Dein NodeMCU Amica Modul soll nun als Webclient arbeiten. Das heißt, er ruft eine bestimmte Webseite auf einem Webserver auf, überträgt dabei Daten zum Webserver und erhält eine Antwort vom Webserver. Diese Antwort kann Daten zur Steuerung des Moduls enthalten.

Der erste Teil des Programmes WiFiWebClient-0.ino stellt die Verbindung zu einem Accesspoint her und protokolliert dies auf dem Seriellen Monitor der Arduino DIE.
Sollte dir kein eigenen Webserver zur Verfügung stehen, kannst du in der Zeile 6 bei **host** statt der IP-Adresse deines Webservers auch www.example.com angeben.
Im folgenden Beispiel wird auf den Webserver des PC mit der IP-Adresse **10.50.20.39** zugegriffen.

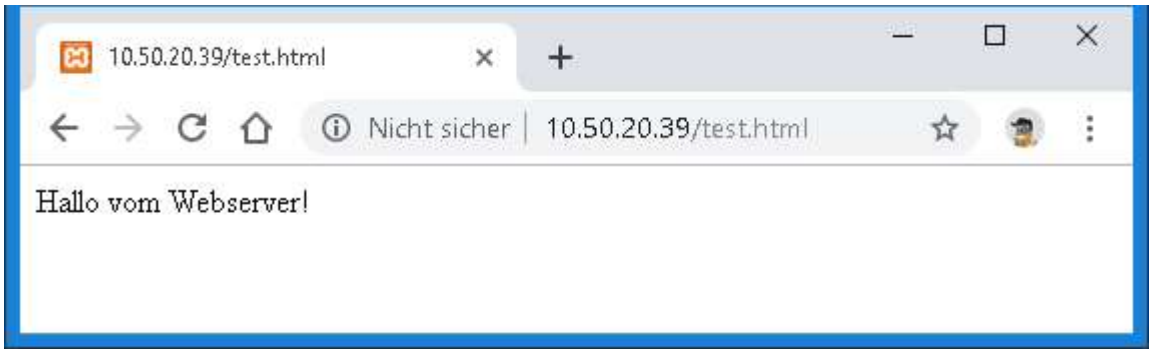
```
1  // nodeMCU Amica V2 Module mit ESP8266 12E - WiFiWebClient-0
2  // Board: NodeMCU 1.0 (ESP12E Module)
3  #include <ESP8266WiFi.h>
4  #define ssid      "mrge-ap-bu46"
5  #define password  ""
6  #define host      "10.50.20.39"  // IP deines PC
7  #define httpPort  80
8
9  void setup() {
10   Serial.begin(115200); Serial.println("\nWiFiWebClient-0");
11   WiFi.mode(WIFI_STA);    pinMode(D0,OUTPUT); digitalWrite(D0,LOW);
12   WiFi.begin(ssid, password);
13   Serial.println("Verbinde mit " + String(ssid));
14   while (WiFi.status() != WL_CONNECTED) {
15     Serial.print("."); delay(500);
16   }
17   Serial.println("\nWiFi verbunden mit " + String(ssid));
18   Serial.println("IP address: " + WiFi.localIP().toString());
19 }
```

Im nachfolgendem loop()-Teil wird regelmäßig eine Webseite aufgerufen und die Antwort des Webservers auf dem Seriellen Monitor ausgegeben.

1. Zeilen 22-30:
   Stellt eine Verbindung zum Webserver her. Falls das nicht gelingt, wird es erneut versucht.
2. Zeile 31-32:
   Der Pfad zur Webseite auf dem Server wird definiert und die URL zum Webserver wird auf den Seriellen Monitor ausgegeben.
3. Zeile 34-37:
   Die Anfrage an den Webserver wird zusammengestellt und abgesendet.
4. Zeile 39:
   Solange der Client keine Antwort erhält, tue nichts.
5. Zeile 41-43:
   Solange Daten vom Webserver an den Client gesendet werden, ließ diese und gib sie auf den Seriellen Monitor aus.
6. Zeile 44-45:
   Client wird gestoppt.

