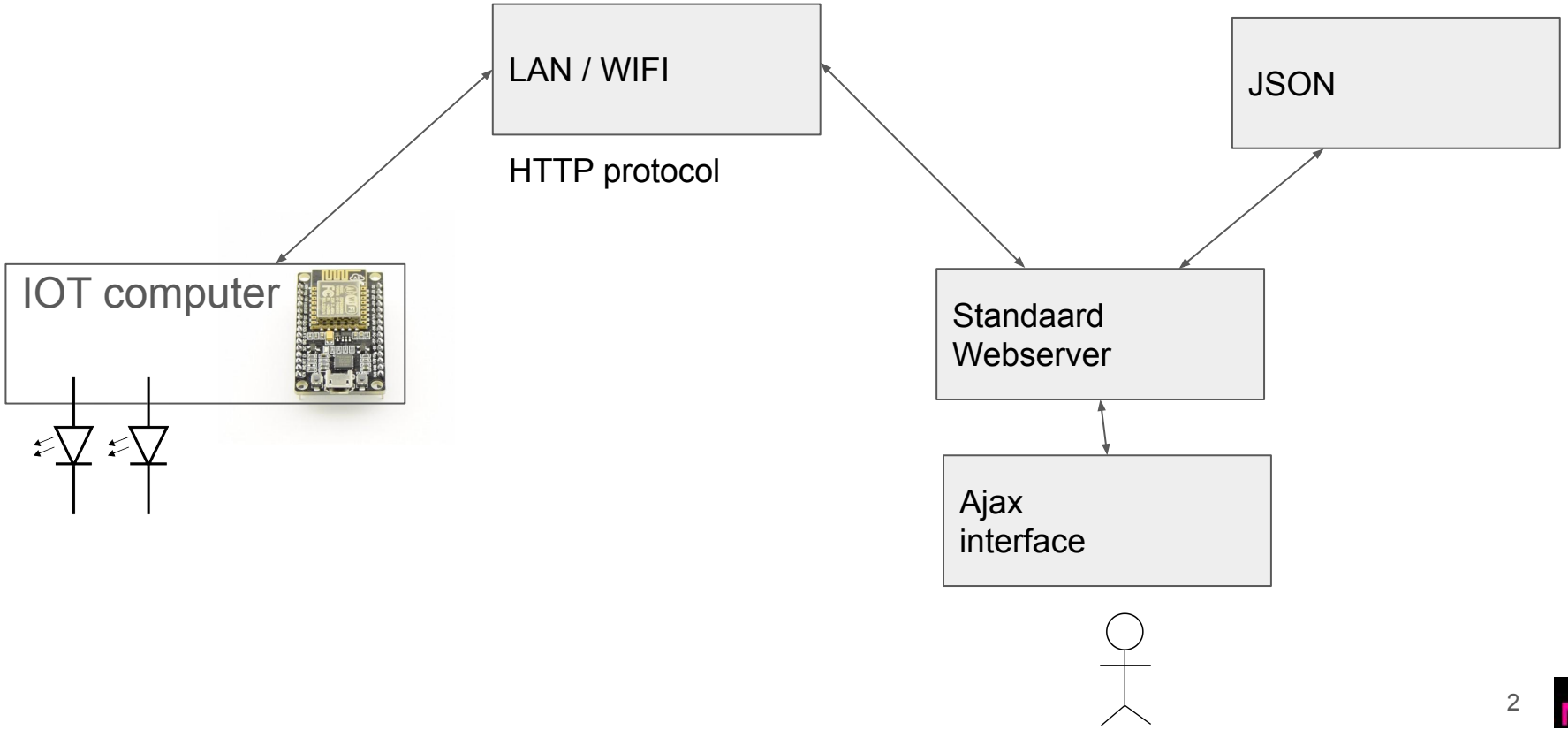
Internet of things

GD2AB

Node MCU ESP8266
HTTP Protocol
MQTT Protocol

IOT Proof of concept project

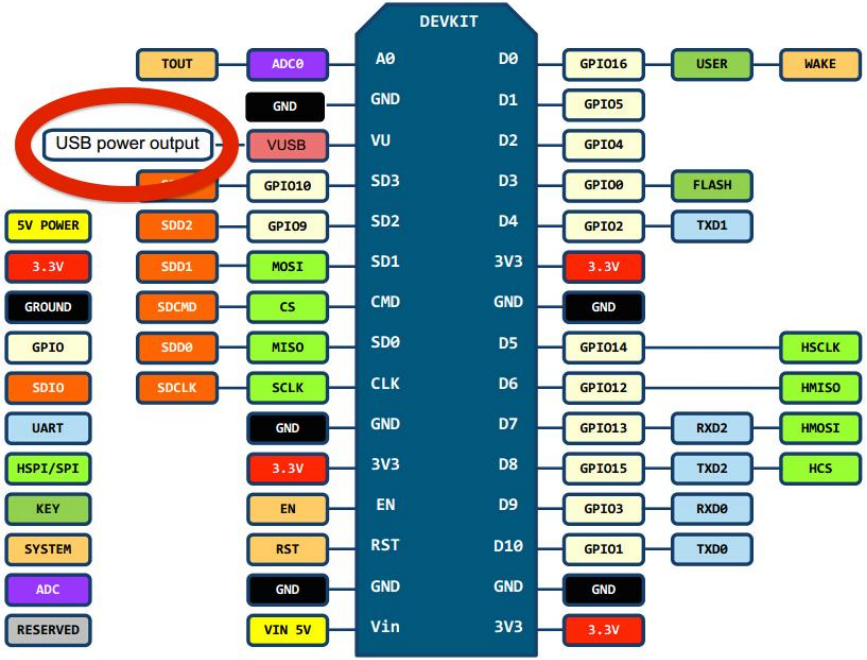


Roadmap

Node MCU ESP8266.

