Partie1

Leçon 5

Blink and Add Libraries

But de la leçon

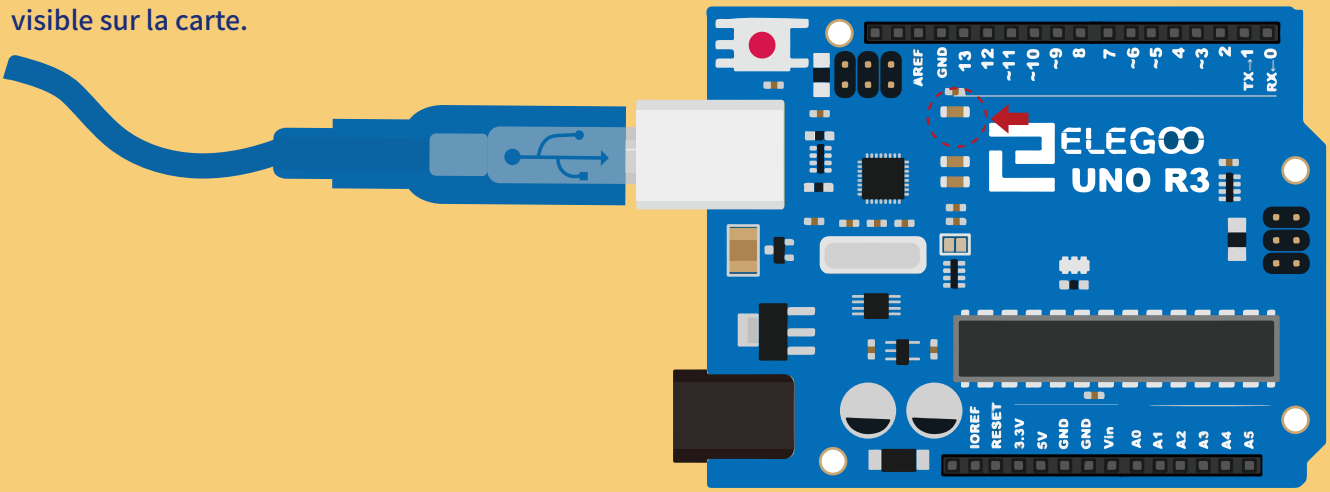
De plus, nous allons apprendre à ajouter des bibliothèques (libraries) dans lesquelles nous pourrons retrouver des fonctions utiles pour les leçons à venir. Cela permettra ainsi d' améliorer et de simplifier les fonctionnalités de l'Arduino tout en augmentant les possibilités.

Matériel nécessaire:

(1) x Elegoo Uno R3

Principe

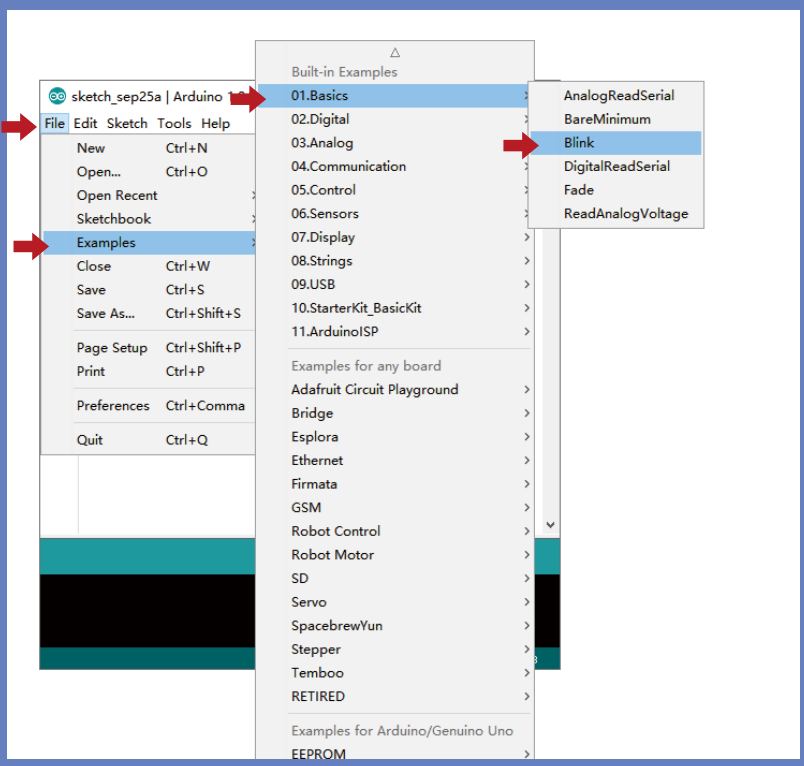
La carte UNO R3 possède deux rangées de connecteurs le long de ses extrémités qui sont utilisés pour brancher une vaste gamme de composants électroniques ou des cartes d' extensions (appelées shields) qui augmentent ses capacités.
Elle est aussi équipée d' une LED intégrée qu' il est possible de commander au travers de vos programmes. Vous pouvez apercevoir cette LED sur l' image ci-dessous, elle est repérable grâce au « L » visible sur la carte.

