Mini Guía Java (2da parte)

Continuamos con esta pequeña guía de Java

Nota: *no se ha tomado ningún orden en particular y la información es muy breve*

1.- ¿Cuales son las plataformas de desarrollo de Java ?

Sun define tres plataformas en un intento por cubrir distintos entornos de aplicación. Así, ha distribuido muchas de sus *APIs* de forma que pertenezcan a cada una de lassiguientes plataformas:

- ***Java ME (Java Platform, Micro Edition) o J2ME:*** orientada a entornos de limitados recursos, como teléfonos móviles, PDAs (Personal Digital Assistant), etc.
- ***Java SE (Java Platform, Standard Edition) o J2SE:*** para entornos de gama media y estaciones de trabajo. Aquí se sitúa al usuario medio en un PC de escritorio.
- ***Java EE (Java Platform, Enterprise Edition) o J2EE :***orientada a entornos distribuidos empresariales o de Internet.

2.- Tecnología Java:

1. ***Programación:***
   - JNI
   - java.*
   - JavaBeans
2. ***Programación GUI:***
   - AWT
   - Swing
3. ***Programación Web del lado del cliente :***
   - Applets
4. ***Programación Web del lado del servidor :***
   - JSP
   - Servlets
5. ***Programación distribuida :***
   - RMI
   - CORBA
   - Enterprise Java
   - Bean
6. ***Sistemas incrustados :***
   - JINI
   - JavaSpaces
7. ***Programación de bases de datos:***
   - JDBC

3.- ¿Qué es JDBC?

Significa *"Java DataBase Connectivity"*. Es una interfaz de comunicación que permite la ejecución entre Java y cualquier motor de bases de datos (*MySQL, Oracle, PostgreSQL, ect.*).

Tipos de conexión

1. **Conexión directa:**El controlador *JDBC* accede directamente al controlador del fabricante (*DB Client Lib*); este tipo de controadores *JDBC* son denominados de nivel 3 ó 4.
    - MySQL
    - DB2
    - Oracle
    - etc.
2. **Conexión indirecta:**El controlador *JDBC* hace *puente* con el controlador **ODBC**, que es el que accede a la base de datos, este es un esquema de controlador *JDBC* de nivel tipo 1.
    - Microsoft Access
    - Microsoft SQL Server
    - Informix
    - etc.

La librería .sql* permie trabajar con bases de datos. Antes de trabajar con bases de datos debes tener tener un **origen de datos**, por ejemplo si vas atrabajar con Access (conexión indirecta); o tener instalado un *controlador* como el de MySQL (conexión directa). Es necesario estudiar las librerías import java.sql.DriverManager,import java.sql.ResultSetMetaData,import java.sql.SQLException import java.sql.Connection

Ejemplo

```
//conexión directa

import java.lang.*;

import java.sql.DriverManager;

public class cert4{ public static void main(String args[]){

try{ Class.forName("com.mysql.jdbc.Driver").newInstance();

System.out.println("\nListo para trabajar con base de datos en MySQL"); }

//fin try catch(Exception e){ System.out.println("\nTodavia no se puede trabajar con base de datos\n...revisar la instalación del controlador\n y la CLASSPATH\n el error es "+e); } }//fin main }//fin clase
```

Los **URL** de *JDBC* proporcionan un modo de identificar un driver de base de datos, en el caso de una conexión directa.

Síntaxis de un URL de *JDBC*

```
jdbc:subprotocolo:subname

jdbc:db2:misDatos

jdbc:odbc:misDatos

//conexión indirecta :ODBC

String URL="jdbc:odbc:misDatos"; //mi origen de datos

String usuario="minombre"; String clave="miclave";
```

Connection conexion=DriverManager.getConnection(URL,usuario,clave);

Para sentencias *SQL* en un objeto Connection

1. Statement
2. PreparedStatement
3. CallableStatement

Pasos para ejecutar una sentencia *SQL* y leer resultados:

1. ***getConnetion()*:**obtener conexión
2. ***PreparedStatement*:**preparar comando SQL
3. ***executeQuery()*:**ejecuta la instrucción SQL que se preparó anteriormente
4. ***ResultSet*:**lleva a memoria los datos de una consult SQL
5. ***netx()*:**lee fila por fila los datos que están en memoria

Nota: El controlador (Connector/J)debe estar en la ruta *C:\Archivos de programa\Java\jdk1.6.0_16\jre\lib\ext*

4.- ¿Cómo puedo crear una interfaz gráfica en Java?

