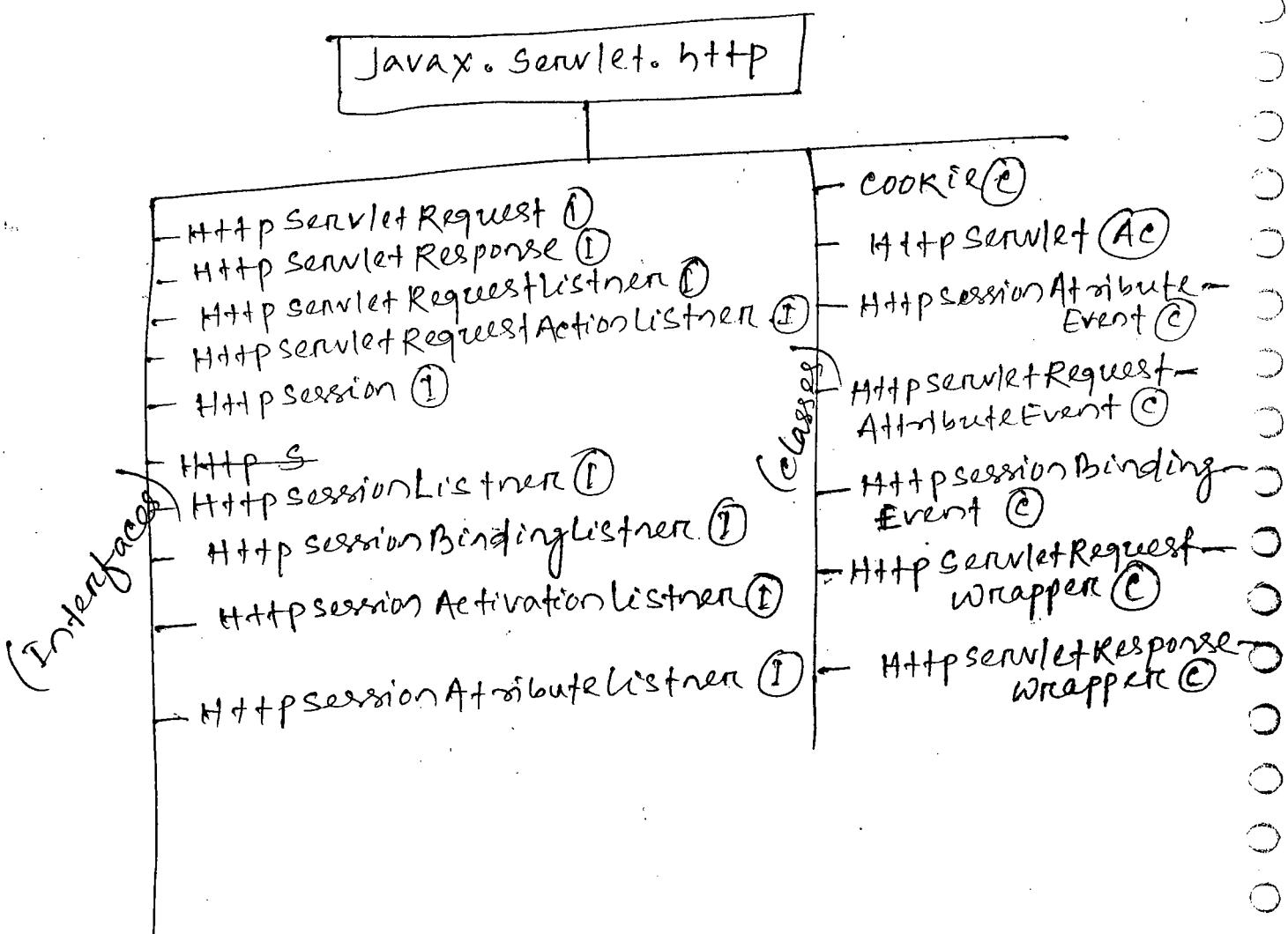
1. javax.servlet package.
2. javax.servlet.http.

→ Javax.servlet package used for developing protocol independent Servlets

Javax.servlet package



→ Javax.servlet.http package used for developing http Servlets or protocol dependent Servlets



Servlet Interface

- It is a root interface for all servlets.
- Every servlet must be inherited from servlet interface.
- Servlet interface provide 5 Methods
 - 3 Methods are called lifecycle methods
 - 2 Methods are called non lifecycle methods.

So what is lifecycle method?

- A Method which is executed by servlet container is called lifecycle method.

- Q. what is non Lifecycle Method?
→ A method not executed by servlet container is called non Lifecycle Method.

Methods of Servlet Interface :-

- (1) **public void init (ServletConfig config)**
throws ServletException.
- It is a lifecycle Method of servlet.
 - called by the Servlet container to indicate to a servlet that the servlet is being placed into service.
 - This method is called servlet container only once.
 - The servlet container calls the init method exactly once after instantiating the servlet. The init method must complete successfully before the servlet can receive any requests.
 - servlet is initialized by calling init method.
- (2) **public void service (ServletRequest req, ServletResponse res)**
throws IOException.

- It is a lifecycle Method.
- This method is called by servlet container to process the request of client.
- called by the servlet container to allow the servlet to respond to a request.
- Operation performed by servlet is defined inside Service Method.

→ this method is called by servlet container by creating thread.

(3) public void destroy()

→ it is a lifecycle method this method is called by servlet container before servlet object is removed.

→ This method is called only once.

(4) public String getServletInfo()

→ Returns information about the servlet such as author, version and copyright.

→ It is a ~~non lifecycle~~ method.

(5) public ServletConfig getServletConfig()

→ It is a non lifecycle Method.

→ Returns a servletconfig object, which contains initialization and startup parameters for this servlet.

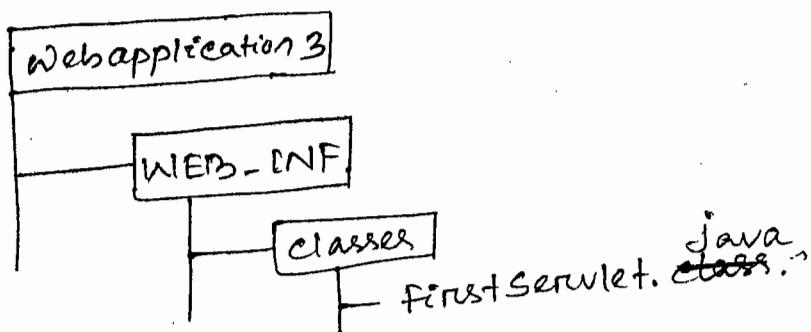
Life cycle of servlet

→ servlet is a program executed by servlet container.

Removal from Service (call the destroy Method) :-

- A servlet container may decide to remove a servlet from service for various reasons such as to conserve memory resources.
- Before removing container calls destroy method.

// ~~This~~ This is first servlet program?



```
package com.nit.servlets;
import javax.servlet.*;
import java.io.*;
public class FirstServlet implements Servlet
{
    static {
        System.out.println("Inside static Block");
    }
    public FirstServlet() {
        System.out.println("Inside constructor");
    }
    public void init(ServletConfig config)
        throws ServletException
    {
        System.out.println("Inside init Method");
    }
}
```

```
public void service ( ServletRequest req,  
                      ServletResponse res)
```

throws ServletException, IOException.

```
{
```

```
    sopln ("Inside service method");
```

```
}
```

```
public void destroy () {
```

```
    sopln ("Inside destroy Method");
```

```
}
```

```
public String getServletInfo () {
```

```
    return "firstServlet";
```

```
}
```

```
public ServletConfig getServletConfig ()
```

```
{
```

```
    ServletConfig config = null;
```

```
    return config;
```

```
}
```

Deployment descriptor file or Web Application

configuration file :-

→ Q. what is Deployment descriptor?

Ans:- Deployment descriptor is a configuration
file for web application and it's name

is web.xml and it reside in WEB-INF
directory

- Servlet container use this file to configure web application servlets, servlet config params, context init params, filters, listeners, welcome page and error handlers.
- one web application contain only one [web.xml].
- It is an XML file and collection of ~~xml~~ XML tags.
- Servlet container parse [Web.xml] during deployment of Webapplication.

Syntax of Writing Web.xml

<Web-app> → Root tag

→ <Servlet>

 |
 container information <Servlet-name> instance name </Servlet-name>
 <Servlet-class> servlet-class-name
 with package name </servlet-class>
 </servlet>

→ <Servlet-mapping>

 |
 client information <Servlet-name> instance name </Servlet-name>
 <url-pattern> / url </url-pattern>
 </servlet-mapping>

</Web-app>

- Servlet is a public to server but private to client. Client can't access Servlet directly.

- Client access servlet using public url defined inside web.xml.

```
< Web-app>
  < servlet>
    < servlet-name> S1 </servlet-name>
      instance name
    < servlet-class> com.onit. servlet * firstservlet
      servlet class name
    < /servlet>
  < /servlet>
< servlet-mapping>
  < servlet-name> S1 </servlet-name>
  < url-pattern> /first </url-pattern>
< /servlet-mapping>
< /Web-app>
  Save it in
  Save it in WEB-INF
```

compile :-

set classpath = c:\Tomcat 8.0\lib\servlet-api.jar

~~javac~~

webapplication3\WEB-INF\classes > javac -d . first
servlet.java

http://localhost

http://localhost:8085/webapplication3/first

Server

Servlet container.

`http://localhost:8080/
Web applications/first
url.`

Establish
connection

`HTTP request
url: Web applications/
first`

- ① Load firstServlet class.
- ② Create object of firstServlet.

`new
first
Servlet
object`

`instance
S1`

delegates

`firstServlet`

`ServletConfig`

`PSV init(ServletConfig
config)`

`↳ sopIn ("inside init")
↳ thread.`

`call Service Method.`

`P. V. service (ServletRequest
req, ServletResponse
res);`

`P. V
destroy()`

`P. V
String getServletInfo()`

`P. V
ServiceContext — {`

`old`

`Web.xml`

`< servlet >
< servlet-name >
< /servlet-name >`

`< servlet-class > /communit.
Servlets /firstServlet
< /servlet-class >`

`< servlet-mapping >
< servlet-name > S1
< /servlet-name >`

`< servlet-mapping >
< web-pattern > /first
< /web-pattern >`

`< servlet-mapping >
< web-pattern >
< /web-pattern >`

Browser

- Servlet is loaded and created object / instantiation
instantiation is done in two ways
 1. When servlet receive first request
 2. At the time of deploying Web application.
 - In order to load Servlet at application startup, that servlet must be configured in web.xml using load-on-startup tag.
- Q. How to make sure a servlet is loaded at the application startup?
- ~~really~~ servlet container loads a servlet on the first client request but sometimes when the servlet is heavy and takes time to loads, we might want to load it on application startup.
 - We can use Load-on-startup element with servlet configuration in web.xml file.

Syntax

```

<Web-app>
  <Servlet>
    <servlet-name> <instance-name> </servlet-name>
    <servlet-class> < servlet-class name > </servlet-class>
    <load-on-startup> <value> </load-on-startup>
    </servlet>
  ...
</Web-app>
  
```

int value
~~or > 0~~

→ The load-on-startup value should be int, if it's 0 or negative integer then servlet container will load the servlet based on client requests and requirement but if it's positive, then container will load it on app's startup.

→ If there are multiple servlets with load-on-startup value as 1, 2, 3 then lower integer value servlet will be loaded first.

these are loaded at the time of deploying webapplication

these are loaded when it receives first request

Servlets	Load-on-startup
Servlet1	10
Servlet2	5
Servlet3	15
Servlet4	0
Servlet5	-1

Servlet container

XML parses

- ② Loaded
- ① Loaded
- ③ Loaded

→ Web.xml is executed at the time of deploying webapplication

Servlet Response

→ ServletResponse is an interface

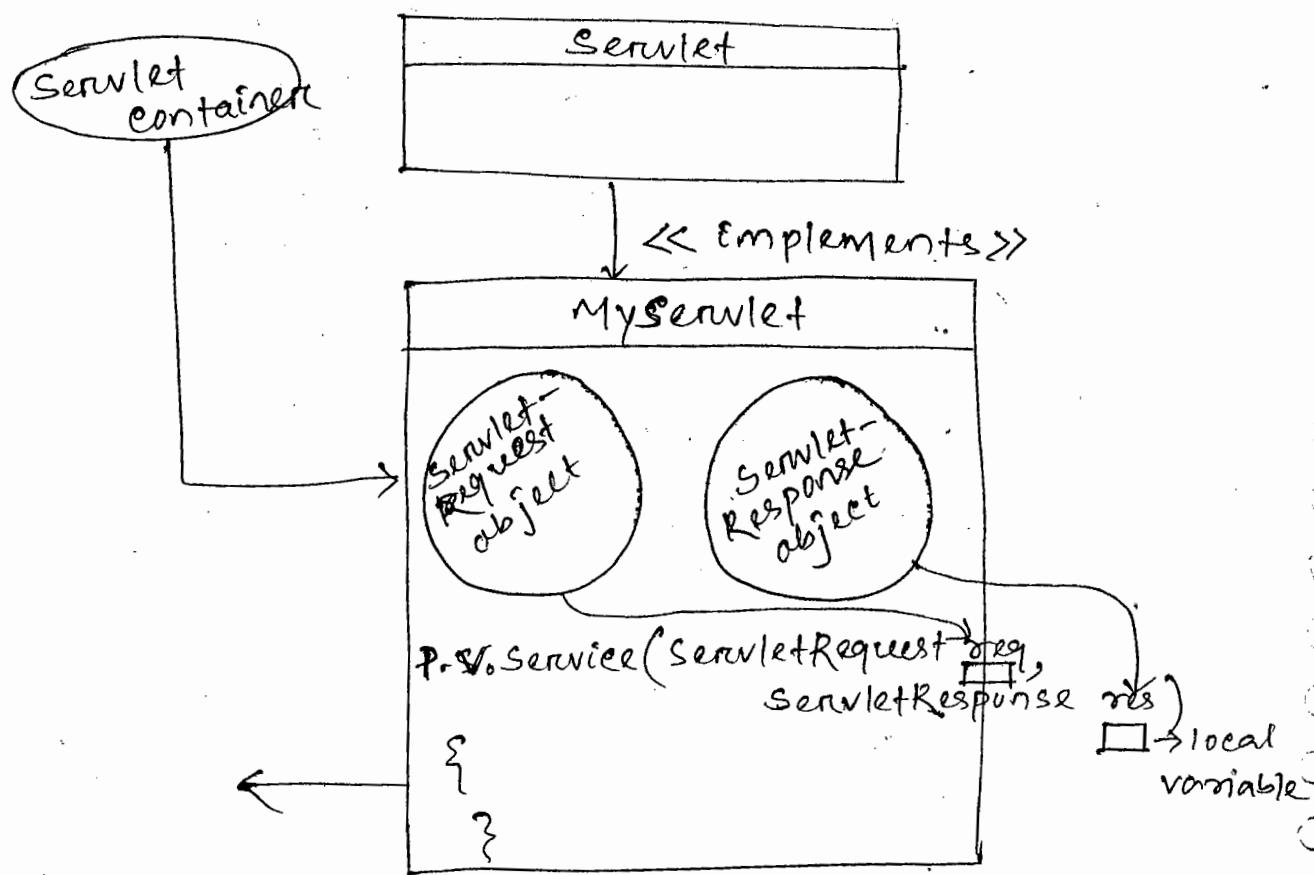
→ This interface is implemented by servlet

container

→ An object of ServletResponse is created by servlet container and send to service method.

→ Lifetime of ServletResponse object is until execution of service method.

→ Servlet uses this object for generating output or response.



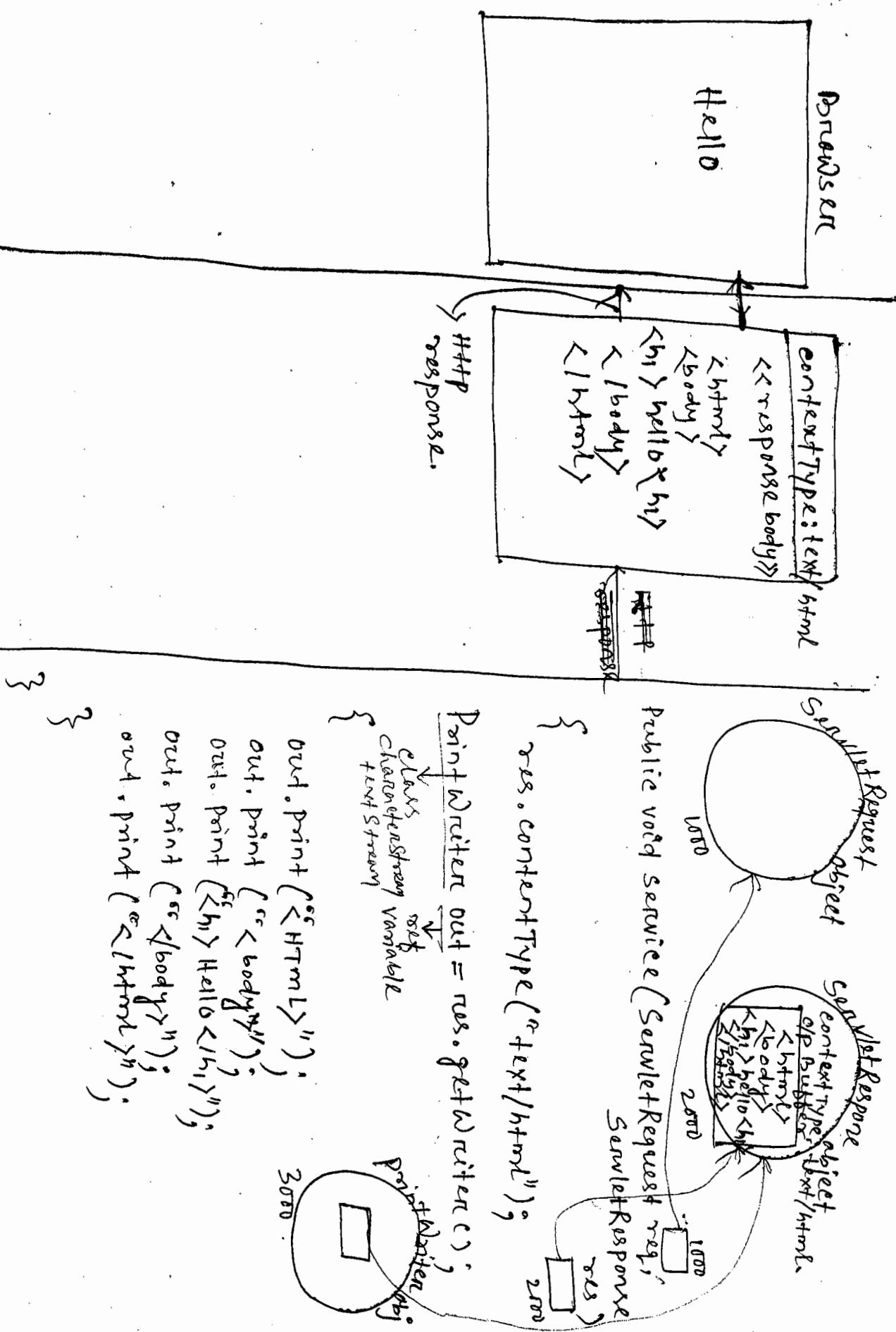
Methods of ServletResponse :-

void setContentType(String type)

- sets the content type of the response being sent to the client, if the response has not been committed yet.
- servlet responses provide 2 types of streams for generating output.
 - 1. byte stream.
 - 2. character stream.

character Stream:-

- servlet response provide the following method which return character Stream.
- **PrintWriter getWriter()**
 - Returns a PrintWriter object that can send character text to the client.



- ServletResponse provide the following method which return byte stream.

ServletOutputStream getOutputStream()

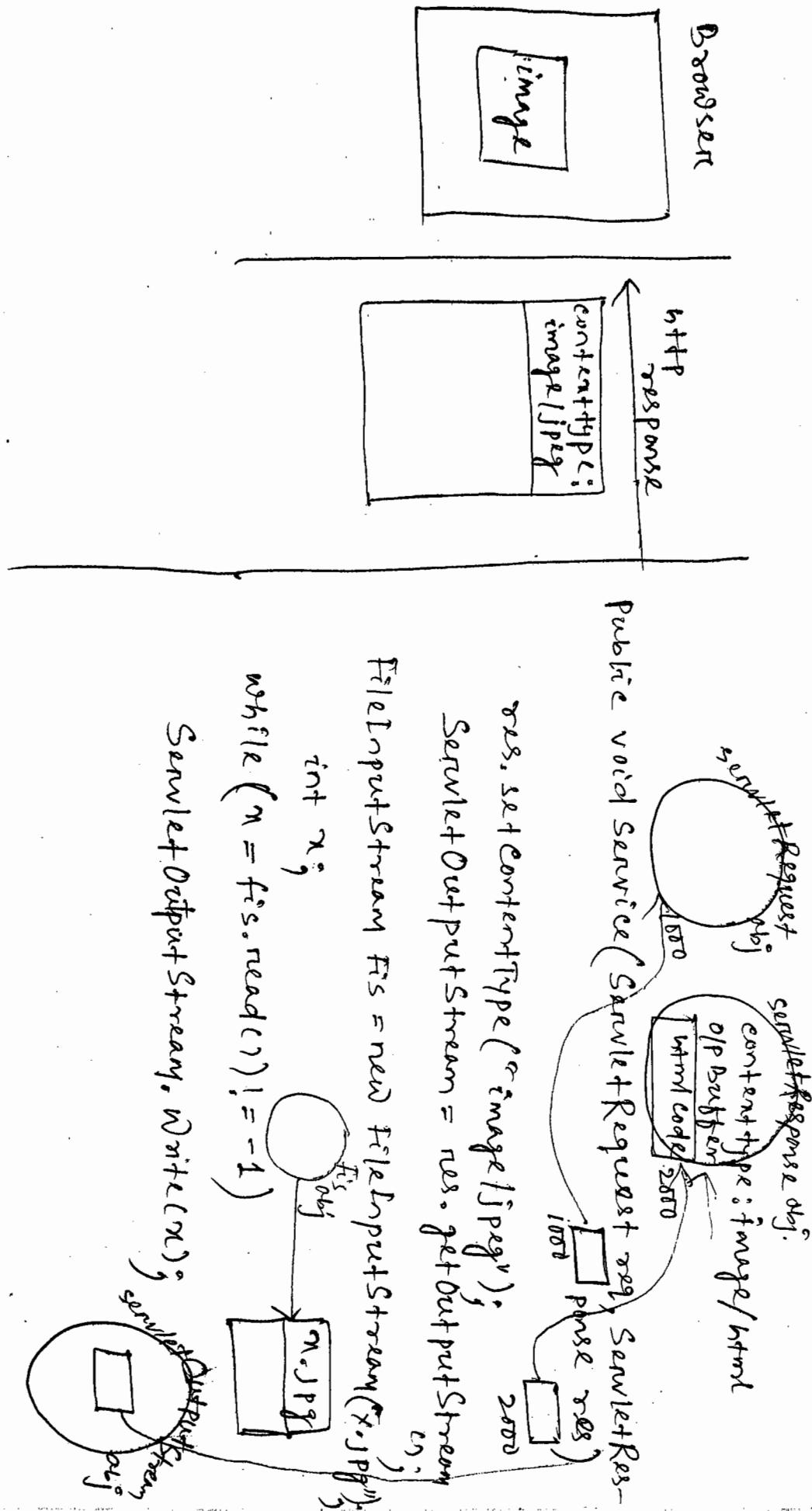
- Returns a ServletOutputStream creatable for writing binary data in the response

- Q. Can we get PrintWriter and ServletOutputStream both in servlet?

- We can't get instances of both PrintWriter and ServletOutputStream in a single servlet Method, if we invoke both the methods; getWriter() and getOutputStream() on response, we will get java.lang.IllegalStateException at runtime with message as other method has already been called for this response.

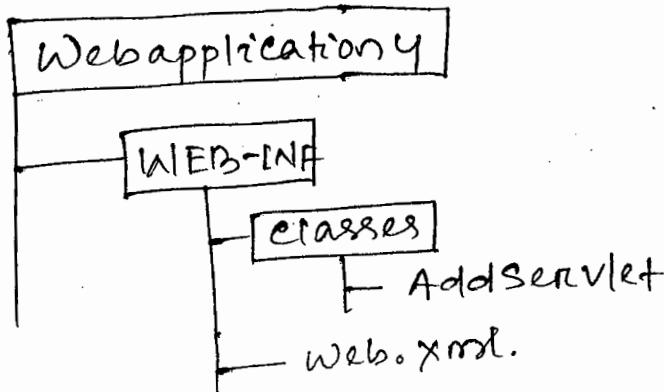
- Q. what is preinitialization of servlet?

- A container does not initialize the servlets as soon as its starts up, it initializing a servlet when it receives a request for that servlet first time. This is called lazy loading. The servlet specification defines the element, which can be specified in the deployment descriptor to make the servlet container load and initialize the servlet as soon as it starts up. The process of loading a servlet before any request comes is called preloading or preinitializing a servlet.



KRISHNAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Example/ program :-



```
package com.nit.servlets;
import javax.servlet.*;
import java.io.*;
public class AddServlet implements Servlet
{
    static // It is executed loading of the class.
    {
        System.out.println("Inside static block");
    }
    public void init(ServletConfig config)
        throws ServletException
    {
        System.out.println("Inside init method");
    }
    public void service(ServletRequest req,
                        ServletResponse res)
        throws ServletException, IOException
    {
        int num1 = 10;
        int num2 = 20;
        int num3 = num1 + num2;
    }
}
```

```
res.setContentType("text/html");
PrintWriter out = res.getWriter();
out.println("<html>");
out.println("<body bgcolor=cyan>");
out.println("<h2>");
out.println("sum is"+num3);
out.println("</h2>");
out.println("</body>");
out.println("</html>");

}

public void destroy() { }

public String getServletInfo() {
    return "AddServlet";
}

}

public ServletConfig getServletConfig()
{
    ServletConfig config = null;
    return config;
}

}
```

SR/ RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Web.xml

```
<web-app>
< servlet >
< servlet-name > S1 < /servlet-name >
< servlet-class > com.nit.servlets.AddServlet
< /servlet-class >
< load-on-startup > 1 < /load-on-startup >
< /servlet >
< servlet-mapping >
< servlet-name > S1 < /servlet-name >
< url-pattern > /add < /url-pattern >
< /servlet-mapping >
< /web-app >
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

10.11.15

ServletRequest :-

- ServletRequest is an Interface. This interface is implemented by Servlet container.
- An object of ServletRequest is created by Servlet container and send to service Method.
- This object contain information about client and information send by client to servlet.

what is parameters or request parameters?

→ parameter is a value send by client to servlet (or) server.

→ Each parameter is having two properties

1. parameter name.
2. parameter value.

→ default request method is GET.

→ In this method data or parameters are concat with URL and send to server.

→ the length of URL is rest to 1024 characters.
This method can't used for sending large amount data.

Methods of ServletRequest :-

① String getParameter (String name)

→ Returns the value of a request parameters as a string, or null if the parameter doesn't exist.

② Map getParameterMap()

→ Returns a java.util.Map of the parameters of the request.

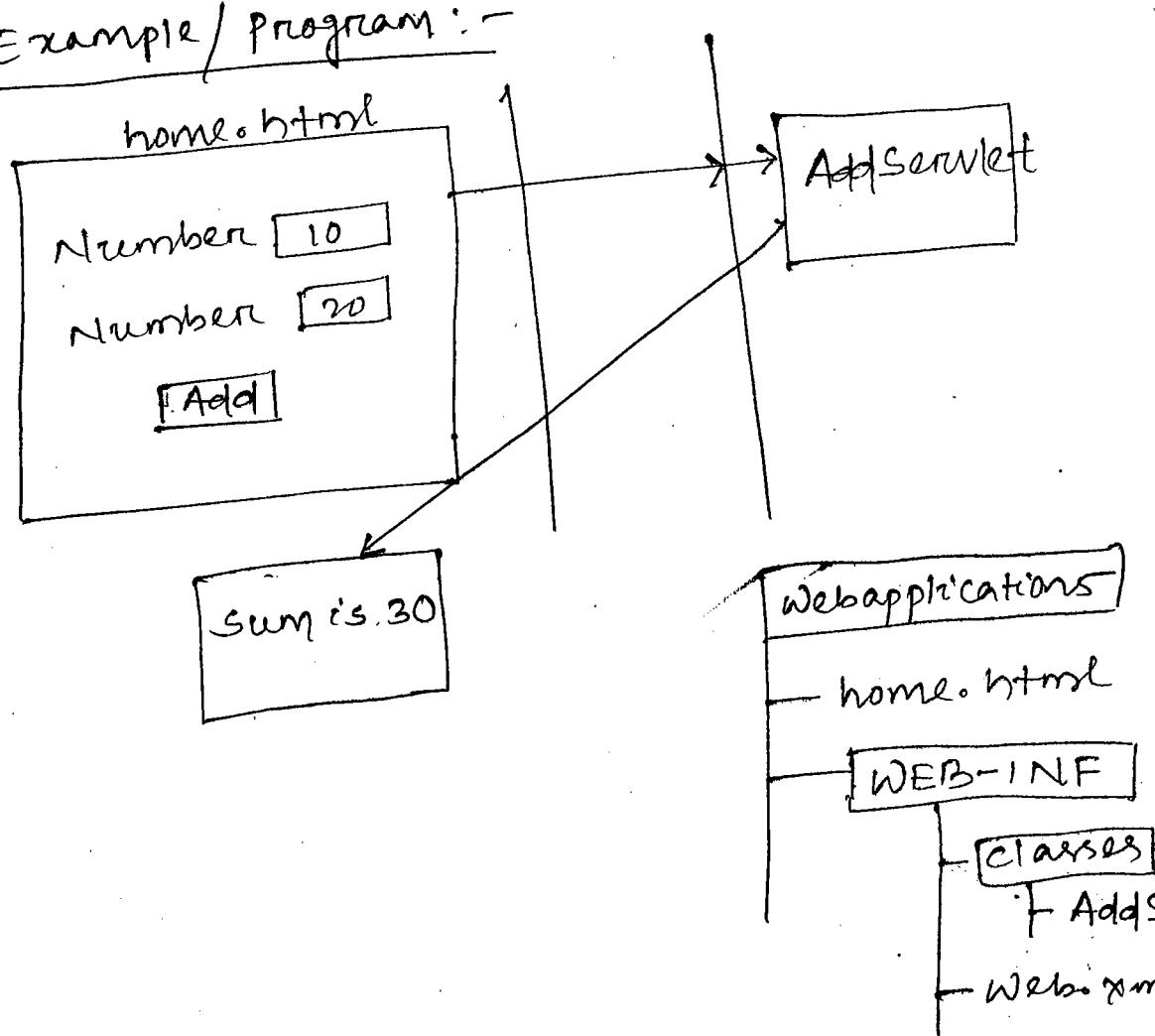
③ Enumeration getParameterNames()

→ Returns an Enumeration of String objects containing the name of the parameters contained in this request.

④ [String[]] getParameterValues(String name)

→ Returns an array of String objects containing all of the values the given request parameter has, or null if the parameter doesn't exist.

Example/ program:-



```
<html>
<body bgcolor = cyan>
<form action = ". / add">
<font color = blue size = 6>
Number <input type = text name = "n1"> <br>
Number <input type = text name = "n2"> <br>
<input type = submit value = "Add">
</font> </form>
</body> </html>
```

```
package com.nit.servlets;
import javax.servlet.*;
import java.io.*;
public class AddServlet implements Servlet
{
    public void init (ServletConfig config) throws
        ServletException.
    {
    }
    public void service (ServletRequest req, Servlet-
        Response res)
        throws ServletException, IOException.
    {
        int num1 = Integer.parseInt (req.getParameter
            ("n1"));
        int num2 = Integer.parseInt (req.getParameter
            ("n2"));
        int num3 = num1 + num2;
        out.println
        PrintWriter out = res.getWriter();
```

```

out.println("<html>");
out.println("<body bgcolor=cyan>");
out.println("<h2>");
out.println("sum is:" + num3);
out.println("</h2></body></html>");
}

public void destroy() {}

public String getServletInfo() {
    return "AddServlet";
}

public ServletConfig getServletConfig()
{
    ServletConfig config = null;
    return config;
}

```

web.xml

```

<web-app>
<servlet>
<servlet-name> s1 </servlet-name>
<servlet-class> com.nit.servlets.AddServlet
</servlet-class>

</servlet>
<servlet-Mapping>
<servlet-name> s1 </servlet-name>
<url-pattern> /add </url-pattern>
</servlet-Mapping>
</web-app>

```


SRI CHINNA AVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

GenericServlet :-

- GenericServlet is an abstract class which implements Servlet, ServletConfig interfaces.
- Define a generic, protocol independent servlet.
To write an HTTP servlet for use on the Web, extend HttpServlet instead.
- GenericServlet implements the Servlet and ServletConfig interfaces.
- GenericServlet makes writing servlets easier.
It provides simple version of lifecycle method init and destroy and of the methods in the ServletConfig interface.
- GenericServlet also implements the log method, declared in the ServletContext interface.
- To write a generic servlet, you need only override the abstract service method.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Servlet

- ① p. v. init (ServletConfig config) throws
 ServletException
- ② p. v. service (Servlet Request req, Servlet-
 Response res) throws
 ServletException, IOException.
- ③ public void destroy();
- ④ public getServletInfo();
- ⑤ public ServletConfig getServletConfig();

⟨ implements ⟩

GenericServlet

```
p. v. init (ServletConfig config){  
}  
p. v. destroy () {  
}  
p. string getServletInfo () {  
}  
p. ServletConfig getServletConfig () {  
}
```

⟨ extends ⟩

MyServlet

```
p. v. service (ServletRequest req,  
              ServletResponse res);
```

{

}

Must
override
Service
Method

Q. How many ways to develop a servlet?

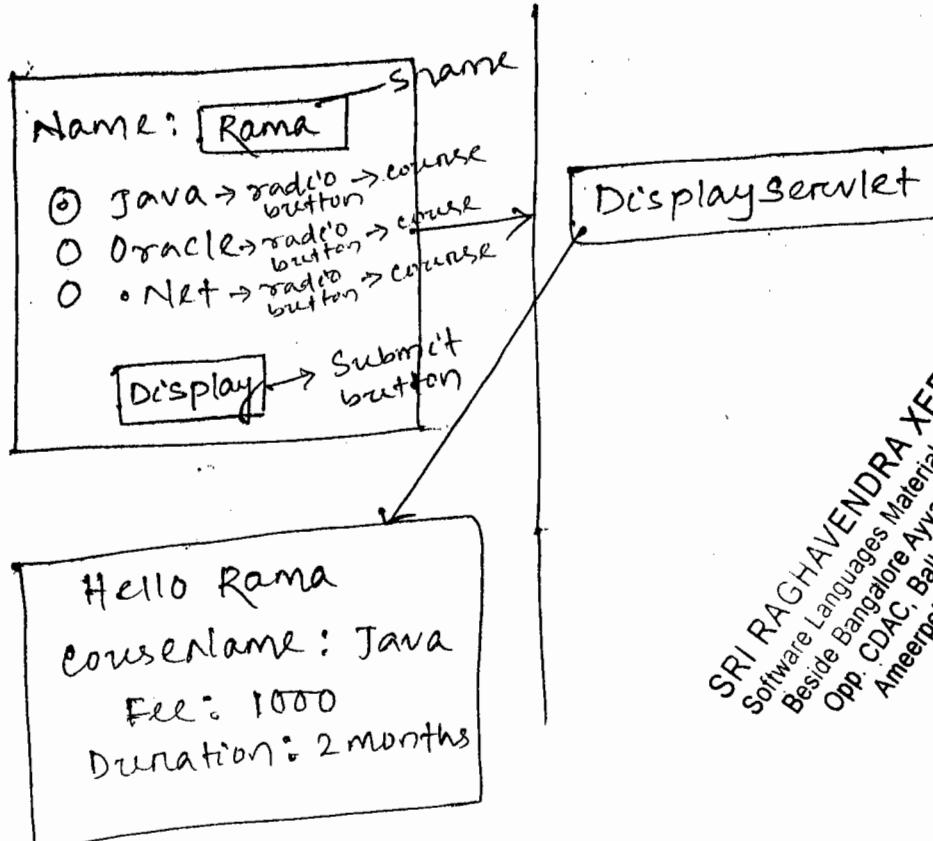
Protocol Independent (i) Servlet → (I)

I - Interface

AC - Abstract Class

Protocol dependent (II) GenericServlet - AC

(III) HttpServlet AC



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Q. How to define welcome files for Web application?

→ The welcome file is a default file to be loaded by a servlet container when we access a URL without telling which page to load.

→ This has to be configured in web.xml.

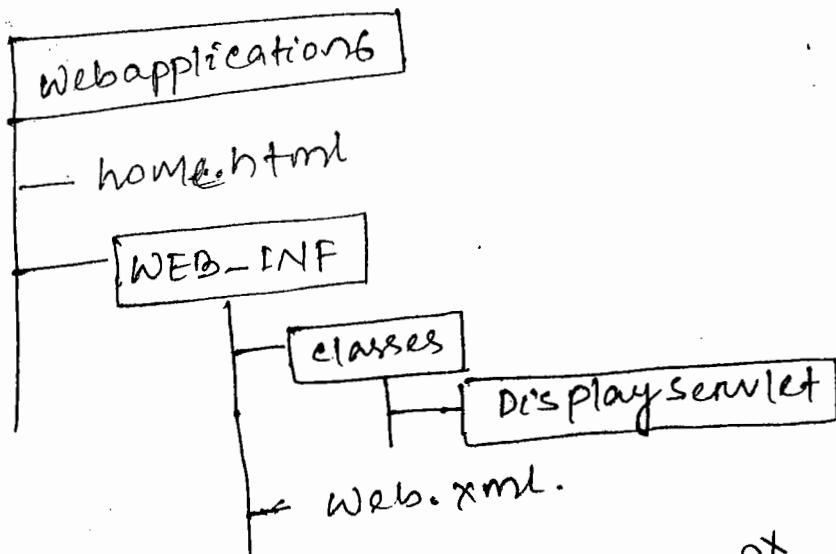
```
<web-app>
  <welcome-file-list>
    <welcome-file> file-name</welcome-file>
    <welcome-file> file-name</welcome-file>
    ... <welcome-file> file-name</welcome-file>
```

```
</welcome-file-list>  
</web-app>
```

Example :-

```
<web-app>  
  <welcome-file-list>  
    <welcome-file> home.html </welcome-file>  
    <welcome-file> index.html </welcome-file>  
    <welcome-file> index.jsp </welcome-file>  
  </welcome-file-list>  
</web-app>
```

Directory structure



DR. K. AVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Program

First develop html

```
<html>
<body bgcolor="cyan">
<form action = "./display">
<font color = blue size = 6>
Name <input type = text name = "sname"><br>
<input type = radio name = "course"
value = "java"> Java <br>
<input type = radio name = "course" value = "dotnet">
value = ".Net" > .Net <br>
<input type = submit value = "display">
</font> </form> </body>
</html>
```

Developing Servlet

```
package com.nit.servlets;
import javax.servlet.*;
import java.io.*;
public class DisplayServlet extends GenericServlet
{
    public void service(ServletRequest req,
                        ServletResponse res)
        throws ServletException, IOException
    {
        String name = req.getParameter("sname");
        String c = req.getParameter("course");
        res.setContentType("text/html");
    }
}
```

```

PrintWriter out = reso.getWriter();
out.println("<html>");
out.println("<body bgcolor=yellow>");
out.println("<h2>");
out.println("Hello "+name+"<br>");
if (e.equals("java"))
    out.println("Java Fee : 2000");
if (e.equals("oracle"))
    out.println("Oracle Fee : 500");
if (e.equals(".net"))
    out.println(".Net Fee : free");
out.println("</h2></body></html>");
}
}

```

Save this the name `DisplayServlet.java`

Web.xml :-

```

<web-app>
<servlet>
<servlet-name> S1 </servlet-name>
<servlet-class> com.nit.servlets.Display-
                           Servlet <
                           servlet-class>
</servlet>
<servlet-mapping>
<servlet-name> S1 </servlet-name>
<url-pattern> & /display </url-pattern>
</servlet-mapping>

```

```

<welcome-file-list>
  <welcome-file> home.html </welcome-file>
</welcome-file-list>
</web-app>

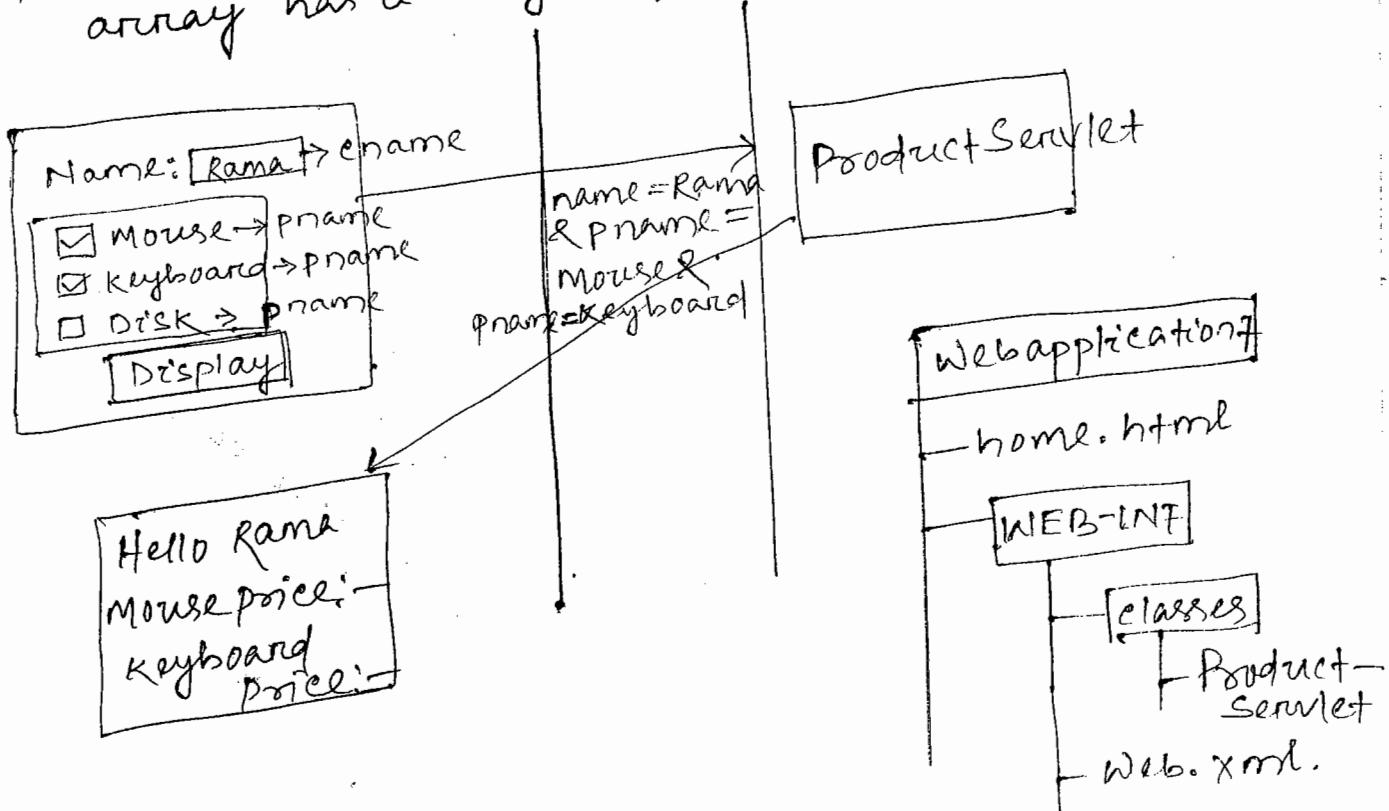
```

13.11.15

getParameterValues :-

`String[] getParameterValues (String name)`

- Returns an array of String objects containing all of the values the given request parameter has, or null if the parameter does not exist.
- If the parameter has a single value the array has a length of 1.



Program :-

```
<html>
<body bgcolor="cyan">
<form action = "product">
<h2>
CustomerName < input type="text" name="cname" >
<br>
<input type="checkbox" name="pname"
      value="mouse" > mouse <br>
<input type="checkbox" name="pname"
      value="Keyboard" > Keyboard <br>
<input type="checkbox" name="pname"
      value="disk" > Disk <br>
<input type="submit" value="display" >
</h2>
</form>
</body> </html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Save it home.html

Developing Servlet

```
package com.nit.servlets;
import javax.servlet.*;
import java.io.*;

public class ProductServlet extends GenericServlet
{
    public void service (ServletRequest req,
                         ServletResponse res)
        throws ServletException, IOException
    {
        String cn = req.getParameter("cname");
        String pn[] = req.getParameterValues("pname");
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
        out.println("<html>");
        out.println ("<body bgcolor=yellow>");
        out.println ("<h2>");
        out.println ("Hello"+cn+"<br>");
        for (int i=0; i<pn.length; i++)
        {
            if (pn[i].equals("mouse"))
                out.println ("Mouse price: 100<br>");
            if (pn[i].equals("keyboard"))
                out.println ("Keyboard price: 2000<br>");
            if (pn[i].equals("disk"))
                out.println ("Disk price: 4000<br>");
        }
    }
}
```

~~37~~
 out.println("<body></html>");
 } // close service method
 } // close class.

Web.xml

```

<web-app>
  <servlet>
    <servlet-name> s1 </servlet-name>
    <servlet-class> com.nit.servlets.ProductServlet
      <servlet-class>

    </servlet>
    <servlet-mapping>
      <servlet-name> s1 </servlet-name>
      <url-pattern> /product </url-pattern>
    </servlet-mapping>
  </web-app>
  
```

(Q) what is the difference b/w getParameter
 & getParameterValues

getParameter	getParameterValues
→ It returns parameter having one value.	→ It returns parameter having more than one value.
→ Its return type is String	→ Its return type is String[]

Execution flow of previous program

SRI RAGHAVENDRA XEROX

Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SRI RAGHAVENDRA XEROX

Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Q. How do I check if parameter exists in ~~ServletRequest~~ → request?

→ ServletRequest object has a map object that maps parameter name and its value. By accessing this map we can check if a parameter was passed in servlet request.

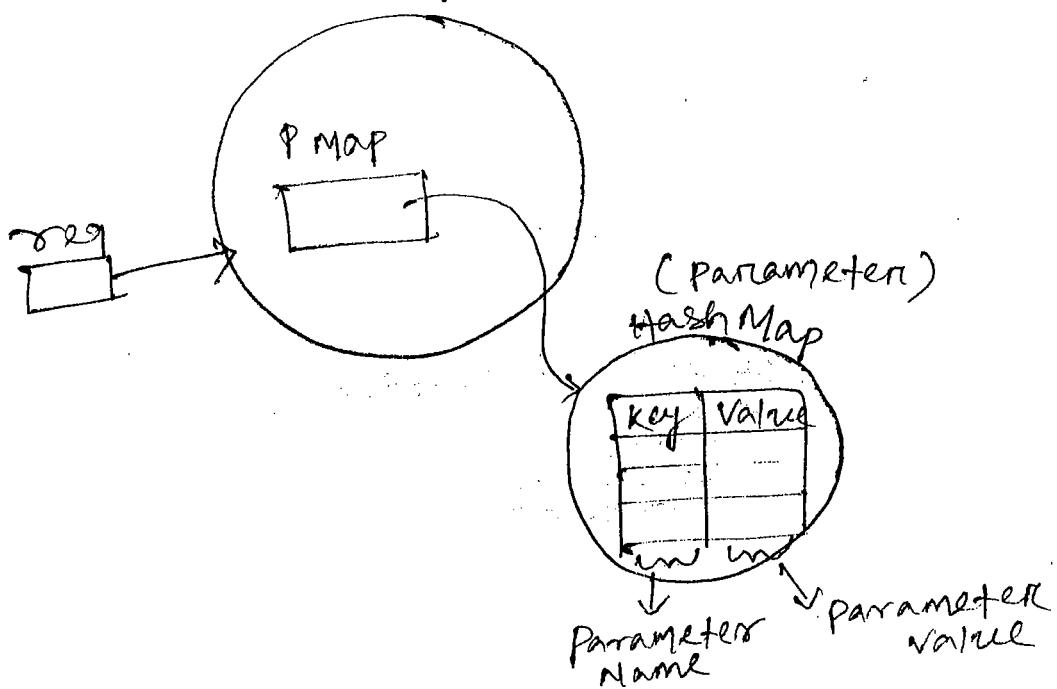
Map getParameterMap() //Method

→ Returns a java.util.Map of the parameters of this request.

→ Map provides following method to search key.

boolean containsKey (Object key) //Method

→ Returns true if this map contains a mapping for the specified key.



// Example for ParameterMap.

```
package com.nit.servlets;
```

```
import java.util.*;
```

```
import javax.servlet.*;
```

```
import java.io.*;
```

```
public class ProductServlet extends GenericServlet
```

```
{
```

```
String cname  
public void service ( ServletRequest req, ServletResponse res )  
throws ServletException, IOException
```

```
{
```

```
String cn = req.getParameter ("cname");
```

```
Map m = req.getParameterMap();
```

```
boolean b = m.containsKey ("pname");
```

```
res.setContentType ("text/html");
```

```
PrintWriter out = res.getWriter();
```

```
out.println ("");
```

```
out.println ("");
```

```
out.println ("

## ");


```

```
out.println ("Hello " + cn + "<br>");
```

```
if ( b == true )
```

```
{  
    String pn[] = req.getParameterValues  
    ("pname");
```

```
for ( int i=0; i<pn.length; i++ )
```

```
{  
    if ( pn[i].equals ("mouse") )
```

```
        out.println ("Mouse Price: 100<br>");
```

```

if (pn[i].equals ("Keyboard"))
out.println ("Keyboard price: 2000<br>");
if (pn[i].equals ("disk"))
out.println ("Disk price: 5000<br>");
```

```
else
    out.println("<h2> Please select product </h2>");
    out.println("</body></html>");
```

3

3

getParameterNames :-

Enumeration getParameterNames()

Enumeration getParameterNames()
Returns an Enumeration of String objects containing the names of the parameters contained in this request.

