La programmation, qu'est ce que c'est ?

D'après la documentation réalisée par : Dominique, Guillaume, Jean-Luc.
Disponible sur le site : http://www.locoduino.org/spip.php?article7

Qu'est ce qu'un programme ?

Un ordinateur est une machine qui exécute un programme, un peu comme un cuisinier exécute une recette de cuisine. Un programme est donc une succession d'instructions que l'ordinateur exécute, les unes après les autres, dans l'ordre.

Une recette de cuisine est rédigée en langage naturel, pour rédiger un programme, on emploie un langage **informatique dédié** qui est bien plus restreint que le langage naturel mais qui est aussi bien plus précis.

Comme un langage naturel, un langage informatique est formé de mots, de symboles de ponctuation et a une syntaxe pour construire des phrases. Il a aussi une sémantique, c'est à dire que les phrases ont une signification. Pour construire un programme, il faut enchaîner les phrases qui constituent la recette de ce que l'on veut faire. Il existe des milliers de langages informatiques et il en naît tous les jours. Pour programmer l'Arduino, nous allons étudier LE langage de l'Arduino.

Le langage de l'Arduino est dérivé du langage C++ qui est lui même dérivé du langage C. Il faut juste retenir que savoir programmer en Arduino permettra d'avoir quelques compétences en C++. Dans le monde Arduino, un programme s'appelle un *Croquis*, ou un *Sketch* en anglais, ne pas oublier que l'Arduino a, à l'origine, été conçu pour des artistes.

Les mots pour le dire

En français, les phrases se terminent par un point. En Arduino les phrases se terminent par un point-virgule. Une des phrases que l'on rencontre très fréquemment est la suivante :

```
1. digitalWrite(4,HIGH);
```

Cette phrase signifie « mettre la sortie numérique 4 à l'état haut ». Comme en français une phrase isolée de son contexte peut vouloir dire autre chose que dans son contexte. La signification qui vient d'être donnée suppose que la broche 4 a été programmée en sortie, ce qui n'est pas forcément le cas. Si la broche 4 a été programmée en entrée, cette phrase a un sens différent. Il ne suffit donc pas que les instructions soient correctes pour que le programme le soit. Comme en français une succession de phrases correctes ne fait pas un discours qui a du sens. On y reviendra. Cette phrase est formée de mots, de nombres et de signes de ponctuation. Un *mot* commence obligatoirement par une lettre et ne contient que des lettres, des chiffres et le caractère de soulignement : _. Les mots s'appellent également *identifiants*.

Puisqu'on en parle, la fonction permettant de programmer la direction d'une broche est la fonction pinMode. Si l'on veut que notre digitalWrite mette la sortie 4 à l'état haut, il faut **préalablement** avoir programmé la broche 4 en sortie. Comme ceci :

```
1. pinMode(4,OUTPUT);
```

Nous allons arrêter d'utiliser **phrase** car en informatique le mot est ***instruction*** étant donné que toutes les phrases sont des ordres que l'ordinateur exécute et non un discours philosophique.

Les fonctions

Revenons à digitalWrite. Cette ***instruction*** est l'appel d'une fonction, nous allons voir plus loin et en détail de quoi il s'agit. Pour l'heure, il faut juste retenir que le logiciel qui vient avec l'Arduino fournit un catalogue de fonctions qui facilite la programmation. *4* est le premier *argument* de la fonction, il s'agit du numéro de la broche concernée. *HIGH* est le second *argument* de la fonction et signifie état haut de la tension électrique. Ça peut être 5V sur un Arduino alimenté en 5V ou bien 3,3V s'il est alimenté en 3,3V. Concrètement, quand un Arduino Uno exécute cette instruction, la tension sur la broche 4 est mise à 5V.

Notez bien les parenthèses qui délimitent le début et la fin des arguments de la fonction, la virgule qui les sépare, ainsi que le point-virgule qui termine l'instruction. Toute omission de ces symboles empêchera que le programme soit exploitable par l'Arduino.

Une fonction est un ensemble d'instructions, un bout de recette de cuisine qui a un nom. Pour continuer dans l'analogie avec la cuisine, si on exécute une recette de tarte, vous trouverez un appel à la fonction : confectionner une pâte sablée. Pour exécuter cette fonction, il faudra prendre sa recette et exécuter les instructions qui la composent puis revenir à l'exécution de la recette de la tarte. L'avantage des fonctions est la concision et la réutilisation du même bout de recette partout où on en a besoin. Toutes les recettes de tarte utilisent la sous-recette de la pâte sablée, il est inutile de recopier cette-sous recette dans toutes les recettes. Il suffit d'y écrire : « Confectionnez une pâte sablée ».

On peut, bien évidemment, faire ses propres fonctions, on dit les *définir*, et d'ailleurs c'est ce que l'on fait quand, dans l'IDE de l'Arduino, on crée un nouveau croquis. L'IDE insère automatiquement deux fonctions que l'on remplit ensuite. **Il s'agit des fonctions setup et loop**.

Comme ceci :

```
1. void setup() {
2.
3. }
4.
5. void loop() {
6.
7. }
```

Une fonction peut retourner ou non une valeur comme la sous-recette de la pâte sablée retourne... une pâte sablée alors que la sous recette « Préchauffer le four thermostat 7 » ne retourne rien. Ici void indique que ces fonctions ne retournent aucune valeur. Rien n'apparaît entre les parenthèses car ces fonctions n'ont aucun argument. Les accolades, une ouvrante et une fermante, délimitent les instructions de la fonction. C'est à dire que toutes les instructions situées entre les accolades font partie de la fonction et seront exécutées l'une après l'autre quand la fonction sera appelée. Ce n'est pas vous qui appelez setup et loop. Ces fonctions sont appelées automatiquement mais par contre vous pouvez leur donner un contenu.

setup est appelé une fois seulement, on va donc y mettre les instructions qui mettent en place les choses, c'est un peu la section *ingrédient* d'une recette de cuisine. En informatique on parle d'*initialisation*.

loop est appelé de manière répétitive, on va donc y mettre les instructions qui doivent être exécutées répétitivement. Ici on s'éloigne de la recette de cuisine qui a un début et une fin. Par essence, un programme Arduino ne se termine jamais.

Un programme complet qui met la broche 4 en sortie puis la met à l'état haut de manière répétitive s'écrirait donc comme suit.

```
1. void setup() {
2.   pinMode(4,OUTPUT);
3. }
4.
5. void loop() {
6.   digitalWrite(4,HIGH);
7. }
```