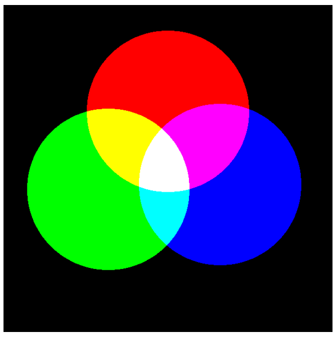
maXbox



# Arduino Teach 2014

Start with Web of Things Course

## 1.1 Program Physical Computing with Things

Today we jump into a complete new topic in programming called web of things or embedded computing with the internet;
There are three main topics in here. First technologies – simply put, this part is mainly for early adopters. It's about coding, developing toys, plugging in kettles on the web (and we and many others actually did that!).
The second part is about new ideas, prototyping and new technologies that are in the lab. It's about research papers, and software philosophy, and about researchers worldwide. Third part is about end-users and products. That's the heaviest part of all.
I will show the topic with 6 examples in one Day:

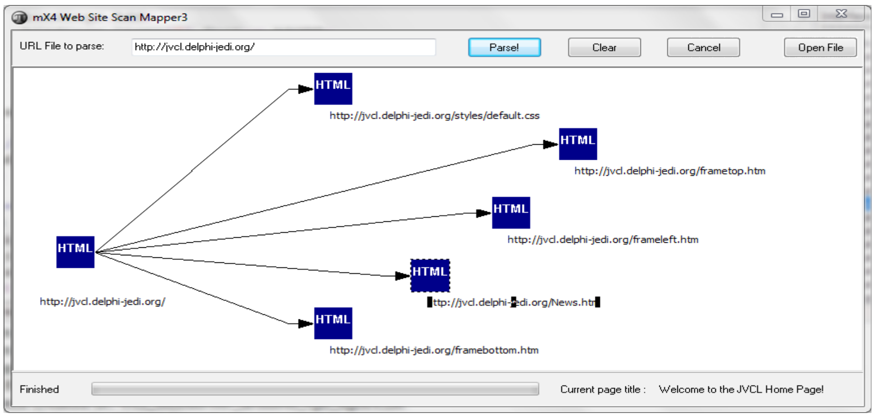
- RGB LED in Arduino with an Oscilloscope
- Generating QR Code
- 3D Printing Simulation
- Web Video Cam
- Digi Clock with Real Time Clock
- Android SeekBar to Arduino LED Matrix

It's about products that exist, or at least should exist (or should not!). It's about marketing, fun (functionality) and new ideas out there.
As more "things" on planet Earth are converted to the inventory set of digitally connected Internet devices, the roles and responsibilities of web developers and technology managers will need to evolve.

So we code a RGB LED light on the Arduino board and a breadboard on which we switch off or on the light by a browser on an android device with our own web server and their COM protocols too. Hope you did already work with the Starter 1 till 29 (especially the 18) at:

http://sourceforge.net/apps/mediawiki/maxbox/

Arduino hardware is programmed using a Wiring-based language (syntax and libraries), similar to C++ and Object Pascal with some slight simplifications and modifications, and a Processing-based integrated development environment like Delphi or Lazarus with Free Pascal.
Current versions can be purchased pre-assembled; hardware design information is available for those who would like to assemble an Arduino by hand. Additionally, variations of the Italian-made Arduino—with varying levels of compatibility—have been released by third parties; some of them are programmed using the Arduino software or the sketch firmware in AVR.
The Arduino is what is known as a Physical or Embedded Computing platform, which means that it is an interactive system that through the use of hardware-shields and software can interact with its environment.


mX4 Web Site Scan Mapper3
Parse!
Clear
Cancel
Open File
HTML
http://jvcl.delphi-jedi.org/
http://jvcl.delphi-jedi.org/styles/default.css
http://jvcl.delphi-jedi.org/frametop.htm
http://jvcl.delphi-jedi.org/frameleft.htm
http://jvcl.delphi-jedi.org/News.htm
http://jvcl.delphi-jedi.org/framebottom.htm

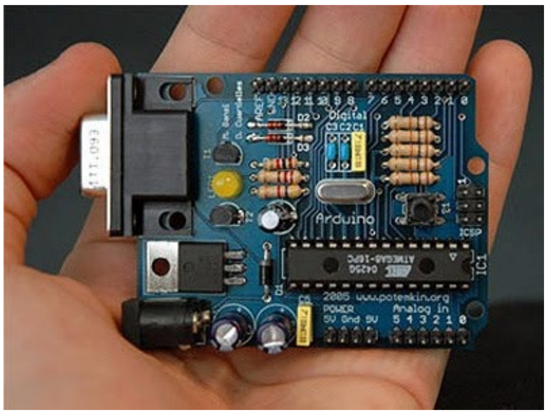
URL File to parse: http://jvcl.delphi-jedi.org/

Finished    Current page title : Welcome to the JVCL Home Page!

This lesson will introduce you to Arduino and the Serial communication (see Tutorial 15). We will now dive into the world of serial communications and control our lamp from a browser to a web server by sending commands from the PC to the Arduino using the Serial Monitor and Interface.

In our case we explain one example of a HTTP server which is an intermediate to the COM serial communication with the AVR micro controller on Arduino[1].
Another Controller is the Delphi Controller. The Delphi Controller and DelphiDevBoard were designed to help students, Pascal programmers and electronic engineers understand how to program micro controllers and embedded systems especially in programming these devices and targets (see Appendix).

