

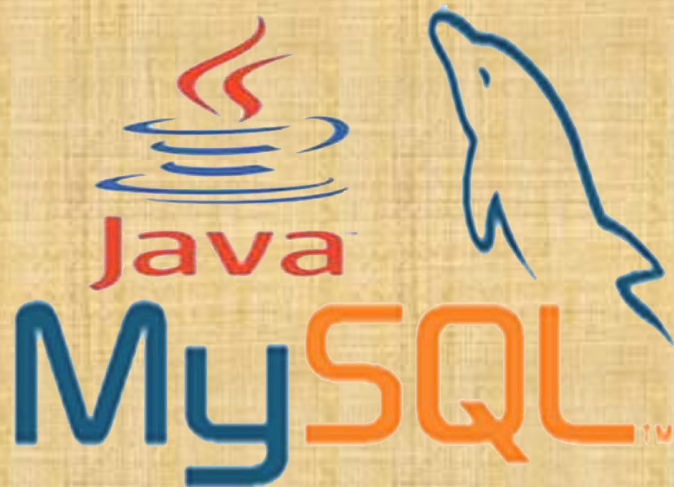
MYSQL Y JAVA



TUTORA: ANGELA JAZMIN MIRANDA FLORES

COMO SE HACE LA CONEXIÓN CON LA BASE DE DATOS

Como habíamos mencionado en sesiones anteriores la comunicación entre java y arduino requería de la importación de su librería “PanamaHitek_Arduino”, en este caso la comunicación de java con la base de datos no es la excepción, ya que esta requiere de “mysql-connector” que es un driver que nos permite hacer la conexión a la base de datos que vamos a manejar.



DRIVER JDBC

Mysql Connector es un driver creado por Mysql AB que te permitirá trabajar con Mysql desde programas escritos en Java. A diferencia de otros drivers, este es de libre distribución, y tiene un buen rendimiento.

En el desarrollo de las últimas versiones se ha incrementado bastante la velocidad del driver, ganando en rapidez así como en eficiencia. El driver soporta resultados de datos "streaming" lo que permite al usuario recoger un gran número de filas sin la necesidad de utilizar un buffer de memoria. El driver implementa un protocolo de paquetes grande que permite enviar filas y campos BLOBs de hasta 2 GigaBytes.



Generally Available (GA) Releases Development Releases

Connector/J 5.1.44

Select Operating System:
Platform Independent

Looking for previous GA versions?

Platform Independent (Architecture Independent), Compressed TAR Archive (mysql-connector-java-5.1.44.tar.gz)	5.1.44	3.3M	Download
MD5: 526d4f3b2b127abcc25b575ece2c54d1 Signature			
Platform Independent (Architecture Independent), ZIP Archive (mysql-connector-java-5.1.44.zip)	5.1.44	3.6M	Download
MD5: e96f0681196a466e90ca887bb9828028 Signature			

i We suggest that you use the MD5 checksums and GnuPG signatures to verify the integrity of the packages you download.

BASE DE DATOS




EJERCICIO DE APLICACIÓN

TUTORA: ANGELA JAZMIN MIRANDA FLORES

BASE DE DATOS

Creamos una tabla con el nombre “datosjava” en nuestra base de datos “sistemadatos”.

 Crear tabla

Nombre:

Número de columnas:

Continuar

BASE DE DATOS

Nombre de la tabla: Add column(s) Continuar

Estructura ⓘ										
Nombre	Tipo ⓘ	Longitud/Valores ⓘ	Predeterminado ⓘ	Cotejamiento	Atributos	Nulo	Índice	A_I	Comentarios	Virtualidad
<input type="text" value="idJ"/> <small>Seleccionar desde las columnas centrales</small>	<input type="text" value="INT"/>	<input type="text" value="10"/>	<input type="text" value="Ninguno"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="PRIMARY"/> <small>PRIMARY</small>	<input checked="" type="checkbox"/>	<input type="text"/>	<input type="text"/>
<input type="text" value="nombreC"/> <small>Seleccionar desde las columnas centrales</small>	<input type="text" value="VARCHAR"/>	<input type="text" value="50"/>	<input type="text" value="Ninguno"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="---"/>	<input type="checkbox"/>	<input type="text"/>	<input type="text"/>
<input type="text" value="notas"/> <small>Seleccionar desde las columnas centrales</small>	<input type="text" value="VARCHAR"/>	<input type="text" value="1000"/>	<input type="text" value="Ninguno"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="---"/>	<input type="checkbox"/>	<input type="text"/>	<input type="text"/>

Comentarios de la tabla:

Cotejamiento:

Motor de almacenamiento:

definición de la PARTICIÓN: ⓘ

Previsualizar SQL Guardar

BASE DE DATOS

Nos debe quedar de la siguiente forma:



The screenshot shows a database management interface with a breadcrumb trail: "Servidor: 127.0.0.1 » Base de datos: sistemadatos » Tabla: datosjava". Below the breadcrumb are several tabs: "Examinar", "Estructura", "SQL", "Buscar", "Insertar", "Exportar", and "Importar". The "Estructura" tab is active, showing two sub-tabs: "Estructura de tabla" and "Vista de relaciones". The "Estructura de tabla" sub-tab displays a table structure with the following columns: #, Nombre, Tipo, Cotejamiento, Atributos, Nulo, Predeterminado, and Extra. The table contains three rows of data:

#	Nombre	Tipo	Cotejamiento	Atributos	Nulo	Predeterminado	Extra
<input type="checkbox"/> 1	idJ 	int(10)			No	Ninguna	AUTO_INCREMENT
<input type="checkbox"/> 2	nombreC	varchar(50)			No	Ninguna	
<input type="checkbox"/> 3	notas	varchar(1000)			No	Ninguna	

JAVA

Creamos un proyecto al cual llamaremos “mysqljava”.

New Java Application

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name:

Project Location:

Project Folder:

☐ Use Dedicated Folder for Storing Libraries

Libraries Folder:

