Balance à CG basée sur Arduino pour les planeurs F3X (et autres).

Enjoy !

# Introduction

Ce document présente les consignes de montage d’une balance à Cg utilisable pour des planeurs de F3x.

Fin 2017, de nombreux posts sont apparus sur les forums d’aéromodélisme spécialisés, présentant une balance numérique dédiée à la détermination du Cg pour des planeurs de F3F et F3B.

En fouillant sur le net à la recherche d’information sur cet équipement commercialisé un peu plus de 130 euros dans divers boutiques, je suis tombé sur le site **d’Olav Kallhovd**, pilote Norvégien de F3F **merci à lui**, (<https://github.com/olkal/CG_scale>), qui a mis en open source toutes les informations nécessaires pour construire une balance très similaire aux modèles commerciaux en utilisant deux cartes Arduino et une imprimante 3D.

La photo ci-dessous présente la balance conçue par Olav. Tous les éléments nécessaires pour la construction sont disponibles sur son site, y compris les fichiers « .stl » pour les impressions 3D.

Cette balance peut être utilisée pour des planeurs de type F3F, F3B et F5J, les différentes largeurs étant disponibles pour l’impression 3D.



Après avoir construit un modèle identique, avec l’aide de Damien Donses, Christophe Nocchi et le support moral de Fred Hours, nous avons introduit quelques modifications pour faciliter et simplifier la construction de cette balance :

* L’écran LCD 1602 initial a été remplacé par un écran OLED 0,96’’ en I2C. Il est également possible d’utiliser un écran LCD 1602 en I2C et vous trouverez sur le site ci-dessous les fichiers Arduino pour les trois versions (version 1602 de base proposée par Olav, OLED 0,96’’ I2C et 1602 I2C).
* L’utilisation de l’I2C pour l’écran permet de simplifier le câblage et de n’utiliser qu’une seule carte Arduino au lieu de 2,
* La led externe visible sur la photo ci-dessus a été remplacée par la led présente de base sur les cartes Arduino. Si vous imprimez en PLA blanc comme sur la photo, le clignotement de la led dans la base reste très visible,
* J’ai dessiné un petit PCB pour faciliter l’intégration des quelques composants qui constituent l’électronique. Ce PCB s’intègre dans la base de la balance, et les fichiers Fritzing et Gerber associés sont également disponibles sur le github ci-dessous. Avec ces fichiers, il vous est possible de commander directement le PCB sur le site de Fritzing Fab (<https://aisler.net/fritzing>) ou Seed Fusion (<https://www.seeedstudio.com/fusion_pcb.html>). Il est tout à fait possible de se passer de ce PCB pour monter la balance, j’ai d’ailleurs construit le premier modèle comme ça en câblant les divers composants l’un à l’autre en fil à fil … mais c’est moins simple !
* Quelques stl ont été repris pour « agrandir quelques trous » et nous avons créé un stl supplémentaire pour intégrer un bras de connexion permettant de rattacher l’écran OLED à la base.

Toutes ces modifications, y compris le fichier Fritzing et les gerber pour le PCB, sont disponibles en Open Sources sur le site <https://github.com/adesandr/CG_Scale_OlkalBreakout>.

# Les composants nécessaires

La liste des composants nécessaires est disponible à l’adresse :

<https://github.com/adesandr/CG_Scale_OlkalBreakout/blob/master/Documentation/BOM.xls>.

Tous les composants sont disponibles sur Ebay, Amazon ou équivalent.

La version originale d’Olav revient environ à 29 euros. Les versions I2C reviennent environ à 24 euros. La balance se monte en une demi-journée, voire une journée grand maximum, calibrage compris, fonction de votre expérience avec un fer à souder et de vos connaissances de l’environnement Arduino. Je ne peux que vous encourager à vous lancer, vous verrez c’est extrêmement simple et le tuto est un peu long pour vous présenter pas à pas les différentes étapes du montage.

J’attire votre attention sur les deux warnings présents dans le fichier pour l’achat des cartes HX711 et de la carte Arduino Pro mini 5V.

# Imprimer la balance

Une Imprimante 3D avec un plateau de 220x220 minimum est nécessaire. Le PLA Ice Filament disponible sur Amazon fonctionne bien, ainsi que le PLA Hobby King, avec 2 épaisseurs de paroi et un remplissage à 20%. L’impression de toutes les pièces de la balance prend une vingtaine d’heures.

Les pièces doivent être imprimées avec la même orientation que dans les fichiers STL.

Les trous pénétrants sont « aveuglés » pour une meilleure impression et doivent être repris par la suite à la Dremel.

# Montage des pièces mécaniques

Installer les écrous M3 dans le cadre.

Les surfaces de contact des capteurs de charge doivent être plates et vrai, et peuvent nécessiter un petit ponçage.

Vérifier l'alignement global des deux jauges de charge pour que la balance soit précise.

IMPORTANT: Les 4 coussinets d'aile sont inclinés pour que la balance puisse être utilisée avec un planeur avec des ailes dièdres. L'articulation pour les coussinets doit être desserrée pour éviter toute liaison et frottements. Utiliser éventuellement un foret de 2 mm pour le trou permettant d’installer les coussinets s’ils sont trop serrés.

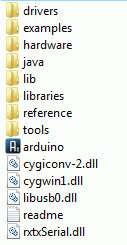
Fixez des tampons autocollants en feutrine (merci Leroy Merlin) sur les coussinets d'aile et sous le cadre si vous le souhaitez.

Mettre des photos pour chaque étape de montage.

# Installer l’environnement Arduino

Téléchargez la dernière version du logiciel Arduino sur cette page  <https://www.arduino.cc/download_handler.php>

Le téléchargement est au format compressé zip. Quand le téléchargement est terminé, dézippez le fichier téléchargé : vous obtenez ainsi un répertoire Arduino-00XX **dont il faut garder la structure**. Ouvrez ce dossier : vous devriez voir quelques fichiers et sous répertoires.



