





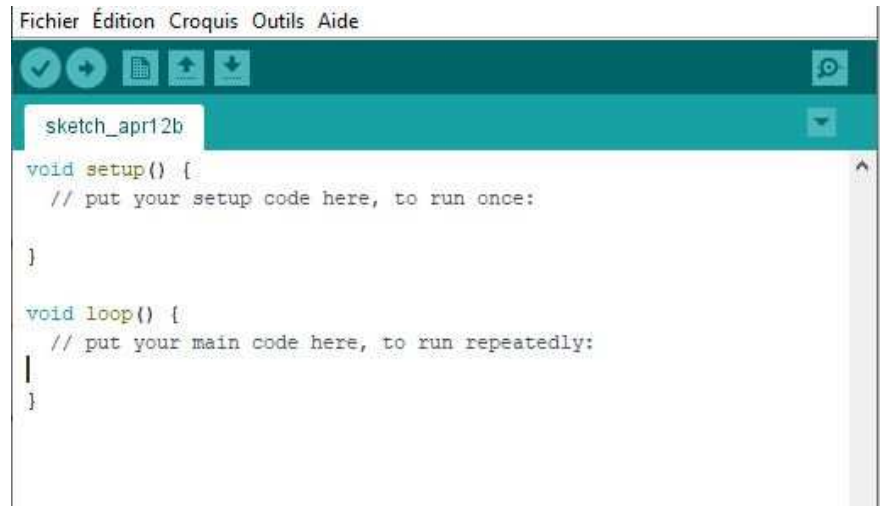


## II - Programmer une carte Arduino

Pour écrire du code qui sera transmis à la carte, il faut installer l'IDE Arduino.

L'interface d'origine se présente ainsi :

	Permet de vérifier si le code est correctement écrit (compris par la machine).
	Permet de téléverser le code dans la carte.
	Permet de créer un nouveau fichier.
	Permet d'ouvrir un fichier.
	Permet d'enregistrer le programme.
	Permet d'ouvrir le moniteur série (pour afficher des données sur l'ordinateur).



Le langage choisi est le C.

Par ailleurs, tout programme contiendra nécessairement les fonctions `setup()` et `loop()`.

- `setup()` : permet d'initialiser des informations notamment le rôle de ports en entrée ou sortie (ce qui différencie le capteur de l'actionneur).  
C'est ici qu'on pourra également déclarer l'utilisation du moniteur série (permettant l'affichage d'informations sur l'ordinateur) ;
- `loop()` : cela signifie « boucle ». Cette fonction est continuellement appelée par le programme. Pour ne pas surcharger la carte d'informations et parfois pour constater un changement d'état, on utilise la fonction `delay()` ;
- `delay(500)` : provoque une pause de 500 ms. Ce délai est à adapter en fonction de la situation.

En C, les blocs sont délimités par des accolades. Par exemple,

Instruction conditionnelle	Boucle pour	Boucle while
<pre>if (condition) {   instructions1; } else {   instructions2; }</pre>	<pre>for(int i=0;i&lt;10;i++) {   instructions; }</pre>	<pre>while(condition){   instructions; }</pre>

Par ailleurs, toute variable doit être déclarée et prototypée. Par exemple :

- `int a=0;` pour déclarer un entier
- `float a=0.0;` pour déclarer un nombre écrit en virgule flottante
- `char a="";` pour déclarer une chaîne de caractères
- `boolean a=true;` pour déclarer une donnée logique (true ou false)

Remarque : À la fin de chaque ligne d'instruction il faut mettre un point-virgule.