**J2EE**

Computer

* Computer is a combination of hardware and os.
* Graphically Computer are represented mention below:
* Application is a collection of programs with a dedicated functionality.
* Every application in this world is platform dependent including java.
* Java as an application is plantform dependent but as a programming language is platform independent.

JAVA

.class

Mac os

Win os

* Every application will have its own known file extentions.

Ex: pdf readers, media readers.

* Application acts like a interface between user and os.
* Os acts like a interface between application and hardwear.

­User

user

App

Applicatio

Os

O.S

H/W

There are two types of applicaton:

1. Stand Alone or Desktop Application(Unshared Application)
2. Web Application (Shared Application)

**Stand-Alone or Desktop Application:**

1. These are the application which are present in our own computer and they are dedicated per user. For exapmple: pdf readers, media players.

**Web Application:**

1. These application are not present in our computer but they are presenet in “some other” computer and our computer and other (server) is network connected.
2. In other words web application are present in networks.
3. To interact with web application we must need to make use of
4. Network
5. Web Browser

Networ is a collection of Computers and there are two types of network:

1. Intranet (Private)
2. Internet (Public)

* So web application can be present either in internet or intranet:

For ex: Jspiders attendence tracking web application is present in intranet and gmail web application is present in internet.

* Whenever we make use of web browser it mean that we are interaction with web application.

**Servers**

Its also a computer but hardware of this computer is pretty high level compare to our own computers. For ex: Servers can have ram upto Gbs of rams…

Graphically server are represented as:

**RDBMS Application (Database or DB)**

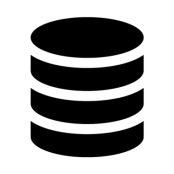
It also an application which helps us to store and maintain gbs to tbs of data.

Graphically rdbms application is represented as :

-------------------------------------------------------------------------------------------------------------

J2EE: java is not enough to develop web application so we need some added featers …so J2EE WAS DEVLOPED.

J2EE: java 2 Enterprise Edition (it was introduced in java 1.2)

C:\Program Files (x86)\Microsoft Office\MEDIA\CAGCAT10\j0292982.wmf  

Web Application Server

Client

**DataBase**

**JDBC**

**Servlets**

**JSP**

**Java Mail**

Packages are used to organize the java programs efficiently.(Collection of java Programs).

API- Collection of packages and every api has its own dedicated functionality.

In J2EE there are many APIs in J2EE like: JDBC, Servlets and JSP, JAVA mail, etc.

**Java 2 Enterprise Edition (J2EE, JEE, JAVA EE)**

J2ee is a collection of api which helps us to develop enterprise web application

An api (Application Programming Interface) is a collection of packages (One or More) with a dedicated functionality, Major API of J2EE are: JDBC , SERVLETS, JSP.

JDBC helps web application to interact to database

Servlets or jsp helps web application to get the request from browser, generates the response, and gives response back to the browser.

All the API’s of J2EE are based upon java and hence they are platform independent.

**Java DataBase Connectivity**

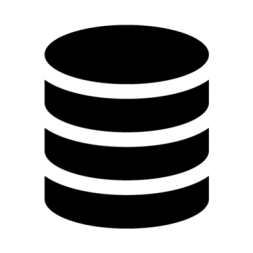
Java DataBase Connnectivity is an API, as the name implies, IT HELPS TO ACHIEVE THE CONNECTIVITY BETWEEN Java Programs & Database

Note: Servlets & JSPs are also java Programs

If we have a Web application & if it has a DB, then it needs to interact with DB to read/ modify the data

JDBC helps to do this & in the world of Java, JDBC is the “one and only” API that helps to interact with RDBMS (DB) Application.

Also JDBC is “DB independent” i.e. Using JDBC we can interact with any RDBMS Applications in the world.



JAVA

JDBC

JDBC Pre-Requirment:

1. Install Any RDBMS Appllication (MySQL)
2. Create a “Database (Schema) “ by any name.
3. Create a table
4. Insert some data into the table.

SQL Queries for MySQL RDBMS Application:-

1. create databse BECME89\_DB;
2. use BECME89\_DB;
3. create table students\_info

(

regno Int(10) not null,

firstname varchar(50),

middlename varchar(50),

lastname varchar(50),

Primary key(regno)

);

1. insert into table\_name values(regno,’firstname’,’middlename’,’lastname’);

Sone UseFull Queries:

1. To Connect to database:

use BECME89\_DB;

1. To get the list of databases:

show databases;

1. To get the list of tables:

show tables;

1. To know the table structure:

describe table\_name;

**Java ThumbRules** :

A class which is declared with abstract keyword is called as abstract class.

A Class declare without abstract keyword is called as concrete class.

