With core java knowledge we can build standalone applications

Standalone applications

The applications which are running on single machine are called as "Standalone applications".

eg: Calculator, MS-word, notepad, ....

If we want to develop web-applications then we need to go for "Advanced Java".

What are web applications?

The application which provide service over the web/internet are called as "Web Applications".

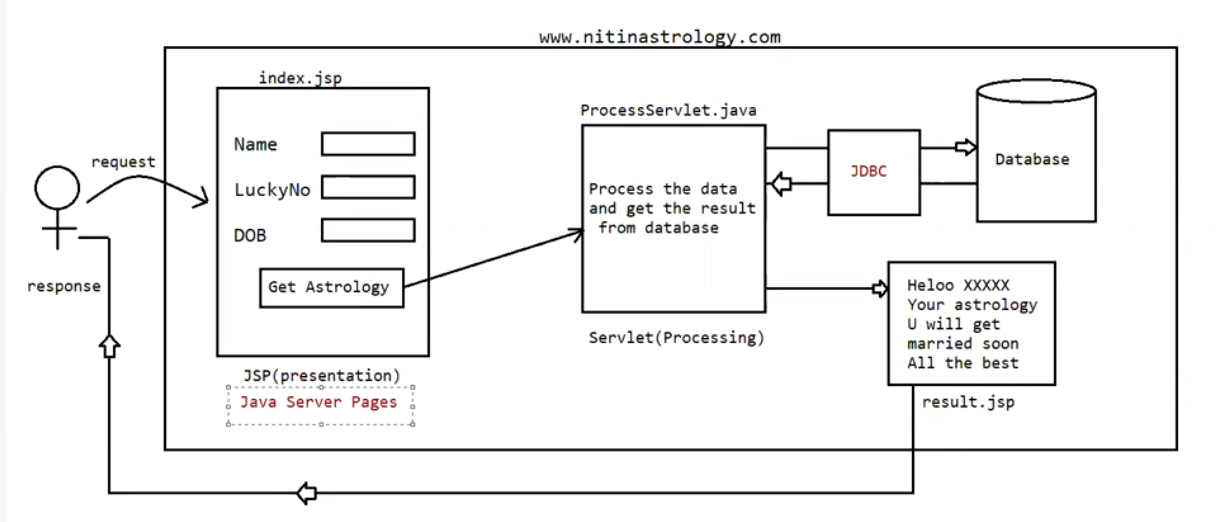
eg: gmail.com, facebook.com,ineuron.ai,telusko.com, .......

In java we can develop web applications using the following technologies

a. JDBC

b. Servlet

c. JSP/Thymeleaf



JDBC(Java Database Connectivity)

For a Java Application(Normal java class or Servlet) if we want to communicate with database, then we need to go for JDBC.

eg: To get mails information from database

To get Astrology information from Database

JSP

Whenever presentation logic is required todisplay something to the end user then we need to go for jsp . Jsp stands for View Component

eg: display login page

display inbox page

display error page

displav result page

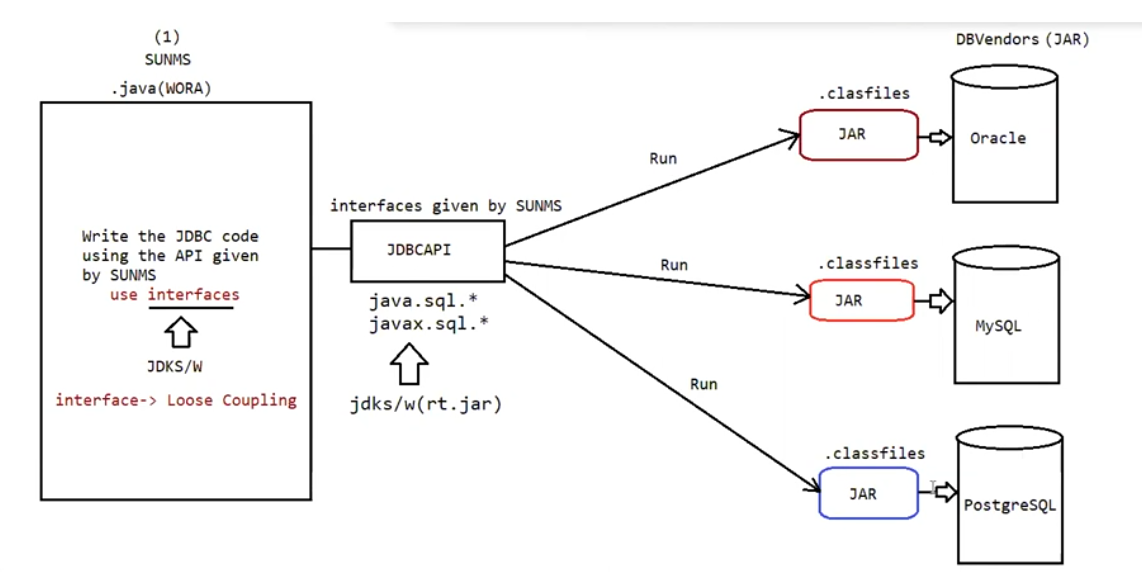
Servlet

Whenever some processing logic is required then we need to go for Servlet. Servlet is meant for Processing logic/ Business Logic.