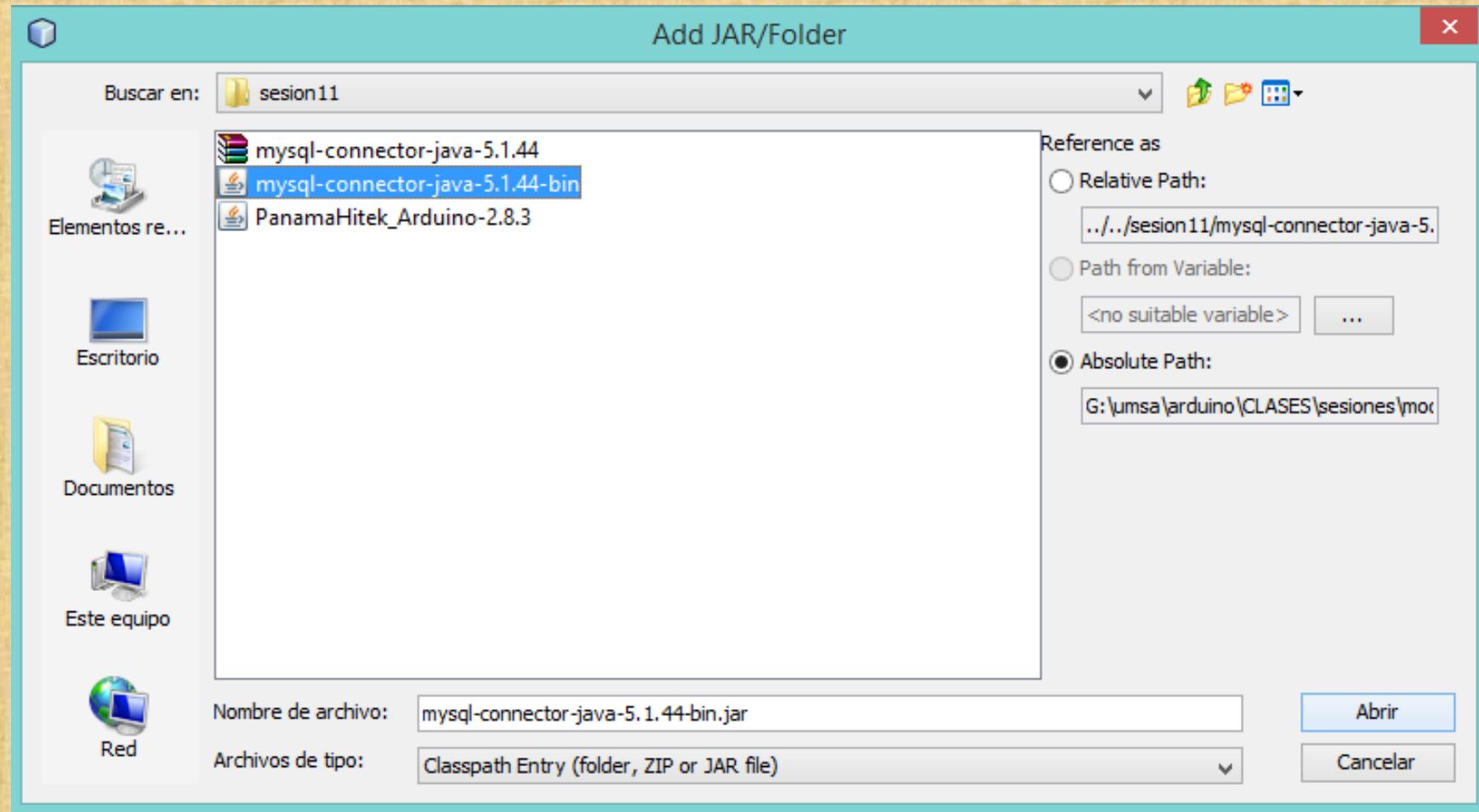
Different users and projects can share the same compilation libraries (see Help for details).

☐ Create Main Class

< Back Next > **Finish** Cancel Help

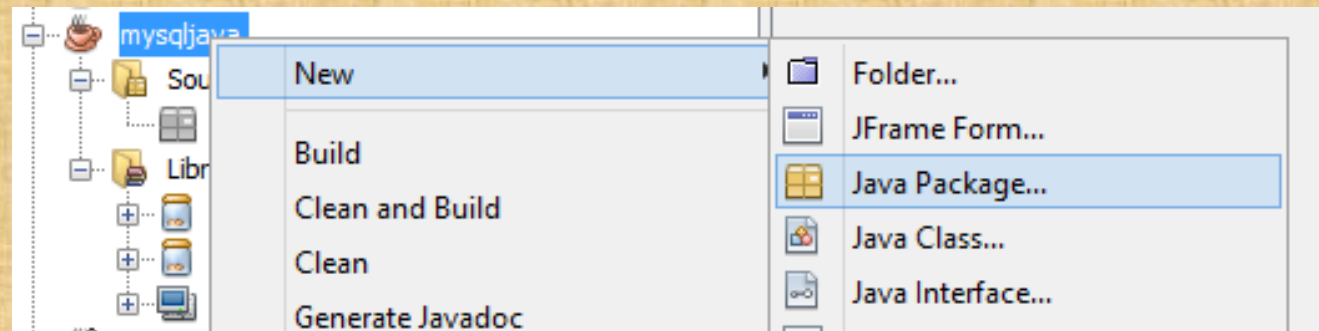
JAVA

Importamos la librería “mysql-connector-java-5.1.44-bin.jar”.

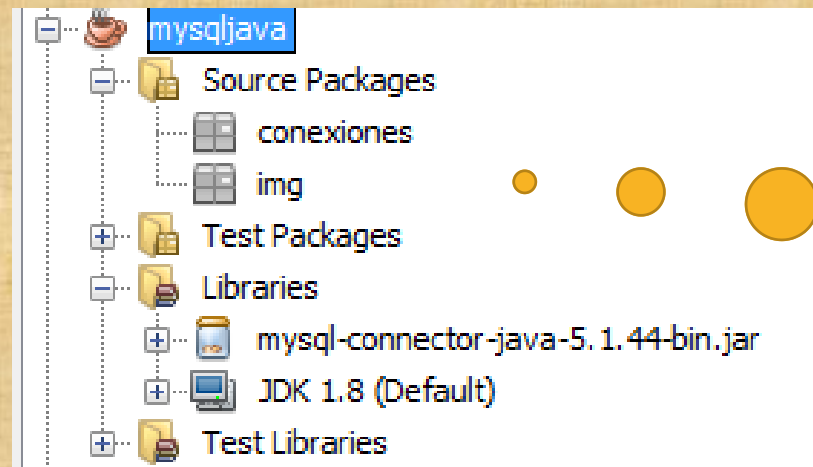


JAVA

Agregamos 2 paquetes dentro nuestro proyecto, una para las imágenes y otra para los archivos java que usaremos.



