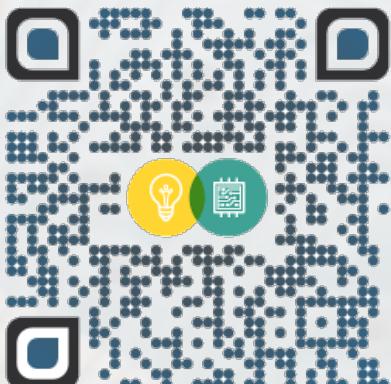


Initiation à l'électronique et à la programmation avec Arduino

[github.com/
hackstub/
atelierArduino](https://github.com/hackstub/atelierArduino)



Alexandre Aubin,
Nicolas Chesnais,
Jérémie Wojtowicz
[Hackstub]

[Médiathèque de Neudorf]
21 novembre 2015

Pourquoi cet atelier

« À l'heure où la technologie impacte de plus en plus les aspects sociaux, politiques et culturels de nos vies,

il est crucial que celle-ci reste comprise et contrôlée par les citoyens

et non qu'elle devienne un instrument de pouvoir pour ceux qui la construise. »

Objectifs

- **Comprendre** les principes de bases de l'électronique et de la programmation
- **Réaliser** des montages avec Arduino
- **Acquérir** les connaissances pour démarrer son propre projet

Programme

[1] **Bases** de l'électronique

Matin

[2] Votre tout **premier montage** Arduino

[3] Construction d'un **robot**

[4] **Programmation** d'un circuit

Aprem

[5] Circuit avancé : **détecteur d'obstacle**
et **activation d'interrupteur**

[1]

Les bases de l'électronique

(et en fait, de la programmation un peu aussi)

Définition

Electronique

↔

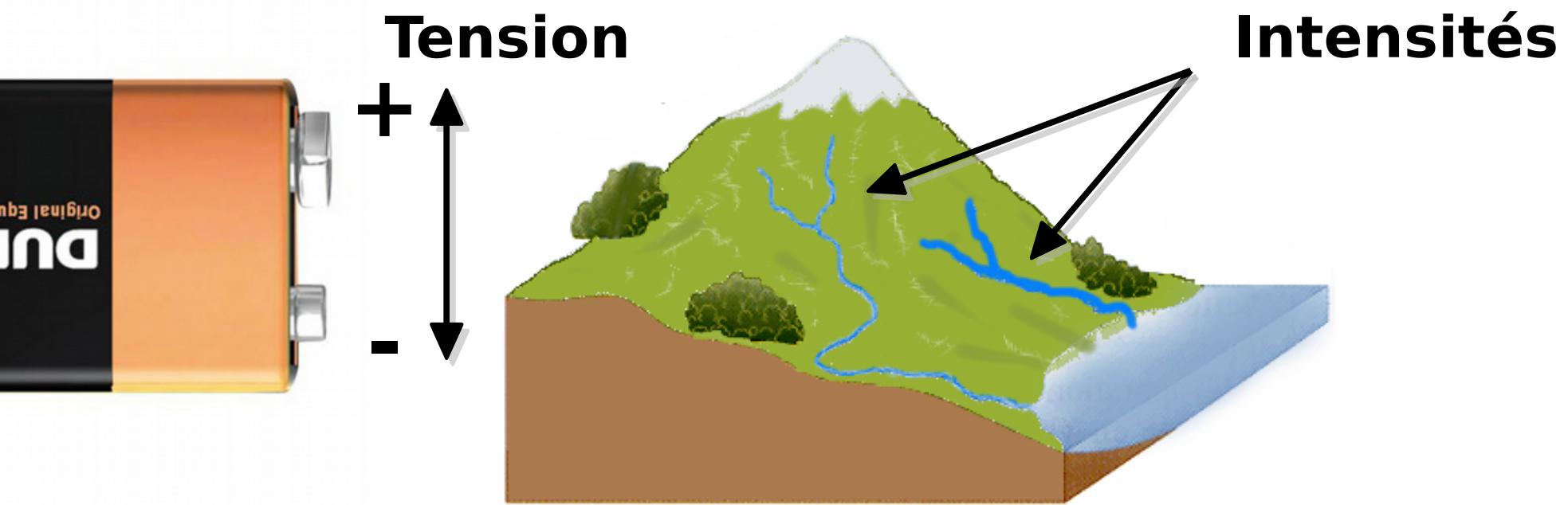
Hydraulique

Manipuler
des électrons
avec des fils

Manipuler
de l'eau
avec des tuyaux

Tensions, intensité

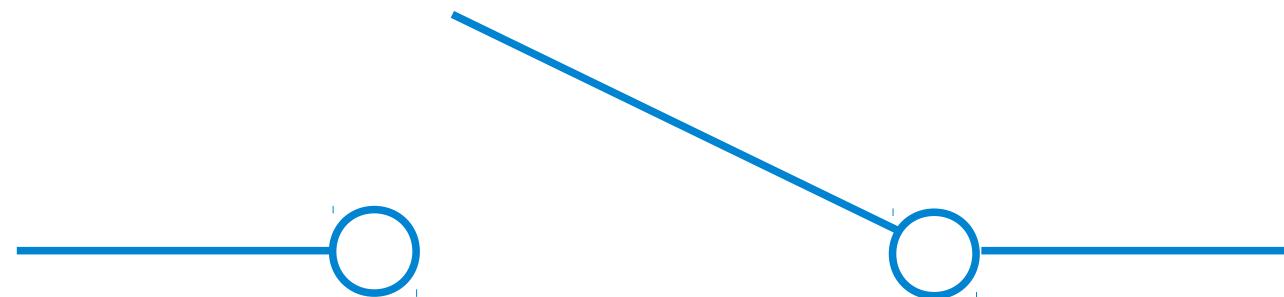
Tension ~ Pression/hauteur
Intensité ~ Débit



Interrupteur

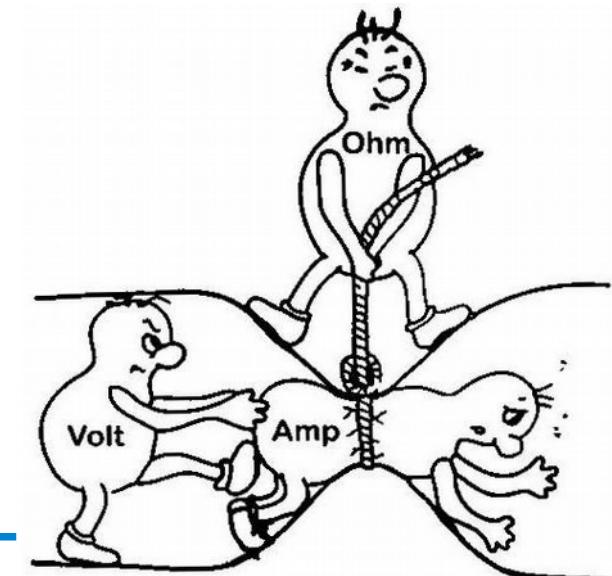
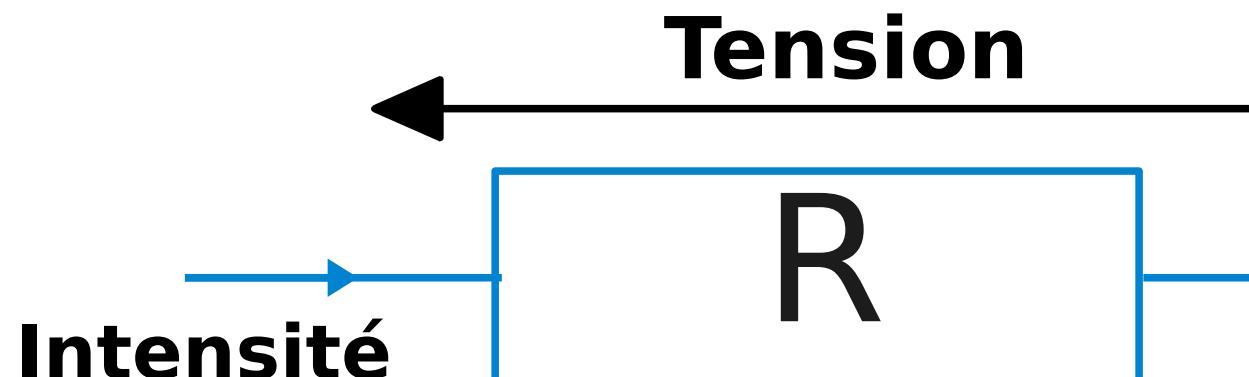


Porte / robinet
Pour donner du contrôle à l'utilisateur



Résistance

Protège les composants d'un débit trop élevés),



$$\text{Tension} = R \times \text{Intensité}$$

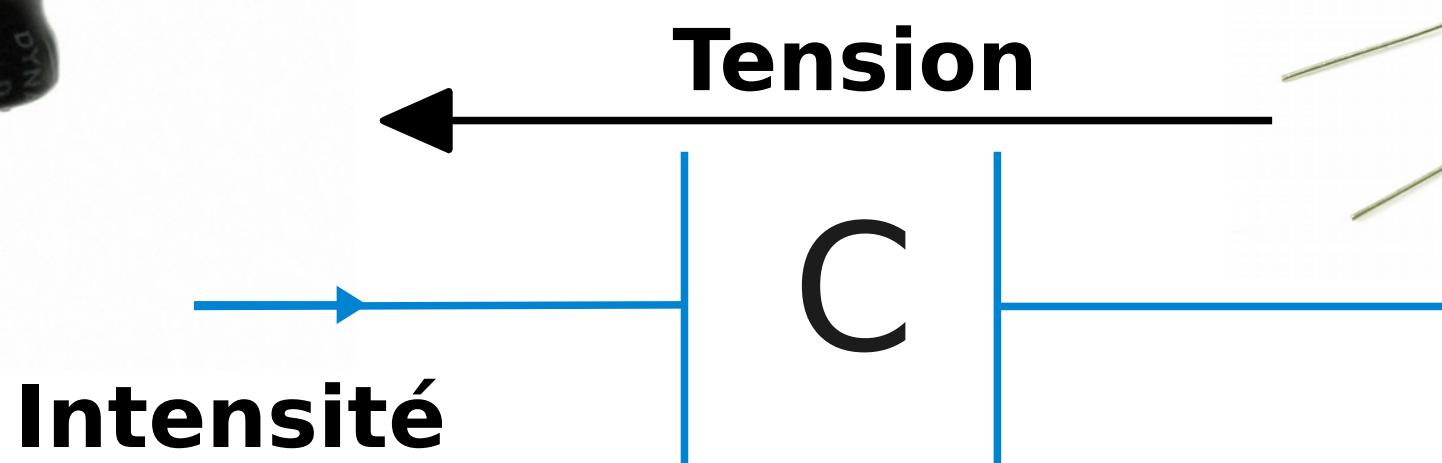
Certaines résistances dépendent de la température ou de la luminosité !