eg: Verify user

Communicate with the data

Process end user data



Steps given by SUNMS to communicate with Database

1. Load and register the Driver.

2. Establish the connection with Database.

3. Create Statement Object and execute the Query.

4. Process the ResultSet.

5. Handle the SQLException if it gets generated I

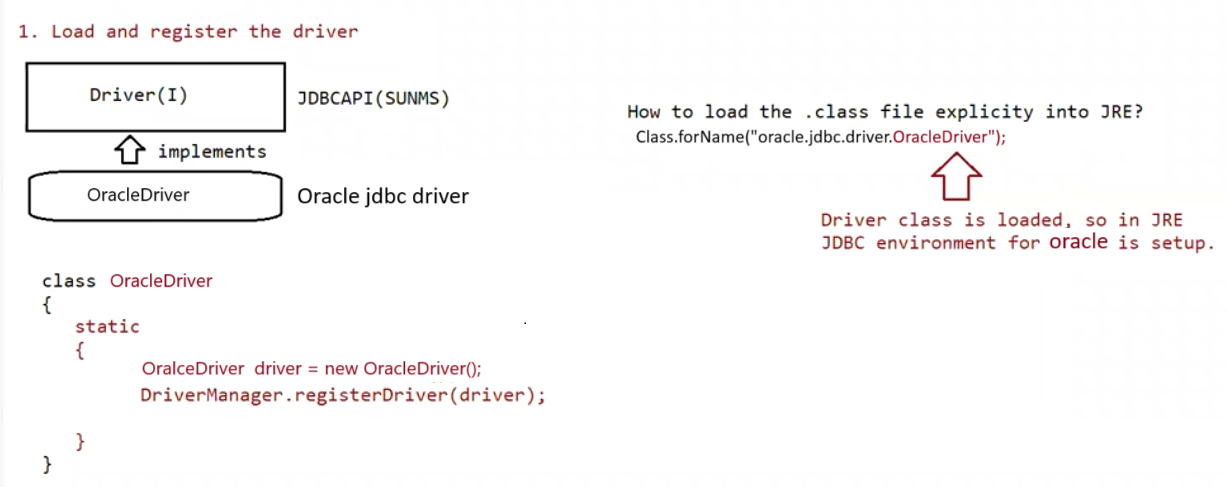
6. Close the Connection

1. Load and register the Driver

We need to load and register the Driver as per the DB requirement.

As per the DB specification, we need to set the JRE environment with the DB environment.

Any class of DB vendor, we say at is Driverclass, iff it has implemented a interface called "Driver".

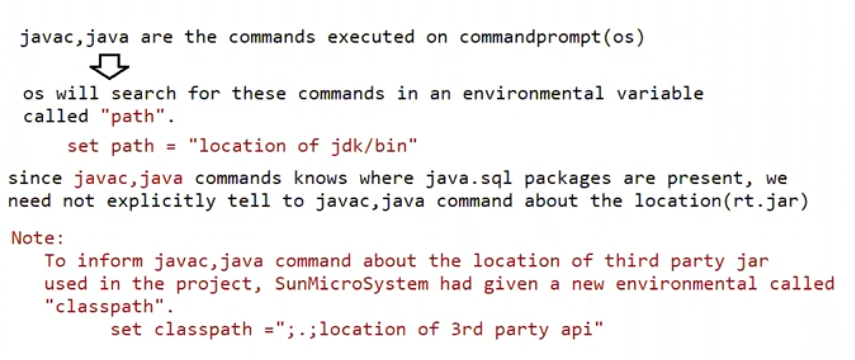


To load the Driver class we should know, what is the class that is implementing driver interface

( here OracleDriver class is implementing Driver Interface ) . to load it manually we are using

The static method forName() present in Class class.

OracleDriver class contains a static block that contains the code to register the driver. Which will be executed during class loading itself.



2. Establish the connection:

public static Connection getConnection(String, java.util.Properties) throws java.sql.SQLException;

public static Connection getConnection(String, String, String) throws java.sql.SQLException;

public static Connection getConnection(String) throws java.sql.SQLException;

Connection connection = DriverManager.getConnection(url,username,password);

The above line would create the connection object, but connection is a interface, can object to interface be created?

No, for an interface object cannot be created

In this line object to a class which implements an interface called Connection is created and we hold the reference of the object with the interface name

This is done to promote the loose coupling in java

The above code would represent abstraction

3.Create Statement Object and execute the Query.

