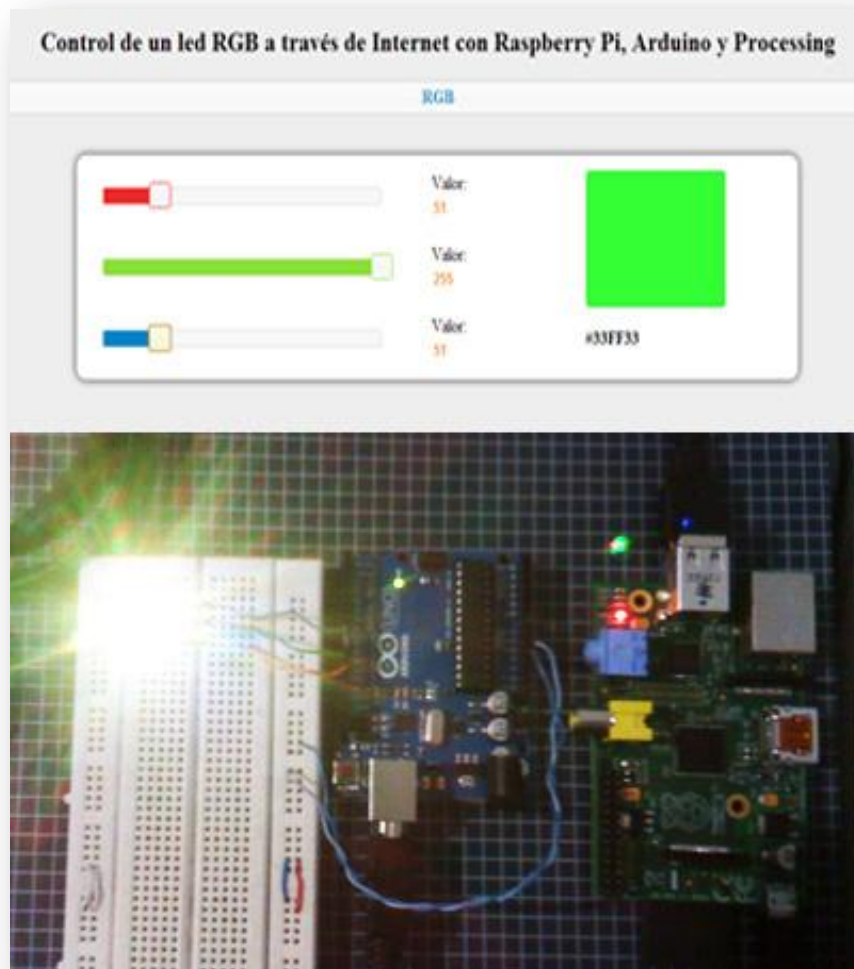


CONTROL DE UN LED RGB A TRAVÉS DE INTERNET USANDO RASPBERRY PI, ARDUINO Y PROCESSING



Desarrollado por:
Jefferson Rivera Patiño

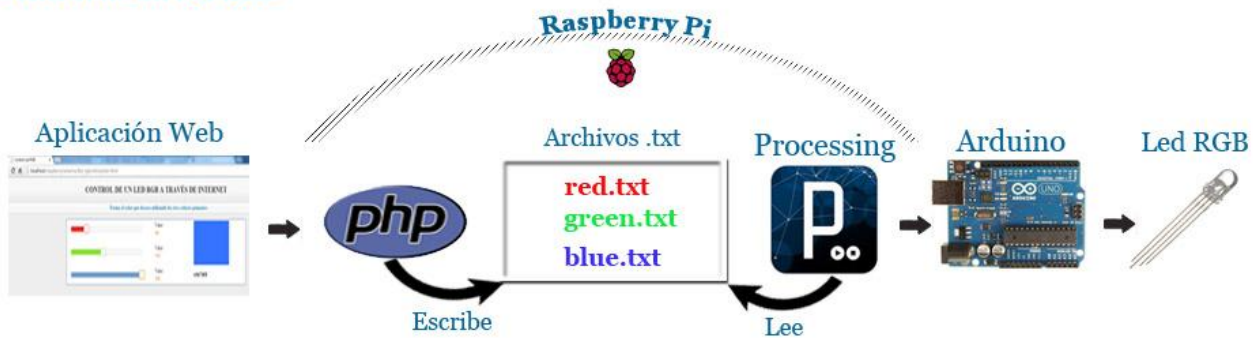
[@riverajefer](#)
riverajefer.blogspot.com
jeffersonrivera.com

Contenido

1. DESCRIPCIÓN DEL PROYECTO.....	3
2. EXPLICACIÓN DEL SOFTWARE	3
2.1. APLICACIÓN WEB	3
2.2 APLICACIÓN ARDUINO.....	4
2.3 INSTALACIÓN Y CONFIGURACIÓN DE PROCESSING 1.5 SOBRE RASPBERRY PI.....	5
2.3.1 APLICACIÓN PROCESSING-ARDUINO.....	8
3. EXPLICACIÓN DEL HARDWARE.....	10
4. PRUEBA DE FUNCIONAMIENTO	11
5. BIBLIOGRAFÍA.....	13
ANEXOS.....	13

