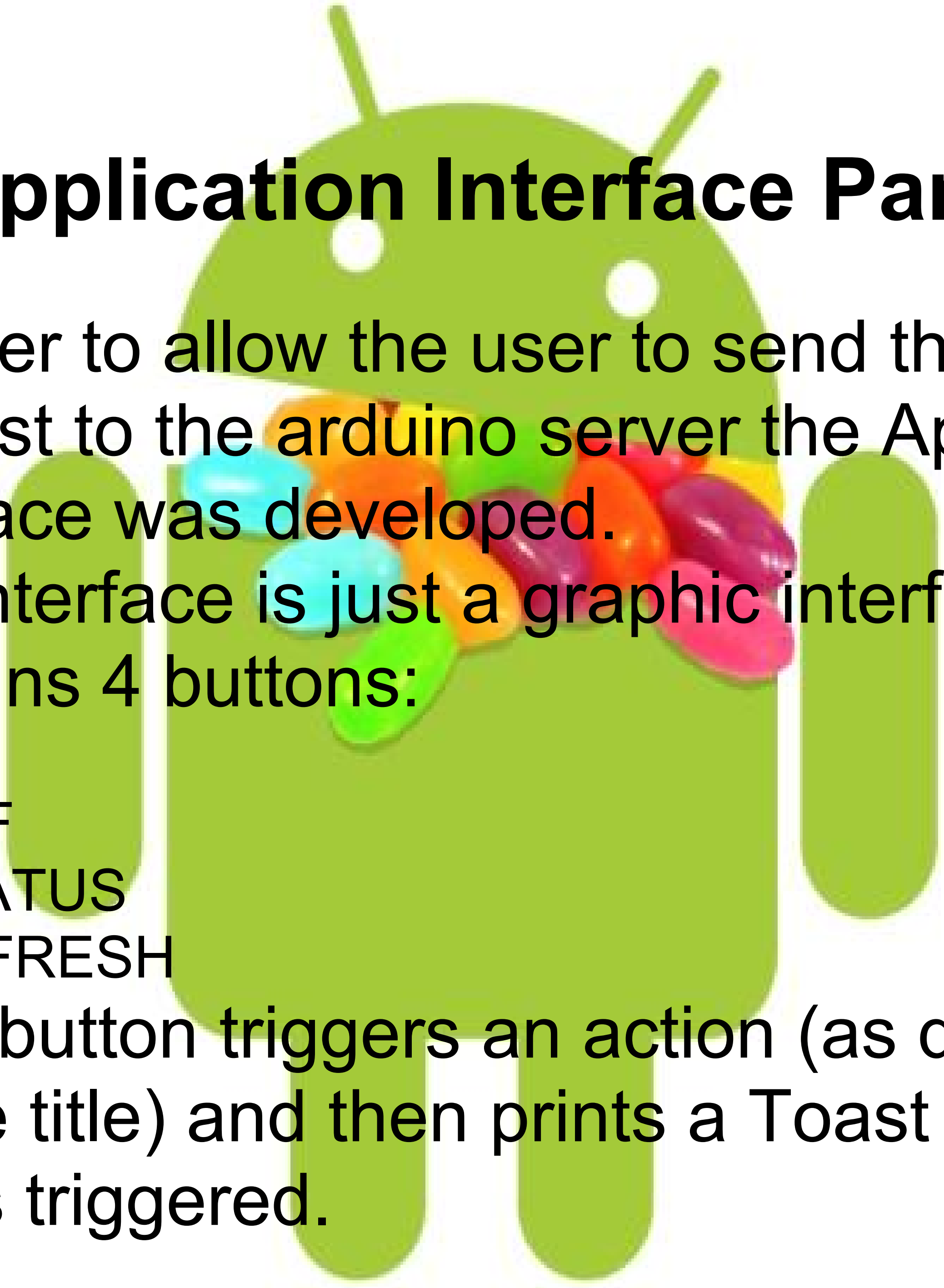
Automate your tasks using Arduino and Android

