

Rs: 190/-

ADVANCE

JAVA

NEW NOTES

BY

Mr.Nataraj

SRI RAGHAVENDRA XEROX

Software Languages Material Available

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

Cell: 9951596199

2
03/01/14

9am to 11am

JEE: Java Enterprise Edition

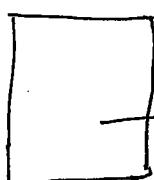
Version: Javaee6 / Jee6

- It is not a installable SW. It is set of SW specifications to develop web server / Application Server SWs.
- Working with JEE module is nothing but working with web server / application server SWs like tomcat, weblogic, websphere etc.
- This module can be used to develop web applications (websites), Distributed Apps, Enterprise Apps, n-tier apps.

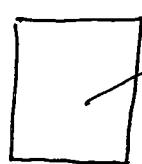
Client Server Application →

- * The Client Server application that contain multiple parallel application having the support of load balance is called Distributed Application.
- * Load Balance means the burden on one server will be shared on another server.

Client env... -

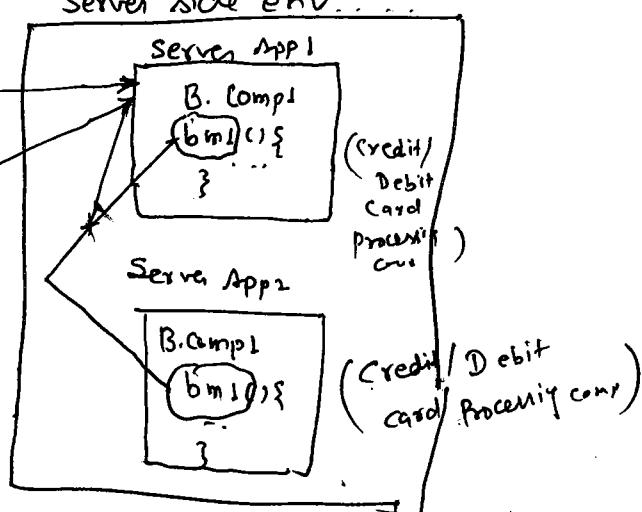


Call 1



Call 2

Server side env... -



The application that deals with large scale complex heavy weight business logic by applying the support of middleware services, is called as enterprise application.

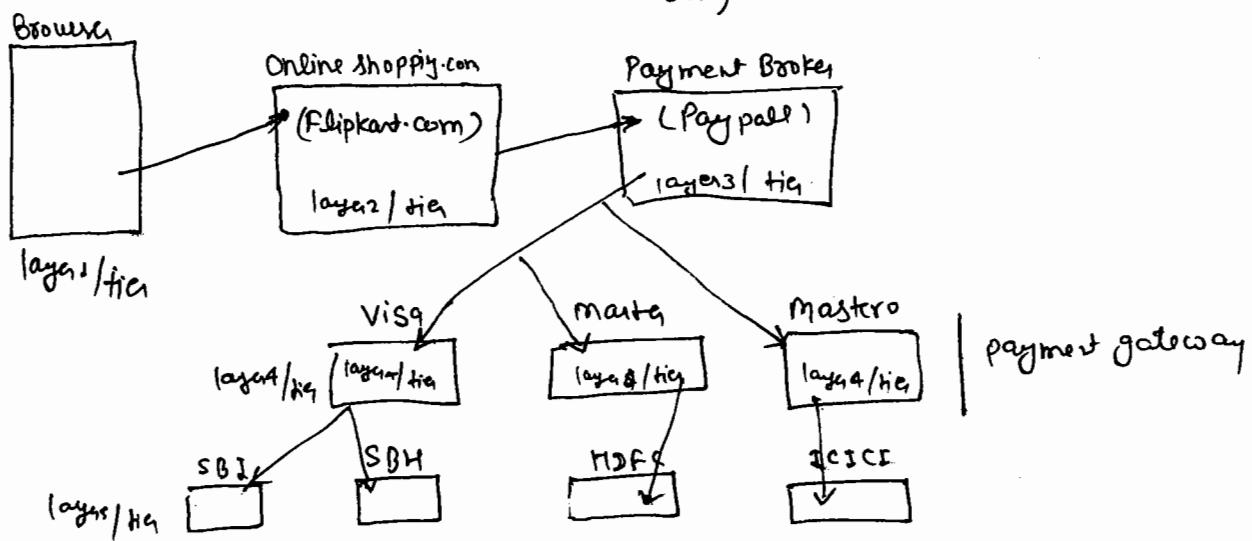
e.g. Banking Application, Online Shopping website etc.

- * An Enterprise application can be developed as web application or Distributed Appn.
- * The additional logics and optional logics that applied on the applications to make the application more perfect & accurate

e.g. Security Service, Transaction Service, Logging Server etc.

- * The application that contains multiple layer interacting with each other is called n-tier application.
- * Generally all enterprise applications are n-tier application.

Enterprise App / n-tier App (Online shopping with the support of Credit / Debit Card Purchases)



Comments:

- * Visa, Master, Mastercard are called payment gateway to maintain wild old infrastructure to perform Debit / Credit Card based transaction.
- * Payall, Pathway are the payment broken organization that maintain tieup with payment gateways for online money transactions.
- * JEE technologies are Servlet, JSP, EJB, JMS, JTA etc.
 - JSP: Java Server page.
 - EJB: Enterprise Java bean.
 - JMS: Java message service.
 - JTA: Java transaction api.
- * Struts, Spring, Hibernate are not part of JSE, JEE technologies. They are frameworks, they were given 3rd party companies by utilizing the technologies internally.
- * Programming languages are like raw material.
- * Technologies are like semi cooked food items.
- * Frameworks are like ready to eat food items.

JME: Java micro edition / mobile Edition

→ It is installable SW or mdkt SW.

→ It is useful to develop mobile applications, micro apps, AI apps (Artificial Application Apps).

Comments.

* Mobile Games, mobile Phonebook comes under mobile apps.

* In appn that are small in size with in performance begin performance are called micro appn. which are useful to provide AI to various devices like mechanical, electronical, electronic etc.

* AI makes mechanical, electronic, electrical devices think & take decisions.

e.g. Fully Automated washing machine
Robo

Note. JME is outdated because of embedded system programming, android, iphone etc.

+ 3
05/01/14

3 imp technologies

- a) S/w vendor company
- b) S/w company
- c) Client organization

The Company who create S/w's is called S/w vendor company.

e.g. IBM, Microsoft, Oracle Corporation etc.

The Company who uses S/w's given by vendor company to develop S/w projects for various ^{view} organizations is called S/w company or S/w service company.

e.g. Polaris, Wipro, etc.

The organization who give their requirements to S/w companies for developing S/w projects. is called client Organization.

e.g. S.B.I, City Bank, Central Government, Airbus etc.

Different Types of S/w projects-

- a) S/w products
- b) Service Sector Projects

The project that is developed & kept in open market to sell to client organization is called S/w product.

e.g. Tally, M.S. Office, Video Games & etc.

The project that is developed exclusively for one client organization by gathering the requirements of that client organization is called Service Sector Project.

e.g. Polarized Project for cityBank.

Version Project for AirBus. → German Based Aeroplane manufacturer Company.

diff Domains of SW Project →

Every SW project contains -

- a) Technical Domain Name
- b) Functional Domain Name

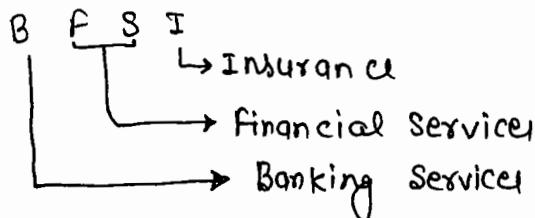
nataraj@gmail.com

Technical Domain will be decided based on technologies used in project development.

e.g Java domain, .Net domain

Functional Domain will be decided based on the category of the project that it belongs to. example - banking domain, healthcare domain, telecommunication domain and etc.

* Java is very popular for developing BFSI domain projects -



JSE module is useful to develop retail, desktop gaming, office automation domain projects.

JEE module is useful to develop banking, financial services, insurance, health care, billing domain projects.

JME module is useful to develop telecommunications, mobile gaming domain projects.

Q) What is the difference between constructor and static block?

A) Constructor is Object level one time execution block.

In this block we place logic to initialize instance variables, constructor will be executed automatically when object is created.

Static is Class level one time execution block. This block will be executed automatically when JVM loads the class for the first time. In this block we place logic to initialize static variables.

```
//DemoApp.java
```

```
class Test
```

```
{
```

```
    static {
```

```
        System.out.println("Test : static Block");
```

```
}
```

```
    public Test() {
```

```
        System.out.println("Test : o-param constructor");
```

```
}
```

```
}
```

```
class Demo
```

```
{
```

```
    static {
```

```
        System.out.println("Demo : static block");
```

```
}
```

```
    public Demo() {
```

```
        System.out.println("Demo : o-param constructor");
```

```
} }
```

```

public class DemoApp
{
    static {
        System.out.println("DemoApp: static block");
    }

    public static void main(String args[]) throws Exception {
        System.out.println("Demo App : main()");
        Test t1 = new Test();
        Test t2 = new Test();
        Class.forName("Demo");
    } // main
} // class

```

```

> javac DemoApp.java
> java DemoApp
    DemoApp: static block
    DemoApp: main()
    Test: staticBlock
    Test: 0-param constructor
    Test: 0-param constructor
    Demo: static block

```

forName() is the method of `java.lang.Class` having the ability to load the class/ability interface. This method makes JVM to load given class but does not allow to create obj for loaded class.

⇒ `forName()` is the static method `java.lang.Class`

⇒ JVM attempts to load the class in the following situations

- a) Just before creating the first object.
- b) When class is given for execution
- c) When method is been called on class name.
- d) When `Class.forName()` is called
and etc ...

Assignment-

- 1) Can we execute Java app without main() .
- 2) Find out the JDK version of given .class file
- 3) When Java doesn't support pointer why does it throw NullPointer Exception.
- 4) Can we develop Java app without main()
Ans) Possible with the support of static block.

```
public class Testapp
{
    static {
        int a=10;
        int b=20;
        int c=a+b;
        System.out.println("Result:" + c);
        System.exit(0);
    }
}
```

The above code runs upto JDK 1.6 and in the initial release of JDK 1.7 But it will not run in the last release of 1.7 and also in 1.8 because main method is mandatory in java application in that release.

4
06/02/14

Understanding Class.forName() method.

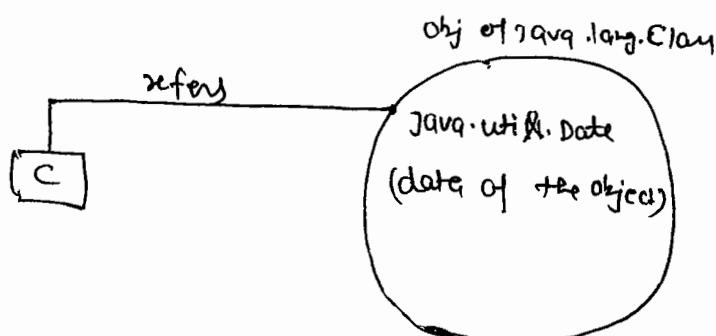
Prototype:-

public static Class forName(String name) throws ClassNotFoundException
modifiers Return type method name

- * This is static method of `java.lang.Class`
- * This method loads given class or interface into App through JVM dynamically at runtime & return the Obj of `java.lang.Class` having the loaded class/interface as data of the object.

```
Class c = Class.forName("java.util.Date");
```

Here `Class.forName()` method loads given `java.util.Date` class into App dynamically at runtime and return the Obj of `java.lang.Class` having loaded "`java.util.Date`" as data of the object.



In the above code C is not object of loaded `java.util.Date` class. It is the object of `Class` pointing to the object of `reference variable` `java.lang.Class` that contains the loaded `java.util.Date` as the data of the object.

Using this reference variable you can perform multiple operations on loaded class like object creation, knowing about methods, constructors and etc.

```

// TestApp1.java

public class TestApp1
{
    public static void main( args[] ) throws Exception
        // load class / interface
        Class c = Class.forName( args[0] );
        // get details about loaded class / interface
        System.out.println("loaded item is:" + c.getName());
        // java.util.Date
        System.out.println("Super class of loaded item is:" + c.getSuperclass());
        // java.lang.Object
        System.out.println("Loaded item is interface?" + c.isInterface());
        // false
        System.out.println("c is referring to the obj of" + c.newInstance());
        // java.lang.Class
    } // main
} // class

```

`Class.forName()` throws `java.lang.ClassNotFoundException` if given class or interface is not available to load.

The objects of `java.lang.Class` represents classes, interfaces, datatype in a running Java application.

To hold numeric values we use numeric datatype variables
 Similarly To hold string value we use `java.lang.String` class object.

So to hold class, interface or Datatype in our JNG app we can use `superclass` of object of `java.lang.Class`.

* We can't create object for `java.lang.Class` using a new keyword because `java.lang.Class` is having just one private constructor.

* There are multiple ways to create the obj of `java.lang.Class`

a) Using `Class.forName()`

```
Class c = Class.forName("java.util.Date");
```

b) Using `getClass()` method of `java.lang.Object` class

```
String s = new String("ok");
```

```
Class cl = s.getClass();
```

c) Using "class" built in property

```
Class c2 = Integer.class;
```

Note → `length`, `class` are two built-in properties of Java.

Creating Object for java class without using new keyword.

`new` keyword/operator creates the object at runtime but expects the presence of class from compile time onward.

```
Test t = new Test();
```

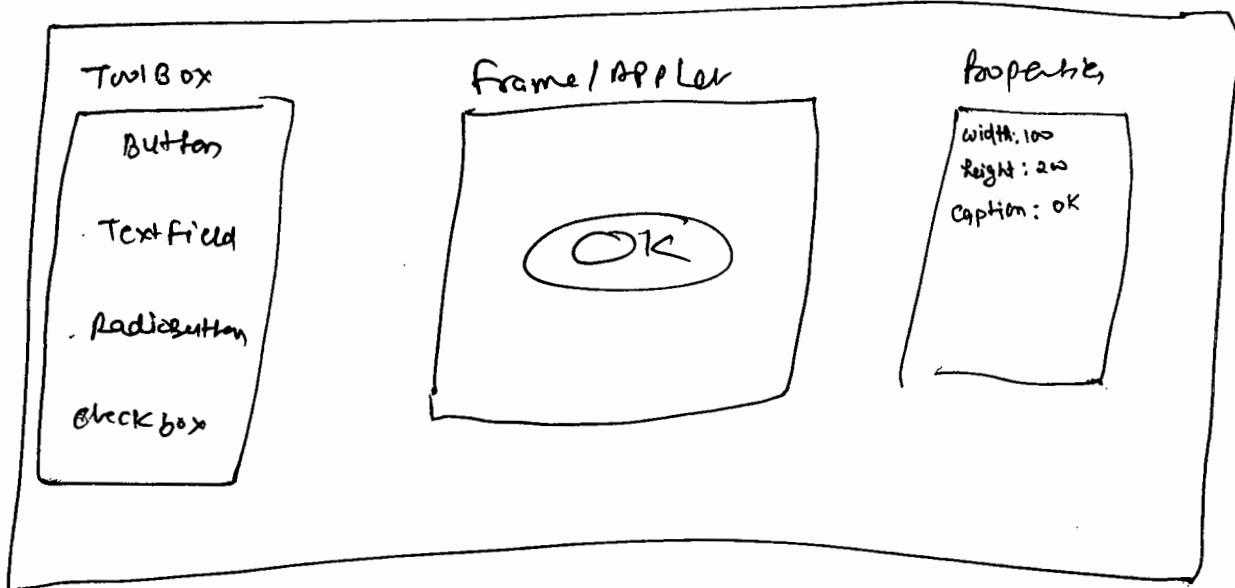
Object for `Test` class will be created at runtime but it expects the presence of `Test` class from compile time onward that means we can't use `new` keyword to create object of Java class that needs to apply dynamically at run time.

In GUI builder applying a component will be selected at run time & object for that component class should be created at run time, so `new` keyword can't

be used here.

To overcome this problem we can use `newInstance()` method of `java.lang.Class`

GUI Builders



Creating object by using `newInstance()` of `java.lang.Class` :-

// Test.java → helper class

```
public class Test
{
    int a;
    float b;
    public Test()
    {
        System.out.println ("Test: 0-param constructor");
    }
    public String toString()
    {
        return "a=" + a + " b=" + b;
    }
}
```

(5)
07/01/2015

//ObjTest.java

```
public class ObjTest  
{  
    public static void main(String[] args) throws Exception  
    {  
        //Load the class into dynamically  
        Class c = Class.forName(args[0]);  
        //Create Obj for loaded class dynamically  
        Object obj = c.newInstance();  
        //Display the details of the object  
        System.out.println("data=" + obj);  
        System.out.println("HashCode=" + obj.hashCode());  
    }  
}
```

- c.newInstance() creates the object for loaded class & returns that object.
- Since java.lang.Object is the ^{class in the} topmost hierarchy of Java class, so its reference variable can refer any Java class object.
- for any object jvm assign one unique identity value called hashCode to get that code use hashCode method.
- When we pass Object or reference variable in System.out.println statement that to String method will be

called automatically.

When `newInstance` method attempts object creation for abstract class or interface then `java.lang.InstantiationException` exception will be raised.

Q) How many ways are there to create obj for Java class?

Ans)

- a) using `new` keyword
- b) using Instance factory method
- c) using Static factory method
- d) Using Cloning process
- e) Using Deserialization process
- f) using `newInstance()` method.

g) using `new` keyword

```
Text t = new Text();
```

b) using Instancefactory method

Factory method. The method that is capable of creating and returning either its own class object or other class object is called factory method.

`Instancefactory()` method will be called through object.

`Static factory()` methods will be called without object

```
String s = new String("Hello")
```

```
String s1 = s.concat("123");
```

instance factory method it return in
own class object

`StringBuffer sb = new StringBuffer("Hello How are you");`

`String st = sb.substring(0,4);`

Instance factory method that returns
other class object.

Use instancefactory method to create new object based
on existing object methods

c) Static factory method.

Eg.1

`Class c = Class.forName("java.util.Date");`

static factory method returning its own class object.

Eg.2

`Console con = System.console();`

static factory method returning other class object.

Note →

If class is having only private constructor & want to
create object of the class outside of the class then we
can use static factory method.

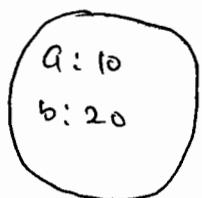
d) using cloning process →

The process of creating new object by having existing object
data as initial data of the object is called cloning.

In this process constructor will not be executed because
there is no need of initializing cloned object separately.

To perform clone we use `clone()` method of `java.lang.Object` class.

`Telt t = new Telt();`
[original object]



hashcode: 33566647
address: 2434563

`Telt t1 = (Telt)t.clone();`
[cloned object]



hashcode: 343635676
address: 5343564

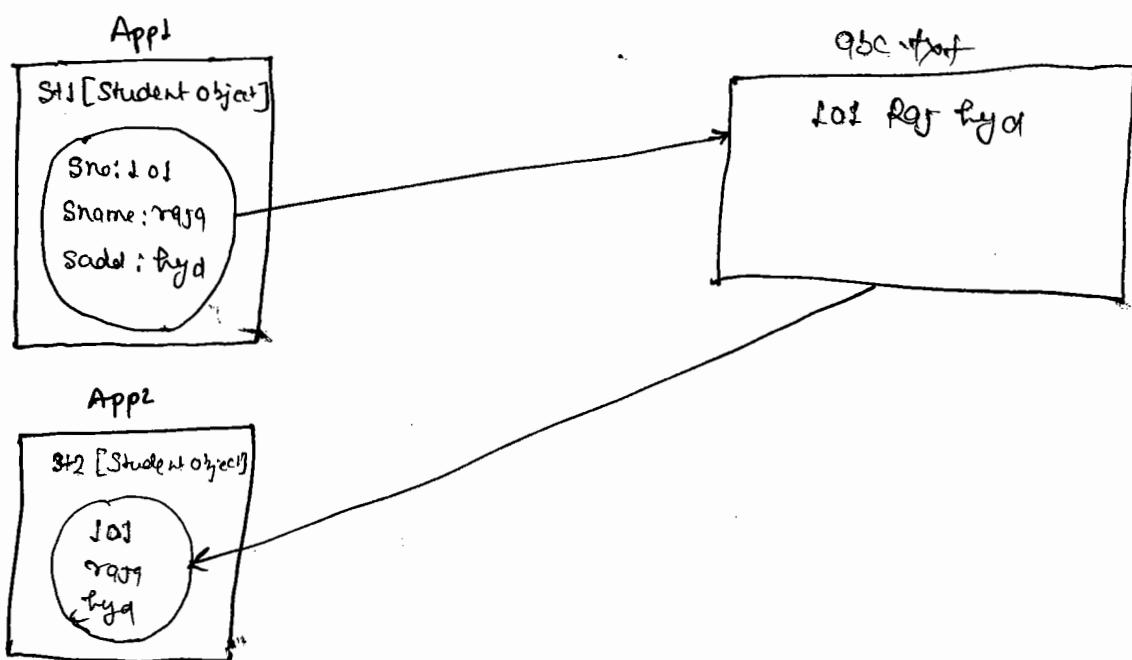
Write operation ← Serialization
Read operation ← Deserialization

Serialization.

The process of capturing data and writing that data to file is called Serialization.

Deserialization.

Reading data from file and creating object having that data is called Deserialization.



when object is created through de-serialization constructor will not be executed.

f) using newInstance() method of java.lang.Class

```
Class c = Class.forName("java.util.Date");
```

```
Object obj = c.newInstance();
```

When Java doesn't support pointers then why does it throw NullPointerException.

When we invoke method on a reference variable that points to null then NullPointerException will be raised.

```
Date d = null;
```

```
d.getDay(); throws NullPointerException.
```

Solution:-

```
Date d = null;
```

```
d = new Date();
```

```
d.getDay();
```

The word Pointer in NullPointerException is no way related to C, C++ pointers. It is used purely by having dictionary meaning which indicates that the method is called on reference variable that points to null value.

+---+
| 6 |
+---+
19/01/15

Terminologies:

Persistence → saving & managing data permanently.

Persistence store

Persistent data

Persistence operations

Persistence logic

Persistence technology

Persistence - The process of saving and managing data long time (permanently) is called persistence.

The data stored in local, global variable and object, allocate memory in RAM area which is a temporary memory so we can't preserve data after the execution of the application.

In order to overcome this problem we need to make our app writing data to persistence store like files, database software.

Persistence Store →

The place where data will be saved here for long time permanently.

e.g. file, DB SW

Persistent Data

The Data of Persistence store

Persistence operation

insert, update, delete & select operations, operation performed on persistent data.

These are also called as CURD / CRUD, SCUD operation

Persistence Logic

The logic that is required to perform persistence operations. e.g. I/O Stream code, JDBC code

Persistence technology.

The technology that can be used to develop persistence logic

CURD/ CRUD --

C → Create
U → Update
R → Read
D → Delete

SCUD

S → Select
C → Create (Insert)
U → Update
D → Delete

Note: Persistence

Persistence logic performing persistence operation on persistence data of persistence store

Every basic application contains the following minimum logic

a) Presentation logic

→ The logic that gives user interface to end user to supply inputs and to view results.

b) Business / Service logic

The main logic that generates results.

c) Persistence logic

The logic that performs persistence operation CURD / CRUD / SCUD operation.

Ex

Read student details, marks from end user

Presentation logic

Calculate total, avg

$$\text{total} = \text{mark1} + \text{mark2} + \text{mark3}$$

$$\text{avg} = \text{total} / 3.0f$$

(B-logic / Service logic)

Generate rank for student

(B-logic)

Write student details, marks, total, avg, result to
Db table

(Persistence logic)

Display results for end user

(Presentation logic)

* File, DB Sl/w allocate memory in secondary memory
device like hard disk and etc. Due to this
their data remains permanent.

Java App - - - - - I/O Stream - - - - - \rightarrow files
(java.io pkg)

Java App - - - - - jdbc - - - - - \rightarrow Db Sl/w
(java.sql, javax.sql...) (oracle, sybase, mysql,
postgresql ~)

Limitations with file as persistence store.

- * no security.
- * Can't manage huge amount of data.
- * It does not allow to apply constraint (like Primary key, ~~foreign key~~)
- * Can't keep relationship b/w data, file
- * Retrieving data with multiple condition is very complex.
- * Comparing and merging data is very complex.
- * Doesn't enable relationships (like foreign key)
- * No query support.

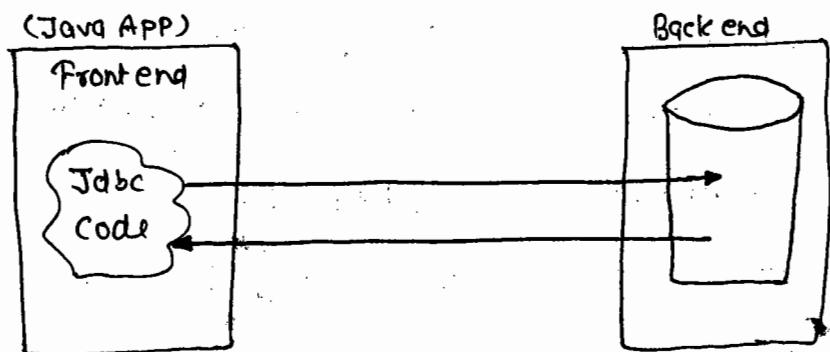
To overcome all these problems huge DB is/w as persistence store

While developing small scale applications like mobile games we file as persistence store because they allocate less memory and they can be created dynamically.

We D/b is/w as persistence store in medium, large scale appn.

e.g. Banking Projects, websites and etc.

Persistence stores are generally called backend b'coz even though bc they are part of project they are not visible to end user but response for instructions given by frontend.



- ⇒ what you can see in the app is called front app
- ⇒ what you can't see in the app but part of app is called backend.

* //

Q) what is API?

Any API (Application Programming Interface).

- * No way related with Java Interface used with dictionary meaning base or platform.
- * API is the base for programmer to develop new app.
- * In C language
 - { API means set of function which comes in the form of header file.
- * In C++ API set of functions and classes in the form .h files.
- * In Java API means classes, interface, enum, which come in the form of package.

(7)
20/01/15

Example java api,

Jdbc api (java.sql, javax.sql pkgs)

awt api (java.awt, java.awt.event pkgs)

IO api (java.io pkg)

Utility api (java.util and its sub pkg)

There are 3 types of apis

a) Pre-defined api/ Built-in api

→ These are part of programming languages and SW technologies.

e.g. Jdbc api, Servlet api and etc.

b) User defined apis

Created by programmers

e.g. Creation of user defined packages.

c) Third Party apis

Given by third party vendors

e.g. print api, Java zoom api and etc.

Q1 How can we develop Java application without having any user defined class?

A1 By using enum

```

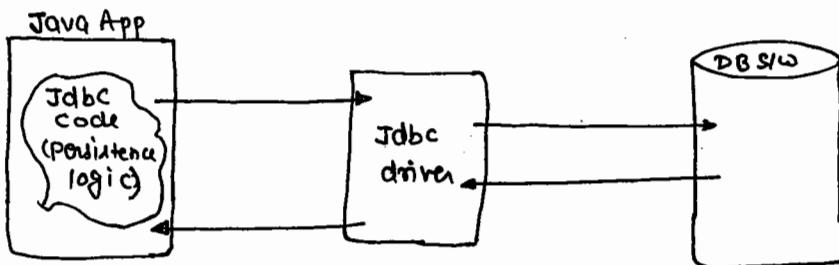
public enum MyEnum
{
    A;
    public static void main(String[] args)
    {
        System.out.println("Hello");
    }
}

//javac myEnum.java
//java myEnum

```

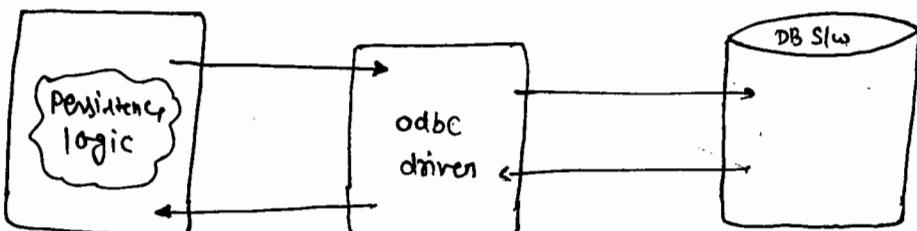
Jdbc Driver.

Jdbc driver is bridge between Java api and DB s/w. It converts Java calls to DB calls (instructions) and vice-versa.



Non Java Apps (like VB, VB.net, C# and etc) use odbc drivers to interact with DB s/w

odbc: open database connectivity



Q) When all non Java technologies are using ODBC drivers to interact with DB s/w can you tell me why do we need separate JDBC driver for Java applications.

A) ODBC drivers are developed in 'C' language taking the support of pointers. Since Java doesn't support pointers so we need separate Java based JDBC drivers to interact with DB s/w.

We can get JDBC drivers from 3 parties -

- a i) Sun Microsystems or ORACLE Corporation
- b ii) DB Vendors (Recommended)
- c iii) Third Party Vendors (not a, b)

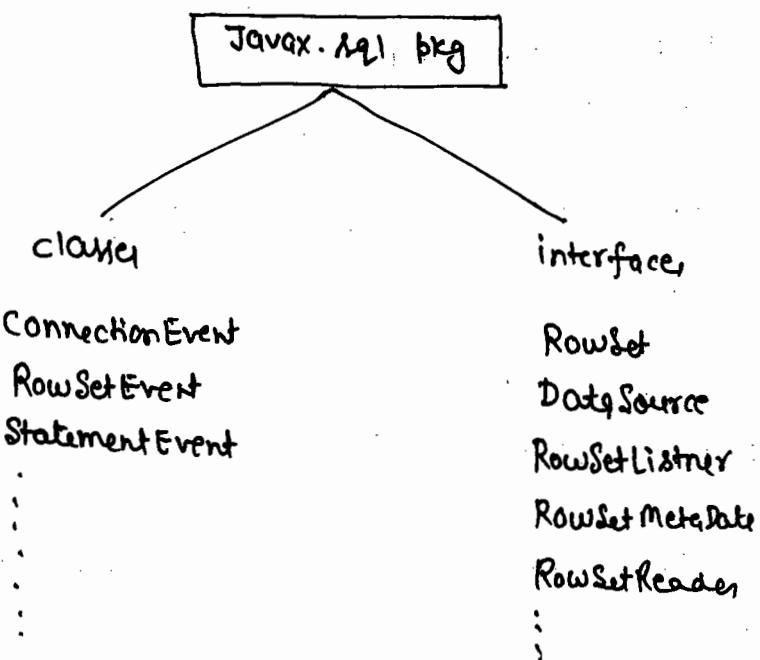
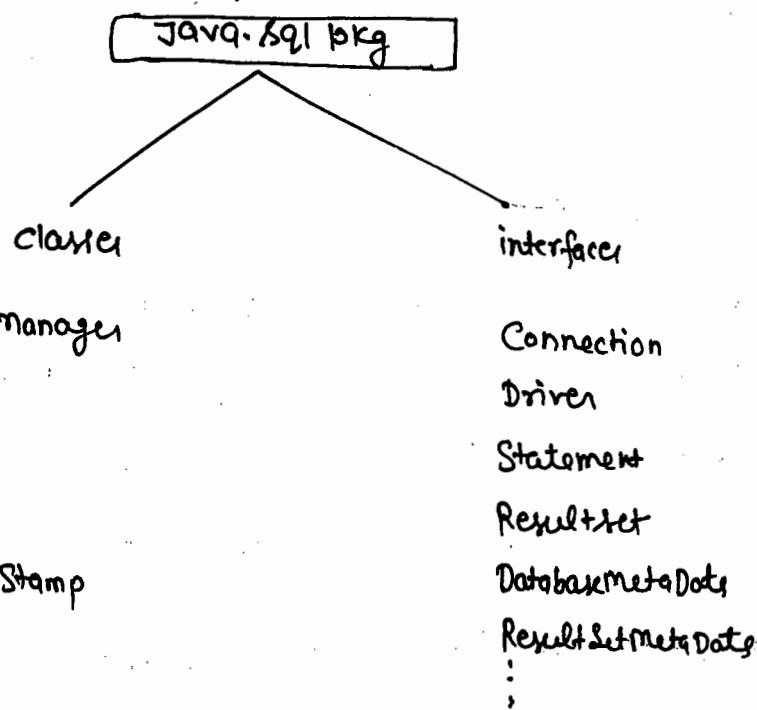
We can get ODBC drivers from 3 parties -

- a) From XOpen
- b) DB Vendors (Recommended)
- c) Third Party Vendors (not a, b)

JDBC is Java technology of JSE module in latest version 4.x.

JDBC 4.x API packages are

Java.sql, javax.sql, javax.sql.rowset, javax.sql.rowset.serial
and -----> core pkgs -----.



Note. Programmers are not responsible to develop JDBC drivers but they are responsible to use them.

⇒ vendor companies are responsible to develop JDBC drivers by implementing various interfaces of JDBC API packages.

- * JDBC technology provides API for programmers to develop JDBC persistence logic by activating JDBC driver.

Q) What is difference b/w programming language and technology?

Ans

Programming Language - It is directly installable SW having raw material having basic features to develop application.

Programming Languages defines syntax and semantics.

PL is the base for program to create O/S, technologies, frameworks, DB SW's and etc.

e.g. C, C++ and Java

Technology -

It is a SW specification that gives set of rules and guidelines (classes) to develop SW by using one or other programming language.

Technology is not installable but technology based SW's are installable.

Working with these SW is nothing but working with technologies.

There are two types of technologies -

i) open technologies

This technologies rules and guidelines are open to all vendor companies to develop SW's.

(8)
21/01/15

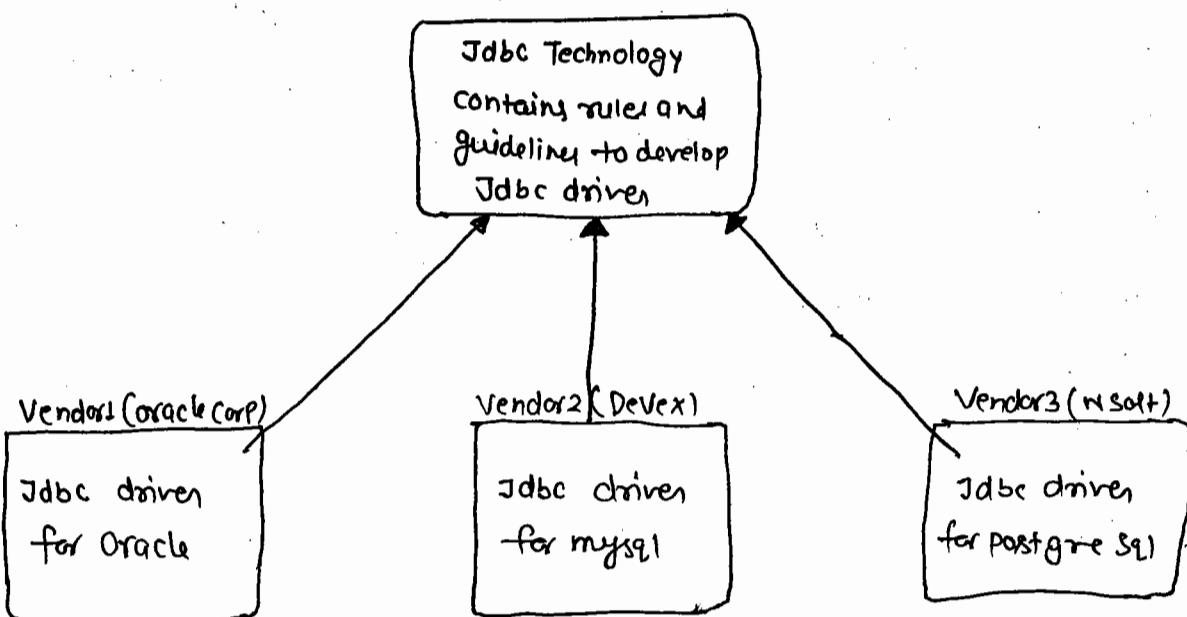
Eg. JDBC, Servlets, JSP and etc.

Proprietary Technology.

This technology rules and guidelines are specific to one vendor company and only that vendor company is allowed to develop it.

Eg. All Microsoft technologies.

- ⇒ JDBC is open technology which gives set rules and guidelines to develop JDBC drivers can be installed or arranged
- ⇒ Working with JDBC driver is nothing but working with JDBC technology.



Since All JDBC drivers are given based on the common rules and guidelines of JDBC technology the way we work with all JDBC drivers going to be much similar. That means the knowledge of working with one JDBC driver can be used to work with another JDBC driver.

Every JDBC driver is collection of classes implementing various interfaces of JDBC API having logic to interact with certain DB S/W.

What is JDBC?

It is open tech Java technology that gives rules and guidelines to develop JDBC drivers for different DB S/W.

What is ODBC?

It is an open technology that gives set of rules and guidelines to develop ODBC drivers for different DB S/W.

Every JDBC driver is identified with its driver class name. In order to use JDBC drivers in our application we must specify its driver class name.

The Java class that implements java.sql.Driver interface directly or indirectly is called JDBC driver class.

Upto JDK 1.7 we get built-in JDBC drivers and its driver class name is

Sun-JDBC-ODBC-JDBC ODBC Driver
pkg name

Driver class name that implements
java.sql.Driver interface as shown

(from <Java Home>\jre\lib\rt.jar file)

Every ODBC driver is identified with its DSN. There are 3 types of DSNs -

- a) User DSN:- Specific to currently logged in windows user.
- b) System DSN:- Visible to all the networked windows of users of a computer.
- c) File DSN:- Shareable in network.

Procedure to create User DSN for Microsoft ODBC Driver for Oracle.

- i) Make sure that Oracle 8i/w is installed.
- ii) Create DSN

C:\Windows\System32\ODBC32 →

User DSN → add

Microsoft ODBC →

DSN oradsh

Description

(optional)

Username SCOTT/System

Server

↓

Optional while working with local oracle. DNS string is required while working with remote oracle.

(9)
22/01/15

Every Java App contains one built in invisible service called `DriverManager` having the capability to manage set of JDBC Drivers.

To access this service we can use `java.sql.DriverManager` class.

To register JDBC driver with `DriverManager` service we must place JDBC driver class object with driver manager service to register JDK's built in JDBC drivers with `DriverManager` service.

In App

```
// Create JDBC driver class obj
```

```
Sun.jdbc.odbc.JdbcOdbcDriver obj
```

```
= new Sun.jdbc.odbc.JdbcOdbcDriver();
```

```
// Register driver with Driver Manager service
```

```
DriverManager.registerDriver(obj)
```

Prototype of registerDriver()

```
public static registerDriver (java.sql.Driver driver) throws SQLException
```

3 important Concept of method parameter types -

q) If method parameter type is interface

we must call that method having implementation class object as the argument. (Refer above)

- b) If method parameter type is abstract class then we must call that method having sub class object of abstract class as argument value.
- c) If method parameter type is concrete class then we must call that method having either that class or object or any subclass of that class object as an argument value.

Standard steps to develop jdbc App -

- Step1 → Register jdbc driver with DriverManager service
- Step2 → Establish the connection with DB S/w.
- Step3 → Create jdbc statement obj
- Step4 → Send and execute SQL query in DB S/w
- Step5 → Gather SQL Query results from DB S/w and process the results.
- Step6 → Close jdbc objects.

Establishing Connection means creating communication channel between Java App and DB S/w.

Statement object is like vehicle b/w Java App & DB S/w to send SQL query to DB S/w as input and to bring SQL query results back to Java App.

All jdbc objects must be closed at the end of app to avoid abnormal closing of streams and to

Avoid ideal connection with DB S/w.

Software vendor company can develop JDBC driver for develop different DB S/w by using following mechanisms or methodologies or architecture.

Type1 (JDBC-ODBC bridge driver)

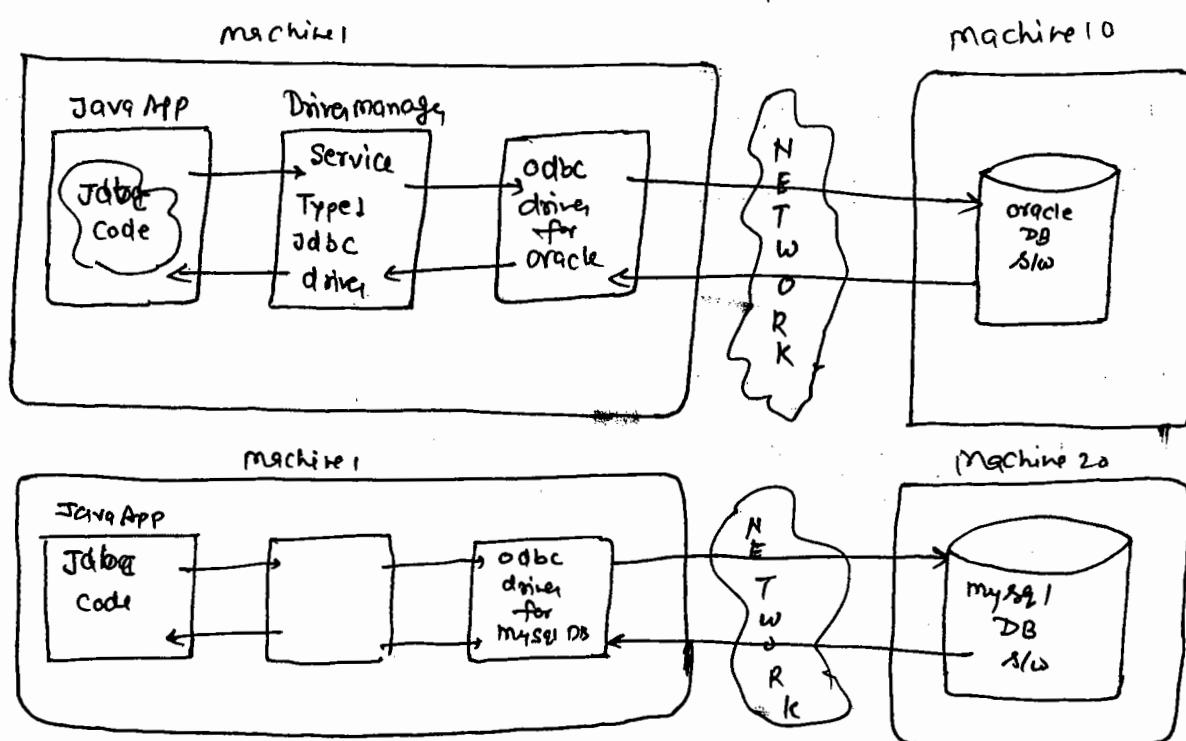
Type2 (Native API / Partly Java Driver)

Type-3 (Net Protocol / All-Java Driver)

Type-4 (Native Protocol / All-Java Driver)

Type-5 (Not given by Sunmicrosystem, so there is technical name)

Partial Architecture of Type-1 JDBC driver



Type 1 driver is not design to locate and communicate with DB S/w directly. It is designed to take the support of appropriate ODBC driver to interact with DB S/w.

The built in JDBC driver of JDK S/w is type-1 mechanism based JDBC driver.

Type 1 driver is not specific to any DB S/w. One type-1 JDBC driver can use multiple ODBC drivers to interact with multiple DB S/w's.

Write an JDBC App^n that uses type1 JDBC driver to establish connection b/w Java App and Oracle DB S/w.

S/w Setup

- * JDK any version (JDK 1.7)
- * Oracle any version (Oracle 11g)
- * type1 JDBC driver (Built in driver of JDK)
- * "DSN" created for "ODBC Driver for Oracle" (oradsn) (refer previous class)

```
// ConnTest.java

import java.sql.*;
public class ConnTest
{
    public static void main (String[] args) throws Exception
    {
        // Register JDBC driver (type1) with DriverManager service
        // Create JDBC driver class obj

        Sun.jdbc.odbc.JdbcOdbcDriver obj =
            new Sun.jdbc.odbc.JdbcOdbcDriver();
        // register driver

        DriverManager.registerDriver(obj);
        // Establish connection with Db & w

        Connection con = DriverManager.getConnection("jdbc:odbc:oradsh",
                                                    "Scott", "tiger");
                                                    DB user           DB password
        if (con == null)
            System.out.println("Connection is not established");
        else
            System.out.println("Connection is established");
    }

    // Exception in thread "main" java.sql.SQLException: Microsoft [ODBC Driver Manager]. The specified DSN contains an architecture mismatch between the Driver and application
    // at sun.jdbc.odbc.JdbcOdbc.createSQLException(JdbcOdbc.java, 6964)
}
```

(10)

28/01/15

Syntax of JDBC URL:

<main protocol> : <sub protocol> : <sub name>

"JDBC" is main protocol

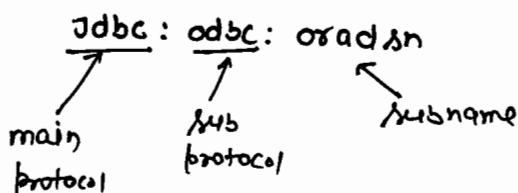
Sub protocol and sub name will change based on JDBC drivers we use

⇒ For types driver

"Sub protocol" is: odbc

"Sub name is" : dsn created for odbc drivers (oradsn)

e.g.



Protocol is a set of rules followed by two parties who want to participate in communication. There are two types of protocol—

a) Network level Protocol

→ Defines rules to get interaction b/w two computers.

e.g. TCP/IP

b) Application Level Protocol

Defines rules to get interaction b/w two softwares / software app.

e.g.

JDBC:odbc, http

`DriverManager.getConnection(, ,)` of App

- makes DriverManager service to use "Jdbc:odbc" as the application level protocol to interact with DB s/w by using the registered types driver and the specified dsn based odbc driver.
- This method jdbc con obj representing the connectivity b/w Java app and DB s/w.
- When `java.sql.Connection` is an interface how can we say `DriverManager.getConnection()` returns jdbc connection object

Ans

Jdbc Connection object means it is not the object of `java.sql.Connection` interface.

It is object of underlying jdbc driver supplied java class that implements `java.sql.Connection` interface and this class name will change driver to driver.

```
SopIn("Connection obj class name:" + con.getClass());
```

In Types driver connection obj class name is

`sun.jdbc.odbc.JdbcOdbcConnection`

`DriverManager.getConnection()` returns the implementation class object of `java.sql.Connection` interface to reference variable of `java.sql.Connection` interface (con) can refer that object.

Prototype of DriverManager.getConnection (-,-,-)

```
public static Connection getConnection  
( String url, String user, String pwd)  
throws SQLException
```

getConnection() return type is java.sql.Connection interface that means this method returns one implementation class object of java.sql.Connection interface.

3 important statements on method return type -

- * If method return type is interface then that method returns one implementation class object of that interface
- * If method return type is concrete class then that method returns either that concrete class object or one of its sub class object.
- * If method return type is abstract class then that method returns one sub class object of that abstract class.

We can also register JDBC driver with DriverManager service by loading JdbcDriver class.

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

`Class.forName()` just load given class then how can we say the above `Class.forName()` is registering driver.

`Class.forName()` loads the class

→ due to this static block of driver execute

→ The static block contains logic to create jdbc Driver class object and logic to register that object with DriverManager service.

The static block of "sun.jdbc.odbc.JdbcOdbcDriver" looks like

```
Static {
```

```
try {
```

```
sun.jdbc.odbc.JdbcOdbcDriver obj1 =
```

```
new sun.jdbc.odbc.JdbcOdbcDriver();
```

```
DriverManager.registerDriver(obj1);
```

```
}
```

```
Catch(Exception e) {
```

```
...:
```

```
}
```

```
}
```

Can you tell me all the ways that are there to register jdbc driver with DriverManager service.

Can you explain different ways to register jdbc driver with DriverManager service.

Q. What are different ways to gathering I/P from end user.

- A) Using cmd like line args
- B) Using System properties
- C) Using java.util.Scanner class (from jdk 1.5)
- D) Using java.io.Console (from jdk 1.5)
- E) Using Stream

// ReadInputs.java

```
import java.util.*;
import java.io.*;
public class ReadInputs
{
    public static void main(String[] args)
    {
        // Using cmd line args
        String s1 = args[0];
        // Using System property
        String s2 = System.getProperty("my.name");
        // Using Scanner class
        System.out.println("Enter name");
        Scanner sc = new Scanner(System.in);
        String s3 = sc.nextLine();
    }
}
```

// Using Console class

```
Console con = System.console();
```

```
Scanner s1 = new Scanner(System.in);
```

```
String s4 = con.readLine();
```

// Using Stream

```
Scanner s1 = new Scanner(System.in);
```

```
BufferedReader br =
```

```
new BufferedReader(new InputStreamReader(System.in));
```

```
String s5 = br.readLine();
```

```
System.out.println(s1 + " " + s2 + " " + s3 + " " + s4 + " " + s5 + " ");
```

}

//> javac ReadInput.java

//> java -Dmyname=sqrq2 ReadInput sqrq1

Enter name

user

defined

system

property

Command
line argument

Parsing command line argument is mandatory

When it is specified in the application.

Parsing System property is optional when it is
specified in the app

(17)
24/01/15

Q) What are the different approaches that can be used to register JDBC driver with DriverManager service.

Approach 1)

```
Sun.jdbc.odbc.JdbcOdbcDriver obj1  
= new Sun.jdbc.odbc.JdbcOdbcDriver();
```

```
DriverManager.registerDriver(obj1);
```

Approach 2) Note - Before creating object for driver class JVM loads due to this static block of driver class executes and driver will be registered with driver manager service automatically.

The above code registers name JdbcDriver twice with DriverManager service. So it's not recommended approach.

- 1) Because of static block of driver class.
- 2) Because of explicit call to DriverManager.registerDriver,

Approach 2

```
DriverManager.registerDriver(new Sun.jdbc.odbc.JdbcOdbcDriver)
```

It is similar to approach 1 so not recommended.

Approach 3

```
Sun.jdbc.odbc.JdbcOdbcDriver = new Sun.jdbc.odbc.JdbcOdbcDriver();
```

Here JdbcDriver will be registered with DriverManager service only for one time with the support of static block but

explicitly created object of Driver class will be wasted.
So it is not recommended.

Approach 4)

```
new Sun.jdbc.odbc.JdbcOdbcDriver();
```

similar to approach 3 so it is not recommended to use.

Approach 5)

```
public class ConnTest extends Sun.jdbc.odbc.JdbcOdbcDriver
```

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

```
    {  
        Connection con = DriverManager.getConnection("jdbc:odbc:DB1", "sa", "sa");  
    }  
}
```

while loading given java class the jvm also loads remaining class of inheritance hierarchy due to this static block of given class & inheritance hierarchy class will be executed automatically

The above approach registers jdbc driver only for one time using static block but it is not recommended approach because it doesn't allow our app class to extend from other classes like frame, Thread, Applet and etc. because java doesn't support multiple inheritance

Approach 6)

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

Here the jdbc driver will be registered with DriverManager service only for one time through static block & no explicit object will be created for driver class. So

is a recommended approach to use.

Approach 7

```
public class ConnTest {
    public void main() throws Exception {
        Class.forName(args[0]);
        Connection conn = DriverManager.getConnection("-", "-", "-");
    }
}

//> javac ConnTest.java
//> java ConnTest sun.jdbc.driver.sun.jdbc.odbc.JdbcOdbcDriver
args[0] command line argument.
```

This is similar to approach no. 6 but not recommended because there is no guarantee that every application runs from command prompt.

e.g. Java based web application.

Approach 8:-

```
public class ConnTest {
    public void m() throws Exception {
        String s1 = System.getProperty("my driver");
        Class.forName(s1);
        Connection conn = DriverManager.getConnection("-", "-", "-");
    }
}

//javac ConnTest.java
//> java -D my driver= sun.jdbc.driver.sun.jdbc.JdbcOdbcDriver ConnTest
```

similar to approach no 7.

Approach 8:-

```
public class ConnTest {
```

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

```
{
```

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

```
}
```

As part of its initialization the driver manager class automatically loads driver classes that are referenced in fixed jdbc.driver.fixed property.

Approach 9:-

Since Types driver is built in driver of jdk so there is no necessity of registering the driver with DriverManager service explicitly.

```
public class ConnTest {
```

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

```
{
```

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

```
}
```

```
}
```

```
// javac ConnTest.java
```

```
// java ConnTest
```

Conclusion - Approach 6 is like real time standard.

The API of jdbc technology will be used by vendor companies as rules & guidelines to develop jdbc drivers. and the same API will be used by programmers to develop Jdbc persistence logic.

Jdbc Statement objects acts as vehicle b/w Java App & DB SW to send inputs SQL queries and to gather results.

To Create Jdbc Statement Obj

Statement st = con.createStatement();

JdbcStatement obj means it is the obj of underlying jdbc driver supplied java class that implements java.sql.Statement().

In types jdbc driver this class name is:

sun.jdbc.odbc.JdbcOdbcStatement

Note-

According to Oracle specification there are 4 types of SQL query -

- DQL Queries (Select Query)
- DML Queries (insert, update, delete query)
- DDL Queries (create, drop, alter)
- DCL Queries (commit, rollback, savepoint)

According to jdbc programming there are two types of SQL query -

- a) Select Queries (DQL Queries)
- b) Non-Select Queries (DML, DDL, TCL Queries)

Note → When select queries are executed we get bunch of records

When non-select query is executed we get numeric value representing the no. of records that are effected.

We use executeQuery(-) on Statement obj to send and execute Select SQL query in DB Alw.

We use executeUpdate(-) on Statement obj to send and execute non-select query in DB Alw

- Q) When we are able to execute SQL query directly in the SQL prompt of DB Alw then what is the need of doing the same operation from java application through jdbc code

Ans End users like bank employee, customers are non-technical people so they can't execute SQL query directly in DB Alw. But they expect one front end appn to operate in order to perform operations on DB to develop this front end appn we take the support of jdbc code.

12
25/01/15

If Architecture mismatch problem comes while work with types
choose open 64 bit command prompt to compile & execute
the app.

use C:\windows\syswow64\cmd.exe file

Write a JDBC Appn that receive and displays records of
Students table by using types JDBC Driver.

S/W Setup

```
create table student ( sno number(10) Primarykey, lname varchar2(15),  
sadd varchar2(15));
```

```
insert into values (101, 'ranga', 'hyd');
```

```
insert into values (102, 'ramesh', 'vizag');
```

//Select Test.java

```
#import java.sql.*;  
  
public class SelectTest  
{  
    public static void main (String[] args) throws Exception {  
        //register jdbc driver  
        Class.forName ("sun.jdbc.odbc.JdbcOdbcDriver");  
        //Establish the connection  
        Connection con = DriverManager.getConnection  
            ("jdbc:odbc:oradyn", "Scott", "tiger");  
        //create Statement object  
        Statement st = con.createStatement();
```

// send and execute SQL Query in Db b/w

ResultSet rs = st.executeQuery("Select * from student");

/* How ResultSet look like.

BFR (Java obj)		
101	raja	Tiru
(1)	(2)	(3)
434	ramu	Vizag
(1)	(2)	(3)
ALR		

*/

// Process ResultSet

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

```
    System.out.println(rs.getInt(1) + " " + rs.getString(2) + " " + rs.getString(3))  
} // System.out.println(rs.getInt(sno) + " " + rs.getString(sname) + " " + rs.getString(sadd));
```

// close jdbc object

```
rs.close();
```

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

```
con.close();
```

```
} // main
```

```
} // class
```

//> javac SelectTest.java

//> java SelectTest

ResultSet Object is java object that holds bunch of record given by DB now after executing Select query

ResultSet object means it is object of underlying jdbcObject supplying that implements java.sql.ResultSet interface.

- * Every ResultSet object contains one BFR, ALR position by default the cursor resides in BFR position

BFR = Before first Record

ALR = After last Record

- * The records in ResultSet and record values in ResultSet contains one based index
- * `obj.next()` moves cursor in ResultSet next position from current position.
 - If that next position is record & this method returns true.
 - And If next position is ALR then method returns false.
- * To receive column values from record of ResultSet we can use `getXXX(-)` methods like `getInt`, `getString`, `getFloat` and etc. either having column index or column name.
- * Always close jdbc objects and at the end of their utilization in the reverse order of their opening.

- * `rs.close()` closes Resultset object that means we can't process the resultset.
- * `st.close()` closes the Statement object that means we can't send further queries to DB s/w for execution.
- * `con.close()` closes the connectivity b/w java Application and DB s/w.
- * JDBC app can't read uncommitted data from DBtable.
- * While working with JDBC app the class name of JDBC because their class names will change based on JDBC driver we use. But we refer these objects through common JDBC API interface reference `obj` variable.
 - * This allows us to develop JDBC code as same code for all JDBC drivers.
 - * Interface e.g. → JDBC code as driver to driver
 - polymorphism e.g. → JDBC code as driver to driver
 - Not multiple inheritance

(13)
27th 2015

Write a JDBC appn to gather student details based on the given start, end of range of student no.

SQL: Select * from student where sno >= 100 and sno <= 300



//SelectTest.java

import java.util.*;

import java.sql.*;

public class SelectTest

{

 p. n. v. m(-) throws Exception
{

//read i/p

 Scanner sc = new Scanner(System.in);

//System.in is inbuilt stream pointing to keyboard

 S.o.p ("Enter Start range of student no - stt");

 int start = sc.nextInt();

 S.o.p ("Enter End Range ---");

 int end = sc.nextInt();

//register JDBC Driver

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

Connection con = DriverManager.getConnection

 ("jdbc:odbc:oradln:", "scott", "tiger");

//create Statement object

Statement st = con.createStatement ();

Start

```
//prepare SQL query  
// Select * from student where sno >= 100 and sno <= 300  
String qry = "Select * from student where sno >=  
" + start + " and sno <= " + end;  
S.o.p (qry);  
  
// send & execute SQL query in DB & I/O  
ResultSet rs = st.executeQuery(qry);  
// process the rs  
int Cnt = 0;  
while (rs.next() != false) // while (rs.next()) { -- }  
{  
    Cnt++;  
    Sop (rs.getInt(1) + " " +  
        rs.getString(2) + " " +  
        rs.getString(3)  
    );  
}  
if (Cnt == 0)  
    Sop ("No Record Found");  
  
// close JDBC object  
rs.close();  
st.close();  
con.close();  
}
```

Write a JDBC app to get employee-no, name, salary for job-details based on given designation.

```
SQL> Select empno, ename, sal from emp  
      where job = 'CLERK';
```

//Table name , col-name are not case sensitive but
// col-values are case sensitive in DB Server

/* Always Develop Java App by keeping non-technical
end users in mind , no convert i/p values as
required for SQL Query Explicitly

01/02/15 \Rightarrow 18

Q. what is difference between Path & classPath?

Ans Path-

In order to execute .exe, .cmd, .bat files of certain directory from any location our computer add the address path of those files to PATH env... variable.

PATH is DOS command and it can be used while working with any technology.

E:\Demos\

|-----> run.bat

run.bat

date

time

ver

dir

e:\Demos>run (executes)

e:\ run (Error)

d:\text>run (Error)

Error: run is not internal or external command.

Note - Batch file (.bat) allows to combine set of commands into single unit.

Solution 1

Copy run.bat file to that directory where you want to execute the appn. This uses more memory.

Solution 2

Add the address location of run.bat file (E:\Demos) to PATH env... variable.

my computer → properties → advance system setting → env variable
→ System/ user variables →

variable name: Path

value : e:\ Demos; <other existing value>
→ OK(3).

C:\>run (execute)
e:\ Demos > run (execute)
d:\ test > run (execute)

Note → All java tools are .exe files like javac.exe, java.exe etc. belonging to the <java-home>\bin directory

In order to use them tools from any location our computer place that "bin" directory to PATH env.- variable

CLASSPATH-

If java Appn uses new API (other than jdk API)
then new API related directory or jar file must be
add to CLASSPATH environment variable to make
java tools recognizing new API

CLASSPATH is very much specific to java env

E:\ Demos

```
+-> Greetings.java  
|---> com  
|----> Greeting.class
```

```
// Greetings.java  
package com.nt;  
public class Greetings {  
    public String sayHello() {  
        return "Good morning";  
    }  
}
```

E:\ Demo\ s1 > javac -d Greetings.java
D:\ Test1
|→ mainApp.java

indicate current directory

// MainApp.java

```
import com.nt.Greetings;  
public class MainApp {  
    public static void main(String[] args) {  
        Greetings gs = new Greetings();  
        System.out.println(gs.sayHello());  
    }  
}
```

D:\ Test1 > java MainApp.java (error)

error: Can't resolve symbols com.nt.Greetings sayHello;

Solution. Add the address location of com.nt pkg (E:\ Demo\ s1) to classpath env... variable to make java tools recognizing com.nt pkg.

mycomputer → properties → adv sys settings → new env variable
→ user / system variable →

variable name: CLASSPATH
value : E:\demo1;.
OK(3)

d:\tut1>javac MainApp.java (success)

d:\tut1>java MainApp (success)

To use java tools in our computer work with Path environment variable recognizing new API or 3rd party API work with CLASSPATH

JAR file.

It is java level zip file which can represent API's JDBC drivers EJB components and web app's and etc.

To create and use Java JAR (Java Archive) file we can use jdk supplied JAR tool.

E:\demo1>jar cf NtAPI.jar .

c → Create jar file

f → Specify jar file name

When new API is there in JAR file then we can add that jar file to classpath or ~~<java-home>\ext\~~
<java-home>\jre\lib\ext folder

Important points about environment variable.

- 1) These are not case sensitive.
- 2) If multiple values added to env.. variables can be separated with ";" symbol.
- 3) Don't set values env.. variables from cmd prompt because they remain temporary.
- 4) Set values to env.. variables from my computer properties.
- 5) User env.. variables are specific to currently logged in windows user, system env.. variables visible to all the window users.
- 6) New values added to env.. variables will be visible only to the new command prompt.
- 7) Always try to add new values in env.. variables at beginning of existing values.

Q. What is the difference b/w classpath and ext folder?

CLASSPATH

1. Its values are visible to multiple jdk installed in the computer

2. Allowed values are directories, JAR file.

3. These values will be used by system class loader.

EXT

Its values are specific to one jdk.

Allows only JAR file

These values will be used by classpath extension class loading.

④ New values will not
be visible in already
opened command
prompt

visible

The Oracle Corporation supplied type 2 mechanism based JDBC driver for Oracle is called "oci driver" (oracle call interface)

The Oracle Corp supplied type 4 mechanism based JDBC driver for my Oracle is called as "oracle thin driver".

Both these drivers come along with oracle SQL installation in the form of jar files.

like (ojdbc6.jar , ojdbc14.jar)
(oracle11g) (oracle10g)

Oracle thin driver details-

driver class name: oracle.jdbc.driver.OracleDriver
(or)

oracle.jdbc.OracleDriver

jdbc url: jdbc:oracle:thin:@<host>:<port no>:<sid>
Protocol Sub name logical DB name

Jar file: ojdbc14.jar (Oracle 10g)

ojdbc6.jar (Oracle 10g)

All S/W installed computer will reside on different S/W port and every port is identified with its port no.

In windows O/S total 65536 S/W ports are available in that 1 to 1024 ports are reserved for O/S services. All externally S/W will use remaining S/W ports.

To represent computer being from that computer use "localhost".

Example App working with Oracle thin driver

- 1) Getter Oracle driver details
- 2) In Any jdbcApp write following instead of *1 code.

```
Class.forName("oracle.jdbc.driver.OracleDriver");
Connection con = DriverManager.getConnection
("jdbc:oracle:thin:@localhost:1521:xe",
"Scott", "tiger");
```

```
Statement st = con.createStatement()
```

- 3) Add ojdbc6.jar file to class path.

Variable name: CLASSPATH

VALUE : C:\oracle\ex&app\oracle\product\11.2.0\server\jdbc\lib
ojdbc6.jar; <other value>;
→ OK(3)

4) Compile and execute the application.

Oci Driver Details (Type2 mechanism based driver)

Type1, Type2 drivers are called thick drivers because we place more things at client side (odbc drivers, vendorDB libraries etc.)

Type4 drivers are called thin drivers because no vendorDB library and odbc drivers are required.

The jar files added to classpath are not visible to the projects of ide for this we need to add jar files separately to the projects of IDE.

Procedure in Eclipse IDE-

Right click on project.

Build path → Configure Build Path → Libraries tab → Add external jar

Don't expose the class names of jdbc objects directly in the application because their class name will change driver to driver. To avoid this refer all jdbc object through reference variable of common jdbc API interfaces.

(19)
02/02/15

Oci driver details (oracle corp supplied type2 mechanism based jdbc driver for oracle)

driver class name:	Oracle.jdbc.driver.OracleDriver
jdbc url:	jdbc:oracle: ^{oci8:} @<sid>
Jar file:	Ojdbc6.jar (oracle11g) odbc14.jar (oracle10g)

^{oci8:} may be XE, ORCL,

Example App

Step1 → keep Oracle thin App ready (previous class)

Step2 → place the following jdbc url in DriverManager.getConnection(...)

DriverManager.getConnection("jdbc:oracle:oci8:@xe", "scott", "tiger");

Protocol ↴ Subname ↴ Sid
 OCI8 | XE | SCOTT

While working with Oracle Corp. Supplied jdbc driver, Oracle thin driver will be activated based on "thin" word of jdbc url.

Similarly Oracle OCI driver will be activated based on "oci8" word of jdbc url.

Login App Development (verifying username and password).

DB table in Oracle

SQL> Select * from userlist

Uname	Pwd
r9ja	roni
king	Kingdom
raunak	hyd

SQL Query.

Select count(*) from userlist where uname = 'raja' and
pwd = 'rani'

count(*) : 1 (valid credential)

Select count(*) from userlist where uname = 'raja' and
pwd = 'rani'

count(*) : 0 (invalid credential)

//LoginApp.java

package com.nt;

public class LoginApp {

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

 //Read inputs

 Scanner sc = new Scanner(System.in);

 System.out.println("Enter username");

 String user = sc.nextLine(); //give raja

 System.out.println("Enter Password");

 String pass = sc.nextLine(); //give rani

 //Convert inputs as required for SQL query

 user = "'" + user + "'"; //give raja

 pass = "'" + pass + "'"; //give rani

```

//register jdbc driver
Class.forName("oracle.jdbc.driver.OracleDriver");
//Establish Connection
Connection con = DriverManager.getConnection ("jdbc:oracle:thin:@localhost:1521
                                             :ORCL", "scott", "tiger");
//Create jdbc Statement
Statement st= con.createStatement();

```

//Prepare SQL query

//Select count(*) from userlist where uname='raj' and pwd='raj'

String qry = "Select count(*) from userlist where uname ='" + uname +
 "'
and pwd ='" + pass + "';

System.out.println (qry);

//Send and execute SQLQuery in DB also

ResultSet rs= st.executeQuery (qry);

//Process the Resultset

int cnt = 0;

if (rs.next())

cnt = rs.getInt (1);

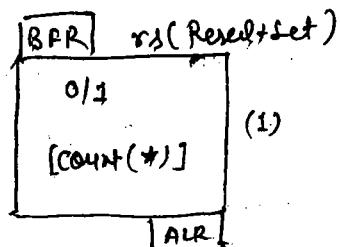
if (cnt == 0)

System.out.println ("Invalid Credential");

else System.out.println ("Valid Credential");

//Close objects

} rs.close(); st.close(); con.close();



Q) What is SQL injection problem?