Java permite construir GUIs (*Interfaces Gráficas de Usuario*) empleando las librerías import java.awt.* e import javax.swing.*. **NetBeans** es una buena herramienta que permite crear fácilmente este tipo de aplicaciones.

Ejemplos :

1.- JFrame f = new JFrame("Titulo del frame");

2.- Label l = new Label("Se escribe Algo o se deja vacio"); //etiquetas antes de la caja de texto

3.- JTextField prueba = new JTextField(20) o ("") o ("",20); //cajas de texto

4.- JComboBox prueba = new JComboBox(); //Lista de Datos a elegir

5.- Checkbox prueba = new Checkbox();

6.- JButton prueba = new JButton("Nombre del Boton"); //Boton

7.- JPanel prueba = new Jpanel(); // Sub-Contenedores

//Marcar punto, pero solo uno a la vez, siempre se ocupa en conjunto con el ButtonGroup y RadioButton

8.- ButtonGroup grupo = new ButtonGroup(); //ESTOS SIEMPRE SE UTILIZAN EN CONJUNTO

9.- JRadioButton M = new JRadioButton("Nombre del Boton");

//Lista pero con una ventana que se sube y baja //EL JLIST Y EL JSCROLLPANE siempre se utilizan en conjunto

String[] EstadoCivil = {"Estado Civil", "Soltero","Casado"};

10.- JList EC = new JList(EstadoCivil);

11.- JScrollPane scroll = new JScrollPane(EC); //De esta forma ya esta creado la ventanita con listas, solo hay que agregarla al frame frame.add(scroll);

//asi se agrega Cuando se utilizan JPanel (Se trabaja)

F1 = new JPanel(new GridLayout(filas,columnas));

//Para agregar el layout al Jpanel, ojo el Jpanel no pertenece al contenedor principal son subcontenedores que al final se ordenan con algún layout específico

F1.add(algun componente);

//Se pueden hacer muchos Jpanel, finalmente se ordenan

//ORDENAR EL JPANEL f.getContentPane().setLayout(new BorderLayout());

// se especifica como queremos ordenar el JPanel f.getContentPane().add(prueba, BorderLayout.WEST);

//En el caso de agregar el JComboBox f.getContentPane().add(F1, BorderLayout.NORTH); //Agregar los sub-contenedores f.getContentPane().add(F2, BorderLayout.SOUTH);

//ESTO SIEMPRE VA ARRIBA DE LA CLASE

1.- frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

//Para cerrar Ventana, ESTO SE PONE AL INCIALIZAR EL FRAME

12.- frame.pack();

//ESTO VA AL FINAL , para ordenar todo

13.- frame.setVisible(true);

// Esto es para que el frame se haga visible

14.- frame.setSize(300,400); //Para darle tamaño a la ventana, no siempre es necesario Comandos de algunos componentes 1.- prueba.addItem("Seleccione su Region");

//Asi se le agregan componentes al JComboBox

2.- //Marcar punto, pero solo uno a la vez, siempre se ocupa en conjunto con el ButtonGroup y RadioButton grupo = new ButtonGroup();

M = new JRadioButton("Masculino",true);

F = new JRadioButton("Femenino",false);

//los agrego al ButtonGroup grupo.add(M);

grupo.add(F);

//Con esto quedan listos para agregarlos al frame se agrega frame.add(M); frame.add(F)

3.- String arreglo[] = {"Ninguno","Cierre centralizado","Aire Acondicionado","Sistema de Alarma"}; JComboBox productos = new JComboBox(arreglo);

//Esto es una forma rapido de agregarle los elementos a un JComboBox Eventos, ActionListener

//Mientras se realiza la parte grafica cuando se le da un evento algun componente se pone dentro del codigo donde se crea la ventana con todo, se pone:

// El componente que se quiere agregar a la accion, en este caso un boton, el boton es llamado hola, entonces; hola.addActionListener(this);

public class NOMBRECLASE implements ActionListener public void actionPerformed(ActionEvent e) { Caundo se le quiere dar Accion a un Boton: String comando = e.getActionCommand();

//ESTO SIEMPRE LO PUEDES OCUPAR CON BOTONES if(comando.equals("Aceptar")){

//entonces dices que el string q inicializaste es = Aceptar que es el nombre del boton Aceptar

