# Manual de configuración de las librerías GiovynetDriver y RXTX

## Contenido

# Pasos para instalar la Librería RS232 o GiovynetDriver

## 1.FUENTES WEB PARA LA LIBRERÍA RS232

http://www.giovynet.com/giovynetDriver_es.html


1.1-Descargar la librería en escritorio:

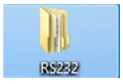
**Nota:** Se debe escoger la librería dependiendo de las características de nuestra computadora, sí es de 32 bit o 64 bit.

1.2-Descomprimir la librería en escritorio, procedemos a crear nuestro proyecto en java

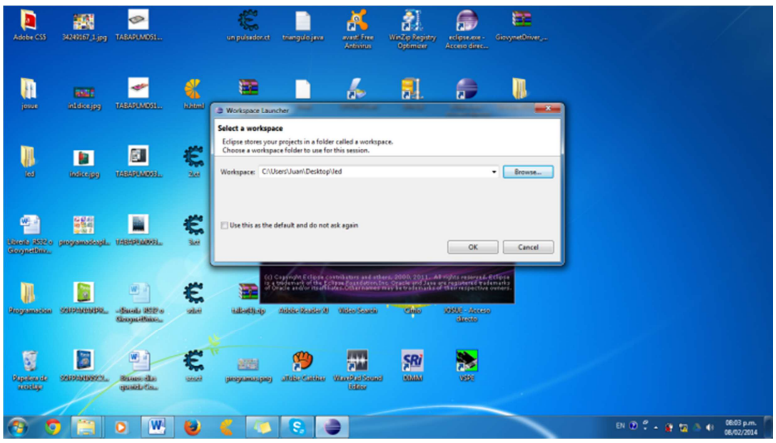
## 2.INSTALACIÓN DE LA LIBRERÍA RS232.

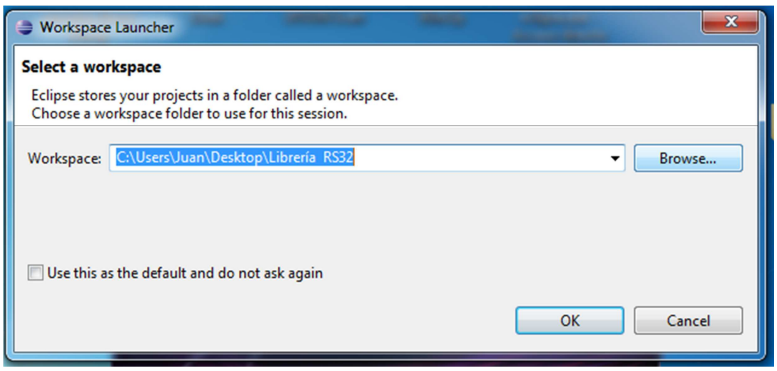
2.1- Crear una carpeta en escritorio

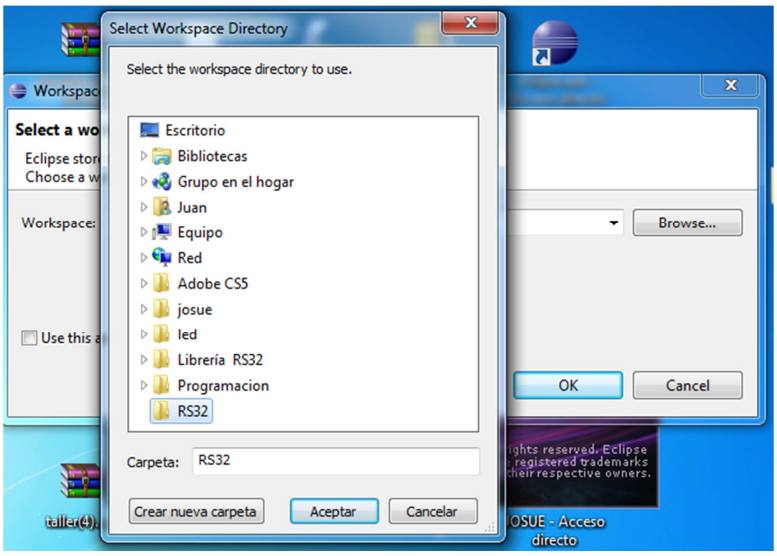
Nombre: RS232




2.1-Ejecutar java y buscar la carpeta creada anteriormente(“RS232”).



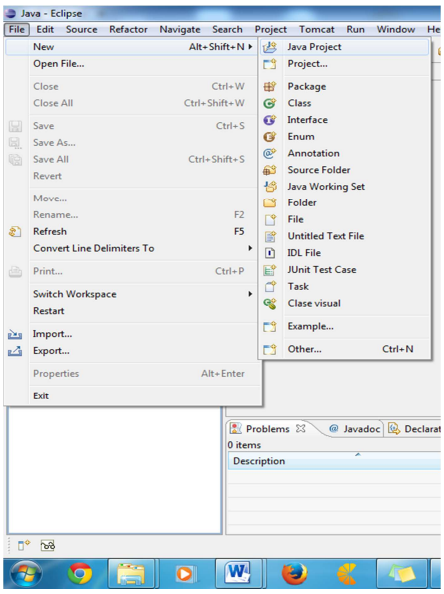


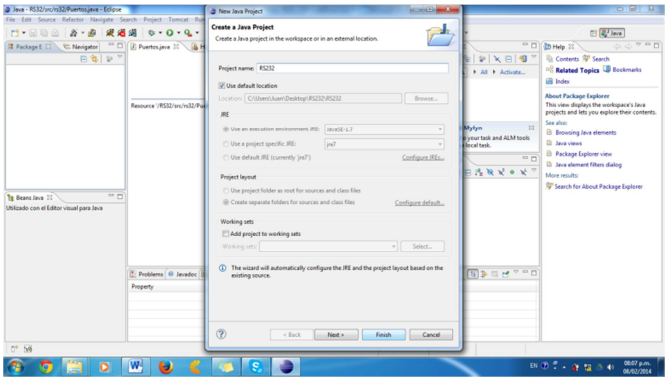

Dar clic en aceptar y Ok.

2.2-Luego buscamos en el menú de herramientas File y damos clic y seleccionamos New

2.3-Dentro New escogemos: Java Project.



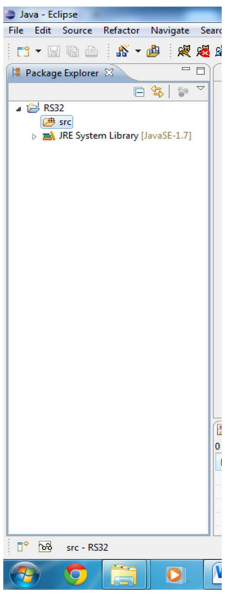

2.4-Ponemos nombre a muestro proyecto: Project name por RS232

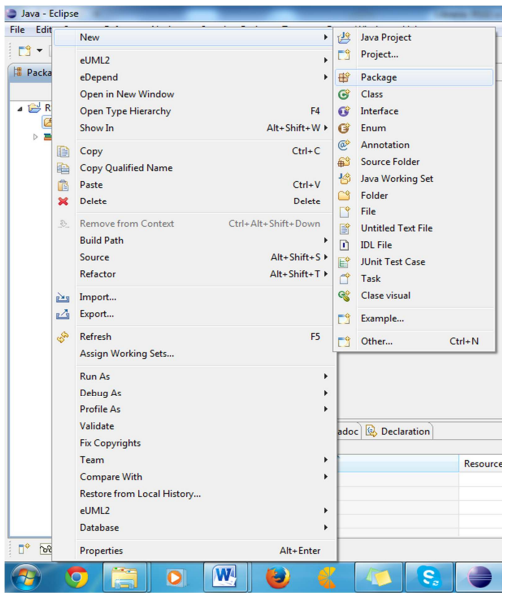



Dar clic en Finish

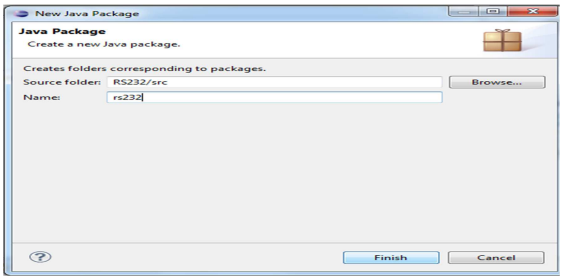
2.5-Damos clic derecho sobre la carpeta creada en java

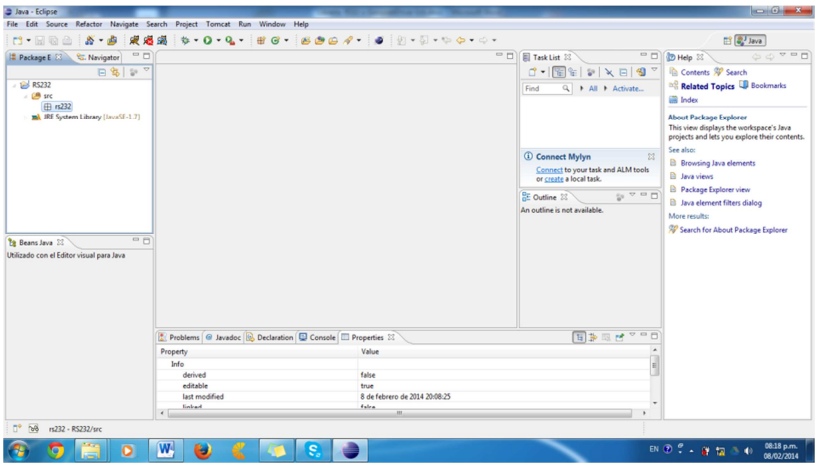
2.6-Escojemos New y dentro de este escogemos Package