1. DESCRIPCIÓN DEL PROYECTO

Control de un led RGB

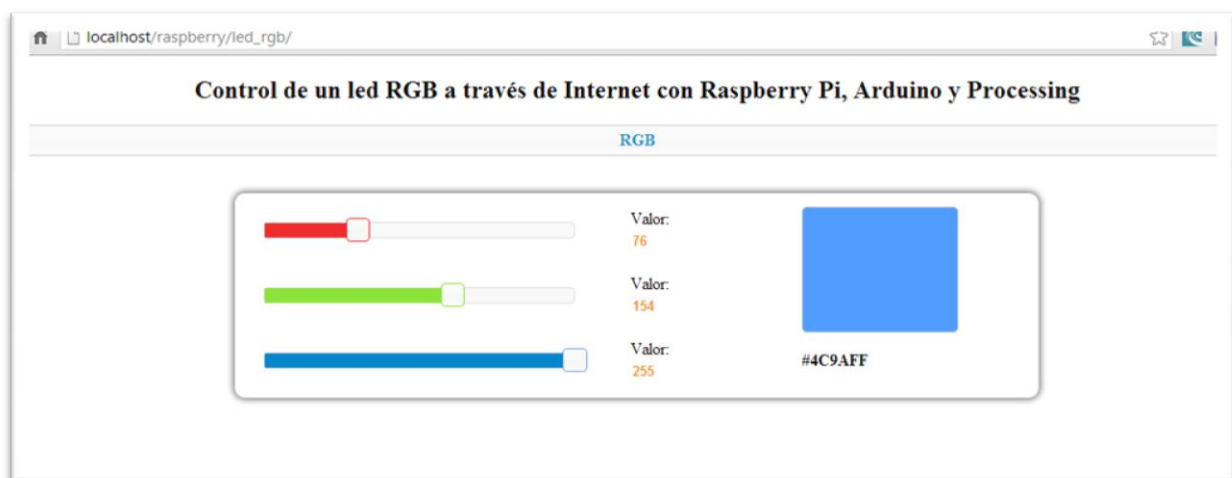


Como se visualiza en la imagen anterior, el flujo del proceso, va del cliente-navegador al hardware- led RGB

Entonces en la aplicación web tenemos tres slides, uno para cada color: red, green, blue, con rangos de 0 a 255. Cuando movemos uno de ellos, por ejemplo el rojo, se envía su valor por medio de jQuery a PHP, donde se procesa y se guarda dicho valor en un archivo plano llamado red.txt. Luego con processing leemos el contenido de este archivo y lo enviamos a la placa Arduino a través de un pin de salida análoga, para posteriormente reflejarla en el led RGB. Este mismo proceso se aplica para los demás colores.

2. EXPLICACIÓN DEL SOFTWARE

2.1 APLICACIÓN WEB



La aplicación base fue tomada de jQuery UI¹

La aplicación cuenta con los siguientes directorios y archivos:



- **css/** Contiene las hojas de estilo, que nos da jQuery Ui
- **js/** Contiene los archivos javascript, entre ellos jQuery.
- **blue.txt, greend.txt, red.txt** ->. archivos de texto plano que contienen los valores de cada color.
- **procesa.php** -> recibe las variables de los slide del index.html, las procesa y guarda su valor en el archivo plano, de acuerdo a su "color".
- **index.html** -> contiene los slides para cada color.

Descarga desde:

http://jeffersonrivera.com/pi/led_rgb.zip

2.2 APLICACIÓN ARDUINO



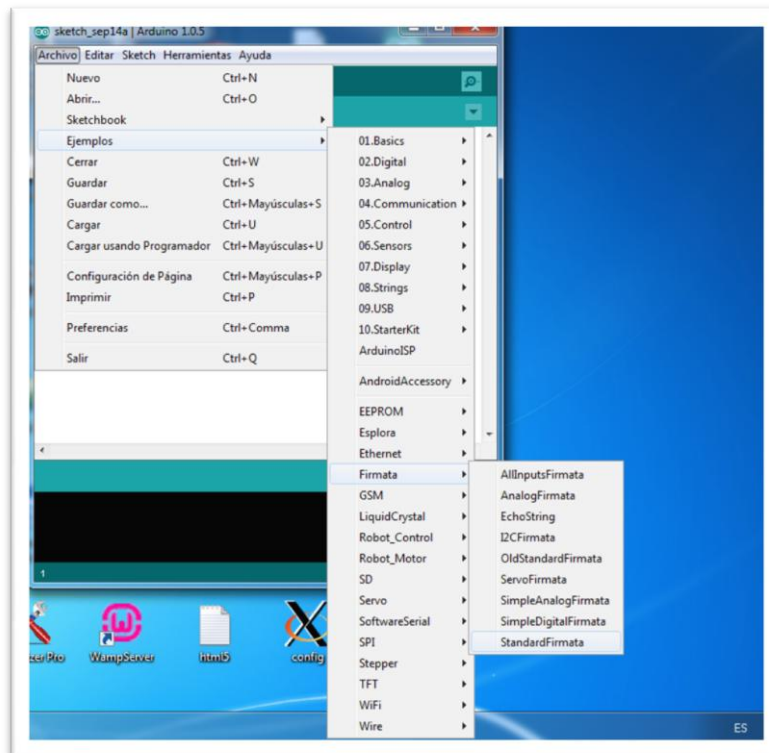
En la placa Arduino, se carga el Firmata que es una librería que contiene un protocolo, que nos permite comunicar Arduino con processing.

