JEE (ADV JAVA)

NOTES

SURESH

INeTSOLV

JEE PART I

Core java basics 2

JDBC 15

JNDI 50

Connection Pooling 58

Servlets 65

JSP 95

Request Dispatcher..... 124

JSP Action tags..... 126

Java beans..... 127

JSP custom tags..... 129

EL(ExpressionLanguage)..... 131

MVC (Model View Cotroller).....133

JSTL(Java standared tag Libraries)..... 133

Annotations(metadata)..... 139

- In Java we can declare following 2 types of variables

1. primitive variables 2. reference variables

- based on the location where we declare variables divided into following 4 types

static variables (inside the method area of JVM)

if variables are declared inside the class outside the methods with using staic keyword are called as static variables

Eg:

```
class Demo{
  static int rno;
  static String name;
   void show(){
    }
  }
```

instance variables (inside the heap memory of JVM)

if variables are declared inside the class outside the methods without using staic keyword are called as instance variables

Eg:

```
class Demo{
  int rno;
  String name;
   void show(){
    }
  }
```

local variables (inside the java stack of JVM)

if declare the variables inside the method or inside any block then we call those variables as local variables.

Eg:

```
class Demo{
 void show(){
   int rno;
 String name;
   }
 }
```

parameters (local variables)

if we declare variables inside the method declaration between parenthesis( ) are called as parameters

Eg:

```
class Demo{
 void show( int rno, String name){
    }
 }
```

- we have following 2 types of variables based on the bhaviour

1. non-final variables (variables)

if any variable value can be changed any number of times then it is called as non-final variables.

2. final variables (constants)

if any variable value cannot be changed at any time then it is called as final variables.

? why static methods

if we want to execute any method directly with hte help of classname then we have to declare our method as static method

Eg:

```
class Demo{
static void show(){
 System.out.println("this is show() method");
   }
}
class Test1{
 public static void main(String args[]){
 Demo.show();
  }
}
```

? can we create an object for final class

-yes we can create an object for final class and we can also access its members

-but final classes cannot be inherited

Eg:

```
final class Demo{
  void show(){
 System.out.println("this is show() method");
   }
}
/*
CTE: cannot inherit from final Demo class
class Child extends Demo{    // X-invalid
}
*/
class Test2{
 public static void main(String args[]){
  Demo d = new Demo(); //creating an object for final class
   d.show();
  }
}
```

javap

This java command mainly used to display the profile of any predefined class or user defined class. Profile means it will display the list of variables and methods that particular class contains.

Syntax:

```
 javap  packagename.ClassName
```

Eg:

```
 javap  java.lang.String
 javap  java.lang.StringBuffer
 javap  Demo
```

API Documentation

Java API Documentation is also used to see profile of any predefined class with description

?what are the diff situations where we can't create any object

1. If a class contains one parameterized constructor and doesnot contain any 0 parameterized constructor then we cannot create any object using 0 parameterized constructor.

```
class Demo{
/*
//solution
Demo(){
}*/
Demo(int x){
}
}
class Test3{
 public static void main(String args[]){
  Demo d = new Demo();  //X-invalid
  }
}
```

2.we can not create an object for a class If it contains private constructor

Eg:

```
class Demo{
private Demo(){
}
}
class Test4{
 public static void main(String args[]){
  Demo d = new Demo(); //Invalid
  }
}
```

solution

In this case we are responsible to create a factory method which is responsible for creating an returning its class object.

Eg:

```
class Demo{
private Demo(){
}
/*factory method*/
static Demo getDemo(){
Demo dd = new Demo();
return dd;
}
void show(){
System.out.println("this is show() method");
}
}
class Test4{
```

```
 public static void main(String args[]){
 Demo d =Demo.getDemo();
  d.show();
 }
}
```

3. if our class is abstract class we can not create any object directly

```
abstract class Demo{
void show(){
System.out.println("this is show() method");
}
}
class Test5{
 public static void main(String args[]){
  Demo d =new Demo();   // X - invalid
  }
}
```

creating an object for abstract class using its child class which is declared normally

```
abstract class Demo{
void show(){
System.out.println("this is show() method");
}
}
class Child extends Demo{
  }
class Test55{
 public static void main(String args[]){
  Demo d = new Child();
  d.show();
  }
}
```

creating an object for abstract class using its child class which is declared inside the factory method

```
abstract class Demo{
/*factory method*/
static Demo getDemo(){
 class Child extends Demo{
  }
Demo dd = new Child();
return dd;
}
void show(){
System.out.println("this is show() method");
}
}
class Test55{
```

```
 public static void main(String args[]){
  Demo d = Demo.getDemo();
   d.show();
 }
}
```

Note:

-If any method has some abstract class as returntype then the particular method returns an object of child class of that abstract class.

4. For interfaces we can not create any object directly

Eg:

```
interface Demo{
void show();
}
class Test6{
 public static void main(String args[]){
  Demo d =new Demo();  // X -invalid
   }
}
```

creating an object for interface using its child class/ implementation class which is declared normally

```
interface Demo{
void show();
}
class Child implements Demo{
public void show(){
System.out.println("this is show() method");
}
}
class Test66{
 public static void main(String args[]){
  Demo d =new Child();
  d.show();
   }
}
```

Note:

-class which is implementing particular interface is called child class or implementation class

-while implementing any interface we must provide the implementation for all the asbtract methods of particular interface

creating an object for interface using its implementation class which is declared as a part of any method

```
interface Demo{
void show();
}
class Manager{
static Demo getDemo(){
 class Child implements Demo{  //method local inner class
```

```
    public void  show(){
        System.out.println("this is show() method");
       }
     }
  Demo d = new Child();
   return d;
  }
}
class  Test666{
 public  static  void  main(String  args[]){
  Demo d = Manager.getDemo();
  d.show();
   }
}
```

Note:

-If any method has some interface as returntype then the particular method returns an object of implementation class or child class of that interface

? why we go for method overriding

when ever child class is expecting its own logic to execute instead of Parent class logic then we have to override the Parent class method inside the Child class with the new definition what is expected by Child class

```
class Parent{
void   show(){
  System.out.println("this is Parent class show() method");
  }
}
class Child   extends Parent{
void  show(){
  System.out.println("this is Child class show() method");
  }
}
class  Test7{
 public  static  void  main(String  args[]){
  Child  c = new Child();
   c.show();
  }
}
```

promoting order of access specifiers in overriding

default<protected<public

upcasting

upcasting means converting an object from Child class reference to Parent class reference.

downcasting

downcasting means converting an object from Parent class reference to Child class reference where writing typecasting is mandatory

//wap to demo on upcasting and downcasting

```
class  Parent{
 void  show(){
  System.out.println("This is Parent class Method");
 }
}
class  Child  extends  Parent{
void  show(){
System.out.println("This is Child class Method");
}
}
class  TypeCasting3{
public  static  void  main(String args[]){
Parent  p;
p = new Parent();
p.show();
p  = (Parent) new Child();  //typecasting optional (upcasting)
p.show();
Child  c =(Child)  p; //typecasting is mandatory ( downcasting)
c.show();
/*
 //RTE:ClassCastException:Parent cannot be cast to Child
 p = new Parent();
 Child  c1 = (Child) p; //downcasting
 c1.show();
*/
}
}
```

Note:

In downcasting while converting Parent class reference type into child class reference type then Parent class reference should contain child class object otherwise we get a runtime exception saying "ClassCastException: Parent cannot be cast to Child"

? how do we organize the projects

- whenever we develop any project we have to follow following directory structure.

step1:first we have to create a folder with our project name

step2:we have to create following folders inside project folder

-- src(folder):contains all .java files

-- classes(folder): contains all .class files

-- lib(folder): contains all .jar files and other supported files

-- doc(folder):contains all doc files which describe about project

-- tmp (folder) : contains extra information about project

Note:

-when ever we are creating classes it allways recommended to create the class using package

-if we want to write a package name we must follow following standared

syntax:

package domainname.companyname.projectname.modulename;

or

package companyname.projectname.modulename;

Sample.java

```
package  com.inetsolv.jee;
public  class   Sample{
public void show(){
System.out.println("this is show() method");
}
}
```

Note:

-If our class is declared as public then our class name and program name must be same.

-In our projects we allways use public classes only and all its members should also be public

-to create a package we have to write package statement in our program using package keyword which must be first line in the program

- to create package and to locate the generated class inside the package we have to compile our java program like follows,

javac -d . Sample.java

program which import and use our package

```
import com.inetsolv.jee.*;
class  PackageTest{
public  static  void main(String  args[]){
 Sample   s =  new Sample();
  s.show();
  }
}
```

Note:

package allways contain set of classes and interfaces in the form .class files where each is class or interface is public all its members must be public.

? how do we deliver our projects

- we can delever out java orjects in the form of .jar files
- we have following 3 types of jar files

1) .jar files(java archieve) for core java projects
2) .war files(web archieve) for web projects
3) .ear files (enterprice archieve) files for enterprice appliations like EJB

- to create all these 3 kinds of jar files we take the help of command called called " jar "

jar command

jar is a java command which used to create jar files

syntax:

jar -cfv myjarfilename.jar listoffiles

c-create new archive

f-specify our own file name

v-verbose mode(display internal details being added to archieve file)

Eg:

jar -cfv myproject.jar com

-To extract the contents of jar file we have to use jar command like follows

jar -xf myproject.jar

Note:

- when we extract jar files along with our folders one extra folder we can see called META-INF which contanins version information and vendor imformation....
- we can also specify our own data inside the meta file like follows

-first create a metafile

myproject.mf

Manifest-Version: 1.0

Created-By: InetSolv Team

-now jar command is used like follows to add our manifest data

Eg:

jar -cfvm myproject.jar myproject.mf com

? what is setting path

- setting path is an instruction given to OS to allow set of commands available in particular path to use.

Eg:

set path="C:\Program Files\Java\jdk1.6.0_17\bin";

?what is setting classpath

- setting classpath is an instruction given to JVM to allow set of classes and interfaces available in the particular .jar file
- we can not import or access the packages or classes or interfaces that are available in the .jar file directly in the particular program if we want to use we can either extract the file or we have to set the class path to corresponding .jar file.
- it is allways recommended to set classpath to use any .jar file
- it is not recommended to extract and using the .jar file which may lead to several issues.

Eg:

creating jar file for com folder or package

> jar -cvf myjarfile.jar com

Note:

delete com folder from the current folder

program which use the package available in .jar file

```
import com.inetsolv.jee.*;
class PackageTest{
public static void main(String args[]){
 Sample s = new Sample();
  s.show();
 }
}
```

compiling the program

> javac PackageTest.java

CTE: package com.inetsolv.jee does not exist

-To resolve this problem we have to set the classpath like follows

> set classpath=myjarfile.jar;.;
> javac PackageTest.java

Now program will be compiled successfully.

? how to solve the problem package xxx does not exist

- if we want to resolve the problem package xxx does not exist just we need to set classpath to corresponding jar file that contains xxx package.
- end section we have to write ;.; to inlcude the current folder classses

Note:

-when we deliver our projects in the form of jar files then programmer is responsible for setting the classpath for every jar file that we are using,

-but to make it simple we create run.bat or run.cmd files which contains classpath setting for all the jar files now user responsible only for executing this .bat file or .cmd files

Eg:

run.bat or run.cmd

```
set classpath=myproject.jar;.;
```

-This .bat file or .cmd files we can create in windows OS but in OS like UNIX or LINUX we have to create shell files like follows

Eg:

run.sh

```
set classpath=myproject.jar;.;
```

Hard coding

-Hardcoding is a concept of providing values directly into variables

- the main disadvantage of Hardcoding is it always generate same output
- It is always recommened to remove the hardcoding in the real time applications.
- To remove Hardcoding we can use different methodologies like ,

    reading values from keyboard
    reading values from a file
    reading values from a textbox
    using command line arguments
    using system properties .....

program to avoid hard coding using command line arguments

Test8.java

```
class Test8{
 public static void main(String ar[]){
  int a=Integer.parseInt(ar[0]);
  int b=Integer.parseInt(ar[1]);
  int c=a+b;
System.out.println("Addtion="+c);
  }
}
```

compilation

> java Test8.java

execution

```
> java  Test8  10 20
Addition=30;
> java  Test8  100 200
Addition=300;
```

program to avoid hard coding using System properties

Test9.java

```
class  Test9{
 public  static  void  main(String  ar[]){
  String   val1=System.getProperty("v1");
  String   val2=System.getProperty("v2");
  int  a=Integer.parseInt(val1);
  int  b=Integer.parseInt(val2);
  int c=a+b;
System.out.println("Addtion="+c);
  }
}
```

compilation

```
> javac Test9.java
```

execution

to set the System properties we have to -D option along with java command while executing like follows

```
>java -Dv1=300 -Dv2=400 Test9
Addtion=700
>java -Dv1=780 -Dv2=480 Test9
Addtion=1260
```

Class.forName(String str)

- forName() is the static method available in predefined class called capital Class which is used to load the classes into JVM explicitly.
- we can load any number of classes into JVM

```
class Sample{
static{     //static block
System.out.println("Hey Im Sample Class");
}
}
class  Test{
public  static  void  main(String  args[]) throws ClassNotFoundException {
   Class.forName("Sample");
   }
}
```

? diff bw NoClassDefFoundError & ClassNotFoundException

when we load the class into JVM using java command implicitly if the loading class is not availble then we get Exception saying " NoClassDefFoundError "

But when we load the class into JVM using Class.forName() method explicitly if the loading class is not availble then we get Exception saying " ClassNotFoundException "

Note:

1. inside the Class.forName() method we must specify fully qualified Class name
(class name along with the package)

Eg:

Class.forName("String"); X-Invalid RTE:ClassNotFoundException

Class.forName("java.lang.String"); -Valid

2. In Class.forName() method if specified class is available in a .jar file we must set the classpath to correponding .jar file otherwise no compile time error but we get run time error

Eg:

```
class Test99{
public static void main(String args[]) throws ClassNotFoundException{
 Class.forName("com.inetsolv.jee.Sample");
  }
}
```

but here com.inetsolv.jee.Sample class is available in myapp.jar file so we need to set the class path to this myapp.jar file

> set classpath=myapp.jar;.;

3. - Here Class.forName() method throws an Exception saying ClassNotFoundException which need to be handled

- We can handle any Exception in following 2 ways
  1. try-catch blocks (recommended)
  2. throws

-It is allways recommened to use try-catch block to handle any exception which is guarantee for complete execution

Eg:

```
class Sample{
static{
System.out.println("Im Sample class loading into JVM");
}
}
class Test10{
public static void main(String args[]){
System.out.println("main() method first line");
try{
 Class.forName("Sample11");
}
catch(ClassNotFoundException cnfe){
cnfe.printStackTrace();
}
System.out.println("main() method last line");
  }
}
```

what are the different ways of creating object

1 using new oeprator
2. using factoryMethod()
3. creating an object using newInstance() method of Class class.

Eg:

```
class Sample{
void show(){
System.out.println("Im Sample class show() method");
}
}
class Test11{
public static void main(String ar[]) throws
ClassNotFoundException,InstantiationException,IllegalAccessException{
Class cl = Class.forName("Sample");
Object o = cl.newInstance();
Sample s = (Sample) o;
s.show();
}
}
```

Note:

1.here if our Sample class is abstract class or having only parameterized constructor then we get runtime exception saying InstantiationException

Eg:

```
abstract class Sample{          (or)      class Sample{
 }                                          Sample(int x){
                                             }
                                           }
```

2.here if our Sample class is having private constructor then we get runtime exception saying IllegalAccessException

Eg:

```
class Sample{
 private Sample(){
 }
}
```

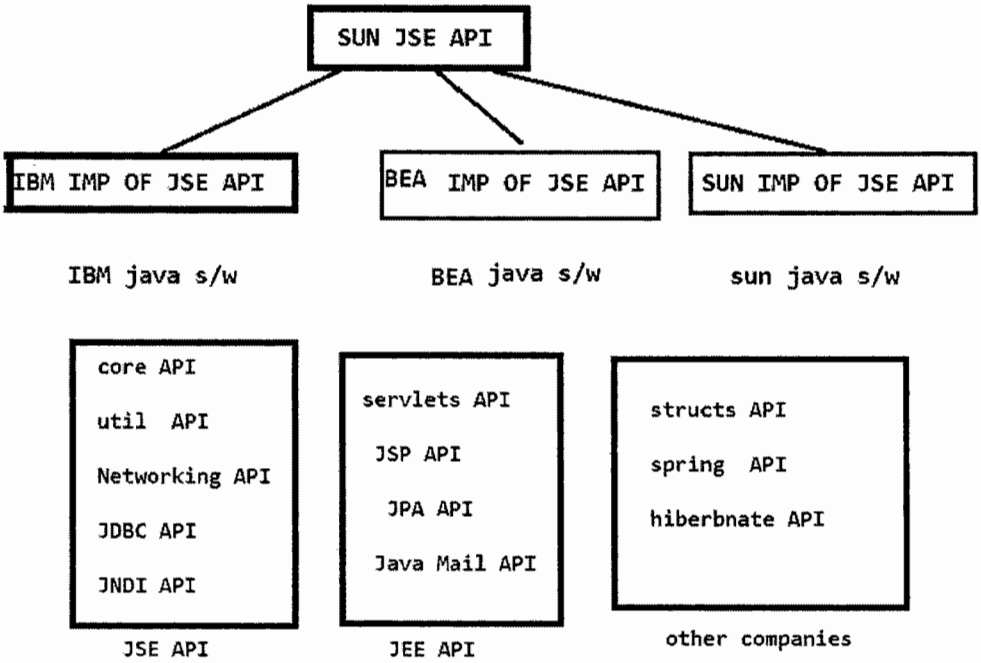
JDBC

API

-API stands for Application Programming interface
-API is a template or model or specification or documentation
-once an API is released any body can provide implementation
-API is not a software which can not be used directly for the development
-we have to use implementation of API for the development of any application
-this implementation of API is called software
-we have following 2 types of API
1. public API 2. proprietery API
- public API means any body can implement
- proprietery API means only particular company who released this API can implement.

? what API contains

-API contains a collection of classes and interfaces in the case of java related APIs
- but any c or c++ related API contains a collection of library functions
- java people have released JSE API as public API so that any body can implement this API and already companies like IBM, BEA, sun,.. are implementing this JSE API



database

- If we want to store the data we can use any one of the following 2 systems
1. file system 2. database

file system

-file system means here we store the data in individual text files
-but file system have so many dis advantages
-we have to store same data repeatedly in multiple files (data redundancy)
-file system does not contain any data sharing
-file system does not provide security

-no maintenance and integrity in file system because if we want to modify the data then we have to modify in multiple files

-no scalability in file system which means there is size limitation problem

-if we want perform any operation we have to write the program like inserting,deleteing, updating,....

database

-in database we store the in form of tables (structured data)

-database resolve all the problems of file system

-database uses one db language called SQL using which we can easily perform any operation like inserting,deleteing,selecting, updating,.... ad we no need to write any new code.

- we have so many databases in the market.

Eg:

oracle, my sql, point db, ibm db2, sql server,.....

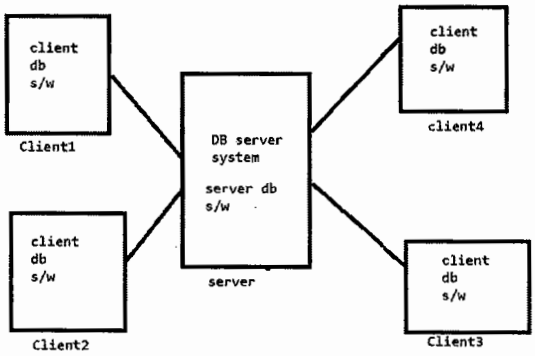
- databases are released in following 2 types of softwares or programs

1. db sever software(program) 2. db client software(program)

-db sever software(program) is installed inside the server system

-this server is connected in any network and ll th clients are connected to this server system

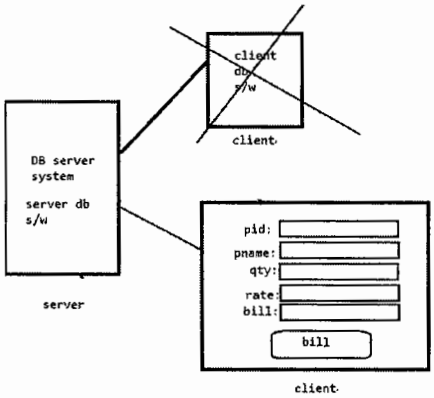
-if any client wants to communicate with db available in the Server we must install db client software inside the client system.



-but if we install client db softwae inside the client system then our client has to write SQL queries to communicte with DB

-but many times our clients are not aware of this SQL and it is not user friendly

-to resolve this problem we have to provide client any java or .net or c application instead of client db softwae then he can easily enter his details and work. Here our application will convert the data entered by the client into SQL queries and send to db



-if any client wants to connect db server avaible in the server

system then he must provide following 3 information.

1.IP address of the server system where db server is instaled

2.service name of db server

3.port number of db server program which is running in server system.

Eg:

IP address: 192.67.90.34

service name: xe

port number: 1521

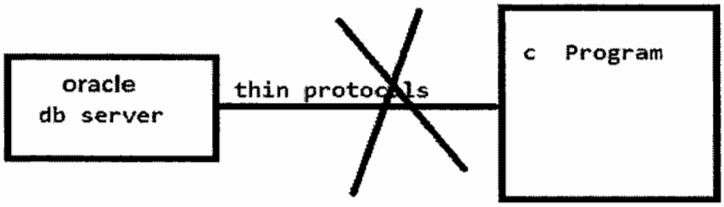
- oracle guys are released oracle db s/w in following 2 editions
  1. express edition (xe)
  2. enterprise edition(orcl)

-oracle people have developed oracle db based on thin protocal

-if any body want to communicate with oracle db we should also follow the same thin protocal

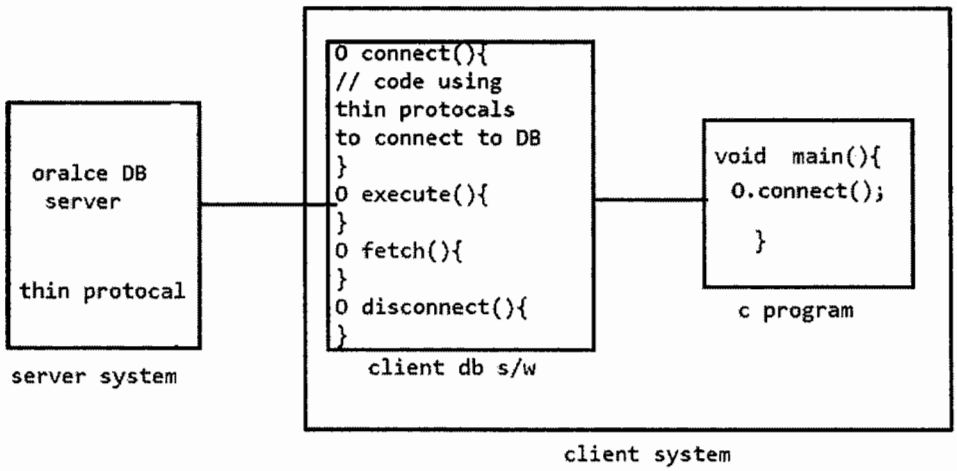
-but oracle people have released thin protocal as proprietary so that no one can communicate with oracle db using thin protocal

- for example if we want to develop a c program to communicate with oracle db using this thin protocal is not possible

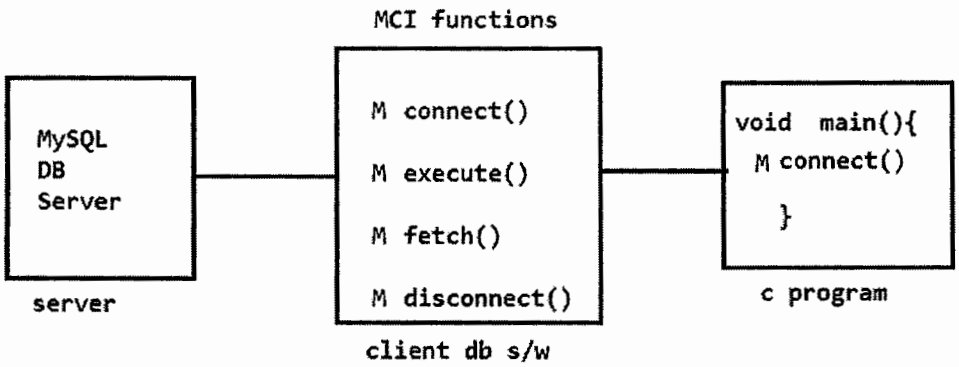


-To resolve the above problem oracle people have released OCI (Oracle call interface) functions as part of client db s/w which communicate with db using thin protocals.

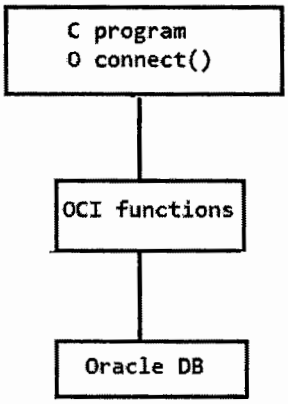
-Now if any body want to write c program which communicate with oracle db we can write by using these OCI functions



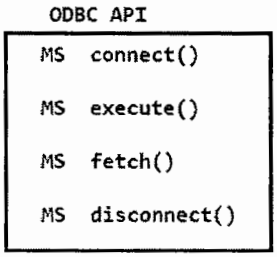
- in the My Sql db people have released MCI functions using which we can develop a c program that ommunicate with MySql DB



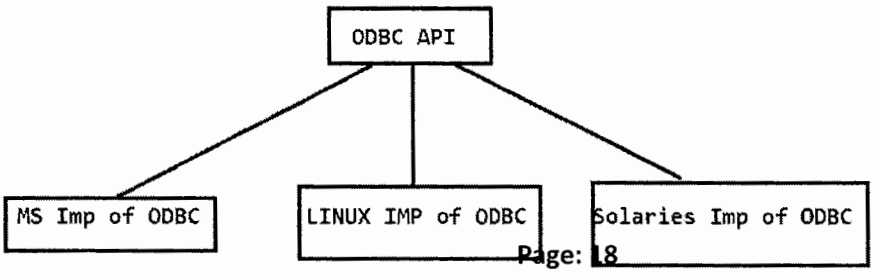
-Following is the simple architecture of a c program which communicate with oracle DB



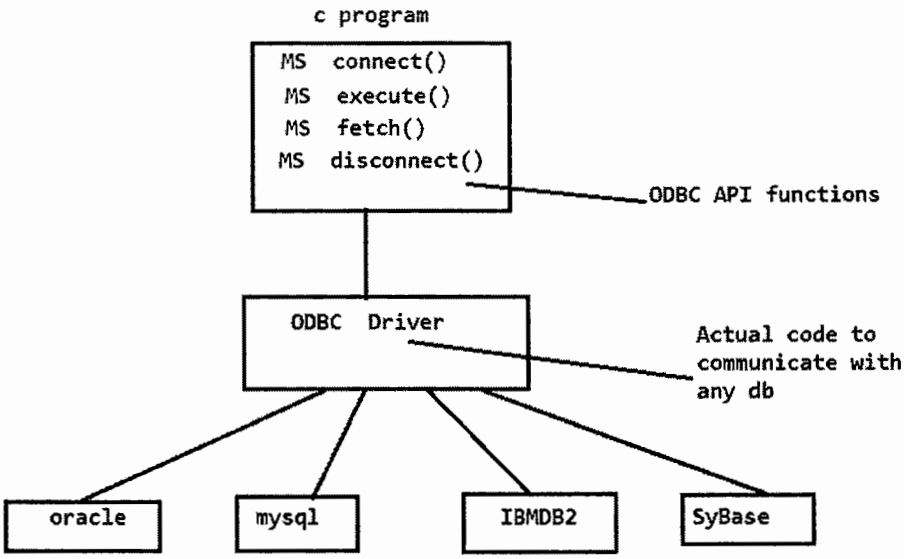
-But Here if we want to change the db then we must change the the code of c program
-It means we can not write a c program which communicate with any db without changing the code
-but to resolve this problem MS people have released ODBC API



-once an API released that can be implemented by anybody.

