Arduino
Alimentation de LEDs

TP N°1	6h	Nom : CORRECTION

Objectif :	**durée** :
Aujourd'hui l'informatique et l'électronique sont indissociables, comme dans les objets connectés (IoT). Un informaticien doit donc avoir des connaissances de base en électronique et en électricité. Il doit savoir utiliser un logiciel de simulation électronique comme Isis Proteus, un logiciel de programmation en langage C et appréhender leurs fonctionnements.	3+3h (modulables)

Matériel :

Carte Arduino Uno – Plaque LAB – résistances – DEL de plusieurs couleurs
Ordinateur connecté – Logiciel Isis Proteus – Logiciel Arduino.

Compétences :

C1 RECHERCHER ET EXPLOITER DES DOCUMENTS ET INFORMATIONS, AFIN DE CONTRIBUER À L'ÉLABORATION D'UN PROJET D'ÉQUIPEMENT OU D'INSTALLATION D'UN SYSTÈME :

- C1-1 Appréhender la mise en œuvre d'un projet simulé ou réel d'installation d'un système

C2 S'APPROPRIER LES CARACTÉRISTIQUES FONCTIONNELLES D'UN SYSTÈME, EN VUE D'INTERVENIR DANS LE CADRE D'UNE ÉVOLUTION OU D'UNE OPÉRATION DE MAINTENANCE :

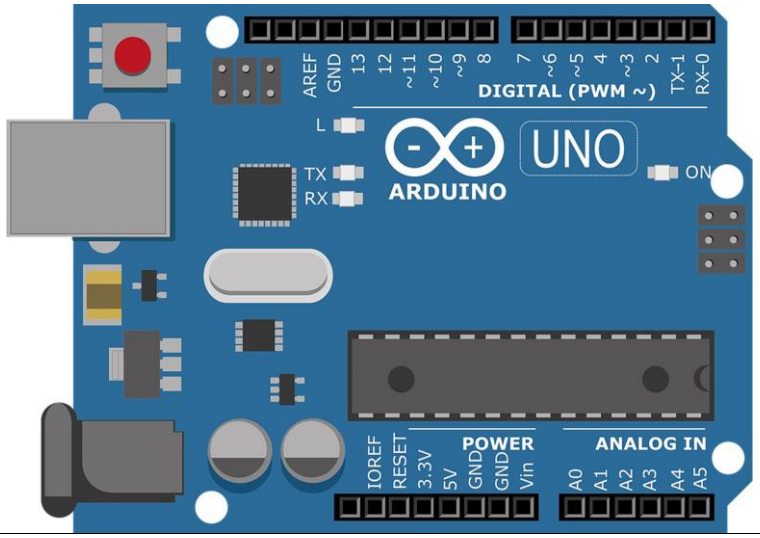
- C2-2 Analyser le fonctionnement de l'installation actuelle ou de l'équipement en vue de l'intervention

C3 PRÉPARER LES ÉQUIPEMENTS EN VUE D'UNE INSTALLATION :

- C3-2 Réaliser l'intégration matérielle ou logicielle d'un équipement
- C3-3 Effectuer les tests nécessaires à la validation du fonctionnement des équipements

C4 INSTALLER ET METTRE EN OEUVRE LES ÉQUIPEMENTS :

- C4-2 Repérer les supports de transmission et d'énergie, implanter, câbler, raccorder les appareillages et les équipements d'interconnexion
- C4-3 Effectuer les tests, certifier le support physique
- C4-4 Installer, configurer les éléments du système et vérifier la conformité du fonctionnement