Tomado la descripción de la pagina de Arduino:

La librería **Firmata** implementa el protocolo Firmata que permite comunicarse con un software alojado en un ordenador servidor. Esto permite escribir un firmware personalizado sin tener que crear tu propio protocolo y objetos, para el entorno de programación que estás usando.²

¹ <http://jqueryui.com/slider/#colorpicker>

² <http://arduino.cc/es/Reference/Firmata>



2.3 INSTALACIÓN Y CONFIGURACIÓN DE PROCESSING 1.5 SOBRE RASPBERRY PI

A continuación se muestran los pasos para instalar y configurar processing, para que trabaje conde la mano con Arduino.

1. Instalar el JDK

```
pi@raspberrypi:~$ sudo apt-get install librtx-java openjdk-6-jdk
```

2. Descargamos Processing 1.5

```
pi@raspberrypi:~$ wget http://processing.googlecode.com/files/processing-1.5.1-linux.tgz
```

3. Descomprimos

```
pi@raspberrypi:~$ sudo tar -xvzf processing*.tgz
```

4. Ingresamos a la carpeta processing-1.5.1/

```
pi@raspberrypi:~$ cd processing-1.5.1
```

5. Eliminamos el directorio java

```
pi@raspberrypi:~$ /processing-1.5.1 $ rm -rf java
```

6. crear un enlace simbólico del sistema Java SDK

```
pi@raspberrypi:~$ ln -s /usr/lib/jvm/java-6-openjdk-armhf java
```

7. En processing eliminar o renombrar

```
pi@raspberrypi: ~$ sudo rm modes/java/libraries/serial/library/linux32/librtxSerial.so
```

8. En processing también eliminar

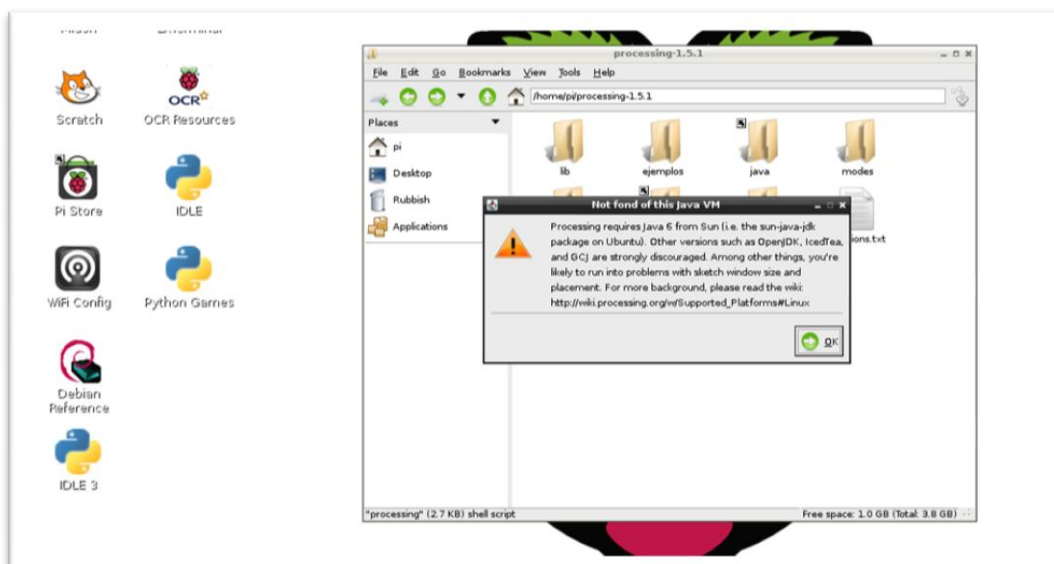
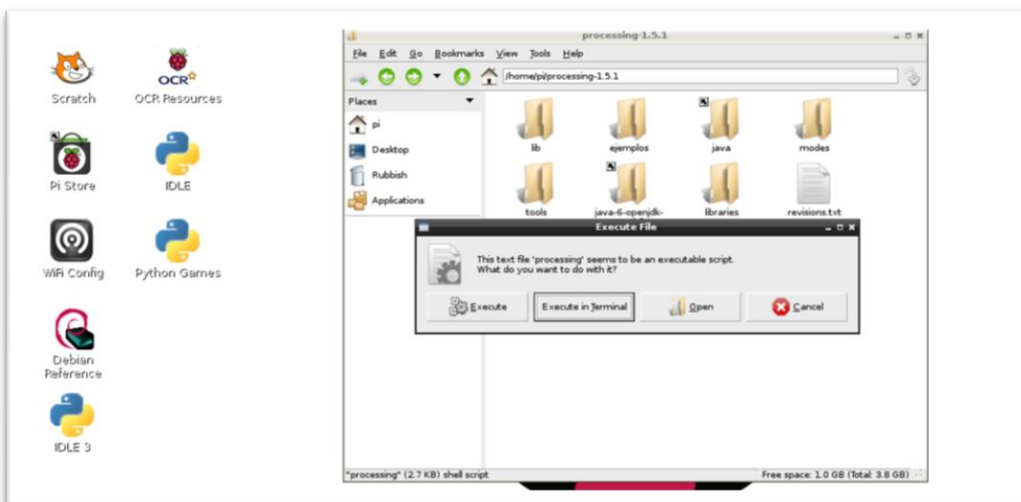
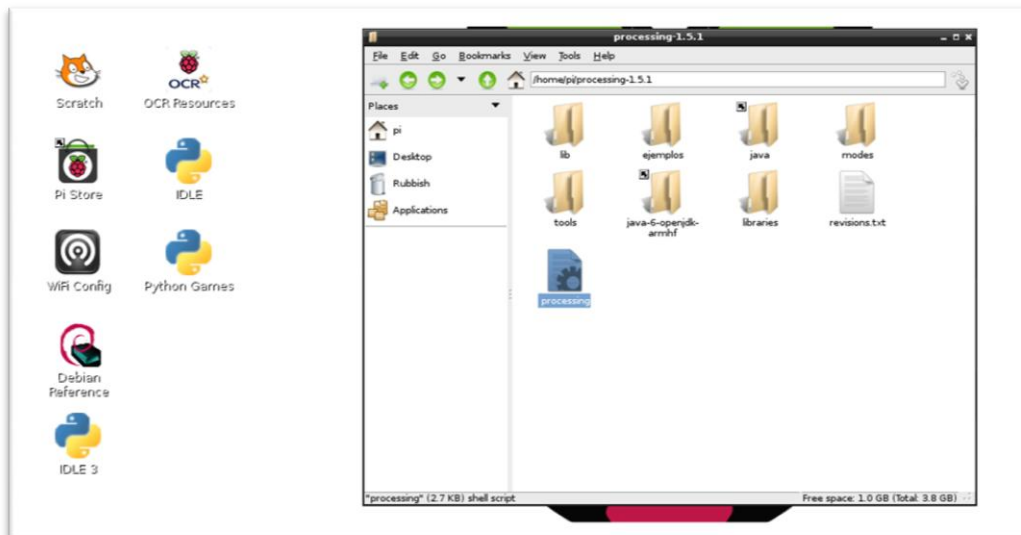
```
pi@raspberrypi: ~$ sudo rm modes/java/libraries/serial/library/RXTXcomm.jar
```