1) Ontwikkelomgeving
2) Hardware
3) Hello World LED flasher
4) WIFI connection
   a) Inloggen in netwerk
   b) Dump netwerk gegevens naar serial port
5) JSON bestand van webserver ophalen, vertalen naar

Hardware


DEVKIT
TOUT
ADC0
A0
D0
GPIO16
USER
WAKE
GND
GND
D1
GPIO5
USB power output
VUSB
VU
D2
GPIO4
GPIO10
SD3
D3
GPIO0
FLASH
5V POWER
SDD2
GPIO9
SD2
D4
GPIO2
TXD1
3.3V
SDD1
MOSI
SD1
3V3
3.3V
GROUND
SDCMD
CS
CMD
GND
GND
GPIO
SDD0
MISO
SD0
D5
GPIO14
HSCLK
SDIO
SDCLK
SCLK
CLK
D6
GPIO12
HMISO
UART
GND
GND
D7
GPIO13
RXD2
HMOSI
HSPI/SPI
3.3V
3V3
D8
GPIO15
TXD2
HCS
KEY
EN
EN
D9
GPIO3
RXD0
SYSTEM
RST
RST
D10
GPIO1
TXD0
ADC
GND
GND
GND
GND
RESERVED
VIN 5V
Vin
3V3
3.3V

Hardware

Voedingsspanning 3,3 Volt

Input / Output , low power 3,3 Volt, kwetsbaar

Vergelijk Arduino Input / Output 5 Volt, redelijk robuust

Hardware Pin Mapping

Arduino =>NodeMCU has weird pin mapping.
Pin numbers written on the board itself do not correspond to ESP8266 GPIO pin numbers. We have constants defined to make using this board easier:

```
static const uint8_t D0   = 16;
static const uint8_t D1   = 5;
static const uint8_t D2   = 4;
static const uint8_t D3   = 0;
static const uint8_t D4   = 2;
static const uint8_t D5   = 14;
static const uint8_t D6   = 12;
static const uint8_t D7   = 13;
static const uint8_t D8   = 15;
static const uint8_t D9   = 3;
static const uint8_t D10  = 1;
```

#include <ArduinoJson.h>
voor de pinmapping

pin mapping.
https://learn.sparkfun.com/tutorials/esp8266-thing-hookup-guide/using-the-arduino-addon

Ontwikkelomgeving

1) Gegevens http://www.tinytronics.nl/shop/index.php?route=product/product/&product_id=365
2) Driver USB https://www.silabs.com/products/development-tools/software/usb-to-uart-bridge-vcp-drivers
3) Instructies https://github.com/esp8266/Arduino
4) Arduino 1.8.1 IDE https://www.arduino.cc/en/main/software
5) IDE instellingen: Board NodeMCU. CPU frequency 80 MHz. Upload speed: 115200. Flash size 4M (3M SPIFFS)

Hello World, Led Flasher

Opdracht:

Installeer ontwikkelomgeving.

Maak een Hello World Led Flasher. Gebruik een GPIO poort. Led met serieweerstand.

