Nerds Only

08.2023

Board ESP8266

Source: Espressif

GIMP

GIMP originally stands for General Image Manipulation Program, and first version was developed 1995 by Spencer Kimball and Peter Mattis as a semester-long project at the University of California, Berkley. Two year later they met Richard Stallman of the GNU project while he visited UC Berkley and asked if they could change ‘General’ in the application's name to ‘GNU’ (the name of the operating system created by Stallman), and Stallman approved.

GIMP's mascot is called Wilber and was created in GIMP by Tuomas Kuosmanen, known as tigert, on 25 September 1997.

The GNU mascot is a anthropomorphic wildebeest head, but has no name.

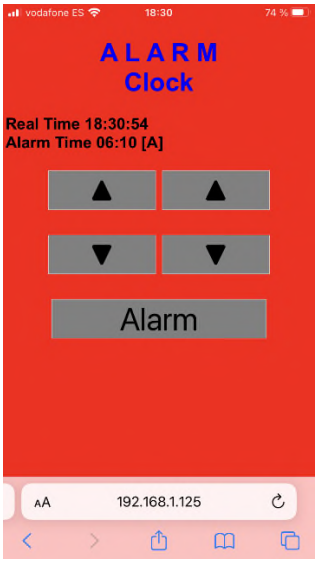
Source:
https://en.wikipedia.org/wiki/GIMP
https://en.wikipedia.org/wiki/GNU_Project

Project

Alarm Clock

We will program RoboMaster as an alarm clock:

- Get real time from internet
- Set alarm time with smartphone (HTML)
- Set alarm on / off by smartphone (HTML)



Use the left up/down buttons to set the hour of alarm time, and the right up/down buttons to set the minutes.
With the ALARM button you can switch the alarm on, status indicated by the letter A. If you switch of (tap button again), the letter A will disappear.

The projects is realized in five steps:

1.- WLAN – Basics

Show local IP on display

2.- WLAN – Basics & HTML

Show local IP on Webpage (Browser on Smartphone)

3.- Internet time

Show actual time on display and HTML

4.- Set Alarm Time

Show alarm time on Display and HTML, Modify hh and mm with buttons

5.- Realize Alarm Clock Funktion

Implement alarm clock logic, with optical and acoustical alarm

In this Update we will do the the first two steps

1.- WLAN – Basics

(SSID, Password, local IP)
Show local IP on display

Introduction

The SSID ist the name of WLAN, in my case MOVISTAR_EF37. To enter WLAN you need password, in my case 981675525.
With WiFi.begin(ssid, password) you send an request from your ESP8266 to your WLAN router, for access. If SSID and Pswd is correct, the router agrees and will assign an IP address to your device. To show the IP address on your display: lcd.print(WiFi.localIP());
This takes about 4 seconds.
If ssid and/or pswd is wrong, you will get (IP unset) or just stay in the loop while (WiFi.status() != WL_CONNECTED).

WiFi.begin(ssid, password)	Access to router, router provides IP address
while (WiFi.status() != WL_CONNECTED) { }	Wait, until connected to router
lcd.print(WiFi.localIP())	Print the assigned IP address on LCD display

Library

ESP8266WiFi library — ESP8266 Arduino Core documentation (arduino-esp8266.readthedocs.io)
C:\Users\Timo\Documents\ArduinoData\packages\esp8266\hardware\esp8266\3.0.1\libraries

Code

```
#include <LiquidCrystal_I2C.h>
#include <ESP8266WiFi.h>

const char* ssid = "MOVISTAR_EF37"; //Your Network SSID
const char* password = "981675525"; //Your Network Password

// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 16, 2);

void setup() {

  // initialize the LCD
  lcd.begin();

  // Turn on the blacklight and print a message: Wait for IP address
  lcd.backlight();
  lcd.setCursor(0, 0);  // Spalte, Zeile
  lcd.print("Connecting");
```

```
  // Connect to WiFi network
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    lcd.print(".");
  }

  // print a IP address.
  lcd.setCursor(0, 0);  // Spalte, Zeile
  lcd.print("                ");
  lcd.setCursor(0, 0);  // Spalte, Zeile
  lcd.print(WiFi.localIP());

}

void loop() {
  // do nothing
}
```

Try it yourself

1.) Try Code with your own SSID and Password. Which IP address you get?

