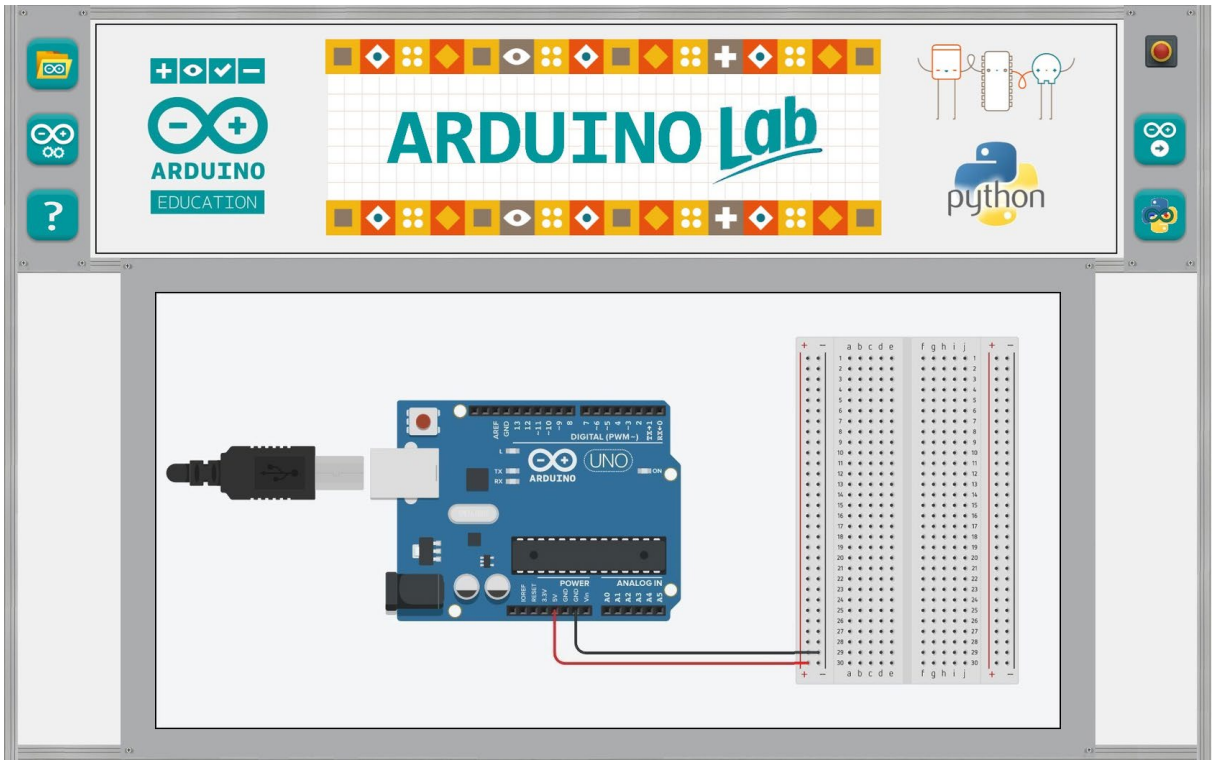
. ARDUINO LAB, Qu'est-ce que c'est ?



Fenêtre d'accueil d'ARDUINO LAB

ARDUINO LAB est un logiciel programmé en Python 3 dans un but pédagogique pour permettre une découverte rapide et simple de l'**ARDUINO UNO** et de son utilisation dans le domaine des sciences.

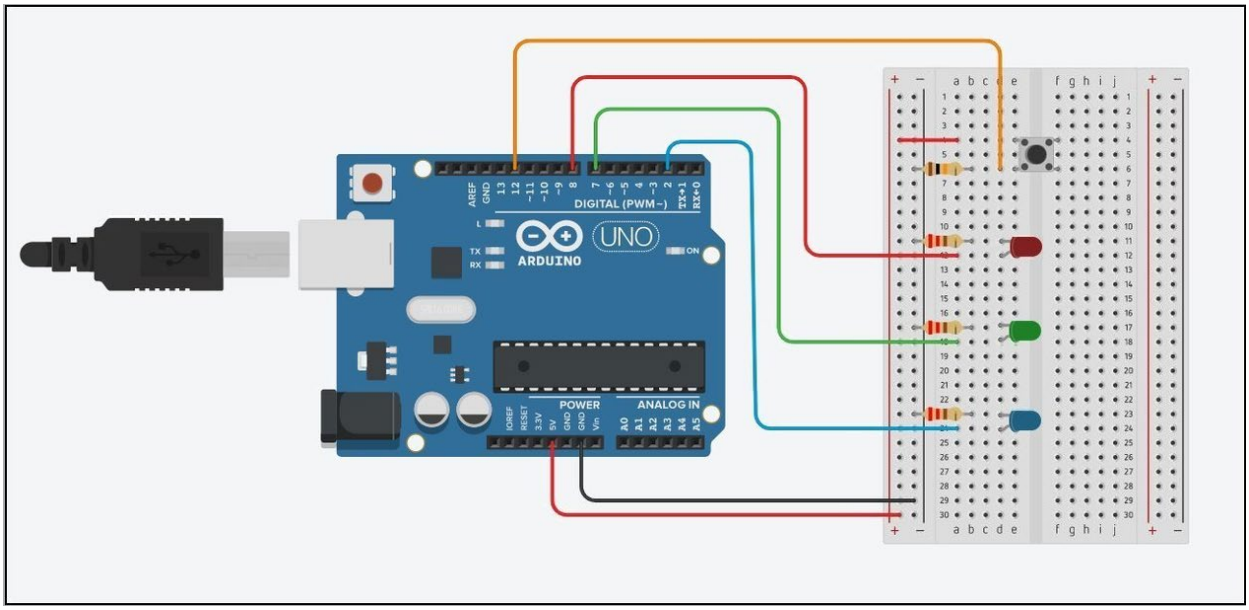
Si les cartes **Arduino** sont, de par leur conception, destinées à travailler de manière autonome, il est cependant intéressant de les utiliser comme interface physique sur un ordinateur pour piloter directement des matériels ou récupérer des informations issues de capteurs, à des fins de traitement et d'exploitations. Les données, qu'elles soient des commandes ou des informations, transiteront par la connexion USB.

Pour cela, deux programmes sont nécessaires :

. Un programme ”donneur d’ordre” sur l’ordinateur,

. et un ”pilote”, animant le microcontrôleur, qui comme son nom l’indique, pilotera les matériels en réponse aux ordres reçus et fournira des données en retour.

ARDUINO LAB est le programme ”donneur d’ordre” qui permet de contrôler l’ARDUINO graphiquement, que ce soit en entrée ou en sortie par l’intermédiaire de circuits électroniques déjà conçus :



Exemple de circuits étudiés dans ARDUINO LAB

La liaison entre **ARDUINO LAB** et l’Arduino fonctionne dans les deux sens. Toute interaction sur l’Arduino (par exemple, l’appui sur un bouton poussoir) est visible sur l’interface graphique **d’ARDUINO LAB**.

Un mode ”Simulation” permet de tester les circuits avant d’effectuer la liaison avec l’Arduino.

Avec **ARDUINO LAB**, Il est également possible de contrôler chaque broche de la platine Arduino pour, par exemple, avec une entrée reliée à un potentiomètre, piloter l’intensité d’une DEL ou l’angle d’un servomoteur…

Enfin, **ARDUINO LAB** permet aussi d’enregistrer les données d’un capteur et d’en effectuer leurs exploitations, fonctionnalité très intéressante dans le domaine des sciences.

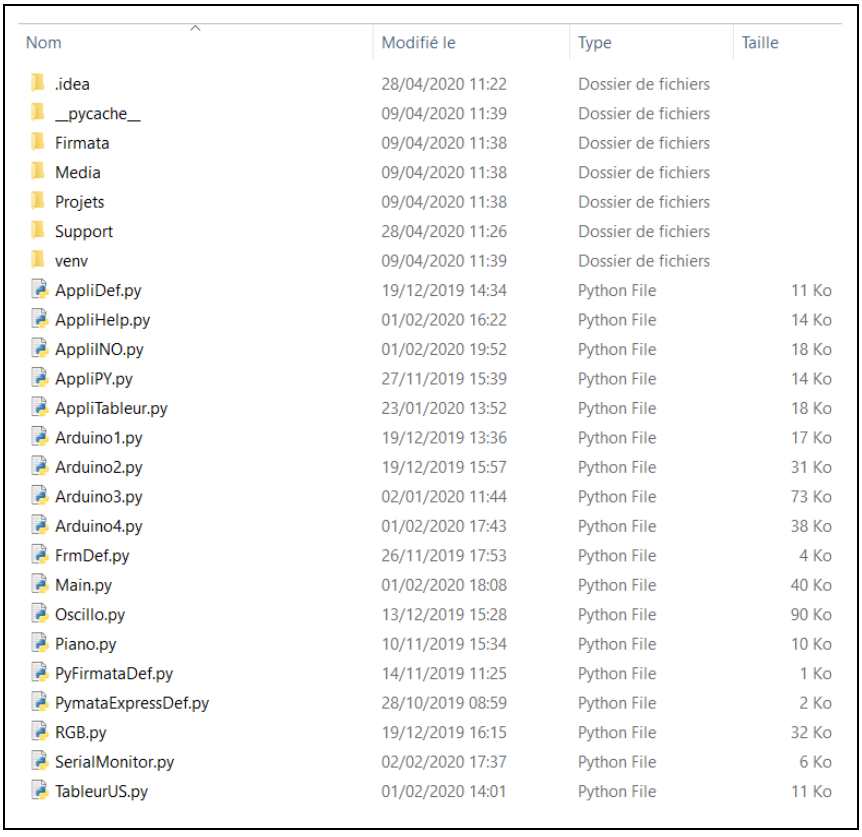
1. Installation d'ARDUINO LAB

Python étant un langage de programmation interprété, pour faire fonctionner **ARDUINO LAB**, il faut qu'un interpréteur **Python** soit installé.

ARDUINO LAB nécessite au minimum la version 3.7 de **Python** et l'installation des bibliothèques suivantes :

- **pyfirmata** (pour contrôler l'Arduino selon le protocole **Firmata Standard**)
- **pymata-express** (pour contrôler l'Arduino selon le protocole **Firmata Express)**
- **asyncio** (pour la programmation asynchrone)
- **matplotlib** (pour tracer et visualiser des données sous formes de graphiques)
- **numpy** (pour le calcul scientifique)
- **Pillow** (pour le traitement des images)
- **scipy** (pour le traitement des données)
- **PyMuPDF** (pour l'affichage des documents PDF)

Une fois l'environnement de travail configuré et après avoir téléchargé puis décompressé le fichier **"ArduinoLab.zip"** qui contient tous les fichiers et dossiers nécessaire à son fonctionnement, **ARDUINO LAB** est démarré à l'aide du fichier **"Main.py"** situé dans le dossier principal **"ArduinoLab"** du programme :



Nom	Modifié le	Type	Taille
.idea	28/04/2020 11:22	Dossier de fichiers	
__pycache__	09/04/2020 11:39	Dossier de fichiers	
Firmata	09/04/2020 11:38	Dossier de fichiers	
Media	09/04/2020 11:38	Dossier de fichiers	
Projets	09/04/2020 11:38	Dossier de fichiers	
Support	28/04/2020 11:26	Dossier de fichiers	
venv	09/04/2020 11:39	Dossier de fichiers	
AppliDef.py	19/12/2019 14:34	Python File	11 Ko
AppliHelp.py	01/02/2020 16:22	Python File	14 Ko
AppliINO.py	01/02/2020 19:52	Python File	18 Ko
AppliPY.py	27/11/2019 15:39	Python File	14 Ko
AppliTableur.py	23/01/2020 13:52	Python File	18 Ko
Arduino1.py	19/12/2019 13:36	Python File	17 Ko
Arduino2.py	19/12/2019 15:57	Python File	31 Ko
Arduino3.py	02/01/2020 11:44	Python File	73 Ko
Arduino4.py	01/02/2020 17:43	Python File	38 Ko
FrmDef.py	26/11/2019 17:53	Python File	4 Ko
Main.py	01/02/2020 18:08	Python File	40 Ko
Oscillo.py	13/12/2019 15:28	Python File	90 Ko
Piano.py	10/11/2019 15:34	Python File	10 Ko
PyFirmataDef.py	14/11/2019 11:25	Python File	1 Ko
PymataExpressDef.py	28/10/2019 08:59	Python File	2 Ko
RGB.py	19/12/2019 16:15	Python File	32 Ko
SerialMonitor.py	02/02/2020 17:37	Python File	6 Ko
TableurUS.py	01/02/2020 14:01	Python File	11 Ko

Attention :

L'emplacement du dossier décompressé **"ArduinoLab"** n'a pas d'importance, mais tous les fichiers et dossiers contenus dans ce dossier ne doivent en aucun cas être modifiés ou déplacés.

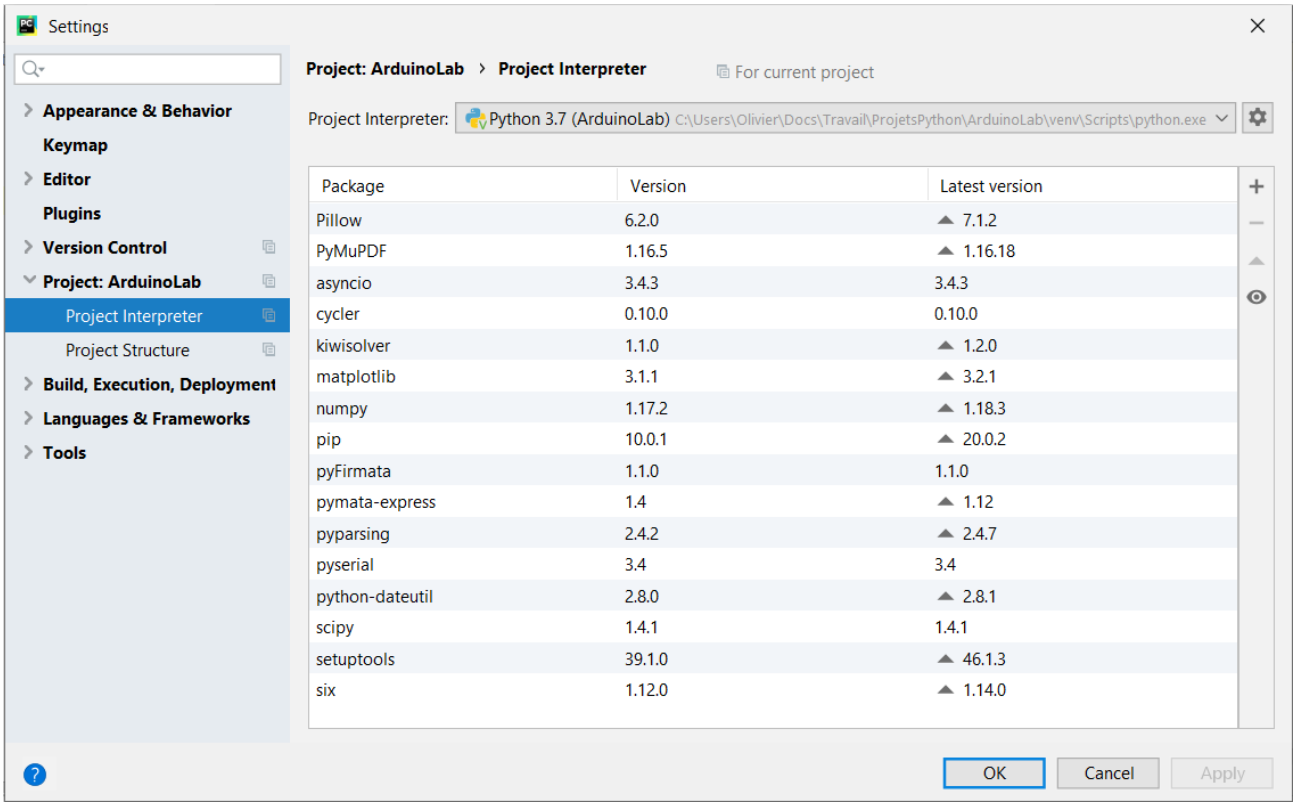
Remarques :

Le dossier principal **"ArduinoLab"** du programme contient un dossier nommé **"venv"** dans lequel se situe un environnement virtuel de programmation avec les bibliothèques indispensables citées ci-dessus.

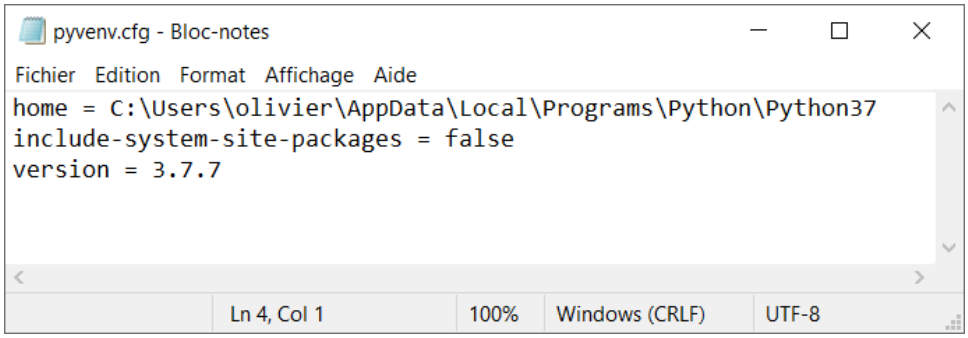
Cet environnement de programmation peut être utilisé comme interpréteur du programme.
Dans ce cas, même si l'installation d'une distribution **Python** (3.7 au minimum) est indispensable, les bibliothèques dont **ARDUINO LAB** dépendent n'auront pas à être ajoutées à la distribution originale installée.

L'utilisation de l'environnement de programmation virtuel pour le fonctionnement d'**ARDUINO LAB** se configure par l'intermédiaire d'un environnement de développement Python (IDE), par exemple, **PyCharm**.

Ainsi dans **PyCharm**, après avoir ouvert le dossier d'**ARDUINO LAB**, il suffit d'indiquer dans les réglages que l'environnement virtuel est l'interpréteur du projet **:**



Il faudra cependant au préalable modifier le fichier ”**pyvenv.cfg**” situé dans le dossier **”ArduinoLab/venv/”** pour indiquer le chemin d’installation de la distribution Python :



2. Pré-requis au fonctionnement d'ARDUINO LAB

Pour fonctionner, **ARDUINO LAB,** nécessite l’installation d’un programme "pilote" dans la mémoire de l’Arduino (sauf en mode "Simulation" bien-sûr) pour envoyer des ordres à l’Arduino ou recevoir des données via le port USB.

Suivant les circuits étudiés ou les capteurs utilisés, **ARDUINO LAB** utilise deux protocoles de communication avec l’Arduino différents, "**Firmata standard**" ou "**Firmata express**".