Presented By:

- Kaponis Georgios(Kaponakis)
- Mageirias Anastasios(mag)
- Karastathis Vaios(vKaras)
- Boviatsis Dimitrios



Application Interface Part

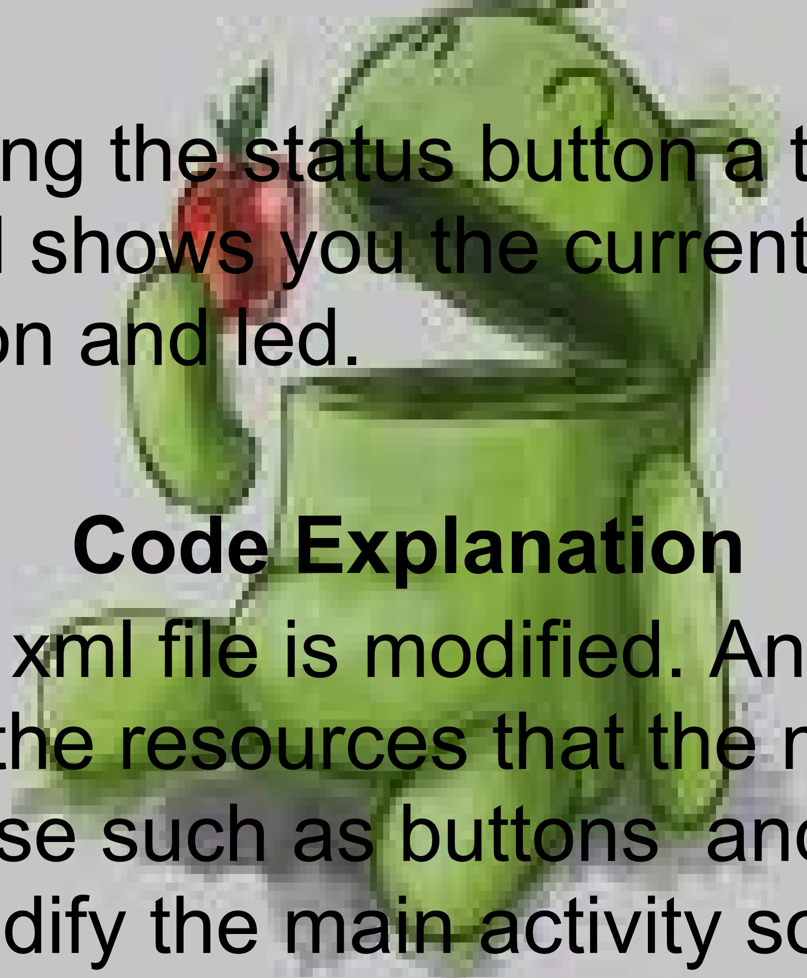
- In order to allow the user to send the http request to the arduino server the Application Interface was developed.
- The interface is just a graphic interface that contains 4 buttons:
  - ON
  - OFF
  - STATUS
  - REFRESH
- Each button triggers an action (as described by the title) and then prints a Toast about the status triggered.

Application Interface Part

After triggering the status button a toast appears and shows you the current status such as connection and led.