2.) Modify code: Substitute the while statement by a simple delay(). Try different delay times. What is the minimum delay you need to get a valid IP address on your display? What message do you get on your display, when the delay is to short?

2.- WLAN – Basics & HTML

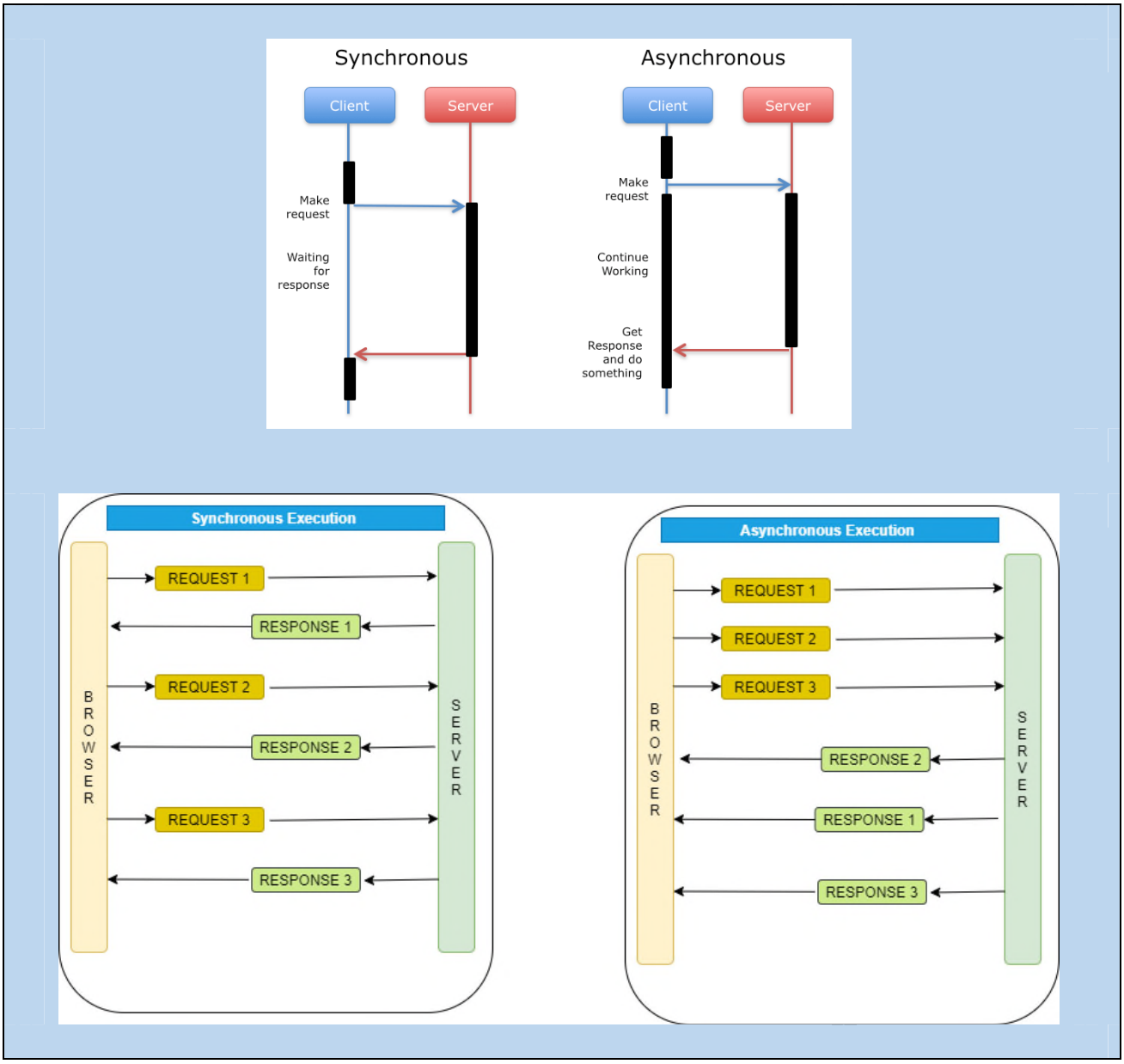
(AsyncWebServer, Client, Server)