Condensateur



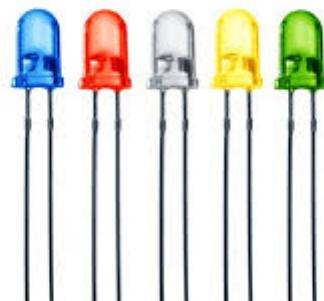
Anologue d'un barrage
Stocke temporairement le courant,
Amorti les grandes variations
de tension



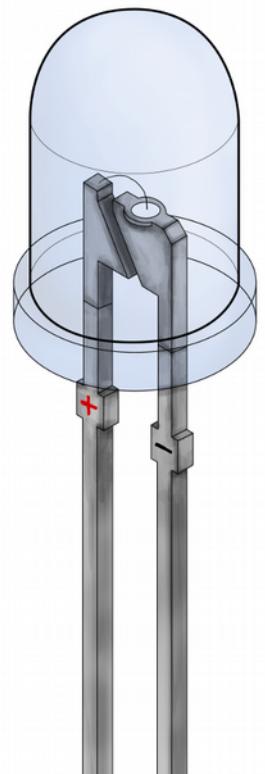
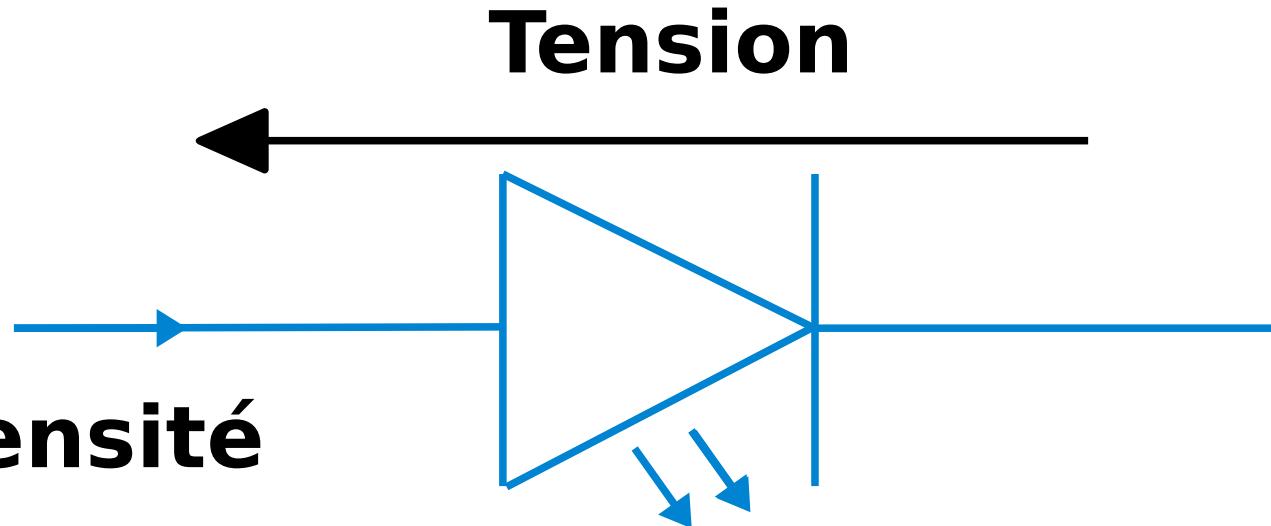
Diode / LED

Bloque le passage du courant
dans un sens !

Sert comme indicateur lumineux



Intensité



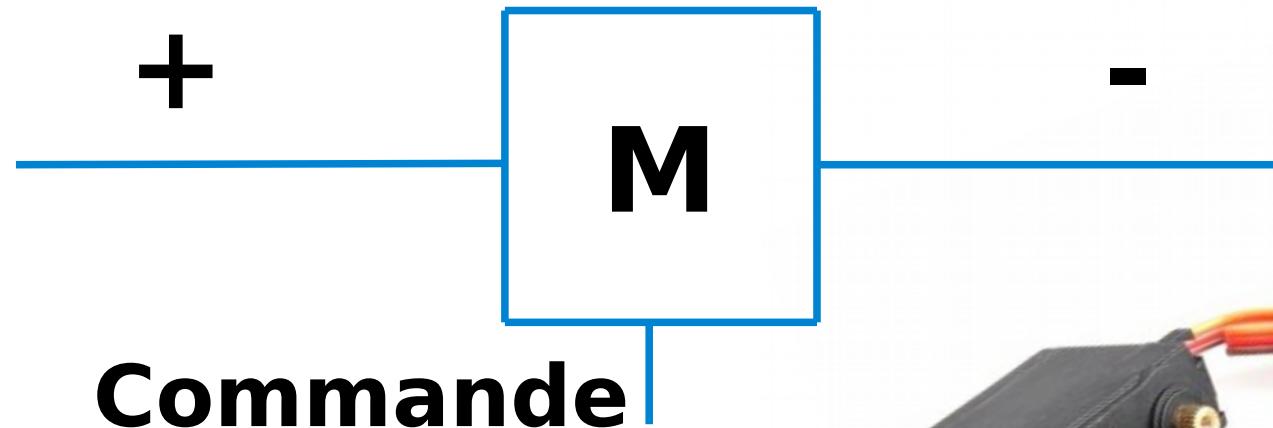


Moteur

Analogue d'un moulin à eau

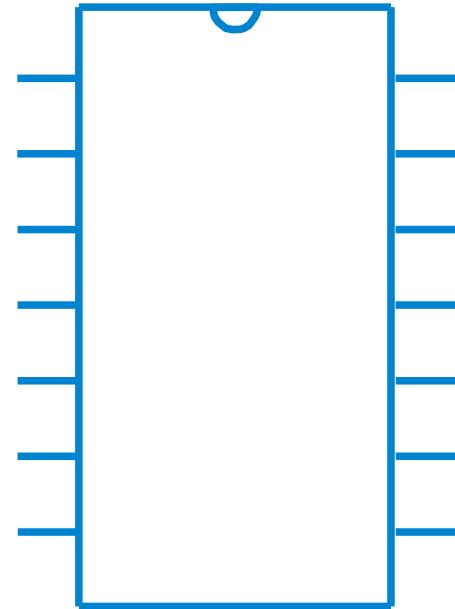


Energie électrique → Energie mécanique



Puces électroniques

Énormément de fonctions possibles,
de la simple opération binaire
au microprocesseur...



Analogique, numérique

Deux 'types' d'électroniques,
adaptées à des applications
différentes

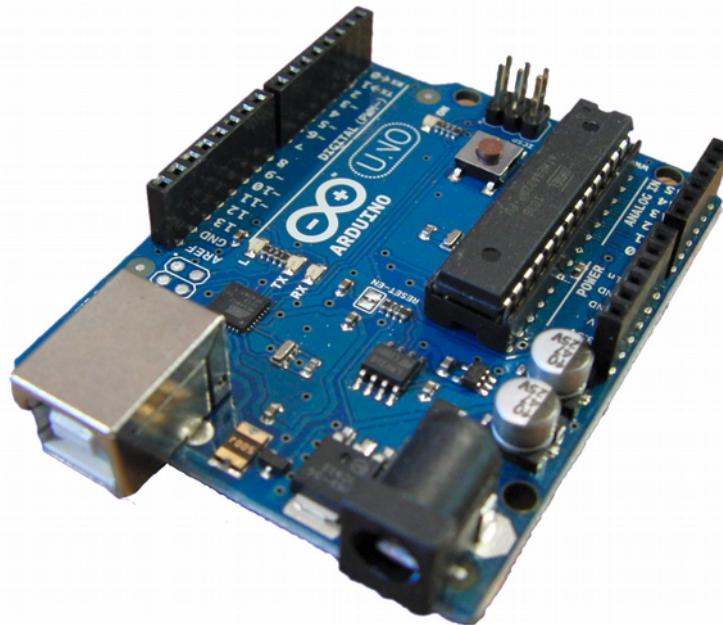
Analogique : tensions arbitraires

capteurs, filtres, traitement
des ondes/signaux (radio, musique, ...)

Numérique : deux états, 0 = “0V” ou 1 = “5V”

interrupteurs, calcul binaire,
transmission de données, ...