Statement statement = connection.createStatement();

The above line would create statement Object, but Statement is an interface, can object to interface be created?

Answer: No, for an interface instantiation is not possible.

In this line, an Object to a class which implements an interface called Statement is created and we hold the reference of the object with the interface name.

This is done to promote loose coupling in java.

The above code would also represent "abstraction".

4.

ResultSet resultSet =statement.executeQuery(sqlSelectQuery);

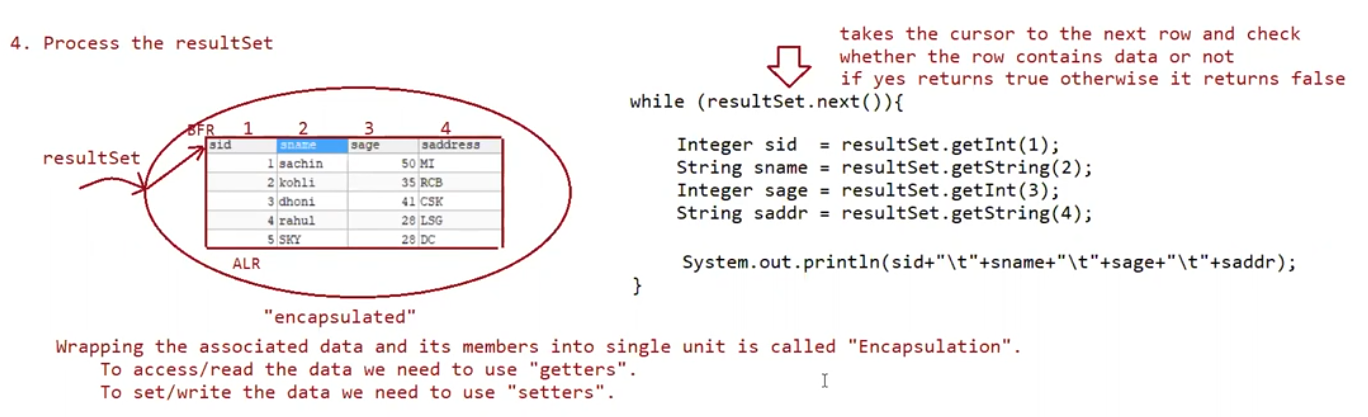
The above line would create statement Object,but ResultSet is an interface,can object to interface be created?

Answer: No, for an interface instantiation is not possible.

In this line, an Object to a class which implements an interface called ResultSet is created and we hold the reference of the object with the interface name.

This is done to promote loose coupling in java.

The above code would also represent "abstraction".



Eg: Jdbc\_Example

From JDBC4.X version onwards, there is a facility of "autoloading".

Q> What is autoloading in JDBC?

Loading and register the driver is done automatically, based on the url supplied by the use

Behind the scenes

a. check the url

b. based on the url supplied, go to classpath environmental variable

c. open the relevant jar

d. go to META-INF/services folder

e. open java.sql.Driver file

f. read the file and load the class supplied in the file

Note:

Using resultSet object, we can retrieve the records based on the column names also.

Eg: Jdbc\_Example2

Note

According to DBA specification, all SQL commands are categorised into following types

a. DDL(Data Definition Language)

eg: Create table, alter table, drop table, etc ...

b. DML(Data Manipulation Language)

eg: insert,update,delete

c. DQL(Data Query Language)

eg: Select

d. DCL(Data Control Language)

eg: Alter password, grant access, ....

e. DA command(Database Administrator commands)

eg: start audit

f. TCL(Transaction Control Language)

eg: commit, rollback,savepoint, ....

According to Java Developer point of view, all SQL operations are classified into 2 types

a. Select operation(DQL)

b. Non-Select Operation(DML,DDL, ... )

Through Statement Object we need to execute the Query and to execute the Query we need to make a call

to a method as shown below.

a. executeQuery()

b. executeUpdate()

c. execute()

a. executeQuery()

This method is used only if we perform select operation.

Because of this method execution, we will get a group of records which are represented as "ResutlSet" object.

public ResultSet executeQuery(String sqlSelectQuery) throws SQLException;

eg: ResultSet resultSet =statement.executeQuery("select sid,sname,sage,saddress from student");

b. executeUpdate()

This method is used for "Non-Select Operations" like(Insert | Update | Delete)

Because of this method execution, we won't get group of records, we will get a numeric value which represents the number of rows affected. So return type of the method is "int".

public int executeUpdate(String sqlNonSelectQuery) throws SQLException;

eg: int rowAffected = statement.executeUpdate("delete from student where sid = 10");

System.out.println("No of rows affected is :: "+rowAffected);

c. execute()

we can use this method for both select and non-select operation

if we don't know the type of query at the beginning and if is available dynamically at the runtime then we need to use this method for execution.

