6.6.3 Mesurer la température et l'humidité

Ouvrir le croquis suivant dans l'IDE Arduino `Exemple_6.6.3.ino`. Vous devez intégrer les fichiers d'en-tête suivants : `ESP8266WiFi.h`, `SSD1306Wire.h`, `TimeLib.h`, `NtpClientLib.h` et `DHT.h`. Si vous n'avez pas déjà installé les bibliothèques appropriées, passez au chap. 5.2.1 à la page 16 et effectuez d'abord cette opération.

Pour cet exemple, vous devez d'abord saisir votre nom WiFi (SSID) et le mot de passe correspondant dans l'exemple de programme. Allez comme au chap. 6.6.1 décrit.Pour la mesure de la température et de l'humidité, nous utilisons le capteur combiné largement utilisé du type DHT11, pour lequel il existe déjà une bibliothèque (`DHT.h`) et des exemples. En outre, il existe de nombreux capteurs différents, par exemple de le monde Arduino, tels que :

- capteurs de température
- barrière infrarouge
- détecteur de mouvement (capteur IR)
- capteur de lumière (LDR)
- capteur de gaz
- capteur à effet Hall
- capteur de choc
- bouton tactile

La brique d'adaptateur de capteur universel (ALL-BRICK-0649) vous permet de connecter simplement de nombreux capteurs disponibles dans le commerce. Pour de nombreux capteurs, il existe déjà des exemples et des bibliothèques, de sorte qu'ils peuvent également être facilement intégrés dans leurs propres projets.

Assemblez d'abord les briques comme indiqué. Faites attention à la connexion correcte du capteur DHT11 fourni. Insérez le capteur exactement comme indiqué sur la fig. ci-dessous à l'extrême gauche dans le connecteur inférieur à 5 broches de la brique d'adaptateur de capteur. L'étiquette "5V" sur le capteur doit être à l'extrême gauche de la prise marquée "5V" sinon le capteur et la brique IoT sont irréversiblement endommagés.

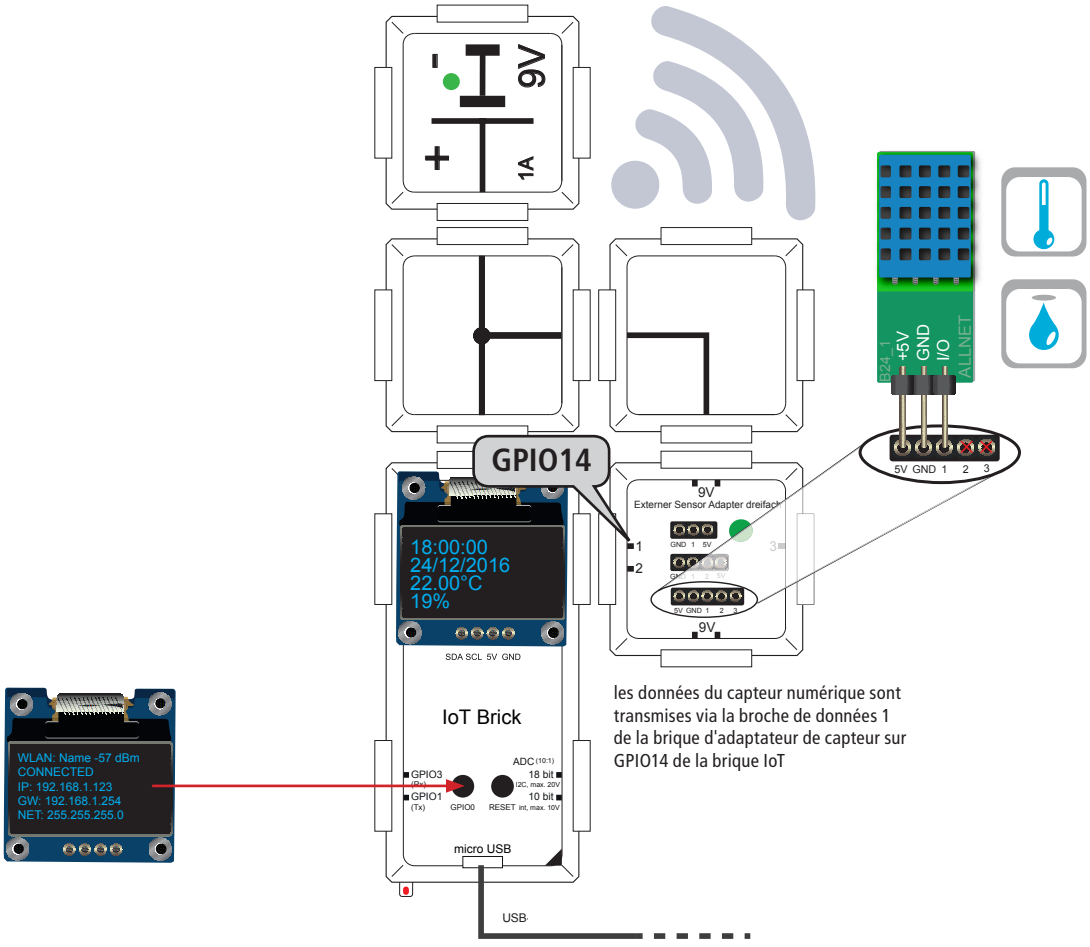

Fig.42 : Température et humidité de mesure du circuit de brique

Conseil : l'état de connexion de la brique IoT peut être affiché à tout moment en appuyant sur le bouton GPIO0