Le chargement, dans la mémoire de l'Arduino, du code "pilote" doit être fait soit manuellement en utilisant le logiciel "**IDE ARDUINO**", soit par l'intermédiaire du menu "**Paramètres**" d'**ARDUINO LAB**. Un mode de chargement automatique du protocole de communication en fonction du circuit étudié est également disponible dans ce menu.

Quoi qu’il en soit, pour utiliser toutes les fonctionnalités d’**ARDUINO LAB,** notamment le téléversement de codes écrits en langage "Arduino", il est nécessaire d’installer, sur l’ordinateur utilisé pour contrôler l’Arduino, le logiciel "**IDE ARDUINO**" qui est disponible à l’adresse suivante :

https://www.arduino.cc/en/Main/Software

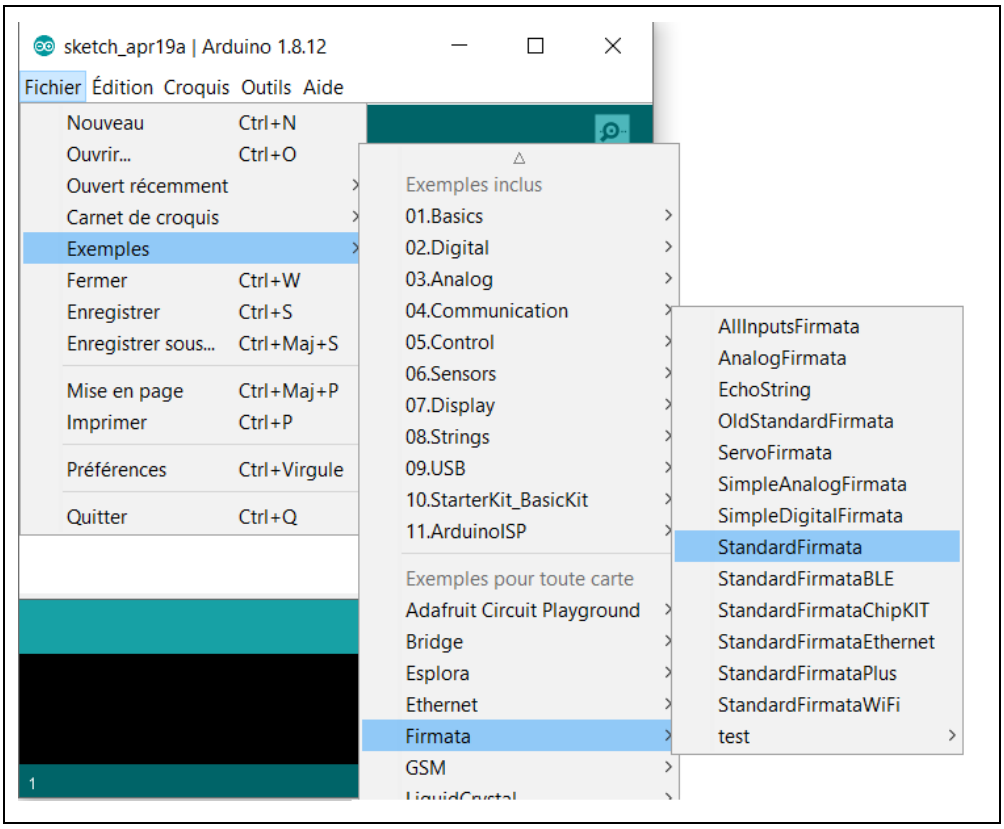
. **Chargement manuel du code "Firmata Standard"** :

- Brancher l’Arduino via un port USB,

- Afin de charger la librairie "**Firmata standard**" sur l’ARDUINO, il faut lancer le logiciel "**IDE ARDUINO**", puis sélectionner :

Fichier > Exemples > Firmata > Standard Firmata,

- puis cliquer sur "**téléverser**".

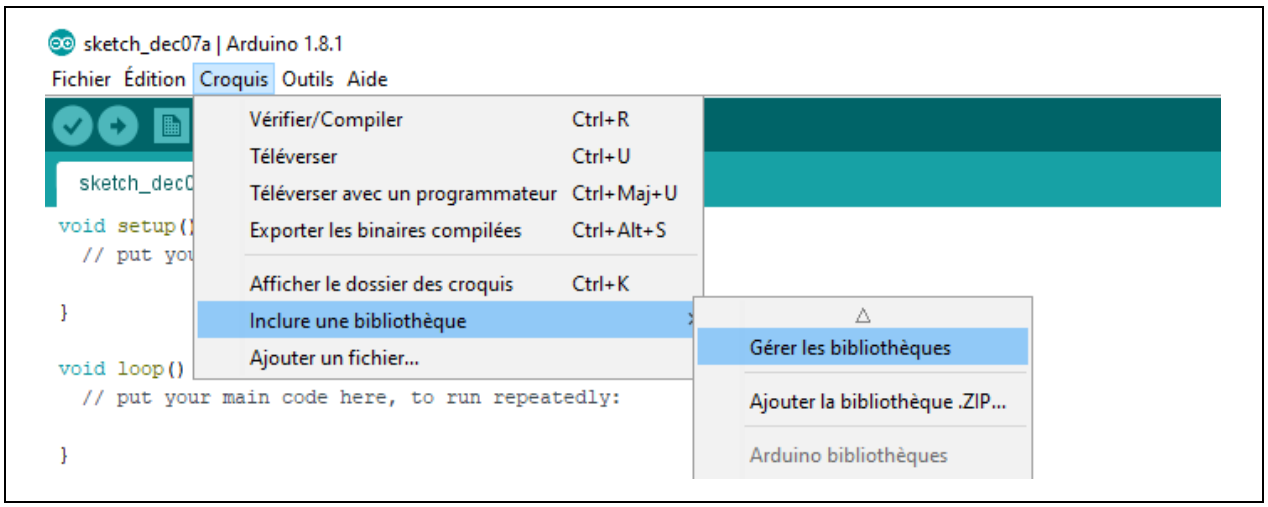


. **<u>Chargement manuel du code "Firmata Express"</u>** :

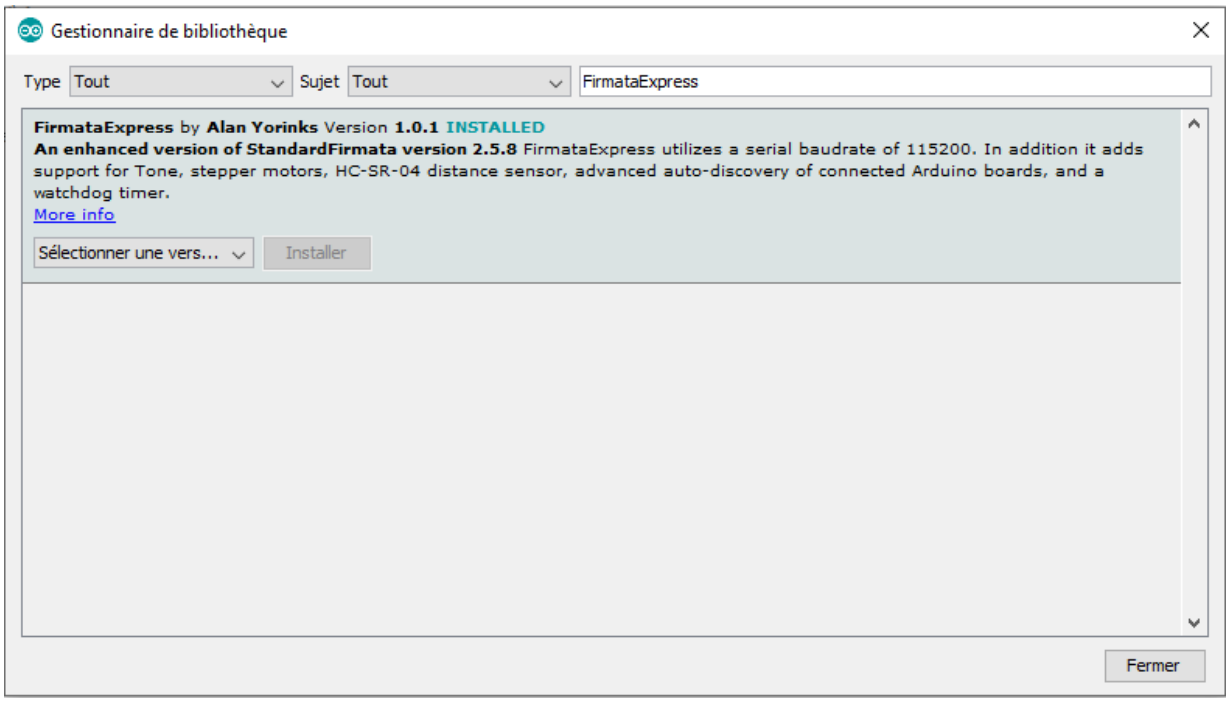
Par défaut, la bibliothèque **"Firmata Express"** n'est pas incluse dans l'"**IDE Arduino".** Il faut donc procéder à son installation :

- Ouvrir le logiciel "**IDE ARDUINO**",

- Sélectionner "**Croquis/Inclure une bibliothèque/Gérer les bibliothèques**",



- Entrer "**FirmataExpress**" dans la zone de recherche :



- et cliquer sur "**installer**".

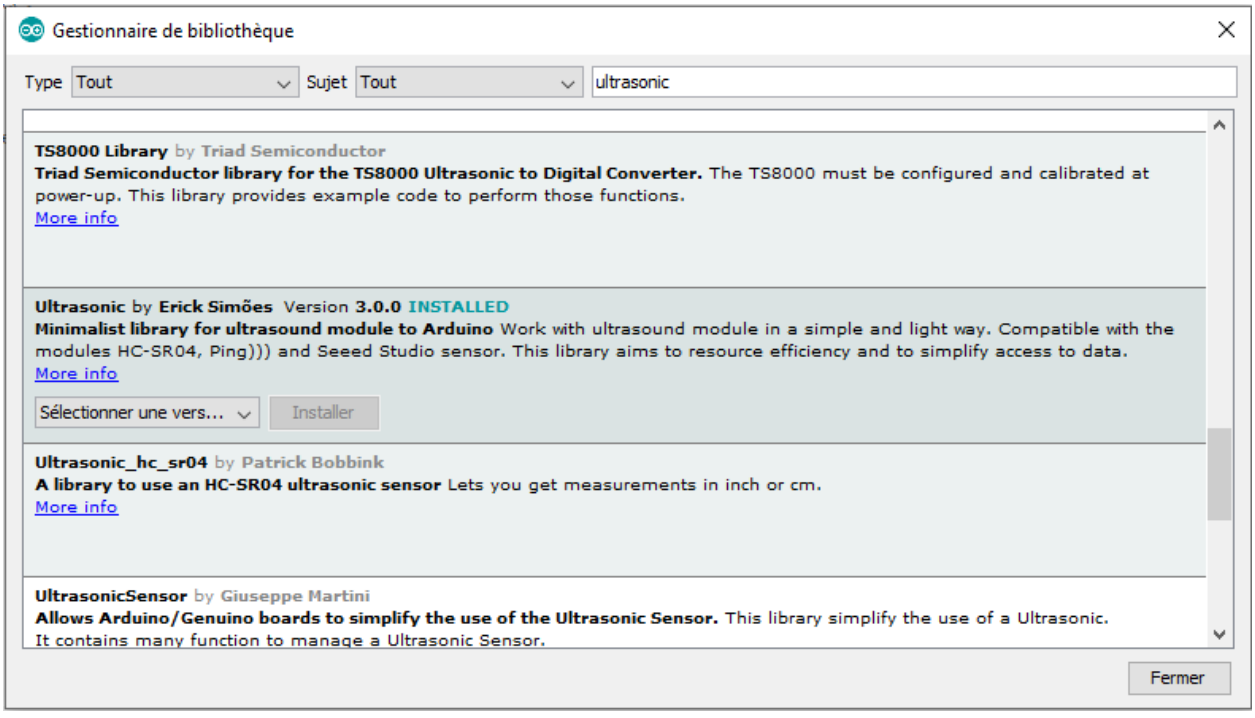
"**FirmataExpress**" nécessite également que la librairie "**Ultrasonic by Erick Simões**" soit installée.

De même que précédemment, en utilisant le logiciel Arduino IDE :

- Sélectionner "**Croquis/Inclure une bibliothèque/Gérer les bibliothèques**",

- Entrer "**Ultrasonic**" dans la zone de recherche,

- Sélectionner "**Ultrasonic by Erick Simões**"



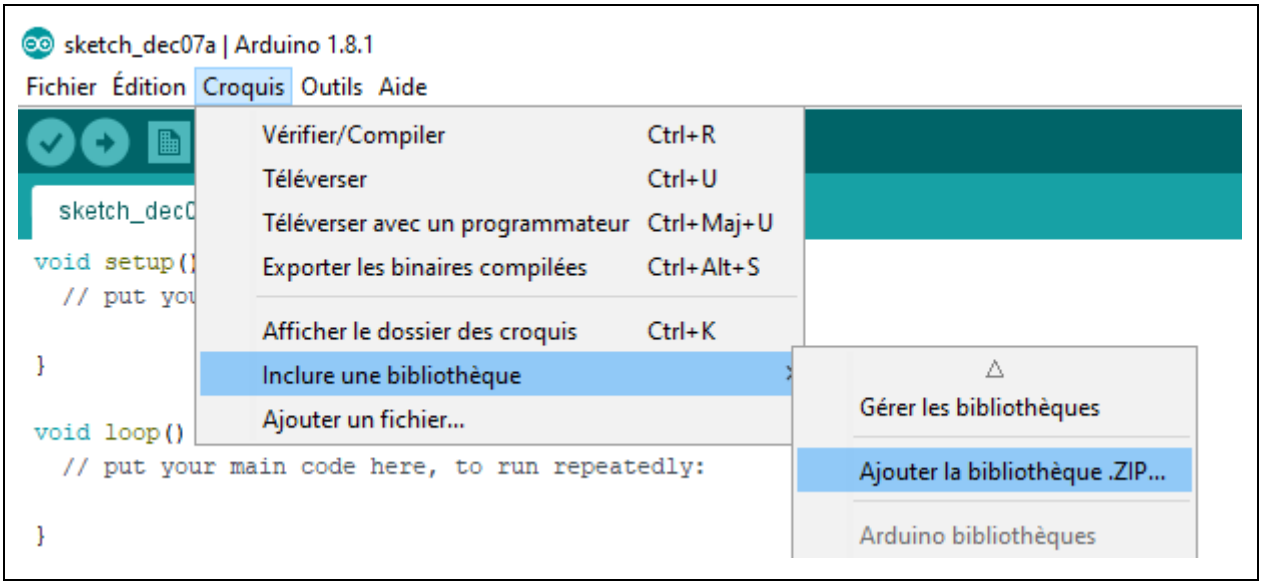
- et cliquer sur "**installer**".

Sans connexion internet, les librairies peuvent être installées à partir des fichiers "**zip**" présents dans le dossier "**Support/Librairies Arduino**" du répertoire d'installation d'**ARDUINO LAB** :

- Ouvrir le logiciel "**IDE ARDUINO**",

- Sélectionner "**Croquis/Inclure une bibliothèque/Ajouter la bibliothèque .ZIP**",

- Sélectionner le fichier .ZIP de la librairie à installer.



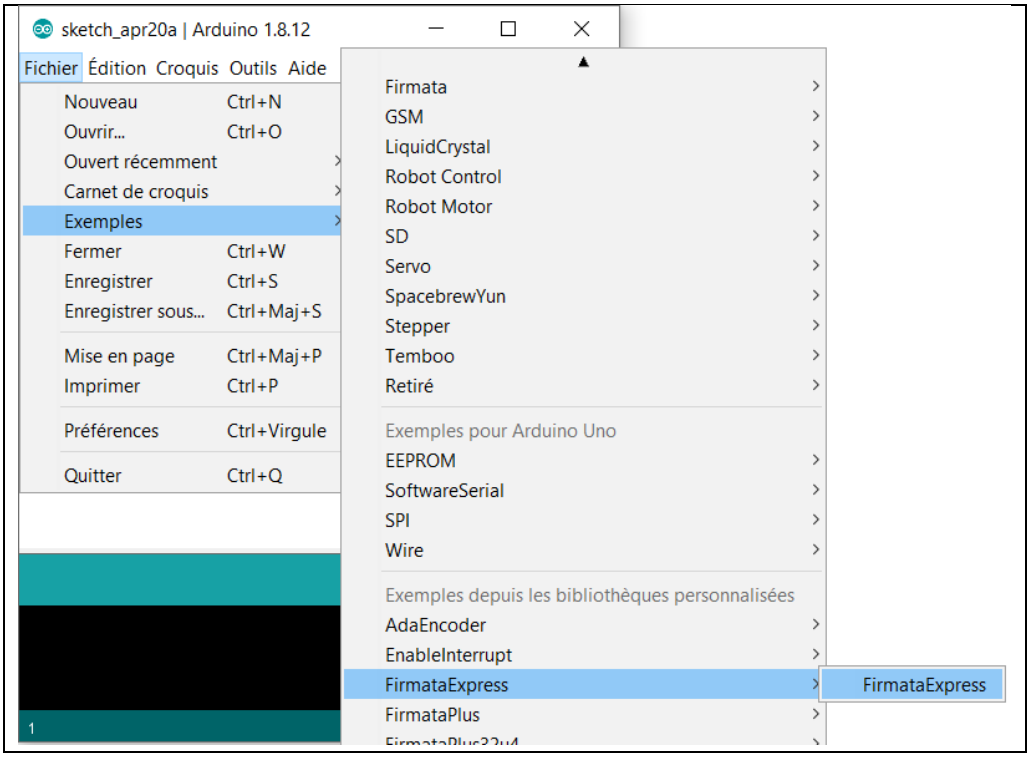
Le chargement du code "**Firmata Express**" dans la mémoire de l'Arduino est maintenant possible :

- Brancher l'Arduino via un port USB,

- Afin de charger la librairie "Firmata express" sur l'ARDUINO, il faut lancer le logiciel "IDE ARDUINO", puis sélectionner :

Fichier > Exemples > FirmataExpress > FirmataExpress,

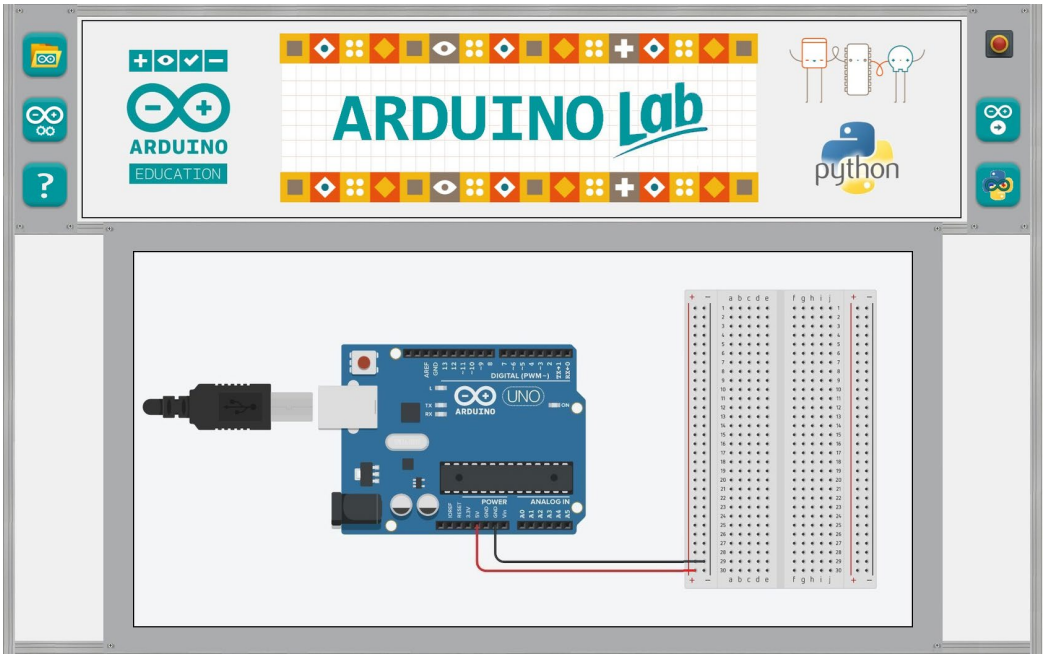
- puis cliquer sur "téléverser".