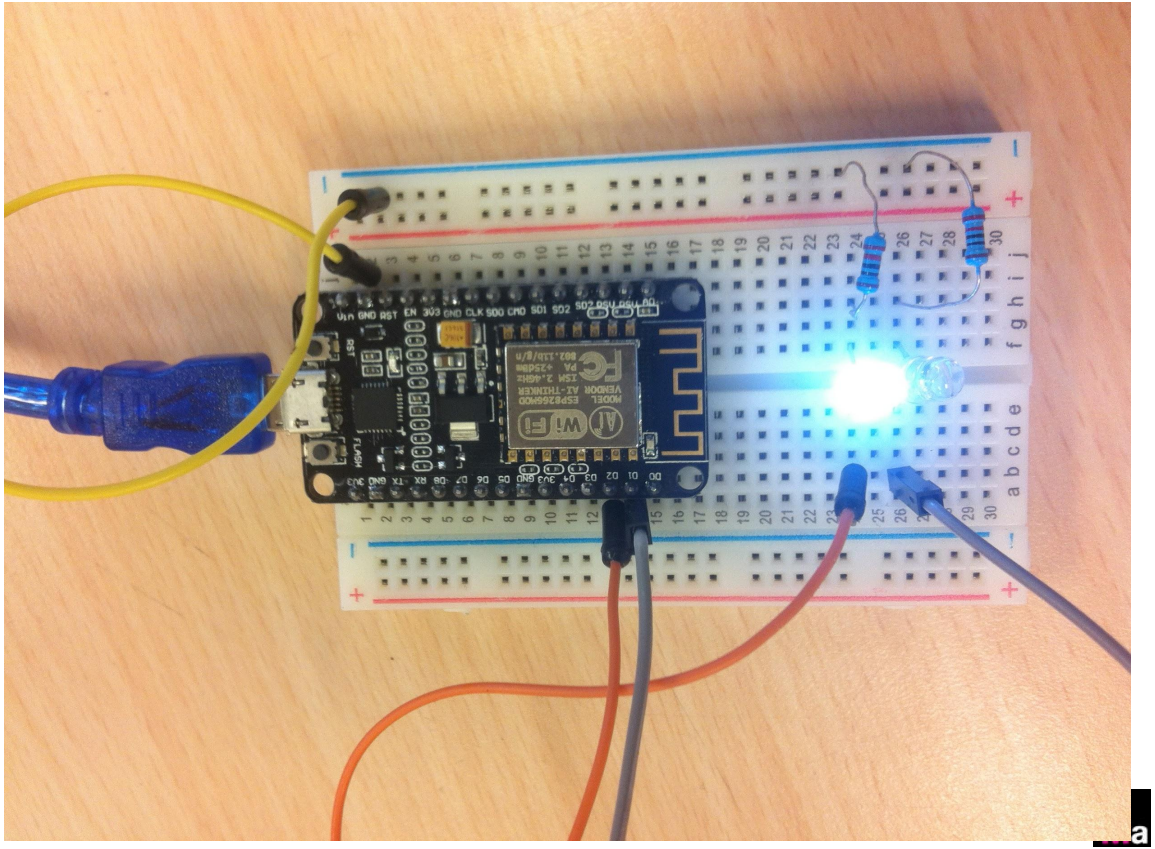
Laten zien in de les

Wifi lan connectie met signalering

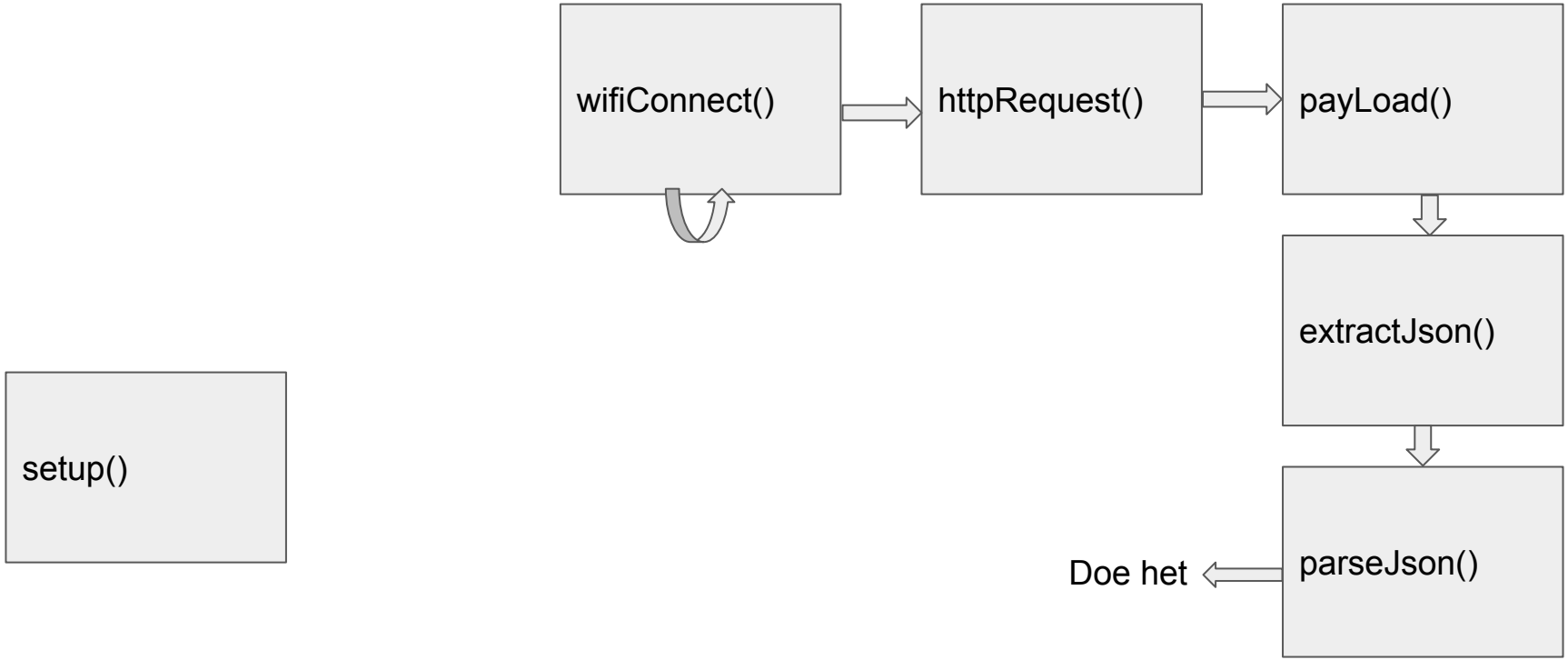
1. Opdracht Script Wifi connect
   a. Node-MCU maakt connectie met wifi LAN:
      SSID: Medialab PWD: Mediacollege
   b. Verbindingsproces moet te volgen zijn op de terminal: connecting, conncected, SSID, IP adres
   c. Signalering extern, 2 LED’s.
      Led 1: Aan als geen verbinding, knippert als bezig met verbinden
      Led 2: Aan als verbinding correct
2. Code via GIT. Maak hier een aparte sectie titel NODE MCU

Wifi LAN connectie met signalering

Signalering extern, 2 LED’s.
Led 1: Aan als geen verbinding, knippert als bezig met verbinden
Led 2: Aan als verbinding correct