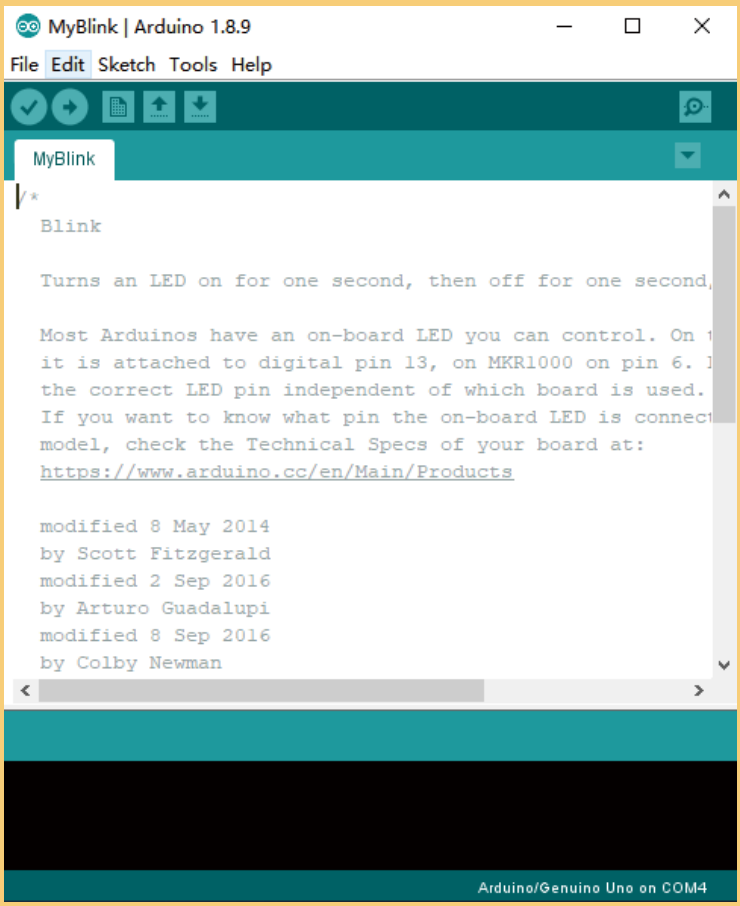


Lorsque vous connectez votre carte pour la première fois, il est possible que la LED se mette à clignoter. C' est parce que les cartes sont fréquemment expédiées avec le programme « BLINK » préinstallé.

Dans cette leçon, nous allons reprogrammer la carte UNO R3 avec notre propre programme « BLINK », ce qui va nous permettre notamment de changer la fréquence de clignotement de la LED.

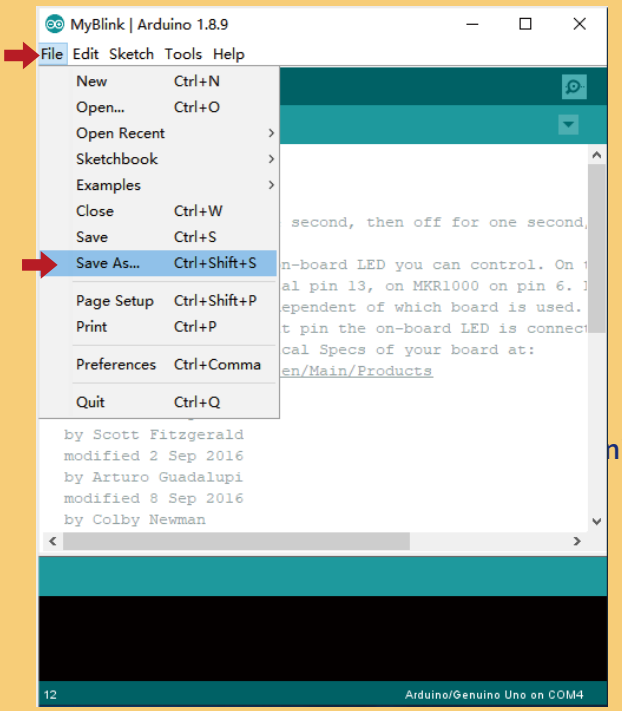
Ouvrez le sketch "BLINK" dans l' environnement de programmation via le menu "FICHIER/EXEMPLES/01.BASICS/Blink"