This is achieved by providing hardware (either pre-assembled or as a DIY kit of components), using course material, templates, and a Pascal compatible cross-compiler and using of a standard IDE for development and debugging (Delphi, maXbox, Lazarus or Free Pascal).



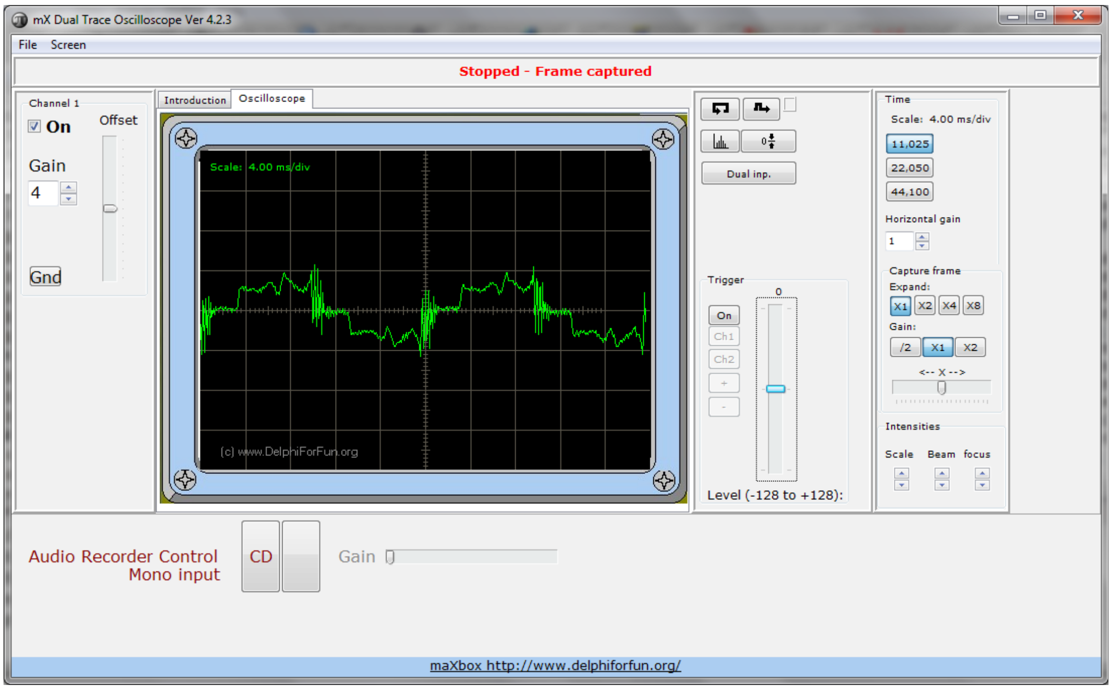
Let's begin with HTTP (Hypertext Transfer Protocol) and TCP. TCP/IP stands for Transmission Control Protocol and Internet Protocol. TCP/IP can mean many things, but in most cases, it refers to the network protocol itself.

[1] An Arduino board consists of an 8-bit Atmel AVR microcontroller, or an ARM cortex on the Due

Each computer on a TCP/IP network has a unique address associated with it, the so called IP-Address. Some computers may have more than one address associated with them. An IP address is a 32-bit number and is usually represented in a dot notation, e.g. 192.168.0.1. Each section represents one byte of the 32-bit address. In maXbox a connection with HTTP represents an Indy object.
In our case we will operate with the local host. It is common for computers to refer to themselves with the name local host and the IP number 127.0.0.1.

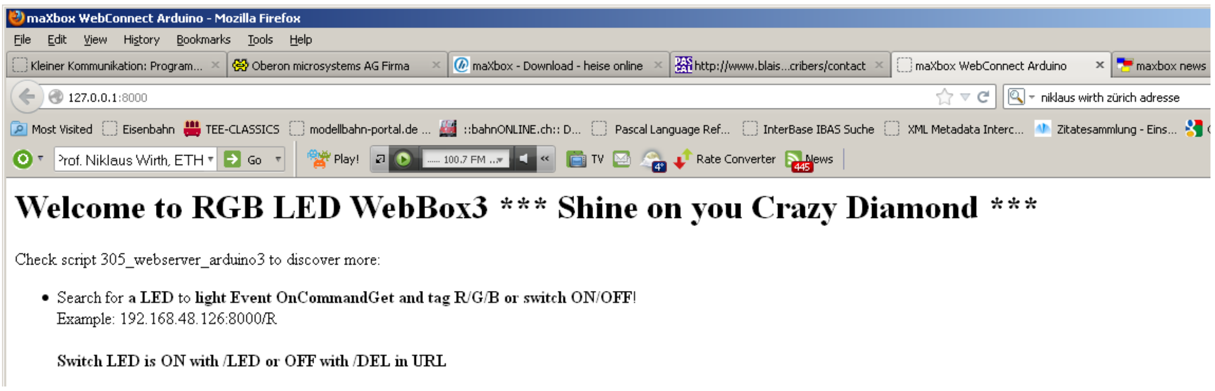
When HTTP is used on the Internet, browsers like Firefox on Android act as clients and the application that is hosting the website like softwareschule.ch acts as the server.