The process of adding SQL instruction to query (like -- (comment)) and changing the behaviour of query dynamically at run time is called SQL injection problem.

Hackers use this technique to get into Accounts with wrong password

with respect to the above code

Username : raja' --

Password : rao (wrong password)

Result : Valid Credentials (SQL injection Problem)

There are 3 jdbc Statement objects

1) Simple Statement Object

It is the object of underlying jdbc driver supplied java class that implements `java.sql.Statement()`.

2) PreparedStatement obj.

It is the obj of underlying jdbc driver supplied java class that implements `java.sql.PreparedStatement()`.

3) CallableStatement obj.

It is the obj of underlying jdbc driver supplied java class that implements `java.sql.CallableStatement()`.

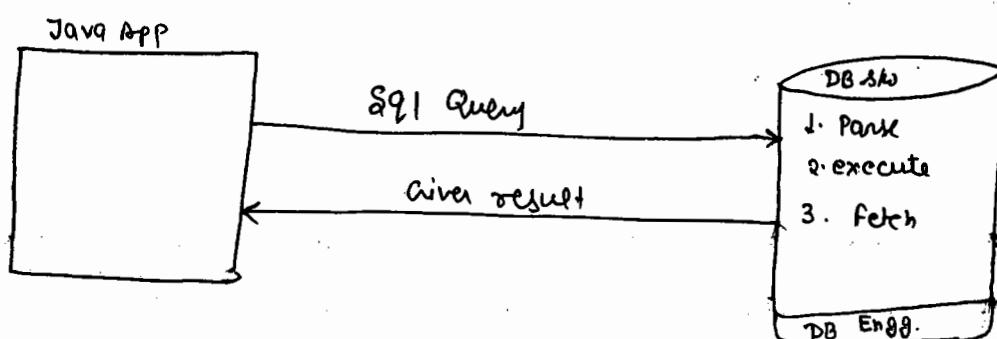
Limitations of Simple Statement.

- * May raise SQL injection Problem.
- * Framing Query with variable is complex process.
- * Not suitable to execute same SQL Query for multiple times either with same or different values.
- * Can't be use to insert diff pattern Date value.
- * Can't be use to insert Large Object (LOB)
- * Gives slow performance

26
+-----+
| 03/02/15 |

When Client App sends SQL Query to DB &/w. the DB engg of DB &/w perform 3 operation.

- 1) Parse
- 2) Execute
- 3) Fetch



Parse operation: Spills the query into tokens and verifies the syntax of query
execute operation: Executes the parsed query

Fetch operation: The result Query execution results will be given to client App.

- * Simple Statement object sends raw SQL query to DB SW for execution
- * When simple statement object is used to execute same SQL query for multiple times.
 - Sample query goes to DB SW for multiple times from java
 $(1,00,000 * 0.1 \text{ sec} = 10,000 \text{ sec.})$
 - Same query will be parsed in DB SW for multiple times
 $(1,00,000 * 0.1 \text{ sec} = 10,000 \text{ sec.})$
 - Same query will be executed in DB SW for multiple times
 $(1,00,000 * 0.1 \text{ sec} = 10,000 \text{ sec.})$
 $(10,00,000 * 0.1 \text{ sec} = 10,000 \text{ sec})$
- * Performing a), b) operation on same query for multiple times is unnecessary and they can't be avoided while using simple statement.

To overcome the above problems of simple statement object we can use pre compiled SQL query with support of JDBC prepared statement object.

The SQL query that goes to DB SW from java app without input values and becomes parsed/compiled Query in DB SW before being executed is called Pre-compiled query.

- * JDBC Prepared Statement Obj of Java App represents pre compiled SQL Query and that object can be used to set input values to pre-compiled query to execute pre-compiled query & to gather results from pre-compiled query.
- * Prepared Statement Object is very useful to send and execute same query in DB SW for multiple times with diff values or same values.

Executing same Query in DB SW for 1,00,000 times using Prepared Statement object.

- Query goes to DB SW from Java App only for one time : ~~1,00,000~~ * 0.1 sec = 0.1 sec
- Query will be parsed in DB SW only for one time : 1 * 0.1 sec = 0.1 sec.
- Same Query will be executed for multiple times in DB SW either same or diff values : 1,00,000 * 0.1 sec = 10,000 sec.
- Same Query output will be fetched out for multiple times from DB SW : 1,00,000 * 0.1 sec = 10,000 sec

Sample Code to work with JDBC PreparedStatement obj

Step1) Create SQL Query with parameters/ place holder (?)

~~Step2) / * PreparedStatement ps = con.prepareStatement~~

("insert into student values (?, ?, ?); ")
↑ ↑ ↑
1 2 3

String qry = "insert into student values (?, ?, ?);";

note: parameters / place holder in query represent the position or location in SQL Query for which the values can be set afterwards.

Step2) Create JDBC PreparedStatement obj

PreparedStatement ps = con.prepareStatement(qry);

This method takes given SQL query to DB API, makes the query as precompiled query as DB API and returns JDBC PreparedStatement object representing that query.

Step3 Set values to SQL Query params using setXXX (-,-) method

ps.setInt(1, 23+);

ps.setString(2, "mjsm");

ps.setString(3, "hyd");

Step4 Execute the SQL Query in DB API

int result = ps.executeUpdate();

Step 5) To execute the above SQL for multiple times repeat Step 3, Step 4 for multiple times

Step 6) Close JDBC objects.

```
ps.close();
```

```
con.close();
```

Write a JDBC app to insert multiple student details in DB table by collecting them from end user.

Note. Here we need to execute the same SQL query for multiple times with different values so we use prepared statement object.

PStudentTest.java

```
package com.nt;

public class PSInsertTest {

    public static void main(...) throws Exception {
        // register driver
        Class.forName("oracle.jdbc.driver.OracleDriver");
        // establish connection
        Connection con = DriverManager.getConnection(
            "jdbc:oracle:thin:@localhost:1521:oracle", "student", "tiger");
        // Student count
        Scanner sc = new Scanner(System.in)
        System.out ("Enter Student count");
        int crt = sc.nextInt();
```

```
//Prepare sql query  
String qry = "insert into student values(?, ?, ?);  
  
//Create jdbc PreparedStatement obj  
PreparedStatement ps = con.prepareStatement(qry);  
  
//read inputs from keyboard (student details)  
  
for(int i = 1; i <= cnt; i++)  
{  
    System.out.print("Enter sno:");  
    int no = sc.nextInt();  
    System.out.print("Enter sname:");  
    String name = sc.next();  
    System.out.print("Enter Address:");  
    String addr = sc.next();  
  
    int result = ps.executeUpdate();  
    //process the result  
  
    ps.setInt(1, no);  
    ps.setString(2, name);  
    ps.setString(3, addr);  
  
    //execute query  
    int result = ps.executeUpdate();  
    //process the result
```

```

if (result == 0)
    System.out ("record not inserted");
else
    System.out ("record inserted");
} // for
// close jdbc object
ps.close();
con.close();
} // main()
} // class

```

(21)
+-----+
| 04/02/15 |

All jdbc drivers supports 3 types of Statement object

We can make both select & non-select SQL queries as pre compiled SQL queries.

Executing Select Query with Prepared Statement Object -

// Create jdbc PreparedStatement object

PreparedStatement ps = con.prepareStatement ("Select * from Student where sno=? and sno<=?");

// set values to query param
ps.setInt (1, 100)

ps.setInt (2, 200);

// send and execute SQL query in DB I/O
ResultSet rs = ps.executeQuery();

```

// Process resultset
while (rs.next())
{
    System.out.println(rs.getInt(1) + " " + rs.getString(2) + " " + rs.getString(3));
}

```

The SQL query of JDBC PreparedStatement object can be there with parameters or without parameters.

PreparedStatement ps = con.prepareStatement("Select * from student");

Q) How can we execute static SQL query using PreparedStatement Object.

A) Call ps.executeQuery() / ps.executeUpdate() methods with Query

ResultSet rs = ps.executeQuery("Select * from student");
 → Execute this SQL query as static/raw SQL Query

Q) What is the difference b/w static SQL query and pre-compiled SQL query?

Ans) DB s/w compiler static SQL Query just before the execution and this query does not allow param.

DB s/w compiler pre-compiled SQL Queries irrespective of its execution in DB s/w and this query allows param / In param (?)

While creating SQL Query for PreparedStatement Object we can place parameters just representing input values that means we can't take parameters representing SQL keywords, column names, tablename.

Select * from student where sno = ? (valid)

Select ? from student

Select * ? student

Select * from where ? >= ?

} (Invalid)

Using one simple Statement object we can execute multiple SQL Queries. For this we must not use types driver.

Since PreparedStatement object will be confined to one SQL Query we can't use that object to execute other SQL Queries.

That means PreparedStatement Obj can be used one SQL query at a time.

The PreparedStatement Obj solves SQL injection problem.

Redesigning LoginApp application using PreparedStatement object -

```
public class LoginApp
```

```
{
```

```
    ==
```

```
    Create PreparedStatement
```

```
    PreparedStatement ps = con.prepareStatement("Select count(*) from  
    userlist where uname=? and pwd=?");
```

```
ps.setString(1, user);
ps.setString(2, pass);
// Send & execute SQL Query in DB
ResultSet rs = ps.executeQuery();
--  
--  
}
```

- Q) Why simple statement obj gives SQL injection problem, and
how can we do PreparedStatement obj solves the problem.

Ans) While working with simple statement obj the DB
&/w compiler SQL query with i/p values. ~~having~~
So they SQL instruction like -- that are given along
with input values participate in query compilation & changes
query behaviour during execution. This is nothing but SQL
injection problem.

While working with preparedStatement obj the SQL query
will be compiled in DB &/w without input values
So the special SQL instruction that are given along
with input values doesn't participate in query compilation
and it doesn't change behaviour of the query at
run time (execution time). This is nothing but solving
SQL injection problem.

What is difference b/w Simple Statement & Prepared Statement

Statement	Prepared Statement
① Deals with static SQL Query.	Deals with Precompiled SQL Query
② Doesn't allow parameters in the query	Allows
③ We can execute multiple SQL Queries at a time (except Type-1 driver)	Can't be used (one after another)
④ Suitable to execute query for onetime to without input values	Suitable to execute query for multiple times with or without input values.
⑤ Gives slow performance	Gives good performance
⑥ May give SQL injection problem.	Solves SQL injection problem
⑦ Doesn't allow to insert date values by collecting them in user choice pattern	Allows
⑧ Can't be used to insert large object (files)	Can be used
⑨ While setting i/p values to queries we need to convert them as required for query.	No conversion is required.

MySQL

22
08/02/15

Type: DB S/w

Version: 5.5

Vendor: Devex/ Sun/ Oracle

Open Source DB s/w (FreeWare)

Default Port No: 3306

Default Username: root

Password: Should be chosen during installation.

To download S/w: www.devex.com

Procedure to Create Logic DB in MySQL having DB table with records:

Step1) Launch MySQL Prompt.

Step2) Create Logical DB in MySQL

Step3) Create DB table with records in Logical DB H+DBI

Step4) Create table student (eno int(5) primary key, sname varchar(20),
 saddr varchar(20));

~~Step5~~) insert into student values (101, 'raja', 'hyd');

 insert into student values (343, 'rameli', 'vizag');

The type mechanism based JDBC driver for MySQL DB now given by
Devex is called as Connector/J JDBC driver. The details are

Driver class name: com.mysql.jdbc.Driver
(or)
org.gjt.mm.mysql.Driver.

JDBC URL: jdbc:mysql://<logical DB> (use this URL while working
with Local MySQL)
or

jdbc:mysql://<host>:<port no>/<logical DB>

Use this URL while working with remote MySQL DB A/I

JAR file: mysql-connector-java-5.1.6.jar

must be arranged separately

Example App to interact with MySQL DB A/I from Java App using
Connector/J JDBC driver.

Step1) Keep any JDBC App ready

Step2) Write following code for "step1" in the app.

- - - -

- - - -

//register driver

```
Class.forName("com.mysql.jdbc.Driver");
```

//Establish Connection

```
Connection con = DriverManager.getConnection("jdbc:mysql://localhost", "root", "root");
```

Statement st = con.createStatement

Step3) Add ~~step1~~ JAR file to class path or build path.

Step4) Compile & execute Java App

While working with any JDBC driver we can gather the following details:

a) driver class name

b) JDBC URL

c) JAR file that represents JDBC driver

Working with Date values:-

While working with DOB, DOJ, BillDate values we need to insert and retrieve Date values.

Don't store Date values in DB table as string values because we can't compare the Date. Always maintain Date values in DB table in Date Datatype column.

You can't use `java.util.Date` class object to insert Date values but we can use `java.sql.Date` class object for inserting Date values.

Once we give `java.sql.Date` class object to JDBC driver then it will inserts Date value in DataBase table in the pattern that it is supported by Underlying DB S/W.

Oracle supports dd-mm-yy (e.g. 20-oct-90)

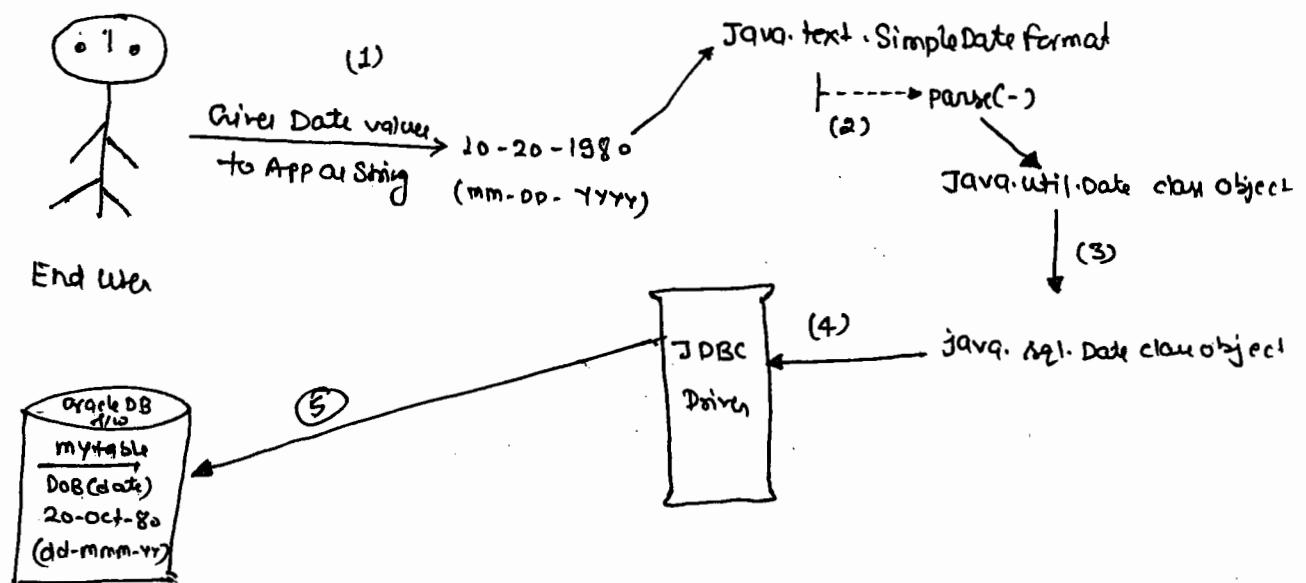
MySQL supports YYYY-mm-dd (e.g 1991-08-24)

Using simple Statement object you can't insert Date value in DB table if that date value is given in other pattern that is not supported by DB S/W.

Using Prepared Statement Obj we can not date value in DB table in the pattern that is supported by DB S/W even though the date value is given other pattern. For this we must take date value as `java.sql.Date` class Obj.

`JAVA.SQL.Date` is sub class of `java.util.Date` class.

Jdbc drivers and DB SQL independent procedure to insert date value to DB table.



w.r.t. the diagram

- 1) End User gives String Date value to application.
- 2) Application uses parse method of Simple Date format class to convert String value to `java.util.Date` class object.
- 3) Appn converts `java.util.Date` `java.util.Date` class object to `JAVA.SQL.Date` class object.
- 4) Appn set `java.sql.Date` class object jdbc driver using `setDate()` of Prepared Statement object.
- 5) Appn insert Date value to DB table in the pattern that it is supported by underlying DB SQL.

```
//Converting String date value to java.util.Date class obj  
String s1 = "10-12-1970" // dd-mm-yyyy  
SimpleDateFormat sdf1 = new SimpleDateFormat("dd-mm-yyyy");  
java.util.Date udt = sdf1.parse(s1);  
System.out.println("Util Date " + udt);  
  
//Converting java.util.Date class obj to java.sql.Date class obj  
long m1 = udt.getTime();  
java.sql.Date sqd1 = new java.sql.Date(m1);  
System.out.println("Sql Date " + sqd1);
```

*1 represents no. of milliseconds that are elapsed b/w 1970 Jan 1st 00:00 hrs (Epoch Standard) to the date & time represented by java.util.Date class obj(udt).

/* If given string date value pattern is yyyy-mm-dd
then it can be converted directly to java.sql.Date
class obj without converting it to java.util.Date
class obj
*/

```
String s2 = "2010-11-30"; //yyyy-mm-dd  
java.sql.Date sqd2 = java.sql.Date.valueOf(s2);  
System.out.println("Sql date " + sqd2);
```

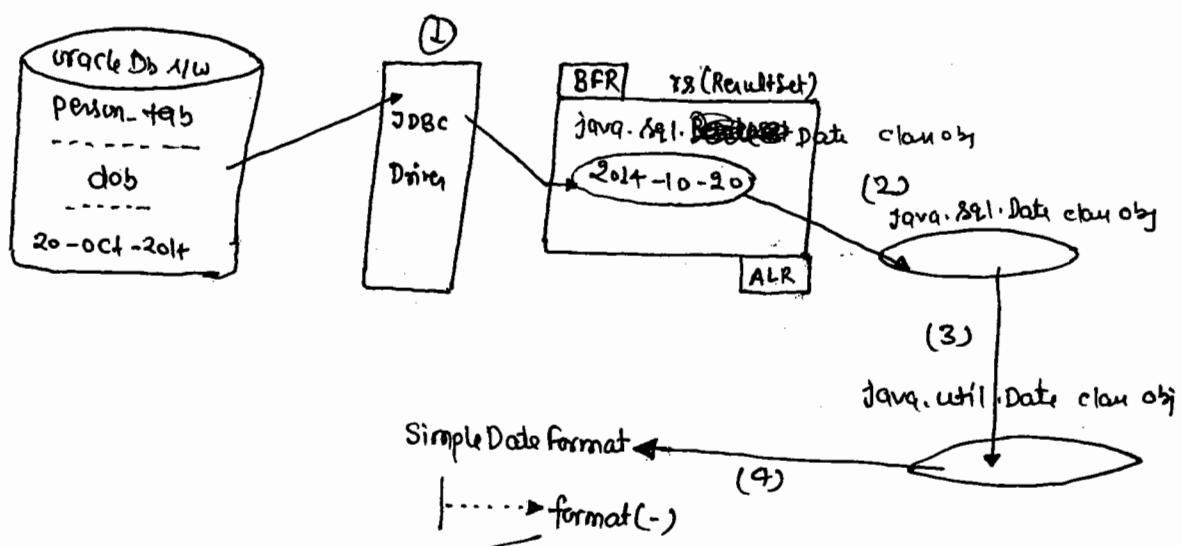
SimpleDateFormat class represents Date pattern which can be used on the pattern of i/p date or pattern of output date.

| (23) |
| 06/05/15 |

→ Develop a date input app

To Retrieve Date values from Db table colz we can use either Simple Statement obj or PreparedStatement obj.

- * Date values from Db table comes in the form of java.sql.Date class obj and we need to convert them date value to other formats as required end user.



Give date
Value as
String value,

20-10-14
(dd-mm-yyyy)

w.r.t. the diagram -

- ① Application executes Select SQL queries to retrieve Date value from DB table column into resultSet or java.sql.Date class object.
- ② App gets java.sql.Date class object from ResultSet.
- ③ App converts java.sql.Date class object to java.util.Date class obj.
- ④ App converts java.util.Date class object Date value to String Date value by using format() method of SimpleDateFormat class

```
String s2 = "2010-15-44" // YYYY-mm-dd
Java.sql.Date sqd2 = java.sql.Date.valueOf(s2);
System.out.println("Sql date " + sqd2);

// Converting Java.sql.Date class obj to java.util.Date
Java.util.Date ufd2 = (java.util.Date)sqd2
System.out.println("Util Date " + ufd2);

// Note: java.sql.Date is the sub class of java.util.Date

// Converting java.util.Date class obj to String date value
SimpleDateFormat sdf2 = new SimpleDateFormat("YYYY-mmMM-dd")
String t3 = sdf2.format(ufd2)
System.out.println("String date " + t3);
```

Note- While working with SimpleDateFormat class obj parse() user given date & pattern for the pattern of i/p date similarly, format() user given date pattern to the pattern of output date.

Example Appn of retrieving date values from DB table column and give them to end user in different pattern.

→ Refer booklet, appn 10 (page no 28)

Write a JDBC appn to collect DOB from end user & calculate age of the person and insert person details to DB as record.

Code that calculate Age Based on given DOB

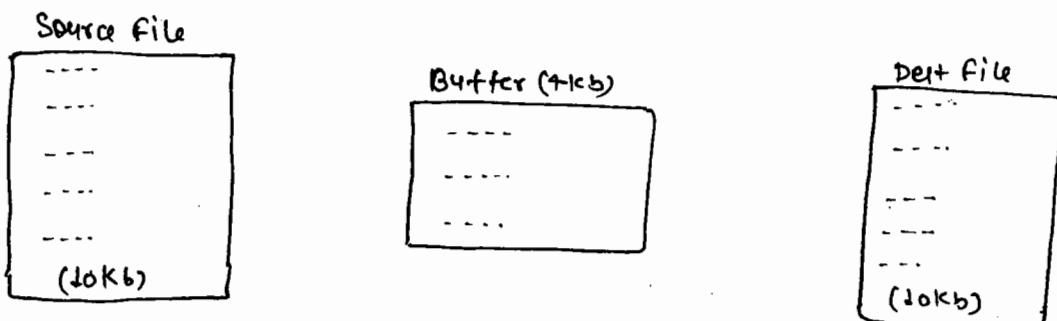
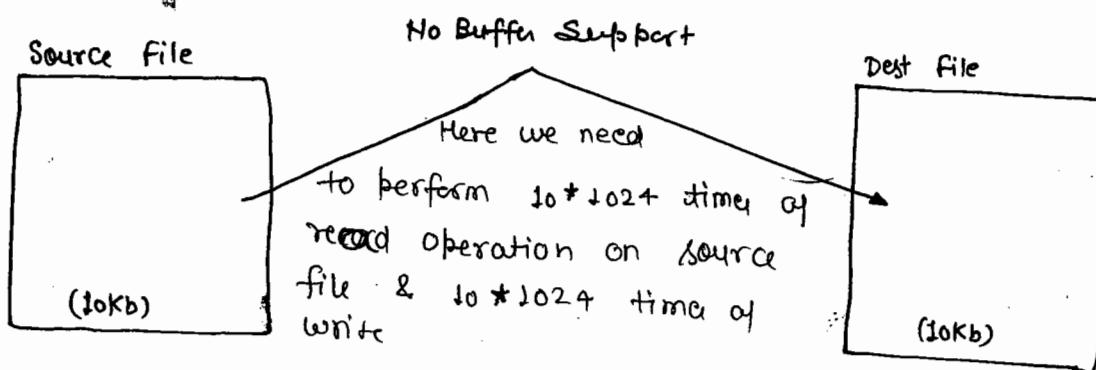
```
class Datecal {  
    public static void main (String[] args)  
    {  
        Scanner sc = new Scanner (System.in);  
        String dob = sc.next();  
  
        // Given java.util.date class object representing date value  
        SimpleDateFormat sdf1 = new SimpleDateFormat ("yyyy-mm-dd");  
        java.util.Date udob = sdf1.parse (dob);  
  
        // get Sysdate & time  
        Date sysdate = new Date();  
        // calc age  
        long uddobmu = udob.getTime();  
        long sysdate
```

25

8/05/15

-: SUNDAY:-

- * To perform & retrieving of large object we can use either simple statement object or prepared statement obj.
- * Buffer or cache is a temporary memory that holds data for temporary period.
- * While transferring data from source to destination if we take the support of buffer the no. of read oprn on source & no. of write oprn on destination file will be reduced drastically.



Here we need to perform read operation on source file only for 3 times & write operation on dest file only for 3 times because of buffer support.

While working with buffer we need to take a counter variable that keeps track of amount of data that is read from source file to buffer each time so that we can write same amount of data from buffer to destination file each time.

DB S/W JDBC driver independent code to retrieve Large objects (LOB) (BLOB) from Db table cols



Java App for Photo Retrieving

- * Create JDBC Statement Obj
Statement st = con.createStatement();
- * Send & execute SQLQuery obj in Db S/w
ResultSet rs = st.executeQuery("Select * from emp");
- * Get Input-Stream pointing to BLOB value of ResultSet
InputStream is = rs.getBinaryStream(4);
If (rs.next()) {
 is = rs.getBinaryStream(4);
}
- * Get OutputStream pointing to Dest file
File OutputStream fo = FileOutputStream
(D:/abc/newpic1.gif);
- * Copy Blob value of ResultSet to Dest file using buffer

```

byte[] buffer = new byte[4096]; //buffer
int bytesRead = 0;
while ((bytesRead = is.read(buffer)) != -1) {
    fo.write(buffer, 0, bytesRead);
}
//close streams
is.close(); st.close(); con.close(); fo.close();
    
```

*1 The Counter variable that keeps ^{track of} no. of bytes data that are ready to buffer reads from resultset.

So at that same amount of data can be written destination from buffer each time.

⇒ To read Blob value from Resultset we can use rs.getBinaryStream(). Similarly we can use rs.getCharacterStream() method to read CLOB value from ResultSet.

Write a JDBC app to retrieve BLOB value (Photo) from DB column.

* Refer Booklet

public class PhotoRetriever {

 throws Exception {

 Scanner sc =

 int no = sc.nextInt();

 Class.forName

 Connection

 Statement st = con.createStatement();

 Resultset rs = st.executeQuery("Select * from empall where empno=" + no);

 InputStream ix = null;

 if (rs.next()) {

 ix = rs.getBinaryStream(1);

 }

 else {

 System.out.println("Record not found");

 }

```
//copy received photograph to Dest file  
FileOutputStream fos = new FileOutputStream ("d:/asc/newpic1  
//use buffer based logic to complete photo retrieval  
int byteRead = 0;  
byte[] buffer = new byte[4096];  
while ((byteRead = is.read(buffer)) != -1) {  
    fas.write(buffer, 0, byteRead);  
} //wfile  
  
//close jdbc objects
```

Develop jdbc application to insert & retrieve CLOB values.

Understanding the OOPS related to jdbc code:-

You always use jdbc obj in jdbc programming without referring their class name because this class ^{name will} ~~are change~~ based on jdbc driver.

This process allows to develop maximum jdbc code as driver independent code.

Statement st = con.createStatement();

To practice prove
use JavaP command
with real class name

- a) createStatement() is declared in `java.sql.Connection()`
- b) ~~createStatement()~~ is implemented in driver supplied java class that implements `java.sql.Connection()`. In type 1 jdbc driver this class name is `Sun.jdbc.odbc.JdbcOdbcDriver`
- c) The createStatement() method definition contain logic to create Statement obj which is nothing but implementation class obj of `java.sql.Statement()`. In type 1 this name is `Sun.jdbc.odbc.JdbcOdbcStatement`

- d) Since interface ref variable can refer its implementation class obj, the `java.sql.Statement()` reference variable (st) can refer the implementation class object of `java.sql.Statement()` returned by `con.createStatement()`.
- e) Result Set rs = st.executeQuery("select * from student");
javap `java.sql.`

- f) executeQuery() is declared in `java.sql.Statement()`;

- g) executeQuery() is implemented in jdbc driver supplied java class

that implements `java.sql.Statement()`

In type1 jdbc driver this class name is

"`Sun.jdbc.odbc.JdbcOdbcStatement`"

- c) `executeQuery()` definition contains the logic to create & return JDBC `ResultSet` obj (that means implementation class of `java.sql.ResultSet()`).
- d) The reference variable of `java.sql.ResultSet()` (rs) after implementation class obj. `ResultSet()` returned by `executeQuery()` method.

d)

```
int result = st.executeUpdate("...");  
int no = rs.getInt(1)  
rs.close();  
boolean b = st.execute();
```

26
09/02/15

While working with clob value insertion you need to deal with following methods -

ps.setCharacterStream(-,-,-)

ps.setClob (-,-)

While retrieving clob value from Db table cols we need to use rs.getCharacterStream(-) or rs.getClob(-) method.

Text file , Rich Text file can be treated as clob value.

Example App on inserting CLOB value

SQL> Create table EmpProfile (no NUMBER(5), name varchar2(10),
resume(clob));

//clobinsert.java

→ refer booklet (page 60, 61)

//ClobReceive.java

To insert clob value we need to use Prepared Statement object but to retrieve clob value we can use either simple Statement or Prepared Statement Obj.

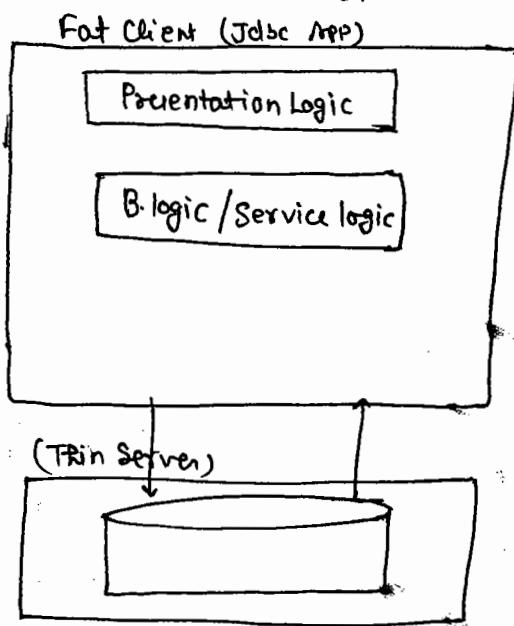
ClobTest1

Callable Statement Object :-

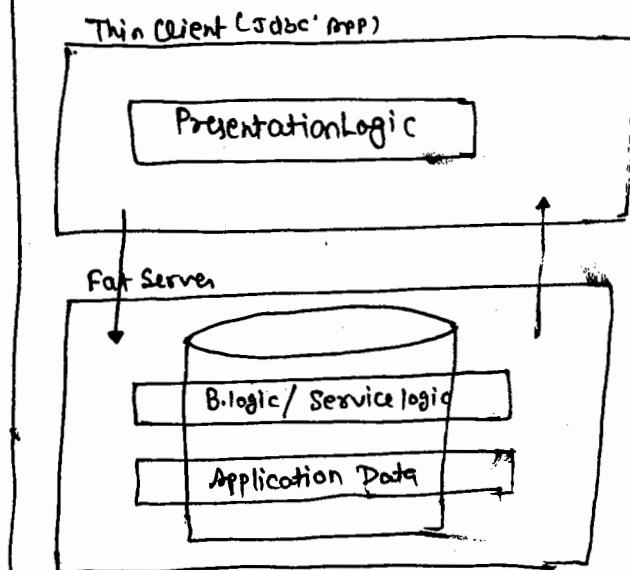
There are two types of jdbc based two tier application—

- i) FatClient Thin Server Application
- ii) ThinClient FatServer Application

① Fat Client Thin Server



② Thin Client Fat Server



- ③ While working with SimpleStatement, PreparedStatement objs our jdbc application contains both presentation, business logic, so it is called FatClient & database s/w just contain table with records. So it is called Thin Server.

- ④ In order to develop thin client - fat Server application we need to place business logic in DB s/w as PL/SQL procedure or function. (Stored Procedure or function).

To call them from jdbc app we use CallableStatement object

In order to reuse business logic in multiple app's of a module we call multiple module of a project then we can place business logic in DB s/w as PL/SQL procedure or function.

If multiple modules of a project use same authentication logic then instead of writing that authentication logic in every module place that logic in DB SQL or PL/SQL procedure or function and use them in multiple modules of project.

- PL/SQL procedure does not return a value, where as PL/SQL function returns a value.

Parameters of PL/SQL procedure or function contains mode, type
There are 3 modes for parameters

- a) IN (as input)
- b) OUT (as output)
- c) INOUT (as INOUT param)

~~Y := X * X~~ (take "X" as in param and "Y" as outparam)
~~X := X * X;~~ (take "X" as IN OUT param)

To create PL/SQL procedure in oracle DB SQL

Launch SQLPlus as administrator

Username: scott

Password: tiger

SQL:> one or ~~the~~ other dummy query

SQL:> col

CREATE OR REPLACE PROCEDURE first_Pro(X IN NUMBER,

Y OUT NUMBER) AS

BEGIN

Y := X * X;

End;

Filemenu save, exit

oracle
1 (+new)

JDBC datatypes are the bridge datatype b/w javatype and underlying DB SQL type. Then datatype helps JDBC driver to convert java values to DB values and vice-versa.

Java Datatype	JDBC Datatype	Oracle SQLtype
int	Type. INTEGER	number
String	Type. VARCHAR	VARCHAR, VARCHAR2
java.util.Date	Type. DATE	date
double	Type. DOUBLE	number
byte[]	Type. BLOB	BLOB
char[]	Type. CLOB	CLOB
:	:	:

27
10/02/15

- * Using SimpleStatement, PreparedStatement, obj we can develop fat client thin server
- * Using CallableStatement obj we can call PL/SQL procedure/function of Db SQL's to develop fat client thin client fat server App.
- * CallableStatement obj also allows to make b-logic/reusable logic or reliable logic by keeping in Db SQL as PL/SQL procedure or function.

Procedure to work with callableStatement obj to call PL/SQL procedure.

Step1) prepare query calling pl/sql procedure

```
String qny = "{call first_prc (?, ?)}";
```

Step2) Create jdbc CallableStatement obj

```
CallableStatement cs = con.prepareCall(qny);
```

Step3) Register out params with JDBC types

```
cs.registerOutParameter(2, Types.INTEGER);
```

Note:- All jdbc datatype of are constants of java.sql.Types class

All jdbc datatypes must be registered with jdbc datatype.

Step4) Set value to IN parameter

```
cs.setInt(1, 20);
```

Step5) Call PL/SQL procedure

```
cs.execute();
```

Step6) Gather result from OUT param

```
int result = cs.getInt(2);
```

Step7) To call PL/SQL procedure for multiple times with different values repeat Step4 to Step6 multiple times.

Step8) Close jdbc objects.

```
cs.close();
```

```
con.close();
```

We use getXXX(-) to gather result from out parameter based on JDBC type i.e. wed to register the out parameter.

Note → Why we are not registering in parameter with jdbc types?

We use setXXX() to set values to in parameters. Based on the XXX part of setXXX the IN parameter will be registered with jdbc types automatically.

//CSTest1.java

```
public class CSTest1 {  
    public static void main (String [] args) throws Exception  
    {  
        //read input  
        Scanner sc = new Scanner (System.in);  
        System.out.println ("Enter Number:");  
        int no = sc.nextInt ();  
        //register driver & establish connection  
        Class.forName ("oracle.jdbc.driver.OracleDriver");  
        Connection con = DriverManager.getConnection (  
            "jdbc:oracle:thin:@1521: \"scott\", \"tiger\"");  
        //create Callable Statement obj  
        CallableStatement cs = con.prepareCall ("{call first_pro (?, ?)}");  
        //register out param with jdbc type  
        cs.registerOutParameter (2, Types.INTEGER);  
        //set value to IN parameter  
        cs.setInt (1, no);  
        //call pl/sql procedure  
        cs.execute();
```

```

    //gather result from OUT param
    int res = cl.getInt(2);
    System.out.println("Result=" + res);
    //close jdbc obj
    cs.close();
    con.close();
}
//main
}
//class

```

Develop Callable Object based app" that calls pl/sql procedure to get emp name, salary ,designation based on given employee target no.

Refer booklet Page no: 32

SYS_REFCURSOR

* The regular datatypes variable of sql can hold one value at a time to hold bunch of records / values we need cursor support.

CURSOR is a special in memory variable of oracle pl/sql programming which can hold bunch of records given by select query execution.

Sys-refcursor is predefined cursor datatype given oracle datatype

`java.sql.Type` class does not provide JDBC datatype for CURSOR. To overcome this problem we use `oracle.type.cursor` an JDBC type given by oracle Corp

To gather result from Cursor type Out param we can use
C1.getObject(-) which gives jdbc Resultset obj.

Example App that gives Student details based on given initial char
of student name

PL/SQL Procedure:-

Create or replace procedure fetchStudDetails (initchars in varchar,
details out sys_refcursor) as

begin

open details for

Select * from student where ename like initchars;

End;

/

The record given by the select query will be stored in
the cursor variable details.

The cursor variable of Oracle PL/SQL programming is like
Resultset obj of jdbc programming.

//register out param with jdbc driver

C1.registerOutParameter(2, OracleType.CURSOR)

//Set input value

C1.setString(1, initchars);

C1.execute();

ResultSet re=(ResultSet)

Assignment -

Write a JDBC app that calls PL/SQL procedure in out param.

Write a JDBC app that calls PL/SQL procedure to delete student details based on given city name.

+-----
⑧
+-----
11/02/15 +-----

In PL/SQL programming of Oracle we can use "SQL%ROWCOUNT" to get numeric value to represent no. of records that are affected when non-select SQL Query is executed.

Write a JDBC app that delete student record based on given student ~~city~~ by calling PL/SQL procedure

PL/SQL Procedure:-

Create or replace procedure deleteStudent(
no in
city in VARCHAR, Cnt out
number) as

begin

delete from Student where Saddr = city;

Cnt := SQL%Rowcount;

end;

/*CSTest3

public class CSTest3 {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

System.out.println("Enter city:");

String city = sc.nextLine();

*1

```

Callable Statement cs= con.prepareStatement("{
    call deleteStudent(?,?) }");

//register out parameter with,
cs.registerOutParameter(2, Types.Integer INTEGER);

//set value to in param
cs.setString(1, city);

//call pl/sql procedure
cs.execute();

//Get the Result from pl/sql procedure
System.out.println("No. of Records that are deleted "+cs.getInt(2));

//close jdbc object
cs.close();
con.close();

} //main
} //class

```

PL/SQL procedure does not return a value whereas PL/SQL function returns a value

To gather results from PL/SQL procedure we multiple out multiple param. To gather multiple results from PL/SQL function we must take one result as return value & other results as out param.

The Query that calls PL/SQL fxn must have one parameter to hold return value & this return parameter must be registered with jdbc type by treating it as out parameter

{ ? = call <function_name>(?, ? --) }

Write a JDBC App that calls PL/SQL function to get Sname, address based on the given student no.

Create or replace function getStudDetail (no in number, name out

```
    varchar) as return varchar  
    address varchar(20);  
begin  
    select Sname, Saddr into name, address from student  
    where sno = no;  
    return address;  
end;
```

// CsfxTest1

```
public class CsfxTest1 {  
    public static void main (String [] args) throws Exception {  
        Scanner sc = new Scanner (System.in);  
        System.out.println ("Enter student no:");  
        int no = sc.nextInt();  
        Class.forName ("oracle.jdbc.driver.OracleDriver");  
        Connection con = DriverManager.getConnection ("jdbc:oracle:thin:@  
            localhost:1521: \"sys\" \"sys\"");
```

// create Callable Statement

```
CallableStatement cs = con.prepareCall ("{  
    ? = call getStudDetail (?, ?) }");
```

// register out parameter with JDBC type

```
cs.registerOutParameter (1, Types.VARCHAR); // return param
```

```
cs.registerOutParameter (2, Types.VARCHAR); // out param
```

```

    cs.setInt(2, no);

    // call PL/SQL function
    cs.execute();

    // gather results
    System.out.println("Name:" + cs.getString(3) +
        "Address:" + cs.getString(1)
    );
}

// close JDBC objects
cs.close();
con.close();
} // main
} // class

```

// write a JDBC app that calls PL/SQL function to retrieve a record based on Student no & also to delete that record.

```

Create or replace function fetchStudForDeletion(
    no in NUMBER
    detail out sys_refcursor) return number
as
    cnt number;
begin
    open detail for
        select * from student where sno = no;
        delete from student where sno = no;
        cnt := SQL%ROWCOUNT;
    returnnt cnt;
end;
/

```

//CsfxTest2.java

```
public class CsfxTest2 {
```

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

```
        Scanner sc = new Scanner (System.in);
```

```
        System.out.println ("Enter Student no:");
```

```
        int no = sc.nextInt();
```

//create callable statement.

```
CallableStatement cs = con.prepareCall ("{"
```

```
? = call.fetchStudentForDeletion (?, ?) }");
```

//register out parameter

```
cs.registerOutParameter (1, Types.INTEGER);
```

```
cs.registerOutParameter (3, OracleTypes.CURSOR);
```

//Set value to IN param

```
cs.setInt (2, no);
```

//call pl/sql fn

```
cs.execute();
```

//Gathering Result

```
ResultSet rs = (ResultSet) cs.getObject (3); //from Out param
```

//Process the result

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

```
    System.out.println (rs.getInt (1) + " " + rs.getString (2) + " " +  
                       rs.getString (3));
```

} //while

```
int count = cs.getInt (1); // from return param
```

```
if (count == 0)
```

```
    System.out.println ("No of records that are deleted " + count);
```

In industry people prefer to use PL/SQL Procedure over PL/SQL fxn, because there is no need of dealing with return values while working with PL/SQL procedure.

In our project 70% persistence logic will be developed through SQL Queries & remaining logic will be developed by PL/SQL procedure.

+-----+
| 29 |
+-----+
12/02/15

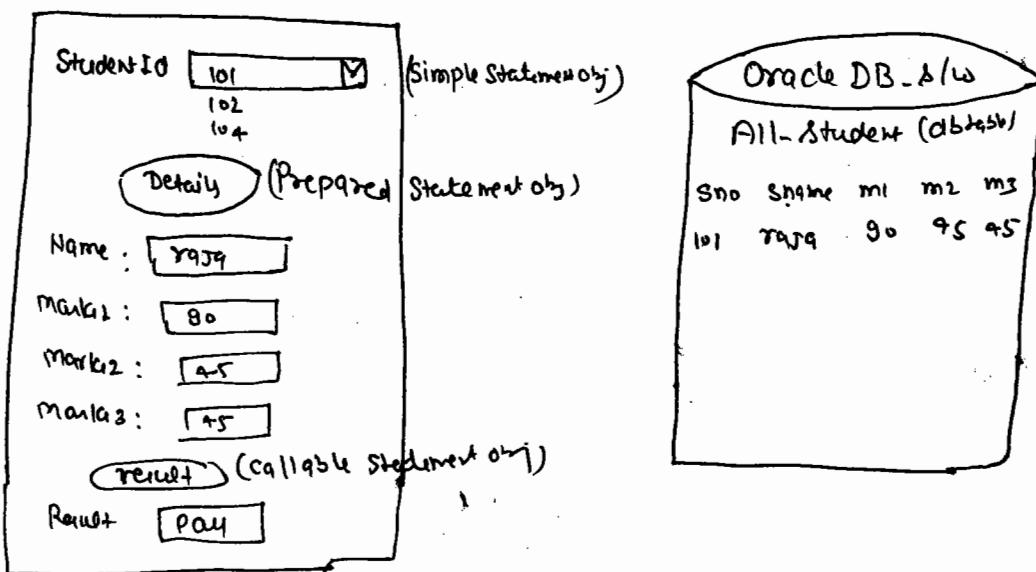
Conclusion on Statement Object:-

We Simple Statement Object to send & execute SQL query in DB Alw only for 1 time without input value.

We PreparedStatement obj to execute same SQL Query for multiple times with diff values.

We CallableStatement obj to call PL/SQL procedure or function of DB Alw

We can develop GUI front App for DB Alw either using AWT or swing



Container: JFrame

Components : JPanel → (6)

JComboBox → 1

JTextField → (5)

JButton → (2)

Layout Manager

Refer Appn 37 (37 to 39)

In the above appn student no. will be populated to combo box during appn startup for that we need to execute "Select * from All_Student" SQL query only for 1 time without input values. So we simple Statement object.

We click on details button multiple times by selecting student's no. for JComboBox. For this "Select * from all_Student where Smb=? " query should be executed for multiple times with diff values. So we Prepared Statement Obj.

Since logic to generate result is common for multiple departments of university it will be placed in DB like a PL/SQL procedure so we use JDBC CallableStatement Obj to call that PL/SQL procedure/ function.

For Source of the appn refer booklet.

It is always recommended to create jdbc objects in one-time execution blocks for better performance. These blocks are like Constructor - static block etc.

In order to perform a persistence operation on Db table we need to create Connection object only for one time and use it for multiple times.

The logic of creating JDBC object, load on Startup activities and instantiating GUI components is always recommended to place in the constructor or static block (one-time execution block).

Develop GUI appn to perform insert, update, delete, view etc.

(31)
14/02/15

Oracle, MySQL etc. are conventional DB A/w's. M.S.Excel, Text files, Word Documents are non conventional DB A/w's. That means they are actually not DB A/w but they can be treated as DB A/w's.

In real time project the conventional DB A/w's will be used as main DB A/w. Non conventional DB A/w's will be used as supporting DB A/w's.

ICICI bank uses Oracle as main DB A/w for their regular banking transaction whereas M.S.Excel will be used as supporting DB A/w for print jobs of A/c statements.

Since type1 JDBC drivers for non-conventional DB A/w's are commercial we use type1 driver to interact with them DB A/w's.

M.S.Excel A/w (Physical DB A/w)



Procedure to interact with non conventional Db like (MS-Excel) using types (JDBC driver): —

* Step1) Create dsn for Microsoft ODBC driver for MS Excel.

C:\Windows\Sy\Wow64\

Step1) Create Excel work sheet having data as shown below.

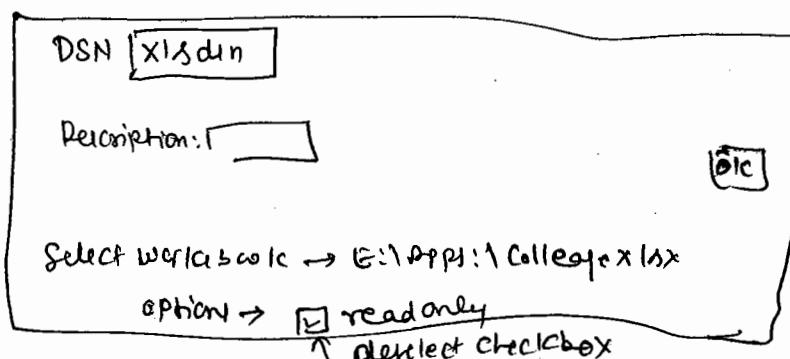
E:\Apps\College.xlsx

Sno	sname	sadd
111	Yamash	Hyd
222	Ravi	Vizag

-college.xlsx

Step2 Create merden for Microsoft ODBC driver for Excel.

C:\Windows\Sy\Wow64\Odbcad32.exe → merden → add → MSExcelDriver (+.xslc) →



Step3) Develop App to get values from MS-Excel sheet using types driver

```
public class ExcelTest {
```

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

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

```
        Connection con = DriverManager.getConnection("jdbc:odbc:  
                                         Xlsdn");
```

```
Statement st = con.createStatement();
//Send & execute SQL Query
ResultSet rs = st.executeQuery ("Select * from [ Sheet1$ ]");
//Process the resultset
while (rs.next()) {
    System.out.println (rs.getInt(1) + " " +
                        rs.getString(2) + " " +
                        rs.getString(3));
}
rs.close();
st.close();
con.close();
3
```

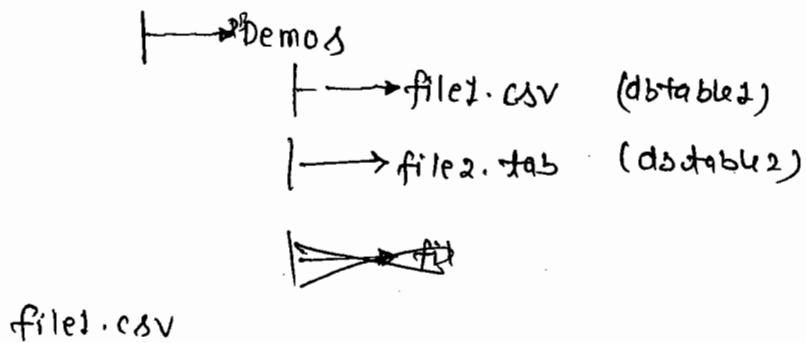
Insert the record:-

```
PreparedStatement ps = con.prepareStatement ("insert into [ Sheet1$ ]");
ps.setInt (1, 55);
ps.setString (2, "ramesh");
ps.setString (3, 'Byd');
System.out.println ("no. of records inserted ?" + ps.executeUpdate());
```

Working with files

The file that contains formatted data can be called as DB.
csv, tab file can be taken as db table.

E:\App



Sno, Sname, Saddr | (col names)
101, raja, hyd | (col
102, ram, hyd | value)

The file that maintains data having common delimiters is called Formatted file.

The whole file system is called physical DB & in that directories are called logical DBs. In that formatted files are called DB tables. In that file first row content is called column names & remaining rows are called column values.

Example App to retrieve data from text file.

Step1) Created the formatted file as shown above

E:\App\DB\Demol file1.csv

Comma separated value

Step2) Create dsn for microsoft text driver

C:\windows\system32\odbcad32.exe → userdn → microsoft
text driver

datasourcename: txtdsn

Database : select directory (E:\App\DBDemo) → ok ..

file1.csv

Step 3) Develop JDBC app using tybet driver.

```
public class TxtFext {
    public static void main(String[] args) throws Exception {
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        Connection con = DriverManager.getConnection(
            "jdbc:odbc:txtdsn");
        Statement st = con.createStatement();
        ResultSet rs = st.executeQuery("select * from file1.csv");
        while (rs.next()) {
            System.out.println(rs.getInt(1) + " " +
                rs.getString(2) + " " +
                rs.getString(3));
        }
        // Close JDBC objects
    }
}
```

filename au
→ tablename

write a jdbc appn to copy the records of excel sheet
to oracle DB table.

//ExceltoOracle.java

→ Refer Booklet (Appn 23 of page no 43)

From one jdbc appn we can interact with both conventional
& non conventional DB slw's at a time.

32
+-----+
| 16/02/2015 |

In real time programmers use separately installed GUI DB
tools to perform DB operations.

Toad for oracle,

Toad for mysql,

SQLyog and etc.....

Procedure to create logical DB having DB table with records in MySQL
using SQLYog (GUI DB Tool):—

Step1) Install SQL YOG

Step2) Launch SQL YOG to connect MySQL

Launch SQLYOG → New → Conn

localhost

username: root

password: root

3306

Step3) Create Logical DB Having db table with records.

R.C on root@localhost → Create database

Enter new database name

Create

R.C. on NtDb2 → Create table

Field Name	Data Type	Length
Pid	int	5
Pname	varchar	20
Qty	int	5

Create table

Enter tablename

OK

R.C. Product → insert / update Data "[File shortcut]"

- 1) Database metaData
- 2) Resultset metaData
- 3) Parameter metaData

MetaData -

Data about Data is called metaData. gathering more information about existing information is called metaData or meta info.

JDBC supports 3 styles of metaData operations -

(i) DB metaData

1) DataBaseMetaData

Give limitations & capabilities of underlying DB API

2) ResultSetMetaData

Give Info about db table that is represented by
ResultSet obj

3) ParameterMetaData

Allows to gather info about parameters (?) that
are there in the SQL Query represented by
PreparedStatement , CallableStatement obj.

DataBaseMetaData -

- * Give more information about underlying DB API & JDBC
driver that is being used.
- * Give limitations & capabilities of underlying DB API.
- * DataBaseMetaData obj means it is the obj of ~~java~~ class
that implements `java.sql.DatabaseMetaData()`
- * To Create the DataBaseMetaData obj

```
DataBaseMetaData dmbd = con.getMetaData();
```

- * This is very useful to developed GUI DB tools to perform
DB operations.
(e.g. SQLyog for mysql, Toad for Oracle)
- * Using this we can know logical DB names, table names,
procedure/ function details and etc.....

Example Appn

↳ Refer Booklet (45, 46)

The details given by DB metadata will be change based on the DB SW we used & JDBC driver we use

ResultSetMetaData -

Useful to know more details about the dbtable that is represented by ~~result~~ ResultSet obj.

Useful to print DB table data along with column names & datatypes.

It is the Obj of underlying JDBC driver supplied Java class that implements java.sql.ResultSetMetaData interface.

```
ResultSet rs = st.executeQuery("Select * from student");
ResultSetMetaData rsmd = rs.getMetaData();
```

This object is useful in report generation & also GUI DB tool creation.

Using this obj we can give table name, col names, col data types and etc....

Note- All JDBC drivers support all metadata opr in JDBC programming.

For Example. Appn on ResultSetMetaData that display table record along with the column names uses following code—

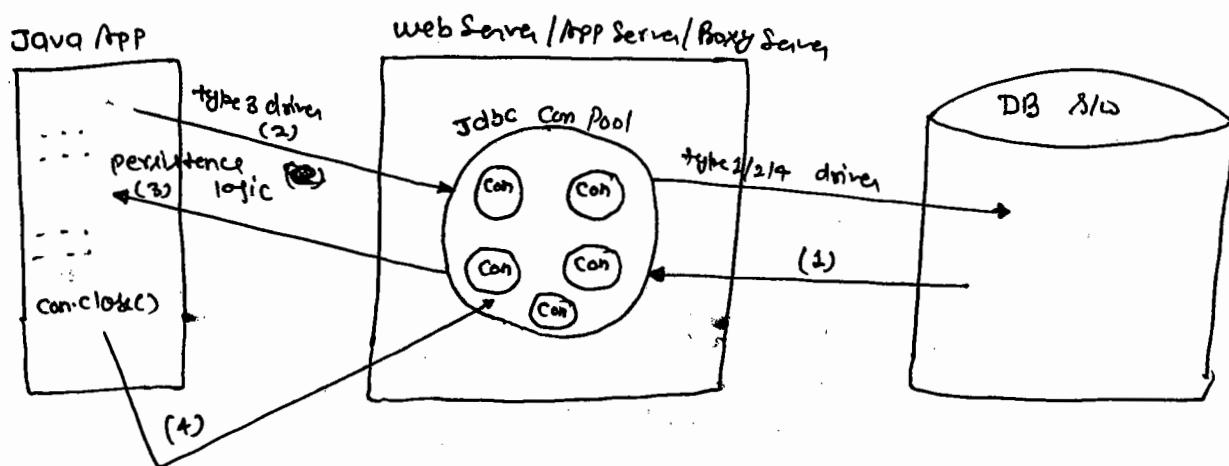
```
//Send & execute SQL query in Db SW  
ResultSet rs = st.executeQuery ("Select * from student");  
//get ResultSetMetaData obj  
ResultSetMetaData rsmd = rs.getMetaData();  
//get col count  
int colcnt = rsmd.getColumnCount();  
//Display col table  
for (int i=1; i<=colcnt; i++) {  
    System.out.println (rsmd.getColumnLabel(i) + "("  
        + rsmd.getColumnType(i) + ")"  
    );  
}  
  
while (rs.next()) {  
    for (int i=1; i<colcnt; i++) {  
        System.out.print (rs.getString(i) + "|");  
    }  
    System.out.println ();  
}
```

* for complete example Appn on ResultSet metaData refer
booklet (App 25 of P.N 46)

Jdbc Type3 driver (Net Protocol All java driver)

The architecture of type3 drivers we work with 3 tier environment where middle tier will be webserver or app server or proxy server maintaining jdbc connection pool.

Jdbc connection pool is a factory that contains set of readily available connection object before actually being used. This connection pool gives reusability of connection objects & allows more clients to interact with DB S/W with minimum connection objects.



Because of the middle server (middleware) of type 3 environment we can enable auditing, logging & etc facilities on the app? w.r.t the diagram—

- 1) Server uses type 1/2/4 driver to interact with DB S/W & to create no connection objects in the connection pool.
- 2) java App uses type 3 driver to get connection objects from connection pool

3) Java App develops jdbc persistence logic to manipulate DB data.

4) Java App releases the connection object back to connection pool to make it free for other client

Advantages:-

We can take all the advantage of jdbc con pool.

Logging & Auditing can be enabled.

Server control activities, related to DB connectivity.

Server takes the responsibility of creating, managing & destruction of Connection objects.

Disadvantage:-

Gives performance issue

Q What is jdbc project that you have use in ur project?

A If ur project is running outside of server then use type 4 jdbc driver.

If your project is running inside the server then use type 3 with type 4.

Here type 4 driver will be used to create connection object in a connection pool. & type 3 driver will be used to get connection objects from connection pool.

IDSServer → Proxy Server

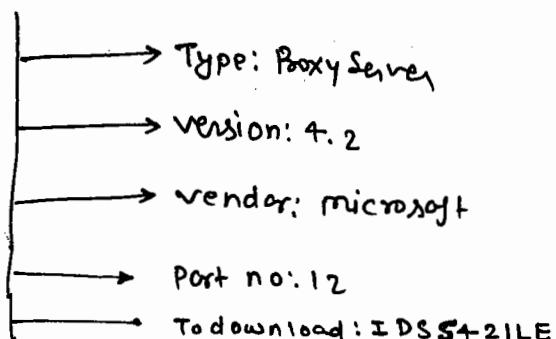
Port no: 12

given by microsoft

Proxy Server is a dummy Server that acts as middleware server as required in type 3 driver environment —

Microsoft's IDSServer is proxy server. This server built-in jdbc driver type 3.

IDSServer



IDSServer supplied jdbc type 3 details -

driver class name: idsgsq1.IDSDriver

url : jdbc:idsgsq1://<host>:12/conn?<query string>

Location: D:\IDSServer\classes
(IDSServer_home)

Along with the url of Ids server supplied type 3 driver, if you pass details of type 1 | type 2 | type 4 driver in the form of <query string> content then IDSServer uses that driver to establish the connection.

Example On

type 3 driver type 1

Java ----- → IDSServer ----- → DB SQL (oracle)

Step1) Install IDSServer (Internet Database Server)

Step2) Create System DSN for microsoft odbc for driver
(oradsn)

Step3) Develop java Appn using type3 jdbc driver

Step4) Add <IDSserver_home>\claver folder to CLASSPATH env variable.
(D:\IDSserver\claver)

// Type3Test1.java

```
Class.forName("iids.sql.IDSServer");
```

```
Connection con = DriverManager.getConnection ("jdbc:iids://localhost:12/
```

```
/*  
Connection con = DriverManager.getConnection ("jdbc:iids://localhost:12/  
conn?drn='oradsn' &  
usr='scott' & pwd='tiger');  
conn?drn='acdln');*/
```

```
Statement st = con.createStatement();  
---  
---
```

(33)
14/02/15

ParameterMetaData

Give Info about parameters(?) that are in the SQL Query represented by Prepared Statement obj like mode, name & etc.

ParameterMetaData obj means it is the obj of jdbc driver supplied java class that implements `java.sql.ParameterMetaData`(I).

To get this object

```
PreparedStatement ps = con.prepareStatement("Select * from student  
where sno=? and sno<=?");
```

```
ParameterMetaData pmd = ps.getParameterMetaData();
```

For Example on ParameterMetaData refer booklet.

- ⇒ Most of the jdbc drivers are not giving support for ParameterMetaData
- ⇒ Only CloudSpace supplied jdbc driver (commercial) is giving support.

Working with ResultSet:-

ResultSet obj is a java object that represents records of db table.

There are two types of jdbc ResultSets

1. Non-Scrollable ResultSet
2. Scrollable ResultSet

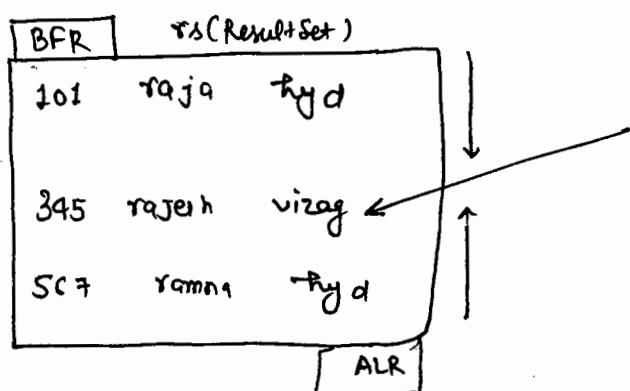
The ResultSet obj that allows records only in one direction (top-bottom) is called Non-Scrollable ResultSet.

In this we can't access the records of Resultset randomly.

e.g. In this Resultset to access 99 record we must go through 98 records.



The Resultset obj that allows to access records non sequential randomly, bidirectionally and directly is called ScrollableResultset obj.



In Scrollable Resultset obj cursor move randomly that means we can directly jump to 99th record from any record.

The jdbc Statement obj that is created by using type, mode values given in Scrollable Resultset obj.

The Statement obj that is created without using type, mode values given in Non-Scrollable Resultset obj.

To Create Non-Scrollable Resultset -

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

```
ResultSet rs = st.executeQuery("Select * from student");
```

To Create Scrollable Resultset obj.

Statement st = con.createStatement(type, mode);

ResultSet rs = st.executeQuery("Select * from Student");

Possible values for type(int):—

ResultSet.TYPE_SCROLL_SENSITIVE (1005)

ResultSet.TYPE_SCROLL_INSENSITIVE (1004)

Possible value for mode(int):—

ResultSet.CONCUR_UPDATABLE (1008)

ResultSet.CONCUR_READ_ONLY (1007)

* Public static final variable are called constant.

* All the variables of interface are ~~ext.~~ Constant by default.

* The methods that are invokable on Non-Scrollable ResultSet obj

next() → Move the cursor to next record in ResultSet.

getXXX() → To gather values from records of ResultSet

close() → To close ResultSet.

getRow() → Give the current position of the cursor in ResultSet

The methods that invokable on Scrollable ResultSet obj -

next()

previous() --> move the cursor to previous position

getXXX()

close()

getRow()

afterLast() ----> Moves the cursor to ALR position

beforeFirst()

first()

last()

isFirst() --> Checks whether record pointer is there in first record or not.

isLast() --> Checks whether cursor/record pointer is there in last record or not.

relative(+/- n) ----->

Absolute (+/-n)----->

Q) What is the difference b/w relative method & absolute method?

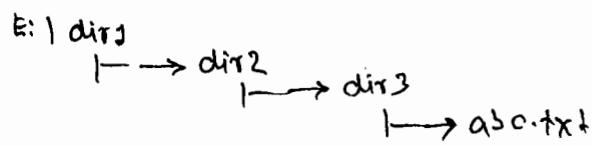
A) { Relative method moves the cursor in resultSet
the no. indicates forward direction & -ve no. indicates reverse direction. }

{ Absolute method moves the cursor & resultSet w.r.t.
BFR or ALR position.
the no. indicates forward direction w.r.t. BFR. -ve no
indicates reverse direction w.r.t. ALR }

/*Satyam Computer → Ramalingam Raju*/

Q) What is the difference b/w absolute path & relative path?

A) If file is located right from its root folder then it is called absolute path. If file is located w.r.t. current position / location then it is called relative path.



case1: (from e:\dir1\dir2)

absolut

Absolute path of abc.txt : E:\dir1\dir2\dir3\abc.txt

relative path of abc.txt : ..\dir3\abc.txt

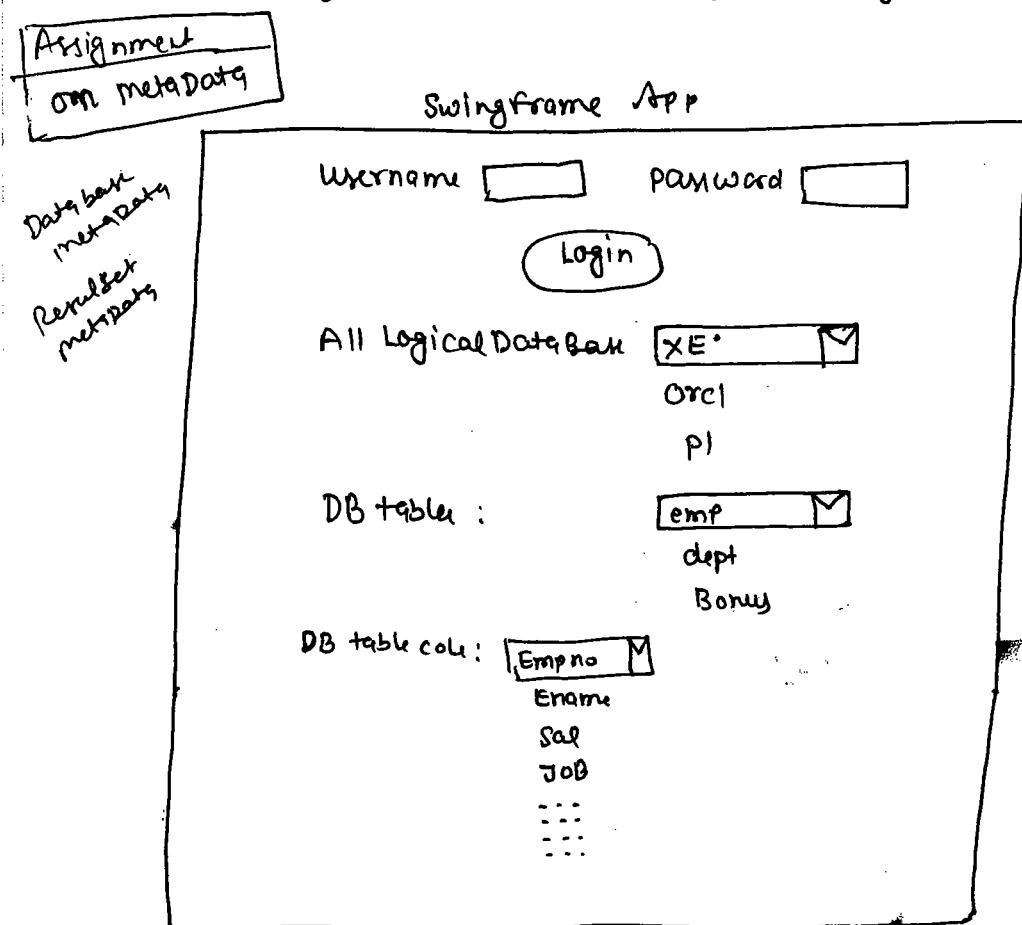
case2: (from e:\dir1\txt)

Absolute path of abc.txt : E:\dir2\dir1\dir3\abc.txt

relative path of abc.txt : ..\dir2\dir1\abc.txt

Write a JDBC app that allows us to access the records
by directionally & randomly.