If LHS == RHS (A reg = new A(); ) then LHS Is always concrete class. If LHS != RHS , ( B ref = new A(); ) then LHS can be an interface, concrete Class, or Abstract Class.

Simply having an abstract class is of no use and there must be at least one subclass.

Simply having an interface in java of no use and there must be at least one implementation class.

In java anything apart from primitive data types are called as object references. For example: Int I ,boolean isTrue(“i “ and “isTrue” is a primitive variable) , Object obj -> it’s an Object ref varialble.

Anything is java which starts with lower case is eithere variable, if its with brackets its a method. And Anything which starts with a Caps is either a Class or an Interface.

In Java, “Super Class” can be either an “Abstract Class” or “Concrete Class”.

**Neccassary Steps to Work with JDBC**

1. Loads the **Driver**
2. Get the **DB Connection** via **Driver**
3. Issue **SQL Queries** via **Connection**
4. **Process the results** returned by **SQL Queries**
5. Close All **JDBC Objects**

Note:-

>”java.sql.\*” is the Package Representation of JDBC

> i.e Any Class / Interface belongs to this package means it’s part of JDBC

**Drivers**

Drivers are additional software components required by JDBC to interact with RDBMS applications. Drivers are provided by DB vendors and they are DB depentdent, that means, using mySql driver we can only interact with my Sql RDBMS application and using Db2 drivers we can only interact with Db2 RDBMS application.

**JAR (Java Archieve) File:**

* It’s a collection of “.class” files + others Necessary Resources (Text File ,XML, Property Files, Etc.)
* JAR File helps us to transfer the “Java Files/ .class filse/ Java Application” From one location to an another location
* JAR File will have “.jar” file extension & Functionality wise it’s similar to “ZIP” file

**Steps To Create JAR File:**

1. Right Click on the Java Project, which we want to transfer , select “Export…”
2. Select “JAR file” option present under “Java” & click on “Next”
3. Provide the “Destination & File Name” , click on “Finish”

**Steps To make use of JAR File:**

1. Right Click on the Java Project, where we want to make use of JAR File, select “Build Path” click on “Add External Archieves…”
2. Select the “JAR File” & Click on “Open”
3. We see JAR File under “Referenced Libraries”

DRIVERS

DB

JAVA

JDBC

**Driver Class**

* “Driver Class” is a Concrete Class, present in driver JAR file, is the one that implements the “java.sql.Driver” interface
* This interface is present in JDBC API & every JDBC driver provider has to implement this Interface.
* “Driver” helps us to establish DB Connection, transfers the DB query and results between Java program and RDBMS Application.

Steps to Load the “Driver Class” into the Program

1. By invoking “registerDriver()” method present in “java.sql.DriverManager” Class by passing an instance of “Driver Class”

Syntax:

**public vod DriverManage.registerDriver(java.sql.Driver driverRef) throws SQLException**

for MySQL Driver:

com.mysql.jdbc.Driver ref = new com.mysql.jdbc.Driver();

DriverManager.registerDriver(ref);

**Second approach**

Class.forName(“com.mysql.jdbc.Driver”)

* This is the most common approach to resister a Driver Class which helps us to pass “Driver Class Name at Runtime”

**Driver Types**

There are 4 types of Drivers

1. Type 1: JDBC-ODBC Bridge
2. Type 2: Native-API Driver
3. Type 3: Network-Protocol Driver
4. Type 4: Native-Protocol Driver

Note: **In JDBC 4, the driver loads automatically, if the jar file is present in the project’s Classpath.**

ODBC: Open Data Base Connectinvety

* DriverManager has 3 overloaded version of getConnection() methods

1. Connection getConnection(String dbUrl) throws SQLException

String dbUrl = “jdbc:mysql://localhost:3306/BECEME89\_DB?user=root&password=root”;

Connection con = DriverManager.getConnection(dbUrl);

1. Connection getConnection(String dbUrl, String userNM,String password) throws SQLException

String dbUrl =

“jdbc:mysql://localhost:3306/BECME89\_DM”;

String userNM = “root”;

String pass = “root”;

Connection con = DriverManager.getConnection(dbUrl, userNM, pass);

1. Connection getConnection (String url, Properties info) throws SQLException

String dbUrl = “jdbc:mysql://localhost:3306/BECEME89\_DB”;

String filePath = “”;

FileReader reader = new FileReader(filePath);

Properties props = new Properties();

Props.load(reader);

Connection con = DriverManager.getConnection(dbUrl, props);

//Data Present in “db.properties” File is:-

#DB Credentials

User = root

Password = root

Note:

* We can make use of any version of “getConnection()” method to establish connection to RDBMS application
* But “getConnection(String url, Properties info)” helps us to take out the hardcoded credentials from program & keep it outside of the application
* Hence this method is widely used because it helps us to “easily maintain the application “ whenever there is change in DB credentials.