**Host names** are "human-readable" names for IP addresses.



With this oscilloscope you can measure traffic or micro controllers in /Options/..

```
if (val=='1'){
     digitalWrite(ledPin11,HIGH); }
else if (val=='A'){
     digitalWrite(ledPin11,LOW);
     }
if (val=='2'){
     digitalWrite(ledPin12,HIGH); }
```


maXbox WebConnect Arduino - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Kleiner Kommunikation: Program...
Oberon microsystems AG Firma
maXbox - Download - heise online
http://www.blais...cribers/contact
maXbox WebConnect Arduino
maxbox news
127.0.0.1:8000
niklaus wirth zürich adresse
Most Visited
Eisenbahn
TEE-CLASSICS
modellbahn-portal.de ...
::bahnONLINE.ch:: D...
Pascal Language Ref...
InterBase IBAS Suche
XML Metadata Interc...
Zitatesammlung - Eins...
Prof. Niklaus Wirth, ETH
Go
Play!
100.7 FM
TV
Rate Converter
News

**Welcome to RGB LED WebBox3 *** Shine on you Crazy Diamond *****

Check script 305_webserver_arduino3 to discover more:

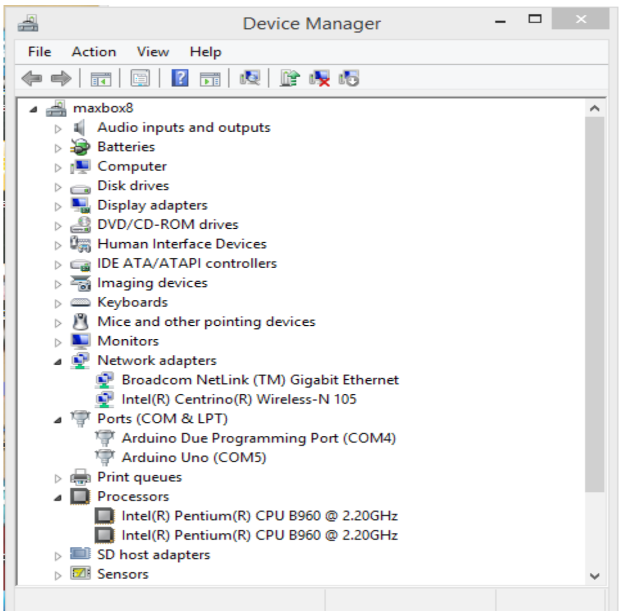
- Search for a **LED** to **light Event OnCommandGet and tag R/G/B or switch** ON/OFF!
  Example: 192.168.48.126:8000/R

  **Switch LED is ON with /LED or OFF with /DEL in URL**

## 3: A Browser set

When the browser starts from script the server is ready for commands to pass as chars in line 59 or 60 to serial communication. When a the server application finishes with our client request, it lights the LED and constructs a page of HTML code or other MIME content, and passes the result back (via server in `TIdHTTPResponseInfo` ) to the client for display.

```
61    writeln(localcom+ ': LED on');
62    RespInfo.ContentText:= getHTMLContentString('LED is:  ON');
```



```
◢ maxbox8
   ▷ Audio inputs and outputs
   ▷ Batteries
   ▷ Computer
   ▷ Disk drives
   ▷ Display adapters
   ▷ DVD/CD-ROM drives
   ▷ Human Interface Devices
   ▷ IDE ATA/ATAPI controllers
   ▷ Imaging devices
   ▷ Keyboards
   ▷ Mice and other pointing devices
   ▷ Monitors
   ◢ Network adapters
        Broadcom NetLink (TM) Gigabit Ethernet
        Intel(R) Centrino(R) Wireless-N 105
   ◢ Ports (COM & LPT)
        Arduino Due Programming Port (COM4)
        Arduino Uno (COM5)
   ▷ Print queues
   ◢ Processors
        Intel(R) Pentium(R) CPU B960 @ 2.20GHz
        Intel(R) Pentium(R) CPU B960 @ 2.20GHz
   ▷ SD host adapters
   ▷ Sensors
```