A partir de là vous n'avez plus besoin du logiciel **Arduino IDE**, vous pourrez contrôler votre Arduino uniquement à partir d'**ARDUINO LAB**.

3. L'interface graphique - Les menus

A partir de la fenêtre d'accueil, on accède aux menus d'**ARDUINO LAB :**



3.1. Menu "Paramètres"

Pour accéder au menu "**Paramètres**", il faut cliquer sur :

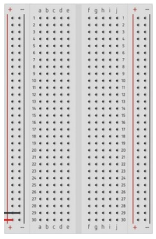
Le Menu "**Paramètres**" d'**ARDUINO LAB** permet d'indiquer:

- le mode de fonctionnement du logiciel, à savoir : "**Contrôle de l'Arduino**" ou "**Simulation**",
- le port "**COM**" sur lequel l'Arduino est connecté (si un seul Arduino est connecté, le port "**COM**" est sélectionné automatiquement)
- le mode de sélection du protocole de communication entre le logiciel et l'Arduino (sélection manuelle ou automatique, chargement de "**Firmata standard**" ou de "**Firmata Express**"),
- l'emplacement du dossier d'installation du logiciel "**Arduino IDE**".

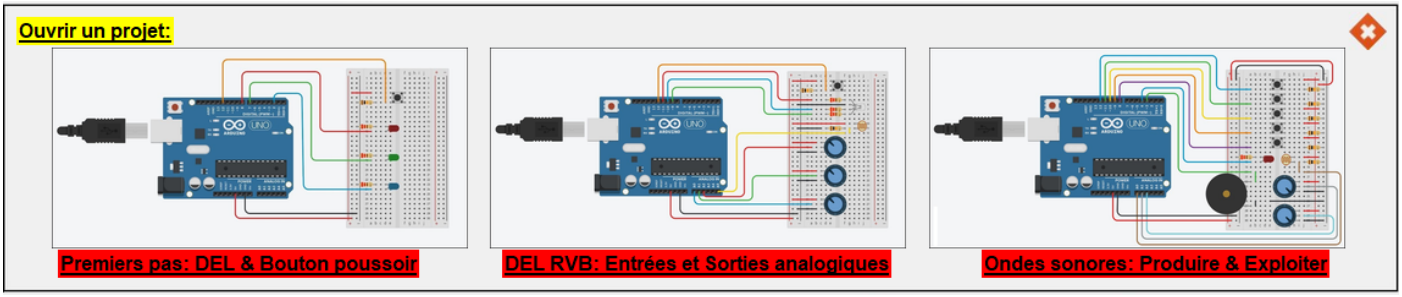
3.2. Menu "Ouvrir un projet"

Pour ouvrir un projet et étudier un circuit, il faut cliquer sur le bouton :

Ou sur la plaque d'essais :

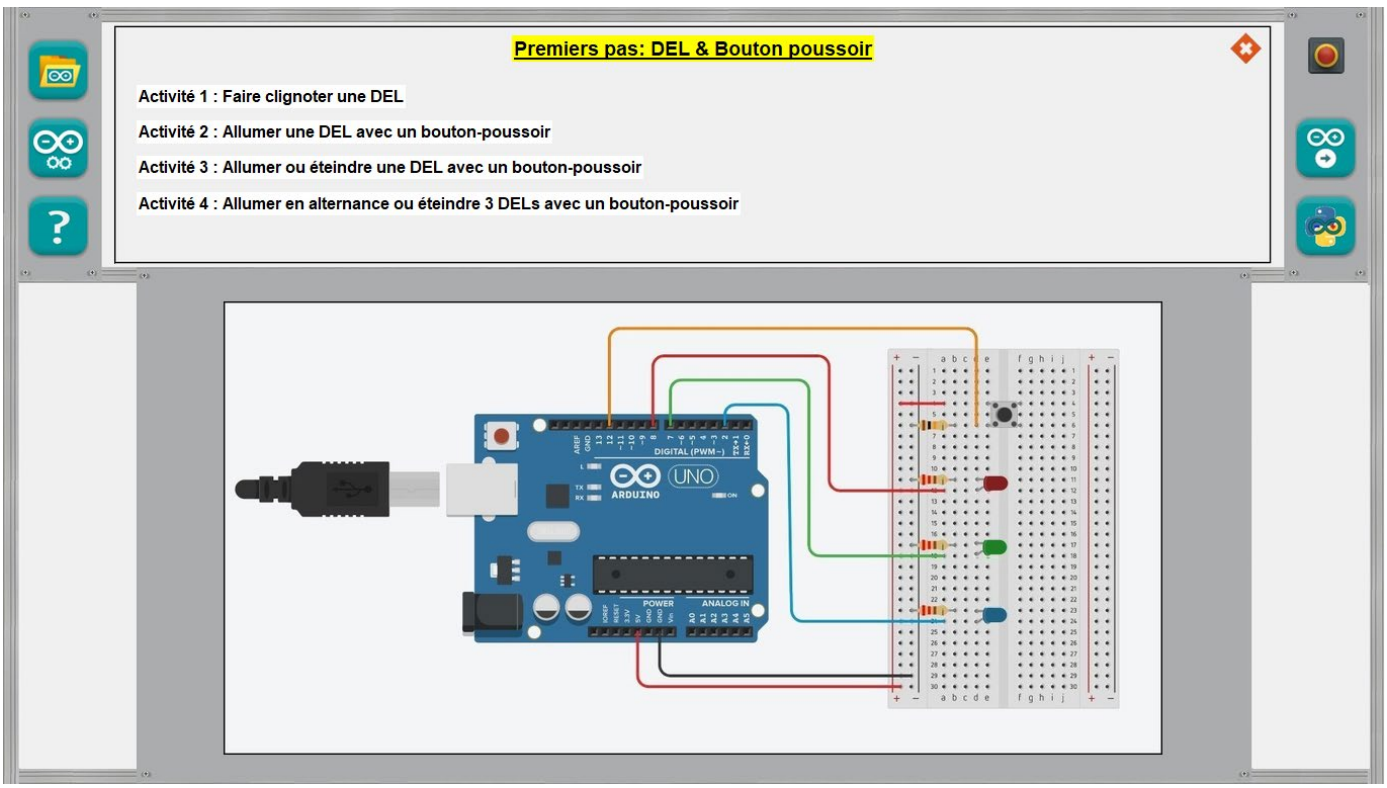


Et choisir le circuit à étudier dans la fenêtre qui s'ouvre, en cliquant sur son image :

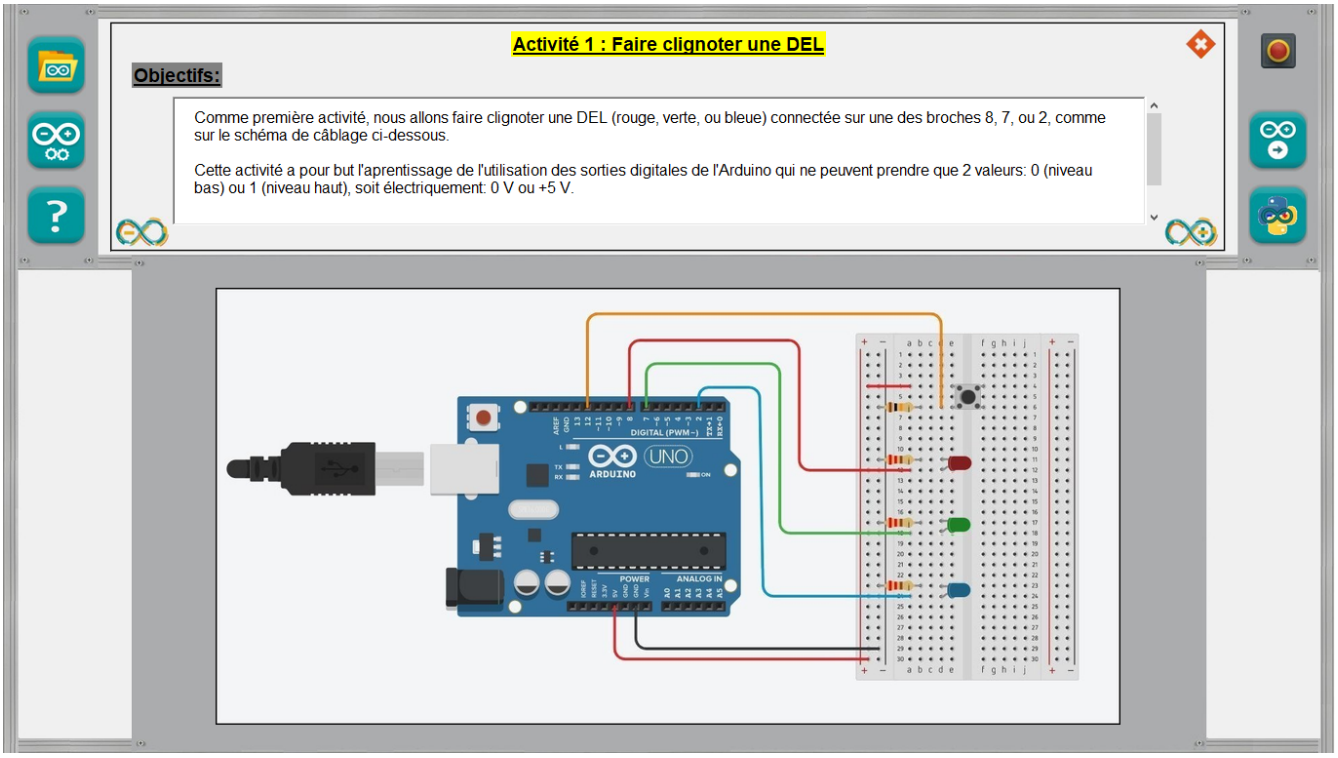


3.3. Menu "Sélectionner une activité"

Après avoir choisi un circuit à étudier, une fenêtre avec le circuit et la liste des activités liées au circuit sélectionné s'affiche :

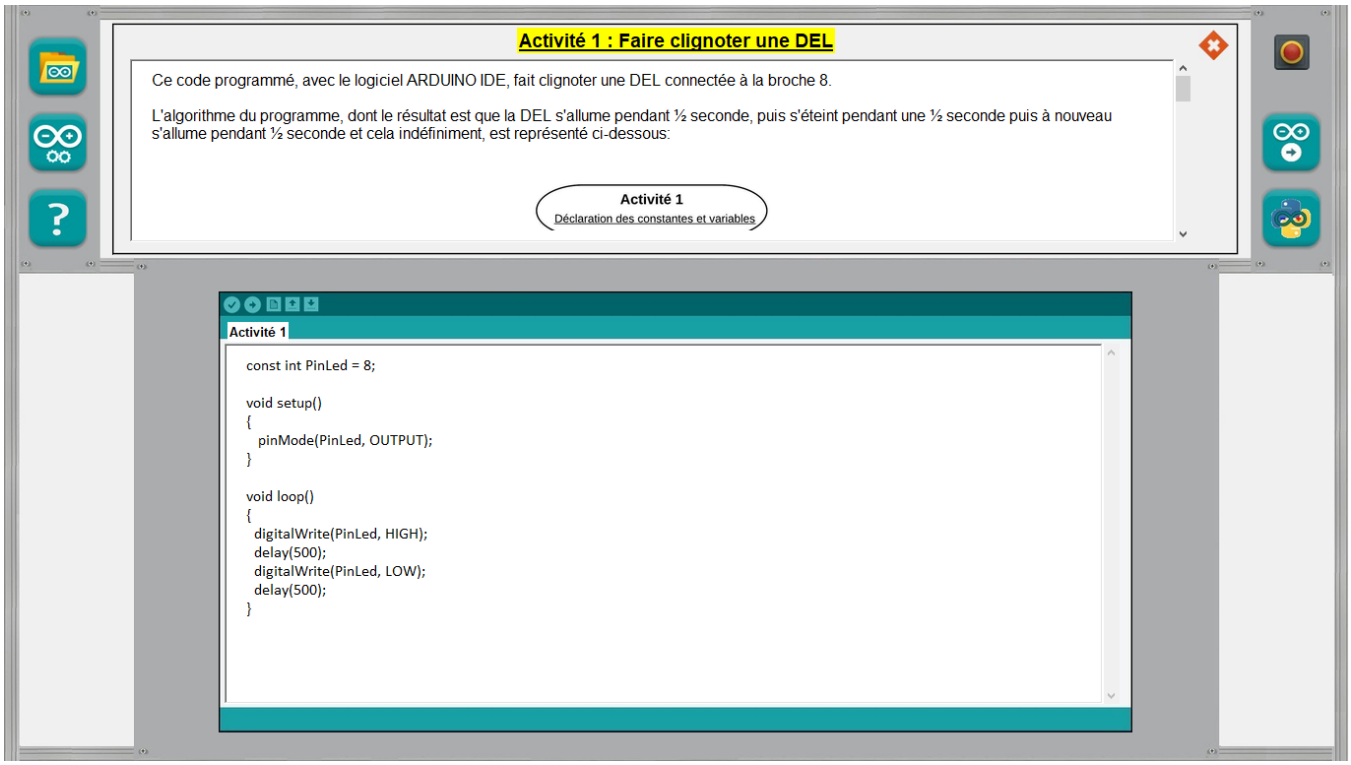


Il suffit de cliquer sur une des activités pour que la fenêtre correspondant à l'activité choisie s'ouvre et affiche sa description :



3.4. Menu "Etude du code en langage ARDUINO IDE"

Le code en langage **ARDUINO IDE** permettant de réaliser l'activité choisie est affiché en cliquant sur le bouton :



L'accès à ce menu de l'interface graphique n'est possible que si une activité est sélectionnée.

Sur la partie supérieure de la fenêtre, une description et l'algorithme du code sont affichés et en dessous, on retrouve le code de l'activité que l'on peut modifier.

Il est possible de sauvegarder le code modifié, d'éditer un nouveau code et d'ouvrir un code préalablement enregistré.

Le code original ou modifié après sauvegarde peut être vérifié et téléversé directement depuis **ARDUINO LAB** dans l'Arduino.

Dans ce cas, l'Arduino est dans un fonctionnement autonome et il n'y a plus d'interaction possible avec **ARDUINO LAB**. Pour réutiliser toutes les fonctions d'**ARDUINO LAB**, il faudra téléverser le protocole de communication adéquate (Firmata standard ou express).

Les fonctionnalités décrites ci-dessus sont accessibles en cliquant sur les boutons :

- Vérifier un code : 
- Téléverser un code : 
- Editer un nouveau code : 
- Ouvrir un code : 
- Sauvegarder un code : 

3.5. Menu "Etude du code en Python"

Le code en Python permettant de réaliser l'activité choisie est affiché en cliquant sur le bouton :

Comme pour le menu précédant, L'accès à ce menu de l'interface graphique n'est possible que si une activité est sélectionnée.

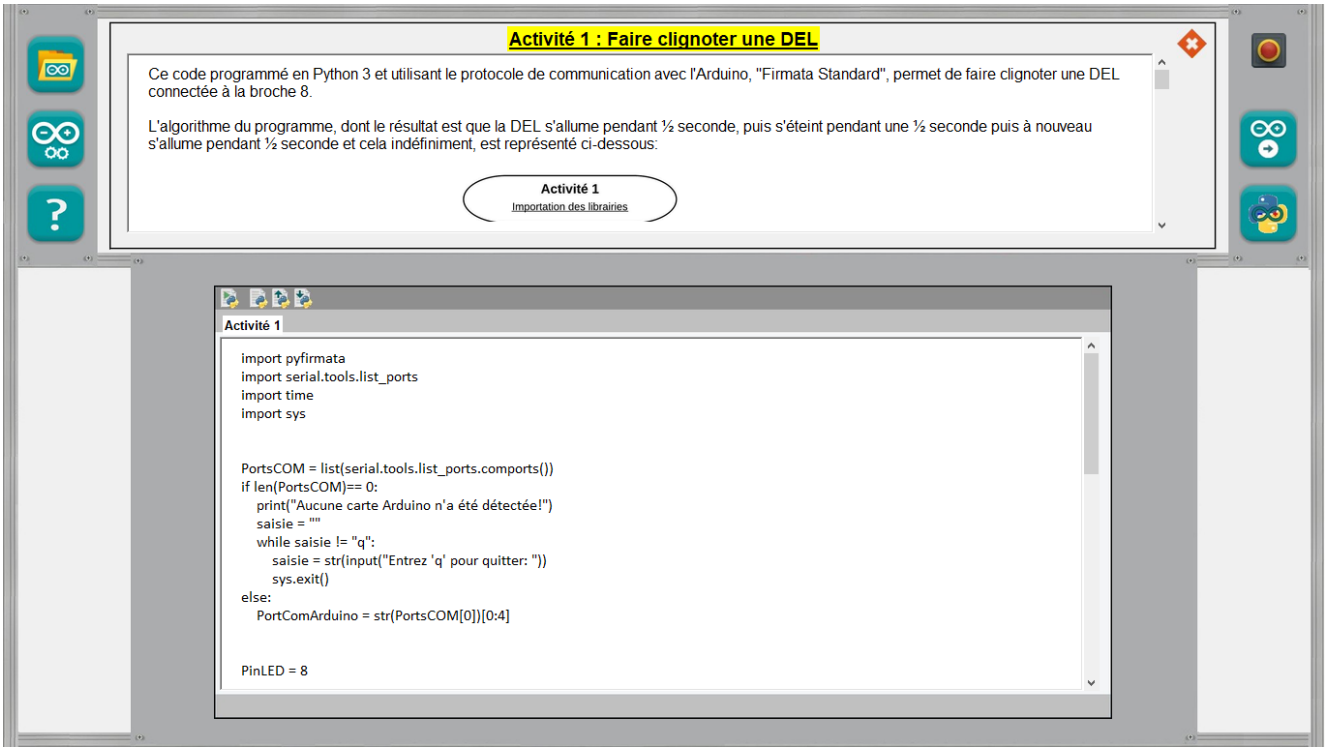
De même, sur la partie supérieure de la fenêtre, une description et l'algorithme du code en Python sont affichés et en dessous, on retrouve le code de l'activité que l'on peut modifier.