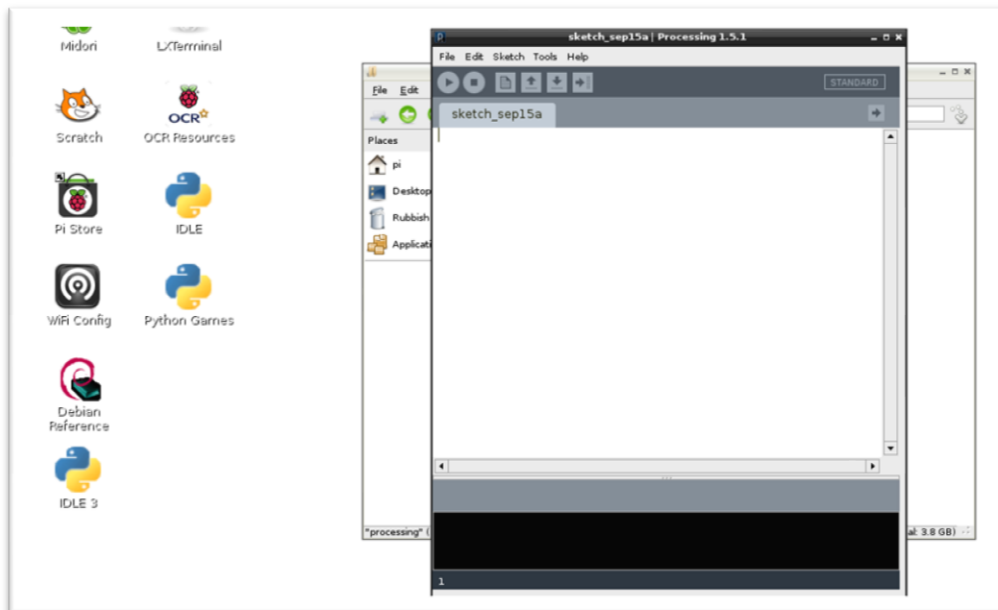
9. y reemplazarlo por

```
pi@raspberrypi: ~$ sudo cp /usr/share/java/RXTXcomm.jar  
modes/java/libraries/serial/library/
```

Tomado de:³

³ <http://scruss.com/blog/2012/08/12/controlling-an-arduino-from-raspberry-pi-using-processing/>





Ahora descargamos la librería arduino-processing⁴

En el siguiente vínculo se explica paso a paso como configurar la librería arduino en processing

<http://playground.arduino.cc/uploads/Nilseuropa/processing-arduinomega.zip>

En resumen. Descargar y descomprimir el paquete, ubicar la carpeta arduino en una subcarpeta donde se están guardando los script de processing llamada libraries, si no existe crear una.