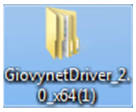


2.7-Damos un  nombre a nuestro Package  en Name:rs232 y clic en Finish





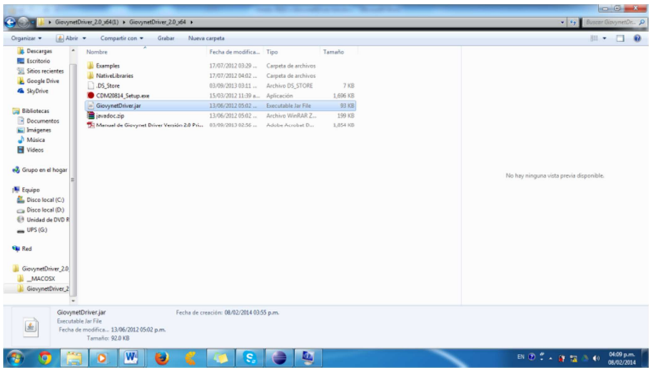
2.9- Entramos  la librería descargada  (GiovynetDriver_2.0_x64)




Se  encuentra en donde descomprimimos la librería

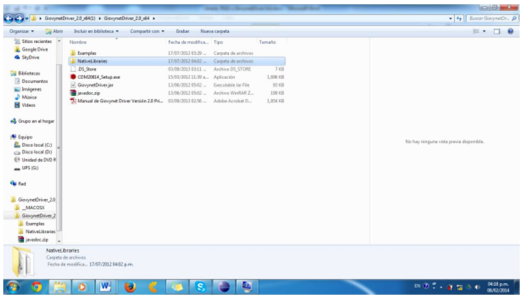
2.10 – Dentro abra dos cartepas en las que entraremos en  la de nombre “GiovynetDriver_2.0_x64”

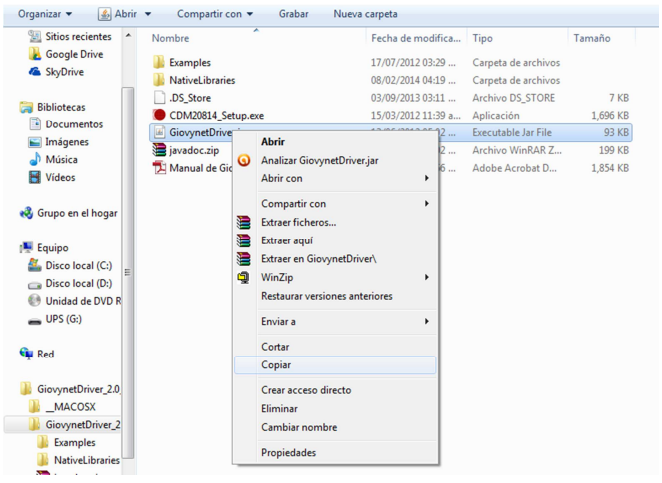
2.11-Dentro de esta abra un archivo de nombre  “GiovynetDriver.jar”

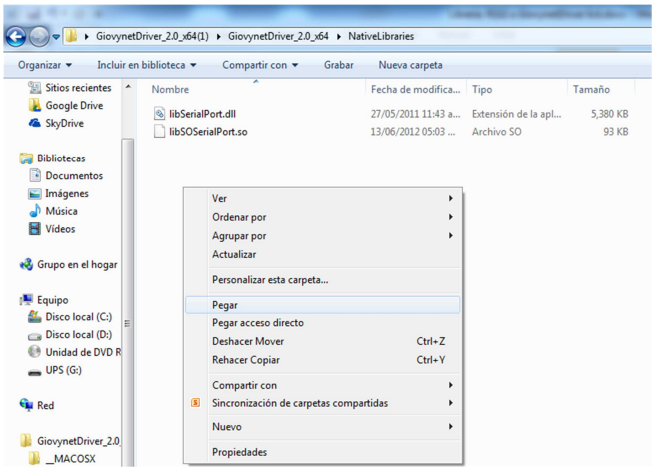


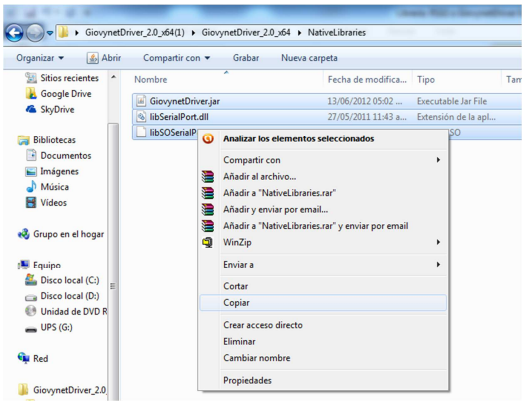
Copiamos el archivo .

2.12-Entramos en la carpeta “NativeLibraries” y pegamos en el el archivo “GiovynetDriver.jar”  y  escogemos todos los archivos y los copiamos







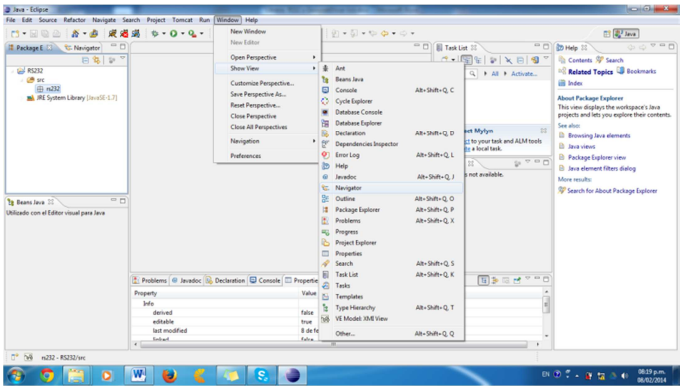

Los Archivos copiados seran :

libSOSerialPort.so

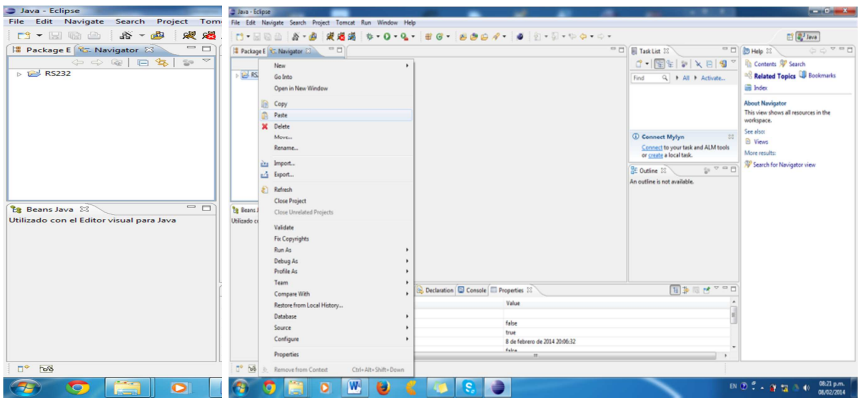
GiovynetDriver.jar

libSerialPort.dll

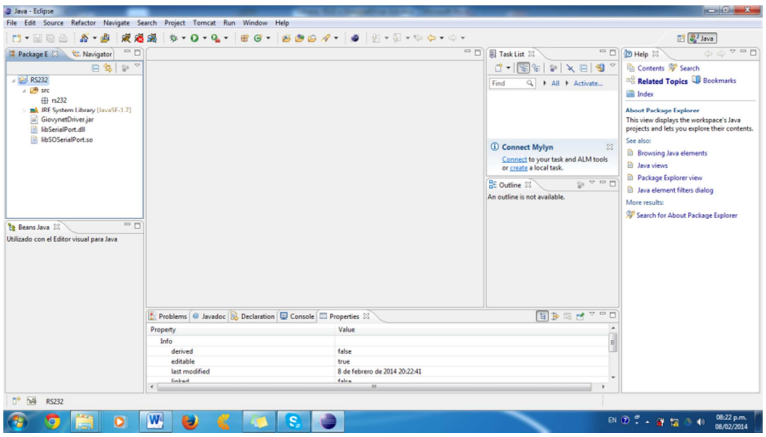
2.13 Bamos a nuestro proyecto en java ,la barra de herramientas escogemos Windows,dentro escogemos Show View y escogemos Navigator



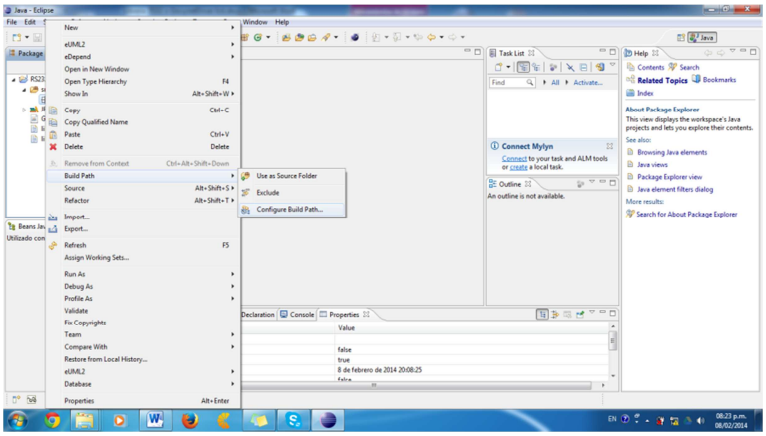
2.14 Se abrira el navegador en este escogemos la carpeta RS32 y damos clic derecho pegar