Il est également possible de sauvegarder le code modifié, d'éditer un nouveau code et d'ouvrir un code préalablement enregistré.

Le code original ou modifié après sauvegarde est exécutable. Cependant, il faudra s'assurer que le bon protocole de communication entre Python et l'Arduino a bien été téléverser.

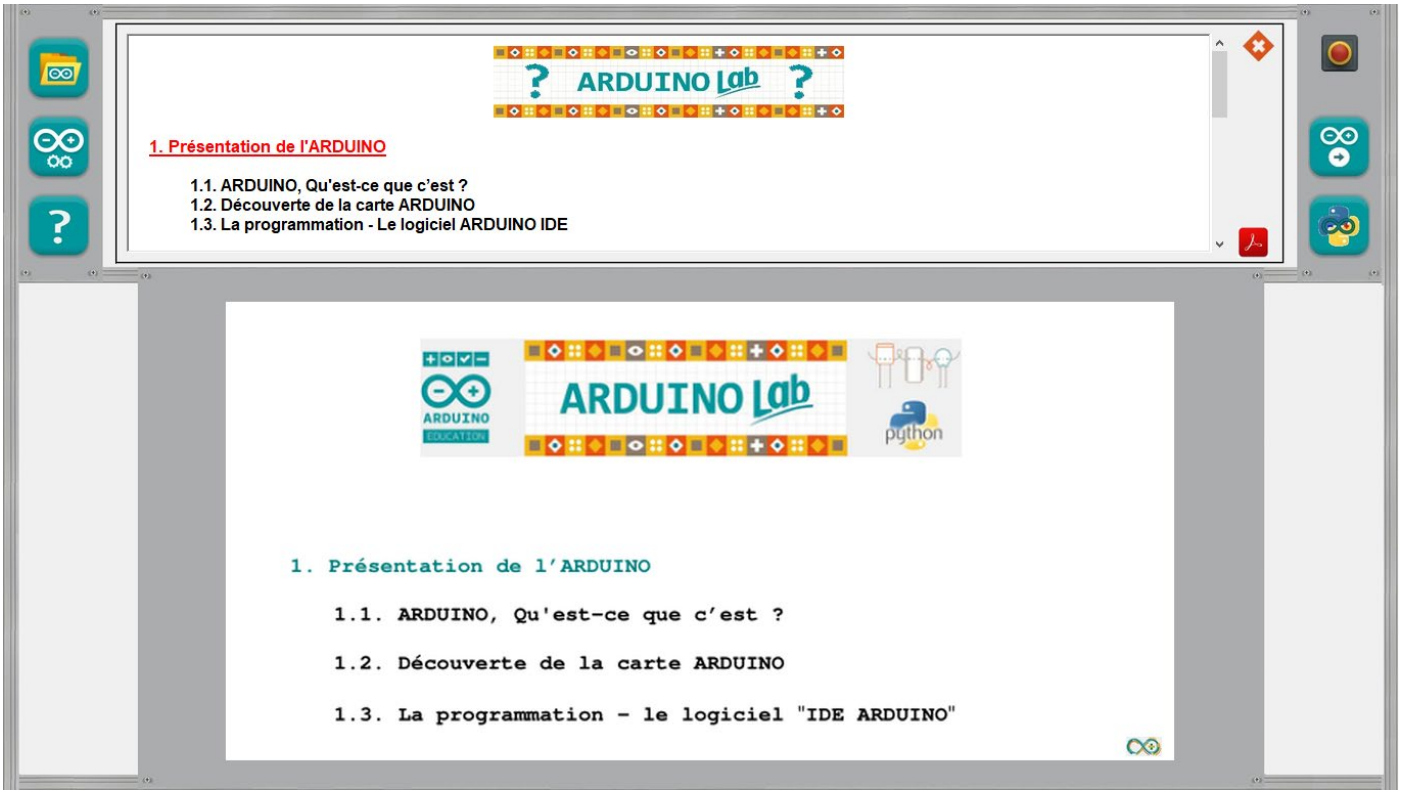
Les fonctionnalités décrites ci-dessus sont accessibles en cliquant sur les boutons :

- Exécuter un code : 
- Editer un nouveau code : 
- Ouvrir un code : 
- Sauvegarder un code : 



3.6. Menu "Aide"

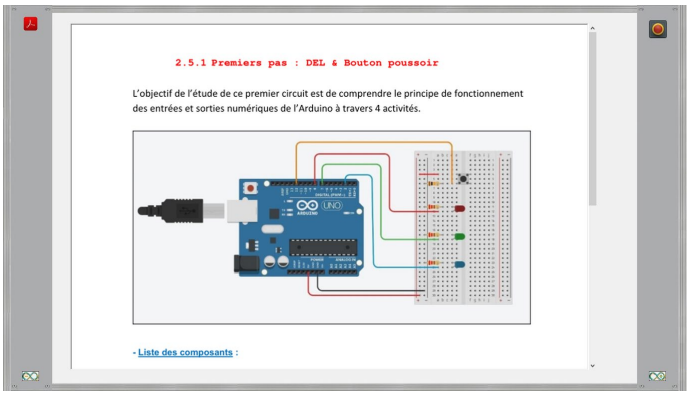
A tout moment, une aide contextuelle est affichée en cliquant sur le bouton :

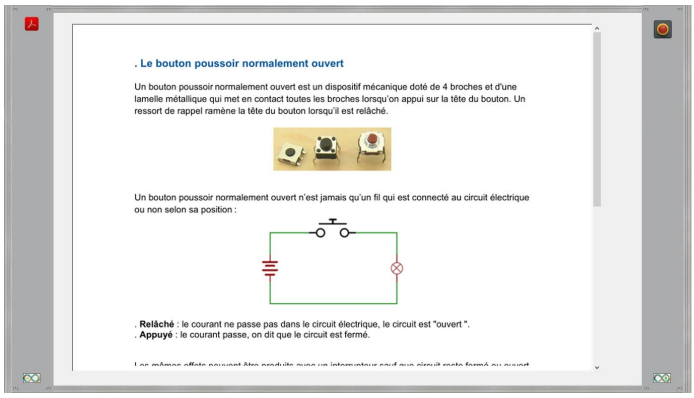


L’accès à l’aide générale d’**ARDUINOLAB** se fait à partir de la fenêtre d’accueil**.**

Quand un circuit ou une activité est sélectionné, l’aide affichée est spécifique à la sélection en cours.

Un clic droit sur certains éléments des circuits ou des fenêtres d’exploitation affiche également une aide contextuelle sur les composants, les données reçues…





<u>Remarques</u> :

. La fenêtre principale et les fenêtres secondaires d’ARDUINO LAB sont fermées en cliquant sur: 

. Quelle que soit la résolution de l’écran, les fenêtres et les objets d’Arduino LAB sont redimensionnés de façon à occuper tout l’espace disponible.

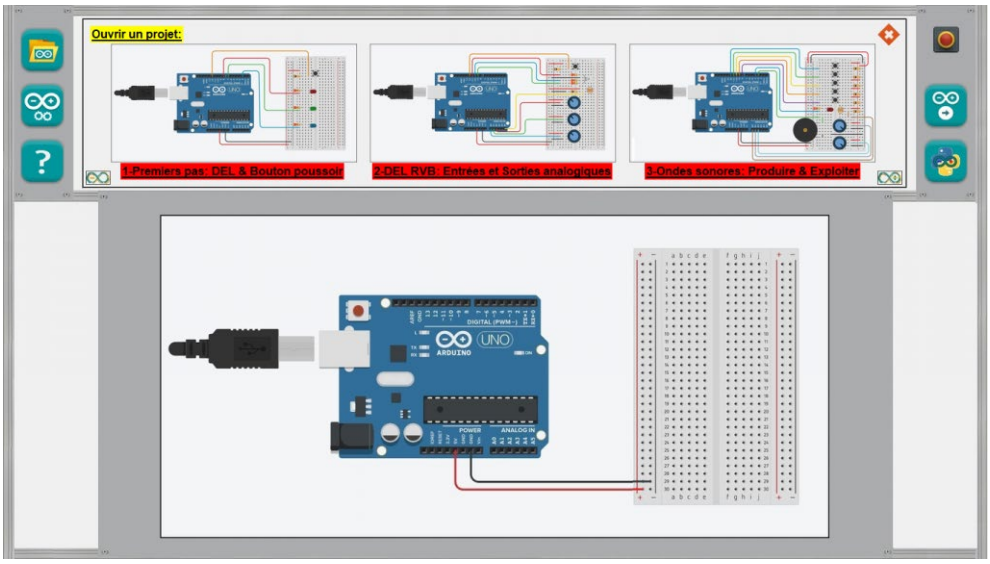
Cependant, l’affichage est optimisé pour les résolutions d’écran suivantes :

. 1366 x 768

. 1680 x 1050

. 1920 x 1080

4. Les projets (étude de circuits) - Les activités



ARDUINO LAB regroupe tous les projets pour **Arduino Uno** programmés en Python qui ont été vus précédemment en ajoutant une interface graphique permettant de contrôler l'Arduino, d'afficher les données reçues et de les exploiter.

Les premiers circuits à étudier et les premières activités associées permettent de se familiariser avec l'Arduino, de comprendre le principe de fonctionnement des entrées et sorties numériques ou analogiques.

Ensuite, c'est l'exploitation des données des principaux capteurs utilisés avec un Arduino qui seront abordés.

Et enfin, on étudiera la gestion des moteurs et servomoteurs.

Pour tous les circuits, quel que soit le mode de fonctionnement d'**ARDUINO LAB** choisi ("**Contrôle de l'Arduino**" ou "**Simulation**"), il faut cliquer sur le connecteur USB pour :

- soit établir la connexion avec l'Arduino et commencer à donner des ordres ou recevoir des données, dans le cas où le mode fonctionnement est le "contrôle de l'Arduino",

- soit lancer la simulation, dans l'autre cas.

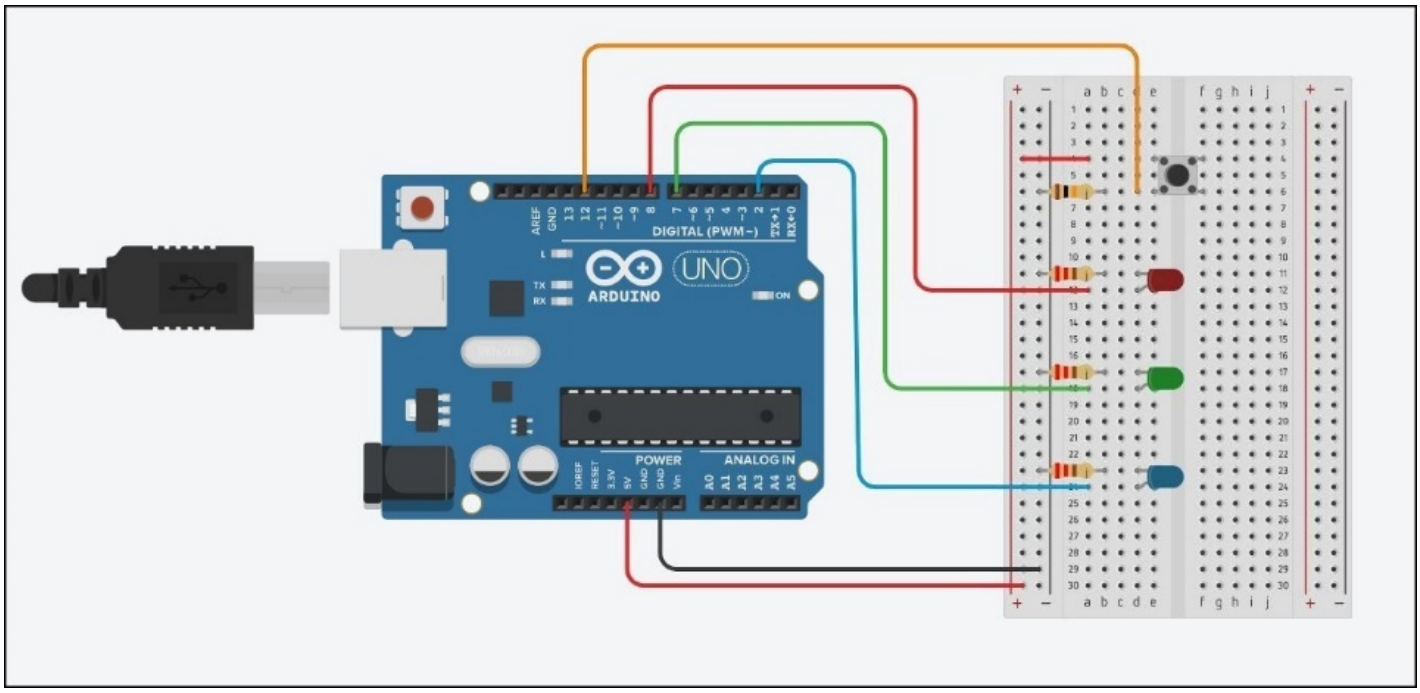
Recommandations :

- Pour le mode de fonctionnement "**Contrôle de l'Arduino**", il faut penser à téléverser le protocole de communication adéquat entre **ARDUINO LAB** et la platine Arduino ou sélectionner le mode sélection automatique du protocole dans le menu "**Paramètres**"

- Il est conseillé de cliquer sur la prise USB avant de quitter l'étude du projet ou de fermer **ARDUINO LAB,** afin de fermer le port "**COM**" sur lequel l'Arduino est connecté.

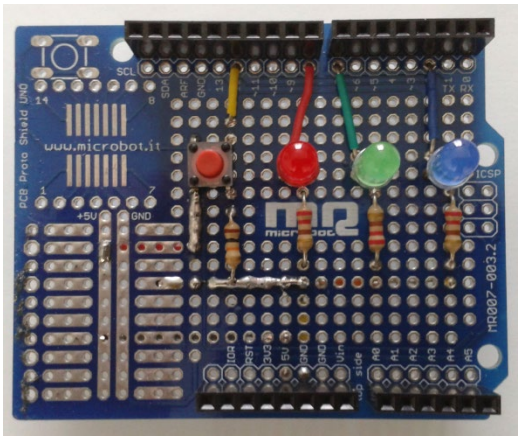
4.1. Premiers pas : DEL & Bouton poussoir

L'objectif de l'étude de ce premier circuit est de comprendre le principe de fonctionnement des entrées et sorties numériques de l'Arduino à travers 4 activités.



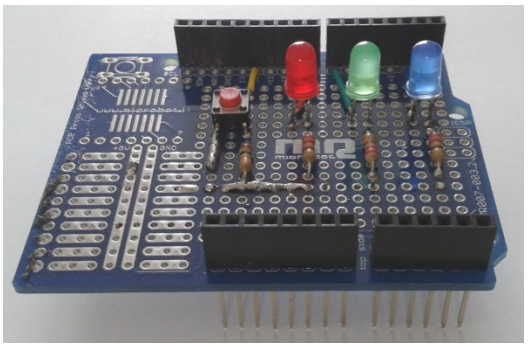
- Liste des composants :

. 1 DEL rouge
. 1 DEL verte
. 1 DEL bleue
. 3 résistances de 220 Ω
. 1 bouton poussoir
. 1 plaque d'essai
. Fils de connexion



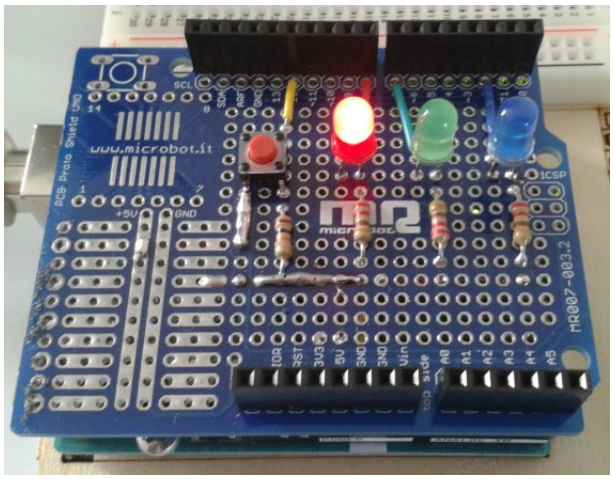
- Protocole de communication (Mode "Contrôle de l'Arduino") :

. Firmata standard



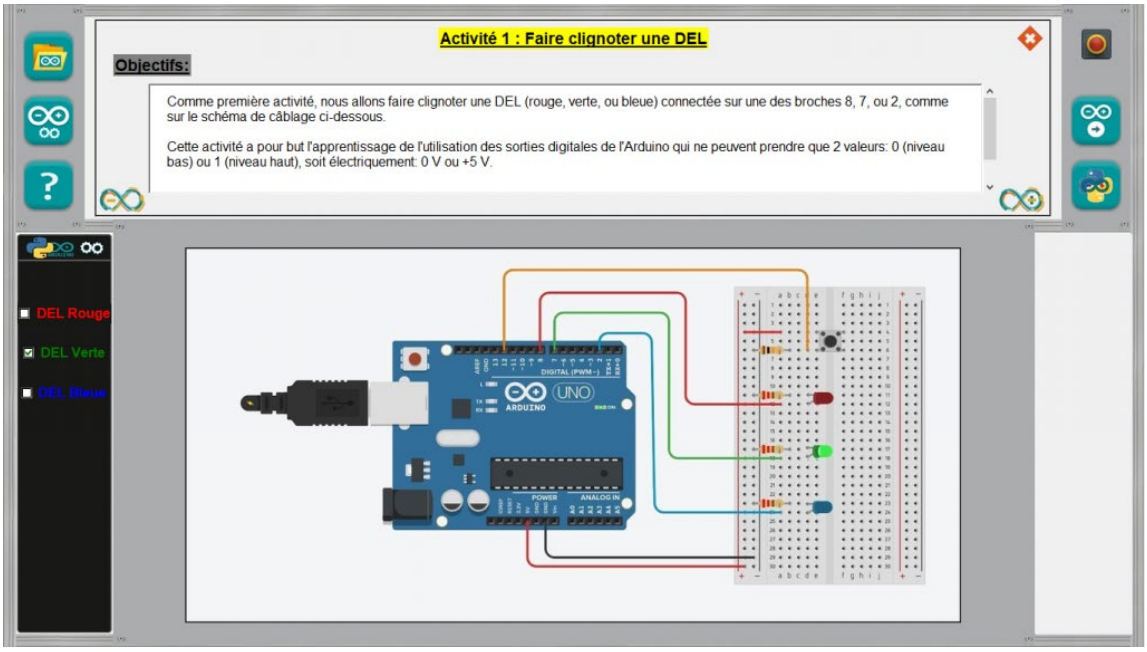
Le circuit sur un "shield" pour Arduino Uno

- Activité 1 : Faire clignoter une DEL



Comme première activité, nous allons faire clignoter une DEL (rouge, verte, ou bleue), préalablement choisie, connectée sur une des broches 8, 7, ou 2, comme sur le schéma de câblage ci-dessus.
Cette activité a pour but l'apprentissage de l'utilisation des sorties digitales de l'Arduino qui ne peuvent prendre que 2 valeurs : **0 (niveau bas)** ou **1 (niveau haut)**, soit électriquement : **0 V ou +5 V**.
Donc, pour allumer la DEL, la broche de l'Arduino sur laquelle celle-ci est connectée, doit être au niveau haut (**+5V**) et pour l'éteindre, elle doit être au niveau bas (**0 V**).