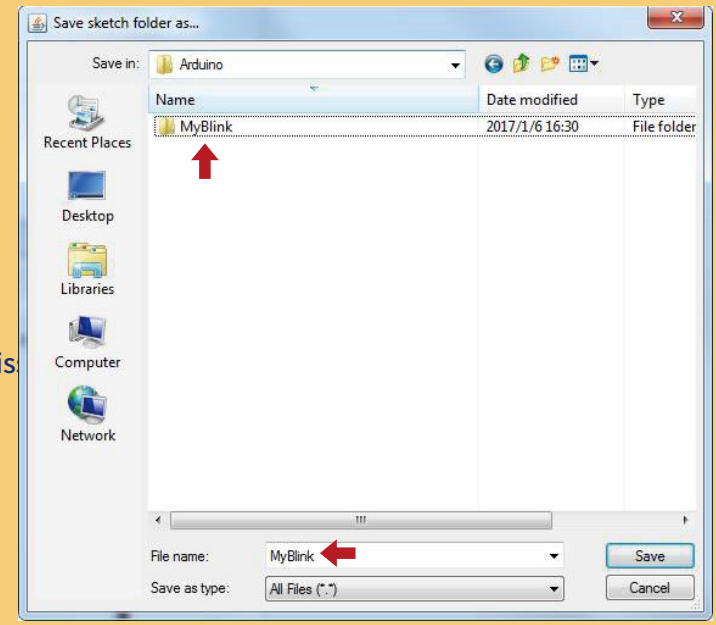




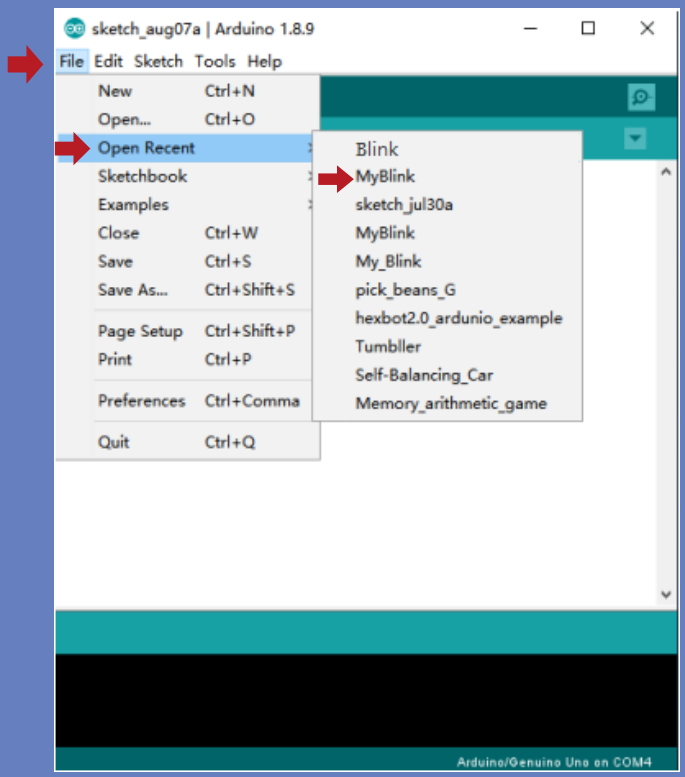
Voici ce que vous devez obtenir.

Comme vous allez apporter des modifications à ce sketch, la première étape consiste à l' enregistrer dans votre répertoire personnel, les sketchs exemples étant en lecture seule.

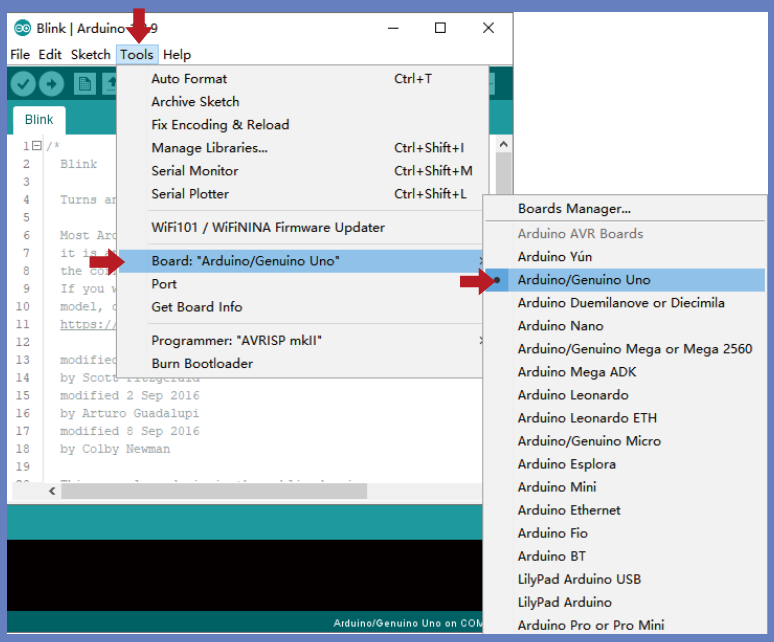


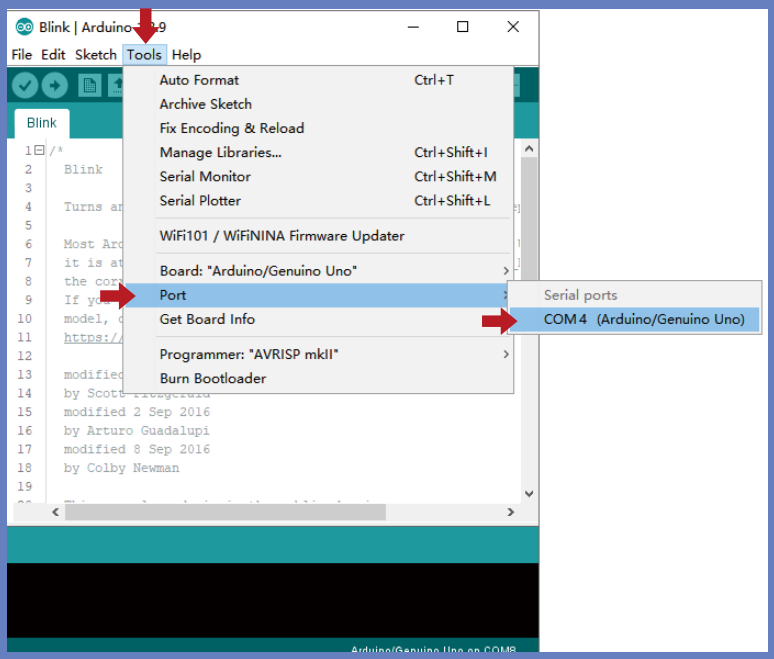


Vous venez d' enregistrer le fichier dans votre répertoire.