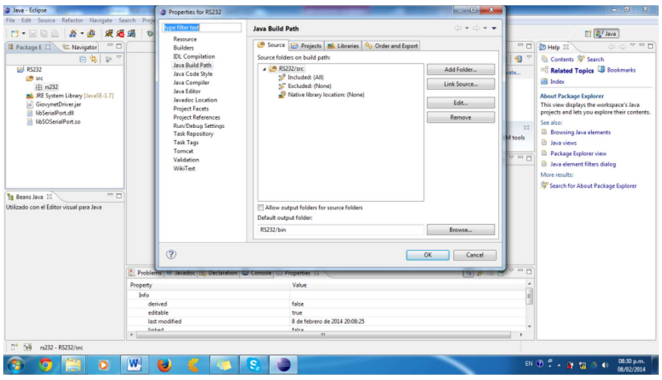
2.15 Regresamos a Package Explorer y veremos nuestra librería en el proyecto

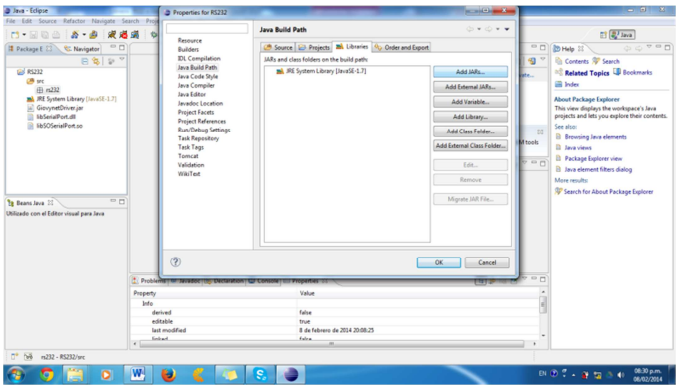


2.16-Damos clic derecho sobre alguna parte de nuestro Package Explorer bamos a Build Path, escogemos Configure Build Path.

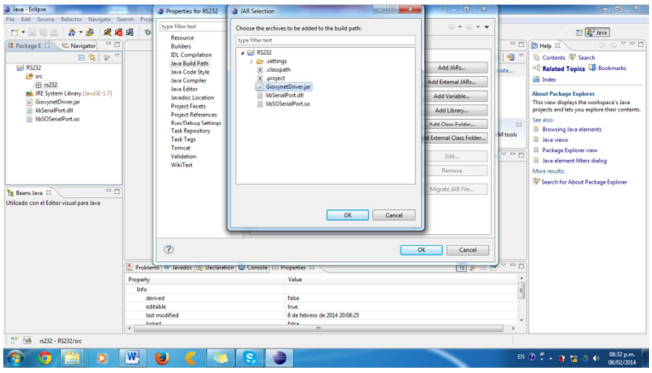


2.17 Hay entraremos en la pestaña Libraries.





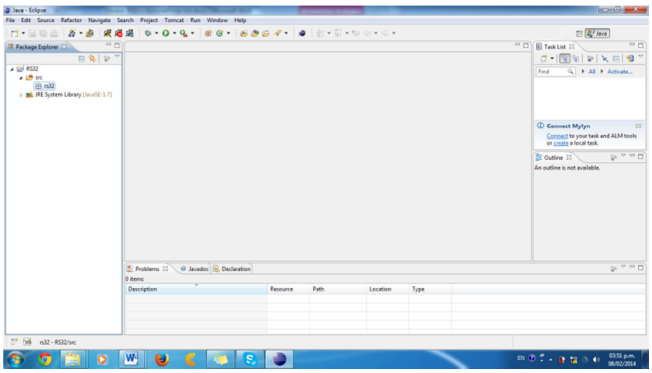
2.18 Dar clic en el boton “Add JARs..”, desplegamos el contenido de la carpeta RS32 y escogemos el archivo “GiovynetDriver.jar” y damos clic Ok y Ok

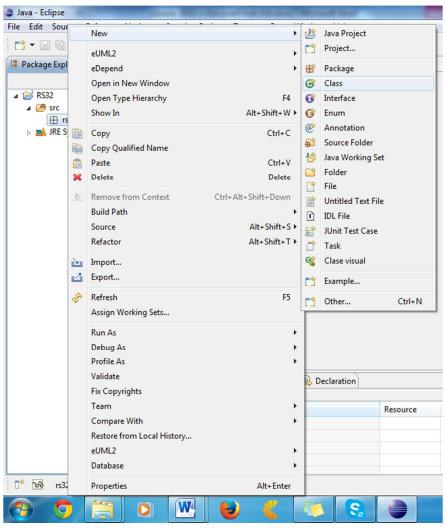


2.19 con esto ya estara nuestra librería instalada en nuestro proyecto en java.

## 3-CÓDIGO PARA EL MANEJO DE LOS PUERTOS SERIAL MEDIANTE JAVA (PUERTOS LIBRES)

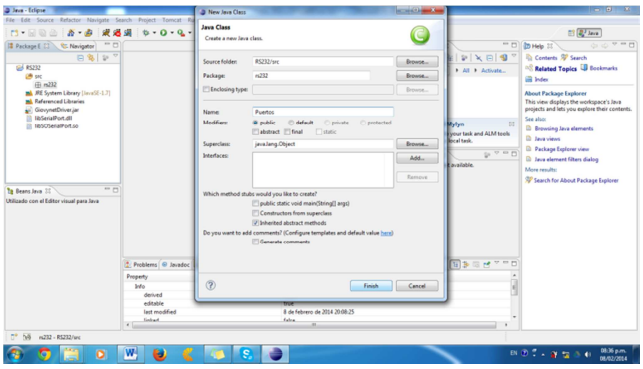
3.1-Crear una  Class dentro de nuestro proyecto “RS32” damos clic en el Package, clic en New  y clic en Class

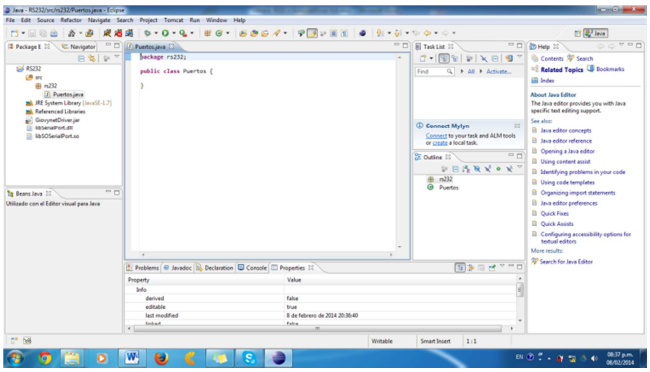





3.2-Le damos un nombre a esta class en  Name

Nonbre:Puertos





### 3.3-Codigo.

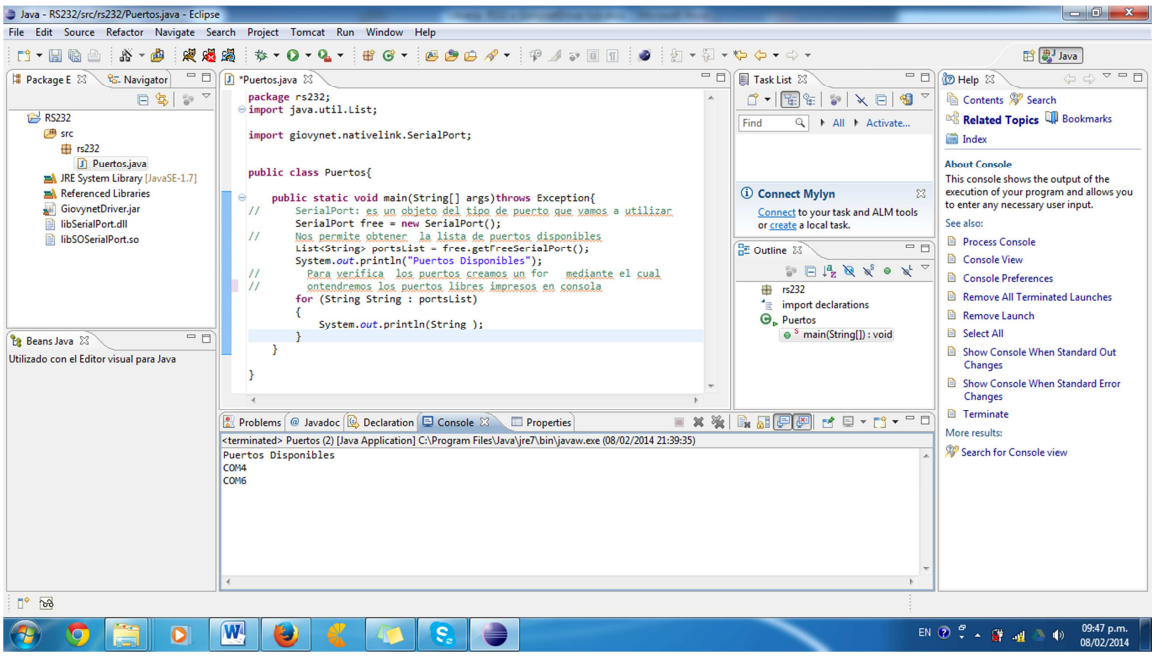
```
package rs232;
import java.util.List;

import giovynet.nativelink.SerialPort;


public class Puertos{

   public static void main(String[] args)throws Exception{
//          SerialPort: es un objeto del tipo de puerto que vamos a utilizar
     SerialPort free = new SerialPort();
//    Nos permite obtener  la lista de puertos disponibles
     List<String> portsList = free.getFreeSerialPort();
     System.out.println("Puertos Disponibles");
//      Para verifica  los puertos creamos un for   mediante el cual
//      ontendremos los puertos libres impresos en consola
     for (String String : portsList)
     {
        System.out.println(String );
     }
   }

}
```