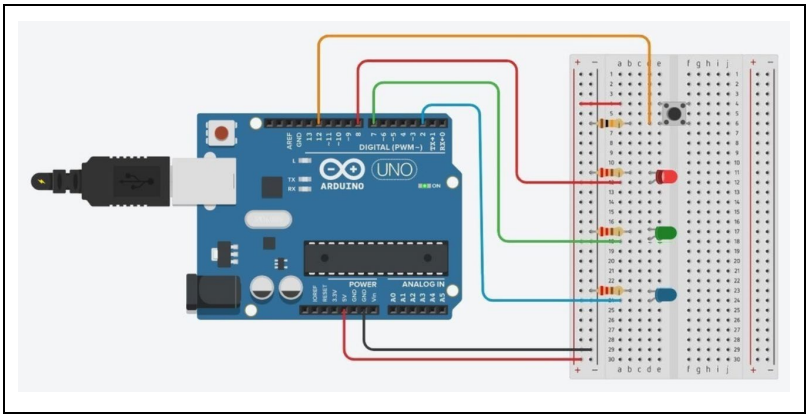
Pour réaliser cette activité, on va demander à l'Arduino d'allumer une des 3 DELs (**donc d'appliquer un niveau haut sur la broche de la DEL**) pendant ½ seconde, puis de l'éteindre (**donc d'appliquer un niveau bas sur la broche de la DEL**) pendant une ½ seconde, puis à nouveau de l'allumer pendant ½ seconde et cela indéfiniment. De cette façon, on verra la DEL choisie clignoter.



ARDUINO LAB – Fenêtre « Projet 1 – Activité 1 »

Après avoir cliqué sur le connecteur USB, le choix de la DEL est fait par l'intermédiaire de ce menu :

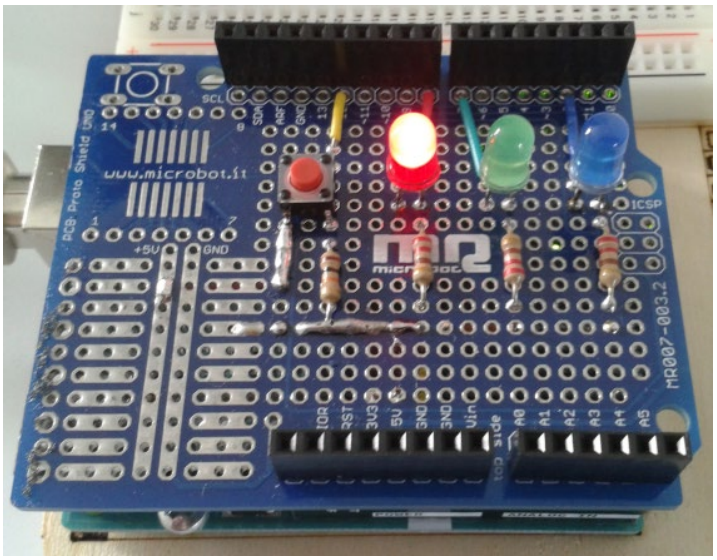
Si le mode de fonctionnement est le "contrôle de l'Arduino", la DEL du circuit réel et la DEL sur l'écran clignotent :



A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

Le code pourra être modifié pour voir l'influence des variables (durée d'allumage, d'extinction, numéro de la broche de la DEL).

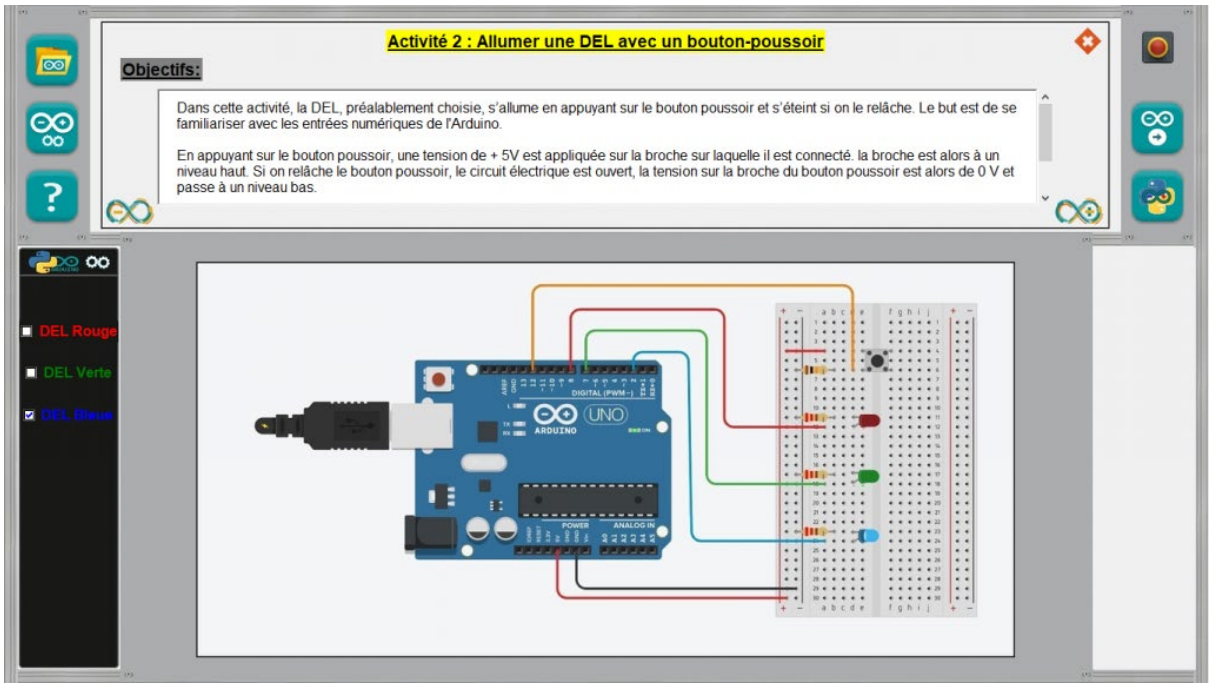
- Activité 2 : Allumer une DEL avec un bouton-poussoir



Dans cette activité, la DEL, préalablement choisie, s'allume en appuyant sur le bouton poussoir et s'éteint si on le relâche. L'objectif est de se familiariser avec les entrées numériques de l'Arduino.

En effet, en appuyant sur le bouton poussoir, une tension de + 5V est appliquée sur la broche sur laquelle il est connecté. La broche est alors à un niveau haut. Si on relâche le bouton poussoir, le circuit électrique est ouvert, la tension sur la broche du bouton poussoir est alors de 0 V et passe à un niveau bas.

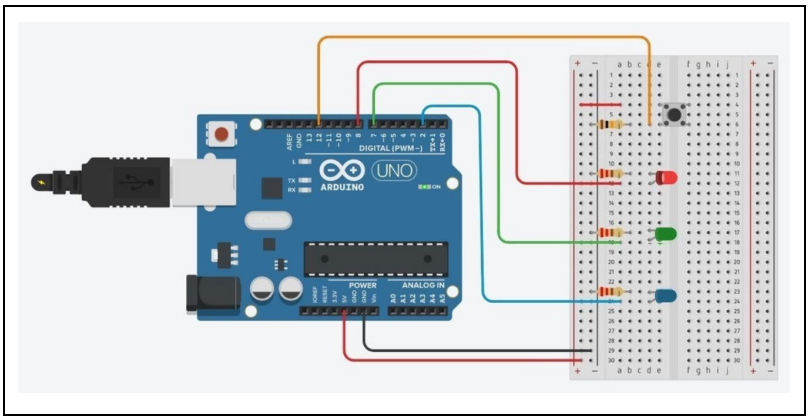
Si on demande à l'Arduino d'interroger l'état logique (**niveau haut ou bas**) de la broche du bouton poussoir qui a été déclaré comme une entrée numérique, on peut savoir si celui-ci est appuyé ou pas et donc lui donner l'ordre d'allumer ou d'éteindre la DEL.



ARDUINO LAB – Fenêtre « Projet 1 – Activité 2 »

Après avoir cliqué sur le connecteur USB, le choix de la DEL est fait par l'intermédiaire de ce menu :

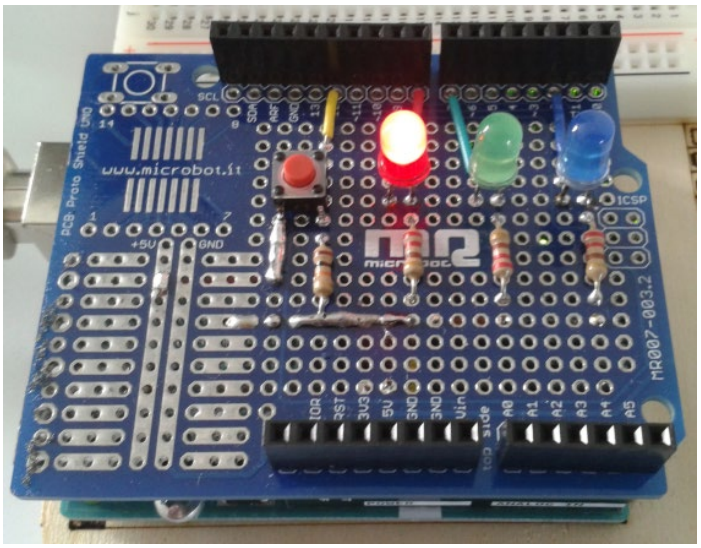
Si le mode de fonctionnement est le "contrôle de l'Arduino", la DEL du circuit réel et la DEL sur l'écran s'allument en appuyant sur le bouton poussoir du circuit réel ou sur celui du circuit affiché sur l'écran :



A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

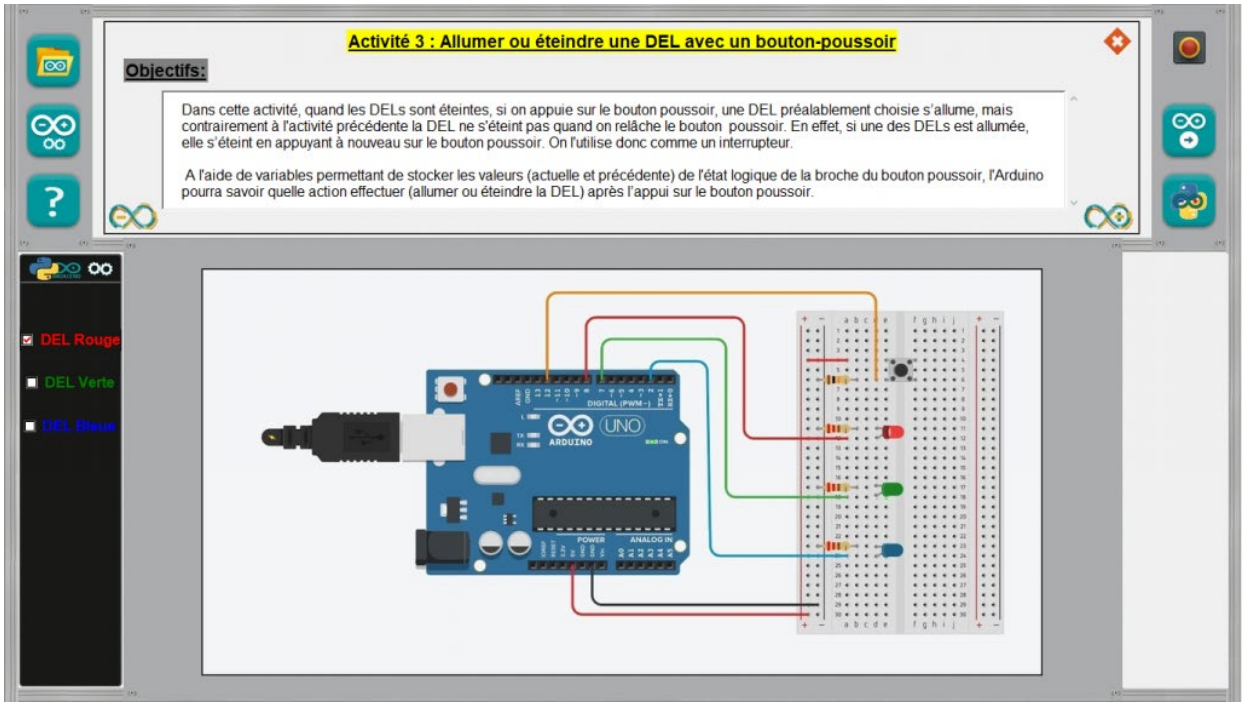
Le code pourra être modifié pour voir l'influence des variables (numéro de la broche de la DEL).

- Activité 3 : Allumer ou éteindre une DEL avec un bouton-poussoir



Dans cette activité, quand les DELs sont éteintes, si on appuie sur le bouton poussoir, une DEL préalablement choisie s'allume, mais contrairement à l'activité précédente la DEL ne s'éteint pas quand on relâche le bouton poussoir. En effet, si une des DELs est allumée, elle s'éteint en appuyant à nouveau sur le bouton poussoir. Cette fois, le principe de fonctionnement du bouton poussoir est comme celui d'un interrupteur.

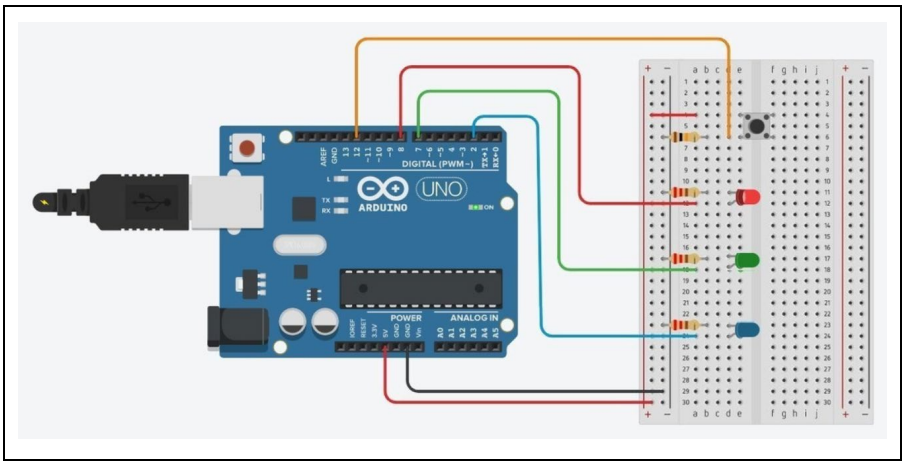
Pour réaliser cette activité, on va demander à l'Arduino d'interroger l'état logique (**niveau haut ou bas**) de la broche du bouton poussoir qui a été déclaré comme une entrée numérique. A l'aide de variables permettant de stocker les valeurs (actuelle et précédente) de cet état, l'Arduino pourra savoir quelle action effectuer (allumer ou éteindre la DEL) après l'appui sur le bouton poussoir.



Après avoir cliqué sur le connecteur USB, le choix de la DEL est fait par l'intermédiaire de ce menu :

Une modification dans le choix de la DEL quand une DEL est déjà allumée, entraine une réinitialisation du programme. Il faut appuyer de nouveau sur le bouton poussoir pour allumer la DEL choisie.

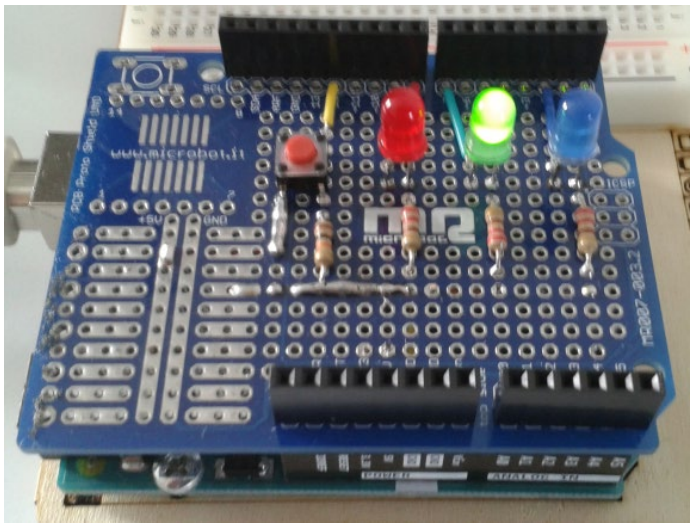
Si le mode de fonctionnement est le "contrôle de l'Arduino", la DEL du circuit réel et la DEL sur l'écran s'allument ou s'éteignent en appuyant sur le bouton poussoir du circuit réel ou sur celui du circuit affiché sur l'écran :



A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

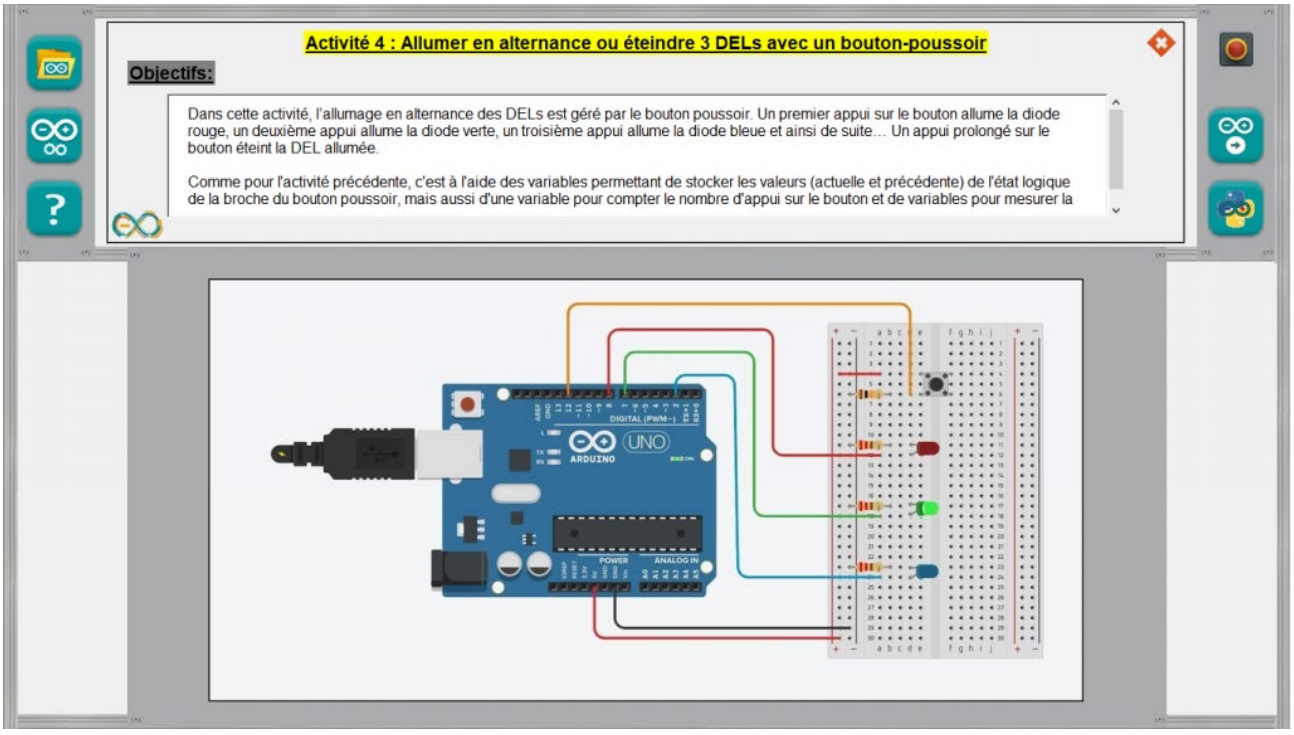
Le code pourra être modifié pour voir l'influence des variables (numéro de la broche de la DEL).