→ Enumeration :-

→ Environmental life is an interface.

- Enumeration is an Interface.
- Enumeration is an Enumeration Interface
- An object that implements Enumeration Interface generates a series of elements, one at a time successive calls to the nextElement method return successive elements of the series.

Enumeration provides 2 methods.

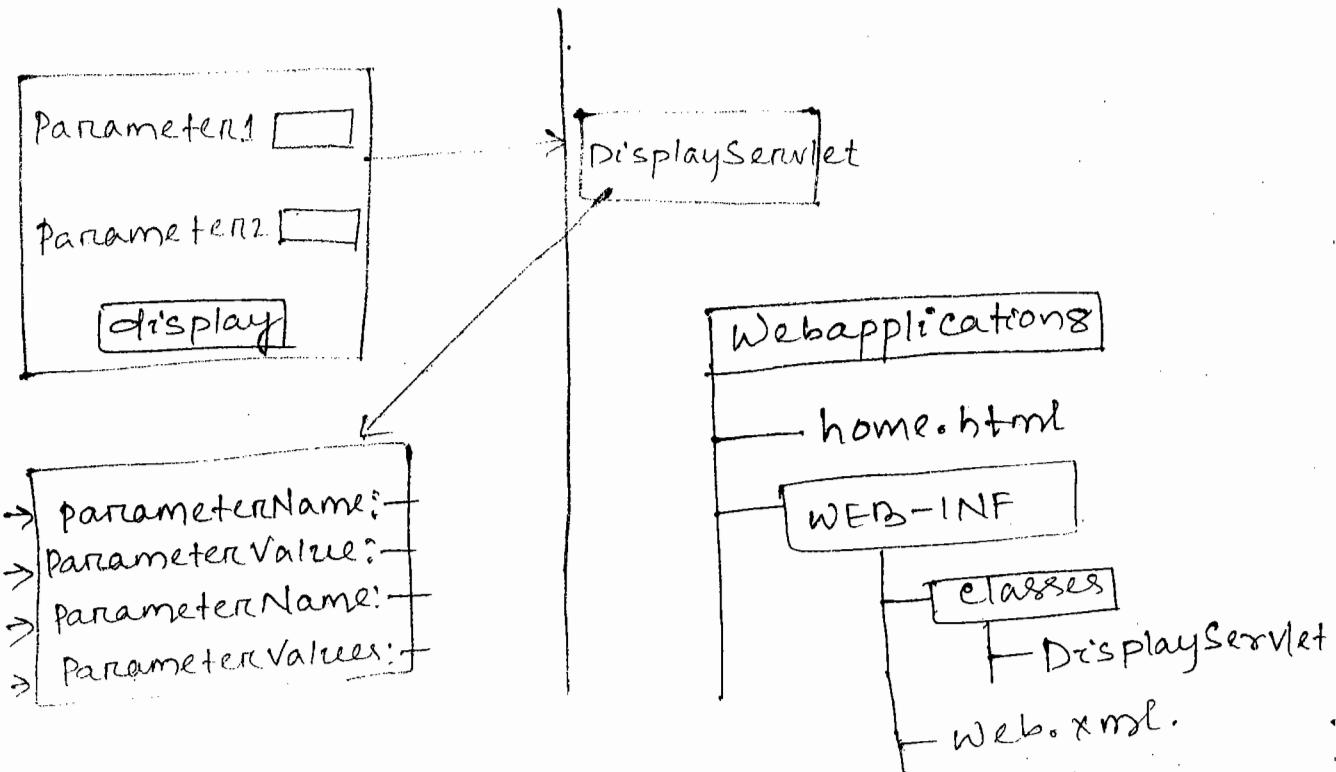
boolean hasMoreElements())

1. `boolean hasMoreElements()`
Tests if the enumeration contains more elements

2. Object nextElement()

Returns the next element of this ensemble.

creation if this enumeration object has at least one more element to provide.



home.html

```
<html>
<body bgcolor=cyan>
<form action = "/display">
<h2>
parameter1 <input type = text name = "P1"><br>
parameter2 <input type = text name = "P2"><br>
<input type = submit value = "display">
</h2>
</form>
</body>
</html>
```

Develop Servlet

```
package com.nit.servlets;
import javax.servlet.*;
import java.io.*;
import java.util.*;

public class DisplayServlet extends GenericServlet
{
    public void service (ServletRequest req,
                         ServletResponse res)
        throws ServletException, IOException
    {
        Enumeration<String>
        e = req.getParameterNames();
        res.setContentType ("text/html");
        PrintWriter out = res.getWriter();
        while (e.hasMoreElements())
        {
            out.println ("

## " ); String pn = e.nextElement(); out.println ("Parameter Name " + pn); out.println (" Parameter Value " + req.getParameter(pn)); out.println ("</h2>" ); } } }


```

web.xml

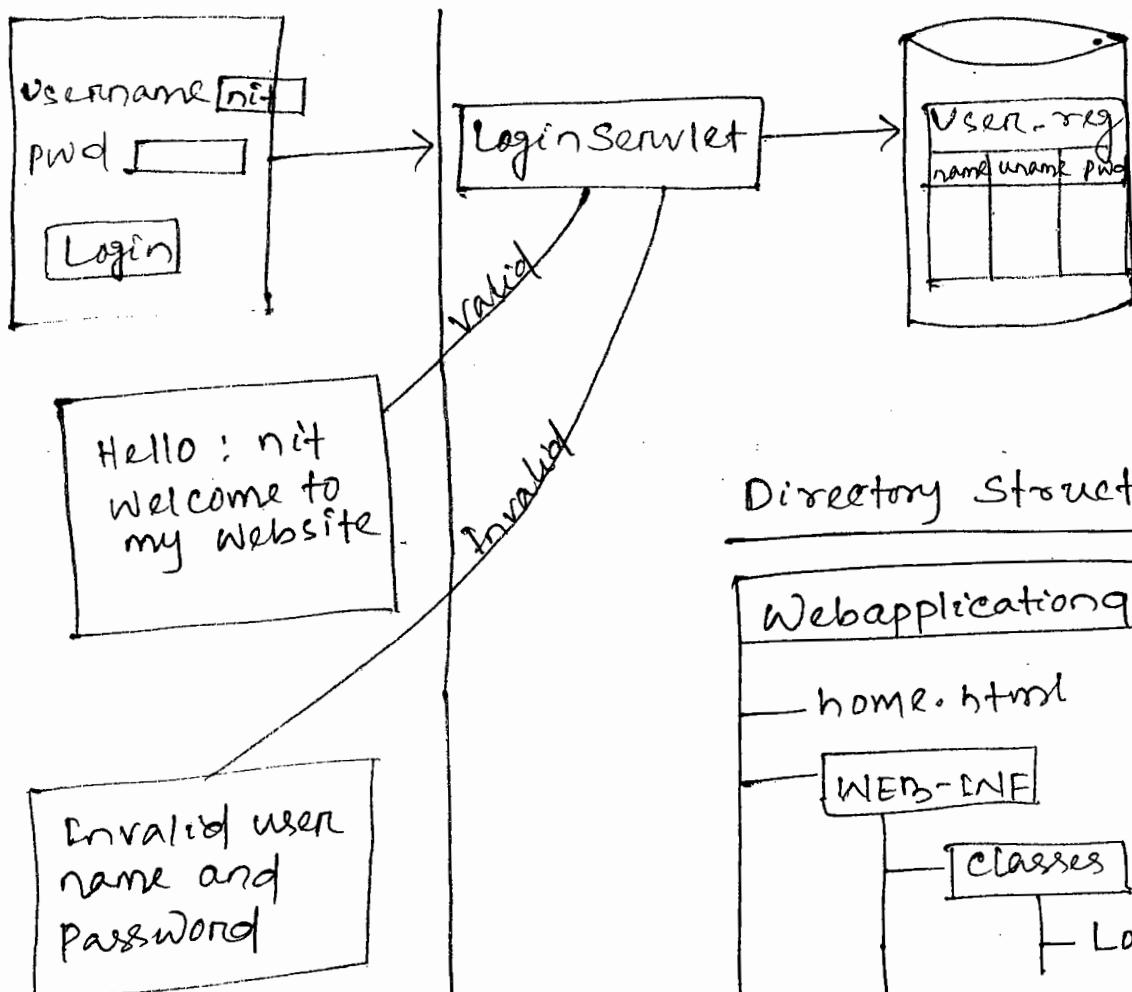
```
<web-app>
  <servlet>
    <servlet-name>S1</servlet-name>
    <servlet-class>com.nit.servlets.DisplayServlet</servlet-class>
  </servlet>
  <servlet-mapping>
    <servlet-name>S1</servlet-name>
    <url-pattern>/display</url-pattern>
  </servlet-mapping>
</web-app>
```

http://localhost:8086/Webapplication8/home.html

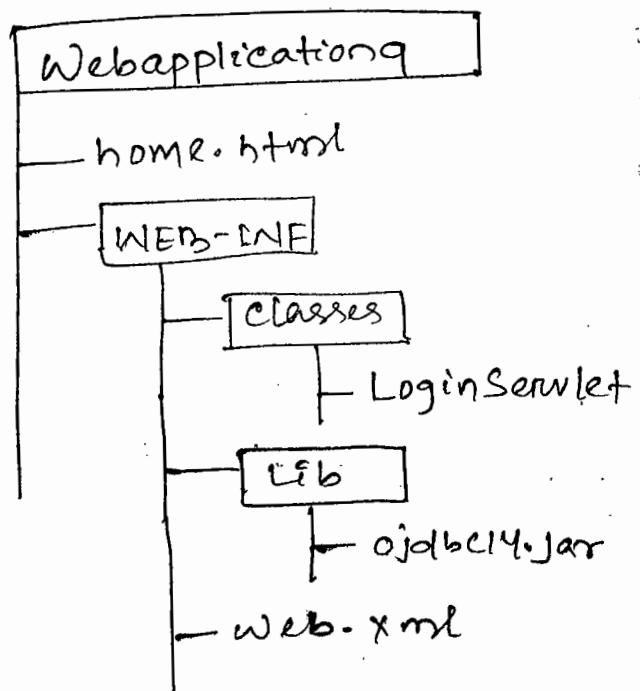
Request Headers :-

- request headers contain information about client.
- GenericServlet doesn't provide any method to manipulate request headers.
- These headers are specific to HTTP protocol.
- In order to read request headers we need to develop HttpServlet.

Servlet communicate with Database



Directory Structure



1. home.html

```
<html>
<body bgcolor="yellow">
<form action="o/LogIn">
<h2>
UserName <input type="text" name="un"><br>
password <input type="password" name="pw"><br>
<input type="submit" value="login">
</h2>
</form>
</body>
</html>
```

2. Develop Servlet

```
package com.nit.servlets;
import javax.servlet.*;
import java.io.*;
import java.sql.*;
public class LogInServlet extends GenericServlet
{
    public void service(ServletRequest req, ServletResponse res)
        throws ServletException, IOException
    {
        String user = req.getParameter("un");
        String pwd = req.getParameter("pw");
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
    }
}
```

```
try {
    Class.forName("oracle.jdbc.driver.OracleDriver");
    Connection cn = DriverManager.getConnection
    (_____, ___, ___);
    PreparedStatement ps = cn.prepareStatement
    ("select count(*) from user-reg where
     uname=? and pwd=?");
    ps.setString(1, user);
    ps.setString(2, pwd);
    ResultSet rs = ps.executeQuery();
    rs.next();
    int c = rs.getInt(1);
    if (c == 1) {
        out.println("<h2>");
        out.println("Hello " + user);
        out.println("<br> welcome to my
                    website </h2>");
    } else
        out.println("<h2> invalid username or
                    password </h2>");
    }
    catch (Exception k) {
        k.printStackTrace();
    }
}
```

web.xml

```
<web-app>
  <servlet>
    <servlet-name> s1 </servlet-name>
    <servlet-class> com.nit.servlets.LoginServlet
      </servlet-class>

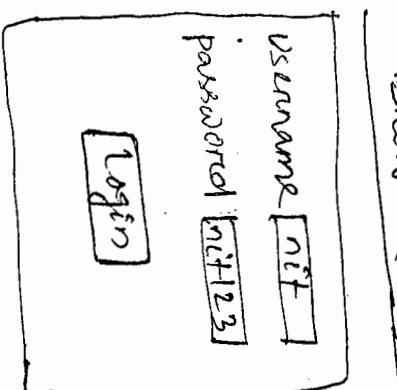
    </servlet>
    <servlet-mapping>
      <servlet-name> s1 </servlet-name>
      <url-pattern> /login </url-pattern>
    </servlet-mapping>
  </web-app>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

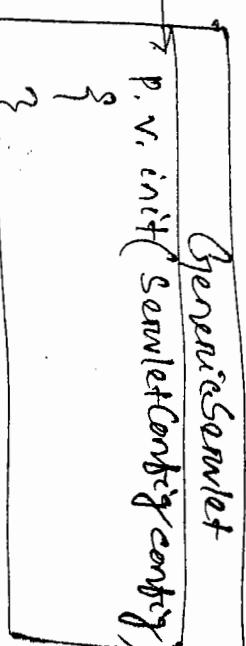
Execution flow of LoginServlet

- ① Load LoginServlet class.
- ② Create object of LoginServlet

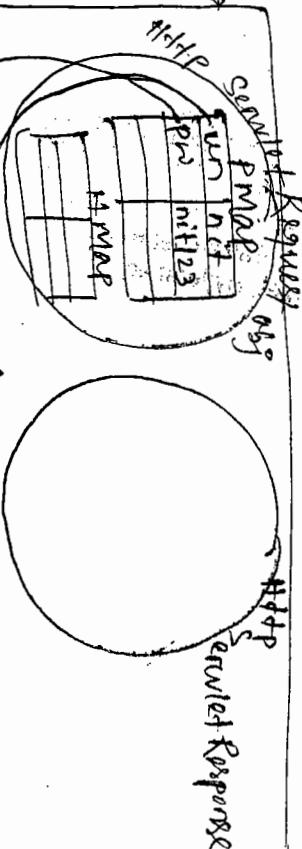
class.forName("com.mysql.jdbc.Driver");
Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/test", "root", "root");



HTTP request
URL: /webapp/
containing login?
username=nit&
pw=nit123
method=GET



- ③ delegate
- ④ create thread



Here as many time as many request are created the request goes to the many connection with server. Here request goes to connection server. It is not efficient.

```

class.forName("-----");
Connection con = DriverManager.getConnection("-----");
PreparedStatement ps = con.prepareStatement("select * from user where name=? and pw=?");

```

Servlet Initialization

- Servlet is initialized by calling init method
- It is a lifecycle method of Servlet.
- This method is called by Servlet container only once.

Q. Is it good idea to create constructor in Servlet?

→ If there is no constructor within class compiler provide a constructor without any parameters. That is called as default constructor of compiler.

- Servlet is loaded and created object dynamically using newInstance() method.
- This method create object by calling zero argument constructor.
- It is not good idea to define constructor within Servlet bcoz same operation is done by init method.

Q. Can you define parameterized constructor with Servlet?

- Yes, provided default constructor.

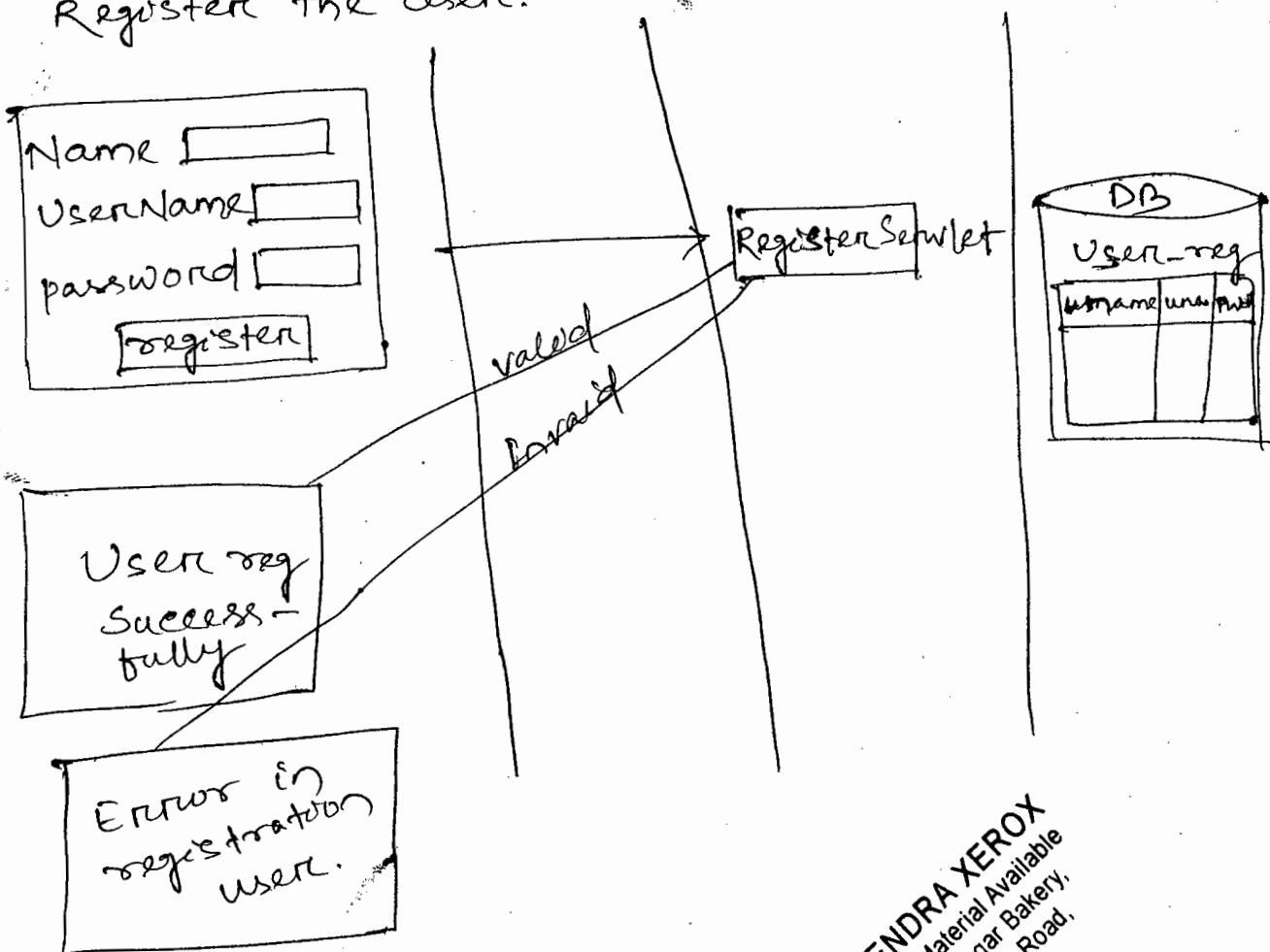
→ Servlet is initialized by overriding init method of GenericServlet.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad

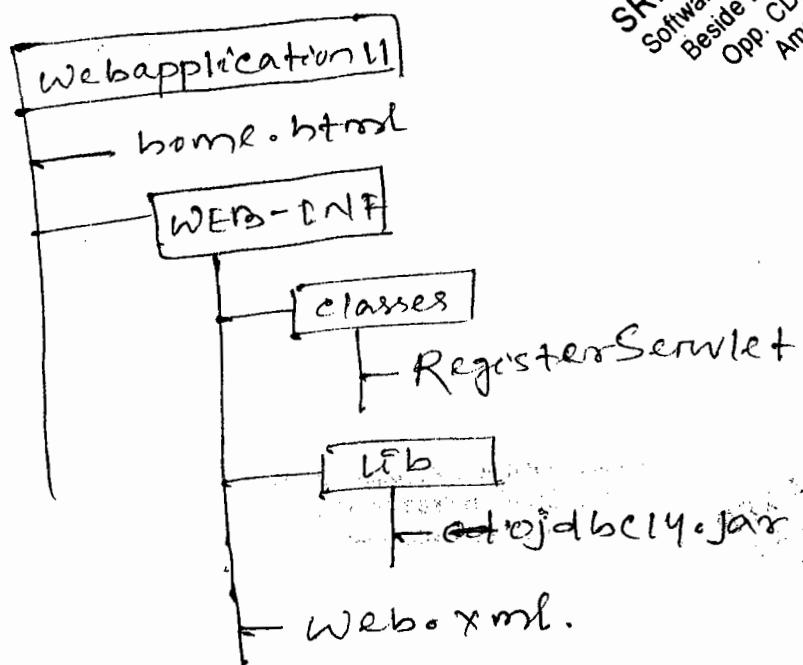
VENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Program

Registers the user.



Directory Structure



SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

home.html (Developing registration Page)

```
<html>
<body bgcolor="cyan">
<form action="/register">
<h2>
  Name <input type="text" name="t1"><br>
  Username <input type="text" name="t2"><br>
  Password <input type="text" name="t3"><br>
  <input type="submit" value="register">
</h2>
</form>
</body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Developing Servlet

```
package com.nit.servlets;
import javax.servlet.*;
import java.io.*;
import java.sql.*;
public class RegisterServlet extends GenericServlet
{
    private Connection cn;
    public void init(ServletConfig config)
        throws ServletException
    {
        try{
            Class.forName("oracle.jdbc.driver.OracleDriver");
            cn = DM.getConnection("-----");
        }
    }
}
```

```
catch (Exception k)
```

```
{
```

```
    k. printStackTrace();
```

```
}
```

```
public void service (ServletRequest req,  
                     ServletResponse res)
```

```
throws ServletException, IOException,
```

```
{
```

```
String name = req. getParameter ("t1");
```

```
String user = req. getParameter ("t2");
```

```
String pwd = req. getParameter ("t3");
```

```
String type = "text/html";
```

```
res. setContentType (type);
```

```
PrintWriter out = res. getWriter ();
```

```
try
```

```
{
```

```
PreparedStatement ps = cn. prepareStatement  
("insert into user values (?, ?, ?)");
```

```
( " insert into user values ( ?, ?, ? ) " );
```

```
ps. setString (1, name);
```

```
ps. setString (2, user);
```

```
ps. setString (3, pwd);
```

```
ps. executeUpdate ();
```

```
int count = ps. executeUpdate ();
```

```
out. println ("

## User registered successfully

");
```

```
}
```

```
catch (SQLException s)
```

```
{ out. println ("

## User is already exist

");
```

```
}
```

```
} }
```

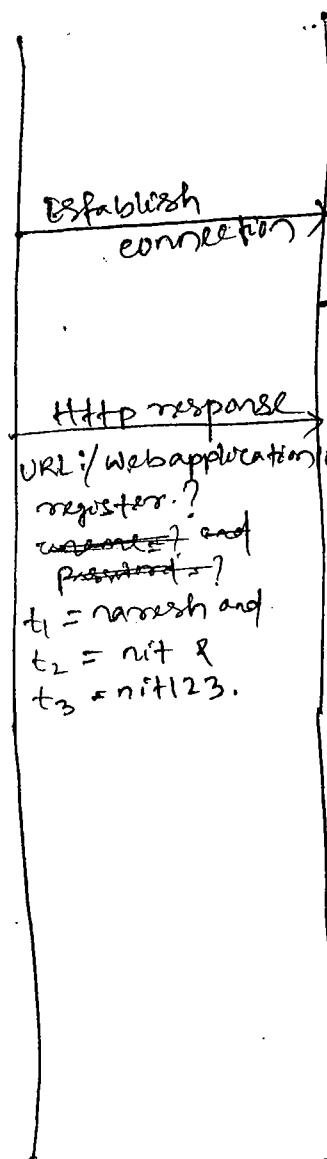
Web.xml

```

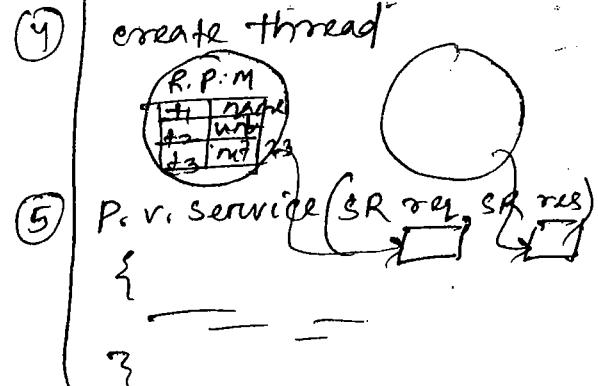
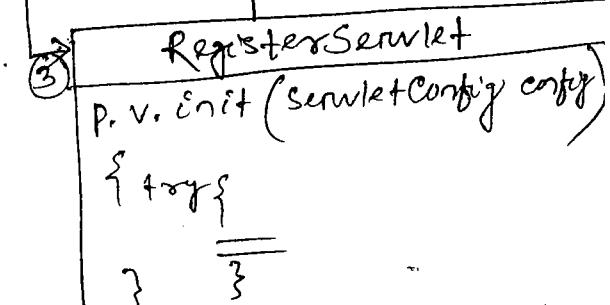
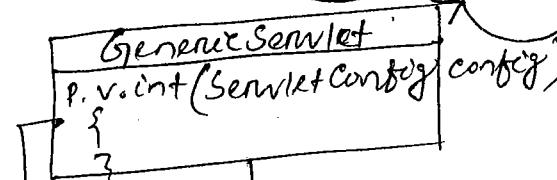
<web-app>
  <servlet>
    <servlet-name> S1 </servlet-name>
    <servlet-class> com.nit.servlets.RegisterServlet
      </servlet-class>
    </servlet>
  <servlet-mapping>
    <servlet-name> S1 </servlet-name>
    <url-pattern> /register </url-pattern>
  </servlet-mapping>
</web-app>

```

Name [Nareesh]
Username [nit]
Password [nit123]
<input type="checkbox"/> register

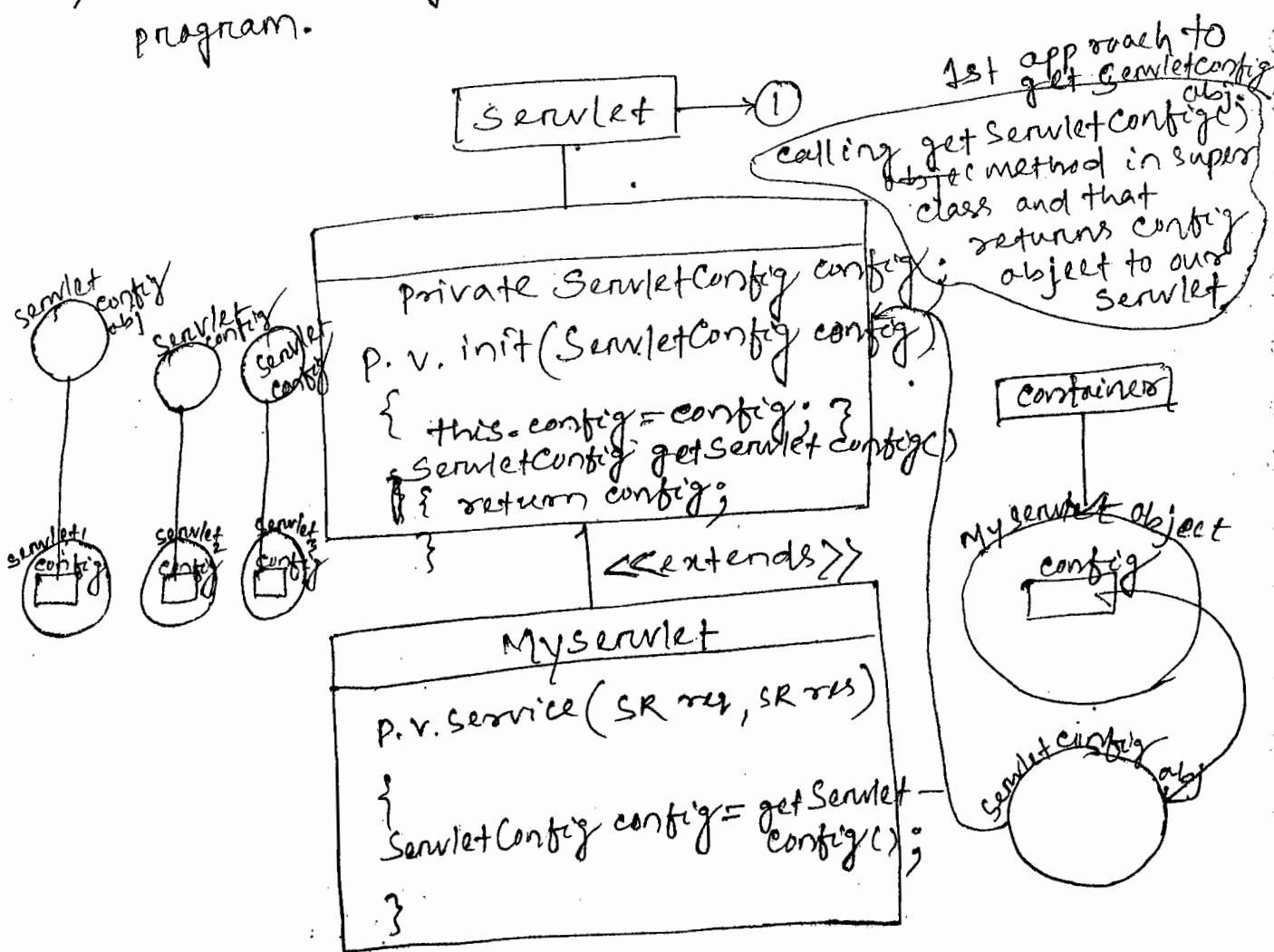


- ① Load RegisterServlet class
- ② create object of RegisterServlet



ServletConfig :-

- ServletConfig is an interface. This interface is implemented by servlet container.
- ServletConfig is configuration object used by servlet container to pass information to a servlet during servlet initialization.
- ServletConfig is one of the pre-defined interface.
- ServletConfig object is used for developing flexible servlets.
- ServletConfig object exist one per servlet program.



- An object of `ServletConfig` created by the container during its initialization phase.

- An object of `ServletConfig` is available to the `Servlet` during its execution, once the `Servlet` execution is completed automatically `ServletConfig` object will be removed by the container.
- Initial values of `Servlet` is defined inside `web.xml` as `init parameters` or `values`.

Syntax:-

```

<Web-app>
  <Servlet>
    <Servlet-name> name</Servlet-name>
    <Servlet-class> classname</Servlet-class>
    <init-param>
      <param-name> name</param-name>
      <param-value> value</param-value>
      </init-param>
      <init-param>
        <param-name> name</param-name>
        <param-value> value</param-value>
      </init-param>
    ...
    </Servlet>
    <Servlet-Mapping>
      <Servlet-name> name</Servlet-name>
      <url-pattern>/url</url-pattern>
    </Servlet-Mapping>
  </Web-app>

```

→ Parameters are Strings.

Methods of ServletConfig Interface :-

[String getInitParameter(String name)]

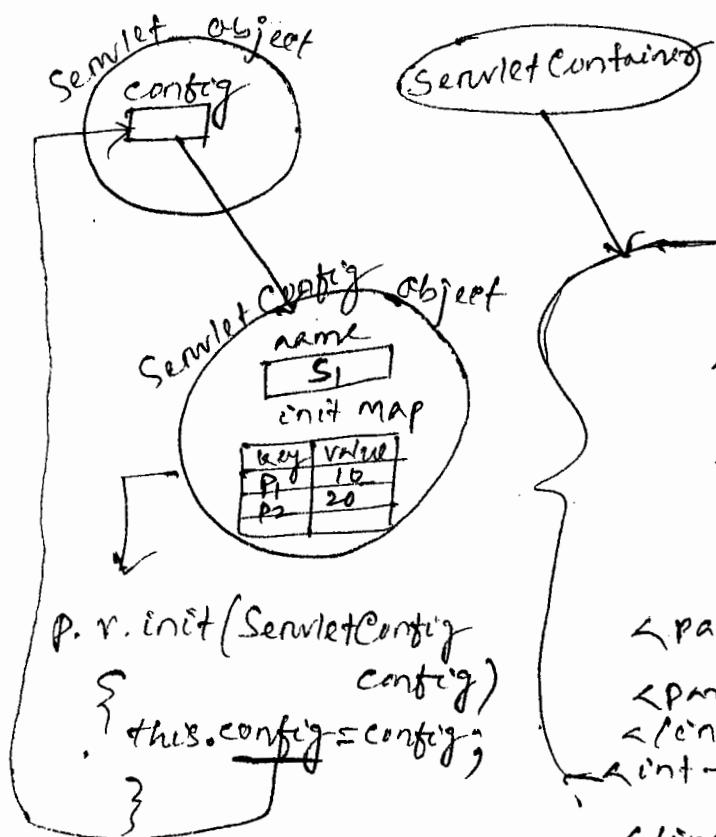
→ Returns a string containing the value of the named initialization parameters, or null if the parameter does not exist.

[Enumeration getInitParameterNames()

→ Returns the names of the servlet's initialization parameters as an Enumeration of string objects, or an empty Enumeration if the servlet has no initialization parameters.

[String getServletName()

→ Returns the name of this servlet instance



web.xml

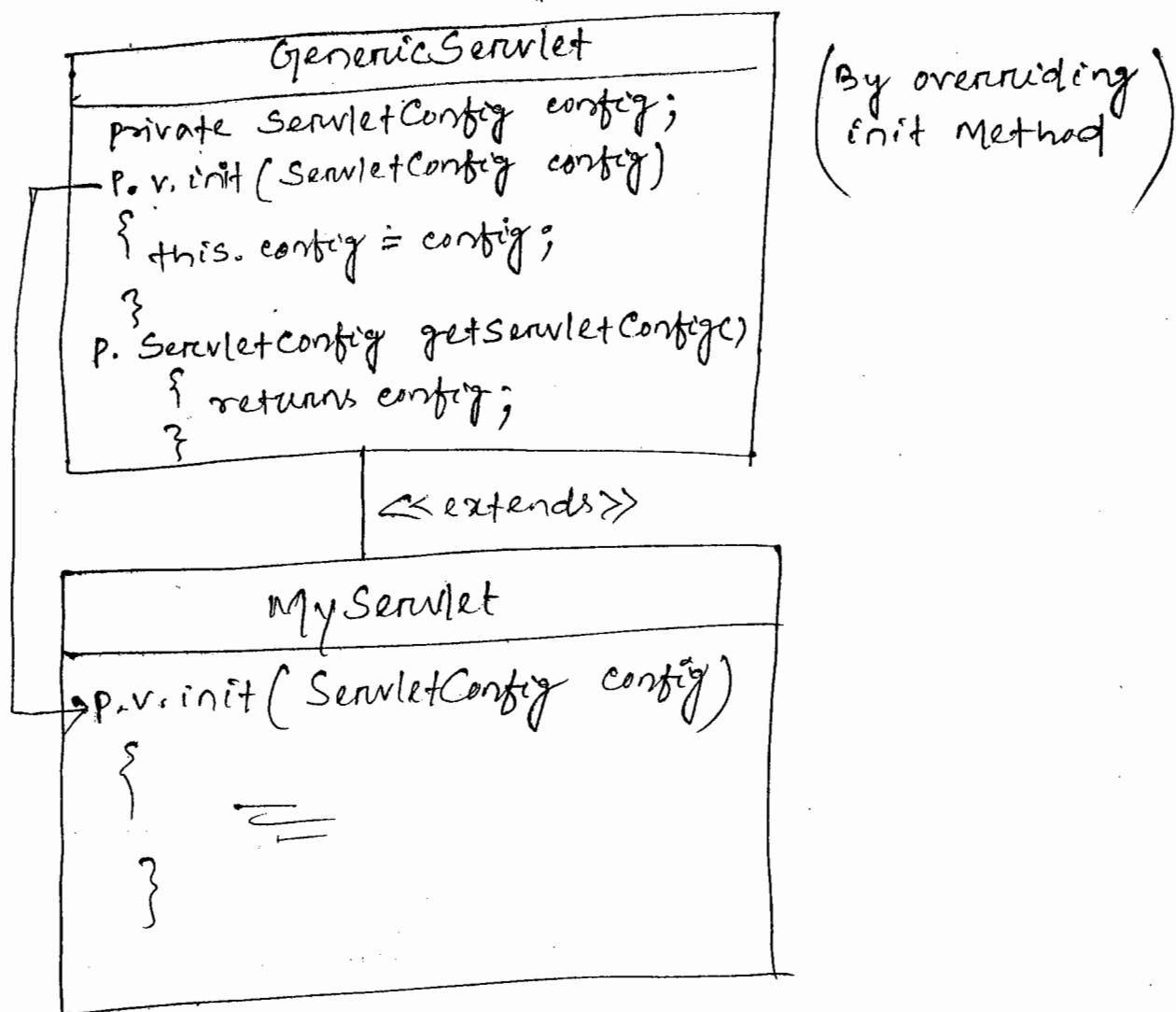
```
<web-app>
    <servlet>
        <servlet-name>S1</servlet-name>
        <servlet-class>
            com.it.servlets.Servlets
        </servlet-class>
        <init-param>
            <param-name>P1</param-name>
            <param-value>10</param-value>
            <param-name>P2</param-name>
            <param-value>20</param-value>
        </init-param>
    </servlet>
```

- Q. How to get object of ServletConfig object?
- getServletConfig method of Servlet interface / GenericServlet.

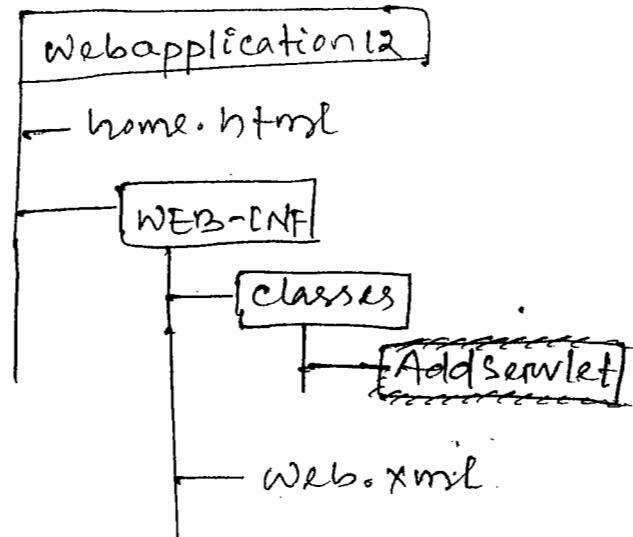
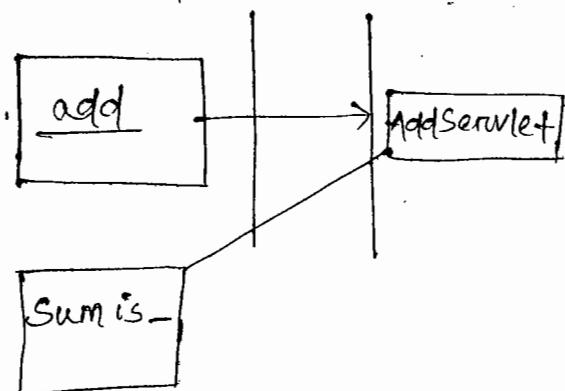
Eg:-

ServletConfig config = getServletConfig();

and approach to get config object :-



Q. Simple application for Servlet initialization.



home.html

```
<html>
<body bgcolor="cyan">
<h2>
<a href = "http://localhost:8086/Webapplication12/
add">Add</a>
</h2>
```

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

Develop Servlet:-

```
package com.nit.servlets;
import javax.servlet.*;
import java.io.*;
public class AddServlet extends GenericServlet
{
    public void service(ServletRequest req,
                        ServletResponse res)
        throws ServletException, IOException
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
O
O {
C     ServletConfig config = getServletConfig();
C     int num1 = Integer.parseInt(config.getInteger-
C                               Parameter("a"));
C
C     int num2 = Integer.parseInt(config.get Integer-
C                               initialParameter("b"));
C
C     int num3 = num1 + num2;
C
C     res.setContentType("text/html");
C     PrintWriter out = res.getWriter();
C     out.println("<h2> Sum is " + num3 + "</h2>");
C
C }
```

web.xml

```
<web-app>
    <Servlet>
        <Servlet-name> s1 </Servlet-name>
        <Servlet-class> com.it.servlets.AddServlet </Servlet-
                               class>
        <init-param>
            <param-name> a </param-name>
            <param-value> 10 </param-value>
            <init </init-param>
        <init-param>
            <param-name> b </param-name>
            <param-value> 20 </param-value>
        </init-param>
    </Servlet>
    <Servlet-Mapping>
```

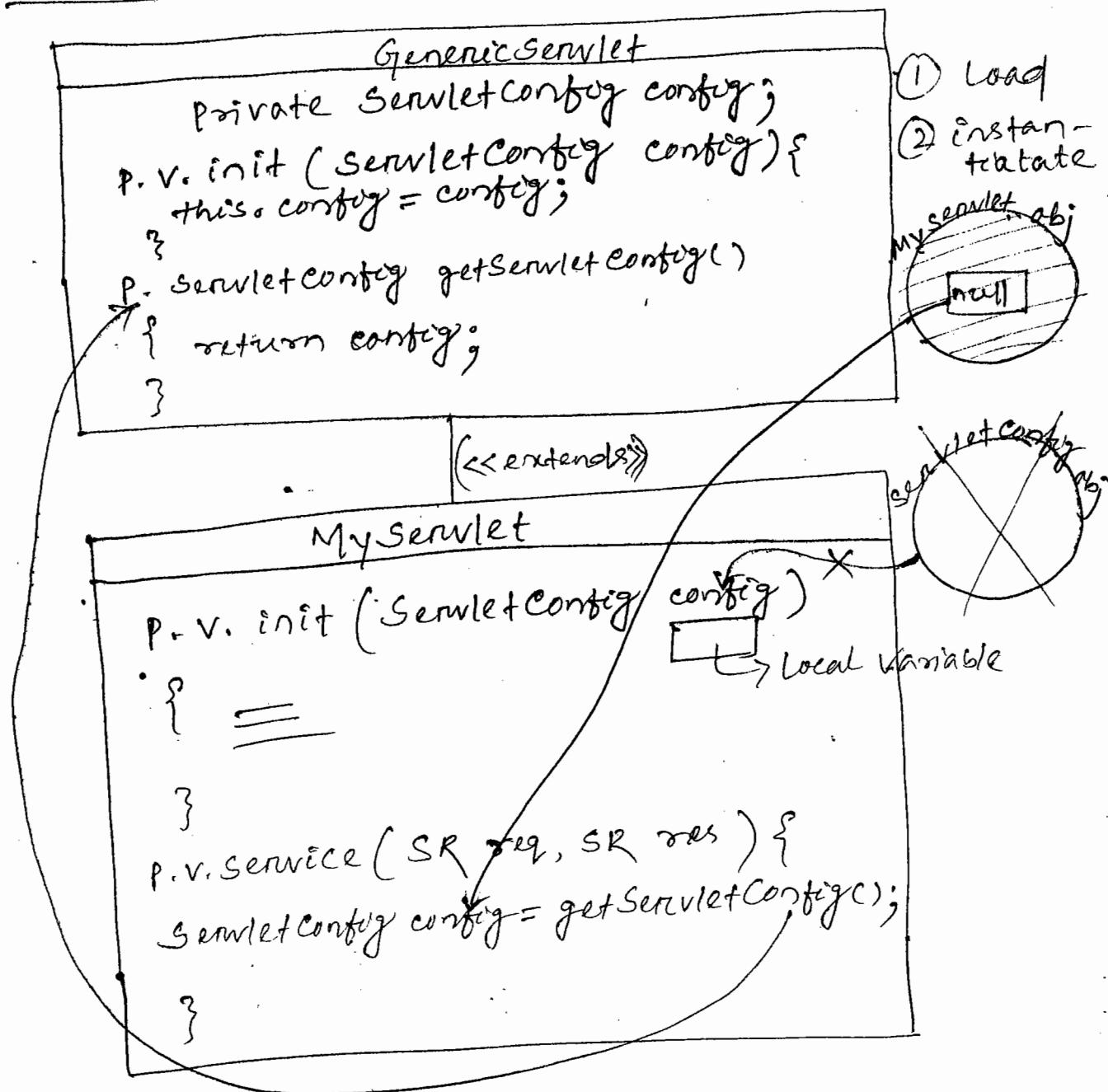
```
< servlet-name > $1 </servlet-name>
< url-pattern > /add </url-pattern>
</servlet-mapping>
```

overriding init. method

- init method is override in order to execute block of statements on creation of Servlet object.
- Servlet is initialized by calling init method.
- Why super.init(config) will be the first statement inside init(config) method?
(or)
- Why overriding init method must call init method of Super class?