Arduino, qu'est-ce que c'est



C'est

- un circuit imprimé (ou carte électronique)
- un outil de prototypage
- ce qui contient l'«intelligence» dans votre montage

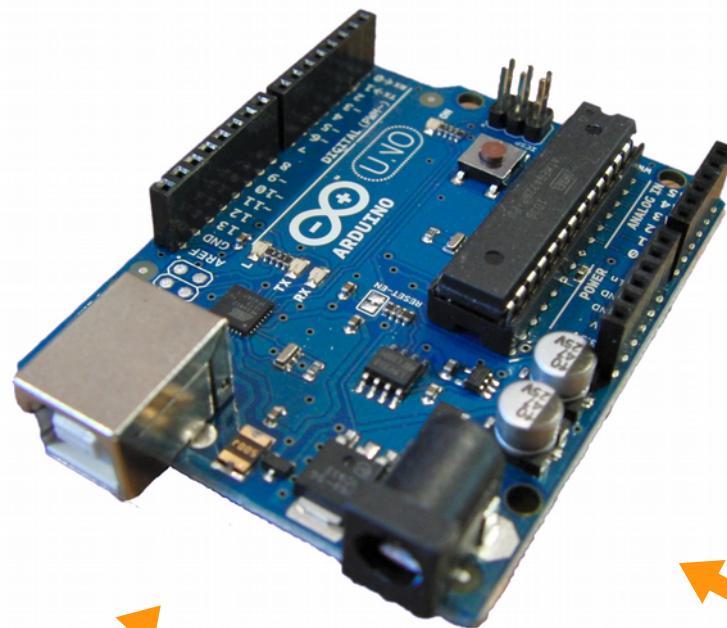


Ça n'est pas un ordinateur (mais ça y ressemble sur le principe)

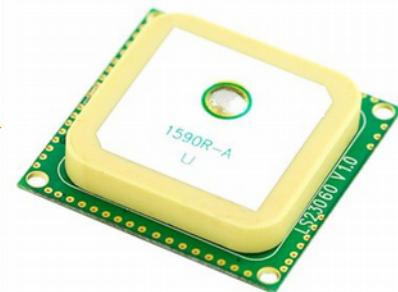
Mettre en relation des choses...



Afficheurs



Recept. GPS



Moteurs



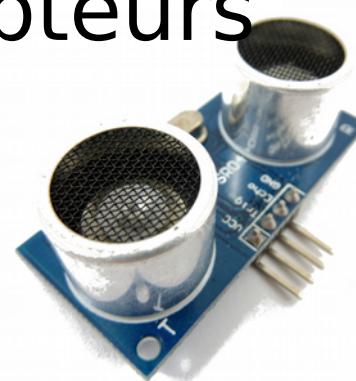
Ordinateur



Contrôles



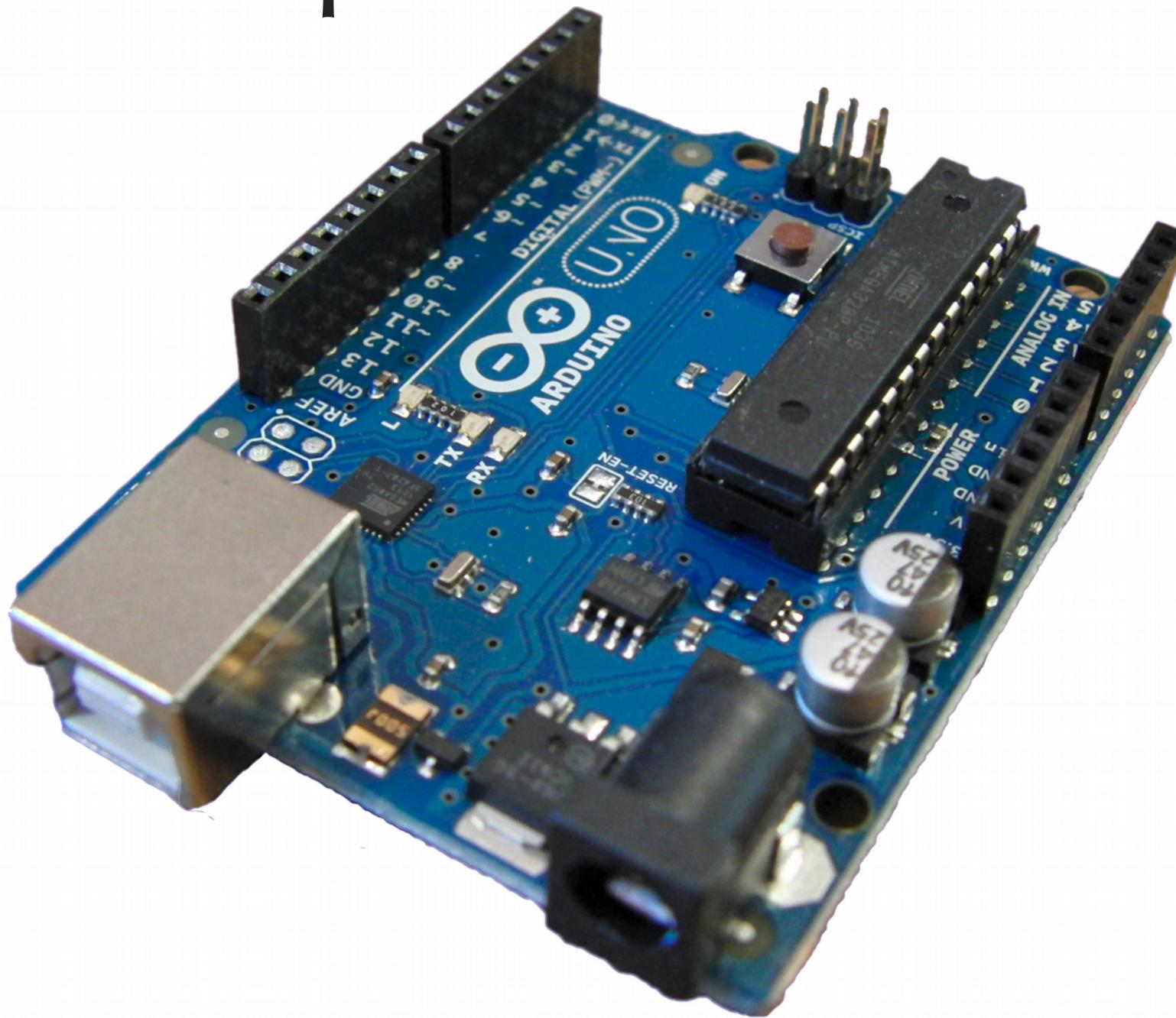
Capteurs



Pourquoi Arduino et pas autre chose ?

- Simple, flexible
- Peu cher [25-30€]
- Libre / open, communauté
- Produits compatibles (« shields », ..)