Il est temps maintenant de connecter la carte UNO R3 via le port USB de votre ordinateur et de vérifier la bonne reconnaissance de celle-ci.



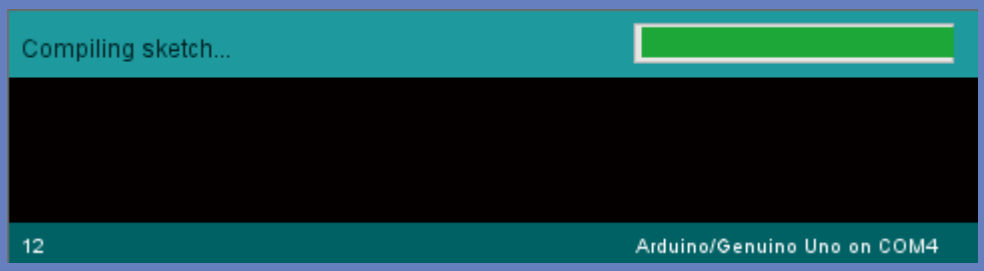


Note: le numéro de port COM ne sera pas nécessairement celui que vous pouvez apercevoir sur les captures d' écran. En effet, c' est votre ordinateur qui va affecter une référence au moment de la connexion et si vous possédez plusieurs cartes, chacune aura potentiellement un numéro différent.

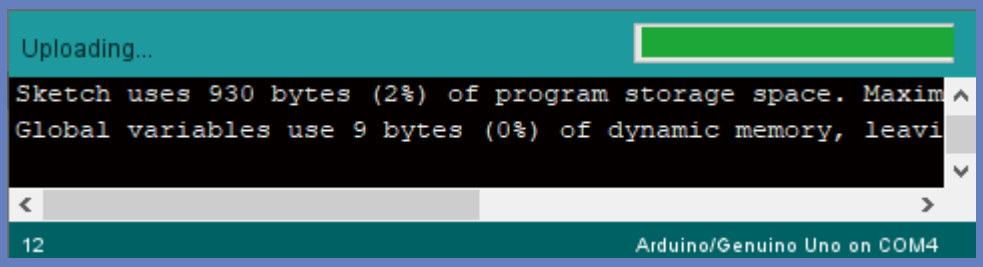
Une fois connecté, vous pouvez apercevoir ce petit texte en bas à droite de votre écran.

Cliquez sur le bouton avec la flèche pour déclencher le téléversement.

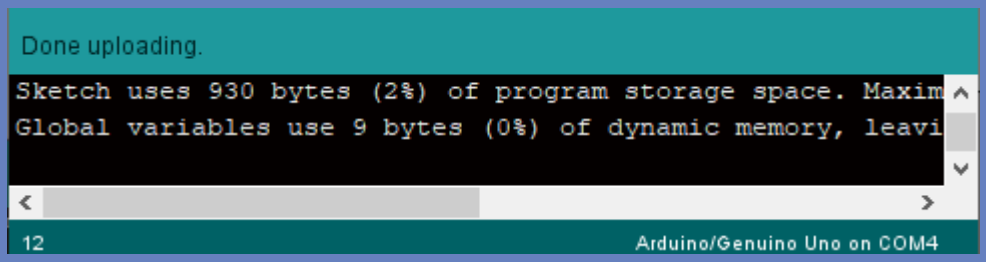
Vous pouvez constater que la première étape est la compilation (le logiciel vérifie l' intégrité du code).



Ensuite, le téléversement à proprement parler commence.



Enfin, le statut "terminé" s' affiche.



Un message vous renseigne sur la taille du programme et la quantité de mémoire utilisée sur la carte.
Si votre carte n' est pas correctement connectée ou reconnue, vous obtiendrez l' écran suivant.
Reportez-vous aux explications données dans les premières leçons pour veillez à avoir correctement installé les drivers.

Vous pouvez constater qu' une part importante du code est consacrée aux commentaires. Les commentaires sont importants ils permettent de bien comprendre le code, surtout quand celui-ci est complexe. Cela permet d' y revenir plus tard lorsqu' il est nécessaire de le faire évoluer.

Code :

Notez qu'une grande partie de ce croquis est composée de commentaires. Ce ne sont pas des programmes réels instructions; Ils expliquent plutôt le fonctionnement du programme.
Ils sont là pour vous.