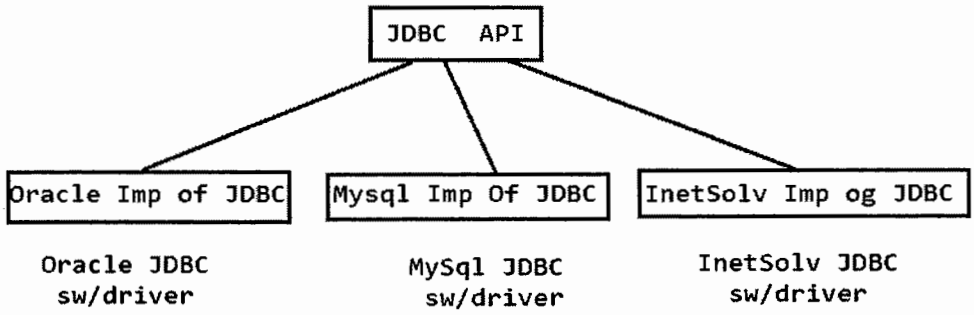


-by using ODBC API we can write a c program which can communicate with any db without changing the code

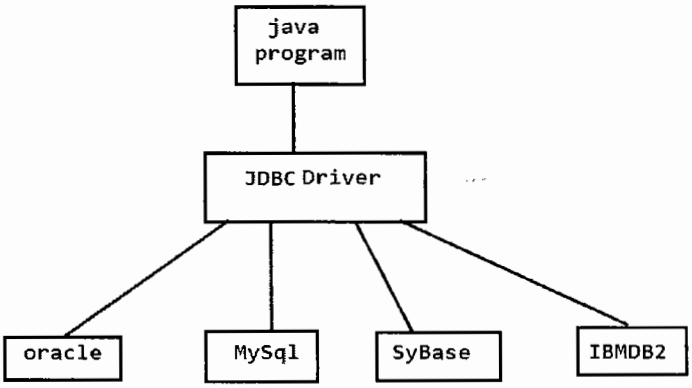


JDBC API (java database connectivity)

- To write a java program which communicate with any db without changing the code sun micro system have released JDBC API
-Once an API released anybody can provide the implementation
-This JDBC API implementation we can call JDBC driver or JDBC s/w



- Following is the simple architecture for java program which communicate with any DB



-to write a JDBC program we use following 2 packages which are available as a part of java s/w

java.sql

Interfaces	classes
Driver	DriverManager
Connection	Types
Statement	Date
PreparedStatement	
CallableStatement	
ResultSet	
ResultSetMetadata	
DatabaseMetadata	

javax.sql

interfaces
DataSource
RowSet

-If we want to a JDBC Program we have to follow following 5 steps

step1: Registering the DB Driver

step2: Getting the Connection from DB Server

step3: Creating Statement object

step4: Sending SQL queries to DB Server

step5: closing the Connection

? what is the Driver class

-Driver class is a class which implements the Driver interface of java.sql package.

-Driver class name is differernt from database to database

-For example oracle people have given Driver class name as " oracle.jdbc.driver.OracleDriver "

Eg:

```
Driver d = new oracle.jdbc.driver.OracleDriver();
```

-but in the case of my sql db server Driver class name is "com.mysql.jdbc.Driver"

Eg:

```
Driver d = new com.mysql.jdbc.Driver();
```

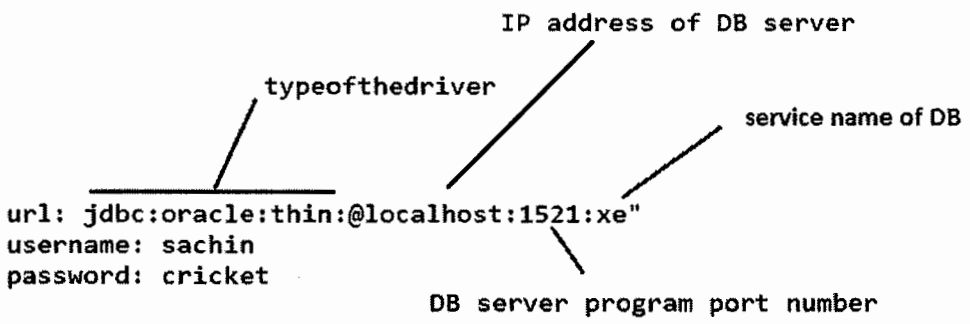
```
//wap to register the Database Driver
import java.sql.*;
class DBConnection1{
public static void main(String args[])throws SQLException{
 Driver d = new oracle.jdbc.driver.OracleDriver();
DriverManager.registerDriver(d);
System.out.println("db driver registered ");
}
}
```

//wap to get the db connection from oracle db

- If we want to connect to oracle db we have to use getConnection() method of DriverManager class like follows

Eg:

```
DriverManager.getConnection("url","username","password");
```



Note: db url is also different from db to db

```
import java.sql.*;
class DBConnect{
public static void main(String args[]) throws SQLException{
        Driver d = new oracle.jdbc.driver.OracleDriver();
        DriverManager.registerDriver(d);
        System.out.println("Driver is Registered....");
        Connection conn =
        DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","library","books");
        System.out.println("Connection is Given....");
        System.out.println(conn.getClass());
        }
    }
}
```

//wap to create the table inside the database

-once we get the connection we can perform any database operations like creating tables ,inserting the records , selecting the records,....

- we can say database operations as CURD operations

C- creating U- Update R- Retrive D- Delete

```
import java.sql.*;
class CreateTable{
  public static void main(String args[]) throws SQLException{
//step1: Registering the DB Driver
DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
    System.out.println("Driver is Loaded");
//step2: Getting the Connection from DB Server
Connection conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","library","books");
    System.out.println("Connection is OK");
// step3: Creating Statement object
   Statement st = conn.createStatement();
```

```
// step4: Sending SQL queries to DB Server
st.executeUpdate("create table stud(rno number(2),name varchar2(10),address varchar2(10))");
  System.out.println("Table is created");
//step5:  closing the Connection
   conn.close();
  }
}
```

-before we compile jdbc programs first we need to set classpath to ojdbc14.jar file which contains all the implementation classes of JDBC API like follows,

```
>set   classpath=C:\oraclexe\app\oracle\product\10.2.0\server\jdbc\lib\ojdbc14.jar;.;
```

(or)

- copy ojdbc14.jar into our working folder then we can set directly classpath like follows

```
> set classpath=objbc14.jar;.;
```

creating db user

- open sql command prompt

```
sql> connect
   username: system
   password:inetsolv
sql>create user library  identified by books
sql>grant connect,resource to sachin
```

Types of Sql Statements

- In Jdbc Sql queries are classified into Following 2 types

1. select queries

- Queries begin with a keyword called select are called as select queries.
- to execute select queries we have to use executeQuery() method available in Statement interface

syntax:

```
     ResultSet    executeQuery(String sql)
```

2. non-select queries

-Queries which are not begin with a keyword called select are called as non-select queries.

-to execute non-select queries we have to use executeUpdate() method available in Statement interface

sytnax:

```
     int    executeUpdate(String sql)
```

here executeUpdate() returns an integer value that indicates how many rows are effected with particular query.

```
//wap to insert the records into db tables
import  java.sql.*;
class  InsertRecords{
public  static  void  main(String args[]) throws  SQLException{
//step1:
//step2:
// step3:
// step4:
String  sql = "insert into stud values(1,'sachin','hyd')";
```

```
System.out.println(sql);
 st.executeUpdate(sql);
//step5
conn.close();
}
}
//wap to insert the records into db tables  using keyboard
import  java.sql.*;
import  java.io.*;
class  InsertRecords1{
public  static  void  main(String args[]) throws  SQLException,IOException{
//step1:
//step2:
// step3:
// step4:
BufferedReader  br = new BufferedReader(new InputStreamReader(System.in));
System.out.println("Enter Any RNO,NAME,ADDRESS");
int rno = Integer.parseInt(br.readLine());
String name = br.readLine();
String addr = br.readLine();
String sql="insert into stud values("+rno+",'"+name+"','"+addr+"')";
System.out.println(sql);
 st.executeUpdate(sql);
//step5
conn.close();
}
}
//wap to update the salaries of employees with 500
import  java.sql.*;
class  UpdateRecords{
public  static  void  main(String args[]) throws  SQLException{
//step1:
//step2:
//step3:
//step4:
String  sql = "update  emp set salary = salary+500";
System.out.println(sql);
int  n  =  st.executeUpdate(sql);
System.out.println(n+ " records are updated");
//step5
conn.close();
}
}
```

```
//wap to delete the student whose rno = 5
import  java.sql.*;
class  DeleteRecords{
public  static  void  main(String args[]) throws  SQLException{
//step1:
//step2:
// step3:
// step4:
String  sql = "delete from stud where rno=5";
System.out.println(sql);
int  n  =  st.executeUpdate(sql);
System.out.println(n+ " records are deleted");
//step5
conn.close();
}
}
```

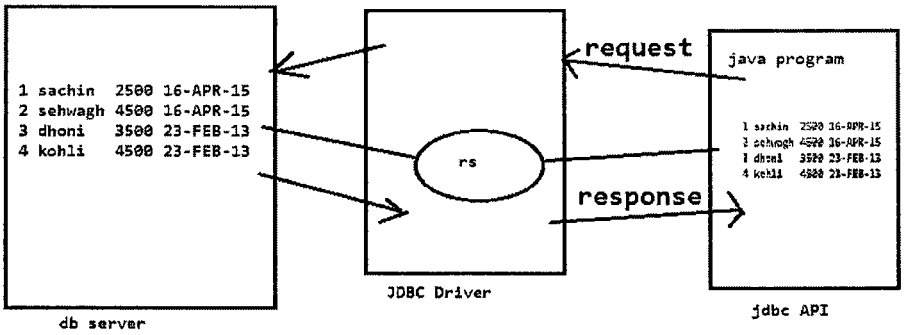
Note:

here executeUpdate() method returns integer value which indicates the number of rows that are deleted or updated or inserted,....

```
//wap to select the records of employee  table
import  java.sql.*;
class SelectRecords{
  public  static  void  main(String args[]) throws SQLException{
//step1
//step2
//step3
//step4
ResultSet  rs =  st.executeQuery("select * from emp");
System.out.println("EMPNO\tENAME\tJOB\tSALARY");
System.out.println("-------------------------------------");
while(rs.next()){
int  eno  =  rs.getInt(1);
String  name = rs.getString(2);
String jo = rs.getString(3);
double sl =  rs.getDouble(4);
System.out.println(eno+"\t"+name+"\t"+jo+"\t"+sl);
}
//step5
conn.close();
 }
}
```

Note:

-when we send select query to db server then db server will send selected records to jdbc driver and now jdbc driver will convert selected records into ResultSet object and given to java program.



-when we select the data the result set pointer is located before the first record in the list of selected records.

boolean next()

-this method returns true if the next record is avaiable and moves the resultset pointer to next record.

-otherwise if records are not available it returns false.

-After moving the resultset pointer to the particular record, to get the data of that record we have to use following getter methods

syntax:

XXX getXXX(coloumnindex)

XXX getXXX(coloumnname) // recommended

db datatypes	java datatypes	ResultSet methods
Number	Integer	getInt(colname/index)
Number(P,S)	Double	getDouble(colname/index)
DATE	java.sql.Date	getDate(colname/index)
Char or varchar2()	String	getString(colname/index)

Note:

1. column index allways begin with 1
2. it is recommended to write column name to improve readability
3. if we call ResultSet get Methods before invoking next() method we get Exception like
   SQLException:ResultSet.next was not called
4. If we write column name or column index inside ResultSet getter Methods which are not available then we get Exception like SQLException: Invalid column name or SQLException: Invalid column index
5. if we call ResultSet get Methods after visiting all records we get Exception like
   SQLException:Exhausted Resultset (if we use column names)
   or
   SQLException: Invalid column index (if we use column indexes)

Projection

- projection means selecting the records according to requirement by filtering columnwise or rowwise

Eg:

1. select only ename,sal columns from emp table

//step4

```
String sql = "select ename,sal from emp";
```

```
System.out.println(sql);
ResultSet rs = st.executeQuery(sql);
System.out.println("ENAME\tSALARY");
while(rs.next()){
  String name = rs.getString(1);
   double sal = rs.getDouble(2);
 System.out.println(name+"\t"+sal);
}
```

2. select only employeess whose salary is 4500

```
//step4
String sql = "select * from emp where sal=4500";
System.out.println(sql);
ResultSet rs = st.executeQuery(sql);
//write jdbc program to select the records using do..while loop
//write jdbc program to select the records using for loop
```

PreparedStatement

-PreparedStatement is another statement object which is also used to execute all sql select and non-select statements

-the main advantage of PreparedStatement is it will improve the performacne when we want to execute same the query multiple times compared to Statement object.

-but if we send different queries using PreparedStatement then there is no difference b/w PreparedStatement and Statement but in this senario Statement object is preferable one.

?how the Performance will be improved using PreparedStatement

when we send a query to database server then database server will perform following operations for every new query foreach time

1. Query tokenization:

it means it will break our query into multiple tokens and collection of tokens will be given for parsing

2. Query parsing:

Query parsing means it will check all the tokens of query whether they are valid db keywords or not if valid it will convert the query into db understandable query format otherwise gives an Error or exception.

3. Query optimization

Query optimization means it will check all the algorithms and attach the appropriate algorithm with our query which takes less time and less memory

4. Query execeution

once optimization is completed then the query will be executed and result will be sent to java program.

conclusion

so if we use Statement object then db server will perform preceeding 4 steps everytime for every query we may send same query multiple times or different queries multiple times.

but if we use PreparedStatement object and sending same query several times then db server will perform preceeding 4 steps only for first time but second time onwards same query will be sent with out doing any preceeding 4 steps so that performance will be improved in Prepared Statement.

but if we send different queries multiple times using PreparedStatement object then db server will perform preceeding 4 steps every time for every new query so inthis case no performance will be improved.

inserting 5000 records using Statement object

```
import java.sql.*;
class StatementDemo{
public static void main(String args[]) throws SQLException{
//step1:
//step2:
//step3:
Statement st = conn.createStatement();
//step4:
long start = System.currentTimeMillis();
for(int i=1;i<=5000;i++){
st.executeUpdate("insert into cust values("+i+",'aaa','"+(100+i)+"')");
}
long end = System.currentTimeMillis();
System.out.println("Time Taken:"+(end-start)+"ms");
//step5:
conn.close();
}
}
```

inserting 5000 records using PreparedStatement object

```
import java.sql.*;
class PreparedStatementDemo{
public static void main(String args[]) throws SQLException{
//step1:
//step2:
//step3:
PreparedStatement pst=conn.prepareStatement("insert into cust
values(?,?,?)");
//step4:
long start = System.currentTimeMillis();
for(int i=1;i<=5000;i++){
//setting the values to positional parameters or place holders
  pst.setInt(1,i);
  pst.setString(2,"abc"+i);
  pst.setString(3,1000+i);
  pst.executeUpdate();
}
long end = System.currentTimeMillis();
System.out.println("Time Taken:"+(end-start)+"ms");
//step5:
conn.close();
}
}
```

positional parameters

-when we use PreparedStatement it should contain query with positional parameters

- positional parameters are specified using ? symbol

- positional parameters are used to supply the values to query

- to suppy the values for positional parameters we have to use following setter methods

db datatypes	java datatypes	PreparedStatement methods
Number	Integer	setInt(position,intvalue)
Number(P,S)	Double	setDouble(position,doublevalue)
DATE	java.sql.Date	setDate(position,datevalue)
Char/varchar2	String	setString(position,Stringvalue)

Note:

positional parameters numbering begin from 1

assignment

-do other CURD operations like create, update, select, delete,... using PreparedStatement

CallableStatement

-CallableStatement is another statement object which is especially used to execute the database stored procedures

-CallableStatement also improve the performance

- In database we can have following 2 types of database stored procedures

1. procedure does not return any value directly
2. function returns a value

creating a procedure in sql

-open sql command prompt

```
sql> ed proc1;
create or replace procedure insert_records
is
begin
insert into emp values(10,'a.sachin',3000);
insert into emp values(20,'b.sachin',4000);
insert into emp values(30,'c.sachin',7000);
insert into emp values(40,'d.sachin',6000);
insert into emp values(50,'e.sachin',2000);
end;
/
sql> @proc1;
```

-executing the procedure in sql command command prompt

```
sql>execute insert_records
```

jdbc program to call procedure

```
import java.sql.*;
 class CSTDemo{
  public static void main(String args[]) throws SQLException{
```

```
//step1
//step2
//step3
CallableStatement cst = conn.prepareCall("{call insert_records}");
  cst.execute();
 System.out.println("go and see in db records are inserted...");
 conn.close();
   }
 }
```

2. procedure with parameters

creating procedure in sql

```
create or replace  procedure  insert_records(eno in number,name in varchar2,job varchar2,sal in number)
is
begin
insert into emp values(eno,name,job,sal);
end;
```

executing in sql

```
sql>execute   insert_records(12,'bbb','xxx',2000);
```

jdbc program to execute procedure which takes in parameters

```
import java.sql.*;
import java.util.*;
 class  CSTDemo1{
  public  static   void  main(String  args[]) throws SQLException{
//step1
//step2
//step3
CallableStatement cst = conn.prepareCall("{call  insert_records(?,?,?,?)}");
   cst.setInt(1,15);
   cst.setString(2,"sachin");
   cst.setString(3,"crick");
   cst.setDouble(4,3000.0);
  cst.execute();
 System.out.println("ok ok");
   cst.close();
   }
 }
```

//wap to call procedure which takes out parameters

creating procedure in sql

```
create  or  replace procedure do_add( a in number,b in number,  c out number)
is
begin
c:=a+b;
end;
/
```

executing procedure in sql

-If we want to use out paramter values we have to use bind variables

defining bind variables

```
sql> variable  n number;
```

executing the procedure using bind variables;

```
sql> execute  do_add(100,200,:n);
```

print the value of bind variables

```
sql> print  n;
```

jdbc program to call procedure which has out parameters

```
import  java.sql.*;
 class  CSTDemo2{
   public  static   void  main(String  args[]) throws SQLException{
//step1
//step2
//step3
CallableStatement cst = conn.prepareCall("{call do_add(?,?,?)}");
    cst.setInt(1,150);
    cst.setInt(2,350);
    cst.registerOutParameter(3,Types.INTEGER);
    cst.execute();
int result =    cst.getInt(3);
  System.out.println("the value return from DBS: "+result);
    conn.close();
     }
  }
```

- when we register the Outer Parameter internally jdbc driver declare one bind variable and once the procedure executed the out parameter value will be stored into particular bind variable.

Note:

when we use positional parameters or place holders insdide any PreparedStatement or CallableStatement each positional parameter will be converted into bind variable by JDBC Driver when setter method set the value to positional parameters then JDBC Driver stores the value inside those bind variables. Once the values are set to positional parameters then we can recieve those values using getter methods.

jdbc program to call functions

creating a function in sql

```
create  or  replace  function do_mul(a in number,b in number)
return  number
is
begin
return  a*b;
end;
/
```

executing a function in sql

```
sql>variable n number;
sql>execute :n:=do_mul(30,40);
sql>print n;
```

or

```
sql> execute dbms_output.put_line(do_mul(30,40));
```

or

```
sql>  select do_mul(10,20)  from dual;
```

jdbc program to call function

```
import  java.sql.*;
 class  CSTDemo3{
   public  static   void  main(String  args[]) throws SQLException{
//step1
//step2
//step3
CallableStatement cst=conn.prepareCall("{? = call do_mul(?,?)}");
    cst.registerOutParameter(1,Types.INTEGER);
    cst.setInt(2,15);
    cst.setInt(3,3);
   cst.execute();
  int result =   cst.getInt(1);
  System.out.println("the value return from DBS: "+result);
    cst.close();
     }
  }
```

-procedure never returns a value direclty but if we want to return from procedure we have to use out parameters

-but function allways returns a value noneed of any out parameters -we can also call this user defined function using normal Statement Object

Eg:

```
Statement st=conn.createStatement();
ResultSet rs=st.executeQuery("select do_mul(10,2) from dual");
if(rs.next()){
int  res = rs.getInt(1);
System.out.println("multiplication ="+res );
}
```

Jdbc program to call predefiened functions

-To call the predefined functions we dont require any CallableStatement we can directly call them using normal Statement Object

```
import  java.sql.*;
class SelectSumFunction{
   public  static   void main(String  args[]) throws SQLException{
//step1
//step2
//step3
//step4
String  sql = "select sum(sal) from emp";
ResultSet  rs = st.executeQuery(sql);
```

```
if(rs.next()){
int  total_sal = rs.getInt(1);
 System.out.println("total salary: "+total_sal);
}
  conn.close();
  }
}
```

Types of ResultSets

- we can create ResultSet object in following 2 types

1. forward only ResultSet

- Bydefault every ResultSet Forward only ResultSet.
- Forward only ResultSet means we can access the records only in forward direction i.e, first record to last record.

2. bi-directional ResultSet

-Bi directional ResultSet means we can access records in both forward and backward directions.

-Bydefault we get ResultSet as forward only ResultSet but if we want to create ResultSet as bi-directional we have to create our Statement object or PreparedStatement object using following methods

syntax:

```
createStatement(int resultSetType,int resultSetConcurrency)
prepareStatement(String sql,int resultSetType,int resultSetConcurrency)
```

-For resultSetType parameter we can supply following any one of the 3 constants

TYPE_FORWARD_ONLY (default)
TYPE_SCROLL_SENSITIVE
TYPE_SCROLL_INSENSITIVE

-For resultSetConcurrency parameter we can supply following any one of the 2 constants

CONCUR_READ_ONLY (default)
CONCUR_UPDATABLE

```
//wap to demo on Bi-Directional  ResultSet
import  java.sql.*;
class  ResultSetTypeDemo1{
  public  static   void main(String  args[]) throws SQLException{
//step1
//step2
//step3
Statement st = conn.createStatement( ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_READ_ONLY);
String  sql = "select * from emp";
ResultSet  rs = st.executeQuery(sql);
rs.next();
System.out.println(rs.getRow());//1
rs.next();
System.out.println(rs.getRow());//2
rs.next();
System.out.println(rs.getRow());//3
```

```
rs.previous();
System.out.println(rs.getRow());//2
}
}
```

? diff b/w TYPE_FORWARD_ONLY and TYPE_SCROLL_INSENSITIVE

-If ResultSet is Selected With TYPE_FORWARD_ONLY then we can access only in forward direction

-If ResultSet is Selected With TYPE_SCROLL_INSENSITIVE then we can access both forward and backward directions

```
//wap to demo on ResultSet with TYPE_SCROLL_SENSITIVE
import java.sql.*;
import java.io.*;
class ResultSetType2{
public static void main(String args[]) throws SQLException,IOException{
//step1
//step2
Statement st = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_READ_ONLY );
String sql = "select empno,ename,job,salary from emp";
//System.out.println(sql);
ResultSet rs = st.executeQuery(sql);
while(rs.next()){
System.in.read();
System.in.read();
rs.refreshRow();
int eno = rs.getInt(1);
String name = rs.getString(2);
String job= rs.getString(3);
double sal = rs.getDouble(4);
System.out.println(eno+"\t"+name+"\t"+job+"\t"+sal);
}
conn.close();
}
}
```

Note:

-If We want to work with TYPE_SCROLL_SENSITIVE type ResultSet then we must follow following Rules

1. ResultSet must be TYPE_SCROLL_SENSITIVE
2. we must specify column names in the select query instead of * symbol.
3. we must call rs.refreshRow() to get the updated data every time

? TYPE_SCROLL_SENSITIVE vs TYPE_SCROLL_INSENSITIVE

-Both constants are giving us bi-directional ResultSet only.

-If ResultSet is TYPE_SCROLL_INSENSITIVE type then reccords updated in the db can not be updated in the current ResultSet.

-If ResultSet is TYPE_SCROLL_SENSITIVE type then reccords updated in the db can updated in the current ResultSet when we call rs.refreshRow()

```
//wap to demo on  ResultSet with CONCUR_UPDATABLE
import  java.sql.*;
class  ResultSetType3{
public  static  void  main(String args[]) throws  SQLException{
//step1
//step2
//step3
Statement   st = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE );
//step4
String  sql = "select empno,ename,job,salary from emp";
System.out.println(sql);
ResultSet  rs =   st.executeQuery(sql);
rs.absolute(5);
rs.updateString("ename","suresh");
rs.updateRow();
//step5
conn.close();
}
}
```

absolute(int row)

This method used to move the cursor position to specific row and there by we we can insert or delete or update the records using the ResultSet

```
//demo on absolute() method
import  java.sql.*;
class  AbsoluteDemo{
 public  static   void main(String  args[]) throws SQLException{
//step1
//step2
//step3
 Statement  st  =  conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE );
String  sql = "select eno,ename,sal,jdate from emp";
ResultSet  rs = st.executeQuery(sql);
rs.absolute(8) ;
 System.out.println(rs.getString(1));
 System.out.println(rs.getString(2));
 System.out.println(rs.getString(3));
 System.out.println(rs.getString(4));
  conn.close();
  }
}
//inserting the record using ResultSet
import  java.sql.*;
class  ResultSetTypeDemo4{
public  static   void main(String  args[]) throws SQLException{
```

```
//step1
//step2
//step3
Statement st = conn.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);
//step4
String sql = "select eno,ename,sal from emp";
ResultSet rs = st.executeQuery(sql);
rs.moveToInsertRow();
rs.updateInt("empno",4);
rs.updateString("ename","kohli");
rs.updateString("job","crick");
rs.updateDouble("salary",1300);
rs.insertRow();
//step5
 conn.close();
 }
}
```

moveToInsertRow():

this method place the cursor in the particular row in the table where we can insert the new record. If we want to insert the record using ResultSet we must this moveToInsertRow() method first.

```
//deleting the row using ResultSet
import java.sql.*;
class ResultSetTypeDemo5{
public static void main(String args[]) throws SQLException{
//step1
//step2
//step3
 Statement st = conn.createStatement
(ResultSet.TYPE_SCROLL_SENSITIVE,ResultSet.CONCUR_UPDATABLE);
//step4
String sql = "select eno,ename,sal from emp";
ResultSet rs = st.executeQuery(sql);
rs.absolute(10);
rs.deleteRow();
//step5
 conn.close();
 }
}
```

Note:

if the specified row position is not available in absolute() method it throws a SQL Exception

primary key

-bydefault db tables will accept duplicate values and null values which causes for several problems in the future for selection or deletion or updation.

-If we want to solve this problem we have to create tables using primary key for particular column in

the table which never allow null values or duplicate values

-primary key will improve the maintenance of records

-for a table atmost one primary key can be applied

Eg:

SQL> create table student(rno number(3) primary key,name varchar2(10),mno number(10),addr varchar2(10));

assignment

? write jdbc program to insert the records into above table

working with Date type of values

-when ever we want to store date type of value into db tables then we have to mention our date in a format like 'dd-mon-yy' or 'dd-month-yyyy'

```
import java.sql.*;
import java.util.Calendar;
class InsertRecords2{
public static void main(String args[]) throws SQLException{
//step1
//step2
//step3
//step4
/*
Date d = new Date();
int y=d.getYear();
int m=d.getMonth();
int da=d.getDate();
//but Date class from java.util package not recommended to use
*/
Calendar c= Calendar.getInstance();
int y=c.get(Calendar.YEAR);
int m=c.get(Calendar.MONTH);
int da=c.get(Calendar.DATE);
String months[]={"JAN","FEB","MAR","APR","MAY","JUN","JUL", "AUG","SEP", "OCT","NOV","DEC"};
String sql = "insert into cust values(1,'sachin','"+da+"-"+months[m]+"-"+y+"')";
//String sql = "insert into cust values(1,'sachin',sysdate)";
System.out.println(sql);
st.executeUpdate(sql);
conn.close();
}
}
//wap to get the records from cust table
import java.sql.*;
class SelectRecords1{
public static void main(String args[]) throws SQLException{
//step1
//step2
//step3
```

```
//step4
ResultSet  rs =  st.executeQuery("select * from cust1");
System.out.println("CID\tCNAME\tBill Date");
System.out.println("--------------------------");
while(rs.next()){
int  cid  =  rs.getInt("cid");
String  cname = rs.getString("cname");
Date bdate = rs.getDate("bdate");
System.out.println(cid+"\t"+cname+"\t"+bdate);
}
```

Note:

To retrieve the date type value from database we have to use getDate() and its return type is java.sql.Date

assignments

//wap to store time into database tables

//wap to username and password from keyboard and check whether they are available in the users table or not

create table users(uname varchar2(10),pwd varchar2(10),role varchar2(10));

-we can register db driver in any JDBC program using any one of the following 2 approaches

1. **DriverManager.registerDriver() method**
2. **using Class.forName() method**

Eg:

```
import  java.sql.*;
class  SelectRecords{
public  static  void  main(String args[]) throws  SQLException,ClassNotFoundException{
//step1
Class.forName("oracle.jdbc.driver.OracleDriver");
//step2//step3//step4//step5
}
}
```

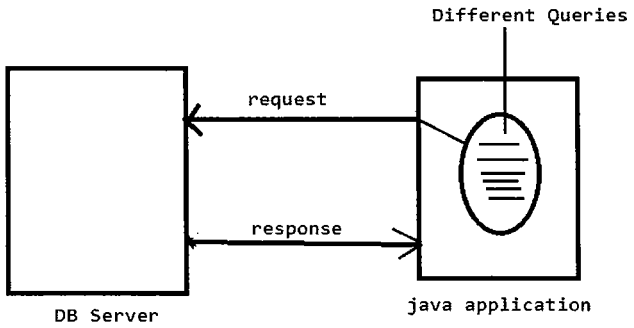
?practice jdbc program where executing multiple queries at a time using Statement Object.