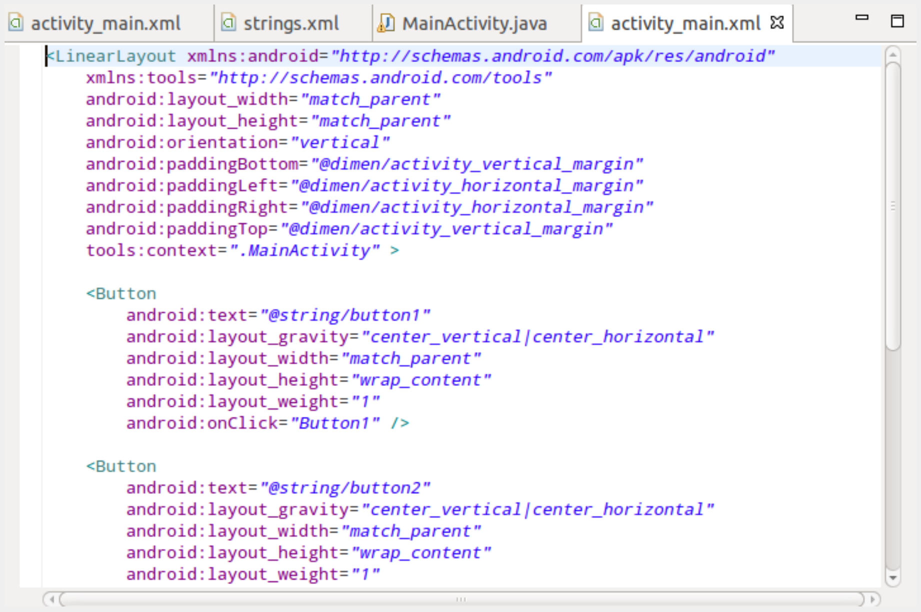
Code Explanation

Basically an xml file is modified. An xml file contains all the resources that the main activity is going to use such as buttons and strings. Then we modify the main activity so that by clicking the button another action is triggered.



Code Explanation

The xml file as seen on Eclipse IDE

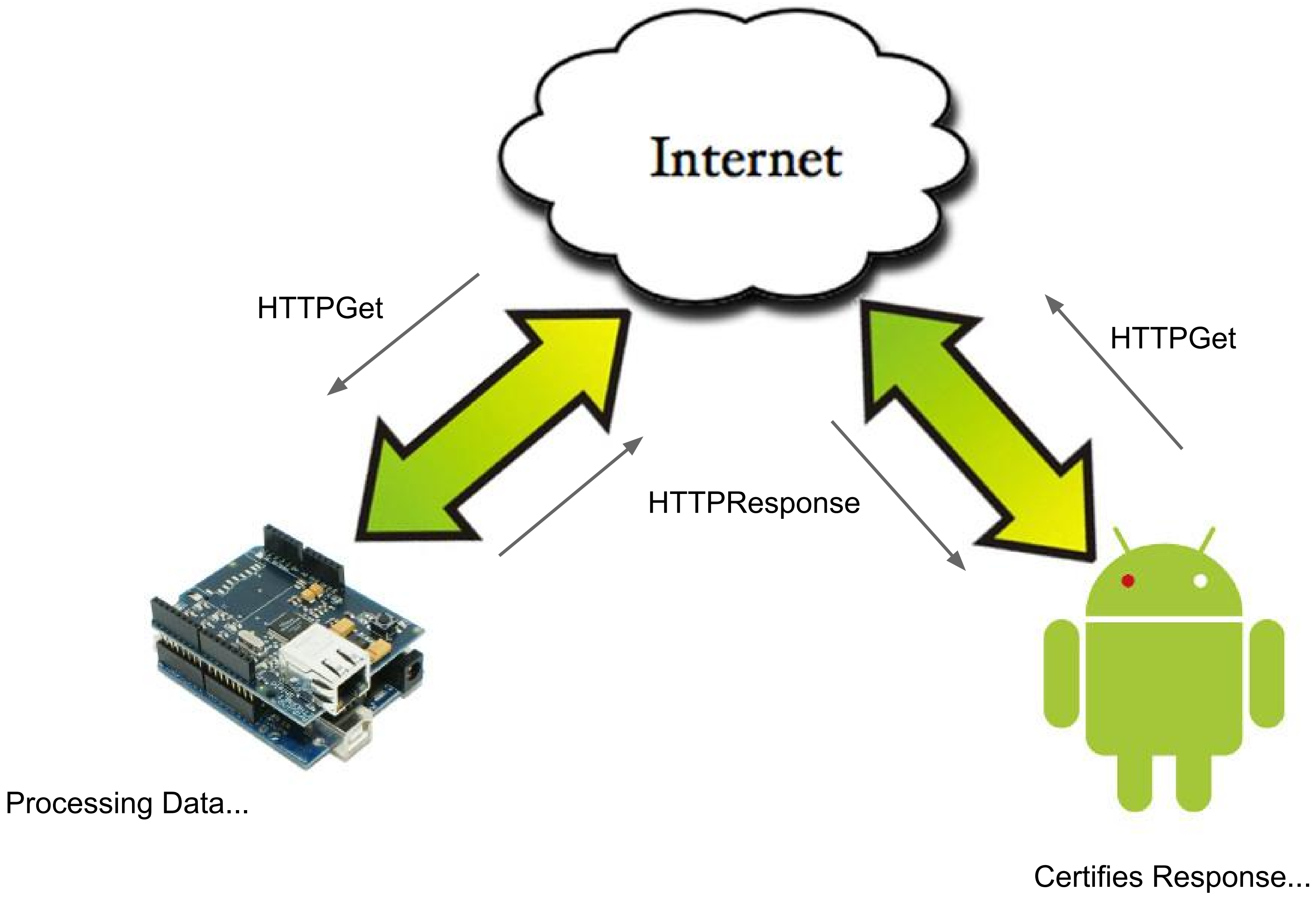


```xml
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity" >

    <Button
        android:text="@string/button1"
        android:layout_gravity="center_vertical|center_horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:onClick="Button1" />

    <Button
        android:text="@string/button2"
        android:layout_gravity="center_vertical|center_horizontal"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
```