exemple de programme

```
#define DHT_TYPE DHT11     // define sensortype: DHT11
const int DHT_PIN = 14;  //data pipe of the sensor to GPIO4 of the IoT Brick
char temp[20];           //define string variable for the temperature
char humi[20];           //define string variable for the humidity

DHT dht(DHT_PIN, DHT_TYPE);    //variable of type DHT

...
void setup() {
...
   dht.begin();                 //initialize sensor
}

void loop() {
...
 if (counter%1000==0){             //Check the sensor approximately once per second
    float t = dht.readTemperature(); //Read Temperature (Celsius)
    float h = dht.readHumidity();     //Read the humidity

    sprintDouble(temp,t,2);  //Convert temp with 2 decimal places to string
    sprintDouble(humi,h,0);  //Convert humidity without decimal to string
    strcat(temp," °C");      //add °C to string
    strcat(humi," %");       //add % character to string

  }
  display.drawString(5, 30, temp);    //Prepare temperature output
  display.drawString(5, 45, humi);    //Prepare humidity output
  display.display();                 //update OLED display
...
}
```

Au début du croquis, le type de capteur DHT11 et la broche GPIO de la ligne de données (ici GPIO4) sont définis. Une variable spéciale est le type `DHT`, à l'aide duquel le capteur est initialisé dans `void setup()`

Mais maintenant, pour la mesure réelle dans la section `void loop ()`. Avec les deux fonctions, la bibliothèque de capteurs `dht.readTemperature ()` et `dht.readHumidity ()` est désactivée. La température et l'humidité sont lues par le capteur et stockées dans les deux variables à virgule flottante `t` et `h`. Avec la fonction d'aide `sprint Double ()`, les nombres à virgule flottante sont convertis en chaînes avec le nombre de décimales souhaité et l'opération de chaîne `strcat ()` ajoute l'unité à la chaîne.

En plus des valeurs de température et d'humidité, l'affichage est complété par l'heure et la date - comme déjà montré dans l'exercice 6.6.2. La sortie réelle de toutes les valeurs vers l'écran OLED est comme d'habitude avec la commande `display.display () ;`.

Parallèlement à cela, la sortie se fait également sur le moniteur série de votre ordinateur. Pour ouvrir le moniteur série, cliquez simplement sur le symbole de la loupe de l'Arduino IDE (voir chap. 5.2.3 à la page 20). Dans cet exemple, la sortie de l'exemple 6.6.2 a été complétée par la température et l'humidité.

En adaptant la connectique, de nombreux capteurs sur le marché peuvent être connectés.

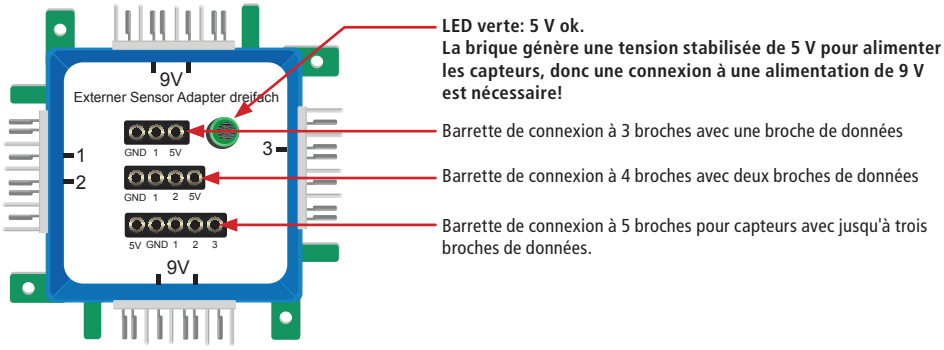

Fig.43 : Options de connexion pour "Adaptateur de capteur brique triple"

Veuillez vous assurer de la bonne connexion à la brique de l'adaptateur. En cas de doute, reportez-vous à la documentation du capteur. Sinon, il existe un risque d'endommagement irréversible du capteur et / ou de la brique IoT!

6.6.4 Taux de change du dollar sur Internet

Ouvrez le croquis suivante dans l'IDE Arduino : `Exercice_6.6.4.ino`. Vous devez intégrer les fichiers d'en-tête suivants : `ESP8266WiFi.h`, `SSD1306Wire.h`, `TimeLib.h`, `NtpClientLib.h` et `CurrencylayerClient.h`. Si vous n'avez pas déjà installé les bibliothèques appropriées, passez au chap. 5.2.1 à la page 16 et faites ceci en premier.