Batch Updates

-Batch Updates are used to send a group of multiple queries to the database at a atime

-In Batch updates we can write any type query and any number of queries

-genrally Batch updates are used to improve the performance because at a time we can send multiple queries to the database



```
//wap to demo on batch updates
import java.sql.*;
class AddBatchDemo{
 public static void main(String args[]) throws SQLException,ClassNotFoundException{
 Class.forName("oracle.jdbc.driver.OracleDriver");
Connection conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","inetsolv","students");
   Statement st = conn.createStatement();
  st.addBatch("insert into cust1 values(1,'aaa',1300.00)");
  st.addBatch("insert into cust1 values(2,'bbb',1200.00)");
  st.addBatch("insert into cust1 values(3,'ccc',2300.00)");
  st.addBatch("insert into cust1 values(4,'ddd',3300.00)");
  st.addBatch("insert into cust1 values(5,'eee',4300.00)");
  st.addBatch("update cust1 set cbill=cbill+500");
  st.addBatch("create table stud11(rno number(2),name varchar2(10))");
   st.executeBatch();
    System.out.println("Records are inserted");
 conn.close();
  }
}
```

void addBatch()

-addBatch() method will adds queries into the Queue of Statement Object.

int[] executeBatch()

- executeBatch() method will execute all the queries added into the Queue in FIFO order
- if any one of the Query is invalid according to SQL it will throw one runtime exception java.sql.BatchUpdateException
- return type of executeBatch() is int[] array which contains the result of each query like how many rows are update or inserted , deleted,..

-to display the result of executeBatch() method we have to use for loop like follows

Eg:

```
  Statement st = conn.createStatement();
  st.addBatch("delete from cust1 where cid=2");
  st.addBatch("insert into cust1 values(7,'fff',1500.00)");
  st.addBatch("update cust1 set bill=bill-200 where cid<=4");
 int result[] = st.executeBatch();
  for(int i=0;i<result.length;i++){
   System.out.println(result[i]);
  }
```

Note:

- using batch updates we can not execute select command and if we do it will throw a runtime exception saying java.sql.BatchUpdateException: invalid batch command: select
- all the queries are executed in FIFO order
- addBatch() must containa any query but not empty otherwise it returns a runtime exception saying java.sql.BatchUpdateException: error occurred during batching: SQL statement to execute cannot be empty

Note

- if any body update and release the new API again one new implementation is provided for the new API
-java people have released JDBC 3.0 then oracle people implementated and named as ojdbjc14.jar
-java people have JDBC 4.0 then oracle people again implementated and named as ojdbjc6.jar
- in jdbc 4.0 one extra feature is added that is driver automatically registered.

working with mysql database

connecting to MySql

-> start menu
-> all programs
-> MySql
-> run mysql command pprompt

password: root

creating database

mysql> create database sachin;

connecting to database

mysql> user sachin;

displaying current databasename

mysql> select databse();

displaying all tables

mysql> show tables;

creating table stud

mysql>create table stud(rno int(2)primary key,name varchar(10));

inserting the record

mysql> insert into stud values(1,"sachin");
mysql> insert into stud values(2,'sachin');

updating the records

mysql> update marks set total=s1+s2+s3;
mysql> update emp set sal=sal+1000 where job='MANAGER';

deleting the records

mysql> delete from stud where rno=5;
mysql> delete from stud;

selecting the records

mysql> select * from stud;
mysql> select * from stud where rno<=4;
mysql> select rno,name from stud where rno<=4;

```
//write a jdbc program to communicate with mysql db
import java.sql.*;
class MySqlCreateTable{
  public static void main(String args[]) throws SQLException{
DriverManager.registerDriver(new com.mysql.jdbc.Driver());
//Class.forName("com.mysql.jdbc.Driver");
    System.out.println("Driver is Loaded");
   Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3306/library","root","root");
    System.out.println("Connection is OK");
```

```
  Statement st =  conn.createStatement();
 st.executeUpdate("create table emp(eno int(2),ename v
archar(10))");
   System.out.println("Table is created");
  st.close();
  }
 }
```

Note:

-To compile the above program we need to set the classpath to the jar file which is released by MySQL DB people like " mysql-connector-java-5.1.16-bin.jar "

-For MySQL DB

Driver class Name: com.mysql.jdbc.Driver

URL: jdbc:mysql://localhost:3306/library(dbname)

assignments

-practice all the queries with mysql

Transactions

-Transaction means set of operations(insert or delete or update) that are performaed in a particular time.

-Bydefault every operation that have executed will automatically saved in any JDBC program.

-But if we want to control the transaction according to our requirement then we have to use setAutocommit() method of Connection interface like follows

syntax:

```
conn.setAutoCommit(false);
```

-Every transaction will contain a starting point and end point

-Every transaction will contain following 2 states

1. success state 2. failure state

-If all the tasks are executed successfully in a transaction then it is called as success state

-But if any one task is not executed successfully in a transaction then it is called as failure state

```
//wap to demo on our own transactions
import java.sql.*;
class MyTransaction{
 public static void main(String args[]) throws Exception{
 Class.forName("oracle.jdbc.driver.OracleDriver");
 Connection conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","library","books");
conn.setAutoCommit(false);
  Statement st =  conn.createStatement();
st.executeUpdate("insert into cust1 values(1,'aaa',1300.00)");
st.executeUpdate("insert into cust1 values(2,'bbb',1200.00)");
st.executeUpdate("insert into cust1 values(3,'ccc',2300.00)");
st.executeUpdate("insert into cust1 values(4,'ddd',3300.00)");
st.executeUpdate("insert into cust1 values(5,'eee',4300.00)");
conn.commit();
st.executeUpdate("delete from cust1 where cid=5");
conn.rollback();
st.executeUpdate("update cust1 set cbill=cbill+500");
```

```
conn.commit();
st.executeUpdate("update cust1 set cbill=cbill+500 where cid=1");
conn.rollback();
conn.close();
}
}
}
```

conn.commit()

-This method should be called if we want to endup the current transaction by saving the current transaction and to start the new transaction.

conn.rollback()

-This method should be called if we want to endup the current transaction by cancelling current transaction and to start the new transaction.

Note:

-we can call these functions multiple times according to the requirement

JDBC program for checking user Validation

first create users table like follows

```
sql> create table users(uname varchar2(10),pwd varchar2(10),role varchar2(10));
```

inserting few records

```
SQL> insert into users values('sachin','sachin','captain');
SQL> insert into users values('dhoni','dhoni','coach');
SQL> insert into users values('kohli','kohli','keeper');
```

Eg1:

```
import  java.sql.*;
import  java.io.*;
class  ValidateUser{
public  static  void  main(String args[]) throws  SQLException,IOException{
//step1
//step2
//step3
BufferedReader  br = new BufferedReader(new InputStreamReader(System.in));
System.out.println("Enter Username");
String u= br.readLine();
System.out.println("Enter Password");
String p= br.readLine();
String  sql="select * from users";
ResultSet   rs = st.executeQuery(sql);
int found=0;
while(rs.next()){
if(u.equals(rs.getString(1))&&p.equals(rs.getString(2))){
String role = rs.getString(3);
System.out.println(u+" Welcome to inetSolv");
System.out.println("Your Role:"+role);
found=1;
```

```
break;
}
}
if(found==0){
System.out.println("Invalid User name and password");
}
conn.close();
}
}
```

Eg2:

```
import java.sql.*;
import java.io.*;
class ValidateUser{
public static void main(String args[]) throws SQLException,IOException{
//step1
//step2
//step3
BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
System.out.println("Enter Username");
String u= br.readLine();
System.out.println("Enter Password");
String p= br.readLine();
String sql="select role from users where uname='"+u+"'and pwd='"+p+"'";
ResultSet rs = st.executeQuery(sql);
if(rs.next()){
System.out.println(u+" Welcomet to inetSolv");
System.out.println("Yuur Role:"+rs.getString(1));
}
else{
System.out.println("Invalid User name and password");
}
conn.close();
}
}
```

Note:

-In a single java aprogram we can create any number of connections with different dbs and we can also create any number of different types of Statement objects

assignment

//write jdbc program to migrate data from oracle db to mysql db

MetaData

-MetaData means data about other data

-By using Meta data we can get more information about tables db,...

-We have following 3 type of MetaData in JDBC

1. ResultSetMetaData 2. DatabaseMetaData 3. ParameterMetaData

1. ResultSetMetaData

-ResultSetMetaData is used to get the metadata about tables like getting number of columns of the table waht are names of each column and thier data type,.....

-If we want to create ResultSetMetaData we have to use a method called getMetaData() of ResultSet object

```
//wap to demo on ResultSetMetaData
import java.sql.*;
class RMDDemo{
public static void main(String args[]) throws SQLException{
Connection conn=null;
Statement st=null;
try{
DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","library","books");
st = conn.createStatement();
String sql = "select * from emp";
System.out.println(sql);
ResultSet rs = st.executeQuery(sql);
ResultSetMetaData rsmd = rs.getMetaData();
System.out.println(rsmd.getColumnCount());
 System.out.println(rsmd.getColumnName(2));
 System.out.println(rsmd.getColumnType(2));//12
 System.out.println(rsmd.getColumnTypeName(2));
String c1 = rsmd.getColumnName(1);
String c2 = rsmd.getColumnName(2);
String c3 = rsmd.getColumnName(3);
String c4 = rsmd.getColumnName(4);
System.out.println(c1+"\t"+c2+"\t"+c3+"\t"+c4);
System.out.println("----------------------------------------");
while(rs.next()){
int eno = rs.getInt(1);
String name = rs.getString(2);
String job= rs.getString(3);
double sal = rs.getDouble(4);
System.out.println(eno+"\t"+name+"\t"+job+"\t"+sal);
}
}
catch(SQLException e){
}
finally{
st.close();
conn.close();
}
}
}
```

Types class

- Types is a class that defines the constants that are used to identify general SQL types which are related to JDBC types.
- Types contains only a list of static constants which represent SQL Datatypes related to JDBC types.
- Types doesnt have any method
- We never create an object for Types

2. DatabaseMetaData

-DatabaseMetaData is used to get the metadata about db like what is the db major version,db minor version, driver major version, driver minor version,...

-Generally DatabaseMetaData is used to check the driver version and db versions whether they are compatible or not.

-If we want to create DatabaseMetaData we have to use a method called getMetaData() of Connection object

```
//wap to demo on DatabaseMetaData
import java.sql.*;
class DBMDDemo{
public static void main(String args[]) throws Exception{
Connection conn=null;
try{
DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","inetsolv","students");
DatabaseMetaData dbmd = conn.getMetaData();
 System.out.println(dbmd.getDatabaseMajorVersion());
 System.out.println(dbmd.getDatabaseMinorVersion());
 System.out.println(dbmd.getDatabaseProductName());
 System.out.println(dbmd.getDatabaseProductVersion());
 System.out.println(dbmd.getDriverMajorVersion()); System.out.println(dbmd.getDriverMinorVersion());
 System.out.println(dbmd.getDriverName());
 System.out.println(dbmd.getDriverVersion());
 System.out.println(dbmd.getJDBCMajorVersion());
if(dbmd.getJDBCMajorVersion()<11){
System.out.println("Sorry i cannot connect to JDBC kindly update your database version 11 or higher versions");
}
else{
System.out.println("You can continue....");
}
}
catch(SQLException e){
}
finally{
conn.close();
}
}
}
```

3. ParameterMetaData

-ParameterMetaData is used to get metadata about positional Parameters used in PreparedStatement or CallableStatement like how many posional parameters are there in the query,what is the datatype,...

-But most of jdbc driver vendors are not implementing all the methods of ParameterMetaData

-If we want to create ParameterMetaData we have to use a method called getParameterMetaData() of PreparedStatement object

```
//wap to demo on ParameterMetaData
  import java.sql.*;
class PMDDemo{
  public static void main(String args[]) throws Exception{
//step1
//step2
PreparedStatement pst = conn.prepareStatement("insert into emp(eno,ename,job,salary) values(?,?,?,?)");
ParameterMetaData pmmd = pst.getParameterMetaData();
 System.out.println(pmmd.getParameterCount());
// System.out.println(pmmd.getParameterType(2));
// System.out.println(pmmd.getParameterTypeName(2));
   }
}
```

Working with Images

-first if we want to store large objects in the database we use CLOB datatype for text files BLOB for binary files which accepts up to 4GB data

Eg:

```
sql>create table animals(aid number(2),aname varchar2(20),image blob);
```

//write a JDBC program to store 1 image into database

StoreImage.java

```
import java.sql.*;
import java.io.*;
class StoreImage{
 public static void main(String args[]) throws Exception{
//step1
//step2
PreparedStatement pst = conn.prepareStatement("insert into animals values(?,?,?)");
 File f = new File("elephant.jpg");
pst.setInt(1,1);
pst.setString(2,f.getName());

FileInputStream fis = new FileInputStream(f);
  pst.setBinaryStream(3,fis,(int)f.length());
 pst.executeUpdate();
conn.close();
  }
}
```

Note: To store the image we have to select the file Using FileInputStream

//write a JDBC program to get the image from database

GetImage.java

```
import  java.sql.*;
import java.io.*;
class   GetImage{
 public  static void  main(String args[]) throws Exception{
//step1
//step2
PreparedStatement   pst   = conn.prepareStatement("select * from animals");
ResultSet   rs = pst.executeQuery();
rs.next();
System.out.println(rs.getInt(1));
System.out.println(rs.getString(2));
FileOutputStream  fos = new FileOutputStream(rs.getString(2));
fos.write(rs.getBytes(3));
conn.close();
 }
}
```

Note:

- To get the image we have to select the file Using FileOutputStream

assignment

- practice the same program to store,get audio file or video files

Types of JDBC Drivers

- If we want to communicate with any database we can use any one of the following 4 types of drivers.

Type1 Driver: JDBC-ODBC Bridge Driver

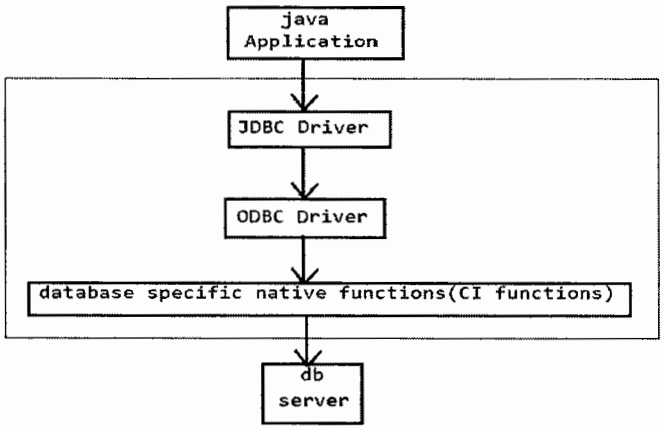
Type2 Driver: Native API Driver

Type3 Driver: Network protocal Driver

Type4 Driver: Database thin Driver/Database protocal Driver/ Pure Java Drivers

1.**JDBC-ODBC Bridge Driver**

- In this type of Driver JDBC function calls are converted into ODBC function calls and ODBC function calls converted into CI functions which are used by particular db to communicate with the db.
- Before we write type1 jdbc program we have to configure our odbc driver for selecting the data osurce with whom we are going to communicate.



steps to configure ODBC

-> start menu
-> control pannel
-> administrative tools
-> (data sources) odbc
-> click on add
-> select required db server driver
-> write name for dsn (myoracledsn,msaccessdsn,..)
-> click on test connection
-> provide service name,username,password
-> click on ok -> ok -> ok

Type1 Driver JDBC program

```
import  java.sql.*;
class  Type1Driver{
public  static  void  main(String  args[])throws SQLException{
          Driver d = new    sun.jdbc.odbc.JdbcOdbcDriver();
            DriverManager.registerDriver(d);
        System.out.println("db driver  registered ");
        Connection conn = DriverManager.getConnection("jdbc:odbc:myoracledsn","inetsolv","students");
        System.out.println("db connection given ");
        Statement  st = conn.createStatement();
        st.executeUpdate("create  table  stud10(rno number(2),name varchar2(10),addr varchar2(15))");
        System.out.println("table is created ");
        conn.close();
    }
}
```

Advantages of Type1 Driver

1.by using Type1 driver we can communicate with any db because all most all databases are implemented ODBC API and contains ODBC drivers

2.we no need to set the class path because type1 drivers are implemented by sun micro systems and given as a part of java s/w.

Disadvantages of Type1 Driver

1.Type 1 Drivers are developed only by sun micro systems no one is intrested in Type1 Drivers

2.Type 1 Driver we perform many number of transformations like JDBC function calls are converted into ODBC function calls and ODBC function calls converted into CI functions which are used by particular db to communicate with the db so that performance willbe decreased.

3.If ODBC suppored " .dll file " is not available in the client system then we have to install separately.

4.Type 1 Driver is platform indipendent because we are using CI functions which are developed based on c-language.

5. ODBC configuration is mandatory.

```
// java program which communicate with ms-access
import  java.sql.*;
class  MACType1Driver{
  public  static  void  main(String  args[]) throws  SQLException{
```

```
    Driver  d = new sun.jdbc.odbc.JdbcOdbcDriver();
          DriverManager.registerDriver(d);
  System.out.println("Driver is loaded.....");
Connection  conn =  DriverManager.getConnection("jdbc:odbc:accessdsn");
  System.out.println("Connection OK");
  System.out.println(conn.getClass());
 Statement  st =   conn.createStatement();
  st.executeUpdate("insert into stud values(1,'sachin',979797,'hyd')");
  System.out.println("record  is  inserted");
  }
}
```

assignment

- write Type 1 driver JDBC programs to communicate with ms access file like inserting ,deleting and updating the records

Note:

-we can not access DDL commands in ms access using jdbc program

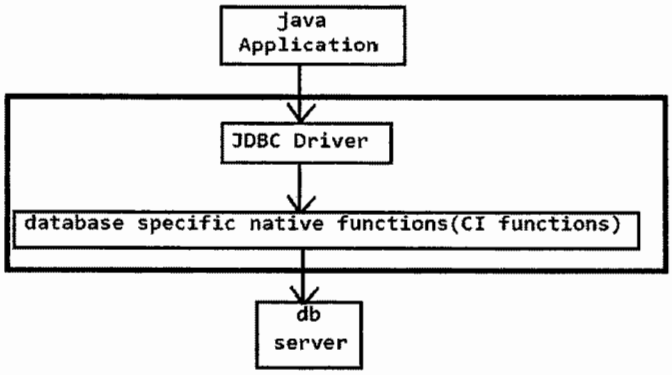
- if we want to work with ms access, ms excel,.. files then it is possible only with type 1 driver.

```
//wap  to select the records from excel sheet
import java.sql.*;
class SelectRecords111{
 public  static  void main(String  args[]) throws  Exception{
DriverManager.registerDriver(new sun.jdbc.odbc.JdbcOdbcDriver());
Connection  conn = DriverManager.getConnection("jdbc:odbc:exceldsn");
Statement  st = conn.createStatement();
String  sql="select * from [Sheet1$]";
System.out.println(sql);
ResultSet  rs =  st.executeQuery(sql);
System.out.println(rs.getClass());
int rn;
String nm;
int mn;
String ad;
System.out.println("RNO\tNAME\tMNO\tAddress");
System.out.println("-------------------------------------------");
while(rs.next()){
rn=rs.getInt(1);
nm=rs.getString(2);
mn=rs.getInt(3);
ad=rs.getString(4);
System.out.println(rn+"\t"+nm+"\t"+mn+"\t"+ad);
}
conn.close();
 }
}
```

2.Type 2 Driver: Native API Driver

-In this Type of Driver JDBC function calls are directly converted into CI functions which are used by particular db to communicate with the db.

-oralce guys are devleoped type2 and type4 drivers and located inside the ojdbc14.jar file




```
//wap to demo on Type2 Driver
import  java.sql.*;
class  Type2Driver{
public  static  void  main(String  args[])throws SQLException{
  Driver d = new    oracle.jdbc.driver.OracleDriver();
DriverManager.registerDriver(d);
System.out.println("db driver  registered ");
Connection conn =DriverManager.getConnection("jdbc:oracle:oci:@localhost:1521:xe","inetsolv","students");
System.out.println("db connection given ");
Statement  st = conn.createStatement();
st.executeUpdate("create  table  stud10(rno number(2),name varchar2(10),addr varchar2(15))");
System.out.println("table is created ");
conn.close();
}
}
```

disadvantages of Type2 Drivers

1.Type2 Drivers are not purely wrtten in java

2.Type2 driver also plat-form dipendent because here we use CI functions

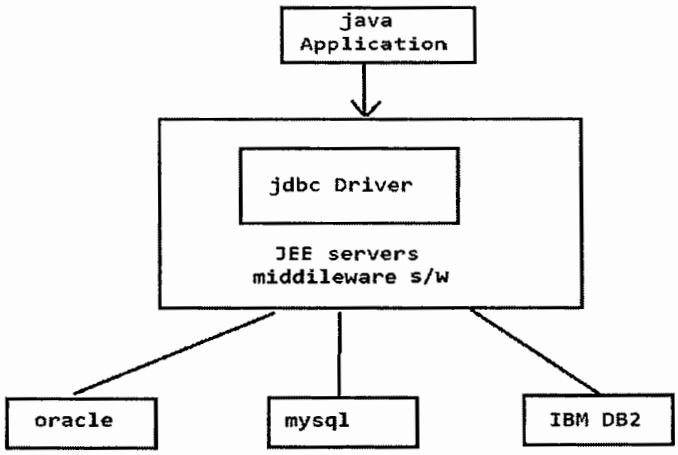
advantages of Type2 Drivers

-comapred to Type1 Driver Type2 Drivers will imporve the performance

-Type2 Driver directly converts JDBC function calls into CI functions

3.Type3 Driver or Network protocal Driver

- Type3 drivers are similar to Type 4 drivers
- but the difference between Type3 and Type 4 drivers is in Type4 drivers programmer is reponsible to arrange the required .jar file other support but in the case of Type 3 drivers we use a middleware s/w which is responsible for containing all the required .jar files and support.
- middleware s/w are nothing but any JEE server
- there are so many JEE servers available in the market like weblogic server,JBOSS,....



JNDI

-JNDI stands for Jva Naming Directory interface

-JNDI is an API which is used to communicate with the directory servers

-Directory servers works same like database like to store the data

differences bw directory servers and database servers

-In the case of database servers data is stored in the format of tables where directory servers store the data in the format of objects

-database servers store the data permanently where directory servers store the data temporarily.

-database servers use SQL to perform the data operations like inserting,deleting,..... , but in directory servers we use predefined methods

-database servers support to store huze amount of data where directory servers are meant for storing huge amount of data.

- simply if we want to store the data one time and use for several times then it is recommended to use directory servers

- in most of the projects we use both database servers and directory servers

- there are so many directory servers available in the market

Eg:

1.LDAP(Lightweight Directory Access protocal) open source

2. ADS (Active Directory Server)

3. NDS (Novell Directory Server)

4. DNS(Domain Naming Server)

.....

-To get the directory servers we no need to install it separately because directory servers are integerated in the application servers

-there are so many application servers (JEE servers) are available in the market.

Eg:

1. WEBLOGIC

2. JBOSS

3. Apache Tomcat Server

4. GLASSFISH

5. Resin .....

procedure to configure domain in weblogic server

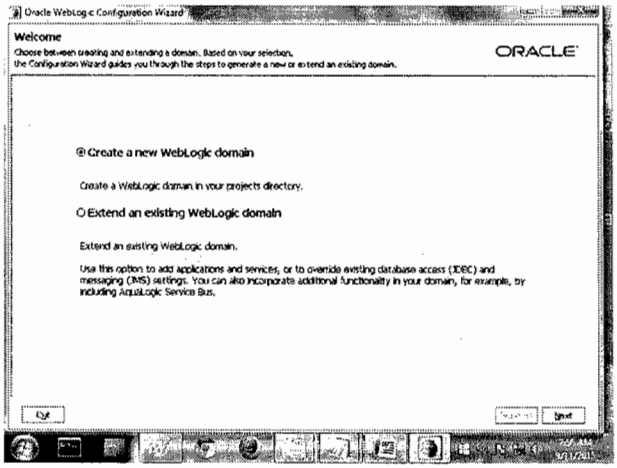
-> start menu

-> oracle web logic

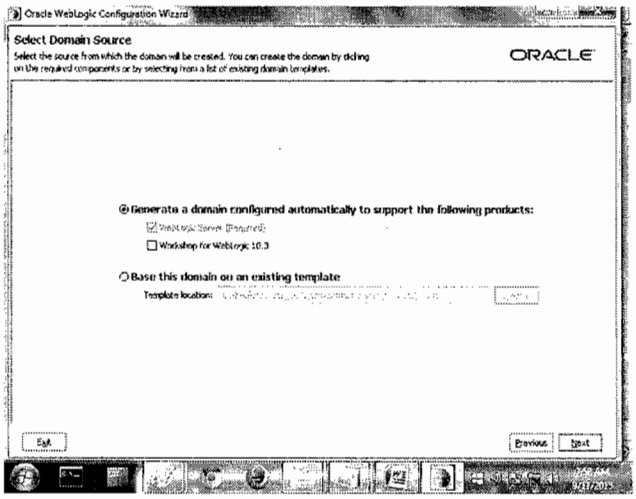
-> weblogic server 10gr3

-> tools

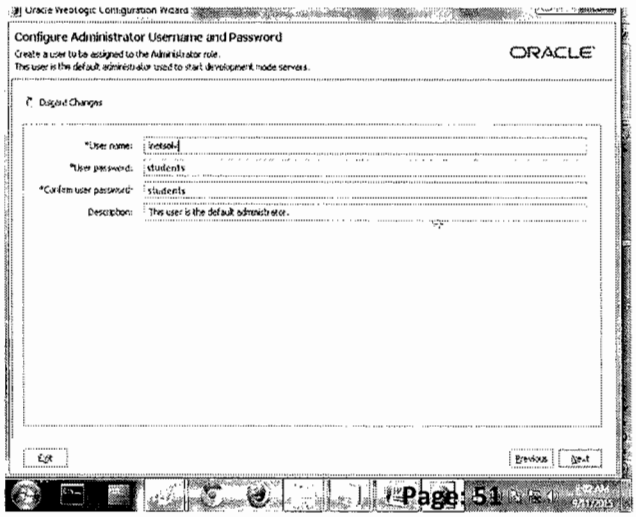
-> configuration wizard



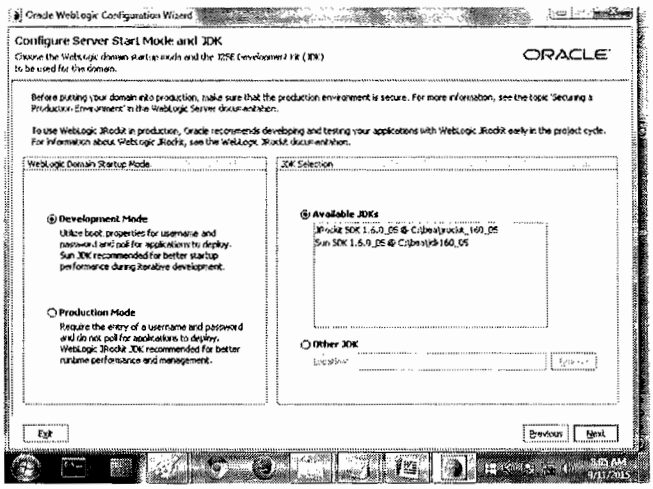
-> Click on next



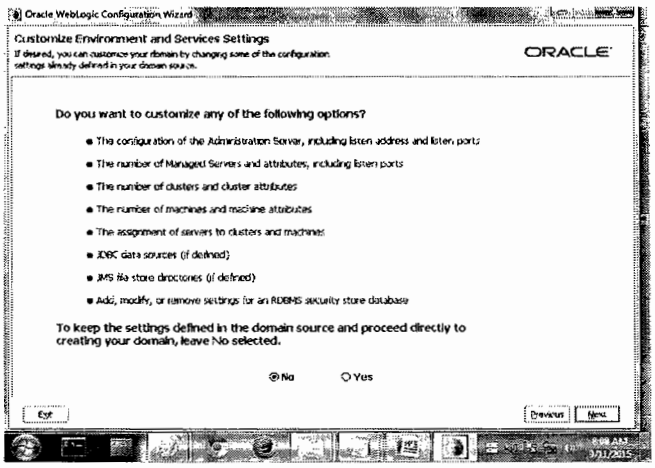
->select generate a domain option and Click on next