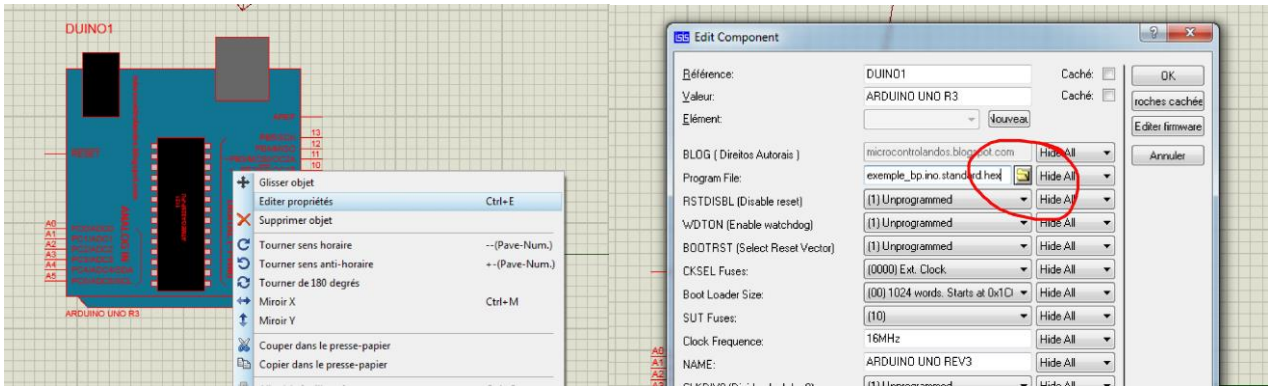


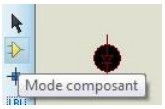
1 Dimensionnement d'une résistance de DEL :

1.1 Téléchargez/copiez le dossier complet sur votre PC.
Utilisez obligatoirement le "clic droit" et sélectionnez "Enregistrer le lien sous…" pour télécharger les fichiers dans votre ordinateur.

1.2 Ouvrez le schéma "TP01 - Arduino Alimentation 1 LED.pdsprj" sous Proteus.

Charger le fichier "TP01_-_Arduino_Alimentation_1_LED.ino.standard.hex" qui se trouve dans le dossier "TP01_-_Arduino_Alimentation_1_LED" dans la carte Arduino avec le "clic droit" sur la carte Arduino puis sur "Editer Propriétés" puis ouvrez le symbole du dossier et chercher le fichier en ".hex", puis chargez-le.



1.3 Mesurez la tension en plaçant le voltmètre entre la borne 3 de la carte Arduino et la masse avec la fonction "Mode Composant" 

puis lancez la simulation en cliquant sur la flèche  en bas à gauche de l'écran.

Notez la valeur de Valim :

1.4 En vous aidant des informations indiquées dans le fichier "dimension del.pdsprj", calculez la résistance de protection de la DEL rouge.

Calcul de RP :

Rp = (Valim - Vf) / Idel = (5 - 1,5) / 0,01 = 350 Ω

1.5 Indiquez la valeur normalisée et le code couleurs de la résistance dans la série E24.

Valeur de RP normalisée : 330 Ω

Code couleur : Orange, orange, marron, or

1.6 Modifiez sous Proteus, la valeur de la résistance RP (valeur normalisée) avec clic droit sur la résistance et "Editez propriétés", changez la valeur dans Résistance, enregistrez et lancez la simulation. Que se passe-t-il ?
La DEL s'allume.

2 Câblage du schéma :

2.1 Câblez sur plaque LAB le schéma de la Partie 1 avec la carte Arduino.
Faire valider le câblage avant la mise sous tension.

