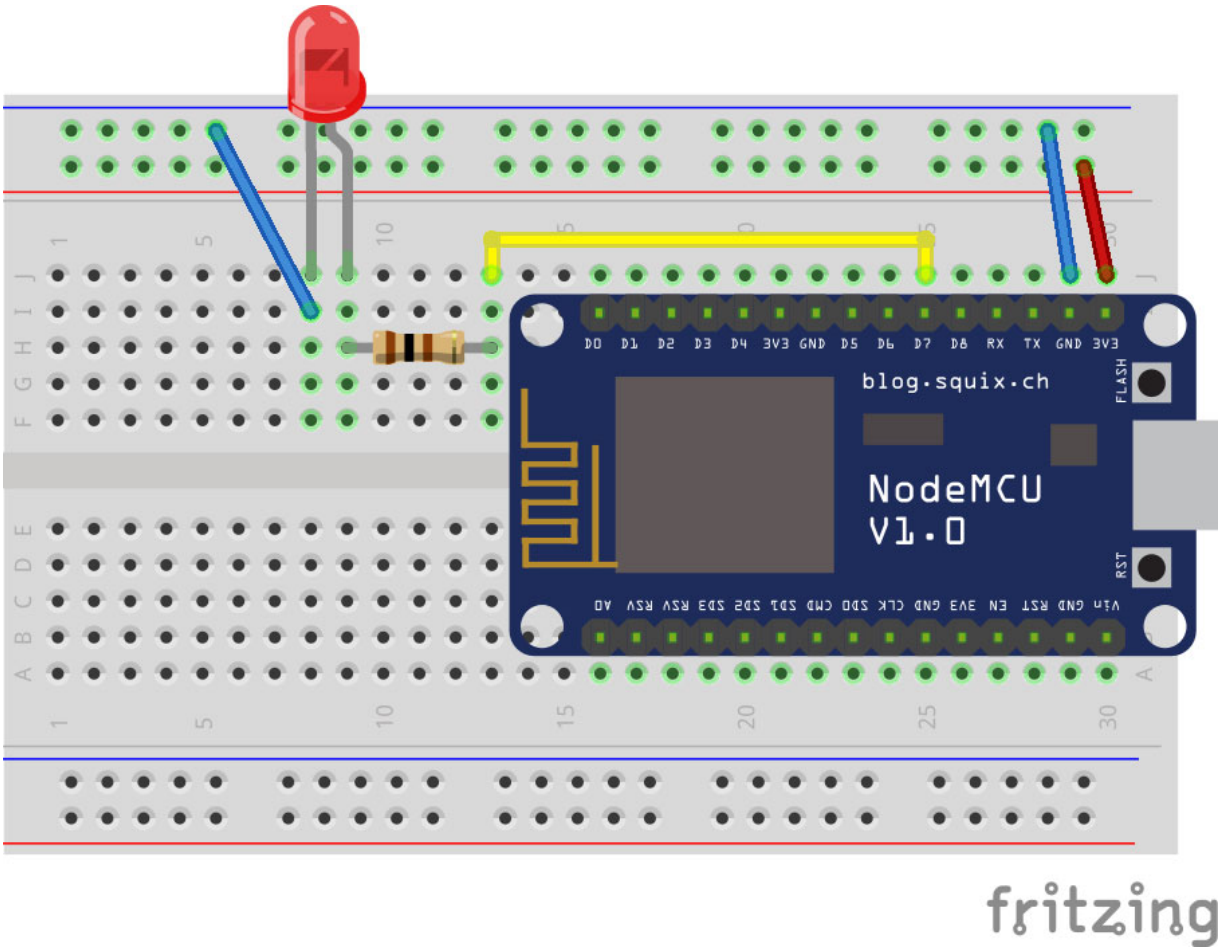
## Purpose

The purpose of this exercise is to build a small IOT thing with a NodeMCU.

## Initial setup




A LED is connected to pin D7, which is GPIO13 on the NodeMCU.

Note, that the LED will need a resistor.

For wifi connection, you can use a standard access point (at home) or create a temporary wifi hotspot with your phone (at DTU). The wireless networks at DTU (e.g. eduroam) use enterprise-level security mechanisms; so you won't be able to connect to the eduroam with your nodemcu..

## Control an LED from Web Browser

This code connects the NodeMCU to a wifi-network, and creates a small server on the NodeMCU. The NodeMCU then tells you the address through serial communication (Note the baudrate set in setup). You can control the LED, if you enter that address on a device on the same wifi-network.

```
# include <WiFi.h>

const char *ssid = "Sofus iPhone";
const char *password = "glem det";

int ledPin = 2; // which is GPIO2. With nodemcu, you should directly use the pin names as they
are stated on the board...
WiFiServer server(80); //create a server

void setup()
{
  Serial.begin(115200);
  delay(10);
```

```
  pinMode(ledPin, OUTPUT);
  digitalWrite(ledPin, LOW);

  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password); // attempt to connect

  while (WiFi.status() != WL_CONNECTED) { //wait to connect
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  // Start the server
  server.begin();
  Serial.println("Server started");

  // Print the IP address
  Serial.print("Use this URL to connect: ");
  Serial.print("http://");
  Serial.print(WiFi.localIP());
  Serial.println("/");
}

void loop()
{
  // Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {
    return;
  }

  // Wait until the client sends some data
  Serial.println("new client");
  while(!client.available()){
    delay(1);
  }

  // Read the first line of the request
  String request = client.readStringUntil('\r');
  Serial.println(request);
  //client.flush();

  // Match the request

  int value = LOW;
  if (request.indexOf("/LED=ON") != -1)  {
    digitalWrite(ledPin, HIGH);
    value = HIGH;
  }
  if (request.indexOf("/LED=OFF") != -1)  {
    digitalWrite(ledPin, LOW);
    value = LOW;
  }

// Set ledPin according to the request

  // Return the response
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: text/html");
  client.println(""); //  do not forget this one
  client.println("<!DOCTYPE HTML>");
```

```
  client.println("<html>");

  client.print("Led pin is now: ");

  if(value == HIGH) {
    client.print("On");
  } else {
    client.print("Off");
  }
  client.println("<br><br>");
  client.println("<a href=\"/LED=ON\"\"><button>Turn On </button></a>");
  client.println("<a href=\"/LED=OFF\"\"><button>Turn Off </button></a><br />");
  client.println("</html>");

  delay(1);
  Serial.println("Client disonnected");
  Serial.println("");
}
```

# Tasks

## Task 1 - Example

Recreate the example "Control an LED from Web Browser", and test if it works with your smartphone or laptop.

## Task 2 - Dual Control

Add "dual control" to your test setup, such that you are able to control the LED from both the Web Browser and the Serial Monitor, by writing either "on" or "off" on the Serial Monitor input.