## 4- Ejemplo de lectura y escritura por medio de un puerto serial

4.1- Crear una Class dentro de nuestro proyecto “Ejemplo” damos clic en el Package, clic en New y clic en Class

### 4.2-Codigo.

```
package rs232;
import giovynet.serial.Baud;
import giovynet.serial.Com;
import giovynet.serial.Parameters;

import java.awt.Frame;

import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JTextField;

public class Ejemplo extends Frame
{
//              Variables y objetos visuales
                JLabel  Etiqueta=new JLabel();
                JTextField Texto=new JTextField();
                JButton Boton=new JButton();

                JLabel  Etiqueta1=new JLabel();
                JLabel  Etiqueta2=new JLabel();
                JButton Boton1=new JButton();
                String caracter = "";

                public Ejemplo()
                {
//                              Frame
                                setVisible(true);
                                setLayout(null);
                                setTitle("Lectura y escritura de datos");
                                setSize(325,125);
//                               Etiqueta
                                Etiqueta.setBounds(20,50, 100, 20);
                                Etiqueta.setText("Enviar un digito");
                                add(Etiqueta);
//                              Caja de texto
                                Texto.setBounds(120,50,15, 20);
                                add(Texto);
//                              Boton
                                Boton.setBounds(150,50,100, 20);
                                Boton.setText("Enviar");
                                add(Boton);
//              Etiqueta 1
                                Etiqueta1.setBounds(20,80, 100, 20);
                                Etiqueta1.setText("Leer un digito");
                                add(Etiqueta1);
//                              Etiqueta2
                                Etiqueta2.setBounds(120,80,15, 20);
                                add(Etiqueta2);
//                              Boto 1
                                Boton1.setBounds(190,80,100, 20);
                                Boton1.setText("Leer");
```

```
                add(Boton1);
//          Evento boton1
                Boton.addMouseListener(new java.awt.event.MouseAdapter() {
                        public void mouseClicked(java.awt.event.MouseEvent e) {

                                                        Parameters configuracion = null;
                                                        try {
//                                                              //Definición de parametros
                                                                configuracion = new Parameters();
//                                                              //definición del puerto que se va a utilizar
                                                                configuracion.setPort("COM1");
//                                                              //definición de la velocidad de impresión, se debe tener en
//                                                              cuenta dicho argumento en las especificacion de velocidad del dispositivo
                                                                configuracion.setBaudRate(Baud._9600);
//                                                              asignamos los parametros al objeto com1
                                                                Com com1 = null;
                                                                com1 = new Com(configuracion);
//                                                              envio de un  de caracter
                                                                com1.sendSingleData(Texto.getText());
//                                                               fin de envio de secuencias de escape ESC/POS
                                                                com1.close();
                                                        } catch (Exception e1) {
                                                                // TODO Auto-generated catch block
                                                                e1.printStackTrace();
                                                        }
                                                }
                        }
                );
                        //Evento boton 1
                        Boton1.addMouseListener(new java.awt.event.MouseAdapter() {
                                public void mouseClicked(java.awt.event.MouseEvent e) {
                                        System.out.println("mouseClicked()"); // TODO Auto-generated Event stub mouseClicked()
                                        Parameters configuracion = null;
                                        try {
                                        configuracion = new Parameters();
                                        configuracion.setPort("COM1");
                                        configuracion.setBaudRate(Baud._19200);
                                        Com com1 = null;
                                        com1 = new Com(configuracion);
                                         while(caracter.equals("")){

                                                                caracter =com1.receiveSingleString();
                                                                System.out.println(caracter);
                                                                Etiqueta2.setText(caracter);

                                           }
                                                                com1.close();
                                                                 caracter ="";

                                                        } catch (Exception e1) {
                                                                // TODO Auto-generated catch block
                                                                e1.printStackTrace();
                                                        }
                                                }
                        });
                }
}
```

## 4.3-Definicion, instalacion ,manejo del hyperterminal y  SetupVSPE

### *4.3.1-Definiciones:*

Hyperterminal:

HyperTerminal es un potente programa de comunicación de windows. Con HyperTerminal y un modem es posible colegar la propia computadora a un servicio de correo electrónico y a mucho bancos datos, o BBS. Si, por ejemplo, te colegas a internet con un fornidor de acceso, sin el colegamento PPP o SLIP, probablemente usarás el HyperTerminal. Una vez que te colegues a la otra computedora, el tuyo será un terminal mudo, en otras palabras, estarás usando tu teclado y la pantalla de tu monitor pero que en realidad estarás usando la otra computadora y su software. Cuando te colegas a un BBS probablemente encontrarás los menúes de comandos que podrás utilizar para hacer cosas diferentes.

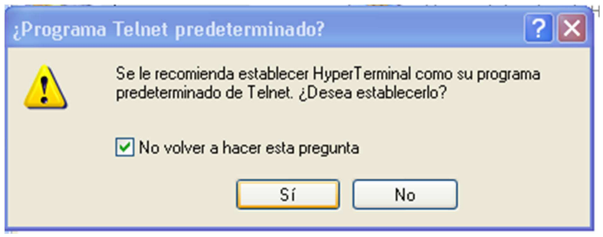
### *4.3.2-Instalacion.*

Hyperterminal.

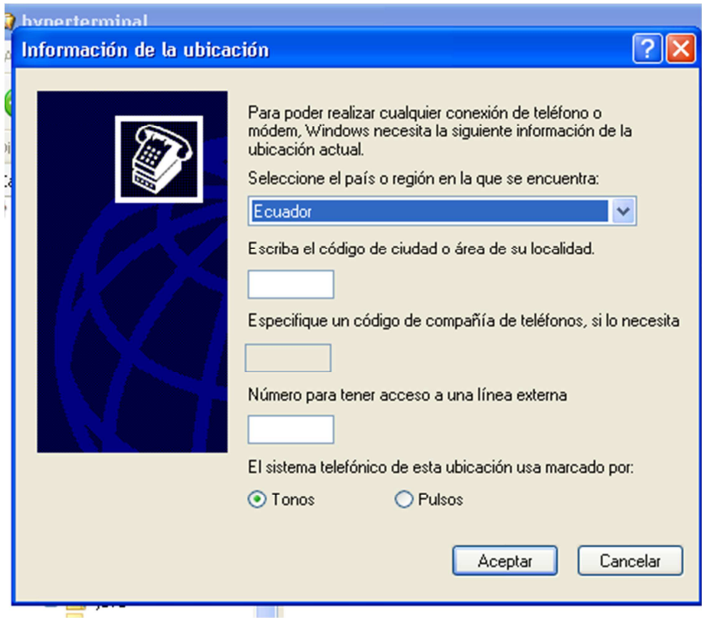
Descargar de:

http://www.softonic.com/s/hyperterminal-windows-7-gratis

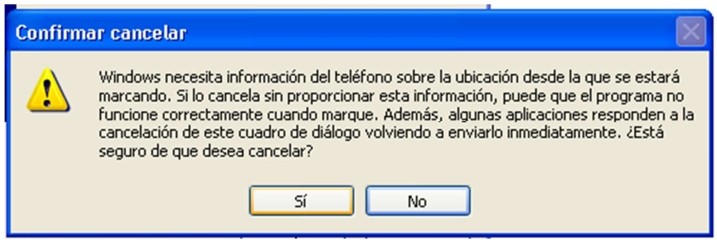
Ejecutar hypertrm.exe



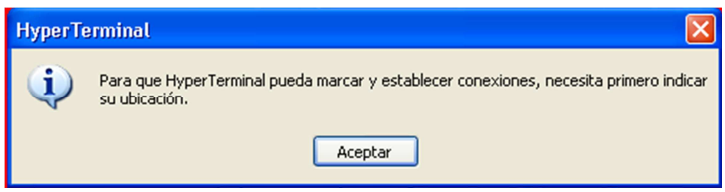

**Poner SI**




**Cancelar**



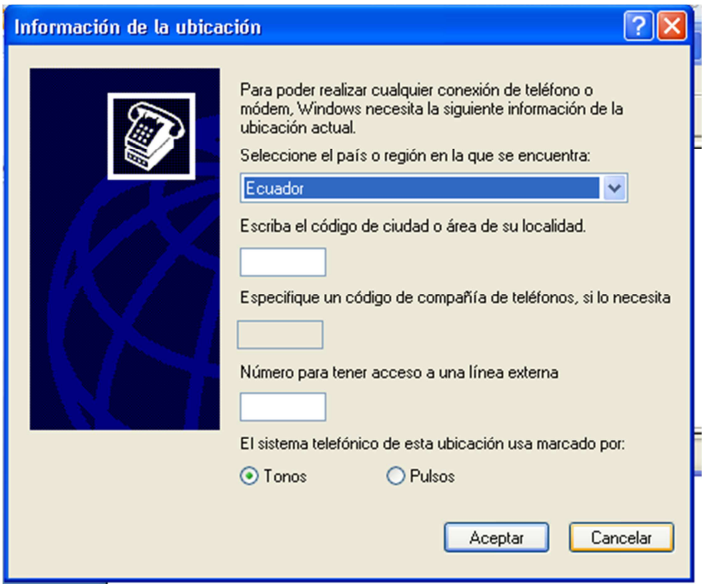

**Si**



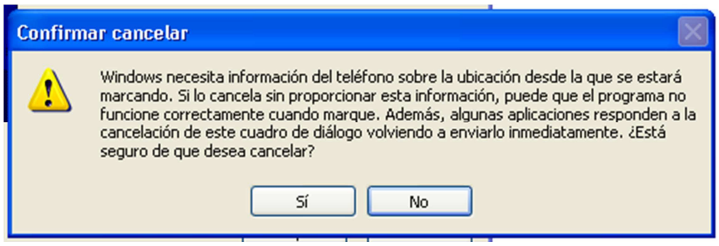

**Aceptar**


**Nombre: Comunicación**

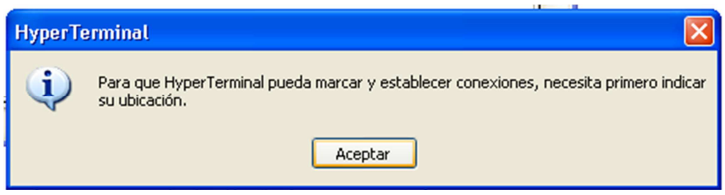
**Aceptar**




**Cancelar**

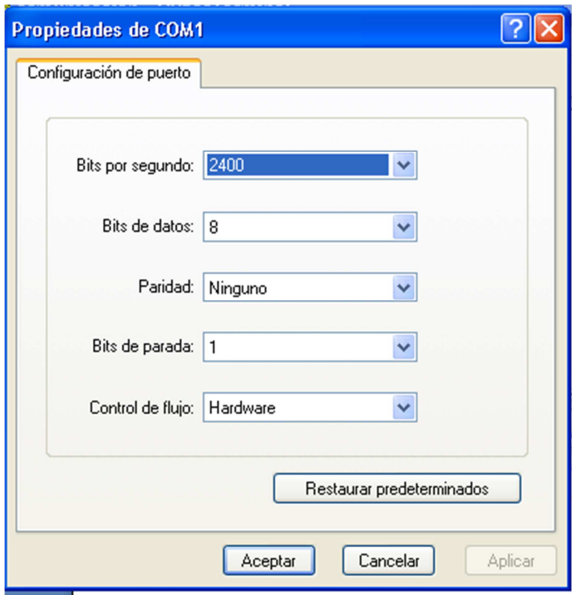



**Si**

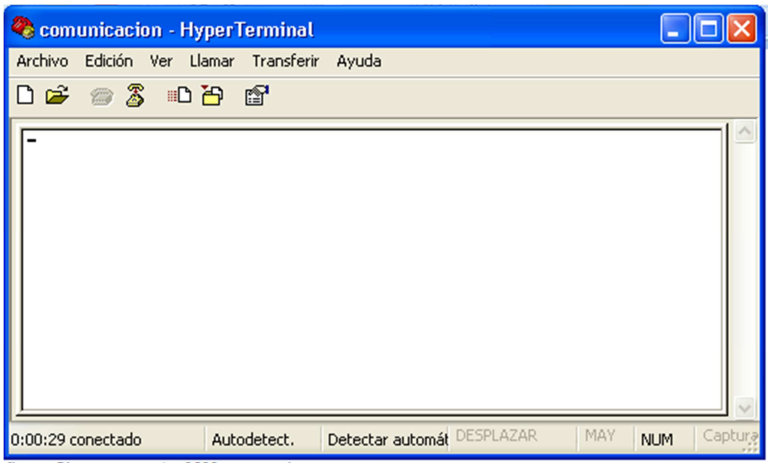



Aceptar


Escoger el puerto a utilizar en la comunicación con java y Poner aceptar




Configurar Bits por segundos 9600 y poner Aceptar




Con los pasos anteriores ya estara listo para la comunicación

### *4.3.3-Instalacion de VSPE*

Nota:solo seguiremos este paso si no tenemos puiertos libres con este programa crearemos un puerto en la memoria de la computadora para poder simular como funciona el puerto

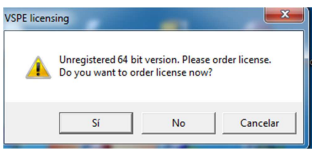
Descargar de :
http://www.lawyerment.com/downloads/Programming/Debugging_and_Tracing/Review_17462_index.htm

Ejecutar  SetupVSPE.msi y seguir los pasos que nos indica en consola .

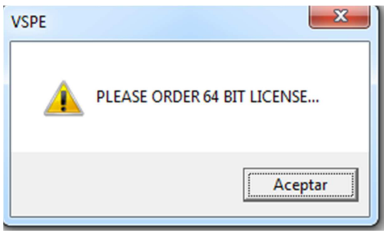
Con esto pasamos a configurarlos

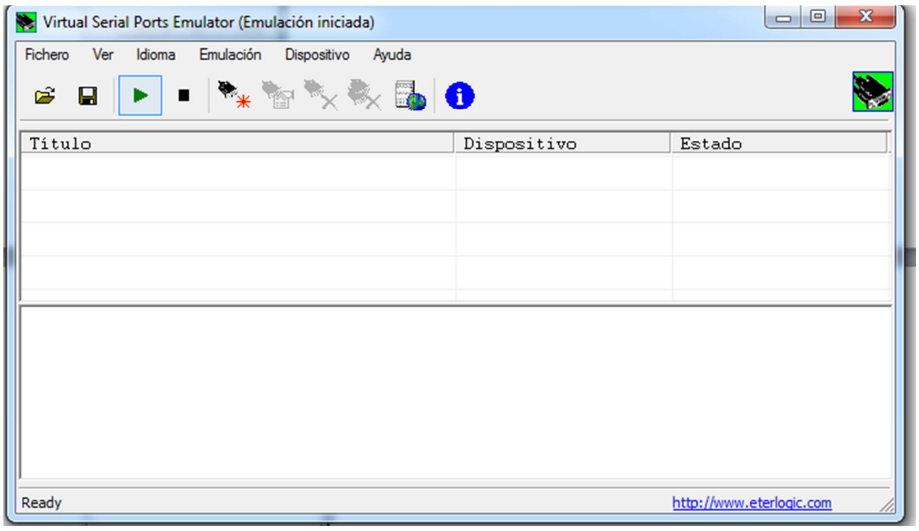
Abri.