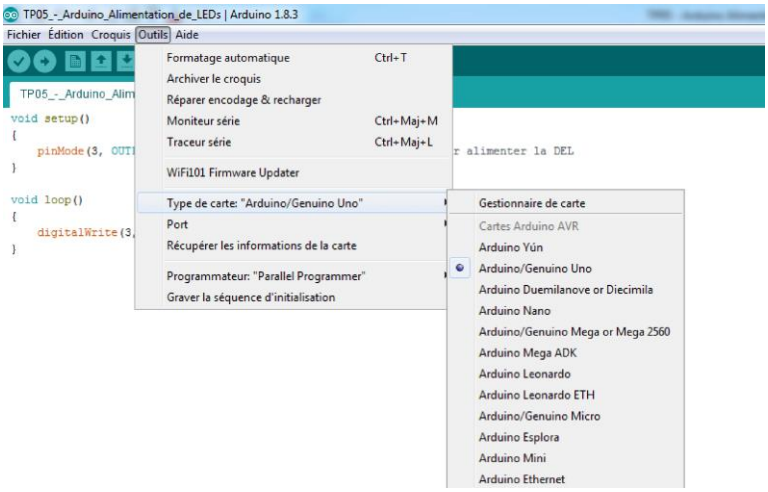
2.2 Afin de faire fonctionner la carte Arduino, il faut charger dans sa mémoire le micro programme qui va allumer la DEL.

a. Brancher la carte via le câble USB sur l'ordinateur, laisser faire l'installation du driver.

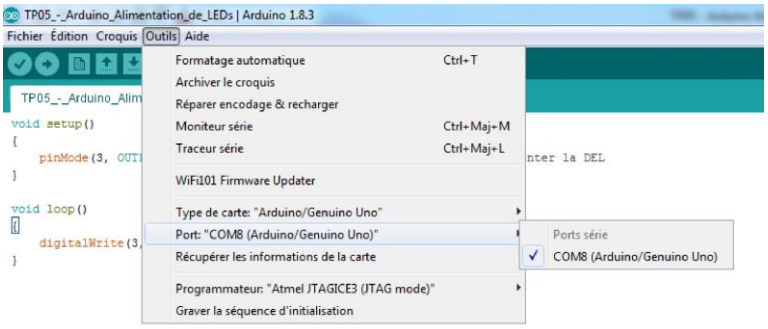
b. Ouvrez le fichier Arduino "TP01_-_Arduino_Alimentation_1_LED.ino" qui se trouve dans le dossier "TP01_-_Arduino_Alimentation_1_LED.
Le logiciel Arduino s'ouvre, suivez les instructions indiquées dans les fenêtres qui s'ouvrent et validez. Au besoin, créer un dossier portant le nom du fichier en ".ino" si ça ne se fait pas automatiquement.

c. Configuration du logiciel :

- Il faut vérifier que la carte sélectionnée soit la bonne, nous utilisons une carte Arduino/Genuino Uno. Il faut aller dans l'onglet "Outils" et chercher le bon type de carte :



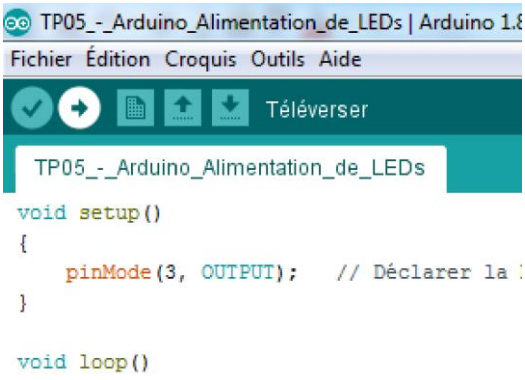
- Puis il faut sélectionner le bon port de communication, appelé "Port" sous l'onglet "Outils".
Pour se faire, il faut choisir le Port COM où Arduino s'affiche entre parenthèse.
Ici dans notre exemple, c'est le port COM8 (attention, ce n'est pas le même sur votre ordinateur). Cliquez dessus pour le sélectionner.