Le croquis commence par

```
/*
Blink
Turns an LED on for one second, then off for one second, repeatedly.
...
This example code is in the public domain. http://www.arduino.cc/en/Tutorial/Blink
*/
// the setup function runs once when you press reset or power the board
```

//
[Syntaxe supplémentaire]
Description
Les commentaires de ligne sont des lignes du programme qui servent à vous informer ou à informer les autres lecteurs sur le fonctionnement du programme. Ils sont ignorés par le compilateur, et ne sont pas exportés vers le processeur, de sorte qu'ils ne prennent pas de place dans la mémoire flash du microcontrôleur. Les commentaires ont pour seul but de vous aider à comprendre (ou à vous souvenir), ou d'informer les autres sur le fonctionnement de votre programme.
Un commentaire d'une seule ligne commence par // (deux barres obliques adjacentes). Ce commentaire se termine automatiquement à la fin d'une ligne. Tout ce qui suit // jusqu'à la fin d'une ligne sera ignoré par le compilateur.
/* */
[Syntaxe Supplémentaire]
Description
Le début d'un bloc de commentaires (commentaire sur plusieurs lignes) est marqué par le symbole /* et le symbole */ marque sa fin. Ce type de commentaire est appelé ainsi car il peut s'étendre sur plus d'une ligne ; une fois que le compilateur a lu le /*, il ignore tout ce qui suit jusqu'à ce qu'il rencontre un */.

The first block of code is:

```
void setup() {
// initialize digital pin LED_BUILTIN as an output.
pinMode(LED_BUILTIN, OUTPUT);
}
```

In this case, there is just one command there, which, as the comment states tells the Arduino board that we are going to use the LED pin as an output.
setup()
[Sketch]
Description
La fonction setup() est appelée quand un sketch commence. Utilisez-la pour initialiser des variables, définir le mode des Pins, charger les bibliothèques à utiliser, etc. La fonction setup() ne sera exécutée qu'une seule fois, après chaque mise sous tension ou réinitialisation de la carte Arduino.

{ ...}
[Syntaxe Supplémentaire]
Description

Les accolades constituent une part importante du langage de programmation C++. Ils sont utilisés dans plusieurs constructions différentes, décrites ci-dessous, et cela peut parfois être déroutant pour les débutants.

Un accolade ouvrante { doit toujours être suivie d'une accolade fermante }. C'est une condition qui est souvent appelée "l'équilibre des accolades". L'IDE (Integrated Development Environment) d'Arduino comprend une fonction pratique pour vérifier l'équilibre des accolades bouclées. Il suffit de sélectionner une accolade, ou même de cliquer sur le point d'insertion qui suit immédiatement une accolade, et son compagnon logique sera mis en évidence.

Des accolades déséquilibrées peuvent souvent conduire à des erreurs de compilation complexes et incompréhensibles qui peuvent parfois être difficiles à trouver dans un programme de grande taille. En raison de leurs utilisations variées, les accolades sont également extrêmement importantes pour la syntaxe d'un programme et le déplacement d'une ou deux lignes d'accolades affecte souvent de manière spectaculaire la signification (et donc son interprétation par le compilateur) d'un programme.

It is also mandatory for a sketch to have a 'loop' function.

```
// the loop function runs over and over again forever
void loop() {
digitalWrite(LED_BUILTIN, HIGH);  // turn the LED on (HIGH is the voltage level)
delay(1000);      // wait for a second
digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
delay(1000);// wait for a second
}
```

Loop : Contrairement à la fonction "setup" qui n'est exécutée qu'une fois, la fonction "loop" est exécutée en boucle.
digitalWrite(Pin num , HIGH/LOW):Sélection d'une valeur HAUTE ou BASSE sur un pin digital.
Si le pin est configuré comme une SORTIE avec la fonction pinMode(), sa tension sera réglée à la valeur correspondante : 5V pour HIGH, 0V (masse) pour LOW.

delay(ms):Met le programme en pause pendant la durée (en millisecondes) spécifiée en paramètre.
(Il y a 1000 millisecondes dans une seconde).

ms : le nombre de millisecondes définissant la durée de la pause.(Plage de valeurs possibles : 0~4394967295)

Il est aussi obligatoire dans un sketch d' avoir un bloc "loop". Contrairement au premier bloc celui-ci s' exécutera en boucle tant que la carte sera sous tension.

```
30 // the loop function runs over and over again forever
31 void loop() {
32   digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the volt
33   delay(500)                         // wait for a second
34   digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the vo
35   delay(500)                         // wait for a second
36 }
```

Le bloc « loop » est constitué de 4 instructions. La première déclenche l' allumage de la led, la deuxième une attente de 1000ms, la troisième l' extinction de la led. La dernière instruction une attente de 1000ms.

Installer des bibliothèques complémentaires

Lorsque vous aurez bien saisi les fonctions intégrées de l' environnement Arduino, vous aurez certainement le besoin ou l' envie d' aller encore plus loin, avec pourquoi pas des bibliothèques complémentaires.