Cancelar



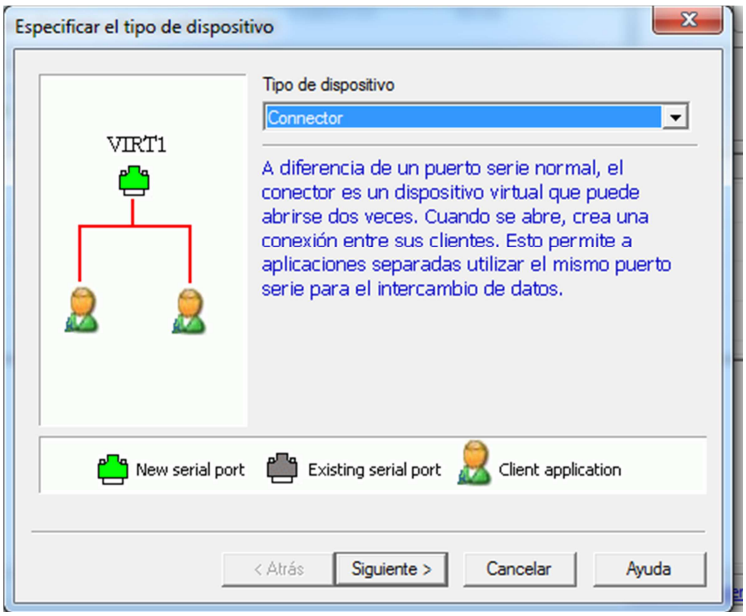

Nos indica para que tipo de omputadora es en este caso 64 bist poner aceptar



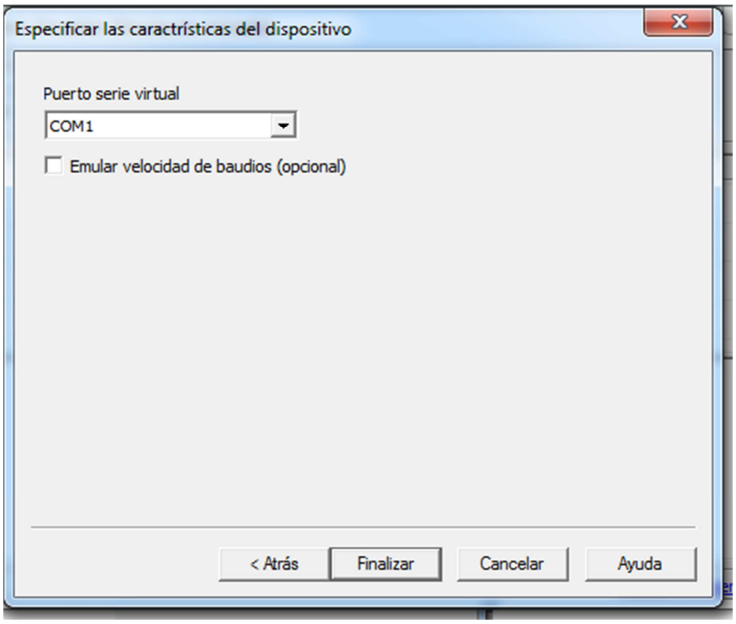

Clic en

Siguiente : estamo configurando para trabajar en la misma computadora

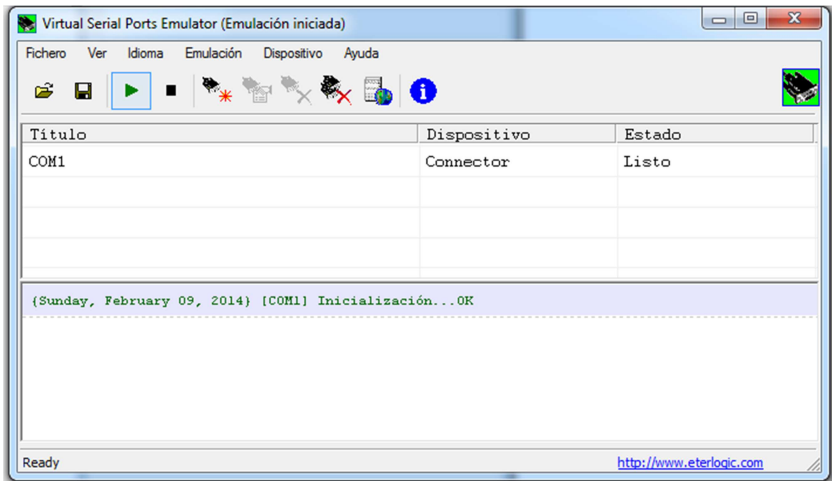



Escoger el puerto como se llamara  pero que no contenga ningun nombre de los puertos libres

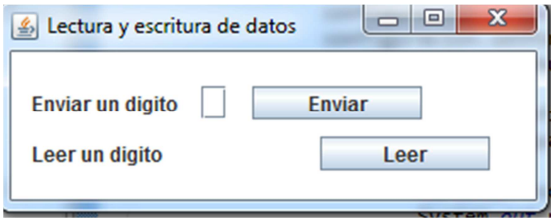
Una ves listo poner Finalizar




Con esto ya esta creado el puerto virtual

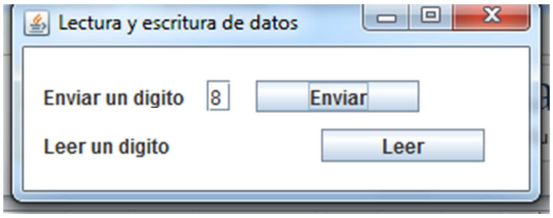


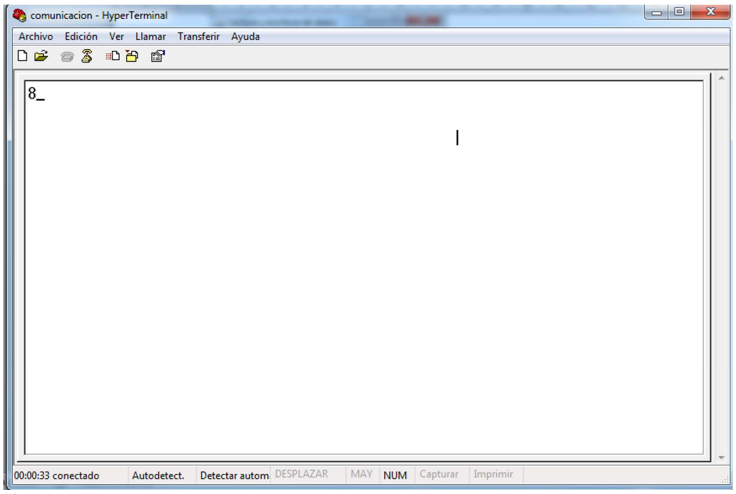

Ejecucion del ejemplo



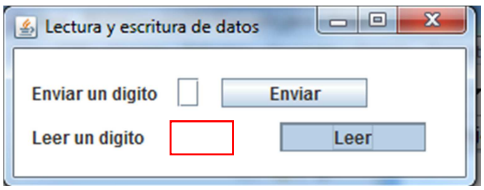

Ecribir un digito en el cuadro de texto

Clic en Enviar y el hyperterminal resivira este ejemplo

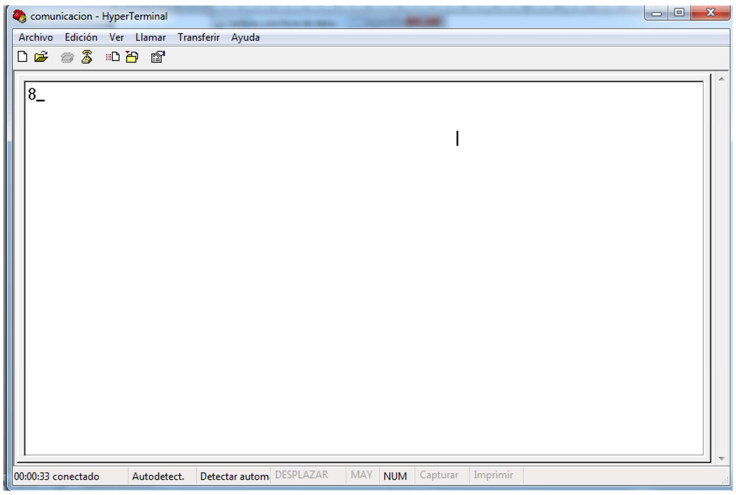


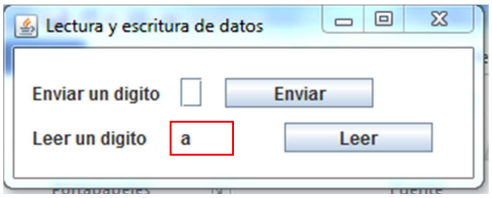

Clic en leer.




Ir al hyperterminaly escribir




Nota no se vera en el hyperterminal lo que se escribe pero el puerto si resivira la informacion es este caso escribi a y salio a lado de Leer un digito



## Pasos para instalar la Librería RXTX

### 5.FUENTES WEB PARA LA LIBRERÍA RXTX

Primero hay algunos archivos que necesitaremos para realizar la comunicación con java correctamente así que hay que descargar las librerías ***rxtx*** dependiendo de la arquitectura de tu computador (32 bits ó 64 bits) en el siguiente blog podemos descargar las librerías y los programas necesarios para simular el puerto serial en tu computador estos son el ***virtual serial port emulator*** (VSPE que como su nombre lo indica nos emula un puerto serial) y el ***hiperterminal***:

- Librerías y Programas necesarios:

  http://wp.me/p4kegf-22

- También necesitaremos el ***software Arduino*** que podemos descargarlo de su página oficial:

  *Para x86: http://arduino.googlecode.com/files/arduino-1.0.5-r2-windows.exe

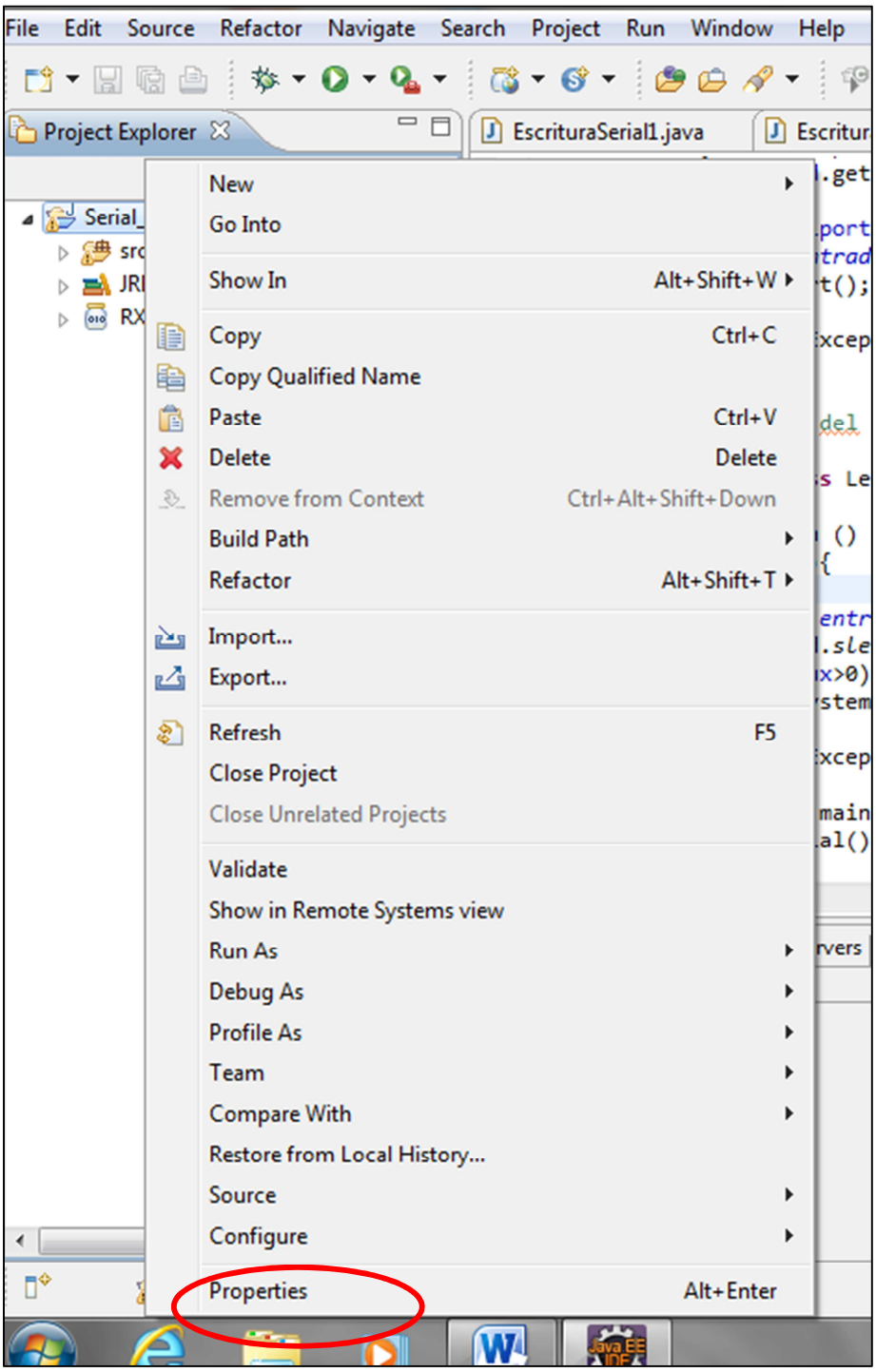
  *Para x64: http://downloads.arduino.cc/arduino-1.5.5-r2-windows.exe

### 6. CONFIGURACIÓN DE LAS LIBRERÍAS EN JAVA

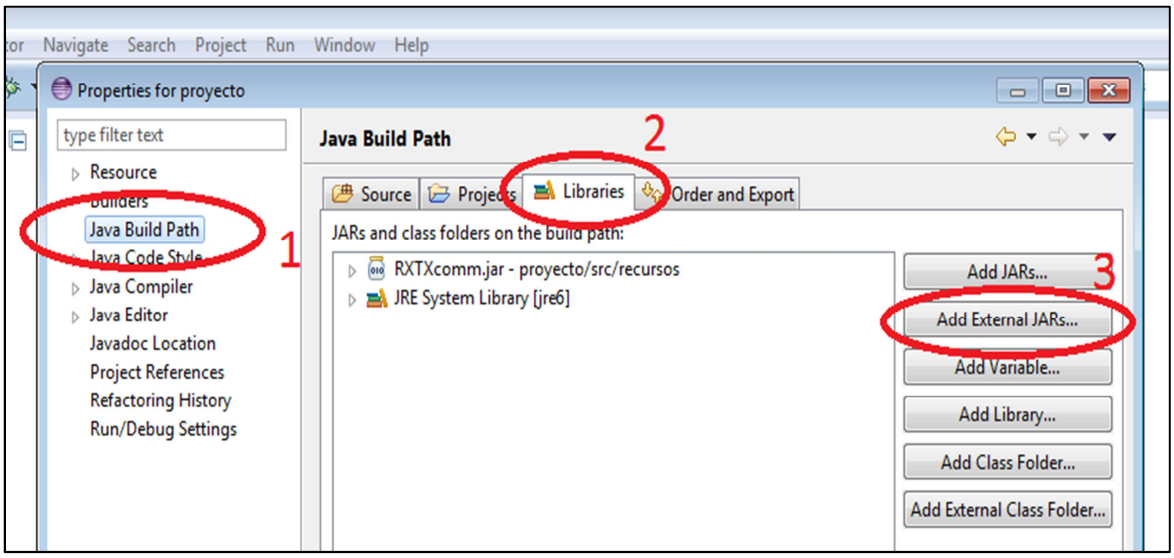
Una vez descargadas las librerías y los programas empezamos descomprimiendo el archivo *"rxtx librerias para XX bits"* y el archivo **rxtxSerial.dll** lo copiamos en la siguiente dirección **C:\Program Files\Java\jreX\bin** donde la **X** varía según la versión de java que tengas instalado en el computador eso para arquitectura de 32 bits.

Si es el caso de que usas arquitectura de 64 bits tienes dos opciones **C:\Program Files\Java\jreX\bin** ó **C:\Program Files(x86)\Java\jreX\bin**.

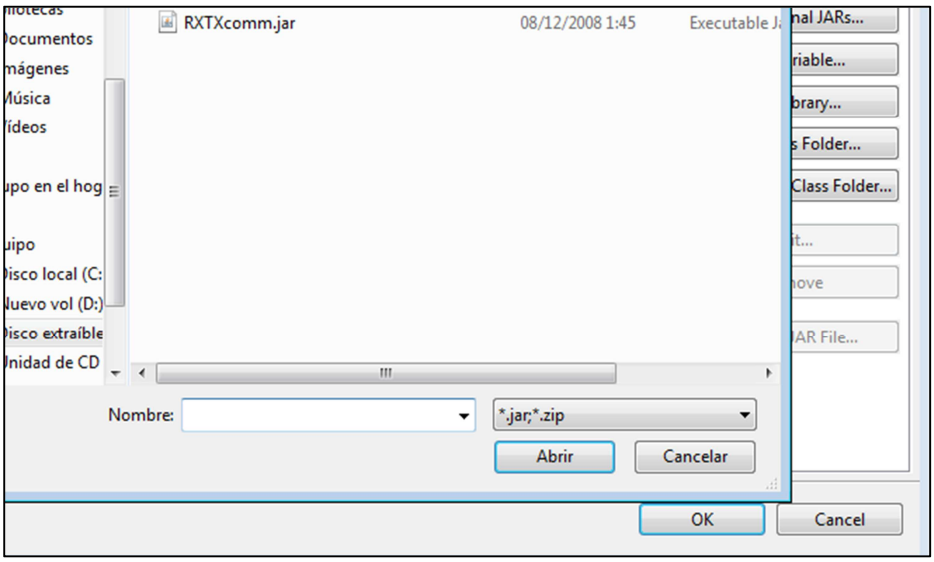
Ahora para el archivo ***RXTXcomm.jar*** (nosotros estamos trabajando con eclipse) entonces abrimos eclipse y damos click derecho en el proyecto en el que estamos trabajando




Y nos dirigimos a PROPIEDADES y nos aparece una venta como esta:




Entonces nos dirigimos a *Java build path* luego a *Libraries* y con el boton *Add External JARs* buscamos en nuestro disco el directorio donde descomprimimos el archivo ***RXTXcomm.jar*** y damos *Abrir* y luego *Ok.*




El ***Hiperterminal*** no hay necesidad de instalarlo, ahora solo instalamos el ***VSPE*** y el ***software Arduino*** que cuando lo instalamos instala los drivers de tu placa Arduino.

**NOTA: LAS EXPLICACIONES DETALLADAS SE ENCUENTRAN COMENTADAS DENTRO DE LOS CÓDIGOS QUE SE PRESENTARAN EN ESTE MANUAL.**

## 7.CÓDIGO PARA EL MANEJO DEL PUERTO SERIAL EN JAVA

```
//librerías necesarias
import gnu.io.CommPortIdentifier;
import gnu.io.PortInUseException;
import gnu.io.SerialPort;
import java.io.IOException;
import java.io.OutputStream;
import java.util.Enumeration;

 //descargar software VSP para virtualizar puertos COM
public class EscrituraSerial1 {

      public static void main(String[] args) {
       Enumeration puertos; //busca todos los puertos y los guarda en el objeto puertos
      puertos=CommPortIdentifier.getPortIdentifiers(); //ojo tiene que tener la -s- al
ultimo porque hay otro metodo sin -s-
      CommPortIdentifier portId; // identifica los puertos com
      SerialPort serialport; // esta clase abre puertos
            while (puertos.hasMoreElements()) { //para recorrer el numero de los puertos, y
especificar con cual quiero trabajar
            //hasmorelements mientras tenga mas elementos:
            portId = (CommPortIdentifier) puertos.nextElement(); //next elemento recorre
uno por uno
            System.out.println(portId.getName()); //puertos disponbibles
                  if (portId.getName().equalsIgnoreCase("COM7")) {
                        try {
      serialport= (SerialPort)portId.open("EscrituraSerial1", 500);//tiempo en ms
                  /**
                  Aquí se reliza la escritura  la lectura del puerto serial
                  */

                        serialport.close();
                        } catch (Exception e) {

                  }
            }
      }
      }

}
```