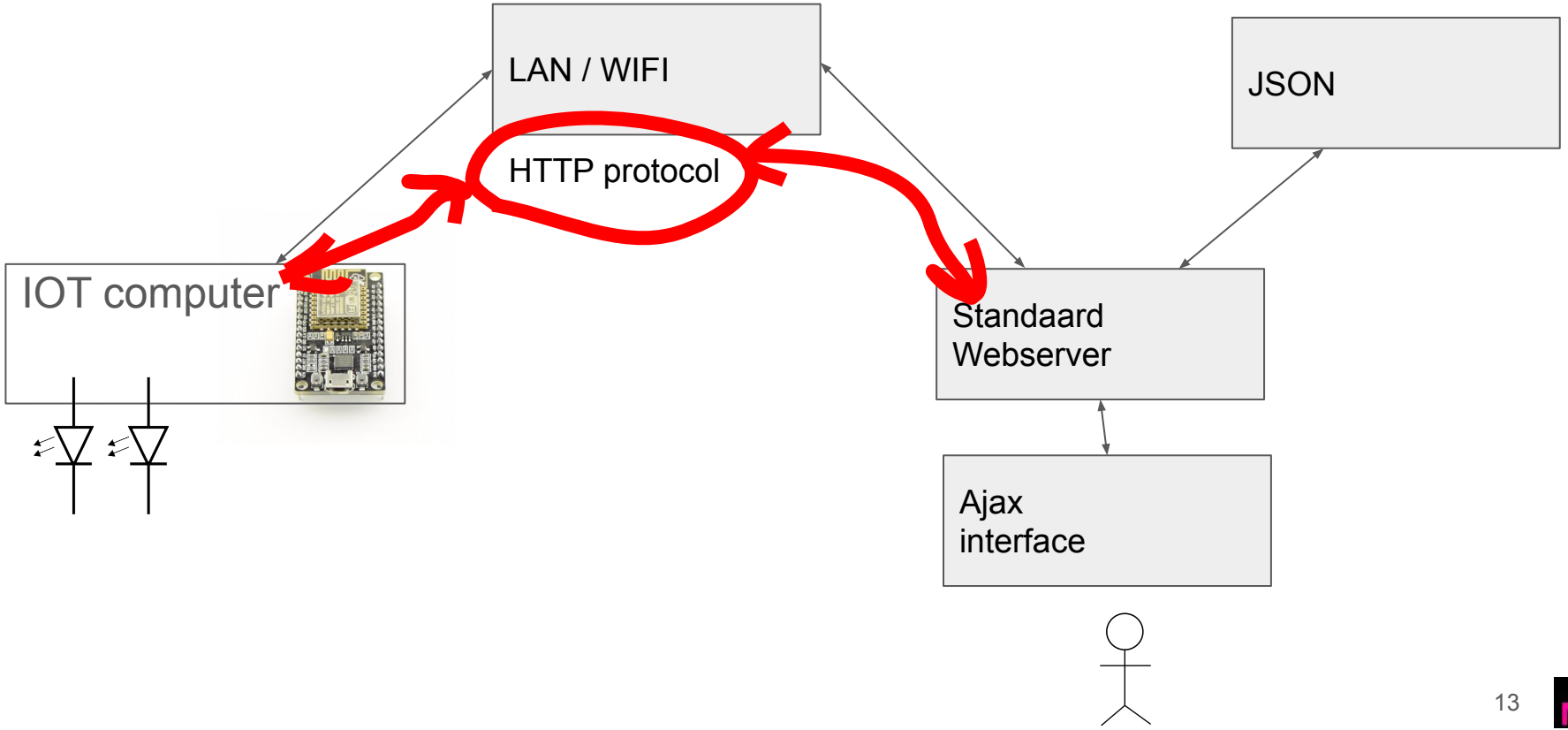


Opbouw software

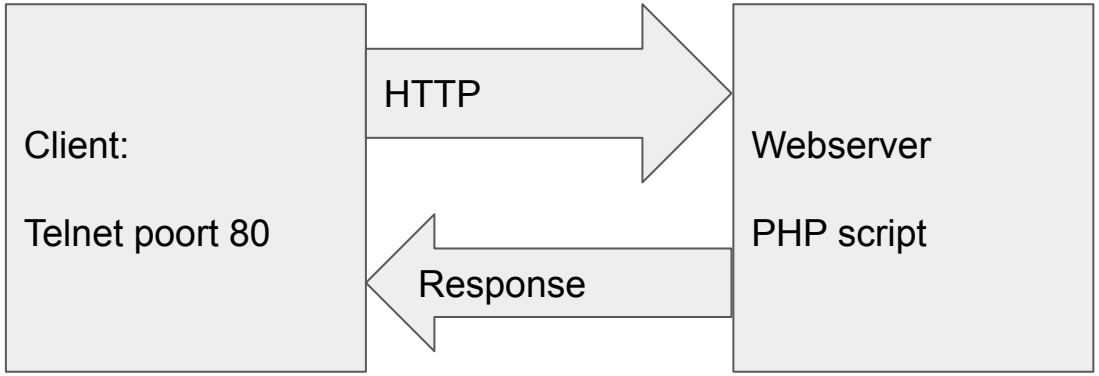


HTTP protocol
HTTP request

HTTP protocol



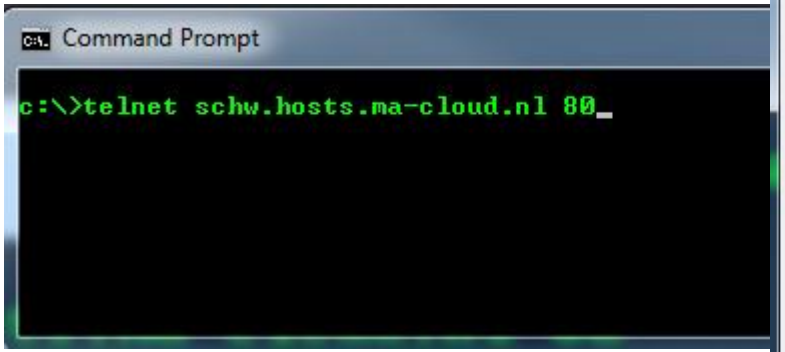
HTTP POST GET request en PHP

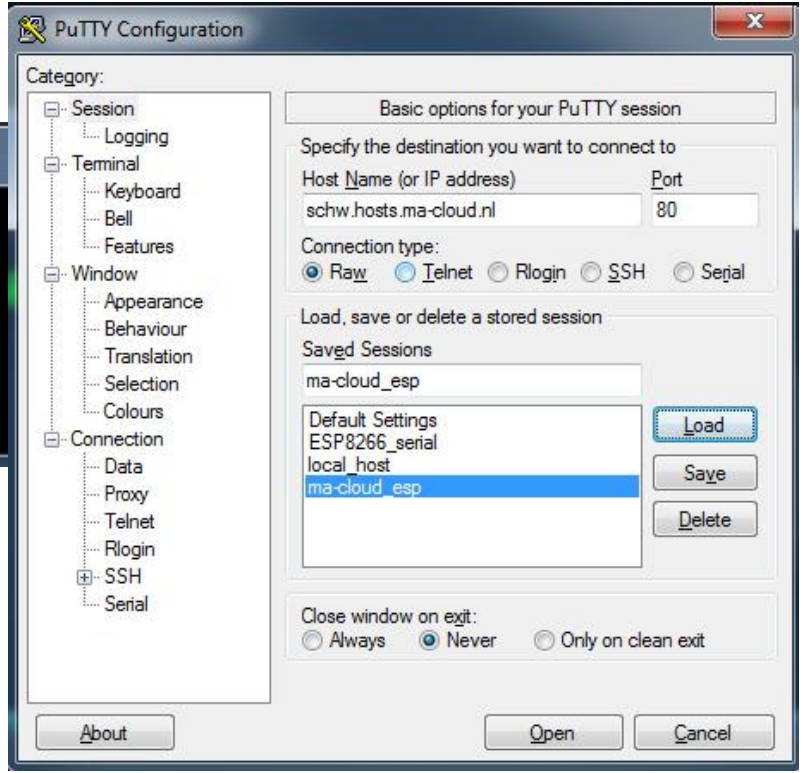


Windows: Telnet activeren of PUTTY installeren

MAC/Linux: Telnet is standaard geinstalllerd

Windows: Telnet activeren of PUTTY installeren
MAC/Linux: Telnet is standaard aanwezig





HTTP request GET

telnet schw.hosts.ma-cloud.nl 80

GET /test.json HTTP/1.0
HOST: schw.hosts.ma-cloud.nl

HTTP response

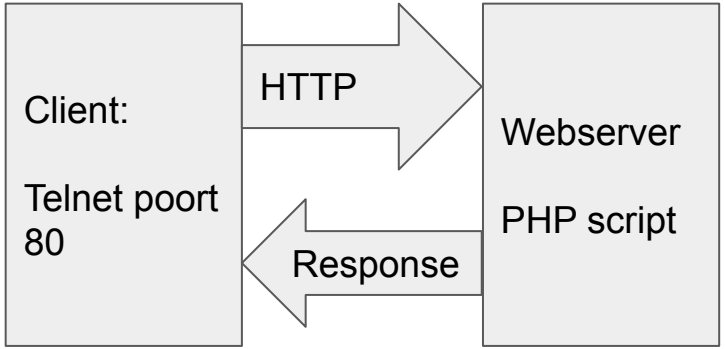
```
HTTP/1.1 200 OK
Date: Sun, 02 Apr 2017 22:52:41 GMT
Server: Apache
Last-Modified: Sun, 02 Apr 2017 22:50:28 GMT
ETag: "402afc-36-54c36de7f5a74"
Accept-Ranges: bytes
Content-Length: 54
X-Powered-By: PleskLin
Connection: close
Content-Type: application/json

{"device":"NodeMCU","status":"HTTP test","Ma":"R&D"}
```