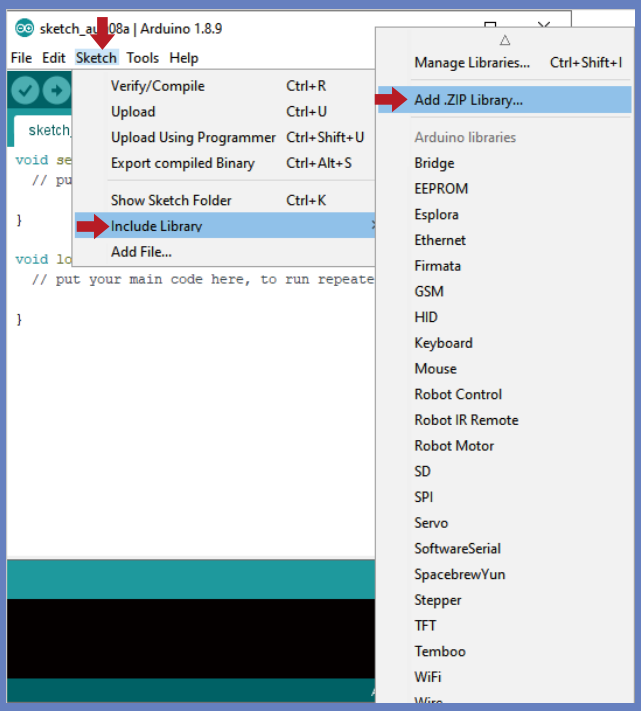
Qu' est-ce qu' une bibliothèque?

Une bibliothèque est une suite d' instructions qui rendent beaucoup plus facile l' utilisation de composants complexes. Cela peut être pour l' utilisation d' un écran à cristaux liquide, pour l' utilisation d' un servomoteur etc…

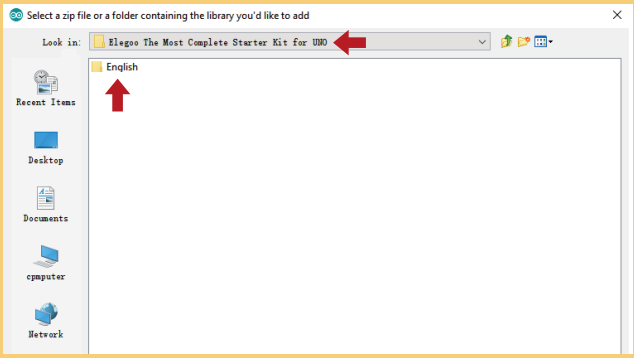
Si la bibliothèque recherchée n' apparait pas

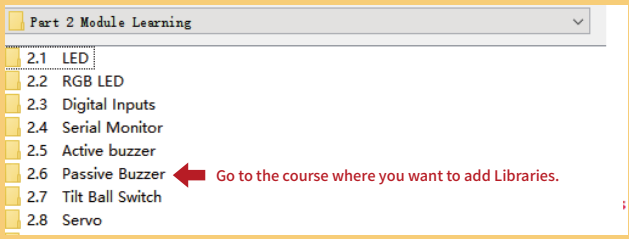
1) Importing a .zip Library

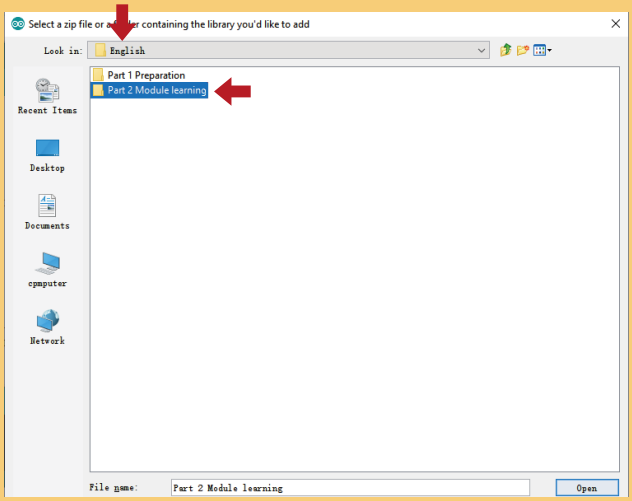
Procurez-vous sur internet (notamment sur GITHUB) le zip de la bibliothèque que vous souhaitez ajouter. Il n' est pas nécessaire de dézipper les fichiers, mais notez bien l' emplacement du fichier. Dans le menu « Sketch », sélectionner « Inclure une bibliothèque ». Sélectionnez ensuite « Ajouter .ZIP Library ».