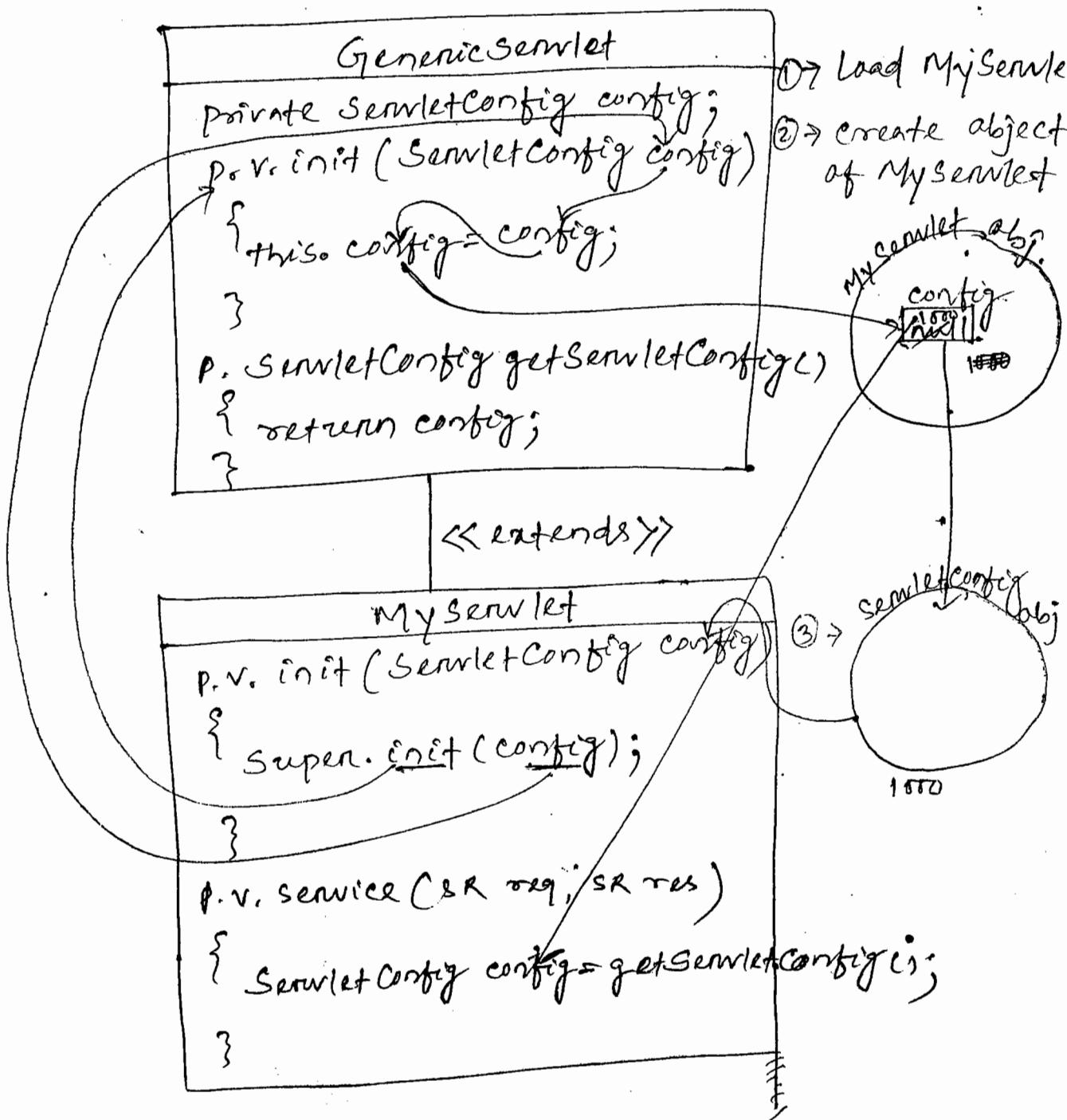
Ans:- This will be the first statement if we are overriding the init(config) method by this way we will store the config object for future reference and we can use by getServletConfig() to get information about config object if we will not do this config object will be lost and we have only one way to get config object bcz Servlet pass config object only in init method. without doing this if we call the servletConfig method we will get NullPointerException.

without calling init method of Super class

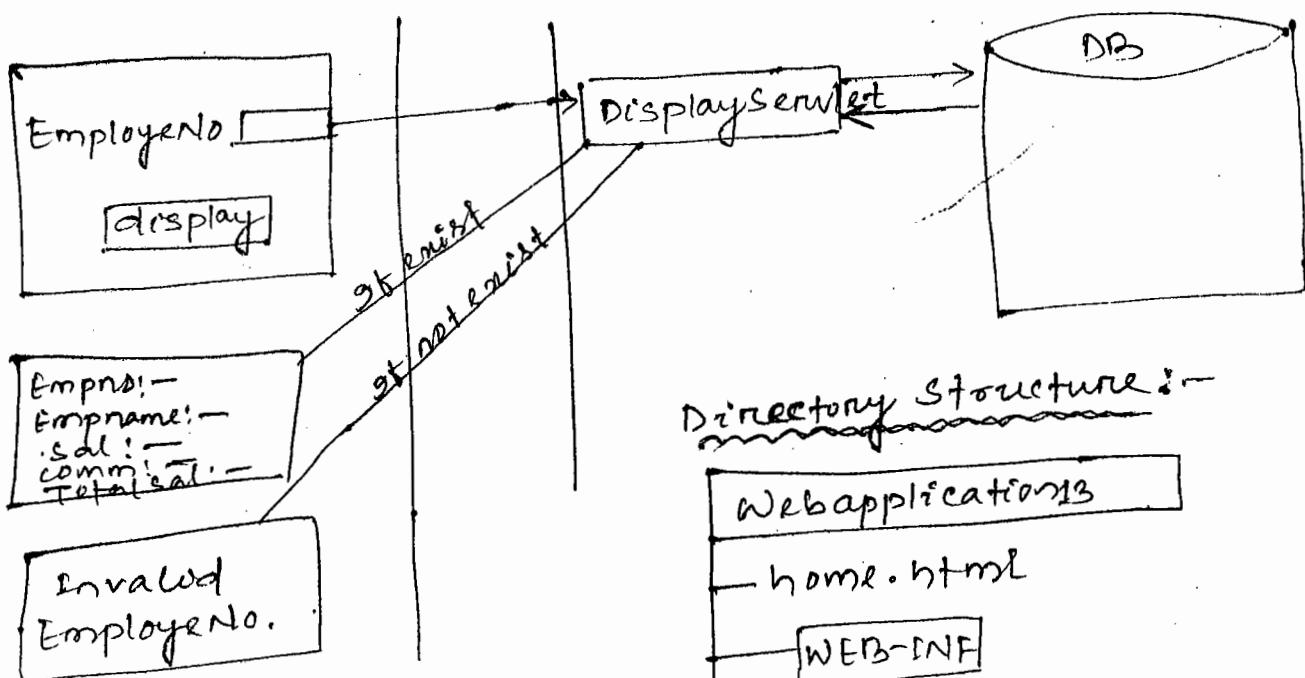


SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

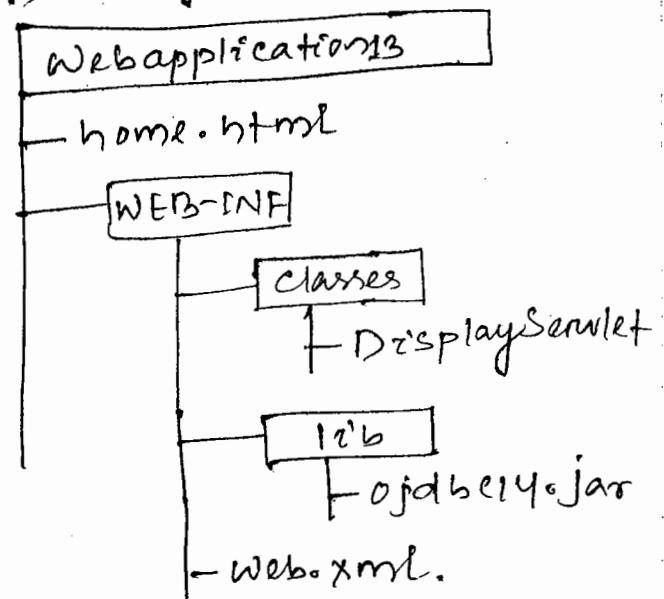
calling init method of super class within subclass



Example :-



Directory Structure :-



~~home.html~~

~~Web.xml~~ :

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

<web-app>
  <servlet>
    <servlet-name> s1 </servlet-name>
    <servlet-class> com.nit.servlets.DisplayServlet
      </servlet-class>
      <init-param>
        <param-name> driver </param-name>
        <param-value> oracle.jdbc.driver.OracleDriver
          </param-value>
        </init-param>
        <init-param>
          <param-name> url </param-name>
          <param-value> serverXE </param-value>
            ← init <init-param>
            <init-param>
              <param-name> user </param-name>
              <param-value> scott<anjan system> </param-value>
                ← init <init-param>
                <init-param>
                  <param-name> prod </param-name>
                  <param-value> tiger<anjan> </param-value>
                  </init-param>
                </servlet>
                <servlet-mapping>
                  <servlet-name> s1 </servlet-name>
                  <url-pattern> /display </url-pattern>
                </servlet-mapping>
  </web-app>

```

SRI RAJAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

```

C home.html
C <html>
C <body bgcolor="cyan">
C <form action="oldisplay">
C   <h2>
C     EmployeeNo <input type="text" name="t1"> <br>
C     <input type="submit" value="display">
C   </h2>
C   </form>
C </body>
C </html>
C Servlet
C package com.net.servlets;
C import javax.servlet.*;
C import java.io.*;
C import java.sql.*;
C public class DisplayServlet extends GenericServlet
C {
C   private Connection cn;
C   public void init (ServletConfig config)
C     throws ServletException
C   {
C     super.init(config);
C     try {
C       class.forName (config.getInitParameter
C                     ("driver"));
C       cn = DM.getConnection (config.getInitParameter
C                             ("url"),
C                             config.getInitParameter ("user"),
C                             config.getInitParameter ("pwd"));
C     }
C   }

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

```
catch (Exception k)
{
    k.printStackTrace();
}

public void service (ServletRequest req,
                     ServletResponse res)
throws ServletException, IOException
{
    int eno = Integer.parseInt (req.getParameter ("e1"));
    res.setContentType ("text/html");
    PrintWriter out = res.getWriter();
    try
    {
        PreparedStatement ps = cn.prepareStatement
        ("select empno, ename, sal, nvl(comm, 0),
         sal+nvl(comm, 0) from emp where
         empno = ?");
        ps.setInt (1, eno);
        ps.executeUpdate ();
        ResultSet rs = ps.executeQuery ();
        boolean b = rs.next ();
        if (b == false)
            out.println ("Invalid employeeNo");
        else
        {
            out.println ("e1");
            out.println ("EmployeeNo " + rs.getInt (1));
            out.println ("EmployeeName " + rs.getString (2));
            out.println ("Salary " + rs.getFloat (3));
        }
    }
}
```

```

out.println("<b> comm" + rs.getFloat(4));
out.println("<b> Total Salary" + rs.getFloat(5));
out.println("<h2>");
}
catch(Exception e)
{
    catch(Exception s)
    {
        s.printStackTrace();
    }
}
}

```

→ Generic Method

→ GenericServlet provide 2 init methods in order to initialize servlet.

1. public void init (ServletConfig config)
2. public void init()

1. public void init (ServletConfig config) is called life cycle method. This method is called by servlet container during initialization of servlet.

2. public void init() is a non lifecycle method. This method is called by init method with parameters.

GenericServlet

```
private ServletConfig config;  
p. v. init (ServletConfig config)  
{ this.config = config;  
  init()  
}
```

```
p. v. init () {  
  —  
}
```

```
p. v. public ServletConfig getServletConfig()  
{ return config;  
}
```

calls

↳ extends >

Myservlet

```
p. v. init ()  
{  
  ServletConfig config = getServletConfig();  
}
```

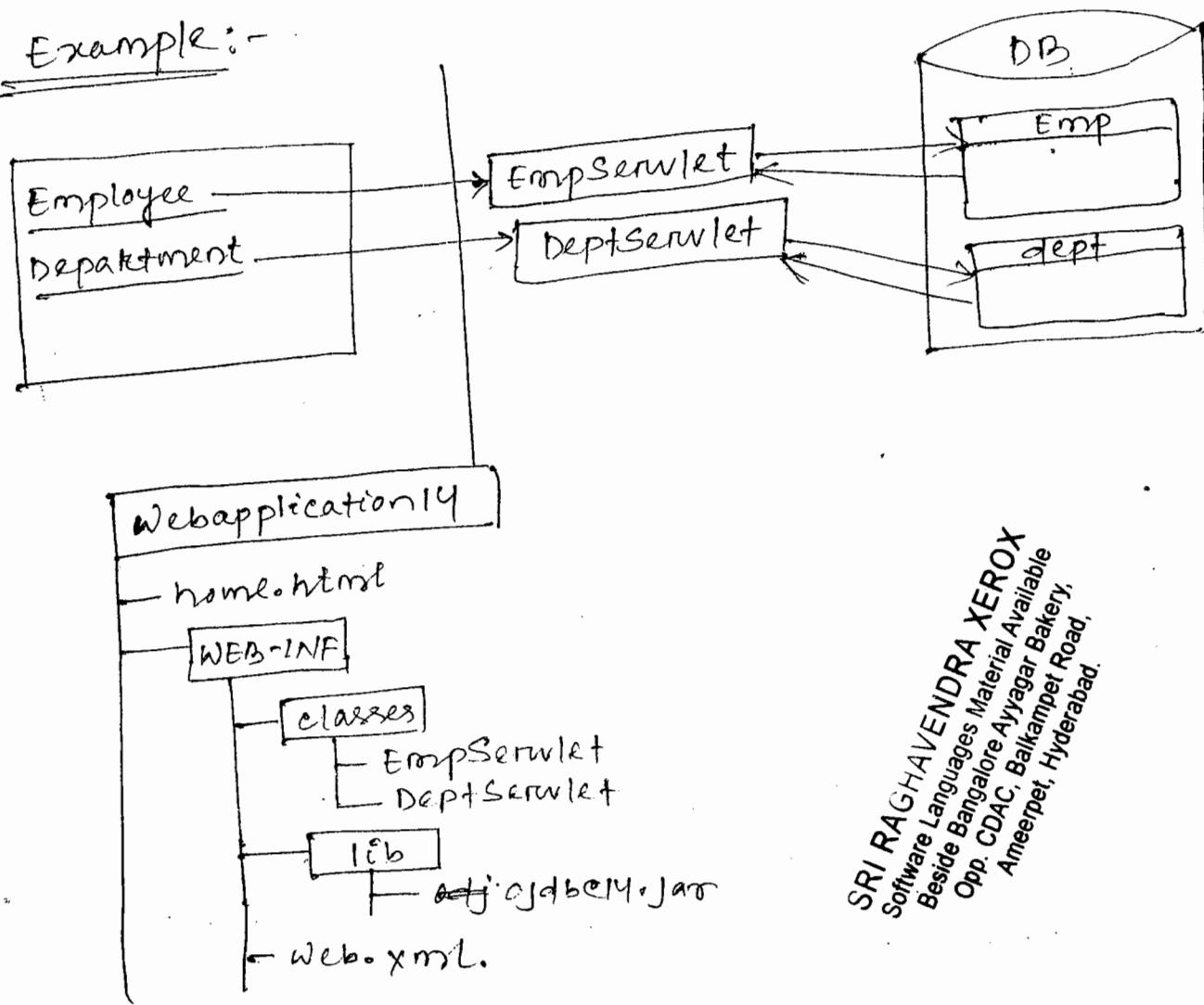
Non life cycle
Method

Q. why we should override only no-args
init() method.

Ans:- If we override init(ServletConfig config) method, then the first statement should be superclass init(ServletConfig config) method is invoked first. That's why GenericServlet provides another helper init() method without argument that gets called at the end of init(ServletConfig config) method.

→ we should always utilize this method for overriding init() method to avoid any issues as we may forget to add super.init() call in overriding init method with ServletConfig argument.

Example:-



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
<Web-app>
  <servlet>
    <s-n> s1 </s-n>
    <s-c> com.nit.servlets.EmpServlet </s-c>
      <init-param>
        <param-name> driver </param-name>
        <param-value> oracle.jdbc.driver.OracleDriver
          </param-value>
      </init-param>
      <init-param>
        <param-name> url </param-name>
        <param-value> Server </param-
          jabc:oracle:thin:@localhost:
          1521:XE </param-value>
      </init-param>
    </servlet>
    <servlet>
      <servlet-name> s2 </servlet-name>
      <servlet-class> com.nit.servlets.DeptServlet
        </servlet-class>
      <init-param>
        <p-n> driver </p-n>
        <p-v> oracle.jdbc.driver.OracleDriver </p-v>
      </init-param>
      <init-param>
        <p-n> url </p-n>
        <p-v> jabc:oracle:thin:@localhost:1521:XE </p-v>
```

```
</init-param>
</servlet>
</servlet-mapping>
<servlet-name> s1 </servlet-name>
<url-pattern> /emp </url-pattern>
</servlet-mapping>
</servlet-mapping>
<servlet-name> s2 </servlet-name>
<url-pattern> /dept </url-pattern>
</servlet-mapping>
</servlet-mapping>
</web-app>
```

Servlet :-

```
package com.cit.servlets;
import javax.servlet.*;
import java.io.*;
public class EmpServlet extends GenericServlet
{
    private Connection cn;
    public void init()
    {
        ServletConfig config = getServletConfig();
        try
        {
            Class.forName(config.getInitParameter("driver"));
            cn = DM.getConnection(config.getInitParameter
                ("url"), "scott", System "tiger");
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

```

        catch (Exception k)
    {
        k. printStackTrace ();
    }
}

} // close init Method

public void service(SR res, SR res) throws
ServletException, IOException

{
    res.setContentType ("text/html");
    PrintWriter out = res.getWriter();
    try
    {
        PreparedStatement ps = cn. prepareStatement
        ("select empno, ename, sal from emp");
        ps. executeQuery();
        ResultSet rs = ps. executeUpdate();
        while (rs. next())
        {
            out. println (rs. getInt(1) + ":" + rs. getString(2)
            + ";" + rs. getFloat(3) + "<br>");
        }
    }
    catch (Exception R)
    {
        R. printStackTrace();
    }
}
}

```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

name.html

```
<html>
<body bgcolor=cyan>
```

```
<h2>
```

```
<a href="http://localhost:8086/webapplication14/
emp"> Employee Details </a>
```

```
<a href="http://localhost:8085/webapplication14/
dept"> Department Details </a>
```

```
</h2>
```

```
</body>
```

```
</html>
```

Dt:- 21-11-15

ServletContext :-

→ ServletContext is an interface. This interface is implemented by servlet container.

→ An object of ServletContext is created by Webcontainer at the time deploying web application or project.

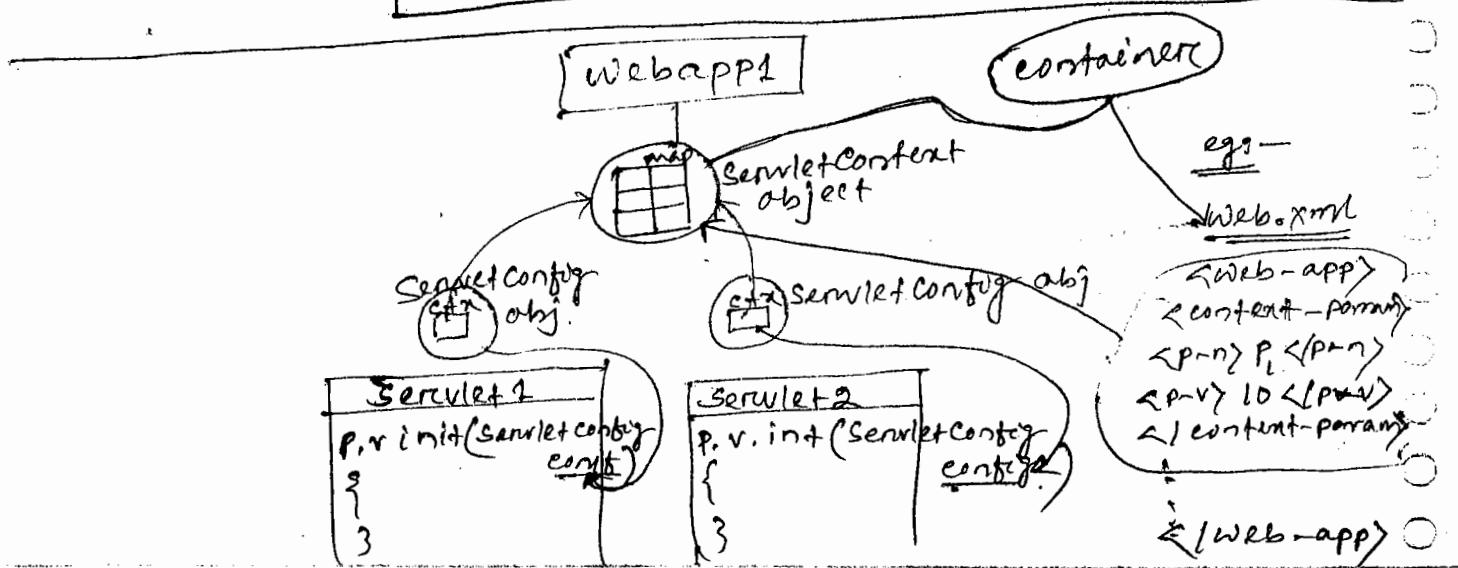
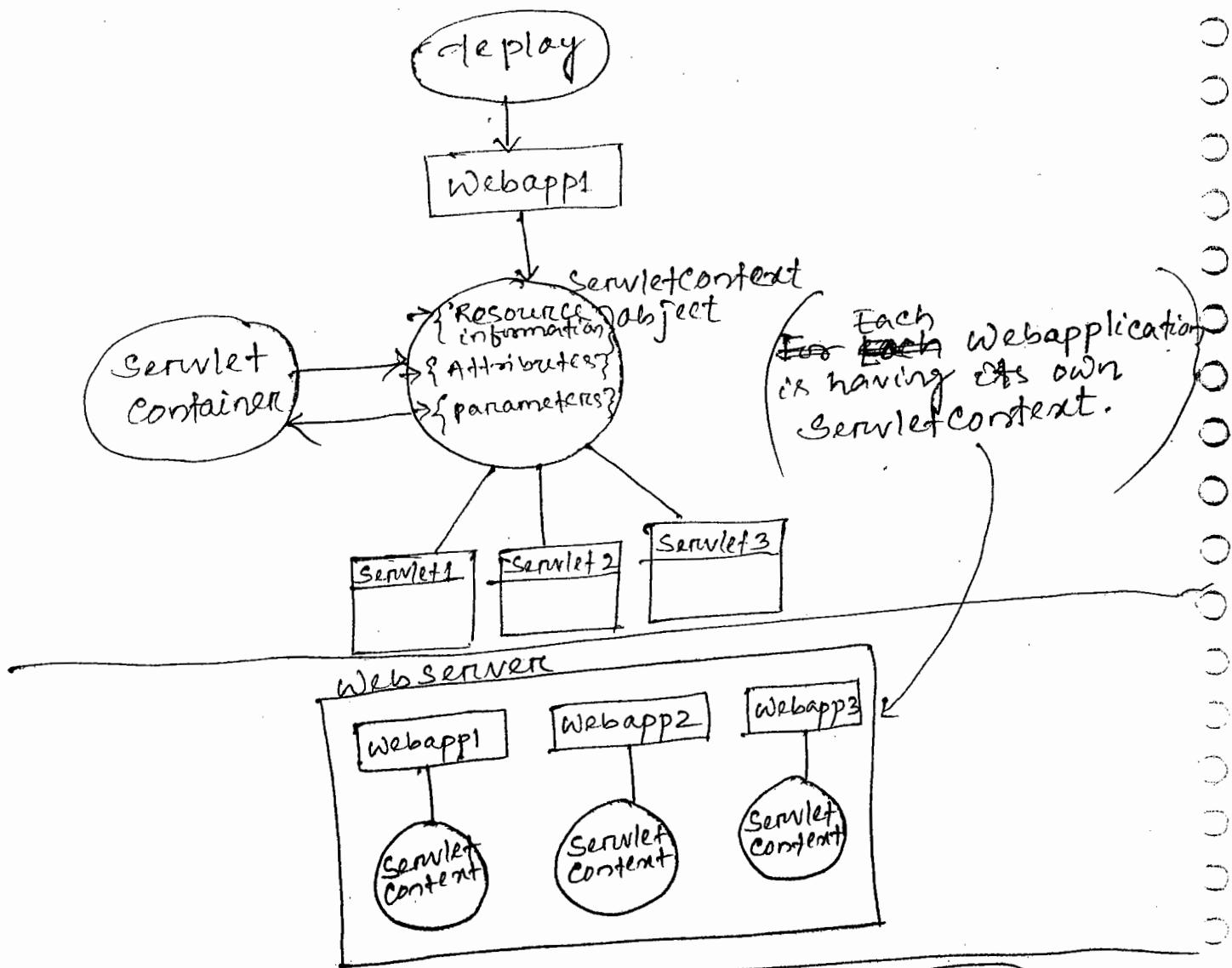
→ There is only one ServletContext object per webapplication.

→ The ServletContext object is contained within the ServletConfig object, which the Web server provides the servlet when the servlet is initialized.

→ ServletContext is an interface which helps us to communicate with the servlet container.

→ It gives information about ~~the~~ environment.

- This object can be used to provide inter-application communication.
- It is used to provide global data to webapplication.

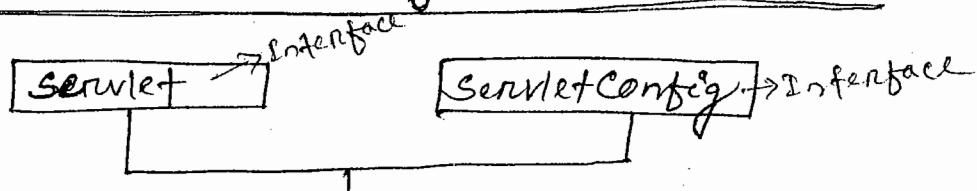


→ using this object we can read global init parameters defined inside [Web.xml] file of web application.

Syntax of defining context parameters :-

```
<web-app>
  <context-param>
    <param-name> name </param-name>
    <param-value> value </param-value>
  </context-param>
  <context-param>
    <param-name> name </param-name>
    <param-value> value </param-value>
  </context-param>
  < servlet>
    < servlet-name> name </servlet-name>
    < servlet-class> classname </servlet-class>
    < init-param>
      < param-name> name </param-name>
      < param-value> value </param-value>
    </init-param>
  </servlet>
</web-app>
```

How to get ServletContext object within Servlet:-



```
GenericServlet
private ServletConfig config;
p.v. init (ServletConfig config)
{
    this.config = config;
    init();
}
public ServletConfig getServletConfig()
{
    return config;
}
public ServletContext getServletContext()
{
    return config.getServletContext();
}
```

«extends»

MyServlet

```
p.v. service (SR req, SR res)
```

```
{
    ①. ServletConfig config = getServletConfig();
        ServletContext ctx = config.getServletContext();
    ②. ServletContext ctx = getServletContext();
}
```

There are two approach to get ServletContext object.

①. getServletContext() method of ServletConfig interface

ServletContext ctn = getServletConfig().getServlet-

② getServletContext() method of GenericServlet. ^{context();}

Methods of ServletContext interface :-

[String getInitParameter(String name)]

→ Returns a string containing the value of the named context-wide initialization parameter, or null if the parameter does not exist.

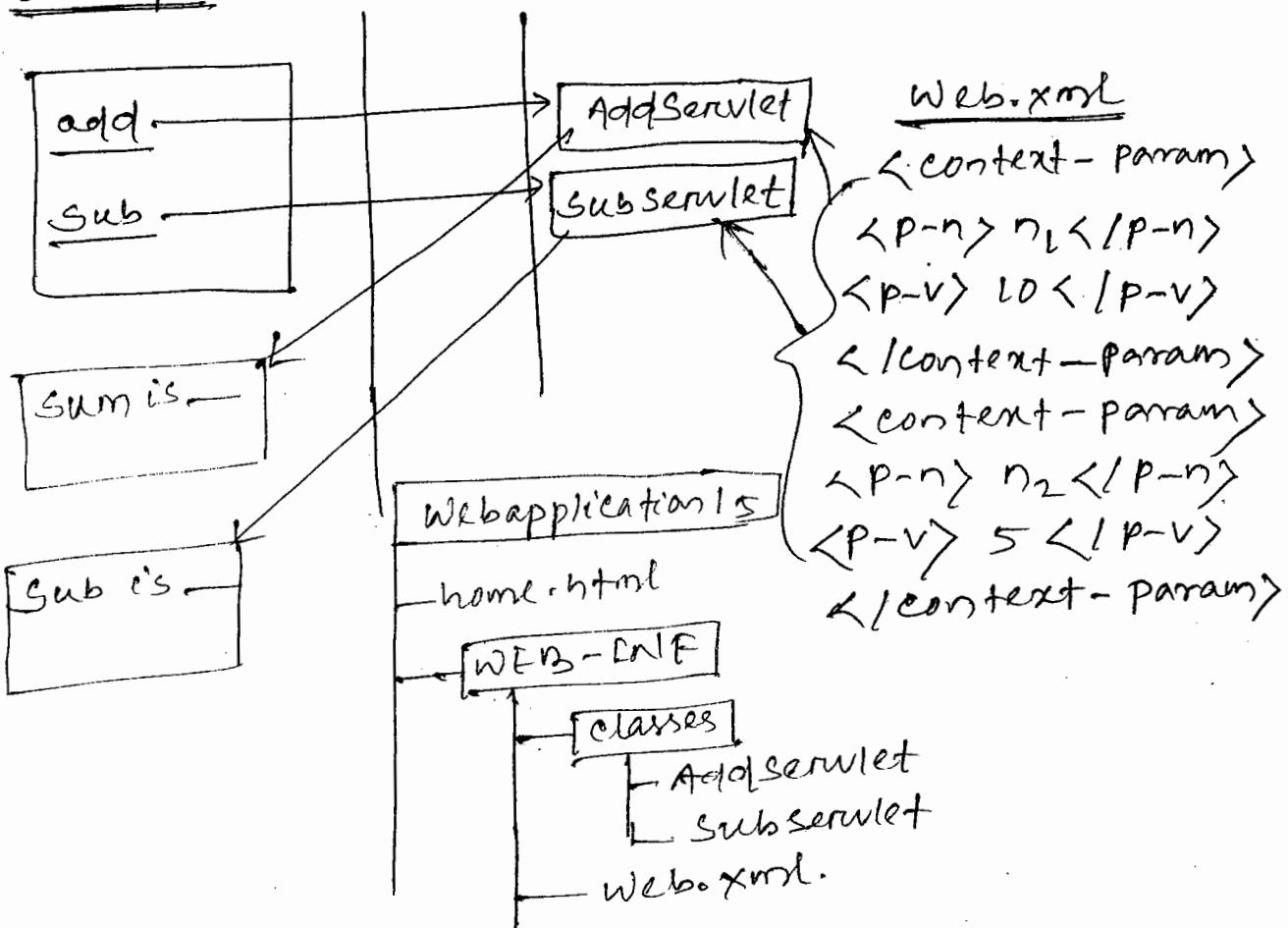
→ [Enumeration<String> getInitParameterNames()]

Returns the names of the contexts initialization parameters as an Enumeration of String objects or an empty Enumeration

→ [boolean setInitParameter(String name, String value)]

Sets the content initialization Parameter with the given named and value of this ServletContext.

Example



home.html

```
<html>
<body bgcolor = cyan>
<h2>
<a href = "http://localhost:8085/Webapplication15/add"> Add </a><br>
<a href = "http://localhost:8085/Webapplication15/sub"> sub</a><br>
</h2>
</body>
</html>
```

Servlet :-

```

C package com.nit.servlets;
C import javax.servlet.*;
C import java.io.*;
C public class AddServlet extends GenericServlet
C {
C     public void service(ServletRequest req, ServletResponse res)
C         throws ServletException, IOException
C     {
C         ServletContext cta = getServletContext();
C         int num1 = Integer.parseInt(cta.getInitParameter("n1"));
C         int num2 = Integer.parseInt(cta.getInitParameter("n2"));
C         int num3 = num1 + num2;
C         PrintWriter out = res.getWriter();
C         out.println("<h2> Sum is " + num3 + "</h2>");
C     }
C }

```

AddServlet

package com.nit.servlets;

SubServlet

```
import javax.servlet.*;
import java.io.*;
```

```
public class SubServlet extends GenericServlet
```

```
public void service(ServletRequest req, ServletResponse res)
```

```
throws ServletException, IOException
```

```
ServletConfig config = getServletConfig();
```

```
ServletContext cta = config.getServletContext();
```

```
int num1 = Integer.parseInt(cta.getInitParameter("n1"));
```

```
int num2 = Integer.parseInt(ctx.getInitParameter("n2"));
int num3 = num1 + num2;
PrintWriter out = res.getWriter();
out.println("<h2> Sub is " + num3 + "</h2>");
}
```

Web.xml

```
<web-app>
<context-param>
<param-name>n1</param-name>
<param-value>10</param-value>
</context-param>
<context-param>
<param-name>n2</param-name>
<param-value>5</param-value>
</context-param>
< servlet >
< servlet-name >s1</servlet-name>
< servlet-class >com.nit.servlets.AddServlet</servlet-class>
< /servlet >
< servlet >
< servlet-name >s2</servlet-name>
< servlet-class >com.nit.servlets.SubServlet</servlet-class>
< /servlet >
```

```

< servlet-mapping >
  < s-> S1 < /s->
  < u-p> /add </u-p>
  < /servlet-mapping >
  < servlet-mapping >
    < s-> S2 < /s->
    < u-p> /sub </u-p>
  < /servlet-mapping >
< /web-app >

```

D= 23.11.15

Q. what is the difference between ServletConfig and ServletContext?

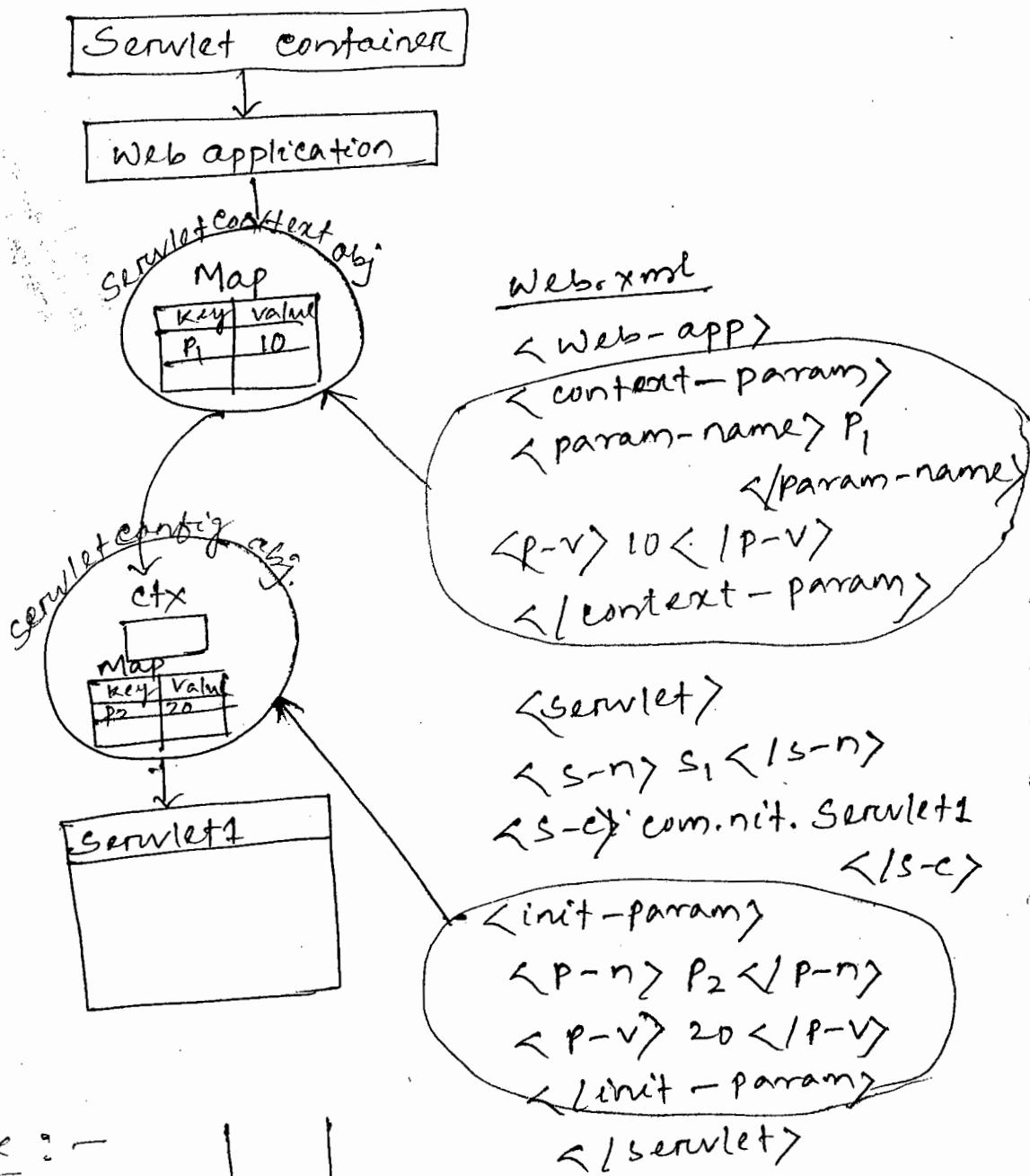
ServletConfig

- ServletConfig represent single Servlet.
- gets like local parameters associated with particular servlet.
- gets a name value pair defined inside the servlet section of web.xml file so it has servlet wide scope
- getServletConfig() method is used to get the config object.
- string based name value pairs

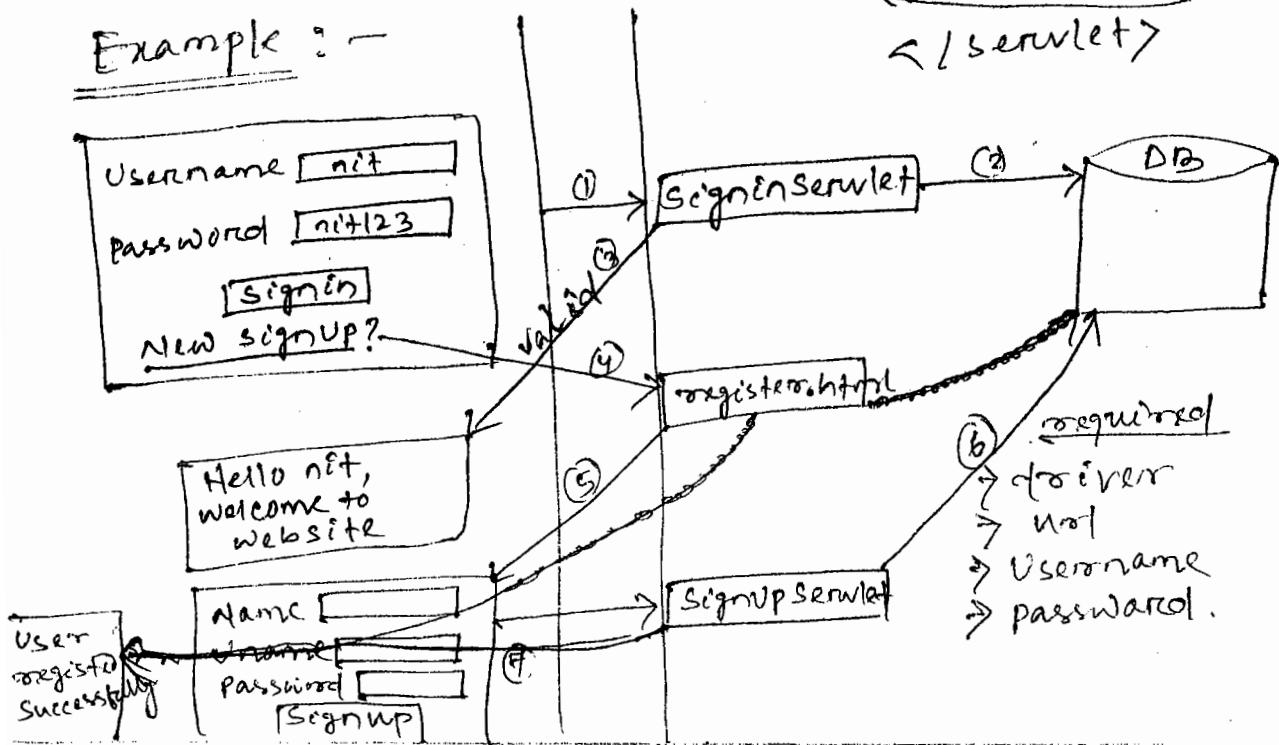
- get represent whole web application running on particular JVM and common for all the servlets.
- gets like global parameter associated with whole appl.
- ServletContext has appl wide scope so define outside of servlet tag in web.xml file.
- getServletContext() Method is used to get ~~this~~ context object.
- Name value pair in the form of both string & object types.

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

SRI RAJENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.



Example :-



Working with Eclipse IDE :-

- Eclipse is a Java IDE (Integrated Development Environment).
- IDE is used for
 - (i) Designing project
 - (ii) Developing project
 - (iii) Deploying project
 - (iv) Testing project.

→ Download Eclipse IDE for JEE

Eclipse LUNA

Eclipse MARS

Eclipse KEPLER

Workspace :-

→ Workspace is a folder where all projects get saved.

~~How to create project~~

How to switch from one workspace to another

→ Select File Menu

→ Select Switch Workspace

Project :- Project is an application in eclipse.

How to create project :-

→ Select File → New → project (Dynamic web project)

Project Name : Webapplication15.

click on **next**

again click on **next**

Context root : ~~naresh~~ ^{naresh} Webapplication15 (change it to (by default) naresh)

Content Directory : WebContent.
(By default)

Select generate Web.xml deployment descriptor.
click on **Finish**

→ Adding servlet-api.jar to webapplication.

→ Select Project

→ Click right mouse button.

→ Select buildpath.

→ Select configure buildpath

→ Select lib(~~tab~~) (tab)

→ Add External.jar

→ Select servlet-api from tomcat7.0/
~~lib~~

→ Click on **OK**

→ How to configure server?

→ Select File → New → Others → Servers → Server.

→ Choose Server

apache

- Tomcat 7.0

click on **Next**

Tomcat installation directory

c:\Tomcat7.0

How to deploy and run web application?

→ Select project in project explorer.

→ Select Run

→ Select RunAs → Run on Server

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Developing project :-

home.html :-

```
<html>
<body bgcolor="cyan">
<form action = "signin">
<h2>
  Username <input type = "text" name = "t1"> <br>
  Password <input type = "password" name = "t2"> <br>
  <input type = "submit" value = "Signin" /> <br>
<a href = "https://localhost:8085/webapplication16/
  signintestuserlogin"
  New User sign UP?> </a> <br>
</h2>
</form>
</body>
</html>
```

Web.xml :-

```
<web-app>
<context-param>
  <param-name> driver </param-name>
  <p-v> oracle.jdbc.driver.OracleDriver </p-v>
</context-param>
<context-param>
  <p-n> url </p-n>
  <p-v> jdbc:oracle:thin:@localhost:1521:XE </p-v>
</context-param>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

<context-param>
<p-n> user </p-n>
<p-v> system </p-v>
<context-param>
<content-param>
<p-n> pwd </p-n>
<p-v> anjan </p-v>
<context-param>
</web-app>

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayagar Bakery
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Servlets

```

package com.nit.servlets;
import java.sql.*;
import java.io.IOException;
@WebServlet("/signin")
public class SigninServlet extends GenericServlet {
  private Connection cn;
  public void init() {
    ServletContext ctn = getServletContext();
    try {
      Class.forName(ctn.getInitParameter("driver"));
      cn = DM.getConnection(ctn.getInitParameter("url"),
        ctn.getInitParameter("user"), ctn.getInitParameter("pwd"));
    }
    catch (Exception k) {
      k.printStackTrace();
    }
  }
}

```

```

C     void
C     public Service (ServletRequest req, ServletResponse res)
C           throws ServletException, IOException {
C
C         String name = req.getParameter ("t1");
C         String p = req.getParameter ("t2");
C         PrintWriter out = res.getWriter ();
C
C         try {
C
C             PreparedStatement ps = cn.prepareStatement
C               ("select count(*) from user-reg where
C                 username=? and pwd=?");
C
C             ps.setString (1, u);
C             ps.setString (2, p);
C
C             ResultSet rs = ps.executeQuery ();
C             rs.next ();
C             int c = rs.getInt (1);
C
C             if (c == 0)
C               out.println ("

## invalid username or password

");
C             else
C               {
C                 out.println ("");
C                 out.println ("");
C                 out.println ("

## Hello "+u+"

");
C                 out.println ("  
 Welcome to website ");
C                 out.println ("");
C               }
C             }
C             catch (Exception k) {
C               k.printStackTrace ();
C             }
C         }

```

SignUpServlet

```
package com.net.servlets;
import java.sql.*;
import java.io.IOException;
@webServlet("signup")
public class SignupServlet extends GenericServlet
{
    private Connection cn;
    public void init()
    {
        ServletConfig config = getServletConfig();
        ServletContext ctx = config.getServletContext();
        try {
            Class.forName(ctx.getInitParameter("driver"));
            cn = DM.getConnection(ctx.getInitParameter("url"),
                ctx.getInitParameter("user"), ctx.getInitParameter("pwd"));
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }
    public void service(ServletRequest req,
                        ServletResponse res)
        throws ServletException, IOException
    {
        String t1 = req.getParameter("t1");
        String t2 = req.getParameter("t2");
        String t3 = req.getParameter("t3");
        PrintWriter out = res.getWriter();
    }
}
```

```
C try{  
C     preparedStatement ps = cn.prepareStatement  
C     ("insert into user-reg values (?, ?, ?)");  
C     ps.setString(1, n);  
C     ps.setString(2, u);  
C     ps.setString(3, p);  
C     ps.executeUpdate();  
C     out.println("<h2>User Registered</h2>");  
C }  
C catch (Exception k)  
C {  
C     out.println("<h2>User exist with this name<del></h2>  
C             </h2>");  
C }  
C }  
C }
```

signup.html

```
C <html>  
C <body bgcolor="cyan">  
C <form action="./signup">  
C <h2>  
C     Name <input type="text" name="t1"> <br>  
C     UserName <input type="text" name="t2"> <br>  
C     Password <input type="password" name="t3"> <br>  
C     <input type="submit" value="sign UP">  
C </h2>  
C </form>  
C </body>  
C </html>
```

Interservlet communication :-

- servlets running together in the same server have several ways to communicate with each other. There are three major reasons to use interservlet communication.

Direct servlet manipulation :-

A servlet can gain access to the other currently loaded servlets and perform some task on each.

servlet reuse :-

One servlet can use another's abilities to perform task.

Servlet collaboration :-

The most common situation involves two or more servlets sharing state information.

RequestDispatcher :-

- RequestDispatcher is an interface.
- This interface is implemented by servlet container.
- Defines an object that receives requests from the client and sends them to any resource (such as servlet, HTML file or JSP file) on the server.
- The servlet container creates the RequestDispatcher object, which is used as a wrapper around a server resource located at a ~~at~~ particular path or given by a ~~a~~ particular name.
- A RequestDispatcher object can forward a client's request to a resource or include the resource itself in the response back.

to the client. A resource can be another servlet, or an HTML file or a JSP file, etc.

How to get RequestDispatcher object?

1. `ServletRequest.getRequestDispatcher(url)`
2. `ServletContext.getRequestDispatcher(url)`

25.11.15
Q. what is the difference between the `getRequestDispatcher(String path)` method of `ServletRequest` interface and `ServletContext` interface?

Ans:- The `getRequestDispatcher(String path)` method of `javax.servlet.ServletRequest` interface accepts parameter the path of the resource to be included of forwarded to, which can be relative to the request of the calling servlet. If the path begins with a “/” it is interpreted as relative to the current context root.

→ `ServletContext.getRequestDispatcher(String path)`

The `getRequestDispatcher(String path)` method of `javax.servlet.ServletContext` interface can't accept relative paths. All path must start with a “/” and are interpreted as relative to current context root.

→ Methods of RequestDispatcher :-

→ RequestDispatcher provide 2 Methods:

1. forward Method.
2. include Method.

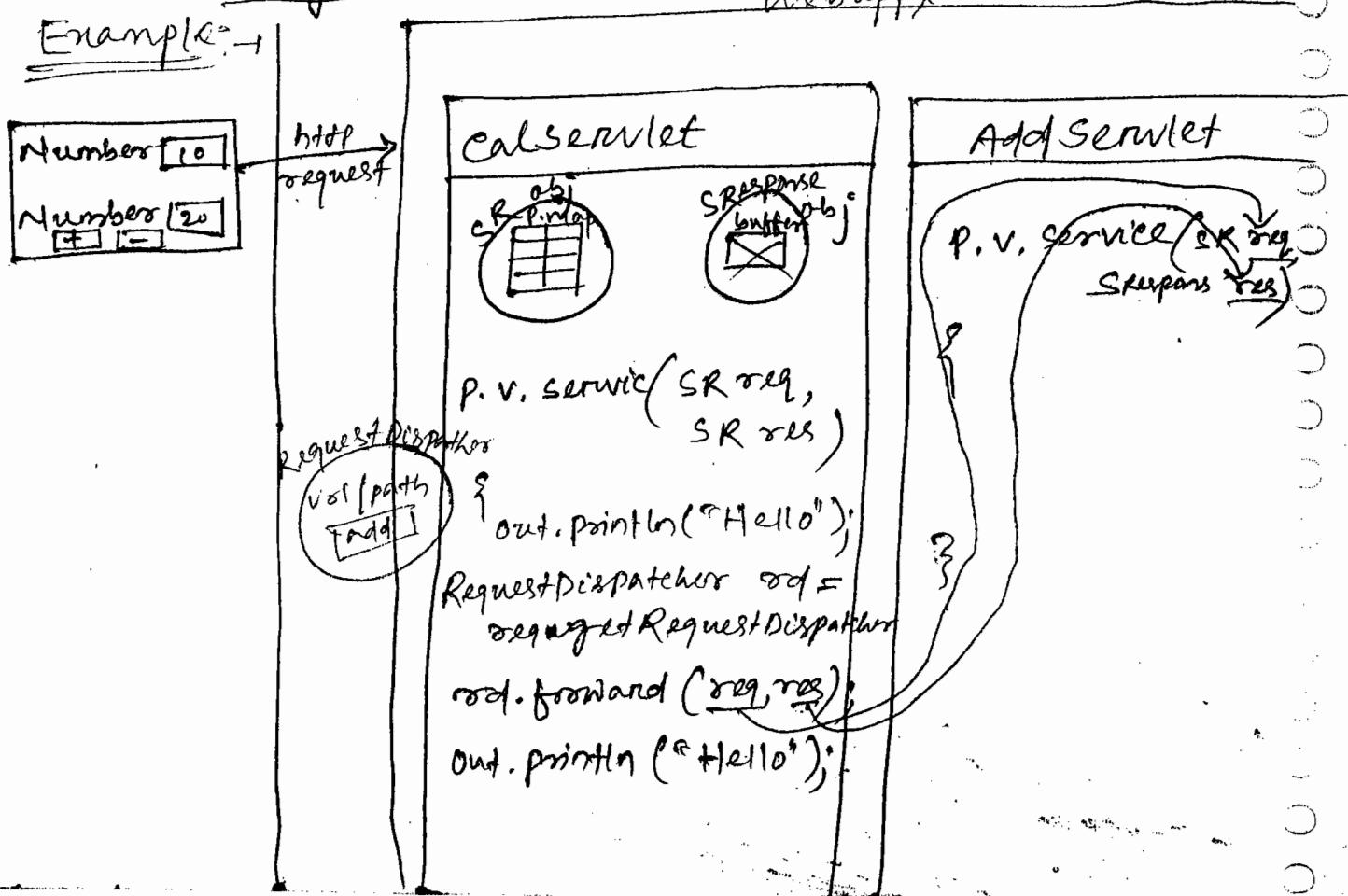
1. `public void forward(ServletRequest req, ServletResponse res)`

→ forward Method forwards a request from a servlet to another resource (Servlet, JSP File, or HTML File)

on the server. This method allows one servlet to do preliminary processing of a request and another resource to generate the response.

- For a RequestDispatcher obtained via `getRequestDispatcher()`, the ServletRequest object has its path elements and parameters adjusted to match the path of the target resource.
- forward Method should be called before the response has been committed to the client (before response body output has been flushed). If the response already has been committed, this method throws an IllegalStateException.

Example:-



Example of forward Method of Request Dispatcher.

home.html :-

```
<html>
  <body bgcolor="cyan">
    <form action="calc"/>
      <h2>
        Number<input type="text" name="t1"><br>
        Number<input type="text" name="t2"><br>
        <input type="submit" value="add" name="b">
        <input type="submit" value="sub" name="b">
      </h2>
    </form> </body> </html>
```

Servlet :-

```
Package com.nit.servlets
import java.io.IOException
@webServlet("calc")
public class calcServlet extends GenericServlet {
  public void service(ServletRequest req, ServletResponse res) throws ServletException, IOException
  {
    String b = req.getParameter("b");
    if (b.equals("add"))
    {
      RequestDispatcher rd1 = req.getRequestDispatcher("add");
      rd1.forward(req, res);
    }
    else
      if (b.equals("sub"))
    {
      RequestDispatcher rd2 = req.getRequestDispatcher("sub");
      rd2.forward(req, res);
    }
  }
}
```

AddServlet :-

```
package com.nit.servlets
import java.io.IOException
@WebServlet("/add")
public class AddServlet extends GenericServlet {
    public void service(ServletRequest req, ServletResponse res) throws
        ServletException, IOException
    {
        int n1 = Integer.parseInt(req.getParameter("t1"));
        int n2 = Integer.parseInt(req.getParameter("t2"));
        int n3 = n1 + n2;
        PrintWriter out = res.getWriter();
        out.println("<h2> Sum is " + n3 + "</h2>");
    }
}
```

SubServlet :-