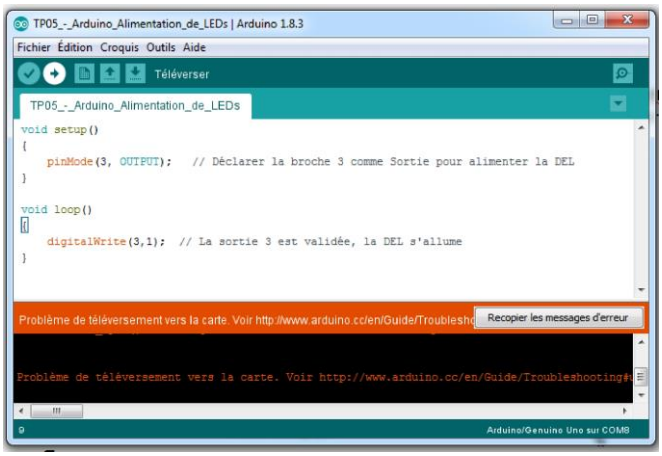
Maintenant, Arduino est configurée, vous pouvez passer à l'étape suivante.

d. Chargement du programme :

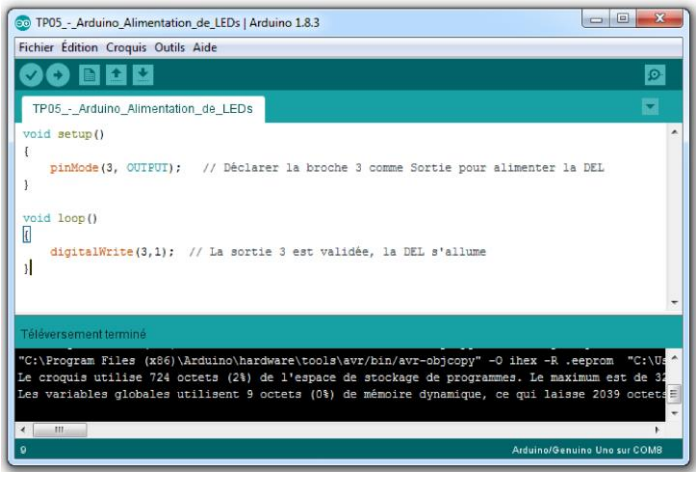
- Pour charger le programme dans la carte, il faut cliquer sur "Téléverser" en haut à gauche de l'écran, ce qui va entrainer la compilation du croquis, ça va prendre un petit moment.



- SI un problème est survenu pendant la compilation (vérification du programme) ou pendant le téléversement, une barre orange s'affiche en bas. Dans ce cas, refaire la partie C, page 3.



- Si tout va bien, le logiciel affiche "Téléversement terminé" dans la barre d'activités et la LED s'allume sur votre montage.



Faire valider.

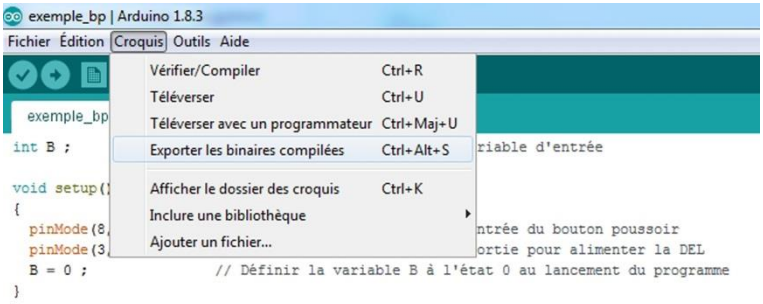
3 Création de programmes Arduino :