- Activité 4 : Allumer en alternance ou éteindre 3 DELs avec un bouton-poussoir



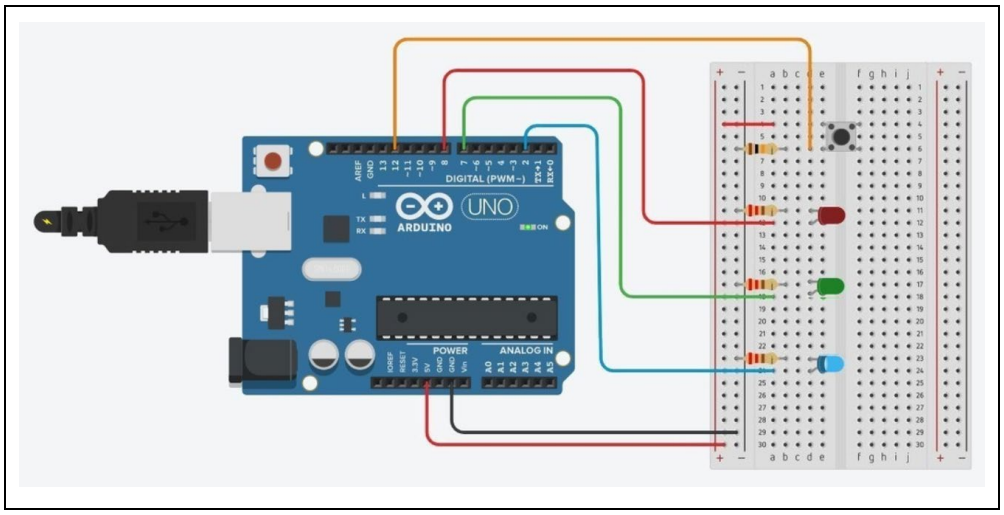
Dans cette activité, l'allumage en alternance des DELs est géré par le bouton poussoir. Un premier appui sur le bouton allume la diode rouge, un deuxième appui allume la diode verte, un troisième appui allume la diode bleue et ainsi de suite…
Un appui prolongé sur le bouton éteint la DEL allumée.

Comme pour l'activité précédente, c'est à l'aide des variables permettant de stocker les valeurs (actuelle et précédente) de l'état logique de la broche du bouton poussoir, mais aussi d'une variable pour compter le nombre d'appui sur le bouton et de variables pour mesurer la durée d'appui, que l'Arduino pourra allumer ou éteindre les DELs.



ARDUINO LAB – Fenêtre « Projet 1 – Activité 4 »

Si le mode de fonctionnement est le "contrôle de l’Arduino", les DELs du circuit réel et les DELs sur l’écran s’allument ou s’éteignent en appuyant sur le bouton poussoir du circuit réel ou sur celui du circuit affiché sur l’écran :

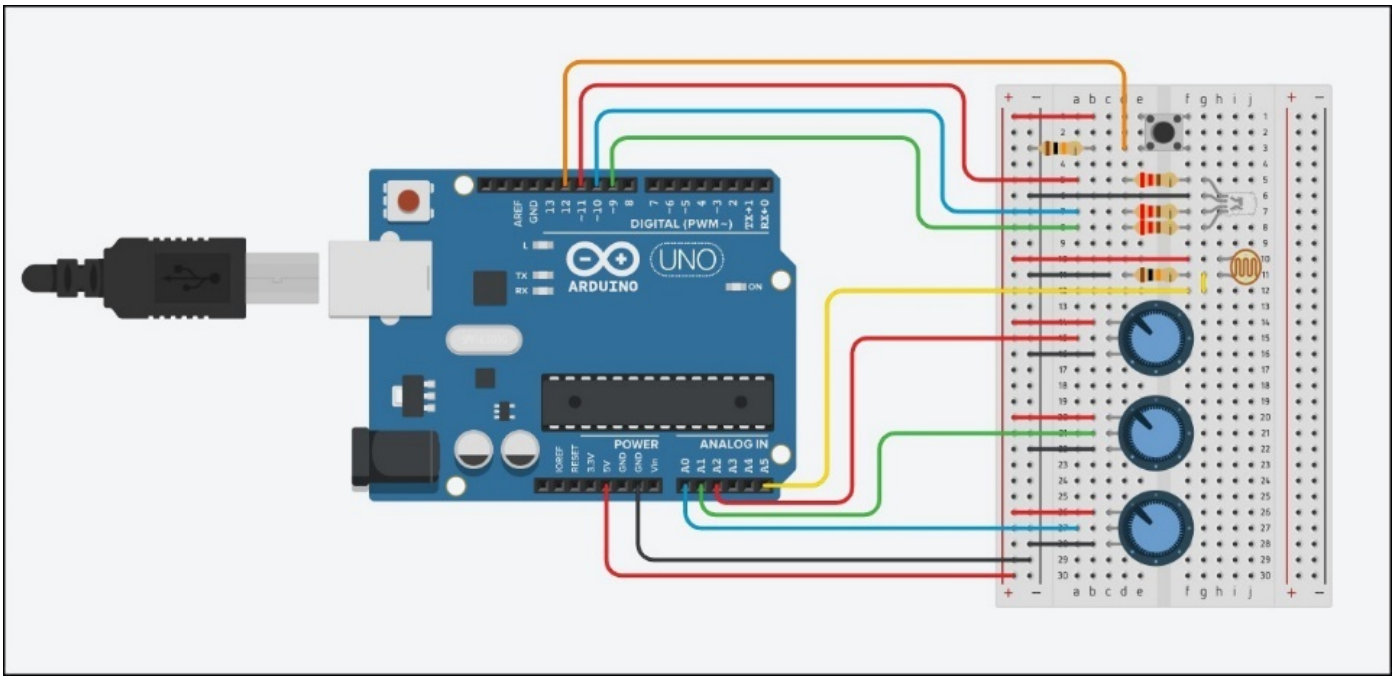


A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

Le code pourra être modifié pour voir l’influence des variables (durée d’appui pour éteindre les DELS).

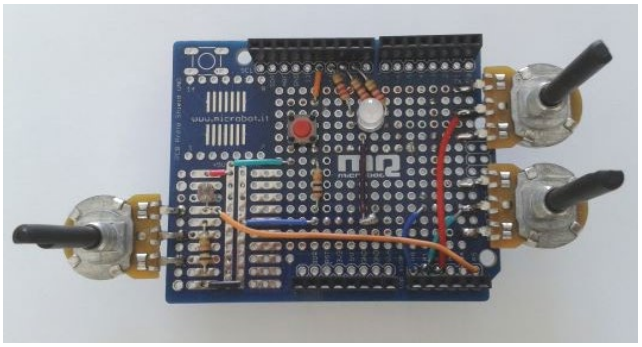
4.2 DEL RVB : Entrées et Sorties analogiques

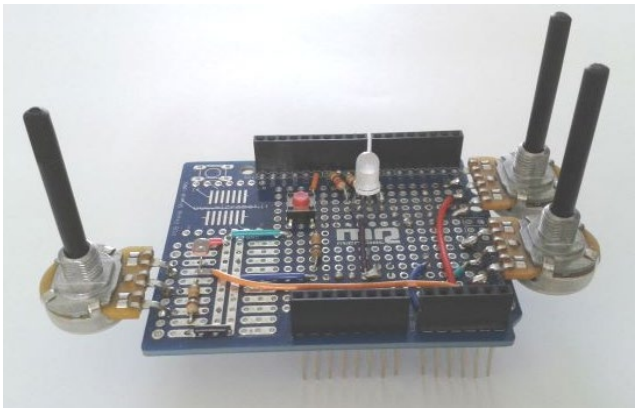
Après avoir vu le principe de fonctionnement des entrées et des sorties numériques de l'Arduino, avec ce nouveau circuit, nous allons nous intéresser aux entrées et sorties analogiques de la platine.



- Liste des composants :

. 1 DEL RVB
. 3 résistances de 220 Ω
. 2 résistances de 10 kΩ
. 3 potentiomètres de 10 kΩ
. 1 photorésistance
. 1 bouton poussoir
. 1 plaque d'essai
. Fils de connexion

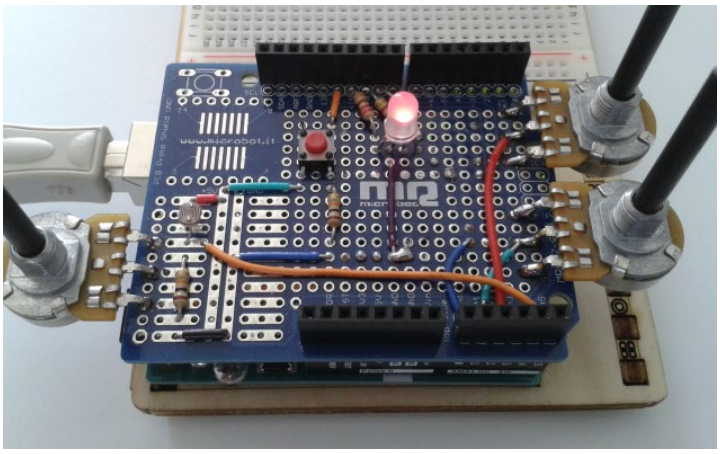




- Protocole de communication (Mode "Contrôle de l'Arduino") :

. Firmata standard

- Activité 1 : Contrôler la luminosité d'une DEL avec un bouton-poussoir



Contrairement aux sorties numériques qui ne peuvent avoir que deux valeurs 0 ou 1 (0 ou 5V), une sortie analogique (ou plutôt PWM) permet d'obtenir une tension entre 0 et 5 V. les broches 3, 5, 6, 9, 10 et 11 peuvent être configurés en sortie analogique.

C'est pour cela que les anodes de la DEL RVB (à cathode commune) de notre circuit sont connectées sur les broches :

- 9 pour la DEL rouge
- 11 pour la DEL verte
- 10 pour la DEL Bleue

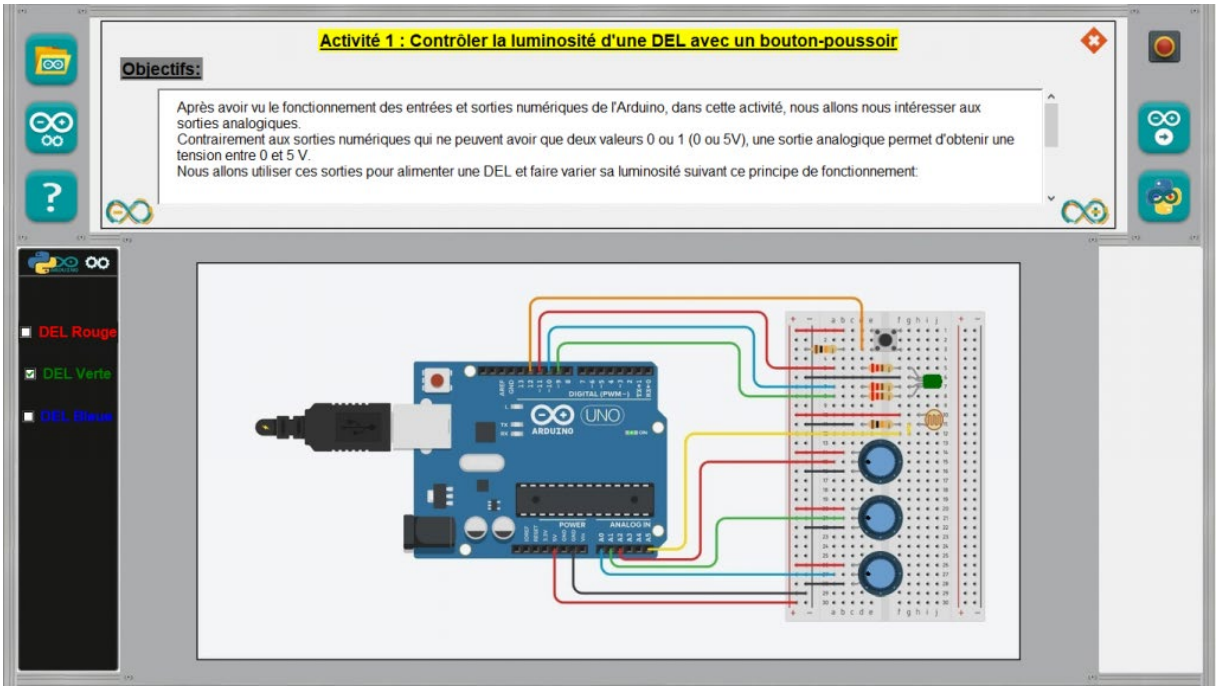
Nous allons utiliser ces sorties pour alimenter une DEL (DEL rouge, verte ou bleue de la DEL RVB) et faire varier sa luminosité suivant ce principe de fonctionnement :

- la DEL étant éteinte, si on appuie sur le bouton poussoir, la diode s'allume.

- On règle la luminosité de la DEL en maintenant le bouton poussoir appuyé, du moins au plus lumineux (broche de la DEL à +0V) jusqu'à un maximum (broche de la DEL à +5V).

- Quand le maximum de la luminosité est atteint et que le bouton poussoir est maintenu appuyé, la luminosité revient au minimum (broche de la DEL à 0V). "

- La DEL étant allumée, si on appuie sur le bouton poussoir, elle s'éteint.

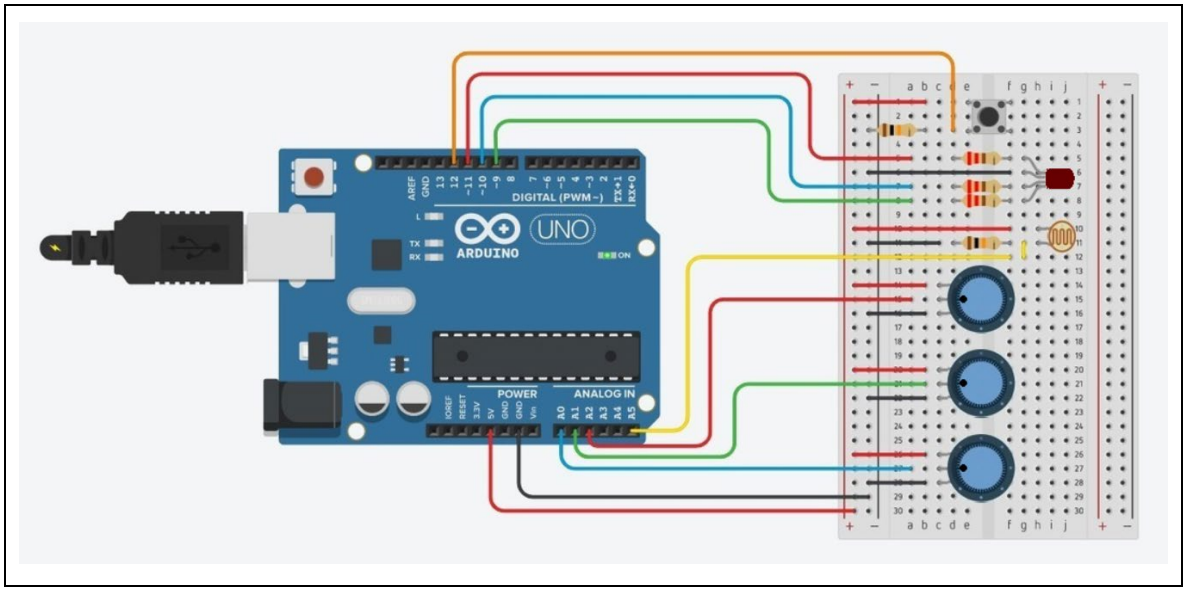


ARDUINO LAB – Fenêtre « Projet 2 – Activité 1 »

Après avoir cliqué sur le connecteur USB, le choix de la DEL est fait par l'intermédiaire du menu ci-contre :

Une modification dans le choix de la DEL quand une DEL est déjà allumée, entraine une réinitialisation du programme. Il faut appuyer de nouveau sur le bouton poussoir pour allumer la DEL choisie.

Si le mode de fonctionnement est le "contrôle de l'Arduino", la DEL du circuit réel et la DEL sur l'écran s'allument ou s'éteignent en appuyant sur le bouton poussoir du circuit réel ou sur celui du circuit affiché sur l'écran.

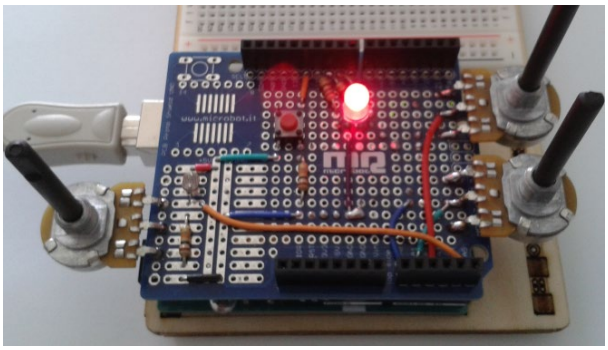


ARDUINO LAB – « La luminosité de la DEL est réglée avec le bouton poussoir »

A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

Le code pourra être modifié pour voir l'influence des variables (choix de la DEL).