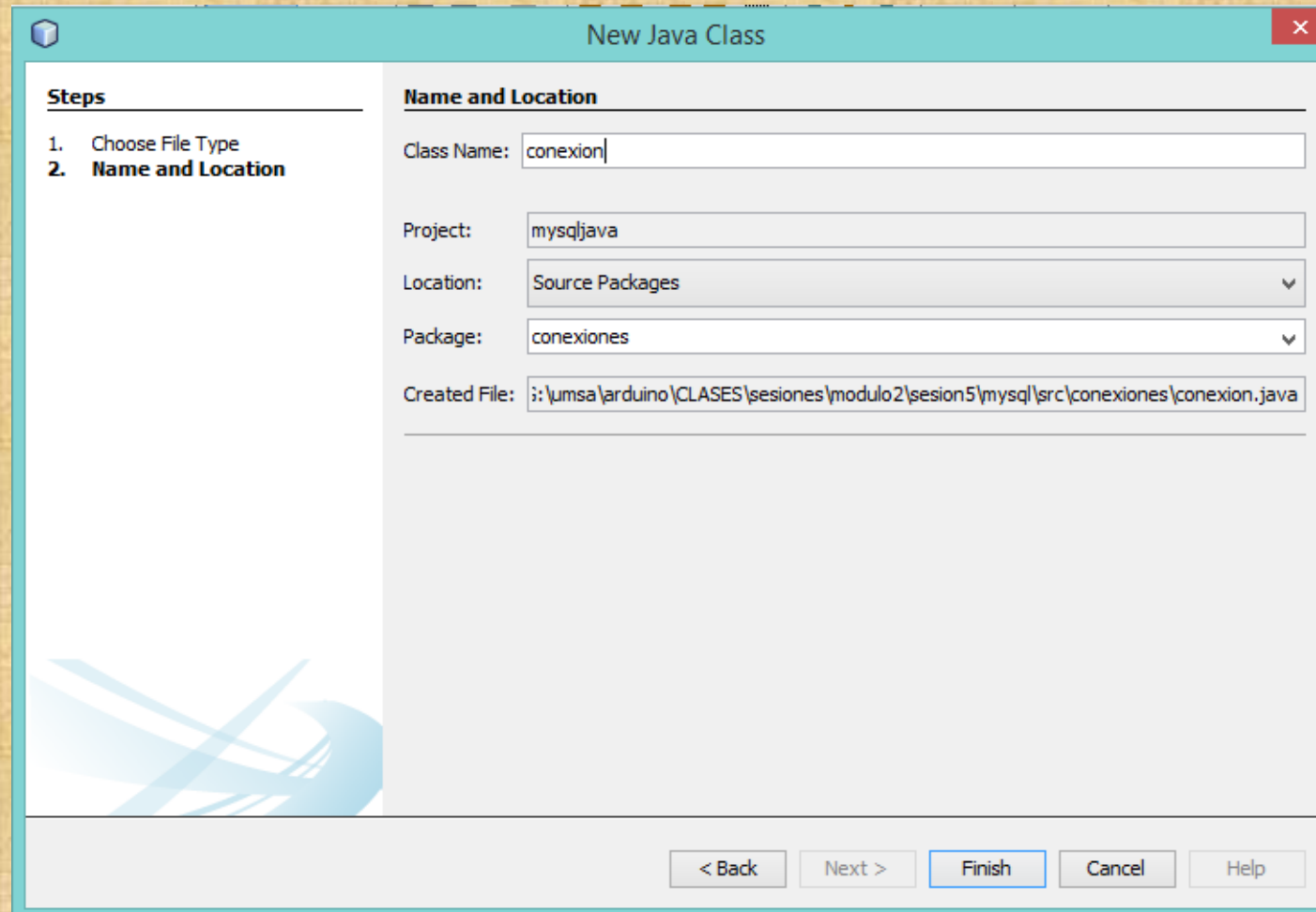
Nos quedara de la siguiente forma:



El paquete "conexiones" contendrá nuestros archivos java y el paquete "img" las imágenes que utilizaremos dentro el proyecto.

JAVA

Creamos un archivo java llamado “conexion”



New Java Class

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel Help

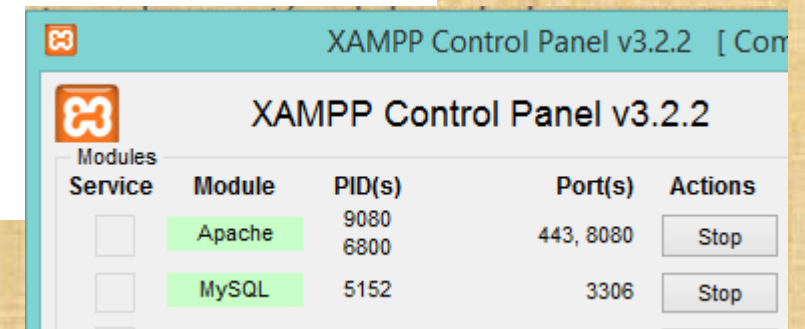
JAVA

Iniciamos la conexión a la base de datos

```
package conexiones;

import com.mysql.jdbc.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.logging.Level;
import java.util.logging.Logger;
import javax.swing.JOptionPane;

public class conexion {
    private static Connection Conec;
    //conexion a la base de datos
    public Connection MySQLConnection(String user, String pass, String db_name) {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            Conec = (Connection) DriverManager.getConnection("jdbc:mysql://localhost:3306/" + db_name, user, pass);
            System.out.println("Se ha iniciado la conexión con el servidor de forma exitosa");
        } catch (ClassNotFoundException ex) {
            Logger.getLogger(conexion.class.getName()).log(Level.SEVERE, null, ex);
        } catch (SQLException ex) {
            Logger.getLogger(conexion.class.getName()).log(Level.SEVERE, null, ex);
        }
        return Conec;
    }
}
```