3.1. LED Clignotante en simulation (Prog01) :

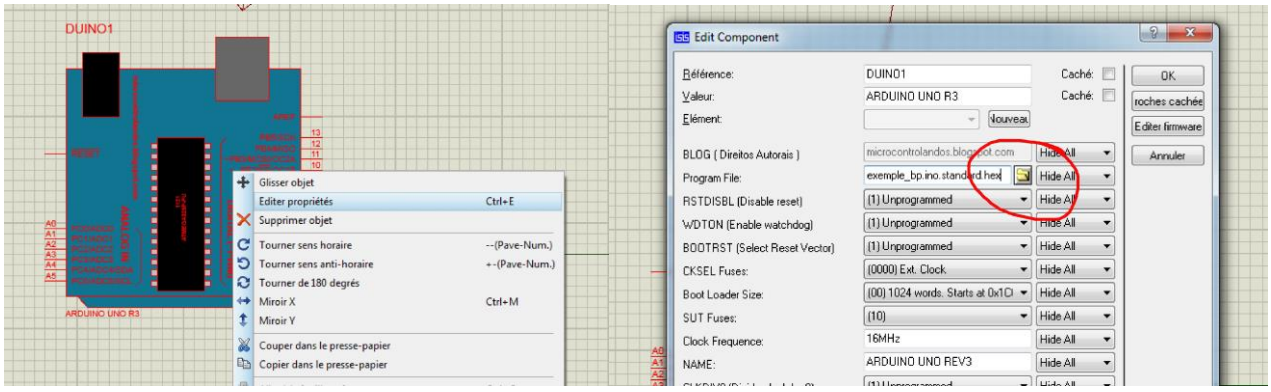
a. On vous demande en modifiant le programme existant, de faire en sorte que la LED clignote avec un cycle d'une seconde (1000ms).
Pour vous aider, vous avez l'annexe en page 8, avec le fichier "memo_arduino_1_0_v2.pdf" et vous pouvez faire des recherches sur le net car il existe des tutos.

b. Une fois le programme écrit, enregistrez le sous le nom "Prog01" puis faire une vérification en cliquant sur l'icône "Vérifier".

c. Si la compilation est bonne, il faut "Exporter les binaires compilées" afin de faire un test dans le logiciel de simulation Isis Proteus.
Arduino génère un fichier ".hex" qu'il faudra mettre dans la carte Arduino (il se trouve dans le dossier portant le nom de votre programme).



d. Charger le fichier ".hex" dans la carte Arduino en cliquant "clic droit" sur la carte Arduino puis sur "Editer Propriétés" puis ouvrez le symbole du dossier et chercher le fichier en .hex que vous venez de créer, chargez le puis lancez la simulation.



e. Validez puis lancez la simulation, la DEL doit clignoter.

f. Faire valider.

3.2. LED Clignotante sur votre câblage (Prog01) :

a. Chargez le programme dans la carte Arduino où la DEL rouge doit être allumée (se reporter à la page 4, partie d, pour téléverser le programme "Prog01".
Une fois chargé, la DEL doit clignoter.

b. Faire valider.

3.3. Alimenter 3 LEDs (Prog02) :

a. Ouvrez le schéma "TP01 - Arduino Alimentation 3 LEDs.pdsprj".

b. Compléter le schéma en câblant correctement les 3 DELs avec leurs résistances associées.

La DEL1 rouge (D1) est branchée sur la sortie 3.
La DEL2 bleue (D2) est branchée sur la sortie 5.
La DEL3 verte (D3) est branchée sur la sortie 7.

c. Calculez la valeur de chaque résistance pour les 3 DELs.

RP1 = (Valim - Vf) / Idel = (5 - 1,5) / 0,01 = 350 Ω

RP2 = (Valim - Vf) / Idel = (5 - 4) / 0,01 = 100 Ω

RP3 = (Valim - Vf) / Idel = (5 - 2,2) / 0,01 = 280 Ω

d. Indiquez leurs valeurs normalisées ainsi que leurs codes :

Valeur de RP1 normalisée : 330 Ω

Code couleur : Orange, orange, marron, or

Valeur de RP2 normalisée : 100 Ω

Code couleur : Marron, noir, marron, or

Valeur de RP3 normalisée : 270 Ω

Code couleur : Rouge, violet, marron, or

e. Modifiez sous Proteus, la valeur des 3 résistances (valeurs normalisées) et lancez la simulation. Les 3 DELS doivent s'allumer.