- Activité 2 : Clignotement d'une DEL à une allure fixée par une entrée analogique

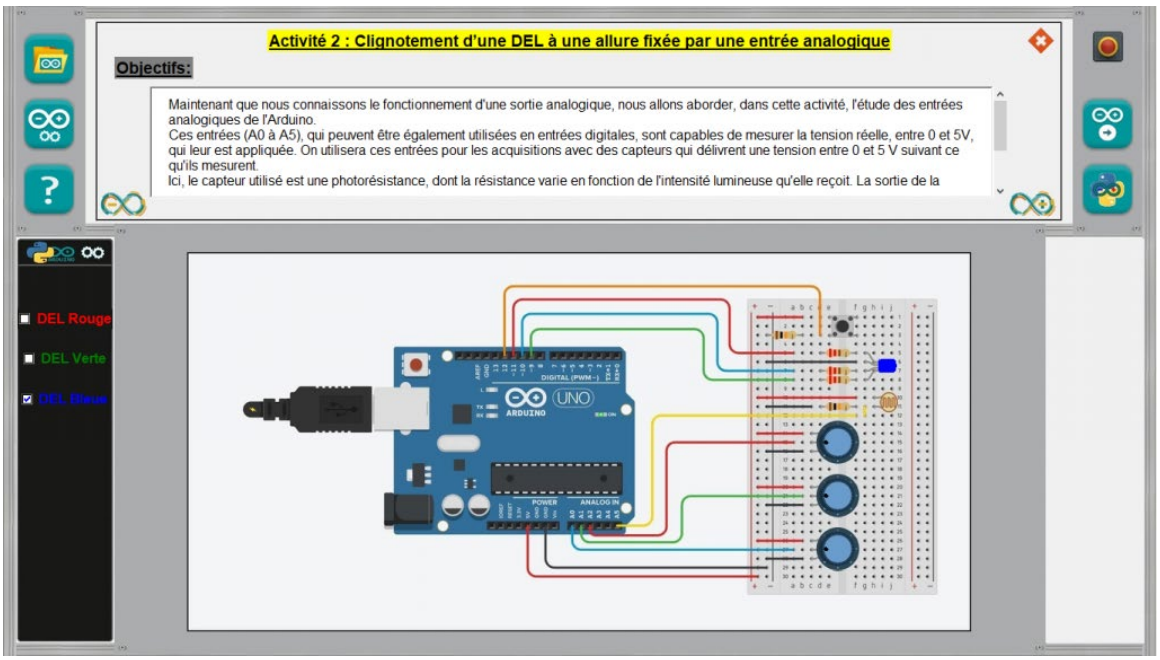


Maintenant que nous connaissons le fonctionnement d'une sortie analogique, nous allons aborder, dans cette activité, l'étude des entrées analogiques de l'Arduino.

Ces entrées (A0 à A5), qui peuvent être également utilisées en entrées digitales, sont capables de mesurer la tension réelle, entre 0 et 5V, qui leur est appliquée. On utilisera ces entrées pour les acquisitions avec des capteurs qui délivrent une tension entre 0 et 5 V suivant ce qu'ils mesurent.

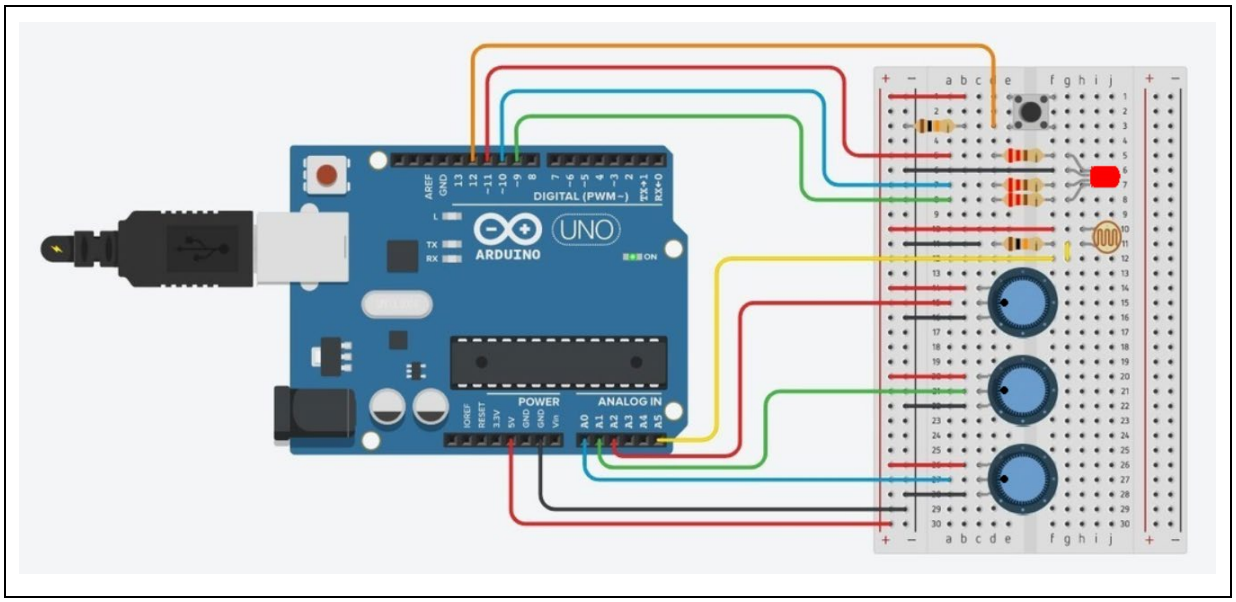
Ici, le capteur utilisé est une photorésistance, dont la résistance varie en fonction de l'intensité lumineuse qu'elle reçoit. La sortie de la photorésistance est branchée sur une des entrées analogiques de la carte Arduino (entrée A5).

L'objectif est de faire clignoter une DEL à une fréquence dépendant de la lumière ambiante. Pour cela, on va faire varier le délai entre 2 allumages de la DEL en fonction de la tension de l'entrée A5 et donc de l'intensité lumineuse reçue par la photorésistance.
Quand l'intensité lumineuse reçue par la photorésistance diminue, la fréquence de clignotement augmente.



Après avoir cliqué sur le connecteur USB, le choix de la DEL est fait par l'intermédiaire de ce menu :

Si le mode de fonctionnement est le "contrôle de l'Arduino", la DEL du circuit réel et la DEL sur l'écran clignotent à une fréquence dépendant de l'intensité lumineuse de la photorésistance réelle ou de celle du circuit sur l'écran (le changement d'intensité lumineuse est alors simulé en passant la souris sur la photorésistance).

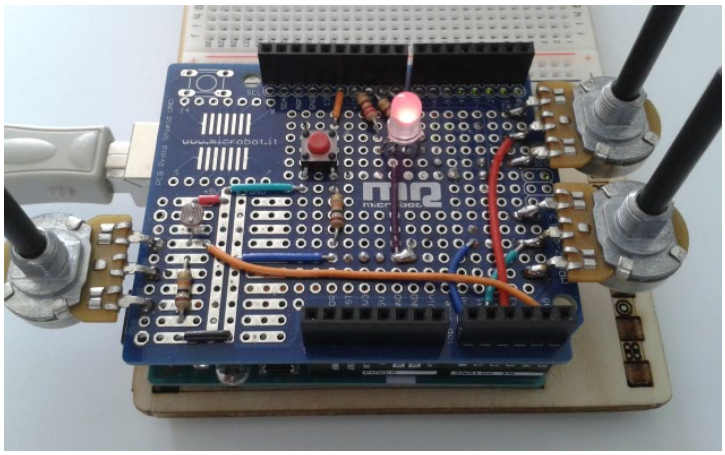


ARDUINO LAB – « La DEL clignote à une fréquence dépendant de la lumière ambiante »

A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

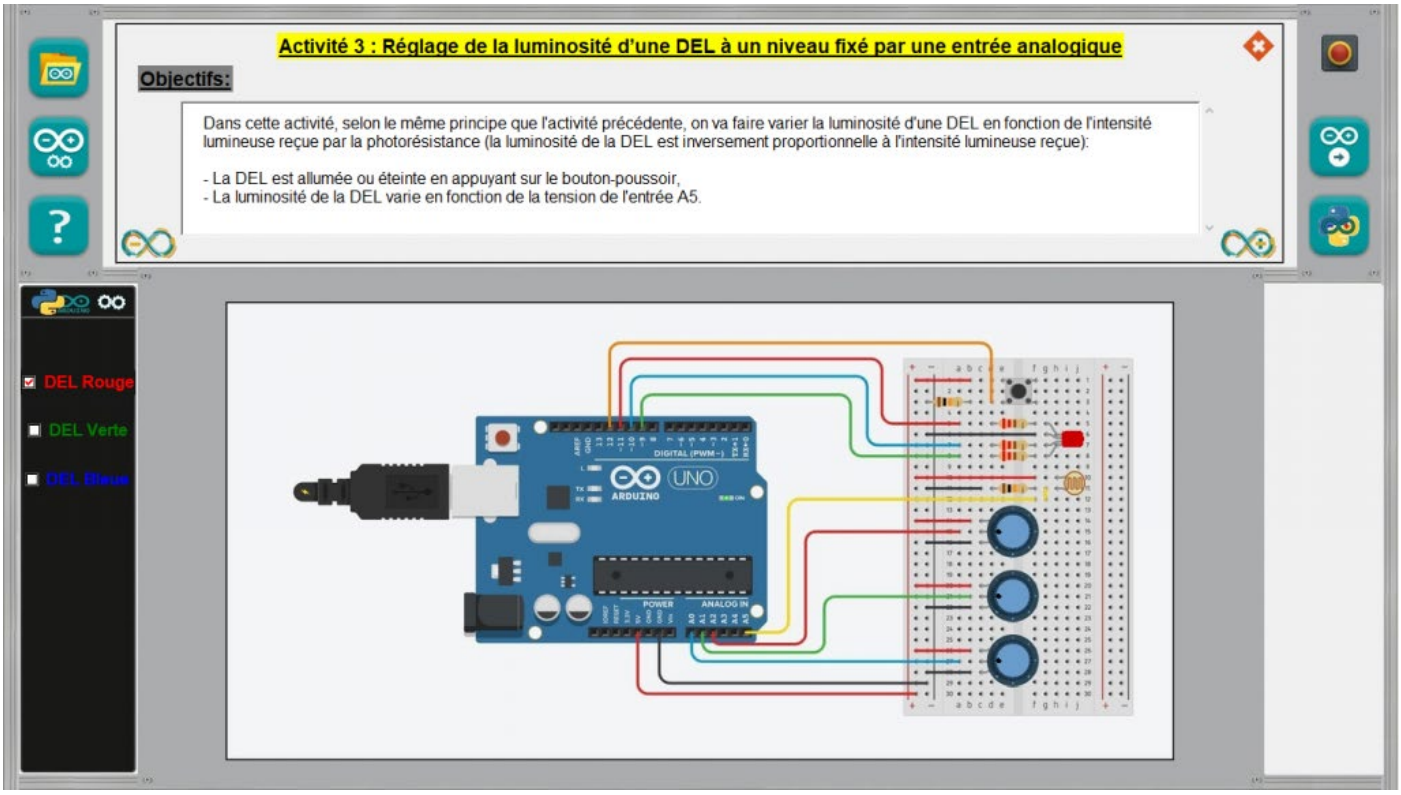
Le code pourra être modifié pour voir l'influence des variables (choix de la DEL).

- Activité 3 : Réglage de la luminosité d'une DEL à un niveau fixé par une entrée analogique



Dans cette activité, selon le même principe que l'activité précédente, on va faire varier la luminosité d'une DEL en fonction de l'intensité lumineuse reçue par la photorésistance (la luminosité de la DEL est inversement proportionnelle à l'intensité lumineuse reçue) :

- La DEL est allumée ou éteinte en appuyant sur le bouton-poussoir,
- La luminosité de la DEL varie en fonction de la tension de l'entrée A5.

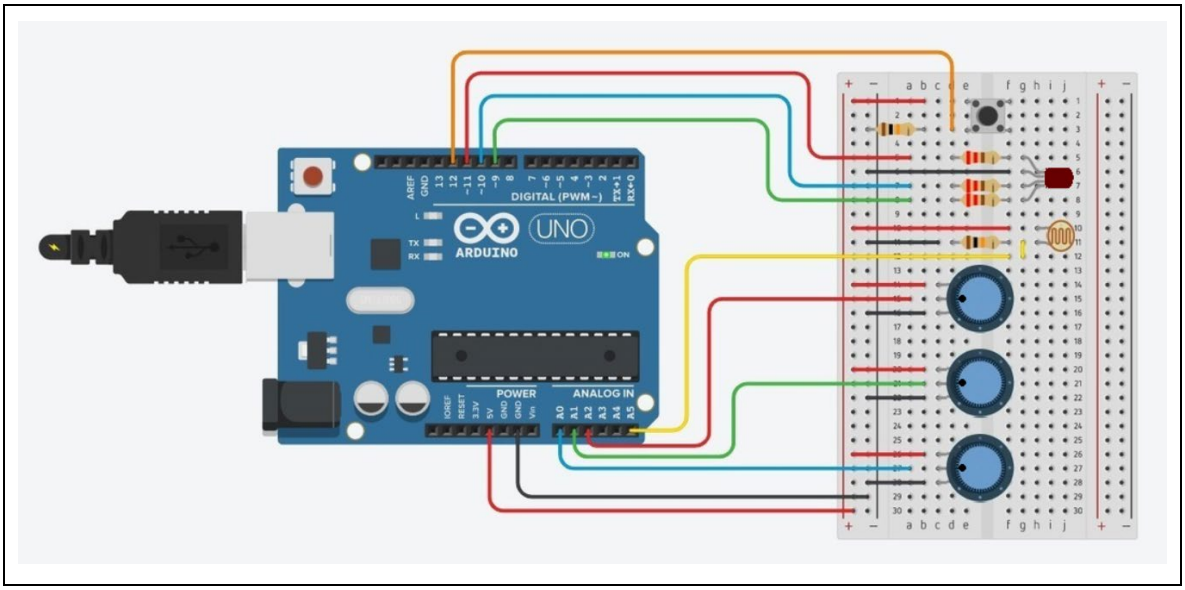


ARDUINO LAB – Fenêtre « Projet 2 – Activité 3 »

Après avoir cliqué sur le connecteur USB, le choix de la DEL est fait par l'intermédiaire de ce menu :

Une modification dans le choix de la DEL quand une DEL est déjà allumée, entraine une réinitialisation du programme. Il faut appuyer de nouveau sur le bouton poussoir pour allumer la DEL choisie.

Si le mode de fonctionnement est le "contrôle de l'Arduino", la DEL du circuit réel et la DEL sur l'écran s'allument ou s'éteignent en appuyant sur le bouton poussoir du circuit réel ou sur celui du circuit affiché sur l'écran. La luminosité de la DEL varie en fonction de la valeur de la photorésistance réelle ou de celle du circuit sur l'écran (le changement d'intensité lumineuse est alors simulé en passant la souris sur la photorésistance).



ARDUINO LAB – « La luminosité de la DEL varie en fonction de la tension de l'entrée A5 »

A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

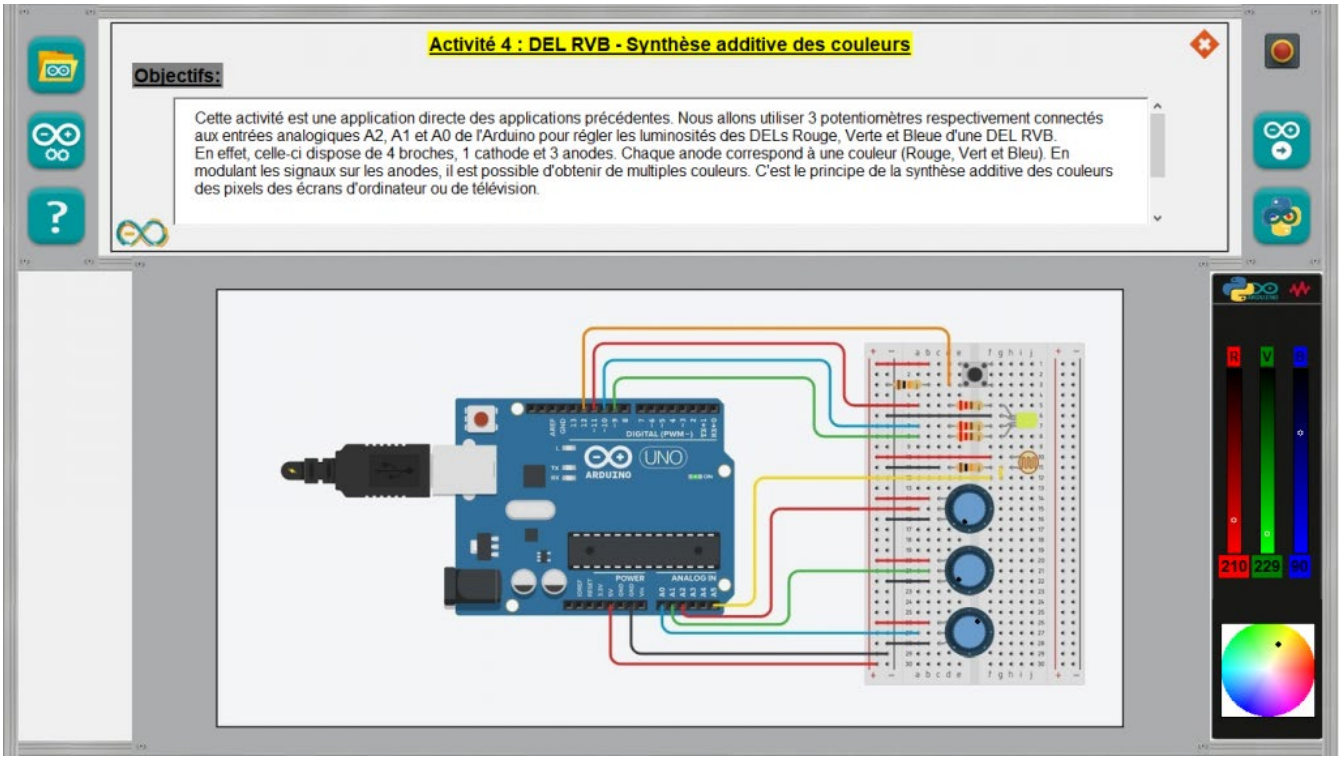
Le code pourra être modifié pour voir l'influence des variables (choix de la DEL).

- Activité 4 : DEL RVB - Synthèse additive des couleurs

Cette activité est une application directe des applications précédentes. Nous allons utiliser 3 potentiomètres respectivement connectés aux entrées analogiques A2, A1 et A0 de l'Arduino pour régler les luminosités des DELs Rouge, Verte et Bleue d'une DEL RVB à cathode commune.