TUTORA: ANGELA JAZMIN MIRANDA FLORES

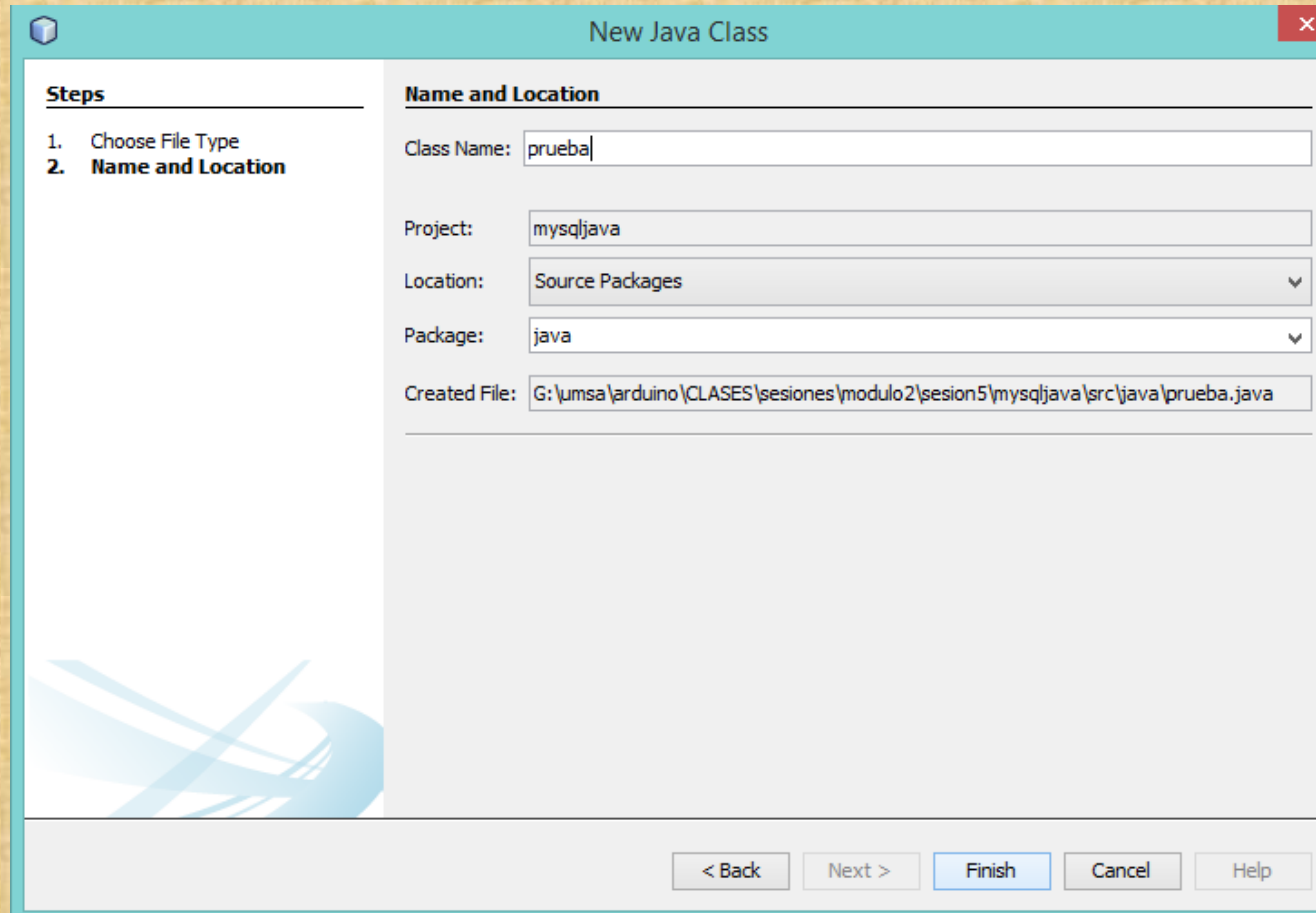
JAVA

Cerramos la conexión a nuestra base de datos.

```
//cierra la conexion a la base de datos
public void closeConnection() {
    try {
        Conec.close();
        System.out.println("Se ha finalizado la conexión con el servidor");
    } catch (SQLException ex) {
        Logger.getLogger(conexion.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```


JAVA

Creamos un archivo java para probar las conexiones que realizaremos.



The screenshot shows the 'New Java Class' dialog box with the following fields and values:

Field	Value
Class Name:	prueba
Project:	mysqljava
Location:	Source Packages
Package:	java
Created File:	G:\umsa\arduino\CLASES\sесiones\modulo2\sesion5\mysqljava\src\java\prueba.java

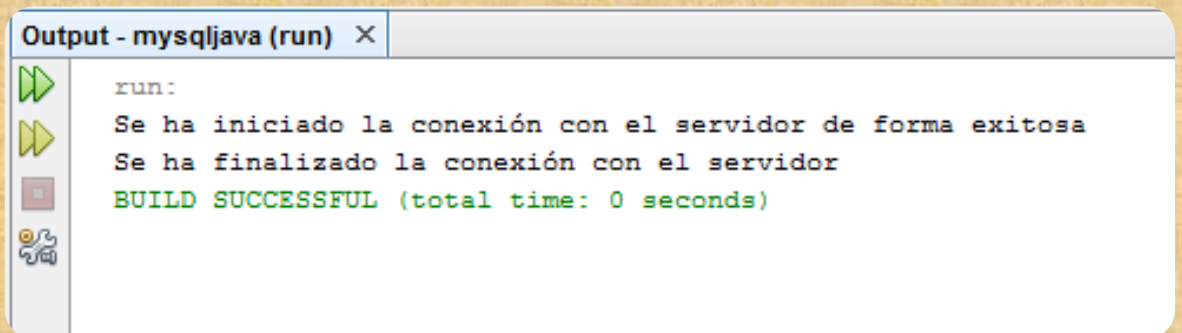
At the bottom of the dialog, there are five buttons: '< Back', 'Next >', 'Finish' (highlighted in blue), 'Cancel', and 'Help'.

JAVA

Hacemos la conexión a nuestra base de datos y la cerramos

```
package conexiones;

public class prueba {
    public static void main(String[] args) {
        conexion c=new conexion();
        c.MySQLConnection("root","", "sistemadatos");
        c.closeConnection();
    }
}
```



The screenshot shows an IDE output window titled "Output - mysqljava (run)". It contains the following text:

```
run:
Se ha iniciado la conexión con el servidor de forma exitosa
Se ha finalizado la conexión con el servidor
BUILD SUCCESSFUL (total time: 0 seconds)
```

On the left side of the output window, there are icons for running (a green play button), stepping through (a green play button with a dot), and debugging (a red bug icon).

JAVA

Para leer los datos de nuestra tabla añadimos a nuestra clase conexión el siguiente método:

```
// obtiene las partituras para el buzzer
public void getmusica(String id){
    try {
        String Query = "SELECT * FROM datosjava WHERE idJ=" + id ;
        Statement st = Conexion.createStatement();
        java.sql.ResultSet resultSet;
        resultSet = st.executeQuery(Query);

        while (resultSet.next()) {
            System.out.println("ID: " + resultSet.getString("idJ") + "\n "
                               + "nombre de la cancion: " + resultSet.getString("nombreC") + "\n notas: " + resultSet.getString("notas"));
        }

    } catch (SQLException ex) {
        JOptionPane.showMessageDialog(null, "Error al obtener los datos");
    }
}
```

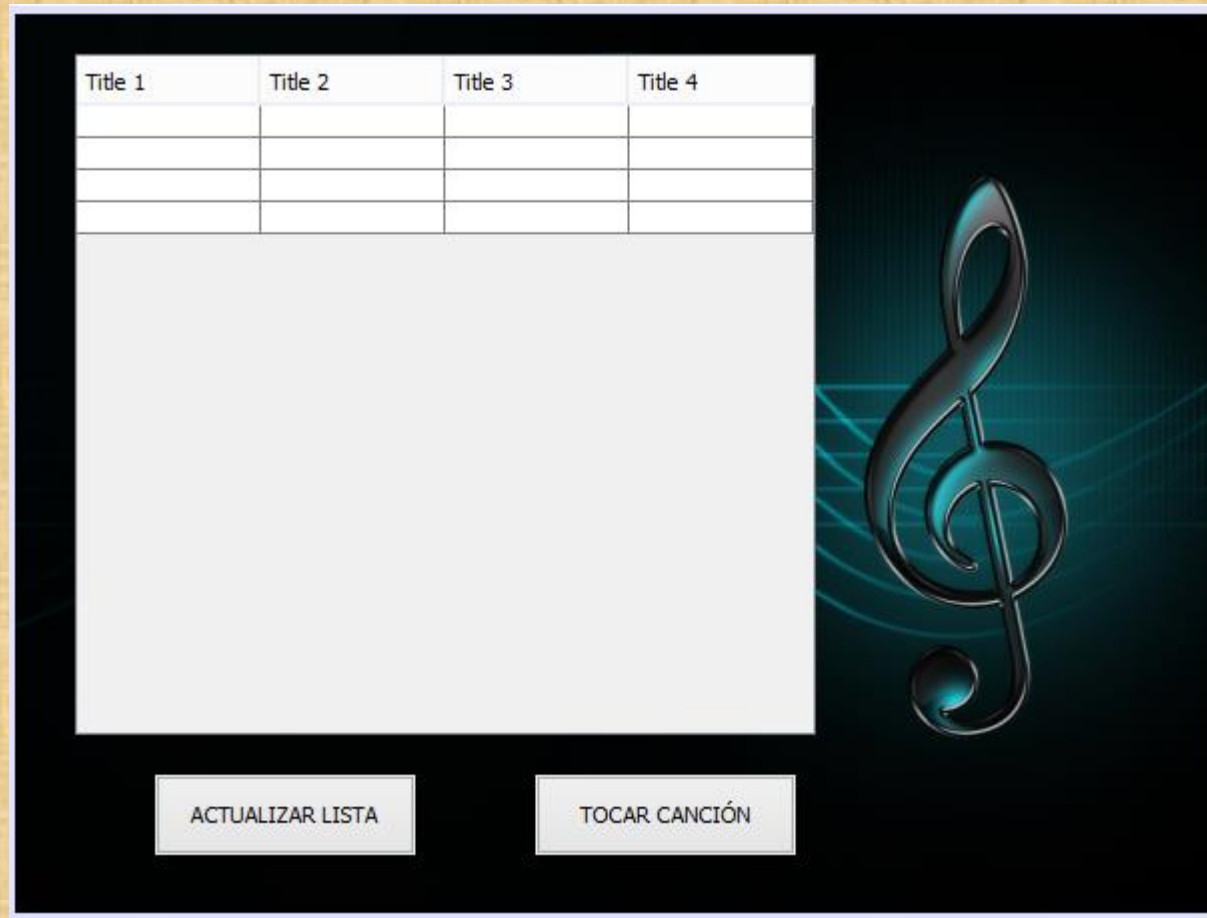
Dentro nuestra clase prueba llamamos a nuestro método creado.