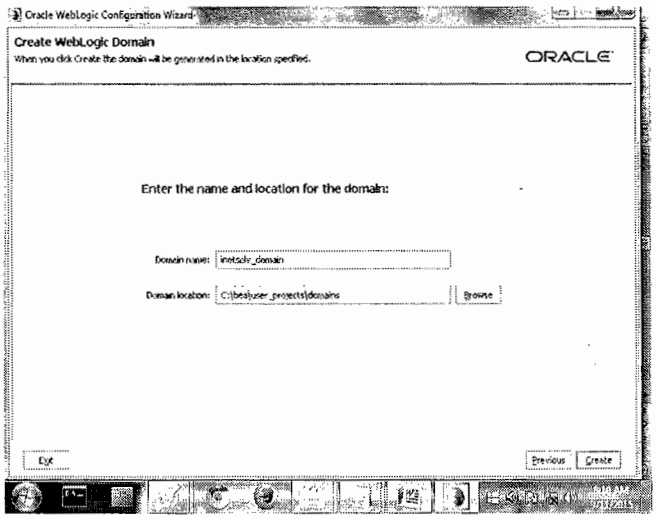
-> Enter the user name and password and Click on next



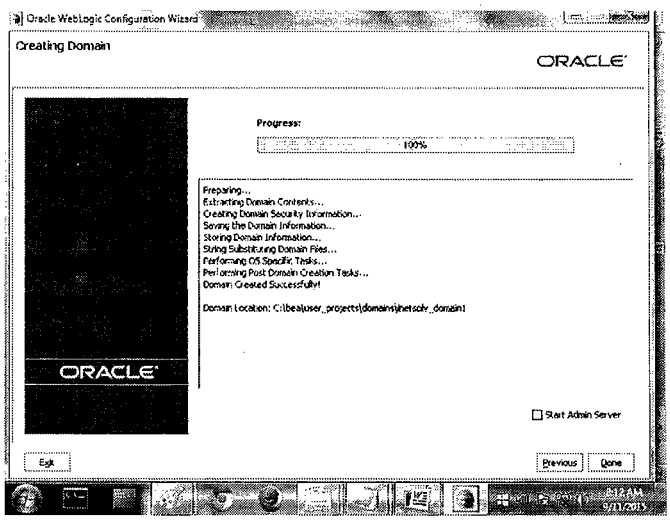
-> Select any JDK and Click on next



-> To get all the features and support Select any No and Click on next



-> Enter the domain name and location amd click on create



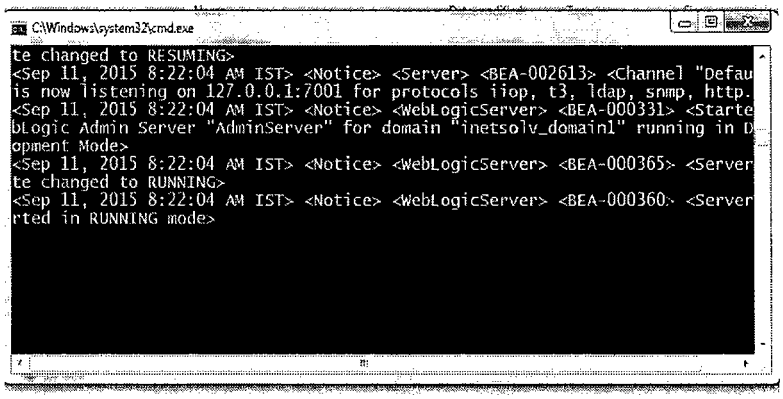
-> configuring oracle weblogic is completed and domain is created click on done to complete the wizard.

-> bydefault all the domains are stored at " C:\bea\user_projects\domains "

-> when we open the domain we can observe .cmd files and .sh file using which we can start the server.

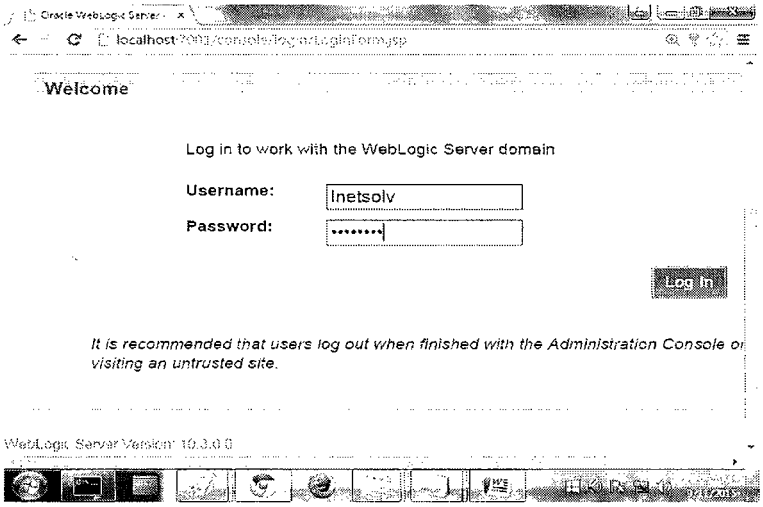
->To start the server which is associated with our domain we just need to double click on " startWebLogic.cmd " file available in the particular domain.

-> If server started following screen will be displayed

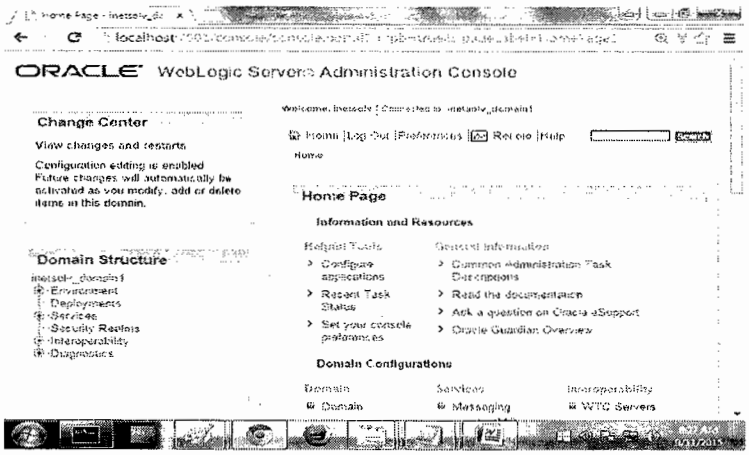


-To open oracle weblogic admin console we have to enter followingURL in side any browser

http://localhost:7001/console

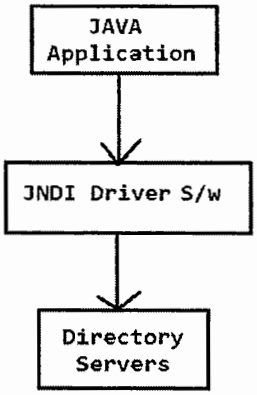


-> Oracle weblogic console page will be displayed like follows



-JNDI is an API which is used to communicate with the directory servers

-Following is the architecture of any JNDI program



-To work with jndi program we have to use following classes and itenrfaces available in javax.naming package

interface	***classes***
Context	InitialContext

-While writing JNDI program we must provide following 4values

JDBC	JNDI
Driver ClassName	INITIAL_CONTEXT_FACTORY
DB URL	PROVIDER_URL
Username	SECURITY_PRINCIPAL
Password	SECURITY_CREDENTIALS

- If we want to write a JNDI program we have to follow following 4 steps

1.create an object for Hashtable class

2.pass the required four values into Hashtable

3.Create an object for Context interface using its Implmentation class called InitialContext by passing the previous Hashtable object

4.adding an object or deleting an object or serach for an object or update an object according to the requirement using bind(), rebind(),unbind(),lookup() methods

1. //wap to store the object on Directory server

void bind(String bindingname,Object obj)

-this method will store the specified object on to the directory server

-if the specified binding name is not avaiable it will store the object successfully

-but if name is already available then it will throw an exception saying "NameAlreadyBoundException "

Eg:

```
import javax.naming.*;
import java.util.Hashtable;
class StoreObject{
 public static void main(String args[]) throws NamingException{
//step1
   Hashtable ht = new Hashtable();
//step2
  ht.put(Context.INITIAL_CONTEXT_FACTORY,"weblogic.jndi.WLInitialContextFactory");
  ht.put(Context.PROVIDER_URL,"t3://localhost:7001");
   ht.put(Context.SECURITY_PRINCIPAL,"inetsolv");//domain username
   ht.put(Context.SECURITY_CREDENTIALS,"students");//domain password
//step3
Context  c = new InitialContext(ht);
//step4
String str = new String("sehwagh");
  c.bind("uname",str);
  }
}
```

Note:

-Before we compile and execute the program we need to set the classpath to wlclient.jar file which available at C:\bea\wlserver_10.3\server\lib like follows

> set classpath=wlclient.jar;.;

- start the weblogic server by double clicking on " startWebLogic.cmd " file avaialable in our domain

```
>javac StoreObject.java
>java StoreObject
```

- now our object will be stored on weblogic JNDI Tree

steps to see the object available on weblogic JNDI Tree

1. Enter the following URL in the browser
   http://localhost:7001/console
2. Enter username and password
   username: inetsolv
   password: students
3. click on Envoronment (availalbe on domain structure panel)
4. click on servers
5. click on server name
6. click on JNDI view Tree
7. on the left side we contain JNDI view Tree structure where we can see all the bindinag names wat we created sofar.

//wap to retrieve the object from Directory server

Object lookup("bindname")

-this method will get object of the specified binding nameob avaialble in the directory server

-if the specified binding name avaiable it will retun the value in the form of Object class Type which need tobe type casted into required object type.

-but if name is not available then it will throw an exception "NameNotFoundException "

Eg:

```
import javax.naming.*;
import java.util.Hashtable;
class GetObject{
 public static void main(String args[]) throws NamingException{
// step1
// step2
// step3
// step4
 Object obj = c.lookup("uname");
 String str =(String) obj; //downcasting
  System.out.println(str);
   }
}
```

//wap to update the object from Directory server

void rebind("bindname",Object newobj)

-this method will modify the content of the object of the specified binding name avaialble in the directory server

-if the specified binding name avaiable it will update the value particular Object

-but if name is not available then it wontt throw any exception instead of that it will create a new object

Eg:

```
import javax.naming.*;
import java.util.Hashtable;
class UpdateObject{
 public static void main(String args[]) throws NamingException{
// step1
// step2
// step3
// step4
String str = new String("suresh");
  c.rebind("xname",str);
  }
}
```

//wap to delete the object from Directory server

void unbind("bindname")

-this method will remove the specified object avaialble in the directory server

-if the specified binding name available then it deletes the specified object -but if name is not available then it will not do any thing

Eg:

```
import javax.naming.*;
import java.util.Hashtable;
class DeleteObject{
 public static void main(String args[]) throws NamingException{
// step1
// step2
// step3
// step4
  c.unbind("uname");
  }
}
```

-By default every object is stored inside the main Context path but we can also create a sub context path and we can also store new objects into this sub context path

-For creating sub context path we have to use a method called createSubcontext() of Context inerface.

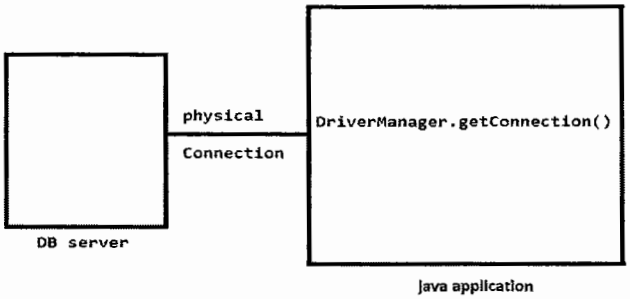
-It is always recommended to use sub context path only which will improve the maintenance.

```
//wap to create sub context path.
import javax.naming.*;
import java.util.Hashtable;
class CreatingSubContext{
 public static void main(String args[]) throws NamingException{
// step1
// step2
// step3
// step4
  c.createSubcontext("inetsolv");
  c.createSubcontext("inetsolv.faculty");
  c.createSubcontext("inetsolv.faculty.corejava");
  c.createSubcontext("inetsolv.student");
  }
}
//wap to store the object into the sub context
import javax.naming.*;
import java.util.Hashtable;
class StoringObjectIntoSubContext{
 public static void main(String args[]) throws NamingException{
// step1
// step2
// step3
// step4
String str = new String("rahaman");
  c.bind("inetsolv.student.sname",str);
  }
}
```

Connection Pool

-when we write a JDBC program using DriverManager we always get physical connection from the database Server.

-In this approach if we dont close the connection after completion of our job then no other people can use the same connection

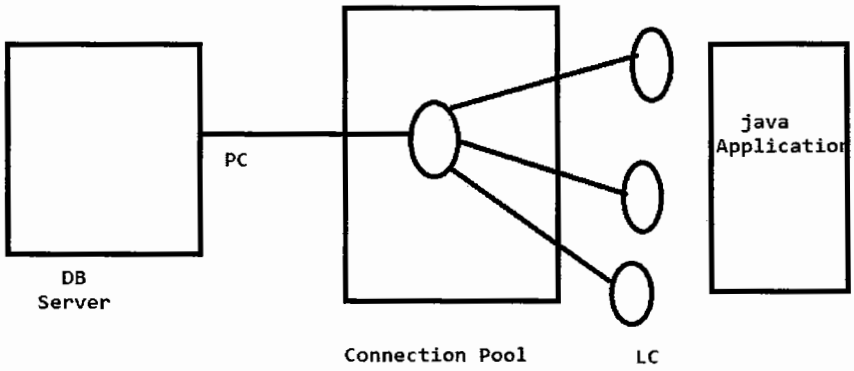


-To resolve this problem we use Connection Pool Program.

-Connection Pool is a java program which manages the connections

-Connection Pool contains a set of Connections

-When we use Connection Pool program first connection pool program will get a physical connection from the database server and next it will give logical connections to the different clients



-In Connection pool our main job is crating an object for datasource by providing db driver ,url, username and password

-from this datasource we can get the connection (logical connections) and next we can perform database operations.

-there are somany Connection Pool program are available in the market.

Eg:

1. DBCP 2. C3P0 3. web logic Connection Pool 4. tomcat Connection Pool 5. JBOSS Connection Pool.....

all these Connection pool programs are released in the form of .jar files which contains all supporting classes and interfaces.

```
//wap to demo on DBCP Connection Pools
import org.apache.tomcat.dbcp.dbcp.*;
import java.sql.*;
class DBCPDemo{
public static void main(String args[]) throws Exception{
  BasicDataSource bds = new BasicDataSource();
```

```
  bds.setDriverClassName("oracle.jdbc.driver.OracleDriver");
  bds.setUrl("jdbc:oracle:thin:@localhost:1521:xe");
  bds.setUsername("inetsolv");
  bds.setPassword("students");
  bds.setInitialSize(3);
Connection con1 = bds.getConnection();
System.out.println("Connection1 is given");
System.in.read();
System.in.read();
Connection con2 = bds.getConnection();
System.out.println("Connection 2 is given");
System.in.read();
System.in.read();
Connection con3 = bds.getConnection();
System.out.println("Connection 3 is given");
System.in.read();
System.in.read();
Connection con4 = bds.getConnection();
System.out.println("Connection 4 is given");
System.in.read();
System.in.read();
Connection con5 = bds.getConnection();
System.out.println("Connection 5 is given");
System.in.read();
System.in.read();
}
}
```

-To compile and execute this program we have to set classpath like follows

set classpath=tomcat-dbcp.jar;.;

```
//wap to demo on C3p0 connecton Pool
import com.mchange.v2.c3p0.*;
import java.sql.*;
class C3P0Demo{
public static void main(String args[]) throws Exception{
ComboPooledDataSource cpds = new ComboPooledDataSource();
  cpds.setDriverClass("oracle.jdbc.driver.OracleDriver");
  cpds.setJdbcUrl("jdbc:oracle:thin:@localhost:1521:xe");
  cpds.setUser("inetsolv");
  cpds.setPassword("students");
  cpds.setMaxPoolSize(3);
Connection con1 = cpds.getConnection();
System.out.println("Connection1 is given");
System.in.read();
System.in.read();
```

```
Connection con2 = cpds.getConnection();
System.out.println("Connection 2 is given");
System.in.read();
System.in.read();
Connection con3 = cpds.getConnection();
System.out.println("Connection 3 is given");
System.in.read();
System.in.read();
//con3.close();
Connection con4 = cpds.getConnection();
System.out.println("Connection 4 is given");
System.in.read();
System.in.read();
Connection con5 = cpds.getConnection();
System.out.println("Connection 5 is given");
System.in.read();
System.in.read();
}
}
```

-To compile and execute this program we have to set classpath like " **set classpath=tc3p0-0.9.1.2;.;** "

Note:

-c3p0 connection pool is not synchronized so multple threads can use this connection pool object at the same time.

-but dbcp Connection Pool is a synchronized one so only one thread can use the particular connection pool

//wap to demo on Weblogic connection pool

configuring weblogic connection pool

-First start the weblogic server for the particular doamin.(inetsolv_domain)

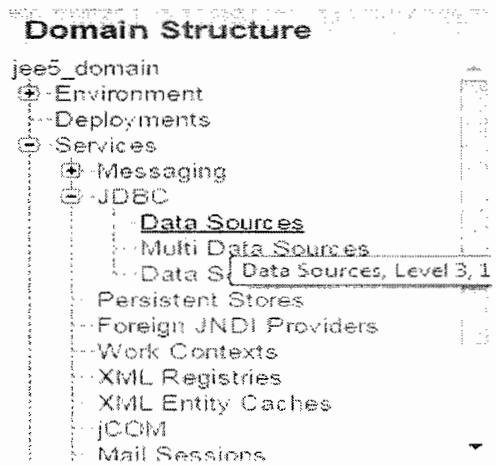
- open weblogic admin console by typing URL like http://localhost:7001/console

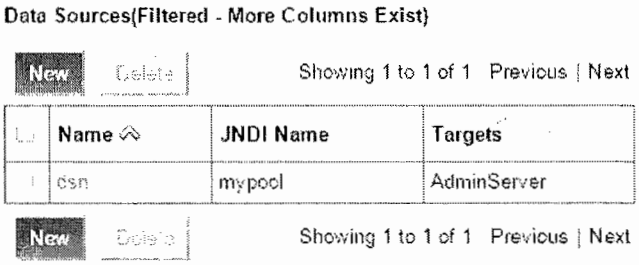
-Enter User name and password

user name: mydomain

password: mydomain

-> select Services -> jdbc -> datasources from domain structure panel





-> -> -> click on new

What would you like to name your new JDBC data source?

* Name: mydsn

What JNDI name would you like to assign to your new JDBC Data Source?

JNDI Name:

mypool

What database type would you like to select?

Database Type: Oracle

What database driver would you like to use to create database connections?

Database Driver: *Oracle's Driver (Thin) Versions:9.0.1,9.2.0,10,11

Back | Next | Finish | Cancel

-> click on next

One-Phase Commit

Back | Next | Finish | Cancel

-> click on next

Connection Properties

Define Connection Properties.

What is the name of database you would like to connect to?

Database Name: xe

What is the name or IP address of the database server?

Host Name: localhost

What is the port on the database server used to connect to the database?

Port: 1521

What database account user name do you want to use to create database connections?

Database User Name: jee5

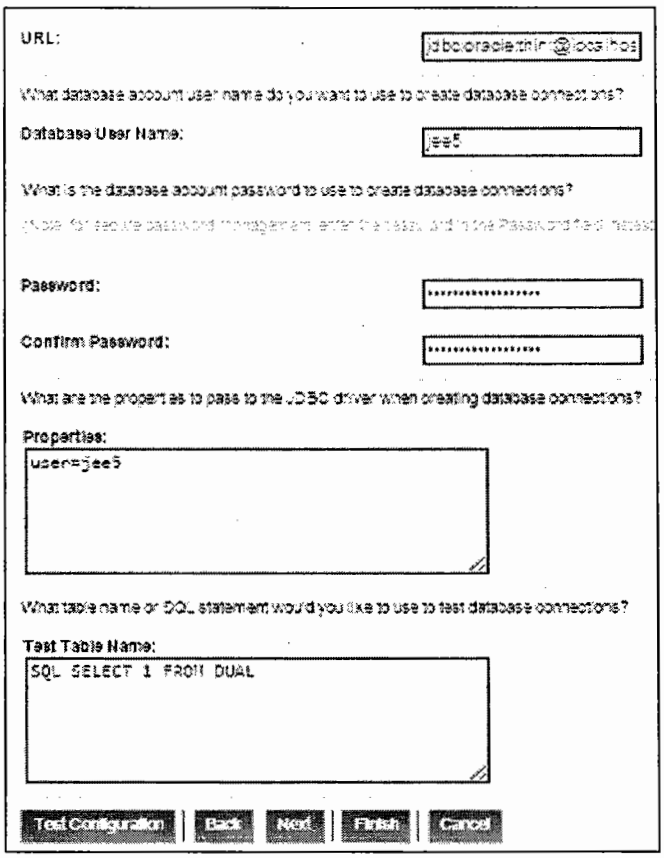
What is the database account password to use to create database connections?

Password: ••••••••

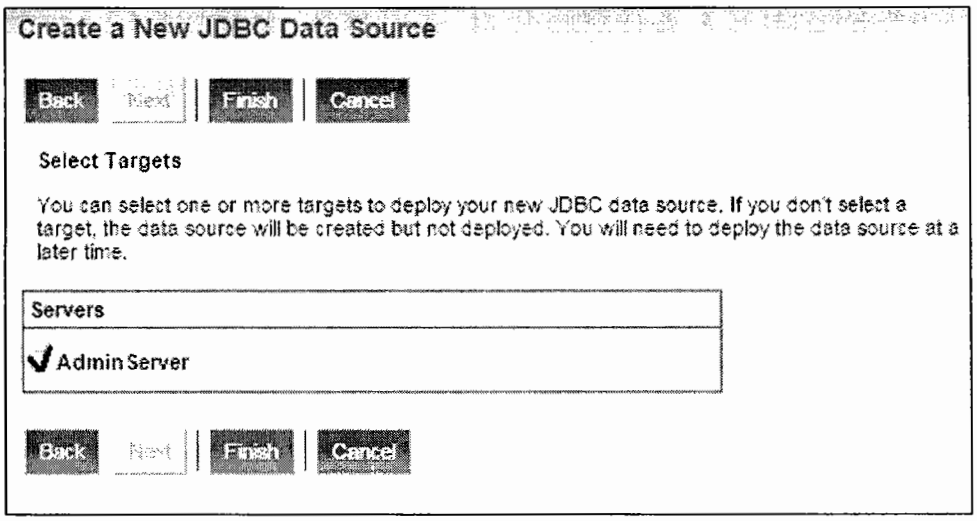
Confirm Password: ••••••••

Back | Next | Finish | Cancel

-> Click on next



-> Click on test configuration to confirm the Connection properties are given valid or not
If connection Test is succeded then click on the next



->click on the admin server and click on finish.

-> We can also see the data source on the jndi view tree by following steps

1. Enter the following URL in the browser
   http://localhost:7001/console

2. Enter username and password
   username: inetsolv
   password: students
3. click on Envoronment (availalbe on domain structure panel)
4. click on servers
5. click on server name
6. click on JNDI view Tree
7. on the left side we contain JNDI view Tree structure where we can see all the bindinag names wat we created sofar.

```
//Program to get connection from web logic connection pool
import javax.naming.*;
import java.sql.*;
import javax.sql.*;
import java.util.*;
class WLCPDemo{
public static void main(String args[]) throws Exception{
  Hashtable ht = new Hashtable();
 ht.put(Context.INITIAL_CONTEXT_FACTORY,"weblogic.jndi.WLInitialContextFactory");
ht.put(Context.PROVIDER_URL,"t3://localhost:7001");
ht.put(Context.SECURITY_PRINCIPAL,"sachin"); //domain user
ht.put(Context.SECURITY_CREDENTIALS,"12345678");//domain pwd
Context ct = new InitialContext(ht);
Object o = ct.lookup("mycp");
DataSource ds = (DataSource) o;
Connection con1 = ds.getConnection();
System.out.println("Connection1 is given");
System.in.read();
System.in.read();
Connection con2 = ds.getConnection();
System.out.println("Connection 2 is given");
System.in.read();
System.in.read();
Connection con3 = ds.getConnection();
System.out.println("Connection 3 is given");
System.in.read();
System.in.read();
Connection con4 = ds.getConnection();
System.out.println("Connection 4 is given");
System.in.read();
System.in.read();
 }
}
```

Note:

To compile and execute this program first we have to copy and locate our program inside the respective domain folder

Eg:

```
\bin> setDomainEnv.cmd
    >javac WLCPDemo.java
    >java WLCPDemo
```

```
//wap to select the records of emp using WLCP
import  javax.naming.*;
import java.sql.*;
import javax.sql.*;
import java.util.*;
class  WLCPDemo1{
public  static  void  main(String args[]) throws  Exception{
//standared JNDI Code
Object  o = ct.lookup("mypool");
DataSource  ds =  (DataSource) o;
Connection conn = ds.getConnection();
Statement   st = conn.createStatement();
String  sql = "select * from emp";
System.out.println(sql);
ResultSet  rs =   st.executeQuery(sql);
System.out.println("EMPNO\tENAME\tJOB\tSALARY");
while(rs.next()){
int eno = rs.getInt(1);
String name = rs.getString(2);
String job= rs.getString(3);
double  sal =  rs.getDouble(4);
System.out.println(eno+"\t"+name+"\t"+job+"\t"+sal);
}
conn.close();
}
}
```

servlets

In java we can develop following 2 types of applications

1. stand alone apps

- stand alone apps means which resides in side the client system and runs in the same client system
-but stand alone apps are having several disadvantages like
-stand alone apps must be installed in all the client systems
-if stand alone apps need to be updated then we need to update in all the client systems so we get maintenance issues.
-stand alone apps require maximum amount of resources from the client system like memory, cpu, hard disk space,....
-to maintain stand alone apps client should have minimum knowledge like how to configure, how to install,....
- generally every stand alone apps contain main() method

Eg:

AWT,Swings,...

2. web based apps

- web based apps means which resides in side the server system and runs inside the client system
- web based apps will resolve the problems of stand alone apps it means we no need to install our app inside all the client systems
- web based apps will consume less amount of memory from the local client system
- we never contain main() method in any web based apps

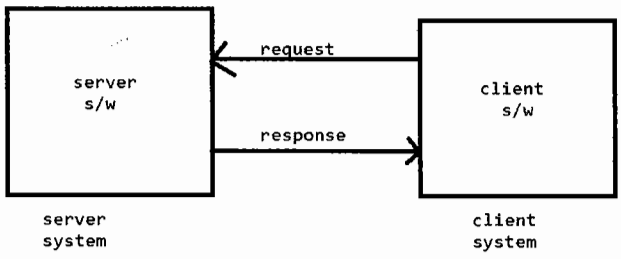
Eg:

Servlets,JSP,....

