TP DECOUVERTE ARDUINO

Le **système Arduino** est un outil pour fabriquer de petits **ordinateurs** qui peuvent **capter** et **contrôler** davantage de choses du monde matériel que votre ordinateur de bureau. C'est une **plateforme open-source** d'électronique programmée qui est basée sur une simple carte à **microcontrôleur** et un **logiciel**, véritable environnement de développement intégré, pour **écrire, compiler et transférer** le programme vers la carte à microcontrôleur.

Arduino peut être utilisé pour **développer** des **objets interactifs**, pouvant recevoir des **entrées** d'une grande variété **d'interrupteurs** ou de **capteurs**, et pouvant contrôler une grande variété de **lumières**, **moteurs** ou toutes autres **sorties matérielles**. Les projets Arduino peuvent être **autonomes**, ou bien ils peuvent **communiquer** avec des logiciels tournant sur votre ordinateur (tels que Flash, Processing ou MaxMSP). Les **cartes électroniques** peuvent être **fabriquées manuellement** ou bien être **achetées préassemblées** ; le logiciel de **développement open-source** peut être téléchargé gratuitement.

OBJECTIFS :

1. **Savoir** utiliser le système de gestion de la carte Arduino.
2. **Savoir** utiliser l'environnement de développement intégré pour créer des programmes exécutables sur la carte Arduino.
3. **Découvrir** les bases du langage de programmation Arduino.

ÉQUIPEMENT :

- **Une carte Arduino Uno**
- **Le matériel à câbler**

TRAVAIL A REALISER :

REALISER les **3 activités** proposées

1 LE PROGRAMME MINIMUM

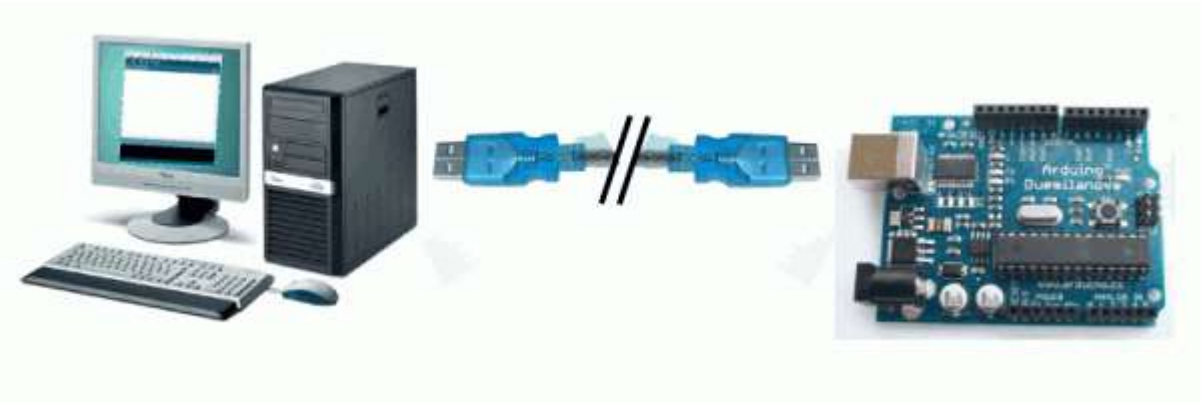
1.1 DESCRIPTION

Cet exemple contient le **code minimum** qui est **nécessaire** et **indispensable** pour un **programme Arduino** compilable. **Le code minimum** doit comporter :

- la fonction **setup()** qui est exécutée une fois au début du programme.
- et la fonction **loop()** qui est exécutée en boucle tant que la carte est sous tension.

Une règle essentielle : les deux fonctions **setup()** et **loop()** (même vides) sont **OBLIGATOIRES** dans tout programme Arduino .

1.2 LE CIRCUIT A REALISER



Une carte Arduino connectée par un **câble USB** à un **ordinateur** avec **le logiciel Arduino** installé.

1.3 LE CODE DU PROGRAMME

```
    void setup()
    {
    // début de la fonction setup () : est exécutée une fois
      // mettre ici les instructions de votre programme à exécuter une seule
fois au démarrage
    } // fin de la fonction setup() : le programme exécute ensuite la fonction
loop

    void loop()
    {
    // début de la fonction loop() : est exécutée en boucle sans fin
      // mettre ici les instructions de votre programme qui seront exécutées en
boucle
    } // fin de la fonction loop : le programme reprend au début de loop
```

1.4 EXPLICATIONS DU PROGRAMME

La fonction **setup()** est la **première fonction** qui est appelée quand un programme démarre. Il faut utiliser cette fonction pour notamment :

- **initialiser les variables,**
- **initialiser le mode de fonctionnement** (entrée/sortie) des broches,
- **démarrer les librairies** utilisées, etc.....