//por tanto ahi lo reconose y realiza el programa q se pida, como sumar, factorial, entregar una letra,etc., depende lo que pidan, en este caso envio un mensaje System.out.println("HOLA SWING"); } if(comando.equals("Limpiar")){

strings nullos para q se borres rut.setText("");

JtextField y asi se hace con todos los JTextField } if(comando.equals("Salir")){ //con esto cierras la ventana System.exit(0); } if(comando.equals("PELICULA 1")){

// Un ejemplo donde se muestra un entero que se pasa a String en el JtextField Vpe1 voto1++; Vpe1.setText(Integer.toString(voto1)); } public void actionPerformed(ActionEvent e) { JComboBox combo =(JComboBox)e.getSource(); String comando = (String)combo.getSelectedItem(); //Esto es para q el evento reconosca un JcomboBox //que se lo asocio a un String y luego trabajo de la misma forma q con los botones if(comando.equals("Cierre centralizado"))

// precio es el JTextField } if (comando.equals("Agregar")) {

//Ejemplo donde se agregan elementos a un comboBox , al apretar el boton Agregar, donde caja es un JTextField String k = caja.getText(); combo.addItem(k); }

5.- ¿Qué necesito para trabajar con aplicaciones Web en Java?

1. Ambiente de desarrollo **Java SDK**
2. Servidor de bases de datos, por ejemplo,*MySQL*
3. El controlador (driver) **JDBC** para MySQL, *Connector/J*
4. El servidor de aplicaciones **Tomcat** que nos permitirá trabajar con los programas en Java. Con éste se incluye el ***API Java Servlet***.
5. Un editor o un IDE como: *Jcreator o NetBeans (no he ocupado Eclipse)*

Cuando inicie a programar en Java (*jsp*) no sabía mucho de como configurar y *montar* los servidores, pero creo que ahora ya entiendo un poco mejor . Si trabajas con **MySQL**, el problema más común (*creo yo*) es la instalación del controlador.

Solución:*en Windows* el directorio por defecto es:

- **C:\Archivos de programa\Apache Software Foundation\Tomcat 6.0\webapps\ROOT**
- El controlador MySQL (*Connector/J*) debe quedar ubicado en **C:\Archivos de programa\Apache Software Foundation\Tomcat 6.0\lib**

En tu navegador coloca lo siguiente http://127.0.0.1:8080 - ó - http://localhost:8080 para comprobar que la instalación ha sido correcta.

Si ya sabes un poco de programación Web ya puedes empezar a trabajar con JSP y los Servlets.

Tips:

- Para crear un *FAVICON* en tu página, coloca esto en la cabeza: <head><link rel='shortcut' icon type='image/ico' href='../imagenes/alerta.png' / > </head>
- Puedes ocupar **archivos .js**:<script language='javascript' src='../miArchivo.js' > < /script> en la cabeza y dentro del cuerpo
- También puedes usar ***librerías AJAX*** .Por ejemplo, para efectos y galerías de imágenes ahorra bastante tiempo
- Si no es realmente necesario puedes omitir animaciones en Flash (*hacen un poco lenta la carga de la página*)
- Para gestionar la transferencia de archivos puedes utilizar un ***servidor FTP*** http://www.aclogic.com
- También puedes incluir un servidor de correos como ***Argo Mail Server*** http://www.argosoft.com/files/apps/agsmail.exe

6.- ¿Qué es una "Agrupación de conexiones" (connection pooling)?

Es el manejo de una colección de conexiones abiertas a una base de datos de manera que puedan ser reutilizadas al realizar múltiples consultas o actualizaciones.

7.- ¿Qué es un Fichero "MANIFEST.MF"?

Es un archivo espacífico contenido en un archivo JAR (*JavaARchive*); se usa para definir datos relativos a la extensión y al paquete. Puedes ver esto http://gpd.sip.ucm.es/rafa/docencia/programacion/tema1/instalacion.html.

Más información:

- Introducción a Spring Framework: http://www.davidmarco.es/tutoriales/SpringFrameworkMVC.html
- JCreator:http://www.jcreator.com/
- Introducción a Hibernate: http://www.froses.com/Assets/Files/Articles/Hibernate_Introduccion_es.pdf
- JBoss: http://docs.jboss.org/jbossas/getting_started/v4/pdf/startguide.pdf