```
public class prueba {
    public static void main(String[] args) {
        conexion c=new conexion();
        c.MySqlConnection("root","","sistemadatos");
        c.getmusica("1");
        c.closeConnection();
    }
}
```

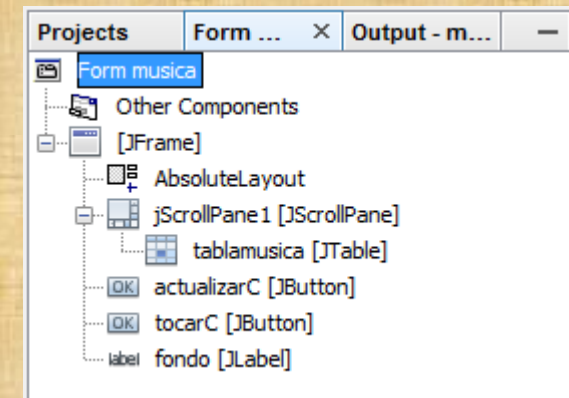
JAVA

Creamos un JFrame al cual lo llamaremos “musica”.

Nos debe quedar de la siguiente forma:

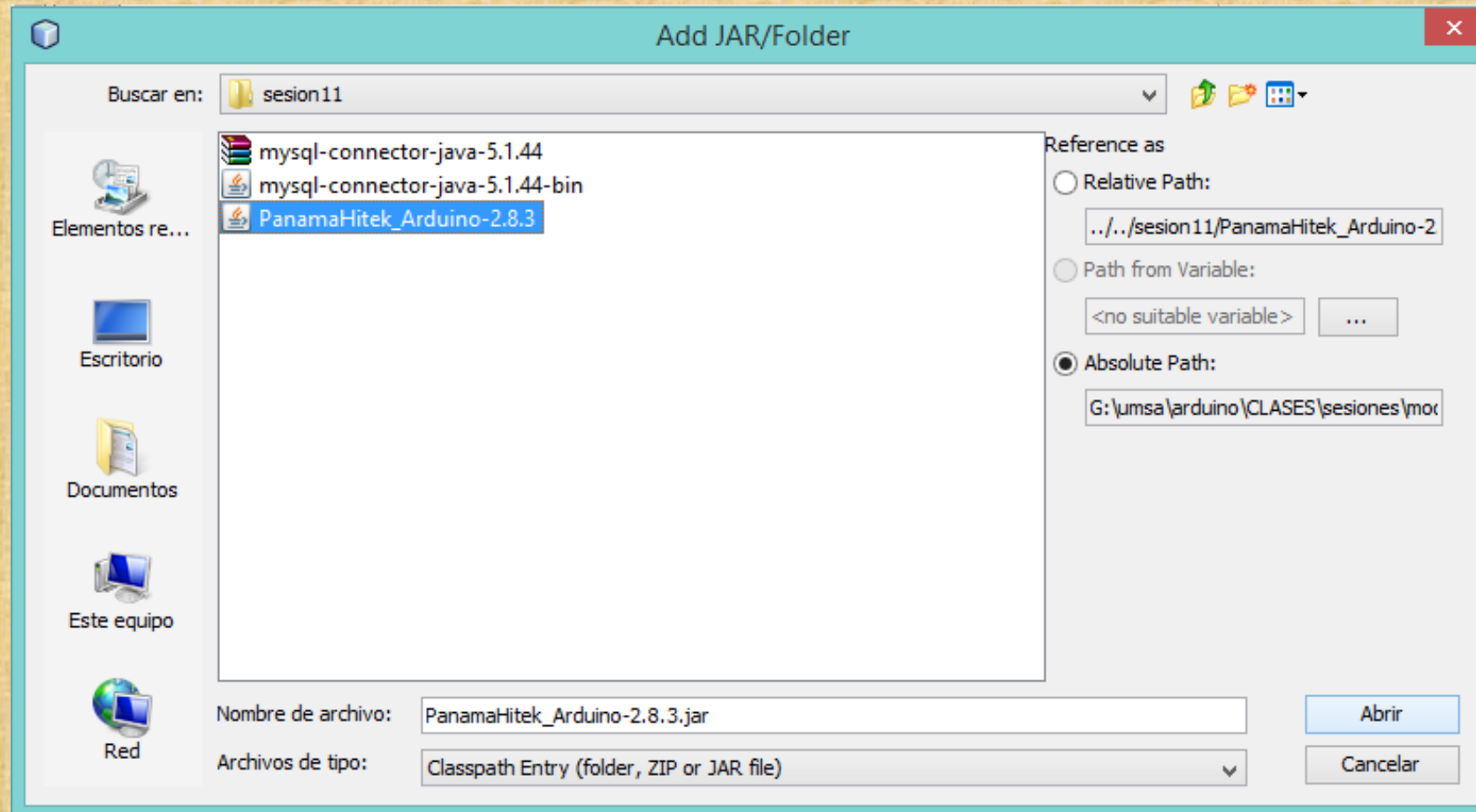


Los componentes que utilizaremos son:



JAVA

Importamos la librería “PanamaHitek_Arduino-2.8.3.jar”.



JAVA

Definimos nuestras variables y establecemos la conexión con la base de datos.

```
public class musica extends javax.swing.JFrame {
    conexion con=new conexion();
    Connection cn=con.MySQLConnection("root","", "sistemadatos");
    String cad;
    PanamaHitek_Arduino arduino=new PanamaHitek_Arduino();
    public musica() {
        initComponents();
        try{
            arduino.arduinoTX("COM14",9600);

        }catch(Exception ex){
            Logger.getLogger(musica.class.getName()).log(Level.SEVERE, null,ex);
        }
    }
}
```


JAVA

```
void mostrarcancion() {  
    try {  
        DefaultTableModel modelo=new DefaultTableModel();  
        modelo.addColumn("id");  
        modelo.addColumn("nombre de la cancion");  
        modelo.addColumn("notas");  
        tablamusica.setModel(modelo);  
        String sql="SELECT * FROM datosjava";  
        String datos[]=new String[3];  
        Statement st=cn.createStatement();  
        ResultSet re=st.executeQuery(sql);  
        while(re.next()){  
            datos[0]=re.getString(1);  
            datos[1]=re.getString(2);  
            datos[2]=re.getString(3);  
            modelo.addRow(datos);  
        }  
        tablamusica.setModel(modelo);  
    } catch (SQLException ex) {  
        Logger.getLogger(musica.class.getName()).log(Level.SEVERE, null, ex);  
    }  
}
```

JAVA