HTTP GET, JSON bestand ophalen schw.hosts.ma-cloud.nl/test.json

PuTTY (inactive)

```
GET /test.json HTTP/1.0
HOST: schw.hosts.ma-cloud.nl
Connection: Keep-Alive

HTTP/1.1 200 OK
Date: Sun, 02 Apr 2017 23:05:28 GMT
Server: Apache
Last-Modified: Sun, 02 Apr 2017 22:50:28 GMT
ETag: "402afc-36-54c36de7f5a74"
Accept-Ranges: bytes
Content-Length: 54
X-Powered-By: PleskLin
Connection: close
Content-Type: application/json

{"device":"NodeMCU","status":"HTTP test","Ma":"R&D"}
```

Script waarmee request wordt verwerkt: schw.hosts.ma-cloud.nl/testGetPost.php




```
<?php
if (ISSET($_GET['testGET'])){echo $_GET['testGET']."<br>";}
else {echo "GET niets ontvangen<br>";}
if (ISSET($_POST['testPOST'])){ echo
$_POST['testPOST']."<br>";}
else {echo "POST niets ontvangen<br>";}
?>
```

HTTP request GET

Telnet schw.hosts.ma-cloud.nl 80

GET /testGetPost.php?testGET=1_april HTTP/1.0

HOST: schw.hosts.ma-cloud.nl

HTTP GET response van server en PHP script

HTTP/1.1 200 OK
Date: Mon, 03 Apr 2017 00:46:17 GMT
Server: Apache
X-Powered-By: PleskLin
Connection: close
Content-Type: text/html

1_april
POST niets ontvangen

HTTP GET response van server en PHP script

PuTTY (inactive)

```
GET /testGetPost.php?testGET=1_april HTTP/1.0
HOST: schw.hosts.ma-cloud.nl

HTTP/1.1 200 OK
Date: Mon, 03 Apr 2017 00:51:14 GMT
Server: Apache
X-Powered-By: PleskLin
Connection: close
Content-Type: text/html

1_april<br>POST niets ontvangen<br>
```

HTTP request POST

```
telnet schw.hosts.ma-cloud.nl 80
POST /testGetPost.php HTTP/1.1
Host: schw.hosts.ma-cloud.nl
Connection: Close
Content-Type: application/x-www-form-urlencoded
Content-Length: 17

testPOST=surprise
```

HTTP POST response van server en PHP script

HTTP/1.1 200 OK
Date: Mon, 03 Apr 2017 01:07:17 GMT
Server: Apache
X-Powered-By: PleskLin
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html