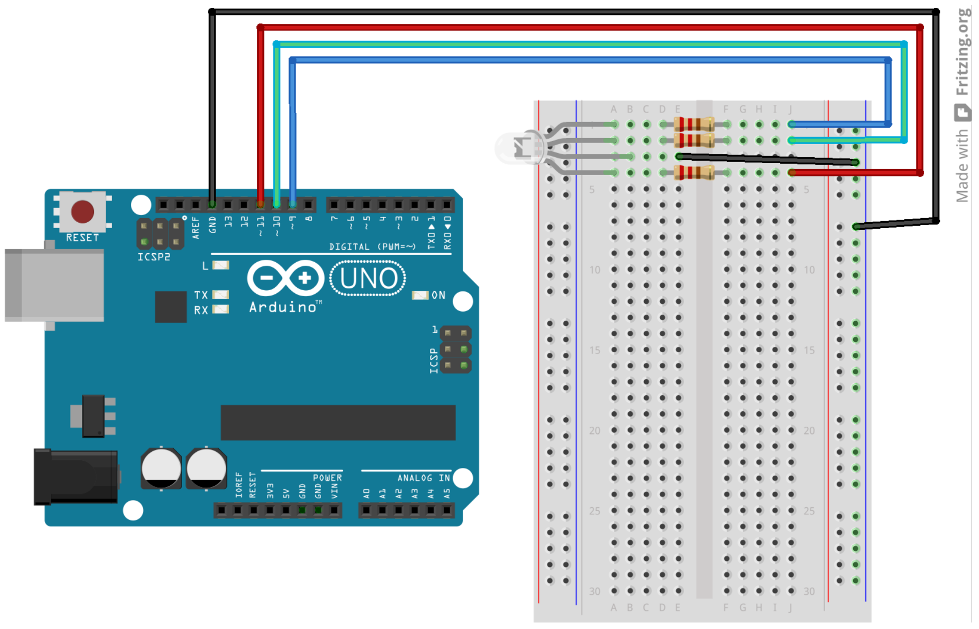
Have you tried the program, it's also possible to test the server without Arduino or a browser. The **Compile** button is also used to check that your code is correct, by verifying the syntax before the program starts. Another way to check syntax before run is F2 or the **Syntax Check** in menu `Program`. When you run this code from the script `102_pas_http_download.txt` you will see a content (first 10 lines) of the site in HTML format with a help of method `memo2.lines.add`:

```
begin
  idHTTP:= TIdHTTP.Create(NIL)
  try
    memo2.lines.text:= idHTTP.Get2('http://127.0.0.1')
    for i:= 1 to 10 do
      memo2.lines.add(IntToStr(i)+' :'+memo2.lines[i])
  finally
    idHTTP.Free
end
```

It downloads the entire file into memory if the data is compressed (Indy does not support streaming decompression for HTTP yet). Next we come closer to the main event of our web server, it's the event `onCommandGet` with the corresponding event handler method `@HTTPServerGet()` and one object of `TIdPeerThread.`  You can use them as server to serve files of many kinds!

## 4: COM Port Settings

Resistors don't have a direction, so it doesn't matter which way it goes.



5: Breadboard Settings

If you're using a standard breadboard, you'll need to use wires to reach the Arduino. Run 3 wires (red, green and blue) to the pin sockets on the Arduino. Run the other wire (black) to one of the GND sockets on the Arduino. The colours aren't essential but they will help you remember what the wires are connected to and black is a convention for ground GND!

Close to the end of the first part some knowledge about sockets. Socket connections can be divided into three basic types, which reflect how a connection was initiated and what a local socket is connected to. These are

- Client connections.
- Listening connections.
- Server connections.

Once the connection to a client socket is completed, the server connection is indistinguishable from a client connection. Both end points have the same capabilities and receive the same types of events. Only the listening connection is fundamentally different, as it has only a single endpoint.
Sockets let your network application communicate with other systems over the network. Each socket can be viewed as an endpoint in a network connection. It has an address that specifies:

- The system on which it is running.
- The types of interfaces it understands.
- The port it is using for the connection.

A full description of a socket connection includes the addresses of the sockets on both ends of the connection. You can describe the address of each socket endpoint by supplying both the IP address or host and the port number.

In the next line we just start a browser to test our server in a so called frame work flow ☺

```
procedure letOpenBrowser;
// TS_ShellExecuteCmd = (seCmdOpen,seCmdPrint,seCmdExplore);
begin
//ShellAPI.ShellExecute(Handle,PChar('open'),'http://127.0.0.1:80/',Nil,Nil,0);
  S_ShellExecute('http:'+IPADDR+':'+IntToStr(APORT)+'/','',seCmdOpen)
end;
```

Try to change the IP address in line 132 of `IP:= IPADDR` with a DHCP or dynDNS address, so you can reach Arduino from an Android, but change also the const in line 13.

Try to get data back from Arduino as a test case like this:

```
Serial.print() in Arduino and cPort.ReadStr() in maXbox
Function ReadStr(var Str: string; Count: Integer): Integer'); //CPort Lib
//S_ShellExecute('http:'+IPADDR+':'+IntToStr(APORT)+'/soa_delphi.pdf','',seCmdOpen)
```

### 1.1.1 RTC

For the real time clock, you guess, you need another script and libraries too.
The Arduino has a fairly precise internal clock, but it’s not the most well-suited device to keep track of date and time. To get a more accurate date and time with a little backup, we need to use a real-time clock or RTC module. For this we use the well known DS1307 RTC available on a breakout board from SparkFun Electronics /Adafruit Industries, schema in appendix!

The DS1307 is a relatively simple device that uses the I2C bus, or sometimes called Two Wire—although these two terms are not always interchangeable - to send a 7-bit binary number for the current date and time.
The DS1307 Serial Real-Time Clock is a low-power, full binary-coded decimal (BCD) clock/calendar plus 56 bytes of NV SRAM. Address and data are transferred serially via a 2-wire, bi-directional bus called I2C.
I2C is an interesting serial protocol that uses two wires, SDA for data and SCL for clock, to control up to 112 slave devices such as sensors, clocks, heart-beats, FM radios, and smart LEDs, from one master device like our micro-controller. In addition to the data and clock pins, each I2C device will also need a positive and ground connection to a power supply rated for a device.