La fonction **setup()** sera exécutée **seulement une fois**, après la mise sous tension ou **une réinitialisation** (par appui sur le bouton 'reset') de la carte Arduino.

Après la fonction setup, la **fonction loop** (loop = boucle en anglais) va faire exactement ce que son nom suggère : elle va **s'exécuter en boucle**, sans fin, permettant au programme **d'exécuter des instructions** et de réagir durant son exécution. C'est le code placé dans la **fonction loop** qui est activement utilisé pour **contrôler la carte Arduino** : on peut dire que c'est le "**cœur**" de votre programme.

Le code que vous allez tester ne **fera** absolument **rien**... mais sa structure vous sera très utile pour réaliser un **copier/coller** avant de démarrer un programme.

Ce programme minimal vous montre également comment mettre des commentaires dans votre code. Toute ligne qui commence par **deux slashs //** ne sera pas lue par le compilateur : vous pouvez donc écrire à ce niveau ce que vous voulez. **Commenter** votre code de cette façon sera particulièrement utile **pour expliquer** à vous-mêmes ou aux autres **votre programme pas à pas**.

Toute fonction utilisée dans un programme **doit être déclarée** en définissant :

- **son type**, défini par un mot clé mis avant le nom de la fonction,
- **les paramètres qu'elle reçoit**, mis dans les ( ) après le nom de la fonction.

Le code d'une fonction est encadré par **une { de début** et **une } de fin** : ce sont les limites du code de la fonction. Bon à savoir également : **chaque ligne de code active** comportant une instruction (sauf quelques cas particuliers) doit se terminer **par un ;**

Le type de la fonction correspond au type de valeur (byte, int, float, etc...) qu'elle renvoie après son exécution. Dans le cas le plus simple, **lorsqu'une fonction ne renvoie aucune valeur**, elle est de **type void** : on utilisera donc ce mot clé pour déclarer une fonction qui ne renvoie rien.

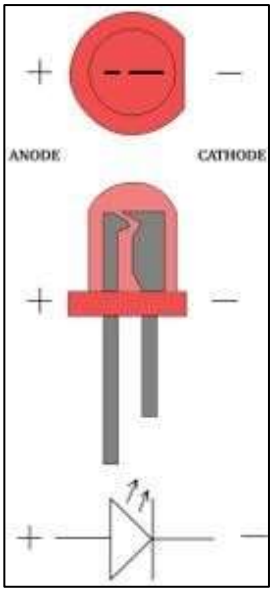
Si **une fonction utilise des paramètres**, ils doivent être **définis entre les ( )**. Les paramètres reçus seront utilisés par la fonction pendant son exécution. **Si une fonction n'utilise aucun paramètre**, on laissera **les ( ) vides** tout simplement.

Les fonctions setup() et loop() obligatoires dans tout programme Arduino sont des fonctions de la forme la plus simple : elles ne renvoient aucune valeur et elles n'utilisent aucun paramètre. On utilisera donc le mot clé void pour les déclarer et on laissera les parenthèses vides.

2 **ACTIVITE** : FAIRE CLIGNOTER UNE LED :

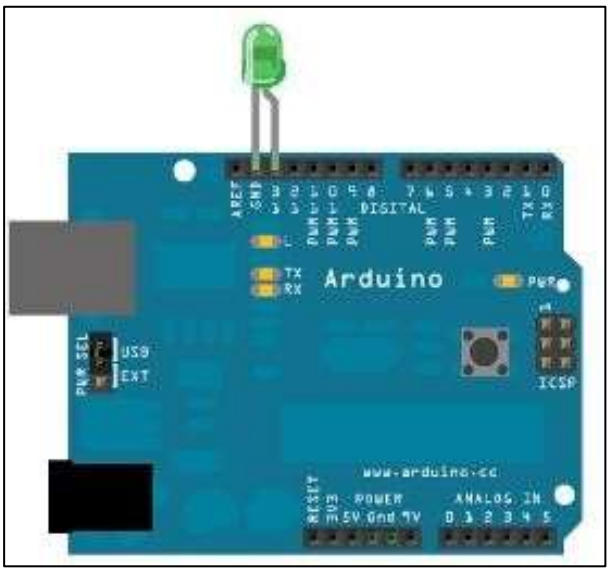
2.1 MATERIEL REQUIS

- **Une carte Arduino Uno**
- **Une LED standard**



2.2 LE CIRCUIT A REALISER

- **Prendre une LED** et connecter sa **patte longue** (patte positive appelée anode) à **la broche 13** de la carte Arduino,
- **Connecter la patte courte** (patte négative, appelée cathode) **à la broche de masse de la carte Arduino** (notée GND pour GROUND en anglais, cette broche correspond au 0V).



- Ensuite**, connecter la carte Arduino par un câble USB à votre ordinateur** sur lequel est installé le logiciel Arduino.