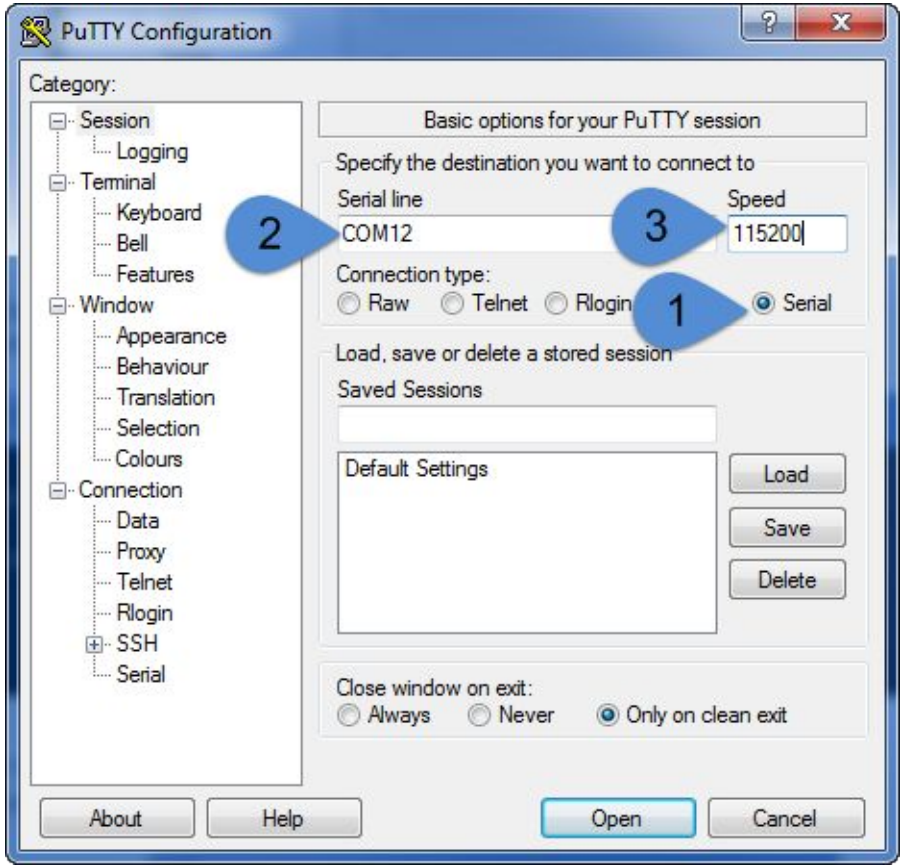
GET niets ontvangen
surprise

HTTP POST response van server en PHP script

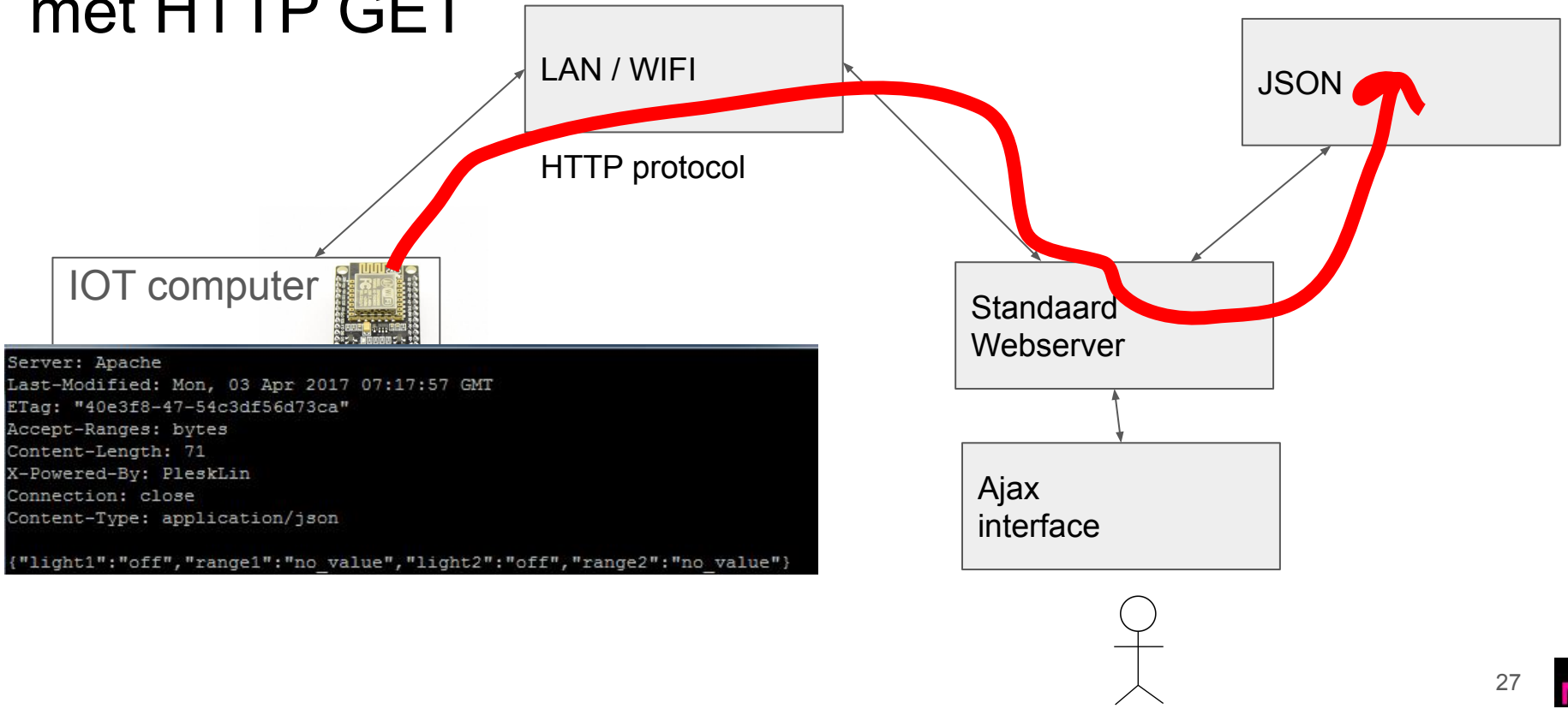
PuTTY (inactive)

```
POST /testGetPost.php HTTP/1.1
Host: schw.hosts.ma-cloud.nl
Connection: Close
Content-Type: application/x-www-form-urlencoded
Content-Length: 17

testPOST=surprise
HTTP/1.1 200 OK
Date: Mon, 03 Apr 2017 01:11:48 GMT
Server: Apache
X-Powered-By: PleskLin
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html

23
GET niets ontvangen<br>surprise<br>
0
```

PUTTY als terminal voor NodeMCU



DATA van Webserver halen met HTTP GET



Main loop

```
void loop() {
  if (WiFi.status() != WL_CONNECTED) wifiConnect();
  httpRequest();//get data from webserver
  if (debug) Serial.println(httpResponse);
  payload();//extract wanted data from HTTP response
  if (debug)Serial.println(httpResponse); //debug
  extractJson();
}
```

HTTP GET request

Zie de overeenkomst met AJAX Javascript

WiFiClient_IOT_versie2_les2017 | Arduino 1.8.1

File Edit Sketch Tools Help

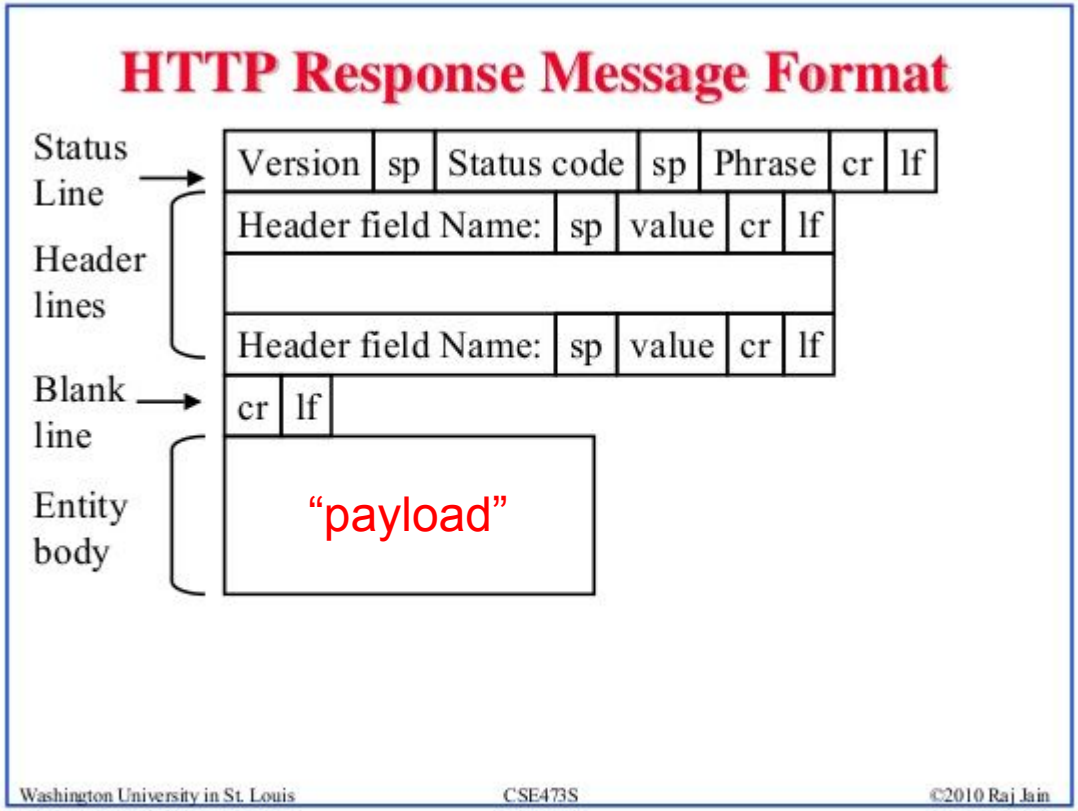
WiFiClient_IOT_versie2_les2017

```
24
25 void httpRequest() {
26   delay(2000);
27   WiFiClient client; //instance
28   if (client.connect(server, 80)) { //connect to webserver on port 80
29     client.println("GET " + path +  " HTTP/1.1");//make a HTTP GET request
30     client.println("Host: " + String(server));
31     client.println("Connection: keep-alive");
32     client.println();
33   }
34   else {
35     Serial.println( "Webserver does not respond");
36     return;
37   }
38   while (client.connected()){
39     if ( client.available() ){
40       char str = client.read();
41       Serial.print(str);
42     }}
43     Serial.println("");
44   }
45
46   void wifiConnect() { //connect to local network
67
```