Comprendre la carte



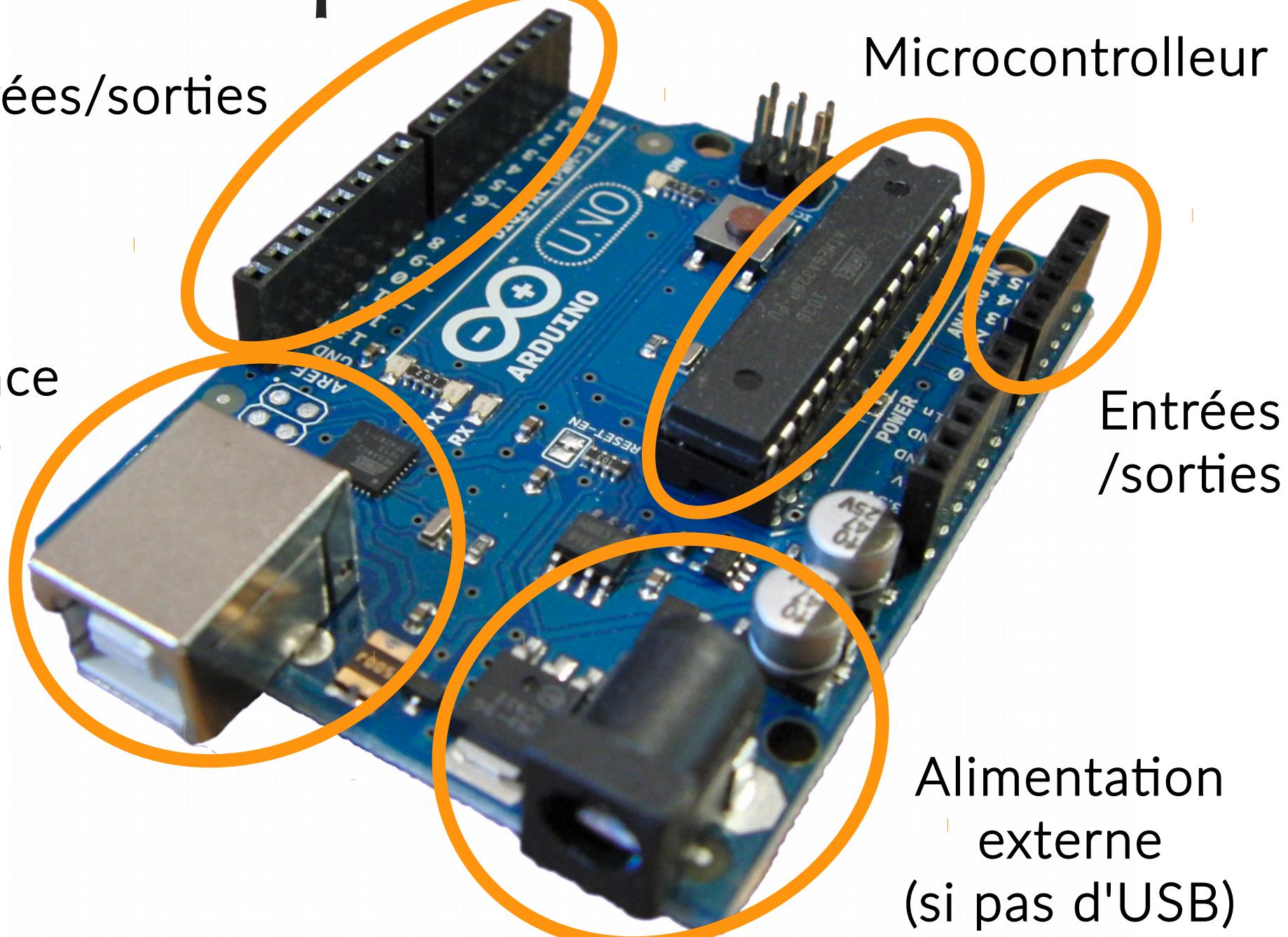
Comprendre la carte

Entrées/sorties

Microcontrôleur

Interface
USB

Entrées
/sorties

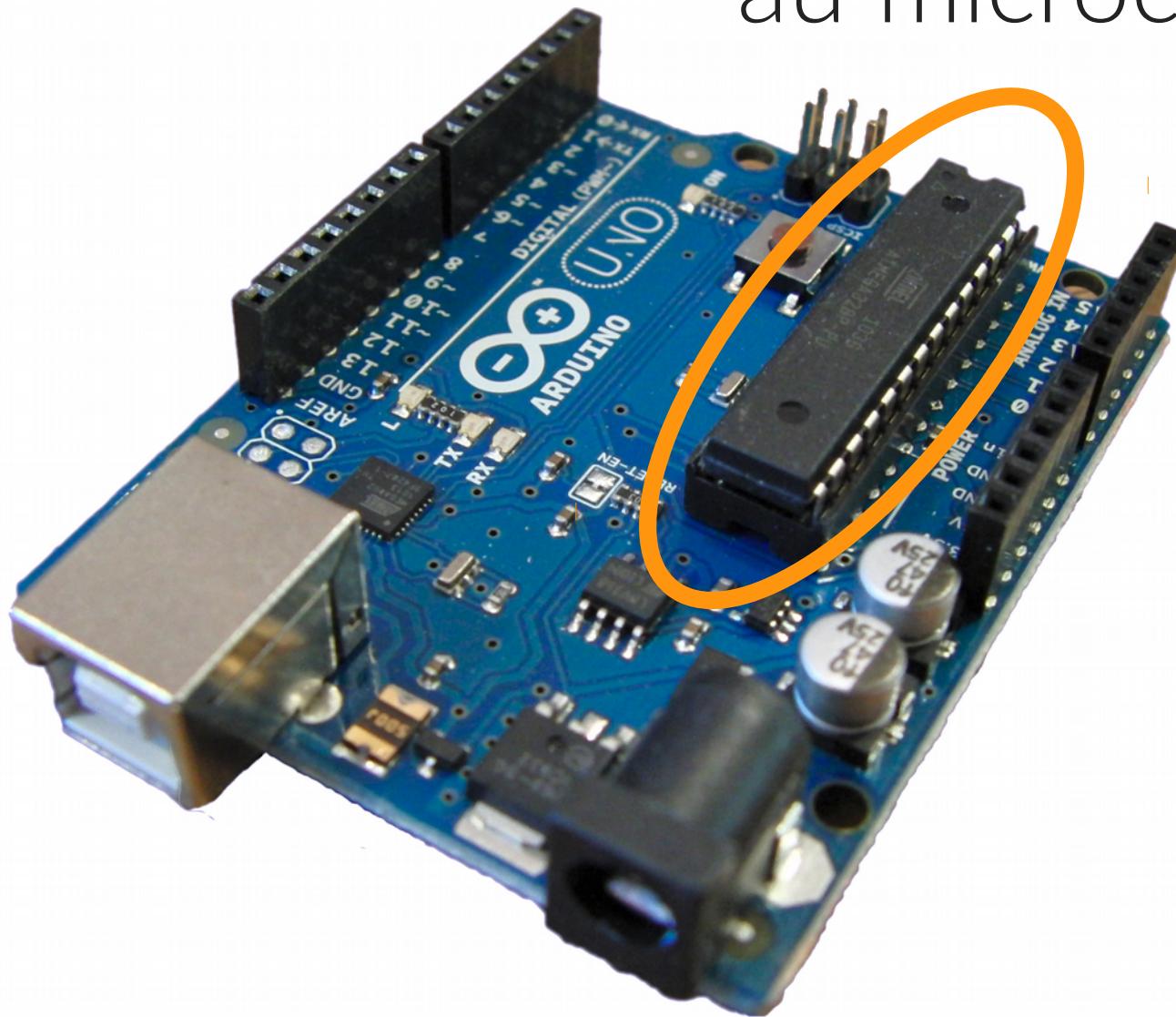


Alimentation
externe
(si pas d'USB)

Mettre en relation,
mais comment ?

Où est l'«intelligence» ?

Ici ! Dans le programme
que vous fournissez
au microcontrôleur !



La programmation

L'art d'expliquer à la machine
ce qu'elle doit faire, sans ambiguïté !

Se fait via **un language compréhensible
à la fois par la machine et par l'humain**

Une langue ... excepté que votre interlocuteur ne
peut pas comprendre si il y a **la moindre erreur
de grammaire** !

Programmer un Arduino



The screenshot shows the Arduino IDE interface with the title bar "Blink | Arduino 2:1.0.5+dfsg2-4". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar contains icons for Save, Run, Open, Upload, and Download. The sketch name "Blink" is selected in the dropdown menu. The code editor displays the following C++ code:

```
// Ceci est un commentaire
// Description en français de
// ce que fait votre programme...
//

void setup()
{
    premiereInstruction();
    deuxiemeInstruction();
    ...
}

void loop()
{
    premiereInstruction();
    deuxiemeInstruction();
    ...
}
```

The status bar at the bottom indicates "Arduino Uno on /dev/ttyACM0".

Programmer un Arduino

```
Blink | Arduino 2:1.0.5+dfsg2-4
File Edit Sketch Tools Help
Blink
// Ceci est un commentaire
// Description en français de
// ce que fait votre programme...
void setup()
{
    premiereInstruction();
    deuxiemeInstruction();
    ...
}
void loop()
{
    premiereInstruction();
    deuxiemeInstruction();
    ...
}
```

Arduino Uno on /dev/ttyACM0

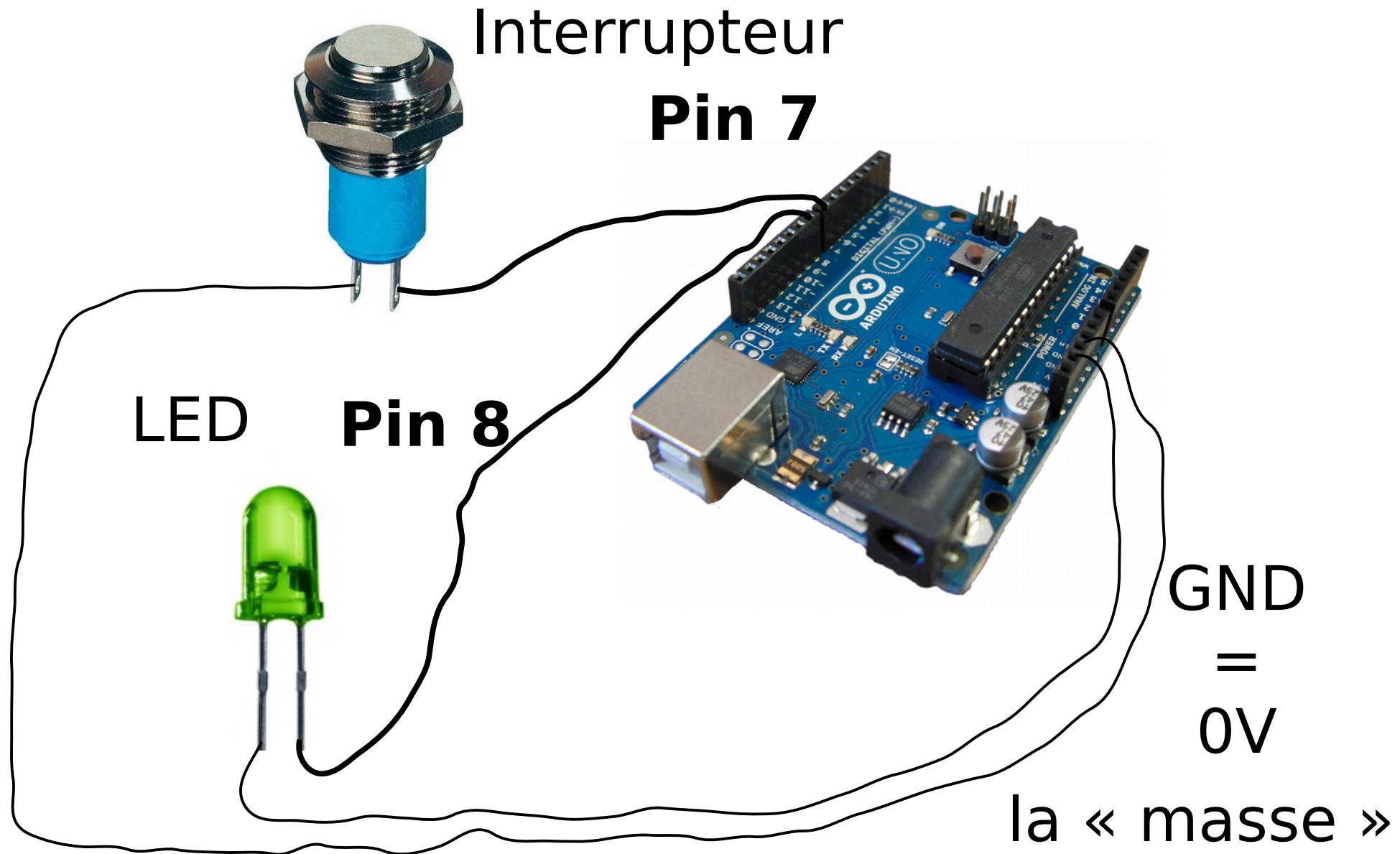
**Boutons pour vérifier
le programme et
l'envoyer sur l'Arduino**