Show local IP on Webpage (Browser on Smartphone)

Introduction

AsyncWebServer

First you need to set up an AsyncWebServer. An AsyncWeb Server is a very robust server concept, that can handle requests at any time. In this case you send a request via the browser (client) of our smartphone with the IP address (just type in the browser the assigned IP address (in my case 192.168.1.125, equal to 192.168.1.125/). This is understood as request for root ("/") or homepage wich than will be send as a request.



```
#include <ESPAsyncWebServer.h>
AsyncWebServer server(80);


server.on("/", [](AsyncWebServerRequest * request)
  {
    request->send_P(200, "text/html", webpage);
  });
```

Setup AsyncServer

On request for root (“/”) send webpage

HTML Page

A separate file named html.h contains the HTML Code with the content and design of the Web-Page, which will be displayed on your smartphone as a response to your request. Again once you type in your browser the IP address (in my case 192.168.1.125) and press ENTER you send the request.
The code has a basic structure:

```
char webpage[] PROGMEM = R"=====(
   HTML Code
)=====";
```

Store the HTML Code in the variable webpage

And the HTML Code build from 3 blocks:

```
<!DOCTYPE html>
<html>
  <head>
  </head>

  <style>
  </style>
  <body>


  </body>
</html>
```

HTML Begin

Header appears as Window title in Browser
Style, like colors

Headers and Paragraphs. Here is basically the content of the webpage.
HTML End

Library

ESP8266WiFi library — ESP8266 Arduino Core documentation (arduino-esp8266.readthedocs.io)
C:\Users\Timo\Documents\ArduinoData\packages\esp8266\hardware\esp8266\3.0.1\libraries

Code

```
#include <LiquidCrystal_I2C.h>

// Set the LCD address to 0x27 for a 16 chars and 2 line display
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

```
// Librarys for WiFi
#include <ESP8266WiFi.h>;
#include <WiFiClient.h>;

#include <ESP8266mDNS.h>

// Include my own html page
#include "html.h"

const char* ssid = "MOVISTAR_EF37"; //Your Network SSID
const char* password = "981675525"; //Your Network Password

WiFiClient client;

#include <ESPAsyncWebServer.h>
AsyncWebServer server(80); // server port 80

void notFound(AsyncWebServerRequest *request)
{
  request->send(404, "text/plain", "Page Not found");
}


void setup()
{
  // initialize the LCD
       lcd.begin();

  // Turn on the blacklight and print a message.
  lcd.backlight();
  lcd.setCursor(0, 0);  // Spalte, Zeile
  lcd.print("Connecting");


   // Connect to WiFi network
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    lcd.print(".");
  }


  // Print local IP on Display
  lcd.setCursor(0, 0);  // Spalte, Zeile
  lcd.print("                ");
  lcd.setCursor(0, 0);
  lcd.print(WiFi.localIP());


  // (1) Page for /
  server.on("/", [](AsyncWebServerRequest * request)
  {
    request->send_P(200, "text/html", webpage);
  });

  server.onNotFound(notFound);

  server.begin();  // it will start webserver

}

void loop()
{

// do nothing
```

```
}
```

HTML Code (html.h)

```
//HTML code--------------------
char webpage[] PROGMEM = R"=====(
<!DOCTYPE html>
<html>
<head>
          <title>A L A R M Clock</title>
</head>