HTTP Response, header + “payload”

```
HTTP/1.1 200 OK
Date: Thu, 06 Apr 2017 06:46
Server: Apache
Last-Modified: Mon, 03 Apr 2
ETag: "40e3f8-47-54c3df56d73
Accept-Ranges: bytes
Content-Length: 71
X-Powered-By: PleskLin
Connection: close
Content-T                tion/js

{"light1":"off","range1":"no
```

“payload”



HTTP Response, header + “payload”

```
HTTP/1.1 200 OK
Date: Thu, 06 Apr 2017 06:46:02 GMT
Server: Apache
Last-Modified: Mon, 03 Apr 2017 07:17:57 GMT
ETag: "40e3f8-47-54c3df56d73ca"
Accept-Ranges: bytes
Content-Length: 71
X-Powered-By: PleskLin
Connection: close
Content-Type: application/json

{"light1":"off","range1":"no_value","light2":"off","range2":"no_value"}
```

“payload”

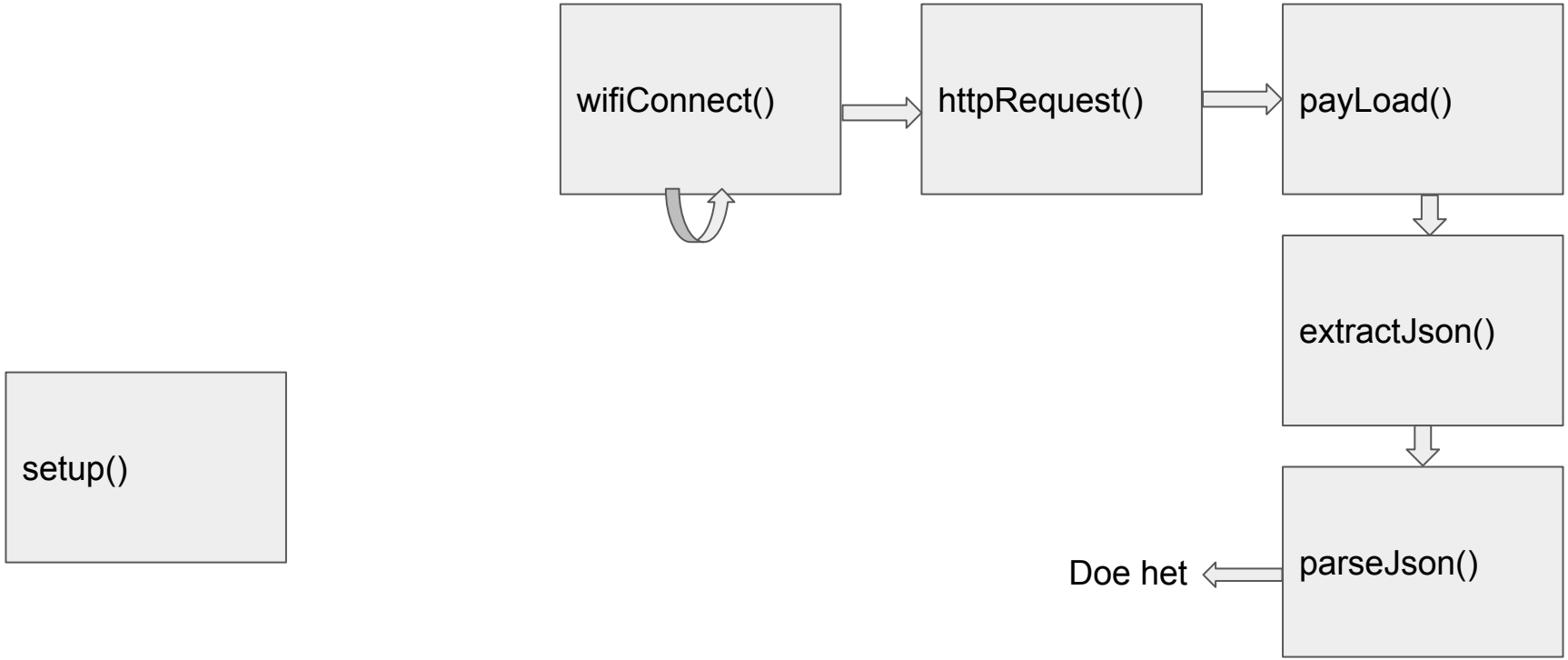
Opdracht 1:

Plaats het json bestand op jouw website bij Ma-cloud. Zorg dat de NodeMCU het JSON bestand van jouw site leest. Probeer het JSON bestand te wijzigen en controleer of de NODEMCU de nieuwe waarden correct uitleest. Laat zien in de les.

Json bestand: light.json

{"device1":"on","range1":"960","device2":"on","range2":"240","device3":"on","range3":"370"}