```
package com.nit.servlets
import java.io.IOException
@WebServlet("/sub")
public class SubServlet extends GenericServlet {
    public void service(ServletRequest req, ServletResponse res) throws
        ServletException, IOException
    {
        int n1 = Integer.parseInt(req.getParameter("t1"));
        int n2 = Integer.parseInt(req.getParameter("t2"));
        int n3 = n1 - n2;
        PrintWriter out = res.getWriter();
    }
}
```

include Method of RequestDispatcher :-

```
public void include(ServletRequest req,  
                    ServletResponse res)
```

- Includes the content of a resource (Servlet, JSP page, HTML file) in the response. In essence, this method enables programmatic server-side includes.
- The ~~ServletRequest~~ object has its path element and parameters remain unchanged from the callers.
- The ~~ServletResponse~~ object has its path element and parameters remain unchanged from the callers.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

26.11.15

Include Method (Request Dispatcher)

Example

home.html

username nit
password nit123

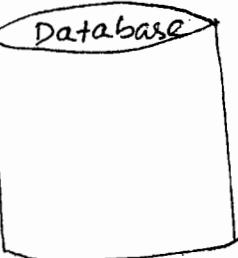
signin

valid

invalid

Sign Servlet

home.html



include

Hello nit
welcome to
my website

home.html

username
password
signin

Invalid Username
or password

home.html

```
<html>
<body bgcolor="cyan">
<form action="/sign">
<h2>
  Username <input type="text" name="t1"> <br>
  Password <input type="text" name="t2"> <br>
  <input type="submit" value="Signin"> <br>
</h2>
</form>
</body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SigninServlet :-

```
package com.mt.servlets  
import java.sql.*;  
@WebServlet("/")  
public class SigninServlet extends GenericServlet  
{  
    private Connection cn;  
    public void init()  
    {  
        try  
        {  
            Class.forName("oracle.jdbc.driver.OracleDriver");  
            cn = DM.getConnection("jdbc:oracle:thin:@localhost:  
                1521:XE", "system", "anjan");  
        }  
        catch (Exception k)  
        {  
            k.printStackTrace();  
        }  
    }  
    public void service(ServletRequest req,  
                        ServletResponse res)  
        throws ServletException, IOException  
    {  
        String user = req.getParameter("t1");  
        String pwd = req.getParameter("t2");  
        PrintWriter out = res.getWriter();  
        try{  
            PreparedStatement ps = cn.prepareStatement  
                ("select count(*) from user_reg where  
                    uname=? and pwd=?");  
        }
```

```

C     ps.setString(1, user);
C     ps.setString(2, pwd);
C     ResultSet rs = ps.executeQuery();
C     rs.next();
C     int c = rs.getInt(1);
C     if(c == 0)
C     {
C         RequestDispatcher rd = req.getRequestDispatcher("home.html");
C         rd.include(req, res);
C         out.println("<h2> Invalid username or password </h2>");
C     }
C     else
C     {
C         out.println("<h2> ");
C         out.println("Hello " + user + " welcome to my website" );
C         out.println("</h2> ");
C     }
C }
C catch (Exception e)
C {
C     e.printStackTrace();
C }
C

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Attributes :-

- Attributes are the objects which can be set, get or remove during runtime.
 - Each attribute is having scope and Lifetime.
 - Webapplication is having 3 scopes
 - 1. request scope.
 - 2. application scope/context scope
 - 3. session scope
- (scope is nothing but border or boundary)

1. request scope:-

- attributes added with in request scope or request exist until execution of service method.
- When request object destroyed, if remove the objects/attributes exist in request.

Methods of ServletRequest to manipulate attributes

1. `Java.lang.Object getAttribute
(Java.lang.String name)`

Returns the value of the named attribute as an object, or null if no attribute of the given name exist.

2. `void removeAttribute
(Java.lang.String name)`

Removes an attribute from this request.

3. `void setAttribute(Java.lang.String name,
Java.lang.Object o)`

Stores an attribute in this request.

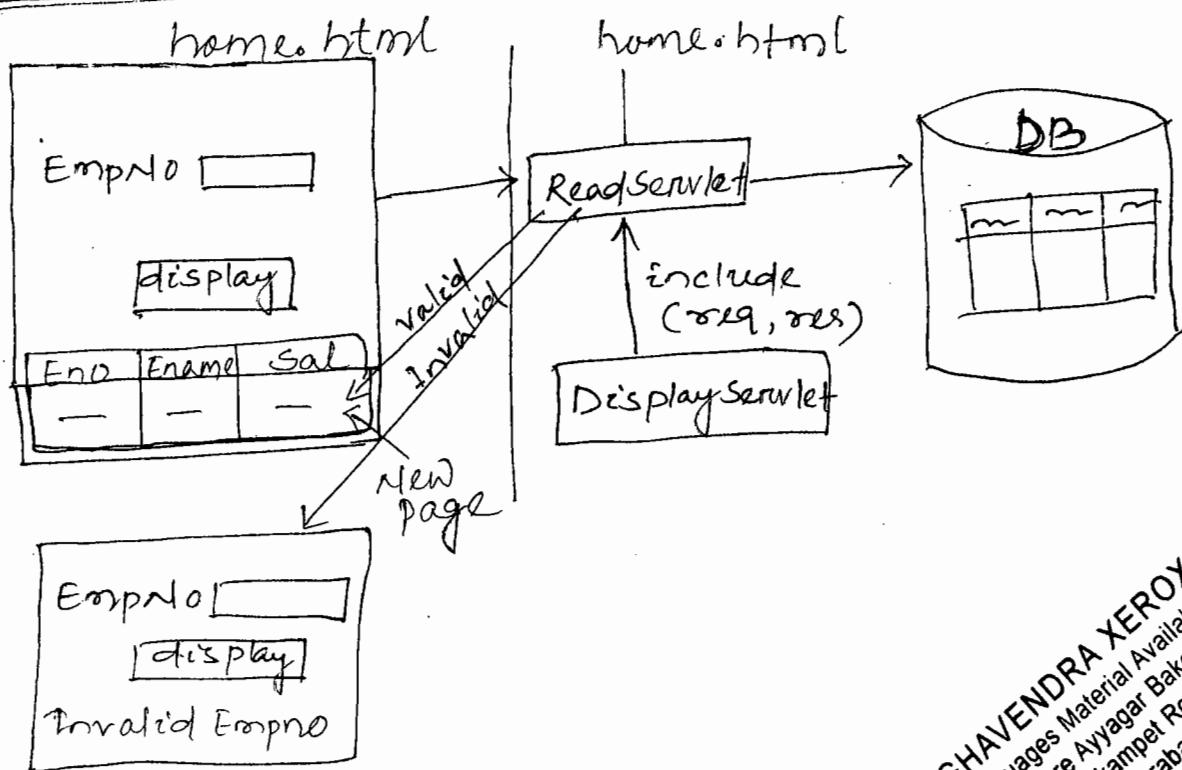
Q. What is the difference between parameter and attribute?

Parameter	Attributes
(i) parameters are strings. (ii) constants which can't change during runtime.	(i) Attributes are object type. (ii) can be changed during runtime.

Q. what is the difference between RequestDispatcher forward and includeMethod.

Included() Method	forward() Method()
(i) The RequestDispatcher included() method inserts the contents of the specified resource directly in the flow of the servlet response as if it were part of the calling servlet. (ii) If you include a servlet or jsp document the included resource must not attempt to change the response status code or HTTP headers any such request will be ignored. (iii) The included() method is often used to include common "boilerplate" text or template markup that may be included by many Servlets	(i) The RequestDispatcher forward() method is used to show a different resource in place of the servlet that was originally called. (ii) The forwarded resource may be another servlet , jsp or static HTML document but the response is issued under the same URL that was originally requested. In other words, it is not the same as a redirection. (iii) The forward() method is often used where a servlet is taking a controller role processing some input and deciding the outcome by returning a particular response page

27.11.15



HOME.html :-

```

<html>
<body bgcolor = cyan>
<form action = ". />read" >
<h2>
< EmpNo <input type = "text" name = "t1" ><br>
    <input type = "submit" value = "display" ><br>
</h2>
</form>
</body>
</html>

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Develop Servlet :-

ReadServlet

```
package com.neto.servlets;
import javax.servlet.*;
import java.sql.*;
import java.io.*;

public class ReadServlet extends GenericServlet
{
    private Connection cn;
    public void init()
    {
        try
        {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            cn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:XE", "system", "rajan");
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }

    public void service(ServletRequest req,
                        ServletResponse res)
    throws ServletException, IOException
    {
        int empno = Integer.parseInt(req.getParameter("t"));
        PrintWriter out = res.getWriter();
        PreparedStatement ps = cn.prepareStatement("select eno, ename, sal from employe where eno=?");
    }
}
```

```

ps.setInt(1, empno);
ResultSet res = ps.executeQuery();
boolean b = rs.next();
RequestDispatcher rd = req.getDispatcher("home.html");
rd.include(req, res);
if (b == true)
    RequestDispatcher rd1 = req.getRequestDispatcher("display");
    req.setAttribute("eno", rs.getInt(1));
    req.setAttribute("ename", rs.getString(2));
    req.setAttribute("sal", rs.getFloat(3));
    rd1.include(req, res);
}
else
{
    out.println("Invalid empno</h2>");
}
catch (Exception k)
{
    k.printStackTrace();
}
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

DisplayServlet

```
package com.net.servlets
import java.io.*;
import javax.servlet.*;
public class DisplayServlet extends GenericServlet
{
    public void initservice(ServletRequest req, ServletResponse res) throws
        ServletException, IOException
    {
        PrintWriter out = res.getWriter();
        out.println("<h2>");
        out.println("<table>");
        out.println("<tr>");
        out.println("<td> Empno </td>");
        out.println("<td> Ename </td>");
        out.println("<td> Salary </td>");
        out.println("</tr>");
        int eno = (Integer)req.getAttribute("eno");
        String en = (String)req.getAttribute("en");
        float sal = (Float)req.getAttribute("sal");
        out.println("<tr>");
        out.println("<td>" + eno + "</td>");
        out.println("<td>" + en + "</td>");
        out.println("<td>" + sal + "</td>");
        out.println("</tr>");
        out.println("</table>");
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

2. context scope

Servlet context provide two methods to get RequestDispatcher object.

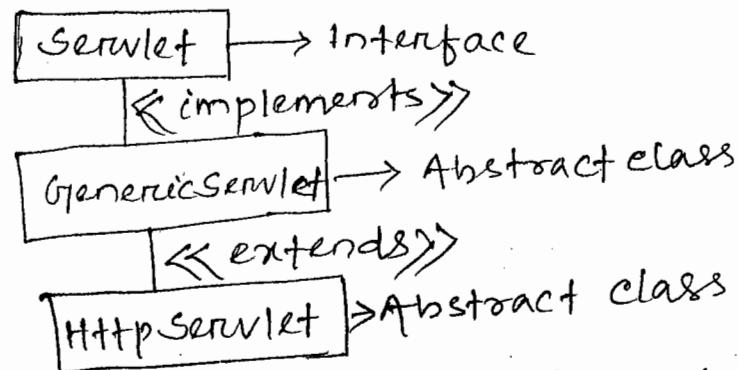
1. getRequestDispatcher.
2. getNamedDispatcher.

getRequestDispatcher	getNamedDispatcher
→ Returns a RequestDispatcher object that acts as a wrapper for the resource (static/dynamic) located at the given path.	→ Returns a RequestDispatcher object that acts as a wrapper for the named servlet.

SRI RAJENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Http Servlet

- HttpServlet is an abstract class.
- ① javax.servlet → GenericServlet
- ② javax.servlet.http → HttpServlet



- Provides an abstract class to be subclassed to create an HTTP servlet suitable for a website.
- HttpServlet is a protocol dependent servlet.
- HttpServlet supports 2 protocols

1. Http
2. Https

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Advantage of Http Servlet :-

1. It provides various request methods.
 2. It provides methods for modifying request and response headers.
 3. It supports cookies.
 4. It supports HttpSession.
- Q. What is difference between GenericServlet and HttpServlet?

GenericServlet

- The GenericServlet is an abstract class that is extended by HttpServlet to provide HTTP protocol-specific methods.

- An abstract class that simplifies writing HttpServlets. It extends the GenericServlet base class and provides a framework for handling the HTTP protocol.

- The GenericServlet does not include protocol-specified methods for handling request parameter, cookies, sessions and setting response handlers.
- GenericServlet is not specific to any protocol.
- The HttpServlet subclass passes generic service method requests to the relevant doGet() or doPost() method.
- HttpServlet only supports Http protocol and Https protocol.

Request methods of HTTP protocol :-

- HTTP protocol supports 7 request methods

1. GET
2. POST
3. PUT
4. HEAD
5. DELETE
6. OPTIONS
7. TRACE

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Q. what is the difference between request methods GET and ~~post~~ POST? (OR) what is difference between doGet and doPost methods?

<u>doGet</u>	<u>doPost</u>
(1.) In doGet() the parameters are appended to the URL and sent along with header information.	(1.) In doPost() on the other hand will (typically) send the information through a socket back to the Webserver and it won't show up in the URL bar.
(2.) The amount of information you can send back using a GET is restricted as URLs can can only be 1024 characters.	(2.) You can send much more information to the server this way and its not restricted to

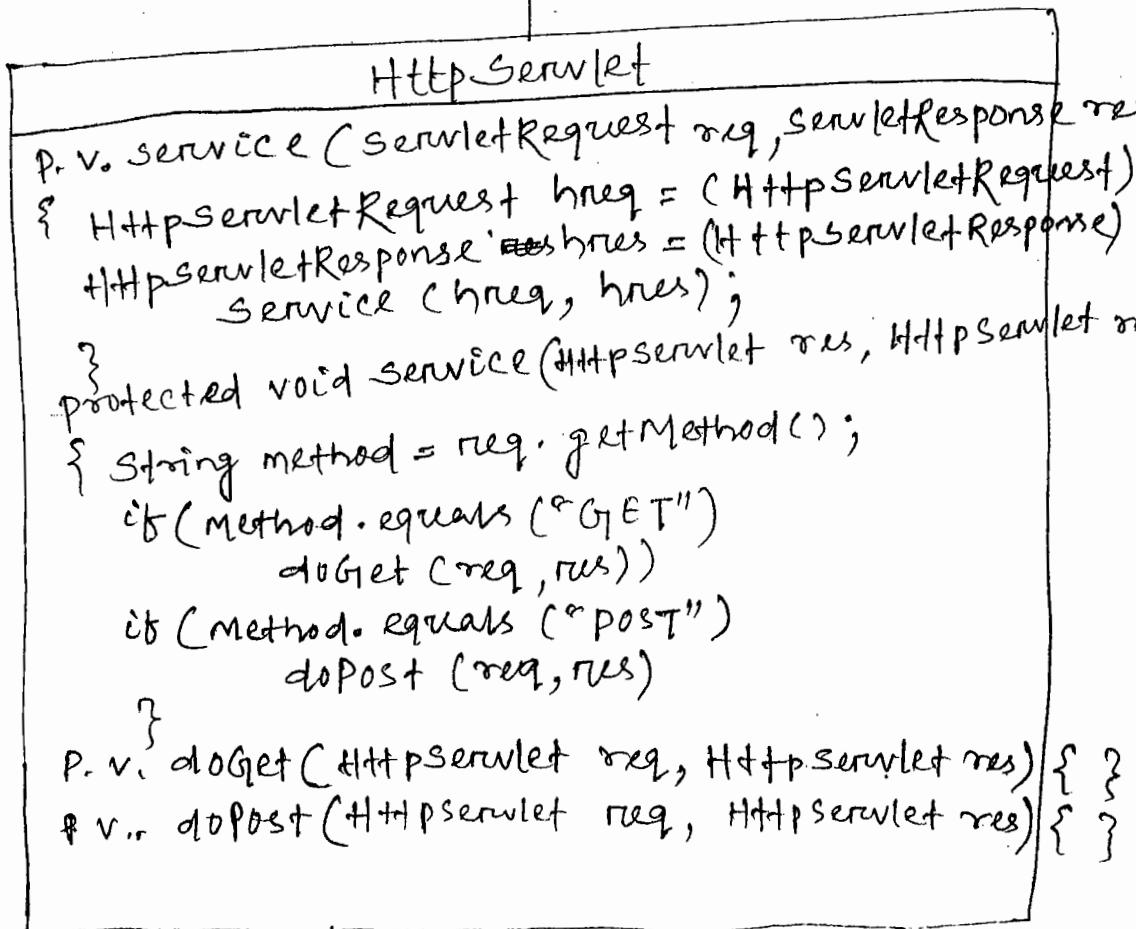
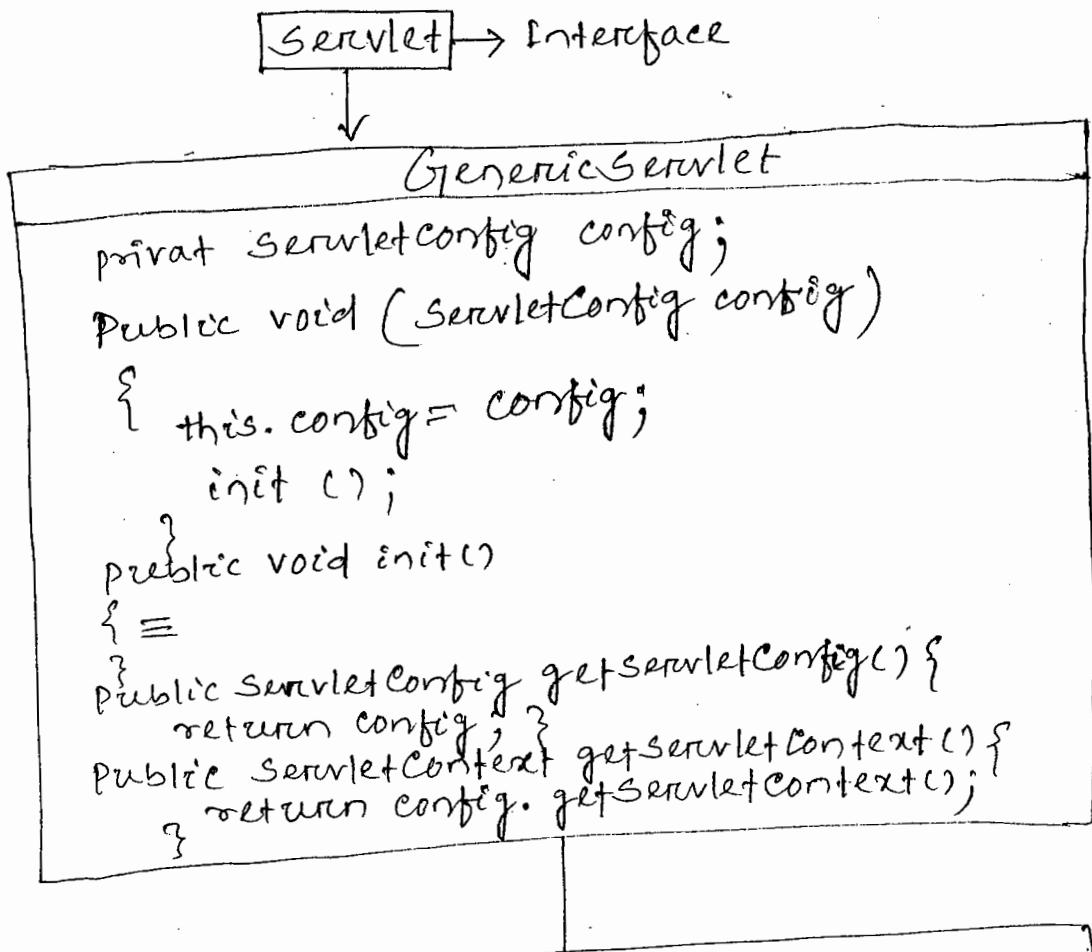
- (3) `doGet()` is a request for information, it does not (or should not) change anything on the server. (~~doGet()~~) (`doGet()` should be idempotent).
 - (4) parameters are not encrypted.
 - (5) `doGet()` is faster, if we sets the response context length since the same connection is used. Thus increasing the performance.
 - (6) `doGet()` should be idempotent i.e. `doGet()` should be able to be repeated safely many time.
 - (7) `doGet()` should be safe without any side effect for which user is held responsible
 - (8) `get` allows bookmarks.
- textual data either it is possible to send files and even binary data such as serialized Java objects
- (3) `doPost()` provides information (such as placing an order for merchandise) that the server is expected to remember.
 - (4) Parameters are encrypted.
 - (5) `doPost()` is used for update or post ~~so~~ some information to the server.
 - (6) `doPost()` is slower as compare to `doGet()` since `doPost()` does not write the context length.
 - (7) This method does not need to be idempotent. operations requested through POST can be have side effects for which the user can be held accountable.
 - (8) This Methods does not allow bookmarks
 - (9) `doPost()` does not need to be either safe.

Q. When to use `doGet()` & `doPost()` Method?

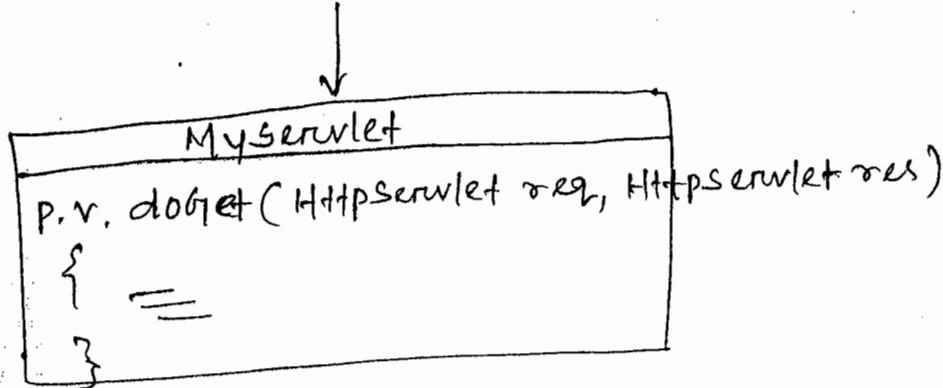
Ans:-

- ① If data is sensitive
- ② If data is greater than 1024 character
- ③ If our application don't need bookmarks then we go for `doPost()` method otherwise go for `doGet()` methods.

Structure of HttpServlet



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.



10/12/2015 :-

How to read HTTP request headers?

→ HttpServletRequest provide the following methods to manipulate request headers.

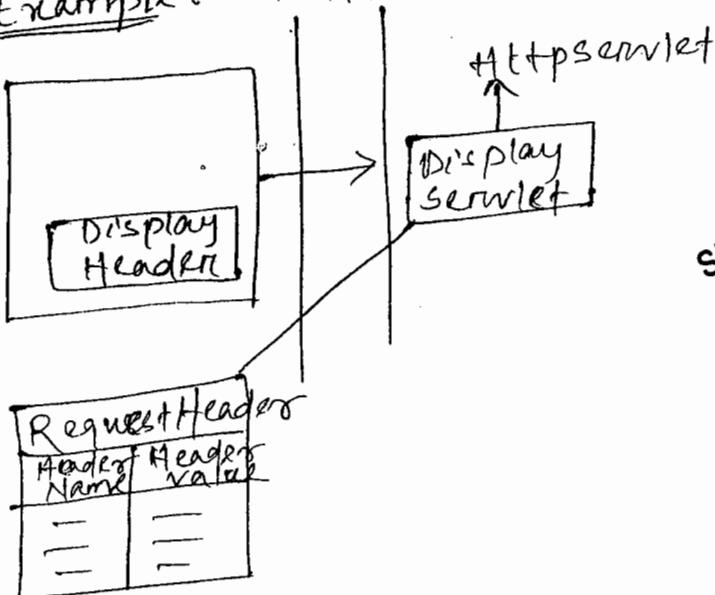
① String getHeader (String name)

Returns the value of the specified request header as a string.

② Enumeration<String> getHeaderNames()

Returns an enumeration of all the header names this request contains.

Example :- Q. Application to read request headers



SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Program :-

home.html :-

```

<html>
<body bgcolor="cyan">
<form action = "/display">
<input type="submit" value="display"/>
</form>
</body>
</html>

```

Serlet

```

package com.nit.serlets; import java.util.*;
import java.io.IOException;
@WebServlet("/display") extends HttpServlet{
public class Displayserlet extends HttpServlet{
protected void doGet(HttpServletRequest req,
HttpServletResponse res)
throws ServletException, IOException
{
PrintWriter out = res.getWriter();
Enumeration<String> e = req.getHeaderNames();
while(e.hasMoreElements())
{
String name = e.nextElement();
String value = req.getHeader(name);
out.println("<h2>");
out.println(name + "=" + value);
out.println("</h2>");
}
}
}

```

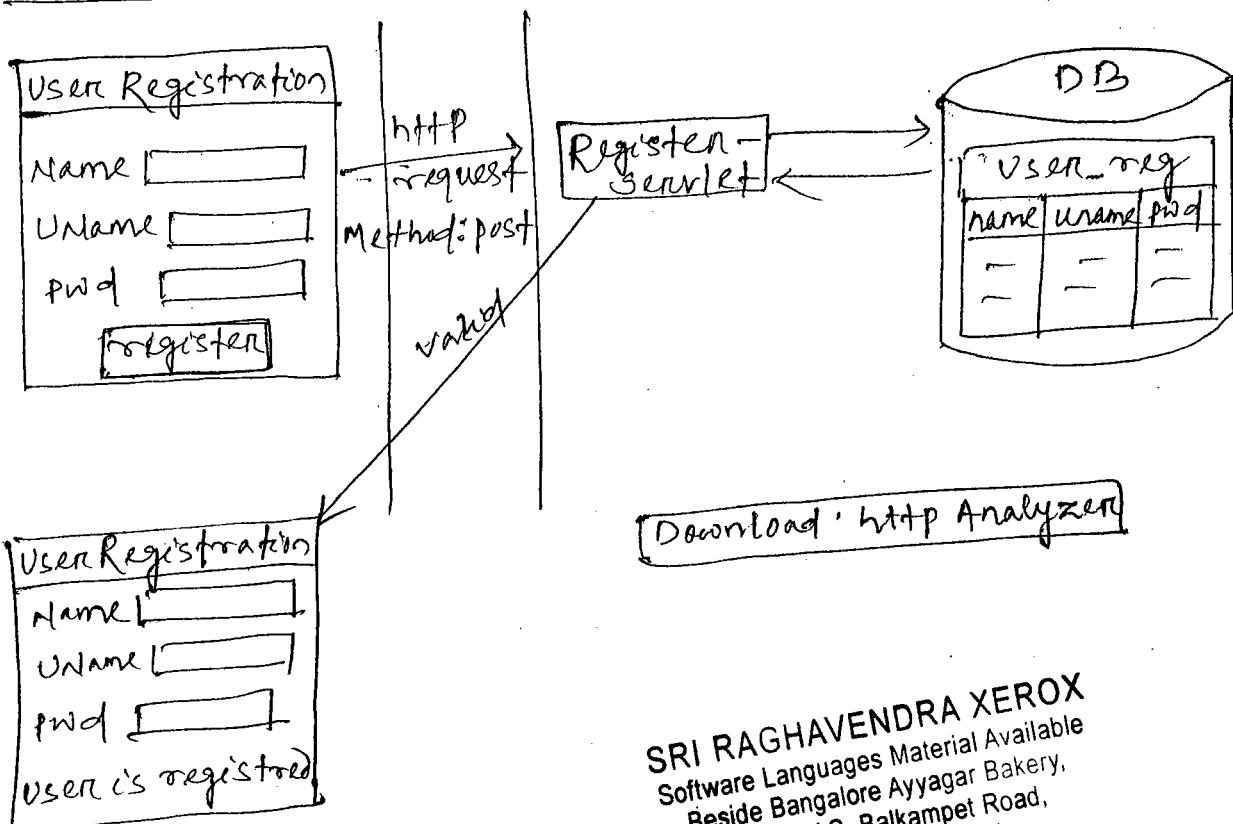
SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Opp. CDAC, Ayyagar Bakery,
 Ameerpet, Hyderabad.

Timeservlet

```
package com.nit.servlets;
import java.util.*;
@WebServlet ("^/time")
public class Timeservlet extends HttpServlet {
    protected void doGet (HttpServletRequest req,
                         HttpServletResponse res)
        throws ServletException, IOException
    {
        Date d = new Date();
        int h = d.getHours();
        int m = d.getMinutes();
int s = d.getSeconds();
        int s = d.getSeconds();
        response.setContentType("text/html");
        response.setHeader("refresh", "1");
// response.setStatus(200);
        PrintWriter out = res.getWriter();
        out.println("<h2>");
        out.println(h+":"+m+":"+s);
        out.println("</h2>");
    }
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Post() Method Example :-



[Download 'http Analyzer']

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

home.html

```
<html>
<body background = cyan>
<form action = ". /register" method = "POST">
<h2>
    Name <input type = "text" value name = "t1"><br>
    UserName <input type = "text" name = "t2"><br>
    Password <input type = "text" name = "t3"><br>
    <input type = "Submit" value = "register">
</h2>
</form>
</body>
</html>
```

Develop servlet

```
package com.net.servlets;
import java.io.IOException;
import java.sql.*;
@WebServlet("/register")
public class RegisterServlet extends HttpServlet
{
    private Connection cn;
    public void init()
    {
        try
        {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            cn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:XE", "system", "anjan");
        }
        catch(Exception k)
        {
            k.printStackTrace();
        }
    }
    protected void doPost(HttpServletRequest req,
                         HttpServletResponse res) throws ServletException,
                         IOException
    {
        String t1 = req.getParameter("t1");
        String t2 = req.getParameter("t2");
        String t3 = req.getParameter("t3");
        res.setContentType("text/html");
        PrintWriter out = res.getWriter();
    }
}
```

SRI RAJAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
try {
    PreparedStatement ps = cn.prepareStatement(
        "insert into user-reg values (?, ?, ?)");
    ps.setString(1, n);
    ps.setString(2, u);
    ps.setString(3, p);
    RequestDispatcher rd = req.getRequestDispatcher("home.html");
    rd.include(req, res);
    ps.executeUpdate();
    out.println("<h2>User Registered</h2>");
}
```

catch (Exception k)

```
{ out.println("<h2>Error in registering user</h2>"); }
```

}

Q. How do I support both GET and POST from
the same servlet?

→ doGet method call your doPost Method.

```
public void doGet(HttpServletRequest req,
                    HttpServletResponse res)
    throws ServletException, IOException.
```

```
{ doPost(req, res); }
```

doGet :-

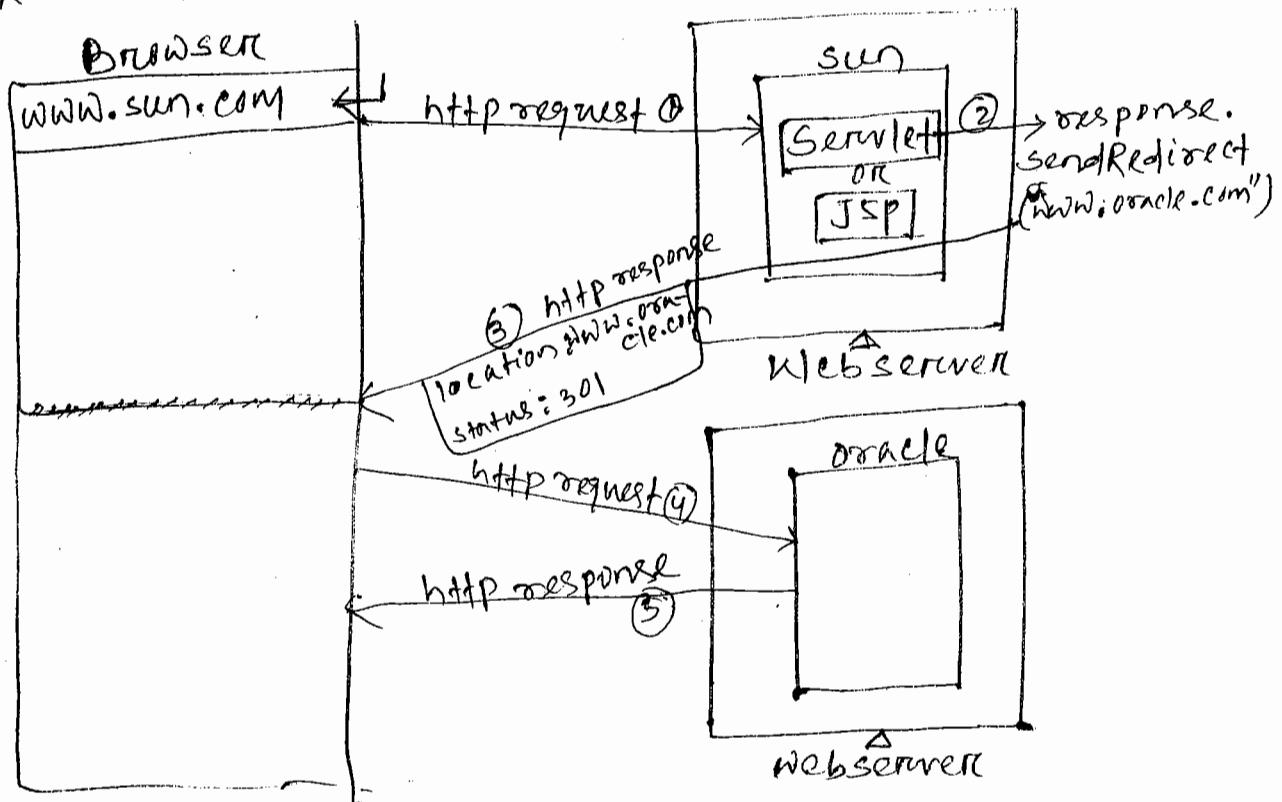
- overriding this method to support a GET request also automatically supports an HTTP HEAD request. A HEAD request is an GET request that returns no body in the response, only the request header fields.
- When overriding this method, read the request data, write the response headers, get the response's writer or output stream object, and finally, write the response data.

doPost :-

- when overriding this method, read the request data, write the response headers, get the response's writer or output stream object, and finally, write the response data.

SendRedirect :-

- SendRedirect is a method of HttpServletRespose.
- Sends the temporary redirect response to the client using the specified redirect location URL and clears the buffer.



Q. what is the difference between forward and sendRedirect methods?

forward() Method

- When we use forward method request is transfer to other resource within the same server for further processing.
- In case of forward) web container handle all process internally and client or browser is not involved.
- When forward is called on RequestDispatcher obj. we pass request and response object so our old request object is present on new resource which is going to process our request.
- Visually we are not able to see the forwarded address, it's transparent.
- Using forward method is faster than send redirect.
- When we redirect using forward and we want to use same data in new resource we can use request.setAttributes() as we have request object available.

sendRedirect() Method

- In case of sendRedirect request is transfer to another resource to different domain or diff. server for further processing.
- When you use Redirect container transfer the request to client or browser so url given inside the sendRedirect method is visible as a new request on new resource to the client.
- In case of sendRedirect call old request and response object is lost because it's treated as new request by the browser.
- In address bar we are able to see the new redirected address it's not transparent.
- sendRedirect is slower bcz one extra round trip is required bcz completely new request is created and old request object is lost. Two browser request required.

→ But in sendRedirect if we want to use we have to store the data in Session and pass along with the URL.

Example :-

Sun Application

```
package com.net.servlets;
import java.io.IOException;
@.WebServlet("home")
public class HomeServlet extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.sendRedirect("http://localhost:8085/
            oracleapplication/home.html");
    }
}
```

oracle application

Add a home.html page

```
<html>
<body bgcolor=yellow>
<font color=blue size=7>
    oracle Site
</font>
</body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Material Bakery
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

03.12.15 :-

Example

Browser

Number	10
Number	20
<input type="button" value="add"/>	

①

Webapplicationx

calcServlet

②

String p₁ = req.get
Parameter("t₁");

String p₂ = req.getParame
ter("t₂");

String url = "http://localhost:
8086/mathapplication/add?
t₁=p₁&t₂=p₂";
response.sendRedirect
(url);

MathApplication

AddServlet

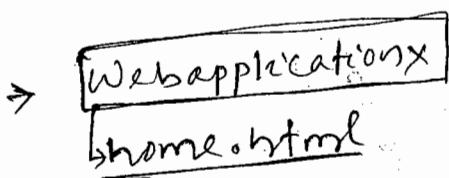
③

④

Sum is
30

⑤

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.



```

<html>
<body bgcolor="cyan">
<form action = ". / calc" >
Number <input type = "text" name = "t1"> <br>
Number <input type = "text" name = "t2"> <br>
<input type = "submit" value = "add" >
</form>
</body>
</html>

```

Develop calcServlet

```

package com.mit.Servlets;
import java.io.IOException;
@webServlet("/")
public class calcServlet extends HttpServlet {
protected void doGet(HttpServletRequest request,
                      HttpServletResponse response)
throws ServletException, IOException

```

```

    String p1 = request.getParameter("t1");
    String p2 = request.getParameter("t2");
    String url = "http://localhost:8086/Mathapplication";
    add? p1 = " + p1 + " & p2 = " + p2;
    response.sendRedirect(url);
}
}

```

`response.setHeader("location", url); // without`
`response.setStatus(301); // using Redirect;`

~~`response.setHeader("location", url);`~~
~~`response.setStatus(301);`~~

Mathapplication

AddServlet

```

package com.rnit.servlets;
import java.io.IOException;
@WebServlet("/add")
public class AddServlet extends HttpServlet {
protected void doGet(HttpServletRequest request,
HttpServletResponse response)
throws ServletException, IOException
{

```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

int n1 = Integer.parseInt(request.getParameter("P1"));
int n2 = Integer.parseInt(request.getParameter("P2"));
int n3 = n1 + n2;
PrintWriter out = response.getWriter();
out.println("<html>");
out.println("<body bgcolor=yellow>");
out.println("<h2> sum is " + n3 + "</h2>");
out.println("</body></html>");
}
}

```

V.V.V.Imp

Session tracking or session Management or State Management :-

Q. what is session Tracking?

- Session tracking is the capability of a server to maintain the current state of a single client's sequential requests.
- The HTTP protocol used by web servers is stateless.
- Session simply means a particular interval of time.
- Session tracking is a way to maintain state of user.
- Http is stateless protocol. Each time user send requests to server, server treats the request as the new request.
- Session tracking is used to recognize the user by server.

→ There are 4 techniques used for session tracking

1. hidden form fields.
2. URL rewriting
3. cookies
4. HttpSession

SRI RAHUVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

① Hidden form Fields

- The Server embeds new hidden fields in every dynamically generated form page for the client. When the client submits the form to the server the hidden fields identify the client.

Syntax for hidden field :-

<input type="hidden" name="name" value="value">

Disadvantages

Advantages :-

1. simple
2. No effect on security level setting in browsers.

Disadvantages :-

1. The documents needs to embed the data inside, waste bandwidth. you have to embed the result from previous pages in the next page.
2. Everyone can see the embedded data by viewing the original source code.
3. If the user surfs to a different site, or to a static section of the same site, the state is lost.

Example :-

home.html

<html>

<body background=cyan>

<form action="r/signin">

<h2>

Username <input type="text" name="t1">

Password <input type="password" name="t2">

<input type="submit" Value="signin">

</h2>

</form>

</body>

</html>

SRI RAGHAVENDRA XEROX

Software Languages Material Available

Beside Bangalore Ayyagar Bakery,

Opp. CDAC, Balkampet Road,

Ameerpet, Hyderabad.

Servlet

signinServlet

```
package com.cito.servlets;
import java.io.IOException
@WebServlet (" signin")
public class SigninServlet extends HttpServlet
{
protected void doGet (HttpServletRequest req,
HttpServletResponse res)
throws ServletException, IOException
{
String user = req.getParameter ("t1");
String pwd = req.getParameter ("t2");
res.setContentType ("text/html");
res.
PrintWriter out = res.getWriter();
if (user.equals ("nit") && pwd.equals ("nit123"))
{
out.println ("");
out.println ("
out.println ("<input type= submit value = "inbody" >");  

out.println ("</form></body></html>");  

}
else
out.println ("<h2> Invalid username or  

password </h2>");  

}
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

InboxServlet

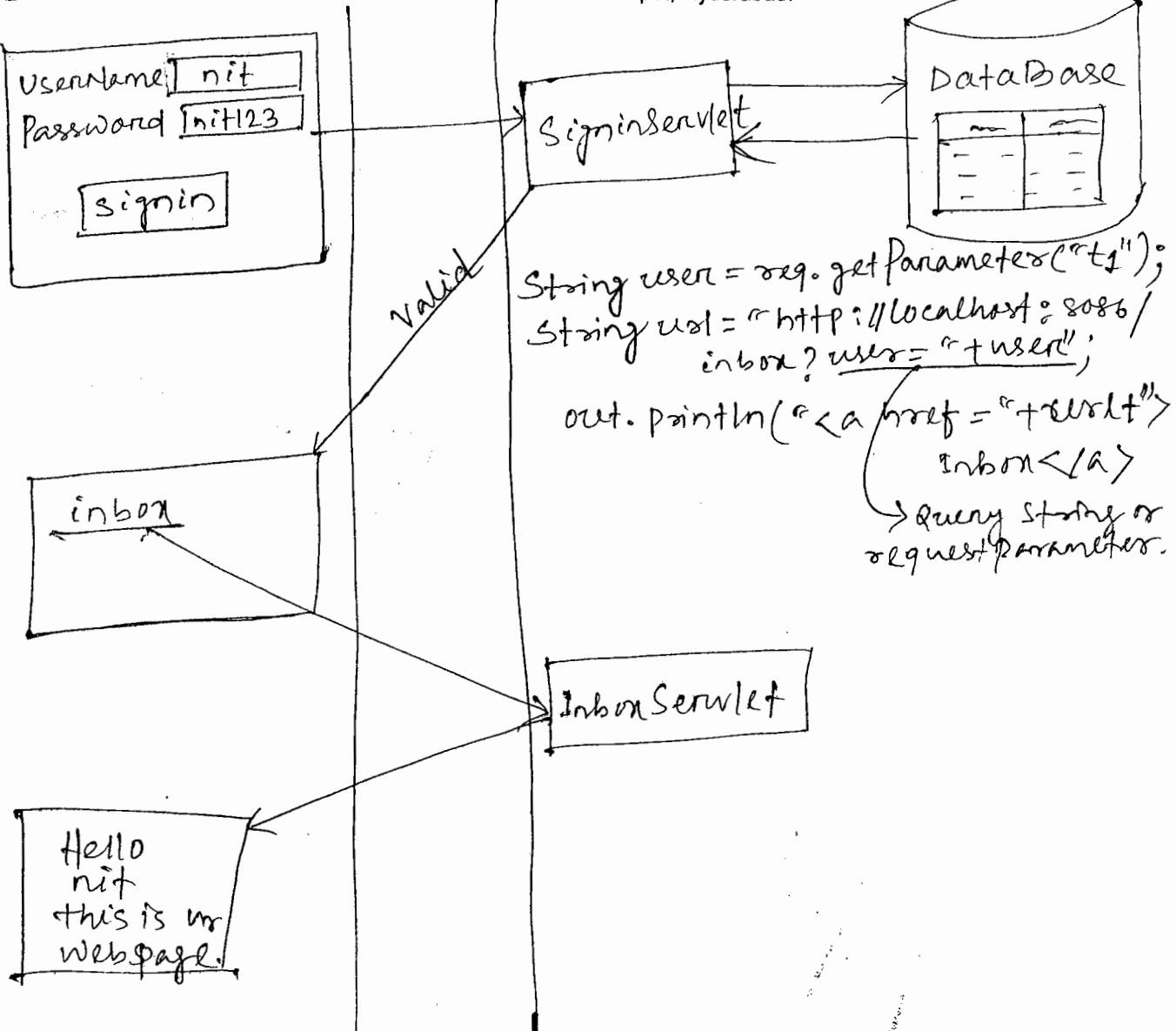
```
package com.nit.servlets;
import java.io.IOException;
@WebServlet("/inbox")
public class InboxServlet extends HttpServlet {
protected void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException
{
    String user = req.getParameter("t1");
    PrintWriter out = res.getWriter();
    out.println("<h2>Hello "+user+" this is your
    inbox</h2>");
}}
```

04.12.15.

② URL rewriting:-

SRI RAGHAVENDRA XEROX

Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.



URL Rewriting :-

- In URL Rewriting, we append a token or identifier to the URL of the next servlet or the next resource we can send parameter name/value pairs using the following format.
- url?name1 = value1 & name2 = value2
 ↑ QueryString or Request Parameter.

Advantages :-

1. Every data is appended on the URL \Rightarrow easy to debug.

Disadvantages :-

1. URL is lengthy and the length of the URL is limited, can't store much information.
2. The URL contain data and this data is visible on URL path which is not secure.
3. If the user surfs to a different site, or to a static section of the same site, the state is lost.

Program :-

```
<html>
<body bgcolor="cyan">
<form action=".//signin">
    <h2>
        Username <input type="text" name="t1"> <br>
        Password <input type="password" name="t2"> <br>
        <input type="submit" value="signin">
    </h2>
    </form>
</body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Servlet :-

```
package com.nit.servlets  
import java.io.IOException;  
@WebServlet("/signin")  
public class SigninServlet extends HttpServlet {  
protected void doGet(HttpServletRequest req,  
HttpServletResponse res)  
throws ServletException, IOException  
{  
String user = req.getParameter("t1");  
String pwd = req.getParameter("t2");  
PrintWriter out = res.getWriter();  
if (user.equals("nit") && pwd.equals("nit"))  
{  
String url = "http://localhost:8086/webapplication/inbox?user=tuser";  
out.println("<h2>");  
out.println("<a href=" + url + ">Inbox</a>");  
out.println("</h2>");  
}  
else  
{  
out.println("<h2> Invalid username or  
password </h2>");  
}  
}  
}  
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

InboxServlet :-

```
package com.nit.servlets;
import java.io.IOException;
@WebServlet("/inbox")
public class InboxServlet extends HttpServlet {
protected void doGet(HttpServletRequest req, HttpServletResponse res)
throws ServletException, IOException {
String user = req.getParameter("user");
PrintWriter out = res.getWriter();
out.println("<h2>");
out.println("Hello " + user + " this is your inbox");
out.println("</h2>");
}
}
```

SRI RAGHAVENDRA XEROX

Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

③ Cookies :-

- ~~Cookies is a class.~~
- cookie is a class. and this class is exist in `javax.servlet.http` package.
- A cookie is small piece of information that persisted between the multiple client request.
- A cookie has a name, a single value and optional attributes such as comment, path and domain qualifiers, a maximum age and version number.
- cookie is persist with in client browser or machine.
- The browser is expected to support 20 cookies for each web server, 300 cookies total, and may limit cookie size to 4 KB each.

Advantages :-

1. simple
2. Don't need to send data back to client browser
can participate in this task.

Disadvantages :-

1. size and number of cookies stored are limited.
2. get stored as plain-text in a specific directory,
everyone can view and modify them.
3. get won't work if the security level set too
~~high~~ high in browser.

constructor of cookie class

`cookie (String name, String value)` // constructor
→ construct a cookie with a specified name and
value

commonly used methods of cookie class

1. `public String getName()`: returns the name
of the cookie. The name can't be changed
after creation.
2. `public String getValue()`: returns the
value of cookie.

other methods required for using cookies

1. `public void addCookie(cookie)`

it is a method of HttpServletResonse interface used
for adding cookie in response object.

2. `public cookie[] getCookies()`

it is a method of HttpServletRequest interface used
for reading all the cookies from browser.

Example:-

```

<html>           [home.html]
  <body background="cyan">
    <form action="http://localhost/login">
      <h2>
        username <input type="text" name="t1"> <br>
        password <input type="password" name="t2"> <br>
        <input type="submit" value="login">
      </h2>
    </form>
  </body>
</html>

```

Servlet:-

[Signinservlet]

```

package com.nit.servlets;
import java.io.IOException;
@.WebServlet(name="signin")
public class Signinservlet extends HttpServlet {
  protected void doGet(HttpServletRequest req,
                        HttpServletResponse res)
    throws ServletException, IOException
  {
    String user = req.getParameter("t1");
    String pwd = req.getParameter("t2");
    res.setContentType("text/html");
    PrintWriter out = res.getWriter();
    if (user.equals("nit") && pwd.equals("nit"))
    {
      cookie c = new cookie("uname", user);
    }
  }
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

```
res.addCookie(c);
out.println("<a href=http://localhost:8086/
webapplication22/inbox>Inbox</a>");
```

```
}
```

```
else
    out.println("<h2> Invalid username or pwd </h2>");
```

```
}
```

InboxServlet

```
package com.nit.servlets;
import java.io.IOException
@WebServlet("/inbox")
public class InboxServlet extends HttpServlet {
protected void doGet(HttpServletRequest req, HttpServletResponse res)
throws ServletException, IOException {
Cookie c[] = req.getCookies();
String user = c[0].getValue();
String d = c[0].getDomain();
PrintWriter out = res.getWriter();
out.println("<h2> Hello " + user + " this is your
inbox");
out.println("<br> domain " + d);
out.println("</h2>");
```

```
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Q. How to I set the maximum age of a cookie?

→ ~~COOKIE~~

void SetMaxAge(int expiry)

sets the maximum age in seconds for
this cookie.

- A positive value indicates that the cookie will expire after that many seconds have passed. Note that the value is the maximum age when the cookie will expire, not the cookie's current age.
- A negative value means that the cookie is not stored persistently and will be deleted when the web browser exits.
- A zero value causes the cookie to be deleted.

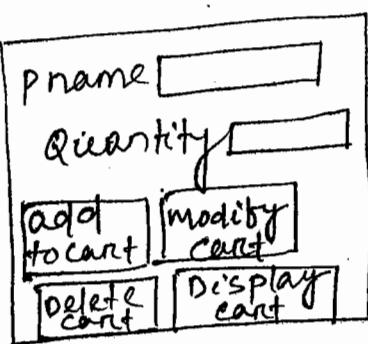
Q. How to I delete a cookie in servlet?

- The servlet API doesn't provide a direct way to delete a cookie in a servlet application. If you want to delete a cookie you have to create a cookie that have the same name with the cookie that you want to delete with the value to an empty string and set the value to set the max age of the cookie to zero(0).

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Rd.
Ameerpet, Hyderabad

Example

Browser



ProductServlet

home.html