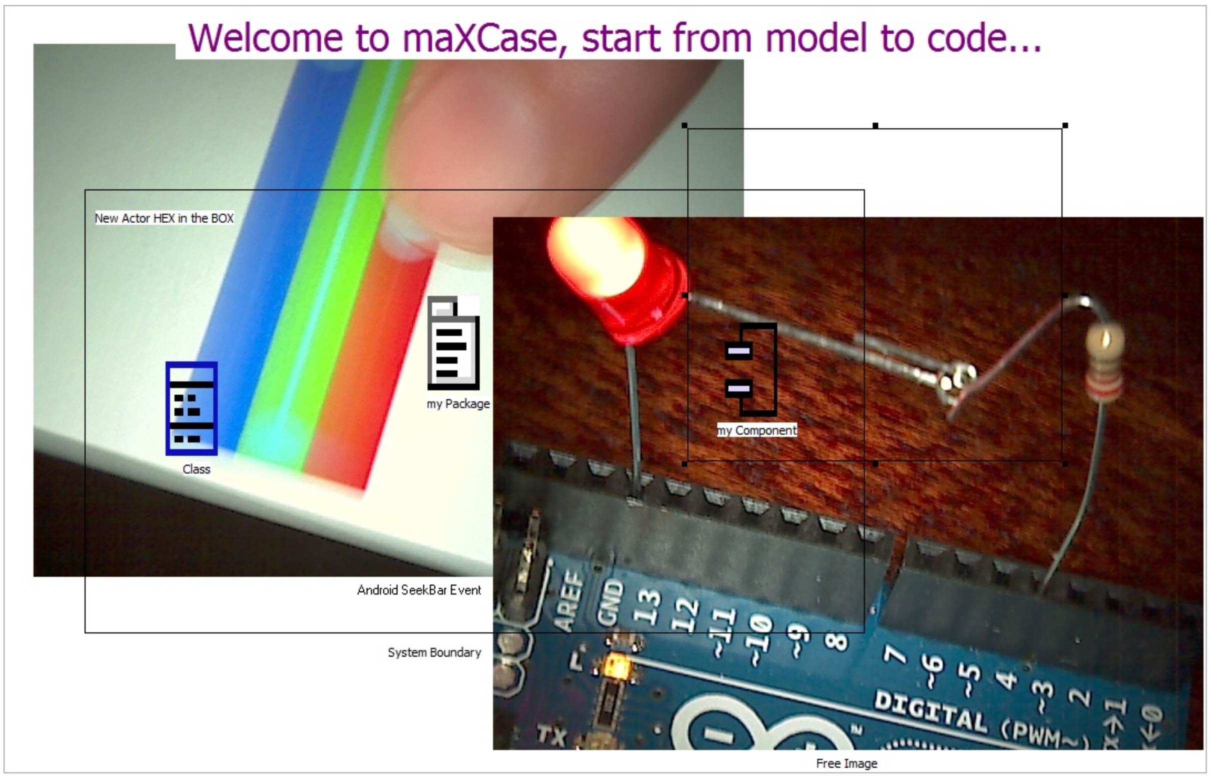
- SCL (Serial Clock Input) – SCL is used to synchronize data movement on the serial interface.
- SDA (Serial Data Input/Output) – SDA is the input/output pin for the 2-wire serial interface. The SDA pin is open drain which requires an external pull-up resistor.

The DS1307 operates as a slave device on the serial bus. Access is obtained by implementing a START condition and providing a device identification code followed by a register address. Subsequent registers can be accessed sequentially until a STOP condition is executed.

## 1.2 Android with Arduino

Now at last the dream team with this USB host (I did a lot with my students).
The Arduino ADK is a combination of hardware and software designed to help people interested in designing accessories for Android devices.

During Google’s 2011 IO keynote, Google introduced a board based on the Arduino Mega 2560 which includes this USB host. A new USB library [3] was introduced, enabling data to be sent and received from any external devices.
http://labs.arduino.cc/ADK/Index
App makers can freely use it in their development. This library has been included in 2.3.4, and some devices have been reported to work with 2.3.3. I work with Andro 4.2.2 and Arduino 1.5.



8: A seek bar on Android dims the Red LED on Arduino

The Mega ADK board is a derivative of the Arduino Mega 2560. The modified Mega 2560 board includes a USB host chip. This host chip allows any USB device to connect to the Arduino.

There are three libraries needed to make the system RGB SeekBar work:

- MAX3421e: handles the USB host chip
- USB: handles the USB communication
- Android Accessory: checks if the device connecting is one of the available accessory-enabled phones

http://www.softwareschule.ch/examples/B004_RGB_Led.ino.htm

```
#include "variant.h"
#include <stdio.h>
#include <adk.h>
#include <Scheduler.h>

#define LED 13
#define RED 2
#define GREEN 3
#define BLUE 4

/* Aware: Serial has to be right below on newline! */
```

```
// Accessory descriptor. It's how Arduino identifies itself to Android.
char model[] = "HelloWorldModel"; // model in xml
char displayName[] = "A Hello World Accessory"; // your Arduino board
char companyName[] = "Hello maXbox Inc";

// Make up anything you want for these
char versionNumber[] = "1.2";
char serialNumber[] = "1";
char url[] = "http://www.osciprime.com/repo/OsciPrimeICS.apk";

volatile uint8_t pause = 255;
USBHost Usb;
ADK adk(&Usb, companyName, model,
displayName,versionNumber,url,serialNumber);
```

Arduino is an amazing device and will enable you to make anything from interactive works of art to robots. With a little enthusiasm to learn how to program the Arduino and make it interact with other components a well as a bit of imagination, you can build anything you want.
Arduino can also be extended with the use of Shields which circuit boards are containing other devices (e.g. GPS receivers, LED Cubes, LCD Displays, Drones, MIDI Synthesizers, Ethernet connections, etc.) that you can simply slot into the top of your Arduino to get extra functionality.