Opbouw software



Init + setup + overzicht functions

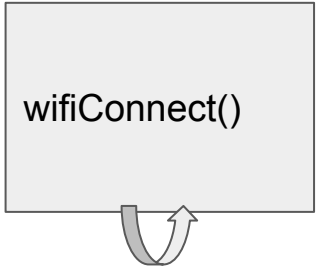
setup()

```
2   #include <ESP8266WiFi.h>
3   #include <ArduinoJson.h>
4   #include <Servo.h> // servo library
5   const char* ssid     = "Medialab"; // wifi lan
6   const char* password = "Mediacollege"; // wifi lan
7   const char*  server  = "schw.hosts.ma-cloud.nl"; // deployment server
8   //String path        = "/ESP8286/light.json"; // path to file
9   String path          = "/ma2d2/light.json"; // path to file
10  String path2         = "/ma2d2/connected.php"; // path to file
11  static const uint8_t wifiConnecting = D1;//LED indicator wifi status flashing while connectin
12  static const uint8_t wifiOk = D2;//LED indicator wifi status ON if connected
13  //static const uint8_t servo1 = D3; //servo 1 op D3
14  boolean debug = false;// print debug messages to terminal
15  String httpResponse;// response from webserver
16  Servo servo1;// define servo
17  Servo servo2;// define servo
18  void setup() {
19    Serial.begin(115200);// start serial monitor
20    pinMode(wifiConnecting, OUTPUT);//LED indicator wifi status flashing while connecting
21    pinMode(wifiOk, OUTPUT);//LED indicator wifi status ON if connected
22    digitalWrite(wifiConnecting, LOW);//init off
23    digitalWrite(wifiOk, LOW);//init off
24    delay(10);
25    //servo
26    servo1.attach(D3);// servo on D3
27    servo2.attach(D4);// servo on D4
28  }
29  void loop() {
37  void parseJson(JsonObject& json_object) {//parse the commands from the json object
58  /*
78  void extractJson() { //extract JSON string from HTTP data
92  void payload() { // extract wanted data from HTTP response
103 void httpRequest() { // get HTTP response from webserver
130 void wifiConnect() { // connect to local network
151
```

loop

```
29 void loop() {
30   if (WiFi.status() != WL_CONNECTED) wifiConnect();
31   httpRequest();//get data from webserver
32   if (debug) Serial.println(httpResponse);
33   payload();//extract wanted data from HTTP response
34   if (debug)Serial.println(httpResponse); //debug
35   extractJson();
36 }
```

wifiConnect()




```
130 void wifiConnect() { // connect to local network
131   int ledState = 0;//flasher
132   digitalWrite(wifiOk, LOW);
133   digitalWrite(wifiConnecting, HIGH);
134   Serial.println();
135   Serial.print("Connecting to ");
136   Serial.println(ssid);
137   WiFi.begin(ssid, password);
138   while (WiFi.status() != WL_CONNECTED) {
139     delay(500);
140     Serial.print(".");
141     if (ledState == 0) ledState = 1;
142     else ledState = 0;
143     digitalWrite(wifiConnecting, ledState);
144   }
145   Serial.println("");
146   Serial.print("WiFi connected, IP address: " + WiFi.localIP() );
147   if (debug) WiFi.printDiag(Serial); // print Wi-Fi diagnostic information
148   digitalWrite(wifiConnecting, LOW);
149   digitalWrite(wifiOk, HIGH);
150 }
151
```

Http Request()

```
92  void payload() { // extract wanted data from HTTP response
103 void httpRequest() { // get HTTP response from webserver
104   digitalWrite(wifiOk, LOW);//flash LED
105   delay(2000);//time between requests
106   digitalWrite(wifiOk, HIGH);//flash LED
107   httpResponse = ""; //empty string
108   WiFiClient client; //instance
109   if (client.connect(server, 80)) { //connect to webserver on port 80
110     client.println("GET " + path +  " HTTP/1.1");//construct a HTTP GET request
111     client.println("Host: " + String(server));
112     client.println("Connection: keep-alive");
113     client.println();
114   }
115   else {
116     Serial.println( "Webserver does not respond");
117     return;
118   }
119   while (client.connected()) {
120     while ( client.available() ) {
121       httpResponse += char(client.read());//mogelijk memory problemen
122       if (httpResponse.length() > 450) {
123         Serial.println("Receive buffer overflow");//prevent buffer overflow
124         httpResponse = ""; //empty string
125         return;
126       }
127     }
128   }
129 }
130 void wifiConnect() { // connect to local network
151
152
```

httpRequest()

payLoad()

```
92  void payload() { // extract wanted data from HTTP response
93    String endOfHeader = "\r\n\r\n";
94    int foundEOH = -1;
95    // look for EOH end of header
96    for (int i = 0; i <= httpResponse.length() - endOfHeader.length(); i++) {
97      if (httpResponse.substring(i, endOfHeader.length() + i) == endOfHeader) {
98        foundEOH = i;
99      }
100   }
101   httpResponse = httpResponse.substring(foundEOH);// strip the HTTP header
102 }
```

payLoad()

extractJson()

//code voor ArduinoJson library
version 5.8.1

```
78 void extractJson() { //extract JSON string from HTTP data
79   int size = httpResponse.length() + 1;
80   char json[size];
81   httpResponse.toCharArray(json, size);
82   StaticJsonBuffer<200> jsonBuffer;
83   JsonObject& json_object = jsonBuffer.parseObject(json);
84
85   if (!json_object.success())
86   {
87     Serial.println("parseObject() failed");
88     return;
89   }
90   parseJson( json_object);//parse the commands from the json object
91 }
```

extractJson()

extractJson()

//code voor ArduinoJson library
version 6.9.1

```
78 void extractJson() { //extract JSON string from HTTP data
79   int size = httpResponse.length() + 1;
80   char json[size];
81   httpResponse.toCharArray(json, size);
82  StaticJsonDocument<256> json_object; //<==== ArduinoJson library version 6.9.1
83  deserializeJson(json_object, json); // json (string) wordt geparsed json_object (array)
84 }
```

extractJson()

parseJson()

```
37 void parseJson(JsonObject& json_object) {//parse the commands from the json object
38   if (strcmp(json_object["device1"], "on") == 0) // ==0 is equal
39   {
40     Serial.print("device 1 on value => ");
41     String range1Str = json_object["range1"];
42     int range1 = map (range1Str.toInt(), 0, 1000, 0, 180);
43     servo1.write(range1);
44     Serial.println(range1Str);//debug
45
46   }
47   if (strcmp(json_object["device2"], "on") == 0) // ==0 is equal
48   {
49     Serial.print("device 2 on value => ");
50     String range2Str = json_object["range2"];
51     int range2 = map (range2Str.toInt(), 0, 1000, 0, 180);
52     servo2.write(range2);
53     Serial.println(range2Str);//debug
54     //digitalWrite(D2, HIGH);
55     //analogWrite(D2, range1);
56   }
57 }
```

Doe het

parseJson()

Minimum opdracht

- Download en installeer: https://github.com/MediacollegeAmsterdam/IOT-Internet-of-things-GD2
- Verander de code:
  - JSON bestand moet van jouw eigen server (M-cloud of eigen domein) geladen worden:
  - {"device1":"on","range1":"80","device2":"on","range2":"700","device3":"on","range3":"-370"}
- Laat zien dat jij het JSON bestand kunt aanpassen en dat de nieuwe data door de NodeMCU geparsed wordt