**Commentaires qui
décrivent ce que fait le
programme**

**Choses à faire lorsque
l'Arduino s'allume**

**Choses à faire en
boucle par la suite...**

Programmer un Arduino



Programmer un Arduino

```
void setup()
{
    pinMode(7, INPUT_PULLUP);
    pinMode(8, OUTPUT);
}

void loop()
{
    int etat = digitalRead(7);

    if (etat == LOW)
    { digitalWrite(8,HIGH); }

    else
    { digitalWrite(8,LOW); }
}
```

Lorsque l'Arduino s'allume :

- **Considérer la pin n°7 comme une entrée** (le bouton)
- **Considérer la pin n°8 comme une sortie** (la LED)

Ensuite, faire en boucle :

- **Verifier l'état du bouton** (pin 7)
 - **Si l'état est 1, allumer la LED** (pin 8 à 1)
 - **Sinon, éteindre la LED** (pin 8 à 0)

Programmer un Arduino

```
#define PIN_BOUTON 7
#define PIN_LED     8

void setup()
{
    pinMode(PIN_BOUTON, INPUT_PULLUP);
    pinMode(PIN_LED,    OUTPUT);
}

void loop()
{
    int etat = digitalRead(PIN_BOUTON);

    if (etat == HIGH)
    { digitalWrite(PIN_LED,HIGH); }

    else
    { digitalWrite(PIN_LED,LOW); }
}
```

Programmer un Arduino

```
#define PIN_BOUTON 7
#define PIN_LED     8

void setup()
{
    pinMode(PIN_BOUTON, INPUT_PULLUP );
    pinMode(PIN_LED,      OUTPUT);
}

void loop()
{
    int etat = digitalRead(PIN_BOUTON);

    if (etat == LOW) allumerLED();
    else              eteindreLED();
}

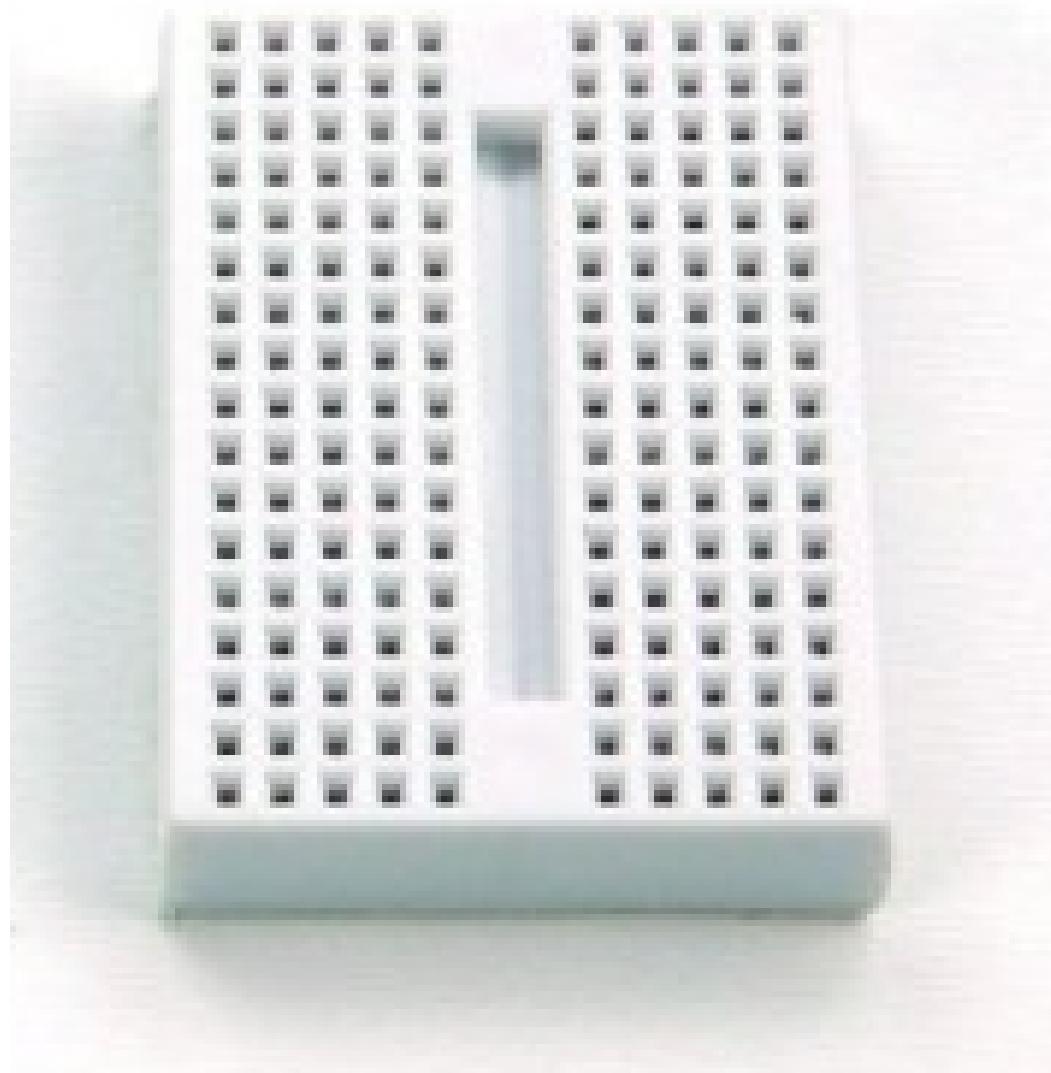
void allumerLED() { digitalWrite(PIN_LED,HIGH); }

void eteindreLED() { digitalWrite(PIN_LED,LOW); }
```

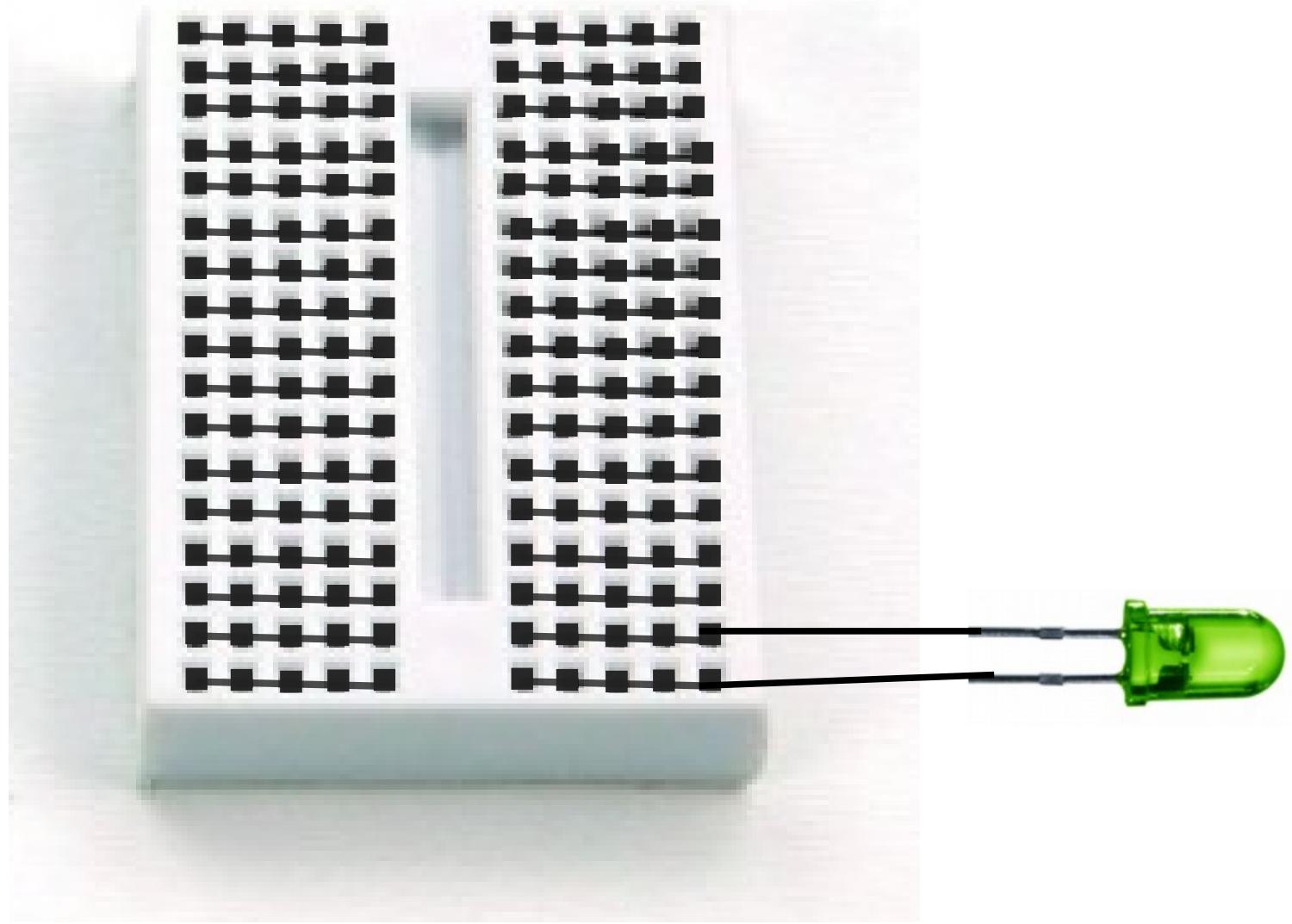
[2]