Arduino is an open-source single-board micro-controller, descendant of the open-source Wiring platform designed to make the process of using electronics in multitude projects more accessible. The Arduino can be connected to Dot matrix displays, glasses, switches, motors, temperature sensors, pressure sensors, distance sensors, web-cams, printers, GPS receivers, Ethernet modules and so on.

The Arduino board is made of an Atmel AVR microprocessor, a crystal or oscillator (basically a crude clock that sends time pulses to the micro-controller to enable it to operate at the correct what type of Arduino you have, you may also have a USB connector to enable it to be connected to a PC or Linux to upload or retrieve data. The board exposes the micro-controllers I/O (Input/Output) pins to enable you to connect those pins to other circuits, buses or sensors.

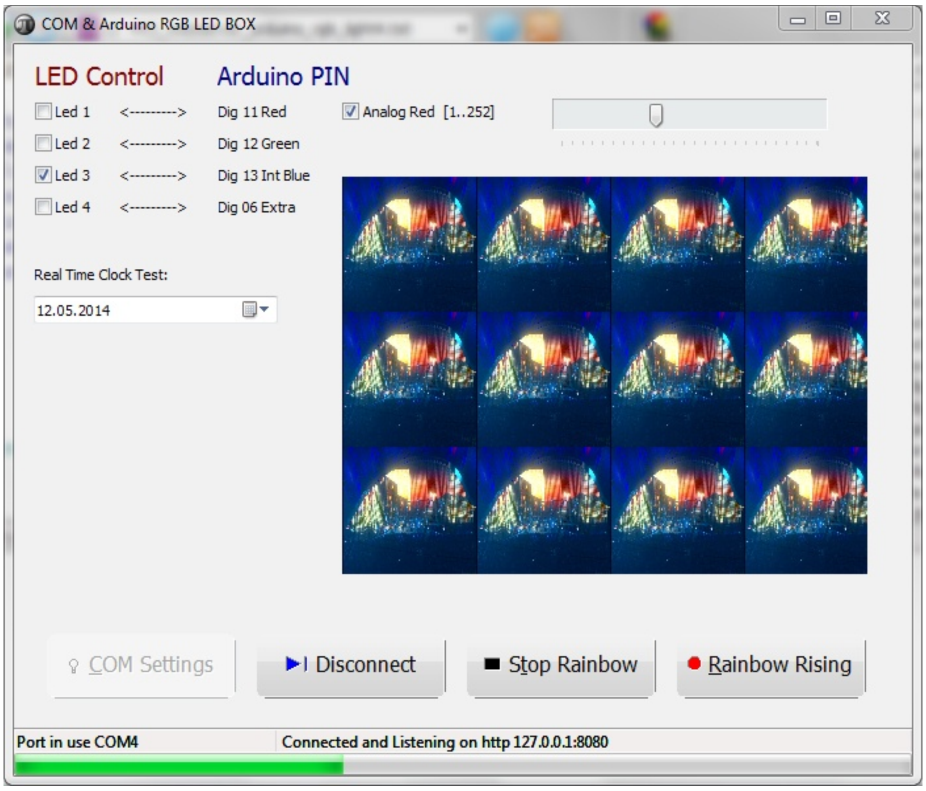
Feedback @
max@kleiner.com

Literature:
Kleiner et al., Patterns konkret, 2003, Software & Support
Links of maXbox, Web of Things, Arduino and Indy:

http://www.softwareschule.ch/download/webofthings2013.pdf
http://www.softwareschule.ch/download/Arduino_C_2014_6_basta_box.pdf

http://www.softwareschule.ch/maxbox.htm
http://www.softwareschule.ch/examples/443_webserver_arduino_rgb_light4.htm
http://en.wikipedia.org/wiki/Arduino
http://fritzing.org/

http://sourceforge.net/projects/maxbox
http://sourceforge.net/apps/mediawiki/maxbox/
http://www.blaisepascal.eu/subscribers/vogelaar_elctronics_info_english.php


COM & Arduino RGB LED BOX
LED Control
Arduino PIN
Led 1 <---------> Dig 11 Red
Led 2 <---------> Dig 12 Green
Led 3 <---------> Dig 13 Int Blue
Led 4 <---------> Dig 06 Extra
Analog Red [1..252]
Real Time Clock Test:
12.05.2014
COM Settings
Disconnect
Stop Rainbow
Rainbow Rising
Port in use COM4
Connected and Listening on http 127.0.0.1:8080