Repérer le fichier avec l'icône Arduino qui correspond au logiciel Arduino. A noter que ce répertoire est "portable" et peut-être mis où vous voulez sur votre ordinateur, voire même sur une clé USB ou un disque dur externe. Le logiciel Arduino s'exécutera sans problème.

## Connecter la carte Arduino à l'ordinateur.

A présent, connectez votre carte Arduino à votre ordinateur en utilisant votre programmateur USB. La LED (verte ou rouge en fonction de la carte Arduino) d'alimentation (notée PWR) doit s'allumer.

Mettre Photo de la carte avec le programmateur branché.

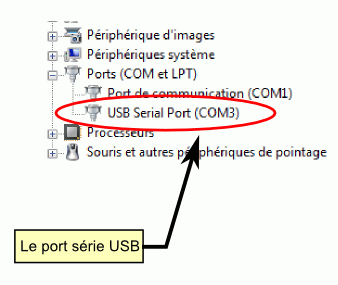
## Installer les drivers USB du port Série virtuel.

Quand vous connectez la carte à l'ordinateur pour la première fois, Windows devrait démarrer le processus d'installation du driver.

Sous Windows Vista et supérieur, le driver devrait normalement être installé automatiquement. Si ce n'est pas le cas :

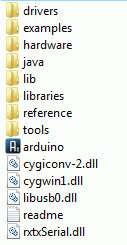
* Aller dans Démarrer > Panneau de Configuration > Système > Gestionnaire de périphérique
* Aller dans Port Com et LPT > Repérer le port USB série > Clic droit > Mettre à jour le pilote
* Sélectionner alors le pilote dans le sous-répertoire /drivers/FTDI\_USB du répertoire Arduino-00xx précédemment téléchargé.
* Valider les différentes étapes.

Vous pouvez à présent vérifier que les drivers ont bien été installés en ouvrant le Panneau de Configuration > Système > Gestionnaire de Périphériques. Vous devriez retrouver dans la section Ports LPT et COM un "USB Serial Port" : c'est le port USB de la carte Arduino. Noter au passage le numéro du port.

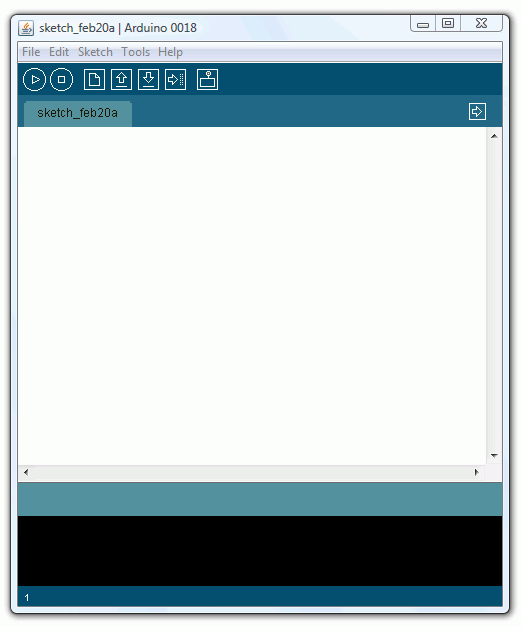


## Lancer le logiciel Arduino :

A présent, lancez le logiciel Arduino en double-cliquant 2 fois sur l'icône Arduino dans le répertoire téléchargé précédemment :

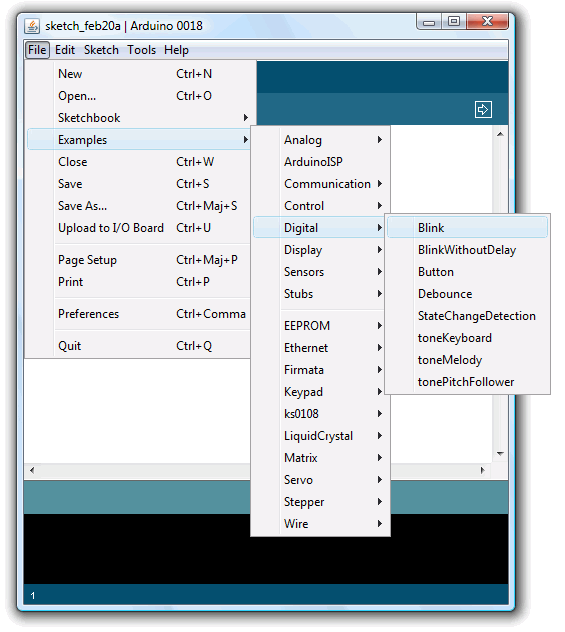


La fenêtre du logiciel doit s'ouvrir :

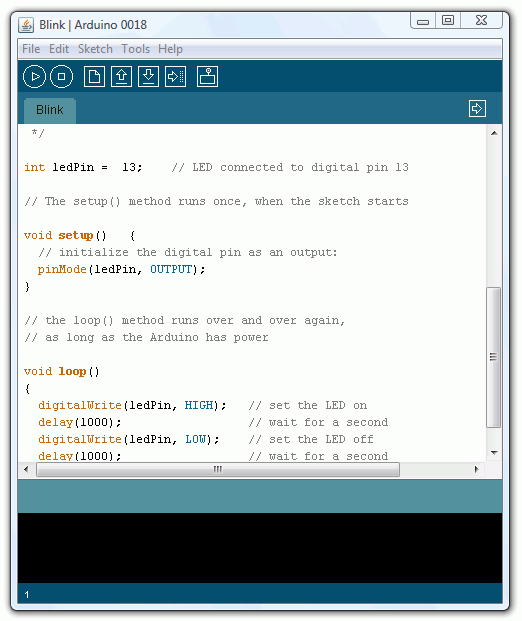


## Ouvrir le programme d'exemple Blink

Ouvrez à présent le programme d'exemple "Blink" qui fait clignoter la LED de la carte connectée à la broche 13. Pour se faire, aller dans le menu File > Examples > Digital > Blink