Como estamos trabajando con Linux, renombrar el archivo Arduino.jar por arduino.jar

Cuando tengamos un ejemplo listo para comunicar processing con arduino, coger el archivo arduino.jar y soltarlo sobre la ventana de processing, para cargar la librería.

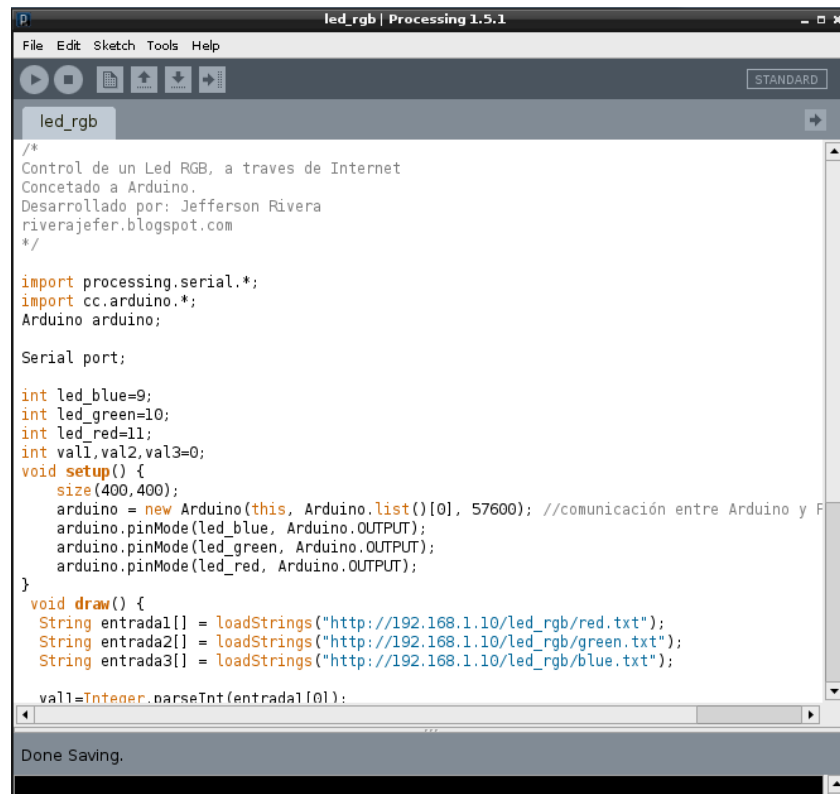
2.3.1 APLICACIÓN PROCESSING - ARDUINO



El código completo se puede descargar de:

http://jeffersonrivera.com/pi/led_rgb.zip

⁴ <http://playground.arduino.cc/interfacing/processing>



```
led_rgb | Processing 1.5.1
File Edit Sketch Tools Help

/*
Control de un Led RGB, a traves de Internet
Concetado a Arduino.
Desarrollado por: Jefferson Rivera
riverajefer.blogspot.com
*/

import processing.serial.*;
import cc.arduino.*;
Arduino arduino;

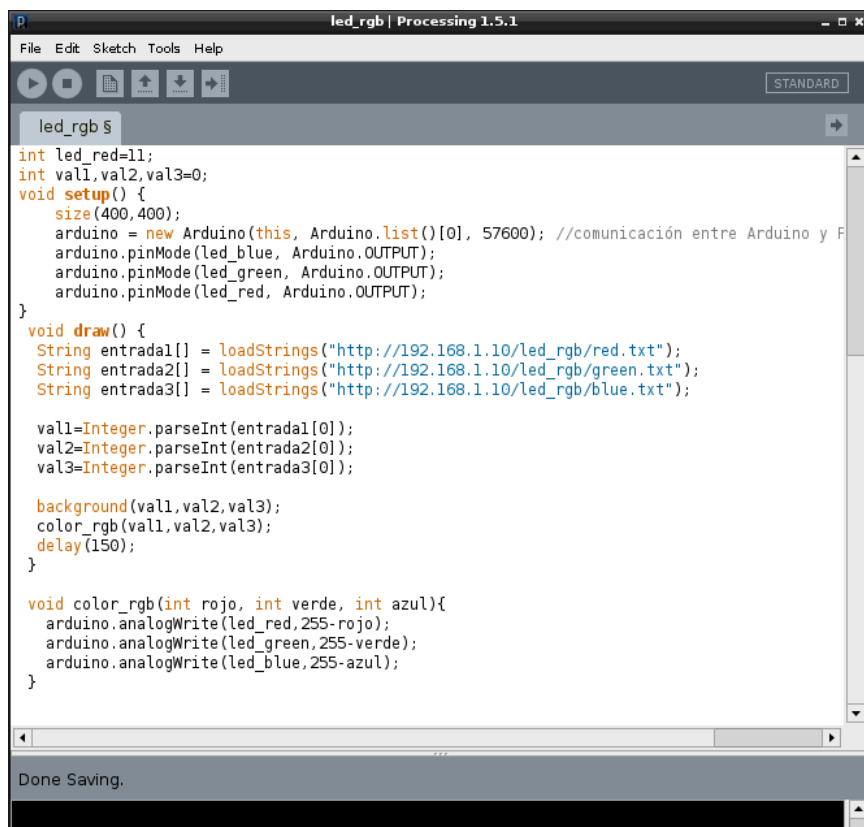
Serial port;

int led_blue=9;
int led_green=10;
int led_red=11;
int val1,val2,val3=0;
void setup() {
  size(400,400);
  arduino = new Arduino(this, Arduino.list()[0], 57600); //comunicación entre Arduino y F
  arduino.pinMode(led_blue, Arduino.OUTPUT);
  arduino.pinMode(led_green, Arduino.OUTPUT);
  arduino.pinMode(led_red, Arduino.OUTPUT);
}
void draw() {
  String entrada1[] = loadStrings("http://192.168.1.10/led_rgb/red.txt");
  String entrada2[] = loadStrings("http://192.168.1.10/led_rgb/green.txt");
  String entrada3[] = loadStrings("http://192.168.1.10/led_rgb/blue.txt");

  val1=Integer.parseInt(entrada1[0]);

Done Saving.
```