- if we want to develop a web based apps we need following 2 types of softwares or programs
  1.server s/ws like any WEBLOGIC, Tomcat Server,JBOSS, GLASSFISH, Resin ,.....
  2.client s/ws or any browser like IE,Chrome,Opera,firefox,.....
- we install server s/w inside the server system and client s/w inside the client system
- Now our job is developing a web based apps according to the requirements and locating inside server system using server s/w (Deployment)

http protocal

-a protocal is a set of standared rules which must be followed by 2 systems in order to make the communication possible
- we have different types of protocals like http,https,ftp,udp,tcp/ip....
- if we want to execute the web based apps we must follow http protocals
-a client is a machine which sends request to the server
-a server is a machine which recieve the request and process the request and send back the result to client



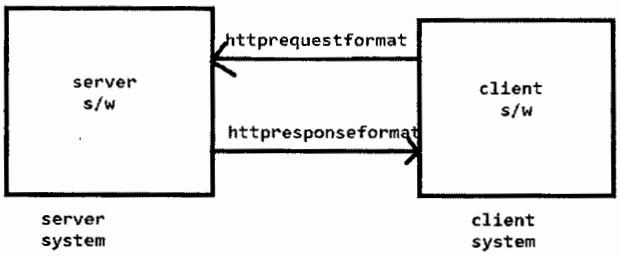
- http protocal divided into following 2 types

1.http request format

when client send request then request will be converted into http request format and send to server.

2.http response format

when server send the response to the client then response will be converted into http response format and send to the client.



1.http request format

initial request line
0 or more Headers
Blank line
http request body(optional)

intial request line

intial request line contains following 3 sections

method	requestreosurce	protocal/version

1. method

- method will indicate what operation has to be performed by the server

Eg:

get,post,delete,trace,.. (server methods)

2. request resource

-request resource is nothing but the resource which has to be processed by sever.

-sometimes requesting resource may be available or may not be available.

3. protocal version

-protocal version indicates what is the protocal version used by client

get	/login.html	http1.0

get	/login.html	http1.1

Headers

-Headers are used to send the extra information about the client to the server like what is theversion of browser or client s/w, what language is used by the client,.....

- Following are the 2 most important Headers

1.User-Agent(contains info about version of browser/client sw)

2.Accept-Language(what language is used by the client)

2.http response format

initial Response line
0 or more Headers
Blank line
http Response body(optional)

initial response line

initial response line also contains following 3 sections

protocal/version	status code	status message

1. protocal version

- protocal version indicates what is the protocal version used by the Server

2. Status code

-Status code indicates what is the rerponse given by server to the client

-developers have given following set of codes for different types of results.

Eg:

100-199 -> information

200-299 -> success

300-399 -> redirect

400-499 -> failed(request resource was not found)

500-599 -> failed (request resource was found but not processed properly)

status message

-For every status code developers have given a specific status messages to understand the status code.

Eg:

200 - ok

404 - not found

Headers

-Headers are used to send the extra information to the client

-The most important Header send by the server to the client is " contentType(text/html) "

-some times we also send errorreport, header-cache info,....

types of protocals

-based on the operation there are following 2 types of protocals

1. state less protocal

state less protocal means a protocal which never remember the conversation happend between a client and server.

Eg:

http, https,....

2. state full protocal

state full protocal means a protocal which allways remember the conversation happend between a client and server.

Eg:

smtp, tcp/ip,.....

Different types of Server Methods

1.get

- if we use get() method then form data will be submitted to the server by appending with URL
- in this case we have no security because our form data will be displayed inside the address bar
- with this get method we can submit limited amount of data (upto 1024kb)

2.post

- if we use post() method then form data will be submitted to the server by appending with header body instead of URL
-in this case we have security because no form data will be displayed inside the address bar
- with this get method we can also submit an unlimited amount of data.

3. put

- put() method will put the resource inside the server, but for security reasons this method is not recommended to use

4. delete

- delete() method will delete the resource from the server, but for security reasons this method is not recommended to use

5. trace or locate

- trace() or locate() methods will search for the resource inside the server, but for security reasons this method is not recommended to use

6. head

-head() method is used to specify the Header portion send by the server to the client. this method is used as a part of request dispatches between servers.

Directory Structure For web based applications

if any body want to develop a web based application he must follow flllowing directory structure.

step1:create a folder with our project name

step2:create a folder inside the project folder with "WEB-INF" as folder name.

step3: create following 3 folders and 1 file inside the "WEB-INF"

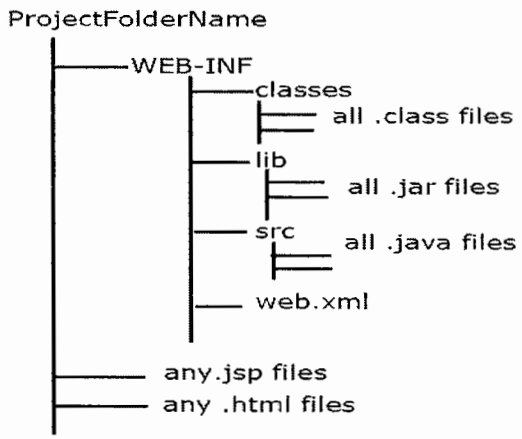
-- classses(folder): it contains all .class files(mandatory)

-- lib(folder): it contains all .jar files(mandatory)

-- src(folder):it contains all .java files(optional)

-- web.xml(file) (mandatory)

step4: place all images,html pages,jsp's ,... directly inside the project folder along with "WEB-INF" folder



? what is WEB-INF folder

- WEB-INF folder is called private folder which contains all .class files, .jar files, configuration files,..
- this folder is called private folder because the files available in this folder can be accessible only By the server.

? what is web.xml file

- web.xml file is called configuration file/deployment descripter which contains the configuration of all servlets that we are using in the project.

? what is deployment

-once we ready with our project directory structure next we have to copy and place this project folder inside the server specific folder and this procedure is called as deployment.

-server specific folder names where we are going to locate our projects are different from server to server.

-in the case of tomcat server we deploy our projects inside the " webapps " folder

-but in the case of weblogic server we deploy our projects inside the " autodeployment " folder

? what is re-deployment

- when ever we change or modify the programs of the project then we need to deploy the project again into the server which is nothing but re-deployment.
- in development procedure we do this re-deployment several times

? what is undeployment

-if we are removing the project folder permanently from the server then it is called as undeployment.

types of web based applications

- we can develop following 2 types of web based applications

1. static web based apps

- If any web page is displaying same result for every client then it is called as static web page
- To develop this kind of applications we can use HTML.

Eg:

login page, register page,....

2. dynamic web based apps

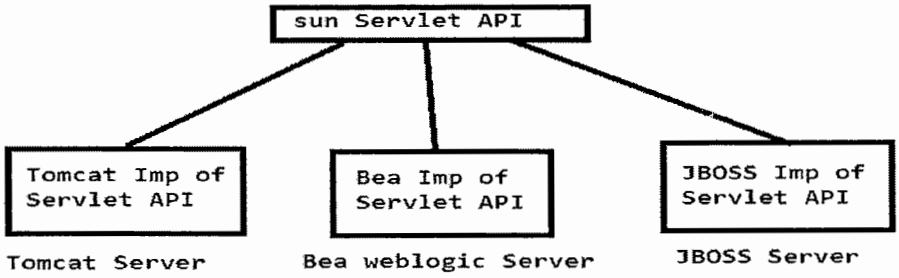
- If any web page is displaying different results for every client dynamically then it is called as dynamic web page
- To develop this kind of applications we can use different technologies like servlets, jsp,asp,php,...

Eg:

customer account page, ...

-To develop a web based application using servlets sun micro system people have released servlet API

-Once an API is released any body can provide the implementation which is called software using which we can develop the applications



-as a part of servlet API following 2 important packags are given using which we can develop servlets.

javax.servlet

Interfaces	***classes***
Servlet ServletRequest ServletResponse ServletConfig ServletContext	GenericServlet

javax.servlet.http

Interfaces	***classes***
HttpServletRequest HttpServletResponse	HttpServlet

-all these 2 packages and all its classes and interface we can found in a .jar file called " servlet-api.jar "

Servlet Interface

If we want to develop a servlet we need to create a class that implements Servlet Interface

procedure to develop a Welcome Servlet

step1: create a class that implements Servlet interface

WelcomeServlet.java

```
import javax.servlet.*;
public class WelcomeServlet implements Servlet{
ServletConfig config;
public void init(ServletConfig config){
this.config=config;
System.out.println("we are in init() method ");
}
public void service(ServletRequest request,ServletResponse response){
System.out.println("we are in service() method ");
}
public void destroy(){
System.out.println("we are in destroy() method ");
}
public ServletConfig getServletConfig(){
System.out.println("we are in getServletConfig() method ");
return config;
}
public String getServletInfo(){
System.out.println("we are in getServletInfo() method ");
return "this is my welcome servlet";
}
}
```

Note: Servlet must be created as public class

step2: compiling the WelcomeServlet program

-If we want to compile servlet program we must set classpath to " servlet-api.jar " like follows ,

```
>set classpath=servlet-api.jar;.;
```

-" servlet-api.jar " file is avaiable inside C:\Program Files\Apache Software Foundation\Tomcat 6.0\lib

step3: configuring the servlet inside the web.xml file

```
<web-app>
 <servlet>
   <servlet-name>aaa</servlet-name>
   <servlet-class>WelcomeServlet</servlet-class>
 </servlet>
 <servlet-mapping>
  <servlet-name>aaa</servlet-name>
  <url-pattern>/welcome</url-pattern>
 </servlet-mapping>
</web-app>
```

step4: create a directory structure for web based application and locate our WelcomeServlet.class file in WEB-INF/classes folder and web.xml file in WEB-INF folder

step5: place the project folder inside the webapps folder of Tomcat Server

step6: start the Tomcat Server by clicking on tomcat6.exe at C:\Program Files\Apache Software Foundation\Tomcat 6.0\bin

step7: Enter the following URL inside any browser's address bar

http://localhost:7777/myproject/welcome

here output will be displayed on the server commandprompt

life cycle methods of Servlet

-When we develop a Servlet we never write any main() method instead of this we writelife cycle methods

-life cycle methods means the entire process of Servlet will be done based on these methods only.

-we have following 3 types of life cycle methods for Any Servlet.

1. init()

-This is the first method executed when ever the object created for the Servlet for the first time.

-In this method we generally write the code for initialization or code which has to be executed for the first time.

-In the total life cycle of Servlet init() method executed only for 1 time.

2. service()

- After init() method server execute service() method.
- in this method we generally write business logic
- service() method executed for multiple times like for every time client send the request.

3.destroy()

-This is the last method executed when Servlet is going to be closed.

- destroy() method executed in following 2 situations
  1. when the server is closed
  2. when the project is un deployed

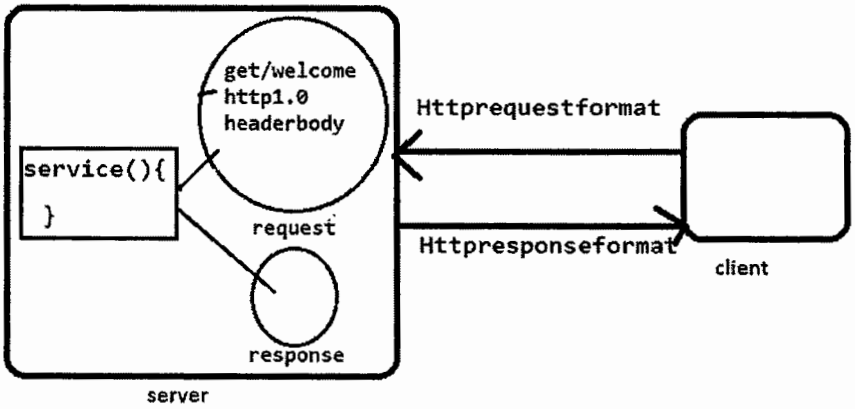
-In this method we generally write the code for memory releasing like closing db connection s, file connections,....

-In the total life cycle of Servlet service()method executed only for 1 time.

? when the servlet object is created

-For the First time when client send the request to Servlet then request will be converted into httprequest format and send to the Server then Server will recieve the request and check already any Servlet object is created if not created Server will create the object and locate inside the Sever JVM next invoke the init() method,next invoke the service() method by creating ServletRequest object and ServletResponse Object

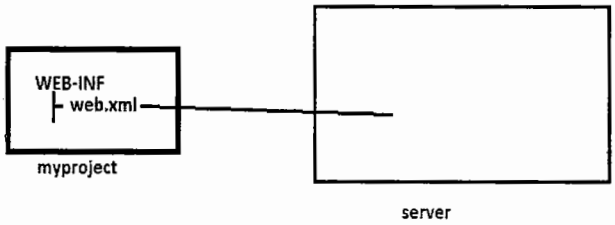
-But if client send the request for the second time it wont create any servlet object and it wont call any init() method server simply calls service() method by creating ServletRequest object and ServletResponse Object.



Note:

-when we deploy the project then no any servlet object is created

-when ever we deploy our project inside the server then server will read the content of web.xml first if web.xml configured properly then deployment willbe success and we can work with our project otherwise project will be deployed with errors and it is not ready for usage.



-To read the content of web.xml Server requires xml parser programs

- In java we have following 2 types of parsers

1. SAX parser (Simple API for XML)
2. DOM parser (Document Object Model)

Note:

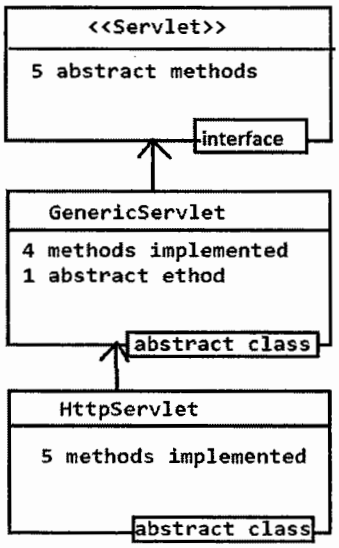
-If requested URL is not configured in web.xml then server send 400 Error which means the requested resource is not available

-If requested URL is configured in web.xml but particular Servlet class is not availalbe inside the classes folder or lib folder then server send 500 Error and throws an Exception like "ClassNotFoundException"

-If requested URL is configured in web.xml and Servlet class also available but Servlet class is not public then server send 500 Error which means the requested resource is available but failed to execute.

-In the last program we are overriding all the 5 methods of Servlet but to simplify the Servlet code java people have given supporting classes like GenericServlet and HttpServlet

-The following is the hierarchy of Servlet interface, GenericServlet and HttpServlet classes.



Generally people will say there are 3 ways to develop 3 ways to develop a Servlet but we can say there are n number of ways to develop a servlet for example we can also develop a new Servlet based on our previous WelcomeServlet.

But sun micro system people recommending us to develop a Servlet based on Http Servlet because we can remove the redundant code of init(), getServletConfig(),...

Following is an example developing a Servlet based on HttpServlet.

```
import javax.servlet.http.*;
public class MessageServlet extends HttpServlet{
public void service(HttpServletRequest request, HttpServletResponse response){
System.out.println("Welcome to Servlet.....");
}
}
```

-In the last 2 examples the output will be displayed on Server command prompt but client can not see output displayed on the Server command prompt.

-It is our responsibility to display the output on the client machine in any web based application.

-When we use System.out.println() then it will executed and out will be displayed on Server. to resolve this problem we can take the help of response object.

-If we write the data on the response object that will be converted into Http response Format and send to the client.

-If we wantto write the data on response object we haveto connect to output stream of the response object for this requirement we can use any one of the following 2 streams

1. using PrintWriter class (recommended)

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

```
import javax.servlet.http.*;
import java.io.*;
public class WishServlet extends HttpServlet{
 public void service(HttpServletRequest request,HttpServletResponse response) throws IOException{
PrintWriter out = response.getWriter();
out.println("Hi Good Evening.....");
  }
}
```

2. using ServletOutputStream class

```
  ServletOutputStream out = response.getOutputStream();
import javax.servlet.http.*;
import javax.servlet.*;
import java.io.*;
public class WishServlet extends HttpServlet{
 public void service(HttpServletRequest request,HttpServletResponse response) throws IOException{
 ServletOutputStream out = response.getOutputStream();
   out.println("Hi Good Evening.....");
  }
}
```

-When we send response to client it is our responsibility that we must specify what kind of content we are sending to the client

-To specify the type of the conent we use setContentType() method which is available in response object.

syntax:

```
    response.setContentType("text/html")
```

-we can send different type of data like text/html, text/css, application/pdf,....

? develop a servlet which send html content to the client

```
import javax.servlet.http.*;
import java.io.*;
public class HtmlServlet extends HttpServlet{
 public void service(HttpServletRequest request,HttpServletResponse response) throws IOException{
 response.setContentType("text/html");
PrintWriter out = response.getWriter();
out.println("<html>");
out.println("<head><title>...html servlet...</title><head>");
out.println("<body>");
out.println("<marquee>");
out.println("<h1>Welcome to Servlets @ InetSolv </h1>");
out.println("</marquee>");
out.println("</body>");
out.println("</html>");
out.close();
}
}
```

? write a servlet to print following output

firstline

secondline

Note:

-In the above application servlet is sending html content to client which allways displays same output to the different clients which is called static Servle

-Using Servlets we can also develop static web based apps it is not recommended to develop static web based apps using servrlets because of following reas

1.It will take lot of time to develop the application

2.If we want to change any thing in the application every we have to perform several steps which leads to maintenance problems

-If we want to develop a static web based apps simply we use HTML docs

-It is allways recommended to develop a dynamic web bas pplication using servlets.

Eg:Following is the Example for dynamic web based applic

```
import javax.servlet.http.*;
import java.io.*;
import java.util.Date;
public class DateTimeServlet extends HttpServlet{
public void service(HttpServletRequest request,HttpServle ponse response) throws IOException{
response.setContentType("text/html");
PrintWriter out = response.getWriter();
Date d = new Date();
out.println(d);
}
}
```

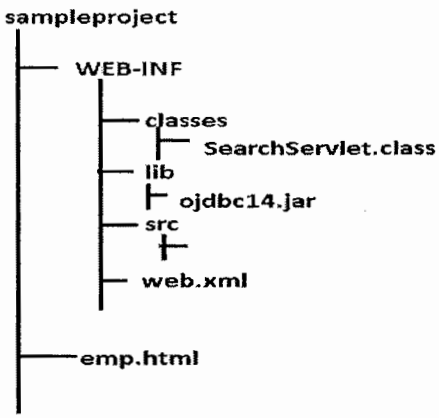
- Here servlet is called dynamic servlet which displays curr date and time for every request

web.xml

```
<web-app>
<servlet>
<servlet-name>ccc</servlet-name>
<servlet-class>DateTimeServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>ccc</servlet-name>
<url-pattern>/*</url-pattern>
</servlet-mapping>
</web-app>
```

in the configuration file if we write url-pattern as /* then n call and access particular servlet with any name.

write a Servlet program which recieve the data from db server and displays on the client system.



emp.html

```
<html>
<head>
<title>Employee Searching Form</title>
</head>
<body bgcolor="cyan">
<h1>You can Search Here</h1>
<center>
<form name="empform" action="search" method="get" target="loc">
EMPNO: <input type="text" name="empid"/><br/>
<input type="submit" value="search"/>
</form>
<iframe name="loc" width="75%"/>
</center>
</body>
</html>
```

SearchServlet.java

```
import  javax.servlet.http.*;
import java.io.*;
import java.sql.*;
public   class  SearchServlet  extends  HttpServlet{
public void service(HttpServletRequest request, HttpServletResponse response) throws IOException{
response.setContentType("text/html");
PrintWriter  out = response.getWriter();
//reading the form data which is requested by client
int  id = Integer.parseInt(request.getParameter("empid"));
//out.println("<h1>"+id+"</h1>");
try{
DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
Connection  conn =  DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","inetsolv","students");
Statement   st = conn.createStatement();
```

```
String  sql = "select * from emp where  empno="+id;
ResultSet  rs =  st.executeQuery(sql);
if(rs.next()){
int eno = rs.getInt(1);
String name = rs.getString(2);
String job= rs.getString(3);
double  sal =  rs.getDouble(4);
out.println("<table align='center' border='1' bordercolor='black'>");
out.println("<tr><td>EMPNO</td><td>"+eno+"</td></tr>");
out.println("<tr><td>ENAME</td><td>"+name+"</td></tr>");
out.println("<tr><td>JOB</td><td>"+job+"</td></tr>");
out.println("<tr><td>SALARY</td><td>"+sal+"</td></tr>");
out.println("</table>");
}
else{
out.println("<h1>Sorry Invalid Empno</h1>");
}
}
catch(SQLException  e){
}
}
}
```

To compile this program we need to set classpath to 2 .jar files like follows

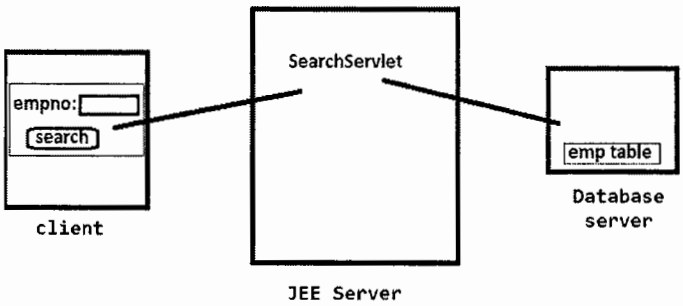
```
> set classpath= servlet-api.jar;ojdbc14.jar;.;
> javac SerachServlet.java
```

web.xml

```
<web-app>
<servlet>
<servlet-name>ddd</servlet-name>
<servlet-class>SearchServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>ddd</servlet-name>
<url-pattern>/search</url-pattern>
</servlet-mapping>
</web-app>
```

Note:

- Server will check for a class in following 3 locations
    1. inside the classes folder
    2. inside the project lib folder
    3. inside the server lib folder

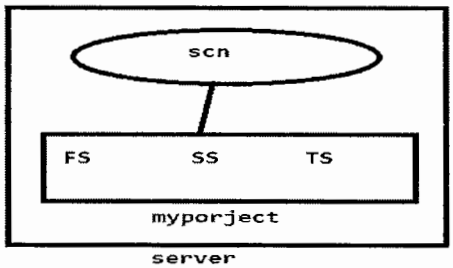


- if class is not available in all these locations then server will throw an exception saying ClassNotFoundException
- In the last program we have directly specified driver name, db url, user name and password which are hardcoded.
- For example in the future if we want to change the user name and password or change the entire database we have to perform many steps which will waste our time.
- we have to remove this hard coding from our application.
-but we never use commandline arguments or system properties to remoe the hardcoding inside the servelts.
-if we want to remove the hardcoding in servelt we can use any one of the following 2 objects

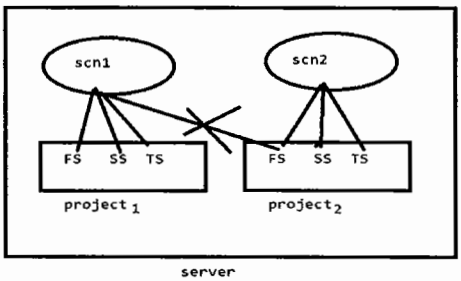
1. ServletContext object
2. ServletConfig object

1. ServletContext object

- ServletContext object can be created and removed only by the server not by the programmer
- ServletContext object is created at the time of deploying the project.
-ServletContext object will be removed by the server when we undeploy the project.
-One ServletContext object is created for one project
-This ServletContext object can be used by all the servlets which are belongs to the particular project



-But ServletContext object of one project can not be used by a servlet which belongs to other project



- We can create ServletConext object in following 2 ways

1. using getServletConext() method

Eg:

```
ServletContext application = getServletContext();
```

2. using ServletConfig object

Eg:

```
ServletConfig config = getServletConfig();
ServletContext application = config.getServletContext();
```

Note:

these getServletConext() , getServletConfig() methods are provided by GerincServlet class.

? how to store the data inside the ServletContext object

- if we want to store the data inside the ServletContext object we have to use <context-param> tag of web.xml file

Eg:

web.xml

```
<web-app>
<context-param>
<param-name>uname</param-name>
<param-value>suresh</param-value>
</context-param>
<context-param>
<param-name>pwd</param-name>
<param-value>students</param-value>
</context-param>
<servlet>
<servlet-name>aaa</servlet-name>
<servlet-class>ServletContextDemo1</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>aaa</servlet-name>
<url-pattern>/scd1</url-pattern>
</servlet-mapping>
<servlet>
<servlet-name>bbb</servlet-name>
<servlet-class>ServletContextDemo2</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>bbb</servlet-name>
<url-pattern>/scd2</url-pattern>
</servlet-mapping>
</web-app>
```

Note:

-In ServletContext object the data will be stored in the form of key-value pairs.

-If we want to get the data of ServletContext object we use getInitParameter()method of ServletContext object like follows

syntax:

String getInitParameter(key);

if specified key is available in web.xml it will return our value in the form String otherwise if key not available in web.xml it will just return a null value.

Servlet Program to read ServletContext object data

```
import javax.servlet.http.*;
import javax.servlet.*;
import java.io.*;
public class ServletContextDemo1 extends HttpServlet{
public void service(HttpServletRequest request,HttpServletResponse response) throws IOException{
res.setContentType("text/html");
PrintWriter out = response.getWriter();
ServletContext application = getServletContext();
String user = application.getInitParameter("uname");
String pwd = application.getInitParameter("pwd");
out.println("<h1>hi "+user+" good Morning???</h1>");
out.println("<h1>your password is "+pwd);
}
}
```

2. ServletConfig object

-ServletConfig object can be created and removed only by the server not by the programmer

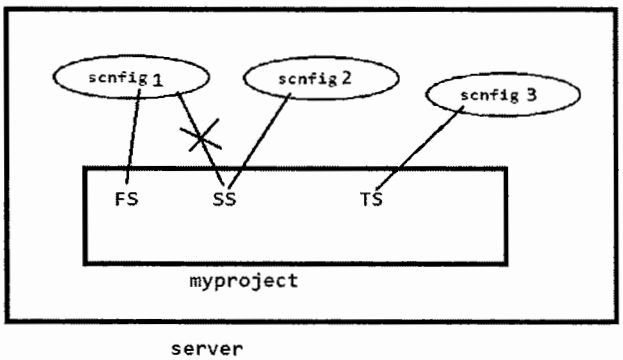
-ServletConfig object is created when client send the request to the particular servlet and when servlet object is created.

-ServletConfig object will be removed by the server when we close the server or when the servlet is deleted from the project

-One ServletConfig object is created for one Servlet it means in one project we can contain multiple ServletConfig objects based on the multiple servlets available.

-This ServletConfig object can be used only by the particular servlet which contains ServletConfig object.

-But ServletConfig object of one servlet can not be used by any other servlet which may be belongs to same or other project



-we can create ServletConfig object using getServletConfig() method like follows

```
ServletConfig   config = getServletConfig();
```

? how to store the data inside the ServletConfig object

- if we want to store the data inside the ServletConfig object we have to use <init-param> tag which is the sub tag of <servlet>tag of web.xml file

web.xml

```
<web-app>
<servlet>
<init-param>
<param-name>uname</param-name>
<param-value>pavan</param-value>
</init-param>
<init-param>
<param-name>pwd</param-name>
<param-value>fans</param-value>
</init-param>
<servlet-name>aaa</servlet-name>
<servlet-class>ServletConfigDemo</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>aaa</servlet-name>
<url-pattern>/configobj</url-pattern>
</servlet-mapping>
</web-app>
```

Note:

-In ServletConfig object also the data will be stored in the form of key-value pairs.

- if we want to get the data of ServletConfig object also we use same method getInitParameter() method of ServletConfig like follows

syntax:

```
String   getInitParameter(key);
```

Servlet Program to read ServletConfig object data

```
import  javax.servlet.http.*;
import  javax.servlet.*;
import java.io.*;
public   class ServletConfigDemo  extends  HttpServlet{
public void service(HttpServletRequest  request, HttpServletResponse response) throws IOException{
res.setContentType("text/html");
PrintWriter   out =  res.getWriter();
ServletConfig  config  = getServletConfig();
String user = config.getInitParameter("uname");
String pwd = config.getInitParameter("pwd");
out.println("<h1>hi "+user+" good Morning???</h1>");
out.println("<h1>your password is "+pwd);
}}
```

assignment

develop the previous SearchServlet program by removing the hardcoding of driver, url , username,password using either ServletContext object or ServletConfig object

-some times we dont know the key names then if we want get all the key names and its corresponding values then we have to use following method which is avaiable in ServletContext object or ServletConfig Object.

```
Enuemeration      getInitParameterNames()
```

//write a Servlet which display all the names an values of ServletContext object

```
import javax.servlet.http.*;
import javax.servlet.*;
import java.io.*;
import  java.util.*;
public class  ServletContextNamesServlet  extends  HttpServlet{
 public  void service(HttpServletRequest request,HttpServletResponse  response) throws IOException{
 response.setContentType("text/html");
PrintWriter  out =  response.getWriter();
ServletConfig  config= this.getServletConfig();
ServletContext application=this.getServletContext();
Enumeration  e = application.getInitParameterNames();
while(e.hasMoreElements()){
String key = (String)  e.nextElement();
String value  = application.getInitParameter(key);
out.println("<br/>name: "+key+", value :"+value);
}
}
}
```

Form Based Applications

-generally a website developed based on different forms which are also called as form based applications

Eg:

login form, register form, billpayment form,....