→ For this we need ScrollableResultSet object.



↳ Refer Booklet (app 26 pg n: 47)

- Transaction Management
- Batch Processing
- Working with JDBC 3 & 4 drivers

(34)
18/02/15

ScrollTest.java

↳ Refer Booklet (Appn 1D of Page no 40)

How can we prepare Scrollable ResultSet obj by using JDBC Prepared Statement Obj?

// Create Prepared Statement Obj

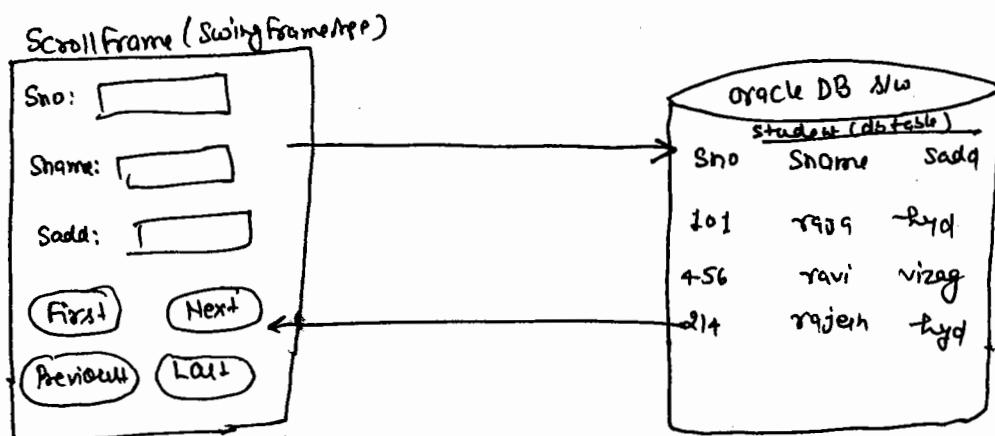
PreparedStatement ps = con.prepareStatement ("Select * from student",

ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_READ_ONLY);

// Send execute SQL Query

ResultSet rs = ps.executeQuery();

GUI Front End Based Example Appn on Scrollable ResultSet Object.



In the above option record should be displayed based on the button that is clicked on dynamically. For this we need to move cursor in ResultSet obj randomly so use scrollable resultSet obj.

* Since there is no next record for last record we must ensure that the record pointer is not already in their in last record before calling rs.next(). For this we can use

`rs.isLast()` method.

Since there is not previous record for first record so we must ensure that the cursor is not their in first record before calling rs.previous() method. This can be done by using

`rs.isFirst()` method

Containers: Swing frame

Components: JTextField : 3

JLabel : 3

JButton : 4

Layout manager: Flow Layout

Event : Action Listener Event

Event Listener : ActionListener

Event handle method : actionPerformed (-)

// Scrollframe.java

↳ Refer Booklet Appn 19 page no 40, 43)

(35)

| 18/02/15 |

NetBeans:-

Type: IDE for Java env..

Version: 8.x (compatible with Java 1.6+) Appn

Vendor: Sun M's (Oracle Corp)

OpenSource IDE

To Download Slw: www.netbeans.org

Procedure to develop Scrollframe Based GUI Appn (previous class
by using NetBeans
IDE)

Step1) Launch NetBeans IDE & create Java Project

File → New Project → Java - TestProj (name of project)

Step2) Add ojdbc14.jar / odbc6.jar to the libraries folder of
Project

Step3) Develop the Appn as JFrame App

R.C. on source package → New JFrame

ClassName: Scrollframe

package: com.int

Step4) Design the appn (use drag & drop option)

Step5) Add ActionListener on buttons to handle action events

R.C. → Button → Event → Action → ActionPerformed

Step6) Declare the following instance variables with the
source code

Connection con;

PreparedStatement ps;

- Step7) Develop user defined method having logic to create ScrollableResultSet obj & call that method from constructor.

```
public void makeConnection() {
    try {
        Class.forName("oracle.jdbc.driver.OracleDriver");
        Con = DriverManager.getConnection("jdbc:oracle:thin:@"
            + "localhost:1521:scott", "tiger");
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

ps = Con.prepareStatement("Select * from student",
 ResultSetType.SCROLL_SENSITIVE,
 ResultSet.CONCUR_READ_ONLY);

rs = ps.executeQuery();

} Catch (Exception e) {

e.printStackTrace();
}

} //make Connection

```
public ScrollFrame() {
    initComponents();
    makeConnection();
}
```

initComponents();
makeConnection();

}

- Step8) write following logic in JButton1 ActionPerformed(..) for
"first" button

```

private void jButton1ActionPerformed( ) {
    try {
        rs.first();
        *1 JTextfield1.setText(rs.getString(1));
        " 2 " (2);
        " 3 " (3);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

```

Step 9) write following code in JButton2ActionPerformed(-) for last button

```

{ try {
    rs.last();
    *1
} catch (Exception e) {
    e.printStackTrace();
}

```

Step 10) write following logic in JButton3ActionPerformed(-) for Next button

```

try {
    if (!rs.isLast())
    {
        rs.next();
    }
}

```

Step 11) Write following logic in JButton4ActionPerformed(-) for previous button

```

try {
    if (!rs.isFirst())
    {
        rs.previous();
    }
}

```

Step 12) Run the apph.

Non Scrollable Resultset Objects are also called as forward only Resultset Object because it allows us to access the records only in one direction i.e. Top to Bottom

A ScrollableResultset obj can also be taken as

- a) Sensitive ResultSet
 - b) InSensitive ResultSet
 - c) Updateable ResultSet
 - d) Readonly ResultSet

What is difference b/w Sensitive Result set & In sensitive Result set?

When Resultset obj is representing Db table, if any modification, done in Db table records are reflecting to Resultset obj then it is called sensitive Resultset.

If modification are not reflecting often it is called
Insensitive Results.

To create Sensitive ResultSet -

Statement `st = Con.createStatement (ResultSet.TYPE_SCROLL_SENSITIVE,`
 `ResultSet.CONCUR_READ_ONLY)`

ResultSet rs = st.executeQuery ("Select * from Student");

To Create Incentive Results

Statement st = Conn.createStatement (ResultSet.TYPE_SCROLL_INSENSITIVE,

Resultset rs = st.executeQuery ("Select * from student");

Example app on Sensitive / InSensitive Resultset.

⇒ refer booklet.

All jdbc drivers support Scrollable Resultset.

Type2 driver supports Sensitive Resultset but doesn't support InSensitive Resultset

Oracle Thin driver (oci drivers) support InSensitive Resultset directly but to get the support of Sensitive Resultset we must do following things: —

a) Replace * symbol with col name in SQL query.

Select * from student ⇒ select sno, sname, saddr from student.

b) Call rs.refreshRow() method before, processing each record

while(rs.next()) {

 rs.refreshRow();

\dots

}

While developing the app which display the Scott market share value, live game score we need to go for SENSITIVE-RESULT SET.

36

20/02/15

Q) What is the difference between Sensitive Resultset & Updatable Resultset?

A) If the modifications done in Resultset are reflecting to DB table then it is called updatable Resultset. Using this Resultset we can perform non-select operations on DB table without using SQL Queries.

If modifications done in DB table are reflecting to Resultset Object then it is called as Sensitive Resultset Obj

Q) What is difference b/w Readonly and Updatable Resultset?

A) Readonly Resultset allows only "select" operations on DB table. Updatable Resultset allows both select & non select operations on DB table.

To create Readonly Resultset:-

Statement `st = con.createStatement (Resultset.TYPE_SCROLL_SENSITIVE,
Resultset.CONCUR_READ_ONLY);`

Resultset `rs = st.executeQuery ("Select * from student");`

To create Updatable Resultset:-

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

Resultset `rs = st.executeQuery ("Select * from student");`

To Select records from updateable Resultset :-

```
while( rs.next() ) {  
    System.out.println( rs.getInt(1) + " ~ " + ... + );  
}
```

To insert record:-

```
rs.moveToInsertRow(); ---> create empty row in ResultSet  
rs.updateInt(1, 100); --->  
rs.updateString(2, "reject"); ---> set value to empty row  
rs.updateString(3, "Tyd"); --->  
rs.insertRow(); ---> Inserts the record in db table  
rs.refreshRow(); ---> Required only while working oracle  
driver
```

To delete record:-

```
rs.absolute(4); ---> points to 4th record  
rs.deleteRow(); ---> deletes 4th record
```

To Update record:-

```
rs.absolute(3);  
rs.updateString(3, "Tyd1"); // modify 3rd col value of 3rd record  
rs.updateRow();  
rs.refreshRow(); *
```

All jdbc drivers supports ~~READONLY~~ Resultset.

Type-1 drivers support updateable Resultset.

ORACLE OCI, THIN, drivers support UPDATEABLE Resultset
Only when A * (star symbol) in SQL query is replaced
with column names.

b) When `re-refreshRows()` method is called.

For Example App on Updatable ResultSet

↳ ~~ResultSet~~ Refer Booklet App 21 of P. 16 - 36

It is not recommended to use Updatable ResultSet object bcoz of the following reason:-

- a) Performing Bulk non select operations is complex.
- b) Performing Criteria based non select operation is complex.
- c)

/* This: Could not resolve the connect identifier specified */

Batch Processing :- / Batch Updation:-

Batch Processing

Instead of sending multiple related queries one by one to DB S/w for execution by using network for multiple times. It is recommended to place all those queries in single batch/unit and send that batch/unit to db s/w to execute queries only by using the network only for one time. This is called batch processing or batch updation.

In batch processing queries go to db s/w via a batch and they will be executed in DB S/w individually and the query results come back to java app by a batch. This reduces network round trip b/w Java App & DB S/w.

The queries of batch processing will not be executed by applying "do everything or nothing principle" that means if one or other query execution failed

the remaining query will be executed.

Sample Code:-

```
Statement st = con.createStatement();
// add query to the Batch
st.addBatch("insert into student values(101,'raja','hyd')");
st.addBatch("update student set saddr='vizag' where sno=110");
st.addBatch("delete from student where saddr='delhi'");
.... // Like this we can add any no. of query to batch
```

// send & execute the batch of query in DB now

```
int res[] = st.executeBatch();
```

res
1
1
3

// Process the result

```
int sum=0;
for( int i=0; i<res.length; i++ ){
    sum = sum+res[i];
}
```

System.out.println("no of records that are effected is:" +sum);

a) why we can't add select query to the batch of
Batch Process?

b) Select query execution gives ResultSet.

executeBatch() returns int array. So we can't place
this ResultSet in the int array. This indicates Select
query can't be added to the batch of batch
processing.

Example Appn on Batch Processing:-

↳ Refer Booklet App 31 of P.M.O - 49

The queries added to the batch can perform operations on one table or multiple tables.

The queries added to the batch can be same type of SQL queries.

All JDBC drivers support Batch Processing

37
+-----+
21/02/15

Q) Can you explain different execute() methods of JDBC program

A) executeUpdate() ---> for executing non-Select SQL Query

executeQuery() ---> for executing Select SQL Query

execute() ---> To call PL/SQL procedure or function DB SQL

executeBatch() ---> To perform batch processing

Transaction Management:-

The process of related operations into single unit & executing them by applying do everything or nothing principle is called Transaction management.

To enable Transaction management.

a) Disable auto Commit mode on DB S/W (begin of Transaction).

b) Execute con.setAutoCommit(false);

b) Execute the queries of batch.

Statement st = con.createStatement();

```
st.addBatch("-----");
st.addBatch(".....");
int res[] = st.executeBatch();
```

c) process the result to enable Transaction Management

```
boolean flag = false;
for( int i=0; i<res.length; ++i)
{
    if ( res[i] == 0) {
        flag = true;
        break;
    }
    if if (flag == true)
        con.rollback(); // Rolls back the Transaction (TCL)
    else
        con.commit(); // commits the Transaction (TCL)
```

Every Transaction management contains the following code:-

- a) Begin Transaction
 - b) Execute the logic
:::
 - c) Commit or Rollback the transaction
- Q) Why Commit, Rollback() methods are given on connection object to invoke().
- A) This allows to commit() or Rollback() SQL query execution that is taken place on multiple statement objects.

By default batch processing doesn't support transaction management but by writing additional logics on resultset

of batch processing we need to go for transaction management.

Note While developing the appn which deal with sensitive logics like transfer money we generally enable transaction management.

Transfer money is the combination of two operations -

- a) withdraw money from source Account.
- b) Deposit money into destination Account.

We need to enable do every thing or nothing principle while executing these logics.

Develop the example appn before transfer money operation having transaction management Supply support.

⇒ Refer Booklet (page no: 50 & 51)

By default auto commit mode is on/enabled on DB. Now to disable this we use Con.setAutoCommit(false).

To commit the query execution results that are executed after disabling autoCommit() mode we need to use Con.commit().

To rollback same queries we need to use Con.rollback(). method.

Batch Processing using PreparedStatement obj

In simple statement based batch processing we can add multiple queries to the batch.

In prepared statement based batch processing we can't add multiple queries to the batch because this object always can bound with one query.

In prepared Statement

We can add multiple set of query param value to the batch. All these values go to DB S/W only for one time at a batch. This reduces the network round trip.

PreparedStatement ps = con.prepareStatement ("insert into student
values(?, ?, ?)");

ps.setInt (1, 201);

ps.setString (2, "Raja");

ps.setString (3, "Hyd");

ps.addBatch();

ps.setInt (1, 202);

ps.setString (2, "Ravi");

ps.setString (3, "Vizag");

ps.addBatch();

int ref[] = ps.executeBatch();

⇒ Refer Booklet

38
22/02/15

Properties:-

The text file that maintain entries in the form of key = value pair is called properties file.

e.g.

myfile.properties

User = scott

pwd = tiger

- * The standard principle of SW industry is do not hard code any value in the application that are changeable in future. It is always recommended to get those values from outside the app in the form of properties file.
- * Since jdbc properties (like username, password, driver, url) and etc. will be change based on driver & DB SW we use it is recommended to deliver jdbc app to client organization by having properties file so that they can change details through properties file without disturbing the source code.
- * `java.util.Properties` is the sub class of `java.util.Map`. This map collection has the ability to get the keys value of element from text properties file using `load` method.

Filestream fs = new FileInputStream("myfile.properties")

Properties p = new Properties();

p (`java.util.Properties` class obj)

username	scott
password	tiger

(Key) (Value) (String)

Example Appⁿ that makes our jdbc code as flexible code
to modify using property file support.

Prefer Appⁿ → 33 page no = 52, 53

- type: DB SW
- vendor: Enterprise DB
- version: 9x
- open source
- port number: 5432
- default username: PostGre
- default password should be chosen during installation
- To download SW: www.postgresql.org
- Default Logical DB: Postgre

Procedure to create logical DB in PostgreSQL having DB table with records :-

- 1) Launch Postgre Admin tool & connect to PostgreSQL DB SW
Start → All programs → postgresql → PgAdmin3 → PostgreSQL → Connect
- 2) Create logical DB

Pg Admin tool → PostgreSQL → DB → rc → newDB
name → ntba
↳ ok

- 3) Create table and insert Record

ntba
↳ Schema
↳ public
↳ table → rc → newtable

Name: product

→ common table

Name: id

Data type: integer

→ Add

name: product name

Datatype: varchar

→ Add

name: quantity

Datatype: integer

④ insert record in Db table product

task

↳ rc. on product

↳ scripts

↳ insert script

insert into product (pid, pname, quantity) values(10, ?, ?)

↓ ↓ ↓
10 Ross 20

↳ execute

⇒ Type + mechanism based jdbc driver for postgres SQL is called Postgres SQL thin driver the details are -

Driver class name: org.postgresql.Driver

URL: jdbc:postgresql:<logical db>

or (while working with local PostgreSQL SQL SW)

jdbc:postgresql://<host>:<port>/logical db (while working with remote PostgreSQL SQL SW)

JAR file: postgresql-8.4-701.jdbc4.jar

Example App to interact with PostgreSQL DB SW:-

Step1) Add jar file to build path (PostgreSQL - 8.4-701.jdbc4.jar)

Step2) Driver details PostgreSQL SQL thin driver.

Step3) develop the app → App no - 36, Page → 57

For various versions of jdbc with their features -

Ref Pg no 1, A2 of booklet

The jdbc api versions are -

JDBC 1.0, 1.2, 2.0, 2.1, 3.0, 4.0

The important two features of JDBC 3.0 are -

- Working with client side JDBC con pool.
- Savepoints,

The imp feature JDBC 4.0 is auto-loading of JDBC driver class.

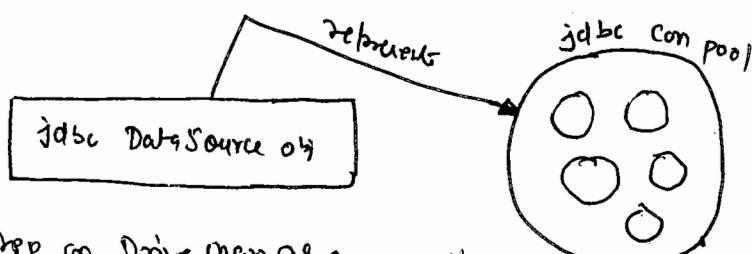
Working with Client Side Connection pool:-

There are two types of JDBC con pools

- Client side JDBC con pool (will be managed by JDBC driver)
- Server side JDBC con pool (will be managed by webserver/app server)

JDBC DataSource obj represents JDBC con pool. Each JDBC con obj of JDBC Con Pool will be retrieved through DataSource obj.

JDBC DataSource obj means it is the obj of underlying JDBC driver supplied java class that implements javax.sql.DataSource()



Examp on DriverManager connection

app 34

* How many types of jdbc connections are there.

A) Two type -

1) Direct Connection obj

2) Pooled Connection object

* ~~How many~~ ~~types~~ of jdbc connection ~~are~~ con obj that are gathered from jdbc conn pool
create by programmer manually

Class.forName("....")

Connection con = DriverManager.getConnection(..., ...);

→ Direct jdbc con obj

Connection con = ds.getConnection();

B) What is Savepoint?

A) Savepoint:-

The normal end result of transaction is commit() or rollback().

If we are looking both commit & rollback for the obj's of transaction we need to create save points.

Save point is a logical point in a transaction upto which we can rollback

There are two types of savepoints -

a) named Savepoint (Appn gives explicit name to it)

b) Unnamed Savepoint (The underlying db also gives name to it)

Begin Transaction

operⁿ1

operⁿ2

Save point p1

operⁿ4

operⁿ5

rollback upto p1

commit transaction

} Note: here operⁿ1, 2 will be committed & operⁿ4, 5 will be rollback.

while performing transaction management in online shopping environment having the option of purchasing some items & rejecting some items of selected items, we need to go for savepoint.

To Create Named Save Point

```
Con.createStatement("mySP");
SavePoint SP = Con.Statement.SavePoint("mySP");
```

To rollback upto SavePoint,

```
Con.rollback(SP)
```

Note There is no con.commit() that takes savepoint.

Note:- SavePoint obj means it is the obj of java class that implements java.sql.SavePoint(I).

For Example App on Savepoint

↳ refer booklet appno 35 Pages - 53, 54

* jdbc 4.0 auto loading of driver class feature -

⇒ The DriverManager class loads & registers jdbc driver automatically at its initialization by collecting driver class name from java.sql.Driver file in META-INF\services folder belonging to the jar files of jdbc driver.

⇒ Ojdbc6.jar given by oracle 11g contains

META-INF\services\java.sql.Driver file having

"oracle.jdbc.OracleDriver" as the connect context so it

supports auto loading of driver class

Ojdbc14.jar given by oracle 10g doesn't contain above file so it doesn't support auto loading of driver class.

* jar file PostgresOL jdbc driver also contains java.sql.Driver
file supporting autoload of jdbc driver class.

while working with this feature there is no need of
placing "Class.forName()" method in our jdbc App.

Example App.

- a) Addojdbc6.jar file to classpath or build path
- b) Run any jdbc App without Class.forName() method call.

4
09/04/15

5pm to 7pm

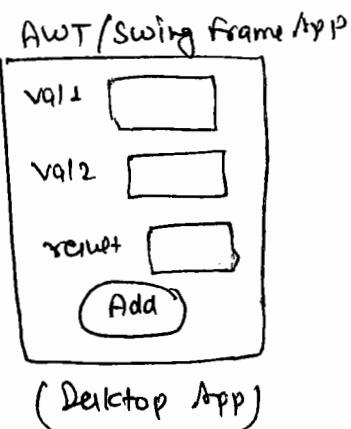
SERVLET

Adv Java :-

1. JDBC + misc topics (pre-requisite: core java oops)
2. Servlet + misc topics (pre-requisite: core java oops)
3. JSP + mini Project + misc topics (pre-requisite: servlet)

- ⇒ Using Java we can develop different type of Apps like standalone Apps, two tier Apps, web App and etc.
- ⇒ The App that contains only one layer & specific to one computer is called Standalone App.
- ⇒ There are two types of Standalone Apps
 - a) Console Apps (CUI Apps)
 - b) GUI Apps (Desktop Apps)
- ⇒ Java class with main() is console App.
- ⇒ Awt/Swing based Frame Apps are called Desktop Apps.

```
public class TestApp {
    public static void main() {
        -- -- --
    }
}
```



- ⇒ The data & logics (code) of stand alone Apps are specific to one computer.
- ⇒ The App that contains two layers or progs talking with each other is two tier App

e.g. Java App talking with DB s/w

Client Java App talking with Server App

- ⇒ In two-tier Apps the data, logics are specific one network where Appⁿ is running. (LAN: Local Area Network)
- ⇒ To make data, logics as more visible and accessible logics we need to take the support of internet network.
- ⇒ We can't run standalone, two-tier App directly in the internet network ... For that we need to develop web apps or website to run on internet network.
- ⇒ To make our app data, logics visible and accessible globally providing 24/7 access we need use web application / website in internet env. - .
- ⇒ To develop the web application we can use -
 - a) Java (for large scale) (more than 30 webpage with security)
 - b) .NET (for medium scale) (up to 30 web page)
 - c) PHP (for small scale) (up to 10 web page)
 - d) CGI (x)
 - ⇒ webapplication
 - ⇒ web resource prg / Web Comp
 - ⇒ Web page
 - a) Static web page / Plain web page
 - b) Dynamic web page / Active web page.

Every web application collection of web resource progs having capability to generate web page -

⇒ There are two types of web page -

a) Static web page / Passive web page

⇒ Generate same content for all requests

E.g. About us. page, Term & Condition page

b) Dynamic web page / Active web page

⇒ The content of webpage will change based on time of request generation and based on the time i/p value of req

E.g. Gmail login page, inbox type

⇒ Internet H/w is WAN and it looks like spider's web
so it is called as web h/w

⇒ There are two types of web resource progs

a) Static web resource progs / comp

⇒ Generate static web page

E.g. html progs

b) Dynamic web resource progs / comp

⇒ Dynamic web page

E.g. Servlet progs, JSP, Progs, asp.net page, PHP progs

⇒ The technologies that can be used to develop web application / web resource

Progs are called web technologies.

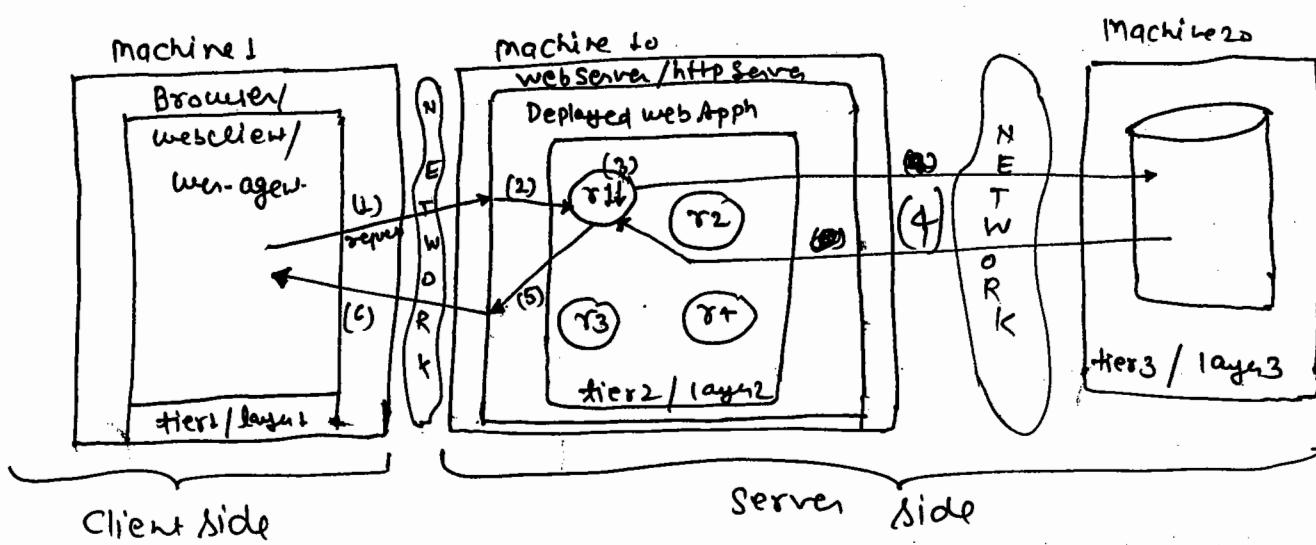
E.g. Servlet, JSP, html, CSS, JavaScript, jquery, asp.net
PHP & etc.

(2)
10/04/15

5pm to 7 pm

- ⇒ Normal java Appⁿ will be executed manually from command prompt whenever there is a need.
- ⇒ The web resource program of web appⁿ will be executed by web server s/w will be created dynamically. whenever request are given to those web resource program from client (browser window),
- ⇒ This indicates the web resource prog will not be ~~executed~~ executed manually, they will be created dynamically in web server s/w (like tomcat).
- ⇒ WebServer s/w automates web applⁿ & its web resource execution it
 - listens to client request → parses them web resource prog
 - execute web resource program → gather output from web resource progs → send the output to clients (browsers) as response in the form of web page.
- ⇒ e.g. Tomcat, IIS, AWS, etc. —
 - IIS: Internet Information Server
 - Aws: Apache web server.
- ⇒ The process of keeping developed by web application in web server is called Deployment.
- ⇒ The process of removing web appⁿ from webserver is called Undeployment.

Understanding web application setup:-



w.r.t. Diagram-

- 1) Browser gives request to web Appⁿ
 - 2) Webserver that listens client request continuously takes the request & process the request to appropriate web resource prog (like servlet prog)
 - 3) The web resource program \neq in web appⁿ executes dynamically
 - 4) if needed , the web resource program interacts with DB via
 - 5) The output generated by web resource prog goes to web server
 - 6) Web server sends output to browser as response in the form of web page.
- ⇒ In one web appⁿ we can have one or more web resource progs.
- ⇒ In one web appⁿ we can deploy one or more web appⁿ (websites)

- ⇒ We can develop web appn or 2 tier application without DB S/w. (or) As 3 tier application with DB S/w.
- ⇒ When web appn is developed as 2 tier appn it looks like thin client - fat server Appn.

(3)
13/04/15

5pm to 7pm

→ Based on the content they generate there are two types of web resource

- Static web resource
(Generate static web page)
- Dynamic web resource
(Generate Dynamic webpage)

Based on the place where web resource progs execute there are two types of web resource progs:-

- Server side web resource progs
→ These progs execute in web serve it self after request is given.
e.g. Servlet, JSP progs etc.
- Client side web resource progs.
→ These progs go to browser window from web server for execution when requested.
e.g. HTML progs, Java script progs and etc.

Note- Generally all Dynamic web resource progs are serving 1 for web resource progs and static web resource progs are client side web resource progs

Note: Decide whether web resource progs are server side or not based on the location where they execute, not based on the location where they reside.

List of browser sw:-

IE → from ms
Navigator → Netscape
Safari → Apple
Firefox → Mozilla
HotJava → RedHat
Chrome → Google (+)
Opera → from Opera soft
--
--
and etc...

List of Server side web technologies (To develop server side web resources)

Servlet → from Sun My (oracle corp) (Java) } Java
JSP → "
asp → from Microsoft
asp.net → "
asp.net MVC → "
SSJS (server side JavaScript) → from Netscape
PHP → from Apache
ColdFusion → from Adobe } Non-Java

List of Client side technologies (To develop client side web resources)

HTML → W3C (World wide web Consortium) → Non-profit organization called Consortium.
JavaScript → from Netscape & Sun Microsystems
VB Script → from Microsoft
Ajax → from Adaptive Path

jquery

DOJO

AngularJS

and etc —

List of WebServer S/w's

Tomcat --> from Apache (Java) (1)

IWS --> from Sun MS (Java)

Resin --> from Rain Soft Java (Java) (4)

Jetty --> from Adobe (Java)

IIS --> from MS (non-Java) (2)

PWS --> from MS (non-Java)

AWS --> from Apache (non-Java) (3)

NFTS (Netscape Fault Track Server) --> from Netscape

and etc —

List of Application Server S/w's:-

⇒ It is enhancement of web server S/w.

weblogic --> from BEA Systems (Oracle Corp) (1)

websphere --> from IBM (3)

Glassfish --> from SUN MS (2)

Jboss --> from Apache (Red Hat) (+)

Oracle GlassFish --> from Oracle Corp

Jrun --> from Adobe

PAS --> from Pragmati Soft

(PAS: Pragmati Application Server)

List of DB S/w's.

oracle --> from Oracle Corp

DB2 --> from IBM

SQLserver --> from MS

PostgreSQL --> Enterprise DB

mysql --> Devlop (Sunny) (Oracle Corp)

Note:
Add one plugin to IIS
then Servlet program can be
executed in IIS.

(4)
14/04/2015

5pm to 7pm

- ⇒ To execute Applet program we need applet viewer or applet Container.
- ⇒ To execute HTML program we need HTML interpreter.
- ⇒ To execute servlet prg we need Servlet Container
- ⇒ To execute JSP prg we need JSP Container.
- ⇒ Both Servlet container and JSP container internally use jvm/JRE.
- ⇒ Applet Viewer also internally uses jvm/JRE.
- ⇒ Every Servlet prg is a java class having code to process request and to generate response.
- ⇒ We need not to arrange Servlet Container, JSP Container separately, they come automatically along with web server like installation (java based).

Q) What is Container?

- A) Container is a software program / application that takes care of whole life cycle of given resources (Birth to death) (Object creation to Object destruction).
- ⇒ Servlet Container that takes care of Servlet prg life cycle and JSP Container takes care of JSP prg life cycle
- ⇒ Container is like aquarium taking care of whole life cycle of given fishes (web resource prgs).

Architecture of Web Server:-

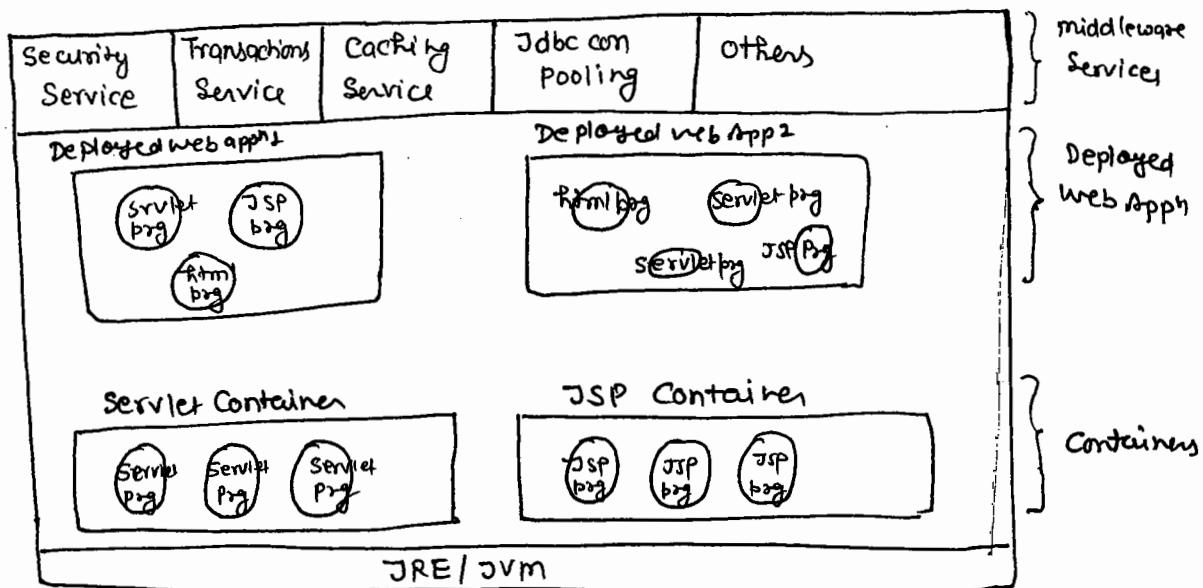


fig: High Level Architecture of Java WebServer

- ⇒ The additional, optional and configurable (allows to enable or disable) logics on our Apps to make our App more perfect are called middleware services. There are not main logic in appn development. These are additional and optional logics to apply on our appn.
- ⇒ When Servlet Prog in request it goes to Servlet Container for execution.
- ⇒ Similarly JSP Prog goes to JSP Container for execution.

JSP Container internally uses the support of Servlet Container.

Tomcat:-

Type: Java based webserver

Version: 8.x (Compatible with JDK 1.7+)
7.x (Compatible with JDK 1.6+)

Vendor: Apache

OpenSource SW (Freeware)

Default Port no: 8080

Creator: David Concurson

Servlet Container: CATALINA

JSP Container: JASPER

To download SW: apache-tomcat-7.0.8.exe
from www.apache.org

(8)
15/04/2015

5pm to 7pm

1/

Top 8 People of Java -

- ① James Gosling → Tomcat, ANTLR, JBoss
② David NcGowan, Rod Johnson, George Kawa, Graeme Melchers
③ Mark Weathers → JBoss, Kent Beck → JUnit, waterfall model & Agile methodology
④ Gavin King → Hibernate, ~~David Durbin~~

Struts

→ While installing Tomcat Server we need to supply the following details —

- JRE location <The one that is parallel to jdk>
- HTTP port no: 2525 (default 8080)
- Tomcat installation folder: D:\Tomcat 8.0 (<Tomcat_Home>)
- Administration details: username: admin
password: admin

Note → Install in other than C drive.

To Start Tomcat Server:-

Go <Tomcat_Home>\bin directory we tomcat8.exe file

→ When server is started one daemon process will be created listening to client request continuously.

To see home page of Tomcat:-

http://localhost:2525

Protocol Hostname & port no. of tomcat

Changing Port no. of tomcat After installation:-

Go to <Tomcat_Home>\conf\server.xml & modify the "port" attribute value of first <connector> tag & restart the server.

Understanding Installation Directories of Tomcat:

D:\Tomcat8.0 (<Tomcat_home>)

- ↳ bin
 - ↳ *.exe (Tomcat8.exe)
 - ↳ *.batch (startup.bat, shutdown.bat)
- ↳ conf
 - ↳ *.xml (server.xml, tomcat-user.xml)
- ↳ lib
 - ↳ *.jar (servlet-api.jar, Catalina.jar, JSP-api.jar, Jasper.jar)
- ↳ logs
 - ↳ *.txt files
- ↳ webapps
 - ↳ dirs, dir2, ...
 - ↳ *.war files
- ↳ temp
- ↳ misc files

⇒ To deploy all Java web app in Tomcat Server we can place them in <Tomcat_home>\webapps folder either in the form of directories or war files.
war file: web app archive

⇒ <Tomcat_home>\lib\servlet-api.jar represents servlet-api(pkg)
JSP-api.jar represents JSP-api (PKG)
catalina.jar represents servlet container
jasper.jar represents JSP container.

⇒ webcontainer = servlet container + JSP container,

Responsibilities of Web Server:

- ⇒ Take requests from clients by listening to client request continuously
(for this it starts Daemon process)
- ⇒ Pass the requests container for request processing
- ⇒ Provides containers like (Servlet/JSP containers) to execute Servlet side pages.
- ⇒ Provides middleware services (like security)
- ⇒ Provides env. to automat the web app & its web resource prgs. execution,
- ⇒ Provides env to deploy, undeploy, start, stop and reload web apps.
- ⇒ Takes the results from containers & send them to browser window, as response.
- ⇒ Sends the code of client side web resource page to browser window for execution and etc..

Responsibilities Container:

- ⇒ Takes the request from web server & execute server side web resource prg dynamically.
- ⇒ Takes care of server side prg life cycle (Birth to death)
- ⇒ Takes care of communication b/w web resource prg.
- ⇒ Perform various activities related to Request handling & response generation.
and etc..

⑥
16/04/2015

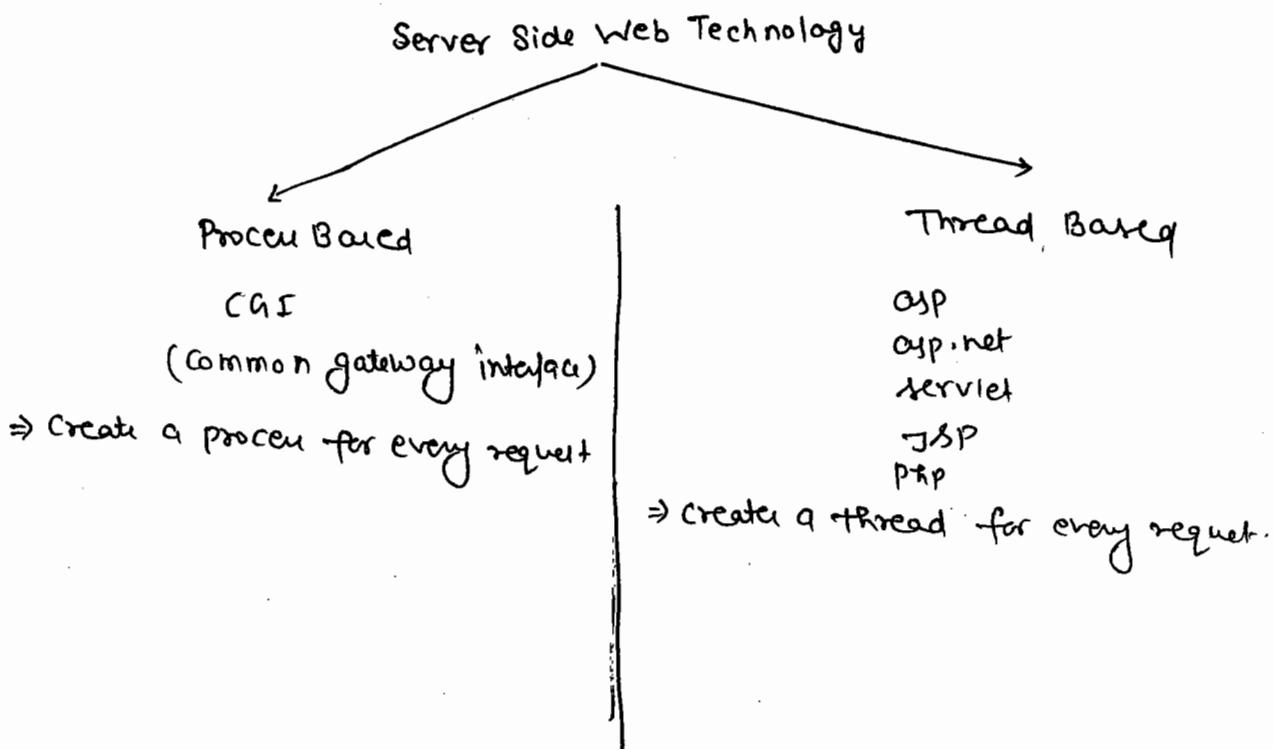
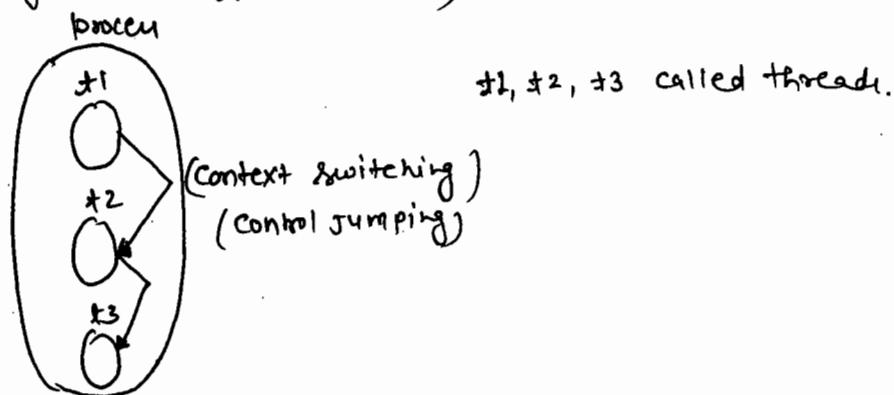
from 5pm to 7pm

- ⇒ Servlet, JSP, EJB, JMS and etc... are JEE module technologies.
- ⇒ Technologies are not installable SW b'coz they just rule and guide line to develop SW's. These SW are installable Working with these SW is nothing but working with technologies.
- ⇒ Web Server/ Application Server SWs are given based on JEE technologies. Installing these servers SW and working with them server SWs is nothing but working with JEE module.
- ⇒ e.g. Installing tomcat, weblogic SW & many more in nothing but JEE module....
- ⇒ Based on the technologies we use there are following containers—
 - a) Servlet Container → Based on Servlet Technology
 - b) JSP Container → Based on JSP Technology
 - c) EJB Container → Based on EJB technology
- Based on physical existence of Container with respect to server there are 3 types of container—
 - a) Standalone Container
 - Here Server and Container together comes as single piece of program.
 - b) Inprocess Container
 - Here Container inside the Server as separate piece of code Servlet Container/ JSP Container of Tomcat Server is called Inprocess Container.
 - c) Outprocess Container
 - Here container reside outside server but will be attached to server externally the servlet/ JSP

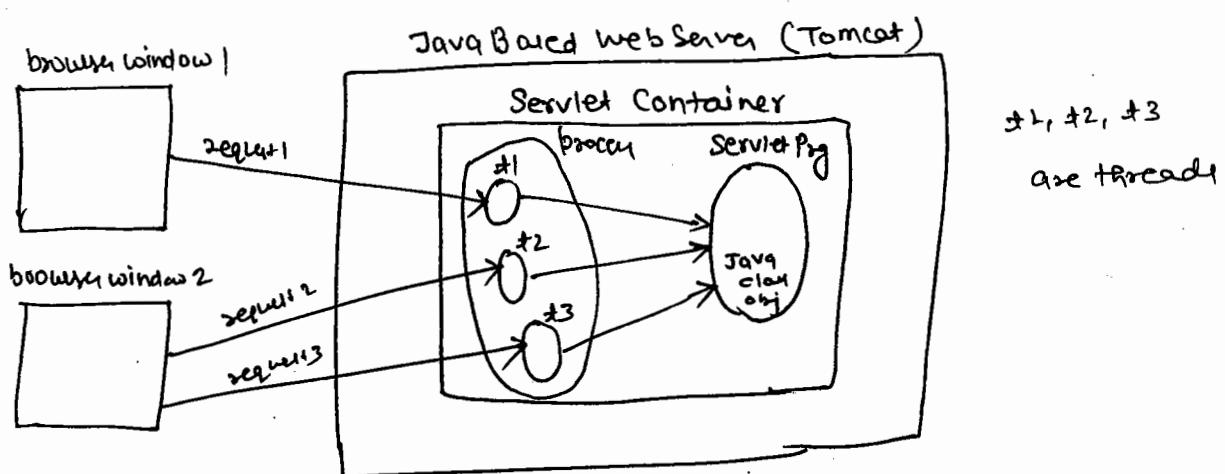
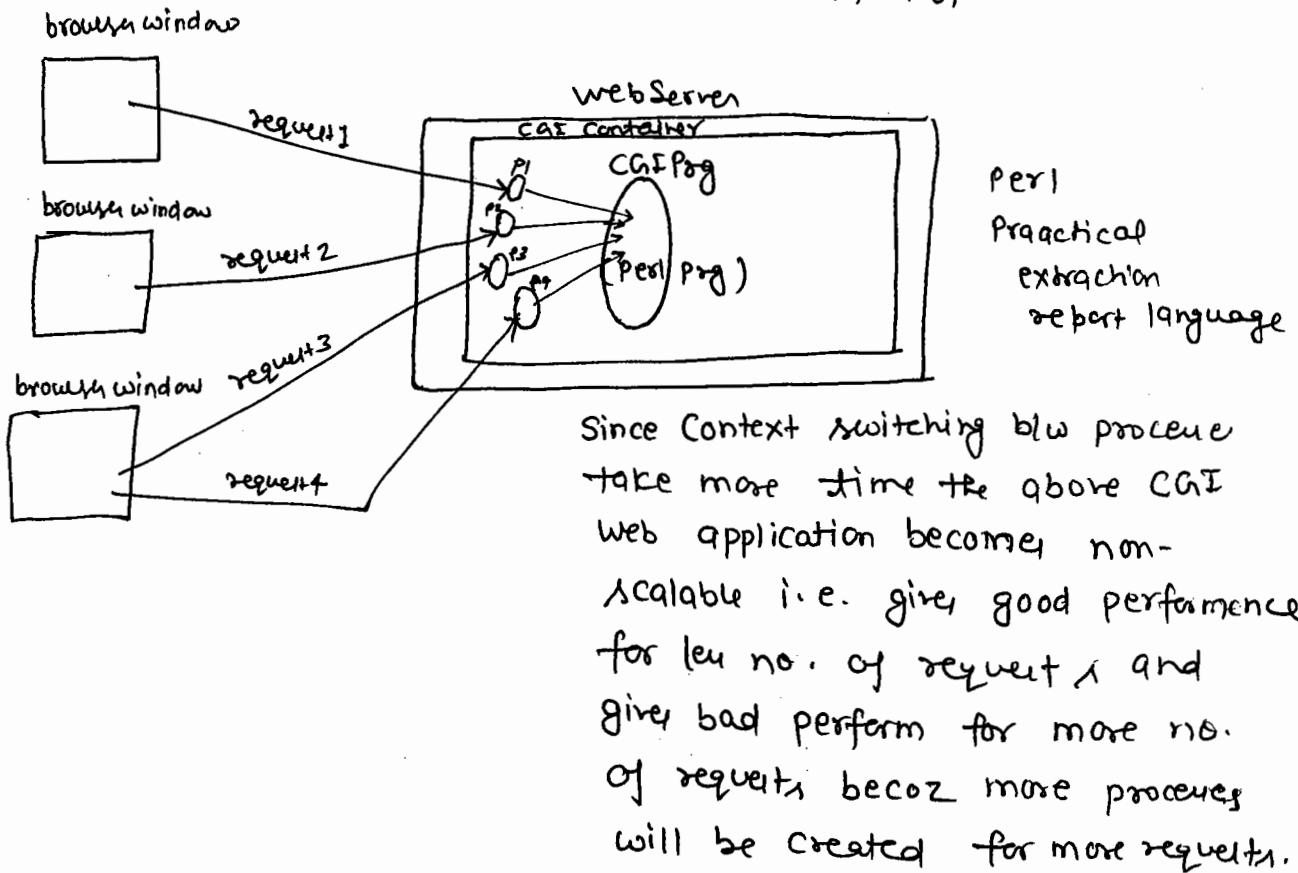
Container attached with IFS externally is called OutProcess Container.

Thread:-

- ⇒ Thread is a subprocess or lightweight process.
- ⇒ Transferring control b/w processes or threads through scheduling is called context switching or control jumping. This takes more time for processes and less time for threads.
- ⇒ When we start the execution of java Appn one process will be started having two default threads (main thread, garbage collector thread).



P₁, P₂, P₃, P₄ are process.



⇒ If multiple request are given to servlet prog (Thread based server side technology) then servlet container creates only one object of our servlet prog class but starts multiple threads on that object for multiple request. Since control jumping b/w threads takes less time the performance of web appn will be good even though no. of request is increased or decreased. This makes web appn as Scalable.

(7)
17/04/2015

5pm to 7pm

Definitions Of Servlets:-

- 1) Servlet is a Java based server side web technology to develop dynamic web resource program / comps having the capability to generate dynamic web pages.
- 2) Servlet is Java based web technology whose web resource prgs can extend the functionality of web server or http server.
- 3) ~~Servlet is Java based web technology that allows to develop "single instance (object) multiple threads based server side web comps" in Java web application having the capability to generate dynamic web pages.~~
(Refer previous diagram)
- 4) Servlet is JEE module technology / specification that gives API for vendor companies by rules and guidelines to develop Servlet Container API (part of Webserver API) and also gives same API for programmers to develop server side / dynamic web resource prg in Java Web appn.

Note- Every Servlet class is Java class with Servlet API. The whole life cycle this Java class will be taken care by ServletContainer (Birth (obj creation) to Death (obj destruction));

For Features of Servlet refer page no. 80 to 82

For Basics of Servlet Container refer pg. no. 76 to 82.

⇒ Java Apps are WORA (Write Once Run anywhere (any platform));

⇒ Java Apps are WODA (Write Once Deploy Anywhere (any Java Server));

Every Technology gives one API. The Servlet Technology API versions, are Servlet 2.3, 2.4, 2.5, 3.0, 3.1.

industry
standard

⇒ In Java API comes in the form of packages having classes, interfaces, enum (special class), annotation (special class).

⇒ Servlet API 2.5 pkgs:-

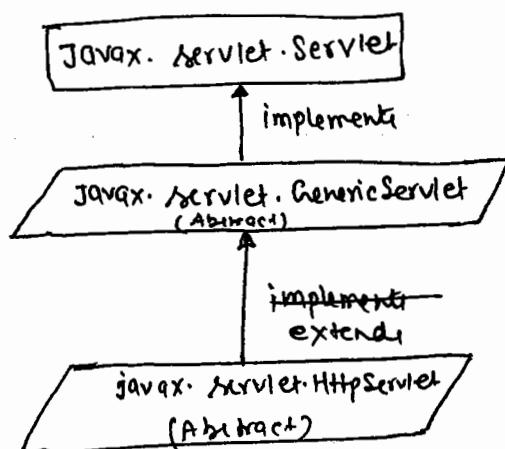
javax.servlet, javax.servlet.http, ~~javax.servlet.annotation~~

⇒ Servlet API 3.x pkgs:-

javax.servlet, javax.servlet.http, javax.servlet.annotation,
javax.servlet.descriptor; ~~javax.servlet.annotation~~

⇒ In Tomcat Server <Tomcat_Home>\lib\Servlet-api.jar file represents Servlet API.

3 important resources of Servlet API -



void init(ServletConfig)
void destroy()
void service(., .)
ServletConfig getServletConfig()
String getServletName()

Init(ServletConfig)
init()
== ==
(Does not implement service() method)

Protected doxxx(.)
Protected service(., .)
Public service(., .)
(All methods are concrete methods)

⑧
20/04/25

5pm to 7pm

There are 3 ways to develop servlet comp / prog -

Approach 1:-

Take a class implementing javax.servlet.Servlet (I) and provide implementation for all the 5 methods of that interface.

Limitation:-

Providing implementation to 5 methods even though we are not interested to implement all the method.

In our servlet class we place logic to process request and logic to generate response in the service (-,-) method.
This response goes to browser window or webpage.

Approach 2:-

Take class extending from javax.servlet.GenericServlet (AC) and provide implementation for service (-,-) method.

Limitation:-

We can't work with all the features of protocol http because GenericServlet (AC) is not specific any protocol.

GenericServlet allows to work with all protocols but doesn't allow to use any protocol for 100%.

Approach 3:-

Take a class extending from javax.servlet.http.HttpServlet and override one of the 7 doXXX(-) methods and one of two service (-,-) method.

Note:-

This best approach to develop servlet prog becoz it allows us to use all features of protocol http.

Servlet Container recognizes java class as Servlet prog only when it implements javax.servlet.Servlet (I) directly or indirectly.

- Q) How many types of Servlets are there?
- A) This is a possibility of developing "n" types of servlet like HttpServlet, ftpServlet, SmtpServlet and etc.. But we prefer developing servlet as HttpServlet becoz entire web/ internet and servers are designed based on protocol HttpServlet.

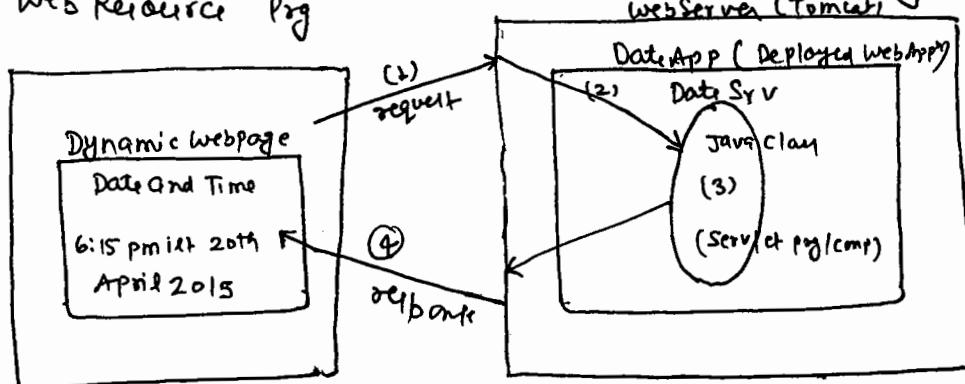
GenericServlet is not separate type of servlet. It is a common super class multiple protocol specific servlet class. As of now there is only sub class for GenericServlet that is "HttpServlet".

WebApplication is collection of web resource prgs like servlet prg, JSP prg, HTML prg having capability to generate web pages, web app also contains some helper resources like images, audio files, video files, and etc... These prgs can't generate web pages but they help other prgs towards generating web pages.

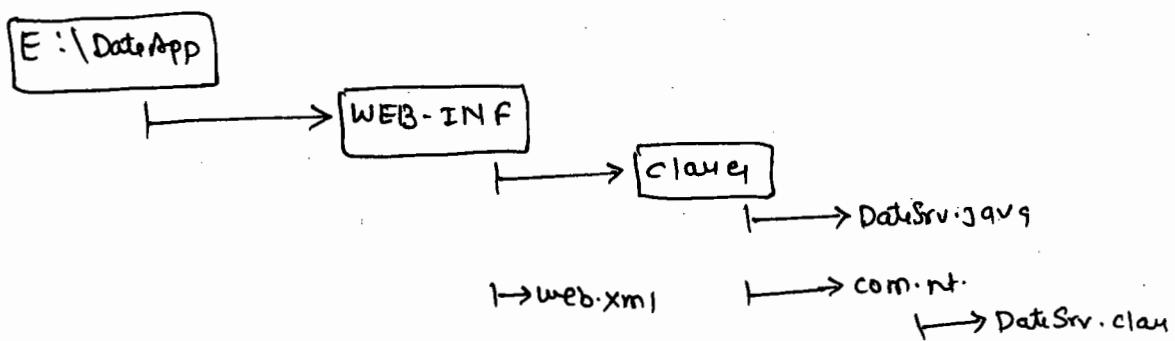
JVM begins app execution with main(). So we place main() in java app.

Servlet Container executes our servlet prg this container calls service(--) of servlet prg to process the request so we place service(--) in servlet prg having request processing logic & response generation logic.

First Java Web Application Development Having Servlet prg or Web Resource prg



Step 1) Create Deployment Directory Structure of web Application



The Deployment Directory Structure is backing mechanism of keeping web resource prog in java web application.

This Structure is common in all servers and not to specific any server. This is designed by Sun as part of Servlet Specification. So all server recognizes the directory structure based web appn deployment.

The web root folder name is user defined, After deployment this name becomes name of web appn.

Step 2:

Develop the Servlet Prg (DateSrv.java)

// DateSrv.java

```
package com.int;
import javax.servlet.*;
import java.io.*;
import java.util.*;
```

③
21/04/15

```
public class DateSrv extends GenericServlet {  
    mandatory
```

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

```
        throws ServletException, IOException {
```

```
    //set response content type
```

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

```
    //get PrintWriter Object
```

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

```
//work great for any type
```

```
    Date d = new Date();
```

```
    //write output
```

```
    pw.println("<h4> Date and Time: " + d.toString()  
              + "</h4>");
```

```
//close stream
```

```
    pw.close();
```

```
} //serve(-)
```

```
-} //class
```

⇒ Our servlet class (DateSrv) hierarchy is

```
javax.servlet.Servlet(I)
```



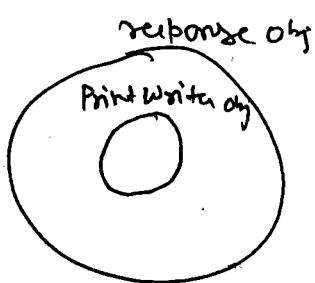
```
javax.servlet.GenericServlet (AC)
```

```
↑ extends
```

```
com.nt.DateSrv(C)
```

⇒ Our servlet prog class must be taken as public class to make class ready for instantiation through ServletContainer becoz only public classes are visible outside the container.

- ⇒ ServletContainer creates obj of our servlet class and calls service(-,-) on that obj to process the request
- ⇒ For every request given to servlet prg the ServletContainer creates one additional set of objects request, response objs and these objs visible in our servlet prg at the param service(-,-) methods.
- ⇒ In service(-,-) method we use request obj to get the data that comes along with the browser generated request & response obj is useful to send output to browser through webserver
- ⇒ res.setContentType("text/html");
 - Give instruction to browser through webserver to display received content as html code based text document (webpage)
- ⇒ PrintWriter pw = res.getWriter();
 - Give Access to PrintWriter stream obj that is there in response obj i.e. we PrintWriter stream Obj that points response obj as destination,



- ⇒ pw.println("Hello")
 - This method writes given message ("Hello") in response obj, response obj writes message to webserver, webserver writes this message to browser as content of webpage.

⇒ pw.close()

* Closes the PrintWriter stream i.e. we can't write further content to response object from servlet prg. This is nothing but committing response.

Step 3) Our servlet prg is using servlet api but it is not part of jdk. Tomcat server gives servlet-api.jar having servlet api so add that jar file to classpath.

My Computer → Properties → Advanced System Setting
→ Environment Variables → User / System Variable

Variable name classpath

Value

D:\Tomcat8.0\lib\servlet-api.jar; .

only copy paste

don't type

→ OK (3)

Step 4) Compile the servlet prg (DateSrv.java)

E:\DataApp\WEB-INF\classes javac -d . DateSrv.java

Step 5) Configure servlet prg in web.xml file

⇒ Providing details of servlet prg in web.xml file to make servlet container recognizing the servlet prg is called servlet prg configuration. Based on this cfg only servlet container locate servlet prg that is there in WEB-INF\classes folder of web application

web.xml

<web-app>

<servlet>

<servlet-name> abc </servlet-name>

<servlet-class> com.mt.DateSrv </servlet-class>

</servlet>

fully qualified class name
of servlet program

< servlet-mapping >

< servlet-name > abc </ servlet-name >
logical name must match with above
< url-pattern > /tut1 </ url-pattern >

</ servlet-mapping > url pattern

</ web-app >

⇒ Every Servlet prg is identified with its url pattern to generate requests. It is not identified with its logical name or with real class name.

⇒ All the Servlet prgs web application must be cfg in web.xml file So it is called web application Cfg file.

⇒ The moment we deploy web application in web server the servlet container reads web.xml file, so it is called Deployment Descriptor file.

Step1 to Step5:- Development of web appn.

Step6

Start the tomcat web server

Wx < Tomcat 8.0 _ home > \ bin \ tomcat8.exe file

Step7 Deploy the Web application (DataApp)

Copy E:\ DataApp folder to < Tomcat . home > \ webapps folder.

Step6 to Step7 :- Deployment of web appn.

Step8:- Test the Appn

http://localhost: 3030 / DataApp / tut1

Protocol	hostname &	web	url pattern
	port no of Tomcat	appn	of Servlet
		name	program

10
 ┌─────────┐
 | 22/04/25 |
 └─────────┘

⇒ The modifications done in servlet prg of deployed appn will be reflected only after recompilation of servlet prg and reloading of web appn.

For Recompilation For Recompilation:-

D:\Tomcat 8.0\webapps\DateApp\WEB-INF\classes>javac -d . *java
for reloading :-

<Tomcat-home> Page → Manage App → Username = admin
 Password = admin → Mgmt →

Tomcat web App manager window → DateApp → reloaded.

⇒ The msgs of System.out.println() goes to log message or confirmation message.

⇒ The msgs of pw.println() go to browser window as the content of web page

⇒ Servlet Container instantiate our servlet prg class by using no-param constructor.

So we must ~~not~~ make greater no-param constructor

*1 → Filled by programmers mainly.

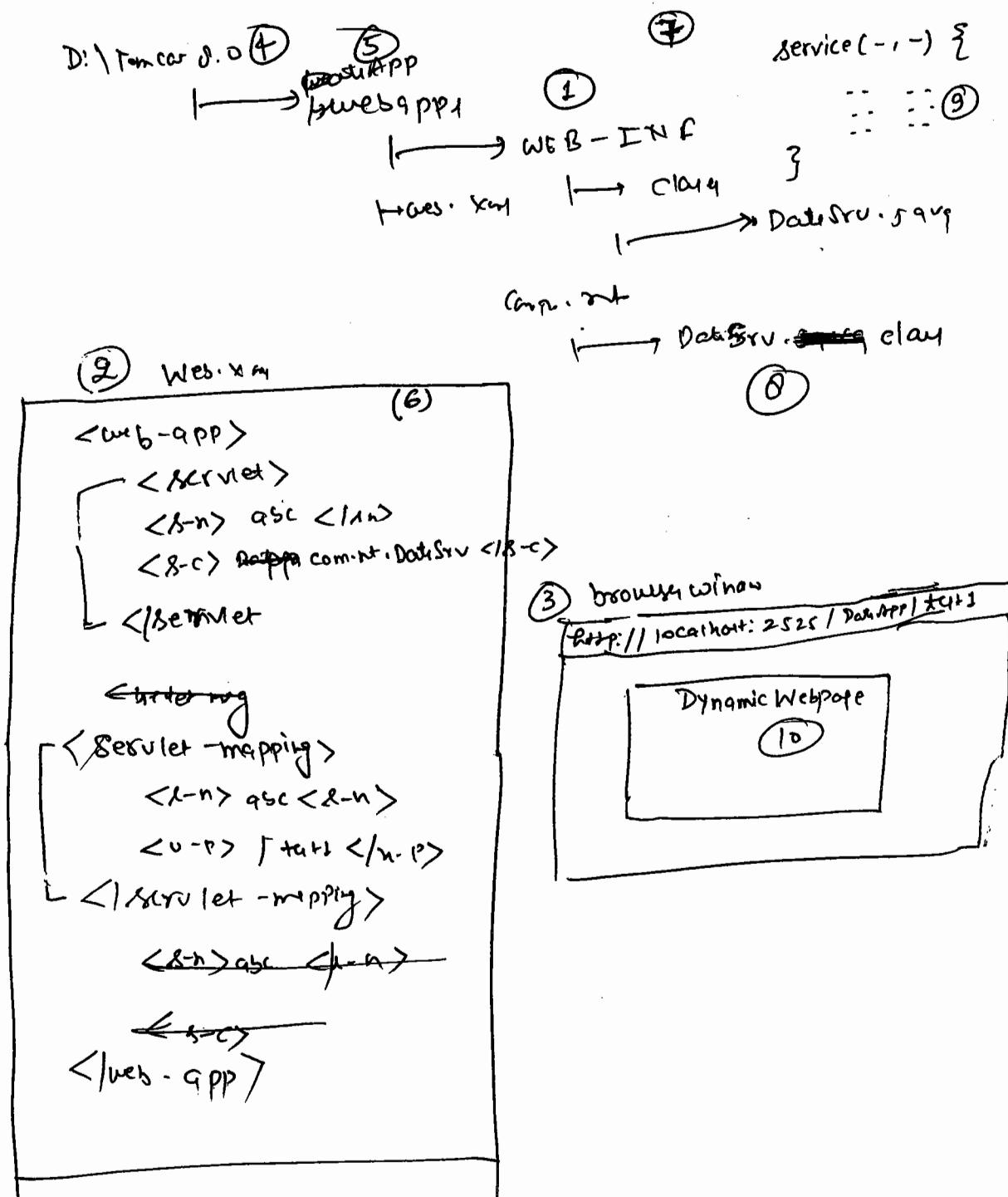
*2 → Default no-param constructor generated by javac compiler implicitly

Can we place only parameterized constructor in our
Servlet too

A) No,

Becoz ServiceContainer creates our servlet class
Obj using no-param const' constructor.

Partial Flow of execution from req generation to response
arrival with the aid of Example (DataServlet)



w.r.t. Diagram—

- 1) Programmer deploys DateApp webapp in Tomcat webServer
- 2) The ServletContainer of Tomcat reads, verifies web.xml file and stores the content of web.xml at RAM level in HashMap or In-memory metaData. (the info like servlet prg details and other more details)
- 3) End user gives request to Servlet prg by typing request url in the browser address bar.
- 4) Based on "localhost:3030" of request url the request goes to Tomcat Server.
- 5) Based on DateApp of ~~web app~~ ^{request url} the Tomcat parses the request to the deployed DateApp webapp
- 6) Based on "/text1" of request url the servlet container gets logical name(abc) class name (com.nt.DateSrv), from "In-memory" metaData. (RAM)
- 7) ServletContainer creates + set of request, response obj for current request.
- 8) Servlet Container loads "com.nt.DateSrv" class from WEB-INF\classes folder for object creation and completes all initialization formalities.
- 9) ServletContainer calls service(-,-) on DateSrv class Obj having req, res obj as arguments.
- 10) service (-,-) processes the request and sends the generated o/p to browser as response through web server to display in the form of dynamic web page.

Note: Servlet prog is not identified with its class name.
becoz if class name is exposed in request url
that may give idea to hacker, the regarding the
technology used in website development.

To avoid this problem url pattern is introduced
to hide Servlet class name from end user. This
also gives chance to have multiple url patterns for
one servlet program.

(Place multiple <url-pattern> tags under <servlet-mapping>
tag for this)

```
<servlet-mapping>
  <server-name> abc </server-name>
  <url-pattern> /nit.c </url-pattern>
  <url-pattern> /abc/xyz/ nit.cpp </url-pattern>
</servlet-mapping>
```

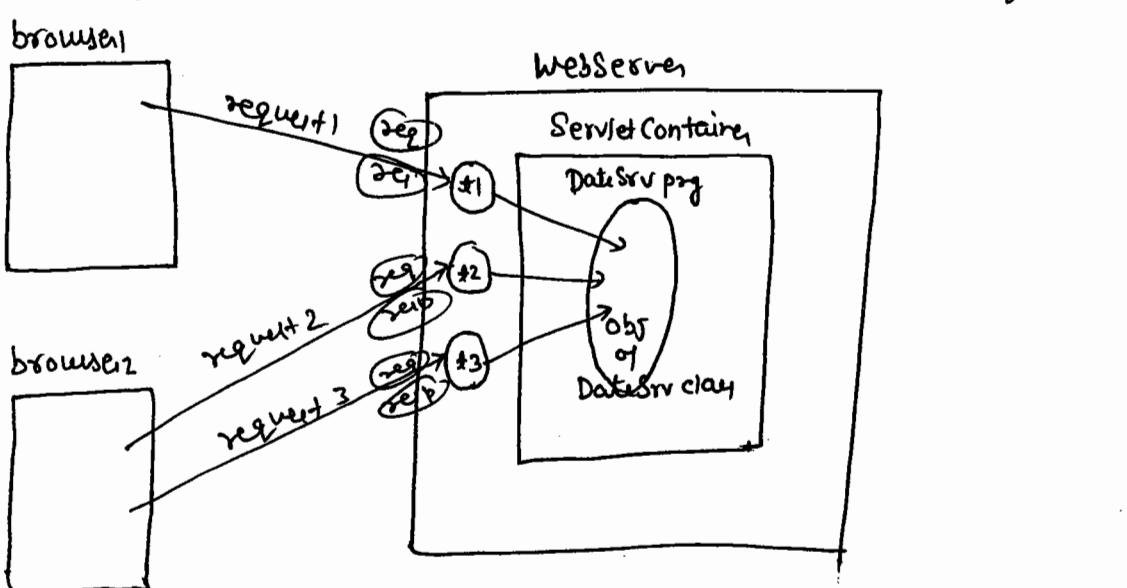
request urls: http://localhost:3030/ Datashop/nit.c / DataApp/abc/xyz
/nit.cpp

~~11
23/04/25~~

⇒ The modifications done in web.xml will be recognized by
underlying servlet container automatically becoz it
internally reloads the web appn.

⇒ When server stopped when web appn is stopped/reloaded
all the existing obj (our servlet class obj, request, response
obj and etc..) will be destroyed.

- ⇒ Reloading web application is nothing but stopping web application + restarting web application.
- ⇒ Our Servlet prog is single instance multiple threads comp i.e. we our servlet prog gets 10 request from same or different clients (browser window) then
- a) ServletContainer creates only obj of our servletprog
 - b) ServletContainer creates 10 threads representing 10 requests on our servlet class obj.
 - c) ServletContainer creates 10 sets of request, response obj for 10 request on 1 set per request-base.



⇒ To demonstrate above diagram practically place following code in service method of Servlet program as shown below —

```

try {
    Thread.sleep(10000);
} catch (Exception e) {
    e.printStackTrace();
}
pw.println("<br> hashCode of request obj " +
    req.hashCode());

```

```
pw.println("<br> hashcode of response obj"
           +res.hashCode());
```

```
pw.println("<br> hashcode of our servlet class
           Obj "+this.hashCode());
```

```
pw.println("<br> hashcode of current Thread obj"
           +Thread.currentThread().hashCode());
```

⇒ For every obj jvm gives one unique identity value that is hashCode. To get this hashCode use objref.hashCode() (method belongs to java.lang.Object class).

⇒ For every request given to web resource pgm the servlet container creates 1 thread and 1 set of request, response obj representing that request and the same will be destroyed once the request related response goes to browser from server.

⇒ In some servers if more (huge) requests are given servlet pgm the servlet container may create more than one obj to our servlet class.
(It is exceptional case)

javax.servlet.ServletRequest(I)

↑ extends

javax.servlet.http.HttpServletRequest(I)

`javax.servlet.ServletResponse(I)`

↑ extend

`javax.servlet.http.HttpServletResponse(I)`

Q) How can we say request, response obj when `ServletRequest` & `ServletResponse` are interface?

A) request obj means it is the object of underlying servlet container supplied java class that implements `javax.servlet.ServletRequest(I)`

response obj means it is the object of underlying servlet container supplied java class that implements `javax.servlet.ServletResponse(I)` directly or indirectly.

Programmer never expose their class name of these objects in their Servlet prg 'becoz the class name change server to server / container to container'. This helps to develop servlet prg at server independent prg (portable prg)

⇒ Tomcat server request obj class name : `org.apache.catalina.connector.RequestFacade`

and response obj class name

`org.apache.catalina.connector.ResponseFacade`

(refer `<Tomcat_home>\lib\catalina.jar` file)

	(12)	
	24/04/15	

regedit → Hkey_Classes_Root

⇒ GenericServlet gives only one service i.e.