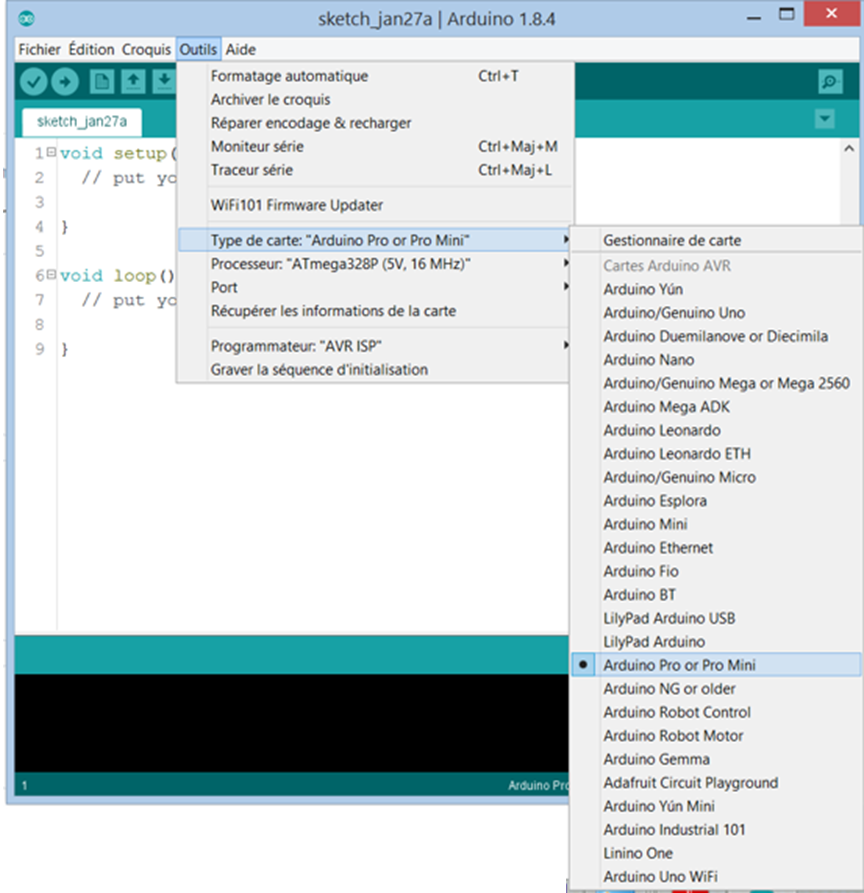


Le code du programme doit à présent apparaître dans la fenêtre de l'éditeur. Noter que vous auriez pu écrire vous-même un programme de test mais pour commencer, ce n'est pas le plus simple...



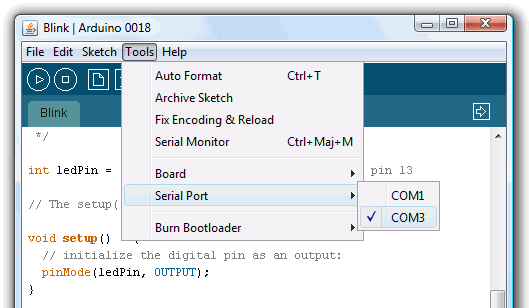
## Sélectionner votre carte Arduino

Vous devez à présent sélectionner votre carte Arduino dans le menu Tools>Board : pour les nouvelle cartes avec un ATmega328, sélectionner Arduino Pro or Pro Mini.



## 8. Sélectionner votre port série.

A présent, vous devez sélectionner le port série utilisée pour la communication avec la carte Arduino depuis le menu Tools>Serial Port (Outils>Port Série). Ce sera probablement le port COM 3 ou supérieur (les ports COM1 et COM2 sont habituellement réservés pour les ports série matériel). Pour trouver de quel port il s'agit, vous pouvez déconnecter votre carte Arduino et réouvrir le menu Tools>Serial Port (Outils>Port Série) : l'entrée qui a disparue est probablement celle du port de la carte Arduino. Reconnecter la carte, réouvrez le menu Tools>Serial Port (Outils>Port Série) et sélectionner le port série qui a du réapparaître :



## 9. Transférer le programme vers la carte Arduino

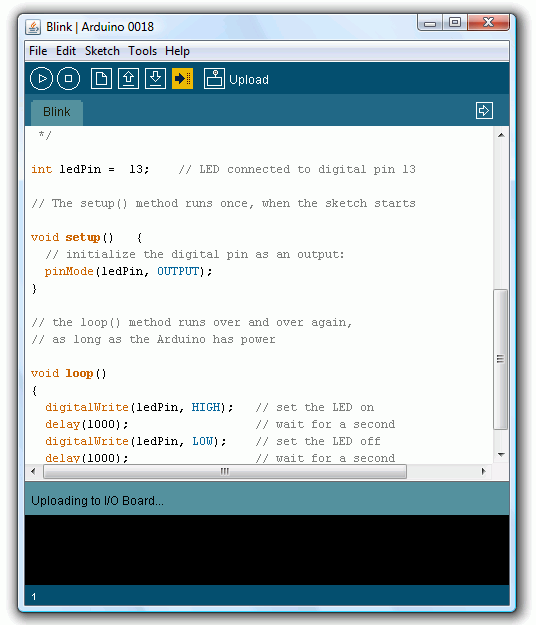
A présent, une fois que vous avez bien sélectionné le bon port série et la bonne carte Arduino, cliquez sur le bouton UPLOAD (Transfert vers la carte) dans la barre d'outils, ou bien sélectionner le menu File>Upload to I/O board (Fichier > Transférer vers la carte). Cliquer simplement sur le bouton "UPLOAD" de la barre d'outils du logiciel Arduino.