```
void tocamelodia(String cad) {  
    switch(cad){  
case "1":  
    try{  
        arduino.sendData("a");  
    }catch(Exception ex){  
        Logger.getLogger(musica.class.getName()).log(Level.SEVERE, null,ex);  
    }  
    break;  
case "2":  
    try{  
        arduino.sendData("b");  
    }catch(Exception ex){  
        Logger.getLogger(musica.class.getName()).log(Level.SEVERE, null,ex);  
    }  
    break;  
case "3":  
    try{  
        arduino.sendData("c");  
    }catch(Exception ex){  
        Logger.getLogger(musica.class.getName()).log(Level.SEVERE, null,ex);  
    }  
    break;  
case "4":  
    try{  
        arduino.sendData("d");  
    }catch(Exception ex){  
        Logger.getLogger(musica.class.getName()).log(Level.SEVERE, null,ex);  
    }  
    break;  
}
```

```
case "5":  
    try{  
        arduino.sendData("e");  
    }catch(Exception ex){  
        Logger.getLogger(musica.class.getName()).log(Level.SEVERE, null,ex);  
    }  
    break;  
case "6":  
    try{  
        arduino.sendData("f");  
    }catch(Exception ex){  
        Logger.getLogger(musica.class.getName()).log(Level.SEVERE, null,ex);  
    }  
    break;  
case "7":  
    try{  
        arduino.sendData("g");  
    }catch(Exception ex){  
        Logger.getLogger(musica.class.getName()).log(Level.SEVERE, null,ex);  
    }  
    break;  
case "8":  
    try{  
        arduino.sendData("h");  
    }catch(Exception ex){  
        Logger.getLogger(musica.class.getName()).log(Level.SEVERE, null,ex);  
    }  
    break;
```

```
case "9":  
    try{  
        arduino.sendData("i");  
    }catch(Exception ex){  
        Logger.getLogger(musica.class.getName()).log(Level.SEVERE, null,ex);  
    }  
    break;  
case "10":  
    try{  
        arduino.sendData("j");  
    }catch(Exception ex){  
        Logger.getLogger(musica.class.getName()).log(Level.SEVERE, null,ex);  
    }  
    break;  
}
```

JAVA

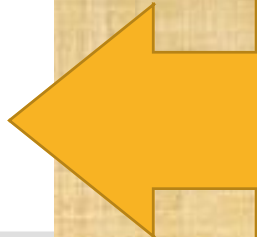
```
@SuppressWarnings("unchecked")
Generated Code

private void actualizarActionPerformed(java.awt.event.ActionEvent evt) {
    mostrarcancion();
}

private void tocarActionPerformed(java.awt.event.ActionEvent evt) {
    int fila=tablamusica.getSelectedRow();
    if(fila>=0){
        cad=tablamusica.getValueAt(fila, 0).toString();
        tocamelodia(cad);
    }else{
        JOptionPane.showMessageDialog(null, "fila no seleccionada");
    }
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new musica().setVisible(true);
        }
    });
}
```



Nos permite seleccionar una fila de la tabla para que toque la melodía que se encuentra almacenada

ARDUINO

Añadimos el archivo pitches.h a nuestro código.

```
musik pitches.h
#define NOTE_B0 31
#define NOTE_C1 33
#define NOTE_CS1 35
#define NOTE_D1 37
#define NOTE_DS1 39
#define NOTE_E1 41
#define NOTE_F1 44
#define NOTE_FS1 46
#define NOTE_G1 49
#define NOTE_GS1 52
#define NOTE_A1 55
#define NOTE_AS1 58
#define NOTE_B1 62
#define NOTE_C2 65
#define NOTE_CS2 69
#define NOTE_D2 73
#define NOTE_DS2 78
#define NOTE_E2 82
#define NOTE_F2 87
#define NOTE_FS2 93
#define NOTE_G2 96
```



No olvidar que este archivo maneja la notación anglosajona.

ARDUINO

Definimos nuestras variables

```
#include "pitches.h";  
int buzzer=9;  
char estado= ' ' ;  
int duracion=0;  
void setup() {  
    Serial.begin(9600);  
    pinMode(buzzer, OUTPUT);  
}
```


ARDUINO

```
void loop() {  
  if(Serial.available()>0){  
    estado=Serial.read();  
    Serial.print(estado);  
    switch(estado){  
      case 'a':  
        duracion=Serial.read();  
        tone(buzzer,NOTE_G2,250);  
        break;  
      case 'b':  
        tone(buzzer,NOTE_G3,250);  
        break;  
      case 'c':  
        tone(buzzer,NOTE_G4,250);  
        break;  
      case 'd':  
        tone(buzzer,NOTE_B4,250);  
        break;  
      case 'e':  
        tone(buzzer,NOTE_D5,250);  
        break;  
      case 'f':  
        tone(buzzer,NOTE_F5,250);  
        break;  
    }
```