<!--HTML body formatting-->
<style>
          body {font-family: "Arial"; background-color: lightblue}
          h1   {color: blue; text-align:center; font-size: 80px}
          h2   {color: black; font-size: 50px}
          h3   {color: red; text-align:center; font-size: 20px}
</style>
<!--end of formatting-->

<body>
      <h1>
              A L A R M<br>
              Clock<br>
       </h1>
      <h2>

           <center>
          <br>
          <img src="https://nerdsacademy.de/wp-
content/uploads/2023/08/ArlamClockWhite.png" width = 40% ><br>
         </center>

              <a>System data:</a><br>


      <p id="demo"></p>
      <p id="demo2"></p>

      <script>
      let host = location.host;
      let host2 = 'IP address: '+host;
      document.getElementById("demo2").innerHTML = host2;
      </script>

      </h2>
</body>
</html>
)=====";
```

Exercise

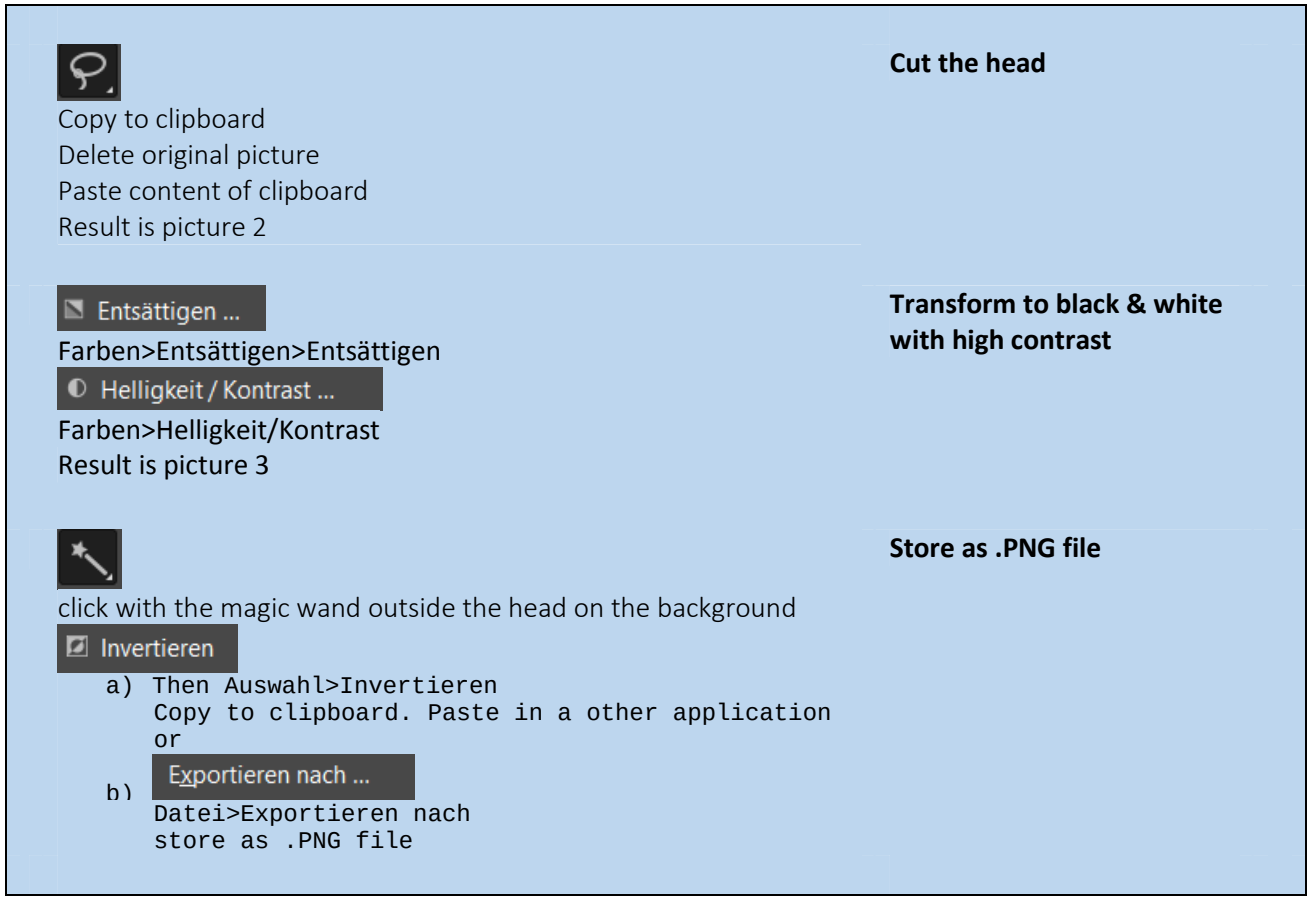
1.) There are two symbols for alarm clock: AlarmClockBlack and AlarmClockWhite. Modify the code that instead of the black the white one is used.
2.) Change background color from lightblue to red.

Extra

Modify foto to black&white with GIMP

Introduction

Gimp is a free software used for image editing. It's a raster graphis editor, means pictures are pixel based (dot by dot). You can use it to modify fotos to simpler black/grey/white fotos.



Links

https://www.chip.de/downloads/GIMP_12992070.html

Description

To use the picture in my projects I store it on my own homepage:
https://nerdsacademy.de/wp-content/uploads/2023/08/Ronaldo.png

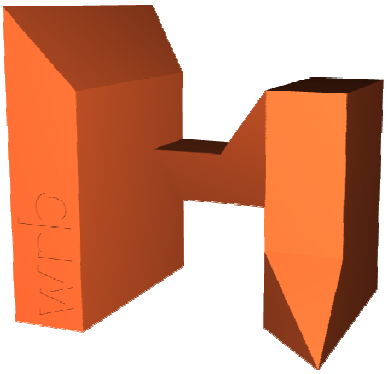
Try yourself!

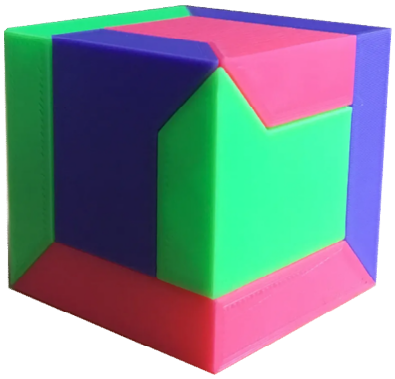
1.) Take a picture of your own (best with sunglasses). Modify it to black&white.