http://www.mon-club-elec.fr/mes_images/ide_arduino/bouton_6_50.gif

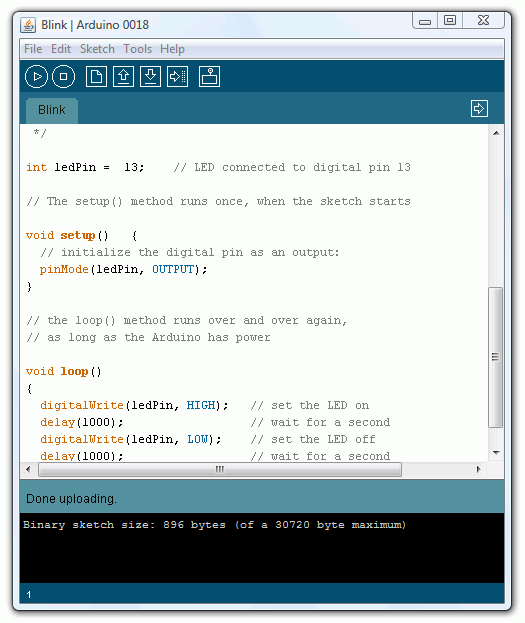
Vous devez appuyer sur le bouton "reset" de la carte juste avant de démarrer le transfert !

A ce moment précis : l'INSTANT MAGIQUE !!!

Vous devez voir les LEDs des lignes RX et TX clignoter rapidement, témoignant que le programme est bien transféré. Durant le transfert, le bouton devient jaune et le logiciel Arduino affiche un message indiquant que le transfert est en cours :



Une fois le transfert terminé, le logiciel Arduino doit afficher un message ("Done uploading")indiquant que le transfert est bien réalisé, ou montrer des messages d'erreurs...



Une fois le transfert terminé, le bootloader est actif une petite seconde ("écoute" pour voir si un nouveau programme arrive...) une fois que la carte est réinitialisée à la fin du transfert; puis le dernier programme programmé dans la carte s'exécute. Vous devriez donc voir la LED sur la broche 13 clignoter (couleur orange).

Si c'est bien le cas, félicitations ! Vous avez réussi à programmer et à faire fonctionner votre carte Arduino.

# Installer les bibliothèques supplémentaires

Deux librairies ou trois librairies (en fonction de la version que vous souhaitez construire) sont à installer pour pouvoir compiler le fichier .ino qui sera téléchargé dans la carte arduino :

* La librairie HX711\_ADC, qui permet de s’interfacer avec les … HX711,
* Si vous construisez la version OLED I2C :
  + Adafruit\_SSD1306
  + Adafruit\_GFX
* Si vous construisez la version 1602 I2C :
  + LiquidCrystal\_I2C
* Si vous construisez la version 1602 de base :
  + LiquidCrystal

Une bibliothèque est un ensemble de fonctions qui s’ajoutent aux fonctions de base du logiciel de l’Arduino et qui adressent plus particulièrement un domaine spécifique de la programmation comme, par exemple :

* EEPROM - lire et écrire dans cette zone de mémoire permanente du micro-contrôleur ;
* Ethernet - accéder à un réseau TCP/[IP](http://www.locoduino.org/spip.php?mot58) comme Internet, mais ça peut être aussi un réseau local à votre domicile, avec l’aide de la carte Ethernet ;
* LiquidCrystal - utiliser un écran LCD ;
* SD - lire et écrire une carte SD de technologie flash et formatée en FAT16 ou FAT32, sauf reformatage de votre part, toutes les cartes SD le sont ;
* Servo - piloter des servomécanismes ;
* WiFi - se connecter à un réseau sans fil à partir d’une carte WiFi ;
* Wire - Communiquer entre Arduinos et avec des périphériques capteurs, actionneurs avec le protocole I2C.

Ces bibliothèques sont [installées en même temps que l’IDE](http://www.locoduino.org/spip.php?article15). Vous n’aurez pas à les ajouter.

Mais surtout, il existe sur le net des multitudes de bibliothèques dans pratiquement tous les domaines, ce qui donne à l’Arduino cette puissance incontestable et son succès. Certaines sont même décrites dans LOCODUINO, en particulier celles qui concerne le modélisme ferroviaire.

Cet article concerne donc l’installation de librairies supplémentaires.

**Bibliothèque ou Librairie ?**

Avant de commencer, revenons sur un point de traduction. Arduino est un projet où l’anglais prédomine. *Library* est normalement traduit littéralement en français par *bibliothèque*. Cependant l’anglicisme a pris le dessus et vous trouverez beaucoup de pages en français où on parle de *librairie* Arduino.

Nous allons voir dans cet article comment installer une bibliothèque qui n’est pas de base dans l’IDE Arduino.

**Où les trouver ?**

Comme Arduino est un projet communautaire, vous trouverez des bibliothèques sur le web en général.  
Quelques bibliothèques sont recensées sur le site officiel sur cette [page](http://www.arduino.cc/en/Reference/Libraries). Si vous ne trouvez pas votre bonheur, une recherche avec le type de [shield](http://www.locoduino.org/spip.php?mot23) ou de composant accompagné de **Arduino** et **library** vous donnera les pointeurs souhaités.

**Où les installer ?**

Revenons sur un point de la structure d’un ordinateur. Quel que soit son environnement de travail (Windows, [Mac](http://www.locoduino.org/spip.php?mot57) OS X ou GNU/Linux), il y a un répertoire privé où vous mettez vos documents auxquels d’autres utilisateurs de l’ordinateur n’ont pas forcément accès, et un répertoire accessible par toute personne possédant une lecture sur les répertoires de programmes.

Dans votre répertoire personnel, après installation de l’IDE Arduino, vous avez un répertoire sketchbook (GNU/Linux) ou Documents/Arduino ([Mac](http://www.locoduino.org/spip.php?mot57)) ou Mes Documents/Arduino (Windows) qui est créé. À l’intérieur, vous avez un répertoire pour chaque programme que vous avez enregistré, ainsi qu’un répertoire particulier nommé *libraries* (si il n’y est pas, vous pourrez le créer si vous choisissez cet emplacement, mais nommez le bien ’libraries’ en Anglais !).  
Si la librairie est installée dans ce répertoire, seul l’utilisateur du répertoire y aura accès dans son programme. Si vous êtes seul sur l’ordinateur, c’est la méthode la plus facile.

