Programmation d’esp32 en utilisant Arduino IDE

1. Qu'est-ce que l'ESP32 ?

L'ESP32 est un microcontrôleur puissant qui peut être programmé pour réaliser différentes tâches, comme lire des capteurs, contrôler des dispositifs (LED, moteurs, etc.), et même se connecter à Internet grâce à son module Wi-Fi intégré.

2. Entrées (Inputs)

Les **entrées** sont des informations que l'ESP32 reçoit depuis l'extérieur. Cela peut être :

- Un bouton que l'on presse
- Un capteur de température ou de lumière
- Un microphone
- Etc.

L'ESP32 peut lire ces signaux en fonction des **broches d'entrée (input pins)**.

Exemple :

- Si vous branchez un bouton sur une broche GPIO de l'ESP32, il détectera quand le bouton est appuyé et pourra exécuter une action en conséquence.

3. Sorties (Outputs)

Les **sorties** sont des actions que l'ESP32 effectue en réponse aux signaux qu'il a reçus ou aux commandes programmées. Cela pourrait être :

- Allumer ou éteindre une LED
- Faire tourner un moteur
- Activer un buzzer
- Envoyer des informations à un écran
- Etc.

Les **broches de sortie (output pins)** envoient des signaux électriques pour contrôler ces dispositifs.

Exemple :

- Dans le cas de votre LED, la carte ESP32 envoie un signal de tension sur la broche associée, ce qui permet d'allumer ou d'éteindre la LED.

4. Les broches GPIO (General Purpose Input/Output)

Les **broches GPIO** de l'ESP32 sont polyvalentes et peuvent être configurées soit en entrée, soit en sortie. Par exemple :

- Si vous branchez un capteur de température, vous configurez la broche en **entrée** pour lire les données.
- Si vous voulez faire clignoter une LED, vous configurez la broche en **sortie** pour contrôler l'état de la LED.

Broches à éviter pour les sorties :

- **GPIO 6 à GPIO 11** : Réservées à l'interface interne de la mémoire flash SPI, évitez d'utiliser ces broches.
- **GPIO 34 à GPIO 39** : Ces broches sont uniquement des entrées (input only), donc elles ne peuvent pas être utilisées pour contrôler une LED.
- **Les autres broches comme 17,23,4,**… peuvent être utilisés en entrée ou sortie.

void setup()

Cette fonction s'exécute une fois lorsque l'ESP32 est mis sous tension ou réinitialisé.

Elle permet de configurer les paramètres initiaux de votre programme, par exemple en indiquant à l'ESP32 quelles broches seront utilisées comme entrées ou sorties.

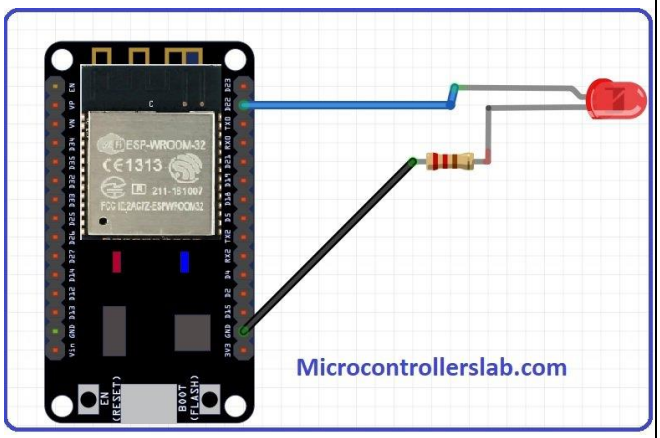
Dans l'exemple de clignotement de la LED, setup() indique à l'ESP32 de définir la broche de la LED comme sortie afin qu'il puisse contrôler si la LED est allumée ou éteinte.

void loop()

Cette fonction s'exécute en continu une fois que setup() est terminé.

L'ESP32 continue d'exécuter le code à l'intérieur de loop() encore et encore, comme un cycle sans fin. C'est parfait pour des choses comme faire clignoter une LED, où vous voulez l'allumer et l'éteindre à plusieurs reprises.