Android Networking Part

- Arduino communicates with android via HTTP requests, using ethernet shield.
- As an Http Client we used Apache Http Client instead of HttpUrlConnection, because it has fewer bugs on Eclair (API Level 7) and on earlier versions.
- We used HTTP GET requests because it was easier both for android and arduino development (it can also be accomplished with POST requests)
- So we send certain predefined HTTP GET requests (pressing buttons) to Arduino, ordering it to execute some snippets of code.

Android - Arduino Communication



Android Networking Part

Explanation of used functions

- Refresh()
- getInputStreamFromUrl(String url)
- convertStreamToString(InputStream is)
- htmlrem(String t)

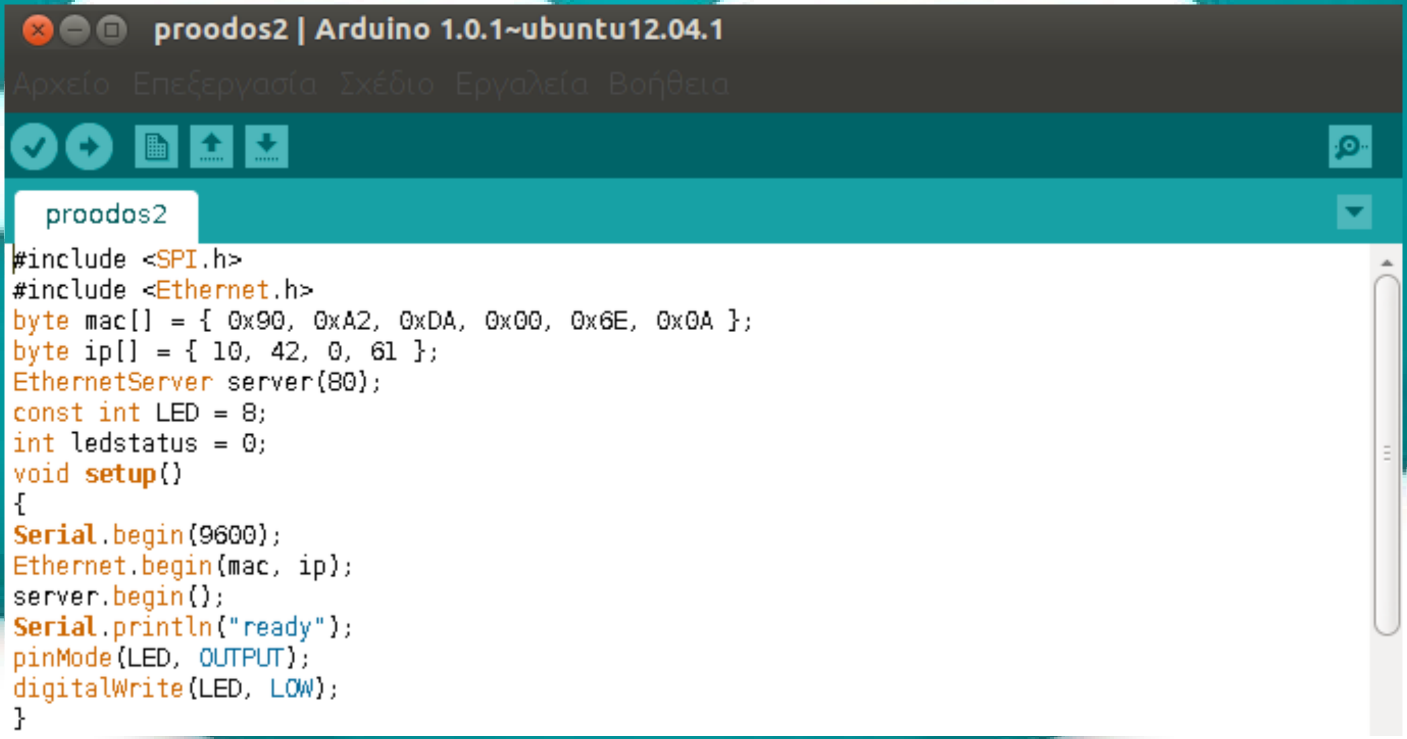
Programming Arduino Part

- In order to give arduino the chance to receive commands from internet we hook it up with the ethernet shield :

- Then, we start a
 server at the shield
that is running at port 80 (the default port for the http requests).

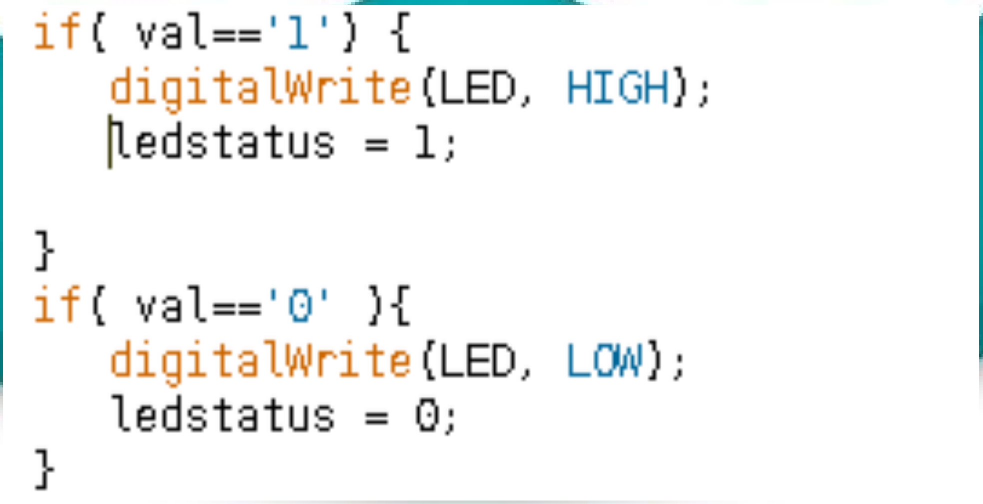
- As mentioned previously, this server is handling 'GET' requests.