```

<html>
  <body bgcolor="cyan">
    <form action=".//product">
      <h2>
        productName<input type="text" name="t1"><br>
        Quantity<input type="text" name="t2"><br>
        <input type="submit" value="Add product" name="b" />
        <input type="submit" value="Modify product" name="b" />
        <input type="submit" value="Delete product" name="b" />
        <input type="submit" value="Display products" name="b" />
      </h2>
    </form>
  </body>
</html>

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

ProductServlet

```
package com.nit.servlets  
import io.IOException.  
@WebServlet("/product")  
public class ProductServlet extends HttpServlet {  
protected void doGet(HttpServletRequest req,  
HttpServletResponse res)  
throws ServletException, IOException  
{  
String pname = req.getParameter("t1");  
String qty = req.getParameter("t2");  
String b = req.getParameter("b");  
RequestDispatcher rd = req.getRequestDispatcher("home.html");  
rd.include(req, res);  
PrintWriter out = res.getWriter();  
if (b.equals("Add Product"))  
{  
Cookie c1 = new cookie(pname, qty);  
res.addCookie(c1);  
out.println("product Added");  
}  
if (b.equals("Modify Product"))  
{  
Cookie c2 = new cookie(pname, qty);  
res.addCookie(c2);  
out.println("product Modified");  
}  
if (b.equals("Delete Product"))  
{  
}
```

```
COOKIE c3 = new COOKIE (pname, qty);
c3 = setMaxAge (c0);
res.addCOOKIE (c3);
out.println ("Product Deleted");
```

```
}  
else if (b.equals ("Display Product"))  
{  
    COOKIE c[] = req.getCOOKIES();  
    for (COOKIE c3 : c)  
    {  
        out.println ("

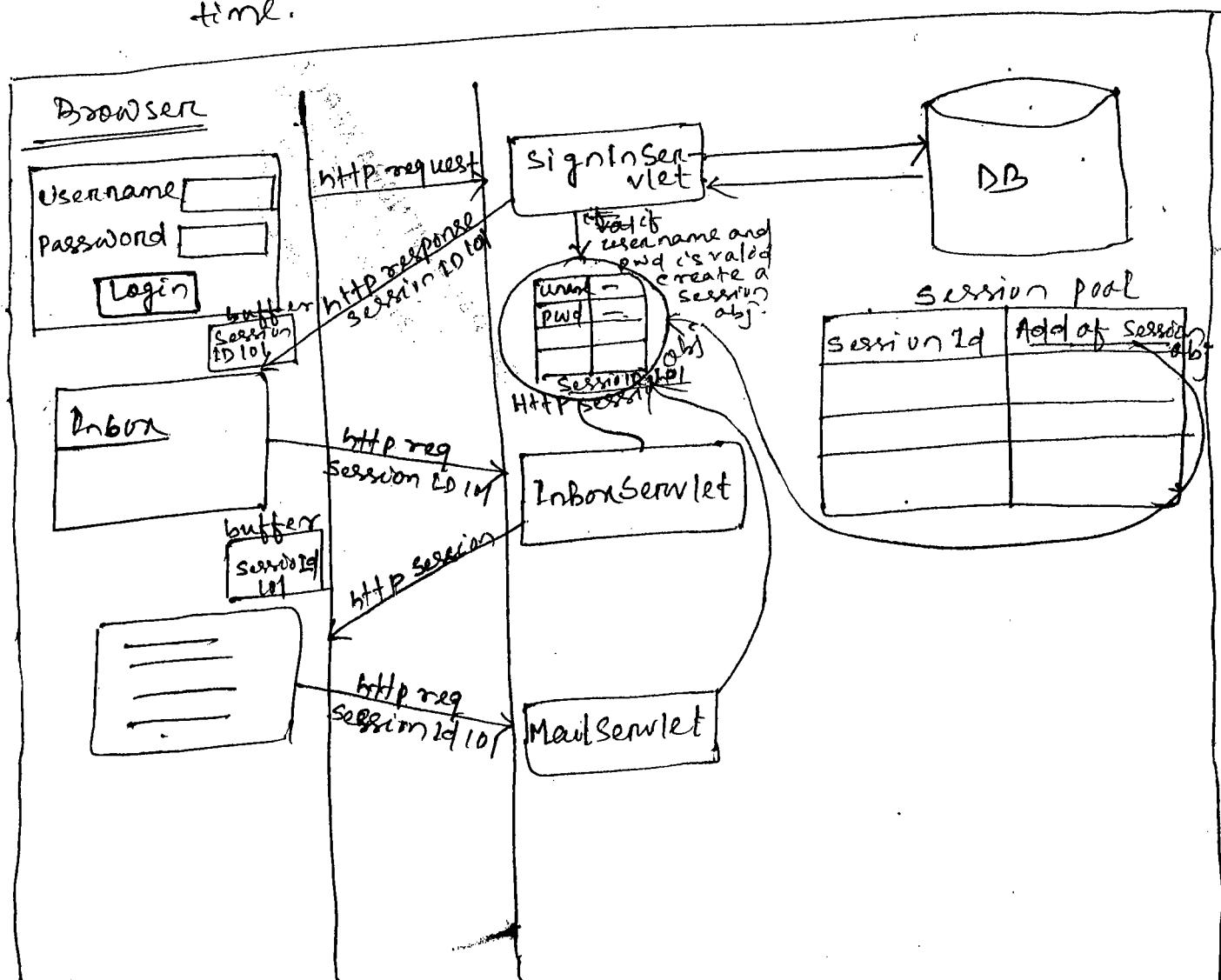
## "); out.println (c3.getName () + ":" + c3.getValue()); out.println ("

");  
    }  
}  
}  
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagari Bakery
Opp. CDAC, Bakampet Road,
Ameerpet, Hyderabad.

HttpSession :-

- HttpSession is an interface. This interface is implemented by Servlet container.
- An object of HttpSession created by Servlet container and stored in session pool.
- container creates a session id for each user/client. The container uses this ID to identify the particular user. An object of HttpSession can be used to perform two task.
 - (1) bind objects
 - (2) view and manipulate information about a session, such as the session identifier, creation time and last access time.



Q. How to get HttpSession object?

→ HttpServletRequest interface provide 2 methods to get the object of HttpSession.

(1) HttpSession getSession()

Returns the current session associated with this request, or if the request doesn't have a session, creates one.

(2) HttpSession getSession(boolean create)

Returns the current HttpSession associated with this request or, if there is no current session and create is true, returns a new session.

① http request	HttpSession session = req.getSession(); create new session
② http request session:101	HttpSession session = req.getSession(); get existing session from Session pool.
③ http request	HttpSession session = req.getSession(true); create new session
④ http request session:101	HttpSession session = req.getSession(true); get existing session.
⑤ http request	HttpSession session = req.getSession(null); when we don't want to create a session returns null
⑥ http request session:101	HttpSession session = req.getSession(false); get existing session

Methods of ^{Http} Session Interface :-

1. Object getAttribute (String name)

Returns the object bound with the specified name in this session, or null if no object is bound under the name.

2. void setAttribute (String name, Object value)

Binds an object to this session, using the name specified.

3. void removeAttribute (String name)

Removes the object bound with the specified name from this session

4. long getCreationTime()

Returns the time when this session was created measured in milliseconds

Example of Session :-

Webapplication2A

home.html :-

```
<html>
<body bgcolor="cyan">
<form action="o/signin">
<h2>
  Username <input type="text" name="t1"> <br>
  password <input type="text" name="t2"> <br>
  <input type="submit" value="signin">
</h2>
</form>
</body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Servlet

SignInServlet

```
package com.net.servlets;
import java.sql.*;
import java.io.IOException;
@WebServlet("/SignIn")
public class SignInServlet extends HttpServlet {
    protected void doGet(HttpServletRequest req,
                         HttpServletResponse res)
        throws ServletException, IOException
```

(write here) ~~and it method.~~

{

```
private Connection cn;
```

```
public void init()
```

```
try{
```

```
Class.forName("-----");
cn = DriverManager.getConnection("-----", "-----", "-----");
```

```
}
```

```
catch (Exception e)
```

```
{ e.printStackTrace(); }
```

```
}
```

~~catch~~

```
String user = req.getParameter("t1");
String pwd = req.getParameter("t2");
```

```
PrintWriter out = res.getWriter();
```

```
try{
```

```
PreparedStatement ps = cn.prepareStatement
```

```
("select count(*) from user_reg where");
```

uname=? and pwd=?");

```
ResultSet rs = ps.executeQuery();
```

```
ps.setString(1, user);
```

```
ps.setString(2, pwd);
```

```
ResultSet rs = ps.executeQuery();
```

```
    rs.next();
    int c = rs.getInt(1);
    if(c!=0)
    {
        HttpSession session = req.getSession();
        session.setAttribute("uname", user);
        out.println("<html>");
        out.println("<body bgcolor=yellow>");
        out.println("<h2>");
        String url = "http://localhost:8086/Webapplication024/inbox";
        out.println("<a href=" + url + ">");
        out.println("<inbox</a>");
        out.println("<h2><body></html>");
    }
    else
    {
        RequestDispatcher rd = req.getRequestDispatcher("home.html");
        rd.include(req, res);
        out.println("<h2> Invalid username and password</h2>");
    }
}
catch (Exception p)
{
    p.printStackTrace();
}
```

InboxServlet

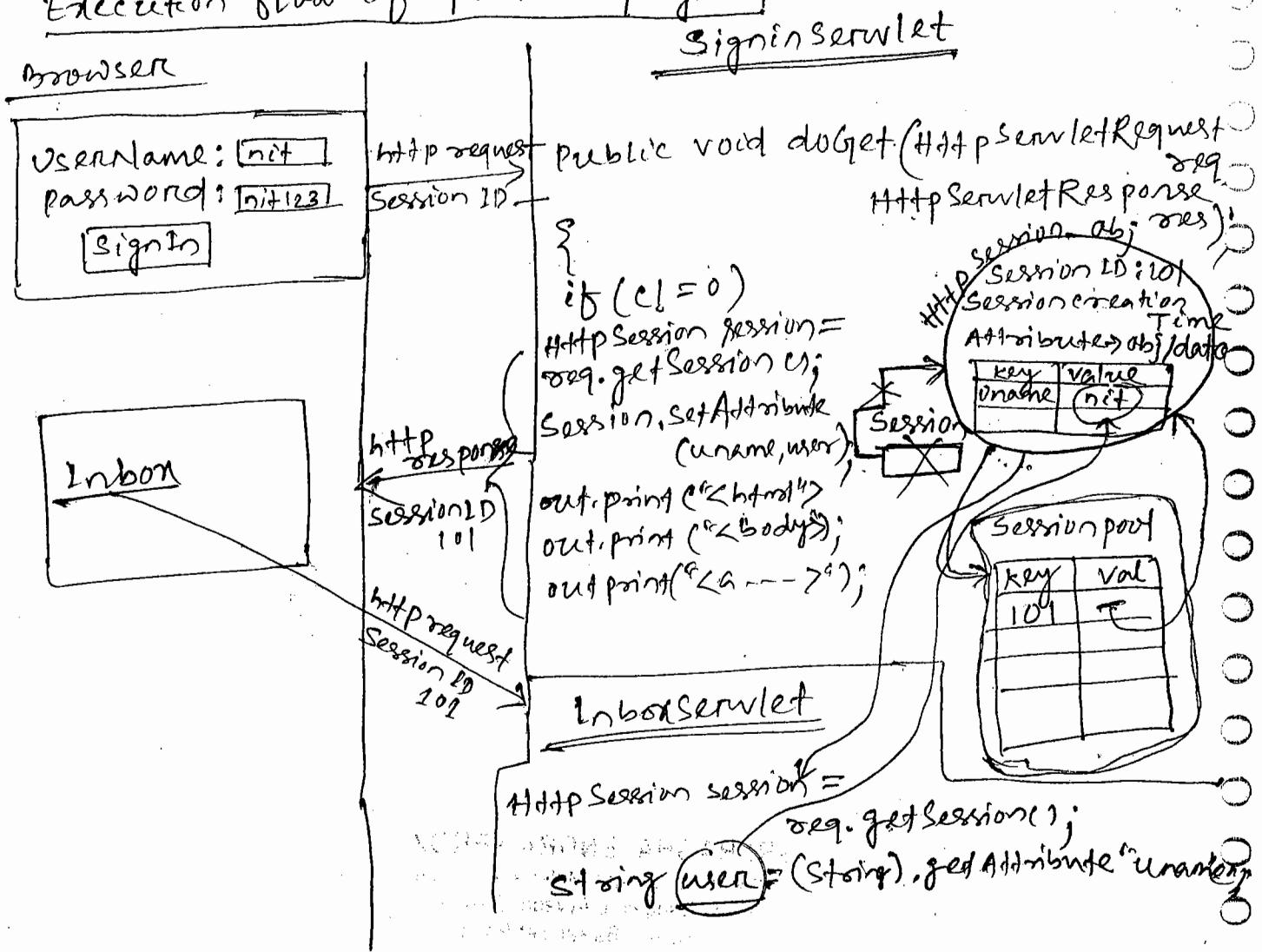
```
package com.nit.servlets;
import java.io.IOException;
@WebServlet("/inbox")
public class InboxServlet extends HttpServlet {
    protected void doGet(HttpServletRequest req, HttpServletResponse res)
        throws ServletException, IOException {
        HttpSession session = req.getSession();
        String user = (String) session.getAttribute("uname");
        PrintWriter out = res.getWriter();
        out.println("<h2>");
        out.println("Hello "+user+" this is your inbox");
        out.println("</h2>");
```

33 ..

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

09.12.15

Execution flow of previous program



How to invalidate session?

→ Session can be invalidated in two ways.

1. by assigning time

2. remove session from servlet.

Based on time there are two types:-

① Setting timeout in the deployment descriptor:

This can be done by specifying timeout b/w the <session-timeout> tags as follows.

```
<session-config>
```

```
  <session-timeout> time </session-timeout>
```

```
</session-config>
```

→ Time is define in min.

② Setting timeout programmatically:

This will set the timeout for a specific session. The syntax for setting the timeout programmatically is as follows.

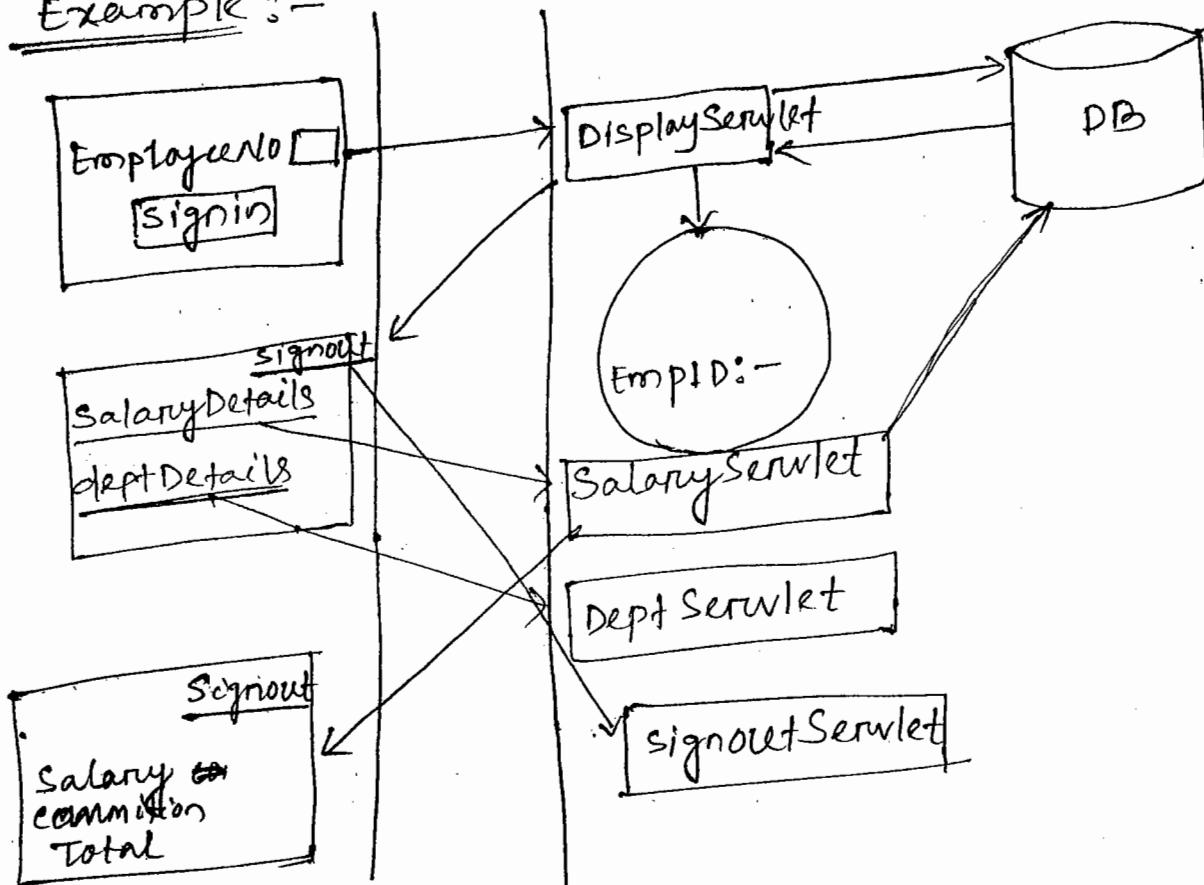
```
public void setMaxInactiveInterval(int interval)
```

interval is defined in seconds.

Q. How can the session in servlet can be destroyed?

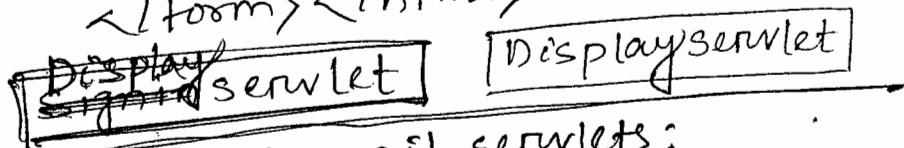
→ using `session.invalidate()` Method, which makes the container abandon the session on which the method is called.

Example:-



home.html

```
<body>
<html>
<body bgcolor="cyan">
<form action="/display">
<h2>
EmployeeId <input type="text" name="t1"> <br>
<input type="submit" value="signin">
</h2>
</form> </html>
```



```
package com.it.servlets;
import java.io.*;
import java.net.*;
import javax.servlet.*;
import javax.servlet.annotation.*;
import java.sql.*;
import java.util.*;

@WebServlet("/display")
public class DisplayServlet extends HttpServlet
{
    Connection cn;
    public void init()
    {
        try
        {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            cn = DM.getConnection("","","");
        }
        catch (Exception k)
        {
            k.printStackTrace();
        }
    }
}
```

```

public void doGet(HttpServletRequest req,
                   HttpServletResponse res)
throws ServletException, IOException {
    int eno = Integer.parseInt(req.getParameter("e1"));
    PrintWriter out = res.getWriter();
    try {
        PreparedStatement ps = cn.prepareStatement(
            "Select count(*) from emp where empno = ?");
        ps.setInt(1, eno);
        ResultSet rs = ps.executeQuery();
        rs.next();
        int c = rs.getInt(1);
        if (c == 0) {
            {
                HttpSession session = req.getSession();
                session.setAttribute("empno", eno);
                out.println("<html>");
                out.println("<body>");
                out.println("<h2>");
                String url1 = "http://localhost:8086/webapp2/salary";
                String url2 = "http://localhost:8086/webapp2/signout";
                out.println("<a href = " + url1 + "> Salary Details </a><br><br>");
                out.println("<a href = " + url2 + "> Signout </a>");
```

~~if (c == 0) {~~

```

                out.println("</h2></body></html>");
```

 }
 else
 {
 RequestDispatcher rd = req.getRequestDispatcher(
 "home.html");
 }
}

```
    rd.include(req, res)
out.println("<h2> Invalid empno </h2>");  
}  
catch (Exception k)  
{  
    k.printStackTrace();  
}
```

```
}
```

SignoutServlet

```
package com.nit.servlets;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import javax.servlet.annotation.*;
import javax.servlet.annotation.WebServlet("signout")
@WebServlet("signout")
public class SignoutServlet extends HttpServlet
{
    public void doGet(HttpServletRequest req,
                      HttpServletResponse res)
        throws ServletException, IOException
    {
        HttpSession session = req.getSession();
        if (session != null)
        {
            session.invalidate();
            RequestDispatcher rd = req.getRequestDispatcher("home.html");
            rd.include(req, res);
        }
    }
}
```

Web.xml

```
<web-app>
  <session-config>
    <session-timeout>10</session-timeout>
  </session-config>
</web-app>
```

SalaryServlet

```
package com.nit.servlets;
import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.annotation.*;
import java.io.*;
import java.sql.*;
@WebServlet("/salary")
public class SalaryServlet extends HttpServlet
{
    connection cn;
    public void init()
    {
        try
        {
            Class.forName("—");
            cn = DriverManager.getConnection("—", "—", "—");
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }
    public void doGet(HttpServletRequest req,
                      HttpServletResponse res)
        throws ServletException, IOException
    {
        HttpSession session = req.getSession();
        if (session == null)
        {
            int id = (Integer) session.getAttribute("empno");
            HandleNPEException();
        }
        HttpSession session = req.getSession();
    }
}
```

Handle NPEException

```
HttpSession session = req.getSession();
if (session == null)
{
    int id = (Integer) session.getAttribute("empno");
}
```

```

int empno=(Integer) session.getAttribute("empno");
PrintWriter out = res.getWriter();
try{
    PreparedStatement ps = cn.prepareStatement("Select sal, nvl(comm,0) + nvl(sal,0) from emp where empno=?");
    ps.setInt(1, empno);
    ResultSet rs = ps.executeQuery();
    rs.next();
    out.println("<h2>");
    out.println("<div>" + rs.getFloat(1) + ":" + rs.getFloat(2));
    out.println("</div>" + rs.getFloat(3));
    String url = "http://localhost:8086/websapp2/signout";
    out.println("<br><a href=" + url + " signout></a>");
    out.println("</h2>");
}
catch (Exception e){
    e.printStackTrace();
}
}

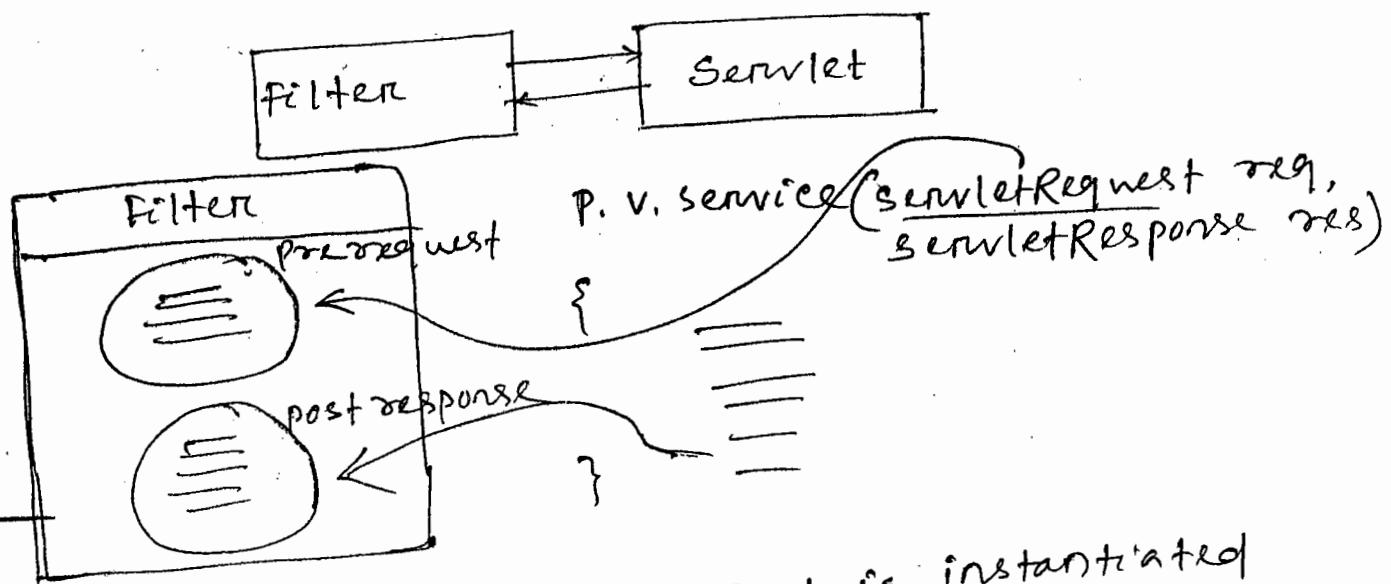
```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

○ 10.12.15

○ Filters :-

- → Filters are part of Servlet API since 2.3.



- → like a servlet, a filter object is instantiated and managed by the container and follows a life cycle that is similar to that of a servlet
- → a filter has four stages :-
- (1) instantiate.
 - (2) initialize
 - (3) filter
 - (4) destroy.
- → These stages are similar to a servlet's instantiate, initialize, service and destroy.
- → Filter are used to pre-process the request and post process the response.
- → A filter is java object that performs filtering tasks on either the request to a resource or on the response from a resource or both.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

→ These are the some of Application using filter.

- (1) Authentication filters
- (2) Logging and Auditing filters.
- (3) Image conversion filters
- (4) Data ~~conversion~~ filters
- (5) Encryption filters.

→ Interfaces belongs to filters.

1. Filter.
2. FilterConfig.
3. FilterChain.

→ All these interfaces are available in java.n.HttpServlet package.

1. Filter :-

- Filter is an interface.
- Every filter must be inherited from this interface.
- It provide lifecycle methods

Methods :-

1. void init (FilterConfig filterConfig)
called by the web container to indicate to a filter that it is being placed into service.

2. void doFilter (servletRequest req,
servletResponse res, FilterChain chain).

The doFilter method of the filter is called by the container each time a request/response pair is passed through the chain due to a client request for a resource at the end of the chain.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Opp. CDAC, Ayyagir Bakery,
Ameerpet, Hyderabad.

3. void destroy()

called by the web container to indicate to a filter that it is being taken out of service.

→ Filter information must be provided inside web.xml.

→ Filter is mapped with one or more ~~then~~ than are servlet.

Web.xml :-

```
<Web-app>
  <servlet>
    <servlet-name> name </servlet-name>
    <servlet-class> Servlet Class name </servlet-class>
    <filter>
      <filter-name> name </filter-name>
      <filter-class> Filter Class name </filter-class>
    </filter>
  </servlet>
  <filter-mapping>
    <filter-name> name </filter-name>
    <servlet-name> name </servlet-name>
  </filter-mapping>
</Web-app>
```

The diagram illustrates the mapping of filters to servlets in the `web.xml` configuration file. It shows a tree structure where filters map to servlets, which then map to specific URLs. Annotations explain how `init-param` values are stored in separate objects for each.

- Servlet Configuration:** A large oval labeled "Servlet Config obj." contains a grid labeled "init-param". An arrow from the `<init-param>` section of the `<filter>` block points to this grid, with the annotation "stored inside".
- Filter Configuration:** A smaller oval labeled "Filter Config obj." contains a grid labeled "init-param". An arrow from the `<init-param>` section of the `<filter>` block points to this grid, with the annotation "stored inside".
- Mapping:** Arrows point from the `<filter-mapping>` sections to the corresponding `<servlet-name>` and `<url-pattern>` sections, indicating the mapping relationship.

2. FilterChain :-

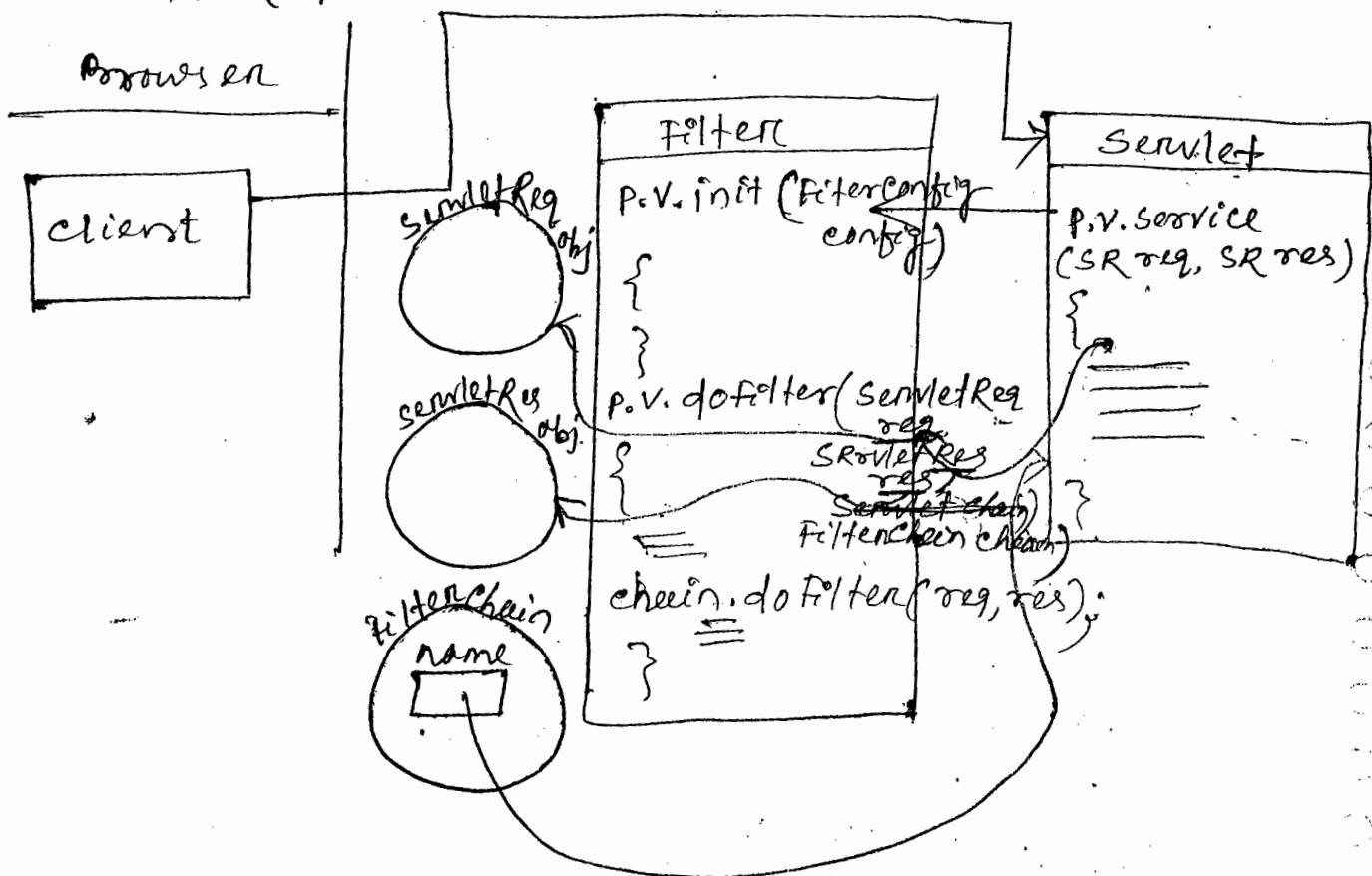
- FilterChain is an interface, which is implemented by servlet container.
- Filters use the filterChain to invoke the next filter in the chain, or if the calling filter is the last filter in the chain, to invoke the resource at the end of the chain.

Methods

- ~~void doFilter~~:

`void doFilter(ServletRequest req, ServletResponse res)`

causes the next filter in the chain to be invoked, or if the calling filter is the last filter in the chain, causes the resource at the end of the chain to be invoked



11.12.15

Webapplication 26

- home.html

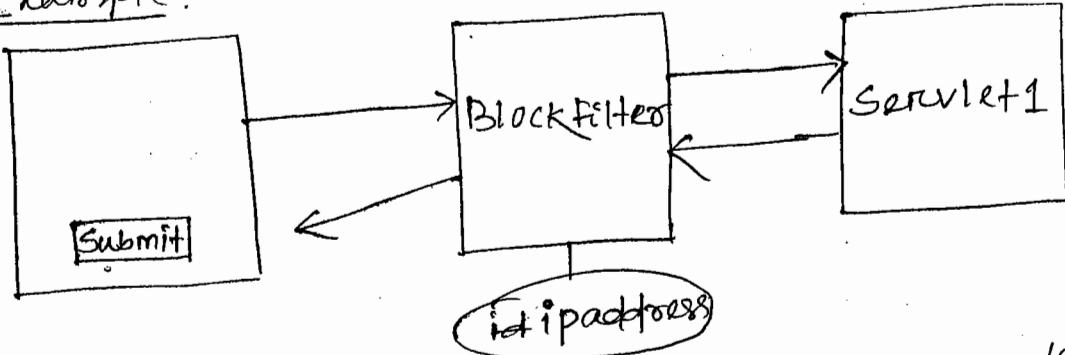
WEB-INF

classes

BlockFilter
Servlet1

web.xml

Example:-



Q. How to get the IP address of client in servlet?

→ HttpServletRequest provides getRemoteAddr() Method,
which return client ip address.

home.html

```

<html>
  <body bgcolor="cyan">
    <form action = "/Servlet1">
      <input type = "Submit" />
    </form>
  </body>
</html>
  
```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Develop filter inside

Block filter

```
package com.nit.filters;
import javax.servlet.*;
import java.io.*;
public class BlockFilter implements Filter {
    public void init(FilterConfig config)
        throws ServletException {
    }
    public void doFilter(ServletRequest req,
                        ServletResponse res,
                        FilterChain chain)
        throws ServletException, IOException {
        String ip = req.getRemoteAddr();
        PrintWriter out = res.getWriter();
        if(ip.equals("127.0.0.1")){
            out.println("<h2>Your ip address is blocked  
by this website</h2>");
        } else {
            fc.doFilter(req, res);
        }
    }
    public void destroy(){}
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Servlet1

```
package com.onit.servlets;
import javax.servlet.*;
public class Servlet1 extends GenericServlet
{
    public void service(ServletRequest req,
                        ServletResponse res)
        throws ServletException, IOException
    {
        PrintWriter out = res.getWriter();
        out.println("<h2>Hello Client, welcome to my
                    website </h2>");
    }
}
```

Web.xml :-

```
<web-app>
< servlet >
    < s-n > S1 < /s-n >
    < s-c > com.onit.servlets.Servlet1 < /s-c >
    < /servlet >
    < filter >
        < filter-name > F1 < /filter-name >
        < filter-class > com.onit.filters.BlockFilter < /filter-class >
    < /filter >
    < filter-mapping >
        < filter-name > F1 < /filter-name >
        < servlet-name > S1 < /servlet-name >
    < /filter-mapping >
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

< servlet-mapping >
  < servlet-name > S1 </ servlet-name >
  < url-pattern > /Sera1 </ url-pattern >
< /servlet-mapping >
< /Web-app >

```

Servlet Wrappers :-

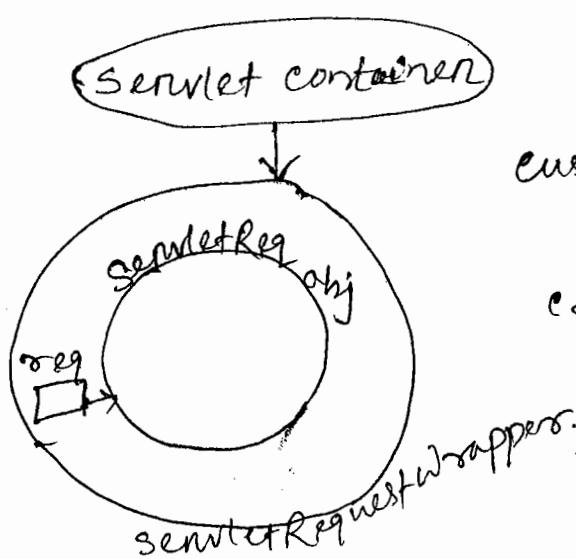
1. `ServletRequestWrapper`.
2. `ServletResponseWrapper`.
3. `HttpServletRequestWrapper`
4. `HttpServletResponseWrapper`.

→ Wrappers are used to customize request and response created by container.

ServletRequestWrapper :-

→ provides a convenient implementation of the `ServletRequest` interface that can be subclassed by developers wishing to adapt the request to a servlet.

`ServletRequestWrapper (ServletRequest request)`
 creates a `ServletRequest` adaptor wrapping
 the given request object



```

customerRequest c =
new customerRequest
  (req);
c.getParameter("user");
  
```

ServletRequest → I

«implements»

ServletRequestWrapper

```
private ServletRequest req;  
public ServletRequest (ServletRequest req)  
{  
    this.req = req;  
}  
P. String getParameter (String name)  
{  
    return req.getParameter(name);  
}
```

«extends»

CustomRequest

```
Public CustomRequest (ServletRequest req)
```

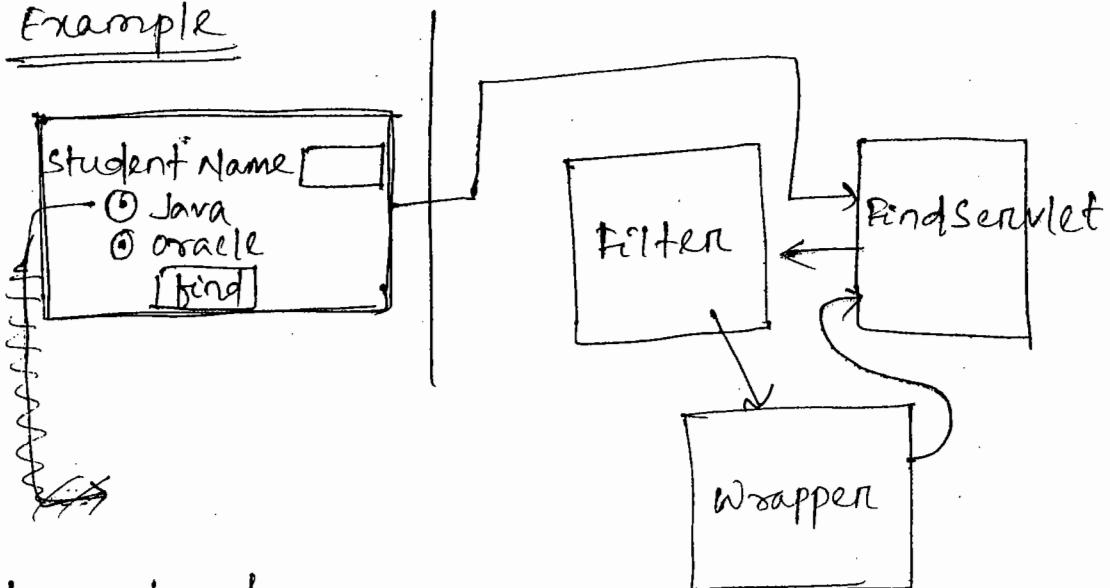
```
{  
    super(req);  
}
```

```
Public String getParameter (String name)
```

```
{  
    Value = super.getParameter(name);  
    return Value.toUpperCase();  
}
```

~~Servlet~~

Example



home.html

```
<html>
<body bgcolor="cyan">
<form action="./find">
<h2>
  Student Name<input type="text" name="T1"><br>
  <input type="radio" name="course"
        value="Java"> Java <br>
  <input type="radio" name="course"
        value="Oracle"> Oracle <br>
  <input type="submit" value="Find">
</h2>
</form>
</body>
</html>
```

RequestWrapper.java

```
package com.nit.wrappers;
import javax.servlet.*;
public class RequestWrapper extends ServletRequestWrapper {
    public RequestWrapper (ServletRequest req)
    {
        super(req);
    }
    public String getParameter (String name)
    {
        String value = super.getParameter(name);
        String v=null;
        if (value.equals("java"))
            v = "Java Fee: 1000";
        if (value.equals("oracle"))
            v = "Oracle Fee: 500";
        return v;
    }
}
```

Filter1.java

```
package com.nit.filters;
import javax.servlet.*;
import java.io.*;
import com.nit.wrappers.RequestWrapper;
public class Filter1 implements filter
{
    public void init (FilterConfig config) throws
        ServletException,
    {
    }
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
Public void doFilter(ServletRequest req,  
                     ServletResponse res,  
                     Servlet FilterChain fc)  
throws ServletException, IOException
```

```
{ RequestWrapper rw = new RequestWrapper(req);  
    fc.doFilter(rw, res);  
}
```

```
Public void destroy()
```

```
}
```

findServlet

```
package com.nit.servlets;  
import javax.servlet.*;  
import java.io.*;  
public class findServlet extends GenericServlet
```

```
{ public void service(ServletRequest req, ServletResponse res)  
throws ServletException, IOException
```

```
{ PrintWriter out = res.getWriter();  
String value = req.getParameter("course");  
String name = req.getParameter("t1");  
out.println("<h2>");  
out.println("Hello "+uname+"<br>");  
out.println(value);  
out.println("</h2>");
```

```
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Web.xml :-

```
<Web-app>
  <Servlet>
    <s-n> s1 </s-n>
    <s-c> com.nit.servlets.FindServlet </s-c>
  </Servlet>
  <Filter>
    <f-n> f1 </f-n>
    <f-c> com.nit.filters.Filter1 </f-c>
  </Filter>
  <Filter-Mapping>
    <f-n> f1 </f-n>
    <f-c> s1 </f-c>
    <filter-mapping>
      <url-pattern> /Find </url-pattern>
    </filter-mapping>
  </Filter-Mapping>
  <Servlet-Mapping>
    <s-n> s1 </s-n>
    <url-pattern> /Find </url-pattern>
  </Servlet-Mapping>
</Web-app>
```

14.12.15

Listeners:-

(Q) why do we have servlet listeners?

Ans:-

1. 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.
2. If your 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.

To handle these scenario, servlet API provides Listener interfaces that we can implement and configure to ~~Listener~~ to an event and do certain operations.

The listeners are

1. `ServletContextListener`.
2. `ServletRequestListener`.
3. `HttpSessionListener`.

1. ServletContextListener :-

- Interface for receiving notification events about ServletContext lifecycle changes.
- In order to receive these notification events, the implementation class must be either declared in the deployment descriptor of the web application, annotated with `@WebListener`, or registered via one of the `addListener` methods defined on `ServletContext`.

Web.xml

```
<web-app>
  <listener>
    <listener-name> instance name </listener-name>
    <listener-class> instance - class </listener-class>
  </listener>
  :
  :
</web-app>
```

SRIRAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Ballampet Road,
Ameerpet, Hyderabad.

Annotation :-

`@WebListener`

```
public class MyListener implements
  ServletContextListener
```

{

}

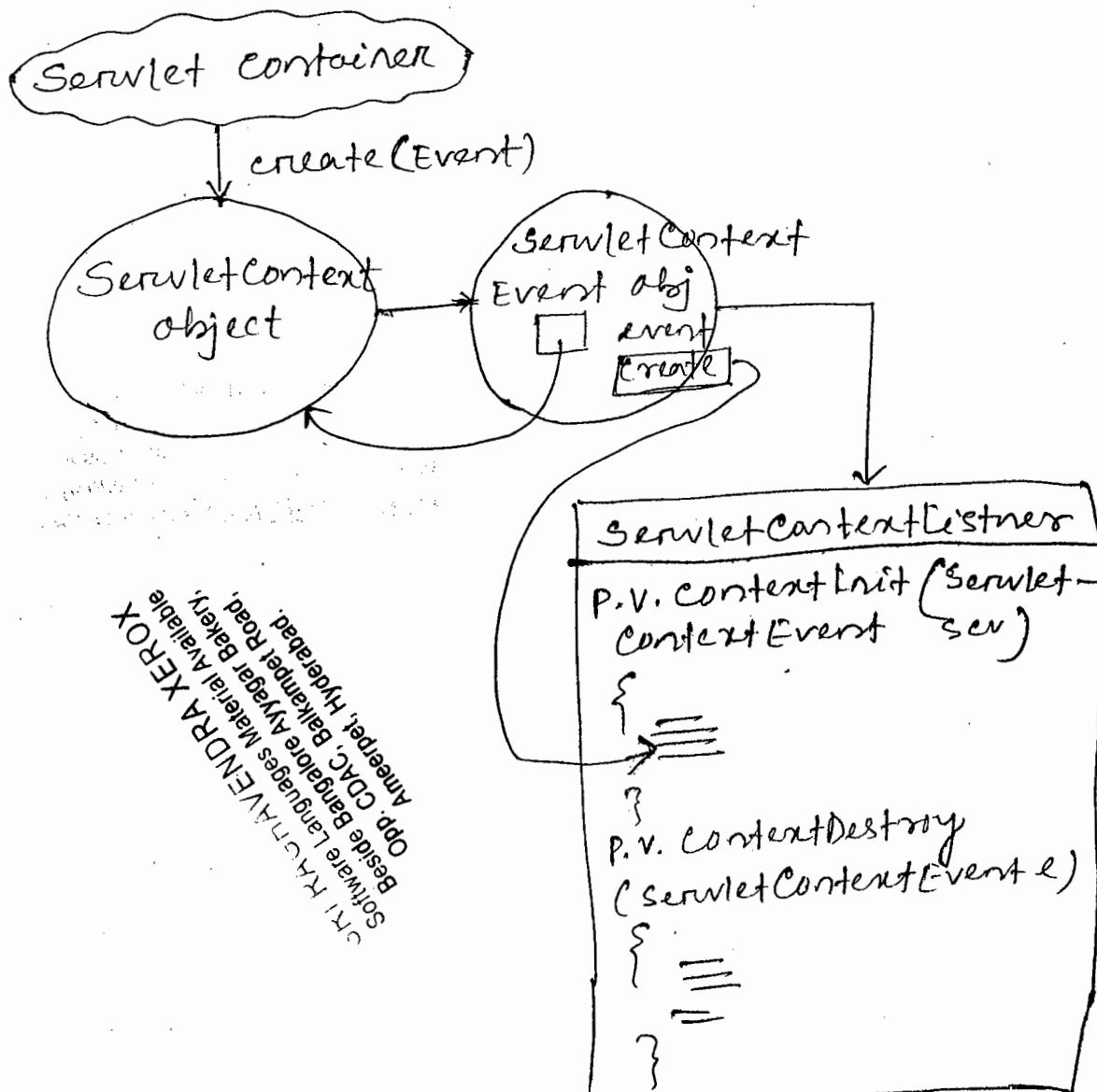
Methods

1) **void contextDestroyed (ServletContextEvent sce)**

- Receives notification that the servlet context is about to be shut down.

2) **void contextInitialized (ServletContextEvent sce)**

- Receives notification that the web application initialization process is starting.



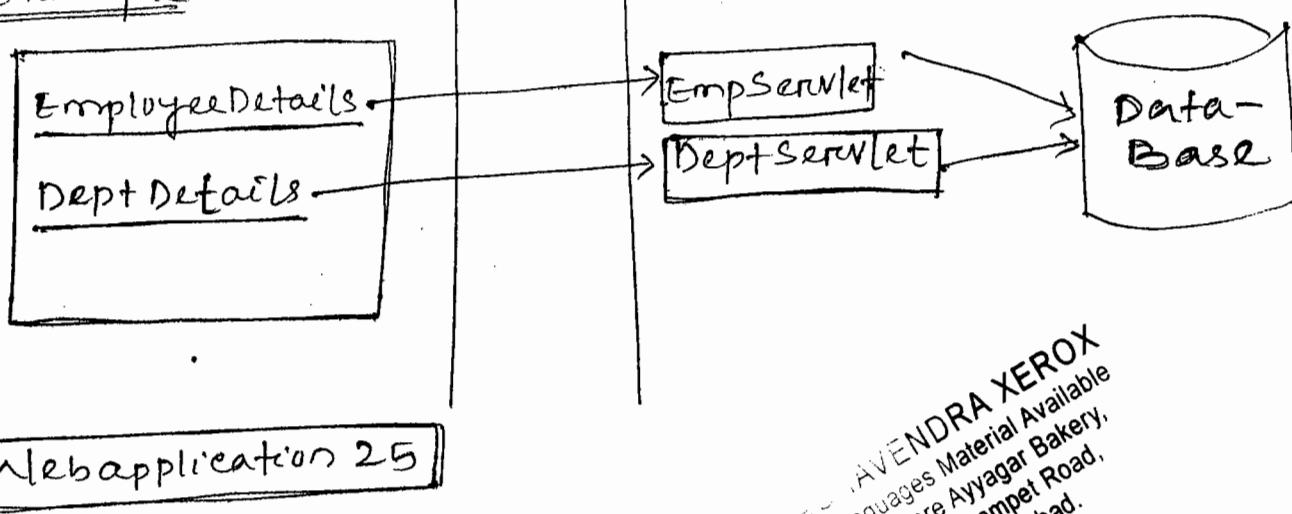
ServletContextEvent :-

ServletContext getServletContext ()

Return the ServletContext that changed.

method.