```
public void service(ServletRequest req, ServletResponse res)
    throws SE, IOException {
    ==
}
```

⇒ HttpServlet gives two service(-,-) method:

1st service(-,-) / public service(-,-) method:

```
public void service(ServletRequest req, ServletResponse res)
    throws SE, IOException {
    ==
}
```

2nd service(-,-) / protected service(-,-) method:

```
protected void service(HttpServletRequest req, HttpServletResponse res)
    throws SE, IOException {
    ==
}
```

⇒ ServletRequest, ServletResponse obj don't allow to use protocol "Http" feature.

⇒ HttpServletRequest, HttpServletResponse obj allow to use protocol "http" feature.

- ⇒ if web appⁿ contains multiple servlet prog^{we must be}
all those page web.xml file having unique url patterns,
logical names.
- ⇒ By specifying diff content type (MIME type) in
servlet prog by using re. setContentType (-) we can
make browser window to display web pages
in diff formats.

MIME: multipurpose Internet Mail Extension

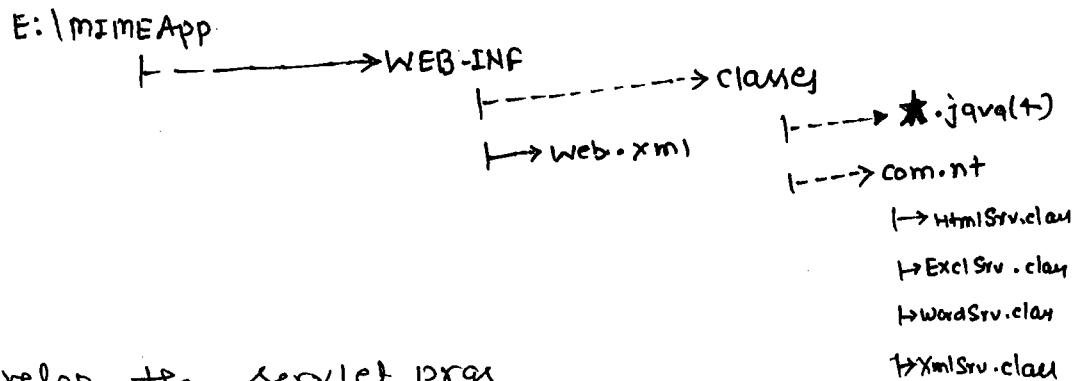
e.g. text/html, application/msword, application/vnd.ms-excel
and etc -- application/vnd.ms-excel

- ⇒ To get these MIME types we can go "regedit" tool
Start → regedit → HKEY_CLASSES_ROOT → choose
extension to know MIME type.

Web application 2

Develop web application having multiple servlet programs
to generate web pages in diff formats.

Step1:- Create Deployment Directory structure



Step2:- Develop the servlet progs

↳ refer Appn 1 of page no 183 to 106.

Step3) Make sure that servlet-api.jar file is placed in classpath.

Step4) Compile Servlet prgs

E:\MIMEApp\WEB-INF\classes>javac -d . *.java

Step5) Develop web.xml having servlet prg configuration
refer page no 105, 106

Step6) Start Tomcat server.

Step7) Deploy the web application

copy E:\MIMEApp folder to <Tomcat_home>\webapps folder.

Step8) Test web application

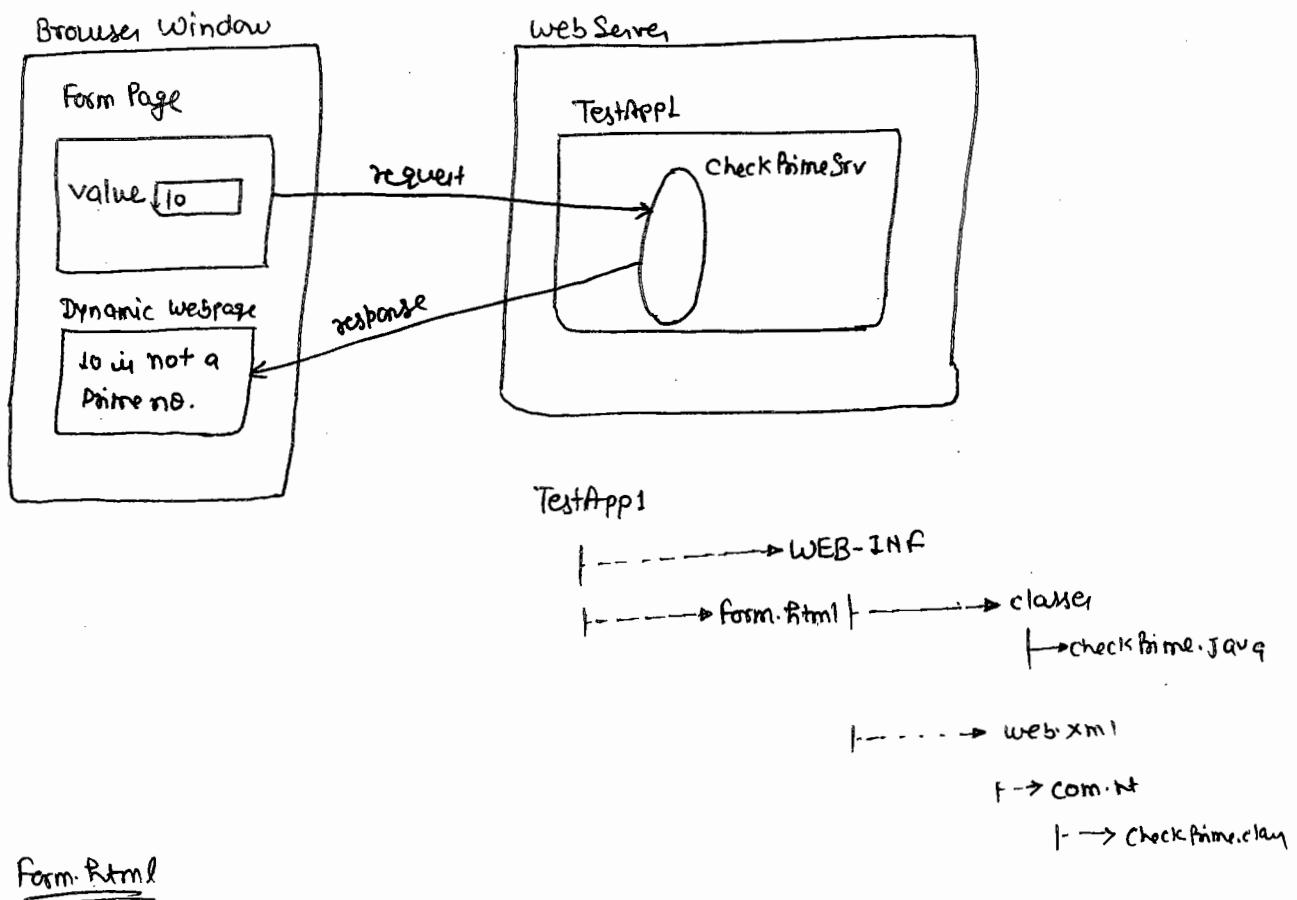
request urls

http://localhost:8080/MIMEApp/rtfurl
" .. " / wordurl
" .. " / xmlurl
" .. " / xlsurl

16/03/15

Except List Box, checkbox, components remaining form components send empty values or default values. The List box check box comps will send values only when they are selected/checked.

HTML to servlet communication using JavaScript:-



```
<form name="frm" action="cps" method="get">
  check prime or not?: <input type="text" name="f1">
</form>
```

onblur = "frm.submit()" >

Java
Script
event
This runs when
component loses
the focus.

Submit is a request
i.e. A JavaScript for

checkPrimeSrv.java

```
package com.nt;  
public class checkPrimeSrv extends HttpServlet {  
    protected void processRequest ( - , - ) throws SE, IOE {  
        //General Setting  
        PrintWriter pw = res.getWriter ();  
        res.setContentType ("text/html");  
        //read form data  
        int value = Integer.parseInt ( req.getParameter ("t1") );  
        //check whether given value is prime or not  
        boolean flag = false;  
        for ( int i = 2; i < value; i++ ) {  
            if ( value % i == 0 ) {  
                flag = true;  
                break;  
            }  
        }  
        if ( flag == true )  
            pw.println ( value + " is not a prime no." );  
        else  
            pw.println ( value + " is a prime no." );  
        //close stream  
        pw.close ();  
    } //process Request ( - )  
    public void doGet ( - , - ) throws SE, IOE {  
        processRequest ( - , - );  
    }  
}
```

```
protected void doPost(-,-) throws SE, IOException {
    processRequest(-,-);
}
```

} // class

web.xml

* cfg com.nt.CheckPrimeSrv prg with /cfg url pattern

* cfg form.html as welcome page

default url: http://localhost:2525/TestApp2/form.html

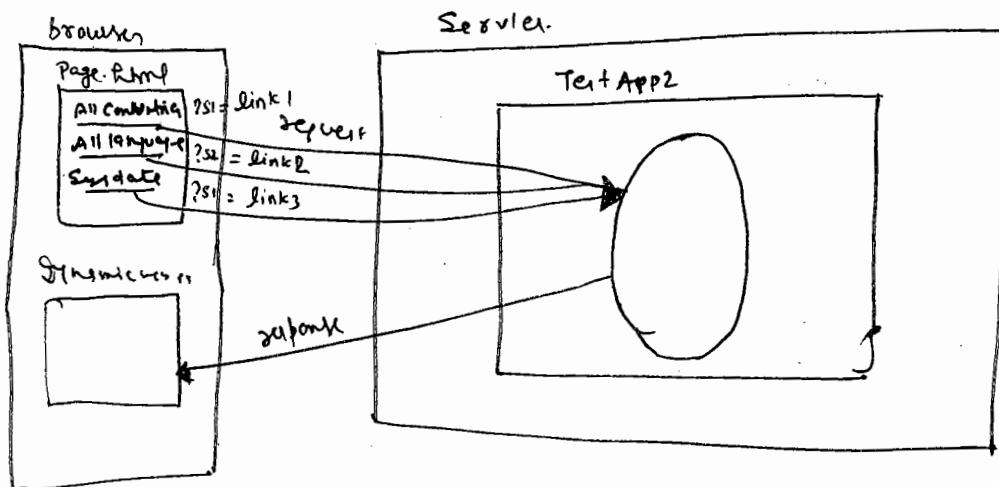
 go

→ This hyperlink does not carry any data given by programmer.

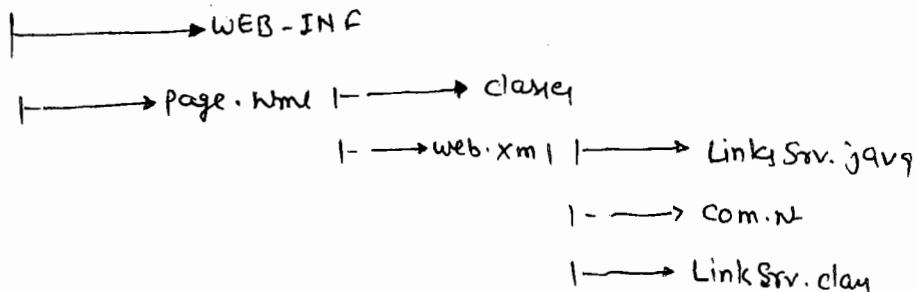
 go

→ This hyperlink carries 101, raja values given by programmer.

M.RMP How differentiate logics in servlet program when multiple hyperlinks are generating requests to that servlet program



TelApp2



- * Locale means language + Country
- * java.util.Locale class getAvailableLocales() method gives array of Locale class objs having info about all the locales.

```
Locale locale[] = Locale.getAvailableLocales();
```

Page.html

```
<a href = "linkurl?&1=link1"> All Countries </a>  
<br>  
<a href = "linkurl?&2=link2"> All Languages </a>  
<br>  
<a href = "linkurl?&1=link3"> All SysDate </a>
```

LinkSrv.java

```
package com.nt;  
public class LinkSrv extends HttpServlet {  
    protected void doGet(-, -) throws SE, IOE {  
        // General settings  
        PrintWriter pw = res.getWriter();  
        res.setContentType("text/html");  
        // read additional req param (s1) value  
        String pval = req.getParameter("s1");
```

```

    // write logic for hyperlink
    if (pval.equals("link1")) {
        Locale locale[] = Locale.getAvailableLocales(); // gives all Locales
        for (Locale loc : locale) {
            pw.println(loc.getDisplayLanguage() + "   ");
        }
    } else if (pval.equals("link2")) { // 2nd link
        Locale locale = Locale.getAvailableLocales();
        for (Locale loc : locale) {
            pw.println(loc.getDisplayLanguage() + "   ");
        }
    } else { // 3rd link
    }
    pw.println("Date & Time " + new Date());
}

```

3 // doGet (-,-)

```

protected void doGet(-,-) throws SE, IOE {
    doGet(req, res)
}

```

3 // class

web.xml

Cfg Com.nt.LinkSrv with /linkurl url pattern & page.html
as welcome file

depurl: http://localhost:2525/TestApp2/page.html

Observation:

```
<form action = "turl">  
---  
---  
<input type = "submit" value = "send">  
</form>
```

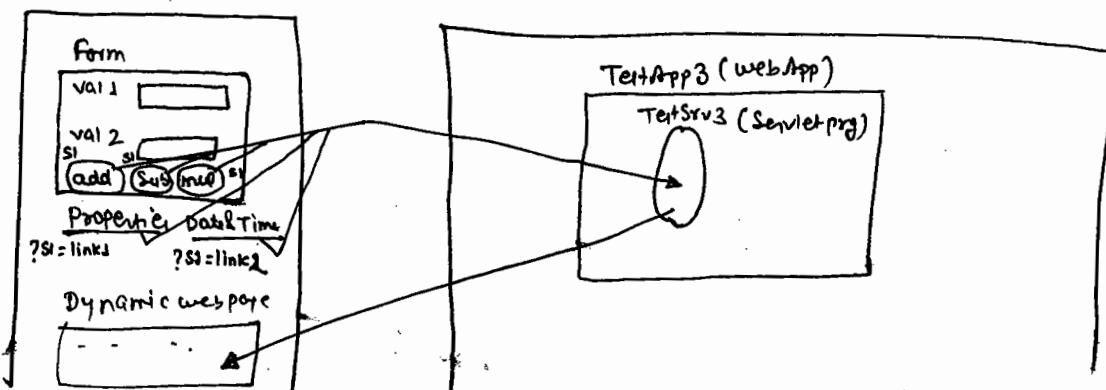
* This does not send submit button caption as req param

```
<form action = "turl">  
---  
---  
<input type = "submit" name = "s1" value = "send">  
</form>
```

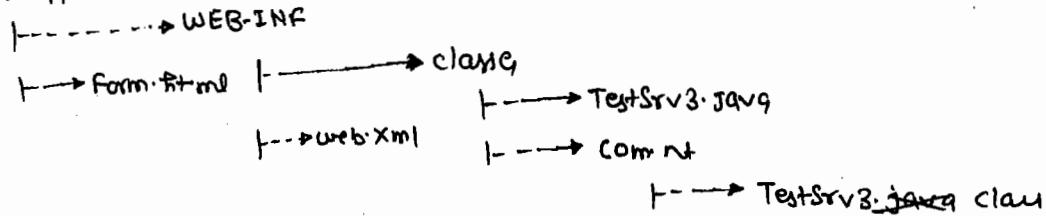
⇒ This sends submit btn caption as req param value along with request (s1 = send)

(16)
17/03/15

How to differentiate logic in Servlet program when multiple submit buttons & hyperlinks are generating request to single servlet program.



TestApp3



Form.html

```
<form action = "turl">  
    value1: <input type = "text" name = "f1" > </br>  
    value2: <input type = "text" name = "f2" > <br>  
    <input type = "Submit" name = "s1" value = "add" >  
    <input type = "Submit" name = "s1" value = "sub" >  
    <input type = "Submit" name = "s1" value = "mul" >  
    <a href = "turl ? s1 = link1" > SystemProperties </a> &nbsp;  
    <a href = "turl ? s1 = link2" > Date * Time </a> &nbsp;
```

TestSrv3.java

```
// import package in netbeans → ctrl + shift + I
```

```
Package com;
```

```
public class TestSrv3 extends HttpServlet {
```

```
    protected void processRequest (HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException
```

```
{
```

```
    // General Setting
```

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

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

```
    // read s1 req param value
```

```
    String pval = req.getParameter ("s1");
```

```
    if (pval.equals ("link1")) {
```

```

pw.println("Sys Properties => " + System.getProperties());
}

else if (pval.equals("link2")) {
    pw.println("Date & Time" + new Date());
}

else if (pval.equals("add")) {
    *1
    int v1 = Integer.parseInt(req.getParameter("t1"));
    int v2 = Integer.parseInt(req.getParameter("t2"));

    pw.println("Sum:" + (v1+v2));
}

else if (pval.equals("sub")) {
    *1
    pw.println("Sub:" + (v1-v2));
}

else if (pval.equals("mul")) {
    *1
    pw.println("Mul:" + (v1*v2));
}

//close Stream
pw.close();

//process Request (-,-)
protected void doGet(-,-) throws SE, IOE {
    processRequest(request, response);
}

protected void doPost(-,-) throws SE, IOE {
    processRequest(request, response);
}

```

web.xml

Configure

* We can use one of these following two streams to work in Servlet Programming

- a) PrintWriter
- b) ServletOutputStream

PrintWriter is a character stream and it is useful to use when there is more char data & less binary data (images) to write.

ServletOutputStream is a byte stream & it is useful to use when there is less char data and more binary data to write.

Example:-

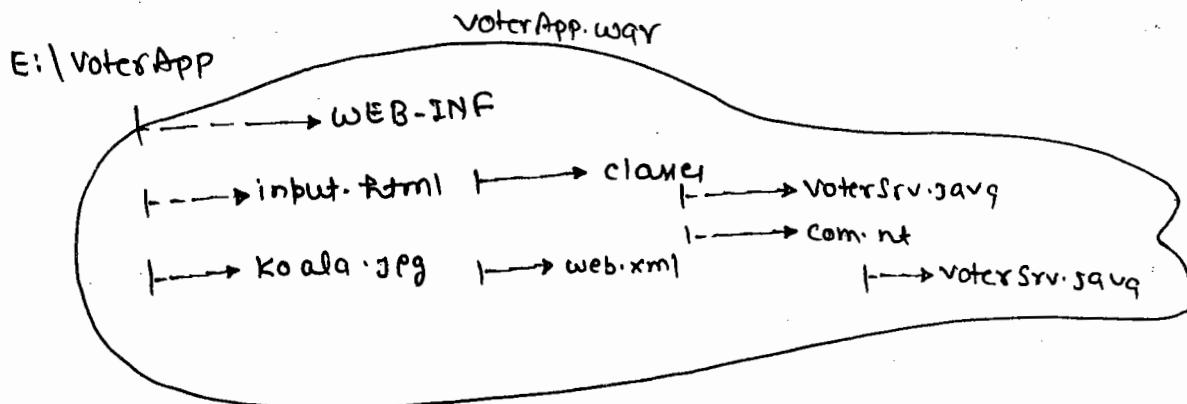
```
PrintWriter pw = res.getWriter();
```

```
ServletOutputStream sos = res.getOutputStream();
```

We can't place multiple streams in one servlet prg i.e.
we can use either ServletOutputStream or PrintWriter
stream in one servlet prg.

We can place either multiple PrintWriter streams, or
multiple ServletOutputStream in one servlet prg.

In real time the web app will be released to client orgs and
will be host on to the internet now in the form of
WAR files (can be created using jar command).



E:\VoterApp> jar cf voterApp.war .

- * Write file in WODA (Write Once Deploy anywhere)
- * In most of the servers If, WAR file is deployed the WAR file name itself becomes name of the app.

Different Modes of Deployment:-

- 1) Hard Deployment (copy war file or directory of the web app to a fixed folder of the server like webapps)
 - 2) Smooth / Console Deployment (Deployment by using the admin console)
 - 3) Tool Based Development (using IDE, Ant tool, Maven tool and etc...)
- ⇒ During Development mode of the project Hard Deployment is used.
- ⇒ During Development Integration of Project Tool based deployment will be used.

How to perform console Deployment in Tomcat Server:-

- 1) Prepare war file

VoterApp.war

- 2) Open Admin Console of Tomcat & deploy the WAR file.

Tomcat home page (<http://localhost:2525>)

→ Manager APP

Username

Pwd

Ok

Go to WAR file to deploy section → browse and select the WAR file.

Note → This WAR file goes <Tomcat_home>\webapps folder.

- 3) Test the App

<http://localhost:2525/voterApp/input.html>

name of
the war file

For hard deployment in tomcat server copy the directory of web appn (VoterApp) or its war file (VoterApp.war) to Tomcat home webapps folder.

* what is the difference b/w hot deployment & cold deployment

If webapp is deployed in server when server is in running mode then it is called as hot deployment.

If the web appn is deployed in server when server is in stopped mode then it is called cold deployment.

Note→

Console deployment should always be hot deployment.

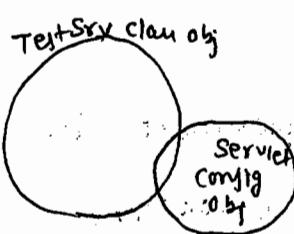
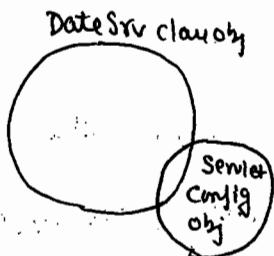
Tool based deployment or Hand deployment can be hot deployment or cold deployment.

17

18/03/15

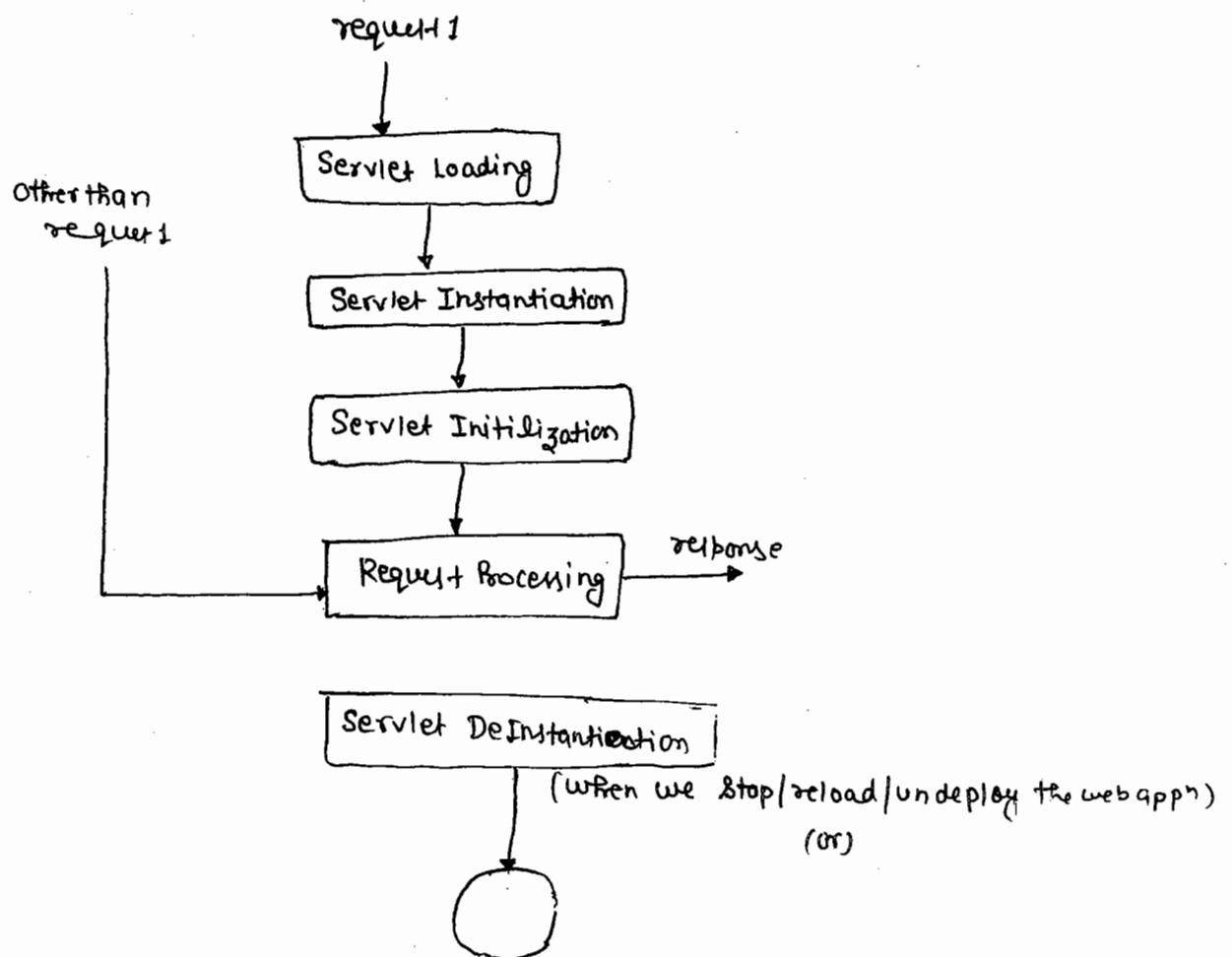
Servlet Life Cycle:-

- ⇒ ServletConfig obj is a right hand object to our servlet class object & it is one per our servlet class obj. This object is useful to know information about servlet program & to pass additional info to servlet program.
- ⇒ Assigning this obj to our servlet class obj is called servlet initialization.

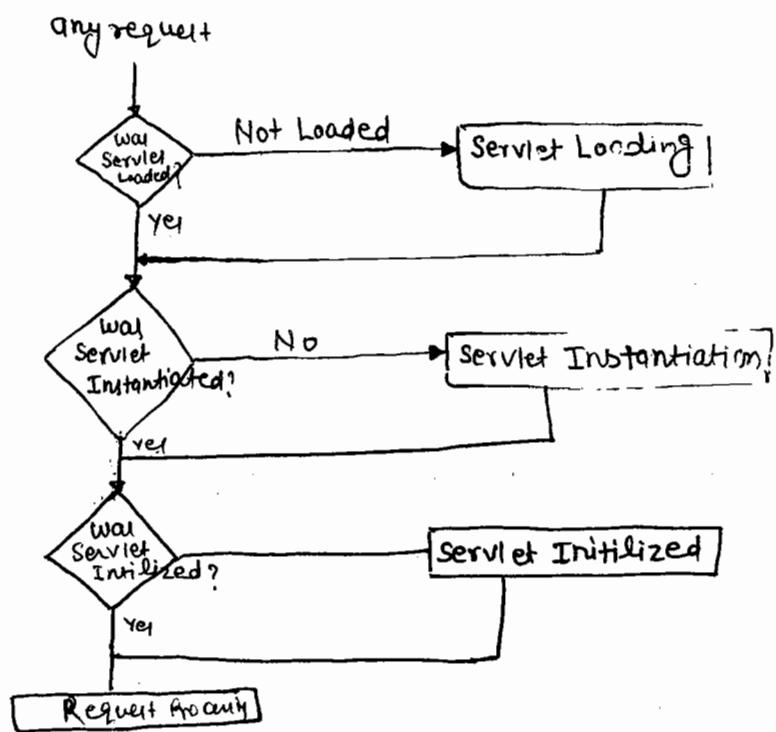


- ⇒ Servlet Lifecycle means taking care of all the operations from birth of our servlet class obj to death of our servlet class obj.
- ⇒ Servlet Container takes care of Servlet lifecycle.
- ⇒ ServletContainer creates only one obj of our servlet

class and use that obj for all requests.



Flow chart of Servlet Life Cycle:-



* Servlet Container loads & creates our servlet class obj as shown below —

```
Class.forName("com.nt.DateSrv").newInstance();  
    (loads the java class)      (create the obj  
                             for loaded class)
```

* As part of Servlet Lifecycle managed by ServletContainer the Container raises 3 life cycle events and its calls 3 lifecycle methods for these 3 lifecycle events

The Servlet Life Events are —

a) Instantiation Event

(Raises when servlet container creates our servlet class obj)

b) Request Processing Event

(Raises when our servlet prg ready for request processing)

c) Destruction Event

(Raises when ServletContainer is about to destroy our servlet class object)

* The 3 life cycle methods are —

a) init (ServletConfig) → for Instantiation Event

b) service (ServletRequest, ServletResponse) → for Request Arrival Event

c) destroy () → for Destruction Event

prototype of life cycle method -

1) public void init (ServletConfig) throws SE

note: init() is not life cycle method.

2) public void service (ServletRequest, ServletResponse) throws SE, IDE

Note - 2nd Service (-,-) and 7 doXXX() methods are not life cycle methods.

3) public void destroy()

⇒ Since life cycle methods will be ServletContainer automatically for life cycle event there are also called Container callback method.

⇒ We override lifecycle methods in our servlet program for these reasons.

a) To place our choice logic to execute in the execution servlet program done by servletcontainer.

b) To seen ServletContainer created objs in our servlet program as the param of life cycle methods.

(Like ServletConfig, ServletRequest/Response and etc...)

init(ServletConfig):-

⇒ It is one time execution lifecycle method of our servlet program for instantiation for Event. Here we place instantiation logic like creating jdbc connection, initializing servletconfig obj and etc.

service(-,-):-

⇒ It repeatedly executing lifecycle method of our servlet program for request processing event. Servletcontainer calls this method for every request, so we place request processing logic in this method.

destroy()

⇒ It is one time execution life cycle method of our servlet program for Destruction Event. Servlet Container calls this method when it is about to destroy or destination our servlet class obj.

We generally place utilization logic in this method like closing jdbc con.

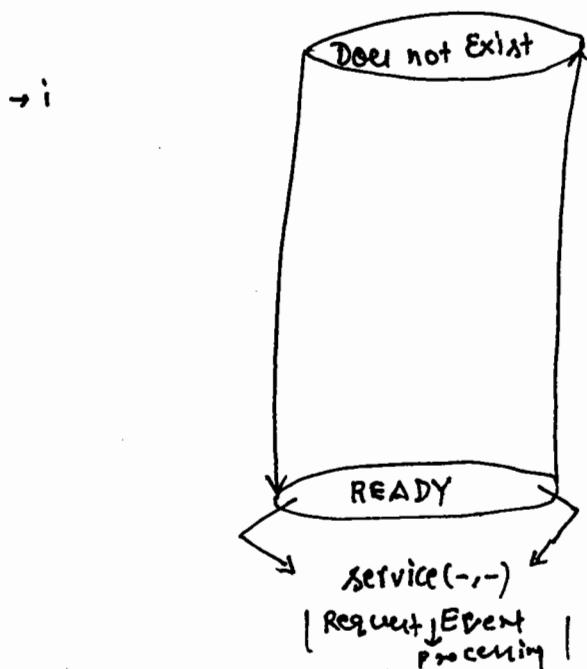
init() -

does not logic to create our servlet class obj, Servlet Container internally maintains that logic.

destroy() -

does not contain logic to destroy our servlet class obj, Servlet Container internally maintaining that logic.

(18)
19/03/15



What happens when Servlet program gets first request from Client (browser window) :-

- a) Create one set of ServletRequest, ServletResponse objs for the current request.
- b) ServletContainer loads our servlet program class from WEB-INF/classes folder.

```
Class c = Class.forName("com.nt.DateSrv");
```
- c) ServletContainer creates our servlet program class obj using 0-param constructor c.newInstance();
- d) ServletContainer creates ServletConfig obj and raises Instantiation Event.
- e) ServletContainer calls the lifecycle method init(-) having ServletConfig obj ref as argument to complete Servlet Initialization.
- f) ServletContainer creates one thread representing current request on our servlet program class obj & raise request processing event.
- g) ServletContainer calls 1st service(-,-) as lifecycle method having ServletRequest, ServletResponse obj reference as arguments.
- h) service(-,-) processes the request & sends dynamic response to browser.

what happens if our servlet program get other than

1st request:-->

- a) ServletContainer creates one set of ServletRequest, ServletResponse objs for current objects. request.
- b) ServletContainer checks availability of Our Servlet class obj?

If available

⇒ ServletContainer creates one thread on our servlet class obj representing current request & raise Request Processing Event.

⇒ Servlet Container calls the lifecycle method service(..) having ServletRequest, ServletResponse obj reference as arguments.

⇒ service(..) process the request & send response to browser.

If not available

⇒ Performs all the operations of 1st request.

Procedure to add SampleServlet program in MIMEApp web application to see the demo of lifecycle methods

- ⇒ Keep MIMEApp web App ready.
- ⇒ Add following LctextSrv.java file in WEB-INF\classes folder of MIMEApp web application.

⇒

LcTextSrv.java

```
package com.nt;

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;

public class LcTextSrv extends HttpServlet
{
    static { Sopin("LcTextSrv: static block"); }

    public LcTextSrv()
    {
        Sopin("LcTextSrv: o-param constructor");
    }

    public void init(ServletConfig cg)
    {
        Sopin("LcTextSrv: i-param constructor");
    }

    public void service(HttpServletRequest req, HttpServletResponse res)
        throws ServletException
    {
        // get PrintWriter
        PrintWriter pw = res.getWriter();

        // Set response Content-Type
        res.setContentType("text/html");

        // Write response
        pw.println("<b><i> <centre> Date and Time " + new Date()
                  +" </i> </b> </centre>");

        // Close Stream
        pw.close();
    }
}
```

```
SopIn("LcTetSrv: 1st Service (-) method");
```

}

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

```
    SopIn("LcTetSrv: destroy() method");
```

}

```
? //clay
```

```
// javac -d . LcTetSrv.java
```

⇒ Cfg com.nt.LcTetSrv program in web.xml file with /lcurl urlpattern.

⇒ request url for testing

http://localhost:2525/mimeapp/lcurl

Q) What is the meaning of enabling <load-on-startup> on servlet prg?

Q) What is servlet pre-instantiation & pre-initialization?

Q) What is Early/Eager loading of servlet?

Ans) When <load-on-startup> is enable on servlet program

the ServletContainer performs servlet loading, servlet

instantiation, servlet initialization operations either during

servlet startup (for Cold Deployment) or during the deployment

of web application (for Hot Deployment). This process is also

called pre-instantiation & pre-initialization or Eager/Early loading of servlet.

```

< servlet >
  < &-n > lc < /&-n >
  < &-c > com.nt.LcTextSrv < /&-c >
    < load-on-startup > 1 < /load-on-startup >
</ servlet >
< servlet-mapping >
  =
</ servlet-mapping >

```

⇒ Generally ServletContainer creates object of our servlet class only when need is there like for 1st request when load on startup is not enable. This is called **lazy loading** of servlet.

(19)

20/03/15

Q) What is the advantage of enabling `<load-on-startup>` of servlet program.

A) If servlet program is developed without enabling `<load-on-startup>` the first request given to servlet prg participates in Servlet loading, Servlet instantiation, Servlet initialization before performing request processing, whereas all the other than 1st request given to servlet prg directly participates in request processing. Due to this the response time (time taken to generate response) of 1st request is little bit more when compare to that response time of other than 1st request.

⇒ To minimize the response time of 1st request we can make servlet container loading, instantiating, initializing our servlet program either during server startup or during deployment of web application. For this we need to enable <load-on-startup> on servlet program. Here the response time of 1st request will be equalized with the response time of other than 1st request.

⇒ Enable <load-on-startup> only that servlet program which will be request immediately after deployment.

e.g. Servlet program that gives main menu, servlet program that gives home page.

⇒ When Servlet Container Creates our servlet class Obj:—

- When Servlet prog gets 1st request
- When Servlet prog gets 1st request after restarting/reloading/redeploying the web application.
- Either during server startup or during the deployment of web application if <load-on-startup> is enabled.

When Servlet Container destroys our servlet class Obj:—

- When Stop/restart server
- When webapplication is stopped/reloaded/undeployed
- When Server is crashed
- If our servlet class obj is continuously idle

Note ⇒ No Special priority towards destruction even though <load-on-startup> is enabled.

⇒ When multiple servlet programs are enabled with <load-on-startup> then their order of pre-instantiation, pre-initialization will be decided based on <load-on-startup> priority values.

↳ High value indicates low priority

↳ Low value indicates high priority

⇒ -ve value will be ignored the enabled <l-o-s>

⇒ Srv1, Srv2, Srv3, Srv4 are servlet progs of a web application (w.r.t. tomcat 6/7/8)

	<l-o-s> value	<l-o-s> value	<l-o-s> value
Srv1	10 (III)	0 (I)	10
Srv2	5 (II)	1 (II)	10
Srv3	14 (IV)	4 (III)	10
Srv4	1 (I)	-10 (Ignore <l-o-s>)	10

$$\text{WondSrv} = 10$$

$$\text{ExcelSrv} = -10$$

$$\text{XmlSrv} = 0$$

$$\text{HtmlSrv} = 7$$

$$\text{LCTextSrv} = 1$$

⇒ In tomcat 6 server if we take <load-on-startup> tag without priority value then 0 will be default value

In tomcat 7/8 server taking <load-on-startup> without priority value gives error.

In tomcat 5 0(zero) doesn't indicate high priority it indicates least priority.

⇒ It is not mandatory to enable <load-on-startup> on every servlet program.

What happens if destroy() is called explicitly from service() method.

Our servlet class obj will not be destroyed. But logic of destroy() method will be execute along with service().

When container raises lifecycle event, the lifecycle method will be called automatically. Since we have called life Even though

Cycle method explicitly the container doesn't raise lifecycle event.

b) What happens if we call init() from service()?

A) Container doesn't create object of our servlet class for every request. But it executes logic of init() along with service.

c) What happens if we call destroy method from init()?

Ans) Our servlet class obj will not be destroyed. But logic of destroy() method executes with init() method.

ServletContainer/ Server internally maintaining lot of code/module to create & destroy our servlet class obj so init() & destroy() method logic are not responsible to perform those operations.

⇒ Life cycle methods are not responsible for lifecycle event but these methods are called automatically by underlying container as server term lifecycle events are raise.

How Servlet program is executing without main().

Can we place main() in Servlet program.

What is the difference b/w placing initialization logic inside the constructor and inside the init().

Develop web app based login App to perform Authentication

(20)

23/03/15

Q) How Servlet program is executing without main()?

A) In Standalone App main() is required to begin the execution, Servlet prg is not a Standalone app, it is java based web resource prg whose execution will be taken care by servlet Container so main() method is not required in servlet program.

ServletContainer executes our servlet prg through life cycle methods, main() is not life cycle method of servlet prg, so it is not required in our servlet program.

Q) Can we place main() method in our servlet program?

A) Yes we can place main() but it will not be executed by servlet container because it is not the lifecycle method of servlet prg.

weblogic

- | → type: Application Server & w
- | → vendor: BEA Systems (oracle corp)
- | → version: 10.3 (compatible with jdk 1.6)
- | → commercial
- | → Default port no: 7001 (changeable)
- | → Allows to create domain. Each domain acts as separate App server.
- | → To download: www.oracle.com, www.commerce.bea.com

* If multiple projects of a company are using same weblogic, then the weblogic & w will be installed only for 1 time in a common computer (Integrated machine) and multiple domains will be created in that weblogic for multiple projects on 1 per project basis.

Procedure to create domain server in weblogic 10.3 (~~User-defined domain~~)

Start → Allprog → oracle weblogic → Quick Start → Getting Started with weblogic → Create a new weblogic domain → Next → Generate a domain → Next →

Domain Name	NtAdvJava9
Username	javasam
password	Javasam1
Confirm password	Javasam1
Next	

→ Next → Admin Server

Name	AdminServer
ListenPort	7070
Next → create	

Procedure to perform console/smooth deployment of web Application in NtAdvjavag domain server of weblogic 10.3

Step1) prepare war file representing web App
(VoterApp.war file)

Note: Make sure that class files are generated by using jdk 1.6 env.... Along with weblogic 10.3 installation we get one built in jdk 1.6 SW.

Creating warfile

> Jar cf voterApp.war .

Step2 Start NtAdvjavag domain server

Start → All programs → oracle weblogic → Web Projects → NtAdvjavag
→ Start Admin Server for weblogic

Step3 Open Admin console of NtAdvjavag server. --

open browser and write url. <http://localhost:7070/console>

Username

Password

→ Login

Step4 Deploy the web Application

Admin Console → Services → Deployments → Install → upload your file →
browse → Select voterApp.war → Next → Next → Next → finish

Step5 Test the web App

Url: <http://localhost:7070/VoterApp/input.html>

Note: To Undeploy webApp from admin console

→ Admin Console screen → deployment → Select Application
--- → delete

⇒ To perform hard deployment of web app in NtAdvJava9 domain server of weblogic copy either war file (VoterApp.war) or directory (VoterApp) web app to <weblogic-home>\middleware
~~test~~ \ user-projects\domains\NtAdvJava9\autodeploy folder.

request url: <http://localhost:7070/VoterApp/index.html>

Note ⇒ The App deployed through hard deployment process can't be undeployed through from admin console.

Note - In weblogic server the modification done in servlet program will be recognized by server without reloading by just doing recompilation.

* The class name of request, response object will change Servlet Container to Servlet Container based on the Servlet Spec we use. So we never specify that class name in our servlet program to make our servlet program portable or work in multiple servers.

21
24/03/15

war file: Web Application Archive (Represents web application)

jar file: Java Archive (Represents EJB comp)

ear file: Enterprise Application Archive (jar file + war file)

App Server = Web Server + EJB Container + more middleware services.....

Difference b/w Web Server & App Server

	WebServer	AppServer
1.	Even to manage to execute web appn (war file)	Given to manage & execute web appn's, EJB Components, Enterprise application (EAR file). EJB Components
2.	Develop based on Servlet, JSP	Developed based on, servlet, JSP, EJB, JMS Specification of JEE.
3.	Give only Servlet Container, JSP Container.	Give Servlet Container, JSP container, EJB container etc.
4.	Does not allow to create domain	Allows
5.	Give less middleware services	More
6.	Supports only http protocol.	Supports both http & non http protocols.
7.	Allows only webclient (browsers).	Allows both webclient & non web client. (Normal App like Awt frame, Swing frame)
8.	Suitable for small scale, medium scale Appn.	Suitable for large scale Appn
	Tomcat, Resin and etc.	weblogic, websphere, JBoss, JRun, Glassfish and etc.

→ Upto tomcat 5 it is called as webserver.

Tomcat 6, 7, 8 give more facilities, so some part industry calls them as Application servers....

Ques:
⇒ If we call method from Super class method i.e. executing based on Subclass object then first it looks to execute that method from subclass. If not available then it executes Super class method.

Class Demos

```
public void x() { (b)
```

```
    y(); (c)
```

```
}
```

```
public void y() {
```

```
--
```

```
    }
```

```
}
```

```
class Test extends Demo {
```

```
    public void y() {
```

```
        -- (d)
```

```
    }
```

```
Test t = new Test();
```

```
t.x();
```

Understanding two init() methods of servlet api

22
25/03/15

1st init(-) method: init(ServletConfig) → It is a life cycle method
→ Originally declared in Servlet() and implemented in GenericServlet class.

2nd init() method: init() → It is not a life cycle method
→ It is given as convenience method to programmer.
→ Originally defined in GenericServlet()

Note: → HttpServlet class does not contain any init method but it will use the inherited method of GenericServlet class.

2) For instantiation Event Servlet Container init(ServletConfig) as life cycle method. It does not call init() method.

For various scenarios of placing init method in our servlet program

↳ Refer page no 95 to 98.



22
26/03/15

GlassFish

Type: Application Server S/W

Version: 4.x / 3.x (compatible with jdk 1.6 onwards)

Vendor: Sun Microsystems (Oracle Corp)

Freeware

Allows to create domain. Default domain name is "domain1".

Each domain act as separate app server.

Default domain port no:

for admin console: 4848

for http access: 8080

Default domain credentials:

username: admin

password: admin/admin

To download: www.glassfish.org (download as zip)

To install S/W: extract zip file...

JAR file that represents JEE API: javee.jar

Procedure to create user-defined domain in GlassFish 3.x/4.x:

<GlassFish_Home>/glassfish3/glassfish/bin> admin create-admin-domain

--adminport=4845 --user=tetsuser --AdvJavaDomain

Port
no

User
name

Domain
name

Enter password > tetsuser

Confirm password > tetsuser

To change the "http" port no of "N+AdvJava9" server;

go to <Glaufish_Home>\glassfish3\glassfish\domain\config\domain.xml

and modify the "port" attribute value of first
<network-listener> tag.

Procedure to perform console / smooth deployment of web application in
GlaufishServer 3.x/4.x (N+AdvJava9 Domain)

x ————— x ————— x ————— x —————

Step1) Prepare war file

like VoterApp.war

Step2) Start "N+AdvJava9 Domain" server.

<Glaufish_Home>\--\--\bin> ./admin start-domain N+AdvJava9
domain.

Step3) Open admin console of "N+AdvJava9 Domain"

open browser window → url → http://localhost:4545

Username: tatusu

password: tatusu

Step4) Deploy the web app

AdminConsole → Applications → deploy → Browser &

Select the VoterApp.war file. → OK

Step5) Access the web app

Http://localhost:5050/VoterApp/Input.html

To perform hand deployment of web app. in "N+AdvJava9Domain"

server of Glaufish copy either war file (VoterApp.war) or

directory (VoterApp) to

<Glaufish_Home>\--\--\domains\N+AdvJava9Domain\autoDeploy\foldr

Diff request methods / model / methodologies:-

GET → header + body

HEAD → only header unique in debugging

POST

PUT

DELETE

TRACE

OPTIONS

order

Page no

101, 4102

⇒ Every response sent by server contains multiple parts like response header, response body / content.

⇒ The content of pw.println() becomes response body.

rw.setXXX() method (like res.setContentType()) sets response header to instruct browser window.

⇒ Domain register in company that gives 3 services

a) selling domain name.

b) Providing space in web server that belongs to static / fixed ip address computer of internal network.

c) maintenance of web app / web site.

e.g. GoDaddy, jekatic, ge bigrock, indigoix and etc..

⇒ To upload web app to web server of domain registered machine (Not my but hosting) we get "FTP App" in our computer. We can use this "FTP App" to add new resource & to delete existing resource from hosted website.

Correction - Get \rightarrow 206 \rightarrow 8

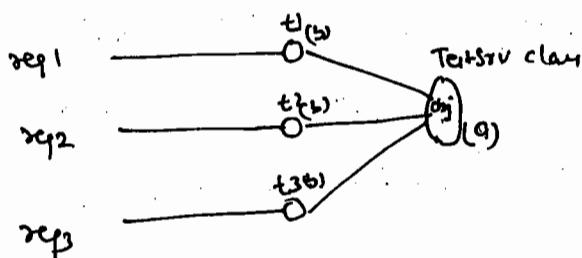
- ⇒ Browser window sends request to server having get methodology by default
- ⇒ Form Page can send either get methodology request or post methodology request to the server.
- ⇒ FTP appn can send either put or delete methodology request
- ⇒ In real time website development the most regularly used methodologies are get, Post due to this we use only doGet(), doPost() or doXXX() while developing the servlet program.

24
24/03/25

Servlet to DB SQL Appln:

/* instance variable \rightarrow not thread safe
local variable \rightarrow thread safe */

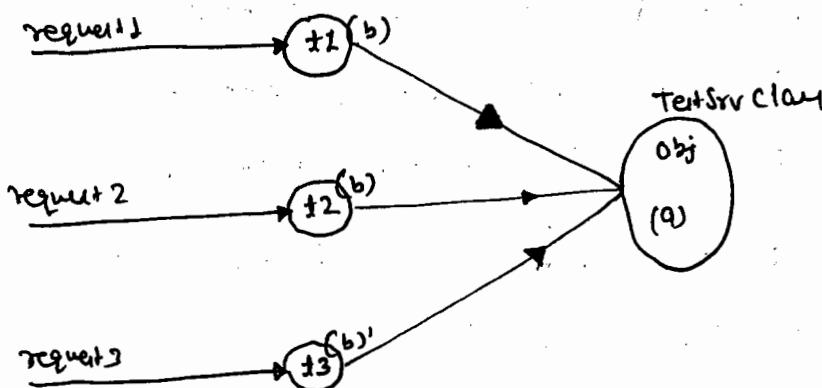
```
public class TestSrv extends HttpServlet {  
    int a; // instance variable  
    public void service(-, -) throws SE, IOException {  
        int b; // local variable  
        ---  
    }  
}
```



- ⇒ If multiple threads are acting on single object / variable concurrently then there is a possibility of getting data corruption. To solve this problem use synchronization concepts to allow only one thread at a time on the object or variable. This makes object or variable as thread safe.
- ⇒ Instance variable of servlet class are not thread safe by default to make them as thread safe use synchronization
- ⇒ Local variable of service() or doXXX() are thread safe because every thread gets its own copy of local variable

Sample code -

```
public class TestSrv extends HttpServlet {
    int a; // instance variable
    public void service(-----) throws SE, IOE {
        int b; // local variable
    }
}
```



- ⇒ To save inputs coming to servlet prg and to save results generated by servlet program permanently we need to our servlet prg talking with DB also by placing jdbc code in servlet prg.

- ⇒ Some times servlet prg interacts with DB s/w to gather inputs from DB s/w.
- ⇒ There are multiple approaches to place jdbc code in our servlet prg.:-

Approach 1 :-

- ↳ Create con obj in the init()
 - ↳ Use con obj in service(-,-) / doXXX(-,-) methods to develop persistence logic
 - ↳ Close con obj in destroy() method.
- ⇒ Here jdbc con obj should be taken as instance variable to using multiple methods of servlet program. Due to this all request coming to servlet program will use single con obj so we get good performance but we must handle multithreading issues explicitly. Advantage
- ↳ disadvantage

Approach 2 :-

- ↳ Create con obj in service(-,-) / doXXX(-,-)
 - ↳ Use con obj in service(-,-) / doXXX(-,-)
 - ↳ Close con obj in service(-,-) / doXXX(-,-)
- ⇒ Here con obj is local variable to service(-,-) method so it is thread safe by default Advantage
- ⇒ For multiple requests given to servlet program multiple connections will be established with DB s/w this degrades the performance. disadvantage

Approach 3:-

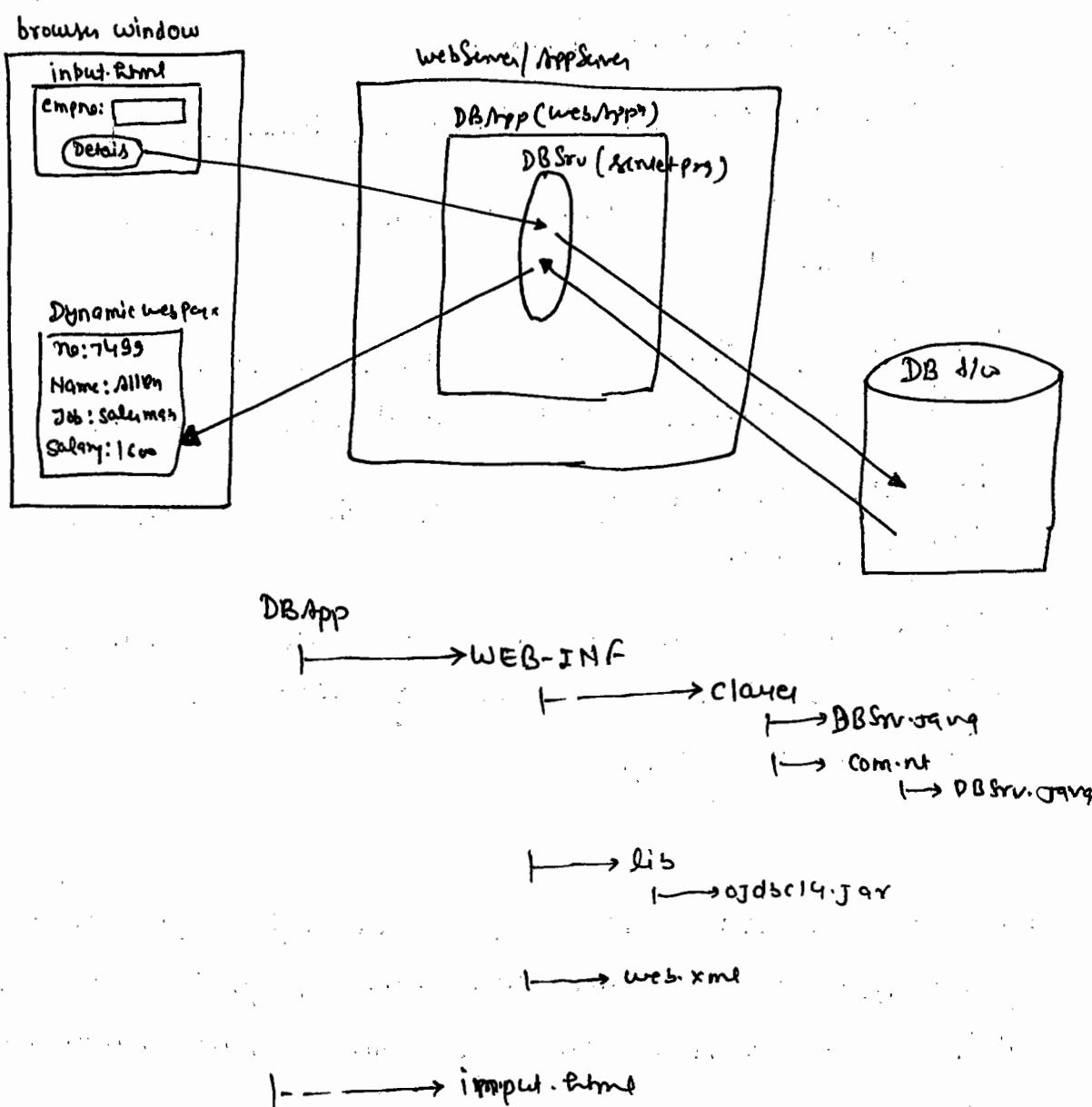
- ↳ Get con obj from server managed jdbc con pool being from service(-,-) / doxxx(-,-)
- ↳ Use con obj in service(-,-) / doxxx(-,-) method.
- ↳ Release con obj back to con pool being from service(-,-) / doxxx(-,-)
- ⇒ Here con ref variable will be taken as local variable to service(-,-) / doxxx(-,-) method. So it gives thread safety.
- ⇒ Since the con obj of con pool will be reused it gives good performance.
- ⇒ Programmers are not responsible to create or manage or destroy con obj all these operations will be taken by jdbc con pool itself.

Approach 4:-

- ⇒ The java class that separates persistence logic from other logic of the appn & to make them logic as flexible logic to modify and reusable logic is called DAO (Data Access Object)
- ↳ place persistence logic (jdbc code) in DAO.
- ↳ Use DAO in servlet prg.
- ⇒ Most of the real time projects we either Approach 3 or Approach 4



Servlet to DB Slw communication using Approach :-



- ⇒ If servlet program uses 3rd party API (other than jdk, servlet, jsp APIs) then the 3rd party API related JAR file must be selected to class path & to WEB-INF\lib folder.

e.g. if servlet prog uses oracle thin driver, then the thin driver related ojdbc14.jar must be added to classpath & also to WEB-INF\lib folder of webapp as shown above.

- ⇒ Here jar file added to classpath will be used by javac to recognize 3rd party api during compilation of servlet program.
- ⇒ Here that jar file added to WEB-INF\lib folder will be used servlet container to recognize & use 3rd party api during execution of servlet program
- ⇒ The compilation & execution of normal java app takes place from command prompt.
- ⇒ The compilation of servlet prg takes place command prompt & execution takes place from servlet container

Note If above servlet program was typed, jdbc driver can be placed ojdbc14.jar in classpath & also in WEB-INF/lib folder is optional because this driver is built-in driver of jdk and tomcat server internally uses that jdk.

(25)
28/03/15

For above diagram based example Appⁿ refers Appⁿ 5 of the page no 113 & 114

⇒ If multiple web apps deployed in a server are using same 3rd party API then instead of placing 3rd party api jar file (1)WEB-INF\lib folder of every web app. ~~it is recommended to place~~ You can place that jar file only in & time in server library folder.

Server Library folder in Tomcat : <Tomcat_Home>\lib folder

Server Library folder in Weblogic domain Server:

<weblogic_Home>\middleware\user_projects\domains\<Domain name>\lib

Server Library folder in Glassfish domain server:

<Glassfish_home>\-\-\domains\<domain_name>\lib\ext folder

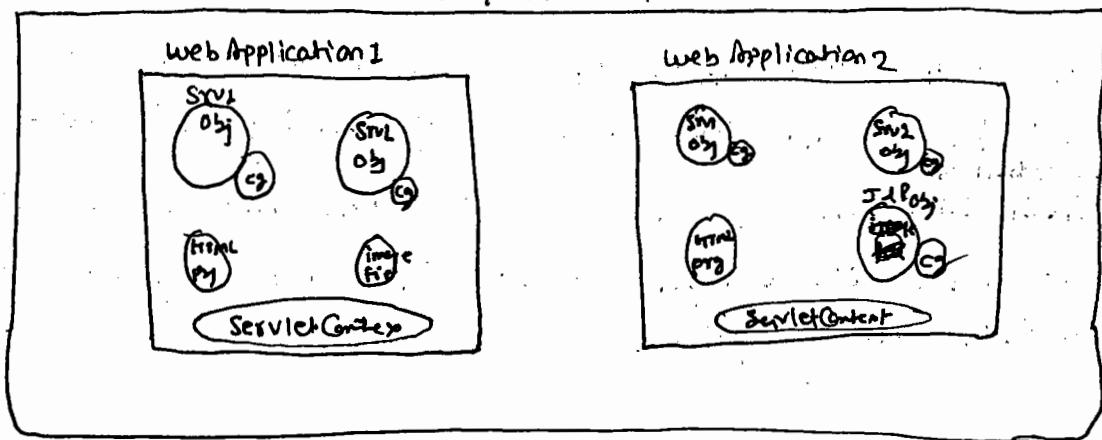
The above process is not recommended b'coz if we move web app from one server to another server we need to move jar files separately.

⇒ When Servlet program uses any API (class or interface) then the servlet container recognizes and uses that API by searching in following places for order.

- a) In WEB-INF\classes folder of current web app.
 - b) In the jar file placed in WEB-INF\lib folder of current web application
 - c) In the jar file placed in Server Library folder.
 - d) In the jar file placed in The library folder that is linked with Tomcat Server.
- Q) What is the diff b/w ServiceConfig and ServletContext obj?

WebServer/App Server

Ans)



ServletConfig obj (cg);—

- ⇒ It is one per our servlet class obj.
- ⇒ It is called right hand obj to our servlet class obj and this obj is useful to gather info about servlet program & to pass additional data to servlet program.
- ⇒ It is the obj of underlying ServletContainer supplied java class that implements javax.servlet.ServletConfig(I).
- ⇒ Using this Obj we can read init parameters from web.xml file being from servlet program.
- ⇒ ServletContainer creates this obj after creating our servlet class obj & init initialize that obj with our servlet class obj using init(-) method.
- ⇒ ServletContainer destroys servletConfig class obj along with our Servlet class object...
26
29/03/15

ServletContext obj (sc);—

- ⇒ It is one per web appn.
- ⇒ The data store in this object is visible & accessible in all the web resources program of web appn, so it is called the Global memory of web application.
- ⇒ It is the object of underlying servlet container supplied java class that implements javax.servlet.Servlet(I).
- ⇒ ServletContainer creates this obj either during server startup or during the deployment of web appn.

⇒ Servlet Container destroys this obj when web appn reloaded, stopped or undeployed.

⇒ Using this object

- we can gather global init params / context param from web.xml file.
- we can get underlying server info and servlet api version that it supports.
- we can get inputstreams pointing to resource.
- we can get absolute path of file.
- we write log messages to log file.

Note: To Access ServletContext obj directly or indirectly.

(Q) There are 12 webappn in webserver. In that 7 appn are there in running mode & 5 appn are there in stopped mode? Can u tell me how many servlet obj are currently available.

(A) $12 - 5 = 7$

Note: ServletContainer destroys our servlet class obj the moment web appn is stopped/reloaded/undeployed.

(Q) In a webappn 10 servlet programs are there. In that 3 servlet program are already requested & `<load-on-startup>` are enabled on other 3 servlet program. Can you tell me how many servlet objects are currently available in the web appn?

(A) $3 + 3 = 6$

our servlet class obj, request obj, response obj, ServletConfig obj, ServletContext obj will be created by servlet container but can access those obj in our servlet prg by using diff technique.

- ⇒ To access our servlet class obj we "this" keyword.
- ⇒ To access req, res obj we service(-,-) / doxxx(-,-) method param.

Different Ways of Accessing ServletConfig Obj

Approach1

```
public class TestServlet extends HttpServlet
```

```
public void init( ServletConfig config )  
    this.config = config;
```

Approach2

```
public class TestServlet extends HttpServlet
```

```
public void init()
```

```
    ServletConfig cg = getServletConfig();
```

```
public void service( HttpServletRequest req, HttpServletResponse res ) throws ServletException, IOException
```

```
    ServletConfig cg = getServletConfig();
```

Appr

Different ways of ServletContext obj

1) Approach 1

```
public class TestSrv extends HttpServlet {  
    public void doXXX(-,-) / service(-,-) throws SE, IOE {  
        // Access ServletConfig obj.  
        ServletConfig cg = getServletConfig();  
        // Access ServletContext obj.  
        ServletContext sc = cg.getServletContext();  
    }  
}
```

Approach 2

```
public class TestSrv extends HttpServlet {  
    public void service(-,-) / doXXX(-,-) throws SE, IOE {  
        // Access ServletContext obj.  
        ServletContext sc = getServletContext();  
        // Public method of GenericServlet  
        // This internally uses ServletConfig  
        // Obj to access ServletContext  
        // Obj  
    }  
}
```

Approach 3:-

ServletContext sc = req.getServletContext(); → from Servlet API 3.0 onwards

(27)

30/03/15

- ⇒ The standard principle of S/W industry is don't hard code values in your app that are changeable in future. It is recommended to gather such values from outside the app.
- ⇒ We can input to servlet prg from outside in two ways-
- Using Request Parameters (from data)
 - To gather non technical data end user like name, age, address and etc.
 - Use request Obj to read these values in servlet prg
 - Using Servlet Init parameters / context parameters
 - Useful to gather technical data from programmer through web.xml file like jdbc driver, url and etc.
 - To read init param values use servletConfig & to read context param values are visible to all servlet prgs of web app.

To place init params in web.xml

```
<Servlet>
  <s-n> --- </s-n>
  <s-c> --- </s-c>
  <init-param>
    <param-name> dbuser </param-name>
    <param-value> Scott </param-value>
  </init-param>
</Servlet>
```

To read init param value in Servlet prg :—

ServletConfig cg = getServletConfig();

String s1 = cg.getInitParameter("dbuser"); // gives scott

→ For improvised DBApp that we init parameter value to gather jdbc properties from web.xml. refer—

AppC of page no 114 to 117

721 to 745, 772 to 776 → logic to read init parameter values from web.xml, using ServletConfig obj

Using ServletConfig obj we can gather logical name servlet prg cfg given in web.xml file (using < servlet-name> tag)

This logical name becomes instance name when servlet container creates our servlet class object

eg.