2.) Create a new code, which creates a a HTML page from your RonboMaster: Titel, your personal maxim, your picture.

3D Printing

Description

3 identicle parts that slide together in a satisfying way to create a cube. Once assembled it needs to be held and moved in a specific direction to seperate. Cube size at default scale is 45mm.





Source

Thingiverse: https://www.thingiverse.com/thing:2975065
Printables: https://www.printables.com/de/model/338890-puzzle-cube-easy-print-no-support

Ich habe es über Printables heruntergeladen.

Exercise

1.) Print the standard version and find solution to mount it.
2.) Print a smaller version (e.g. 30 instead of 45 mm), and try again

Contact:
nerdsonly.de/kontakt/

Disclaimer

Safety Awareness: The content on this website is intended for educational purposes. Users should exercise caution and follow safety guidelines when engaging in electronic and 3D print projects to avoid potential risks.

Accuracy Disclaimer: While we strive for accuracy, the information and projects provided may change and might not always be up-to-date. Users should verify information before relying on it.

Liability Limitation: We are not responsible for any injuries, damages, or losses resulting from the use of information, projects, or materials provided on this website. This includes damages related to personal injury, property damage, data loss, or profits.

External Links: Links to external websites are not endorsements, and we are not accountable for the content or accuracy of those sites.

Intellectual Property: The intellectual property rights of projects and content belong to respective owners. Users can use the information for personal, non-commercial purposes, but commercial use requires explicit permission.