- The code as seen on Arduino IDE:


proodos2 | Arduino 1.0.1~ubuntu12.04.1
Αρχείο Επεξεργασία Σχέδιο Εργαλεία Βοήθεια
proodos2

```
#include <SPI.h>
#include <Ethernet.h>
byte mac[] = { 0x90, 0xA2, 0xDA, 0x00, 0x6E, 0x0A };
byte ip[] = { 10, 42, 0, 61 };
EthernetServer server(80);
const int LED = 8;
int ledstatus = 0;
void setup()
{
Serial.begin(9600);
Ethernet.begin(mac, ip);
server.begin();
Serial.println("ready");
pinMode(LED, OUTPUT);
digitalWrite(LED, LOW);
}
```

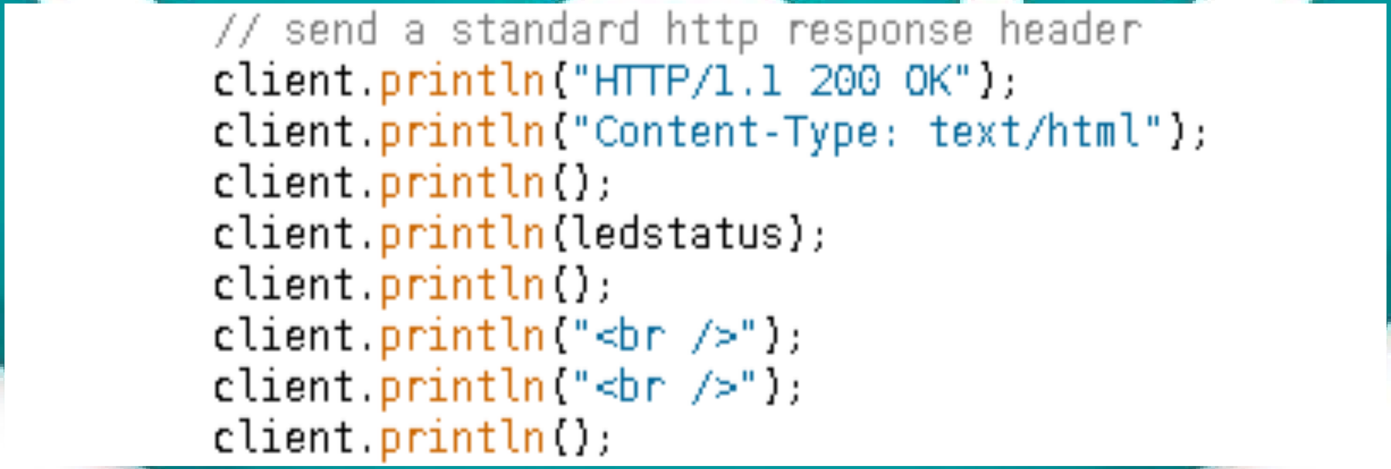
- In order to understand that the incoming request wants arduino to perform an action (turn the lamp 'ON', 'OFF' or return its current status) the prefix "pin" is used in the http request.
- After the prefix follows a number (0, 1 or 2).
- Each of these numbers corresponds to a specific action. 1 turns the lamp 'ON', 0 turns it 'OFF' and 2 returns its current status



```
if( val=='1') {
    digitalWrite(LED, HIGH);
    ledstatus = 1;

}
if( val=='0' ){
    digitalWrite(LED, LOW);
    ledstatus = 0;
}
```

- In the final part of the code, the result is printed in html form.
- This type of response is used because it is easier for the Android App to read the reply.

- Here is the response code:

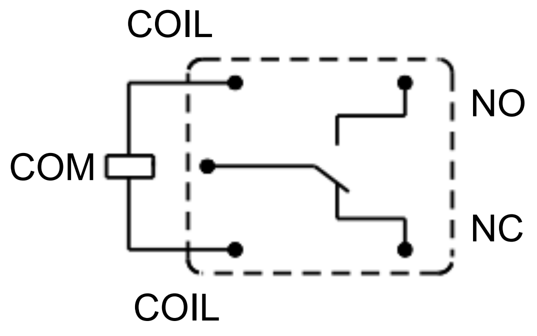


```
// send a standard http response header
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println();
client.println(ledstatus);
client.println();
client.println("<br />");
client.println("<br />");
client.println();
```

Using Arduino to Control Relays

So, what exactly is a relay?