```
ServletConfig cg = getServletConfig();
pw.println("<br> Logical name of servlet prg"
+ cg.getServletName());
```

Different ways of reading init param values :—

Approach 1)

String s1 = cg.getInitParameter("drive");

Note ⇒ Here we must know init param name to get its value.

Approach 2)

Enumeration e = cg.getInitParameterNames(); // gives all init param names

```
while( e.hasMoreElements() ) {
```

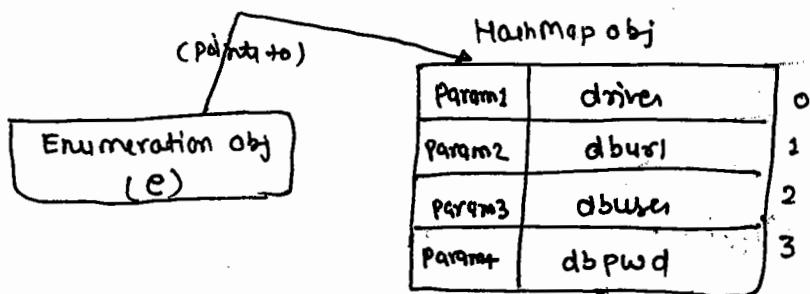
```
    String name = (String) e.nextElement(); // gives name
```

```
    String value = cg.getInitParameter(name); // gives value
```

```
    pw.println("<br>" + name + " ... " value);
```

```
}
```

→ Give all init param name & value.



Context Parameters / Global init Parameters

If multiple servlet prgs of a web appn are looking to use same technical data then instead of placing that technical data in web.xml. or init param values of every servlet prg it is recommended to place them in web.xml file only for 1 time as context param values & use `ServletContext obj` in servlet prgs to read those context param values.

Example. Instead of placing jdbc properties as servlet init param in web.xml file for every servlet prg it is recommended to place jdbc properties as context param values in web.xml file to use them in web multiple servlet prgs.

To place context params in web.xml file:-

```
<web-app>
  <context-param>
    <param-name> driver </param-name>
    <param-value> oracle.jdbc.driver.OracleDriver </param-value>
  </context-param>
  -----
  -----
  <servlet>
    -----
  </servlet>
  <servlet-mapping>
    -----
  </servlet-mapping>
  <servlet>
    -----
  </servlet>
  <servlet-mapping>
    -----
  </servlet-mapping>
```

To read context param value in servlet prg:-

```
ServletContext sc = getServletContext();
```

```
String st = sc.getInitParameter("driver");
```

⇒ For improved DBApp web app that uses context parameters
to maintain jdbc Properties refer Appn 7 of the page no
117.

862 to 867 , 862 to 881 → context parameters having jdbc properties

910 to 916 → code that uses ServletContext obj to read jdbc properties from context parameters of web.xml.

Different ways of reading context param values:-

Approach:

//Access ServletContext obj

ServletContext sc = getServletContext();

String s1 = sc.getInitParameter("driver");

Note- we must know the name of context parameter here.

Approach:

Enumeration e = sc.getInitParameterNames(); //give all context param names

while (e.hasMoreElements()) {

String name = (String) e.nextElement(); //give name

String value = sc.getInitParameter(name); // give value

pw.println(name + " " + value);

}

Using ServletContext obj we can gather multiple details about underlying web app & its web resource program as shown below—

ServletContext sc = getServletContext();

pw.println("
 class name of Servlet Context obj " +

sc.getClassName());

(org.apache.catalina.core.ApplicationContextFacade)

```
pw.println("<br> Servlet API Version " + sc.getMajorVersion())
          + "<br> " + sc.getMinorVersion());
(3.0)
```

```
pw.println("<br> Server Info " + sc.getServerInfo());
(Apache Tomcat 7.0)
```

```
pw.println("<br> Context path of web app " + sc.getContextPath());
/DBApp
```

```
pw.println("<br> Real path of input.html"
          + sc.getRealPath("input.html"));

```

D:\Tomcat7\webapps\DBApp\input.html

```
pw.println("<br> MIME type of input.html"
          + sc.getMimeType("input.html"));
text/html
```

81/03/15

Q) What is difference b/w code placed in constructor & the code placed in init() of Servlet program?

- A) ⇒ We can't access ServletContext obj without ServletConfig obj
- ⇒ ServletContainer creates ServletConfig obj after constructor execution & before init() method execution, so ServletConfig obj is not visible & ServletContext obj is not accessible in the constructor. But same obj are visible & accessible in the init() method.
- ⇒ The discussion says, we can't use servlet init param & context param in the constructor but same thing can be used in init() method.

With respect to Tomcat 9

~~x~~ ~~x~~ ~~x~~

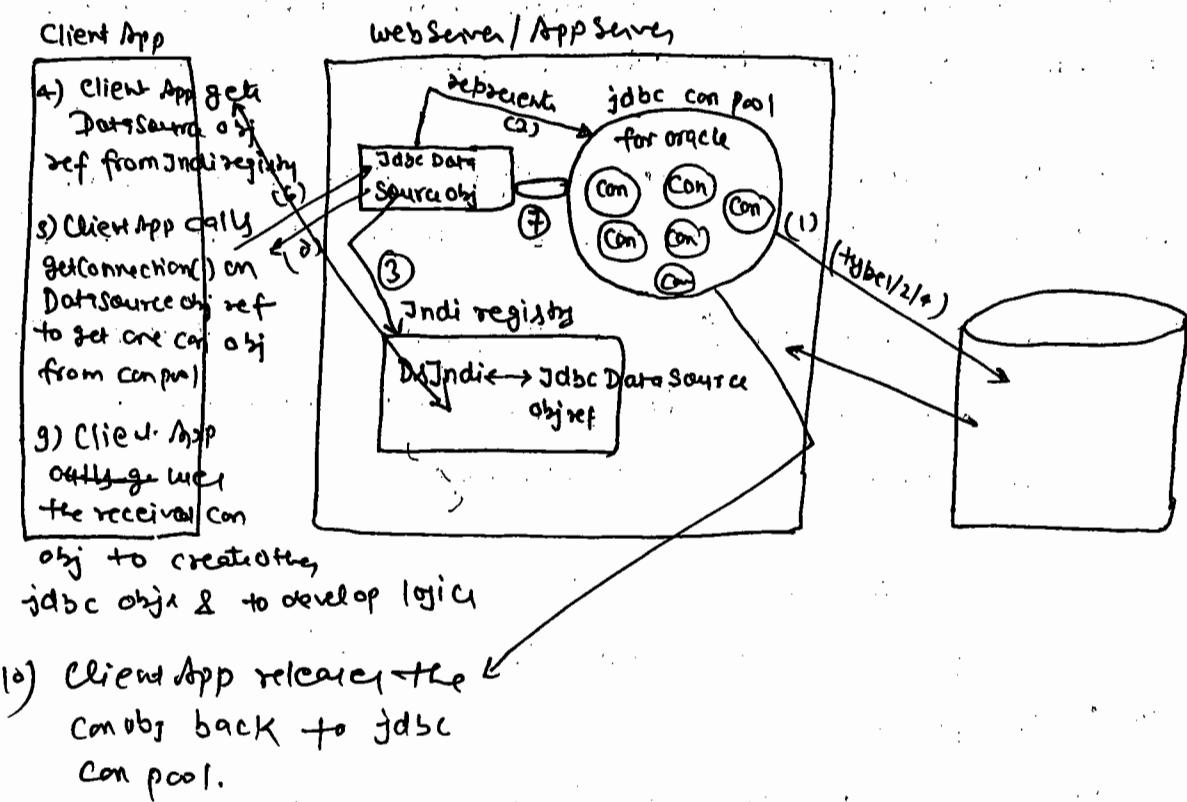
- ⇒ If we give same name to multiple init param then the first value of that list will be taken as final value.
- ⇒ If we give same name to multiple context param then last value in the list will be taken as final value.
- ⇒ We can give same name to init param & context param becoz we can access init param using ServletConfig obj & context param using ServletContext object.

Working with Server Managed Jdbc Con pool:-

- ⇒ It is the jdbc con pool that is created & managed in webserver/app server.
- ⇒ Jdbc Datasource represents this jdbc con pool & that

dataSource obj ref will be placed in jndi registry for global visibility.

- ⇒ To provide global visibility to any obj or obj ref placed it in Jndi registry.
Every server gives in built-in registry.
- ⇒ Data Source Obj means the obj of java class that implements javax.sql.DataSource(I)
- ⇒ All jdbc con objs in jdbc con pool represents connectivity with same DB like for example jdbc con pool for oracle means all jdbc con objs in that jdbc con pool represents connectivity with same oracle DB like.
- ⇒ In realtime only PL/TL creates jdbc con pool & DataSource in the server & remaining all member will use that jdbc con pool.



with respect to the diagram:-

- 1) Server uses type 1/2/4 jdbc driver to interact with DB & to create set of jdbc con objects in the connection pool.
 - 2 & 3) DataSource obj represents Jdbc con pool & that object reference will be placed in Jndi registry for global visibility.
- ⇒ For remaining points refer diagram.

There are two types of jdbc connection—

- a) Direct Connection (using `DriverManager.getConnection(...)`);
- b) Pooled Connection (Using `DataSource.getConnection(...)`);

Procedure to create Jdbc DataSource & Jdbc con pool in "NtAdvJava9" Domain Server of weblogic 10.3.

Step1) Start NtAdvJava9 domain ^{new} of weblogic

Step2) Open Admin Console to "NtAdvJava9" domain Server.

`http://localhost:7090/console`.

Username: weblogic_javasam

Password: JavaSam1

→ Login

Step3) Here Create jdbc con pool for oracle & jdbc data source.

DataSource → New → Name DST ↗ logical name

JndiName [DSJndi]

DBtype [oracle]

Driver Oracle Driver thin for service connection

Next

DB name logical name (sid)

Hostname

Port

DB Username

Password

→ Test Configuration → Next →

admin server →

Launch DSI → connection pool tab

initial capacity

max

capacity increment → specifies of no. of new connection obj that should be created in the connection pool when there is a need of creating new connection object.
→

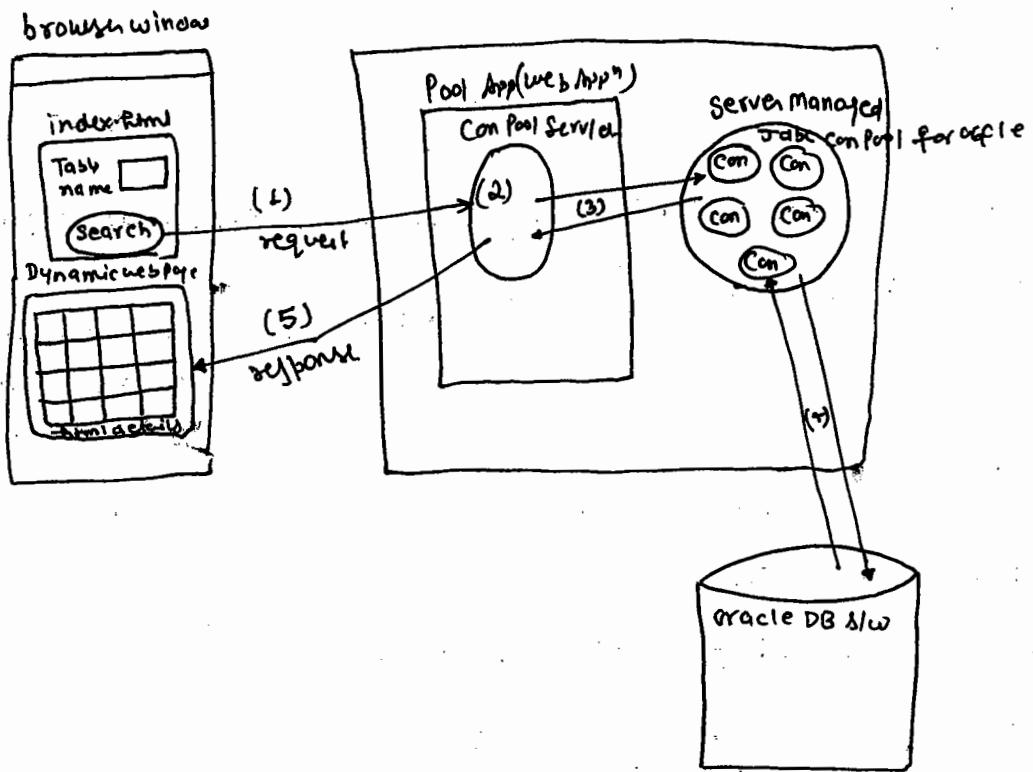
→ Advance

Shrink frequency → After every 900 sec. the idle connection obj from connection pool will be removed.

At the end of above steps the datasource obj of that points to con pool will be placed in Trns apj for global visibility.

8pm

29
01/04/15



for above diagram based Example Appn refer Appn 10 of the page no 121.

- ⇒ Java Appn interacts with DB s/w by using jdbc code. Connectivity b/w java App & Db s/w will be represented by jdbc con object.
- ⇒ InitialContext obj represents the connectivity b/w Java App & Jndi Registry. we can perform following opertions on jndi registry by using InitialContext obj
- Bind operation (keeping obj in registry)
 - Unbind operation (removing obj from registry)
 - Rebind operation (replacing obj with new object)
 - lookup (getting obj from registry)

⇒ To write above code we need Jndi API & it is part of JSE module. Jndi API pkgs are javax.naming & its subpackage.

Example code to prefer lookup operation:

```
InitialContext ic = new InitialContext(); // connect with Jndi Registry  
DataSource dt = (DataSource) ic.lookup("D1Jndi"); // gets Datasource  
obj ref from Jndi registry
```

In Java App we can use two types of JDBC connection pool

a) Client side JDBC connection pool.

e.g. Apache DBcp, C3P0, proxool.

If ur app is running at side the server or standalone or two-tier app.

b) Server side JDBC connection pool:

cf webserver/app using managed JDBC connection pool

If your app is deployable (like webapp / ejb config)

In webserver or AppServer then use server managed/pool JDBC connection pool.

⇒ In real time web app development they prefer using server managed JDBC connection pool.

⇒ Procedure to create JDBC connection pool for oracle in "NetBeans Java Domain" in Glassfish 3.x

Step1 Place ojdbc6/14.jar in <Glassfish_home>\glassfish\glassfish3\domains\NtAdvJava9\lib\ext folder

Step2 Start NtAdvJava9 Domain Server

<Glassfish_home>\--\--\bin\asadmin start-domain NtAdvJava9Domain

Step3 Open Admin Console

http://localhost:4845

Submit Username, password.

Step4 Create JDBC Con pool for oracle

Admin Console → Resources → JDBC → JDBC Connection Pools → New screen

Pool name

Resource Type

DB Driver vendor

Additional Property

User: scott

DB name: xe

Password: tiggy

Server name: localhost

Service name: xe

URL: jdbc:oracle:thin:@localhost:1521:xe

Port number: 1521

Finish

Step5 Create JDBC Data Source / Resource pointing to the above JDBC Con Pool

Admin Console → Resources → JDBC → JDBC Resources → New →

Instance Name

Pool name

→ OKS

url: http://localhost:8080/PointApp

(30)

02/04/15

Procedure to create jdbc con pool in Tomcat 6/7/8 server.

1) Download following zip file from Google

apache-tomcat-jdbc-1.1.0.1-bin.zip

And extract zip file to get "tomcat-jdbc.jar"

2) place "tomcat-jdbc.jar" and ojdbc14.jar files <Tomcat-home>/lib folder.

3) Place the following <Resource> tag under <Context> in <Tomcat-home>/conf/context.xml file.

<Resource name = "DsJndi" type = "javax.sql.DataSource"

factory = "org.apache.tomcat.jdbc.pool.DataSourceFactory"

driverClassName = "oracle.jdbc.driver.OracleDriver"

url = "jdbc:oracle:thin:@localhost:1521:xe"

username = "scott"

password = "tiger"

initialSize = "10"

maxActive = "20" maxIdle = "10" minIdle = "5"

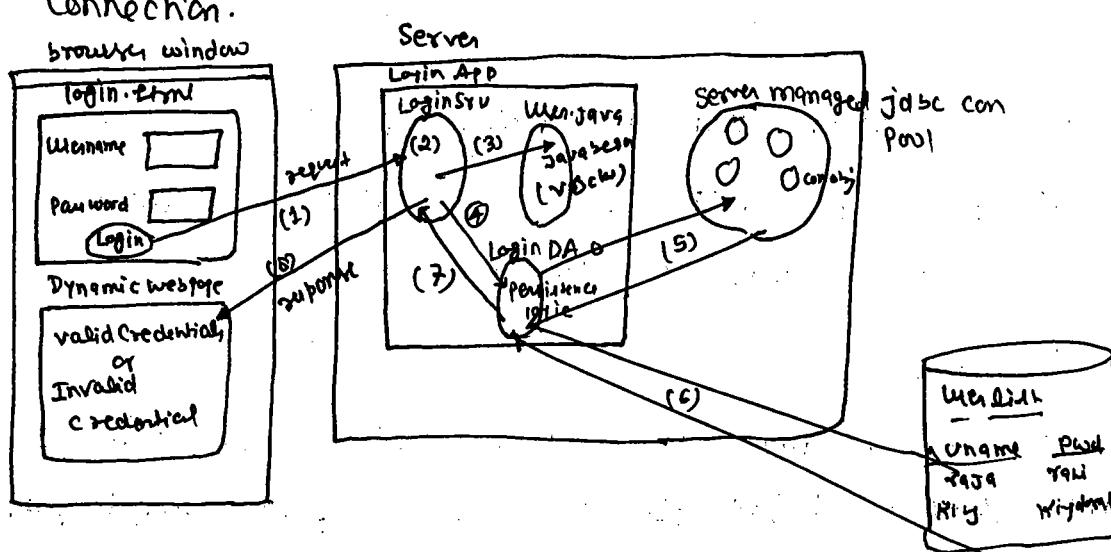
maxWait = "100" validationQuery = "Select sysdate from DUAL"

/ >

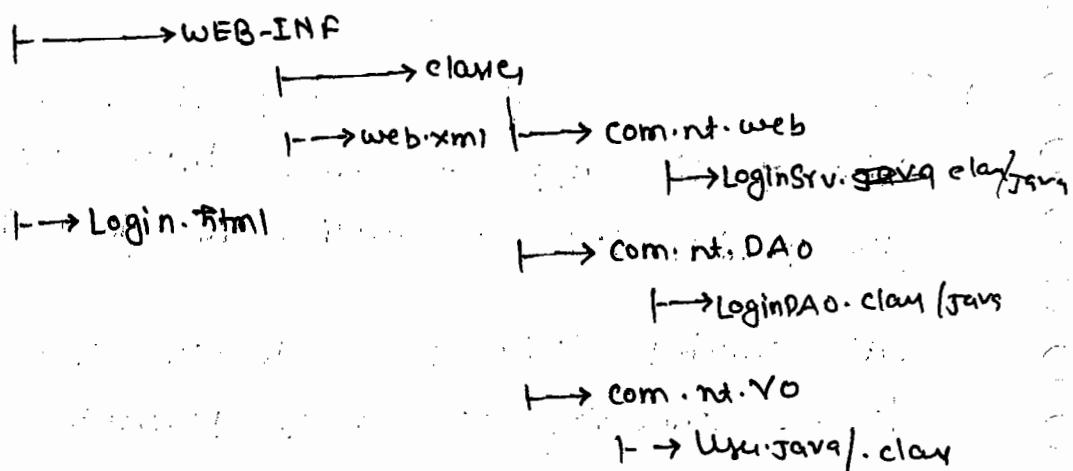
4) Restart the server...

Servlet to DB & w communication using DAO (Approach S)

- ⇒ The java class that separates persistence logic from other logics of the App & makes those logics are reusable and flexible logic to modify is called DAO.
- ⇒ DAO contains logic to create con, close connection & to use connection to develop persistence logic.
- ⇒ Don't write persistence logic in servlet prgs and place them in DAO class.
- ⇒ The Java class that contains getter & setter method, is called Java bean. setter methods are useful to write data properties (variable) and getter methods are useful to read data from properties.
- ⇒ Java Bean whose obj holds input values given by form page is called VO class (Value Object class)
- ⇒ DAO class can use direct jdbc con or pool jdbc Connection.

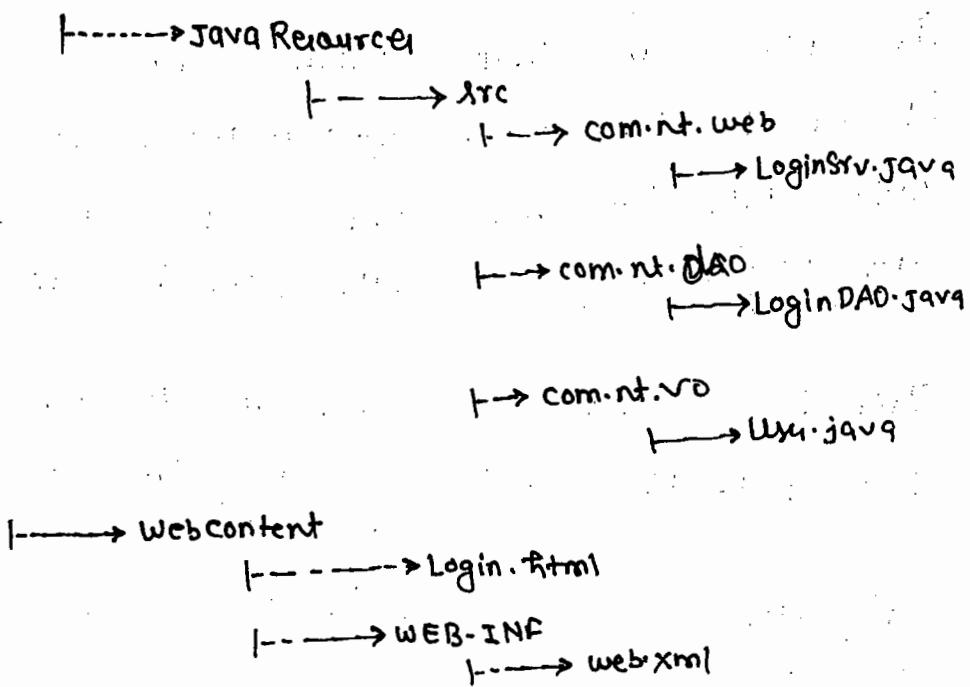


Login App



Using Eclipse IDE

LoginApp (Dynamic Web Project)



Note- The java bean that is acting as VO class always useful to pass i/p data from one resource to another resource in the form of object.

Steps using Eclipse:-

- 1) Create Java Dynamic web project having name LoginApp
- 2) Add servlet-api.jar file to the classpath build path

3) Develop User.java

User.java

```
public class User {
```

// bean properties

```
private String user, password;
```

// 0-param constructor

⑤ // 2-param constructor

// write getter and setter

--- right click → source → generate getter and
--- --- --- setter methods

```
}
```

LoginDAO.java

```
public class LoginDAO {
```

```
private static final String AUTH_QRY
```

```
= "Select count * from userlist where uname=?  
and pwd=?"
```

(g) public boolean authenticate(User user) {

```
try {
```

```
Connection con = makeConnection();
```

```
PreparedStatement ps = con.prepareStatement(AUTH_QRY);
```

```
ps.setString(1, user.getUserName());
```

```
ps.setString(2, user.getPassword());
```

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

```
boolean flag;
```

```
int cnt=0;
if(rs.next()) {
    cnt = rs.getInt(1);
}
if(cnt != 0)
    return true;
else
    return false;
```

} // try

```
} // catch (Exception e) {
    return false;
}
```

} // method

public Connection makeConnection() throws ContextException

```
public Connection makeConnection() throws Exception
{
```

```
// get con obj from jdbc con pool
InitialContext ic = new InitialContext();
DataSource ds = (DataSource) ic.lookup("java:/comp/env/
                                         Dugnali");
Connection con = ds.getConnection();
```

```
return con;
}
```

}

LoginSrv.java

extends HttpServlet
public class LoginSrv {

 public void doGet(-, -) throws SE, IOException

 {

 // General Setting

 PrintWriter pw = res.getWriter();

 res.setContentType("text/html");

 // read form data

 String user = req.getParameter("user");

 String pass = req.getParameter("pass");

 // Keep form data in user obj (VO class obj)

 User user = new User(user, pass); (8)

 // User DAO class persistence logic

 (K) boolean flag = new LoginDAO.authenticate(user); (i)

 if (flag)

 pw.println("Valid Credentials");

 else

 pw.println("Invalid Credentials");

 pw.close();

 } // doGet(-, -)

 public void doPost(-, -) throws SE, IOException

 {

 doGet(req, res);

 }

}

web.xml (e)

Configure Login.html as welcome page.

Configure com.ntt.web.LoginSrv with /loginurl url pattern.

Note:- Normal classes need not be configured in web.xml file.

Login.html (b)

```
<h1> <center> Login Page </center> </h1>
```

```
<form action = "loginurl" >
```

User: <input type = "text" name = "user" />

Password: <input type = "password" name = "pwd" />

<input type = "submit" value = "Login" >

⇒ Configure tomcat server with eclipse IDE.

```
window → preferences → Server → Runtime Environment → Add  
Apache Tomcat 7. → Next → Tomcat Installation folder  
→ Finish → OK
```

⇒ Run the project

→ Run

Right click on project → Run on server → Next → Finish

request url: http://localhost:2525/LoginApp/login.html (a)

(31)

03/04/15

⇒ In real time programmer prefers DAO class with ~~dbms~~ managed con pool support for Servlet to DB S/W communication.

Understanding Protocol http:-

/* The text that allows non sequential access is called hypertext */

⇒ Protocol HTTP is a application level protocol that runs over network level protocol called TCP/IP having set rules and guidelines to transfer hyper text between browser and web server/ App server.

⇒ The text that allows non-sequential access through hyperlinks is called hyper text.

⇒ To create hyper text we use HTML. Similarly to send hypertext browser to server server to browser the protocol "http" will used.

⇒ The latest version of protocol http is 1.1.

⇒ When browser sends http request to server it carries multiple details along with request. The details are -

http request method , path of resource, request header, request body and misc info

⇒ When web resource prg (web comp) sends response to browser the response carries multiple details like

Protocol name, version, response header, response body and etc...

⇒ http request carries all details given by browser having certain standard structure and same with http response...

⇒ The ~~other~~ data that will be carried along with the request can taken by h2p2 data--.

h → http request method (GET/ POST/ --)

t → http request headers (User-Agent, Referer, Accept--)

P → path of resource (/DateApp/tut1)

P → ~~param~~ ~~request~~ param

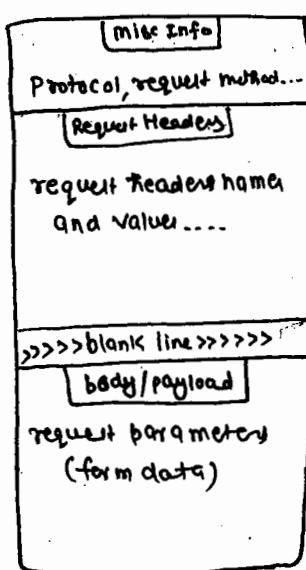
⇒ The data that carried along with response can taken as SCH data,

S → http response status code (200-559)

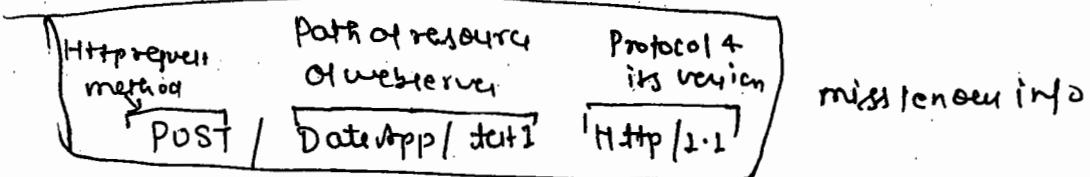
C → response content (the msg of pw.println (-) method)

H → response headers (refresh, contentType, contentLength, --)

blocks diagram of http request



Http request with detailed data:-



Host: www.NatrajSoft.com → Host to connect

User-agent: Mozilla / IE 5.0

Accept : text/html, text/xml, image/jpeg, application/x-mixed

Accept-language: en-US

Accept-encoding: gzip, deflate

Accept-Charset: ISO-8859-1

Keep-Alive : 300

Connection : keep-alive

>>>>> blank line <<<<<<<

like = query & prod = payload request body / payload

(form data / req parameters)

a) what is difference b/w request params & request headers

A)

request params

request headers

1) Content of request body.

Content of request headers section

2) holds end user supplied data.

holds browser supplied data

3) param names are not unique and are userdefined

header names are fixed & predefined.

4) These are optional in the request

Mandatory in the request

⇒ All the data that comes along with the request will be received by server and will be stored in request obj. Servlet program can use this request obj to gather that data.

HTTP://localhost:2020/form/app/furl? sno=101&sname=fyq1&add=fyq1

Different ways of gather request param

a) req.getParameter(-) method

```
String s1 = req.getParameter("sno") // gives 101
```

b) req.getParameterNames() method

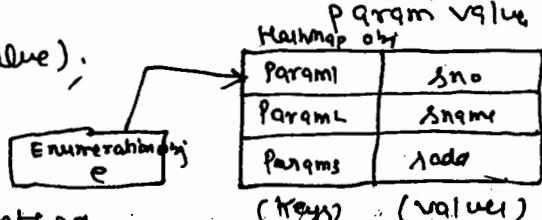
```
Enumeration e = req.getParameterNames(); // gives all req param name  
while (e.hasMoreElements()) {
```

```
String name = (String) e.nextElement(); // gives each req param name
```

```
String value = req.getParameter(name); // gives each req
```

```
Sopln (name + " --> " value);
```

```
}
```



c) Using req.getParameterValues() method

```
String addrs[] = req.getParameterValues("sadd");  
→ {"fyq1", "fyq1"}
```

```
String addrs = req.getParameterValues("sadd")[1];  
// gives fyq1
```

d) Using req.getParameterMap() method

Map<String, String[]> map = req.getParameterMap();

Note:- We can use either ServletRequest obj or HttpServletRequest obj here.

(32)
~~04/04/15~~

Note:-

In Approach 1, 2 if request param contains multiple values we will get only 1 value. In Approach 3, 4 we will get multiple values for the same.

improved Approach 4:-

```
Map<String, String[]> map = req.getParameterMap();
```

```
Map<String, String[]> map = req.getParameterMap();
```

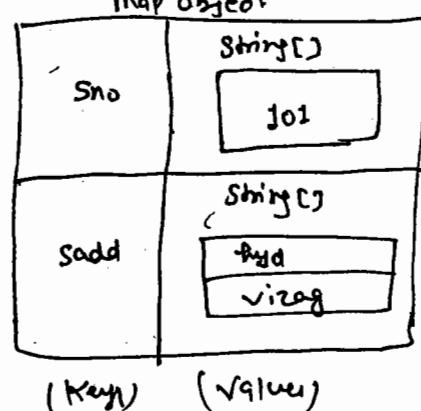
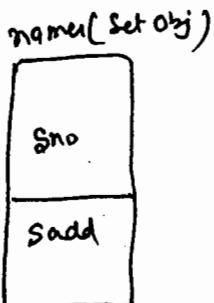
```
Set<String> names = map.keySet();
```

```
for (String name : names) {
```

```
    String values[] = map.get(name);
```

```
    pw.println("<br> " + name + " --> " + Arrays.toString(values));
```

```
} // for
```



Note:- Arrays are objects in Java so they can be stored as the elements of collection data structure.

Different ways of reading req headers & their values from client generated request:

Note For this we must need HttpServletRequest object.

Approach1) Using req.getHeader(-) method

```
String s1 = req.getHeader("User-Agent"); // gives browser name
```

Approach2) Using req.getHeaderNames(-) method

```
Enumeration e = req.getHeaderNames(); // gives all header name
```

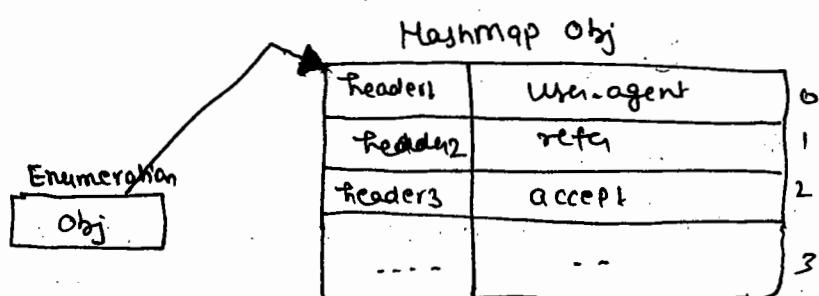
```
while (e.hasMoreElements()) {
```

```
    String name = (String) e.nextElement(); // gives header name
```

```
    String value = req.getHeader(name); // gives header value
```

```
    pw.println(name + " ----> " + value);
```

```
}
```



Q) How to find the browser name using which our servlet prg is requested?

A) Use "User-Agent" request header as shown below —

```
String s1 = req.getHeader("User-Agent");
```

⇒ We can change accept language request header value by using browser setting

IE → tools → Internet Options → language → add
→ choose language.

List of http request headers:-

Accept, Accept-language, refer, connection, Keep-alive, cookie,
Accept-encoding user-agent, cache-control and etc....

⇒ For more information on http request refer Page no 85 to 87.
and also refer Page no 171 to 181.

Gathering millennium info from client generated request being from Servlet program:—

//methods of javax.servlet.Servlet(I)

pw.println ("req Content Type" + req.getContentType());

// MIME type or null

pw.println ("req content length" + req.getContentLength());

// length in bytes or -1

pw.println ("req protocol" + req.getProtocol());

// HTTP/1.1

pw.println ("req Scheme" + req.getScheme());

// http

pw.println ("req Remote port" + req.getRemotePort());
// 45679 (browser port)

pw.println ("req Remote Host" + req.getRemoteHost());
// client machine hostname

pw.println ("req Remote addr" + req.getRemoteAddr());
// client machine IP Address

```
pw.println ("Server name" + req. getServerName());  
//localhost
```

```
pw.println ("Server port" + req. getServerPort());  
//2525
```

//using the methods of HttpServletRequest.

```
pw.println ("req method/mode" + req. getMethod());  
//Get
```

```
pw.println ("query String" + req. getQueryString());  
//Sno=101&sname=raj
```

```
pw.println ("Context path" + req. getContextPath());  
//DateApp
```

```
pw.println ("Servlet path/ URL pattern" + req. getServletPath());  
//Hello
```

```
pw.println ("correct uri" + req. getRequestURI());  
// /DateApp/Hello
```

```
pw.println ("request url" + req. getRequestURL());  
// http://localhost:2525 / DateApp /Hello
```

B) What is the difference b/w URL & URI?

A) URL → Universal Resource Locator :- Using this we can locate resources from anywhere.

URI = Universal Resource Identifier:-

Details.

Eg. Refer above

Just gives resource identification

33
05/04/15

-: SUNDAY :-

Http Response:-

The output generated by servlet prg goes to browser window as response using protocol "http". So it is called "Http response". This response contains multiple details and they are called "SCH" details.

S ----> http response status code (100-599)

(100-199) ----> info

(200-299) ----> success

(300-399) ----> Redirect

(400-499) ----> Incomplete

(500-599) ----> Server Error

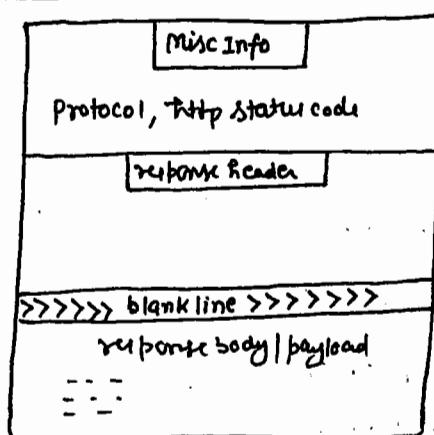
C ----> http response content (The code msg placed in pw.println(-))

H ----> http response headers (refresh, contentType, contentLength, cache-control,)

(Gives instructions to browser window)

For more info on status code refer page no 179 to 183

Block Diagram of http response:-



Http / 1.1 200 ok → Http Status Code

Set - Cookie : JSessionID = 148937ABEA179EA
Content - Type : text/html
Content - Length : 1354
Date : Sun Apr 2015 9:17:34 GMT
Server : Apache - Tomcat / 5.0
Connection : close
<<< Blank Line >>>
<i> welcome to servlets </i>

response headers

List of http response headers

Set - cookie

refresh

contentType

ContentLength

pragma / cache - control

(1.0) (1.1)

expires

LastModified

location

Server

Date

Connection

and etc

Different methods to set response header values :-

res . setHeader (,)

res . setIntHeader (,)

`res.setHeader(-,-)`

`res.setContentType(-)`: set given MIME type as response content type....

(Q) How to refresh webpage dynamically automatically at regular intervals?

A) Use response header "refresh" as shown below-

`res.setHeader("refresh", "10");`

`res.setIntHeader("refresh", 10);`

⇒ This auto refresh is very useful while displaying live game scores, stock market share value.

⇒ `res.setContentLength(-)` method gives instruction to browser to become ready to display the given minimum content

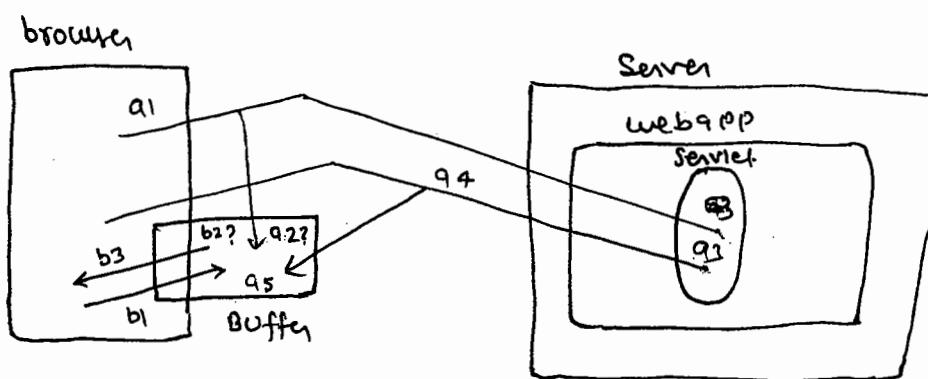
⇒ When browser gives request to web component the servlet container creates one set of `request.response` object.

⇒ This response obj contains default header with values & misc info. To override existing header values `res.setHeader(-,-)` method and to add new headers we `res.addHeader(-,-)` method.

For more info of http response refer page no 87 to 89.

34
07/04/15

- ⇒ Buffer/cache is a temporary memory that holds data for temporary period.
- ⇒ In Client Server env... the cache at client side holds the server supplied results and uses that data across the multiple same requests to reduce network round trips between browser window and server.
- ⇒ Every browser holds the server generated webpage in buffer/cache and uses that to display the web page across the multiple same request. This reduces network roundtrips.
 e.g. The web resource that gives seven wonders of world uses the cache/buffer of browser.



$a_1 \rightarrow a_6$: 1st request } Both
 $b_1 \rightarrow b_3$: 2nd request } Same request

- Browser maintains buffer by default. If web resource program generating dynamic output time to time. (like showing stock share values) then we should browser buffer to get updated results every time, we should "cache-control" or pragma response header.

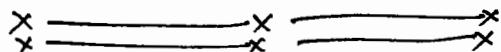
In Servlet program

re1. setHeader ("pragma" , "no-cache"); //In http 1.0 version

re1. setHeader ("Cache-control" , "no-cache"); //In http 1.1 version

Note - Every web resource program can give instruction to browser whether the output generated by web resource program should be placed in buffer or not in browser window.

— : Servlet Communication : —



Servlet Communication

Browser to Servlet Communication

- 1) Request \leftrightarrow Response
- 2) Sending Error message
- 3) Redirection
 - a) Using Hyperlinks.
 - b) Using response headers
 - c) Send redirection

Applet - Servlet Communication

Servlet to Servlet (or) Web component Communication

- 1) By forwarding request
 - 2) By including response
- Request
Dispatcher

Browser to Server Communication:-

① request → response :-

browser gives request to servlet & servlet generates response
(The one that we have used so far)

② Sending Error message -

→ Use res.sendRedirect(status code, error message);

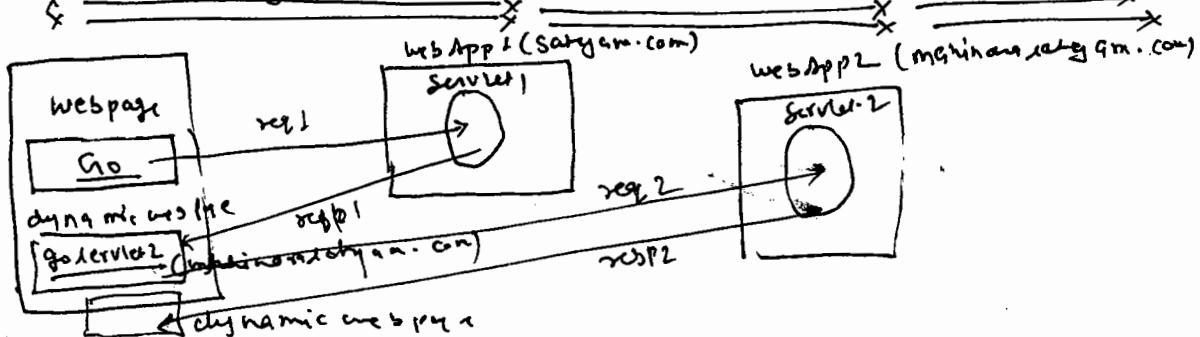
③ Redirection -

→ The process of redirecting a request to another web resource program when request is given to regular web resource program is called request redirection. It can be done in 3 ways as discussed above.

→ This concept is useful when one company acquires another company because request coming to 1st Company website should be redirected to another Company website.

e.g. Oracle Corp. acquired Sun Microsystems. So the request given SUN.com should be forced to Oracle.com.

3.9) Redirection by link (Hyperlink) (Not recommended to use) :-



So far we have given request to one
and receive the output but sometimes this one
resource program needs to talk with other web resource
program of same web application.

So in that situation we can use either servlet to
servlet communication or redirection concept.

Note- The problem with above hyperlink based redirection
is if -

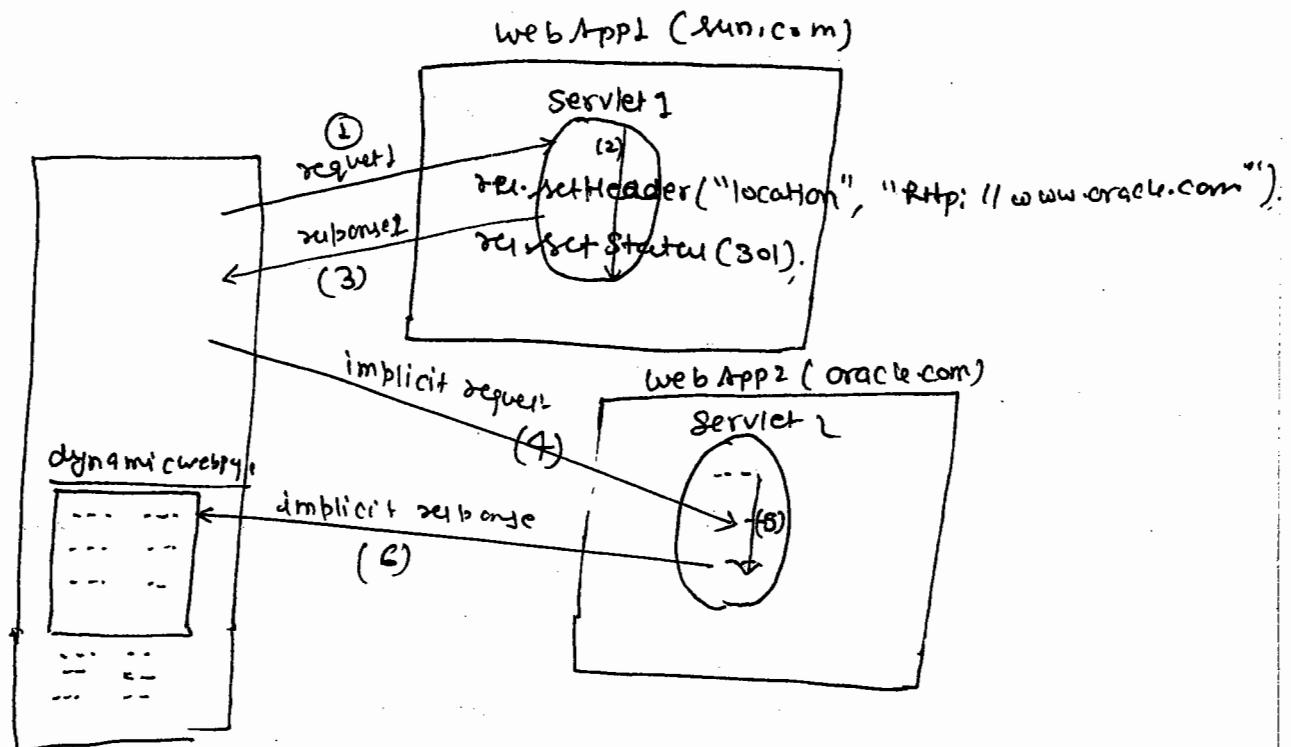
if user forgets to click on dynamically generated
hyperlink then redirection does not take
place.

Redirection by using Response Header: →

Instead of asking end user to click on one hyperlink to
navigate the request to another website we can use
response header "location" in the source servlet program
to redirect request to destination program. In this process
status code of source servlet program generated response
must be in b/w 300 to 399.

B'coz of the status code after receiving response browser
generates implicit request to new web resource program
based on the value of response header location.

Redirection using response headers:-



⇒ In the above diagram browser automatically generate implicit request to Servlet2 of oracle.com by seeing response status code as 301 and by using the response header location value.

Note:- Working with response header location directly is not recommended process.

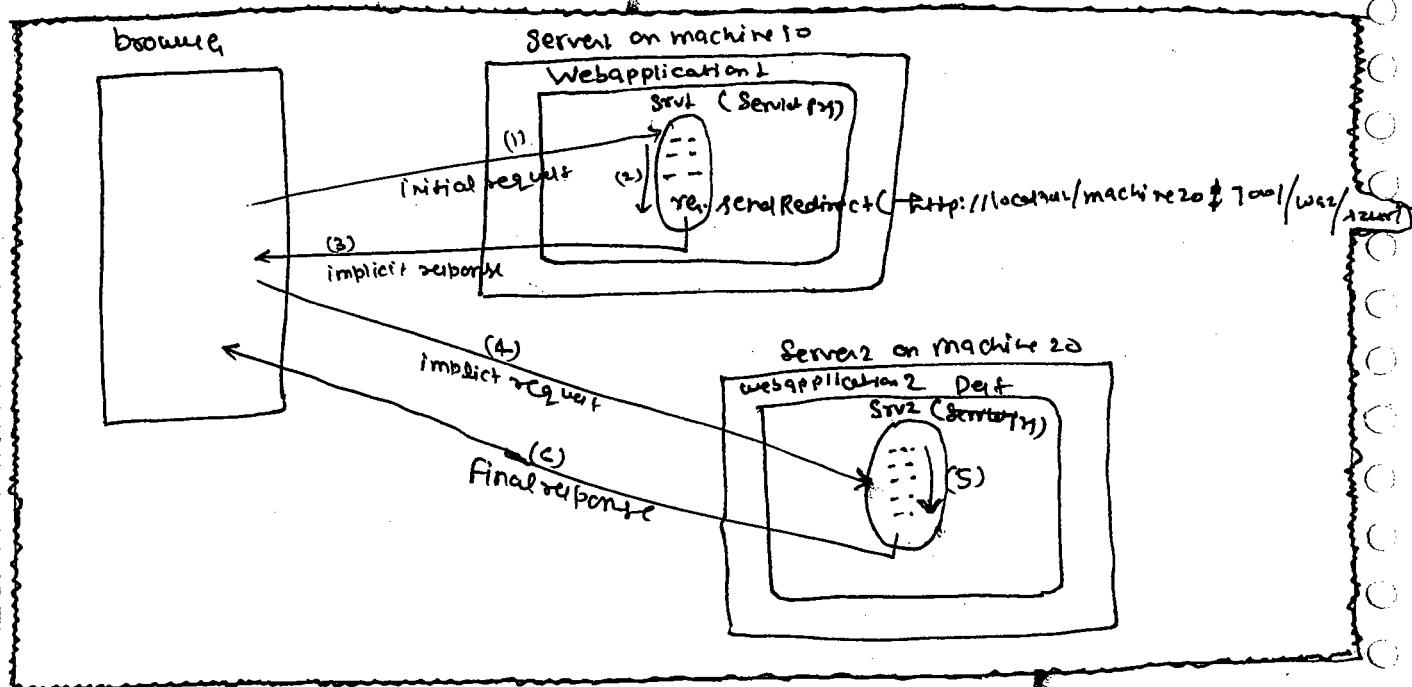
In this Approach source program should set response status code in between 300 to 399 manually. Doing this is not also recommended process.

To overcome the problem of Above 2 approaches we use redirection concept.

~~08/04/15~~

3.c Send Redirection (Using `re.sendRedirect()`)

→ Here there is no need of working with hyperlinks & response header, status code manually. All redirection activities will be taken care by `re.sendRedirect()` automatically.



W.R.T diagram

- 1) Browser gives initial request to Srv1 program.
- 2) All the statements of Srv1 prg including `re.sendRedirect()` method execute.

Note:- Becoz of `re.sendRedirect()` method execution the response status code changes to 300-303 and "location" header will be generated aligned with url kept in `re.sendRedirect()`

- 3) Srv1 prg generates implicit response to browser window.
- 4) Browser window gathers the "location" response header value and gives implicit request to Srv2 prg.

- 5) All the logics of Srv2 prg execute.
- 6) output generated by Srv2 prg goes to browser window as final response.

⇒ res.sendRedirect(-) performs redirection mode of servlet communication
⇒ Here source prg talk with dest prg after having one network round trip with browser window.
⇒ Source prg and Dest prg will not use same request, response Obj., So the request data coming to source prg (Srv1) is not visible and accessible in Dest prg (Srv2)
⇒ To pass additional data b/w source & Dest prgs append query String to url of res.sendRedirect(-) method as shown below -

In Srv1 prg

```
res.sendRedirect ("http://localhost:7002/waz/Srv1? sno=101 & sname=mango")
```

In Srv2 prg

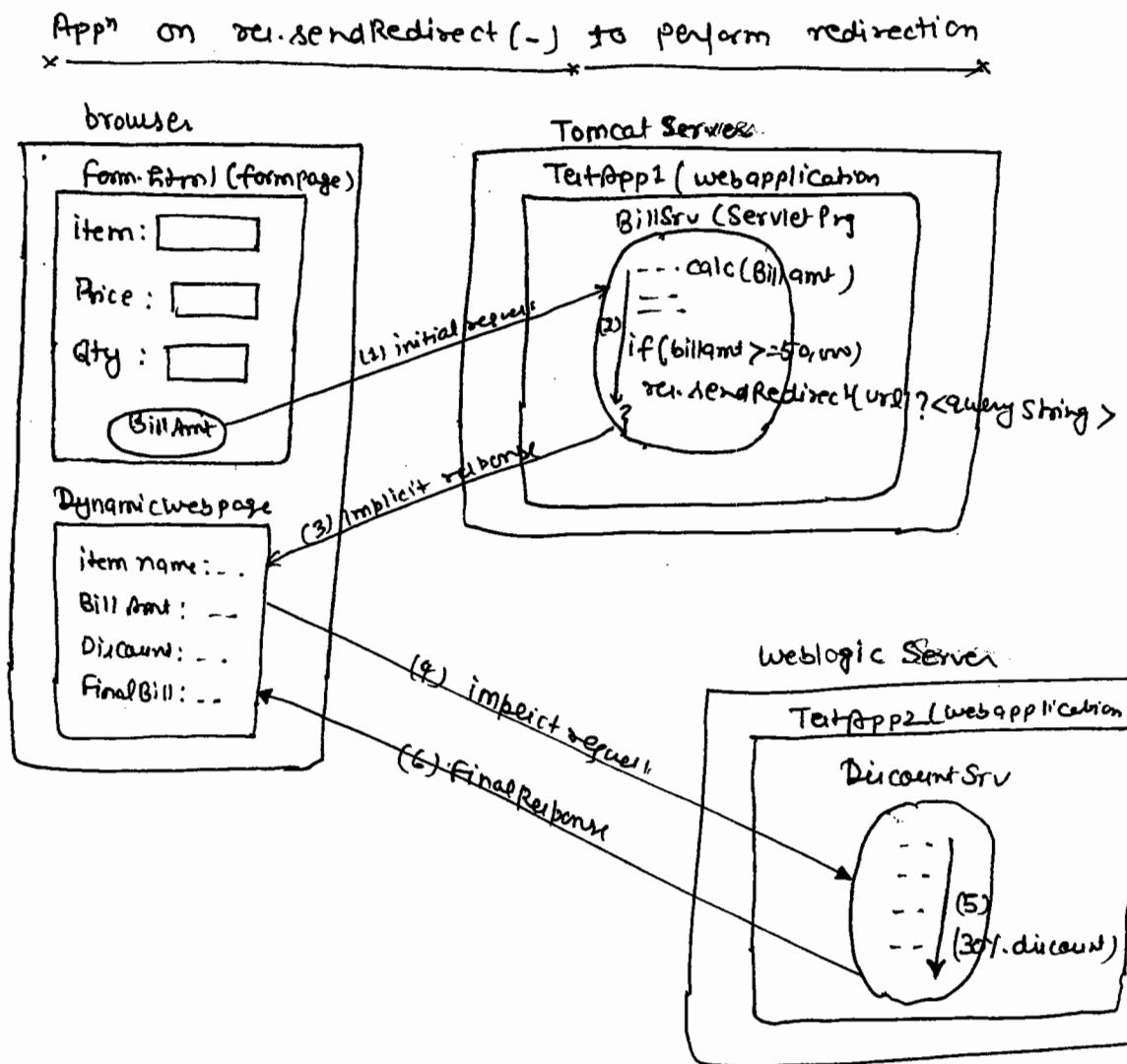
```
String s1 = req.getParameter ("sno");
```

```
String s2 = req.getParameter ("sname");
```

- ⇒ The entire response output of Source prg will be discarded and only html output of Dest prg goes to browser.
- ⇒ Srv1, Srv2 prgs can be there in same web application or can be there in two different web applications of same or different ~~resource~~ servers.
- ⇒ Srv2 prg can be developed in any web technology like servlet.jsp, html, asp, asp.net and etc.
- ⇒ If Company A acquires Company B the requests given to B.com will redirected to A.com.

e.g. Oracle Corp has acquired Sun Microsystems, so the request given to sun.com will be redirected to oracle.com.

Example.



In the above appln query String will be appended to the url of response. `sendRedirect` method to item name Bill Amt to ~~to discount Srv~~ DiscountSrv from BillSrv

TestApp1 (Deploy in Tomcat Server)

→ WEB-INF

→ form.html → classes

→ web.xml → BillSrv.java

→ com.nt → BillSrv.class

TestApp2 (Deploy in Weblogic Server)

→ WEB-INF

→ classes

→ web.xml

→ DiscountSrv. JSP

→ com.int

→ DiscountSrv. class

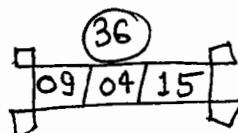
⇒ For the above Example App refer App2 of page no 136 to 138.

2008 BillURL

2051

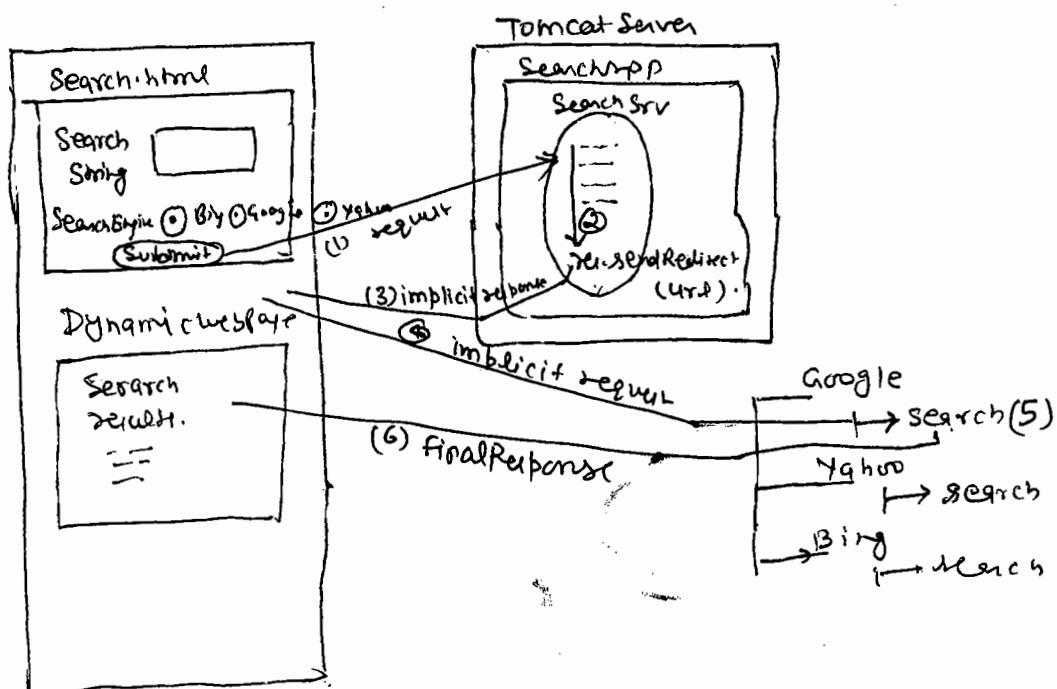
2056 ? Bill=

2082, 2083



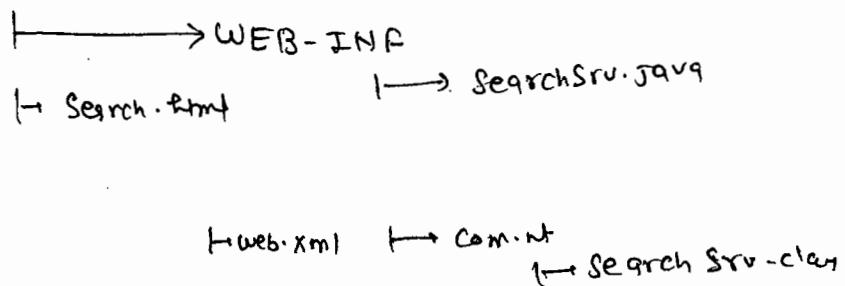
⇒ In real time sendRedirect() is very useful to redirect the request to unknown technology based web resource program of internet website

e.g. If u want to develop our local web app or hub for multiple search engine then we need to use send redirection to our servlet



⇒ The SearchSrv servlet prg should redirect the request to search engine website and should also send the endure supplied from data (search String) to search engg website as additional req param value by appending query String to the url of res.sendRedirect() method.

SearchApp (Deploy in Tomcat 6.0)



⇒ Search URL in 3 different search Engine—

<http://www.google.co.in/search?q=hello>

<http://www.bing.com/search?q=hello>

<http://www.yahoo.com/search?p=hello>

For above diagram based example Appn refer flipb 17 of page no 130

2132 ~~in~~ search

2135, 2136, 2137

2133

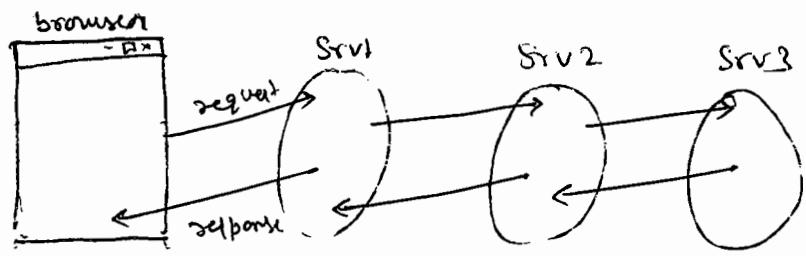
2150

2171, 2172

2179, 2183, 2190, 2187, 2194

WEB Component Communication or Request Dispatching or Servlet-to-Servlet Communication

- ⇒ Here the source servlet prog directly communicate with destination program (like another servlet prog or jsp prog or html program)
- ⇒ In Redir. Send Redirection the source prog talks with Dest prog by having one network round trip with browser window.
- ⇒ In web component communication all the web resource Progs will use same request, response obj to process the request.



(All these 3 servlet progs are processing same request, so they use same request, response objects)

Use above webcomp communication when source and dest Progs are local to each other.

Use sendRedirection when source & dest progs are remote to each other.

To ways of web comp Communication

Forwarding Request

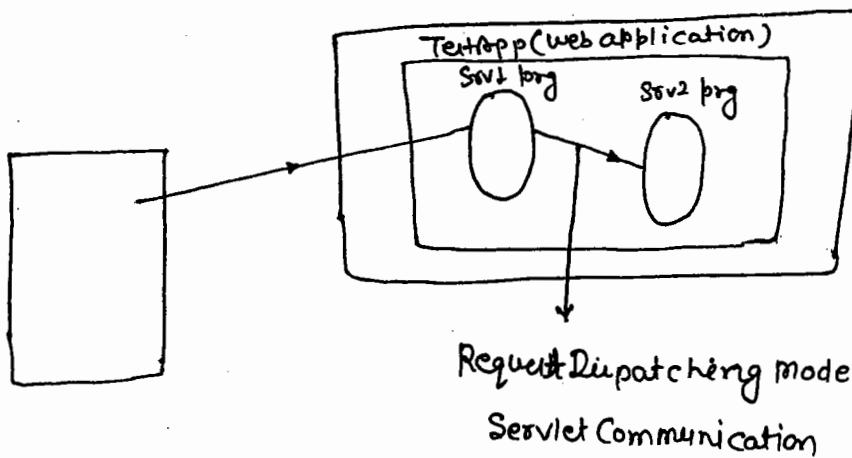
Here Source Program forwards the request & output of the source program will be discarded. Only output of Dest program goes to browser as response.

Including response

Here Source prg included the output of Dest prg so the outputs of both source & dest prgs together go to browser as response.

⇒ For webcomponent communication you should have Request Dispatcher in source prg to communicate with dest prg. It is the object of underlying servlet container supplied java class that implements javax.servlet.RequestDispatcher(I)

37
10/04/15



⇒ To perform web comp. communication or Request Dispatching mode communication we need RequestDispatcher obj

Different ways of creating RequestDispatcher obj

Approach 1: Using request obj

In Source servlet prog (Srv1 prog)

RequestDispatcher rd = req.getRequestDispatcher("①s2url");
optional

→ URL pattern of Dest Servlet prog

rd.forward (req, res) → performs forwarding request mode of communication
(or)

rd.include (req, res) → Perform including response mode of communication

Approach 2:- Using ServletContext obj

In Source prog (Srv2 prog)

ServletContext sc = getServletContext();

RequestDispatcher rd = sc.getRequestDispatcher("①s2url");
mandatory

rd.forward (req, res);

or

rd.include (req, res);

Approach 3: Using ServletContext obj

In Srv1 prog.

ServletContext sc = getServletContext();

RequestDispatcher rd = sc.getRequestDispatcher("abc");

→ Logical name given web.xml file

for Dest prog (Srv2)

rd.forward (req, res);

Q) What is the difference b/w getRequestDispatcher() & getNamedDispatcher() methods?

getRD()	getND()
i) Allows url pattern of servlet prg , url pattern or filename of jsp prg or filename of html prg as argument value.	Allows the logical name of servlet prg / jsp prg as argument value
ii) Invokable only request, ServletContext obj.	Invokable only ServletContext obj
iii) Allows to take html program as destination program.	Does not allow to take html prg as dest prg becoz we can't cfg html program in web.xml file having logical name.

a) RequestDispatcher rd = req.getRequestDispatcher("/abc.html");

b) RequestDispatcher rd = req.getRequestDispatcher("/abc.jsp");

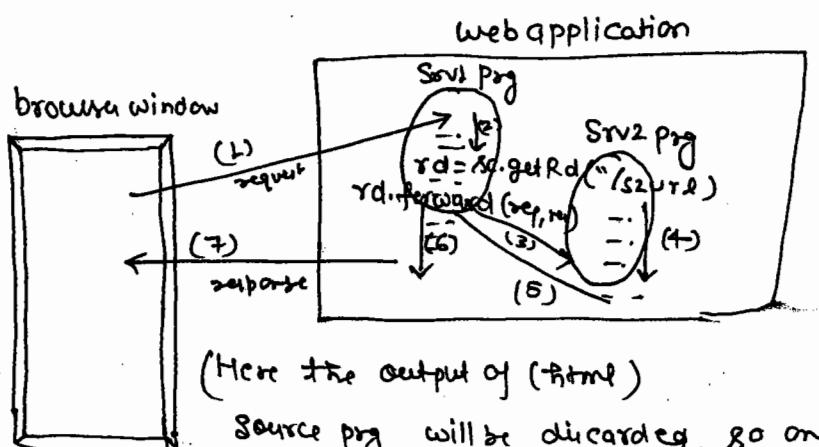
Q) What is the difference between creating RequestDispatcher obj using request obj and creating using ServletContext obj?

A)

⇒ request obj based RequestDispatcher objects allow us to keep both source & dest prgs in same web application.

→ ServletContext Obj based RequestDispatcher obj allows us to keep both source & dest prog either in same web appn or in two different web applications of same server, But not in two diff web application

Understanding rd.forward (-,-)



(Here the output of (html))
source prg will be discarded, so only
the output of Dest prg (Srv2) goes
to browser as response)

- rd.forward (-,-) performs forwarding request mode of communication
- Here Source Servlet program directly communicate with dest program.
- Here Source & Dest prgs we have request, response objs so, the req data coming to source prg is visible & accessible in dest prg.
- To pass additional Data from Source prg to dest prg we req attribute Srv1, Srv2 prgs can be there in same web application or can be there in two different web appn or can be there in two different web appn of same server.
- All statements placed Srv1 prg before & after rd.forward

`rd.forward(, -)` method executes but their html output will be discarded.

→ Srv² prg can be a servlet prg or jsp prg or html prg.

→ Real time use-case: Developing Error Servlet & cfg that Server

Q) What is Error Servlet?

A) ⇒ The servlet prg that execute only when Exception is raised in other servlet prg is called Error Servlet.

⇒ This Servlet is useful to display Exception related technical message or non-technical guiding message.

⇒ * While performing Credit / Debit purchase if any exception

→ To link error servlet with main servlet we place `rd.forward()` that points to error servlet in the main servlet program.

~~22/04/25~~

38

Sample Code on Error Servlet cfg

Step1) Keep DBApp application ready

Step2) Develop Error servlet

ErrSrv.java (In WEB-INF\classes folder of DBApp)