maXbox Arduino Framework

## 1.3 Appendix Arduino LED Rainbow

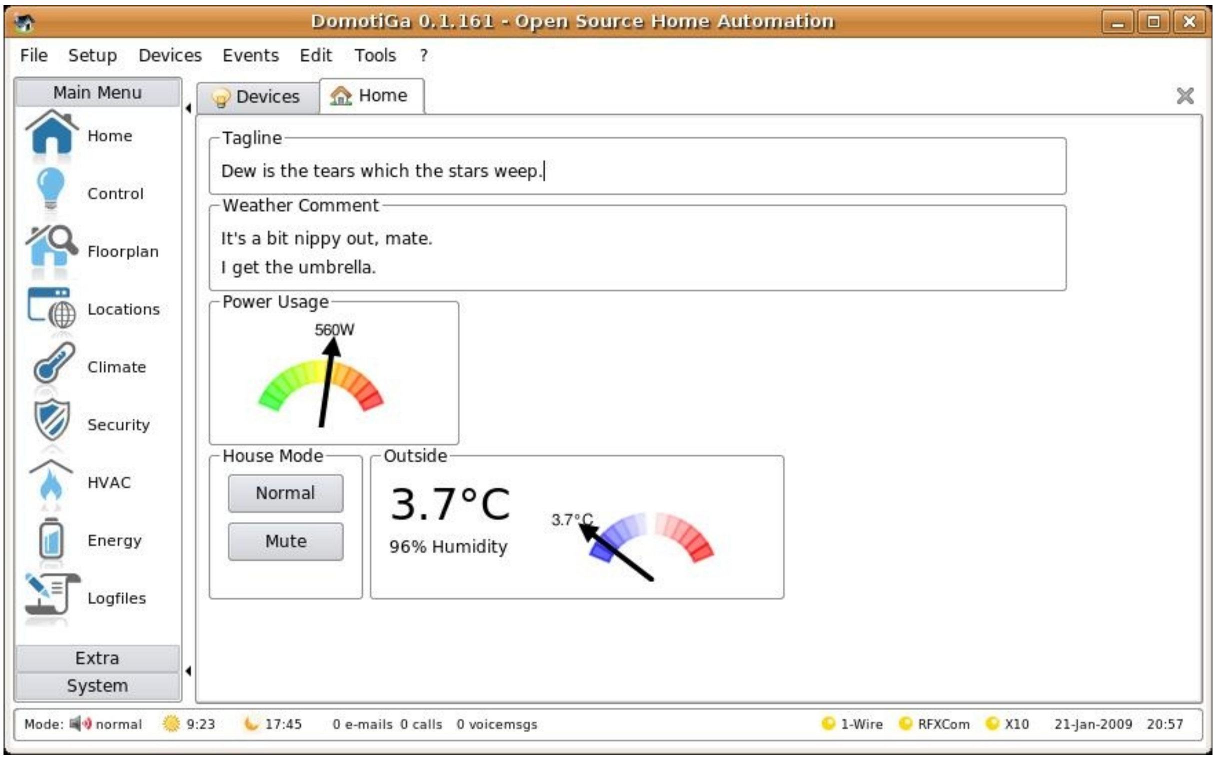
```
procedure TF_Rainbowloop(Sender: TObject);
var mtime, multiple: integer;
begin
   LED_Checker(false, true);
  {Color Spectrum from Red to White code (r,y,g,c,b,m,w...}
   mtime:= 500; //1000;
   multiple:= 2;
   statBar.Panels[1].Text:='Rainbow - Click LED4 to end loop!';
   try
     with cPort do begin   //using
       repeat
        //WriteStr('1'); Sleep(mtime);
         digitalWrite(red, AHIGH);     // red
         delay(mtime);
         digitalWrite(green, AHIGH);   // yellow
         delay(mtime);
         digitalWrite(red, ALOW);      // green
         delay(mtime);
         digitalWrite(blue, AHIGH);    // cyan
         delay(mtime);
         digitalWrite(green, ALOW);    // blue
         delay(mtime);
         digitalWrite(red, AHIGH);     // magenta
         delay(mtime);
         digitalWrite(green, AHIGH);  // white
         mtime:= mtime * multiple;
         delay(mtime);
         digitalWrite(blue, ALOW);    // reset
         digitalWrite(green, ALOW);
         digitalWrite(red, ALOW);
         mtime:= mtime div multiple;  //time/=multiple;
       until chk_led4.Checked=true;
       chk_led4.Checked:= false;
     end;
   except
      Showmessage(R_EXCEPTMESS);
   end;
end;
```

## 1.4 Appendix Arduino Base Code

```
//*******************************************Arduino Code**********************
 * Turns on and off 5 light emitting diodes (LED) connected to digital  pins 2 to 6. The LEDs will be
controlled using check boxes on maXbox that sends serial data to Arduino.  */
int val = 0;       // variable to store the data from the serial port
int ledPin1 = 2;   // LED connected to digital pin 2
int ledPin2 = 3;   // LED connected to digital pin 3
int ledPin3 = 4;   // LED connected to digital pin 4

void setup() {
  pinMode(ledPin1,OUTPUT);    // declare the LED's pin as output
  pinMode(ledPin2,OUTPUT);    // declare the LED's pin as output
  pinMode(ledPin3,OUTPUT);    // declare the LED's pin as output
  Serial.begin(9600);        // connect to the serial port
}
void loop () {
  val = Serial.read();      // read the serial port
  if (val !=-1){
    if (val=='1'){
      digitalWrite(ledPin1,HIGH);
    }
    else if (val=='A'){
      digitalWrite(ledPin1,LOW);      }
    if (val=='2'){
      digitalWrite(ledPin2,HIGH);     }
    else if (val=='B'){
      digitalWrite(ledPin2,LOW);      }
    if (val=='3'){
      digitalWrite(ledPin3,HIGH);     }
    else if (val=='C'){
      digitalWrite(ledPin3,LOW);      }
    //Serial.println();
  }}
```