- Coil
- Switch



Materials

- Arduino Uno + Ethernet Shield
- Rectifier Diode 1N4004
- Relay SRD-S-106D
- Resistor 1kΩ
- NPN Transistor 2N2222
- Work Lamp
- Insulating Tape

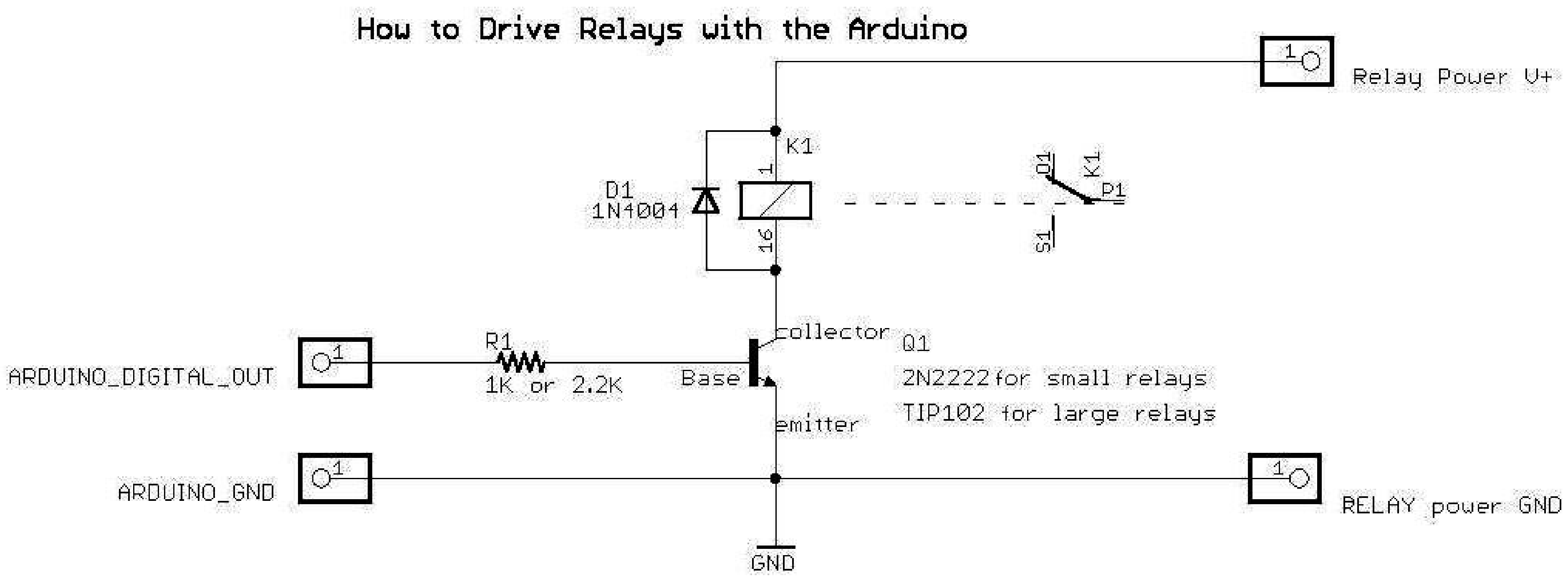
Safety

- Employ an electrician
- Don't work with relay or extension cords while plug is connected to socket
- Enclose any wires before testing