Vous voyez d’ailleurs que les bibliothèques officielles ne sont pas dans celui-ci. Afin que chaque utilisateur de l’ordinateur y ait accès, elles sont installées dans un répertoire qui diffère du précédent et même selon les ordinateurs :

* sous GNU/Linux, dans /usr/share/arduino/libraries/ ;
* sous Mac OS X, dans le *bundle* de l’application : Arduino/Contents/Resources/Java/libraries [[1](http://www.locoduino.org/spip.php?article71#nb1)] ;
* sous Windows 32 bits, dans C :\Program Files\Arduino\libraries ;
* sous Windows 64 bits, dans C :\Program Files (x86)\Arduino\libraries ;

Vous pourrez faire une installation ici si vous en avez les droits pour faire profiter tous les utilisateurs de l’ordinateur de cette bibliothèque. Toutefois, ce n’est pas une procédure prévue dans l’IDE Arduino et c’est inutile si vous êtes le seul à programmer sur Arduino avec votre ordinateur.

**Plusieurs méthodes d’installation**

Nous allons prendre un exemple :  
la bibliothèque Tone qui permet de faire jouer un buzzer en sortant des sons, vous la trouverez [ici](https://code.google.com/p/rogue-code/wiki/ToneLibraryDocumentation), téléchargez-la pour essayer.

**Méthode facile**  
Dans l’IDE Arduino, vous avez le menu Croquis/Importer bibliothèque…/add library (Et non, ce n’est pas traduit…) en version 1.0.6, ou Croquis/Inclure une bibliothèque/Ajouter la bibliothèque .ZIP… dans la version 1.6.7 et suivantes. Vous sélectionnez l’archive de la bibliothèque et vous cliquez sur OK.  
Elle est installée.

Par exemple sous Windows, vous aurez un répertoire Mes Documents/Arduino/libraries/Tone qui contient le fichier tone.h entre autres…  
Sous Mac le dossier est dans Documents/Arduino/libraries

**Autre méthode**  
Décompressez votre archive, vous trouverez un répertoire du nom de la bibliothèque qui contient les fichiers nécessaires pour l’exécution dans un code. Vous copiez ce répertoire dans le dossier que vous préférez, soit votre home si vous êtes seul sur le PC sinon le répertoire d’installation du programme que nous avons vu précédemment pour que tous les utilisateurs profitent de la bibliothèque, à coté des bibliothèques officielles.  
Par exemple sous Windows, vous aurez un répertoire c :/Program Files (x86)/Arduino/libraries/Tone qui contient le fichier tone.h entre autres…  
Attention au nom de la bibliothèque qui peut être différent dans le Zip, en particulier s’il a été téléchargé sur une ’forge’ comme [framasoft](https://git.framasoft.org/groups/locoduino.org), [GitHub](https://github.com/) ou [Sourceforge](https://sourceforge.net/). Ces environnements de travail collaboratifs ajoutent le nom de la branche (la version de travail) à celui du projet, ce qui donnerait ’tone-master’ pour Tone. Enlevez bien tout ce qui n’est pas le nom la bibliothèque sinon elle ne sera pas reconnue. Pour savoir à coup sûr quel nom utiliser, le répertoire doit porter le nom du fichier .h qui sera inclus : Tone.h -> Tone !

Redémarrez l’IDE pour profiter de la bibliothèque.

**Depuis l’IDE 1.6.0**

Depuis cette version de l’IDE Arduino, un Gestionnaire de bibliothèques est disponible par la commande Croquis/Inclure une bibliothèque/Gérer les bibliothèques… Il permet de retrouver une liste détaillée des bibliothèques présentes, de les mettre à jour lorsqu’une évolution a été détectée chez son créateur, et d’en installer de nouvelles proposées automatiquement. Cette liste peut être triée par thème, ou par état (installées, à mettre à jour…) .

**Désinstallation**

Supprimez simplement le répertoire de la bibliothèque et redémarrez l’IDE pour que ça soit pris en compte

# Intégrer l’électronique dans la balance

Mettre des photos à chaque étape de l’intégration.

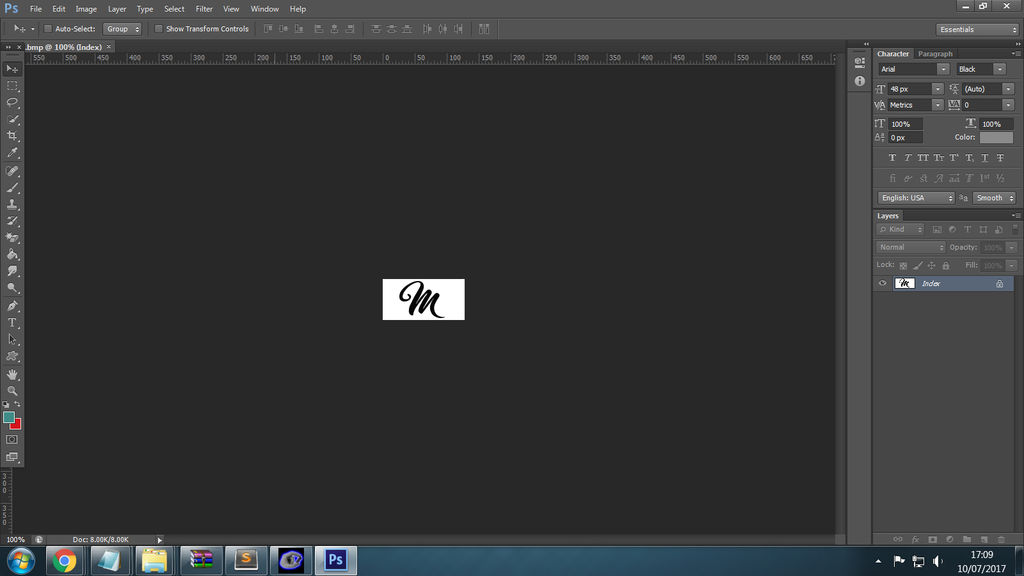
# Télécharger le logiciel et calibrer les ADC

A compléter avec la procédure.