Faire valider.

f. Reprendre le fichier Arduino d'origine "TP01_-_Arduino_Alimentation_1_LED.ino" et modifier le pour allumer les 3 DELs en même temps, enregistrez le sous le nom Prog02 (ne pas téléverser avec de faire l'enregistrement).

g. Câblez sur plaque LAB le schéma de la Partie 3 (3 DELs).
Faire valider le câblage avant la mise sous tension.

h. Téléverser votre nouveau programme dans la carte Arduino, les 3 DEL doivent s'allumer toutes seules.
Faire valider le fonctionnement.

3.4. Faire clignoter 3 LEDs :

A partir d'ici, seul le câblage en vrai sera testé.

a. Les 3 DELs doivent clignoter en même temps (1s).
Faire le programme, enregistrez le sous le nom "Prog03" puis téléverser le.
Faire valider par le professeur.

b. Les 3 DELs doivent clignoter chacune à leur tour (0,5s).
Faire le programme, enregistrez le sous le nom "Prog 04" puis téléverser le.
Faire valider par le professeur.

D1 s'allume, D2 et D3 sont éteintes ; D2 s'allume, D1 et D3 sont éteintes ; D3 s'allume, D1 et D2 sont éteintes et ainsi de suite.

c. Les 3 DELs doivent clignoter chacune à leur tour (0,5s) mais en se chevauchant.
Faire le programme, enregistrez le sous le nom "Prog 5" puis téléverser le.
Faire valider par le professeur.

D1 s'allume, D2 et D3 sont éteintes ; D1 est encore allumée, D2 s'allume, D3 est éteinte ; D1 s'éteint, D2 est allumée, D3 est éteinte ; D1 éteinte, D2 est encore allumée, D3 s'allume et ainsi de suite.

d. Pour le dernier programme, appelé prog06, vous devez créer un programme qui donne l'illusion que la lumière se déplace comme par exemple un va et viens ou autre, vous êtes libre.
Faire le programme de votre choix, enregistrez le sous le nom "Prog06" puis téléverser le et faire valider par le professeur.

Annexe :

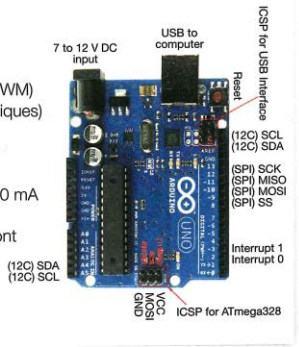
La carte Arduino

Fonction

Arduino est un circuit imprimé sur lequel se trouve un microcontrôleur. Ce microcontrôleur peut être programmé pour analyser et produire des signaux électriques, de manière à effectuer des tâches diverses telles que : le contrôle d'équipements domotiques, d'éclairages, de chauffage, le pilotage de systèmes électroniques et robotiques.

Caractéristiques

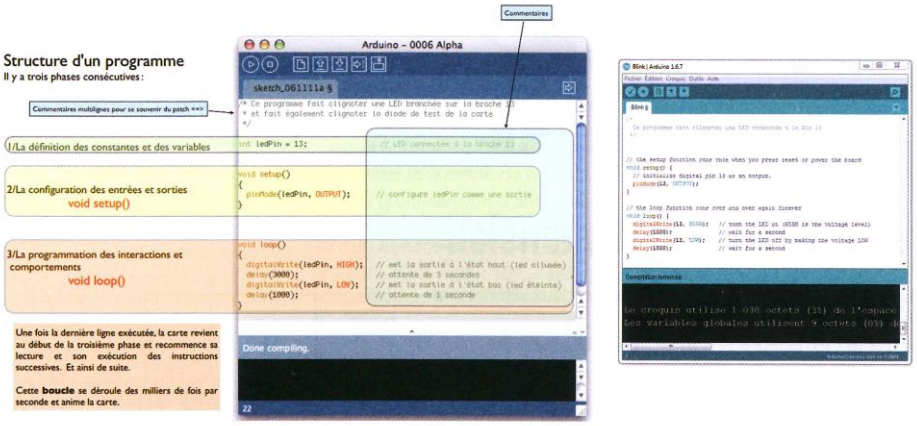
Microcontrôleur	ATmega328
Tension de fonctionnement	5 V
Tension d'alimentation (recommandée)	7-12 V
Tension d'alimentation (limites)	6-20 V
Broches E/S numériques	14 (dont 6 disposent d'une sortie PWM)
Broches d'entrées analogiques	6 (utilisables en broches E/S numériques)
Intensité maximum disponible par broche E/S (5 V)	40 mA (attention : 200 mA cumulé pour l'ensemble des broches E/S)
Intensité maxi. disponible pour la sortie 3,3 V	50 mA
Intensité maximum disponible pour la sortie 5 V	Fonction d'alimentation utilisée : 500 mA maximum si port USB utilisé seul
Mémoire programmable flash	32 kB (ATmega328) dont **0,5 kB** sont utilisés par le bootloader
Mémoire SRAM (mémoire volatile)	2 kB (ATmega328)
Mémoire EEPROM (mémoire non volatile)	1 kB (ATmega328)
Vitesse d'horloge	16 MHz