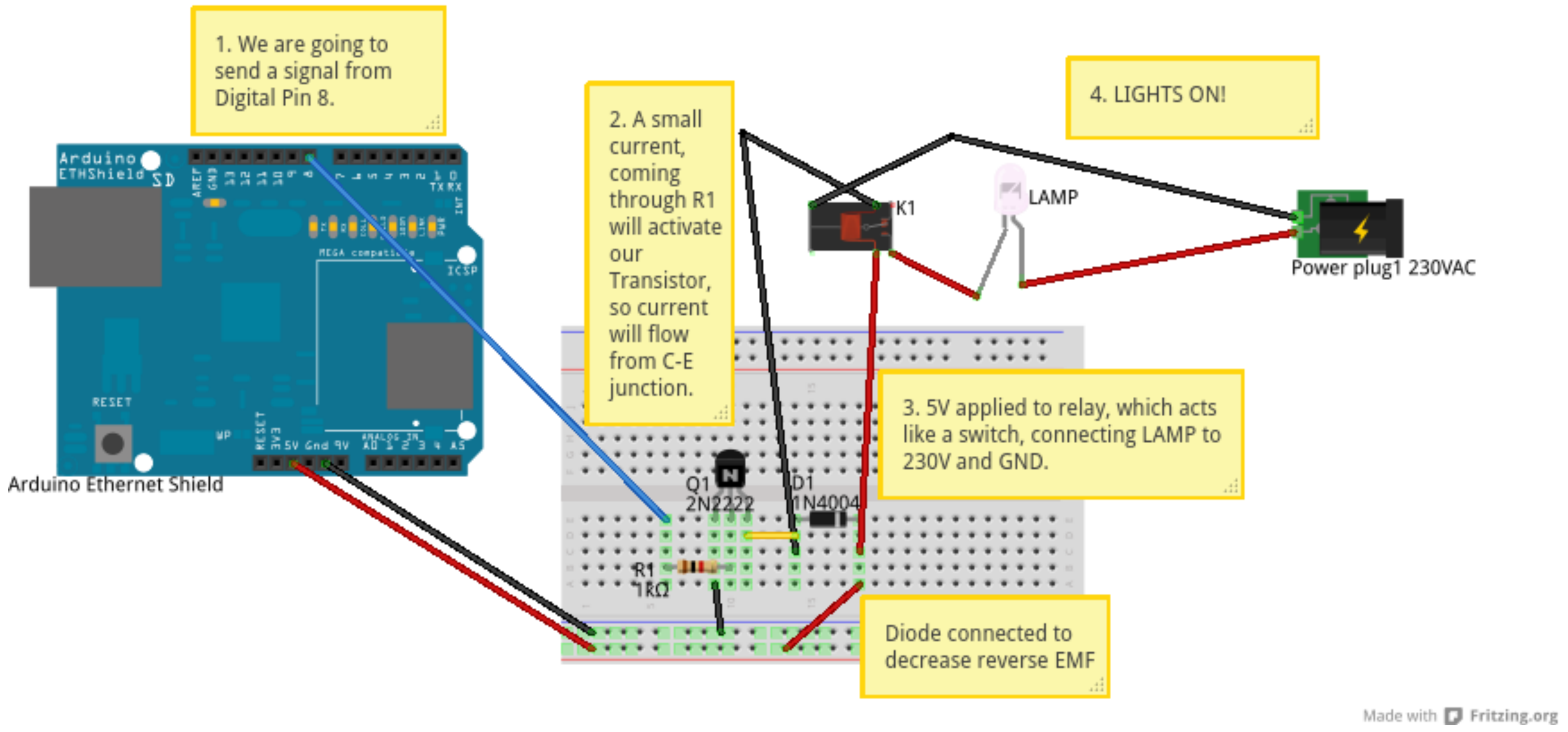
Preparation

- Cut the cord in half
- Solder "active"(black or brown) wire with coil pins
- Solder the wires with the same color together
- Insulate properly

Schematic


How to Drive Relays with the Arduino
1
Relay Power V+
K1
1
D1
1N4004
16
O1
K1
P1
S1
R1
1K or 2.2K
ARDUINO_DIGITAL_OUT
Base
collector
Q1
2N2222 for small relays
TIP102 for large relays
emitter
ARDUINO_GND
RELAY power GND
GND

Putting it All Together


1. We are going to send a signal from Digital Pin 8.
2. A small current, coming through R1 will activate our Transistor, so current will flow from C-E junction.
3. 5V applied to relay, which acts like a switch, connecting LAMP to 230V and GND.
4. LIGHTS ON!
Diode connected to decrease reverse EMF
Arduino Ethernet Shield
Q1 2N2222
D1 1N4004
R1 1kΩ
K1
LAMP
Power plug1 230VAC
Made with Fritzing.org

Fritzing Breadboard View