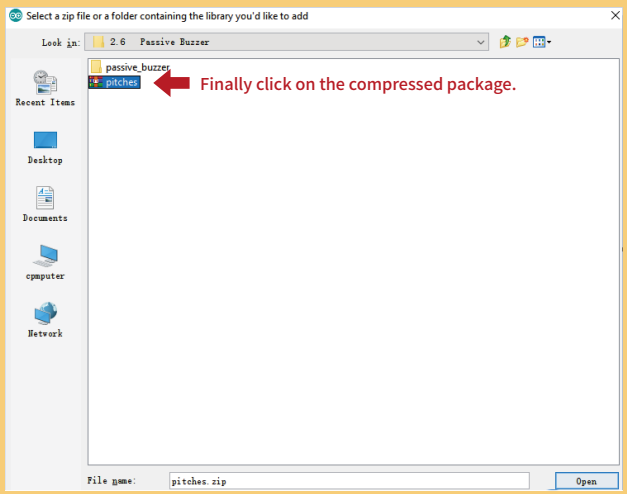


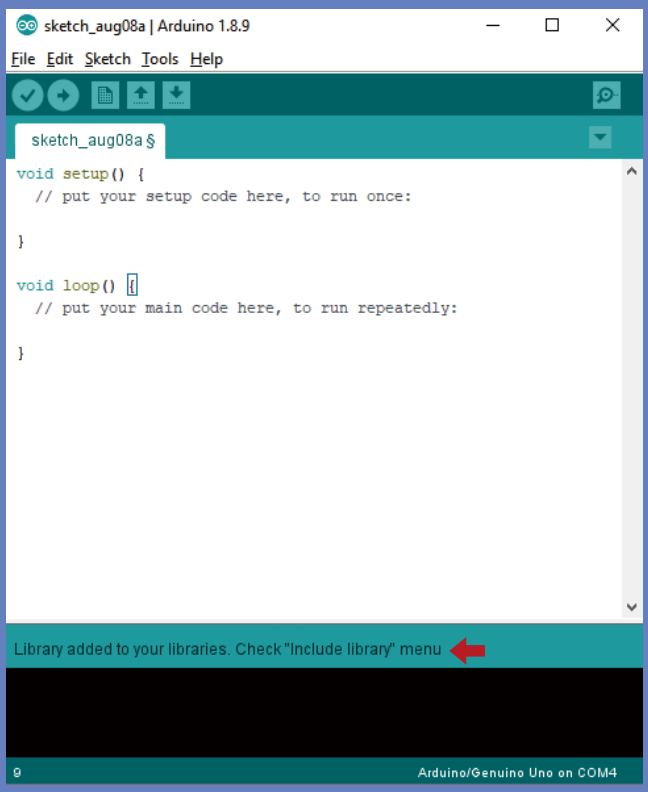
Vous serez invité à sélectionner la bibliothèque que vous voudrais ajouter. Accédez aux fichiers .zip l'emplacement et l'ouvrir.
Si la bibliothèque est utilisée dans chaque dossier de cours, elle être fourni avec la bibliothèque correspondante package de compression.
Prenons l'exemple de la leçon 11







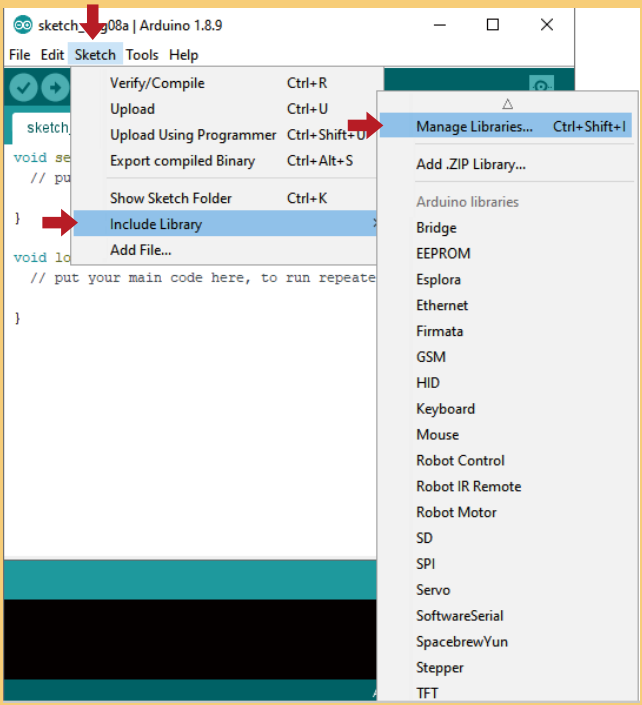




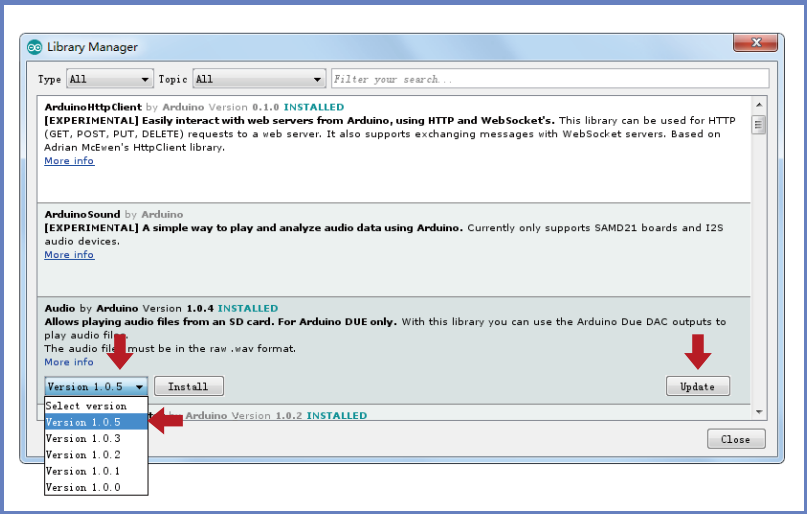
Revenez au menu Sketch> Import Library. Vous devriez maintenant voir la bibliothèque en bas du menu déroulant. Il est prêt à être utilisé dans votre croquis. Le fichier zip sera étendu dans le dossier des bibliothèques dans votre répertoire d'esquisser Arduino. NB: la bibliothèque sera disponible à utiliser dans des croquis, mais les exemples pour la bibliothèque ne seront pas exposés dans le fichier> Exemples jusqu'à la redémarrage de l'IDE.