Example



Shri K. AVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Webapplication 25

home.html

```
<html>
<body bgcolor="cyan">
```

```
<h2>
```

```
<a href="http://localhost:8085/Webapplication25/
emp"> EmployeeDetails </a>
```

```
</h2>
```

```
</body>
```

```
</html>
```

Listener

```
package com.nit.listeners;
import javax.servlet.ServletContextEvent;
@WebListener
public class contextListener implements
    ServletContextListener {
    public void contextInitialized(ServletContextEvent sce) {
        try {
            Class.forName("oracle.jdbc.driver.
                OracleDriver");
            Connection cn = DM.getConnection ("-", "-", "-");
            ServletContext ctx = sce.getServletContext();
            ctx.setAttribute ("conn", cn);
        } catch (Exception k) {
            k.printStackTrace();
        }
    }
    public void contextDestroyed(ServletContextEvent sce) {
    }
}
```

contextListener

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Servlet

EmpServlet

```
package com.onit.servlets;
import java.sql.*;
@WebServlet ("emp")
public class EmpServlet extends HttpServlet
{
protected void doGet (HttpServletRequest req, HttpServletResponse res)
throws ServletException, IOException
{
    PrintWriter out = res.getWriter();
    try
    {
        ServletContext ctx = getServletContext();
        Connection cn = (Connection) ctx.getAttribute
("conn");
        PreparedStatement ps = .
        Statement st = cn.createStatement();
        ResultSet rs = st.executeQuery ("select empno,
name, sal from emp");
        while (rs.next())
        {
            out.println ("<h2>");
            out.println ("<rs.getInt(1)>" + "<: " + rs.getString(2)
+ "<: " + rs.getFloat(3));
            out.println ("</h2>");
        }
    }
}
```

```

        catch (Exception k)
    {
        k.printStackTrace();
    }
}
}

```

2. HttpSessionListener :-

→ Interface for receiving notification events about HttpSession lifecycle changes.

Methods :-

1. **void sessionCreated (HttpSessionEvent se)**

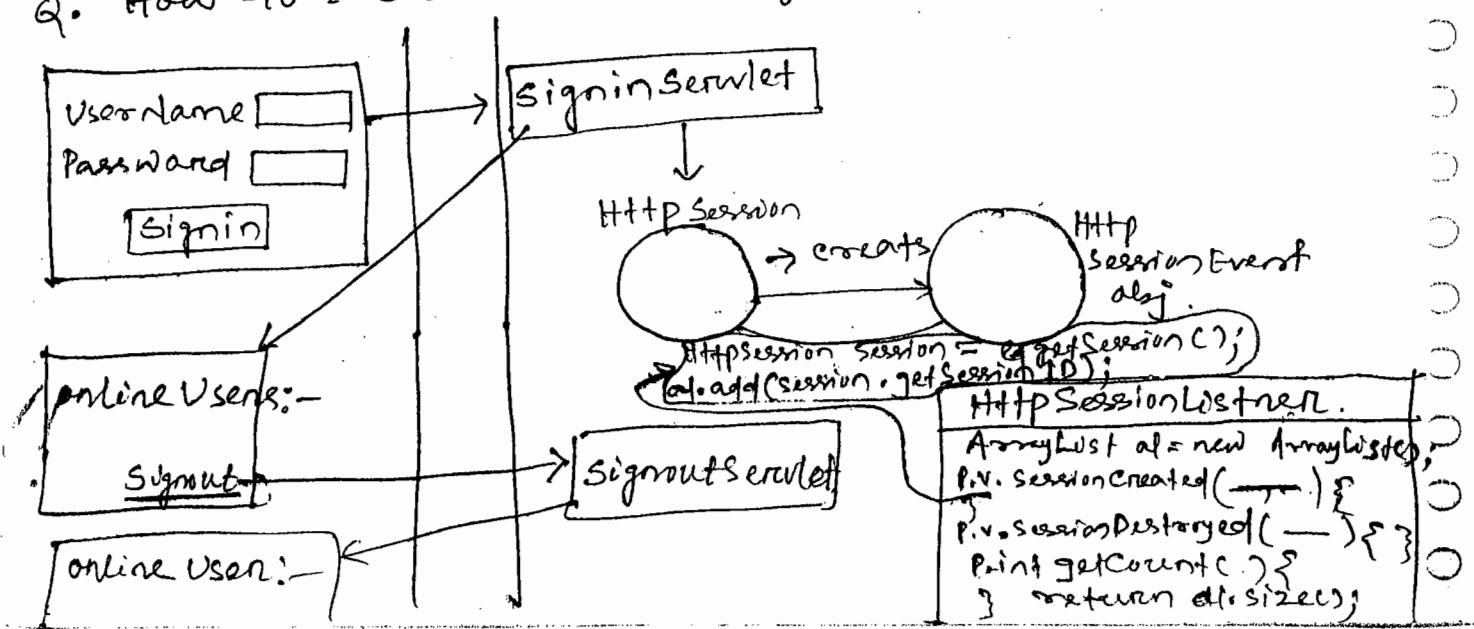
Receives notification that a session has been created.

2. **void sessionDestroyed (HttpSessionEvent se)**

Receives notification that a session is about to be invalidated.

Q. 12.15 :-

Q. How do I count number of online users ?



Servlet HTML :-

```

<html> [home.html]
<body bgcolor="cyan">
<form action = "s/signin">
<h2>
  Username <input type = "text" name = "t1"> <br>
  Password <input type = "text" name = "t2"> <br>
  <input type = "submit" value = "Signin">
</h2>
</form>
</body> </html>

```

Signinservlet :-

```

package com.nit.servlets;
import java.io.IOException;
@WebServlet("/signin")
public class Signinservlet extends HttpServlet {
  protected void doGet(HttpServletRequest req,
                        HttpServletResponse res)
    throws ServletException, IOException {
    HttpSession session = req.getSession();
    SessionListener listener = (SessionListener) session.
      getAttribute("counter");
    int count = listener.getCount(); // return session
    PrintWriter out = res.getWriter();
    out.println("<h2>");
    out.println("Number of Online Users " + count);
    out.println("<a href = http://localhost:8086/webapplica-
               tion27/signout> Signout</a>");
    out.println("</h2>");
  }
}

```

33

Listener:-

```
package com.onit.listeners;
import javax.servlet.*;
import java.util.*;
@WebListener
public class SessionListener implements HttpSessionListener
{
    ArrayList al = new ArrayList();
    public void sessionCreated (HttpSessionEvent e)
    {
        HttpSession session = e.getSession();
        al.add(session.getId());
        session.setAttribute("counter", this);
    }
    public void sessionDestroyed (HttpSessionEvent e)
    {
        HttpSession session = e.getSession();
        al.remove(session.getId());
    }
    public int getCount()
    {
        return al.size();
    }
}
```

SP DAGHAVENDRA XEROX
Languages Material
galore Ayyagar
AC, Balkampet
Ameerpet, Hyderabad.

SignoutServlet :-

```
package com.nit.servlets;
import java.io.IOException;
@WebServlet("/signout")
public class SignoutServlet extends HttpServlet {
protected void doGet(HttpServletRequest req, HttpServletResponse res) throws
ServletException, IOException {
HttpSession session = req.getSession(false);
if (session != null) {
SessionListener listener = (SessionListener) session.
getAttribute("counter");
session.invalidate();
int count = listener.getCount();
PrintWriter out = res.getWriter();
out.println("<h2>");
out.println("Online Users" + count);
out.println("</h2>");
```

```
}
```

Q. What is JSESSIONID?

(*)

What is JSESSIONID in JSP or Servlet?

Ans:- JSESSIONID is a cookie generated by Servlet container like Tomcat or Jetty and used for Session Management in JEE Web application for http protocol.

→ If Web server is using cookie for session management it creates and sends JSESSIONID cookie to the client and then client sends it back to server in subsequent http requests.

attribute listeners :-

→ All listener events belongs to attributes.

1. ServletRequestAttributeListener.
2. HttpSessionAttributeListener.
3. ServletContextAttributeListener.

HttpSessionAttributeListener :-

→ Interface for receiving notification events about HttpSession attribute changes.

void attributeAdded (HttpSessionBindingEvent event)

Receives notification that an attribute has been added to a session.

void attributeRemoved (HttpSessionBindingEvent event)

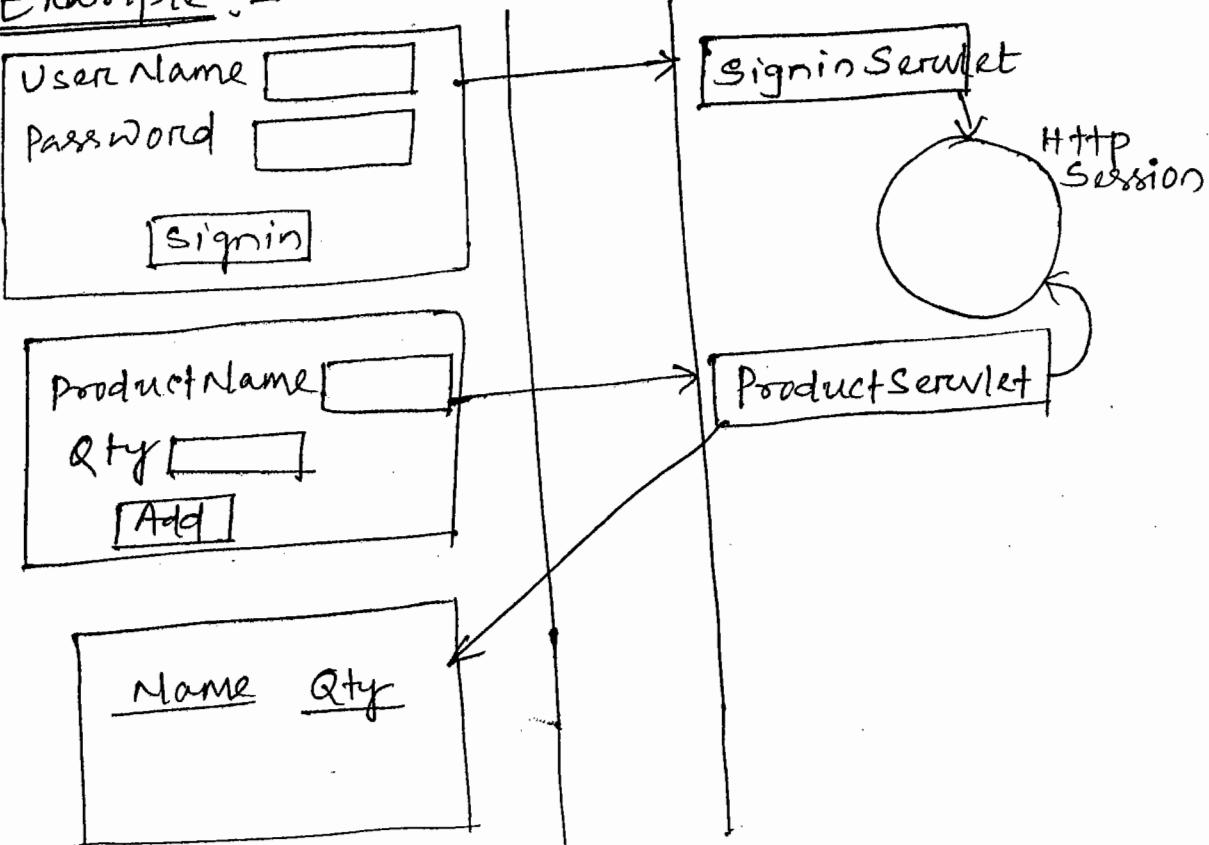
Receives notification that an attribute has been removed from a session.

void attributeReplaced (HttpSessionBindingEvent event)

Receives notification that an attribute has been replaced in a session.

AGHAVENDRA XEROX
Languages Material Available
Bangalore Ayyagar Bakery,
CDAC, Balkampet Road,
Serpet, Hyderabad.

Example :-



home.html :-

```
<html>
<body bgcolor= cyan>
<form action = ". / signin">
<h2>
  Username <input type = text name = "t1" > <br>
  Password <input type = password name = "t2" > <br>
  <input type = submit value = "signin" >
</h2>
</form>
</body>
</html>
```

product.html :-

```
<html>
<body bgcolor="red">
<form action="/product">
<h2>
< productName <input type="text" name="t1" />
<qty <input type="text" name="t2" />
<input type="submit" value="add" />
</h2>
</form>
</body>
</html>
```

SignServlet :-

```
package com.nit.servlets;
import java.io.IOException;
public class SignServlet extends HttpServlet
{
    protected void doGet(HttpServletRequest req,
                         HttpServletResponse res)
        throws ServletException, IOException
    {
        HttpSession session = req.getSession();
        RequestDispatcher rd = req.getRequestDispatcher("product.html");
        rd.include(req, res);
    }
}
```

Product Servlet :-

Package com.nit.servlets.

```
import java.io.IOException;
@WebServlet("/product")
```

```
public class ProductServlet extends HttpServlet {
```

```
protected void doGet(HttpServletRequest req,
                      HttpServletResponse res)
```

```
throws ServletException, IOException {
```

```
String p1 = req.getParameter("t1");
```

```
String q = req.getParameter("t2");
```

```
HttpSession session = req.getSession(false);
```

```
session.setAttribute(p1, q);
```

```
}
```

```
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Listener

com package com. niti. listeners.

WebListener

Public class AttributeListener implements

HttpSessionAttribute

Public void attributeAdded (HttpSessionBindingEvent e)

{
System.out.println(e.getName() + ":" + e.getValue());

}

}

16.12.15

HttpSessionBinding Listener :-

- Causes causes an object to be notified when it is bound to or unbound from a session. The object is notified by an HttpSessionBindingEvent object.
- It is having 2 Methods

Void valueBound (HttpSessionBindingEvent event)
~~Notification~~ Notifies the object that it is being bound to a session and identifies the session.

Void valueUnbound (HttpSessionBindingEvent event)
~~Notification~~ Notifies the object that it is being unbound from a session and identifies the session.

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

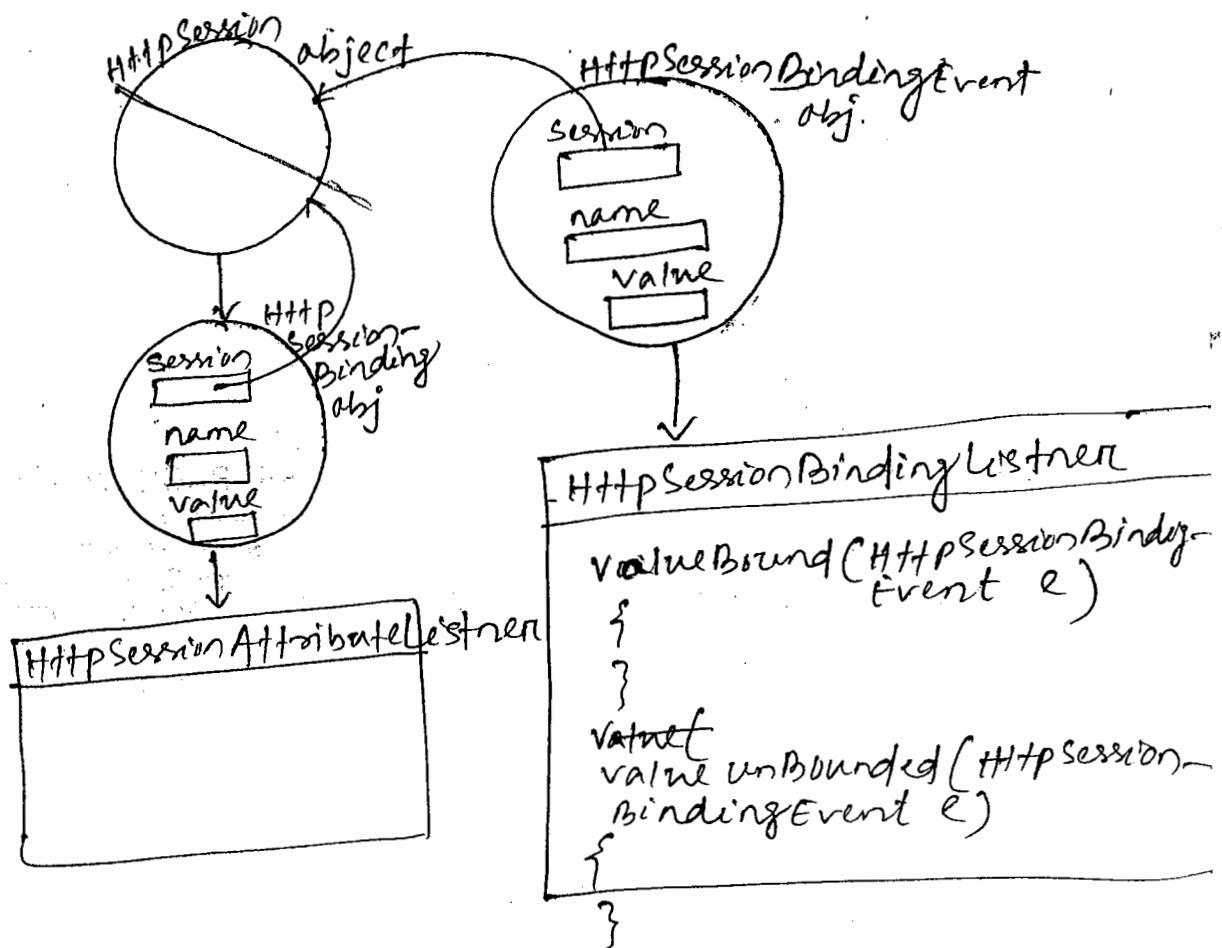
HttpSessionBindingEvent :-

Java.lang.String getName()

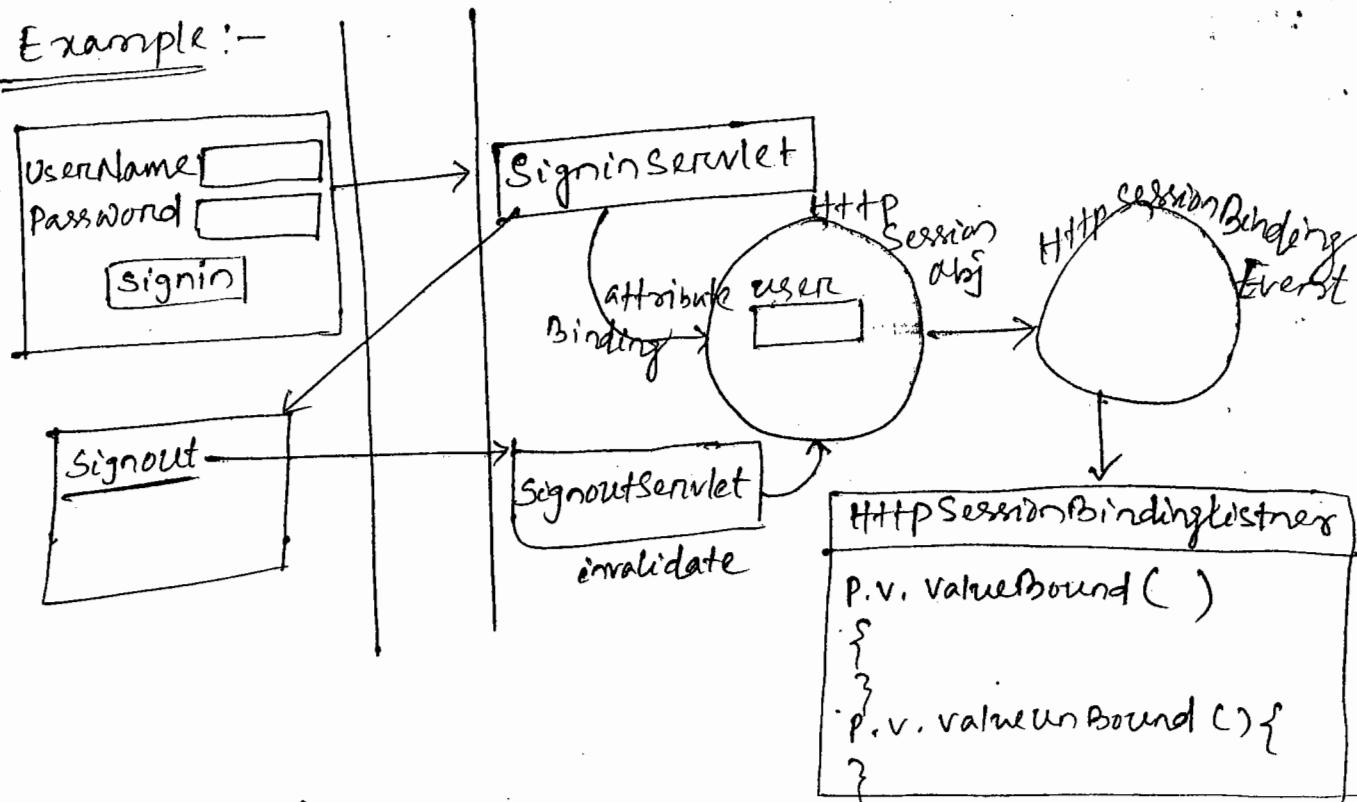
Returns the name with which the attribute is bound to or unbound from the session.

Java.lang.Object getValue()

Returns the value of the attribute that has been added, removed or replaced.



Example :-



home.html :-

```
<html>
<body style="background-color: cyan;>
<form action="c:/signin">
<h2>
  Username <input type="text" name="t1"> <br>
  Password <input type="text" name="t2"> <br>
  <input type="submit" value="signin">
</h2>
</form>
</body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SigninServlet

```
package com.nit.servlets;
import java.io.IOException;
@WebServlet("/signin")
public class SigninServlet extends HttpServlet {
protected void doGet(HttpServletRequest req, HttpServletResponse res)
throws ServletException, IOException
{
String u = req.getParameter("t1");
HttpSession session = req.getSession();
session.setAttribute("user", u);
PrintWriter out = res.getWriter();
String url = ("http://localhost:8086/webapplication2/signout");
out.println("<html>");
out.println("<body bgcolor=cyan>");
out.println("<a href=" + url + ">Signout</a>");
out.println("</body></html>");
}
```

}}}

Listner

```
package package com.nit.servlets;
import javax.servlet.annotation.WebServlet;
@WebServlet
public class BindingListner implements HttpSessionBinding{
void valueUnbound(HttpSessionBindingEvent e){
System.out.println(e.getName() + ":" + e.getValue());
}
void valueBound(HttpSessionBindingEvent e)
{
System.out.println(e.getName() + ":" + e.getValue());
}}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Signout Servlet :-

```
package com.nit.servlets;
```

```
@WebServlet("/signout")
```

```
public class SignoutServlet extends HttpServlet {  
protected void doGet(HttpServletRequest req, HttpServletResponse res)  
throws ServletException, IOException;
```

```
HttpSession session = req.getSession(false);
```

```
if (session != null)
```

```
session.invalidate();
```

```
RequestDispatcher rd = req.getRequestDispatcher
```

```
("home.html");
```

```
rd.include(req, res);
```

```
}
```

```
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

ServletOutputStream :-

→ provides an output stream for sending binary data to the client. A ServletOutputStream object is normally retrieved via the ServletResponse.
getOutputStream Method.

Example :-

[Web Application 30]

ImageServlet :-

```
package com.nit.servlets;
```

```
@WebServlet("/display")
```

```
public class Imageservlet extends HttpServlet {
```

```
protected void doGet(HttpServletRequest req,
```

```
HttpServletResponse res)
```

```
throws ServletException, IOException
```

```
{
```

```
res.setContentType("image/jpeg");
```

```
FileInputStream fis = new FileInputStream("fx.jpg");
```

```
ServletOutputStream s = res.getOutputStream();
```

```
int x;
```

```
while ( (x = fis.read()) != -1 )
```

```
s.write(x);
```

```
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

ResourceBundle class

- It is an abstract class.
- Resource bundles contains locale-specific objects when your program needs a locale-specific resource, a string for example, your program can load it from the resource bundle that is appropriate for the current user's locale.

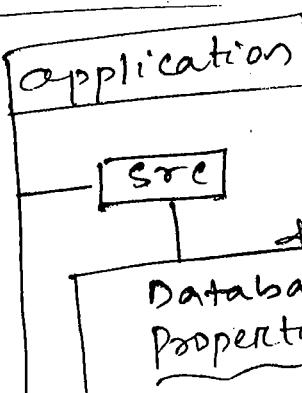
static ResourceBundle getBundle(String basename)

Get a resource bundle using the specified base name, the default locale and the caller's class loader.

String getString(String key)

Gets a string for the given key from this resource bundle or one of its parents

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.



Database Properties file

```
driver = oracle.jdbc.driver.OracleDriver  
url = jdbc:oracle:thin:@localhost:1521  
User = scott  
password = tiger
```

Java class

```
public class DBUtil
{
    static ResourceBundle r;
    static
    {
        r = ResourceBundle.getBundle("src/db.properties");
    }
    try {
        Class.forName(r.getString("driver"));
    } catch (Exception k)
    {
        k.printStackTrace();
    }
}
public static Connection getConnection()
{
    Connection cn = null;
    try {
        cn = DM.getConnection(r.getString("url"),
            r.getString("user"), r.getString("password"));
    } catch (Exception k)
    {
        k.printStackTrace();
    }
    return cn;
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayappa Temple
Opp. CDA Office
Mysore - 570001

Class :-

Method.

InputStream getResourceAsStream (String name)

- finds a resource with a given name.

Program :-

```
import java.io.*;
```

```
class Test
```

```
{
```

```
    public void main (String args[])
```

```
    {  
        Test t = new Test();  
        InputStream is = t.getClass().getResourceAs-  
        stream ("A.java");
```

```
        System.out.println (is);
```

```
}
```

~~open Stream like BufferedReader~~

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Q. what is the difference bet' PrintWriter and
ServletOutputStream ?

- PrintWriter is a character-stream class whereas
as ServletOutputStream is a byte stream class.
- We can use PrintWriter to write character
based information such as character array
and string to the response whereas we can
use ServletOutputStream to write ~~byte array~~
array data to the response.
- we can use servletResponse getWriter() to
get the PrintWriter instances whereas we
can use servletResponse getOutputStream() method
to get the ServletOutputStream object
reference.

Q. can we get PrintWriter and ServletOutputStream — stream both in a Servlet?

→ We can't get instances of both PrintWriter and ServletOutputStream in a single Servlet method, if we invoke both the methods; getWriter() and getOutputStream() on response; we will get Java.lang.IllegalStateException.

Servlet Annotations (Servlet 2.5)

→ Servlet Annotations are introduced in Servlet API 2.5 (JEE 5.0, JSE 5.0)

① @WebServlet :-

→ These annotations are to avoid writing web.xml.

→ javax.servlet.annotation package.

② @WebServlet :-

Annotation used to declare a Servlet.

→ Annotation is processed by the container at deployment time, and the corresponding Servlet made available at the specified URL patterns.

`@WebServlet(name = "s1", urlPattern = {"/first", "/second"})`
public class Servlet1 extends HttpServlet

{

}

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

④ @ WebServlet (urlPatterns = {"/ servlet1"},
initParams = { @WebInitParam ("p1", "10")
@WebInitParam ("p2", "20") })

public class Servlet1 extends HttpServlet

```
{  
    p. v. init ()
```

```
{
```

ServletConfig config = getServletConfig();

String p1 = config.getParam ("p1");

String p2 = config.getParam ("p2");

```
}
```

```
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

④ @ WebServlet (loadOnStartup = "1" urlPattern =
{"/ first"})

public class Servlet1 extends HttpServlet

```
{
```

```
}
```

② @WebFilter

→ This annotation is processed by the container at deployment time, and the corresponding filter applied to the specified URL patterns, servlets, and dispatcher types.

java.lang.String[] servletNames:

The names of the servlets to which the filter applies

@WebServlet (name = "s1", urlPatterns = {"/*/servlet1"})
public class Servlet1 extends HttpServlet

{

}

@WebFilter (servletNames = {"s1"})

public class Filter1 implements Filter

{

}

③ @WebListener

→ This annotation is used to declare a WebListener.

@WebListener

public class Listener1 implements ServletContextListener

{

}

(To avoid web.xml we write
@WebListener)

→ web.xml

<listener>

<listener-name> ListenerName </listener-name>

<listener-class> ListenerClassName </listener-class>

</listener>

Exceptions in Servlet

Q. How to handle exceptions thrown by application with another Servlet?

→ If you notice, doGet() and doPost() methods throw ServletException and IOException, since browser understand only HTML when our app! throw exception, Servlet container processes the exception and generate a HTML response. Same goes with other error codes like 404, 403 etc.

→ Servlet API provides support for custom Exception and Error Handler servlets that we can configure in deployment descriptor to the whole purpose of these servlets are to handle the exception or error & raised by app! and send HTML response that is useful for the user. We can provide link to app! home page or some details to let user know what went wrong.

Q. We can configure them in web.xml like below

Syntax

```
<error-page>
  <error-code> code </error-code>
  <location> url </location>
</error-page>
```

Syntax :-

```
<error-page>
  <exception-type> exception type </exception-type>
  <location> url </location>
</error-page>
```

Example

home.html

```
<html>
<body bgcolor=cyan>
<a href = "http://localhost:8086/webapplication32/
           ser1">Servlet1</a>
</body>
</html>
```

Servlet

Exception Handler

```
package com.nit.servlets;
import java.io.*;
import javax.servlet.*;
public class ExceptionHandler extends GenericServlet
{
    public void service (ServletRequest req,
                         ServletResponse res)
        throws ServletException, IOException
    {
        PrintWriter out = res.getWriter();
        out.println ("<html>");
        out.println ("<body bgcolor= yellow>");
        out.println ("<h2>");
        out.println ("requested resource is not available");
        out.println ("<a href = http://localhost:8085/
                     webapplication32/home.html> home
                     </a>");
        out.println ("</h2></body></html>");
    }
}
```

Web.xml

```
<web-app>
< servlet >
< s-n > S1 </s-n >
< s-c > com.onit.servlets.ExceptionHandler </s-c >
</servlet>
< servlet-mapping >
< s-n > S1 </s-n >
< url-pattern > /servlet/<servlet </url-pattern >
</servlet-mapping>

< error-page >
< error-code > 404 </error-code >
< location > /servlet/</location >
</error-page>
</web-app>
```

Example

```
<html> HOME.HTML
<body bgcolor="cyan">
<form action="addiv">
<h2>
Number<input type="text" name="t1"><br>
Number<input type="text" name="t2"><br>
<input type="submit" value="div">
</h2>
</form>
</body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Opp. CDAC, Ayyagar Bakery,
Ameerpet, Hyderabad.

Servlet

```
package com.nit.servlets;
import javax.servlet.*;
import java.servlet.annotation.*;
import java.servlet.http.*;
import java.io.*;

@WebServlet("/div")
public class DivServlet extends HttpServlet
{
    public void doGet(HttpServletRequest req,
                       HttpServletResponse res)
        throws ServletException, IOException
    {
        int n1 = Integer.parseInt(req.getParameter("t1"));
        int n2 = Integer.parseInt(req.getParameter("t2"));
        int n3 = n1/n2;
        PrintWriter out = res.getWriter();
        out.println("<h2> Result is " + n3 + "</h2>");
    }
}
```

Web.xml

```
<web-app>
<error-page>
<exception-type>java.lang.ArithmaticException</exception-type>
<location>/a.htm</location>
</error-page>
</web-app>
```

a.html

```
<html>
<body bgcolor=yellow>
<h2> number can't divide with zero </h2>
</body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SRI RAGHAVENDRA XEROX
Software Languages Material Available

JAVA REAL TIME PROJECT :-

Core Technologies :-

- ① JDBC.
- ② Servlets
- ③ JSP.

Tools :-

- ① Log4j
- ② Maven
- ③ SVN
- ④ JUnit

Design Patterns :-

- ① MVC
- ② DAO
- ③ DTO

IDE :-

Eclipse

Database :-

Oracle.

Server

Tomcat or Weblogic.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

MVC

- Mvc stands for Model-view-controller.
- MVC is a design pattern.
- Design patterns provide solution for ~~a~~ given problem.
- This design pattern is used for developing web appliⁿ.
- Web application is divided into 3 layers
 - (1) Presentation Layer
 - (2) Service Layer
 - (3) Persistent Layer.

Presentation Layer (View)

- This layer consists of programs which are used to give input and generate output.
- This layer is developed using the following technologies
 - (1) HTML
 - (2) JSP (we are using JSP in our project)
 - (3) JSF
 - (4) Applet
 - (5) Java FX.
- JSP uses the following programs
 - (1) AJAX
 - (2) Java Script.

Persistent Layer (Model) :-

- This layer consists of programs which communicate with database.
- These programs perform persistent operations.
- DAO (Data Access Objects)
- DAO uses JDBC.

Service Layer (controller)

- controller is a program which perform action based on client request.
- controller acts as interface b/w Model and View.
- controller receive request from client perform action using other resources exist in web application.
→ Servlet

DAO:-

- DAO stands for Data Access Object.
- DAO is a Design pattern.
- DAO is used to perform persistent operations like insert, update, delete and select.
- DAO uses JDBC in order to communicate with DB.

Advantages of DAO :-

1. Loosely coupled
2. Reusability.

How to develop DAO ?

To develop DAO we required 3 things

1. DAO Interface
2. DAO Implementation
3. DAO Factory.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

1. DAO Interface :-

- This Interface define the operations performed on data.
- data exist in database in the form of table.

ex:- interface UserDAO1

(I → Interface)

```
{  
    int registerUser (UserBean u);  
    int changePassword (UserBean u);  
    boolean findUser (UserBean u);  
}
```

DTO (Data Transfer Object)

- It is a value object.
- This object doesn't have Business Logic.

Rules for developing DTO

- (i) It must be Serializable.
- (ii) Define properties.
- (iii) For each property define Setter and getter methods.

(Serializable is an interface which is available in java.io package)

Example:-

```
class UserBean implements Serializable.  
{  
    private String name;  
    private String rename;  
    private String pwd;  
  
    // setter Method  
    public void setName (String name)  
    {  
        this.name = name;  
    }  
  
    // getter Method  
    public String getName()  
    {  
        return name;  
    }  
}
```

→ (Hence we can write more setter and getter Method as per our requirement.)

2. DAO implementation :-

→ It is a class which provide implementations of DAO interface.

DBUtil

→ It is a database utility class, which create connection and return.

developing DBUtil class :-

```
class DBUtil
```

```
{
```

```
    private static Connection cn;
```

```
    static
```

```
    {  
        try
```

```
            Class.forName("oracle.jdbc.driver.OracleDriver");
```

```
}
```

```
    catch (Exception k)
```

```
{
```

```
    } // static block close
```

```
    public static Connection getConnection()
```

```
{
```

```
    try
```

```
        cn = DriverManager.getConnection("url", "user",  
                                         "pwd");
```

```
}
```

```
    catch (Exception k)
```

```
{
```

```
}
```

```
    return cn;
```

```
}
```

↳ close static block (close connection) method
↳ close class

Implementation class

```
class UserDaoImp implements UserDao
{
    private Connection cn;
    UserDaoImp()
    {
        cn = DBUtil.getConnection();
    }
}
```

DaoFactory:-

→ This class provide static methods which create DAO objects and return reference.

```
class Daofactory
{
    public static Object getDao (String dao)
    {
        Object o = null;
        try
        {
            Class c = Class.forName(dao);
            o = c.newInstance();
        }
        catch (Exception k)
        {
            k.printStackTrace();
        }
        return o;
    }
}
```

```
Servlet
UserDao ud =
(UserDao) Datafactory.
getDao("UserDao");
```

AJAX

- AJAX stands for Asynchronous JavaScript And XML.
- AJAX is a technique for creating fast and dynamic web pages.
- AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.

Create an XMLHttpRequest object :-

- The keystone of AJAX is the XMLHttpRequest obj.
- The XMLHttpRequest object is used to exchange data with a server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.
- All modern browsers (chrome, IE7+, firefox, safari, and opera) have a built in XMLHttpRequest object.
- Syntax for creating an XMLHttpRequest object

```
variable = new XMLHttpRequest();
```

Verify browser support AJAX or not

```
var xhttp = new XMLHttpRequest();
if (window.XMLHttpRequest) {
    xhttp = new XMLHttpRequest();
}
```

Methods of XMLHttpRequest

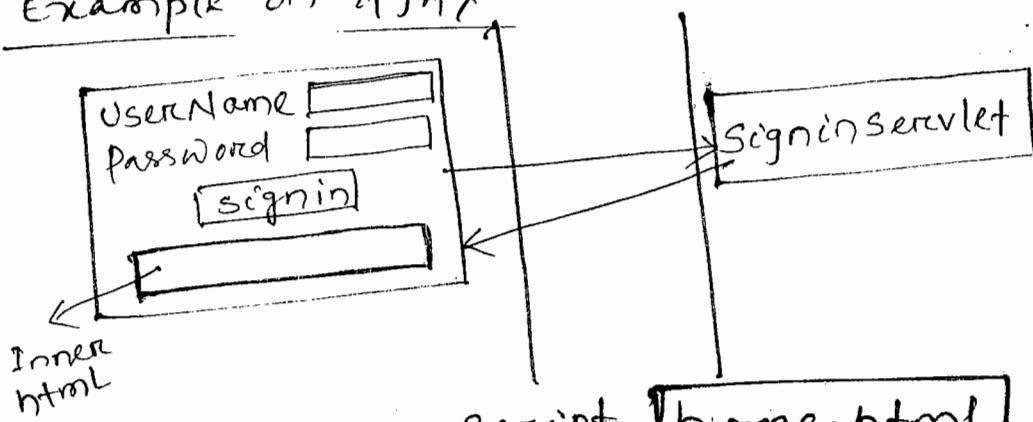
① `open (method, url, async)`

↓
Request
Method ↓
Reg. url
 ↓
 true/false

when using `async = true`, specify a function to execute when the response is ready in the `onreadystatechange` event.

② `Send()`

Example on AJAX



home.html

```

<html>
  <head>
    <script>
      function sendRequest() {
        var r = new XMLHttpRequest();
        r.onreadystatechange = function() {
          if (r.readyState == 4 && r.status == 200)
            document.getElementById("msg").innerHTML = r.responseText;
        }
      }
    </script>
  </head>
  <body>
    <form>
      <input type="text" name="username">
      <input type="password" name="password">
      <input type="button" value="signin" onclick="sendRequest()">
    </form>
  </body>
</html>

```

ajaxapplication

```

home.html
WEB-INF
  classes
    signinServlet

```

SigninServlet

```

    r.open("GET", "signin?u=" + document.t1.t1.value)
    & p = "document.t1.t2.value.tostring();"
    r.send();
}
// Function closed

</script>
</head>
<body bgcolor="cyan">
<form name="f1">
<h2>
  username <input type="text" name="t1"> <br>
  password <input type="password" name="t2"> <br>
  <input type="button" value="signin"
         onclick="sendRequest()"/>
  <p id="msg"></p>
</h2>
</form>
</body>
</html>

```

SigninServlet :-

```

package com.nit.servlets;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
@WebServlet("/signin")
public class SigninServlet extends HttpServlet
{

```

SPI RAGHAVENDRA XEROX
 Software Languages Material Available
 5-512, Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

```
public void doGet (HttpServletRequest req,  
                   HttpServletResponse res)
```

```
throws ServletException, IOException.
```

```
{
```

```
    String u = req.getParameter("u");
```

```
    String p = req.getParameter("p");
```

```
    PrintWriter out = res.getWriter();
```

```
    if (u.equals("nit") && p.equals("nit"))
```

```
        out.println("<h2> valid username and pwd</h2>");
```

```
    else
```

```
        out.println("<h2> Invalid username & pwd</h2>");
```

```
}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

JSP (Java Server Page)

Q) what is MVC?

- MVC stands for Model - view - controller.
- It is a design pattern used for developing web applications.
- Webapplication Logic is divided into 3 Logics
 1. presentation Logic.
 2. Business Logic.
 3. Persistent Logic.
- Each one is called Module or tier.

Q) what is Model?

- It is a module or tier which contains persistent logic or database operation.
- Model uses JDBC, Hibernate, JPA.

Q) what is controller?

- controller, control flow of execution of webapp!
- In webapplication development servlet is called controller.
- controllers receive request/input from client and uses other Modules to perform operations.
- They contain business logic.

Q) what is view?

- View is used for generating input and output.
- It contains presentation Logic.
- This is developed using HTML or JSP.

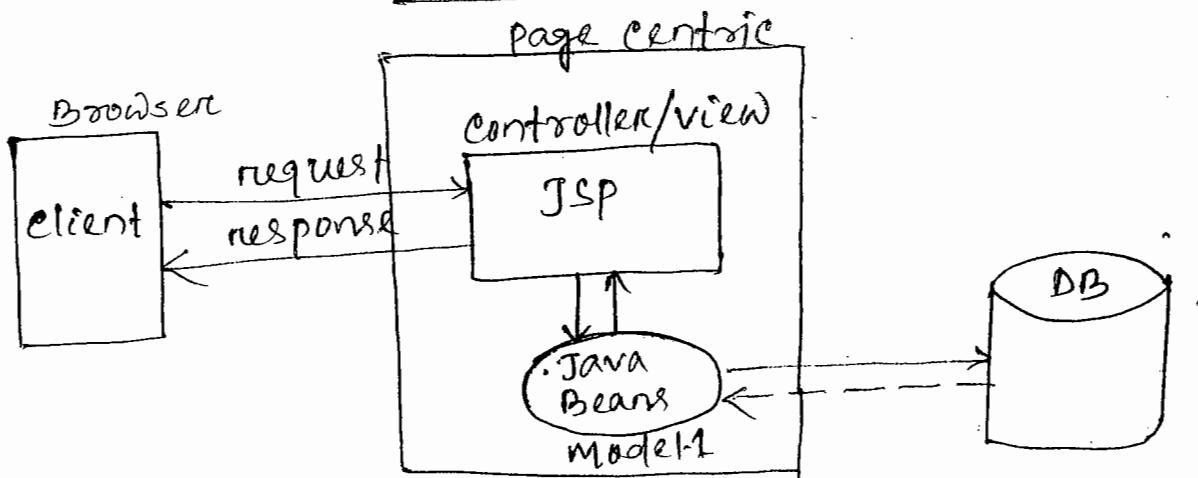
→ MVC is having 2 Model

(i) MVC Model 1 Architecture.

(ii) MVC Model 2 Architecture.

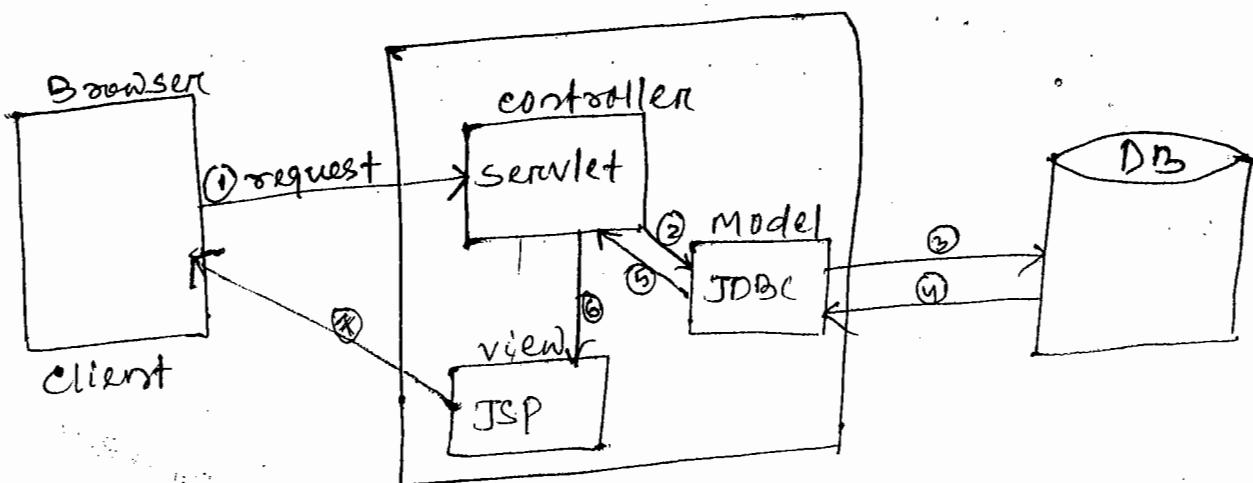
Q What's difference between MVC-Model 1 and MVC-Model 2 Architecture.

MVC-Model 1



MVC Model-2

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.



MVC - I

- (i) MVC 1 associates the presentation logic with the business logic.
- (ii) In MVC1 only one component is responsible for receiving request and sending response.
- (iii) In MVC1 Business Logic and persistent presentation logics combined so web designer and web developer can't work simultaneously.
- (iv) Doesn't support reusability of application component.
- (v) In MVC controller and Model both are JSP.
- (vi) In MVC1 there is a tight coupling betⁿ page and Model as data access is usually done using custom tag or through java bean call.

MVC - 2

- (i) MVC-2 isolates or disassociates the presentation logic from business logic
- (ii) In MVC-2 there's separate components for receiving and sending response i.e controller and view.
- (iii) since both logic are separate that's why designer and developer can work together.
- (iv) Reusability of components.
- (v) while controller is servlet and Model is Java class.
- (vi) In MVC-2 architecture there is only one controller which receives all the request for the application and is responsible for taking appropriate action in response to each request.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

03.12.15 :-

Q. what is JSP ?

- JSP stands for Java Server Pages
- JSP technology build on Java
- JSP or Java server pages, was developed by Sun Microsystem. JSP technology is object oriented programming language and is based on Java Language.
- JSP is a part of JEE. JSP is a web technology used for developing dynamic websites.

Advantages of JSP / Features of JSP :-

(1) simplifies the process of development :-

- It allows programmers to insert the Java code directly into the JSP file, making the development process easier. Although JSP files are HTML files, they use special tags containing the Java source code which provides a dynamic ability.

(2) portability :-

The Java feature of 'write once, run anywhere' is applicable to JSP. JSP is platform independent making it portable across any platform and therefore multi platform.

(3) Independency of layers :-

- There is a clear separation between presentation and implementation layer.

- Q. why do I need JSP Technology if I already have Servlets?
 → JSP technology was designed simplify the process of creating pages by separating Web presentation from web content.
 → JSP is collection of html tags and JSP tags and Java code.
 → Every JSP page is saved with ext.jsp.
 → this JSP can be placed in root folder or WEB-INF.
 → The JSP placed in root folder is public resource and this resource can access by client directly.
 → the JSP placed inside WEB-INF is private resource, which can't access directly by client.
 client access this resource using public url defined inside Web.xml.

Web.xml

```

<Web-app>
  <Servlet>
    <Servlet-name> instanceName </Servlet-name>
    <jsp-file> JSP filename </jsp-file>
    <init-param>
      <param-name> name </param-name>
      <param-value> value </param-value>
    </init-param>
    :
    </Servlet>
  <Servlet-Mapping>
    <Servlet-name> instanceName </Servlet-name>
    <url-pattern> /public url </url-pattern>
  </Servlet-Mapping>
</Web-app>
  
```

JSP

```

<html>
<body bgcolor = cyan>
<h2>
This is my first JSP
</h2>
</body>
</html>

```

→ Save it JSP1.jsp

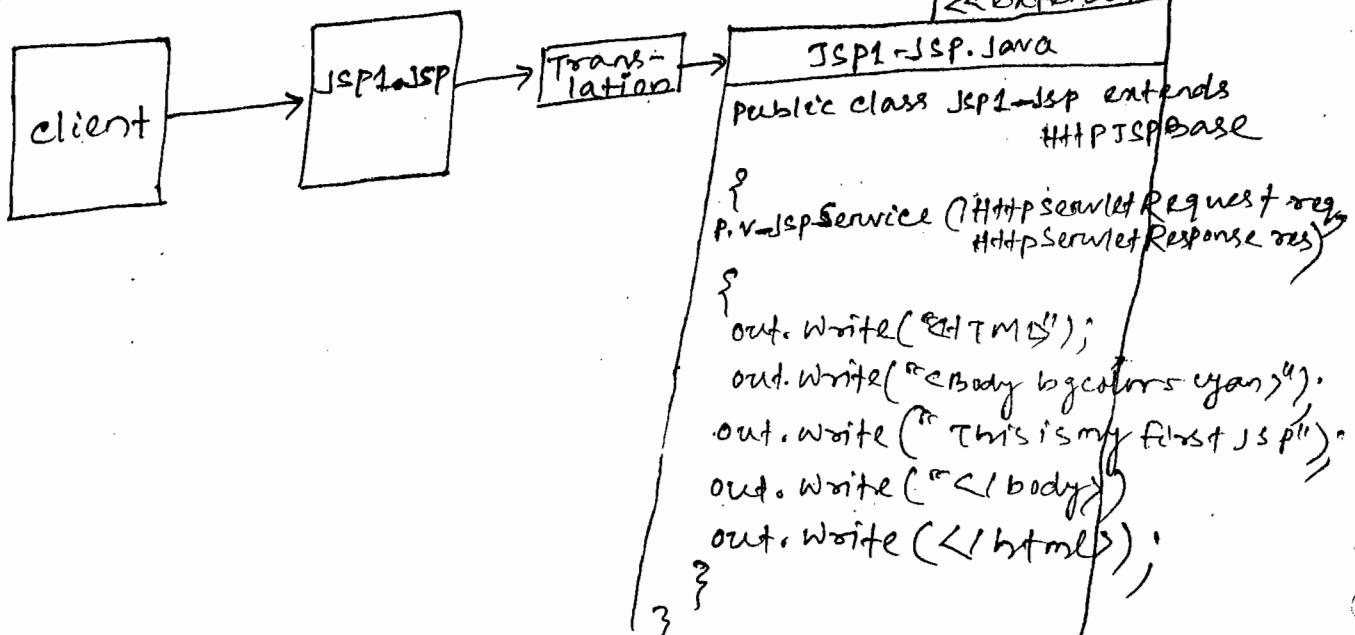
JSP Lifecycle phases :-

1. Translation
2. compilation
3. Loading the class.
4. Instantiating the class
5. Initialization
6. placing into Service
7. destroy.

