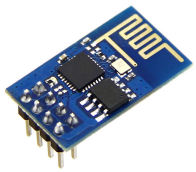
Arduino WIFI

ATTENTION, pas mal de problèmes avec ce module, blocages, pertes de connexion...

Module ESP8266

ESP8266 ESP-01



Cablage

Vue de dessous du composant Arduino

```
+-----------------------------+
|                     GND TX  |    TX <-----> 10
|                     0   3,3V|
|                     0   0   |
|                     RX  3,3V|    RX <-----> 11
+-----------------------------+
```

ATTENTION, l'alimentation est du 3,3 V

Exemple : mini serveur web

```
#include <SoftwareSerial.h>

// A MODIFIER
#define DEBUG true
#define RX 10 // broche arduino
#define TX 11 // broche arduino
#define ssid "xxxxxxxxxx"
#define password "xxxxxxxxxxxx"

String pageaccueil = "<h1>Accueil</h1>Commandes disponibles:<br>test1<br>";

// FIN DES MODIFICATIONS

SoftwareSerial ESP8266(RX, TX);

String NomduReseauWifi = ssid;
String MotDePasse      = password;

void envoieAuESP8266(String commande)
{
  ESP8266.println(commande);
}

void recoitDuESP8266(const int timeout)
{
  String reponse = "";
  long int time = millis();
  while ( (time + timeout) > millis())
  {
    while (ESP8266.available())
    {
      char c = ESP8266.read();
      reponse += c;
    }
  }
  Serial.print(reponse);
}
```

```
void initESP8266()
{
  Serial.println("**************** DEBUT DE L'INITIALISATION ***************");
  envoieAuESP8266("AT+RST");
  recoitDuESP8266(2000);
  envoieAuESP8266("AT+CWMODE=3");
  recoitDuESP8266(5000);
  envoieAuESP8266("AT+CWJAP=\""+NomduReseauWifi+"\",\""+MotDePasse+"\"");
  recoitDuESP8266(10000);
  envoieAuESP8266("AT+CIFSR");
  recoitDuESP8266(1000);
  envoieAuESP8266("AT+CIPMUX=1");
  recoitDuESP8266(1000);
  envoieAuESP8266("AT+CIPSERVER=1,80");
  recoitDuESP8266(1000);
  Serial.println("***************** INITIALISATION TERMINEE ****************");
}

String sendData(String command, const int timeout, boolean debug)
{
  String response = "";

  ESP8266.print(command); // send the read character to the esp8266
  long int time = millis();
  while ( (time + timeout) > millis())
  {
    while (ESP8266.available())
    {
      // The esp has data so display its output to the serial window
      char c = ESP8266.read(); // read the next character.
      response += c;
    }
  }

  if (debug)
  {
    Serial.print(response);
  }
  return response;
}

void setup()
{
  Serial.begin(9600);
  ESP8266.begin(9600);
  initESP8266();
}

void loop()
{
  String requete, webpage, cipSend, closeCommand;
  if (ESP8266.available())
  {
    Serial.println("Connexion");
    if (ESP8266.find("+IPD,"))
    {
      delay(1000);
      int connectionId = ESP8266.read() - 48;
      // subtract 48 because the read() function returns
      // the ASCII decimal value and 0 (the first decimal number) starts at 48
      if (ESP8266.find("GET /"))
      {
        requete = "";
```

```
        char cmd = ESP8266.read();
        while (cmd != ' ')
        {
          requete += cmd;
          cmd = ESP8266.read();
        }
        if (requete == "")
        {
          webpage = pageaccueil;
        }
        // a dupliquer et a modifier pour les differentes commandes
        else if (requete == "test1")
        {
          webpage = pageaccueil + "Commande test1 OK";
        }
        // fin des modifs
        else // page erreur requete inconnue
        {
          webpage = pageaccueil + "Requete inconnue " + requete;
        }
      }

      // Envoi de la reponse

      cipSend = "AT+CIPSEND=";
      cipSend += connectionId;
      cipSend += ",";
      cipSend += webpage.length();
      cipSend += "\r\n";
      sendData(cipSend, 1000, DEBUG);
      sendData(webpage, 1000, DEBUG);

      // Fermeture de la connexion

      closeCommand = "AT+CIPCLOSE=";
      closeCommand += connectionId; // append connection id
      closeCommand += "\r\n";
      sendData(closeCommand, 3000, DEBUG);
    }
  }
}
```