GUIDE GRATUIT • ÉLECTRONIQUE

Réaliser son premier projet avec un microcontrôleur

Dernier chapitre de la formation :
programmez votre premier montage

• • •

📐 FONDAMENTAUX

🧰 MATÉRIEL

⚡ PRATIQUE

BLOG DAUTEK

Réaliser son premier projet avec un microcontrôleur : guide pratique

Le dernier chapitre de la formation : programmez votre premier montage intelligent

📖 Chapitre 13/14 | 🗂️ Passer à la pratique | ⏱️ ~6 min de lecture

🎯 Nous y voilà : le dernier chapitre de cette formation gratuite. Après douze chapitres de théorie, de matériel et de premiers montages, vous allez programmer votre tout premier projet avec un microcontrôleur, de bout en bout, en mobilisant absolument tout ce que vous avez appris jusqu'ici.

⚡ Le matériel nécessaire

- Une carte Arduino Uno ou compatible, avec son câble USB
- Une breadboard
- Une LED et sa résistance de protection (calculée au chapitre 4)
- Deux fils de câblage (jumpers)
- Un ordinateur avec l'IDE Arduino installé (gratuit)

🔌 Étape 1 : le montage matériel

Reprenez le montage du chapitre 10 (résistance + LED), mais cette fois, reliez l'extrémité libre de la résistance non plus au rail d'alimentation de la breadboard, mais directement à une broche numérique de l'Arduino (par exemple la broche 13), et la patte courte de la LED à une broche GND de la carte.

🧩 Étape 2 : votre premier programme

Un programme Arduino (appelé "sketch") s'organise toujours en deux grandes parties : la fonction **setup()**, exécutée une seule fois au démarrage (on y définit par exemple qu'une broche est une sortie), et la fonction **loop()**, exécutée en boucle continue tant que la carte est alimentée (c'est là que se trouve la logique répétée, comme un clignotement).

📌 **À retenir :** la logique d'un programme Arduino tient en une phrase : "préparer une fois, puis répéter en boucle". C'est cette simplicité qui rend le langage accessible dès les premiers essais.

Instruction	Rôle
pinMode(13, OUTPUT);	Déclare la broche 13 comme une sortie, dans setup()
digitalWrite(13, HIGH);	Envoie du courant sur la broche 13 : la LED s'allume
digitalWrite(13, LOW);	Coupe le courant sur la broche 13 : la LED s'éteint
delay(1000);	Marque une pause de 1000 millisecondes (1 seconde)

En enchaînant "allumer, attendre une seconde, éteindre, attendre une seconde" dans la fonction loop(), vous obtenez une LED qui clignote indéfiniment, sans aucun composant supplémentaire : c'est le microcontrôleur qui gère seul la temporisation, là où un montage purement analogique nécessiterait un circuit dédié bien plus complexe.

Voici le programme complet, prêt à être recopié dans l'IDE Arduino :

```
void setup() {
  pinMode(13, OUTPUT);  // La broche 13 pilote la LED
}

void loop() {
  digitalWrite(13, HIGH);  // Allume la LED
  delay(1000);              // Attend 1 seconde
  digitalWrite(13, LOW);   // Éteint la LED
  delay(1000);              // Attend 1 seconde
}
```

📐 Étape 3 : téléverser et observer

1. Reliez la carte Arduino à l'ordinateur par câble USB.
2. Sélectionnez le bon modèle de carte et le bon port dans l'IDE Arduino.
3. Cliquez sur le bouton de téléversement (la flèche) : le programme est transféré sur la carte.
4. Observez la LED : elle doit clignoter au rythme défini par vos instructions delay().

🛠 Diagnostiquer avec le moniteur série

Avant même d'ajouter le capteur de luminosité, prenez l'habitude d'utiliser le moniteur série (Serial Monitor) de l'IDE Arduino : c'est l'équivalent d'un multimètre pour votre code, qui affiche en temps réel ce qui se passe dans votre programme. Ajoutez `Serial.begin(9600);` dans votre fonction setup(), puis utilisez `Serial.println(valeur);` n'importe où dans votre programme pour afficher une valeur à l'écran de votre ordinateur : c'est l'outil de débogage le plus utile pour comprendre pourquoi un programme ne se comporte pas comme prévu.