1. Translation :-

→ when JSP receive first request from client,
JSP container translate JSP into Servlet.

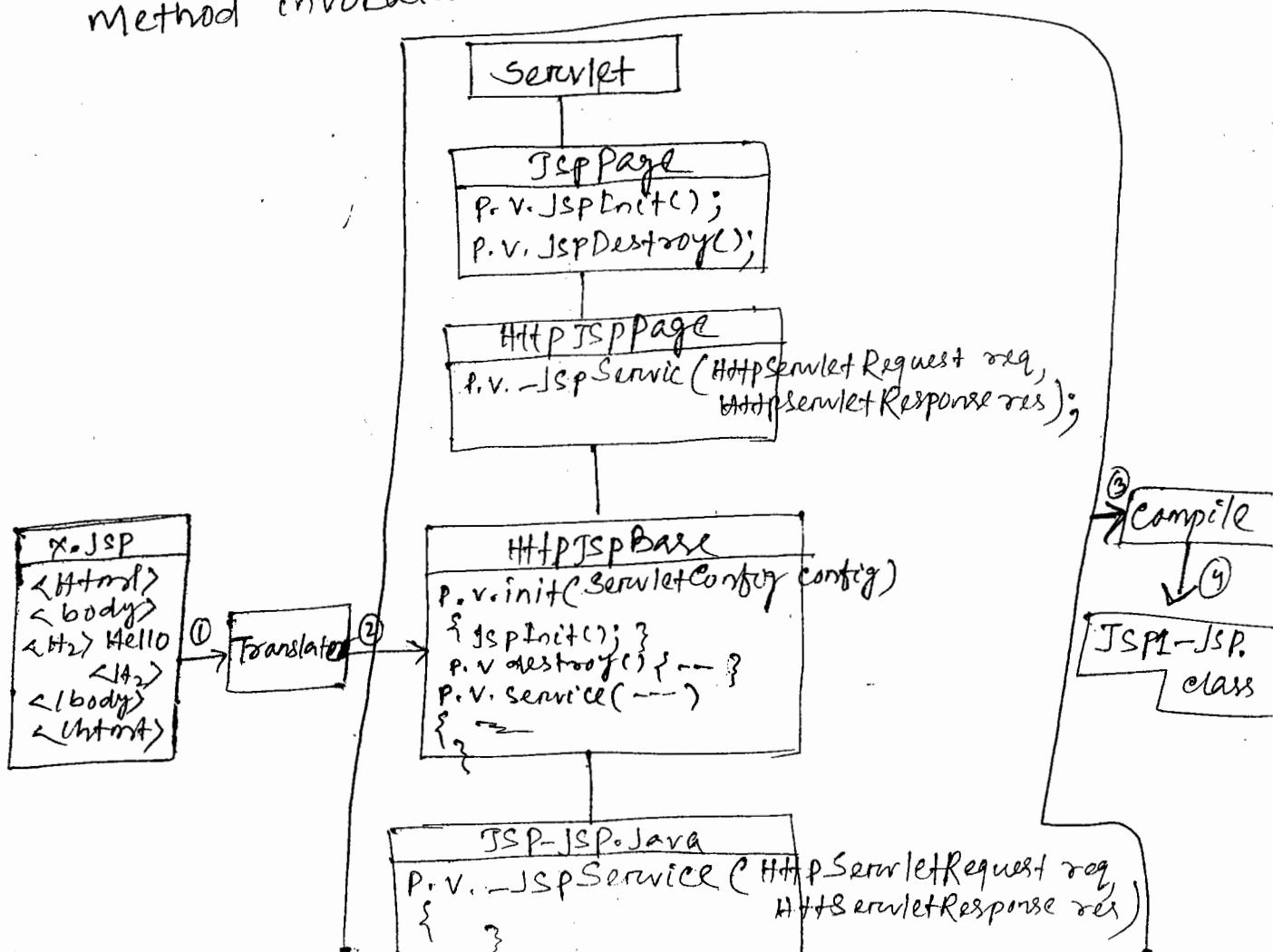
• JSP → .Java



① Translation

JSPPage

- JSPPage JSPPage is an interface
- The JSPPage interface describes the generic interaction that a JSP page implementation class must satisfy.
- This interface provides 2 Methods
 - (1) `void JspDestroy()`
 - The JspDestroy() method is invoked when the JSPPage is about to be destroyed.
 - (2) `void JspInit()`
 - The JspInit() method is invoked when the JSP page is initialized.
- A class implementing this interface is responsible for invoking the above methods at the appropriate time based on the corresponding servlet-based method invocation.



HttpJspPage :-

→ The HttpJspPage interface describes the interaction that a JSP page implementation class must satisfy when using the HTTP protocol.

→ It provides 1 Method

Void -JspService(HttpServletRequest request,
HttpServletResponse response)

→ The -JspService() method corresponds to the body of the JSP.

② compiling

→ Source program is compiled and generate byte code or .class file.

③ Loading :-

compiled .class file is loaded into JVM.

④ Instantiating the class :-

→ Create an object of Loaded class.

⑤ Initialization :-

JSP is initialized by calling jspInit() method.

⑥ Placing into service :-

JSP is placed into service by calling -JspService Method by sending HttpServletRequest and HttpServletResponse object.

⑦ Removing from Service :-

Before removing JSP from service container calls JSⁿ JspDestroy Method.

JSP Tags

→ JSP Tags are classified into 3 categories

1. scripting elements
2. directives
3. standard actions

Scripting elements :-

These tags are used for inserting java code inside JSP.

3 types of scripting tags:-

1. declaration tag.
2. expression tag.
3. Scriptlet.

1. declaration tag:-

declaration tag is used for declaring variables and methods.

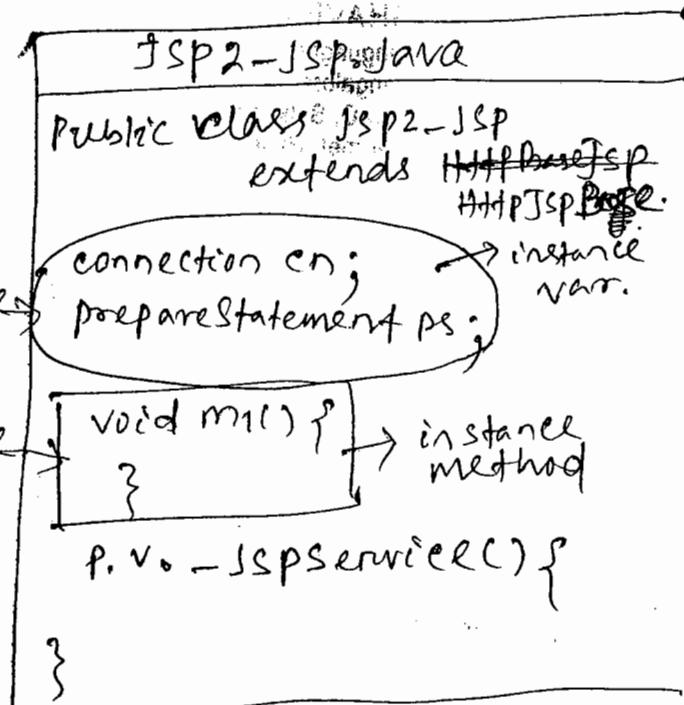
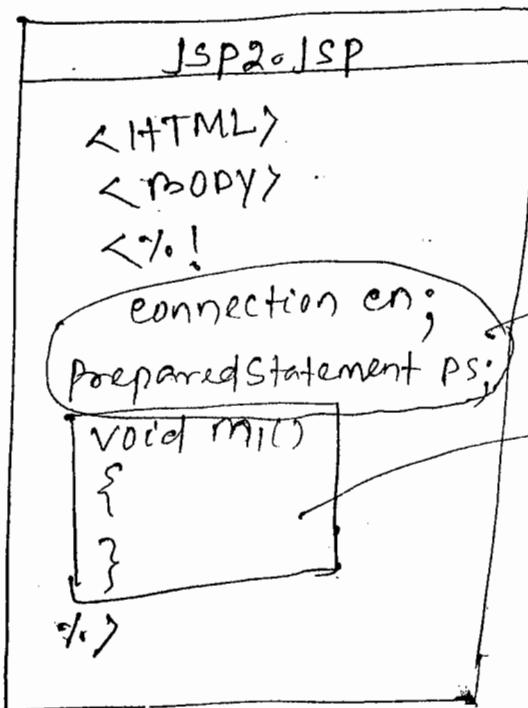
Syntax :-

<%!

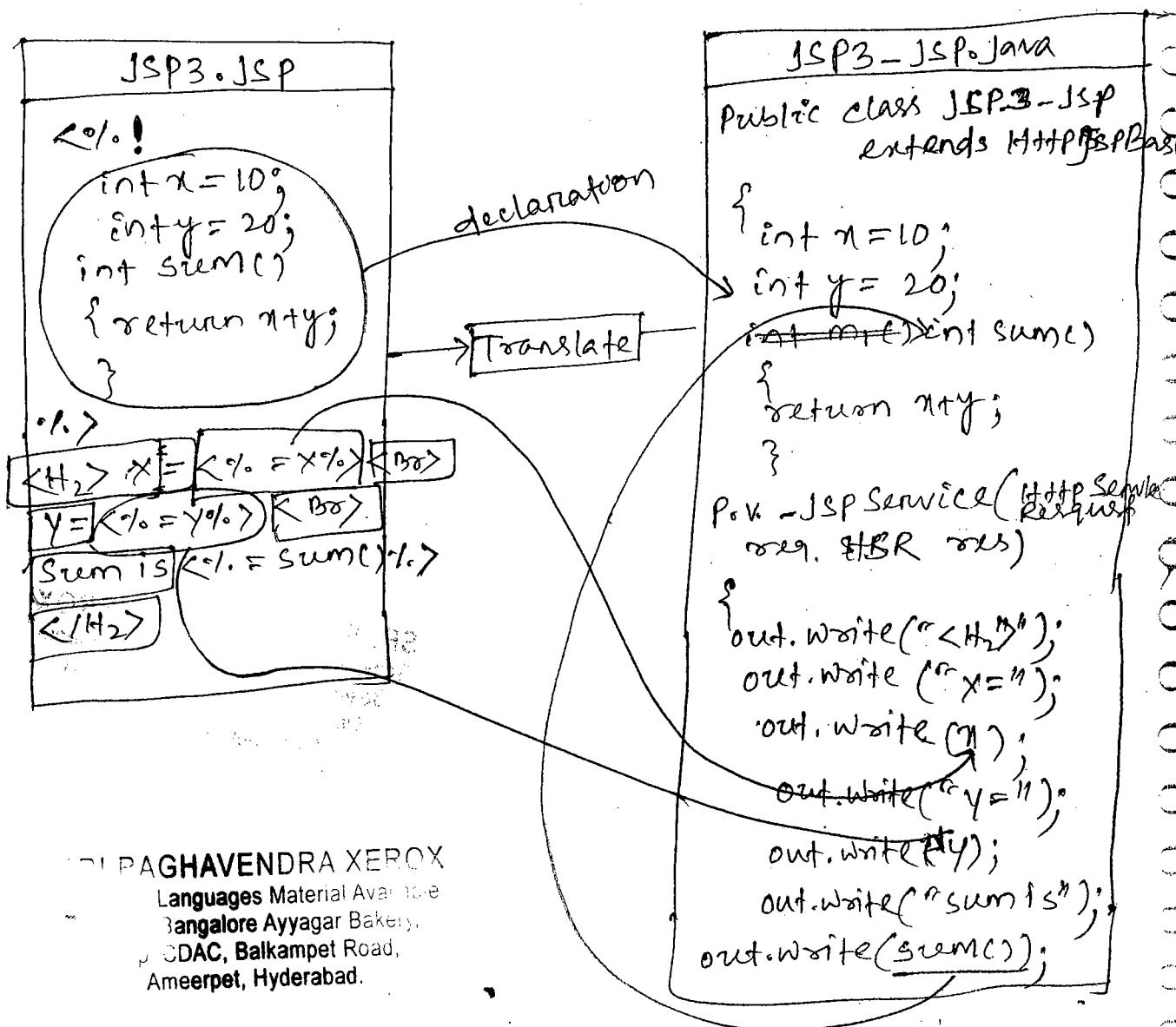
declaring variables;
defining methods;

%>

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.



2. expression tag :-



→ statements written inside expression tag include within service method. This expression included as part of response body. It is eval and send to client browser.

Syntax

`<% = expr %>`

05.12.2015

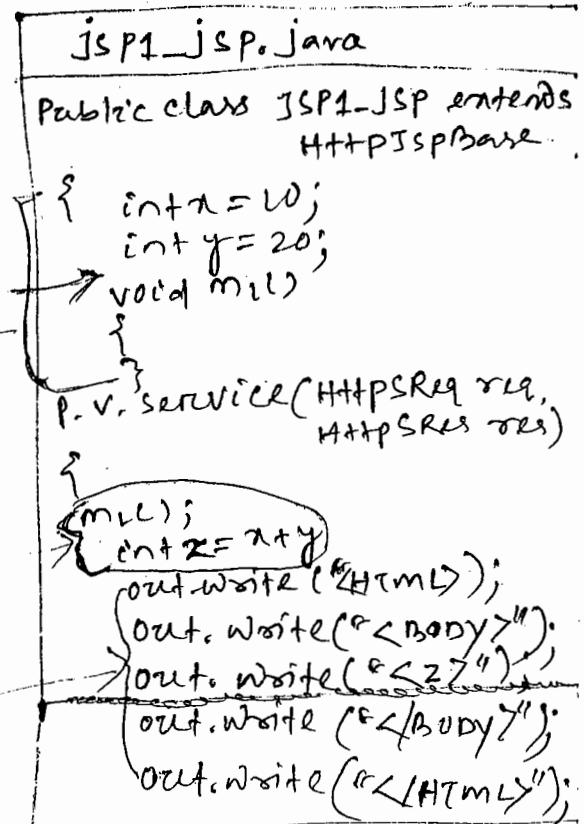
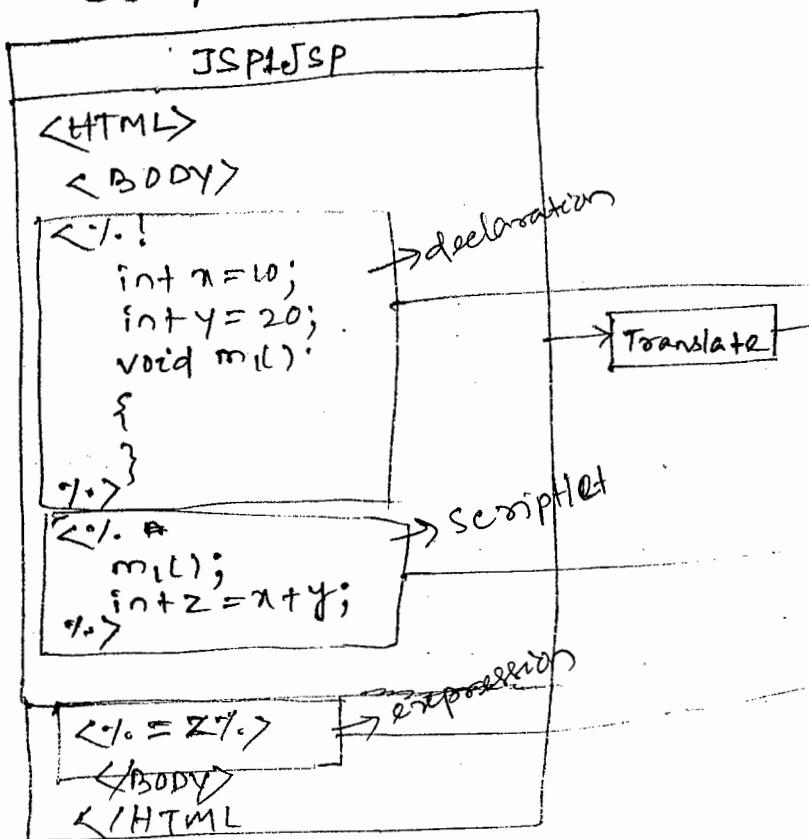
③ scriptlet :-

<%.

Statement 1;
Statement 2;
Statement 3;

%>

- These statements are included within service method.
- All executable statements are included within scriptlet.



How to initialize JSP:-

- JSP page is initialized by calling JSPInit() Method.
- Any code which has to be executed after/on creating object defined inside JSPInit() Method.
- We can write a constructor inside JSP.

How to Write comments in JSP:-

→ There are 2 types of comment (i) output comment.
 (ii) Hidden comment.

Q. what is output comment?

→ A comment that is sent to the client in the viewable page source. The JSP engine handles an output comment as uninterpreted HTML text, returning the comment in the HTML output sent to the client. You can see the comment by viewing the page source from your web browser.

Syntax :-

<%-- Comment --%>

Q. what is Hidden comment?

→ Comment that documents the JSP page but is not sent to the client. The JSP engine ignores a hidden comment, and does not process any code within hidden comment tags.

→ A hidden comment is not sent to the client, either in the displayed JSP page or the HTML page source. The hidden comment is useful when you want to hide or "comment out" part of your JSP page.

Syntax :-

<%-- Comment --%>

JSP1.jsp

```
<html>
<body>
<!-- This is my first JSP -->
<%-- This is first JSP--%>
</body>
</html>
```

output comment
Hidden comment

Translate

JSP1-JSP.java

```
public class JSP1-JSP extends HttpServlet
{
    out.write("<html>");
    out.write("<body>");
    out.write("<!-- This is my first JSP -->");
    out.write("</body>");
    out.write("</html>");
```

JSP application 2

```
<html>
<body bgcolor=cyan>
<%!
    java.sql.Connection cn;
    java.sql.Statement st;
    java.sql.ResultSet rs;
    public void jspInit()
    {
        try{
            class.forName("oracle.jdbc.driver.OracleDriver");
            cn = java.sql.DriverManager.getConnection
                ("jdbc:oracle:thin:@localhost:1521:XE", "system",
                "cayan");
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }
%>
```

RAGHAVENDRA XEROX
Software Languages Material A
Beside Bangalore Ayyagar Bus Stand
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

① <%!
    try {
        st = cn.createStatement();
        rs = st.executeQuery("select empno, ename, sal
                            from emp");
        while (rs.next())
            {
%> ② <%= rs.getInt(1) %>; <%= rs.getString(2) %>;
        <%= rs.getFloat(3) %> <br>
    } <%
}
    catch (Exception p)
    {
        p.printStackTrace();
    }
③ <%>
</body>
</html>

```

response
 bcz written
 in expression
 tag.

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Material Available
 Opp. CDAC, Balkampet Ayyagar Bakery,
 Ameerpet, Hyderabad.

Listemp.jsp

```
<html>
<body>
<%!
    java.sql.Connection cn;
    java.sql.Statement st;
    java.sql.ResultSet rs;
%>


v. JSPInit()


{
try {
    class.forName("oracle.jdbc.driver.OracleDriver");
    cn=DM.getConnection("","","");
    catch (Exception k)
    {
        k.printStackTrace();
    }
    %>
    try {
        st=cn.createStatement();
        rs=st.executeQuery("select empno,ename,sal from emp");
        while (rs.next())
        {
            %>
            <%=rs.getInt(1)%>%>
            <%=rs.getString(2)%>
            <%=rs.getFloat(3)%>
        }
    }
    catch (Exception p)
    {
        p.printStackTrace();
    }
%>
```

Listemp-JSP.java

```
p. class Listemp_JSP extends HttpServlet
{
    java.sql.Connection cn;
    java.sql.Statement st;
    java.sql.ResultSet rs;
    p.v. JSPInit()
    {
try {
    class.forName("oracle.jdbc.driver.OracleDriver");
    cn=DM.getConnection("","","");
    catch (Exception k) { }
    p.v. service (HttpServletRequest req,
                  HttpServletResponse res);
    {
try {
        st=cn.createStatement();
        rs=st.executeQuery("select empno,ename,sal
                           from emp");
        while (rs.next())
        {
            out.print
            out.write(rs.getInt(1));
            out.write(rs.getString(2));
            out.write(rs.getFloat(3));
        }
    }
    catch (Exception p)
    {
        p.printStackTrace();
    }
}
```

② Directives : Tag:-

Q. What is directives?

→ Directive are basically used to configure the code that is generated by container in a servlet. As a part of JSP we have three types of Directives.

(1) page directives

(2) include directives.

(3) taglib directives

(1) page directives :-

→ Page directives are basically used for supplying compile time information to the container for generating a servlet. The page directive will take the data in the form of (key, value) pair.

Syntax :-

<%@

page attribute_name1 = attribute_value1,
attribute_name2 = attribute_value2,

~~Attribute name~~

Attribute nameN = attribute_valueN %>

→ whenever we use page directive as a part of JSP program that statement must be the first statement.

→ The scope of page directive is applicable to current JSP page only.

→ JSP program allows one page directive.

Error during runtime

```
JSP1.jsp
<%@page .....%>
<%@page .....%>X
```

```
JSP1.jsp
<%!
statements;
%>
<%@page .....%>X
```

2) import :-

This attribute is used for importing either pre-defined or userdefined packages.
The default value is java.lang.*

```
JSP1.jsp
<%@page import=
"java.util.*;
java.sql.*"%>
<%!
Date d = new Date();
%>
<H1> Date and Time
<%=d%> </H1>
```

Translate

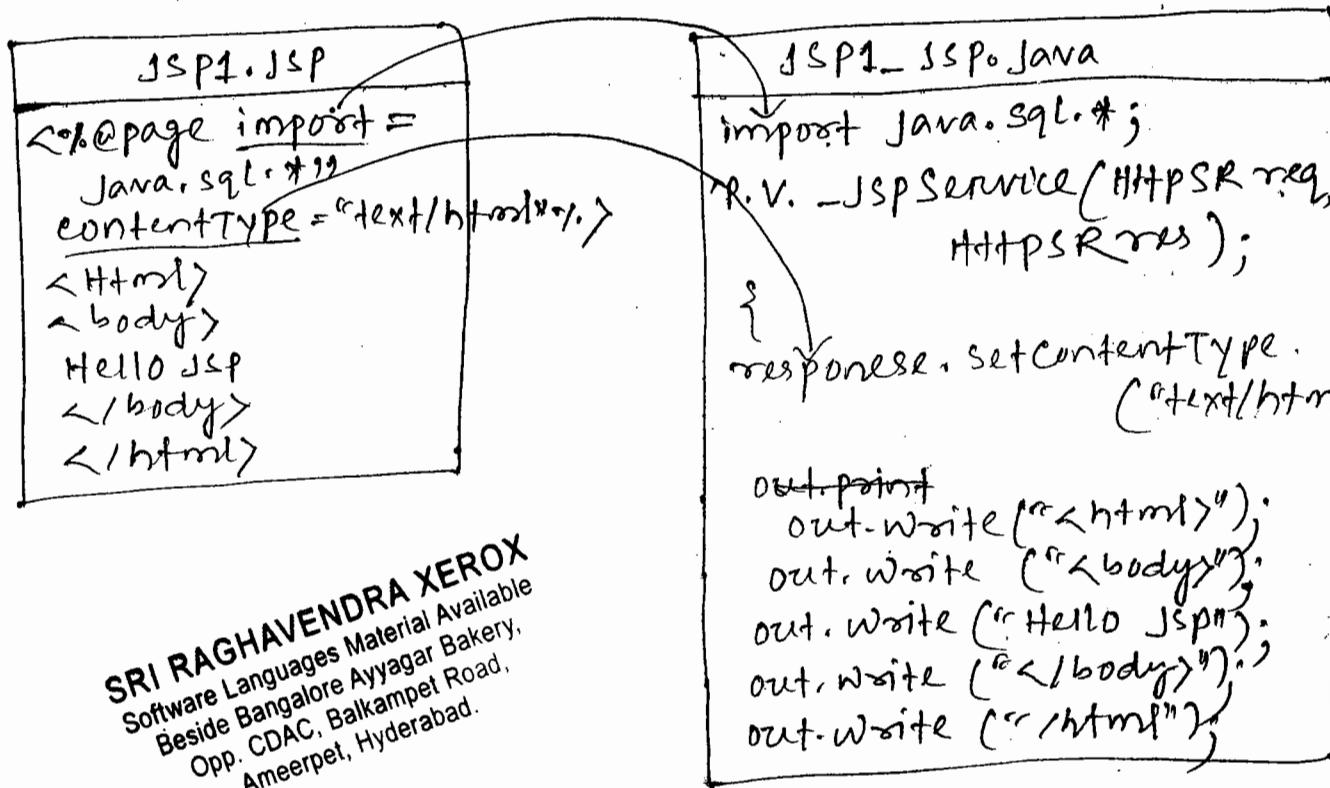
```
JSP1.jsp.java
import java.util.*;
import java.sql.*;
p. class JSP1_jsp extends
HttpBaseJSPPhase
{
    Date d = new Date();
    p. v. service (HTTPSR req,
HTTPSR res)
    {
        out.write ("<H1>");
        out.write ("Date and Time");
        out.write (d);
    }
}
```

compile

```
JSP1-jsp.class
p. class JSP1_jsp extends
HttpJSPBase
{
    Java.util.Date d = new Date();
}
```

2) Content Type:-

The attribute is used for setting the MIME type (plain/text, img/jpeg, img/gif, audio/wave; etc.). The default value is text/html.



3) Language:-

This attribute represents by default Java i.e. in order to represent any business logic a JSP program is making use of Java language. The attribute language can support any of the other programming languages for developing a business logic at server side.

Syntax:-

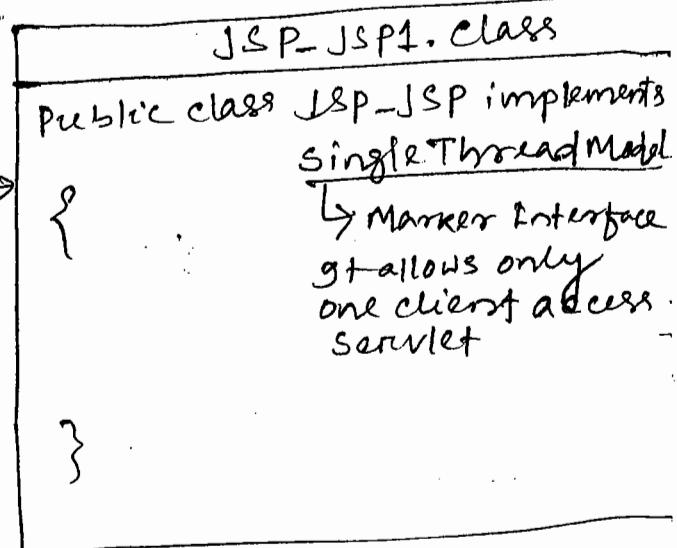
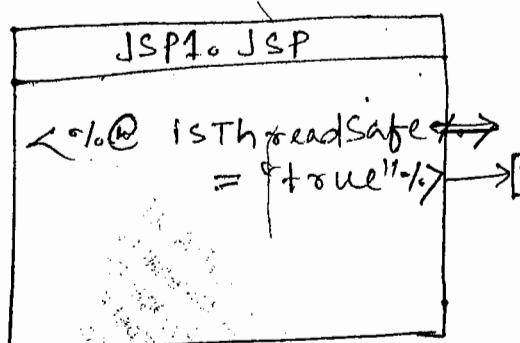
```
<%@ page language = "Java" %>
```

4) isThreadSafe :-

This attribute represents by default ~~false~~ ^{false} which represents one server side resource can be accessed by many number of clients (each client is treated as one thread)

Syntax :-

<%@ page isThreadSafe = "true" %>



5) isErrorPage

When we write n' number of JSP

pages, there is a possibility of occurring exceptions in each and every JSP page.

It is not recommended for the JSP programmer to write try and catch blocks in each and every JSP page. It is always recommended to handle all the exceptions in a single JSP page.

→ isErrorPage is an attribute whose default value is true which indicates exceptions to be processed in the same JSP page which is not recommended.

→ If isErrorPage is false then exceptions are not processed as a part of current JSP page and the exceptions are processed

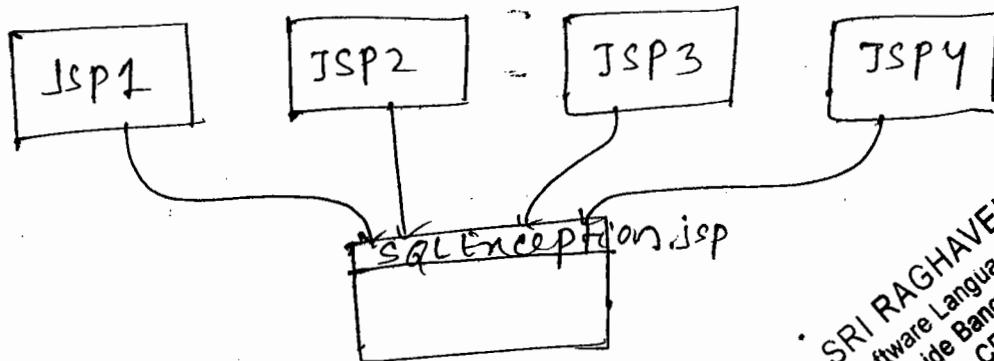
in some other JSP page which will be specified through an attribute called `& errorPage`.

Ex:-

```
<%@Page isErrorPage="true"%>
```

```
<%@Page isErrorPage="false"
```

```
errorPage="---"%>
```



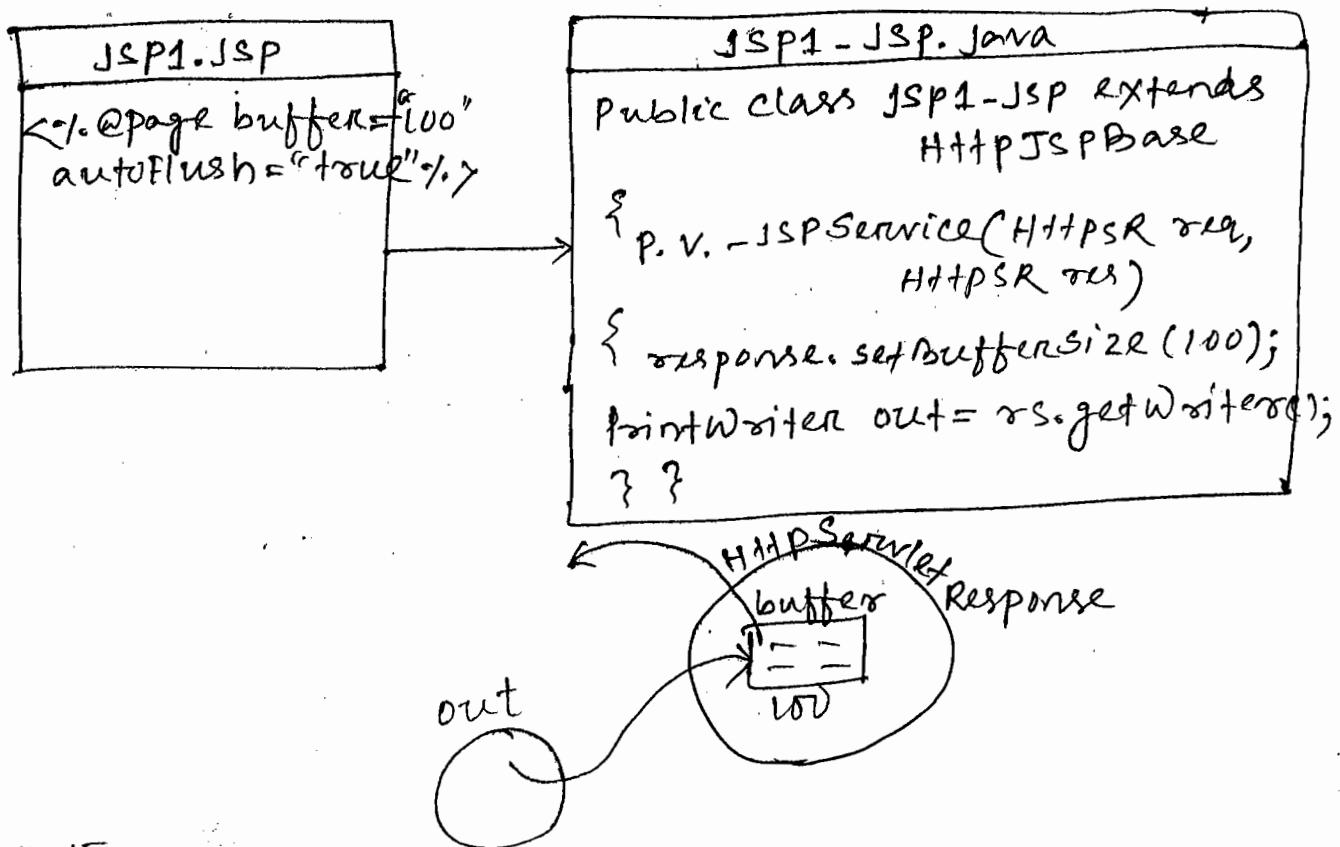
SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

7) autoflush

8) buffer

whenever the server side program want to send large amount of data to a client it is recommended to make autoflush value as false and we must specify the size of the buffer in terms of kb.

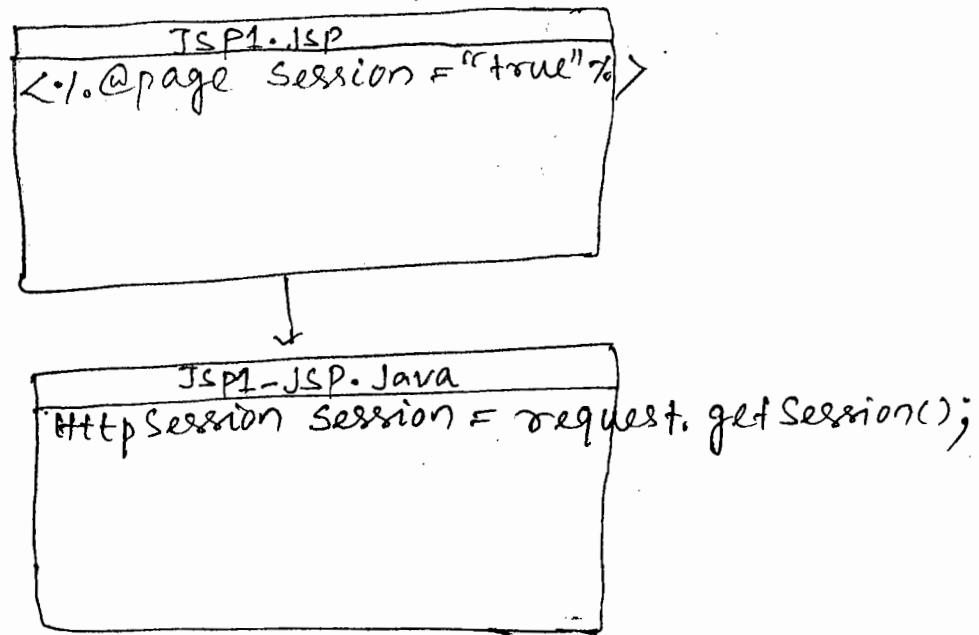
→ The default value of autoflush is true which represent the server side program gives the response back to the client each.



08.12.15

9) Session :-

when we want to make 'n' number of independent request as consecutive requests one must use the concept of Session. In order to maintain the session we must give the value of the session attribute has true in each and every JSP page (recommended). The default value of session is true which represent the session is applicable to current JSP.



Jsp implicit objects :-

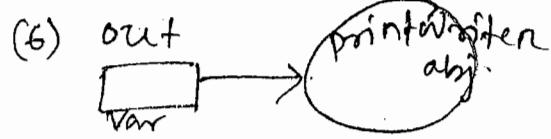
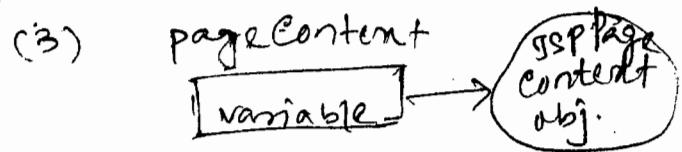
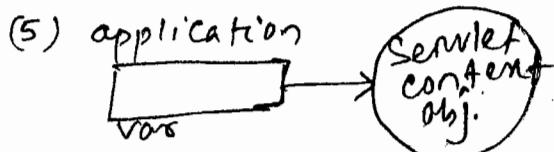
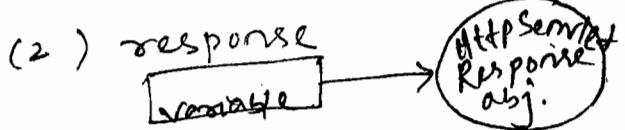
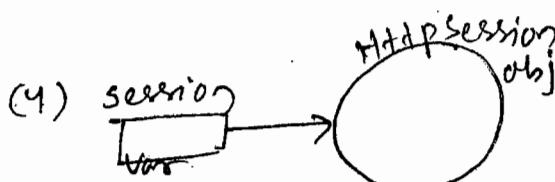
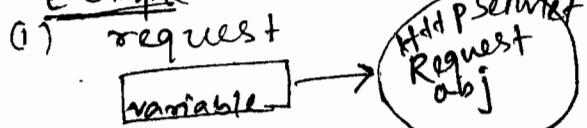
(Q) what are the implicit objects in JSP?

- Implicit objects in JSP are the Java objects that the JSP container makes available to developers in each page.
- ~~The ab~~ These objects are not created by the developer.
- These objects need not be declared or instantiated by the JSP author. They are automatically instantiated by the container and are accessed by using standard variables; hence they are called implicit objects.

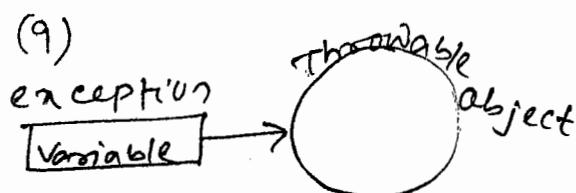
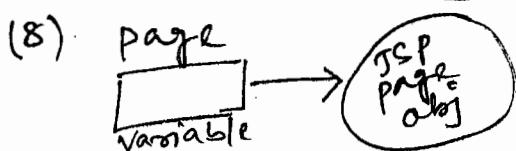
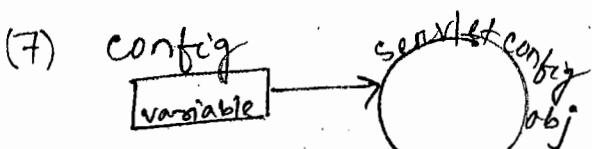
The implicit objects are:-

- (1) request
- (2) response
- (3) pageContext
- (4) session
- (5) application
- (6) out
- (7) config
- (8) page
- (9) exception.

Example

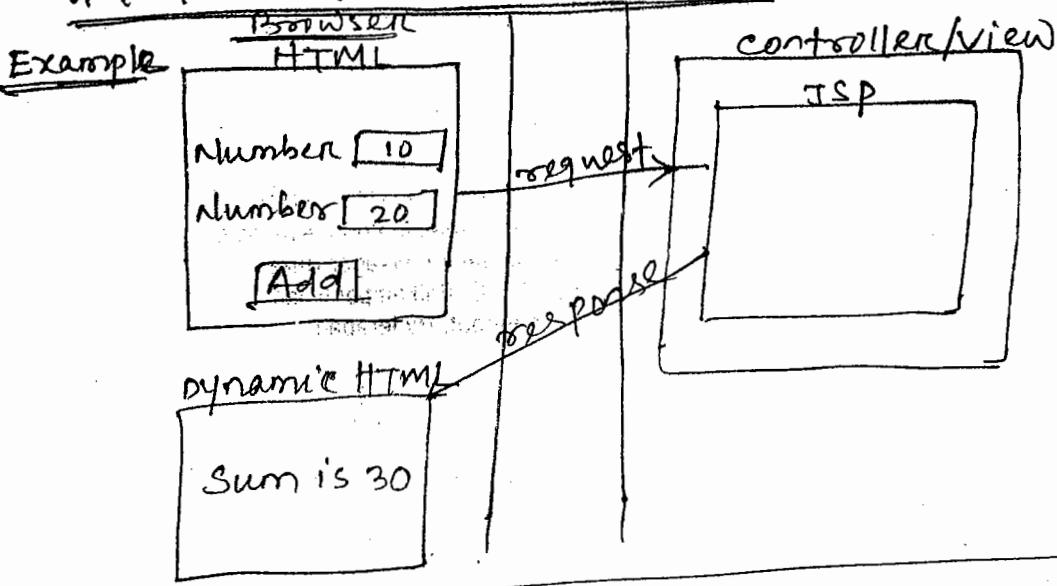


SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.



- The implicit objects are parsed by the container and inserted into the generated servlet code.
- They are available only within the JSP service method and not in any declaration.

HTML - JSP communication :-



SRI R&L
4 VENDRA XEROX
Sri Venkateswara Languages Material Available
Beside Bangalore Opp. CDAC, Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad

home.html

```

<html>
<body bgcolor="cyan">
<form action=".\\add.jsp">
<b><input type="text" name="t1"></b>
<b><input type="text" name="t2"></b>
<input type="submit" value="Add">
</form>
</body>
</html>

```

add.jsp

```
<html>
<body bgcolor="yellow">
```

```
<h2>
```

```
<%>
```

```
int n1, n2, n3; } Declaration
```

```
<%>
```

```
n1 = request.Integer.parseInt(request.getParameter("t1"));
```

```
n2 = Integer.parseInt(request.getParameter("t2"));
```

```
n3 = n1 + n2;
```

```
<%>
```

```
Sum is <%= n3 %>
```

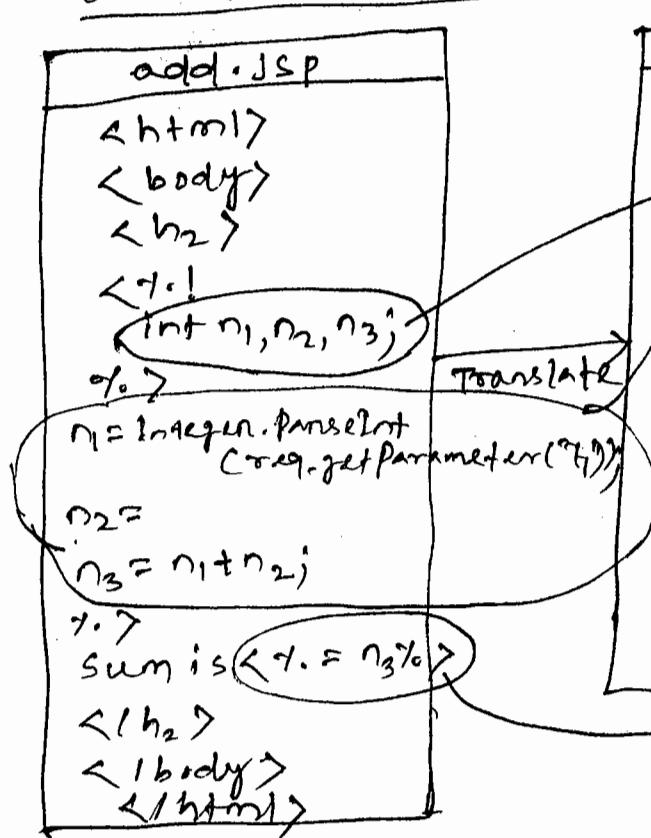
```
</h2>
```

```
</body>
```

```
</html>
```

Execution flow of the above program

RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.



```
add-JSP.java
```

```
public class add-JSP extends HttpServlet
{
    int n1, n2, n3;
    P. V. JSP Service(HttpServletRequest req, HttpServletResponse res)
    {
        n1 = Integer.parseInt(req.getParameter("t1"));
        n2 =
        n3 = n1 + n2;
        out.write("HTML");
        out.write("<body>");
        out.write("<h2>");
        out.write("sum is ");
        out.write(n3);
        out.write("</h2>");
        out.write("</body>");
        compile("2.html");
    }
}
```

- Q. How to define init parameters or values of JSP?
- Initial parameters of JSP defined inside web.xml as config parameters.

Syntax :-

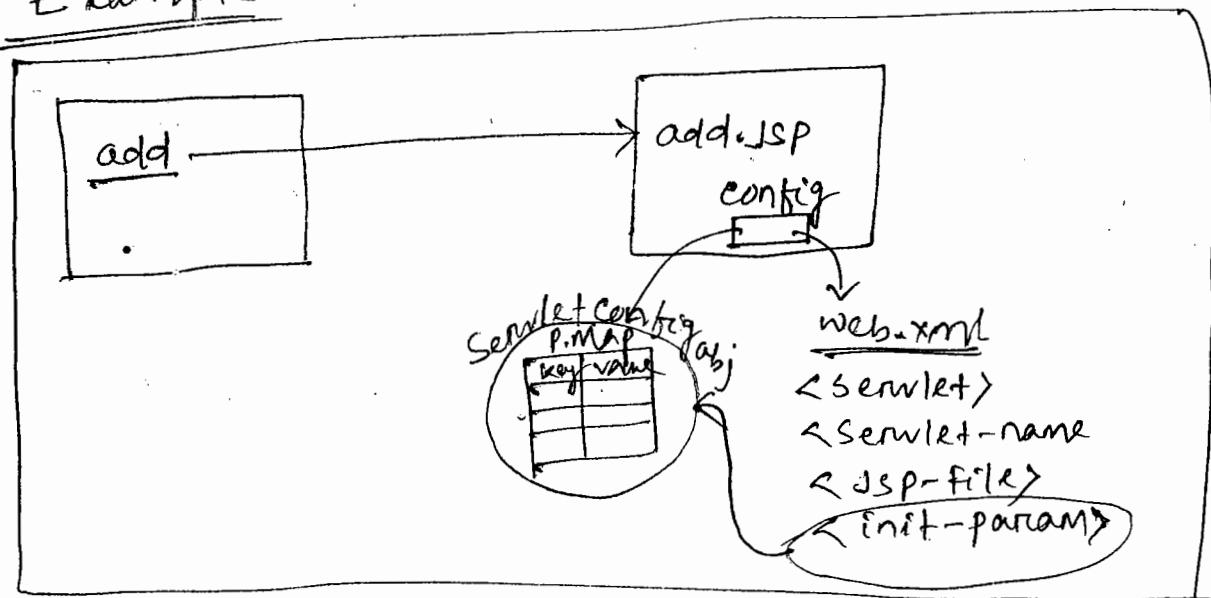
```

<web-app>
  <servlet>
    <servlet-name> instance-name </servlet-name>
    <jsp-file> JSP-File-name </jsp-file>
    <init-param>
      <param-name> name </param-name>
      <param-value> value </param-value>
    </init-param>
    <init-param>
      <param-name> name </param-name>
      <param-value> value </param-value>
    </init-param>
    ...
  </servlet>
</web-app>

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Alayam Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Example



Web.xml :-

```
<web-app>
  <servlet>
    <servlet-name> s1 </servlet-name>
    <jsp-file> /add.jsp </jsp-file>
      ↪ init
      <init-param>
        <param-name> n1 </param-name>
        <param-value> 10 </param-value>
      </init-param>
      <init-param>
        <param-name> n2 </param-name>
        <param-value> 20 </param-value>
      </init-param>
    </servlet>
    <servlet-mapping>
      <servlet-name> s1 </servlet-name>
      <servlet-type>
        <url-pattern> /add </url-pattern>
      </servlet-mapping>
    </web-app>
```

home.html :

```
<html>
  <body background=cyan>
    <h2>
      <a href="http://localhost:8086/JSPApplication4/add"> Add </a>
    </h2>
  </body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Add . JSP

<%!