-to devevelop the form based applications we have to use html <form> tag

syntax:

```
<form name="name" action="url of servlet/jsp" method="get/post">
 --- form components  ---
</form>
```

- A servlet can be called in following 4 different ways

1. calling the servlet directly by writing the url inside the address bar of any browser
2. calling the servlet using hyperlinks

index.html

```
<html>
<head>
</head>
<body bgcolor="cyan">
```

```
<h1>Calling Servlet Using hyperlink</h1>
<center>
<a href="hs">clicke here </a>to open HtmlServlet
</center>
</body>
</html>
```

-For example let us use HtmlServlet to call using hyperlink

web.xml

```
<web-app>
  <servlet>
      <servlet-name>aaa</servlet-name>
      <servlet-class>HtmlServlet</servlet-class>
  </servlet>
  <servlet-mapping>
      <servlet-name>aaa</servlet-name>
       <url-pattern>/hs</url-pattern>
  </servlet-mapping>
</web-app>
```

Note

If we use our html or jsp page name as index.html then this page will be displayed as first page when we enter our project URL like " http://localhost:7777/sampleproject/ "

3.Calling the servlets using forms

index.html

```
<html>
<head>
</head>
<body bgcolor="cyan">
<h1>Calling Servlet Using hyperlink</h1>
<center>
<form name="myform" action="hs">
<input type="submit" value="call"/>
</form>
</center>
</body>
</html>
```

4. calling servlet by using our own buttons by using java script

sampleform.html

```
<html>
<head>
<script  type="text/javascript">
function  callServlet(){
 document.forms[0].action="hs";
 document.forms[0].submit();}
</script>
```

```
</head>
<body bgcolor="cyan">
<h1>Calling Servlet Using our own buttons</h1
<center>
<form>
<input type="button" value="click here" onclick="callServlet()"/>
<br/>
to open HtmlServlet
</form>
</center>
</body>
</html>
```

Note:

-To display welcome page our required file must be created as index.html or index.jsp,...

-To display welcome page we can also use<welcome-file-list> tag inside the web.xml like follows

web.xml

```
<web-app>
  <servlet>
      <servlet-name>aaa</servlet-name>
      <servlet-class>HtmlServlet</servlet-class>
  </servlet>
  <servlet-mapping>
      <servlet-name>aaa</servlet-name>
       <url-pattern>/hs</url-pattern>
  </servlet-mapping>
 <welcome-file-list>
    <welcome-file>SampleForm.html</welcome-file>
    <welcome-file>hs</welcome-file>
 </welcome-file-list>
</web-app>
```

? develop a servlet that check user name and password and forward the result to either success page or fail page.

loginform.html

```
<html>
<head>
</head>
<body bgcolor="cyan">
<h1>Login Here,</h1>
<center>
<form name="loginform" action="ts" method="get">
  User Id: <input type="text" name="userid"/><br/>
 Password:<input type="password" name="pwd"/><br/>
  <input type="submit" value="login"/>
  <input type="reset" value="cancel"/>
```

```
</form>
</center>
</body>
</html>
```

TestServlet.java

```
import  javax.servlet.http.*;
import  javax.servlet.*;
import java.io.*;
public   class TestServlet  extends  HttpServlet{
public void doGet(HttpServletRequest request, HttpServletResponse response) throws IOException{
response.setContentType("text/html");
PrintWriter   out =  res.getWriter();
String user =  request.getParameter("userid");
String pwd =  request.getParameter("pwd");
if(user.equals("InetSolv")&&pwd.equals("students")){
response.sendRedirect("success.html");
}
else{
response.sendRedirect("fail.html");
}
}
}
```

web.xml

```
<web-app>
  <servlet>
      <servlet-name>aaa</servlet-name>
      <servlet-class>TestServlet</servlet-class>
  </servlet>
  <servlet-mapping>
      <servlet-name>aaa</servlet-name>
       <url-pattern>/ts</url-pattern>
  </servlet-mapping>
 <welcome-file-list>
   <welcome-file>loginform.html</welcome-file>
 </welcome-file-list>
</web-app>
```

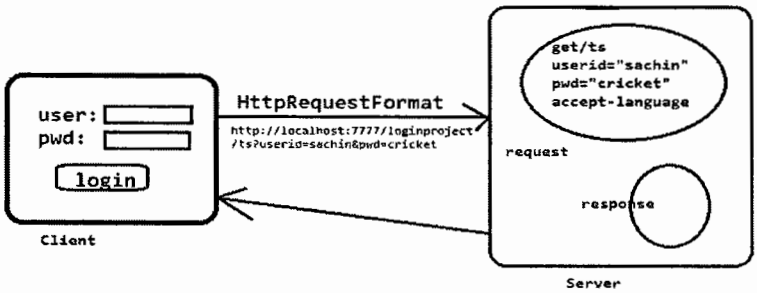
success.html

```
<html>
<head>
</head>
<body bgcolor="green" text="white">
<center>
<h1>Welcome to InetSolv </h1>
</center>
```

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

fail.html

```
<html>
<head>
</head>
<body bgcolor="red" text="white">
<center>
<h1> Invalid UserName</h1>
</center>
</body>
</html>
```



Note:

-If we want to read the requested data of the form inside the servlet program we have to use getParameter() method of request object.

syntax:

```
String    request.getParamater("fieldname");
```

-here fieldname is nothing but the name of the form component like any textbox or password box or radio buttons,.... in html

- if specified fieldname is not available it returns null value.

-Sometimes we need to forward our servlet to any particular html or other servlet or jsp then we have to use sendRedirect() method of response object.

syntax:

```
response.sendRedirect("URL");
```

-Inside the Servlet programming instead of overriding service() method we can also ovveride other server methods like doGet() or doPost() methods which are given by HttpServlet.

?what is the difference between doGet(),doPost() and service()

usage of all these 3 methods is same but doGet() method can access only get method of the form ,doPost() method can access only post method of the form and service() method can access both get and post method of the form.

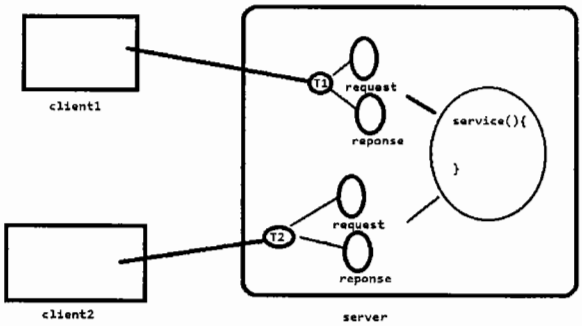
Thread Pooling

-some times we multiple clients will send the request to same servlet at the same time, in this case we use the concpet of Thread Pooling

-Thread Pooling is a built in program available in all the JEE servers

-According to this Thread Pooling when ever client send the request to particular Servlet then server will pick one Thread from Thread Pooling and creates servlet request and response objects and handover to paticular service() method of Servlet like this for every client for every request there will be one separate thread created.

-Following is diagram which explain thread pooling



-when server finish the job it will delete he servlet response object and request object and thread will be return to Thread pooling program again.

Note:

- Server to Server different alogorithems maintained to conduct Thread Pooling.

<u>? can we write constructor inside the Servlet</u>

- yes we can write a constructor inside the Servlet but which must be a public 0 parameterized constructor

-Internally Server uses Class.forName() and newInstance() methods to create the object for any Servlet.

- For example if we write a default or private or protected constructor inside the Servlet then no Servlet object is created by Server and Server display an Exception saying " java.lang.IllegalAccessException "

- otherwise if we write a parameterzed constructor without any 0 parameterized constructor inside the Servlet then no Servlet object is created by Server and Server and display an Exception saying " java.lang.InstantiationException "

Eg:

```
//sample TomcatServer program internal code how it will create a Servlet object
public class TomcatServer{
public static void main(String args[]){
Class cl = Class.forName("HtmlServlet");
Object o = cl.newInstance();
Servlet s = (Servlet) o;
 s.init();
 s.service();
}
}
```

Note:

-because of above problems sun micro system people recommedning us not to write the constructor inside Servlet.

-sun micro system people have provided init() method instead of constructors.

-If our Servlet contains both 0 parameterized constructor and 1 parameterized constructor and both are public but Server allways call 0 parameterized constructor and create Servlet object but if we want to invoke other constructors aswell then we have to use " this(...) ".

Eg:

```
import  javax.servlet.http.*;
import java.io.*;
public   class SampleServlet  extends  HttpServlet{
public  SampleServlet(){
this(10);
System.out.println("we are in 0 parameterized constructor");
}
public SampleServlet(int x){
System.out.println("we are in 1 parameterized constructor");
}
public void service(HttpServletRequest request, HttpServletResponse response) throws IOException{
response.setContentType("text/html");
PrintWriter  out = response.getWriter();
out.println("welcome to servlet");
 }
}
```

Note:

-If we want to write the constructors inside the Servlet they must be public

? when a servlet object is created

case 1:

- when client send the request to the server for the first time then a servlet object will be created.

case 2:

-by using <load-on-startup> tag of web.xml also we can create an object for Servlet, In this case the servlet object is created at the time project is deployed.

-<load-on-startup> will be specified inside the web.xml file in side the particular servlet tag.

Eg:

```
<web-app>
  <servlet>
      <servlet-name>aaa</servlet-name>
      <servlet-class>HtmlServlet</servlet-class>
       <load-on-startup>2</load-on-startup>
  </servlet>
  <servlet-mapping>
      <servlet-name>aaa</servlet-name>
       <url-pattern>/hs</url-pattern>
  </servlet-mapping>
</web-app>
```

-here <load-on-startup>will take take only +ve value which will decide the order of creating Servlet objects at the time of deployment.

-the least startup value containing Servlet First get loaded and object is created.

-If 2 servlets are having same priority then it will use names of the Servlets in alphabetical order and create the servlet objects but in the case of welogic server it will create the objects based on the order of thier configuration.

Note:

if we specify <load-on-startup> value negative then it wont consider and no Servlet object is created during the deployment

? can we run servlets on other Server

yes we can run servlets on any JEE Servers

-For example if we want to run our servlet on weblogic server we can use any one one of the following procedures

procedure1:

-start the weblogic server

-First open weblogic admin console by writing following URL

http://localhost:7001/console

-Enter the username and password of particular domain

username: students

password: javajava

-deploy our project folder(Eg:sampleproject) into autodeploy folder of weblogic domain

-Enter the URL like http://localhost:7001/sampleproject/hs to execute our application

procedure2:

- we can also deploy our porject using weblogic admin console like follows
- start the weblogic server
- First open weblogic admin console by writing URL http://localhost:7001/console
- Enter the username and password of particular domain

username: students

password: javajava

- select the deployments option from domain structure
- click on install button
- select the project folder and clik on next
- select"Install this deployment as an application"& click on next
- specify the deployment name and click on next and finish

-now Enter the URL like http://localhost:7001/sampleproject/hs to execute our application

-When we want to develop a servlet which uses JDBC code then if write JDBC 5 steps directly inside the service() method for every client request it will execute the service() there by JDBC5 steps are also executed every time which will not improve the performance.

- to improve the performace we have to write db driver registration and getting connection as a part of init() which are executed only 1 time entire life cycle of the Servlet.

-Statement object creation and executing query steps as a part of service() method

-closing connection should be specified as a part of destroy() method

```
import javax.servlet.http.*;
import javax.servlet.*;
import java.io.*;
import java.sql.*;
public class SearchServlet extends HttpServlet{
ServletConfig config;
Connection conn;
public void init(ServletConfig config){
```

```
this.config=config;
try{
DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
conn =  DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","inetsolv","students");
}
catch(SQLException e){
}
}
public void service(HttpServletRequest request, HttpServletResponse response) throws IOException{
response.setContentType("text/html");
PrintWriter  out = response.getWriter();
int   id = Integer.parseInt(request.getParameter("empid"));
try{
Statement   st = conn.createStatement();
String  sql = "select * from emp where  empno="+id;
System.out.println(sql);
ResultSet  rs =   st.executeQuery(sql);
if(rs.next()){
int eno = rs.getInt(1);
String name = rs.getString(2);
String job= rs.getString(3);
double  sal =  rs.getDouble(4);
out.println("<table align='center' border='1' bordercolor='black'>");
out.println("<tr><td>EMPNO</td><td>"+eno+"</td></tr>");
out.println("<tr><td>ENAME</td><td>"+name+"</td></tr>");
out.println("<tr><td>JOB</td><td>"+job+"</td></tr>");
out.println("<tr><td>SALARY</td><td>"+sal+"</td></tr>");
out.println("</table>");
}
else{
out.println("<h1>Sorry Invalid Empno</h1>");
}
}
catch(SQLException  e){
}
}
public  void destroy(){
try{
conn.close();
}
catch(SQLException  e){
}
}
}
```

? whether servlets are thread-safe or not

-By default servlets are not thread-safe it means at a time multiple clients can access our Servlet which may leads to data inconsitancy problem.

-but if we want to make servlets thread-safe we have to use synchronized keyword.

Eg:

```
import javax.servlet.http.*;
import javax.servlet.*;
import java.io.*;
public class ThreadSafeTestingServlet extends HttpServlet{
int nseats;
public void init(ServletConfig config){
nseats=1;
}
public synchronized void service(HttpServletRequest req,HttpServletResponse res) throws IOException{
res.setContentType("text/html");
PrintWriter out = res.getWriter();
if(nseats>0){
out.println("<h1>Seat is Alloted to You</h1> ");
try{
Thread.sleep(1000);
}
catch(InterruptedException ie){
}
nseats=nseats-1;
}
else{
out.println("<h1>Seat is Not Alloted to You</h1> ");
}
out.close();
}
}
```

Note:

- It is allways recommended that not to write the isntance varaibles in side the servlets because those can be shared by multiple threads (created for multiple request given by multiple clients) that are working on same servlet.
- it allways recommended to use local variables inside the service() method

MyConneciton.java(singletone class)

```
package com.myproject.connections;
import java.sql.*;
public class MyConnection{
private MyConnection(){
}
static Connection conn=null;
public static Connection getMyConnection() throws SQLException{
```

```
if(conn==null){
DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());
conn =  DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","inetsolv","students");
}
return  conn;
}
public static  void   closeMyConnection() throws SQLException{
conn=null;
}
}
```

- Servlet using MyConnection class which display employee info

SearchServlet.java

```
import  javax.servlet.http.*;
import  javax.servlet.*;
import java.io.*;
import java.sql.*;
import  com.myproject.connections.*;
public   class  SearchServlet  extends  HttpServlet{

public void service(HttpServletRequest request,HttpServletResponse response)throws IOException{
response.setContentType("text/html");
PrintWriter  out = response.getWriter();
int    id = Integer.parseInt(request.getParameter("empid"));
try{
Connection  conn = MyConnection.getMyConnection();
Statement   st = conn.createStatement();
String  sql = "select * from emp where  empno="+id;
System.out.println(sql);
ResultSet  rs =   st.executeQuery(sql);
if(rs.next()){
int eno = rs.getInt(1);
String name = rs.getString(2);
String job= rs.getString(3);
double  sal =  rs.getDouble(4);
out.println("<table align='center' border='1' bordercolor='black'>");
out.println("<tr><td>EMPNO</td><td>"+eno+"</td></tr>");
out.println("<tr><td>ENAME</td><td>"+name+"</td></tr>");
out.println("<tr><td>JOB</td><td>"+job+"</td></tr>");
out.println("<tr><td>SALARY</td><td>"+sal+"</td></tr>");
out.println("</table>");
}
else{
out.println("<h1>Sorry Invalid Empno</h1>");
}
```

```
}
catch(SQLException  e){
}
}
public  void destroy(){
MyConnection.closeMyConnection();
}
}
```

? can we call destroy() method

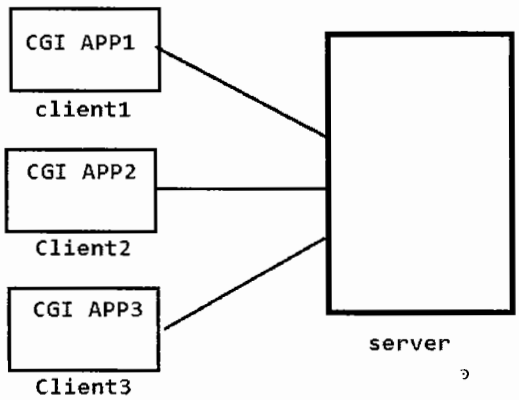
yes we can call destroy() method from service() but no Servlet object will be removed just destroy() method executed like a normal method

Eg:

```
import  javax.servlet.http.*;
import  javax.servlet.*;
import java.io.*;
public  class TestingServlet  extends  HttpServlet{
public void service(HttpServletRequest req,HttpServletResponse res) throws IOException{
destroy();
}
public void destroy(){
System.out.println("we are in destroy() method");
}
}
```

CGI

- CGI stands for common geteway interface
- CGI is another technology that is used to develop the web based apps.
- CGI applications are having so many drawbacks like follows
    1. CGI applications are platform form dependent
    2. CGI applications runs in the computer which are associated with server which leads to lot of performance issues
    3. for every new request it will create a new object.



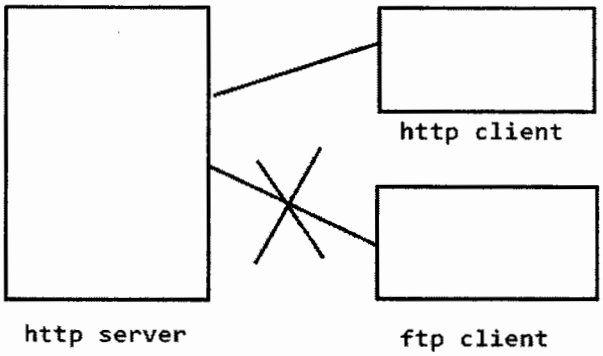
Protocols

a protocal is a set of standared rules which must be followed by 2 systems in order to make the communication possible

Eg:

Http protocal
Ftp protocal
Tcp/ip protocal
smtp protocal
udp protocal
.....

javax.servlet.http.*;
javax.servlet.ftp.*;
javax.servlet.tcpip.*;
javax.servlet.smtp.*;
javax.servlet.udp.*;
.....



JSP

- JSP statnds for Java Server Pages
- JSP is an API introduced by sun micro systems which is also used to develop web based apps
- JSP files can be located inside the projectfolder
- The extension name of JSP file should be .jsp
- We no need to configure JSP file inside the web.xml
- we can directly use jsp file name as url in the addressbar.
- we have many advantages compared to servlets
- servlet takes manysteps to develop where jsp takes less steps and save the development time
- JSP will divide our project into presentation logic and business logic
- JSP will allways contain presentation logic where servlets will contain business logic

servlet container

servlet container is a program which provide implementaion for Servlet API which is responsible for creating servlet servlet object and runnding the Servlet.

jsp container

jsp container is a program which provide implementaion for JSP API which is responsible for running the Servlet.

-If we want to develop any web based application we have to use servet container, jsp container , http server,...

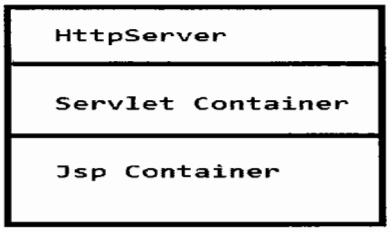
- Now a days all these things integrated in any JEE servers
- JEE servers are avaialble following 2 types

1. web servers

- If Any Server contains only HttpServer, jsp container ,servlet cotnainer we can call it as web servers
- web servers are used to only develop web based applications but we can not develop any EJB applications

Eg:

tomcat, resin, jws,.....

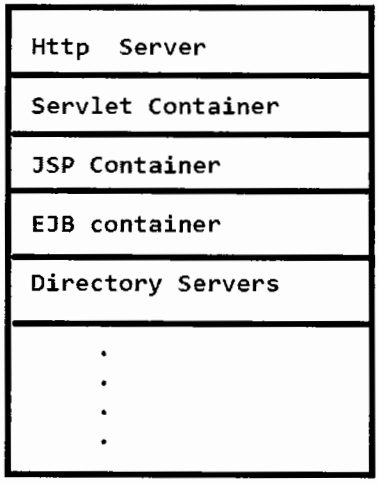


2. application server

- If Any Server contains HttpServer, jsp container ,servlet cotnainer EJB applications, directory servers,... we can call it as application servers
- simply we can say application servers completly implementing JEE API where web servers are partially implementing JEE API

Eg:

weblogic, websphere, JBOSS,.....



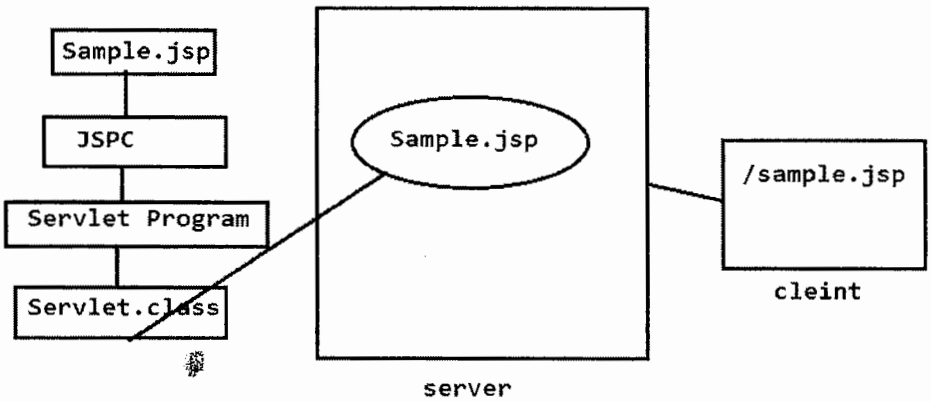
//wap to develop a first JSP file

Sample.jsp

```
<html>
<head>
</head>
<body bgcolor="yellow">
Welcome  to My JSP
<%
 System.out.println("Welcome  to My JSP");
%>
</body>
</html>
```

-When ever client send the request to the jsp file first JSP compiler convert jsp file into corresponding jsp program(http://localhost:7777/myproject/sample.jsp)

-JSP compiler will compile the servlet program and generate the .class file and creates the Servlet object for jsp file locate inside the heap memory of JVM of Server.



-all the preceeding steps will be done by server when client send the request to the server for the first time but second time onwards for every request server will give jsp object

-if we want see the genereated servlet program then we hae to go following location

C:\Program Files
-> Apache Software Foundation
-> Tomcat 6.0
->work
->Catalina
->localhost
->myjspproject
->org
-> apache
->jsp

Here we contain servlet program and its compiled .class file for correponding jsp program

- JSP compiler must generate a servlet program for jsp file and must generate a .class file
- Every JSP compiler follows its own naming pattern
  --In the case of tomcat server the servlet program name is Sample_jsp.java
  --But in the case of weblogic server the servlet program name is " _Sample.java "
- Every JSP compiler creates a package to locate the generated .class file
  Eg:
  in tomcat server .class file will be located in org.apache.jsp package

-Every generated servlet program import following 3 packages

```
import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.jsp.*;
```

-Every generated servlet program contains following 3 life cycle methods

```
_jspInit()
_jspService()
_jspDestroy()
```

-Inside the _jspService() it will create 9 implicit objetcs

-Following is the gernerated Servlet program package org.apache.jsp

```
import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.jsp.*;
public final class sample_jsp extends org.apache.jasper.runtime.HttpJspBase{
 public void _jspInit() {
  }
  public void _jspDestroy() {
  }
 public void _jspService(final HttpServletRequest request, final HttpServletResponse response)
      throws java.io.IOException, javax.servlet.ServletException {
    }
}
```

JSP components

- To Develop a Jsp file the developers have given following different jsp components
  1.template text
  2. jsp scriptlets

3. jsp expressions
4. jsp declarations
5. jsp directives
6. jsp action tags
7. expresion language(EL)
8. Jsp custum tags
9.JSTL

1. template text

template text means if we write any text direclty inside the jsp file then it is called template text.this component mainly used to send the info to client .

Eg:

Sample.jsp

```
 Hi    // template text  line 1
 Welcome to JSP Page  //  template text line2
```

- template text will be written direclty inside the _jspService() method like follows

Eg:

```
public final class Sample_jsp extends HttpJspBase{
 public void _jspService(..............){
   out.write("Hi");
   out.write("Welcome to JSP Page");
 }
}
```

2. jsp scriptlets

- This jsp component is used to write the java code inside the jsp file

syntax:

```
<%  (starting scriptlet)
 //java code
%> (ending of scriptlet)
```

- we can write any number of jsp scriptlets and any where inside the jsp file

Eg:

```
<%
  int  a=10;
  int b=20;
  int c=a+b;
System.out.println("Additon="+c);
%>
```

- the code what we write inside the jsp scriptlets will be wrriten directly inside the _jspService() method like follows

Eg:

```
public final class sample_jsp extends HttpJspBase{
 public void _jspService(..............){
  int  a=10;
```

```
  int b=20;
  int c=a+b;
System.out.println("Additon="+c);
 }
}
```

-when we write the code inside the jsp scriptlets we must follow all the java rules otherwise jsp compiler fail to compile the program and display the error messages on the browser.

- For Every Jsp File when JSP compiler genrate the servlet program it will create following 9 implicit objects inside the _jspService() method

1. request
2. response
3. pageContext
4. session
5.application
6.config
7.out
8.page
9.Exception

hence all these objects are created and available inside the _jspService() method we can use all these implicit objects direclty inside the jsp scriptlets

1. request

This implicit object is used to process the request of send by the client.

request -> HttpRequest

2. response

-This implicit object is usedto process send the response toclient

-we can send content type

-we can send error report (we write as a part of error pages)

-we can send text

response -> HttpResponse

Eg:

```
<%
  response.setContentType("text/html");
   out.print("hi");
%>
<%
response.sendError(-504,"sorry");
%>
```

3.out

out is the implicit variable created for JspWriter which is used to send the text to client and display on the client system.

out -> JspWriter

login.html

```
<html>
<head>
</head>
<body bgcolor="cyan">
<h1>Login Here,</h1>
<center>
<form name="loginform"  action="readdata.jsp" method="get">
  User Id: <input type="text"  name="userid"/><br/>
 Password:<input type="password" name="pwd"/><br/>
  <input type="submit" value="login"/>
  <input type="reset" value="cancel"/>
</form>
</center>
</body>
</html>
```

readata.jsp

```
<html>
<head>
</head>
<body bgcolor="yellow">
<h1>Your Details are ,</h1>
<%
response.setContentType("text/html");
String  user = request.getParameter("userid");
String  pass = request.getParameter("pwd");
out.print(" User Id: "+user);
out.print("<br/> Password:"+pass);
%>
</center>
</body>
</html>
```

-For jsp files we no need to configure inside the web.xml file but if we want to configure jsp file inside the web.xml is possible.