# BONUS : Modifier le logo qui apparaît sur l’écran à l’initialisation de la balance.

A traduire et contextualisé avec le logo de l’ANEG.

## Step 1: Create Your Logo Graphic

[](https://cdn.instructables.com/FVN/9UB2/J4SQHGW9/FVN9UB2J4SQHGW9.LARGE.jpg)

Use the graphics editor of your choice to create a 8 bit bitmap of your logo in this instance making the size 128 x 64 or whatever size your OLED or LCD is.

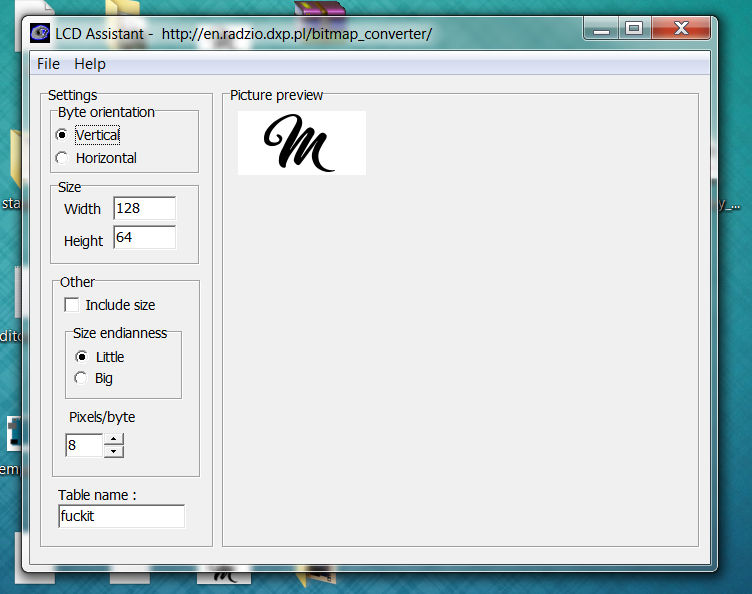
It must be saved a a 8 bit bitmap as no other file format or type will work.

Also make sure the background is actually white (#ffffff) and the foreground (the logo or image) is black (#000000) these color steps are important as the software we use to convert the graphic to code won't work properly if these two color values are not used.

Save the bitmap and move onto converting it into code

Add TipAsk Question

## Step 2: Use LCD Assistant to Convert Your Bitmap to Code

[](https://cdn.instructables.com/FDV/E046/J4SQHGVU/FDVE046J4SQHGVU.LARGE.jpg)

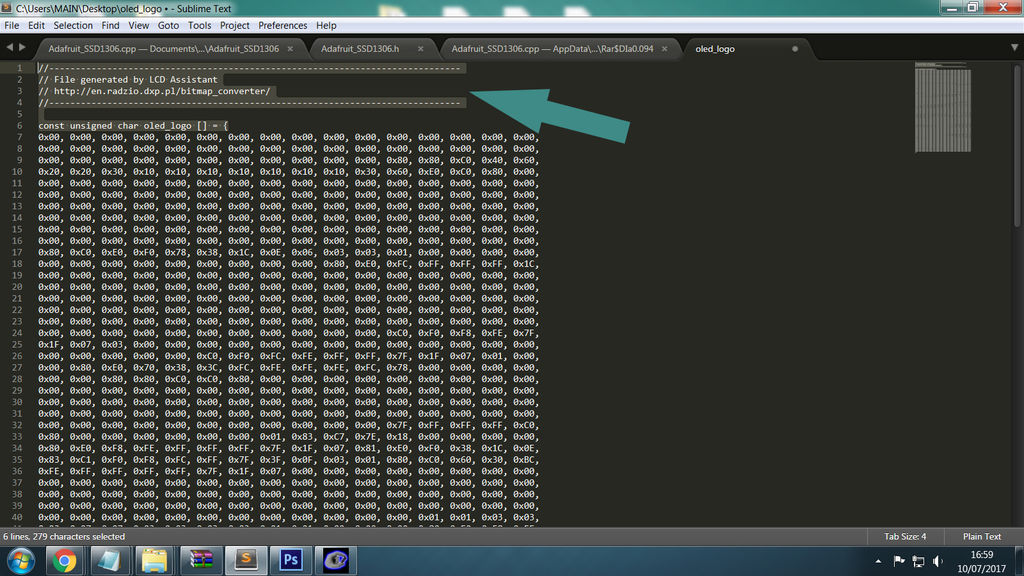
Download the free bitmap to code converter called LCD Assistant the link for it is [HERE](http://en.radzio.dxp.pl/bitmap_converter/)

Open the program and then from the file menu load in your bitmap graphic. LCD Assistant should automatically detect the size of the graphic in this case 128 x 64 and for every other setting leave everything else as is.

Then again from the file menu select output and save the output file to somewhere handy on your computer such as the Desktop for example.