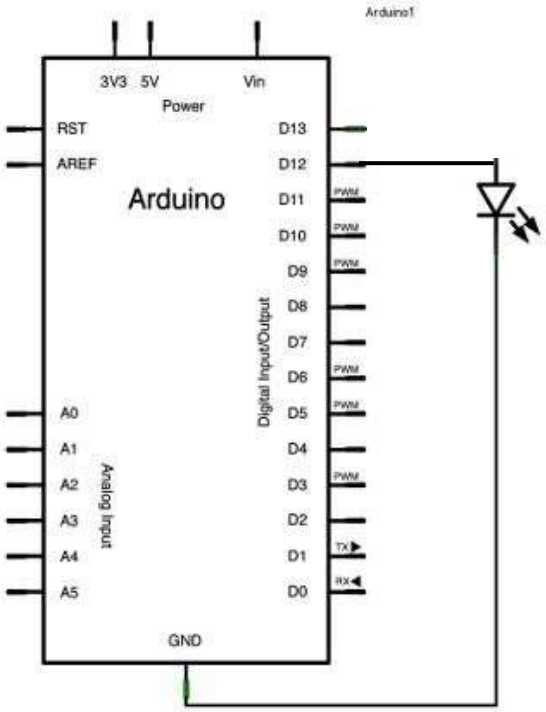
2.3 MISE EN ŒUVRE DU PROGRAMME

- Lancer le **logiciel Arduino,**
- **Copier/coller le code** du document ressource « **Programme LED** ».
- **Programmer votre carte** en suivant la procédure disponible dans le « **Dossier ressources** » au paragraphe :
  « **7 : *Le Logiciel Arduino : Environnement de développement Intégré (EDI)*** »

La LED connectée sur la broche 13 doit se mettre à clignoter …

2.4 SCHEMA THEORIQUE DU MONTAGE

Voici le schéma théorique du montage :



Vous noterez que la **LED n'a pas de résistance en série**. Ceci car l'**intensité disponible sur cette broche n°12 en sortie** est assez **basse** pour **ne pas endommager la LED**. Ceci simplifie le montage. Cependant, **d'une manière générale**, il faut ajouter une **résistance en série avec la LED afin de limiter l'intensité** utilisée sur la broche en sortie.

- Les **broches de la carte Arduino** peuvent fournir **jusqu'à 40 mA** en **entrée** comme **en sortie**. Cependant, il ne faut pas oublier que **l'intensité maximale** disponible pour **l'ensemble des broches** en sortie est de **200mA** pour une carte UNO.

- Ainsi en pratique, **il est judicieux de limiter l'intensité utilisée à 15 mA** par broche en moyenne. Pour mémoire, la chute de tension aux bornes d'une LED est de 1,5V environ. On pourra donc, avec une LED, utiliser une résistance en série de 220 Ohms (R = U/I = (5V - 1,5V) / 0,012 = 233 Ohms).

2.5 EXPLICATIONS DU PROGRAMME :

2.5.1 *Au niveau de la fonction setup ()*

Dans le programme on commence par **initialiser la broche 12** en sortie avec l'instruction **pinMode()** selon :

- ***pinMode(13, OUTPUT);***

2.5.2 *Au niveau de la fonction loop()*

Dans la fonction **loop**, "**coeur**" du programme, on commence par **allumer la LED** à l'aide de l'instruction **digitalWrite() :**

- ***digitalWrite(12, HIGH);***

Ceci met la **broche 12** au niveau **HAUT**, soit **+5V**. Ceci crée une **différence de potentiel entre les 2 broches** de la LED et elle s'allume donc.

Ensuite, on **éteint la LED** toujours à l'aide de l'instruction **digitalWrite() :**

- ***digitalWrite(12, LOW);***

Ceci met la **broche 12** au niveau **BAS**, soit **0V** : la LED s'éteint.

Entre l'allumage et l'extinction de la LED, il faut prévoir **un délai assez long** pour que l'œil humain puisse percevoir le changement. Pour cela, on utilise **l'instruction delay()** pour dire à la carte Arduino de ne rien faire pendant **1000 millisecondes**, c'est à dire une seconde :

- **delay(1000);**

Si vous ne mettez pas d'instruction **delay()** entre la mise au **niveau HAUT puis BAS** de la broche 12, la **LED va clignoter très rapidement**... et elle apparaîtra toujours allumée. Si on connecte un **oscilloscope** sur la broche, on pourra voir cependant que la broche change d'état **HAUT/BAS** plusieurs millions de fois par seconde.

2.6 ACTIVITES COMPLEMENTAIRES :

MONTER la LED sur la sortie 11 et **MODIFIER** le programme en conséquence pour allumer la led.

PROPOSER un programme permettant de réaliser un **SOS** en code **Morse** avec la **LED**.