public boolean execute(String sql) throws SQLException;

eg: boolean value = statement.executeQuery(dynamicQuery);

if(value == true) {

//select Query

ResultSet resultSet = statement.getResultSet();

//process the resultSet

}

else{

//nonSelect Query

int rowCount = statement.getUpdateCount();

System.out.println("Number of rows affected is :: "+rowCount);

}

Eg: jdbc\_Execute\_Update\_Method

// go through the code

Eg: jdbc\_Execute\_Update\_Method\_Eg2

// go through the code

Eg: jdbc\_Execute\_Update\_Method\_Eg3

// go through the code

Formatting SQL Queries using dynamic input

1st approach

sname = scanner.next();

sage = scanner.nextlnt();

saddress = scanner.next();

String sqlInsertQuery = "insert into student(`sname',’sage`, saddress')values('"+sname+"',"+sage+",'+saddress+"')";

2nd approach

sname = scanner.next();

sage =scanner.nextInt();

saddress = scanner.next();

sname = "'"+sname+"'";

saddress = "'"+saddress+"';

String sqlnsertQuery = "insert into student(`sname', sage`, saddress`)values("+sname+","+sage+","+saddress+")";

3rd approach

The above 2 approaches are not recommended, to do formatting we prefer using String class format() as shown below.

public static String format(String format, Object ... args)

sname = scanner.next();

sage =scanner.nextInt();

saddress = scanner.next();

String sqllnsertQuery = String.format("insert into student('sname', sage', saddress') values ('%s',%d,'%s')";

sname,sage,saddress);

Eg: Dynamic\_Input\_Using\_String\_Format\_Method

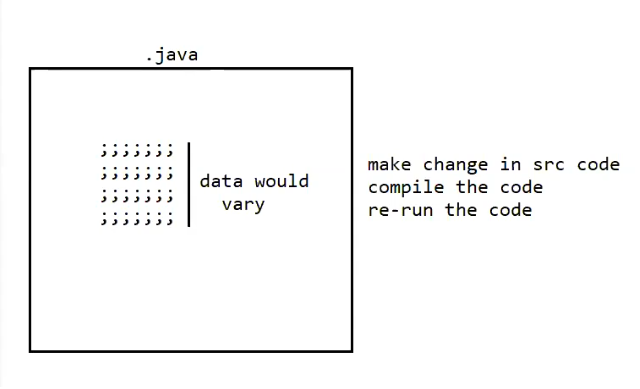
// go through the code

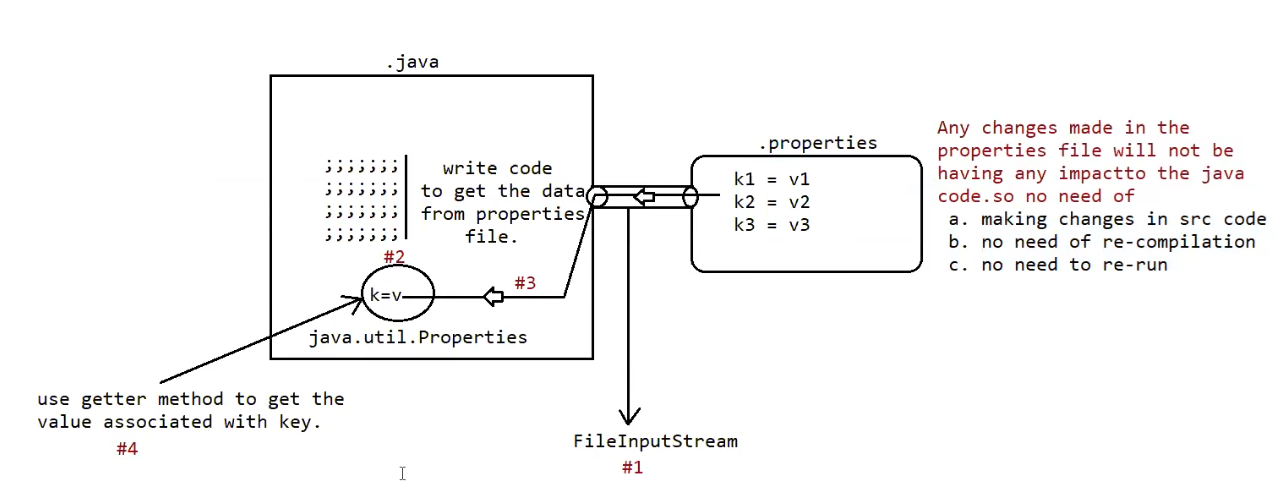
Properties

It represents the data in the form of K,V pair

It represents an object of type java.util. Properties

Properties object holds the data which would be changing frequently in our application.





Eg: Properties\_Example\_With\_Java\_Program

// go through the code

Eg: Properties\_With\_Select\_Operation

//go through the code