Add TipAsk Question

## Step 3: Edit Logo Hex Code Output File

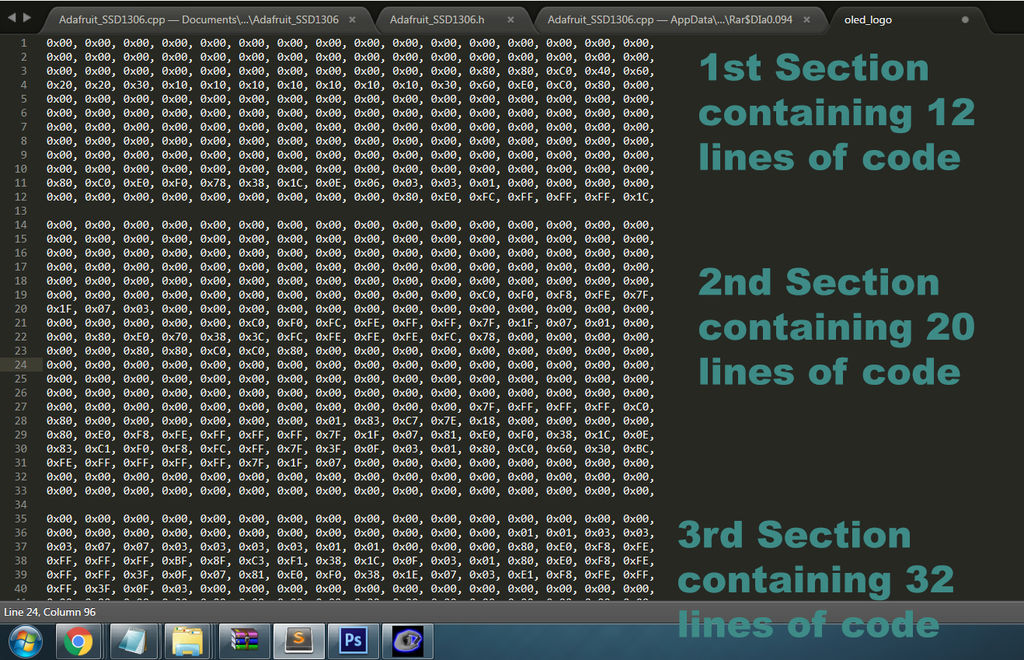
[](https://cdn.instructables.com/FIQ/K9SE/J4SQHGQO/FIQK9SEJ4SQHGQO.LARGE.jpg)

Open the file generated by LCD Assistant in any text editor even Notepad will do. Here I am using the excellent Sublime Text.

You want to delete everything apart from the actual hex code so remove the part at the top of the file so the first line should just be code and at the bottom delete the ending " }; " symbols.

Add TipAsk Question

## Step 4: Break Logo Code Into Three Sections

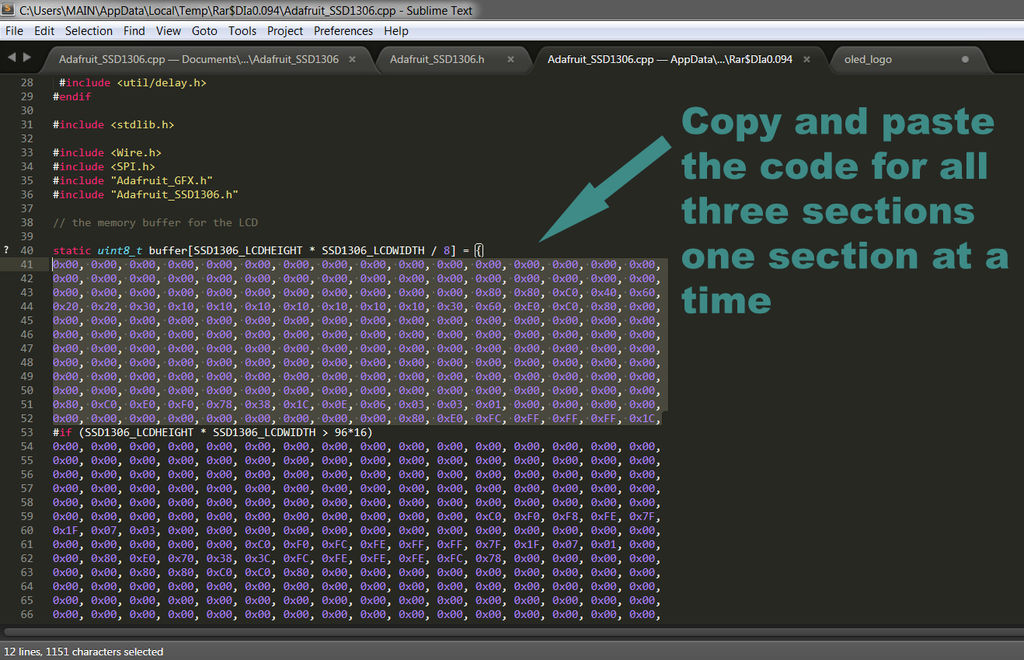
[](https://cdn.instructables.com/FMK/OD1A/J4SQHGX6/FMKOD1AJ4SQHGX6.LARGE.jpg)

In this step we just want to break the logo hex code into three sections so we end up with three separate sections. The first of which will have 12 lines of code, the second 20 lines of code and the third secton 32 lines of code to make a total of 64 lines of code which is the same height as your OLED... 64. (If you are using a different size then you will have more or less lines of code to match the height of your display)

All you do to separate these sections is to add a blank line in between them.

Add TipAsk Question

## Step 5: Copy Your Logo Code to Replace the Adafruit Logo Code

[](https://cdn.instructables.com/FNU/REO2/J4SQHGRH/FNUREO2J4SQHGRH.LARGE.jpg)

For this step you need to already have the Adafruit\_SSD1306 library downloaded and installed. If you are using the Arduino web ide rather than the standard version that runs on your computer you will need to add the library once customized as a custom library because you are unable to edit the library using the web / cloud version and be sure to include the custom version into your sketch and not include the pre installed version or this won't work.