Dans cet exemple, loop() alterne entre l'allumage et l'extinction de la LED, avec un délai entre les deux.

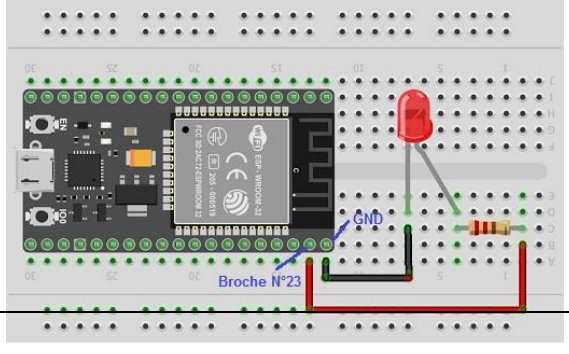


Exemple : clignoter une diode LED

```
// Définir la broche GPIO où la LED est connectée
int ledPin = 17; // 17 est le numéro de pin utilisé.


void setup() {
// Initialiser la ledpin(17)  en tant que sortie
  pinMode(ledPin, OUTPUT);
}


void loop() {
```



```
  // Turn the LED on
  digitalWrite(ledPin, HIGH);
  // Wait for 1 second (1000 milliseconds)
  delay(1000);

  // Turn the LED off
  digitalWrite(ledPin, LOW);
  // Wait for 1 second (1000 milliseconds)
  delay(1000);
}
```

Conclusion

En résumé, l'ESP32 permet de gérer des **entrées** et des **sorties**, c'est-à-dire de lire des informations (input) et de contrôler des dispositifs (output). Cela ouvre la porte à de nombreuses applications, comme des systèmes automatisés, des projets domotiques, ou encore des robots.

Code pour définir les entrées et sorties

Dans le programme, on utilise la fonction `pinMode()` pour spécifier si une broche est une entrée ou une sortie. Voici quelques exemples :

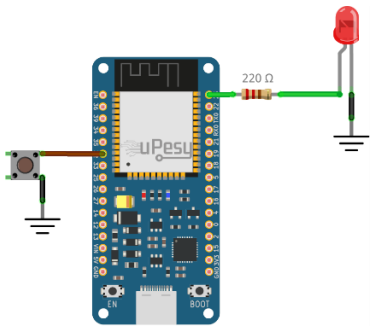
Pour une entrée :

```
pinMode(pinBouton, INPUT); // Configurer la broche du bouton comme entrée
```

Pour une sortie :

```
pinMode(pinLED, OUTPUT); // Configurer la broche de la LED comme sortie
```

6. Exemple concret avec un bouton et une LED



- **Entrée** : Le bouton connecté à l'ESP32 détecte quand vous appuyez dessus.
- **Sortie** : Lorsque le bouton est appuyé, l'ESP32 allume la LED. Quand on relâche le bouton, l'ESP32 éteint la LED.

```
const int buttonPin = 13;
const int ledPin = 17;

// État actuel et précédent du bouton
int buttonState = HIGH;
int previousButtonState = HIGH;

// État de la LED
int ledState = LOW;

void setup() {
    Serial.begin(115200);

    // Configurer les broches
    pinMode(buttonPin, INPUT_PULLUP); // Utilise le pull-up interne
    pinMode(ledPin, OUTPUT);
    digitalWrite(ledPin, ledState);   // Initialiser l'état de la LED
}

void loop() {
    // Lire l'état actuel du bouton
    buttonState = digitalRead(buttonPin);
    Serial.println(buttonState);

    // Si l'état du bouton a changé de relâché (HIGH) à pressé (LOW)
    if (buttonState == LOW && previousButtonState == HIGH) {
        // Inverser l'état de la LED
        ledState = !ledState;
        digitalWrite(ledPin, ledState); // Mettre à jour la LED
    }

    // Mémoriser l'état actuel du bouton pour le prochain cycle
    previousButtonState = buttonState;

    delay(50); // Petit délai pour éviter les rebonds
}
```