- genrally we locate .jsp file inside the project file along with WEB-INF folder but for security resons it is recommended to provide inside the WEB-INF folder

configuring jsp file inside the web.xml

-assume we have stored our .jsp file in following folder

```
|-myproject
|    |---  WEB-INF
|        |-- myjspfiles
|            |-- Sample.jsp
|
```

web.xml

```
<web-app>
<servlet>
<servlet-name>aaa</servlet-name>
<jsp-file>/WEB-INF/myjspfiles/Sample.jsp</jsp-file>
</servlet>
<servlet-mapping>
<servlet-name>aaa</servlet-name>
<url-pattern>/sam</url-pattern>
</servlet-mapping>
</web-app>
```

Note:

1. while specifying the .jsp file name we have to specify the absolute path.
2. Here out create for JspWriter class but in servlets we use PrintWriter class and the main difference between JspWriter and print Writer is JspWriter associated with 8kb of buffer size where PrintWriter doesnot associated with any buffer

4. conifg:

This implicit object is created for ServletConfig class and used to access the data of ServletConfig object same like in servlets.

conifg -> ServletConfig

Eg:

web.xml

```
<web-app>
<servlet>
<servlet-name>aaa</servlet-name>
<jsp-file>/WEB-INF/myjspfiles/Sample.jsp</jsp-file>
<init-param>
<param-name>uname</param-name>
<param-value>suresh</param-value>
</init-param>
<init-param>
<param-name>pwd</param-name>
<param-value>123456</param-value>
</init-param>
</servlet>
<servlet-mapping>
<servlet-name>aaa</servlet-name>
<url-pattern>/sam</url-pattern>
</servlet-mapping>
</web-app>
```

sample.jsp

```
<html>
 <head>
</head>
<body>
<%
String uname=config.getInitParameter("uname");
String pwd=config.getInitParameter("pwd");
out.println("<h1>USER:"+uname+"</h1>");
out.println("<h1>PWD:"+pwd+"</h1>");
%>
</body>
</html>
```

5. application

application is implicit object which is created for ServletContext class and used to access the data of ServletContext object same like in servlets.

application -> ServletContext

Eg:

web.xml

```
<web-app>
<context-param>
  <param-name>cname</param-name>
  <param-value>InetSolv</param-value>
 </context-param>
 <context-param>
  <param-name>pwd</param-name>
  <param-value>students</param-value>
 </context-param>
</web-app>
```

Sample.jsp

```
<html>
<head>
</head>
<body BGCOLOR="lightpink">
HI Im JSP Page
<%
String cname = application.getInitParameter("cname");
String pwd = application.getInitParameter("pwd");
out.print("<br/>Username:"+cname);
out.print("<br/>Password:"+pwd);
%>
</body>
</html>
```

6. session

session is implicit object which is created for HttpSession class using which we can store and access data

session -> HttpSession

7. page

page is implicit object which is created for Object class using which we can indicate the object of servlet of corrsponding jsp program.

page-> Object

8.Exception

- this implict variable used in error pages

Exception -> Throwable

9. pageContext

pageContext is the implict variable created for PageContext class which is used for creating all the implicit variables like session,ServletConfig,out ,..... (if we require)

```
<%
ServletConfig  config1 = pageContext.getServletConfig();
HttpSession  session1= pageContext.getSession();
.......
%>
```

comments in jsp

- in jsp we can also write comments to explian about jsp page
- we can use html comments in jsp like follows

Eg:

```
<!--
  <%
    //statements;
  %>
-->
```

- we can use jsp comments in jsp like follows

```
<%--
  //statements;
--%>
```

jsp declarations

- jsp declarations are mainly used to define the istance members or static members in side the servlet program genrated for our jsp file

Syntax:

```
<%!
  //instance variables
  //static variables
  //instance methods
 // static methods
%>
```

Eg:

```
<%!
static int rno;
```

```
static String name;
void  show(int a,int b){
System.out.println("Addition="+(a+b));
}
%>
```

servlet code generated for jsp

Tthe code written inside the jsp declaratios willbe direclty provided inside the servlet class of JSP along with life cycle methods

```
public final class sample_jsp extends HttpJspBase{
 static int rno;
 static String name;
 void  show(int a,int b){
   System.out.println("Addition="+(a+b));
  }
 public void _jspService(..............){
   }
}
```

Note

1.if we want to access the methods declared in jsp declarations then we need to call or invoke them from _jspService() method for this we write method invocation inside the jsp scriptlets

2.implicit objects of Jsp can not be accessed in jsp declarations because implicit objects are local to _jspService() method and local variables of one method we can not access inside other locations

Eg1:

```
<%!
void show(int a,int b,JspWriter myout) throws java.io.IOException{
 myout.println("Addition="+(a+b));
  }
>
<%
  show(10,20,out);
%>
```

Eg2:

```
<%!
JspWriter myout;
void  show(int a,int b) throws java.io.IOException{
 myout.println("Addition="+(a+b));
}
%>
<%
 myout=out;
 show(10,20);
%>
```

- if we want to override life cycle method of JSP inside the jsp then we can override inside the JSP declarations only.

- following are the life cylce methods of JSP.
  1. jspInit()
  2. jspService()
  3. jspDestroy()

Eg:

```
<html>
 <head>
</head>
<body>
Hi Im JSP Page
<%!
public void jspInit(){
System.out.println("we are in init() method");
}
public void jspDestroy(){
System.out.println("we are in destroy() method");
}
%>
<%
System.out.println("we are in service() method");
%>
hi
</body>
</html>
```

Here

when we send a request to the JSP then servlet object will be created for correpsonding JSP and init() and service method executed and service method executed for every request in the mean while if jsp page is updated and page get refreshed immediately old jsp object deleted and destroy method is called next new jsp object is created and init() and service() method called

jsp expressions

- jsp expressions are used to print any value or expression result direclty on jsp page.

syntax:

```
<%=expression%>
```

Eg:

```
<%
int a=10;
int b=20;
%>
<h1>Your Result is Here</h1>
<h1>Addition  <%=a+b%></h1>
<h1><%=new java.util.Date()%></h1>
```

following is the example how jsp compiler converts the jsp expressions into servlet code

Eg:

```
<%=a+b%>
    ↓
  jsp compiler
    ↓
 out.print(a+b);
```

-using jsp expressions displaying submitted values

Eg:

```
<h1>USERNAME:<%=request.getParameter("userid")%></h1>
<h1>PASSWORD:<%=request.getParameter("pwd")%></h1>
```

jsp directives

- jsp directives are athe jsp components which are used to give the instruction to JSP compiler
- jsp directives will simplify writing jsp file
- We have following 3 types of jsp directives
  1. page directives
  2. include directives
  3. taglib directives

page directives

- page directives are used to give the instruction to jsp compiler
  like what package to import , what language we are writing,....
  syntax:
  <%@ page attribute1="value1" attribute2="value2".....%>

- Following are the different page directive attributes we can use for page directive inside the jsp
  1.language
  2.import
  3.info
  4.buffer
  5.autoflush
  6.content type
  7. isThreadSafe
  8. iserrorpage
  9. errorpage
  10.session
  11. extends

1.language

- language is page directive attribute which give an instruction to jsp compiler to use what kind of language inside the scriptlets

-we can specify different languages like java script, css ,...

-bydefault it takes java language inside the scriptlets

Eg:

```
<%@page language="javascript"%>
<%
 var a=10;
 document.write(a);
%>
```

Note:

- it is not recommend to language attribute
- tomcat server or weblogic server not supporting this language attribute but resin server supporting this language attribute.

2.import

- import is page directive attribute which give an instruction to jsp compiler like what packages tobe imported into jsp program.

Eg:

sample.jsp

```
<%@ page import="java.util.*" %>
<%
ArrayList al = new ArrayList();
 al.add(10);
 al.add(20);
 al.add(30);
%>
<%=al%>
```

-we can also import any number of packages using this import attrbute like folllows

Eg:

```
<%@ page import="java.util.*,java.io.*" %>
```

Note:

But it is not recommended to import multiple packages using single import attribute it is better to write multiple page directives with import attribute to import multiple packages

3.info

- info is page directive attribute which give an instruction to jsp compiler to override getServletInfo() method to return the info about jsp file

Eg:

```
<%@ page info="my servlet page"%>
<%
out.println(getServletInfo());
%>
```

servlet code

```
public String getServletInfo(){
  return "my servlet page";
 }
```

4.content type

- contentType is page directive attribute which give an instruction to jsp compiler like what type of content we are seding to the client.
- we can specify the diferent content type like text/xml, text/css...

Eg:

```
<%@ page contentType="text/xml"%>
<student>
<rno></rno>
<name></name>
</student>
```

5.buffer

-buffer is page directive attribute which give an instruction to jsp compiler what is the buffer size can be taken by out implicit variable of JspWriter. (bydefault 8kb)

Eg:

```
<%@ page buffer="5kb"%>
```

6.autoflush

- bydefault autoflush is true but if we want to control flushing on our own requirement we have to use this autoflush page directive.

Eg:

```
<%@ page autoFlush="false"%>
<%
 out.flush();
%>
```

7. isErrorPage

-isErrorPage is a page directive attribute which is used to create the error page

Eg:

error.jsp

```
<%@ page isErrorPage="true"%>
<html>
 <head>
</head>
<body>
<%
out.println("sorry division by zero is not possible");
%>
</body>
</html>
```

8. errorPage

-If our jsp file containging any exception and if we want to display any error page then we can use this errorPage page directive attriute

Eg:

calc.html

```
<html>
<head>
</head>
<body bgcolor="cyan">
<h1>Enter Here,</h1>
<center>
<form name="calcform"  action="test.jsp" method="get">
  Enter I Value: <input type="text"  name="v1"/><br/>
  Entetr II Value:<input type="text" name="v2"/><br/>
  <input type="submit" value="add"/>
</form>
</center>
</body>
</html>
```

test.jsp

```
<%@ page errorPage="error.jsp"%>
<html>
 <head>
</head>
<body>
<%
int n1 = Integer.parseInt(request.getParameter("v1"));
int n2 = Integer.parseInt(request.getParameter("v2"));

out.print("Division"+n1/n2);
%>
```

9.session

-session is a page directive attribute using which we can make the session enable or disable for particular jsp page

-bydefault session is true

-but if want make it false or true we can use this session attribute

-once we set session false in the particular page we can control the seession and its data.

Eg:

```
<%@ page session="false"%>
```

10. extends

- extends is a page directive attribute using which we can create our jsp servlet program by extending from any other class.

Eg:

test.jsp

```
<%@ page extends="sample.work.HtmlServlet"%>
```

folowing servlet code will be generated when we execute the above jsp program

```
public  class  test_jsp  extends  sample.work.HtmlServlet{
 }
```

11. isThreadSafe

- bydefault every JSP page isThreadSafe is true.
- but if we want make whether it is thread safe or not we have to use isThreadSafe page directive attribute like follows

```
<%@ page isThreadSafe="false"%>
```

Eg:

counter.jsp

```
<%@ page isThreadSafe="false"%>
<%!
 int count=1;
%>
<h1> Hi You Are my <%=count%> customer<h1>
<h1> who visit this Website</h1>
<%
try{
Thread.sleep(5000);
}
catch(InterruptedException ie){
}
count++;
%>
```

12. pageEncoding

-pageEncoding is a page directive attribute using which we can specify what charset we are using in jsp

Eg:

```
<%@ page pageEncoding="UTF-8"%>
```

Multiple form based applications

-In Our Applciation sometimes we need to carry or transfer the data between multiple forms

-But in form based applications we are using http protocal which is stateless protocal which means it remember only the current conversation.

-For example if we leave the first form and enter into second form it will forget about the previous form data.

-But if we want to overcome this limitation and carry the data between multiple forms we can use any one of the following methodologies

1.hidden variables
2.cookies
3.sessions
4.session with url re-writing or encoding URL

hidden variables

-If we want to use variables we have to use hidden fileds given by HTML

-hidden fields must be specified as a part of <form> and we must specify name and value attributes.

-hidden fields not displayed to the client

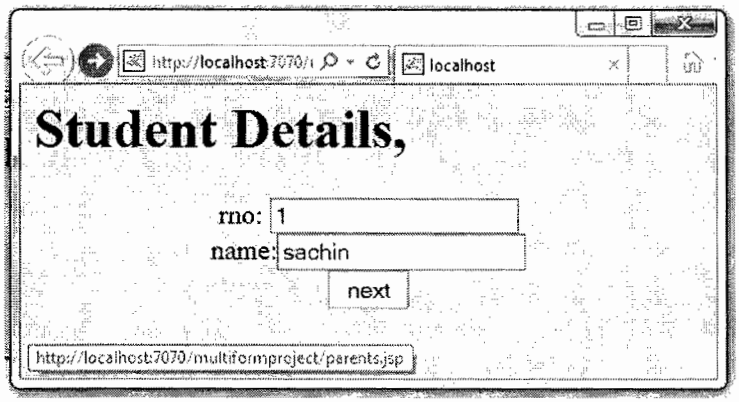
syntax:

```
<input type="hidden" name="xxx" value="yyy"/>
```

Eg:

student.html

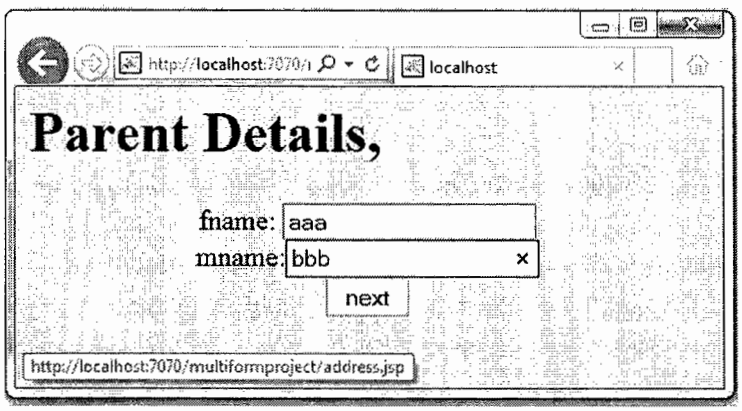
```
<html>
<head>
</head>
<body bgcolor="cyan">
<h1>Student Details Here,</h1>
<center>
<form name="studform"  action="parents.jsp" method="get">
   rno: <input type="text"  name="rno"/><br/>
  name:<input type="text" name="name"/><br/>
   <input type="submit" value="next"/>
</form>
</center>
</body>
</html>
```



parents.jsp

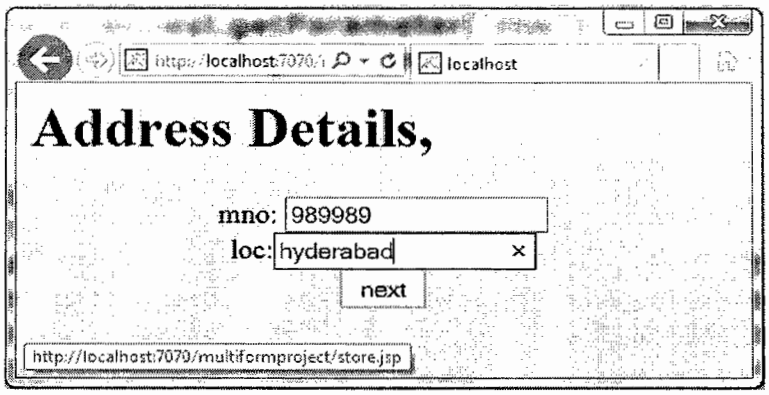
```
<html>
<head>
</head>
<body bgcolor="cyan">
<%
String rno = request.getParameter("rno");
String name = request.getParameter("name");
%>
<center>
<form action="address.jsp" method="get">
<input type="hidden"  name="rno" value="<%=rno%>"/>
<input type="hidden" name="name" value="<%=name%>"/>
<br/>
  fname: <input type="text"  name="fname"/><br/>
  mname:<input type="text" name="mname"/><br/>
```

```
  <input type="submit" value="next"/>
</form>
</center>
</body>
</html>
```



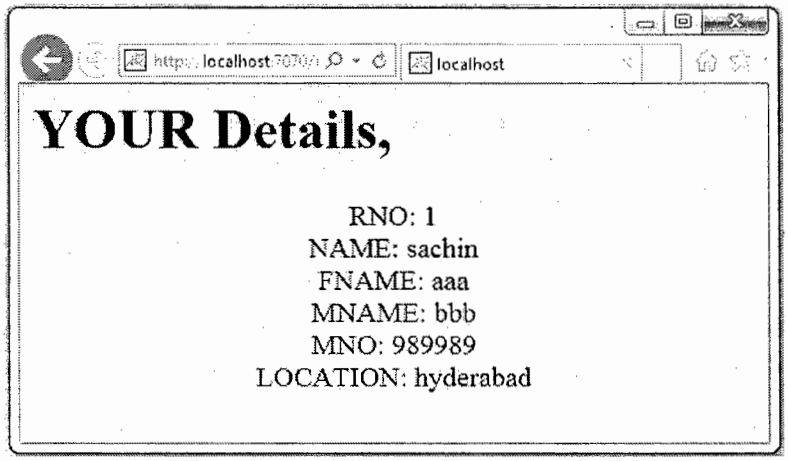
address.jsp

```
<html>
<head>
</head>
<body bgcolor="cyan">
<h1>Address Details Here,</h1>
<%
String rno = request.getParameter("rno");
String name = request.getParameter("name");
String fname = request.getParameter("fname");
String mname = request.getParameter("mname");
%>
<center>
<form  action="store.jsp" method="get">
<input type="hidden" name="rno" value="<%=rno%>"/>
<br/>
<input type="hidden" name="name" value="<%=name%>"/>
<br/>
<input type="hidden" name="fname" value="<%=fname%>"/>
<br/>
<input type="hidden" name="mname" value="<%=mname%>"/>
<br/>
  mno: <input type="text"  name="mno"/><br/>
 address:<input type="text" name="address"/><br/>
<input type="submit" value="next"/>
</form>
</center>
</body>
</html>
```



store.jsp

```
<html>
<head>
</head>
<body bgcolor="cyan">
<h1>Student Details Are,</h1>
<%
String rno = request.getParameter("rno");
String name = request.getParameter("name");
String fname = request.getParameter("fname");
String mname = request.getParameter("mname");
String mno = request.getParameter("mno");
String address = request.getParameter("address");
%>
<center>
RNO:<%=rno%><br/>
NAME:<%=name%><br/>
FNAME:<%=fname%><br/>
MNAME:<%=mname%><br/>
MNO:<%=mno%><br/>
ADDRESS:<%=address%>
</center>
</body>
</html>
```



disadvantages of hidden variables

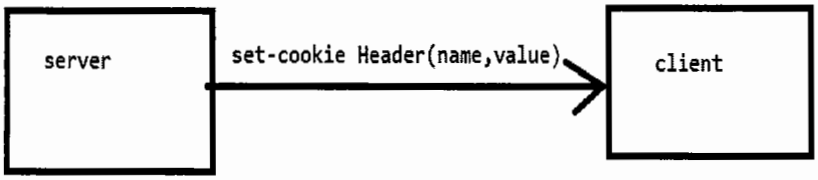
1. if we want to carry huze amount of data then it will be complicated using hidden variables.
2. hidden variables are not recommended to carry any sensitive data

advantages of hidden variables

1. hidden variables consume less memory.
2. once we close the browser the data of hidden variables willbe cleared automatically.

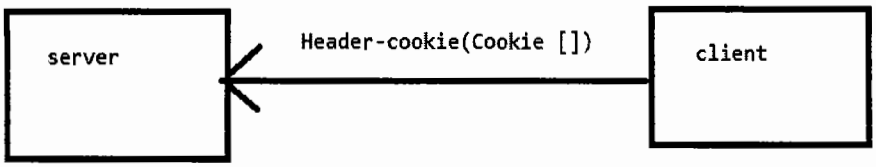
2. cookies

-cookies are also used to tranfer the data between multiple form based applciations.
-simply a cookie is a small peice of information send by the server to any client
-Every cookie will store the data in the form of key value pairs
-Server will add the cookie to reponse header by using Set-Cookie header and send to the client



- cookies willbe stored in the browser by using domain name of website.
- browser can contain cookies from multiple domains

-client need to use HeaderCookie to get all the cookies available in the browser



- we can create following 2 types of cookies

1. non-persistant cookies

-non-persistant cookies means cookies data will be deleted when we close the browser immediately.
-bydefault we get non-persistant cookies

2.persistant cookies

-persistant cookies means cookies data will be available even though we close the browser upto specified time.
-if we want to get persistant cookies we have to use setMaxAge() method.

creating Cookies in servlet

SendCookiesServlet.java

```
import javax.servlet.http.*;
import java.io.*;
public class SendCookiesServlet extends HttpServlet{
```

```
 public  void service(HttpServletRequest request,HttpServletResponse response) throws IOException{
  PrintWriter out = response.getWriter();
  Cookie  c1 = new Cookie("uname","sachin");
  Cookie  c2 = new Cookie("pwd","123456");
   response.addCookie(c1);
   response.addCookie(c2);
 out.print("Cookies are added....");
 }
}
```

RecieveCookiesServlet.java

```
import javax.servlet.http.*;
import java.io.*;
public  class  RecieveCookiesServlet  extends  HttpServlet{
public  void service(HttpServletRequest request,HttpServletResponse response) throws IOException{

  PrintWriter out = response.getWriter();
  Cookie  cks[] = request.getCookies();
if(cks!=null){
for(int i=0;i<cks.length;i++){
  out.print(cks[i].getName()+": "+cks[i].getValue()+"<br/>");
  }
 }
}
}
```

creating cookies in jsp

CookieTest1.jsp

```
<html>
  <head>
</head>
<body>
<%
Cookie  c1 = new Cookie("uname","sachin");
Cookie  c2 = new Cookie("pwd","12345");
response.addCookie(c1);
response.addCookie(c2);
out.print("Cookies are send to the Client");
%>
</body>
</html>
```

- reading Cookies

CookieTest2.jsp

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

```
<body>
<%
Cookie  ck [] = request.getCookies();
if(ck!=null){
out.print("Number of Cookies:"+ ck.length+"<br/>");
for(int i=0;i<ck.length;i++){
out.print(ck[i].getName()+":"+ck[i].getValue()+"<br/>");
}
}
else{
out.print("sorry no cookies are available");
}
%>
</body>
</html>
```

Using Cookies in multiple form based applications

student.html

```
<html>
<head>
</head>
<body bgcolor="cyan">
<h1>Student Details Here,</h1>
<center>
<form name="studform"  action="parents.jsp" method="get">
   rno: <input type="text"  name="rno"/><br/>
  name:<input type="text" name="name"/><br/>
  <input type="submit" value="next"/>
</form>
</center>
</body>
</html>
```

parents.jsp

```
<html>
<head>
</head>
<body bgcolor="cyan">
<%
String rno = request.getParameter("rno");
String name = request.getParameter("name");
Cookie  c1 =  new Cookie("rno",rno);
Cookie  c2 =  new Cookie("name",name);
response.addCookie(c1);
response.addCookie(c2);
%>
```

```
<center>
<form  action="address.jsp" method="get">
  fname: <input type="text"  name="fname"/><br/>
 mname:<input type="text" name="mname"/><br/>
  <input type="submit" value="next"/>
</form>
</center>
</body>
</html>
```

address.jsp

```
<html>
<head>
</head>
<body bgcolor="cyan">
<h1>Login Here,</h1>
<%
String fname = request.getParameter("fname");
String mname = request.getParameter("mname");
Cookie  c1 =  new Cookie("fname",fname);
Cookie  c2 =  new Cookie("mname",mname);
response.addCookie(c1);
response.addCookie(c2);
%>
<center>
<form  action="store.jsp" method="get">
  mno: <input type="text"  name="mno"/><br/>
 address:<input type="text" name="address"/><br/>
<input type="submit" value="next"/>
</form>
</center>
</body>
</html>
```

store.jsp

```
<html>
<head>
</head>
<body bgcolor="cyan">
<%
Cookie  cks[] = request.getCookies();
String rno=null;
String name=null;
String fname=null;
String mname=null;
```

```
if(cks!=null){
for(int i=0;i<cks.length;i++){
 if(cks[i].getName().equals("rno")){
rno=cks[i].getValue();
}
else
if(cks[i].getName().equals("name")){
name=cks[i].getValue();
}
else
if(cks[i].getName().equals("fname")){
fname=cks[i].getValue();
}
else
if(cks[i].getName().equals("mname")){
mname=cks[i].getValue();
}
}
}
String mno = request.getParameter("mno");
String address = request.getParameter("address");
%>
<center>
RNO:<%=rno%><br/>
NAME:<%=name%><br/>
FNAME:<%=fname%><br/>
MNAME:<%=mname%><br/>
MNO:<%=mno%><br/>
ADDRESS:<%=address%>
</center>
</body>
</html>
```

managing cookies in the google chrome

- in google chrome open tools
- settings
- advanced settings
-privacy
-content settings

Note:

- for 1 domain we can store upto 30 cookies
- browser allows upto 300 cookies
- each cookie stores upto 4kb data.

-we can also create a persistant cookie by calling setMaxAge()method

Eg:

```
<%
Cookie  c = new Cookie("uname","sachin");
 c.setMaxAge(10000);
response.addCookie(c);
%>
```

disadvantags of cookies

1. if client disable the cookies inthe browser we can work with cookies
2. for security reasons it is nbot recommended to use cookies.
3. if we want to huze amount of data bw server and client it will be complicated.

advantags of cookies

1. compared to hidden variables cookies are better to transfer the data beause once we store the cookie we can get the data in any page of the website directly.

session

- creating a session object means creating an object for a class which is implementing javax.servlet.http.HttpSession interface
-session object is also used to tranfer the data between multiple form based application.
- In the case of servlets we have to write the code for creating session object

Eg:

```
import  javax.servlet.http.*;
import  javax.servlet.*;
import java.io.*;
public  class SessionServlet  extends  HttpServlet{
public void service(HttpServletRequest  request, HttpServletResponse  response) throws IOException{
res.setContentType("text/html");
PrintWriter   out =  res.getWriter();
HttpSession  session = request.getSession(true);
out.println(session.getId()+"<br/>");
out.println(session.getCreationTime()+"<br/>");
out.println(session.isNew());
out.println("hi");
 }
}
```

-but in jsp by default session object is enable when ever client send the request to the server internally. so we can write jsp program directly like follows to get Session information

- in jsp session is a implicit object.

sessiondemo.jsp

```
<%
out.println(session.getId()+"<br/>");
out.println(session.getCreationTime()+"<br/>");
out.println(session.isNew());
out.println("hi");
%>
```

-but if we want to handle session object in jsp explicitly then we have to write following page redirective in jsp program

```
<%@ page sesion="false"%>
```

-now the jsp program must be written like follows to get session information.

sessiondemo.jsp

```
<%@ page sesion="false"%>
<%
HttpSession session = request.getSession(true);
out.println(session.getId()+"<br/>");
out.println(session.getCreationTime()+"<br/>");
out.println(session.isNew());
out.println("hi");
%>
```

? how and when the session object is created by server

- when client send the request to the server for the first time then server creates a session object and now server createsa unique id which is assosiated with this session object
- next server will create cookie with the name JSESSIONID which holds session object as a value and send to the client

-If we wat to request the Server to create session object we have to following methods of request object.

```
HttpSession  getSession(true)
HttpSession  getSession(false)
```

- session objects are used to store the data
- session can support to store any kind of data
- to work with the session object ad to store the data or to get the data we have to use following methods

```
void  setAttribute("key",Object obj);
Object  getAttribute(key);
void  removeAttribute(key);
void invalidate()
```

- program to transfer the data between multiple form based applications using session object

student.html

```
<html>
<head>
</head>
<body bgcolor="cyan">
<h1>Student Details Here,</h1>
<center>
<form name="studform"  action="parents.jsp" method="get">
   rno: <input type="text"  name="rno"/><br/>
  name:<input type="text" name="name"/><br/>
   <input type="submit" value="next"/>
</form>
</center>
</body>
</html>
```

parents.jsp

```
<html>
<head>
</head>
<body bgcolor="cyan">
<h1>Parent Details Here,</h1>
<%
String rno = request.getParameter("rno");
String name = request.getParameter("name");
session.setAttribute("rno",rno);
session.setAttribute("name",name);
%>
<center>
<form action="address.jsp" method="get">
  fname: <input type="text"  name="fname"/><br/>
 mname:<input type="text" name="mname"/><br/>
  <input type="submit" value="next"/>
</form>
</center>
</body>
</html>
```

address.jsp

```
<html>
<head>
</head>
<body bgcolor="cyan">
<h1>Address Details Here,</h1>
<%
String fname = request.getParameter("fname");
String mname = request.getParameter("mname");
session.setAttribute("fname",fname);
session.setAttribute("mname",mname);
%>
<center>
<form  action="store.jsp" method="get">
  mno: <input type="text"  name="mno"/><br/>
 address:<input type="text" name="address"/><br/>
<input type="submit" value="next"/>
</form>
</center>
</body>
</html>
```

store.jsp

```
<%@ page session="false"%>
<html>
<head>
</head>
<body bgcolor="cyan">
<%
HttpSession  session = request.getSession(true);
String rno=(String)session.getAttribute("rno");
String name=(String)session.getAttribute("name");
String fname=(String)session.getAttribute("fname");
String mname=(String)session.getAttribute("mname");;
String mno = request.getParameter("mno");
String address = request.getParameter("address");
%>
<center>
RNO:<%=rno%><br/>
NAME:<%=name%><br/>
FNAME:<%=fname%><br/>
MNAME:<%=mname%><br/>
MNO:<%=mno%><br/>
ADDRESS:<%=address%>
<%
//session.invalidate();
%>
</center>
</body>
</html>
```

? when session object is deleted

case 1: when we close the browser

case2: when we call session.invalidate() in our program generally session.invalidate() is called as a part of logout pages.

case 3:when we call session.setMaxInactiveInterval(int) inour program

Eg:

```
HttpSession  session = request.getSession(true);
  session.setMaxInactiveInterval(30);
```

case 4: by changing the session-timeout in web.xml of server.

web.xml (Tomcat6.0/conf/web.xml)

```
<session-config>
    <session-timeout>30</session-timeout>
  </session-config>
```

Note:

it is not recommended because all the projects runing on the server will be effected.

case 5: by changing the session-timeout in web.xml of our project

Eg:

web.xml (myproject/WEB-INF)

```
<session-config>
    <session-timeout>30</session-timeout>
  </session-config>
```

Note:

now only our project will be effected.