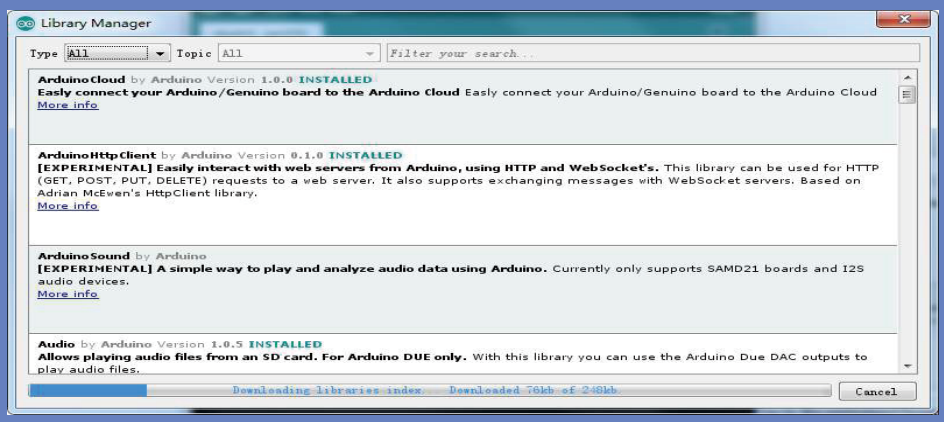
2) Utilisation du gestionnaire de bibliothèque

Pour installer une nouvelle bibliothèque dans votre Arduino IDE, vous pouvez utiliser le gestionnaire de bibliothèque(disponible à partir de la version IDE 1.8.9). Ouvrez l'IDE et cliquez sur le menu "Sketch" puis Inclure la bibliothèque> Gérer les bibliothèques.



Ensuite, le gestionnaire de bibliothèque s'ouvrira et vous trouver une liste des bibliothèques déjà installées ou prêt pour l'installation. Dans cet exemple, nous allons installer la bibliothèque audio. Faites défiler la liste pour la trouver, puis sélectionnez la version de la bibliothèque que vous souhaitez installer. Parfois, une seule version de la bibliothèque est disponible. Si le menu de sélection de version n'apparaît pas, ne inquiétude: c'est normal.
Il faut parfois être patient avec ça, comme indiqué sur la figure. Veuillez le rafraîchir et attendre.





Vous allez pouvoir voir l' ensemble des bibliothèques déjà présentes ainsi que la version utilisée :

Si la dernière version d' une bibliothèques n' est pas installée, le bouton “Installer” se dégrise, vous pouvez procéder à son installation.

3)Installation manuelle

Pour installer la bibliothèque, quittez d'abord l'application Arduino. Ensuite, décompressez le fichier ZIP contenant la bibliothèque. Par exemple, si vous installez une bibliothèque appelée "ArduinoParty", décompressez ArduinoParty.zip. Il devrait contenir un dossier appeléArduinoParty, avec des fichiers comme ArduinoParty.cpp et ArduinoParty.h à l'intérieur. (Si les fichiers .cpp et .h ne sont p as dans un dossier, vous devrez en créer un. Dans ce cas, vous devez créer un dossier appelé "ArduinoParty" et y déposer tous les fichiers qui se trouvaient dans le ZIP Fichier, comme ArduinoParty.cpp et ArduinoParty.h.)Faites glisser le dossier ArduinoParty dans ce dossier (votre dossier de bibliothèques). Sous Windows, il sera probablement appelé "Mes documents \ Arduino \ bibliothèques". Pour les utilisateurs de Mac, il sera probablement appelé «Documents / Arduino / bibliothèques». Sur Linux, ce sera le dossier "bibliothèques" dans votre carnet de croquis.

Votre dossier de bibliothèque Arduino devrait maintenant ressembler à ceci
My Documents\Arduino\libraries\ArduinoParty\ArduinoParty.cpp
My Documents\Arduino\libraries\ArduinoParty\ArduinoParty.h
My Documents\Arduino\libraries\ArduinoParty\examples
Ou comme ceci (sur Mac et Linux):
Documents/Arduino/libraries/ArduinoParty/ArduinoParty.cpp
Documents/Arduino/libraries/ArduinoParty/ArduinoParty.h
Documents/Arduino/libraries/ArduinoParty/examples
....

Il peut y avoir plus de fichiers que les fichiers .cpp et .h, assurez-vous qu'ils sont tous là.
(La bibliothèque ne fonctionnera pas si vous mettez les fichiers .cpp et .h directement dans le dossier des bibliothèques ou si elles sont imbriquées dans un dossier supplémentaire. Par exemple: Documents \ Arduino \ bibliotecas \ ArduinoParty.cpp et Documents \ Arduino \ Les bibliothèques \ ArduinoParty \ ArduinoParty \ ArduinoParty.cpp ne fonctionneront pas.)
Redémarrez l'application Arduino. Assurez-vous que la nouvelle bibliothèque apparaît dans l'élément de menu Sketch-> Import Library du logiciel. C'est tout! Vous avez installé une bibliothèque!