La programmation C pour Arduino

Le langage Arduino est basé sur les langages C/C++ et supporte toutes les constructions standards du langage C et quelques-uns des outils du C++. Le langage Arduino repose sur l'utilisation du compilateur C pour les microcontrôleurs AVR, AVR Libc et permet d'utiliser la plupart de ses fonctions. Tout en permettant l'utilisation des fonctions classiques disponibles en langage C, le langage Arduino (open source !) est un véritable « méta-langage » orienté pour la programmation microcontrôleur qui offre des fonctions de syntaxe très simples mais très puissantes, comme par exemple les fonctions analogRead, analogWrite, map ou encore shiftOut. La plupart des librairies utiles sont également disponibles pour la communication série avec un PC, l'utilisation d'afficheur LCD standard, de clavier matriciel, de servomoteurs ou encore de moteurs pas-à-pas.

Consulter le e-book tutoriel Arduino sur le site : eskimon.fr.



Commentaires
Toujours écrire des commentaires sur le programme : soit en multiligne, en écrivant entre /****/, soit sur une ligne de code en se séparant du code avec //

Définition des variables
Pour notre montage, on va utiliser une sortie numérique de la carte, qui est par exemple la 13e sortie numérique. Cette variable doit être définie et nommée ici : on lui donne un nom arbitraire BrocheLED. Le mot de la syntaxe pour désigner un nombre entier est int.

Configuration des entrées-sorties void setup ()
Les broches numériques de l'Arduino peuvent aussi bien être configurées en entrées numériques ou en sorties numériques. Ici, on va configurer BrocheLED en sortie. pinMod(nom, état) est une des quatre fonctions relatives aux entrées-sorties numériques.

Programmation des interactions void loop()
Dans cette boucle, on définit les opérations à effectuer, dans l'ordre :
- digitalWrite(nom, état) est une autre des quatre fonctions relatives aux entrées-sorties numériques ;
- delay(temps en millisecondes) est la commande d'attente entre deux autres instructions ;
- chaque ligne d'instruction est terminée par un point-virgule ;
- ne pas oublier les accolades, qui encadrent la boucle.

(Syntaxe en marron, paramètres utilisateur en vert)

```
/* Ce programme fait clignoter une LED branchée sur la broche 13
 * et fait également clignoter la diode de test de la carte
 */


int BrocheLED = 13;    // Définition de la valeur 13 et du nom de la broche à utiliser



void setup()
{
  pinMode(BrocheLED, OUTPUT);     // configure BrocheLED comme une sortie
}


void loop()
{
  digitalWrite(BrocheLED, HIGH);  // met la sortie num. à l'état haut (led allumée)
  delay(3000);                    // attente de 3 secondes
  digitalWrite(BrocheLED, LOW);   // met la sortie num. à l'état bas (led éteinte)
  delay(1000);                    // attente de 1 seconde
}
```

Source :
© Delagrave