```
package com.nt;  
public class ErrSrv extends HttpServlet {  
    public void doGet(-,-) throws ServletException, IOException {  
        //general Settings  
        PrintWriter pw = res.getWriter();  
        res.setContentType("text/html");  
        pw.println("<b><font color= red > Internal Problem  
                    Try Again </font> </b>");  
        //close Stream.  
        pw.close();  
    }  
}
```

Step3) cfg ErrSrv pag in web.xml having "/error" as url pattern

Step4) cfg Error Servlet main Servlet (DBSrv) by using res.forward(-,-).

//DBSrv.java

```
public class DBSrv extends HttpServlet {  
    public void init() {  
        --  
        --  
    }  
}
```

```
public void doGet(–, –) throws SE, IOException {  
    try {  
        //  
        //  
        //  
        // Code that may throw Exception  
        //  
        //  
        //  
    }  
    catch (Exception e) {  
        RequestDispatcher rd = req.getRequestDispatcher()  
        rd.forward(req, res);  
    }  
}  
// doGet(–, –)
```

```
public void doPort( , ) throws SE, IOE {  
    doSet( seq, ser );  
}  
  
public void destroy( ) {  
    -- -- -- -- --
```

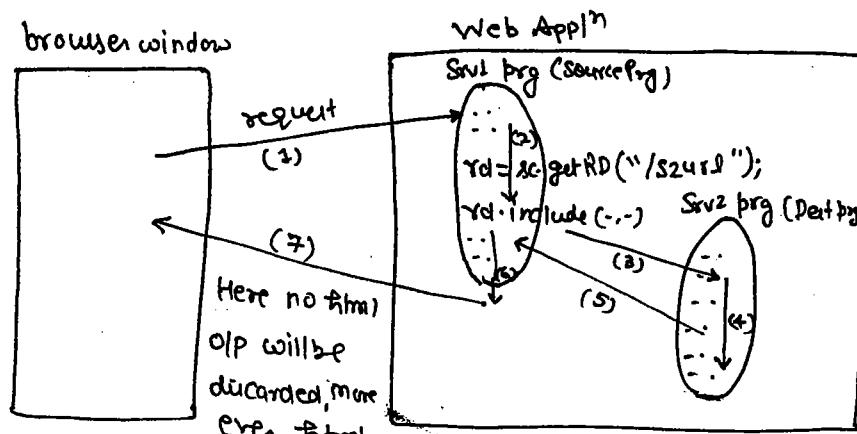
→ When `rd.forward()` is executed in source servlet program the `<HTML>` output generated by that program will be discarded. But The output generated by `Script` statements will not be discarded.

Q) what happens if we place multiple rd.forward() in one source servlet pgm;

A) It throws exception (IllegalStateException) most of the time.

In some special situations if all rd.forward() are pointing to servlet programs then Ith rd.forward() will be applied.

Understanding rd.include(-,-) :-



Output of both Source
prg (Srv1), dest prg (Srv2)
together goes to browser window
as response.

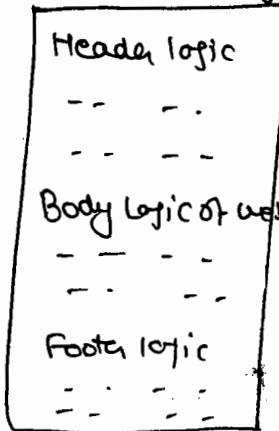
Key Points:-

- ⇒ `rd.include(-,-)` performs including response mode of communication.
- ⇒ Srv1, Srv2 prgs use same req, res obj. So the request data coming to Srv1 prg is visible and accessible in Srv2 prg.
- ⇒ To pass additional data from Srv1 prg to Srv2 prg we can use `req` attribute.
- ⇒ Srv1 communicates with Srv2 prg directly
- ⇒ Srv1, Srv2 prgs can be there in same web app or can be there in two diff web apps of same server.
- ⇒ Srv2 can be servlet prg or Jsp prg or html tag prg.
- ⇒ All statements of Srv1 prg before & after `rd.include(-,-)` will be executed but their fml output will not be discarded.
- ⇒ Srv1 prg output will be included in the o/p of Srv1 prg in the place where `rd.include(-,-)` is called.

`<%@include(-,-)` is useful to place common logic (header, footer) in separate web page & include their o/p in main web resource page.

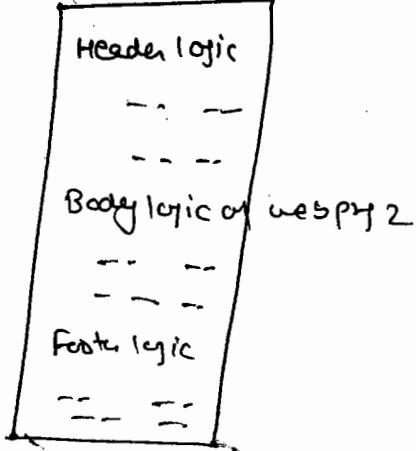
Problem

Main Servlet Prg 1



(generate webpage)

Main Servlet Prg 2



(generate webpage)

→ If multiple pages of web appn contains same header & footer content. In ^{the above} code header, footer logic are not reusable.

Solution:-

HeaderSrv (servlet proj)

== header logic

Main Servlet Prg 1

Header logic

`<%@include(-,-)`

Body of logic of webpage

== ==

== ==

Footer logic

`<%@include(-,-)`

Here Header, footer logic
are reusable logic.

footer.html

== footer logic

Main Servlet Prg 2

Header logic

`<%@include(-,-)`

Body logic of webpage 2

== ==

== ==

Footer logic

`<%@include(-,-)`

(33)
~~12/04/15~~

Sample Code that uses rd.include()

Step1 Keep DBAPP Ready

Step2 Develop separate web resource program having header, footer logic.

HeaderSrv.java

```
public class HeaderSrv extends HttpServlet {
    public void doGet(-,-) throws SE, IOE {
        PrintWriter pw = res.getWriter();
        res.setContentType("Text/html");
        pw.println("<marquee><%1> Netraaz.in <%2>
                    </marquee>");
        // I don't close pw
    }
    public void doPost(-,-) throws SE, IOE {
        doGet(req, res);
    }
}
```

Note:- Configure HeaderSrv prg in web.xml file with /headerurl url pattern.

Footer.htm1

```
<b><i> <center> copy right reserved 2014-15 </center> </i> </b>
```

DBSrv.java (main Servlet Prg)

```
public class DBSrv extends HttpServlet {
    public void init() {
        -- -- --
    }
    public void doGet(-, -) throws SE, IOE {
        try {
            //include header content
            RequestDispatcher rd1 = req.getRequestDispatcher("/header");
            rd1.include(req, res);
            -- -- -- //code that generate
            //include footer content
            RequestDispatcher rd2 = req.getRequestDispatcher("/footer.html");
            rd2.include(req, res);
        } catch(Exception e) {
            RequestDispatcher rd = req.getRequestDispatcher("/error");
            rd.forward(req, res);
        }
    }
    public void destroy() {
        --
    }
}
```

} //class

For complete example qpp on web component Appln that uses
rd.forward (-, -) rd.include (-, -) refer pages: 127 to 132

Note `pw.close()` commit the response that means we can't add further content to the response.

- ⇒ While working with `rd.include(-,-)` we should not place `pw.close()` in destination program because it commit the response and does not allow to add further content to the response given by source program.
- ⇒ If we place both `rd.forward(-,-)` and `rd.include(-,-)` in one source servlet program then effect of `rd.include()` will not be there, because `rd.forward(-,-)` method not only discard the o/p of source servlet prg. It also discard the include o/p of source servlet prg.
- ⇒ If we place multiple `rd.include(-,-)` methods in source servlet program then o/p of multiple destination program will be included in the source program.
- ⇒ If you place `pw.close()` method before `rd.forward(-,-)` or before `rd.include(-,-)` method calls then web component communication will fail. Moreover we may get illegal state exception.

A) What is the difference b/w `rd.forward(-,-)` & `rd.include(-,-)`

A)

S.No.	<code>rd.forward(-,-)</code>	<code>rd.include(-,-)</code>
i)	Performs forwarding request mode of communication.	Performs including response mode of communication.
ii)	Discard the o/p of source program and final response contains only the o/p of dest prg.	Does not discard any o/p. So the final response contains the o/p of both source and dest prg.
iii)	Allows to place <code>pw.close()</code> in dest prg.	Does not allow to place <code>pw.close()</code> in dest prg.
iv)	Useful for ErrorServlet/page ctrl	Useful to make common logic as the reusable logic by placing them in separate web resource prg.

- ⇒ if you place multiple `re.sendRedirect(-)` method in one source program then there is a possibility of getting exception.
- ⇒ if we place `re.sendRedirect(-)` and `rd.forward(-,-)` in one source servlet prg then we may get illegalStateException
- ⇒ if we place `rd.include(-,-)` and ~~`rd.include(-,-)`~~ with `re.sendRedirect(-,-)` in one servlet prg then the effect of `rd.include(-,-)` will not be there.

Q) What is the difference b/w `rd.forward()` & `rd.sendRedirect()`

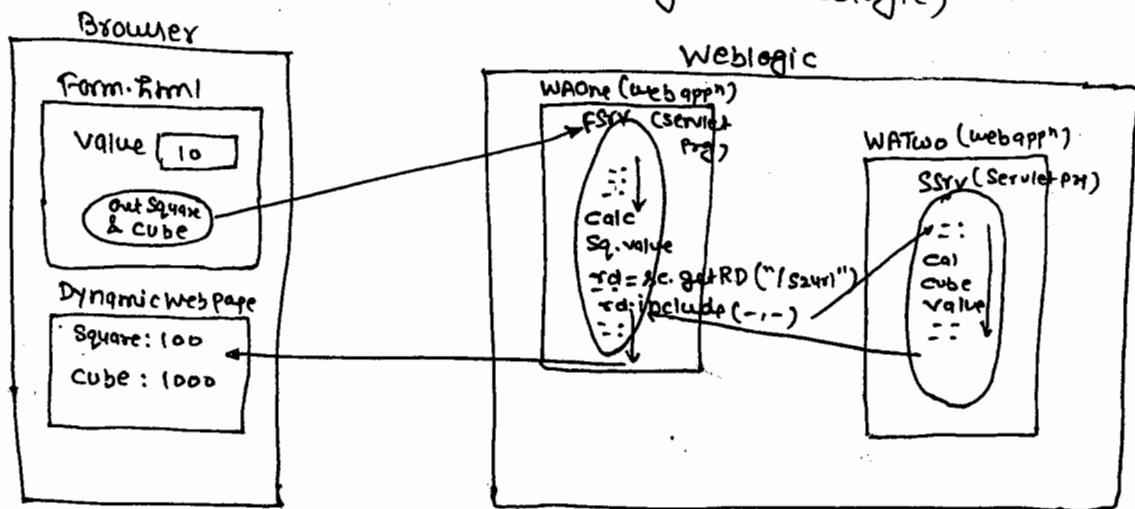
A)

S.No.	<code>rd.forward</code>	<code>re.sendRedirect</code>
1)	Performs forward request mode communication.	Performs redirection mode communication.
2)	Source talk with destination directly.	Source talk with destination by having one round trip with browser.
3)	Source and destination use same req, res obj.	Does not use same req, res Obj.
4)	req. data coming to source is visible and accessible in dest prg.	Is not visible and accessible in dest prg.
5)	To pass additional data from source to dest we use req attribute.	Append query to the url of <code>re.sendRedirect(-)</code>
6)	Source and dest can be there in same web app or in two diff. web app of same server.	Source and dest can also in two diff. web app of two diff. server.
7)	Dest prg can be html, servlet or jsp program.	Dest prg can be html, .servlet, jsp prg, asp.net, php prg and etc.
8)	While forwarding request url in browser address will not change.	will be changed.
9)	Useful when source & dest are local.	Useful when source and dest are remote.

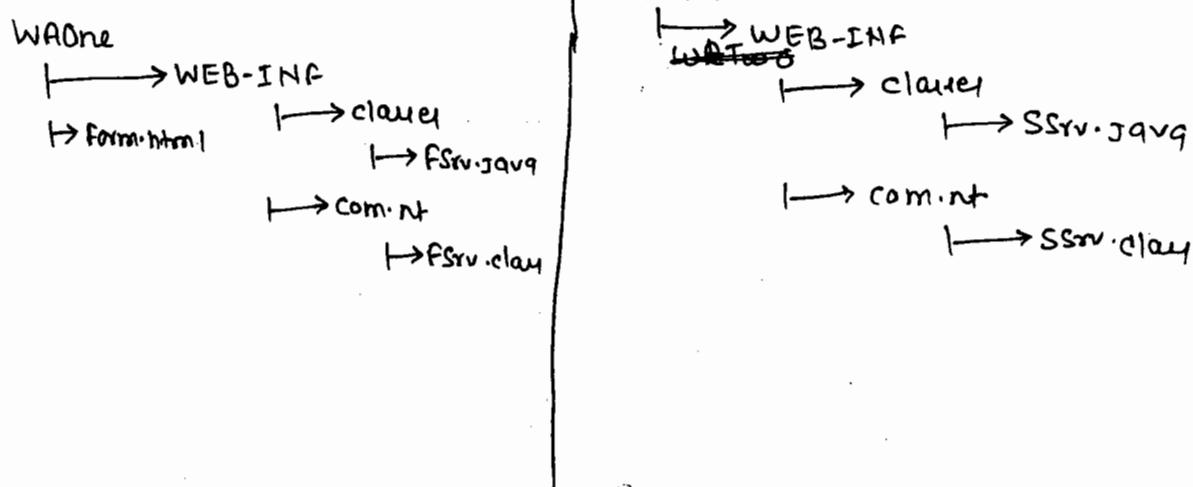
(40)

13/04/15

- req object based RequestDispatcher allows us to place both source & dest program in one appn whereas the ~~getter~~ ServletContext obj based RequestDispatcher obj allows us to place source and dest program either in same web appn or in two different web application of same server.
- Example Application on getting communication b/w two different servlet program of two different web application using RequestDispatcher object (only in weblogic)



Here src fsrv program & Dest ssrv prog are there in two diff web Appn of same server, so we need to use ServletContext Obj RequestDispatcher obj



⇒ Whenever we call `rd.include()` in source servlet prg it locates or creates Dest servlet class obj and performs all lifecycle operation to communicate with dest servlet prg.

Note:- The above appn doesn't work in tomcat, Glassfish, Jboss Server because the method `getContext()` of `HttpServletRequest` is not implemented properly. Due to this NullPointerException will be raised

Conclusion on Servlet Communication:-

- if src & dest prg are there in same web Appn then use RequestDispatcher based communication.
- if src & dest prg are there in two different web appn of same or different servers then use Send Redirection based communication.

Attributes in Servlet Programming:-

Q) How to pass data b/w web resources prgs as the web Appn?

A) if src & dest prg reside in same web Appn -

- use `req` attributes (if src & dest prgs use same req, res obj)
- use `session` attribute (when src & dest prgs get req from same browser)
- use `ServletContext` attribute (for any situation)

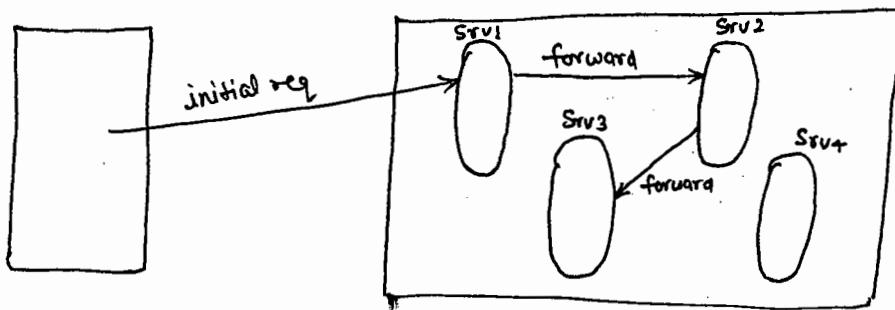
if src & dest prg reside in two diff web appn of same or diff servers

Append query string to url of `res.sendRedirect()` method.
`res.sendRedirect(url?<query string>)`

Attribute is a logical name that holds object as value having scope -

request Attributes ---> Scope: request scope (visible throughout request)
session Attributes ---> Scope: session scope (specific to a browser)
ServletContext Attributes ---> Scope: Application scope (visible in web resource prg of web appn)

request Attributes:-



- ⇒ req attributes scope that means request attributes are specific to specific request or request attributes are visible throughout request.
- ⇒ Request attributes are visible in all web resource prg when use same request response object.
- ⇒ In above diagram,

Srv1, Srv2, Srv3 are processing same request i.e. they use same req, res obj. So the request attribute created in Srv1 prg is visible and accessible in Srv2, Srv3 prg but not in Srv4 program.

- ⇒ request attributes allocate memory in request obj, so they can be used in web resource prgs

4
14/04/15

To create request attribute:-

```
req.setAttribute ("name", "Raja");
```

```
req.setAttribute ("Age", 10);
```

↓
Convert to Integer Wrapper class obj
through Autoboxing

To Read request Attribute value:-

```
String name = (String) req.getAttribute ("name");
```

```
int age = (Integer) req.getAttribute ("age");
```

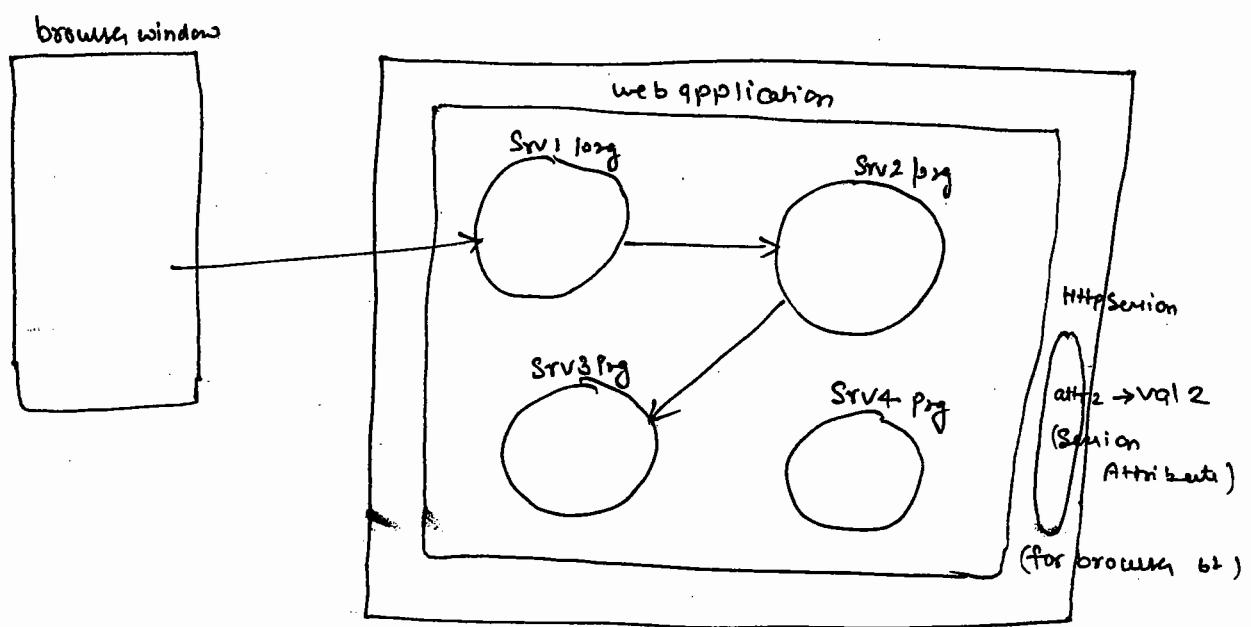
↓

Here the wrapper class of object

Integer is converted to simple
integer using Autounboxing feature.

; Session Attribute:-

- ⇒ HttpSession object allocate memory on server on a per browser window basis.
- ⇒ Session Attribute resides in HttpSession object, These attributes are visible in all web resource program of web appn irrespective of req, res obj they use. But they must get request from that browser window for which HttpSession Obj is created.
- ⇒ Session attribute scope is session scope that means they visible in web appn but specific to a browser window.



⇒ The session attribute created in Srv1 prog by getting req from browser window b1 is visible and accessible in another web resource prog of web appn only they get request from same browser window (b1).

To create session attribute:—

HttpSession ses = req.getSession(); ⇒ create or locate HttpSession obj
 ses.setAttribute ("course", "java");

To read session attribute:—

String course = (String) ses.getAttribute ("course");

To delete session attribute:—

ses.removeAttribute ("course");

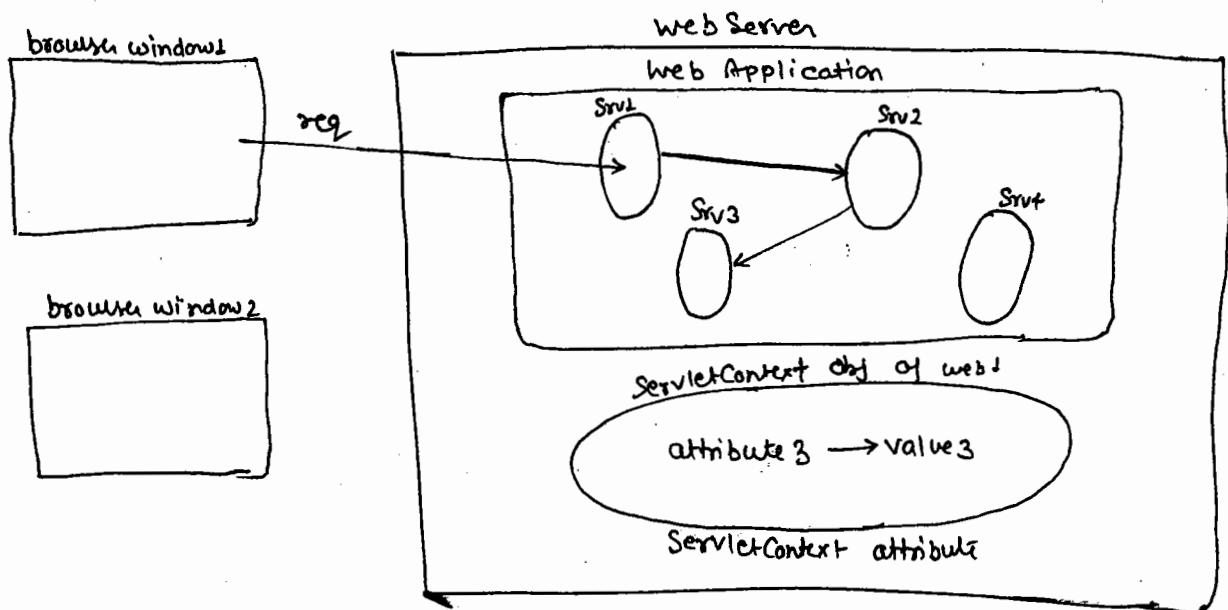
Servlet Context Attribute / Application attribute—

⇒ These attributes allocate memory in ServletContext obj which is global memory of the web application.

⇒ These attributes are visible and accessible in all the web resource program of the web appn irrespective of the req, res.

Obj they are using and irrespective of the browser window from which they are getting request.

⇒ ServletContext Attribute provide global visibility for attribute within a web application



The ServletContext attribute that is created in Srv1 by getting request from any browser window is visible and accessible in all web resource obj of web App in irrespective of any condition.

To Create ServletContext attribute value:-

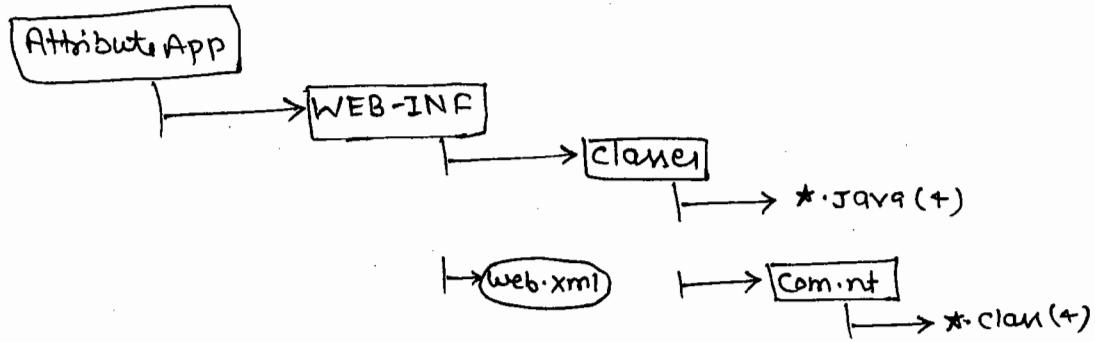
```
ServletContext sc = getServletContext();
sc.setAttribute ("email", "notaraz@gmail.com");
```

To read ServletContext Attribute value:-

```
String mail = (String) sc.getAttribute ("email");
```

To remove ServletContext Attribute:-

```
sc.removeAttribute ("email");
```



→ Give first Request to Srvs and give remaining request to Other prg either from same window or from different windows And observe the visibility of attributes.

Page 135 → 1945 to 1953

Page 138 → 2216, 2219, 2222, 2255, 2256, 2251 line

↳ For Example App that demonstrate different scope of attribute refer App 18 page 139.

(4-2)
15/04/15

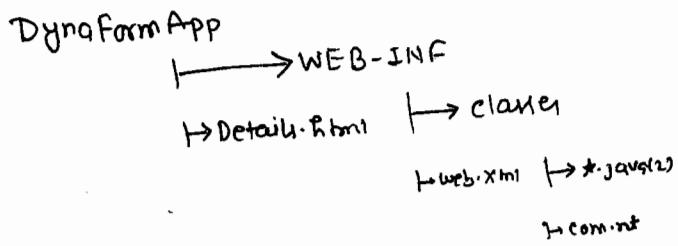
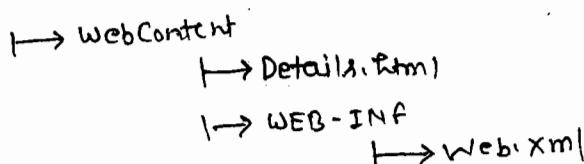
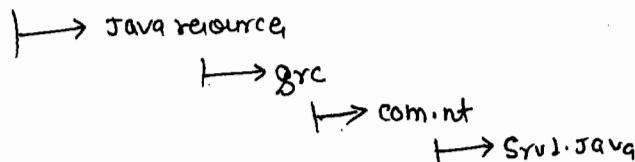
Working with Dynamic Web Pages:-

~~X X X~~

Page no: 183

Baidu diagram

DynformApp (for Eclipse)



Correction: 182 → ~~checkbox~~ checkbox married

<url pattern = "/Srv1/Prog" >

→ Srv1.java

if (maritalStatus = "single")

/Srv2/Prog → url pattern of Srv2

name: s1+2
else

url pattern of Srv2 Prog
s1+2 s1+2

→ Page no: 186

4. pw.println → ready to format data but gives null value
⇒ gives form2 / req2 data.

Page 183

Diagram demonstrating dyno. dynamic form page generation along with stateless behaviour of the web appn.

Page no: 186

In Stateless webApp we can't use previous req data while processing correct that means while processing req3 we can't use req1, req2 data.

In Stateful webApp every current request can use previous request data that means we can use req1, req2 data while processing req3.

(43)
Hence it is

- Session tracking:-

i) Hidden form field

refer booklet

(44)
~~17/04/15~~

ii) Http Cookies:-

→ refer booklet

(45)
~~18/04/15~~

Http Cookies (Continued....)

→ refer booklet

(46)
~~19/04/15~~

Diaadvantage of cookie - & Advantage of cookie -

refer booklet

HttpSession with cookies -

→ refer booklet

Session API :-

(47)
~~20/04/15~~

To know Last Accessed time of Session -

```
long m = sri.getLastAccessedTime();  
Date d = new Date(m);
```

To invalidate the Session -

- a) Close browser window
- b) When sri.invalidate() method is called
- c) When max inactive interval period / session idle timeout period is complete.



(43)
22/04/15

4) HttpSession With URL Rewriting :-

→ refer booklet page no 200

URL Rewriting (Eclipse Project)

Java Resource

src

com

firstServlet.java

SecondServlet.java

ThirdServlet.java

Webcontent

personal.html

WEB-INF

web.xml

JAR file: ojdbc14.jar, Servlet-api.jar

Request URL: http://localhost:3030/URLRewriting/Personal.html

(Run this app by disabling / blocking cookies in browser window)

res.sendRedirect(res.encodeRedirectURL("/surf"));

This method is useful to continue the session started with one webapp in continue in another webapp.

Eg The session started in gmail will be continued in YouTube. To get user identity.

Generic Servlet	HttpServlet
i) Protocol independent	Protocol (http) dependent
ii) Does not allow to use all the features of http	Allows
iii) Common super class for multiple protocol specific servlet prog.	Sub class of Generic Servlet for using features of protocol http.
iv) Supports only one way session tracking that in hidden box etc.	Supports all way of session tracking.
v) It is Abstract class having one abstract method.	It is abstract class having no abstract methods.

vi)	Given service (-) method for placing request processing logic.	Given doxxx(-) (7) for placing request processing logics...
-----	--	---

— ; Servlet Listners; —

Servlet Listners (Event handling in servlet prog. programming);—

Event is an action performed on the object or comp. Every event is an object internally.

Event handling means executing some logics when event is raised. For this we have Event Listners that give Event Handling methods.

To perform event handling we need 5 details-

- Source comp: like Button
- Event class: like ActionEvent
- Event Listener: like ActionListener
- Event handling method: like actionPerformed();

Perform

From servlet api 2.3 event listeners are given to perform event handling on request, HttpSession, ServletContext obj.

⇒ Every handling req obj

- Allows to know when request obj is created / destroyed.
- Allows to know when request attribute is created / modified / deleted.

Note:- Using this we can get request processing time of each request to evaluate the performance of each request.

⇒ Event handling on ServletContext obj

a) Allows to know when ServletContext Obj is created / destroyed.

b) Allows to know when ServletContext attribute is created / modified / deleted.

Note:- Using this we can keep track web appn deployment or Undeployment related activities.

⇒ Event handling on HttpSession obj

a) Allows to know when HttpSession obj created / destroyed.

b) Allows to know when session attribute is created (modified) / deleted.

Note:- Using this we can keep track of session duration of each user.

(50)
23/04/15

⇒ Keeping track of flow of execution is called logging. Talking about components that are involved.

⇒ Keep tract of activities performed in the appn is called Auditing like noticing web application deployment time, Undeployment time and etc....

⇒ Servlet Listener are given to perform ^{auditing web} in the appn
(Monitor various activities in the appn)

Servlet Obj	Event class	Event Listener	Event Handling
request obj	ServletRequestEvent	ServletRequestListener	requestInitialized(-) requestDestroyed(-)
HttpSession obj	HttpSessionEvent	HttpSessionListener	sessionCreated (-) sessionDestroyed (-)
ServletContext	ServletContextEvent	ServletContextListener	ContextCreated (-) ContextDestroyed (-)

⇒ We can use ServletListeners to enable monitoring / auditing on web app without disturbing the source code of Web app.

⇒ In one web app we can have multiple Servlet Listeners, All these listeners must be cfg in web.xml, file by using <listener>, <listener-class>

Example App

Note:- By using sc.log(-) method we can write log message to current days log file <Tomcat_home>\logs\localhost.<Date>.txt file.

Step1) Keep "SessionApp" ready (App24 of the booklet)

Step2) Add following Listener classes in com.nt.listener class in com.nt.listeners pkg of Eclipse "src" folder

MyReqListener.java

```
package com.nt.listeners;  
public class MyReqListener implements ServletRequestListener {  
    long start, end;  
  
    public void requestInitialized(ServletRequestEvent sre) {  
        start = System.currentTimeMillis();  
    }  
  
    public void requestDestroyed(ServletRequestEvent sre) {  
        end = System.currentTimeMillis();  
        ServletContext sc = sre.getServletContext();  
        // write log file  
        sc.log(((HttpServletRequest) sre.getServletRequest()).  
            getRequestedURI() +  
            " hat folgen " + (end - start) + " ms time");  
    }  
}
```

MySuliListener.java

```
package com.nt.listeners;  
public class MySuliListener implements HttpSessionListener {  
    long start, end;  
  
    public void sessionCreated(HttpSessionEvent hse) {  
        start = System.currentTimeMillis();  
    }
```

```
    fse.getSession().getServletContext().log ("Session started at"
                                              + new Date());
}

public void sessionDestroyed (HttpSessionEvent fse) {
    end = System.currentTimeMillis();
    fse.getSession().getServletContext().log ("Session ended at"
                                              + new Date ());
    + "Session Duration" + (end - start) + "ms.");
}

}
```

MyServletContextListener.java

Package com.nt.Listeners;

```
public class MyServletContextListener implements ServletContextListener {
    long start, end;

    public void contextInitialized (ServletContextEvent sce) {
        start = System.currentTimeMillis();
        sce.getServletContext().log ("Web appn is
                                     deployed / restarted at " + new Date ());

    }

    public void contextDestroyed (ServletContextEvent sce) {
        end = System.currentTimeMillis();
        sce.getServletContext().log ("Web appn is
                                     undeployed / stopped " + new Date () + "
```

"it is in running mode for " + (end - start) + " ms");

}

Configure above Listener in web.xml file.

```
<listener>
  <listener-class> com.nt.listeners.MyScListener
  </listener-class>
<listener>
  <listener-class> com.nt.listeners.MySelListener </listener-
  class>
<listener>
  <listener-class> com.nt.listeners.MyReqListener </listener-
  class>
</listener>
```

Run the app & observe today's log file in the Tomcat sever.

(51)

24/04/15

VIMP:

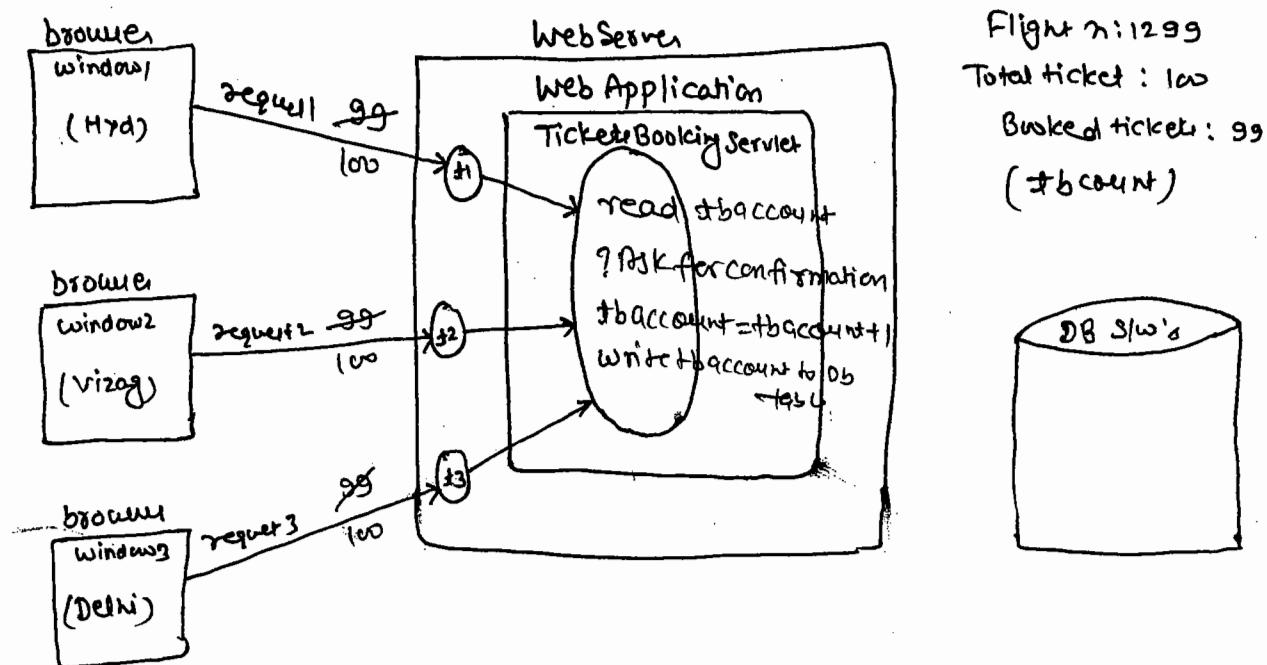
Thread Safety in servlet programming:-

⇒ If multiple threads are acting on single obj / variable simultaneously then that obj is not thread safe by default. (Data corruption can occur).

⇒ instance variable of java class / servlet class are not thread safe by default. The local variable of service(-) / doXXX(-) are thread safe by default.

→ if we don't make servlet prog or its data or thread safe we may get lot of practical issues like booking same ticket for multiple passengers at same time.

Shown below —



⇒ To make our servlet prog or thread safe we can use

- Place only local variables in `service(-,-)` / `doXXX(-,-)` methods.
- Use synchronized `service(-,-)` / `doXXX(-,-)` methods.
- Place synchronized blocks inside the `service(-,-)` / `doXXX(-,-)` methods.
- Make our servlet prog class implementing `javax.servlet.SingleThreadModel` (I).

Approach 1) Local Variables:-

```
public class TestSrv extends HS {
    Connection con;
```

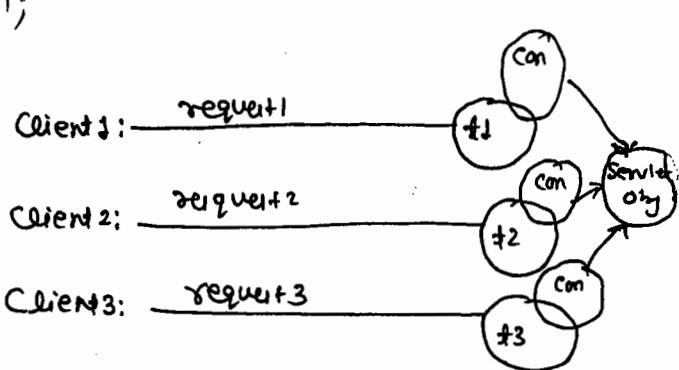
```
    public void service(-,-) throws SE, IOE {
```

```
        Connection con;
```

```
}
```

Limitation:-

We can't take only local variable every time.



Here every thread is getting its own copy of con object. So con obj (local) is thread safe.

Approach 2) Using Synchronized service(-,-) / doXXX(-,-) methods:-

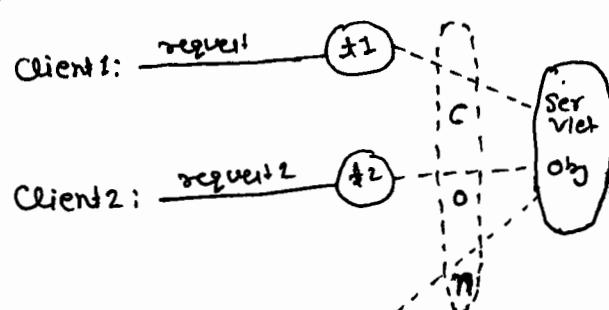
```
public class TestSrv extends HS {
```

```
    Connection con;
```

```
    public void synchronized service(-,-) / doXXX(-,-)
        throws SE, IOE {
```

```
    }
```

use con



Here multiple threads are acting on single

copy of connection obj (instance variable) but only one thread will act on that con obj at

a time during service(-,-) or doXXX(-,-) method execution bcz that method is given in synchronized method.

Limitation:-

Making whole service or doXXX(-,-) method as synchronized method for share one or two obj at meaning level.

Approach 3: Using Synchronized blocks:-

public class TestServ extends HttpServlet {

 Connection con;

 public void service(-,-)/doXXX(-,-) throws SE, IOE {

 Synchronized (con) {

 -- // con obj

 }

 ServletContext sc = req.getServletContext();

 Synchronized (sc) {

 -- // sc obj

 }

 HttpSession ses = req.getSession();

 Synchronized (ses) {

 -- // ses obj

 --

 }

}

This approach gives good performance without having performance issues. So it is industry standard.

Here only the object that is placed in synchronized block gets lock and allows one thread at a time.

HttpSession, ServletContext obj are global obj in web app so it is recommended to use them by keeping in synchronized blocks.

Approach 4: Using SingleThreadModel(I) :-

Public class TestSrv extends HttpServlet implements SingleThreadModel {

Connection con;

public void service(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {

--- --- --- /use con

}

}

When this interface is implemented on our servlet class we are guaranteed that no two threads will execute simultaneously on one obj of our servlet class. For this Servlet Container either uses synchronization concept to lock our servlet class object or uses Servlet class obj pool to assign one obj to each request.

(This is violation of servlet specification that says Servlet should be single instance multiple thread appn.)

Because of above reason and no proper standardization and implementation this interface deprecated with no replacement from Servlet API 2.4.

27/04/15

Servlet Filters:-

- ⇒ Intercepting filter is special web resource prg of web app' that can trap all the request and response of other web resource prg.
- ⇒ Intercepting filter contains command and global pre request logic and post response generation logics.
- ⇒ In Servlet programming we can use servlet filter as "Inter ceptor" filter of Web app.
- ⇒ Without disturbing the existing web resource of web app if you want to add additional functionality to web app then use Servlet filter support.

Basic Point:-

- ⇒ The java class that implements javax.servlet.Filter(I) is called Servletfilter. This class should also implement 3 methods of filter. They are -
 - i) init(FilterConfig cg)
 - ii) doFilter(ServletRequest sreq, ServletResponse srep, FilterChain fc) throws SE, IOE
for pre-request processing, post-response generation logic
 - iii) destroy()
for uninitialization logic.
- ⇒ The above three methods are lifecycle method of servlet filter.
- ⇒ Servlet Container creates our servlet filter class obj either during servlet startup or during deployment of web App (No <load-on-startup> is required)
- ⇒ FilterConfig obj is one per our Servletfilter class obj. It can be used to read filter init parameters placed in web.xml file. Useful to access ServletContext obj.
- ⇒ Servlet container destroys our servlet class obj when web app is stopped or reloaded or undeployed
- ⇒ Every servlet filter prg must be cfg in web.xml file using <filter> <filter-mapping> tags.
- ⇒ We can map one or more filter prgs with one or more main web resource prg web app.

Example:-

To link one filter with one servlet program

• servlet url pattern : /s1url

filter url pattern : / s1url

To link more filter with one servlet program

(filters) frv1, frv2 url pattern: /s1url

Srv1 url pattern : /s1url

Note:- When multiple filters are mapped with single servlet program then the filters trap the request in the order they cfg. in web.xml Similarly trap response in the reverse order of their cfg.

To one filter with all web resource prg of web App

Frv1 url pattern : /*

Srv url pattern :

To link specific filter prg with specific servlet prg:-

Srv1 Prg: / abc/ s1url

Srv2 Prg: /abc/ s2url

Srv3 Prg: /xyz/ s3url

Srv4 Prg: /xyz/ s4url

Frv1 Prg: /abc/ *

Frv2 Prg: /xyz/ *

There are 3 types of filters—

a) request filter

Contains only pre-request processing logic.

e.g. Req count filter, Authentication filter, Authorization filter and etc.

b) response filter

Contains only post-response generation logic

e.g. Conversion filter, compression filter and etc.

c) request-response filter

Contains both pre request and post response generation logic.

e.g. Performance Test filter.

⇒ Filter prg and its target servlet prg/jsp prg use same request, response objs, so the filter prg can send data to servlet prg/jsp prg by using request attributes.

Sample Servlet Filter Program:-

```
public class Myfilter implements javax.servlet.Filter {  
    public void init(FilterConfig fg) {  
        ... ... ... //Initialization logic  
    }  
    public void dofilter (ServletRequest req, ServletResponse res,  
                        FilterChain fc) throws SE, IOE {  
        ... ... ... Pre-request logic  
        fc.dofilter (req, res); // Control goes to next filter/  
        ... ... ... Target Servlet/JSP prg  
    }  
}
```

```

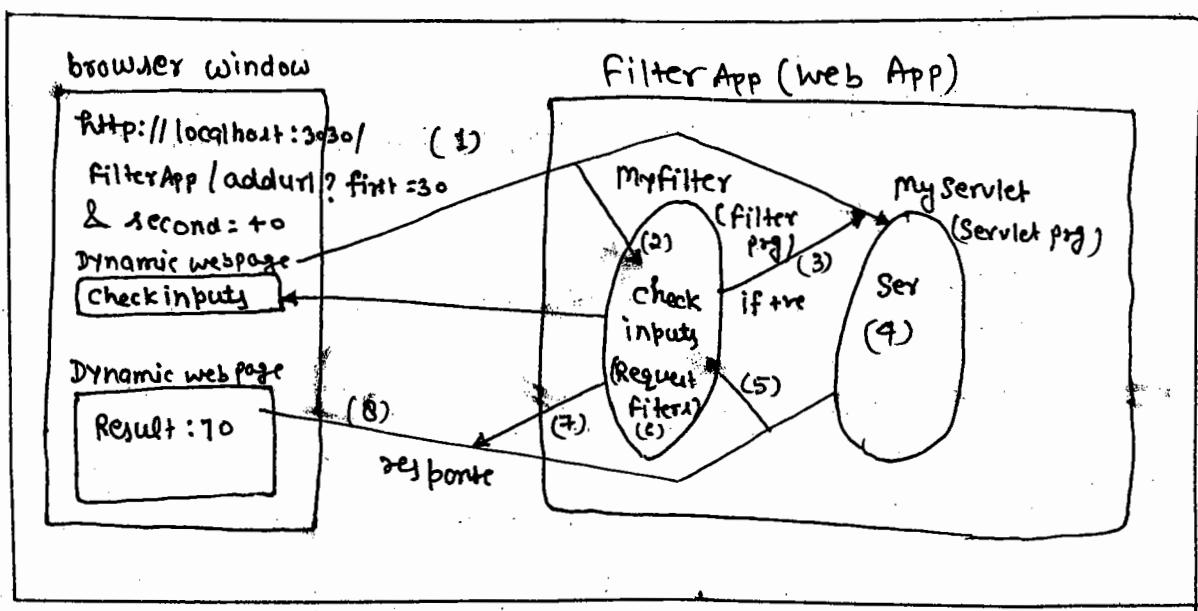
public void destroy()
{
    ...
    ...
    ...
}

}

} // class

```

Example App - POC



→ Let's assume myServlet having logic of addition without validating inputs. To perform validation without disturbing the code of existing web resource program (servlet) we can take the support of shown above.

→

Q) How can we find No. of request (Hits) coming to Web App?

A) It can be done by developing Servlet filter and by making that servlet filter. All the Request and response by having special url pattern.

Example App

Step (1) Keep Session App Ready (Appn 24 from Booklet)

Step (2) Add Special filter in com.nt. pkg of web app (Session App). Request Count Filter.

```

public class ReqCountFilter implements Filter {
    int cnt = 0;
    ServletContext sc;
    public void init(FilterConfig fg) {
        sc = fg.getServletContext();
    }
    public void doFilter(ServletRequest req, ServletResponse res,
                         FilterChain fc) throws
        IOException, ServletException {
        cnt++;
        sc.getAttribute("counter", cnt);
        fc.doFilter(req, res);
    }
    public void destroy() {
    }
}

```

Step3

cfg above Servlet filter prg with /* url pattern

in web.xml

<filter>

<filter-name> Counter </filter-name>

<filter-class> com.int. ReqCountFilter </filter-class>

</filter>

<filter-mapping>

<filter-name> Counter </filter-name>

<url-pattern> /* </url-pattern>

</filter-mapping>

Step4

place following code in every servlet program to
display req. count by reading servlet attribute value.

Note:-

We can also use "Servlet Request Listener" support to
count no. of request that are coming to
web app. (This is alternate to using Servlet filter)

— : Jboss (Server) : —

type: Application Server SW

version: compatible with jdk 1.7+

Vendor: Apache / Redhat

Creator: Marc Fleury

Commercial: (from Jboss 6.x), open source (upto Jboss 5.x)

default portno: 8080

To download: www. jboss.org

(or)

www. redhat. com

as zip file : Jboss - 01 - 7.1.1 - final .zip

⇒ To change the port no: of Jboss 7.x

< Jboss 7.x - home > \ Standalone \ cfg \ standalone - xml

modify

port attribute value of < socket - binding >

that points to http (ctrl + F = 8080)

⇒ To start Jboss server go to

< Jboss 7.x - home > \ bin folder and run Standalone . bat file

Note:- Jboss not having its own servlet, JSP container
it uses Tomcat server container internally

Procedure to deploy java web App in Jboss7

Step1> Create username and pwd that are required for
launching admin console

goto < Jboss - home > \ bin use "add - user . bat " file.

what type of user do you wish to add -

(a) Management (✓)

(b) Application

(c): a

Realm: ↲

user : tatuser

pwd : tatuser

repwd: tatuser

Is this correct Y/N: Y ↲

Step2:-

Start Jboss Server

user <Jboss-home>\bin\standalone.bat file
(run as administrator)

Step3.

Create WAR file representing web App

E:\VoterApp

↳ Input.htm , WEB-INF

↳ classes , web.xml

↳ VoterSrv.java

↳ com.nt

↳ VoterSrv.class

E:\VoterApp} jar cf VoterApp.war

Step4:-

open admin console of Jboss7.x.

http://localhost:8080 → admin console

user: tatuser

pwd : tatuser

Step5 Deploy the web App

admin console → managed deployment → add
Content → Browse & Select the war file
next → save → enable → confirm

Step6 Test the web App

http://localhost/6666/voterApp/input.htm

Note:- Tomcat support only warfile based console deployment that means it doesn't support an mode of hard deployment.

Procedure to access Tomcat Server web App with domain name in windows environment (No. of inter)

Step1. Go to

c:\windows\System32\drivers\etc\ folder
and edit the host file.

Having following change -

127.0.0.1 localhost to → 127.0.0.1 www.natqrqz.in

Open host file or in Notepad through administrator

Step2:-

Change tomcat http port no to go through
<Tomcat-home>\conf\server.xml

Step3

Add the following <host> in server.xml under
<engine> tag

<Host name="www.natqrqz.in" appBase="webapps"

unpackWAR="true" autoDeploy="true">

<context path="/" docBase="semisapp"/> welcome file cfg is mandatory in this app

Step4 Restart the server and Test the App.

Note:-

Perform all the activity Tomcat 6/7 service.

request url: www.nataraz.in

⇒ We can make the above <Host> tag collecting the sever app webApp from specified folder.

e.g.



In Server.xml

```
<Host name = "www.nataraz.in" appBase = "E:\Test">
```

...
...
...

```
</Host>
```

(54)
02/05/15

Adapter classes:-

problem:-

```
interface Test{
```

```
    public void x();  
    y();  
    z();  
}
```

```
class Demo implements Test{
```

```
    ...  
    ... // we must implement  
    ...  
    ... all the 3 methods  
    // extend though we need one  
    method implementation.  
}
```

Solution:-

Adapter class

```
public class MyAdapter implements Test {
```

```
    public void x() {} }  
    y() {} } } Null method definition  
    z() {} }
```

```
public class Demo extends MyAdapter {
```

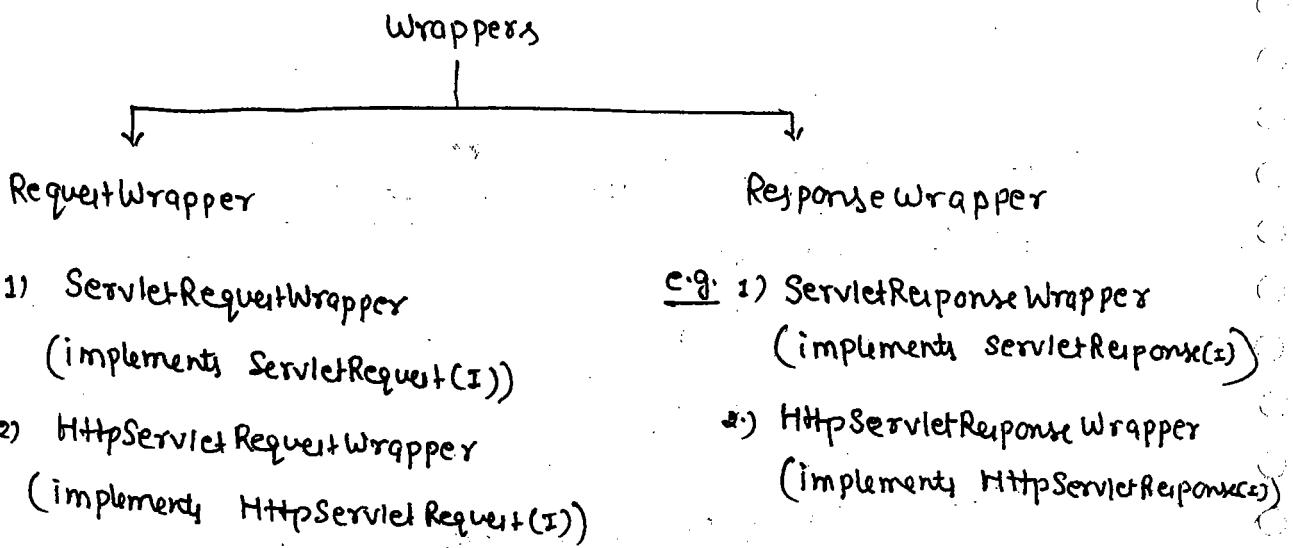
```
    public void x()
```

```
{
```

```
    ... ... ...
```

// Here we are free to
implement that method
in which we are interested in

- ⇒ Integer, Float and etc.... are wrapper classes of java level to simplify the utilization of simple values in the form of objects.
- ⇒ Servlet wrappers are given to simplify the process of working with custom request obj and custom response obj.
- ⇒ To develop custom request obj by custom request class we need to take a class that implements javax.servlet.ServletRequest(I). But providing implementation for multiple methods is very complex.
- ⇒ To overcome this problem we can take the custom Request class, custom Response class extends ServletWrapper class (Adapter class) so we can override only those methods in which we are interested in



All wrapper are Adapter classes.

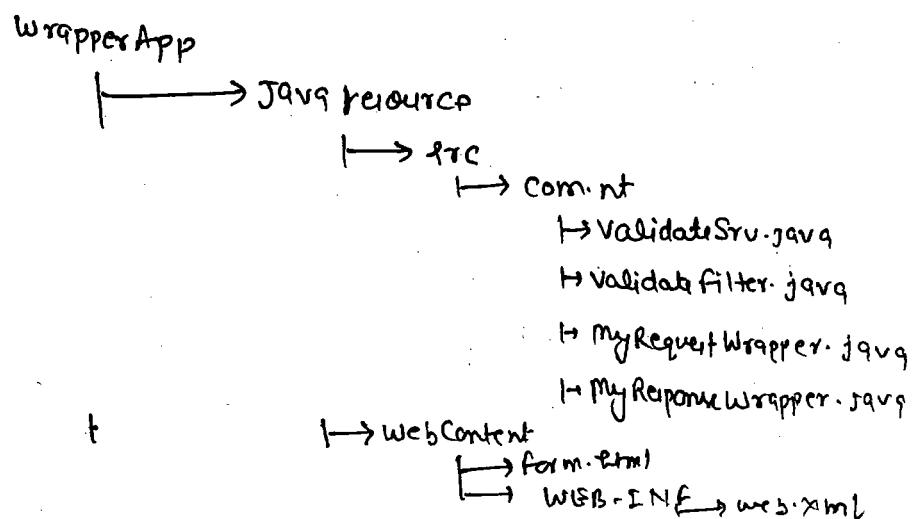
⇒ To pass custom request obj or custom response obj to Servlet
by we create them in Servlet filter and we pass
them to servlet prg by mapping filter with servlet
prg.

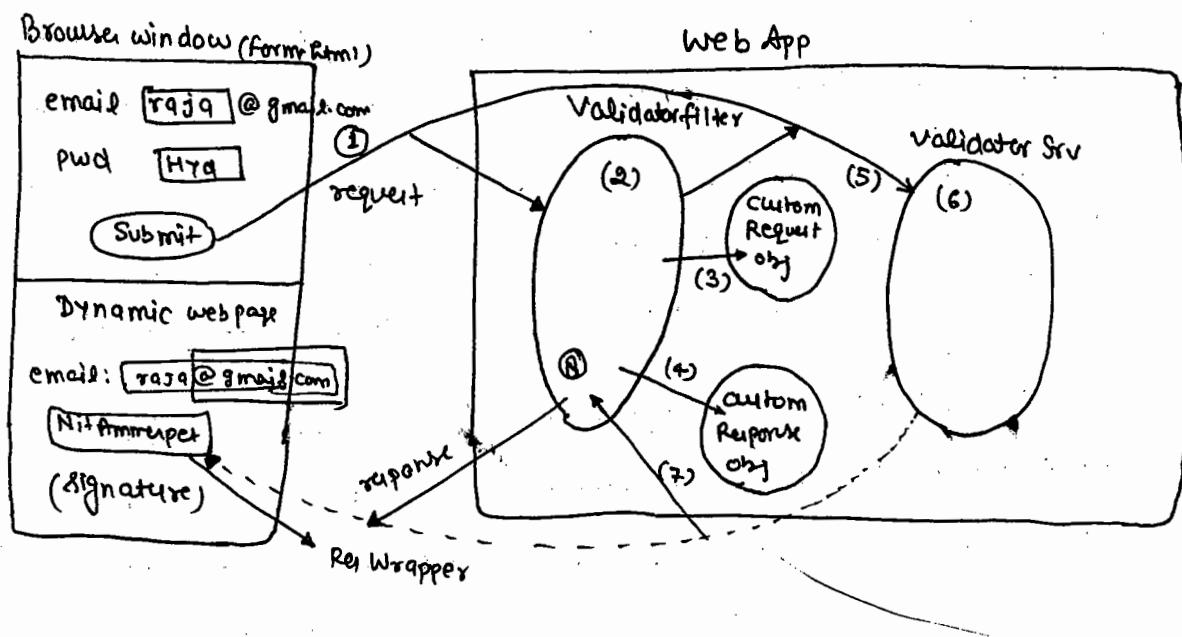
use case of Request Wrapper: —

Checking weather given emailid is having @gmail.com
or not. If not then append it.

use case with Response Wrapper: —

Converting given o/p to upper case output or adding
extra response @ end of existing response as signature.





⇒ There is no need of customer Request class/ customer Response class in web.xml even though they are developed based on servlet Wrappers.

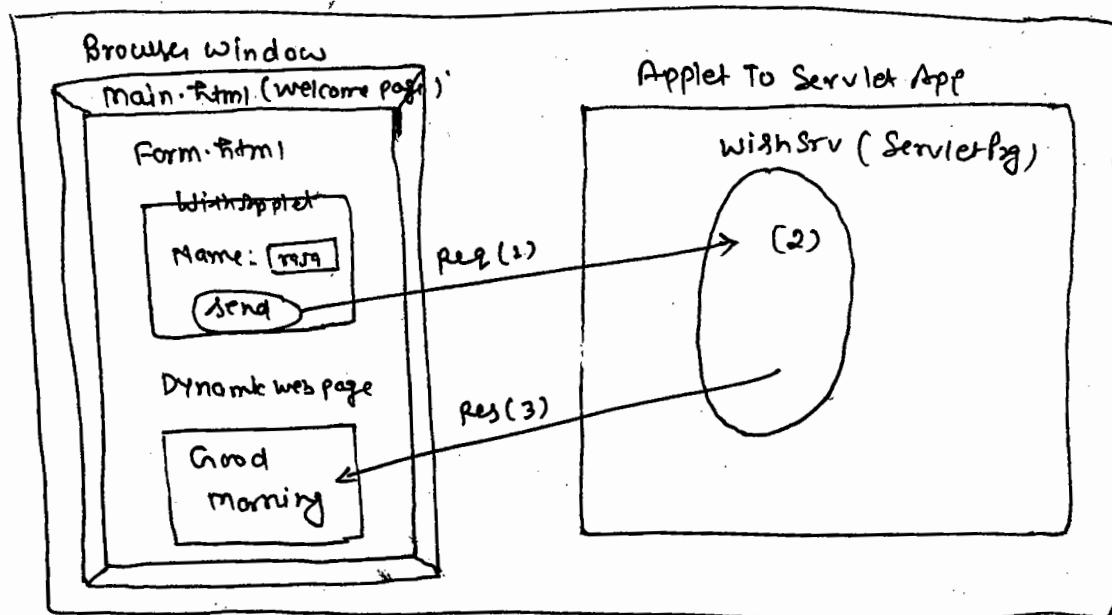
55
| 03/05/15 |

Applet to Servlet communication:

- ⇒ Unsigned applets gives security becoz when they are downloaded, to browser window from Network they can not interact with the files of file system (No virus).
- ⇒ Trusted Applets do not give security becoz they can interact with the files of file system. (virus can be written).
- ⇒ Html pages give good performance but gives bad security. Applets gives good security but gives bad performance.

- In html form page to servlet communication the form data of form page will be prepared as query string and will be appended to request url automatically.
- In Applet to Servlet communication we need to do this work explicitly and we need submit request and receive the result explicitly by using Applet Context obj.
- Applet Context obj means it is the obj of java class that implements `java.applet.AppletContext(I)`. This obj holds info about current applet that is being executed.

Example App on Applet to Servlet Communication



→ To display response of same page we need to take support of Applet frame, javascript, AJAX

AppletServletApp

→ `form.html`, `main.html`, `WithApplet.java`

→ WEB-INF

→ `web.xml`

→ classes

→ `com.mkyong.WishSrv.java`

or → `WishSrv.class`

Note:
Need to config
in `web.xml` file,
but every servlet
must be config
in `web.xml`

File Uploading :-

- ⇒ The process of sending file of client machine file system to Server machine file system is called file uploading.
- ⇒ The process of getting server machine file system to Client machine file system is called file downloading.
- ⇒ While developing Job portal, Matrimony, Social network and etc... web sites there file uploading and downloading are quite common.
- ⇒ To perform file uploading through form page we need to
 - a) Take request methodology as Post.
 - b) Specify multipart/form-data as enctype.
 - c) Take file uploading comps in form page. (`<input type="file">`)
- ⇒ To receive and save uploaded file in server machine file system we need to do following operations in servlet prg
 - a) Use the `getInputStream()` to receive the file and to save file.
(Use `req.getInputStream()` for this)

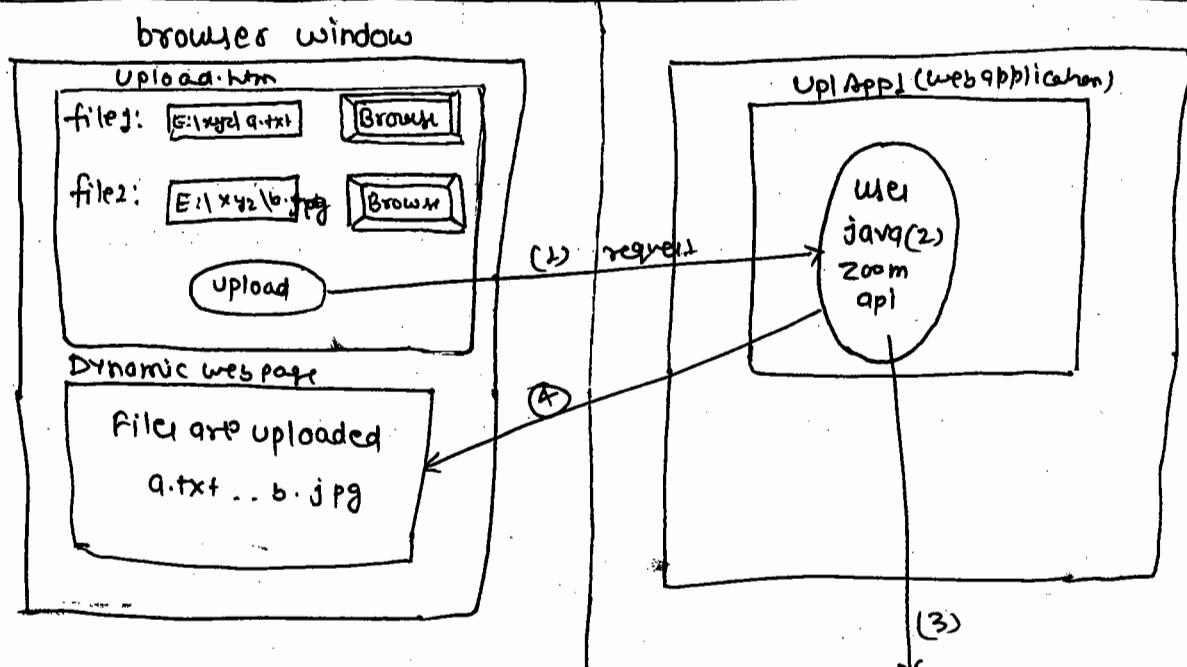
Note:- Working streams is very complex hence to use 3rd party api called javax zoom api in our servlet to complete this file uploading.

Note:- Javax zoom api can be downloaded from www.javaxzoom.com or www.javaxzoom.net in the form of 3 jar file

uploadbean.jar (main jar file)

com.jar
start.jar { Dependent jar files }

The classes of uploadbean.jar file are using classes of com.jar, start.jar.



Client machine file system

E:\xyz

→ a.txt

→ b.jpg

UpApp

→ WEB-INF

→ upload1.htm

→ classes

→ UploadSrv1.java

→ Com. nt

→ UploadSrv1.class

→ lib

→ com.jar, start.jar

→ uploadbean.jar

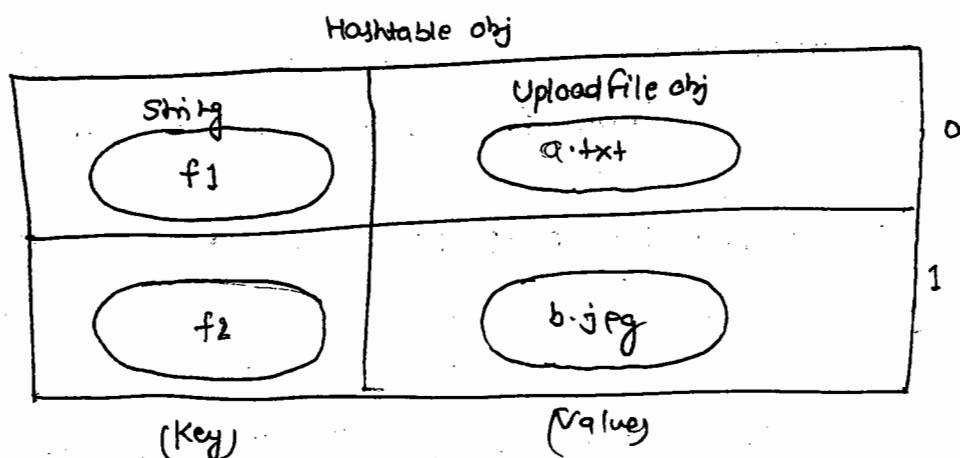
→ web.xml

JAR files in classpath: Servlet-api.jar, uploadbean.jar.

The jar files added to classpath will be used by javac during compilation of servlet program to recognize 3rd party api.

The jar files added to WEB-INF\lib folder will be used by Servlet Container during the execution of Servlet program to recognize and use 3rd party API.

* refer Appⁿ 20 of page no 165 & 166



⇒ Imp classes of javazoom api.

9) UploadBean ---> To specify Dat folder to save the upload file and to complete file uploading

b) `MultipartFile` -> Created on the top of request obj to hold all the values (multiple types of values that are coming along with the request)

c) Upload file → Each obj this class represents one uploaded file

File Downloading:-

⇒ There are two types of file downloading -

a) Response downloading

(Here output/result of servlet/jsp prog will become downloadable file content)

b) Resource downloading

(Here file of web application or file located by web application from file system will become downloadable file. It is like becoming download video from youtube.com and etc....)

⇒ For response downloading place two lines of code in any servlet/jsp prog -