Votre premier montage Arduino

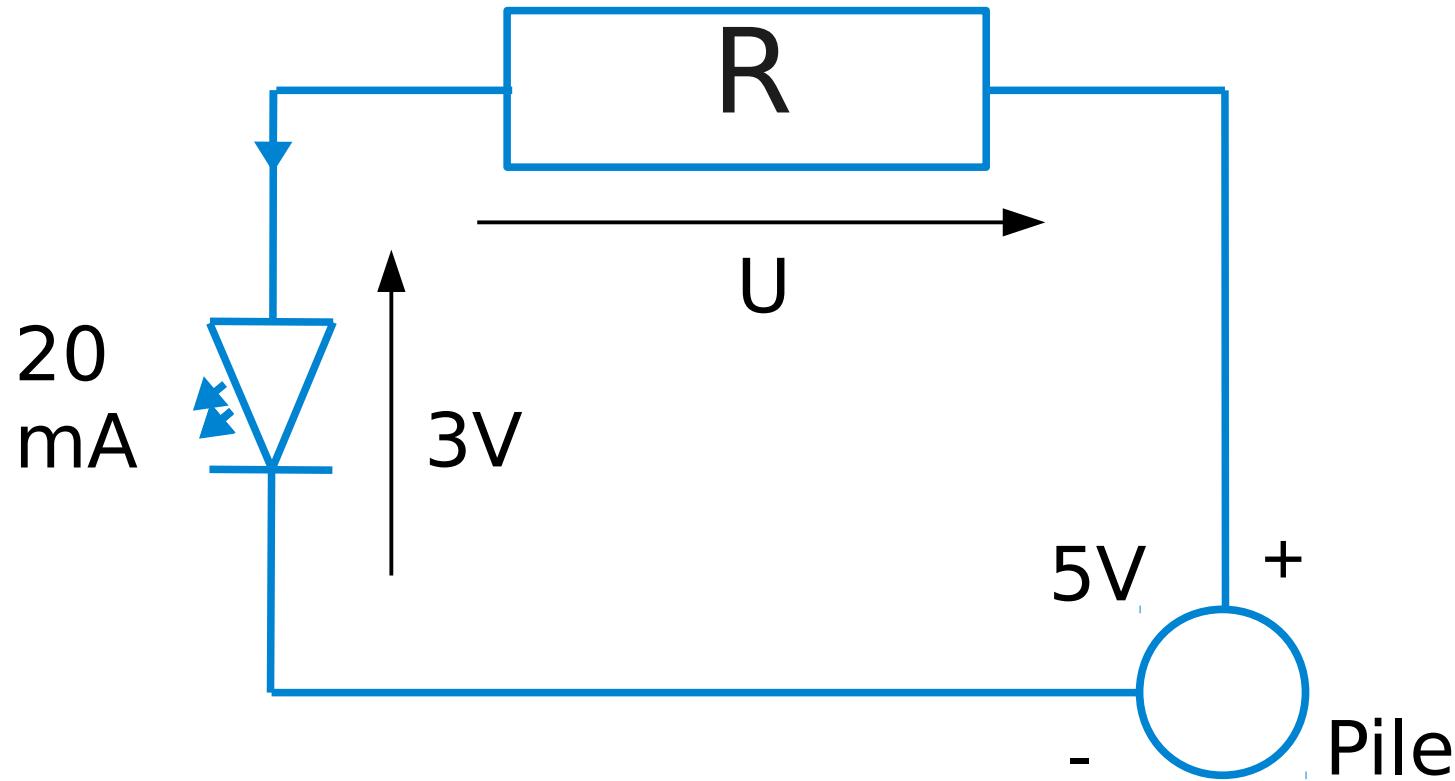
Le mini-breadboard



Le mini-breadboard



Protéger une LED



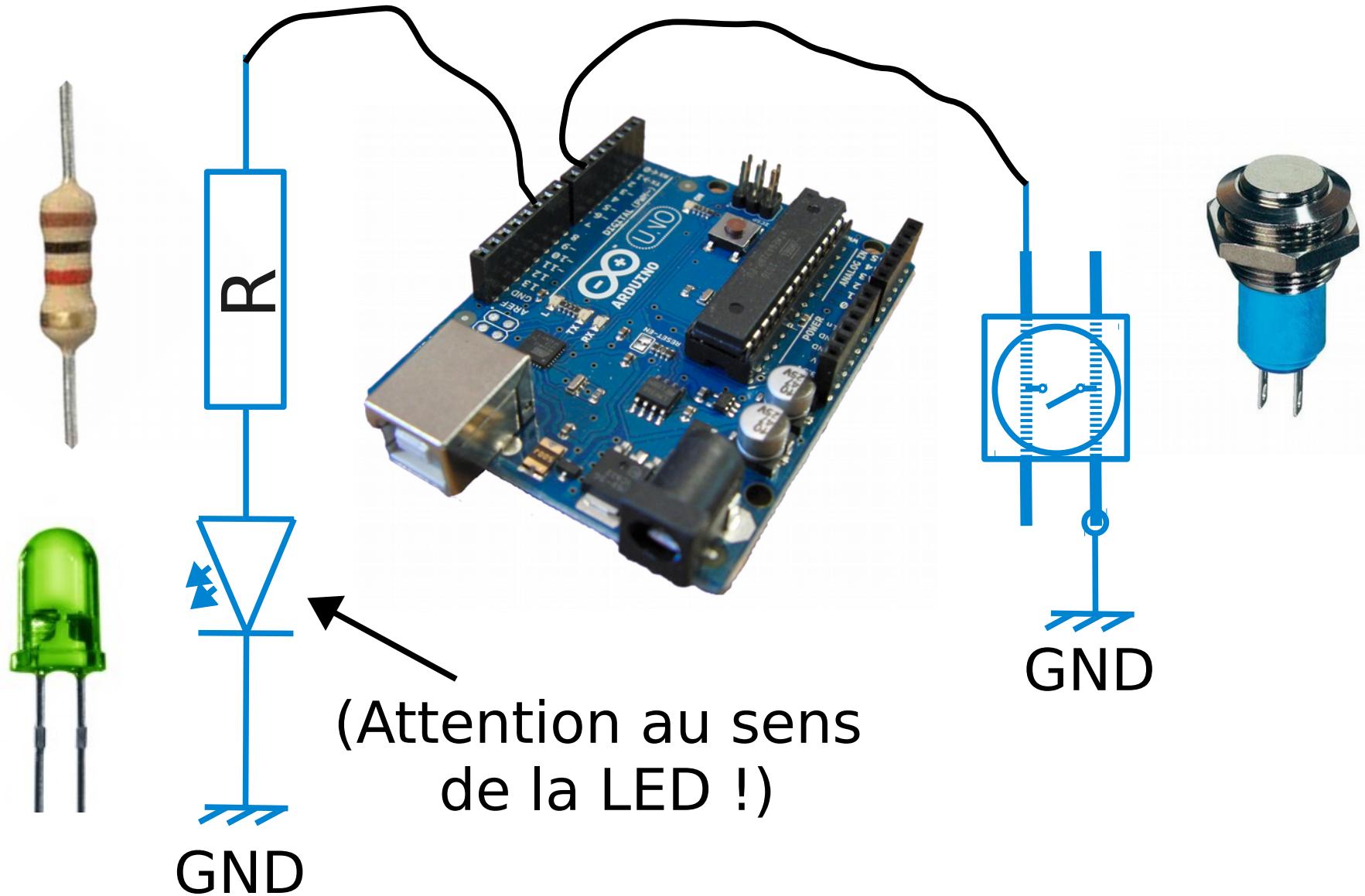
$$5V = 3V + U$$

$$\rightarrow U = 2V$$

$$\text{Loi d'Ohm : } U = Ri$$

$$\rightarrow R = U / i = 2V / 20 \text{ mA} = 100\Omega$$

Le montage



Le programme

```
#define PIN_BOUTON 7
#define PIN_LED     8

void setup()
{
    pinMode(PIN_BOUTON, INPUT_PULLUP );
    pinMode(PIN_LED,      OUTPUT);
}

void loop()
{
    int etat = digitalRead(PIN_BOUTON);

    if (etat == LOW) allumerLED();
    else              eteindreLED();
}

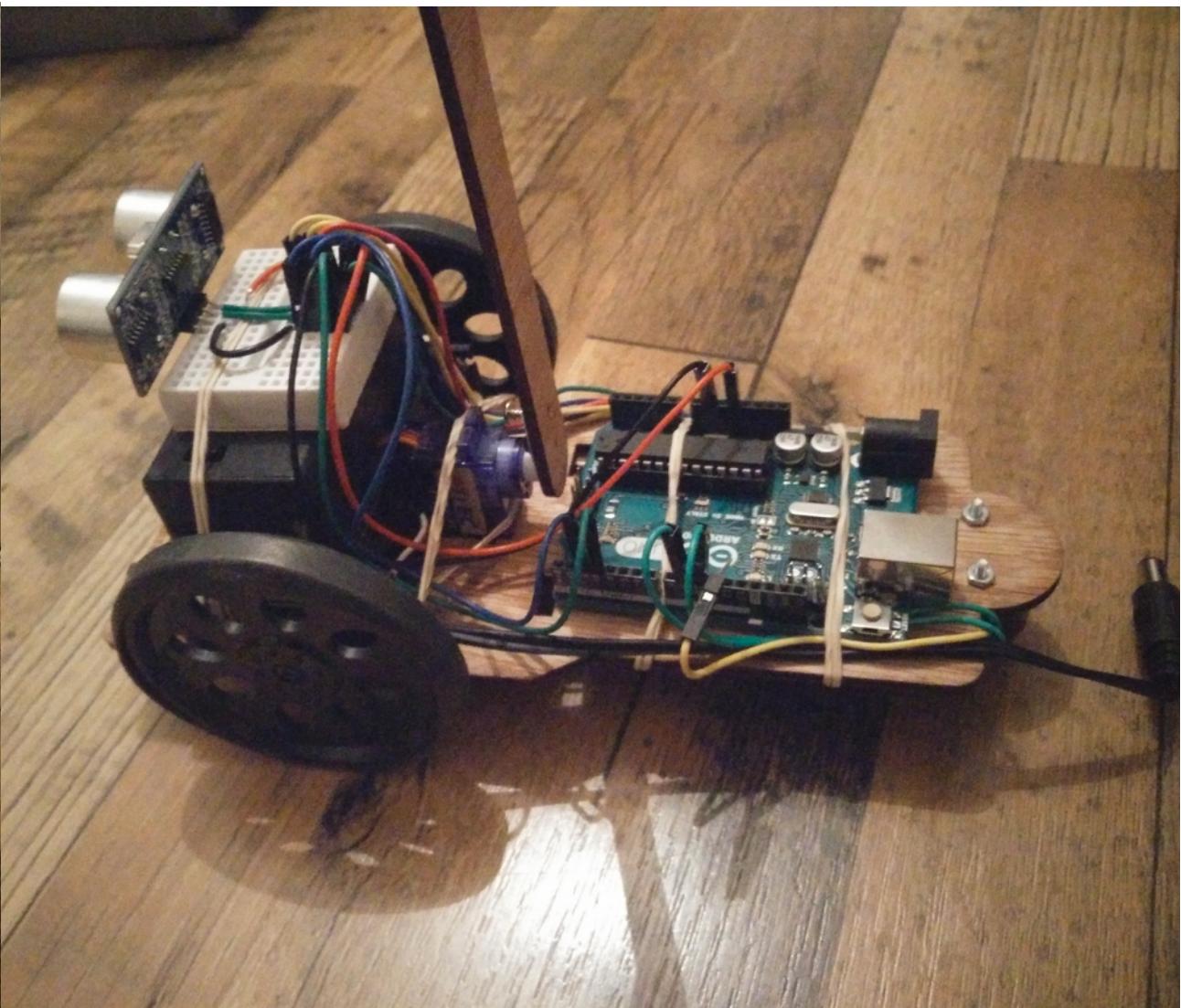
void allumerLED() { digitalWrite(PIN_LED,HIGH); }

void eteindreLED() { digitalWrite(PIN_LED,LOW); }
```