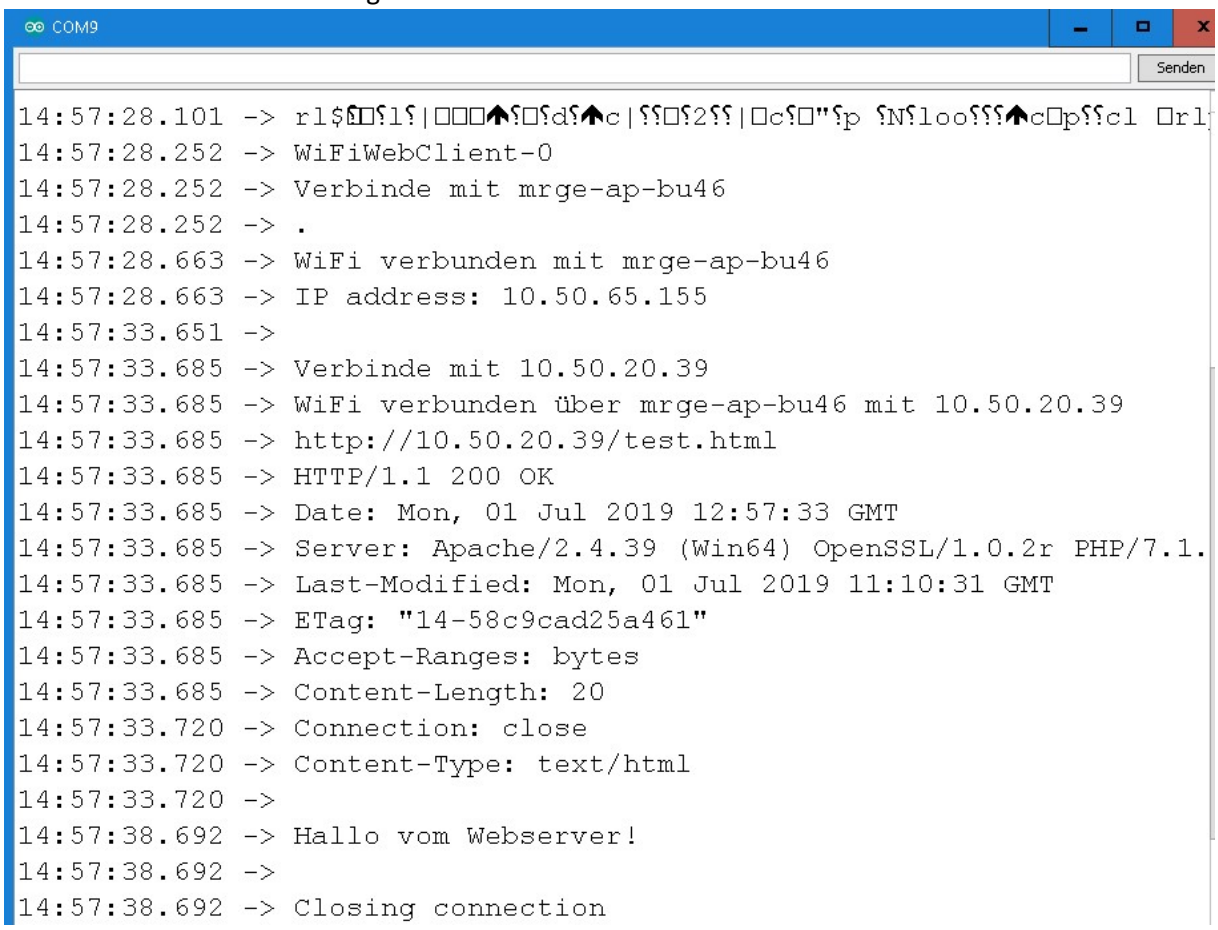
```
21⊟void loop() {
22   digitalWrite(D0,HIGH);   delay(5000);  digitalWrite(D0,LOW);
23   Serial.println("\nVerbinde mit " + String(host));
24   WiFiClient client;                    // Verbindung zum Webserver
25⊟  if (!client.connect(host, httpPort)) {
26     Serial.println("connection failed");
27     return;
28   }
29   Serial.println("WiFi verbunden über " + String(ssid) +
30                  " mit " + String(host));
31   String pfad = "/test.html";
32   Serial.println("http://" + String(host) + pfad);
33
34   // Anfrage an Webserver
35   client.print(String("GET ") + pfad + " HTTP/1.1\r\n" +
36                "Host: " + host + "\r\n" +
37                "Connection: close\r\n\r\n");
38
39   while (client.available() == 0) {} // Warte auf Anwort des Webservers
40
41⊟  while (client.available()) {       // Lies die Anwort des Webservers
42     Serial.println(client.readStringUntil('\n'));
43   }
44   client.stop();
45   Serial.println("\nClosing connection");
46 }
```

Der Webserver hat in der Datei c:\xampp\htdocs\test.html lediglich „Hallo vom Webserver“ stehen!. Diese Seite kann mit deinem Browser auf dem PC oder im Handy per http://10.50.20.39/test.html aufgerufen werden.