## 8. Ejemplo de lectura y escritura para un puerto serial

### 8.1Escritura serial

#### *8.1.1 Escritura serial mediante el método write*

```
public class EscrituraSerial1 {

	public static void main(String[] args) {
	 Enumeration puertos; //busca todos los puertos y los guarda en el objeto puertos
	OutputStream ops;//Declaramos la variable del tipo OutputStream para escribir los datos
	puertos=CommPortIdentifier.getPortIdentifiers(); //ojo tiene que tener la -s- al ultimo
porque hay otro metodo sin -s-
	CommPortIdentifier portId; // identifica los puertos com
	SerialPort serialport; // esta clase abre puertos
		while (puertos.hasMoreElements()) { //para recorrer el numero de los puertos, y
especificar con cual quiero trabajar
		//hasmorelements mientras tenga mas eleementos
	portId = (CommPortIdentifier) puertos.nextElement(); //next elemento recorre uno por
uno
		System.out.println(portId.getName()); //puertos disponbibles
			if (portId.getName().equalsIgnoreCase("COM7")) {
				try {
		serialport= (SerialPort)portId.open("EscrituraSerial1", 500);//tiempo en ms
		ops=serialport.getOutputStream(); // el puerto obtendra los datos que enviemos
		ops.write (" UPS".getBytes()); //get bytes transforma el string a bytes
				ops.close();// cerramos la salida
				serialport.close(); //cerramos el puerto
				} catch (Exception e) {
			}
}
}
}
}
```

### *8.1.2 Escritura serial mediante el método print (Ejemplo donde se extrae un vector String desde un txt)*

```
import gnu.io.CommPortIdentifier;
import gnu.io.SerialPort;
import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import java.io.PrintStream;
import java.util.Enumeration;
  public class EscrituraSerial2 {
      static BufferedReader extraer;

      public static void main(String[] args)  {
            //leemos y guardamos los datos desde el txt
             try {
                   FileInputStream F = new FileInputStream("D://q//Diez.txt");
            extraer= new BufferedReader (new InputStreamReader (F));
} catch (Exception e) {
}
//declaramos e inicializamos las variables
      Enumeration puertos;
       puertos=CommPortIdentifier.getPortIdentifiers();
       CommPortIdentifier portId;
       SerialPort serialport;
       PrintStream salida;
       String vec[]=new String[2];
       //la misma configuracion para conectarse con el puerto serial de la pagina 9
       while (puertos.hasMoreElements()) {
            portId = (CommPortIdentifier) puertos.nextElement();
              System.out.println(portId.getName());
            if (portId.getName().equalsIgnoreCase("COM7")) {
                        try {
                              serialport= (SerialPort)portId.open("EscrituraSerial2",
500);
                              // declaramos el objeto salida del tipo PrintStream para
relaizar la escritura
                              salida = new PrintStream(serialport.getOutputStream());

                              for (int i = 0; i < vec.length; i++) { //en este for
recorremos los datos del txt
                                    String c=extraer.readLine();
                                    vec[i]=c;
                                    salida.print(vec[i]+"\n"); //enviamos al puerto serial
                                    }
                                          salida.close(); //cerramos la salida
                                          serialport.close();//cerramos el puerto
                              } catch (Exception e) {
                              }}}}}
```

### 8.2 Lectura serial

```
import gnu.io.CommPort;
import gnu.io.CommPortIdentifier;
import gnu.io.SerialPort;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.Enumeration;
  public class LecturaSerial  {
        //inicializamos y decalramos variables
      CommPortIdentifier portId;
      Enumeration puertos;
      SerialPort serialport;
      static InputStream entrada = null;
      Thread t;
      //creamos un constructor para realizar la conexion del puerto
      public LecturaSerial() {
        super();
        puertos=CommPortIdentifier.getPortIdentifiers();
        t = new Thread(new LeerSerial());
        while (puertos.hasMoreElements()) { //para recorrer el numero de los puertos, y
especificar con cual quiero trabajar
            //hasmorelements mientras tenga mas eleementos
            portId = (CommPortIdentifier) puertos.nextElement(); //next elemento recorre
uno por uno
            System.out.println(portId.getName()); //puertos disponbibles
            if (portId.getName().equalsIgnoreCase("COM7")) {
                  try {
                  serialport= (SerialPort)portId.open("LecturaSerial", 500);//tiempo en ms
                        entrada = serialport.getInputStream();//esta variable del tipo
InputStream obtiene el dato serial
                        t.start(); // inciamos el hilo para realizar nuestra accion de
imprimir el dato serial

                  } catch (Exception e) {
                  } } }
  }
    //con este metodo del tipo thread relaizamos

      public static class LeerSerial implements Runnable {
         int aux;
       public void run () {
         while(true){
              try {
                  aux = entrada.read(); // aqui estamos obteniendo nuestro dato serial
                  Thread.sleep(100);
                  if (aux>0) {
                        System.out.println((char)aux);//imprimimos el dato serial
                        }
                  } catch (Exception e) {
                  } } }
  }public static void main(String[] args) {
          new LecturaSerial();
              }}
```

## 9. COMUNICACIÓN JAVA-ARDUINO MEDIANTE UN PUERTO SERIAL

### 9.1 PROGRAMACIÓN EN ARDUINO

```
byte PIN_SENSOR = A0;
int dato_serial =0; // para los datos de entrada serie
float C;
int temp;

void setup() {
Serial.begin(9600); // abre el puerto serie, establece la velocidad a 9600 bps
}

void loop() {
C = (5.0 * analogRead(PIN_SENSOR) * 100.0) / 1024;
temp=C;
// envía datos solo cuando recibe datos:
if (Serial.available() > 0) {  // si ha llegado un dato serial
lectura_dato(); // realiza la operación lectura_dato
comparacion_dato();          // llama a la funcion de comparar el dato

} }

void lectura_dato (void){
// lee el byte de entrada:
dato_serial = Serial.read();
}

void comparacion_dato (void){
if (dato_serial=='T') { // si el dato serial es la T envía la temperatura a java
Serial.write(temp);
}}
```

### 9.2 PROGRAMACIÓN EN JAVA

En los ejemplos anteriores hemos trabajado la lectura del puerto serial solo en consola en este ejemplo para una mejor visualización de la temperatura se ha implementado algunos componentes de interfaz gráfica que son:

- Jtextfield para asignar el puerto que tenga nuestro Arduino
- Jlabel para indicar la temperatura dentro del Jframe
- Dos Jbutton para conectar y desconectar el puerto.

**SE EXPLICARÁ SOLAMENTE LA PARTE RELACIONADA CON EL PUERTO SERIAL EL CÓDIGO COMPLETO SE ENCUENTRA EN EL SIGUEINTE ENLACE:**

http://wp.me/p4kegf-29

Declaramos e inicializamos las variables para crear la conexión con el puerto serial de Arduino:

```
  Enumeration puertos_libres =null;
  CommPortIdentifier port=null;
SerialPort puerto_ser = null;
OutputStream out = null;
InputStream in = null;
int temperatura=10;
Thread timer;
```

Esta es la configuración de Arduino para comunicarse con java, estas líneas van dentro de la configuración que se explica en la página 9, pero como estamos trabajando en un Jframe toda la configuración para el puerto serial en vez de ir en el *main* se lo escribe dentro del constructor de la clase:

```
        try {
              puerto_ser = (SerialPort) port.open("puerto serial", 2000);
                          int baudRate = 9600; // 9600bps

                          puerto_ser.setSerialPortParams(
                          baudRate,
                          SerialPort.DATABITS_8,
                          SerialPort.STOPBITS_1,
                          SerialPort.PARITY_NONE);
                          puerto_ser.setDTR(true);
                          out = puerto_ser.getOutputStream();
                          in = puerto_ser.getInputStream();

                          timer.resume();
              }
```

Creamos el metodo del tipo thread done relizaremos los procesos que necesitemos ya sea lectura o escritura, como es un ejemplo de un sensor de temperatura relizamos las dos opciones:

```
private class ImplementoRunnable implements Runnable{
            int aux;
            public void run() {
                  while(true){
                        try {
                        out.write('T'); // enviamos la T a Arduino
                        Thread.sleep(100);
                        aux = in.read();// resivimos los datos de arduino

                        if (aux!=2){
                        temperatura = aux;
                        lblNewLabel.setText(""+temperatura+" ºC");
                  System.out.println(aux);
                  }repaint();

                  } catch (Exception e1) {
                  }}}}
```

Tener en cuenta que java solo tiene una entrada así que cualquier valor que envié Arduino hay que separarlo en java para usar solo los que necesitamos, en nuestro ejemplo como solo hemos configurado un sensor de temperatura solo recibiremos ese valor.

Aspecto final de nuestro programa de ejemplo:

## Recomendaciones:

- Se recomienda siempre en la comunicación serial que una vez realizada la lectura o escritura se debe interrumpir el o los hilos y no olvidar cerrar el puerto y las entradas y salidas de java.

- Si desean visualizar archivos ***.java*** de forma rápida sin perder el realzado de sintaxis (los colores) y la tabulación (dependiendo del IDE) entonces instalen el ***notepad++*** (Es un editor de textos Open Source).

- Si deseas aprender más acerca de la programación en java visita nuestro sitio: http://upscodigojava.wordpress.com/