Pour cet exemple également, vous devez d'abord saisir votre nom WiFi (SSID) et le mot de passe correspondant dans l'exemple de programme. Aller à comme au chap. 6.6.1 décrit.

Afin de récupérer le taux de change du dollar (EUR-USD) sur Internet, vous avez besoin d'une bibliothèque spéciale "Brick-ESP8266", que vous pouvez télécharger sur http://www.brickrknowledge.de/downloads. Sauf si vous avez déjà installé toutes les bibliothèques du chap. 5.2.1, veuillez les installer d'abord comme décrit au chap. 5.2.1.2.3 à la page 19.

exemple de programme

```
#include <CurrencylayerClient.h>
...

if(counter%5000==0){      //update exchange rate approximately every 5 seconds
   currencylayer.getLastChannelItem();
   counter = 0;
   }
display.setTextAlignment(TEXT_ALIGN_LEFT);
display.setFont(ArialMT_Plain_16);
display.drawString(0, 45, currencylayer.getFieldValue(0));
display.setTextAlignment(TEXT_ALIGN_RIGHT);
display.drawString(127, 45, " EUR/USD");
display.display();
...
```

IDans cet exemple, nous obtenons les taux en dollars courants sur Internet pour calculer combien d'euros je dois payer pour un dollar (ou vice versa).

$$EUR = USD * taux\ de\ change$$

Pour obtenir le taux de conversion, nous appelons la fonction `currencylayer.get- LastChannelItem ()`. L'intervalle de temps pour la mise à jour peut être contrôlé via le compteur dans l'instruction `if (compteur% 5000 == 0)`. L'affichage est formaté comme d'habitude - Cependant, l'instruction `currencylayer. getFieldValue(0)` est importante. Seulement à cette position, le taux du dollar arrondi à 4 décimales est disponible sous forme de chaîne dans le croquis.

En plus du taux de conversion actuel, l'affichage est également complété par l'heure et la date - comme déjà montré dans l'exercice 6.6.2. La sortie réelle de toutes les valeurs vers l'affichage OLED se fait comme d'habitude avec la commande `display.display();`

Parallèlement à cela, la sortie se fait également sur le moniteur série de votre ordinateur. Pour ouvrir le moniteur série, cliquez simplement sur le symbole de la loupe de l'IDE Arduino (voir chap. 5.2.3 à la page 20) La sortie de l'exemple 6.6.2 a été complétée par quelques messages d'état.

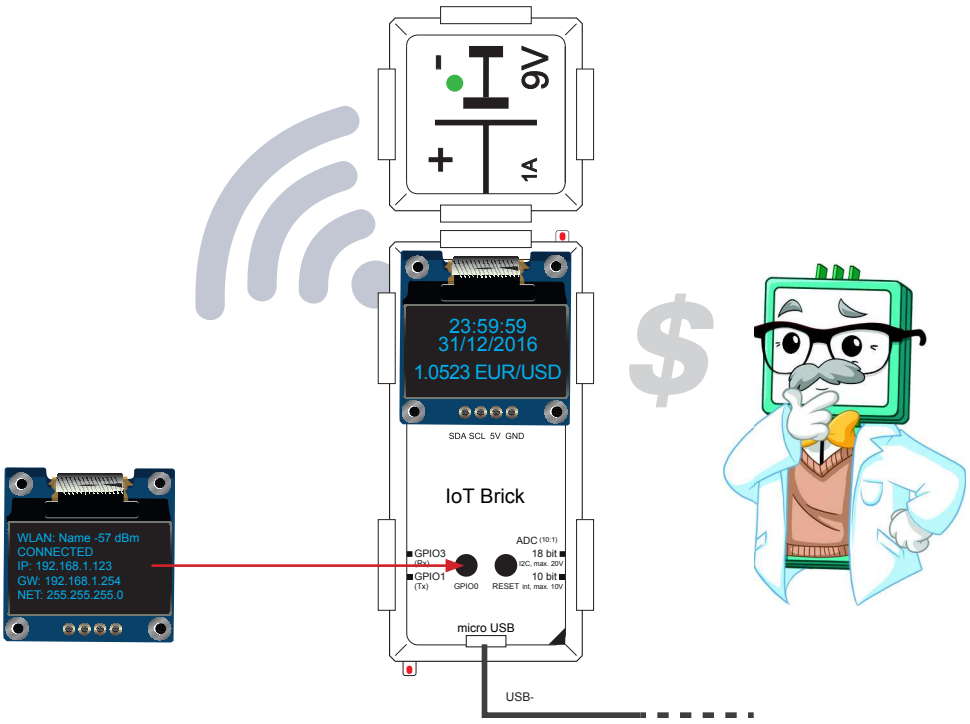

Fig.44 : Taux de change du dollar sur Internet avec un circuit de brique

CONSEIL: L'état de connexion de la brique IoT peut toujours être affiché en appuyant sur le bouton GPIO0