```
      case 'g':  
        tone(buzzer,NOTE_G5,250);  
        break;  
      case 'h':  
        tone(buzzer,NOTE_A5,250);  
        break;  
      case 'i':  
        melodia();  
        break;  
      case 'j':  
        simpson();  
        break;  
    }  
  }  
}
```


ARDUINO

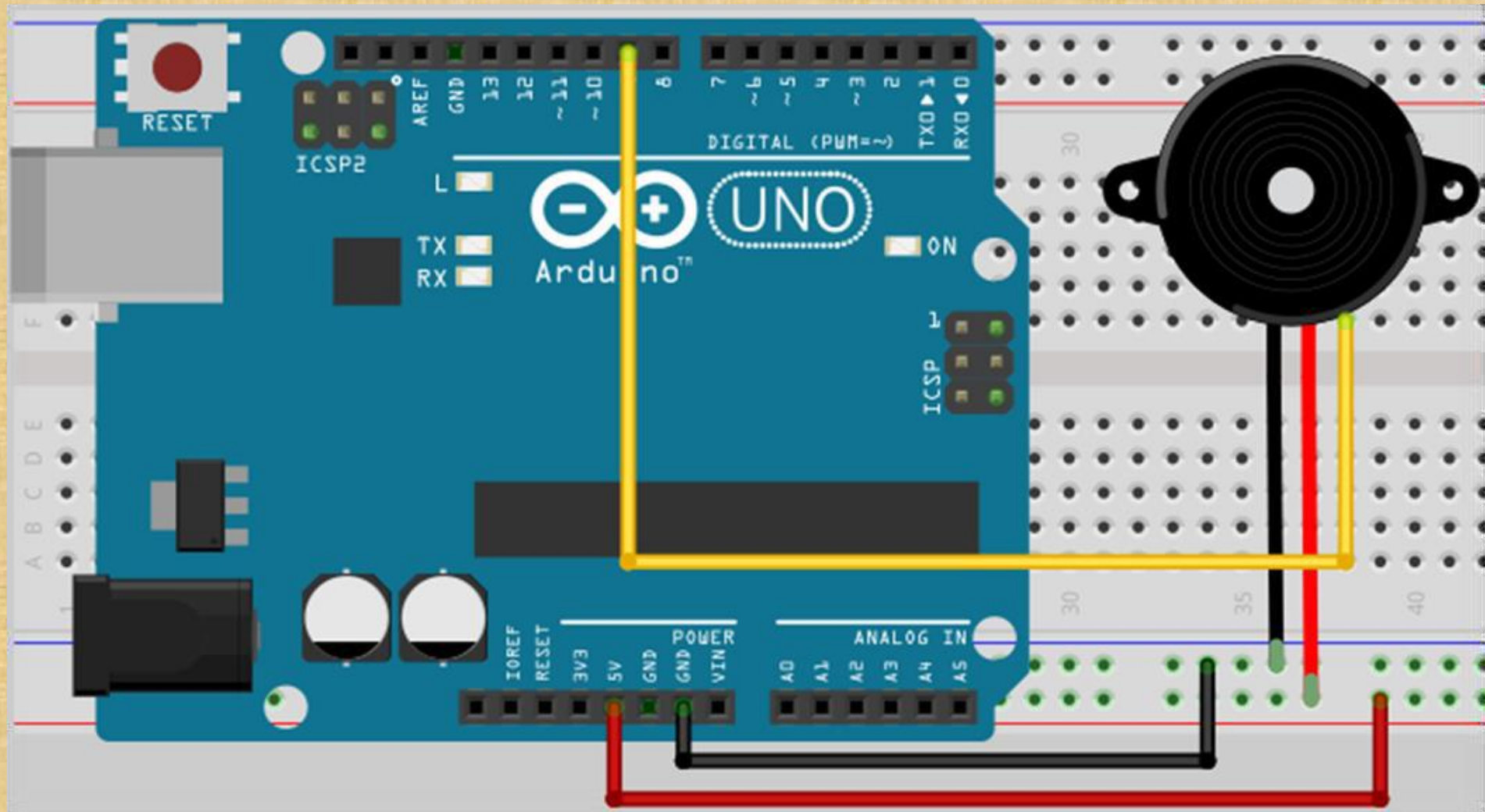
```
void melodia() {  
  int melody[] = {NOTE_C4, NOTE_G3, NOTE_G3, NOTE_A3, NOTE_G3, 0, NOTE_B3, NOTE_C4};  
  int noteDurations[] = { 4, 8, 8, 4, 4, 4, 4, 4};  
  for (int thisNote = 0; thisNote < 8; thisNote++) {  
    int noteDuration = 1000 / noteDurations[thisNote];  
    tone(buzzer, melody[thisNote], noteDuration);  
    int pauseBetweenNotes = noteDuration * 1.30;  
    delay(pauseBetweenNotes);  
    noTone(buzzer); //detiene la reproduccion de los tonos  
  }  
}
```

ARDUINO

Tono de la intro de la serie “Los Simpsons”.

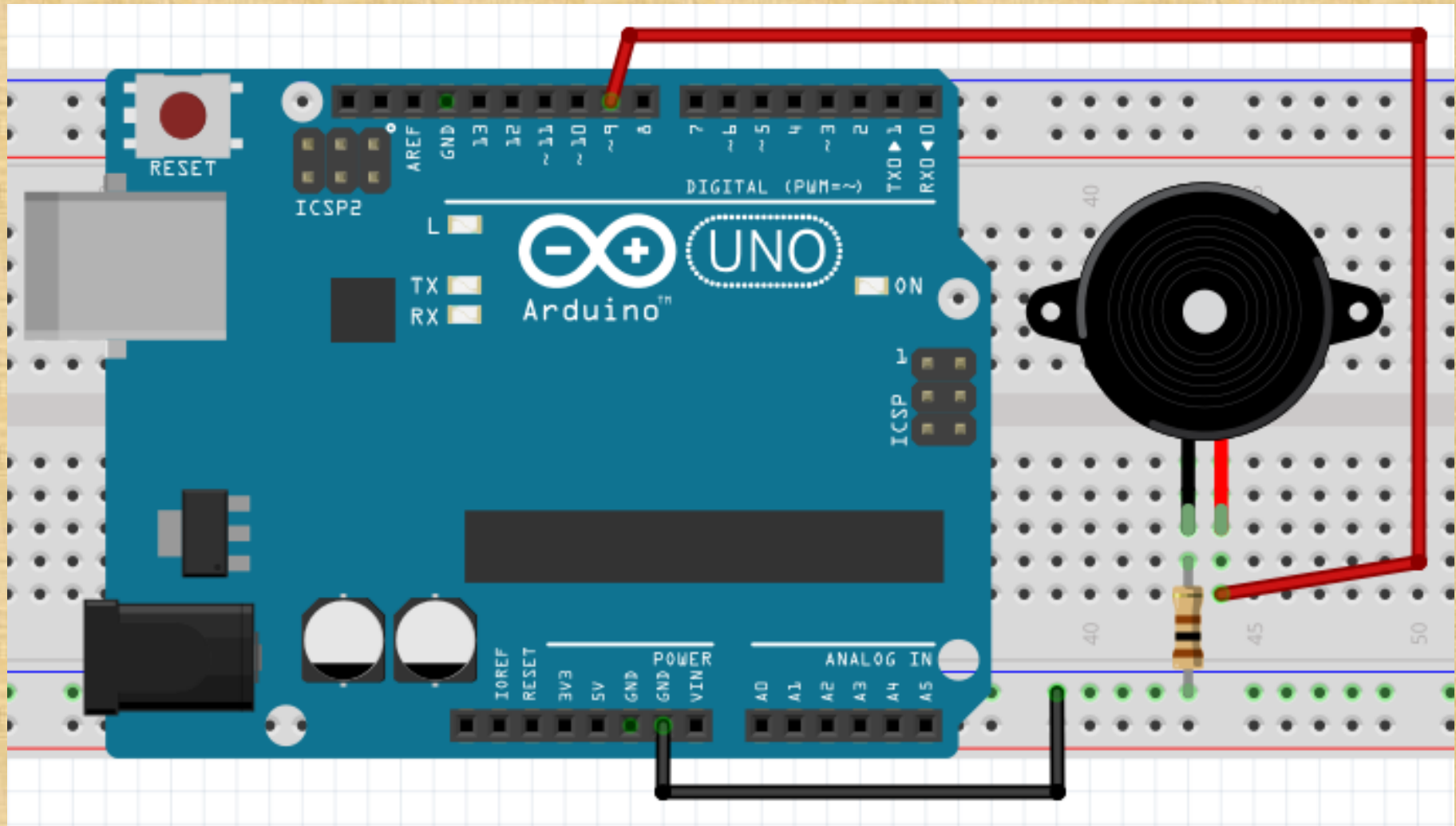
```
void simpson(){
  int melody[] = {NOTE_C6, NOTE_E6, NOTE_FS6, NOTE_A7, NOTE_G6, NOTE_E6, NOTE_C6, NOTE_A6, 0, NOTE_FS5, NOTE_FS5, NOTE_FS5, NOTE_G5,
0, NOTE_FS5, NOTE_FS5, NOTE_FS5, NOTE_G5, NOTE_AS5, NOTE_C6, NOTE_C6, NOTE_C6, NOTE_C6};
  int noteDurations[] = {4, 4, 4, 4, 4, 4, 4, 4, 8, 4, 4, 4, 4, 8, 4, 4, 4, 4, 4, 4, 4, 4};
  for (int thisNote = 0; thisNote < 23; thisNote++) {
    int noteDuration = 1000 / noteDurations[thisNote];
    tone(buzzer, melody[thisNote], noteDuration);
    int pauseBetweenNotes = noteDuration * 1.30;
    delay(pauseBetweenNotes);
  }
}
```

CONEXIÓN



TUTORA: ANGELA JAZMIN MIRANDA FLORES

CONEXIÓN



MYSQL Y LAS GRÁFICAS

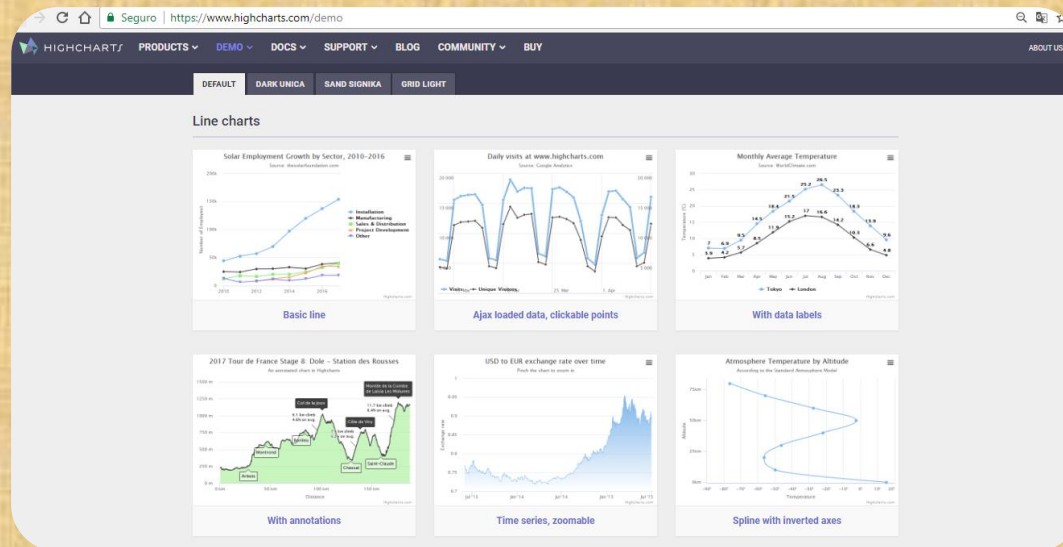
TUTORA: ANGELA JAZMIN MIRANDA FLORES

HIGHCHARTS

HighCharts es una librería escrita en Javascript que permite la creación de gráficas. La librería ofrece un método fácil e interactivo para insertar graficas a una página web.

La librería es compatible con todos los navegadores modernos incluyendo iPhone/iPad e Internet Explorer desde su versión 6.

No es comercial, no se necesita el permiso de los autores para su implementación en sitios web personales o sin fines de lucro.



PHP

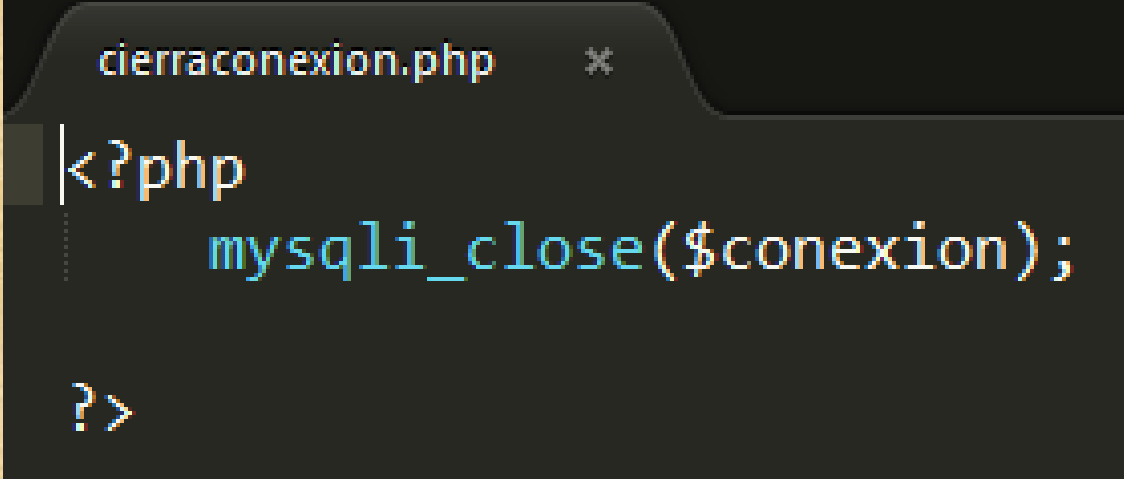
RECORDEMOS NUESTRO ARCHIVO DE CONEXIÓN EN PHP