- session object can be removed in following 3 levels

1. program level
2. project level
3. server level

- we can also get all the key-value pairs available on the session object

Eg:

```
<%
java.util.Enumeration   e =  session.getAttributeNames();
while(e.hasMoreElements()){
String  name = (String) e.nextElement();
String  value = (String) session.getAttribute(name);
out.println("NAME:"+name+"VALUE:"+value+"<br/>");
}
%>
```

Scoped Variables

-Scoped Variables are used to store the values in different levels.

-we have following 4 types of scoped variables

1.PageScope
2.requestScope
3.sessionScope
4.applicationScope

1.PageScope

- Storing the data as a part of pageContext implicit Object is called page Scope, This data is available until the _jspService() method under execution.

```
<%
pageContext.setAttribute("uname","sachin");
String name=(String)pageContext.getAttribute("uname");
out.println(name);
pageContext.removeAttribute("uname");
name=(String)pageContext.getAttribute("uname");
out.println(name);
%>
```

2.requestScope

Storing the data into request object (implicit Object) is called as request Scope.

? when to use requestScope

if we want to send the request to multiple servlets then in order to share the requested data in between multiple servlets we have to use requestScope

3.sessionScope

Storing the data into session object (implicit Object) is called as session Scope.

? when to use sessionScope

if we want to share the data between multiple forms in our application then we have to use sessionScope

4.applicationScope

Storing the data into ServletContext object is called as applicationScope.

? when to use applicationScope

if we want to allow any client to access the data from any browser then we have to use sessionScope

if we store data in applicationScope then all servlets or jsp of that project can access it.

RequestDispatcher

-RequestDispatcher is an interface which is used to send the request to multiple servlet pages or jsp at a time.

-sending the request to multiple servlet pages or jsp at a time is the concept called servlet chaining

-RequestDispatcher contains following 2 methods

```
void inlcude(HttpServletRequest,HttpServletResponse)
void forward(HttpServletRequest,HttpServletResponse)
```

include()

- if we use this method the current servlet output append with the output of other requested servlet and send to the client.

forward()

-If we use this method the current servlet output replace with the output of other requested servlet and send to the client.

-RequestDispatcher is an interface and if we want to create an object then we have to use ServletContext object or directly request object like follows

1. using ServletContext object

Eg:

```
ServletContext application = this.getServletContext();
RequestDispatcher rd = application.getRequestDispatcher("url");
```

2. using request object directly

Eg:

```
RequestDispatcher rd = request.getRequestDispatcher("url");
```

Note:

here url of the any servlet must begin with " / "

Eg:

FirstServlet.java

```
import javax.servlet.http.*;
import javax.servlet.*;
import java.io.*;
public class FirstServlet extends HttpServlet{
public void service(HttpServletRequest request,HttpServletResponse response) throws
IOException,ServletException{
ServletContext application = this.getServletContext();
PrintWriter out = response.getWriter();
out.println("We are in First Servlet");
```

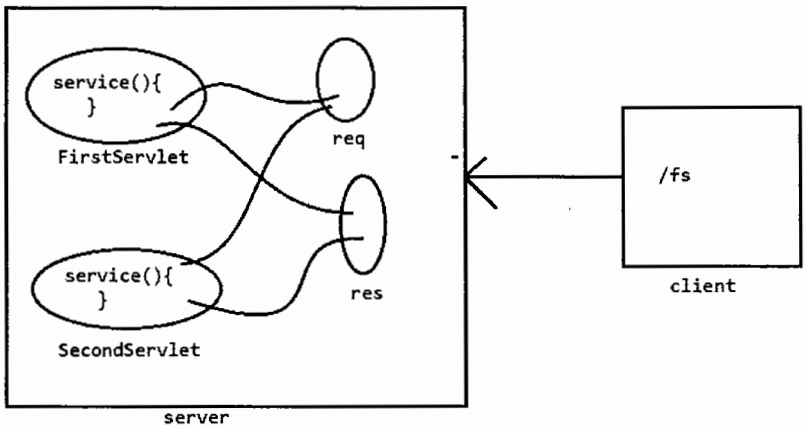
```
RequestDispatcher  rd = application.getRequestDispatcher("/ss");
rd.include(request,response);
}
}
```

SecondServlet.java

```
import  javax.servlet.http.*;
import  javax.servlet.*;
import java.io.*;
public  class SecondServlet  extends  HttpServlet{
public void service(HttpServletRequest request,HttpServletResponse response) throws
IOException,ServletException{
PrintWriter out = response.getWriter();
out.println("We are in Second Servlet");
}
}
```

when ever client send the request to FirstServlet then server will create request object and response object for First Servlet and these objects are handover to service() method of FirstServlet

when RequestDispatcher encounters then server create servlet object for SecondServlet by supplying same request and response objects then include() method will club the output of both First Servlet and SecondServlet and send to the client



Note:

-We can call Firsr Servlet as Calling Servlet and Second Servlet is called as Called Servlet.

-From the servlet we call any number of rd.inlcude() methods to include multple servlets

-But From the servlet we call only one rd.forward() method to include any one servlet otherwise we get IllegalStateException

- In this case of JSP's we can directly get RequestDispatcher by using implicit objects application or request like follows

Eg:

```
<%
 RequestDispatcher  rd = application.getRequestDispatcher("/ss");
 rd.include(request,response);
%>
```

One.jsp

```
<%
out.println("We are in One.jsp");
 RequestDispatcher  rd = application.getRequestDispatcher("/Two.jsp");
 rd.include(request,response);
%>
```

Two.jsp

```
<%
out.println("We are in Two.jsp");
%>
```

Three.jsp

```
<%
out.println("We are in Three.jsp");
%>
```

jsp action tags

It is recommended not to write java code inside the JSP's. to resolve this problem sun micro system people have given following 5 jsp action tags

1.include
2.forward
3.useBean
4.setProperty
5.getProperty

1. include

this action tag is used to send the request to other servlet or jsp pages and include the output of other servlet and send to the client.

syntax:

```
<jsp:include  attribute="value"/>
```

One.jsp

```
<%
out.println("We are in One.jsp");
%>
<jsp:include  page="/Two.jsp"/>
<jsp:include  page="/Three.jsp"/>
```

we can call this jsp include action tag is dynamic include because every time when client send the request then request send to multiple servlets and output will be clubbed and send to the client.

2.forward

forward is jsp action tag which forward the result to sepecified jsp or servlet.

syntax:

```
<jsp:forward  attribute="value"/>
```

Eg:

One.jsp

```
<%
out.println("We are in One.jsp");
%>
<jsp:forward  page="/Two.jsp"/>
```

jsp include directive

jsp include directives also used to perform the servlet chaining

syntax:

```
<%@ include  file="" %>
```

Eg:

One.jsp

```
<%
out.println("We are in One.jsp");
%>
<%@ include  file="/Two.jsp"%>
```

Note:

- when we use getRequestDispatcher("url") method for servlet chaining we must begin url with " / "
- when we use getNamedDispatcher("url") method for servlet chaining we no need to begin url with " /"

One.jsp

```
<%
out.println("We are in One.jsp");
%>
<%
RequestDispatcher rd=application.getNamedDispatcher("ss");
 rd.include(request,response);
%>
```

case1:

RequestDispatcher can be used to dispatch the request from one servlet to another servlet if they are belongs to the same project

case2:

RequestDispatcher can be used to dispatch the request from one servlet to another servlet if they are belongs to two different projects which are working on same server.

case3:

But RequestDispatcher can not be used to dispatch the request from one servlet to another servlet if they are belongs to two different projects which are working on 2 different servers.

Java Beans

- A Java bean is a java class which created by following standared rules

1.It must be created using any package
2.It must be public class
3.It must implement Serialiazable interface
4.It must contain a public 0 parameterized constructor
5.It must contain all the fileds or variables as private.
6.It must contain setter methods and getter methods for each field.
7.It must contain all setter methods and getter methods as public

Eg:

```
package  com.inetsolv.samplejspproject;
public  class  Student implements  java.io.Serializable{
private int rno;
private String name;
private double fee;
```

```
public Student(){
}
public void   setRno(int rno){
this.rno=rno;
}
public int   getRno(){
return  rno;
}
public void   setName(String name){
this.name=name;
}
public String   getName(){
return  name;
}
public  void   setFee(double fee){
this.fee=fee;
}
public  double   getFee(){
return  fee;
}
}
```

creating an object and using properties of Java beans in jsp program

```
<%@ page  import="com.inetsolv.samplejspproject.*"%>
<%
Student s = new Student();
s.setRno(1);
s.setName("sachin");
s.setFee(1300.0);
out.println("<br/>RNO:"+s.getRno());
out.println("<br/>NAM:"+s.getName());
out.println("<br/>FEE:"+s.getFee());
%>
```

- we can also use jsp action tags for creating an object for any java bean and for accessing its properties.

3.useBean

-useBean is jsp action tag using which we an create an object for java bean without writing any java code.

syntax:

```
<jsp:useBean  id="instancename" class="fullyqualifiedClassName" scope="page/session/request/application"/>
```

here scope indicates where to locate the created java bean object default scope is page.

4.setProperty

-setProperty is the jsp action tag using which we can call setter methods of Javabean and set the value without writing any java code

Syntax:

```
<jsp:setProperty name="instancename" property="variablename" value="anyvalue"/>
```

5.getProperty

-getProperty is the jsp action tag using which we can call getter methods of Javabean and get the value without writing any java code

Syntax:

```
<jsp:getProperty name="instancename" property="variablename"/>
```

Note:

-setProperty just set the value and doesn't display any value on the JSP page.

-getProperty get the value and display the value on the JSP page.

Eg:

beandemo.jsp

```
<html>
<head>
</head>
<jsp:useBean id="st" class="com.inetsolv.samplejspproject.Student" scope="page"/>
<center>
<jsp:setProperty name="st" property="rno" value="1"/>
<jsp:setProperty name="st" property="name" value="suresh"/>
<jsp:setProperty name="st" property="fee" value="1500.00"/>
<br/>RNO:<jsp:getProperty name="st" property="rno"/>
<br/>NAM:<jsp:getProperty name="st" property="name"/>
<br/>FEE:<jsp:getProperty name="st" property="fee"/>
</center>
</body>
</html>
```

7. Jsp custum tags

- We can also create our own tags in jsp
- If we want to create our own tag then we have to follow following steps

1. creating a class that extends a class " SimpleTagSupport "
2. override the doTag() method of " SimpleTagSupport " class

Eg:

HelloTag.java

```
package com.inetsolv.samplejspproject;
import javax.servlet.jsp.*;
import javax.servlet.jsp.tagext.*;
import java.io.*;
public class HelloTag extends SimpleTagSupport{
public void doTag() throws IOException{
JspWriter out = getJspContext().getOut();
out.println("Hi Im Hello Tag");
}
}
```

-To compile this program we have to set classpath to jsp-pi.jar file which is available in tomcat server.

```
> set classpath=jsp-api.jar;.;
> javac -d . HelloTag.java
```

3. Once weget .class for our Custom Tag we have to configure .class file in custom.tld file where we decide name of the tag and its attributes,...

Eg:

custom.tld

```
<taglib>
 <tlib-version>1.0</tlib-version>
 <jsp-version>2.0</jsp-version>
<short-name>Example TLD</short-name>
<tag>
<name>Hello</name>
<tag-class>com.inetsolv.samplejspproject.HelloTag</tag-class>
  <body-content>empty</body-content>
 </tag>
</taglib>
```

4. following is the example how to use custom tags in jsp

Eg:

```
<%@ taglib prefix="inet" uri="WEB-INF/custom.tld"%>
<html>
 <head>
  <title>A sample custom tag</title>
 </head>
 <body>
  <inet:Hello/>
 </body>
</html>
```

custom tag attributes

1.

```
package  com.inetsolv.samplejspproject;
import  javax.servlet.jsp.*;
import  javax.servlet.jsp.tagext.*;
import java.io.*;
public  class HelloTag extends  SimpleTagSupport{
private String message;
  public void setMessage(String msg) {
    this.message = msg;
  }
  public void doTag()  throws JspException, IOException  {
     if (message != null) {
       /* Use message from attribute */
       JspWriter out = getJspContext().getOut();
       out.println( message );
     }
     else {
      JspWriter out = getJspContext().getOut();
```

```
        out.println("default message inetsolv");
      }
    }
  }
```

2.

custom.tld

```
<taglib>
  <tlib-version>1.0</tlib-version>
  <jsp-version>2.0</jsp-version>
  <short-name>Example TLD</short-name>
  <tag>
    <name>Hello</name>
    <tag-class>com.inetsolv.samplejspproject.HelloTag</tag-class>
    <body-content>empty</body-content>
   <attribute>
      <name>message</name>
    </attribute>
  </tag>
</taglib>
```

3.

```
<%@ taglib prefix="inet" uri="WEB-INF/custom.tld"%>
<html>
  <head>
    <title>A sample custom tag</title>
  </head>
  <body>
    <inet:Hello message="my attribute message"/>
  </body>
</html>
```

8. expresion language(EL)

- some times while setting the value for any java bean property we need to write expression which we can not write directly. this problem can be solved by using expresion language(EL).
-expresion language(EL) is simply a process of writing different mathematical expressions in jsp
-expresion language(EL) supports arithmeitcal ops, comparision operators, incr-decr operators,...
- if we want to write any expression using EL we have to use following syntax

  syntax:

  ${expression}

-but in the jsp if we want to execute the expressions which are written by using EL we must write following js directive

  <%@ page isELIgnored="true|false"%>

//wap to demo on EL

java bean

Circle.java

```
package  com.inetsolv.samplejspproject;
public  class  Circle implements  java.io.Serializable{
private double radius;
private double area;
private double perimeter;
public Circle(){
}
public void   setRadius(double  radius ){
this.radius=radius;
}
public double   getRadius(){
return  radius;
}
public void   setArea(double area){
this.area=area;
}
public double   getArea(){
return  area;
}
public  void   setPerimeter(double perimeter){
this.perimeter=perimeter;
}
public  double   getPerimeter(){
return  perimeter;
}
}
```

eljspdemo.jsp

```
<%@ page  isELIgnored="false"%>
<html>
<head>
</head>
<jsp:useBean id="cir" class="com.inetsolv.samplejspproject.Circle" scope="page"/>
<jsp:setProperty name="cir" property="radius" value="5"/>
<br/>Radius: <jsp:getProperty name="cir" property="radius"/>
<jsp:setProperty name="cir" property="area" value="${3.14*cir.radius*cir.radius}"/>
<jsp:setProperty name="cir" property="perimeter" value="${2*3.14*cir.radius}"/>
<br/>AREA: <jsp:getProperty name="cir" property="area"/>
<br/>PERIMETER:<jsp:getProperty name="cir" property="perimeter"/>
</center>
</body>
</html>
```

-we can also write expression language statements inside the jsp body like follows

Eg1:

```
 ${10*10*10}
```

Eg2:

```
<jsp:text>
 ${10*10*10}
</jsp:text>
```

Eg3:

```
<jsp:text>
Circle radius is : ${3.14*cir.radius*cir.radius}
</jsp:text>
```

MVC Architecture

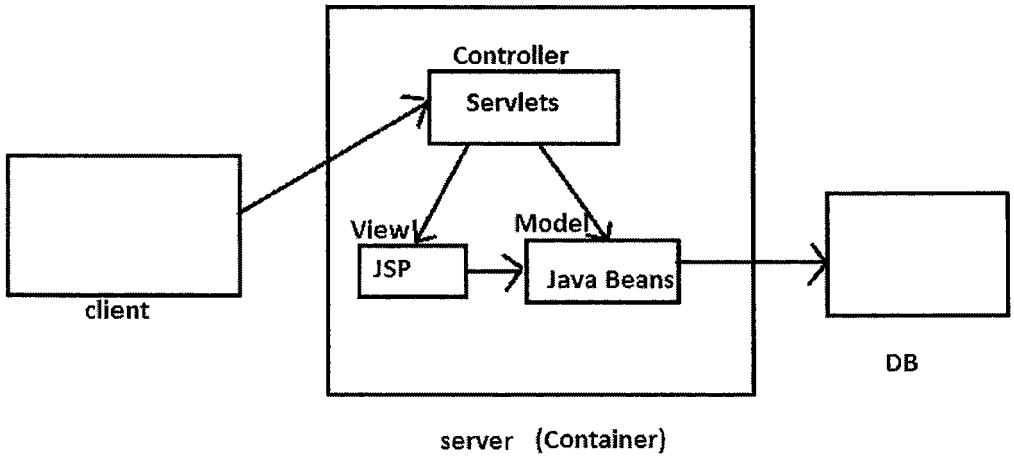
-MVC Architecture is a design pattern which is used for the development of any web based applciation.

-MVC stands for Model view Controller

Model: model contains all the java beans which are used to process the data

View: View contains all the jsp pages, html pages which are used to display the data presentation logic

Controller: Controller contains all the servlet classes which are used to write the business logic
somtimes we also write the business logic as a part of java beans.



JSTL

-JSTL stands for Java standared tag libraries which are also used to develop the jsp files without writing any java code.

-JSTL tags are given by sun micro system people and relesed as a part of jstl.jar files.

-we have totally 5 types of JSTL tags.

1. core jstl tags
2. sql jstl tags
3. xml jstl tags
4. formatting(internationalization) jstl tags
5. functions based jstl tags

core jstl tags

-core jstl tags are used to perform some core operations like printing, assigning, if test , iteration,.... in any jsp page without writing java code.

-to use core jstl tags we have to write following taglib directive inside the jsp file.

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
```

<c:out>

this core jstl tag is used display the content on the page

Eg:

```
<c:out value="sample jsp">
</c:out>
```

<c:forEach>

this core jstl tag is used to iterate or repeatedly execte the group of statements

Eg:

```
<c:forEach var="n" begin="1" end="10">
<c:out value="${n}">
</c:out>
</c:forEach>
```

<c:set>

this core jstl tag is used to declare variable and set the value

Eg:

```
<c:set var="n" value="11"/>
```

<c:if>

this core jstl tag is used to execute the statements based on condition.

Eg:

```
<c:set var="n" value="11"/>
<c:if test="${n%2==0}">
<c:out value="It is Even Number">
</c:out>
</c:if>
<c:if test="${n%2==1}">
<c:out value="It is Odd Number">
</c:out>
</c:if>
```

<c:choose>

The core jstl tag works like a Java **switch** statement in that it lets you choose between a number of alternatives. Where the **switch** statement has **case** statements, the <c:choose> tag has <c:when> tags. A switch statement has **default** clause to specify a default action and similar way <c:choose> has <c:otherwise> as default clause.

Eg:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<c:set var="salary" scope="session" value="${2000*2}"/>
<p>Your salary is : <c:out value="${salary}"/></p>
<c:choose>
  <c:when test="${salary <= 0}">
    Salary is very low to survive.
  </c:when>
  <c:when test="${salary > 1000}">
    Salary is very good.
  </c:when>
  <c:otherwise>
    No comments
  </c:otherwise>
```

```
</c:choose>
```

<c:import>

- this core jstl tag is used to include the result of specified URL.
- we can write any number of <c:import>

Eg:

```
<c:import url="Two.jsp"/>
```

<c:redirect>

The core jstl tag used to forward our servlet to any particular html or other servlet or jsp

Eg:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<c:redirect url="http://www.photofuntoos.com"/>
```

sql jstl tags

- sql jstl tags are used to communicate with any database in any jsp page without writing java code.
- to use sql jstl tags we have to write following taglib directive inside the jsp file.

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql"%>
```

<sql:setDataSource>

- This sql jstl tag is used to create the datasource by specifying JDBC Driver, URL, username and password

syntax:

```
<sql:setDataSource var="" driver="" url="" user="" password="">
```

<sql:query>

-This sql jstl tag is used to execute the query

syntax:

```
<sql:query var="" datasource="" >
  query;
</sql:query>
```

Eg:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql"%>
<%@ page isELIgnored="false"%>
<html>
<head>
</head>
<sql:setDataSource var="myDataSource" driver="oracle.jdbc.driver.OracleDriver"
url="jdbc:oracle:thin:@localhost:1521:xe" user="inetsolv" password="students"/>
<sql:query var="emprecords" dataSource="${myDataSource}">
select * from emp
</sql:query>
<table border="1" bordercolor="blue" align="center">
<tr>
<td>EMPNO</td>
<td>ENAME</td>
```

```
<td>JOB</td>
<td>SALARY</td>
</tr>
<c:forEach var="row"  items="${emprecords.rows}">
<tr>
<td><c:out value="${row.empno}"></c:out></td>
<td><c:out value="${row.ename}"></c:out></td>
<td><c:out value="${row.job}"></c:out></td>
<td><c:out value="${row.sal}"></c:out></td>
</tr>
</c:forEach>
</body>
</html>
```

<sql:update>

The sql jstl tag executes an SQL statement that does not return data, for example SQL INSERT, UPDATE, or DELETE statements.

Eg:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/sql" prefix="sql"%>
<%@ page  isELIgnored="false"%>
<sql:setDataSource var="myDataSource" driver="oracle.jdbc.driver.OracleDriver"
url="jdbc:oracle:thin:@localhost:1521:xe" user="inetsolv" password="students"/>
<sql:update dataSource="${myDataSource}"  var="count1">
  INSERT INTO Emp VALUES (104, 'sachin, 'crick',15000);
</sql:update>
No.of Records are inserted: <c:out value="${count1}">
<sql:update dataSource="${myDataSource}"  var="count2">
  update emp set sal=sal+500;
</sql:update>
No.of Records are updated: <c:out value="${count2}">
```

functions based jtsl tags

- this functions based jtsl tags are used for performing some predefined string operations like finding length, converting string into lowercase, uppercase,....
- to use functions based jtsl tags we have to write following taglib directive inside the jsp file.

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn"%>
```

Eg:

sample.jsp

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn"%>
<%@ page  isELIgnored="false"%>
<html>
<head>
</head>
<body>
<c:set var="str" value="java language">
```

```
</c:set>
<br/><c:out value="${fn:length(str)}"/>
<br/><c:out value="${fn:toLowerCase(str)}"/>
<br/><c:out value="${fn:toUpperCase(str)}"/>
</body>
</html>
```

formatting(internationalization) jstl tags

- this formatting jstl tags are used to change the format date or any number.
- to use formatting jstl tags we have to write following taglib directive inside the jsp file.

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt"  prefix="fmt"%>
```

<fmt:formatNumber>:

-this formatting jstl tag used to change the format of thenumber

Eg:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
<%@ page isELIgnored="false"%>
<html>
<head>
</head>
<body>
<c:set var="num" value="12345.6789">
</c:set>
<br/><fmt:formatNumber value="${num}" type="number" maxFractionDigits="2"></fmt:formatNumber>
<br/><fmt:formatNumber value="${num}" pattern="####.##" type="number"></fmt:formatNumber>
<br/><fmt:formatNumber value="${num}" type="currency">
</fmt:formatNumber>
</body>
</html>
```

<fmt:formatDate>:

-This jstl formatting tag is used to change the format of the Date

Eg:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
<%@ page isELIgnored="false"%>
<html>
<head>
</head>
<body>
<c:set var="cdt" value="<%=new java.util.Date()%>">
</c:set>
<br/><fmt:formatDate value="${cdt}" type="time"/>
<br/><fmt:formatDate value="${cdt}" type="date"/>
<br/><fmt:formatDate value="${cdt}" type="both" dateStyle="long"/>
<br/><fmt:formatDate value="${cdt}" type="date" dateStyle="medium"/>
```

```
<br/><fmt:formatDate value="${cdt}" type="date" pattern="yyyy-MM-dd" />
</body>
</html>
```

xml jstl tags

-xml jstl tags used to read, process ,format the xml files

-to use jstlxml tags we have to write following tablib directive in jsp file.

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/xml"  prefix="x"%>
```

Eg:

student.xml

```
<inetsolv>
<student>
<rno>1</rno>
<name>sachin</name>
<mno>9767865754</mno>
<address>hyd</address>
</student>
<student>
<rno>2</rno>
<name>sehwagh</name>
<mno>8767855554</mno>
<address>chennai</address>
</student>
<student>
<rno>3</rno>
<name>dhoni</name>
<mno>9767777754</mno>
<address>delhi</address>
</student>
</inetsolv>
```

sample.jsp

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/xml"  prefix="x"%>
<%@ page isELIgnored="false"%>
<html>
<head>
</head>
<body>
<c:import var="studinfo" url="http://localhost:7777/myproject/student.xml"/>
<x:parse var="result" xml="${studinfo}"/>
<br/>Name: <x:out select="$result/inetsolv/student[1]/name"/>
<br/>RNO: <x:out select="$result/inetsolv/student[1]/rno"/>
<br/>MNO: <x:out select="$result/inetsolv/student[1]/mno"/>
<br/>Address: <x:out select="$result/inetsolv/student[1]/address"/>
</body>
```

</html>

annontations(metadata)

-annontations are the compile time directives which are used to give an intruction to compiler

-annontations are written just like a comment using @ symbol

syntax:

```
@Annotationname
```

here @ is called a notation for annontation

we have following 3 types of predefined annontations

@Deprecated

@Ovveride

@SupressWarnings

@Deprecated

this annonation give an instruction to compiler that particular component is deprecated which is means not recommended to use.

Eg:

```
class Student{
private int rno;
private String name;
private double fee;
public Student(){
}
@Deprecated
public void setRno(int rno){
this.rno=rno;
}
public int getRno(){
return rno;
}
@Deprecated
public void setName(String name){
this.name=name;
}
public String getName(){
return name;
}
}
class Annotation1{
public static void main(String args[]){
Student s =new Student();
s.setRno(1);
s.setName("suresh");
System.out.println(s.getRno());
System.out.println(s.getName());
}
```

```
}
```

@Ovveride

This annotation is used to specify the particular method overriden method inside the child class when ever paent class method signature is changed then compilation time we get errors.

Eg:

```
 class  Parent{
 void show(int a,int b){
   System.out.println("Parent class show() method a="+a);
  }
}
class Child extends Parent{
@Override
 void show(int a){
   System.out.println("Child class show() method a="+a);
  }
}
class Annotation2{
  public  static void main(String args[]){
  Parent  p;
 p =  new Parent();
 p.show(10);
 p = new Child();
 p.show(10);
}
}
```

@SuppressWarnings

this annotation is used to specify in the particular program to ignore the warngings

Eg:

```
import java.util.*;
class Annotation3{

@SuppressWarnings("deprecation")
  public  static void main(String args[]){
Date  d = new Date();
System.out.println(d.getYear());
}
}
import  java.util.*;
@SuppressWarnings("unchecked")
class    Test1{
 public  static  void  main(String ar[]){
ArrayList  al = new  ArrayList();
   al.add(10);
   al.add(25);
```

```
    al.add(50);
 System.out.println(al);
}
}
```

executing Servlets without configuring in web.xml

we can also execute servlet using annotations without configuring into web.xml file

Eg:

```
import  javax.servlet.http.*;
import  javax.servlet.*;
import  javax.servlet.annotation.*;
import java.io.*;
@WebServlet("/welcome")
public  class HtmlServlet  extends  HttpServlet{
public void service(HttpServletRequest req,HttpServletResponse res) throws IOException{
System.out.println("Welcome to Servlet.....");
res.setContentType("text/html");
PrintWriter   out =  res.getWriter();
out.println("<html>");
out.println("<head><title>my response</title></head>");
out.println("<body>");
out.println("<marquee>");
out.println("<h1>Hi Welcome to Servlet....</h1>");
out.println("</marquee>");
out.println("</body>");
out.println("</html>");
out.close();
}
}
```

Note:

if we want work with this program we must use tomcat 7.0