💡 Pour aller plus loin : une LED qui réagit à la luminosité

Une fois ce premier programme maîtrisé, l'étape naturelle suivante consiste à ajouter un capteur de luminosité (LDR) branché sur une entrée analogique de l'Arduino, puis à modifier le programme pour qu'il allume la LED uniquement lorsque la luminosité ambiante descend sous un certain seuil : vous venez de reconstituer, à petite échelle, la logique de base d'un lampadaire ou d'un éclairage extérieur automatique.

```
void setup() {
  pinMode(13, OUTPUT);
  Serial.begin(9600);       // Active le moniteur série
}

void loop() {
  int luminosite = analogRead(A0);   // Lit la LDR (0 à 1023)
  Serial.println(luminosite);         // Affiche la valeur lue

  if (luminosite < 400) {             // Seuil à ajuster selon votre pièce
    digitalWrite(13, HIGH);           // Il fait sombre : LED allumée
  } else {
    digitalWrite(13, LOW);            // Il fait clair : LED éteinte
  }
}
```

📌 **À retenir :** la valeur lue par analogRead() varie de 0 à 1023, et non de 0 à 5V directement : c'est une conversion numérique de la tension analogique reçue sur la broche, un concept déjà abordé au chapitre 1 (analogique vs numérique).

💡 **Le saviez-vous ?** Le principe de cette simple LED qui réagit à la luminosité est exactement celui utilisé dans la grande majorité des ampoules et capteurs domotique du commerce, à la différence près qu'ils communiquent en plus leur état par un protocole radio (Zigbee, Wi-Fi...), un sujet que nous couvrons largement ailleurs sur ce blog.

🎯 Mon avis : ce projet n'est qu'un point de départ

Ce montage peut sembler modeste au regard des treize chapitres qui l'ont précédé, et c'est précisément le sens de cette formation : vous donner une base assez solide pour que ce premier projet ne soit pas un aboutissement, mais un point de départ. Vous avez désormais toutes les briques (composants, sécurité, calculs, lecture de schéma, matériel, montage et programmation) pour explorer sereinement vos propres idées, à votre rythme.

✏️ **Petit défi final :** modifiez le programme pour faire clignoter deux LED en alternance plutôt qu'une seule, en réutilisant tout ce que vous avez appris dans ce chapitre.

📥 GUIDE GRATUIT

Gardez ce projet complet sous la main

La version PDF reprend le montage, le code commenté et la piste d'extension avec un capteur de luminosité.

📋 Questions fréquentes

1 Faut-il déjà savoir programmer pour suivre ce chapitre ?

Non : le langage Arduino utilisé ici reste volontairement simple, et ce premier projet ne demande aucune connaissance préalable en programmation.

2 Le programme peut-il rester sur la carte sans l'ordinateur branché ?

Oui, une fois téléversé, le programme reste en mémoire sur la carte : elle peut ensuite être alimentée par une simple pile ou un adaptateur secteur.

3 Que faire si la LED ne clignote pas ?

Vérifiez d'abord le montage physique (sens de la LED, bonne broche), puis la sélection du bon port et du bon modèle de carte dans l'IDE Arduino.

4 À quoi sert exactement le moniteur série ?

Il affiche en temps réel les valeurs que votre programme choisit d'imprimer (via Serial.println), ce qui permet de vérifier qu'un capteur renvoie bien la valeur attendue avant même de se soucier de la logique qui l'exploite.

🔍 Merci d'avoir suivi cette formation jusqu'au bout !

Vous voilà arrivé au terme des treize chapitres de cette formation gratuite. Des bases de l'électricité jusqu'à votre premier programme, vous disposez maintenant d'une fondation solide pour explorer vos propres projets, en toute autonomie.

Pour continuer sur votre lancée, voici quelques pistes concrètes, directement reliées à ce que vous venez d'apprendre : ajoutez un module relais (vu au chapitre 3) à votre montage pour piloter une vraie lampe en toute sécurité ; explorez nos articles dédiés à Home Assistant pour faire dialoguer votre microcontrôleur avec une box domotique complète ; ou lancez-vous dans la conception d'un circuit imprimé sur-mesure (chapitre 11) pour transformer ce prototype en un objet fini et durable. N'hésitez pas à revenir consulter n'importe quel chapitre de cette formation au fil de vos besoins : elle reste gratuite et accessible en intégralité, à votre rythme.

© Blog Dautek — blog.dautek.fr — Guide gratuit à usage personnel, non destiné à la revente.