Parte 1



```
led_rgb | Processing 1.5.1
File Edit Sketch Tools Help

int led_red=11;
int val1,val2,val3=0;
void setup() {
  size(400,400);
  arduino = new Arduino(this, Arduino.list()[0], 57600); //comunicación entre Arduino y F
  arduino.pinMode(led_blue, Arduino.OUTPUT);
  arduino.pinMode(led_green, Arduino.OUTPUT);
  arduino.pinMode(led_red, Arduino.OUTPUT);
}
void draw() {
  String entrada1[] = loadStrings("http://192.168.1.10/led_rgb/red.txt");
  String entrada2[] = loadStrings("http://192.168.1.10/led_rgb/green.txt");
  String entrada3[] = loadStrings("http://192.168.1.10/led_rgb/blue.txt");

  val1=Integer.parseInt(entrada1[0]);
  val2=Integer.parseInt(entrada2[0]);
  val3=Integer.parseInt(entrada3[0]);

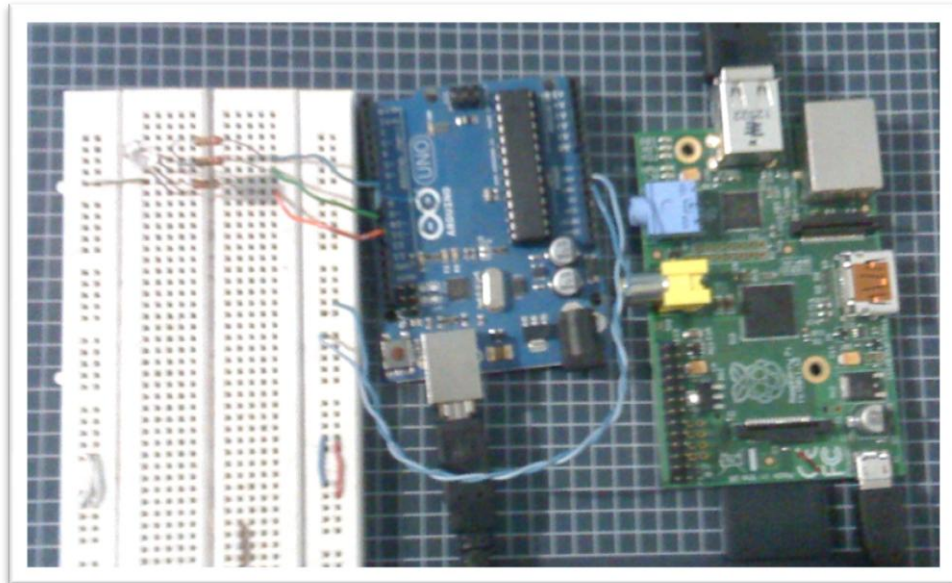
  background(val1,val2,val3);
  color_rgb(val1,val2,val3);
  delay(150);
}

void color_rgb(int rojo, int verde, int azul){
  arduino.analogWrite(led_red,255-rojo);
  arduino.analogWrite(led_green,255-verde);
  arduino.analogWrite(led_blue,255-azul);
}

Done Saving.
```