À vous ! 😊

[3] Construction du robot

Les etapes de construction

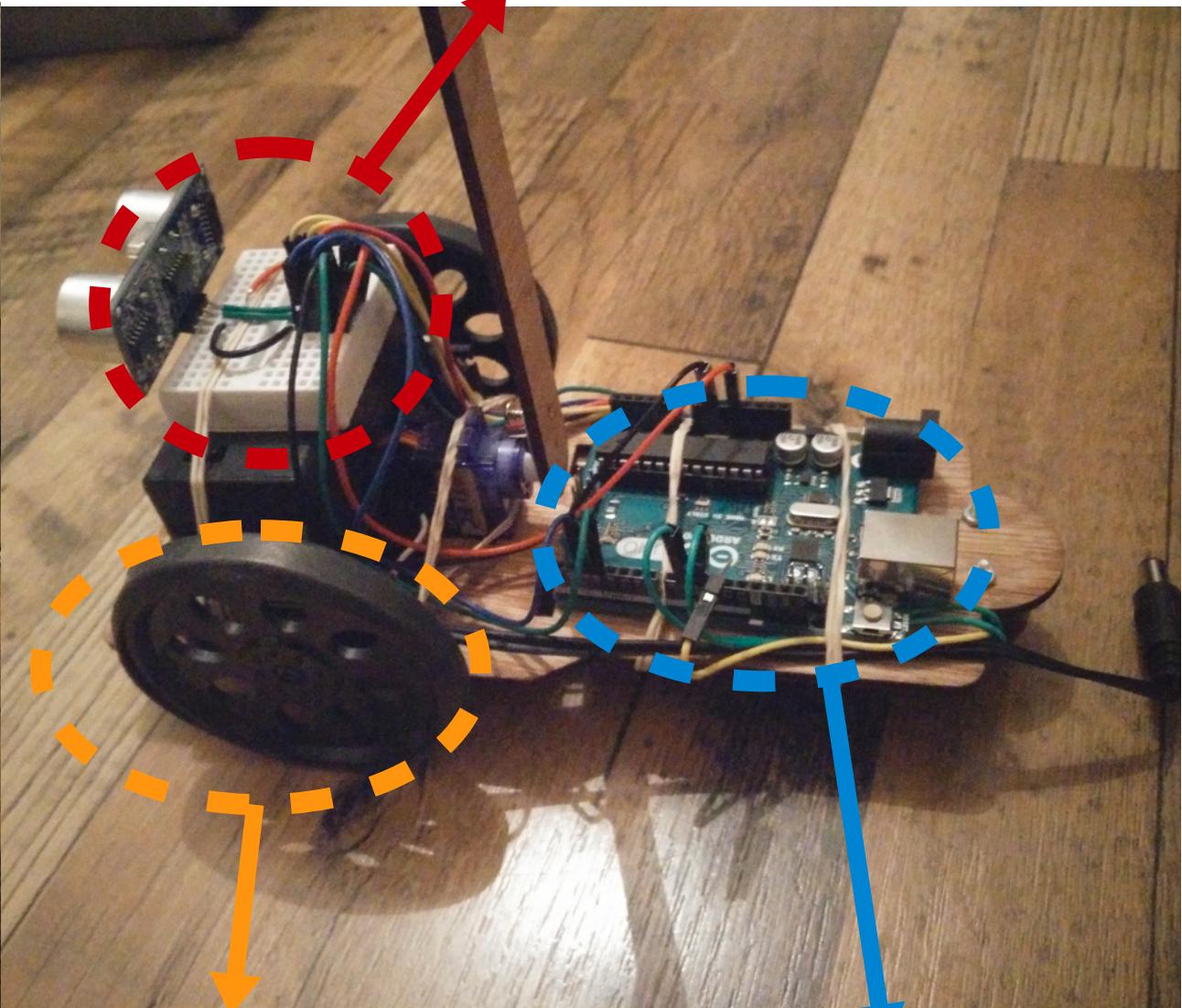


Les etapes de construction

Mini breadboard



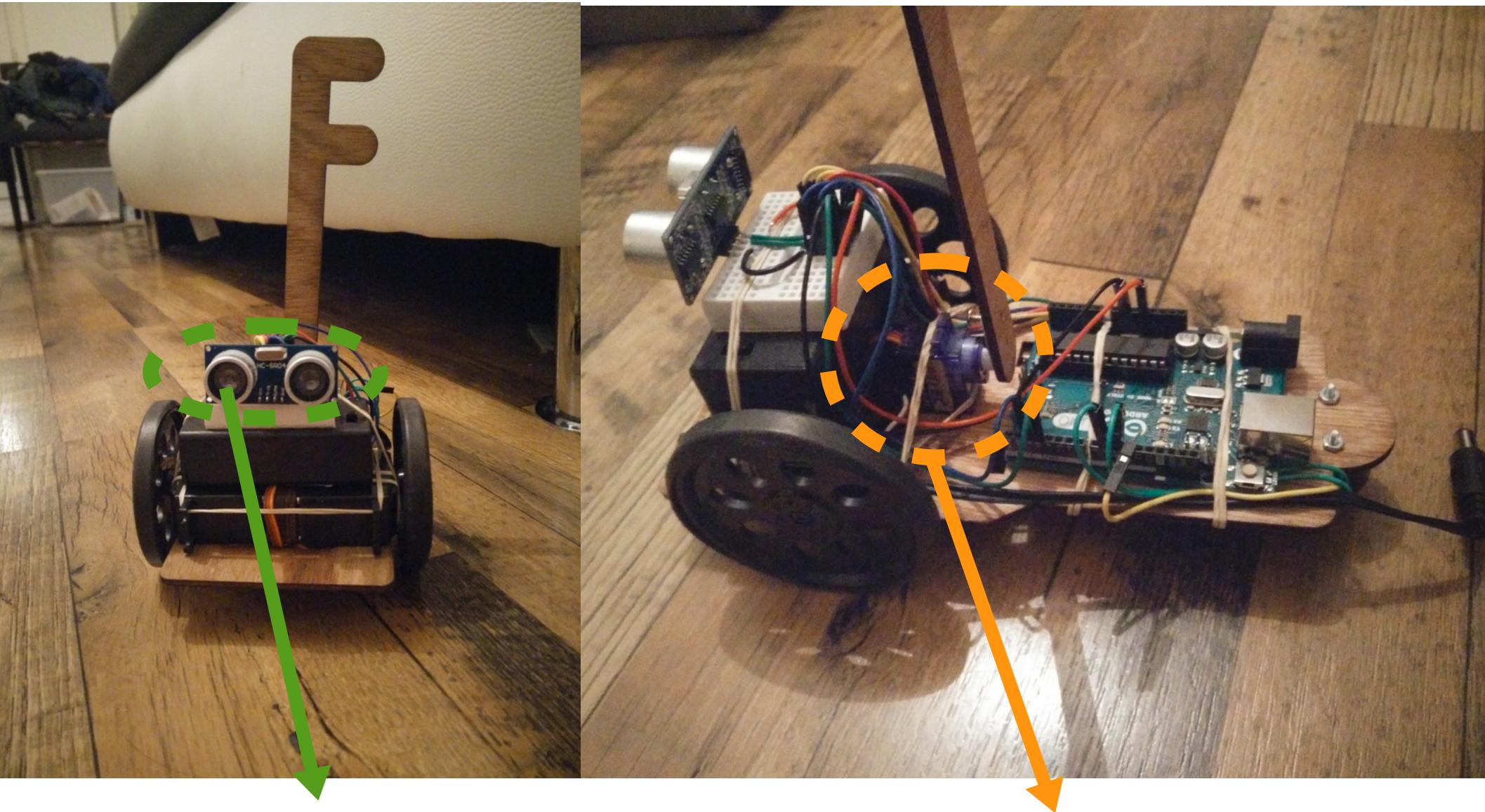
Pile 9V Moteurs



Roue

Arduino

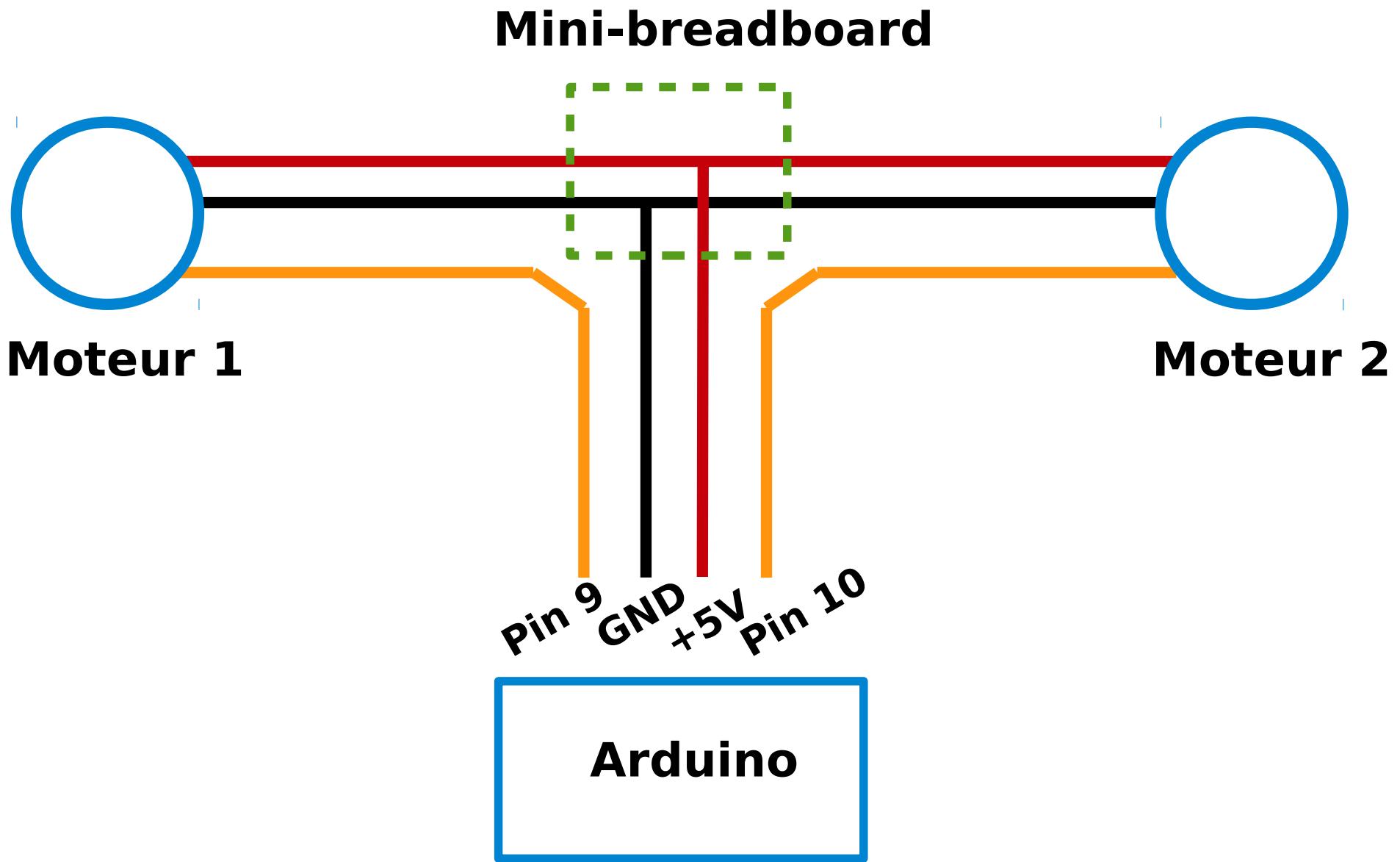
Les etapes de construction



Capteur ultrason

Servomoteur

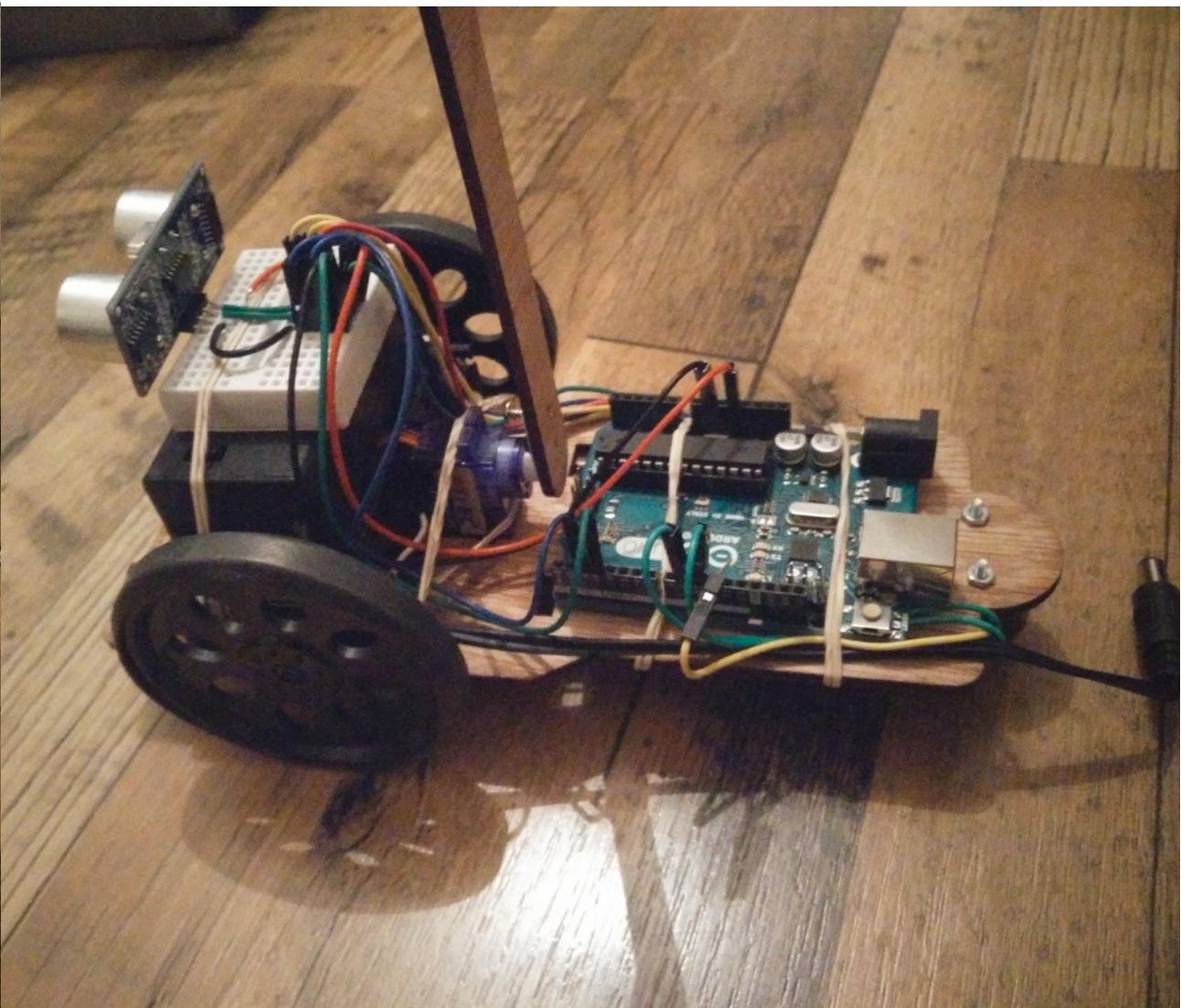
Les etapes de construction



À vous ! 😊

[4]
Programmation d'un robot

Le robot



Le programme

```
// ^ Tout plein de choses qu'il n'est pas nécessaire de comprendre ^

void setup()
{
}

void loop()
{
    avancer(1000);

    delay(1000);

    tourner(0.25);

    delay(1000);
}
```

Faire en boucle :

- **Avancer d'un metre**
- **Attendre 1 seconde**
- **Tourner d'un quart de tour (0.25)**
- **Attendre 1 seconde**

Le circuit