```
int n1, n2, n3;
```

```
%>
```

<%

```
n1 = Integer.parseInt(config.getParameter("n1"));
```

```
n2 = Integer.parseInt(config.getParameter("n2"));
```

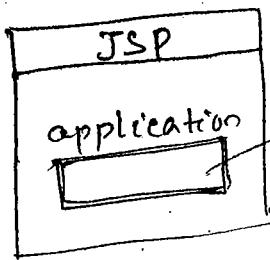
```
n3 = n1 + n2;
```

```
%>
```

</h2> Sum is <%= n3 %> </h2>

09.12.15

JSP Project



(Access context-param in JSP)

Web.xml

<web-app>

<context-param>

<p-n> P1 </p-n>

<p-v> 10 </p-v>

</context-param>

<context-param>

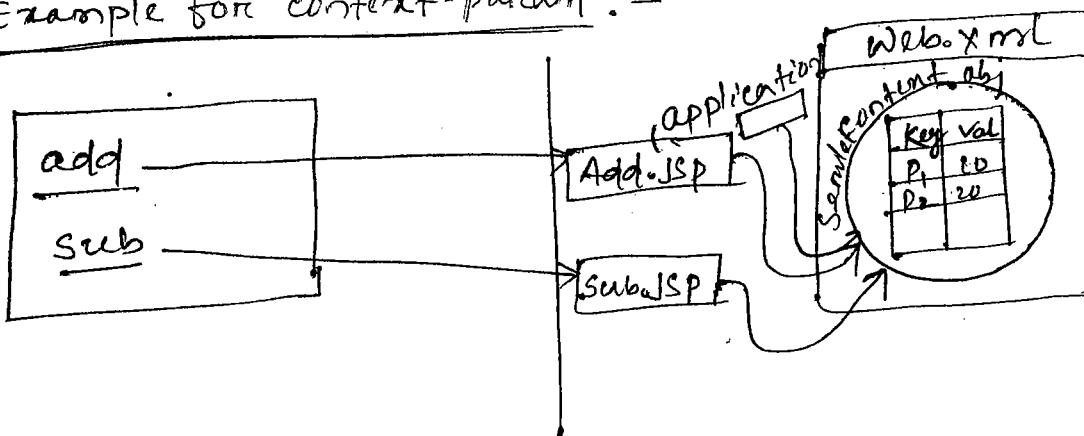
<p-n> P2 </p-n>

<p-v> 20 </p-v>

</context-param>

</web-app>

Example for context-param :-



Web.xml

```
<web-app>
<context-param>
<p-n> P1 </p-n>
<p-v> 10 </p-v>
</context-param>
<context-param>
<p-n> P2 </p-n>
<p-v> 20 </p-v>
</context-param>
</web-app>
```

SRI RAGHAVENDRA XEROX
Scanners / Printers / Images Material Available
Opposite Ayyagar Bakery,
Bukkampet, Hyderabad.

home.html

```
<html>
<body bgcolor=cyan>
<h2>
<a href = "http://localhost:8086/jspapplication5/
add.jsp"> Add</a><br>
<a href = "http://localhost:8086/jspapplication5/
sub.jsp"> Sub</a><br>
</h2>
</body>
</html>
```

Add.jsp

```
<html>
<body bgcolor=yellow>
<h2>
<%!
int n1, n2, n3;
%>
```

Add.jsp

<%>

```
n1 = Integer.parseInt(application.getParameter("p1"));
n2 = Integer.parseInt(application.getParameter("p2"));
n3 = n1 + n2;
```

<%>

$\langle h_2 \rangle$ sum is $\langle ? = n_3 \rangle \langle h_2 \rangle$

~~$\langle h_1 \rangle \langle h_2 \rangle$~~

$\langle /body \rangle$

$\langle /html \rangle$

Sub.jsp

$\langle html \rangle$

$\langle body \rangle$ bgcolor = red

$\langle h_2 \rangle$

~~$\langle a href$~~

$\langle % !$

```
int n1, n2, n3;
```

$\% >$

$\langle t \rangle$

```
n1 = Integer.parseInt(application.getInitParameter("p1"));
n2 = Integer.parseInt(application.getInitParameter("p2"));
```

```
n3 = n1 - n2;
```

$\% >$

$\langle h_2 \rangle$ difference is $\langle ? = n_3 \rangle \langle h_2 \rangle$

$\langle /h_2 \rangle$

$\langle /body \rangle$

$\langle /html \rangle$

② include directive:-

Q. what is include directive?

Ans:- The include directive is used to statically insert the contents of a resource into the current JSP.

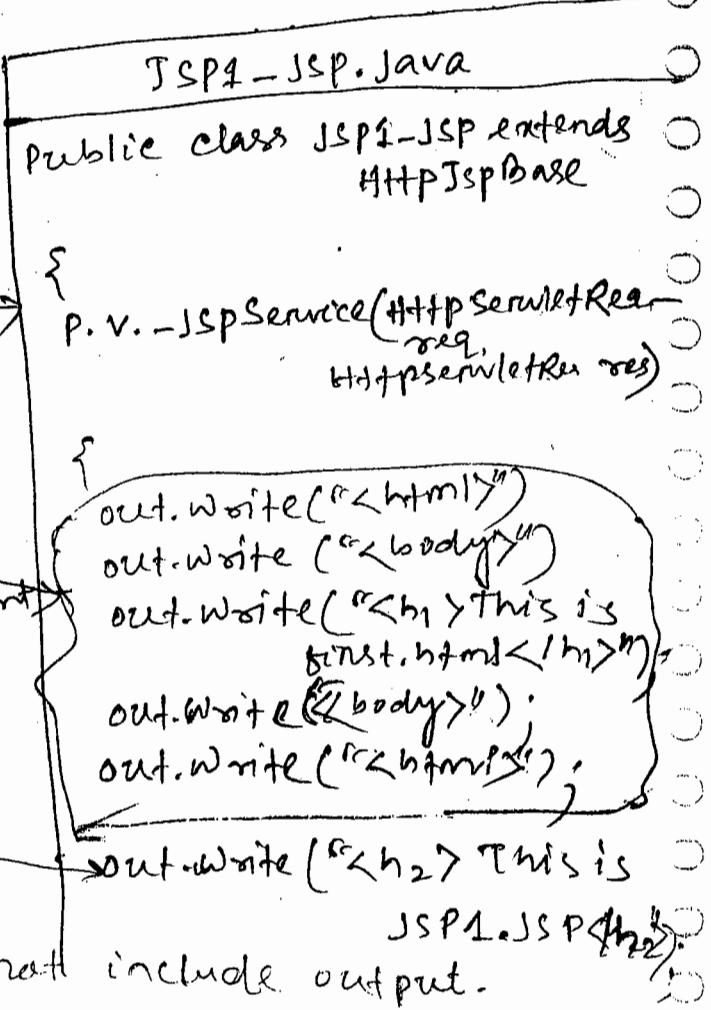
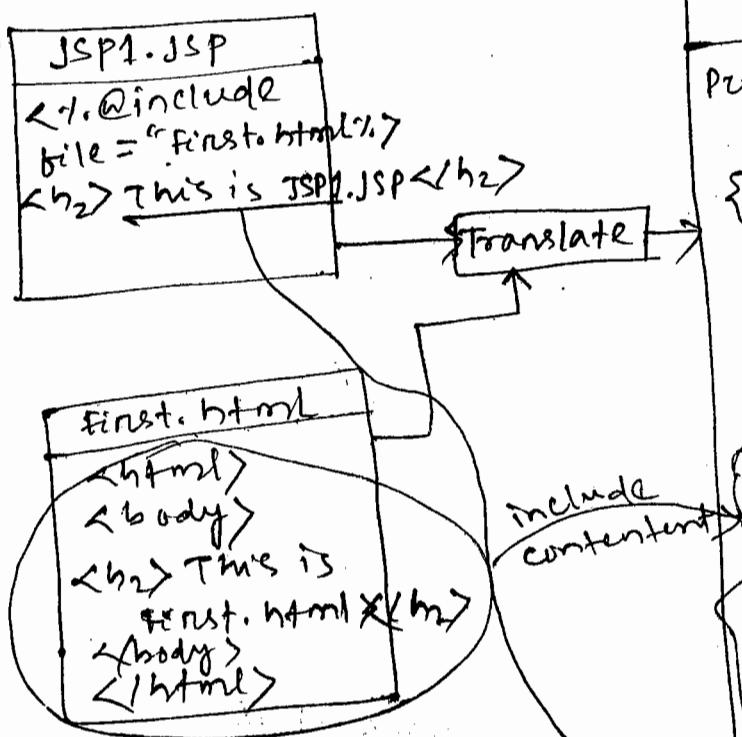
→ This enables a user to reuse the code without duplicating it and includes the contents of the specified file at the translation time.

Syntax :-

```
<%@include file="filename"%>
```

→ This directive has only one attribute called file that specifies the name of the file to be included.

→ One JSP page can have more than one include directives.



It is include content not include output.

Standard Actions :-

→ These are basically used to pass runtime information to the container.

→ As a part of JSP we have the following standard actions; they are

<jsp:forward>,
<jsp:include>,
<jsp:param>,
<jsp:useBean>,
<jsp:setProperty> and
<jsp:getProperty>

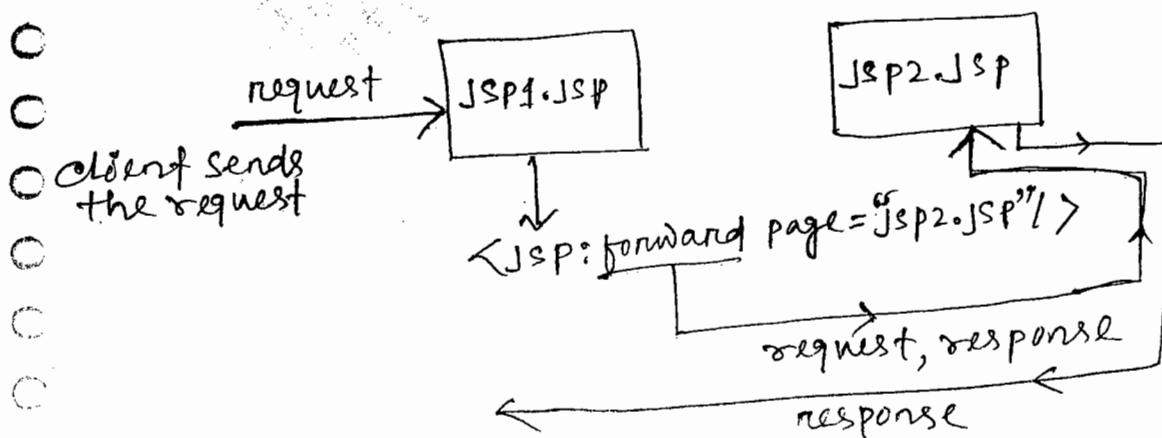
1. <jsp:forward> :

when we want to forward a request and response to the destination JSP page from the source JSP we must use <jsp:forward>

Syntax :-

Without body :-

<jsp:forward page="relative or absolute path of JSP page">

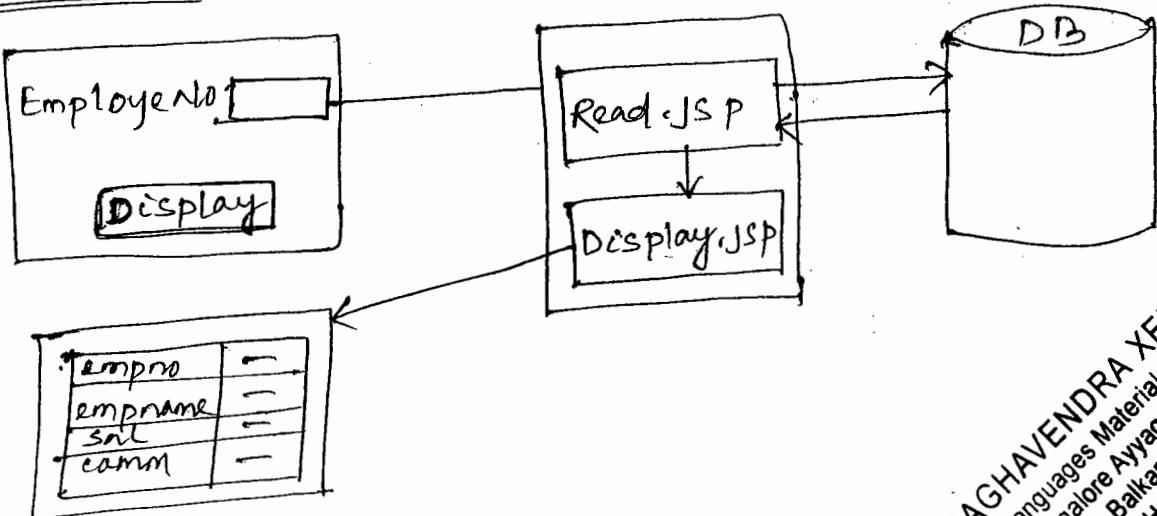


Syntax of with body

```
<jsp:forward page="page">  
<jsp: param name="name" value="value"/>  
<jsp: param name="name" value="value"/>  
.....  
</jsp:forward>
```

→ these parameters are stored as a part of request.

Example :-



home.html

```
<html>  
<body bgcolor="red">  
<form action=".read.jsp">  
<h2>  
EmployeeNo <input type="text" name="eno" />  
<input type="submit" value="display" />  
</h2>  
</form>  
</body>  
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

read.jsp

```
<%@page import = "java.sql.*"%>
<%!
    Connection cn;
    PreparedStatement ps;
    ResultSet rs;
%>
public void jspInit()
{
    try{
        Class.forName ("oracle.jdbc.driver.OracleDriver");
        cn = DM.getConnection ("-", "-", "-");
    }
    catch (Exception k)
    {
        k.printStackTrace();
    }
}
<%>
<% int empno = Integer.parseInt (request.getParameter ("eno"));
    try{
        ps = cn.prepareStatement ("Select empno, ename, sal,
                                nvl(comm,0), sal+nvl(comm,0),
                                from emp where
                                empno = ?");
        ps.setInt (1, empno);
        rs = ps.executeQuery ();
        boolean b = rs.next ();
        if (b == true)
    }
    ><%>
<jsp:forward page = "display.jsp">
<jsp:param name = "eno" value = "<%= rs.getInt(1)%>">
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

<jsp:param name="n" value="<% = rs.getString(2)%>">
<jsp:param name="s" value="<% = rs.getFloat(3)%>">
<jsp:param name="r" value="<% = rs.getFloat(4)%>">
<jsp:param name="t" value="<% = rs.getFloat(5)%>">

</jsp:forward>
<%
}
else
{
%>
<%@ include file="home.html"%>
<h2> Invalid empno </h2>
<%
}
%>
&gt;
catch (Exception k)
{
    k.printStackTrace();
}
%>

```

display.jsp

```

<html>
<body bgcolor="yellow">
<h2>
<table border="2">
<tr>
<td> EmployeeNo </td>
<td> <% = request.getParameter("eno") %> </td>
</tr>
<tr>
<td> EmployeeName </td>

```

SRI RAGHAVENDRA XEROX
 Material Available
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

```
<td><%.= request.getParameter("en")%></td>
<td> Salary </td>
<td><%.= request.getParameter("s")%></td>
</td>
<td> comm </td>
<td><%.= request.getParameter("c")%></td>
</td>
<td>
<td> Total Salary </td>
<td><%.= request.getParameter("t")%></td>
</td>
</td>
</table>
</h2>
</body>
</html>
```

2. <jsp:include>

→ This tag is used for processing a client request by a source JSP page by including other JSP pages and static resources like HTML's. One source JSP can include a number of server side resources and finally we get the response of source JSP.

Syntax :-

① Without body:-

```
<jsp:include page="relative or absolute path  
of JSP page"/>
```

② with body :-

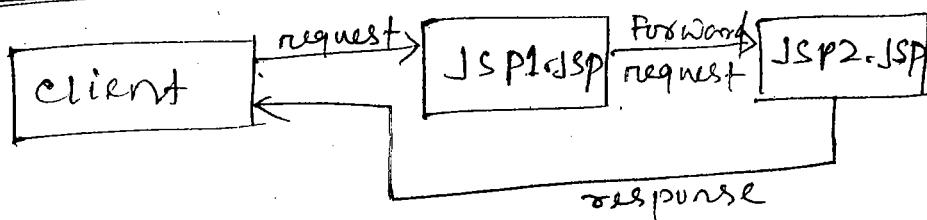
<jsp:include page="relative or absolute path
of JSP page"/>

<jsp:param name="param name 1"
value="param value 1"/>

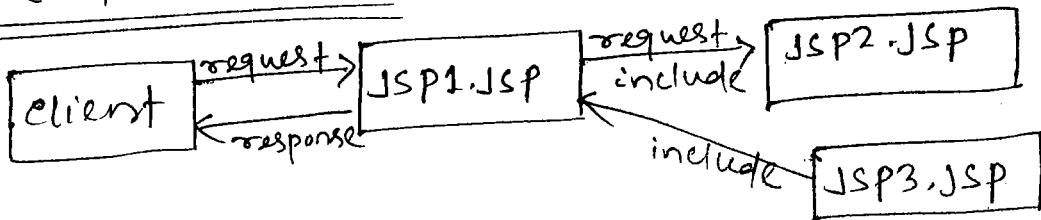
<jsp:param name="param name 2"
value="param value 2"/>

</jsp:include>

<jsp:forward>



<jsp:include>



11.12.15

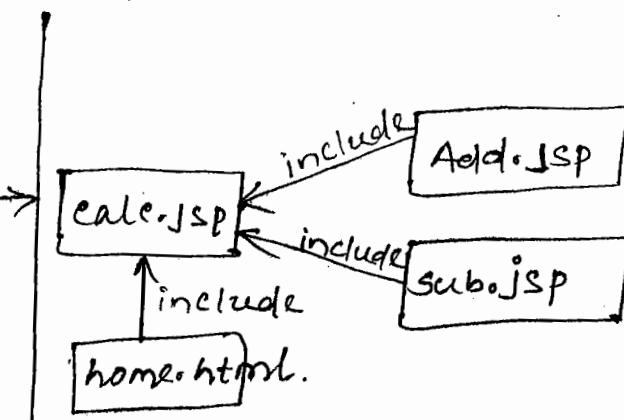
Q. what is the difference between <%@include%>
and <jsp:include> ?

<%@include%>	<jsp:include>
(1) It is a directive.	(1) It is a standard action.
(2) It is executed during translation and insert code inside generated Servlet.	(2) It is executed during reentime.
(3) used for including static resource (.html)	(3) Used for including static (.html) and dynamic pages (jsp)
(4) It include file.	(4) It include response/output.
(5) can't send values.	(5) Can send values as parameters.

Example

Browser

Number	<input type="text"/>
Number	<input type="text"/>
Add	Sub



Web application

home.html

```

<html>
<body bgcolor="cyan">
<form action=".calc.jsp">
<h2>
Number <input type="text" name="t1"><br>
Number <input type="text" name="t2"><br>
<input type="submit" value="Add" name="b">
<input type="submit" value="Sub" name="b">
</h2>
</form>
</body>
</html>
  
```

calc.jsp

```

<%!
String b;
%>
<%
b = request.getParameter("b");
%>
<%@ jsp:include page="home.html" %>
  
```

```

<%!
if (b.equals("add"))
{
%>
<jsp:include page="add.jsp">
<%
}
else
{
if (b.equals("sub"))
{
%>
<jsp:include page="sub.jsp">
<%
}
%>

```

SRI RAJAVENDRA XERQX
 SRI RAJAVENDRA XERQX
 Available
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Beside Balkampet Road,
 Opp. CDAC, Balkampet Road.
 Ameerpet, Hyderabad.

Add.jsp

```

<%!
int n1, n2, n3;
%>
<%
n1 = Integer.parseInt(request.getParameter("t1"));
n2 = Integer.parseInt(request.getParameter("t2"));
n3 = n1 + n2;
%>
</h2> sum is <%= n3%></h2>

```

Sub.jsp

```

<%!
int n1, n2, n3;
%>

```

```

n1 = Integer.parseInt(request.getParameter("t1"));
n2 = Integer.parseInt(request.getParameter("t2"));
n3 = n1 - n2;
> Difference is <math>n_3</math>

```

Java Bean

Java Bean:-

- Java A JavaBean is a specially constructed java class written in the Java and coded according to the JavaBeans API specification.
- characteristics of JavaBean
 - It provides a default, no-argument constructor.
 - It should be Serializable and implement the Serializable interface.
 - It may have a number of properties/variables which can be read and/or written.
 - It may have a number of "getter" and "setter" methods for the properties.

```

Package com.nit.beans;
import java.io.*;
public class EmployeeBean implements
{
    private int empno;
    private String ename;
    private float sal;
}

```

Properties.

Go to Source for
Setting Setter and Getter Method

```

public void setEmpno (int empno)
{
    this.empno=empno;
}
public int getEmp()
{
    return empno;
}

```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```

public void setFname( String fname )
{
    this.fname = fname;
}
public String getName()
{
    return fname;
}
public void setSal( float sal )
{
    this.sal = sal;
}
public float getSal()
{
    return sal;
}
<jsp:useBean/> ; -

```

This tag is used for creating an object of JavaBeans class as a part of JSP page.

Syntax :-

```

<jsp:useBean id = "object name of a JavaBeans class"
              class = "Fully qualified name of JavaBeans class"
              scope = "scope attribute"/>

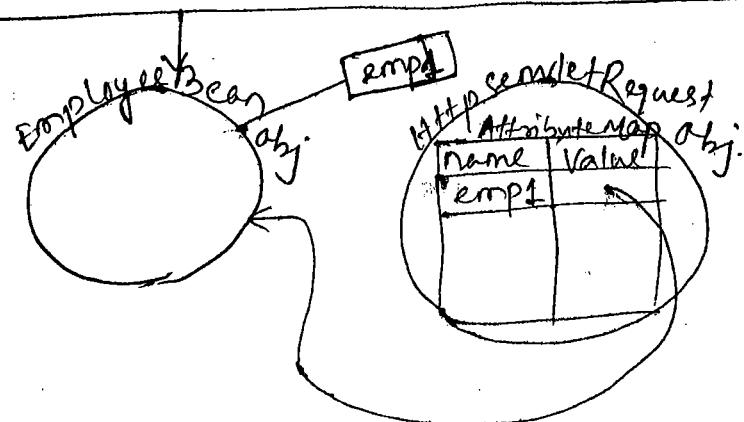
```

JSP1.jsp

```

<jsp:useBean id = "emp1"
              class = "com.nit.beans.EmployeeBeans"
              scope = "request"/>

```



15.12.15

1. param :-

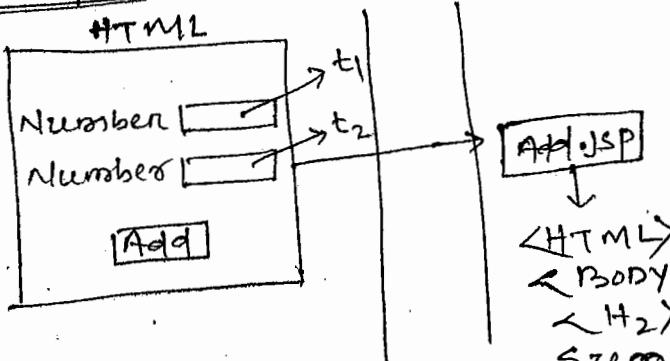
→ It is used for reading request parameters.

Syntax

`$ ${expression}`

`<%@ page isELIgnored = False/true %> [enable or disable Expression language]`

Example



`<HTML>
<BODY>
<H2>
Sum is ${param.t1+param.t2}
</H2></BODY></HTML>`

HOME.html

```
<html>
<body bgcolor = cyan>
< form action = ".//add.jsp">
    Number <input type = text name = "t1"><br>
    Number <input type = text name = "t2"><br>
    <input type = submit value = "add">
</form>
</body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Add.jsp

`<html> <%@page isELIgnored="false" %>`
`true`

`<body bgcolor="red">`

`<h2>`

`Sum is ${param.t1 + param.t2}`

`</h2>`

`</body>`

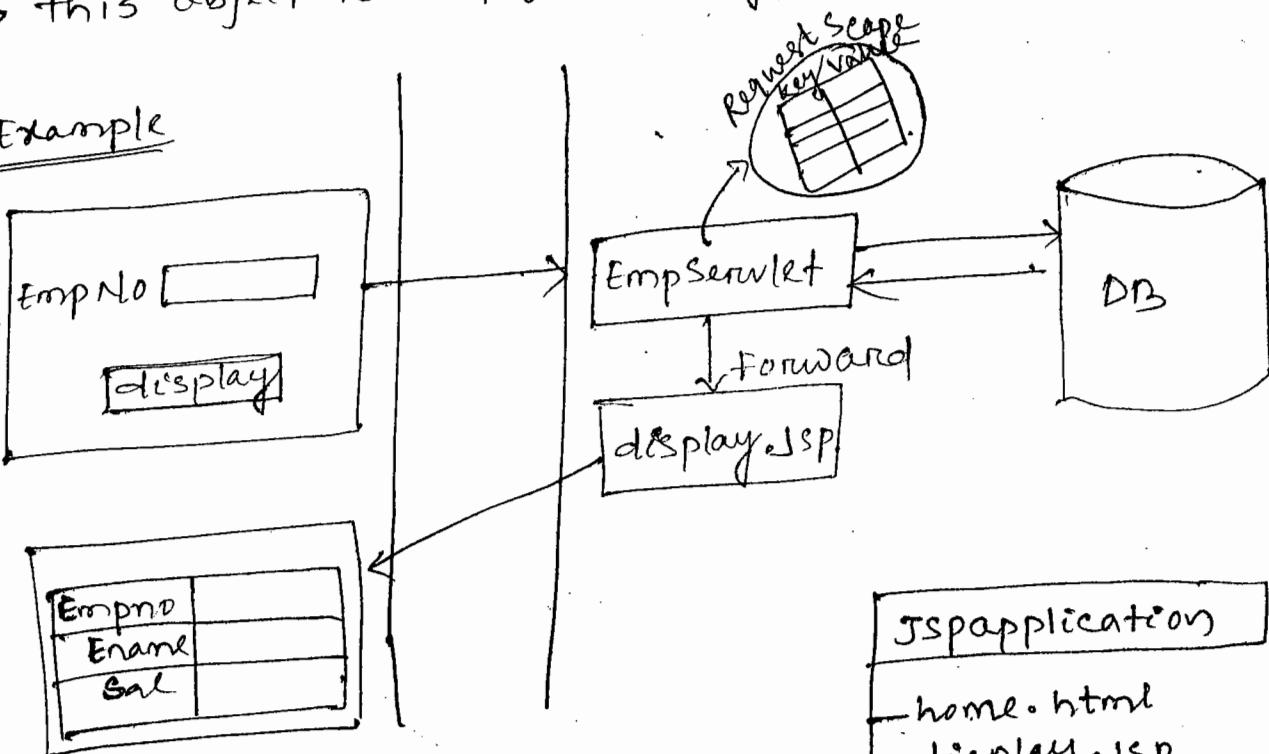
`</html>`

[If isELIgnored is true then
the expression is treated as
simple text.]

2. requestScope:-

→ this object is used for reading request attributes.

Example



'I RASHAVENDRA XEROX
Software Languages Material Available
Outside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad'

home.html

```
<html>
<body bgcolor="cyan">
<form action="/emp">
<h2>
Empno <input type="text" name="t1" />
<input type="submit" value="display">
</h2>
</form>
</body>
</html>
```

EmpServlet

```
package com.nit.servlets;
import java.sql.*;
@WebServlet("/emp")
public class EmpServlet extends HttpServlet {
Connection cn;
public void init()
{
try{
    Class.forName("oracle.jdbc.driver.OracleDriver");
    cn = DriverManager.getConnection("jdbc:oracle:thin:@127.0.0.1:1521:xe");
}
catch (Exception K)
{
    K.printStackTrace();
}
}}
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

```
protected void doGet(HttpServletRequest req,  
                      HttpServletResponse res)  
throws ServletException, IOException
```

```
{  
    int eno = Integer.parseInt(req.getParameter("t1"));  
    PrintWriter out = res.getWriter();  
    try {
```

```
        PreparedStatement ps = cn.prepareStatement  
            ("select empno, ename, sal from emp where  
             empno=?");  
        ps.setInt(1, eno);
```

```
        ResultSet rs = ps.executeQuery();
```

```
        boolean b = rs.next();
```

```
        if (b == true)
```

```
{
```

```
        RequestDispatcher rd1 = req.getRequestDispatcher  
            ("display.jsp");
```

```
        req.setAttribute("empno", rs.getInt(1));
```

```
        req.setAttribute("ename", rs.getString(2));
```

```
        req.setAttribute("sal", rs.getFloat(3));
```

```
        rd1.forward(req, res);
```

```
}
```

```
else
```

```
{
```

```
    RequestDispatcher rd2 = req.getRequestDispatcher  
        ("home.html");
```

```
    rd2.include(req, res);
```

```
    out.println("Invaled empno");
```

```
}
```

```
}
```

```
        catch (Exception k)
    {
        k.printStackTrace();
    }
}
```

display.jsp :-

```
<html>
<body> bgcolor = red >
<table border = 2>
<tr>
<td> EmployeeNo <td>
<td> ${requestScope.empno} </td>
</tr>
<tr>
<td> EmployeeName <td>
<td> ${requestScope.ename} </td>
</tr>
<tr>
<td> Salary </td>
<td> ${requestScope.sal} </td>
</tr>
</table>
</body>
</html>
```

SRI RAUCHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

3. applicationScope :-

→ this object is used for reading attribute from Servlet context.

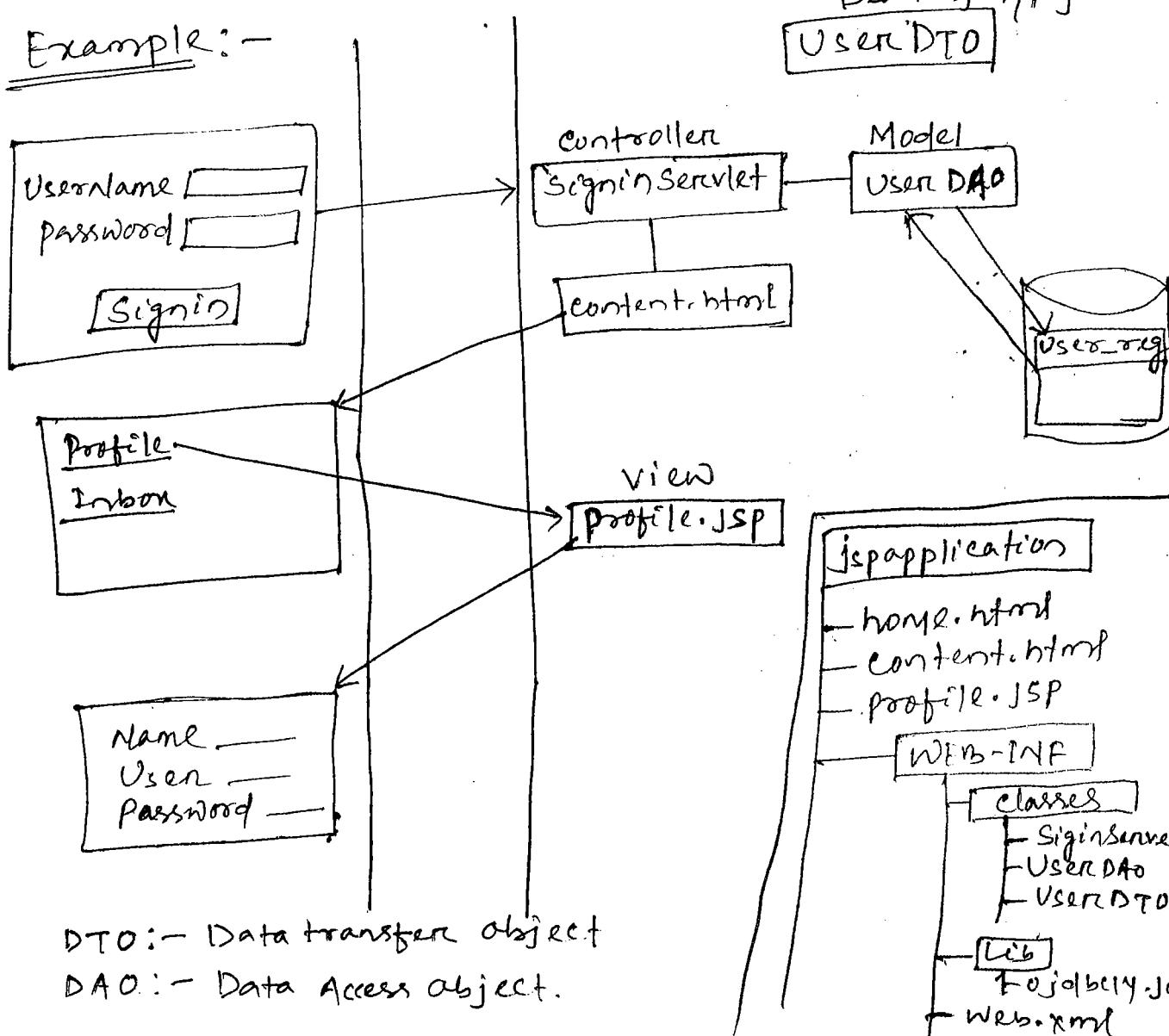
4. sessionScope :-

→ this object is used for reading attributes from HttpSession.

```
$ { requestScope.attribute-name }  
$ { applicationScope.attribute-name }  
$ { sessionScope.attribute-name }
```

(reading
Attributes)

Example:-



Q1. HTML

```
<html>
<body bgcolor="cyan">
<form action=". /signin">
<h2>
    username <input type="text" name="t1"> <br>
    password <input type="text" name="t2"> <br>
    <input type="submit" value="signin">
</h2> </form> </body></html>
```

16.12.15

Develop Interface

```
package com.commit.dao;
import com.commit.dto.UserBean;
public interface UserDAO {
    public boolean verifyUser (String uname, String pwd);
    public UserBean getProfile (String uname);
}
```

class UserBean

```
package com.commit.dbo;
public class UserBean {
    private String name;
    private String user;
    private String pwd;
    public String getName() {
        return name;
    }
    P. V. setName (String name) {
        this.name = name;
    }
    public String getUser() {
        return user;
    }
}
```

```
public void setUser( String user ) {  
    this.user = user  
}
```

```
P. v. getPwd() {  
    return pwd;  
}
```

```
P. v. setPwd( String pwd ) {  
    this.pwd = pwd;  
}
```

Develop a class :-

```
package com.mit.util;  
import java.sql.*;  
public class DBUTIL {  
    public static Connection getConnection()  
    {  
        Connection cn = null;  
        try {  
            Class.forName("oracle.jdbc.driver.OracleDriver");  
            cn = DM.getConnection("-", "-", "-");  
        }  
        catch (Exception k)  
        {  
            k.printStackTrace();  
        }  
    }
```

MANAVENDRA XEROX
Oriental Technologies Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

Implementation class

```
package com.net.daoimpl;
import com.net.daoi.UserDaoI;
import com.net.util.DBUtil;
public class UserDaoImpl implements UserDaoI {
    private Connection cn;
    public UserDaoImpl()
    {
        cn = DBUtil.getConnection();
    }
    public boolean verifyUser(String uname, String pwd)
    {
        boolean b = false;
        try
        {
            PreparedStatement ps =
                ("Select count(*) from user-reg where uname=? and"
                 "pwd=?");
            ps.setString(1, uname);
            ps.setString(2, pwd);
            ResultSet rs = ps.executeQuery();
            rs.next();
            int c = rs.getInt(1);
            if (c > 0)
                b = true;
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
        return b;
    }
}
```

```

public UserBean getProfile( String uname )
{
    UserBean b = null;
    try
    {
        preparedstatement ps = cn.prepareStatement
        ("Select * from User-reg where Uname = ?");
        ps.setString (1, uname);
        ResultSet rs = ps.executeQuery();
        b = new UserBean();
        rs.next();
        b.setName (rs.getString(1));
        b.setUser (rs.getString(2));
        b.setPwd (rs.getString(3));
    }
    catch (Exception k)
    {
        k;
    }
    return b;
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Adding Servlet :-

```

package com.rnit.servlets; import java.*;
import java.io.IOException;
@WebServlet ("/signin")
public class SigninServlet extends HttpServlet {
    protected void doGet(HttpServletRequest req,
                         HttpServletResponse res)
        throws ServletException, IOException
}

```

```

String u = req.getParameter("t1");
String p = req.getParameter("t2");
UserDAO userdao = new UserDAOImp();
boolean b = userdao.verifyUser(u, p);
if (b == true)
{
    UserBean ub = null;
    ub = userdao.getProfile(u);
    HttpSession session = req.getSession();
    session.setAttribute("profile", ub);
    RequestDispatcher rd1 = req.getRequestDispatcher("display.html");
    rd1.forward(req, res);
}
else
{
    RequestDispatcher rd2 = req.getRequestDispatcher("home.html");
    rd2.include(req, res);
}
}

```

SRI RAGHAVENDRA XEROX
 Software Languages Material Available
 Beside Bangalore Ayyagar Bakery,
 Opp. CDAC, Balkampet Road,
 Ameerpet, Hyderabad.

Display.html

```
<html>
<body bgcolor="cyan">
<h2>
<a href="http://localhost:8086/jspapplication1/
profile.jsp">profile</a>
</h2>
</body>
</html>
```

Profile.jsp

```
<html>
<body bgcolor="yellow">
<h2>
Name: ${sessionScope.profile.name} <br>
User Name: ${sessionScope.profile.user} <br>
Password: ${sessionScope.profile.pwd}
</h2>
</body>
</html>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

JSTL (JSP Standard Tag Lib)

- JSTL is a specification given by Sun Microsystem for developing tag library.
- These tag lib. are developed by Server providers.
- The JSTL tags are classified, according to their functions, into following JSTL tag library groups that can be used when creating a JSP page:

(1) Core Tags

(2) Formatting Tags

(3) SQL Tags

(4) XML Tags

(5) JSTL Functions.

for accessing these tags sun given URL

core tags: <http://java.sun.com/jstl/core>.

formatting tags: <http://java.sun.com/jstl/fmt>,

SQL tags: <http://java.sun.com/jstl/sql>.

XML tags: <http://java.sun.com/jstl/xml>.

Installing JSTL :-

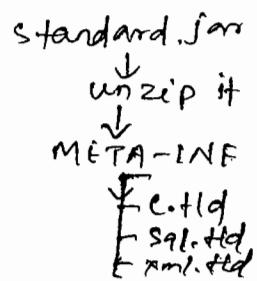
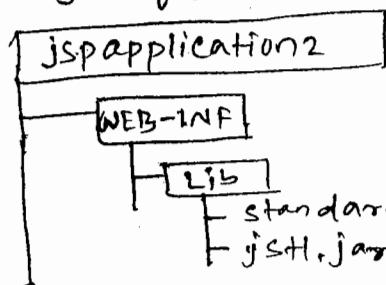
inorder to work with JSTL, we need import two jar files.

1. standard.jar.
2. jstl.jar

Location :-

C:\Tomcat 7.0\ webapps\ examples\ WEB-INF\ lib

→ copy the 2 jar files



unzip it.
extract one jar file.

td (tag library descriptor file)

→ this file contain mapping of tag handler class with tag name.

c.td → contains core tags

sql.td → contains sql tags

fmt.td → contains formatting tags

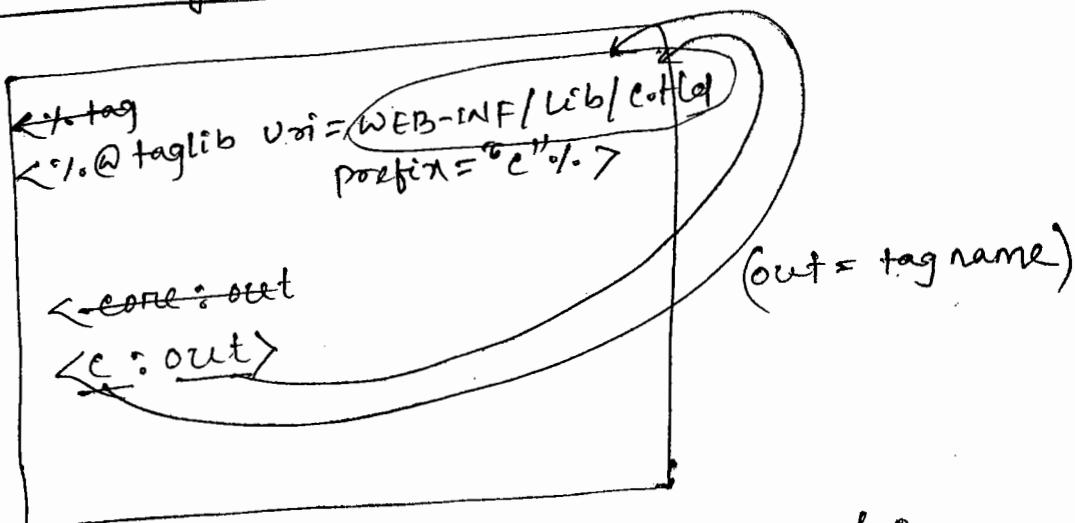
xml.td → contains xml tags

<%@ taglib> directive

→ it is use for including tag libraries.

<%@taglib uri="location" prefix="prefix"%>

including file



how to configure lib in web.xml :-

Syntax:-

```
<web-app>
  <jsp-config>
    <taglib>
      <taglib-uri> uri </taglib-uri>
      <taglib-location> location </taglib-location>
    </taglib>
  </jsp-config>
</web-app>
```

Web.xml

```

<web-app>
  <jsp-config>
    <taglib>
      <taglib-uri> http://java.sun.com/jstl/core </taglib-uri>
      <taglib-location> WEB-INF/lib/META-INF/
        <!--</taglib-location>
      </taglib>
    </jsp-config>
  </web-app>

```

<c:set.....>

→ It is used for declare variables.

```

<c:set var="variablename"
       value="value" scope="scope"/>

```

→ scope define where to store variable.

- request scope
- application scope
- session scope
- page scope (It is default scope of a variable)

<c:out ...>

→ This tag is used for generating response.

```

<c:out value="value/expression/variable"/>

```

Example:-

```
<%@taglib uri="http://java.sun.com/jstl/core"
prefix="c"%>
<c:set var="n1" value="10"/>
<c:set var="n2" value="20"/>
<c:set var="n3" value="${n1+n2}"/>
<h2> sum is <c:out value="${n3}"/>
```

(3) <c:if.....>

→ This tag is used for defining condition.

```
<c:if test="expression">
  statements
</c:if>
```

Example:- man.jsp

```
<%@taglib uri="http://java.sun.com/jstl/core"
prefix="c"%>
```

```
<c:set var="n1" value="${param.t1}"/>
<c:set var="n2" value="${param.t2}"/>
<c:if test="${n1>n2}">
<h2> Max is <c:out value="${n1}"/></h2>
</c:if>
<c:if test="${n1<n2}">
<h2> Max is <c:out value="${n2}"/></h2>
</c:if>
```

html

```

<html>          home.html
<body bgcolor = cyan>
<form action = ". / Max.jsp">
<h2>
Number <input type = text name = "t1" > <br>
Number <input type = text name = "t2" > <br>
<input type = submit value = "Max" >
</h2>
</form> </body> </html>

```

④ <c:forEach....>

→ Used for executing one or more than one statement number of times.

Syntax

```

<c:forEach var = "name of the variable"
begin = "starting value"
end = "ending value"
step = "updation" >
BLOCK of statements
</c:forEach>

```

```
for(i=1; i<=10; i++)
```

→ Example.

```

<c:forEach var = "i" begin = "1"
end = "10" step = "1" >


## 


```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

⑤ <cf_choose...../>

- It is selection tag, which execute statements based on condition.
- It is similar to switch... case.

Syntax:-

```
<cf_choose>
<cf_when test="{$test condition 1}">
    Block of statements;
    </cf_when>
    <cf_when test="{$test condition 2}">
        Block of statements;
        </cf_when>
    <cf_otherwise>
        Block of statements;
    </cf_otherwise>
```

SR/ RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SQL Tags:-

→ In order to use SQL tags we need to configure SQL tag library in Web.xml.

Name of the tld file is sql.tld

uri = http://java.sun.com/jstl/sql

prefix = sql.

Web.xml

<Web-app>

<taglib <jsp-config>

<taglib>

<lib> <uri> http://java.sun.com/jstl/sql </taglib-uri>

<taglib-location> WEB-INF/lib/META-INF/sql.tld

</taglib-
location>

</taglib>

</jsp-config>
</Web-app>

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

sql:setDataSource

→ This tag is used to define datasource.

→ Datasource consist of driver, url, username and password.

→ Datasource hold connection of database.

Syntax

```
<sql: setDataSource var = "variablename"  
driver = "driver class name" url = "jdbc url"  
user = "username" password = "database password"  
scope = "request/application/session/page"/>
```

Example of sql:DataSource

```

<%@taglib uri="http://java.sun.com/jstl/sql"
           prefix="sql"%>
<sql: setDataSource name="dsn1">
    driver="oracle.jdbc.driver.OracleDriver"
    url="jdbc:oracle:thin:@localhost:1521:XE"
    user="system" password="tiger"/>

```

web.xml

<taglib>

<taglib-uri> http://java.sun.com/jstl/sql </taglib-uri>

<taglib-location> WEB-INF/lib

/META-INF/sql.tld

</taglib-location>

② <sql:update>

→ This tag is for executing DML statements like insert, update and ~~delete~~ delete.

Syntax:-

```

<sql:update dataSource="datasource name"
            var="variable which hold count">
    DML Statement;
</sql:update>

```

Example of sql:update

```

<sql:update dataSource="#{dsn1}" var="count">
    update emp set sal=sal+100
</sql:update>

```

③ <sql:param>

→ To define the values of parameters hold by SQL statement.

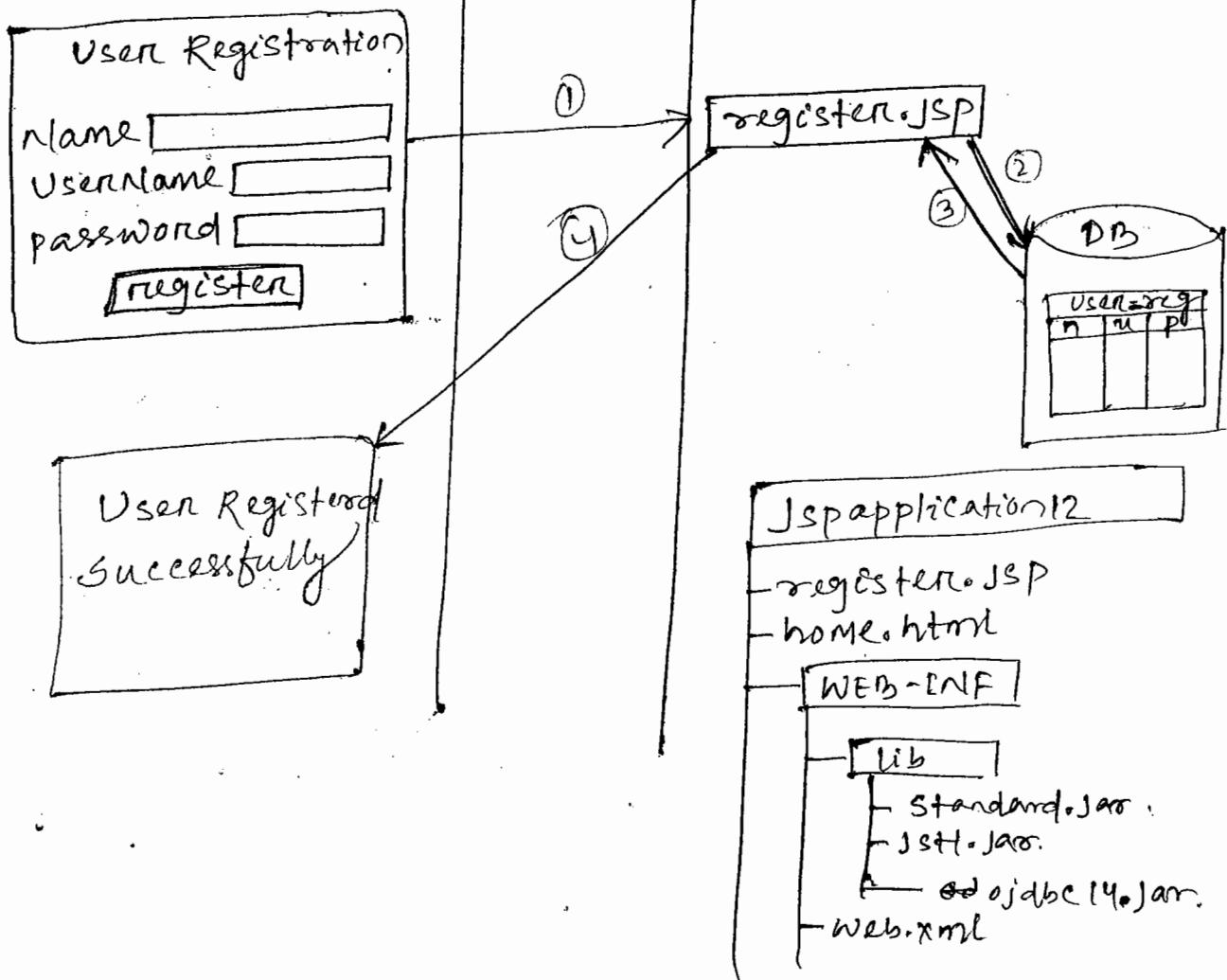
Syntax:-

```
<sql:param value="value"/>
```

Example:-

```
<sql:reupdate datasource="$\{dsn1\}" var="c">
    insert into dept values (? , ? , ? )
    <sql:param value="10"/>
    <sql:param value="xyz"/>
    <sql:param value="abc"/>
</sql:update>
```

Program:-



home.html

```
<html>
<body bgcolor="cyan">
<form action="i/register.jsp">
<h2>
  & Name <input type="text" name="t1"> <br>
  UserName <input type="text" name="t2"> <br>
  password <input type="password" name="t3"> <br>
  <input type="submit" value="register">
</h2>
</form></body></html>
```

```
register.jsp
<%@taglib uri="http://java.sun.com/jstl/core"
   prefix="c"%>
<%@taglib uri="http://java.sun.com/jstl/sql"
   prefix="sql"%>

<c:set var="n" value="${param.t1}"/>
<c:set var="u" value="${param.t2}"/>
<c:set var="p" value="${param.t3}"/>

<sql:setDataSource var="dsn"
   driver="oracle.jdbc.driver.OracleDriver"
   url="jdbc:oracle:thin:@localhost:
        1521:XE"
   user="system"
   password="anjan"/>

<sql:update var="e" dataSource="${dsn}">
  insert into user_reg value (?, ?, ?)
<set:param value="${n}"/>
<set:param value="${u}"/>
<set:param value="${p}"/>
```

```
<sql:update>
<c:if test = "${c} > 0">
<h2> user registered </h2>
</c:if>
```

⑨ <sql:query>

→ This tag is for executing query.

Syntax:-

```
<sql:query dataSource = "datasourcename"
var = "variable which hold resultset">
```

DRL

```
</sql:query>
```

Example (only for executing query)

```
<sql:query dataSource = "${dsn1}" var = "rs">
    select empno, ename, sal from emp
</sql:query>
```

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayagar Bakery
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.

SRI RAGHAVENDRA XEROX
Software Languages Material Available
Beside Bangalore Ayyagar Bakery,
Opp. CDAC, Balkampet Road,
Ameerpet, Hyderabad.