Parte 2

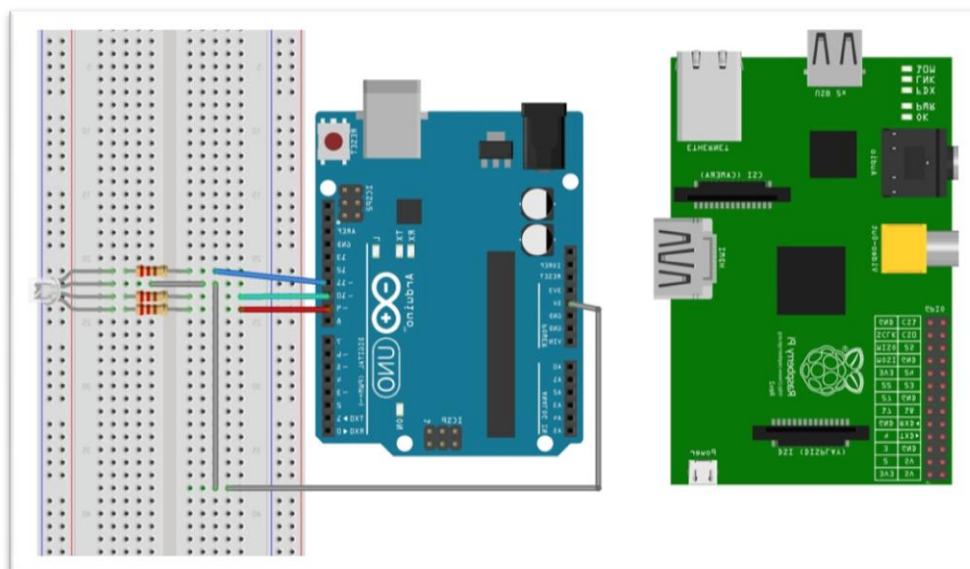
3. EXPLICACIÓN DEL HARDWARE



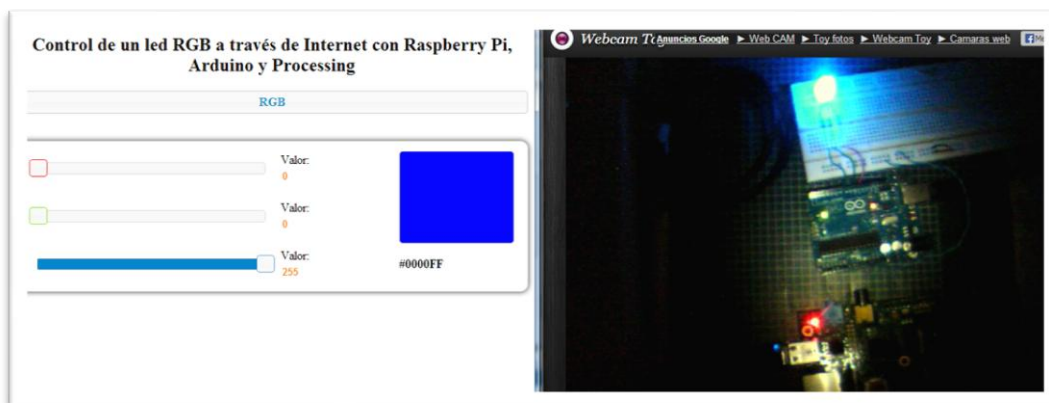
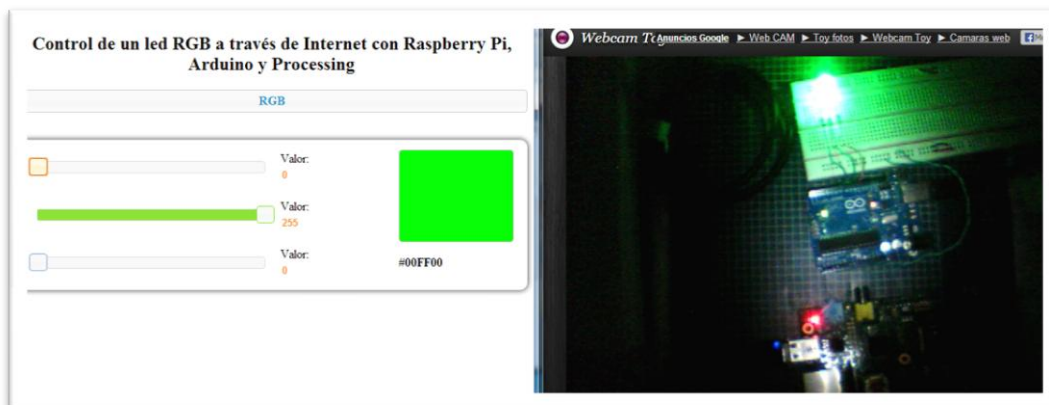
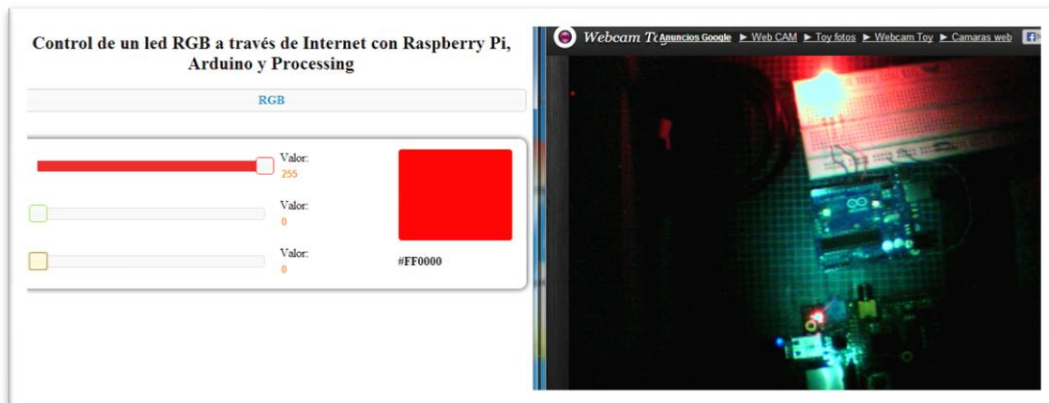
La placa Arduino está conectada a la Raspberry Pi a través del cable USB.