Ruft das NodemMCU-Modul diese URL mittels des Skripts WiFiWebClient-0 auf, erhalten wir die nachfolgenden Ausgaben.

Nach dem Starten unseres Programm sehen wir im Seriellen Monitor



Im nächsten Beispiel wird die c:\xampp\htdocs\test.php aufgerufen. Diese Datei muß du erst erstellen.

```php
<?php
  echo "Hallo NodeMCU";
  echo " - REMOTE_ADDR:".$_SERVER['REMOTE_ADDR'];
?>
```

Der Aufruf http://10.50.20.39 im Webbrowser vom PC mit der IP-Adresse **10.50.20.41** liefert folgende Ausgabe.

Jetzt muß im Editor der Arduino-IDE die Zeile 31 verändert werden, statt **test.html** ist nun **test.php** anzugeben und das geänderte Programm ist auf den NodeMCU hochzuladen. Auf dem Seriellen Monitor erscheint jetzt eine entsprechnde Antwort des Webserver

```
15:45:03.693 -> WiFi verbunden über mrge-ap-bu46 mit 10.50.20.39
15:45:03.693 -> http://10.50.20.39/test.php
15:45:03.693 -> HTTP/1.1 200 OK
15:45:03.693 -> Date: Mon, 01 Jul 2019 13:45:03 GMT
15:45:03.693 -> Server: Apache/2.4.39 (Win64) OpenSSL/1.0.2r PHP/7.1.
15:45:03.693 -> X-Powered-By: PHP/7.1.29
15:45:03.693 -> Content-Length: 41
15:45:03.693 -> Connection: close
15:45:03.693 -> Content-Type: text/html; charset=UTF-8
15:45:03.693 ->
15:45:08.698 -> Hallo NodeMCU - REMOTE_ADDR: 10.50.65.155
15:45:08.698 ->
```

Ändere nun das php-Skript wie folgt ab:

```php
<?php
    $zw="\r\n";
    if($_SERVER['REMOTE_ADDR']=='10.50.20.41') {
        $zw="<br>";
    }
    echo "Hallo NodeMCU!".$zw;
    echo "REMOTE_ADDR:      ".$_SERVER['REMOTE_ADDR'].$zw;
    echo "HTTP_HOST:        ".$_SERVER['HTTP_HOST'].$zw;
    echo "REQUEST_METHOD:   ".$_SERVER['REQUEST_METHOD'].$zw;
    echo "REQUEST_TIME:     ".$_SERVER['REQUEST_TIME'].$zw;
    echo "QUERY_STRING:     ".$_SERVER['QUERY_STRING'].$zw;
    echo $zw;
?>
```

Statt der IP-Adresse 10.50.20.41 ist die IP-Adresse deines PCs oder Handy's einzutragen.

Die Zeilenwechsel in der Variablen **$zw** sind für einen Webbrowser anders als für den Seriellen Monitor der Arduino IDE.

Erweitern wir die Datei test.php und lernen, wie in PHP auf den Inhalt von QUERY_STRING zugreifen können.

```
12
13      parse_str($_SERVER['QUERY_STRING'], $output);
14      echo "<pre>";
15      print_r($output);
16      echo "</pre>";
17
18      $feld='value';
19      if (array_key_exists($feld, $_GET)) {
20          echo "key '".$feld."' = ".$_GET[$feld].$zw;
21      }
22  ?>
```

Die nächste Erweiterung schreibt Informationen in eine Textdatei **dht22+ldr.txt**.

```
22      $inhalt = $_SERVER['REQUEST_TIME']."\t".$_SERVER['QUERY_STRING'];
23      $handle = fopen ("dht22+ldr.txt", "a+");
24
25      fwrite ($handle, $inhalt."\n");
26      fclose ($handle);
27  ?>
```

Jeder Aufruf der Datei test.php über den Webserver fügt eine neue Textzeile an das Ende der Textdatei **dht22+ldr.txt** an .

Wir können aber vor dem Schreiben in die Textdatei, die Werte aus QUERY_STRING aufspalten.

```
22      $x = explode("&",$_SERVER['QUERY_STRING']);
23      echo "<pre>";
24      var_dump($x);
25      echo "</pre>";
26      foreach ($x as $l) {
27          echo $l.$zw;
28          list($variable,$wert) = explode("=",$l);
29          echo $variable," => ", $wert,$zw;
30      }
```