```
res.setHeader ("Content-Disposition", "attachment;  
filename = Title.txt");
```

```
res.setContentType ("application/ms-excel");
```

→ `ContentType` is the response header that gives instruction to browser window to display web page in certain format.

→ `"Content-Disposition"` is another response header that gives instruction to browser either display response on browser window (inline) or to make the response as downloadable file (attachment).

Annotations based servlet Programming:-

MetaData: Data about data is called MetaData. Gathering more details or providing more details about existing data is metaData.

In java we can do metaData operations in 3 ways-

a) Using Java Comments (// or /* ... */)

```
// hold age
int age;
```

b) Using modifiers

```
public static int age;
```

c) Using xml files

like cfg of servlet prg. in web.xml

⇒ XML file based metaData operations (resource cfg) give flexibility of modification but gives bad performance becoz we need to use XML parser to read, process XML documents which is heavy process to perform.

⇒ Annotations are java statements that are alternate XML file based meta operations (resource cfg) which give good performance and bad flexibility of modification.

Annotation Syn:-

```
@(Annotation) (param1 = val1, param2 = val2 ...)
```

Two types annotations -

a) Annotations for documentation (from jdk 1.0)

↳ Should be used along with documentation comments

`/** */` to get special rendering in api documentation.

`@See`, `@author`, `@return`, `@Param`, etc..

b) Annotations for programming (from jdk 1.5)

⇒ Can be used .java files for meta data operations for resource cfg.

`@Override`, `@SupressWarnings`, `@Deprecated`, and etc..

Servlet Annotations:-

→ Introduced from servlet api 3.0 to cfg various resources of java web application like servlet prgs, servlet Filter prgs, Servlet Listners, and etc....

→ By using these annotation we can avoid web.xml file in java web application development.

→ We can't override the cfgs done through annotations by using web.xml. file cfgs....

→ Annotations in servlet programming is incremental approach supporting resource cfg.

→ Since all web frameworks based on servlet api 2.5 (no annotation support).

So the annotation based servlet programming is not taking place real time.

→ Servlet Annotations -

@WebServlet (servlet cfg.)

@Webfilter (Servlet filter cfg)

@WebListener (Servlet Listener cfg)

@WebInitParam (Init param cfg for a servlet prg)

@HandleType (to recognize other classes that are annotated)

@MultipartConfig (to enable multipart form data)

and etc...

→ Tomcat 7/8, Weblogic 12c, WebSphere 8, Glassfish 3/4 supports
Annotation based servlet programming.

Example:-

FormAnnoApp

-----> WEB-INF

-----> form.html

-----> classes

-----> formSrv.java

-----> com

-----> formSrv.class

↳ refer App^n 19 of page no 143.

Eg.1.

@WebServlet ("/sturl")

public class formSrv extends HttpServlet {

...

}

Eg.2

@WebServlet (url pattern = {" /sturl", " / sturl" })

name = "abc"

loadOnStartup = 1)

public class formSrv extends HttpServlet {

... ...

}

e.g.3

```
@WebServlet(urlPatterns = {"/*url", "/text1"},  
    name = "abc",  
    loadOnStartup = 1,  
    initParams = {@WebInitParam(name = "p1", value = "v1"),  
        @WebInitParam(name = "p2", value = "v2")  
    })
```

```
public class formSrv extends HttpServlet {
```

```
}
```

⇒ What happens if we add both web.xml and XML configuration in one application?

The cfg done in annotation will be applied when the Servlet pg is requested through the url pattern cfg.

cfg done in web.xml will be applied when the Servlet pg is requested through url pattern of web.xml.

cfgs done in web.xml file can be used as additional cfgs for annotation cfgs, so we can use web.xml file to cfg those facility which can be cfg directly through annotations. These cfgs are like welcome file cfgs context param cfgs, session cfgs and etc....

(58)
24/05/15

SUNDAY

Different type of URL Pattern:-

X X X

⇒ Every servlet prg is identified with its url pattern. It helps programmers to hide technology that is used to develop web applications.

⇒

- ⇒ For every & servlet prg url pattern cfg is mandatory.
- ⇒ All Servers / ServletContainer, JspContainers are designed based on common Servlet, Jsp specification so they recognize only the above 3 types of url pattern —

1) Exact Match Url Pattern:-

⇒ must begin with "/" symbol ^{not} must contain * (star) symbol.

⇒ Can have one or more letters / words separated with "/" symbols.

e.g. <url-pattern> /text1 </url-pattern>

Request URLs:

http://localhost:3030/DataApp/text1 (valid)

/DataApp/text1.c (invalid)

/DataApp/abc/text1 (invalid)

/DataApp/text1/xyz (invalid)

Note:- The one we have used so far comes under "Exact match url pattern".

Other exact match url patterns:-

Eg1: <url-pattern> /txt1.c </url-pattern>

Eg2: <url-pattern> /abc/xyz/txt2.c </url-pattern>

Eg3: <url-pattern> /Nt/Nt1/ txt1.aspx </url-pattern>

Note- The Regularly used url pattern of real time is exact match.

2) Directory matching:-

⇒ Must begin with "/" symbol and must end with "*" symbol.

⇒ can have multiple letter or words separated with "/" symbol.

Eg: <url-pattern> /image1/books/* </url-pattern>

Request URL's:

http://localhost:3030/DataApp/images/books/hello.c (valid)

/DataApp/images/books.c (invalid)

/DataApp/books/images/hello.c (invalid)

/DataApp/123/456/hello.d/123.c (invalid)

/DataApp/images/books (valid)

Other examples:-

Eg1: <url-pattern> /abc/xyz/* </url-pattern>

Eg2: <url-pattern> /x/y/* </url-pattern>

Eg3: <url-pattern> /y/* </url-pattern>

Eg4: <url-pattern> /Nt/it/tech/* </url-pattern>

Extension match

Syn: $\star.\langle \text{extension} \rangle$

$\langle \text{url-pattern} \rangle \star.\text{do} \langle \text{url-pattern} \rangle$

request urls

Http://localhost:3030/DataApp/abc.do (valid)

/DataApp/xyz/abc/x.do (valid)

/DataApp/abc.x (invalid)

/DataApp/abc.do/xyz.c (invalid)

/DataApp/.do (valid)

Other examples:-

Eg1: $\langle \text{url-pattern} \rangle \star.c \langle \text{url-pattern} \rangle$

Eg2: $\langle \text{url-pattern} \rangle \star.xhtml \langle \text{url-pattern} \rangle$

Eg3: $\langle \text{url-pattern} \rangle \star.htm \langle \text{url-pattern} \rangle$

⇒ The Servlet that takes all requests having diff/request URLs and delegates the requests other web resource page is called front Controller Servlet. While cfg this servlet we need to use either directory match or extension match url patterns.

$\langle \text{url-pattern} \rangle /text1/text2.c \langle \text{url-pattern} \rangle$

(it is exact match url pattern)

$\langle \text{url-pattern} \rangle /x/y/\star.do \langle \text{url-pattern} \rangle$

(Invalid url-pattern formation)

⇒ if clash comes b/w exact match and directory match url pattern then priority will be given to Exact match url - pattern.

Srv1 url pattern: /xyz/* (Directory match)

Srv2 url pattern: /xyz/test1.c (Exact match)

http://localhost:3030/ DateApp /xyz/test1.c
(Srv2 executed)

⇒ if clash comes b/w Directory match and extension match the priority will be given to Extension directory match

Srv1 url pattern: /xyz/* (Directory match)

Srv2 url pattern: *.c (Extension match)

http://localhost:3030/ DateApp /xyz/test1.c
(Srv2 executes)

Exactmatch → DirectoryMatch → ExtensionMatch

⇒ if clash comes b/w Exact match and Extension match then Priority will be given to exact match.

⇒ In Tomcat & JBoss if two servlet programs are having same url pattern exception will be raised.

If multiple servlet programs are having directory match url pattern if clash is coming b/w them for priority will be given to more nearer matches

⇒ If multiple servlets are having same logical name
the last servlet in the list gets higher priority.

(59)
31/05/15

(60)
 01/05/15

→ While working with FORM model of authentication we can cfg our own form page as login page and our own page as error page.

<form-login-config>

<auth-method>

<form-login-page> /login.jsp </form-login-page>

<form-error-page> /login-fail.jsp </form-error-page>

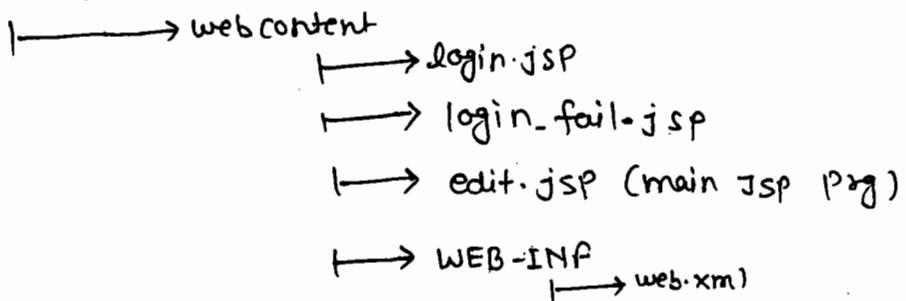
</form-login-config>

→ Since request given by form login should taken by Server env / container directly we must place following fixed values

- a) action url : j_security_check
 - b) user name text box name : j_username
 - c) password box name : j_password
 - d) request method : POST
- } fixed name

for example Appn "FORM" model authentication ref APP 31 of
Page no 169

FormSecApp2



//comment :-

edit.jsp (main page that will be protected)

login.jsp (our own form page)

3987 POST → mandatory, action = "...fixed
 & requesturl: http://localhost:3030/FormSecApp2/test2

CLINT-CERT Model:-

⇒ It's not given for authentication, it is given for utilizing digital certificate to encrypt client supplied data over the n/w. we can create these digital certificate by using algorithms like RSA, Verisign and etc.

Example

Step1) Create digital certificate for CLINT-CERT using RSA (Rivest, Shamir, Adleman) algorithm.

*
keytool -genkey -alias niti -"

Step2) Enable "Http on SSL" protocol in Tomcat Server by cfg the name & location of digital certificate.

```
<connector = "org.apache.coyote.http11.Http11Protocol"  
port = "8443" maxThreads = "200"
```

```
schema = "https" secure = "true" SSLEnabled = "true"
```

```
keystorefile = "C:/webs/HIT/.keystore" keyStorePass = "raw9"
```

```
clientAuth = "false" sslProtocol = "TLS" />
```

Step3 restart the server

https → using https

ssl → secured servlet

http://localhost:8443/fvotapp/inout.html.

* * Keystore

- ⇒ Now Server send digital certificate to Client(browser) → Browser.
- ⇒ Receive & install this digital certificate at Client

Live Hosting of class room web application:-

Domain Registrars:-

These companies will do business in the following sectors -

- By selling space of web server / Application Server that is installed in machine having static IP Address (fixed)
- By selling domain names
- By Maintaining websites
- By doing SEO for websites.

Eg:-

godaddy , jelastic, bigrocks , indibricks, j2e and etc ..

Jelastic gives 14 day trial period for Java hosting with tomcat, jboss, and etc servers and with mysql, postgresql db slow's support

Procedure Deploy web appn in jelastic (without DB etc)

Step1:- prepare war file.

Step2:- submit email id to jelastic.com get password

jelastic → sign in → layer shift

Step3- Submit the above email id & received password for login

Step4:- Create env... by choosing Tomcat7 server (NITENV2)

Create environment → tomcat → java 7 → choose environment name → create

Step5

Upload war file & deploy war file

Deployment manager → upload → browse & select

upload ← NITEnv2 ← upload ← VoterApp.war

Step6 Test the application

NITEnv2 → Tomcat → open in browser

nitenv2.jayShift.co.uk / input.html

⇒ To modify html file content of hosted appn ~~NITEnv2~~

NITEnv2 → Tomcat → config → downside → webapps

→ Root → input.html modify & save.

⇒ To modify servlet prg content of the hosted application

NITEnv2 → Tomcat → config → downside → webapps

→ Root → WEB-INF → classes → com.nt → upload

→ Select the modified / updated VoterSrv.class file

i.e. we need to modify VoterSrv.java outside
& we need to compile that program with
outside itself.

Hosting of web appn in Jealousic.com with MySQL Database

S/w:-

Step1 Create Env (NitEnv3) choosing Tomcat , MySQL DB & w

Step2 Collect MySQL DB username, password from email account.

Step3 Launch MySQL admin window to create DB table with cols having records

NITENV3 → MySQL → Node → open browser

→ New → Create DataBase →

→ Create table

Step4 Gather jdbc url of above logicalDB

NITENV3 → MySQL → Node5d → Additional info

jdbc:mysql://~~localhost~~ node 9821 + -NitEnv2.j. layershift.co.ulc/
NitDB

Step5 Make DB App appn working with MySQL

- 1) change JDBC properties in web.xml file for MySQL
- 2) MySQL-connector-jarv5.1.6.jar file add lib folder
- 3) Prepare war file (DBApp.war);

1) >jar cf DBApp.war (create war file)

Step6 Upload war file (DBApp.war) in Nit2Env

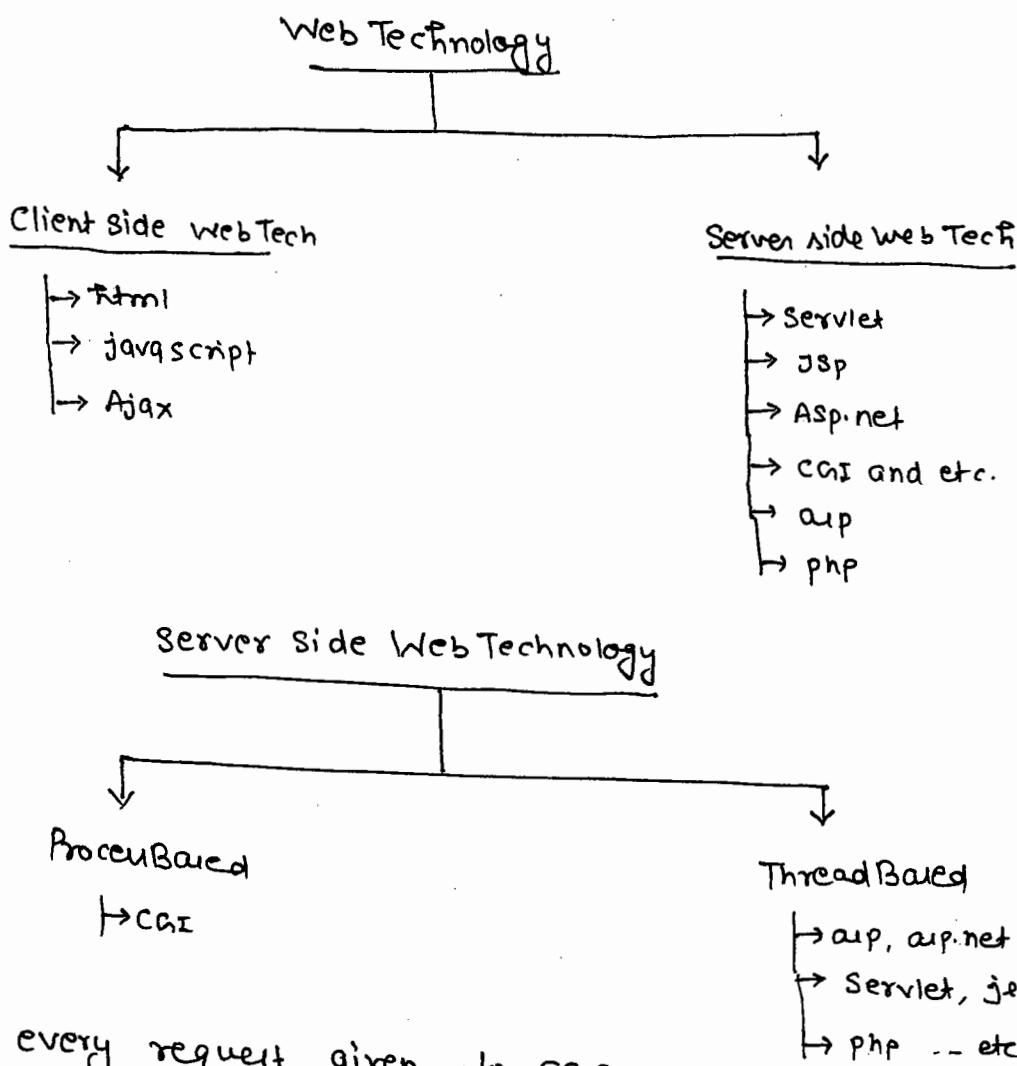
Step7 Test the Appn

JSP

(java Server page)

①
30/04/15

Web application is collection of web resources prg having capability to generate web pages. To develop these web resources prgs we need web technology.



- ⇒ For every request given to CGI prg one process will be created and this process is heavy weight consume more resource. Due to this CGI Appn performance is very poor.
- ⇒ For every request given to servlet/jsp prgs one thread will be created. These threads are lightweight consume less resource. Due to this servlet/JSP Appn give very good performance.

Q) Why Sun MicroSystem has given JSP Technology where they have already given servlet technology?

A) In the initial days of servlets Sun MS failed to attract asp programmers towards servlet. The reason are -

- a) Asp supports tag based programming, servlet doesn't support.
- b) To work with servlet strong knowledge is required. It is very difficult for asp programmer to learn java.

To overcome above problems Sun MS has given a tag based technology called jsp having all the features of servlet. So any programmer can use jsp without having strong knowledge of java/servlet.

Limitation of Servlet:-

- ⇒ Strong java knowledge is required.
- ⇒ Not suitable for non-java programmers.
- ⇒ Placing html code in servlet program is complex and error prone process.
- ⇒ Makes the programmer to mix up presentation logic & b. logic (java code).

e.g.

```
bw.println("<font color = \"red\">Hello </font>").
```

- ⇒ Any modification in servlet prog will be reflecting only after recompilation of servlet prog overloading of web appn.
- ⇒ Doesn't give implicit objs. request, response, Servlet Context and etc are not implicit obj they are container created obj. So we need to write additional code to access those obj.

- ⇒ Servlet cfg in web.xml is mandatory.
- ⇒ Servlet is not good for business logic not good for persistence logic

Features of JSP:-

- ⇒ Supports tag based programming.
- ⇒ Gives 9 implicit obj.
- ⇒ Allows us to write separate both b.logic & p.logic.
- ⇒ Modifications done in JSP program will be reflected automatically without reloading the web app.
- ⇒ Strong java knowledge is not required
- ⇒ Suitable for both java and non-java programmers.
- ⇒ Cfg of jsp in web.xml file is optional.
- ⇒ Allows us to use all the features of servlet bcoz every JSP pg internally one equivalent servlet pg will be generated. generated.
- ⇒ Gives automatic exception handling. supports

Note:-

- In initial days programmers have used JSP as complete alternate to servlet. But now a days programmers are using both servlet, JSP technology together in web app development.
- ⇒ For every JSP pg and equivalent servlet pg will be generated internally using page compilation process. Every server provides one JSP page compiler in JSP container.

⇒ Tomcat Server Jsp Container is: Jasper
Servlet Container is: Catalina

Jsp container internally uses servlet container

Servlet API: (servlet-api.jar)

→ (javax.servlet, javax.servlet.http, and etc...)

JSP - API: (jsp-api.jar)

→ javax.servlet.jsp, javax.servlet.jsp.el
javax.tag and etc...

Note:- JSP - API internally uses servlet-api-page compiler
Page compiler: JSpC (org.apache.jasper.JSpC)

⇒ JSP program is executing means Jsp equivalent servlet is executing internally.

JSP Definition:-

- ⇒ It is a Sun's supplied Java based web technology that allows to develop tag based web resource prgms in Java web appn having capability to generate dynamic web page.
- ⇒ Jsp is a web technology that gives api for vendor companies having rule and guide lines to develop Jsp container and also gives tags for programmers to develop web resource prg
- ⇒ Jsp container executes Jsp program.
- ⇒ Servlet Container executes servlet prg.
- ⇒ Jsp api latest version: 2.3, servlet api latest version: 3.1.

- ⇒ Every Jsp prg / compl page Should have extension .jsp.
- ⇒ Since html, Jsp prg contains tag it is better to call them as html, pages / Jsp pages.
- ⇒ Every Jsp page Contains Template Text and Java code.
This template text generates static Context and java code generate dynamic context.

(2)
04/05/15

Structure of Jsp prg

<filename>.jsp

```

< --- (html code)
    --->
    === ordinary
    === text
    <% === == (javacode)
    --- - - - %>
    === === plaintext
  
```

Template text = html code + ordinary text
Jsp page generate dynamic web page
If few lines of web page are dynamic
then total webpage is called as dynamic web page.

ABC.jsp

```

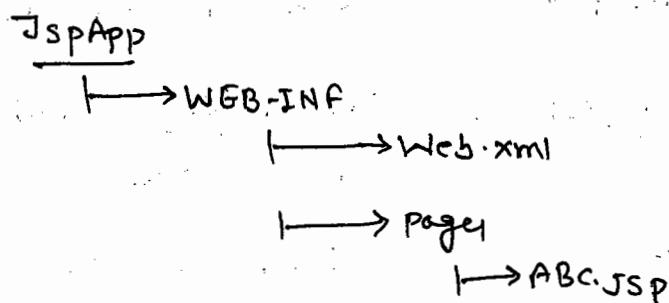
<b> <center> Welcome </center> </br>
<br>
Date and Time <br>
<% java.util.Date d = new java.util.Date(); %> (Java code)
<br> Out.println( ... )
<br> end (template text)
  
```

Output of Jsp prg (Dynamic web page)

Welcome
Date And Time : <u>4/05/2015 IST 9:39 AM</u>
dynamic context
end

⇒ Placing Jsp prg outside WEB-INF is not recommended because it become directly access to end client since it is placed in public area. This gives security access and other problems.

To overcome this problem place Jsp prg inside WEB-INF folder so they can't be accessed directly but they can be accessed either through Servlet prg or by using url pattern that is configured in web.xml file for Jsp prg.



Q) How many types of objects can be there in Jsp program and Jsp page?

Ans) Two types of objects —

a) Implicit Obj:—

These obj. will be created in JES (Jsp equivalent Servlet) class automatically. we can use this obj directly in our Jsp scriptlets (`<% ... %>`) without writing any code.

The 9 implicit objects are —

out, request, response, exception, session, page, pagecontext, config, application.

(b) Explicit objs:

The objs that are created by the programmer manually in Jsp page.

<% java.util.Date d = new java.util.Date(); %>

	Html	Jsp
(i)	Client side web technology.	Server side web technology
(ii)	Generates static pages	Generates dynamic pages
(iii)	Needs Html interpreter to execute html code	Needs Jsp container to execute code.
(iv)	Given by W3C	Given By Sun's
(v)	Doesn't allow to use 3rd party custom tag	Allow to use
(vi)	Doesn't allow to place Java code.	Allow to place.

(3)
05/05/15

Jsp Life cycle method:-

~~jspInit()~~
~~-jspService(-,-)~~
~~-JspService(-,-)~~

jspInit() -----> For Instantiation Event
 -jspService(-,-) --> For request arrival Event
 jspDestroy() ---> For Destruction Event.

Technically speaking the above should be called as "life cycle convenience methods.", not the life cycle methods, becoz container calls never calls these methods directly for life cycle events. Container calls these methods through Servlet Life cycle method.

- jspInit() is called from init() method.
- -jspService(-) from service(-,-) method.
- jspDestroy() is from destroy() method.

jspInit() ---> useful for initialization logic like creating jdbc connection.

-jspService(-,-) ---> useful for request processing

jspDestroy() ---> useful to uninitialization logic like close connection

* By default every JES class gives only -jspService(-,-) method. In order to place other methods we need to provide override jspInit() & jspDestroy in JSP program.

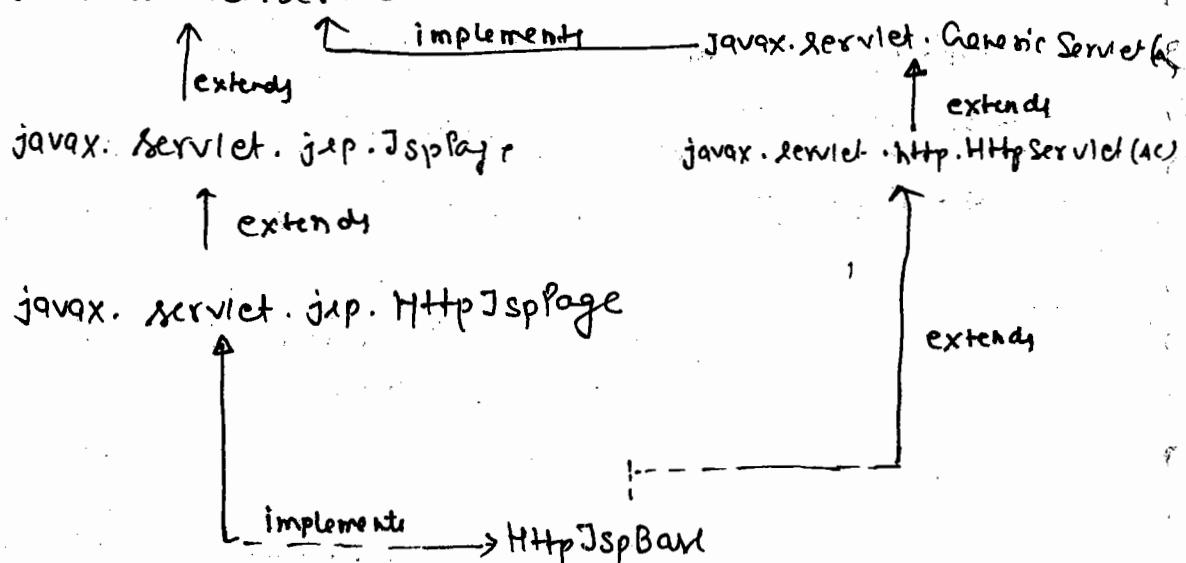
Generally the methods generated by container & can't be overridden by programmer contains (-) underscore symbol in the method name.

- jspService(..) is such method given by JSPContainer in JES class.

In Tomcat service server for abc.jsp will be generated abc-jsp in `<tomcat-home>\work\localhost\jspApp\org\appach\jsp` folder having two files -

`ABC.jsp.java`, `ABC.jsp.class`

`javax.servlet.Servlet`



In Tomcat Server Jsp page Compiler name is: JSPC

In Weblogic Server Jsp page Compiler name is: weblogic-JSPC

In Tomcat Server JES class for ABC.jsp is ABC-Jsp.

In Weblogic server JES class for ABC.jsp is _abc

There are two phases in JSP prg/page execution

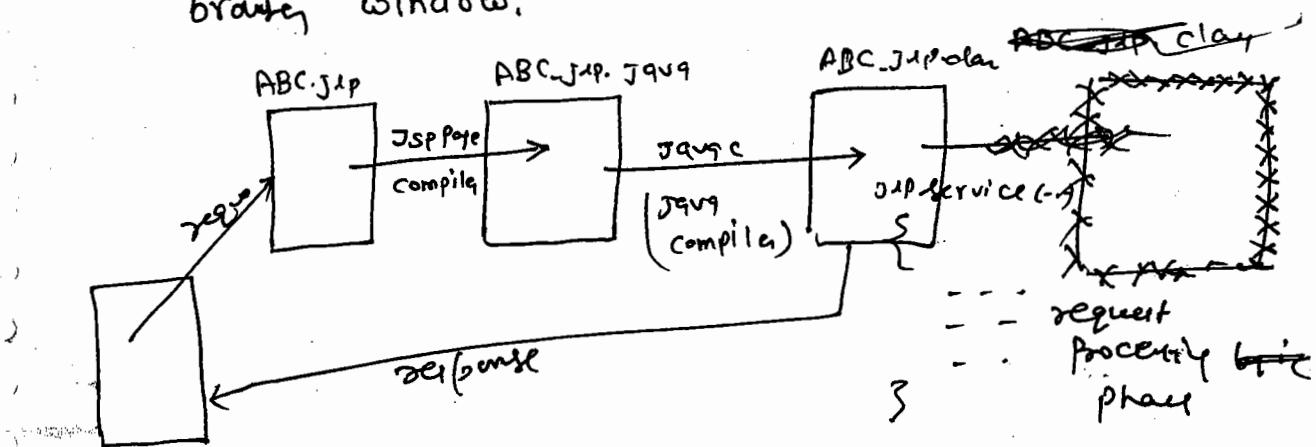
a) Translation Phase:-

Converts the JSP page into equivalent servlet source file & compiled file.

b) Request Processing Phase:-

Executes `-jspService(...)` of JES class through service()

to process the request & to send response to browser window.



The request given to JSP Page participates in request processing phase if JSP to source code is not modified compare to previous request & JES class is available otherwise the request given in JSP page participates in both translation phase & request processing phase.

- ⇒ if we delete .java file of JES then the request will not participate in Translation phase
- ⇒ Every JSP Container remembers the source code of JSP program until next request comes to that JSP program. In this process the current source

code & previous source code will be compared by using Wdiff, or axis tool.

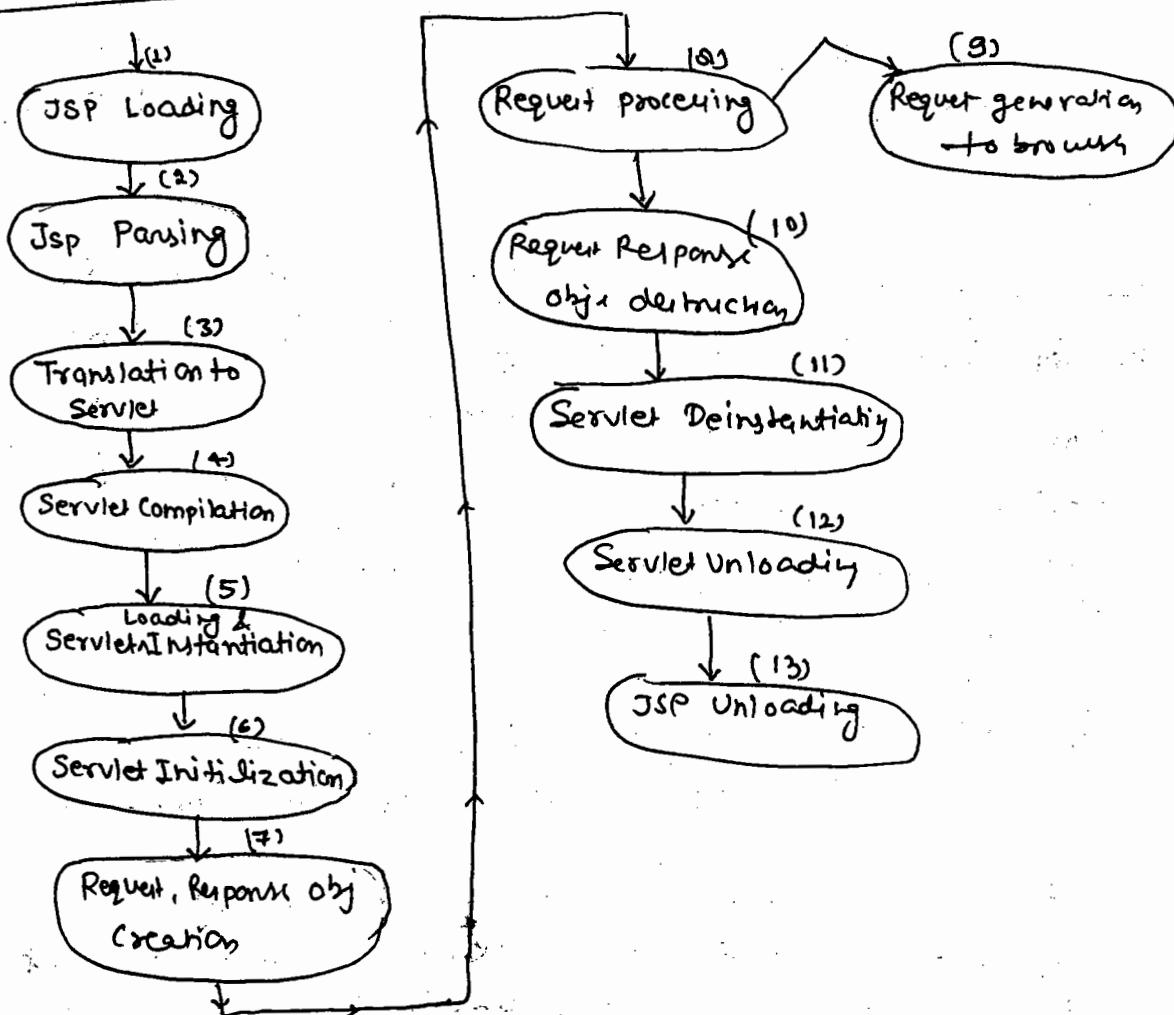
Q) what happens if we modify JSP equivalent Servlet?

A) In Tomcat 6 server the modifications will reflect in the output if u recompile servlet prg & reload the web appn.

In Tomcat 7, 8 servers the modifications will not be reflected irrespective of any condition.

④
06/05/15

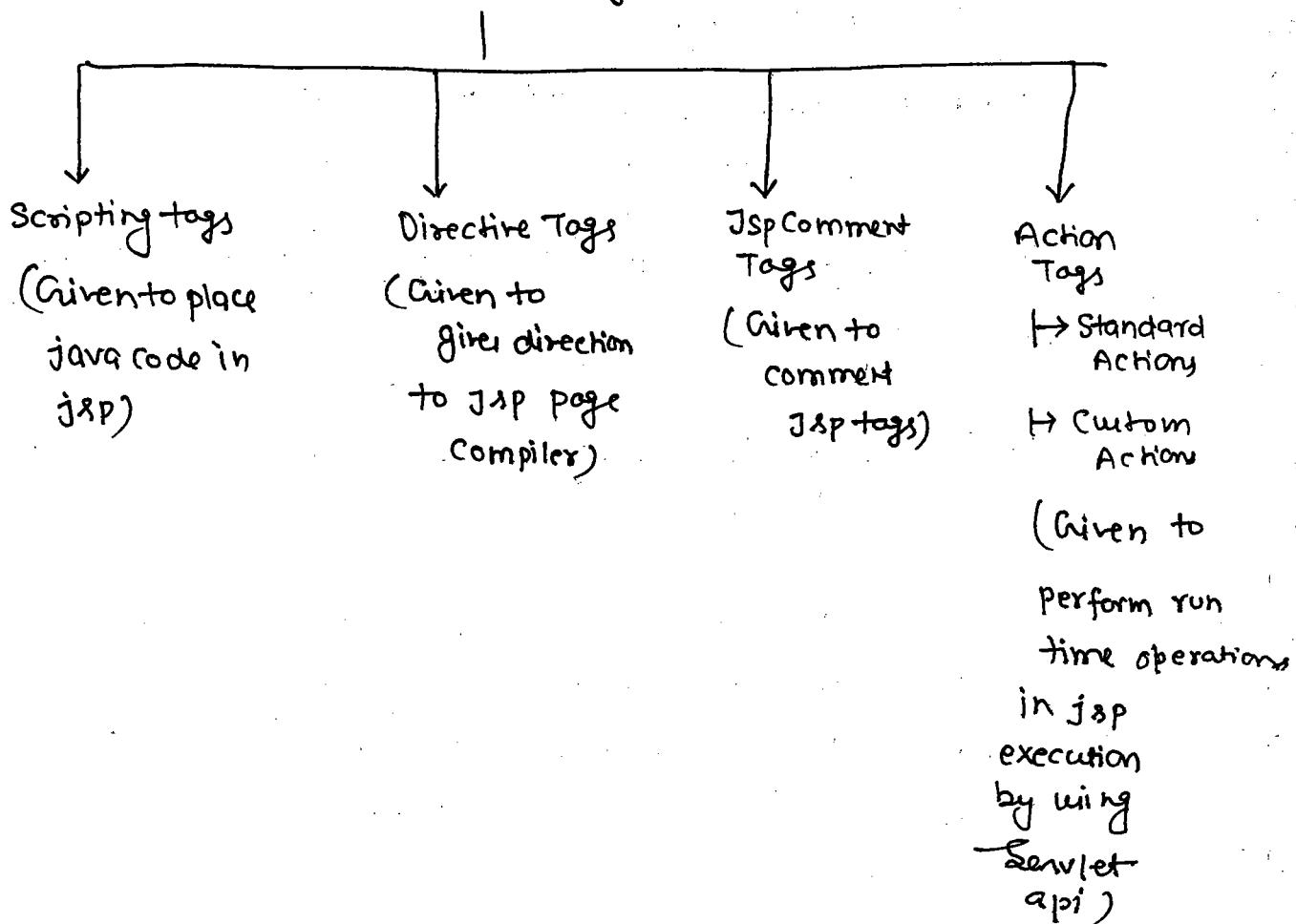
JSP flow of execution:-



- (1) Container loads .jsp file from memory.
- (2) Container verifies the syntax of JSP tags placed in .jsp file.
- (3) Using JSP page compiler JSP program will be converted into an equivalent servlet class (JES class)
- (4) The java compiler (javac) compiles JES class source file.
- (5) Container loads JES class and creates obj using no-param constructor.
- (6) Container calls jspInit() through init(ServletConfig) method to initialize ServletConfig obj with our Servlet class obj JES
- (7) Container creates one set of request, response obj for current request.
- (8) Container calls _jspService(-,-) method through service(-,-) to process the request.
of JES
- (9) The output generated by _jspService(-,-) method goes to browser as response.
- (10) Once response is delivered the request, response obj will be destroyed.
- (11) if there is no further request to jsp or if web appn stopped / reloaded the container performs Servlet DeInitialization. When this is about to happen container calls jspDestroy() through destroy().

- (12) JVM unloads the loaded JES class.
- (13) Container unloads the loaded .jsp file.
- ⇒ When 1st request is given to JSP, 1 to 10 operation will take place.
- ⇒ When other than 1st request is given to JSP without modifying the source code of JSP program 7 to 10 operation will take place.
- ⇒ If web app is stopped or reloaded or undeployed, if server is stopped or restarted or if JSP is not getting request from long time then 11 to 13 operation takes place.

Various JSP Tags



- ⇒ The code that is embedded with tags is script code
- ⇒ Java code script code will be embedded with html tags so it is called script code.
- ⇒ Java code placed in JSP code is called script code.

Scripting tags:-

- 1) Script Scriptlet (`<% %>`)
- 2) Declaration (`<%! %>`)
- 3) Expression (`<%= %>`)

Directive tags

- 1) Page Directive (`<%@ page %>`)
- 2) Include Directive (`<%@ include %>`)
- 3) Taglib Directive (`<%@ taglib %>`)

JSP Comments

```
<%--  
--%>
```

Standard Action (Built-in Actions)

<code><jsp:include></code>	<code><jsp:attribute></code>
<code><jsp: forward></code>	<code><jsp: param></code>
<code><jsp: writeln></code>	<code><jsp: param></code>
<code><jsp: getProperty></code>	<code><jsp: plugin></code>
<code><jsp: SetProperty></code>	<code><jsp: fallback></code>
<code><jsp: element></code>	and etc ...

A) Scripting tags

1. Scriptlet

Standard syn:

```
<%  
...  
%>
```

xml syn:

```
<jsp:scriptlet>
```

```
</jsp:scriptlet>
```

The code placed in scriptlet goes to _jspService(-,-) of JES class as it is.

This code is useful to process or request processing logic to process the result request.

In Jsp

```
<% java.util.Date d = new java.util.Date(); %>
```

In JES class

```
public void _jspService(-,-){
```

```
...  
...  
...
```

```
java.util.Date d = new java.util.Date()
```

```
...  
...  
...
```

```
}
```

- ⇒ Variable declared in scriptlet becomes local variable of `_jspService(-,-)`.
- ⇒ All implicit objs of jsp are local variables of `_jspService(-,-)` method, and the code of scriptlet also goes `_jspService(-,-)`. So we can use implicit Obj in scriptlet.

`<%`

```
int a=10;
```

```
out.println("A value" + a);
```

```
out.println("Current Browser name" + request.getHeader  
("user-agent"));
```

`%>`

- ⇒ We can't place method defination in jsp because they become nested method of `_jspService(-,-)` and java doesn't support nested methods.

`<% public int add(int x, int y) {`

```
    return x+y;
}
```

`%>`

xml Syntax -

```
<jsp:scriptlet>  
    int a=10;  
    int b=20;  
    if(a < b)
```

```
        out.println("A value small");
```

```
    else out.println("B value small");
```

while working with xml syntax u should be very careful towards less than symbol becoz it give the indecation of subtag begining for less than symbol even though it used as conditional operator.

`</jsp:scriptlet>`

with standard tag scriptlet this problem will not be here.

To overcome this problem use `<![CDATA[...]]>`. This makes JSP parser to take the body of tag as character data without applying any XML meaning.

```
<jsp:scriptlet>
    if (5<10)
        out.println("5 is small");
</jsp:scriptlet>
```

Gives error becoz "<" symbol will be taken XML subtag.

Solution 1 (not recommended)

```
<!
    if (5<10)
        out.println("5 is small");
    />
```

Solution 2

```
<jsp:scriptlet>
    <![CDATA[
        if (5<10)
            out.println("5 is small");
    ]]>
</jsp:scriptlet>
```

07	05	15
----	----	----

3) Declaration:-

Standard Syntax:-

```
<%!
  ...
  ...
  ...
%>
```

--> // instance variable
 --> // declaration
 --> // method definition

XML syntax:

```
<jsp:declaration>
  ...
</jsp:declaration>
```

⇒ The code placed in declaration tag goes to JES class outside to the -jspService(-,-) method.

⇒ Variables declared in declaration tag becomes global variables in JES class.

(instance variable / static variables)

e.g. `<%! int a=10;%>`

⇒ Variables declared in scriptlet becomes local variables of -jspService(-,-) whereas the variables declared in declaration tag becomes global variable in JES class

Q) How can we differentiate declaration tag variable from Scriptlet tag variable inside scriptlet when both have got same name.

A) Use either page or this.

implicit
Object

`<!int a= 10;%>`

~~<%~~ `<% int a= 20;`

`out.println ("Local a value" + a);`

```
out.println("Global a value" + this.a);  
ABC.jsp o1 = (ABC.jsp) page;  
out.println("Global a value" + o1.a);  
%>
```

Note:- Don't we implicit object page in the above situation becoz it makes you type cast with JES class name which changes server to server.

Declaration tag for defining method:—

```
<%! public int sum(int x, int y) {
```

```
    return x+y;
```

```
}
```

```
%>
```

```
<% out.println("Result is" + sum(10, 20)); %>
```

⇒ Implicit objects are not visible in declaration tag becoz they are local variables of _jspService(-,-) method and the code placed in declaration tag ~~becomes~~ goes to outside _jspService(-,-) method.

⇒ We can also use declaration tag to place jspInit() jspDestroy() method definitions in our JSP program.

```
<%! public void jspInit() {  
    System.out.println("ABC.jsp: jspInit()");  
}  
%>  
<% System.out.println("ABC.jsp: _jspService(-,-)");  
%>
```

Hello

```
<% public void jspDestroy(){  
    System.out.println("ABC.jsp: jspDestroy()");  
}%>
```

- Q) What is the use of enabling <load-on-startup> on JSP program?
- A) The container performs following operations on JSP prg either during server startup or during deployment of web application.
- a) JSP prg will be translated into JES source code
 - b) JES source file will be compiled
 - c) JES class will be instantiated and initialized

Due to this the first request give JSP prg directly participates in request processing phase.

<Servlet>

<s-n> XYZ </s-n>

<jsp-file> ABC.jsp </jsp-file>

<load-on-startup> 1 </load-on-startup>

</Servlet> Priority

<Servlet-mapping>

<s-n> XYZ </s-n>

<url-pattern> /tot1 </url-pattern>

</Servlet-mapping>

Note:-

We can get <load-on-startup> benefit only JSP page request with its url pattern

Http://localhost:3030 / JspApp / test1

⇒ <load-on-startup> enabled on the JSP page performs pretranslation, pre-instantiation & pre-initialization

- Q) Can we place _jspService(..) in our JSP page?
- A) Not possible, becoz it becomes duplicate method for the already available _jspService(..) method of JES class and Java doesn't support duplicate method.
- Q) Can we place servlet life cycle method in our JSP program?
- A) No, becoz the super class of JES class has made init(), service(), destroy() method of final methods. (refer HttpServlet) and we can't override final method.

<jsp:declaration>

```
<! [CDATA [
    public int findSmall(int a, int b) {
        if (a < b)
            return a;
        else
            return b;
    }
]}>

</jsp:declaration>
```

Small value: <jsp:scriptlet>

```
out.println( findsome(10, 20));
```

```
</jsp:scriptlet>
```

⇒ In one JSP pg we can place any no of scripting tags in any order having both XML, standard syntax.

⇒ By default JES class gives only _jspService(..) method so we can place jspInit(), and jspDestroy() method explicitly in our JSP pg or page by using declaration tag.

(6)

08/05/15

3) Expression:-

⇒ Evaluates the given expression displays generated results on to browser window.

⇒ Any thing that returns result is called expression.

e.g. Arithmetic operation, logical operation, method call, instantiation and etc..

Standard syn:-

```
<%. = <expr>%>
```

xml syn:-

```
<jsp:expression>
```

```
<expr>
```

```
</jsp:expression>
```

```
<%! int a=10;  
    int b=20; %>
```

A value `<%= a %>` `
` In JES: `out.println(a);`

B value `<%= b %>` `
` In JES: `out.println(b);`

Sum value `<%= a+b %>` In JES: `out.println(c);`

`<%= a %>` is Big value than `<%= b %>` value? `<%= a>>b%>`

Note:- if expression tag is used perfectly then there is no need of placing using `out.println()` in Jsp pg.

- ⇒ The code placed in expression tag goes to `_jspService(-,-)` method of JES class, so we can use implicit object in expression tag.
- ⇒ We can use expression tag for instantiation and to display data of the object
- ⇒ We can use expression tag to call both user-defined and pre-defined methods, which return results.

In Jsp Page -

```
<%! public int add(int x, int y){  
    return x+y;  
}  
%>
```

Sum of 10,20 `<%= add(10,20) %>` `
`

Browser name `<%= request.getHeader("user-agent") %>`

Current Date and Time `<%= new java.util.Calendar.getInstance() %>`

`get(java.util.Calendar.HOUR_OF_DAY)`

Date and Time `<%= new java.util.Date() %>`

- ⇒ Using Expression tag we can't call java method whose return type is void.
- ⇒ Generally we use expression tag to evaluate one expression at a time. But we can place multiple expression in one tag by concatenating their results as shown below.
- ⇒ The code placed expression tag goes to JES class & becomes the argument value of `out.print()` method.

e.g.

```
<% int a=10; %>
```

Square and Cube: `<% = a*a + " " + a*a*a %>`

In XML Syntax of expression tag even ~~the CDATA can't solve~~ `<![CDATA[--]]>` can't solve < (less than symbol) problem.

```
<% int a=10;
    int b=20; %>
```

`<% = a %>` is bigger than `<% = b %>` ?

`<jsp:expression>`

`a>b`

`</jsp:expression>`

~~<% = a %>~~ `<% = a %>` is smaller than `<% = b %>` ?

`<jsp:expression>`

`<![CDATA [`

`a<b`

`]]>`

`</jsp:expression>`

Developing Jsp pg based web app in Eclipse IDE

Step1) Launch Eclipse IDE & configures Server

window → preferences → server → Runtime Environment → Add
→ Tomcat7 → installation directory

D:\Tomcat7.0

Step2) Create Dynamic Web project with name JspApp1

Step3) Place the following Jsp program having all the three scripting tag.

First Jsp

```
<%@ page import = "java.util.*" %>
<%!
    public String sayHello() {
        String un
        Calender cl = Calender.getInstance();
        int h = cl.get(calender.HOUR_OF_DAY);
        if (h <= 12)
            return "AM" + un;
        else if (h <= 16)
            return "A.M" + un;
        else if (h <= 20)
            return "P.M" + un;
        else
            return "NIGHT" + un;
    }
%>
```

Declaration tag

```

<h1> <center> Welcome to Jsp </center> </h1>} Template
          +text
Date and Time <% = new Date() %> <br> //expression
<% String name = "raja"; %> //Scriptlet
With msg: <% = sayHello(name) %> //expression

```

Step4) Run the application.

Website Click → Run as → Run on Server → next → Print → Right

Comments in Jsp page

- 1) Jsp Comments / Hidden Comments <%-- --%>
(To comment jsp tags)
(Will be recognized by Jsp Page compiler)
- 2) Scripting Comments / Java comments //→single line, /*...*/(multiline)
(To code java code of Jsp pgf)
(Will be recognized by Java C)
- 3) Html comments or output comments (<!-- -->)
(To comment html code or template text of Jsp pgf)
(Will be recognized by html interpreter)

Whenever comment is recognized by compiler/interpreter
then it places that portion code with white space.

Comment	Visibility in JES source	Visibility in JES bytecode	Visibility in Html that goes to browser	Visibility in output
JSP comments	no	no	no	no
Java Comments	yes	no	no	no
Html comment	yes	yes	yes	no

(7)

09/05/15

Scope in Jsp:-

page scope: Specific to each Jsp page

request Scope: Through out the request

response Scope: Through out the response

session Scope: Within web appl'n but specific to a browser

application Scope: Within web appn with out any condn

	implicit obj	type	scope
i)	request	HttpServletRequest(I)	request
ii)	response	HttpServletResponse(I)	response
iii)	page	Object(c)	page
iv)	PageContext	PageContext(AC)	page
v)	Config	ServletConfig(I)	page
vi)	application	ServletContext(I)	application
vii)	session	HttpSession(I)	session
viii)	out	JspWriter(AC)	page
ix)	exception (only in error page)	Throwable	page

⇒ All implicit objs are the objs of underlying container supplied java classes which implement the above specified type interface or extends from above specified type class.

⇒ JspContainer class internally uses servletContainer due to this some classes of Jsp implicit objs are given by servlet

Container

Html to Jsp to DB S/W Communication:-

- ⇒ For html to jsp communication place jsp file name / url pattern or action url or fireurl.
- ⇒ For jsp to DB S/W communication place the jdbc code in Jsp prg.
- ⇒ JSP / servlet prg can get non-technical inputs from end user or form data, can get technical inputs (like jdbc driver details) from web.xml either in init param or context param.
- ⇒ To read init param from web.xml we config obj and read context param from web.xml we application obj.
- ⇒ config, application..etc implicit obj are not visible in declaration tag becoz they are local to `JspService()` of JES class. But we can access those objects in declaration tag by writing the servlet api code explicitly. ~~init~~ Init(..) parameters place in JSP configuration in web.xml can be accessed only when the Jsp prg is requested with url pattern.

Example

ABC.jsp

```
<%! public void jspInit() {  
    ServletConfig cg = getServletConfig(); // servlet api code  
    System.out.println("Config obj class name:" + cg.getClass());  
    System.out.println("P1 init param value:" + cg.getInitParameter("P1"));  
    // given val  
}%>
```

web.xml

< servlet >

< s-n > xyz </ s-n >

< jsp-file > ABC.jsp </ jsp-file >

< init-param >

< param-name > p1 </ param-name >

< param-value > val1 </ param-value >

</ init-param >

</ servlet >

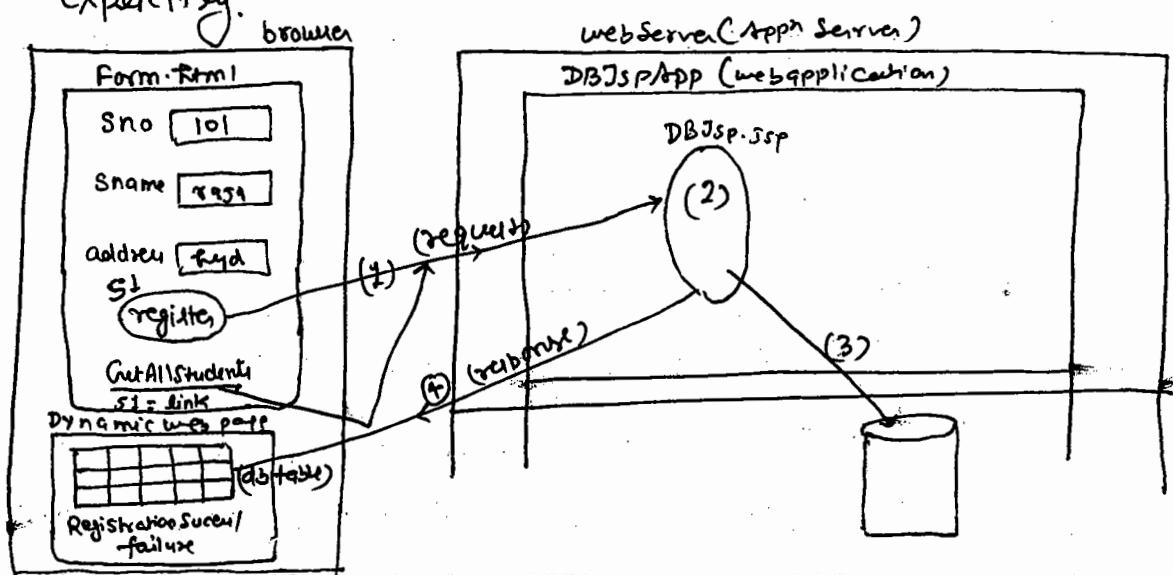
< servlet-mapping >

< s-n > xyz </ s-n >

< u-p > /test1 </ u-p >

</ servlet-mapping >

⇒ For the code placed in scriptlet, expression tag there is no need of exception handling becoz this code goes to `- jspService(-,-)` method and takes care of exception handling automatically. But for the code placed in declaration tag we must take care of exception handing explicitly.



DBJSpApp

---> WEB-INF

form.html

Page

DBJSp.jsp

web.xml

lib

ojdbc14.jar

sequetURL: <http://localhost:3030/DBJSpApp/form.htm>

⇒ Use all the 3 life cycle convenience methods.

⇒ Rather JDBC properties from web.xml file or init param values.

⇒ Don't place `out.print()` in entire Jsp program.

Note For above diagram based App refer App of the page no 264 to 267

for various fundamentals of Jsp refer page no 250 to 258.

⑧
11/05/15

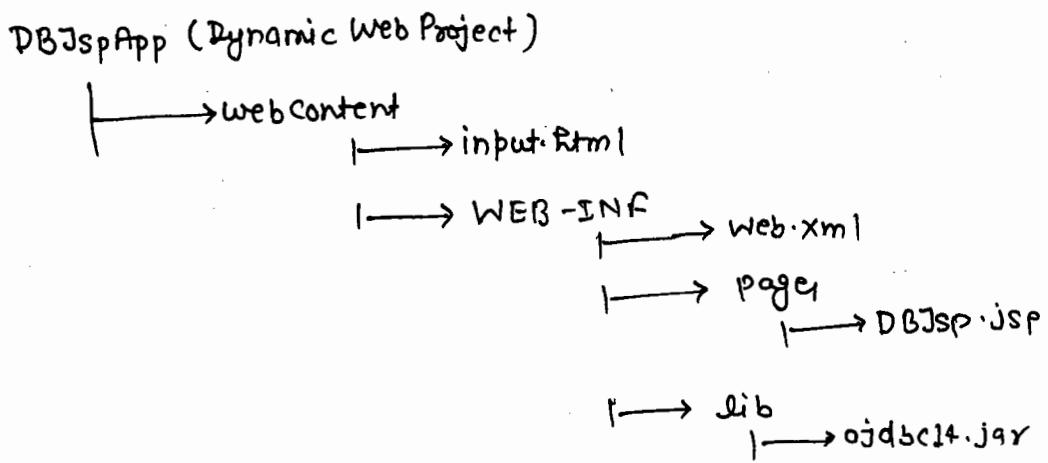
Comment

267 to 282 → JSP init parameters having JDBC properties

304 to 307 → refer web.xml for init parameter name,

352 to 372 → logic to display the records of ResultSet in the form of HTML table content.

⇒ In real time no one hard code JDBC property value in Servlet & JSP pg. They get it from web.xml file either as init parameters or as Context parameters to achieve the flexibility of modification.



Directive Tags

⇒ These tags are given to give directions to jsp page compiler.

⇒ Syn: <%@ Directive Tag attribute %>

⇒ 3 directive tags -

- Page directive
- Include directive
- Taglib directive

a) Page Directive

⇒ Useful to provide Global info for jsp page

Standard Syn:-

<%@ page attribute %>

xml Syn:-

<jsp:directive.page attribute %>

attribute

language	scripting
import	errorPage
extends	isErrorPage
contentType	isThreadSafe
buffer	info
autoFlush	iELIgnored

1) language:-

- Allows specify the language that can be used to write script code in scripting tags of jsp page.
- "Java" is the only one value and default value that is allowed here

```
<%@ page language="java" %> (valid)
```

```
language="c" %> (invalid)
```

```
language="c++" %> (invalid)
```

2) info:-

```
<%@ info = "Customer Registration Page" %>
```

```
<%@ page info="Customer Registration Page" %>
<b>Hello </b>
```

- No default value
- allow to specify short description about Jsp page
- In JES class getServletInfo() method will be overridden having this short desc

```
public String getServletInfo() {
    return "Customer Registration Page";
}
```

3) session:-

- default value is true.
- When session tracking is not required, there is no need of "Session" obj in our JSP pg.

```
session="false"
```

make JES not to create the implicit obj "session."