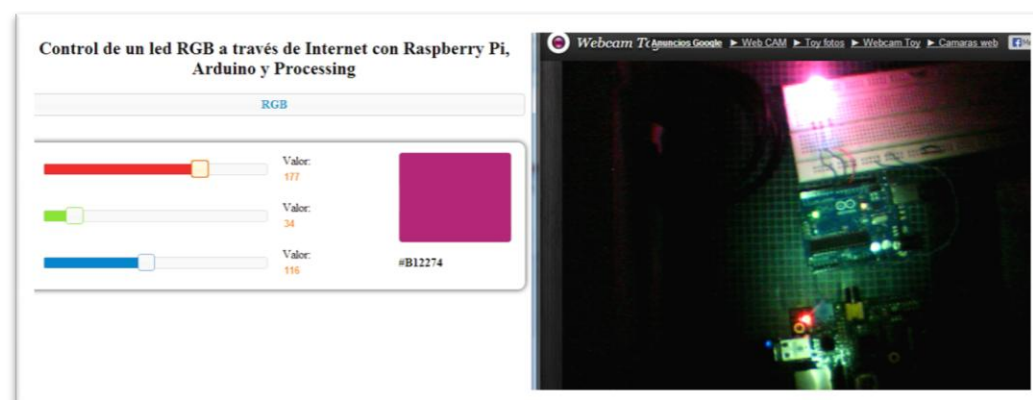
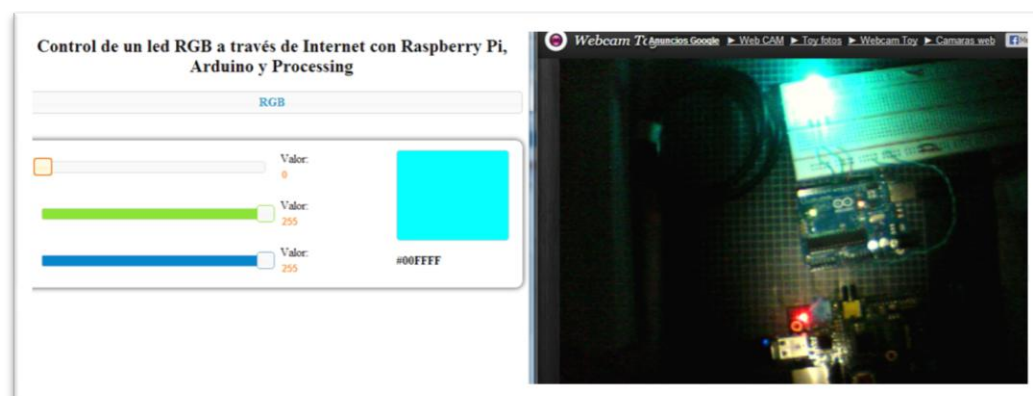
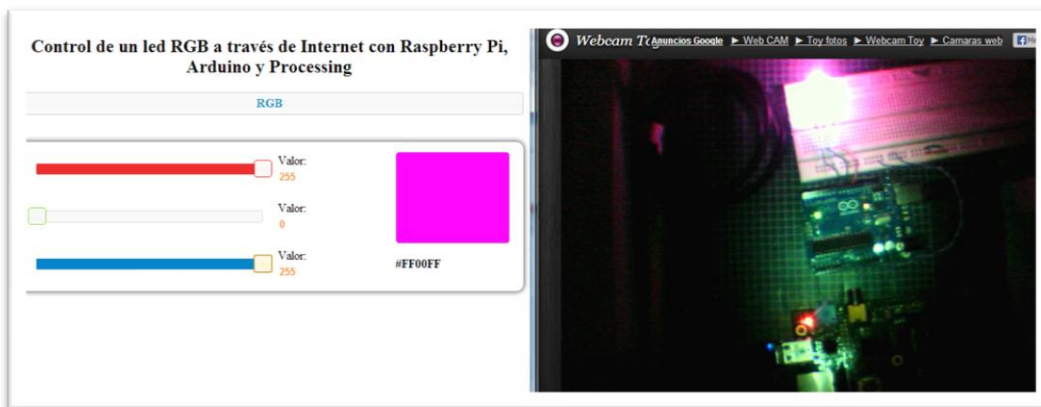
Del Arduino salen 4 cables, 3 salidas análogas, que van al led RGB, y 5v al cátodo común.

- 9-> Pin Red.
- 10-> Pin Green.
- 11-> Pin Blue.
- 5v-> Cátodo común.



4. PRUEBA DE FUNCIONAMIENTO





5. BIBLIOGRAFÍA

<http://firmata.org/>

<http://jqueryui.com/>

<http://arduino.cc/es/>

<http://processing.org/>

<http://scruss.com/blog/2012/08/12/controlling-an-arduino-from-raspberry-pi-using-processing/>

<http://www.ledfacil.com.ar/LEDs%20RGB%20demo.pdf>

<http://playground.arduino.cc/Interfacing/Processing>

ANEXOS

Código fuente de la aplicación:

http://jeffersonrivera.com/pi/led_rgb.zip

Desarrollado por:

Jefferson Rivera Patiño

@riverajefer

riverajefer.blogspot.com

jeffersonrivera.com