Inside the Adafruit\_SSD1306 library open the file named Adafruit\_SSD1306.cpp using a text editor. Scroll down into you find the following section starting with this line:

static uint8\_t buffer[SSD1306\_LCDHEIGHT \* SSD1306\_LCDWIDTH / 8] = {

You will then copy and paste replacing the code with your new logo code into each section:

static uint8\_t buffer[SSD1306\_LCDHEIGHT \* SSD1306\_LCDWIDTH / 8] = {

CODE SECTION ONE GOES HERE

#if (SSD1306\_LCDHEIGHT \* SSD1306\_LCDWIDTH > 96\*16)

CODE SECTION TWO GOES HERE

#if (SSD1306\_LCDHEIGHT == 64)

CODE SECTION THREE GOES HERE

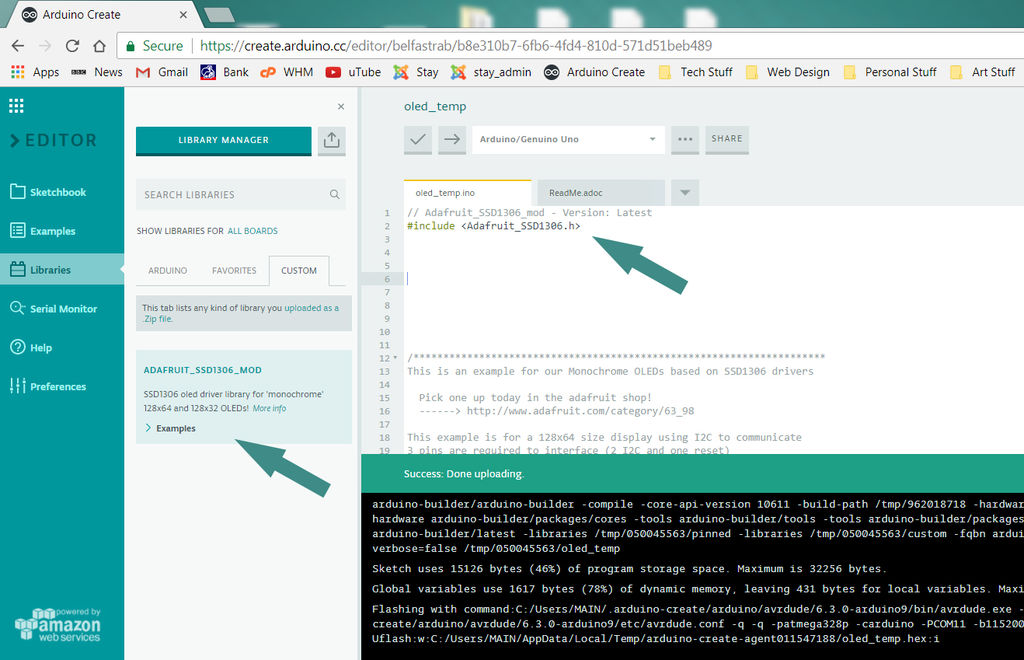
#endif #endif };

Leave everything exactly as it is being careful to only replace the code in each section and nothing else otherwise it won't work.

Once all that is done save the file.

Add TipAsk Question

## Step 6: Use Modified Library in Your Sketch

[](https://cdn.instructables.com/FKE/JREZ/J4YFMSIH/FKEJREZJ4YFMSIH.LARGE.jpg)

Use any sketch that uses the Adafruit\_SSD1306 library there is loads out there but for the sake of completion you can use the one provided in this excellent Instructable by Bay Yoyal found[here](https://www.instructables.com/id/The-First-Usage-of-096-I2C-OLED-Display-With-Ardui/) but whatever sketch you use it will need to make use of the splash screen on first boot / turn on of OLED as not all sketches will do that.

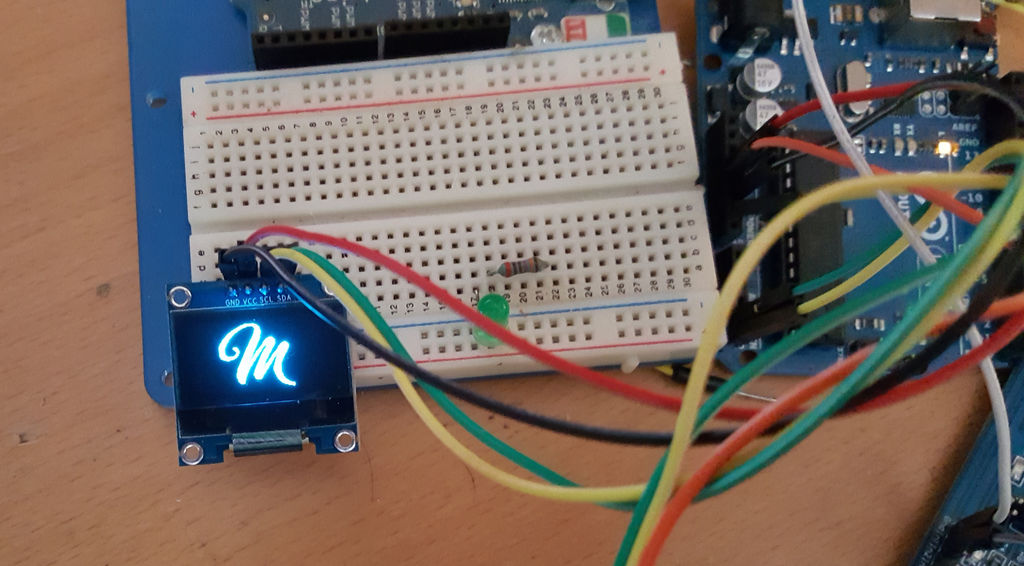
If using the computer / software version of the Arduino IDE you should be good to go assuming you only have one version of the Adafruit\_SSD1306 library on your system.

For the web / cloud version of the Arduino IDE that I use you will need to put all the files within the main Adafruit\_SS1306 library directory / folder into a zip file and name it whatever you like. Then using the ide navigate to libraries and add this new modified library as a custom version. Once the new custom library has been added there is a button beside it which says include which will add it your sketch. (PLEASE NOTE: double check the sketch to make sure you are only including the library once)

Verify and upload your sketch to your Arduino.

Add TipAsk Question

## Step 7: Enjoy Your Custom Logo

[](https://cdn.instructables.com/FG5/TZ4A/J4YFMWKI/FG5TZ4AJ4YFMWKI.LARGE.jpg)

That's it all done. Assuming everything has been done right your logo will now replace the Adafruit Industries one.

I hope you liked this Instructable and sorry if it was not written very well it's my first one