```
<%@ page session="false" %> <!-- implicit obj session will not be created -->
```

`<%@ page session="true" %> <!-- implicit obj session will be created -->`

4) import:-

- No default value
- Allow to import pkgs (.1 or more)
- `<%@ page import = "java.sql.*; javax.*" %>`

5) contentType:-

- default value: text/html
- Allows to specify response content type to generate dynamic content.

e.g 1: `<%@ page contentType = "text/plain" %>`

e.g 2: `<%@ page contentType = "application/msword" %>`

12/05/15 → Holiday

(9)

13/05/15

6) extends:-

- ⇒ Allows to specify custom class as super class of JES class.
- ⇒ This custom class must extends from HttpServlet(c) and must implement JspPage(I). Since doing all these things is very complex we never attempt to form perform.
- ⇒ In Tomcat server the default value for this attribute is "HttpJspBase" which implements JspPage(I) and extends from HttpServlet (A C).

e.g. `<%@ page extends = "myClass" %>`

Here myClass becomes super class for JES class. For this myClass should extend from HttpServlet (A C) and should implement JspPage (I).

7) buffer:-

- Allows to specify the size of buffer that is associated with "out" object.
- The default size is "8kb."
- When Server executes Jsp prg code part by part in scheduling the o/p will be stored in the buffer until last part o/p is generated. Later this o/p goes to browser as response.

e.g1: <%@ page buffer = "8kb" %>

e.g2: <%@ page buffer = "10kb" %>

e.g3: <%@ page buffer = "none" %>

Q) what is the difference b/w PrintWriter and JspWriter ?

A) (Servlet prg) (type of implicit obj out)

PrintWriter can't work with buffering whereas JspWriter can work with buffering.

JspWriter internally uses PrintWriter when buffering is not required. (buffer = "none").

The print() of JspWriter throw IOException whereas

print() of PrintWriter doesn't throw IOException.

8) autoFlush:-

- This attribute allows boolean value (false/true)
- flushing means sending the content of buffer to browser window.
- autoflushing means sending the content to the buffer to browser window when buffer is filled or when JSP prg execution is completed.

⇒ autoflush = "true" ----> enables auto flushing

⇒ autoflush = "false" ----> disables auto flushing

e.g.1

```
<%@ page buffer = "10kb" autoflush = "true" %>
```

e.g.2

```
<%@ page buffer = "3kb" autoflush = "flush" %>
```

if JSP prog generates more than 3kb data then exception will be raised.

(Java.io.IOException: Error: Jsp Buffer Overflow)

```
<% for (int i=1; i<=100000; i++) {  
    out.println("Hello");  
}>
```

If autoflush = "true" → Then exception will not be raised

e.g.3

```
<%@ page buffer = "none" autoflush = "false" %>
```

gives error as "bad combo."

Enabling or Disabling of buffer when buffer size is '0' (zero) gives error.

Q) JSP prog buffer size is 3kb and it is generating more than 3kb O/P. Can u tell me what happens?

Ans) if autoflush = "false"
(buffer overflow error comes)

if also autoflush = "true"

the JSP prog comes in browser part by part through the flushing separate on the buffer

Q) What is the difference b/w implicit object page and pageContext obj?

Ans) → The implicit obj page holds "this" (nothing but ref of JES class obj).

→ The implicit obj pageContext holds multiple obj's detail of Jsp page like req, res, buffer size, autoflush mode session mode, error page and etc--.

→ We can use pageContext to access other implicit objs of jsp.

We can use PageContext to create page, request, session, application scope attributes.

In JES class

Object page = this;

pageContext = _jspxFactory.getPageContext(this,
request, response, "err.jsp", false,
3072, true); error
page session
mode
buffer
size auto
flush
mode

Q) What is the difference b/w ServletContext obj and PageContext obj?

Ans) → pageContext obj is one per Jsp page and it can be used to create page, session, request, application scope attributes.

⇒ ServletContext obj is one per web application and it can be used only to application scope attribute.

9) isThreadSafe :-

⇒ If it is a boolean attribute in whose default value is "true."

⇒ if multiple threads are acting on 1 obj our servlet class / JES class ^{concurrently that} then ~~then~~ ^{the} servlet or Jsp is not thread safe

<!--@ page isThreadSafe = "false" -->

⇒ Make JES class to implement "SingleThreadModel(I)" and let container to take care thread safety
(Allows only 1 request at a time to jsp prg)

<!--@ page isThreadSafe = "true" -->

⇒ Make JES class not implement "SingleThreadModel(I)" Due to this programmer should explicitly take care of thread safety using synchronization concept.

⇒ Here jsp prg allows multiple threads / requests simultaneously.

10) isELIgnored :-

⇒ Keeping java code in jsp is not industry standard so to perform Arithmetic and logical expression without java code we can use expression language.

(\${expr})

e.g. \${10+2}

e.g:- <!--@ page isELIgnored = "false" -->

---> Recognizes EL Syntax placed in jsp Prg

e.g:- <!--@ page isELIgnored = "true" -->

---> Ignore EL Syntax placed in jsp Prg

Important points about page Directive:-

- 1) Invalid attribute will not be allowed.
- 2) attributes are case-sensitive.
- 3) Except "import" attribute no other attribute can be repeated for multiple times with diff values either same <%@page%> or diff <%@page%> tags.
- 4) Except "import" attribute no other attribute allows to have multiple values separated with "," symbol.

(10)
14/05/15

Exception Handling in Jsp:-

- For the code placed in declaration tag methods we need to handle exceptions explicitly by using try/catch blocks
- For the code placed in Scriptlets/expression tags the JspService(-,-) automatically handles the exception.

Error pages in Jsp:-

- The Jsp/html page that executes only when exceptions are raised in other jsp page is called error page. This page is useful to display non-technical guiding messages to end user when exceptions are raised.
- Use case: Displaying "Exception related technical message" when submit credit card details gives problem to end user to understand the message.

By error page cfg support we can display same message or non-technical guiding messages.

⇒ In Jsp Error page cfg can be done in 2 ways—

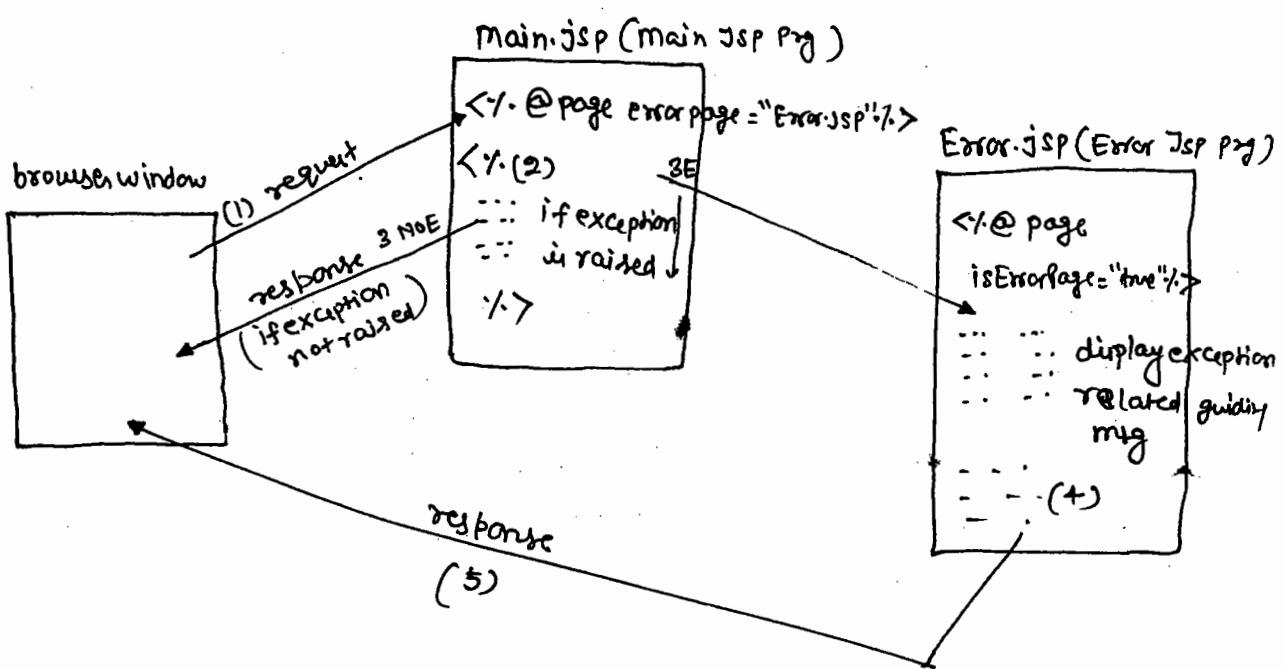
a) Local Error Page cfg (specific to each Jsp Pg)

(use `<error-page>`, `isErrorPage` attribute of `<%@ Page %>`)

b) Global Error Page cfg (common for all jsp pg's)

(use `<error-page>` tag of `web.xml`)

Local Error Page cfg



⇒ The implicit obj exception is visible only in error.jsp pg and this obj can be used to display execution related guiding messages

Jsp App2

----> WEB-INF
----> web.xml

----> Error.jsp

----> Main.jsp

main.jsp

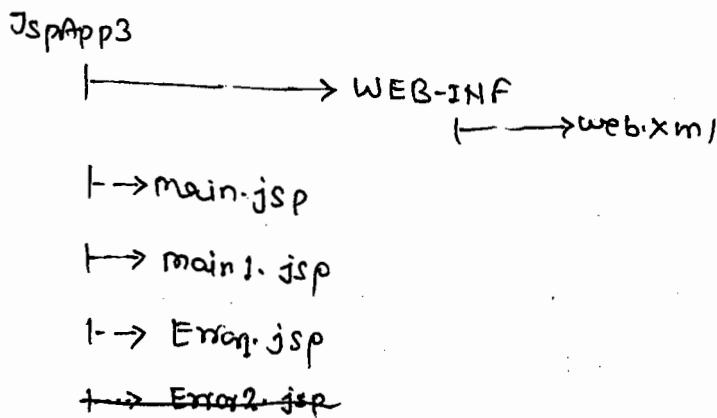
```
<%@ page errorPage = "Error.jsp"%>
<% int x = Integer.parseInt("10"); %>
value is <%= x %>
```

Error.jsp

```
<%@ page isErrorPage = "true"%>
<b> internal problem </b>
<br>
<hr>
<%= exception.toString()%>
```

request url: `http://localhost:8080/JspApp/main.jsp`

Example on Global Error Page cfg



main.jsp

```
<% int x = Integer.parseInt("120"); %>
value is <%= x %>
```

main1.jsp

```
<%@ page import = "jquery.util.*" %>
<% Date d = null; %>
```

Current Month <%= d.getMonth() + 1 %>

Error.jsp

```
<%@ page isErrorPage = "true" %>
<b> internal problem resource </b>
<hr>
<% = exception.getMessage() %>
```

web.xml

```
<web-app>
  <error-page>
    <exception-type> java.lang.Exception </exception-type>
    <location> /Error.jsp </location>
  </error-page>
</web-app>
```

request urls: <http://localhost:8080/JspApp3/main.jsp>
[/main1.jsp](http://localhost:8080/JspApp3/main1.jsp)

- ⇒ if we apply both local and global error page cfgs on single jsp prg then the cfgs done in local error page cfg takes place.
- ⇒ we can take html page or error page, but we can't work with "implicit obj" "exception."
- ⇒ The above error page cfgs does not respond for the exceptions raised in servlet comp.

Directive include:- <%@include%>

Syntandard syn:-

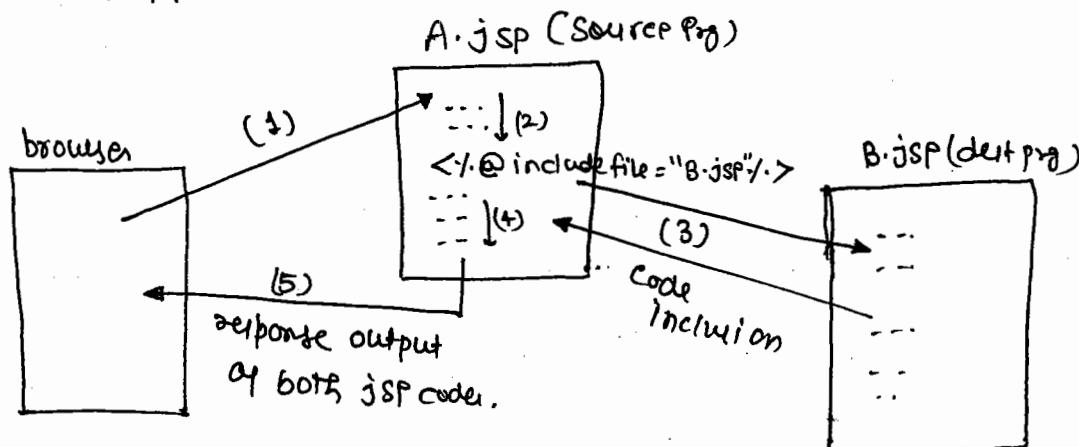
<%@include file = "<file name>"%>

Xml syntax:-

<jsp: directive.include file = "<file name>"%>

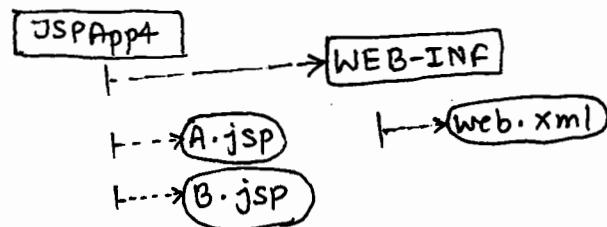
⇒ Given to include the code of Det prog / file to the code of source jsp prog's JES class

⇒ This tag is given for code inclusion not for output inclusion.



⇒ Here Separate JES class will not be generated for B.jsp, But the code of B.jsp will be included to the JES class of A.jsp.

⇒ This method does not use rd. include (-,-) method internally



11
15/05/15

A.jsp

```
<b> from the start of A.jsp </b>
<br>
<%@ include file="B.jsp" %>
<br>
<b> from the end of A.jsp </b>
```

B.jsp

```
<b> from the B.jsp </b>
```

request url: `http://localhost:3030/JspApp4/A.jsp`

→ The JES class of A.jsp nothing but (A.jsp.java) contains code of A.jsp and code of B.jsp

Action tags:-

These tags internally used functionalities of servlet-api to perform operations directly at execution phase or request processing phase.

There are two types of action tags

Standard Action tags: Built-in tags of jsp

e.g. `<jsp:include>`, `<jsp:forward>` and etc..

Custom Action tags: Developed by programmers

Note: All action tags give only xml tags.

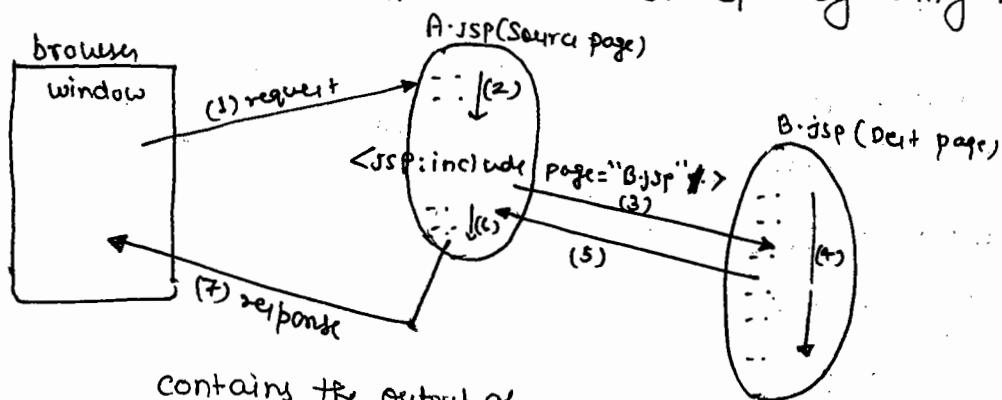
Action include :- (<jsp:include>)

Syntax

<jsp:include page = "..."/>

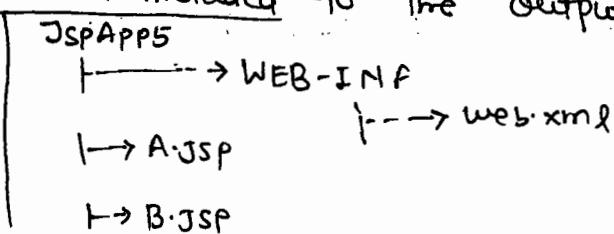
⇒ Given to include the o/p Dest web resource prg to o/p source jsp prg by rd.include(-,-) internally.

⇒ Generate separate JES class for dest jsp page and includes that o/p in Source Jsp by using rd.include(-,-)



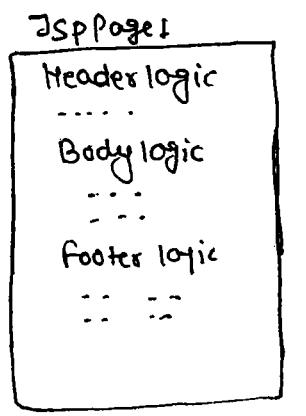
contains the output of
both A.jsp and B.jsp

⇒ It performs output inclusion. Here we get two separate JES classes and more even the output of Dest JES class (B.jsp) will included to the output of Source JES class (A.jsp)

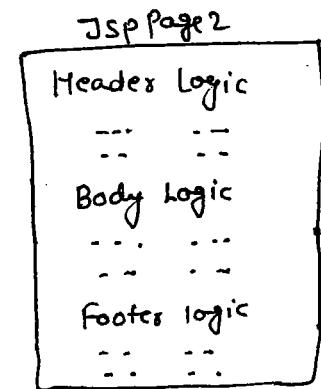


Directive include	Action include
1. Given for code inclusion at translation stage.	Given for o/p inclusion at execution phase.
2. Performs compile time / static binding.	Performs runtime / dynamic binding.
3. Doesn't use <%@include(-,-) internally	use
4. Gives both XML and standard syntax.	only XML syntax.
5. Doesn't allow to take Servlet prg or dest prg	Allows
6. Doesn't generate JES class dest prg (JSP prg)	Generates
7. Useful if dest prg is static program like RDBMS	useful if the destination program is dynamic program

Problem



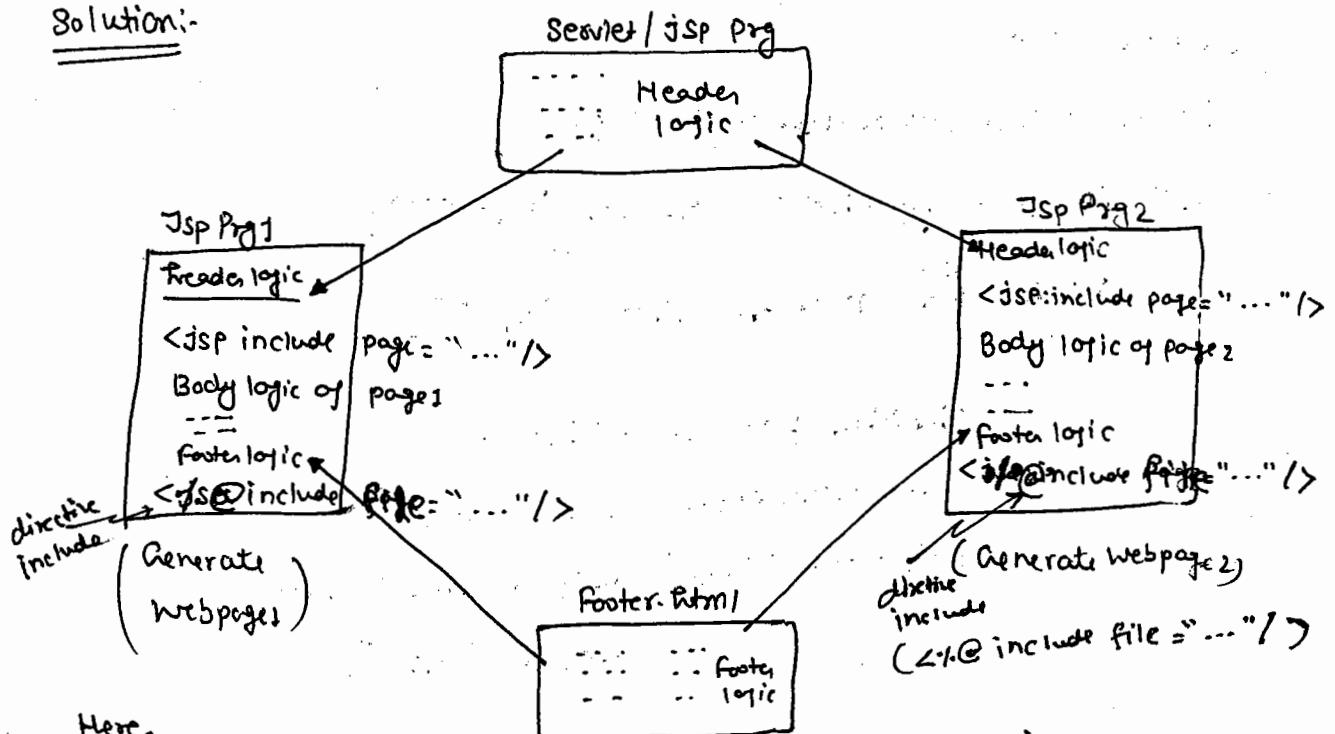
(Generate Webpage1)



(Generate Webpage2)

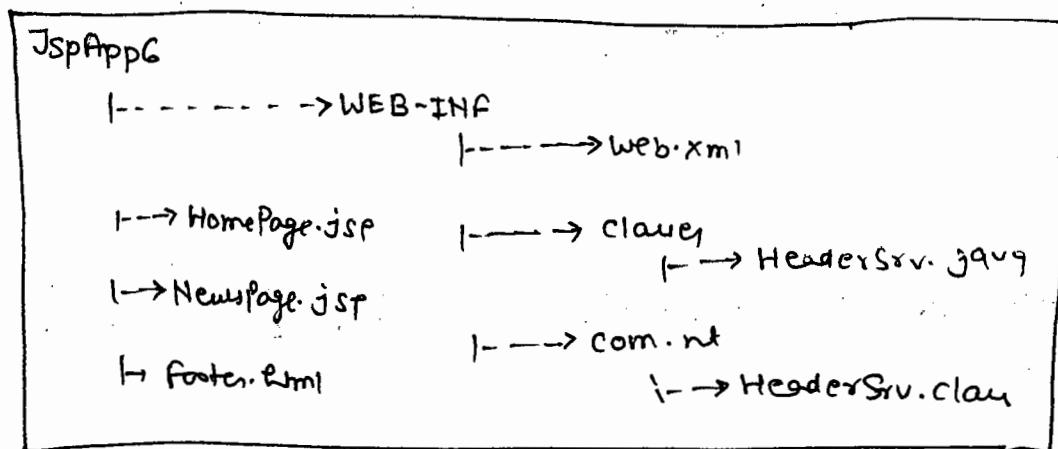
All pages of web application contains same header & footer content. Here our header & footer logic are not reusable.

Solution:-



Here,

Header and Footer logics are reusable.



JspApp6 (Eclipse)

→ java resource

→ src

→ com.nt

→ HeaderSrv.java

→ webcontent

→ HomePage.jsp

→ NewsPage.jsp

→ footer.html

```

package com.nt;

import javax.servlet.http.HttpServlet;

public class HeaderSrv extends HttpServlet {
    public void doGet(..) throws SE, IOE {
        PrintWriter pw = res.getWriter();
        res.setContentType ("text/html");
        pw.println ("<h1> <marque> HCL Technologies
                    </marque></h1>");
    }

    public void doPost(..) throws SE, IOE {
        doGet (req, res);
    }
}

```

Web.xml

Configure com.nt.Header Srv with /head 481.

Footer.html

```
<center><i> @Copy all rights reserved </i> </center>
```

Homepage.jsp

```

<table width = "100%" > height = "100%" >
    <tr height = "30%" >
        <td> <jsp:include page = "header" /> </td>
    </tr>
    <tr height = "60%" >
        <td> <h4> welcome to TVG.net </h4> </td>
        <br> (Breaking News)
    </tr> <br> IPL match: MS won match against KKR
          with support pollard's excellent last over.

```

```

<tr height="10%>
  <td> @include file="footer.html" /> </td>
</tr>
</table>

```

12
16/05/15

⇒ We can place multiple directive include and multiple action include in one.jsp prg and the output all dest prg will be included the output of source page.

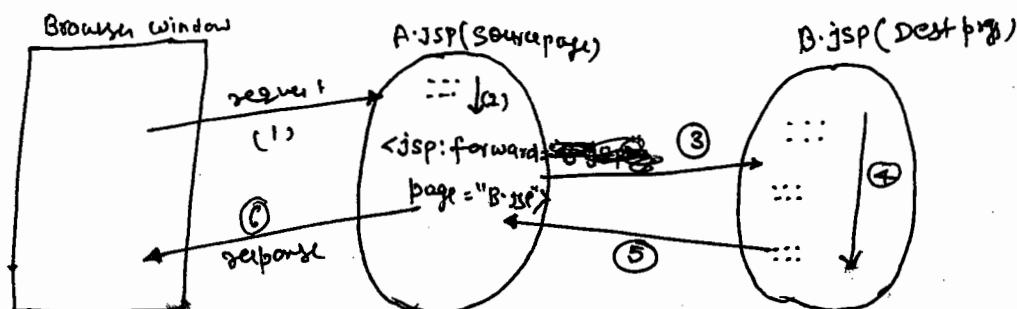
Action forward (<jsp:forward>)

⇒ Given to forward the request from source jsp prg to dest jsp prg. In this process the output of source jsp prg will be discarded and only the output of dest prg goes to browser as response.

Syntax:

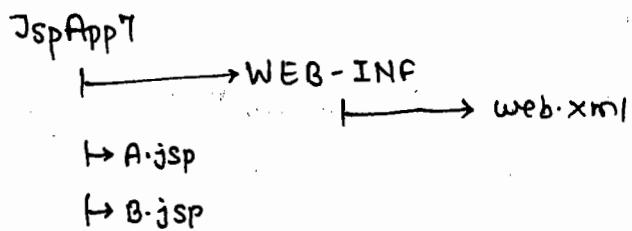
```
<jsp:forward page="--" />
```

Note ⇒ There is no directive forward tag.



{Contains only the output of dest program.
The output of source program will be discarded.}

- ⇒ It internally uses rd.forward (-,-) method.
- ⇒ The difference b/w rd.forward (-,-) and <jsp:forward> tag in jsp programming is statement placed after rd.forward (-,-) method in source servlet prg executes, but the statements placed after <jsp:forward> tag in source jsp prg doesn't execute.



A.jsp

```

<b> <i> form A.jsp </i> </b>
<jsp:forward page = "B.jsp" />
<b> <i> end A.jsp </i> </b>

```

B.jsp

```

<b> <i> form B.jsp </i> </b>

```

<jsp:param>

⇒ This tag is given to pass data from source.jsp page to dest.jsp page or additional request param values. This tag can be only used as the sub tag of `<jsp:forward>` or `<jsp:include>` tags.

Example

A.jsp

```

<% float PI = 3.14f %>
<jsp:forward page = "B.jsp">
<jsp:param name = "p1" value = "10" />
<jsp:param name = "p2" value = "<% = PI %>" />
</jsp:forward>

```

B.jsp

<i> from B.jsp </i>

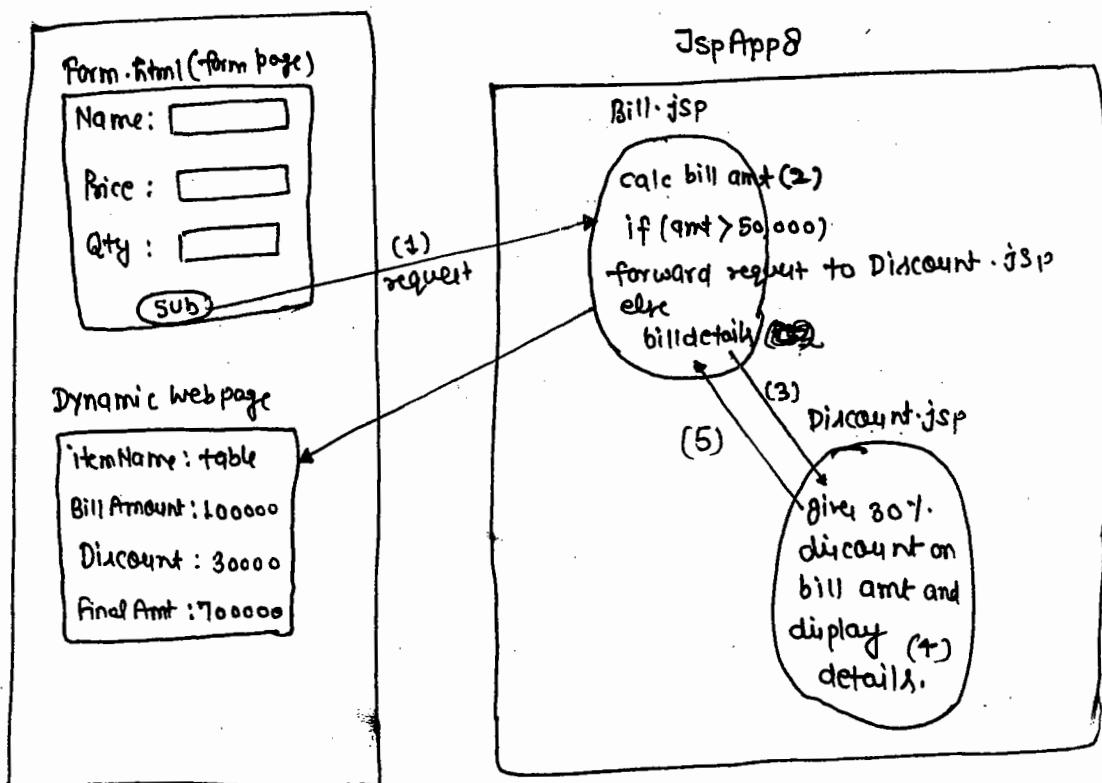
Additional request param value: <% = request.getParameter("p1")%>
<% = request.getParameter("p2")%>

⇒ It is always recommended to place <jsp:forward> tag in our source jsp pg our conditional statement place because there is no meaning of discarding the output of source page every time.

⇒ <jsp:forward>, <jsp:include> tags are given to make source jsp pg communication with dest pg, so this is called jsp communication. The two mode of communication is —

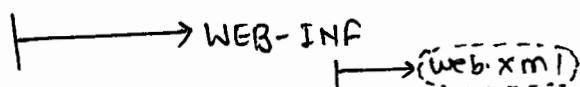
- a) forwarding request (using <jsp:forward>)
- b) including response mode (using <jsp:include>)

⇒ In the above communication the source page and dest page is the same req, res Obj, so the form data coming to source page is visible and accessible in dest page.



→ Bill.jsp passes the calculated bill amount to Discount.jsp by using <jsp: param> tag.

JspApp8



→ form.html

→ Bill.jsp

→ Discount.jsp

refer Application given in page no 273.

(13)
18/05/15

⇒ There is no directive forward tag becoz directive tags perform their operations at translation phase, in this process we can't discard the old source jsp page. For this reason there is no directive forward tag.

⇒ <jsp:include>, <jsp:forward> tags are useful for jsp communication / jsp chaining.

In this process the source jsp page and dest jsp share same request / response obj

Send Redirection in Jsp:-

⇒ There is no separate jsp tag to perform this operation we need to call response.sendRedirect(-) method explicitly to perform sendRedirection.

In source jsp prg

<% response.sendRedirect("http://www.google.ca.in"); %>

Attributes in jsp Prg:-

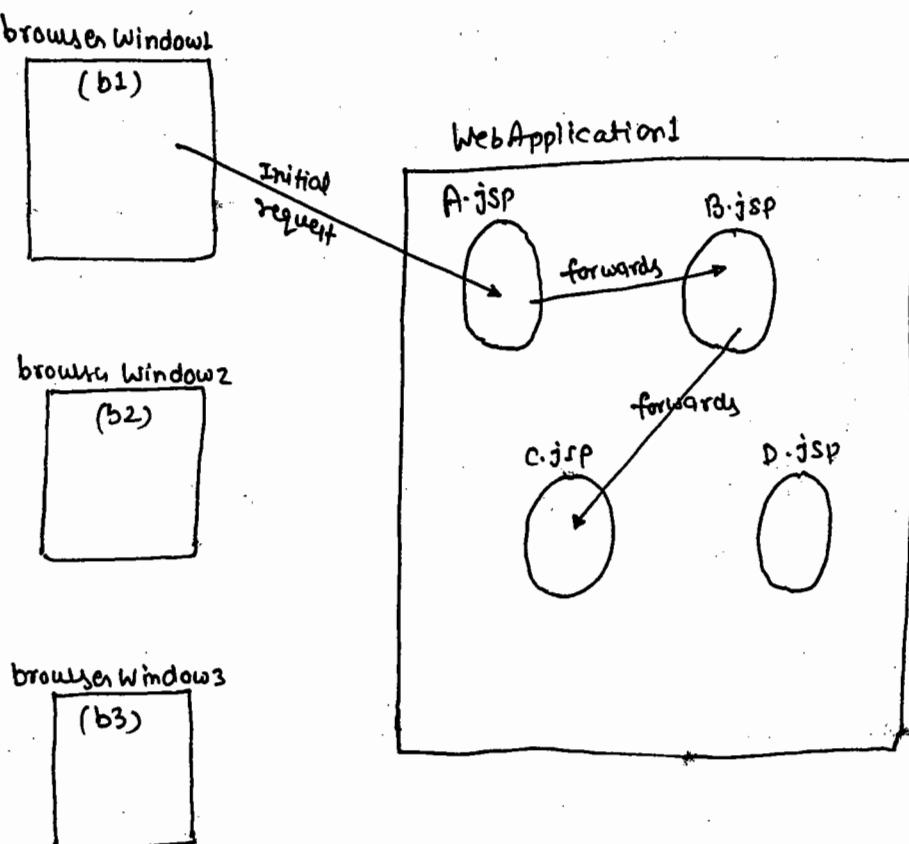
⇒ Attribute is a logical name that holds value. It is not related with tag attributes.

⇒ To pass data b/w various resources of web application we need to work with attributes.

⇒ In jsp we can have 4 types of attributes -

- a) request attribute (request scope) (Through out request)
- b) session attribute (session scope) (Within web appn specific to browser)
- c) Application Attribute (application scope) (Within web appn not specific to browser)
- d) pageContextAttribute

(allow to create request, session, application, page scope attributes)



- ⇒ request attribute created in A.jsp is visible and accessible in B.jsp, C.jsp but not in D.jsp because A.jsp, B.jsp, C.jsp use same request obj.
- ⇒ Session attribute created in A.jsp by getting request from browser window (b1) is visible and accessible in all other JSP pages but they should also get request from same browser window (b1).
- ⇒ Application attribute created in A.jsp is visible and accessible in all other JSP pages irrespective any conditions.
- ⇒ request obj allows to create only request scope attributes.
- ⇒ session obj allows to create only session scope attributes.
- ⇒ application obj allows to create only application scope attributes.
- ⇒ PageContext obj can be used to create request, session, application and pageContext attributes.

Page Context attribute:-

To create pageContext attribute:-

pageContext.setAttribute("age", 30); → creates page scope attribute

pageContext.setAttribute("name", "Raja", pageContext.REQUEST_SCOPE);
→ request scope attribute

Other scope:-

pageContext.SESSION_SCOPE

pageContext.APPLICATION_SCOPE

pageContext.PAGE_SCOPE

To modify pageContext attribute value:-

pageContext.setAttribute("age", 40); // modify page scope attribute

pageContext.setAttribute("name", "King", pageContext.REQUEST_SCOPE);
(modifies the request scope attribute)

To read pageContext attribute value:-

int age = (Integer) pageContext.getAttribute("age"); // gives page
String name = (String) pageContext.getAttribute("name");
(gives request scope attribute value)

To find page PageContext attribute:-

int age = (Integer) pageContext.findAttribute("age");

Q) What is the difference b/w findAttribute() and getAttribute()?

A) getAttribute() method searches for given attribute in the specified scope.

findAttribute() method searches for the given attribute in a multiple scopes in a order (page scope → request scope → session scope → application scope).

To remove pageContext attribute:-

pageContext.removeAttribute("age");

pageContext.removeAttribute("name", pageContext.REQUEST_SCOPE);

⇒ To provide global visibility to scriptlet tag generated data we can assign that data to instance variable or we can keep that data in page scope pageContext attribute.

Example App:-

JspApp9

-----> WEB-INF
-----> web.xml

→ A.jsp

→ B.jsp

→ C.jsp

→ D.jsp

refer App given in page no 272 & 273

- ⇒ if we place multiple <jsp:forward> tags in one source page the first forward will be applied.
- ⇒ if we place multiple <jsp:include> tags in one jsp pg all of them will be applied.
- ⇒ if we place both <jsp:include>, <jsp:forward> tags in one jsp pg the effect of <jsp:include> will not be there.

<jsp: plugin>

- ⇒ This tag is given to display Applet/GUI java bean on the browser window.
- ⇒ If the logo of website is designed as applet having animation support through threads then we need to work with <jsp: plugin> tag
- ⇒ This internally uses <object> / <embedded> / <applet> tag to display the applet on browser.
- ⇒ <jsp:fallback> is the subtag of <jsp: plugin> tag that is capable of displaying error message when browser doesn't support applets.
- ⇒ <jsp:param>, <jsp: param> tags can be used as sub tag of <jsp: plugin> tag to send data from JSP prg to Applet prg..

<jsp: plugin> tag attributes

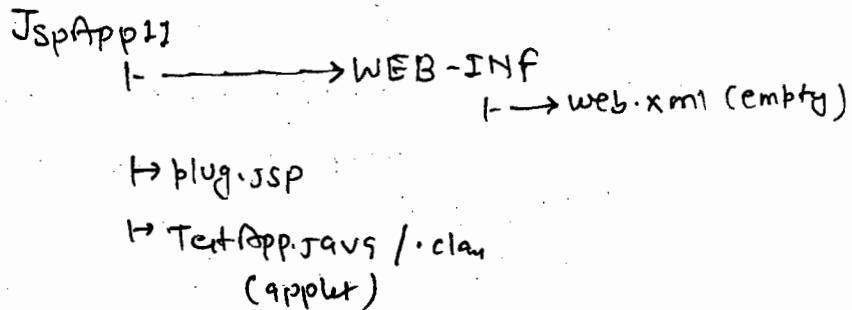
type = "applet/bean"

code = "The class is acting as applet/ Bean"

codeBase = "the directory where the Applet/Bean" is available.

Example

Applets of page no 269



request url : http://localhost:3030/JspApp1/plug.jsp

⇒ if want to pass data to Applet from jsp pgm we can use <jsp:params>, <jsp:param> tag as shown below-

In plug.jsp

```
<jsp:plugin type="applet"  
            code="TextApp.class"  
            codebase="/JspApp1" width="300" height="300">  
  
<jsp:params>  
    <jsp:param name="organization" value="NIT"/>  
</jsp:params>  
</jsp:plugin>
```

In TextApp.java

```
in paint(Graphics g) {
```

```
    ...  
    String s1 = getParameter(  
    ...  
}
```

Q) What is Java Bean?

A) Bean means Component. Component means reusable object.

Java Bean is a java class that is developed with set of standards. The standards are -

- a) class should be public & must not be final, abstract
- b) recommended to implement Serializable(z)
- c) Must have direct/indirect -o param constructor
- d) Member variables should be non-static & private.
(bean properties.)
- e) Every bean property should have 1 setter method
& 1 getter method with public modifier

- setter method is useful to set data to bean property.
- getter method is useful to read data from bean property.

Ex-

```
public class StudentBean implements Serializable {
```

// Bean Properties

```
private int sno;
```

```
private String sname;
```

```
public StudentBean() {
```

```
}
```

```
public void setSno(int sno) {
```

```
    this.sno = sno;
```

```
}
```

```
public int getSno() {
```

```
    return sno;
```

```
public void setSname(String sname) {
```

```
    this.sname = sname;
```

```
}
```

```
public void getName() {
```

```
    return sname;
```

```
}
```

- ⇒ Java Bean is useful to combine multiple simple values to an object and to send that object from one resource to another resource or one layer to another layer
 (e.g. To keep form data into a single obj we can use java bean)

Q) What is the difference b/w Java Bean & EJB?

	JavaBean	EJB
1)	Lightweight class	Heavyweight Comp
2)	An ordinary class with standards	A distributed technology
3)	Allows only local clients.	Allows both local & remote clients
4)	Needs JRE for execution	Needs EJB container for execution.

- ⇒ For Servlet to Java Bean communication the servlet program has to create obj for javabean and should call setter & getter methods manually.
- ⇒ For Jsp to Java Bean communication we can use 3 tags
- <jsp:useBean> ---> To create or locate Java Bean class obj
 - <jsp: setProperty> → To call setXXX(.) & to write data to bean properties,
 - <jsp: getProperty> → To call getXXX() & read data from bean properties.

For info about above 3 tags along with Application
refer pages no 255 to 257 & App 7 of pages no 270 - 271

JspApp19

→ WEB-INF

→ SetValue.jsp → class
 → StudentBean.java

→ GetValue.jsp → com.nt
 → StudentBean.class

- In real time formData will be given to service class or DAO class by Java Bean class obj through Jsp using <jsp:useBean>, <jsp:setProperty>

→ Similarly the results generated by service class or DAO class come to JSP in the form of Java bean class obj. To display obj data we can use `<jsp:usebean>`, `<jsp:getProperty>`

15
21/05/15

The web resource program of web application contains following logic—

- 1) request data gathering logic
(reading form data, reading request param, reading request header)
- 2) Form validation logic
(Verifying the pattern & format of the form data)
- 3) Business logic / Service logic
(The logic that process request & gathering the result)
- 4) Persistence logic
(CURD operation on DB & file)
- 5) Session tracking logic
(The logic that session remember clients data across the multiple request during a session)
- 6) Persistence logic
(The logic that gives user interface to end user)
- 7) Middleware Service
(Security, logging, and etc...)

Different Model / Architecture of java web application

Model 1 Architecture

- * Here either Servlet / Jsp program will be used as main web resource program of the web application.
- * Every main web resource program contains multiple logic without separation.
- * The Application that are developing comes under "Model 1" architecture application

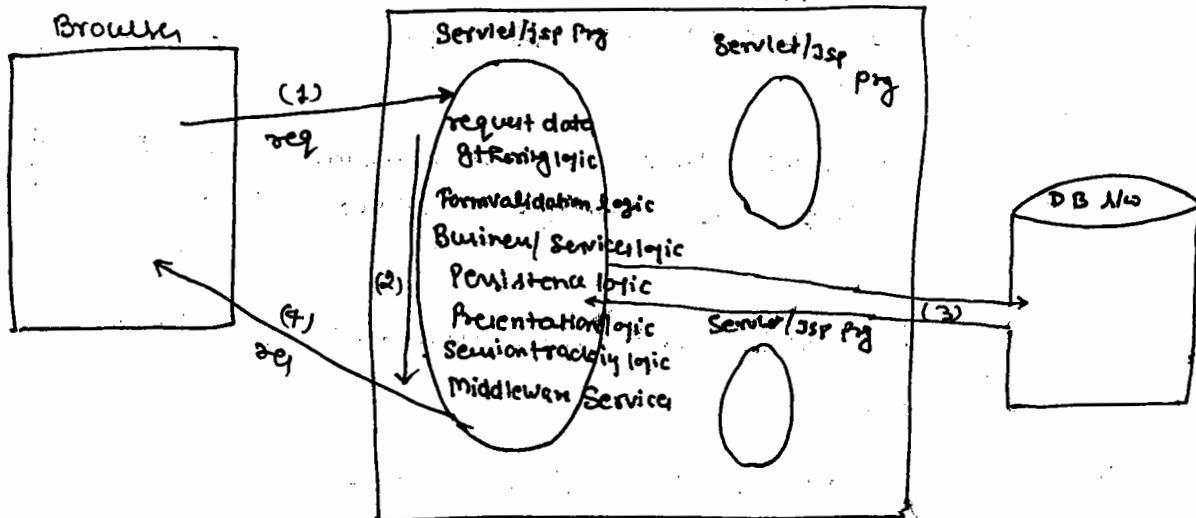
Model 2 Architecture

MVC1 MVC2

- * Here multiple layers will be there and logics of these multiple layer will be developed by using multiple technology.

Model 1 Architecture

Model 1 Architecture based Application



- * In Model 1 mainly Jsp prog will be used as main web resource program.

Disadvantages of Model 1:-

- ⇒ Here we keep multiple logics in every main web resource program. So there is no clear separation b/w logics.
- ⇒ The modification done in one kind of logic will effect other logic.
- ⇒ The maintenance and enhancement of project becomes complex.
- ⇒ Parallel development is not possible, so productivity is very poor.
- ⇒ Some middleware service must be implemented by programmers manually. This gives burden to programmers.

Advantages:-

- ⇒ Having less no. of programmers the project can be developed.
- ⇒ Knowledge on either servlet/JSP is enough to develop the project.

Note:- Model 1 Architecture is good for developing small scale website (< 10 pages)

— : MVC : —

M → Model layer → Data + B. logic + Persistence logic (like Algo's)

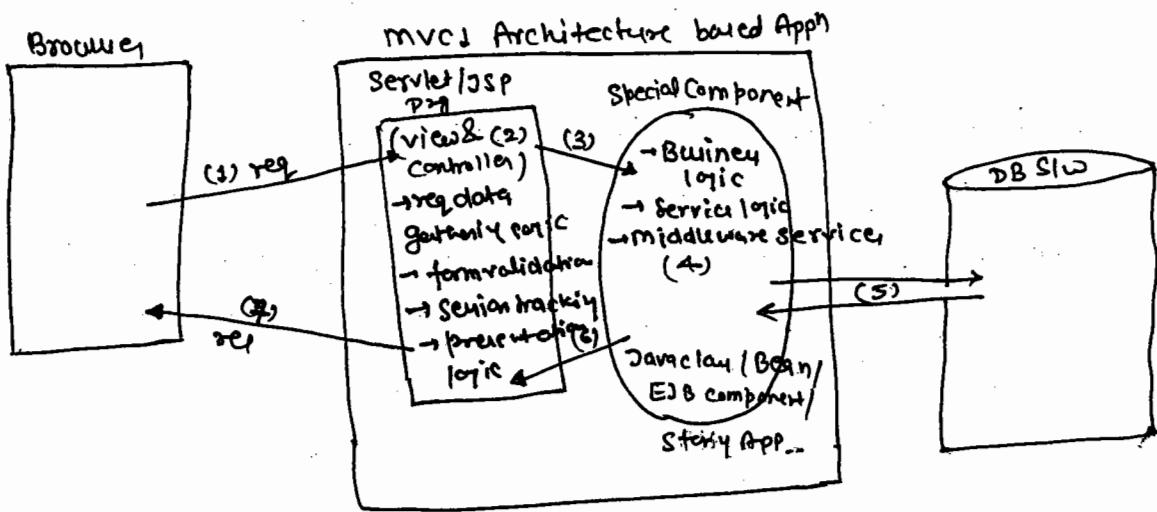
V → View layer → Presentation logic (It is like beautician)

C → Controller → Integration logic (It is like traffic police)

⇒ It is the logic that monitors & controls all and monitor all the operation/ activity.

⇒ It is the logic that gives interaction b/w view layer and model layer resources.

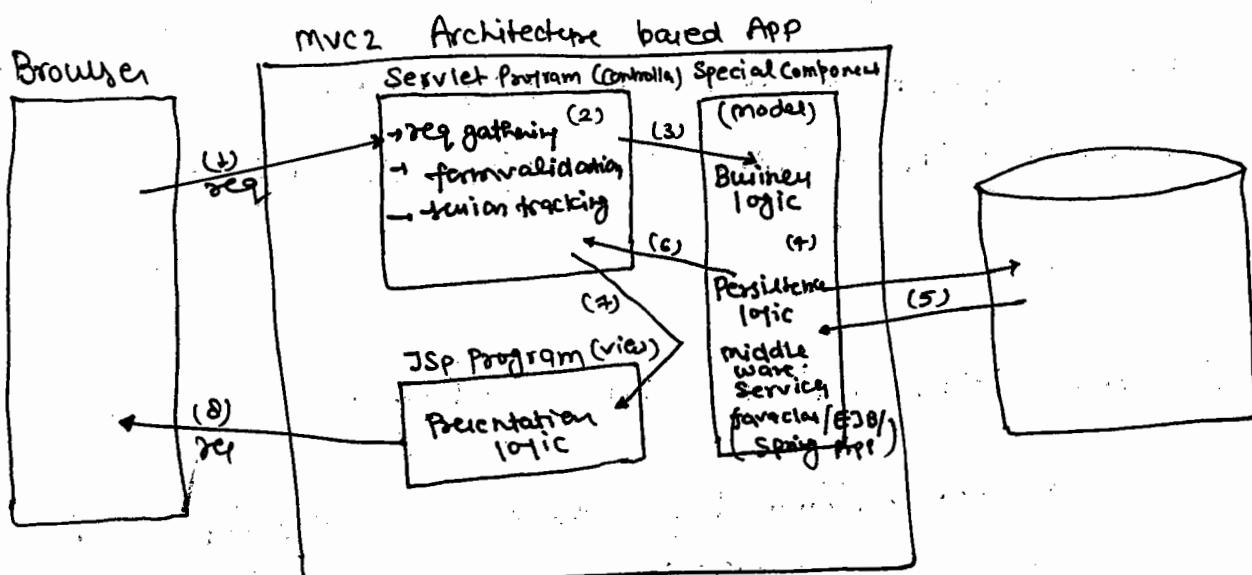
- ⇒ In model1 architecture we take single resource having view, controller, layer logic and we take separate resource having model layer logic.
- ⇒ In MVC2 architecture we take separate resource for view layer, separate resource for controller layer and separate resource for model layer.
- ⇒ Html & Jsp program will be use as view layer resource, servlet/filter will be use as controller layer resource.
- ⇒ Java Bean/ Java class/ EJB Components /Spring App / Spring with hibernate will be used as model layer resources.



- ⇒ Compare to model1 architecture MVC2 give clear separation b/w logic using layer support. If you wan to get more clean separation b/w logic then use MVC2 architecture.

Note:- use MVC2 architecture for medium and small scale App below 25 pages

MVC2 Architecture



⑯
22/05/15

Advantages:-

- ⇒ Gives multiple layers due to this we can get clean separation b/w logics.
- ⇒ The modification done in one layer logics does not effect other layer logics.
- ⇒ The maintenance & enhancement of the project becomes easy.
- ⇒ Parallel development is possible, so productivity will be good.
- ⇒ It is industry defacto standard to develop the web app?
- ⇒ if we can use EJB, Spring in Model we can get the benefits of working with built-in middleware services.

Disadvantages:-

- ⇒ For development we need more programmers
- ⇒ Knowledge in multiple technologies is required.
- ⇒

Q) How can we do parallel development?

A) The P.L. divides the team into two parts -

Part 1 Web Authors (Take the support of Servlet, JSP technologies and develop view, controller layer logic.)

Part 2 JEE Developers (Take the support of EJB, Spring, Hibernate and etc... to develop model layer logic.)

Since Two parties (part1, part2) can work parallelly we can say parallel development is possible.

MVC2 rules/ principles:-

- ⇒ Every layer is designed to have specific logics, so always place only those logics and don't place additional logics.
- ⇒ All operations of web app must take place under the monitoring or controller servlet.
- ⇒ The controller servlet must be ready to trap and take all the requests, for that it must be cfg either with extension match or directory match url pattern.
- ⇒ There can be multiple resources in view layer, multiple resources in model layer but there must only one servlet

Getting at Servlet Controller.

- ⇒ The two resource view layer must not communicate with each other directly they must communicate through controller servlet.
- ⇒ The view layer resources and model layer resources must not interact with each other directly. This must happen through controller class.

AddRotator App.

- ⇒ Should display the advertisement randomly after every 2 sec.
- ⇒ Advertisement should come as graphical hyperlink.
- ⇒ Should be implemented based on MVC architecture
 [Java [JSP <----> Java bean communication]]
- ⇒

3 types of bean properties:-

a) Simple Properties (Allows to set 1 value at a time)

 int age;
 public void setAge(int age){}

 this.age = age;
 }

 public int getAge(){}

 return age;
 }

b) Boolean Properties

 private boolean married;

 public void setMarried(boolean married){}

 this.married = married;

}

 public boolean isMarried(){}

 return married;

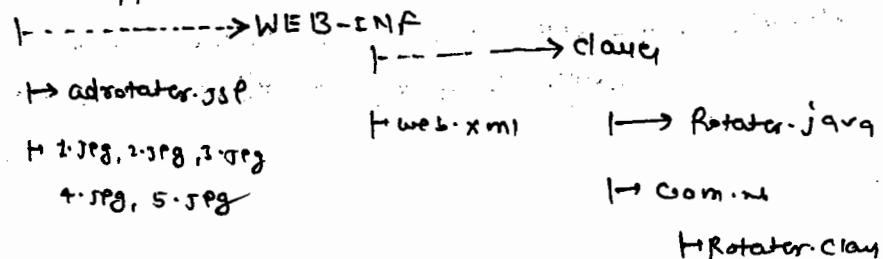
}

c) index properties

```
private String[] colors;  
public void setColors(String[] colors){  
    this.colors = colors;  
}  
public String[] getColors(){  
    return colors;  
}
```

refer App 8 of page no 271 & 272

AddRotator App



http://localhost:3030/AddRotatorApp/adrotator.jsp

Comments

C22 → file contain logic

C27, C28 → indexed properties

C42, C45 → Generate Random numbers

C52 → Give one image from the array getLink →

C56 → One URL from the array

C49 → Create or locate my href target

→ getproperty function call get

C53 → call getImage().

C56, C58 → Displays graphical hyperlink.

Rotator.nextAdvertisement() → Calls java Math method to get pseudo random number.

`response.setIntHeader()` - response header that gives instruction to browser to refresh the web page at regular intervals.

(17)
23/05/15

Mini Project Basics:-

⇒ Resultset is not a Serializable Object to send over the network, So to send Resultset data/records over the n/w we have 2 the solutions -

a) Work with Rowsets instead of Resultsets.

→ Rowsets are Serializable obj but very few jdbc drivers are supporting rowset

b) Copy the Records of Resultset into Collection (java.util pkg)

→ All the collections are Serializable obj to send over the network

Note:- (b) is recommended solution

⇒ If we are looking to perform more read operation on a collection then use Non-synchronized collection for better performance.

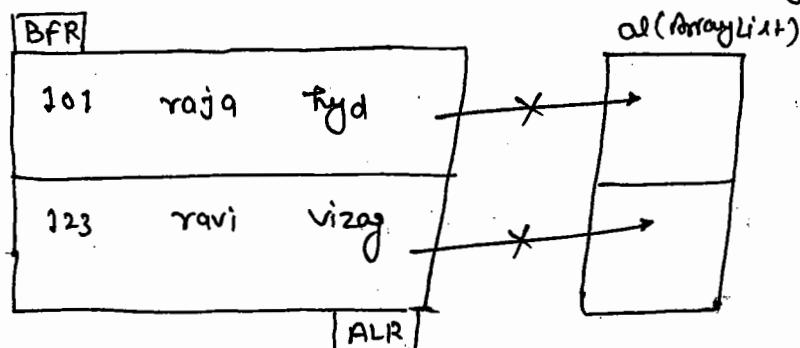
e.g. ArrayList, HashMap

⇒ If we are looking to perform more read and write operations on a collection then use Synchronized collection for achieving thread safety.

e.g: Vector, Hashtable

- ; Problem to Copy the records of ResultSet to ArrayList elements:-

⇒ Each element of ArrayList allows only 1 obj at a time, where as each record ResultSet contains multiple col values, So we can't copy each record of ResultSet to each elements of ArrayList directly.



- ⇒ To overcome the above problem, copy the records of ResultSet to a Custom javaBean/ java class objects and Add those objects ~~to~~ elements of ArrayList.
- ⇒ Here custom java bean or class is called as BO class. (Business Object class).
- ⇒ To send collection over the network the obj added to elements of collection should also be taken as Serializable object.

Student.java

```
public class Student implements Serializable {  
    private int no;  
    private String name;  
    private String address;  
    // write setXXX(), getXXX() methods  
}
```

- ⇒ Logic to copy the records of ResultSet to ArrayList elements.

```
ArrayList < Student > list = new ArrayList < Student > ( );
```

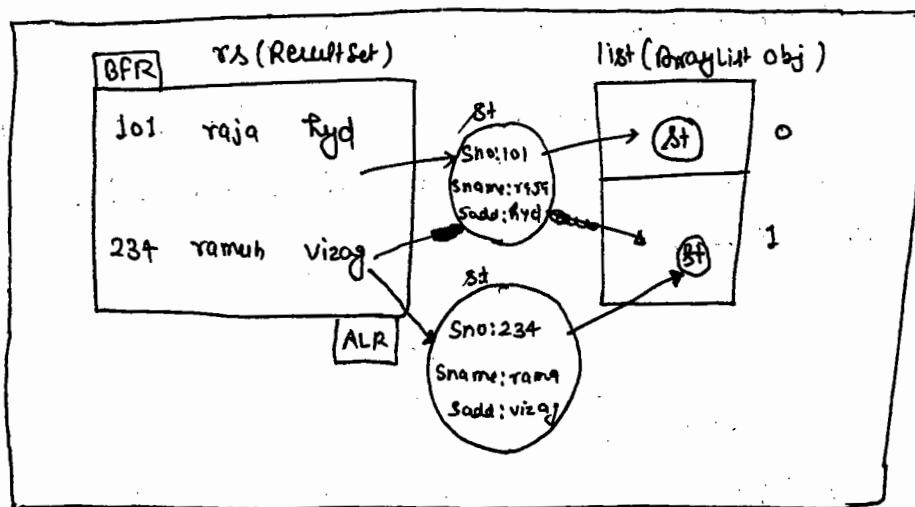
```
ResultSet rs = st.executeQuery ( Select * from Student );
```

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

```
    Student st = new Student ( );  
    st.setSno ( rs.getInt ( 1 ) );  
    st.setSname ( rs.getString ( 2 ) );  
    st.setSadd ( rs.getString ( 3 ) );  
}
```

} copy each record
of ResultSet to
Student class obj.
(BO class)

```
list.add ( st ); } Add Each Student class obj to ArrayList
```



Q) Where did you use java bean in your projects?

- A) As BO class holding DB table record / ResultSet record.
- ⇒ As VO class holding input values (form data) given by end user in a object.
- ⇒ As DTO class to send multiple values from one resource to another resource, as Serializable java Object.
- ⇒ As model Layer resource - having b-logic in MVC1, MVC2 architecture based web application.
- ⇒ As Domain/ Entity class in Hibernate programming.
- ⇒ In Session tracking to hold multiple values from page as single Session Attribute value.

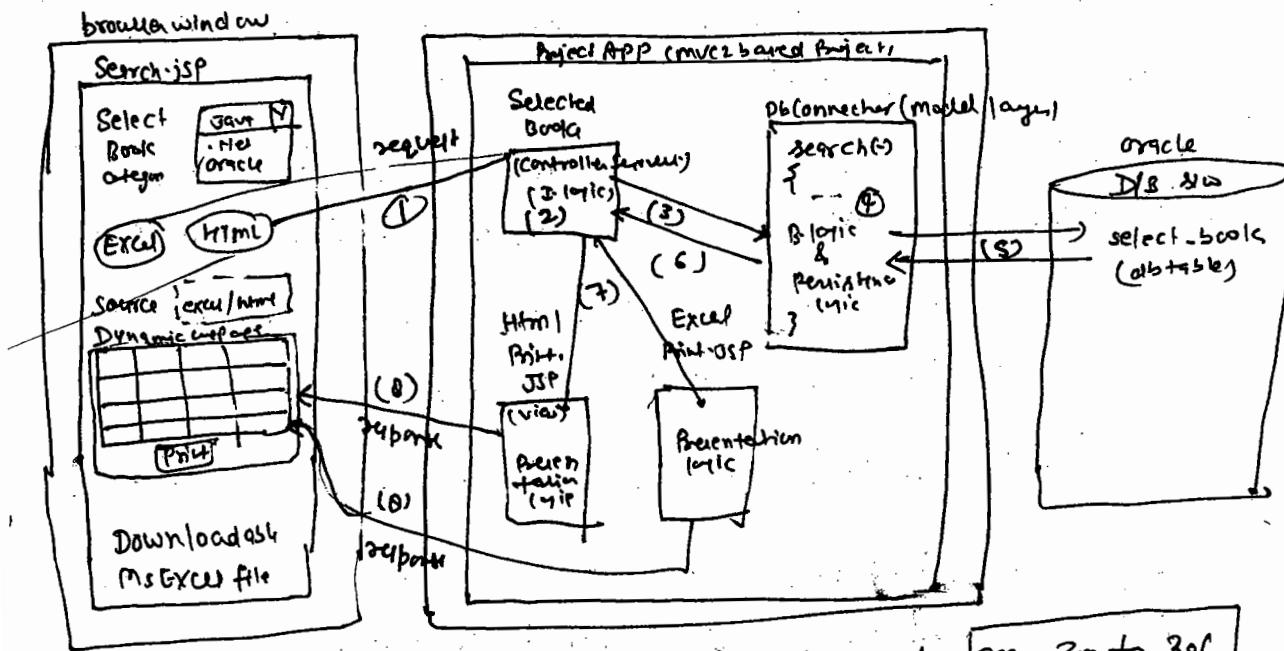
⇒ As Command class in Spring MVC and etc...

Where did you use Collection in your project?

- To send ResultSet obj records from one resource to another resource we copy them into ArrayList.
- To get jdbc details from properties file we use java.util.Properties class.
- To prepare Jndi properties we need Hashtable/Properties class.
- To maintain dynamically growable buffer/cache.
- In Session tracking to maintain multiple values on single session attribute value we keep multiple values in a collection. And etc....

Mini Project:-

- ⇒ Based on MVC2 Architecture
- ⇒ Model → java class (having b.logic & p. logic)
- View → JSP Prg (Having presentation logic)
- Controller → Servlet (Having Integration logic)



For Above diagram based Example Application refer page 30 to 30f

w.r.t. diagram:-

- 4) & 5) Model layer class gets ResultSet having dbstb records copies them to an ArrayList in the form Bookbean class obj (BO class obj) to send that ArrayList to controller service.
- 6) Controller Servlet keeps the Received ArrayList in request Attribute and forward the control to ExcelScreen.jsp or HtmlPrint.jsp based on button that is clicked.

Imp points:-

- ⇒ This project is based on MVC2 architecture.
- ⇒ BO class is used to copy the ResultSet records to ArrayList elements.
- ⇒ Java Script is used for form validation and also for form submission.
- ⇒ Hidden box is taken to get token to servlet to know which button is clicked to submit button for request.
- ⇒ Output comes downloadable excel file when "exceloutput" btn is clicked.
- ⇒ Output comes as html table having option to print page when "htmloutput" btn is clicked.

18

25-05-15

Comments:

/* TIZ → Bruce Hackel (BH) */

Search.jsp → form page

mainsrv.java → Controller Servlet program

DBConnector.java → java class having model layer, b. logic & persistence logic

book bean → Business object class

HtmlPrint.jsp → result page

ExcelScreen.jsp → result page

1282 to 1284 ⇒ logic to copy the records of ResultSet to ArrayList element

1289, 1290 → ordinary button

1294 → Hidden box

f → form name

source → refer 1294 validate() → refer 1171

1247 → 1250 ⇒ forwards the control to result page

1394 HtmlPrint.jsp

1395, 92 → reading request Attribute values (refer 1238, 39)

1414 → books belonging to "JAVA"

1443 → calls java script function

1446, 50, 51, 52 → makes the o/p of Jsp Program as downloadable file.

1444 →

1457 → 1483 ⇒

⇒ logic to display ArrayList obj data in the form of html table content

ProjectApp

→ WEB-INF

↳ Search.jsp

↳ HtmlPrint.jsp

↳ ExcelScreen.jsp

→ classes

↳ A.java(3)

↳ com.nt.controller

↳ mainsrv.class

↳ com.nt.Model

↳ DBConnector

↳ BookBean.class

Jar file in classpath = jdbc14.jar
servlet-api.jar

1141 (a), 1154 (b), 1209 (c) 1159 (d) 1171 (e) 1181 (f)
 1189 (g) 1228 1144 to 1152 (h) 1229 (i) 1237 (j)
 1291 (k) 1348 (l) 1287 (m) 1243 (n) 1250 (o) 1394 (p) 1443 (q)
 1404 (r)

⇒ For more mini projects based on servlet, jsp refer page no 285 to 310.

	(19)
26-05-15	

Custom Jsp Tag Library:-
 x ————— x ————— x

⇒ Jsp tag library is a library that contains set of Jsp tags. There are two types of Jsp tag libraries —

a) Custom Jsp tag libraries

→ Created by the programmers or Third party vendors.

b) JSTL (Jsp Standard Taglibrary)

→ Given by sun ms/orcorp

⇒ Placing java code in jsp is not industry standard
 (Avoid <% %> , <%!= %> , <%= %> tags)

⇒ We should always develop jsp page as script less jsp page (java code less) for this we need to use following 4 tags

a) html tags

b) Jsp built-in tags

c) Jstl tags

d) Custom tags

⇒ The advantage of developing jsp page as java code less jsp page is

a) improves the readability of jsp page

b) improves the reusability of java code represented by jsp tag

Custom tag lib

Custom Jsp Taglibrary

Terminologies

Jsp taglibrary: It is the library where set of jsp tags are available.

Tag Handler class: The java class that defines the functionality of jsp prg is called tag handler class. This class extends from "TagSupport" class directly or indirectly.

TLD file : The xml file where all the jsp tags of taglibrary are cfg is called tld file (Tag library descriptor). This file contains tag name, tag handler class name, optional or mandatory attribute names and etc...

⇒ Every Tag handler class contains two methods —

a) doStartTag() → Contains logic to execute when open tag & its body encounter.

b) doEndTag() → Contains logic to execute when closing tag encounter.

⇒ These two methods can give instructions to Container to perform further operations by returning numeric value in form constants

SKIP_BODY, EVAL_PAGE, EVAL_BODY_INCLUDE and etc..

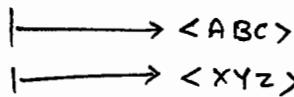
SKIP_PAGE, EVAL_BODY_AGAIN

⇒ In every taghandler class there will be one inherited property of super class that is "pageContext" we can use this "pageContext" to access other implicit objs even though the tag handler class is ordinary java class.

Procedure to develop Custom Jsp taglibrary having custom Jsp tags:-

Step1) Design Custom Jsp tag library

NTagLibrary



Step2) Develop tag handler classes for the Jsp tags

ABCTag.java (WEB-INF/classes folder)

```
public class ABCTag extends TagSupport {
```

```
    public int doStartTag() {
```

```
        // logic to process open tag  
        // & its body (g)
```

```
}
```

```
    public int doEndTag() {
```

```
        // logic to process closing tag  
        // & its body. (g)
```

```
}
```

XyzTag.java (WEB-INF/classes folder)

```
public class XyzTag extends TagSupport {  
    public int doStartTag() {  
        --  
    }  
    public int doEndTag() {  
        --  
    }  
}
```

Step 3

Develop Tld file having cfgs of Jsp tags

WEB-INF/NIT.tld

<ABC> -----> ABCTag.class

<XYZ> -----> XyzTag.class

Step 4

cfg Jsp tag library in web.xml file

```
<web-app>  
    <tlds> (d)  
        <taglib-uri> demo </taglib-uri>  
        <taglib-location> /WEB-INF/NIT.tld </taglib-uri>  
    </tlds> (e)  
</web-app>
```

Note:- Every Jsp taglibrary is identified with its taglib uri.

Step 5:- Use Jsp tags in the jsp page of the web application -

Test.jsp

```
<%@taglib uri="demo" prefix="nt"  
          ↙ (c) ↘  
          →(b) ↗  
<nt:ABC/>  
          ↗ a. ↘
```

```
<nt:xyz/>
```

⇒ "prefix" is user-defined, if multiple jsp tags or multiple jsp taglibraries are having same name then they can differentiated with "prefix".

⇒ W.r.t. above code

a) → nt:abc code encountered.

b) → from that tag prefix will be gathered.

c) → Based on the prefix taglib uri will be gathered

d&e) → Based on the tag lib uri the name and location of tld file will be gathered

f) → Based on the tld file the tag handler class name will be gathered.

g) → Tag handler class execute & the response goes to browser.

⇒ <%@taglib ...%> tag is required to use custom taglibrary tags or JSTL tag library tags in our JSP page / pg.

for Example Appn on Custom Tag library development
refer page no 282 to 285 of the booklet

//Comments

6-9: JSP tag library Cfg.

12: Uri = demo (refer line no 7)

19: <start:example> (an empty JSP tag that displays one line of text)

22: <start: prime>

21,23: empty tag with optional attribute n to display all the prime numbers upto n.

28: <start: display> //with 1 mandatory attribute (size)
// + optional attribute (font)

25 to 31: tag with body with 1 optional attribute size & mandatory attribute font

: To display the given text in different font & size.

44:

49: tag.ExampleTag → Tag Handler Classes

60 to 63: Here attribute n is optional attribute.

60 → indicate tag doesn't allow body.

70 → body content = JSP → indicates the content allowed in JSP program also allowed as the content of tag body.

71 to 74 → font is mandatory attribute.

CustTagLibApp

→ WEB-INF
→ tut.jsp

→ classes
→ web.xml
→ tags
→ first.tld
→ ExampleTag.class
→ PrimeTag.class
→ DisplayTag.class

JAR file: jsp-api.jar, servlet-api.jar

CustTagLibApp (Eclipse)

→ Java Resource
→ web content

→ tags
→ first.tld
→ web.xml
→ WEB-INF

JAR file: servlet-api.jar, JSP-api.jar

http://localhost:
3030/CustTagLibApp.jsp
/tut.jsp
? print=1

20
27-05-15

Comments:

Example Tag.java

extends TagSupport \Rightarrow mandatory

doStartTag() \Rightarrow execute for <start> Example > open tag

return skip_body \Rightarrow give instruction to container not to evaluate the body of the tag becoz the tag is an empty tag

Public int doIntTag(),

eval_page \Rightarrow give instruction to container to evaluate the remaining page

PrimeTag.java

n=10 \Rightarrow hold the attribute n value (default value is 10)

public void setN{ -- } execute automatically to set Attribute n value to variable n.

public int getN{ ... } optional

private boolean isPrime(int x){ -- }

[131 to 139] return true if given no. is prime no.

140 \Rightarrow [46 to 130] displays all the prime numbers that are there b/w 1 to n.

catch {

return skip_body;

displaytag.java \Rightarrow This <start:display> tag generated output will be included in the response only when print=8 request param is taken

\Rightarrow To hold font, size attribute value

setFont, setSize

<start:display font="arial" size="10">

HCL

</start:display>

```
<table border='0'>  
  <tr> <td> <span style="font-size:10px; font-family:arial;">  
    HCL
```

http://localhost:3030/Cut+Taglib/tst.jsp?print=yel

web.xml

csg4e

```
<web-app>  
  <JSP-config>  
    <taglib>  
      =:  
      </taglibs>  
    </JSP-config>  
</web-app>
```

first.tld

```
<tlib-version>  
<jsp-version>  
<short-name>  
replace *<info> with <description>  
  
* → replace <info> tag with  
<description> & place it after  
<body-content>
```

JSTL (Jsp Standard Tag Library)

- ⇒ Developing the JSP program as java code less JSP program is industry standard. For this we should avoid `<% ... %>`, `<!/ ... />`, `<%= ... %>` tags from JSP program, and we can use JSP standard action tags.
- ⇒ JSP standard action tags are very few in numbers so they are not sufficient to make JSP page or Java code less JSP page. To overcome this problem we can use custom action tags but developing custom action tags is very complex process.
- ⇒ To overcome this Sun has given JSTL or tag library specification having set of JSP tags.
- ⇒ Every vendor company who provides webserver/app server implements JSTL specification and provides implementation to JSP tags.
- ⇒ JSP tags are very useful to make JSP page or Java code less JSP page i.e. it allows to perform every operation in the form of tags like variable declaration to DB connectivity and etc.

Five Taglibraries of JSTL

- 1) Core
- 2) SQL
- 3) XML
- 4) FMT (I18n)
- 5) Functions

1) Core:-

⇒ For Basic operations like variable declaration, loops, conditions and etc...

2) Sql:-

For DB connectivity and other operations.

3) Xml:-

For XML processing

4) Fmt:-

To add the support Internationalization. To format numbers, date and etc...

5) Functions:-

Utility functions for manipulating collections and strings.

For various tags of JSTL Tag libraries refer page 312 to 314

⇒ In JSTL Vendor company supply ready made tld file and taghandler class.

⇒ In Tomcat Server these classes and tld file comes in the form of jstl.jar and standard.jar files
(we can get these files from internet)

⇒ The tld files are c.tld, x.tld, fmt.tld, sql.tld and etc.. (we can collect them from jstl.jar file)

⇒ Every tag library of JSTL can be identified with its taglib uri, we can get this taglib uri directly from tld file or by cfg tld file in web.xml file.

Example:

For core taglibrary fixed taglib uri collected from <uri> tag of C.tld is

http://javq.sun.com/jstl/cor

But we can also get user defined taglib uri from web.xml

```
<jsp-config>
  <taglib>
    <taglib-uri> core </taglib-uri>
    <taglib-location>/WEB-INF/c.tld</taglib-location>
  </taglib>
</jsp-config>
```

Note →

- Fixed taglib uri is recommended to use becoz it doesn't make you to arrange tld file separately.

⇒ To use JSTL tags (core) in our jsp prg we should specify tag lib uri in our jsp prg

In test.jsp

<%@ taglib uri="http://javq.sun.com/jstl/core" prefix="c"%>

→ fixed taglib uri

(*)

<%@ taglib uri="core" prefix="c"%>

→ user-defined

Other JSTL libraries fixed taglib uris.

http://javq.sun.com/jstl/core → core

/xml → xml

/fn → fn

/sql → sql

/functions → functions

recommended prefix

c

x

fn

sql

fn

⇒ Every JSTL taglibrary gives set of tags and we can use all these tags to make our JSP progs by java code less JSP progs..

(21)
~~30 - 05 - 15~~

⇒ To perform arithmetic and logical operation in our jsp progs without using java code we can use EL (Expression Language). we can apply this EL either on tags or on template tags. text.

Syn:-

`$ { <expr> }`

- ⇒ We always use JSTL tags with EL.
- ⇒ EL supports all the operations of java also gives few more implicit objs to support java code less jsp programming , The implicit obj are -
- requestScope
 - sessionScope ⇒ To keep values in diff scope
 - pageScope
 - applicationScope
 - cookie
 - freader ⇒ To read single http request header value
 - freadervalue ⇒ To read multiple http request value
 - param ⇒ To read single req param value
 - paramValue ⇒ To read multiple req param value

Procedure to develop JSTL App:-

Step1) Create Deployment directory Structure

JSTLApp1 (Eclipse Dynamic Web project)

 |→ webcontent-

 |→ Example1.jsp

 |→ WEB-INF

 |→ web.xml

 (→ c.tld, sq1.tld,
 fmt.tld

(from standard.jar)

JAR files: jstl.jar, standard.jar, servlet-api.jar,
JSP-API.jar

web.xml

<web-app> <taglib>

<taglib-uri> http://java.sun.com/jstl/core </taglib-uri>

<taglib-location> /WEB-INF/c.tld </taglib-location>

</web-app> </taglib>

Example1.jsp

<%@ taglib uri = "http://java.sun.com/jstl/core prefix = "c" %>

<c: set var = "phame" value = "raja" />

<c: out value = "\${phame}" />

for more Apps on JSTL refer page no 344 to 346