## 1.5 Appendix Arduino App



Tagline

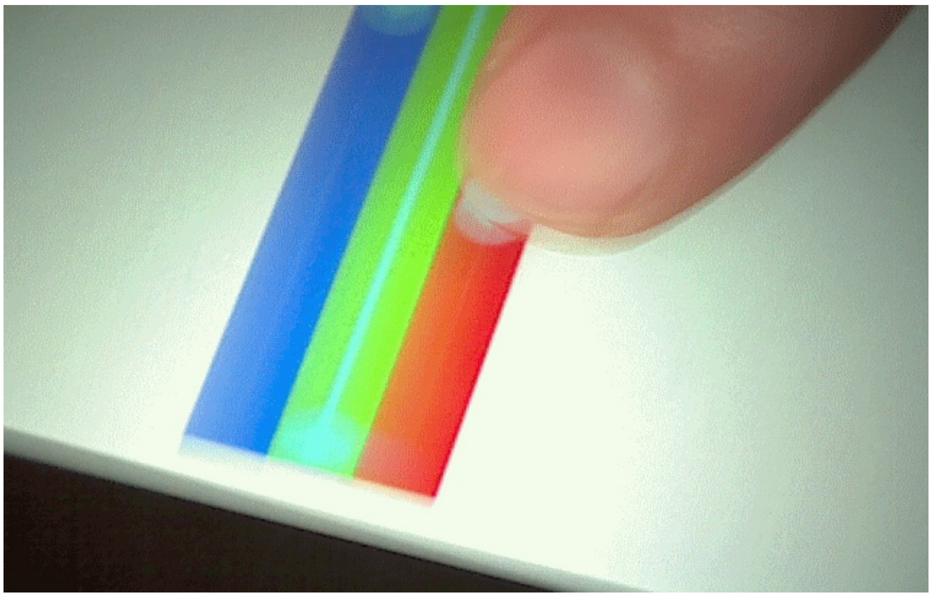
Dew is the tears which the stars weep.

Weather Comment

It's a bit nippy out, mate.
I get the umbrella.

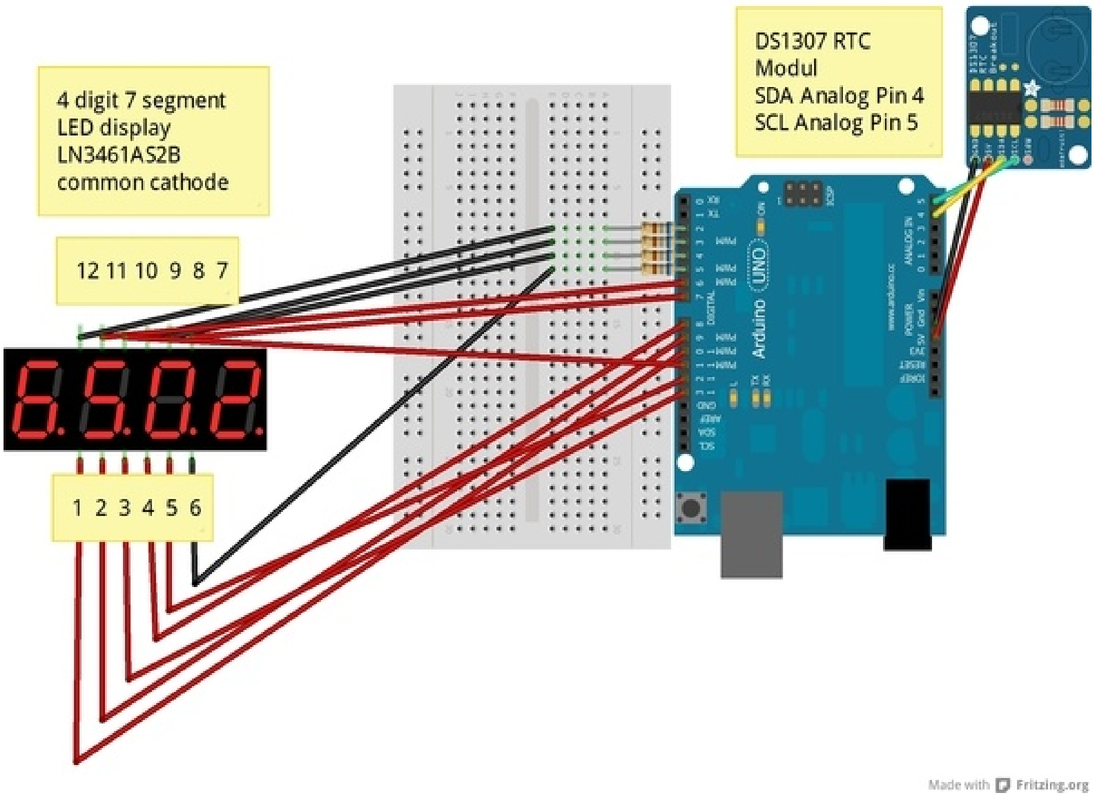
## 1.6 Appendix Controller

### 1.6.1 Android Schema



9: maXbox Web Cam live in Android on Arduino

## 1.7 RTC Layout


4 digit 7 segment
LED display
LN3461AS2B
common cathode
12 11 10 9 8 7
1 2 3 4 5 6
DS1307 RTC
Modul
SDA Analog Pin 4
SCL Analog Pin 5
Arduino UNO
Made with Fritzing.org