```
conexion.php
<?php
$user="root";
$host="localhost";
$pw="";
$db="sistemadatos";
$conexion= new mysqli($host,$user,$pw,$
    db);
if($conexion->connect_errno){
echo " no conectado";

exit();
}
?>
```

PHP

RECORDEMOS EL ARCHIVO CIERRACONEXION DE LA ANTERIOR SESIÓN

A screenshot of a code editor window with a dark theme. The title bar at the top shows the filename 'cierraconexion.php' and a close button. The code is written in PHP and is used to close a MySQL connection. It consists of two lines: the opening PHP tag and the 'mysqli_close' function call, followed by the closing PHP tag on a new line.

```
cierraconexion.php ✕  
<?php  
    mysqli_close($conexion);  
  
?>
```

EJERCICIO DE APLICACIÓN

TUTORA: ANGELA JAZMIN MIRANDA FLORES

PHP

Para hacer una grafica como la siguiente usaremos una plantilla de **HIGHCHARTS**



GRAFICA.PHP

La consulta nos devuelve el ultimo valor de la tabla en la base de datos

```
grafica.php x
<?php
function datoagua(){
include "conexion.php";
$query="SELECT *
FROM datoshield
ORDER BY fechas DESC
LIMIT 1";

$resp = mysqli_query($conexion,$query);
while ($consulta = mysqli_fetch_array($resp)) {
echo "valor= ".$consulta['datos'];
}
include ("cierraconexion.php");
}

?>
```


GRAFICA.PHP

```
<html>
<head>
<META HTTP-EQUIV="REFRESH" CONTENT="30;URL=grafica.php">
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
<script type="text/javascript">
    google.charts.load('current', {'packages':['gauge']});
    google.charts.setOnLoadCallback(drawChart);

    function drawChart() {

        var data = google.visualization.arrayToDataTable([
            ['Label', 'Value'],
            ['nivel de agua', <?php datoagua();?>],

        ]);

        var options = {
            width: 1000, height: 620,
            redFrom: 0, redTo: 20,
            yellowFrom:20, yellowTo: 40,
            greenFrom:40, greenTo: 100,
            minorTicks: 5
        };
    }
}
```

GRAFICA.PHP

```
var chart = new google.visualization.Gauge(document.getElementById('chart_div'));

chart.draw(data, options);

setInterval(function() {

    data.setValue(0, 1, <?php datoagua();?>);
    chart.draw(data, options);
}, 1000);

}
</script>
</head>
<body>
    <div id="chart_div" style="width: 400px; height: 120px;"></div>
</body>
</html>
```

GRÁFICA

