3. Les variables

En programmation, les données sont stockées dans des variables, elle-mêmes composées d'un certains nombre de bits.

Récapitulatif des types de variables et de leur plage d'encodage en C sur Arduino Uno:

Type de variable	Nombre de bits	Plage d'encodage
bool	8 bits	true / false
unsigned char / byte	8 bits	0 à 255
char	8 bits	-128 à 127
unsigned int	16 bits*	0 à 65535
int	16 bits*	-32 768 à 32 767
unsigned long	32 bits*	4 294 967 296
long	32 bits*	-2 147 483 648 à 2 147 483 647
float	32 bits	-3,4028235E+38 à 3,4028235E+38
double	32 bits*	identique à float*

*: les Arduinos de première génération (puce AVR), comme le Uno, ne permettent pas de dépasser 32 bits. Les générations plus récentes (puce ARM) permettent d'outrepasser cette limitation originelle.

© Sébastien Marchand - 2020

3.1- Déclaration de variables en langage C:

Le C est un langage typé statiquement. En d'autres termes, avant d'être utilisées, toutes les variables doivent être déclarées. Déclarer une variable signifie définir son type et éventuellement définir une valeur initiale (lui affecter une valeur). Il n'est pas nécessaire d'initialiser les variables lorsqu'elles sont déclarées, mais c'est souvent utile.

Exemple en langage C:

```
int inputVariable1;
int inputVariable2=0 ;
```

Déclaration d'une variable nommée « inputVariable1 » de type « int » sans l'initialiser.

A la ligne suivante nous déclarons une seconde variable « inputVariable2 » de même type et initialisée à 0.

Note: la syntaxe grammaticale du langage C impose ici un point virgule en fin de ligne d'instruction.

© Sébastien Marchand - 2020

Que se passe-t-il si on met dans une variable une valeur plus grande que sa plage d'encodage autorisée?

C'est ce que l'on appelle un débordement. La variable va reboucler sur sa valeur minimale. Cela peut avoir des conséquences fâcheuses sur l'exécution d'un programme.

Pourquoi alors ne pas utiliser tout le temps des variables 32 bits?

Plus on utilise d'octets et plus une variable prend de la place en mémoire. Hors un Arduino a très peu de mémoire vive (SRAM). De même cela ralentit les performances de calcul. Il est donc important de bien évaluer les besoins et de choisir ses types de variables en conséquence, lors de leur déclaration, si l'on souhaite écrire un code performant.

© Sébastien Marchand - 2020

3-2 Les données complexes

Les variables ne peuvent contenir qu'une seule donnée. Cela peut vite devenir lourd à gérer.

3-2-1 Les tableaux

On appelle tableau une variable composée de données de même type, stockées de manière contiguë en mémoire (les unes à la suite des autres).

Exemple en langage C:

```
char name[12] ;
```

Déclaration d'un tableau nommée « name » qui peut contenir 12 variables de type « char »: Nous venons de créer une chaine de caractères.

```
byte position[4] = {0, 180, 90, 45} ;
```

Déclaration d'un tableau nommé « position » qui peut contenir 4 variables de type « byte ». Nous avons également initialisé les valeurs de chaque case du tableau.

© Sébastien Marchand - 2020

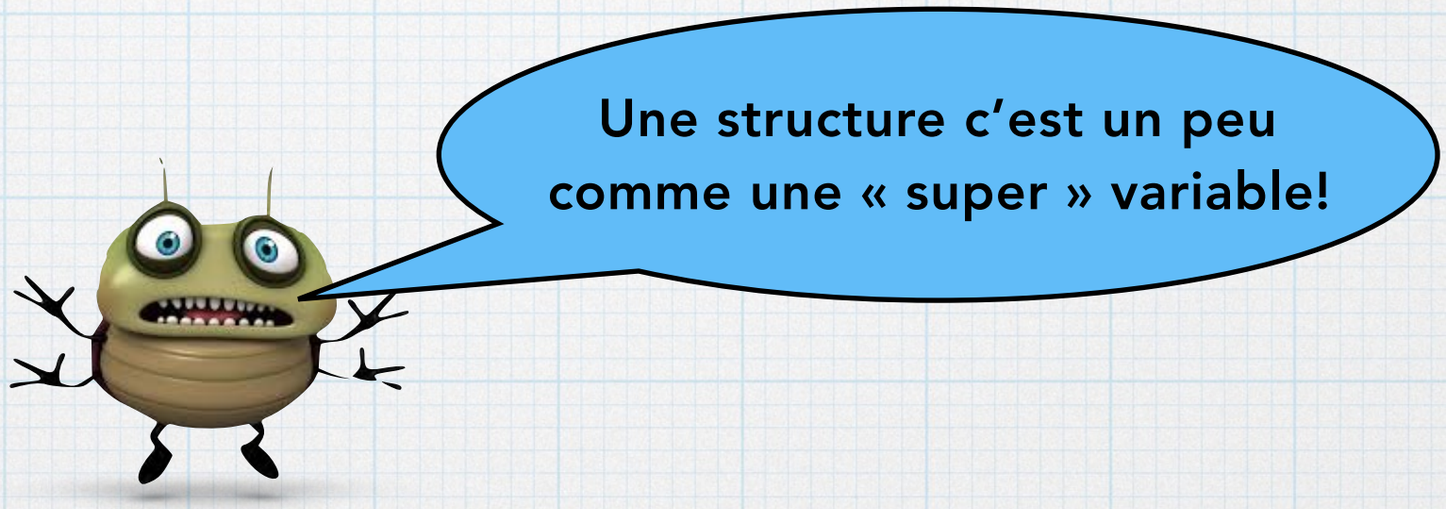
3-2-2 Les structures

Les structures permettent de regrouper des variables au sein d'une entité repérée par un seul nom de variable.

Les variables contenues dans une structure sont appelées champs de la structure.

Exemple en langage C:

```
struct Motor
{
   float  speed;
   bool  direction;
};
../..
Motor motor1;
motor1.direction = true;
motor1.speed     = 100;
```



Déclaration d'une structure nommée motor1 et de type « Motor » qui contient deux champs: « speed » et « direction ».

© Sébastien Marchand - 2020