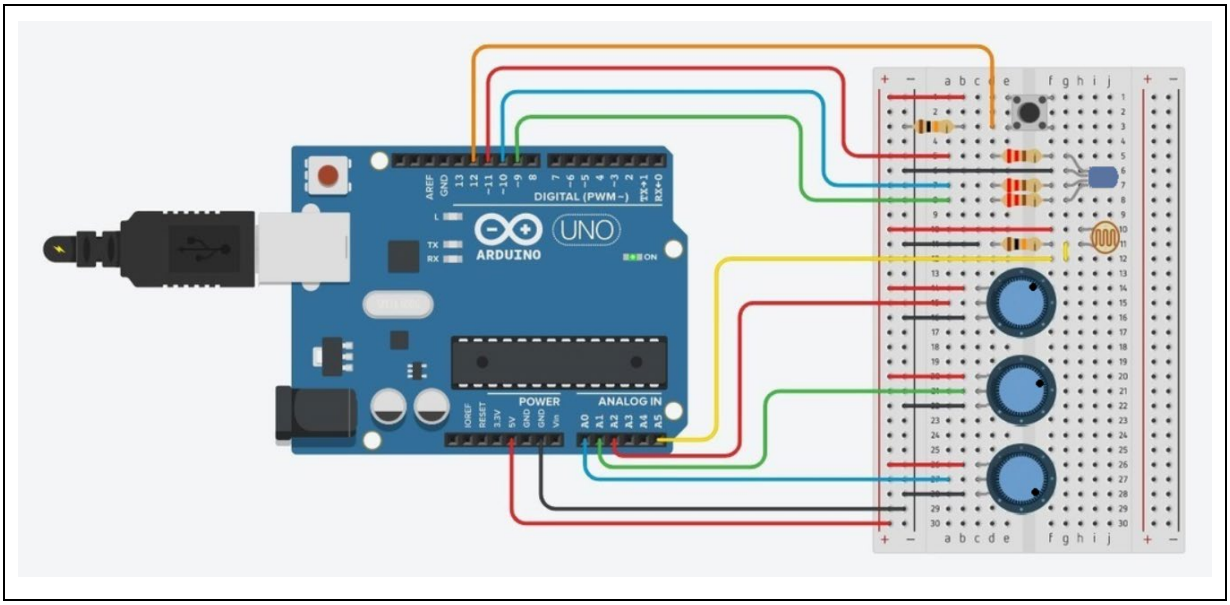
En effet, celle-ci dispose de 4 broches, 1 cathode et 3 anodes. Chaque anode correspond à une couleur (Rouge, Vert et Bleu).
En modulant les signaux sur les anodes, il est possible d'obtenir de multiples couleurs. C'est le principe de la synthèse additive des couleurs des pixels des écrans d'ordinateur ou de télévision.

- Dans un premier temps, la DEL RVB est allumée en appuyant sur le bouton poussoir,
- La luminosité des DELs varie en fonction de la tension des entrées A2, A1 et A0,
- La DEL RVB est éteinte en appuyant de nouveau sur le bouton poussoir.



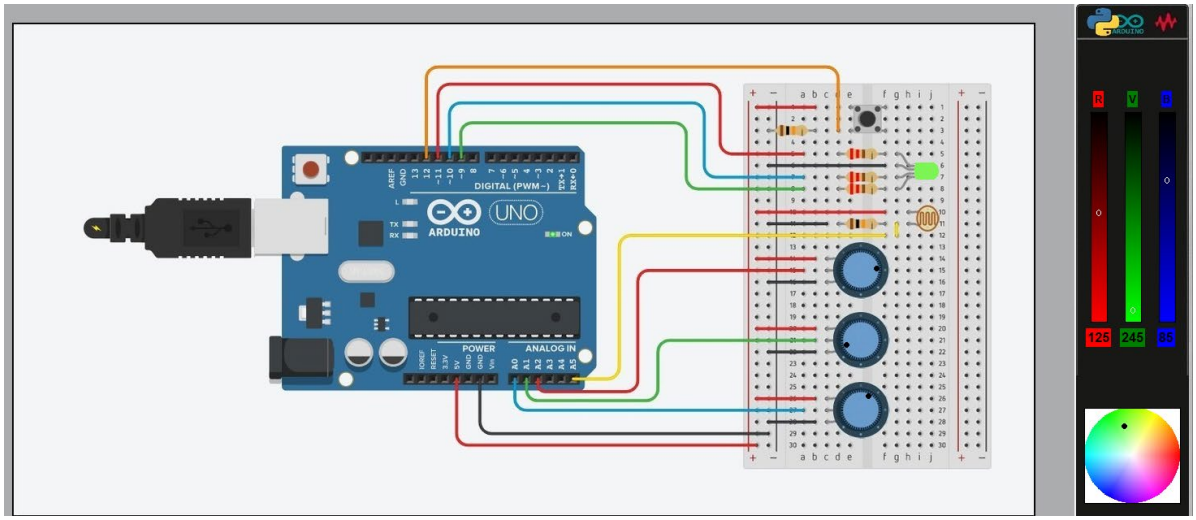
ARDUINO LAB – Fenêtre « Projet 2 – Activité 4 »

Si le mode de fonctionnement est le "contrôle de l'Arduino", la DEL RVB du circuit réel et la DEL RVB sur l'écran s'allument ou s'éteignent en appuyant sur le bouton poussoir du circuit réel ou sur celui du circuit affiché sur l'écran. La luminosité des DELs (réelle et virtuelle) est réglée, soit à l'aide des potentiomètres réels, soit à l'aide des potentiomètres du circuit sur l'écran (**la molette de la souris permet de régler le potentiomètre quand la souris est dessus**).



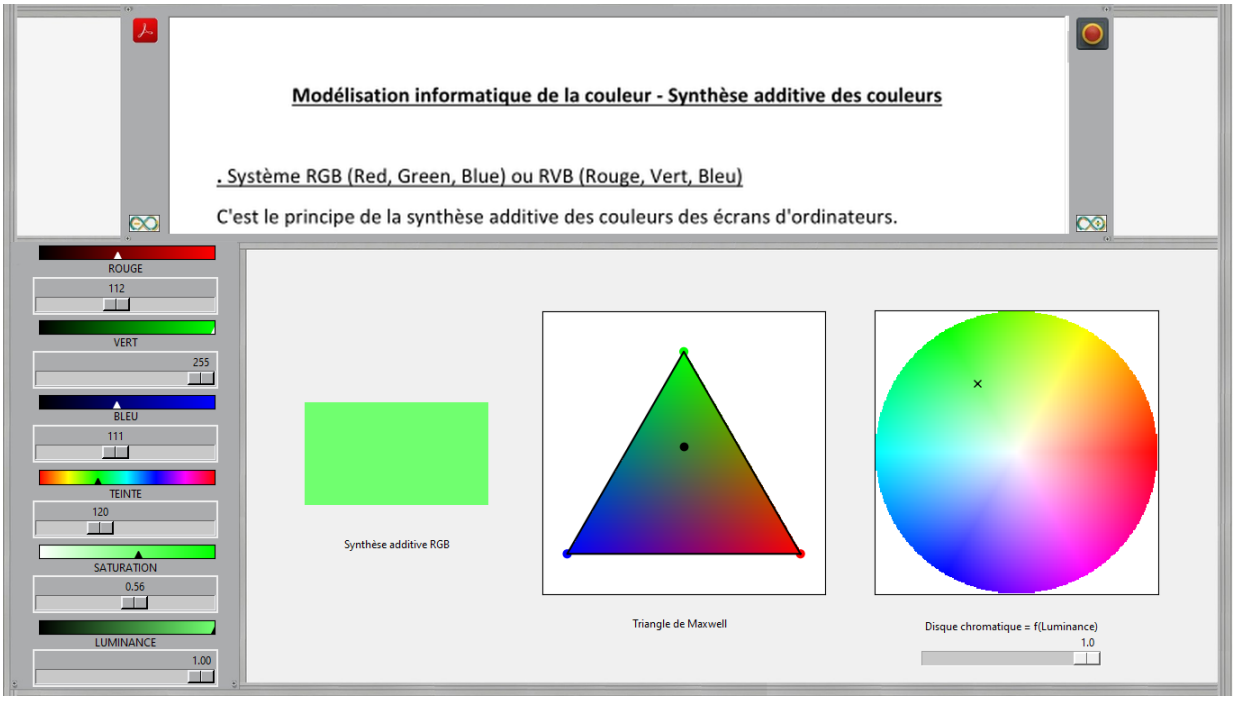
ARDUINO LAB – « Synthèse additive RVB »

Après avoir cliqué sur le connecteur USB, un menu avec les valeurs des composantes Rouge, Verte et Bleue de la couleur produite par synthèse additive et son emplacement sur un disque chromatique est affiché :



En cliquant sur le disque chromatique, une nouvelle fenêtre expliquant le principe de la synthèse additive RVB (Rouge, Vert, Bleu) s'ouvre.

Dans cette fenêtre, il est possible de modifier les valeurs des composantes RVB, de voir le déplacement de la couleur produite sur le disque chromatique, de choisir une couleur sur celui-ci et de connaitre ses composantes RVB :

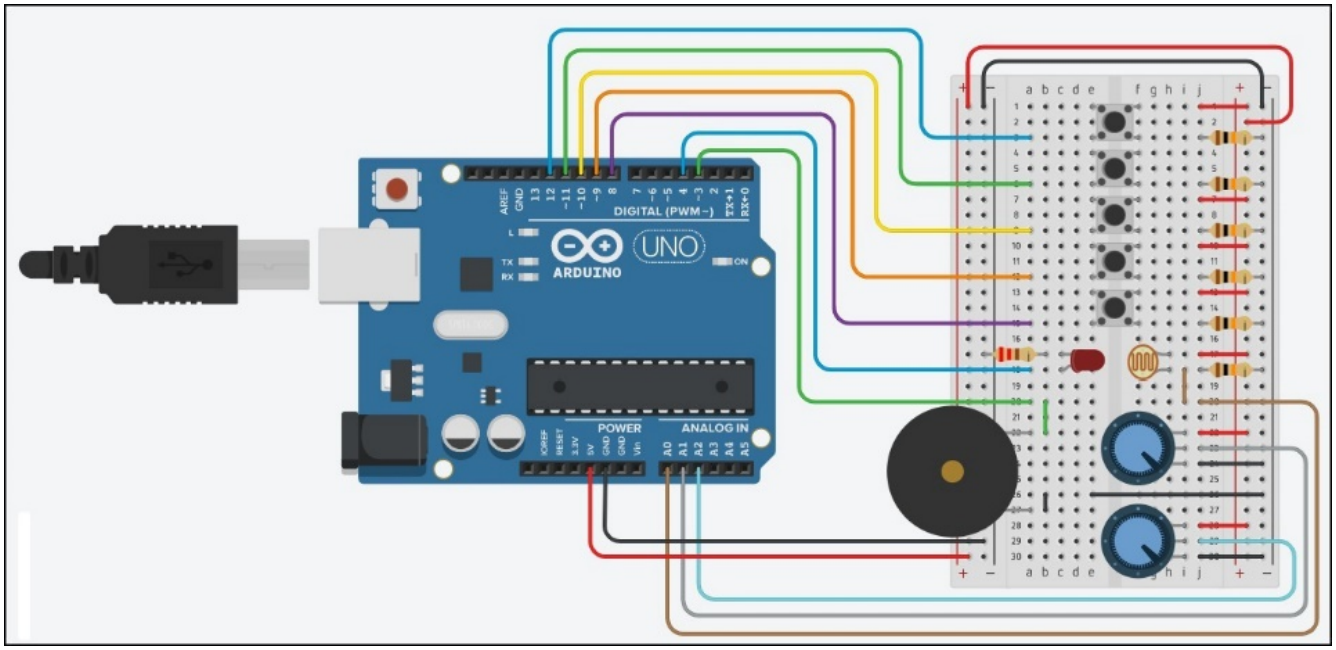


La fenêtre "Synthèse additive des couleurs " est fermée en cliquant sur : 

A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

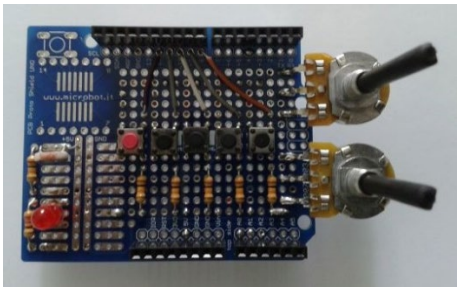
2.5.3 Ondes sonores : Produire & Exploiter

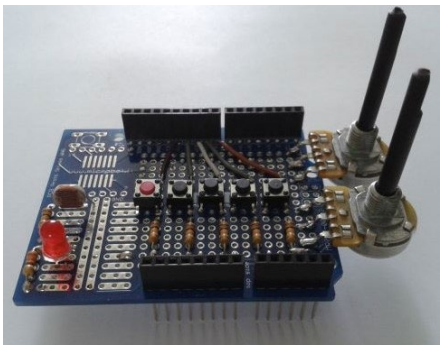
Après avoir bien travaillé, maintenant que nous maitrisons le principe des entrées et sorties de l'Arduino, nous allons nous détendre en écoutant un peu de musique…



- Liste des composants :

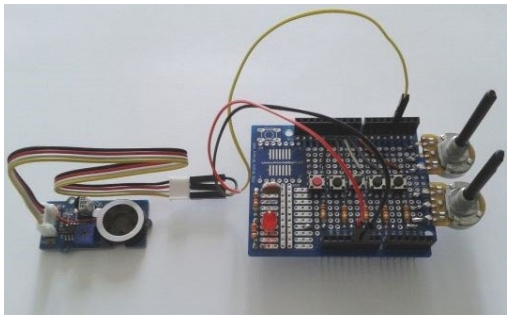
. 1 DEL Rouge
. 1 résistances de 220 Ω
. 6 résistances de 10 kΩ
. 2 potentiomètres de 10 kΩ
. 1 photorésistance
. 6 boutons poussoir
. 1 haut-parleur (ou piezzo)
. 1 plaque d'essai
. Fils de connexion



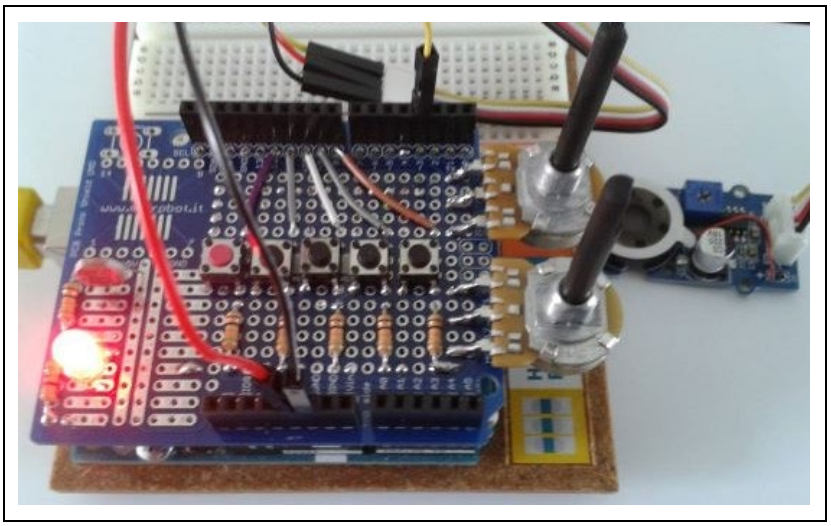


- Protocole de communication (Mode "Contrôle de l'Arduino") :

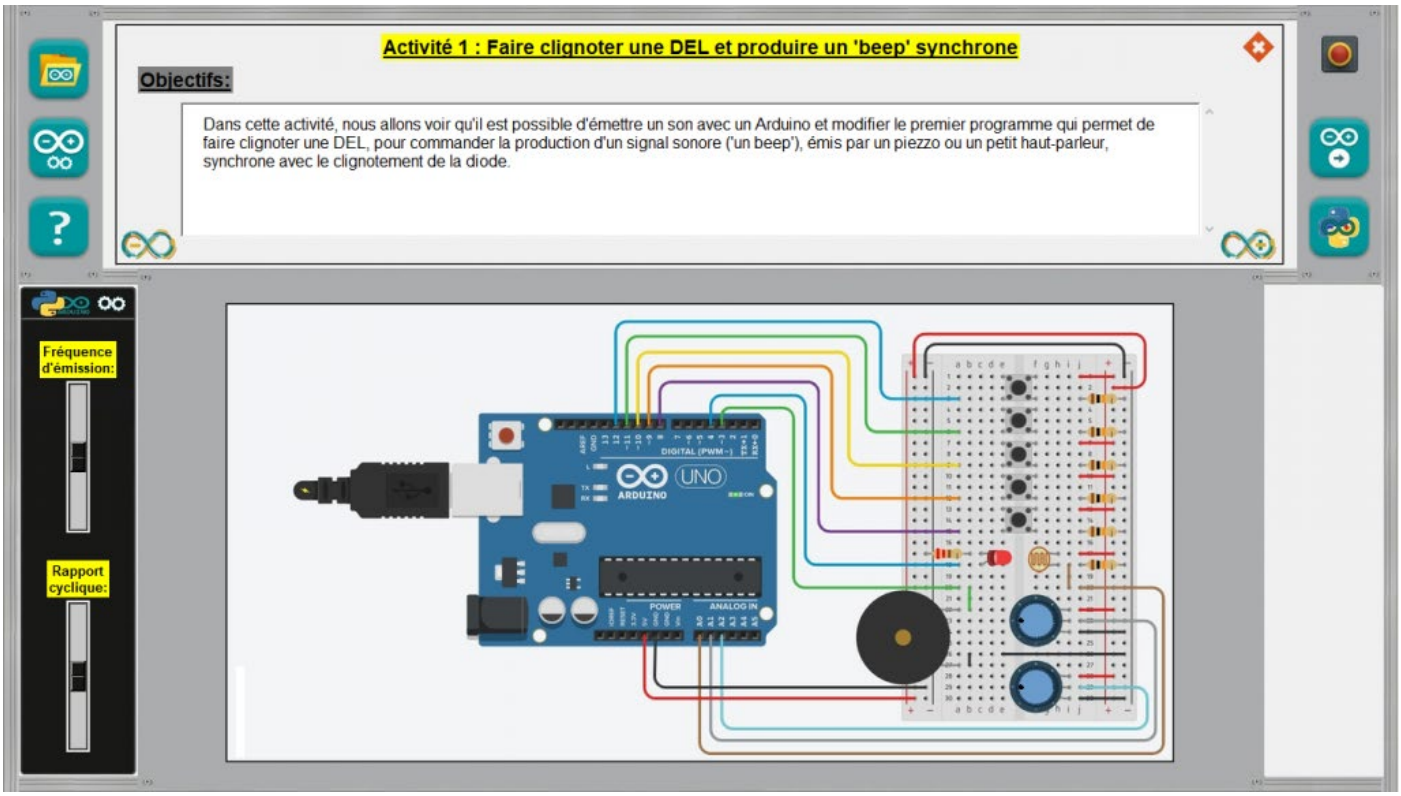
. Firmata Express



- Activité 1 : Faire clignoter une DEL et produire un "beep" synchrone



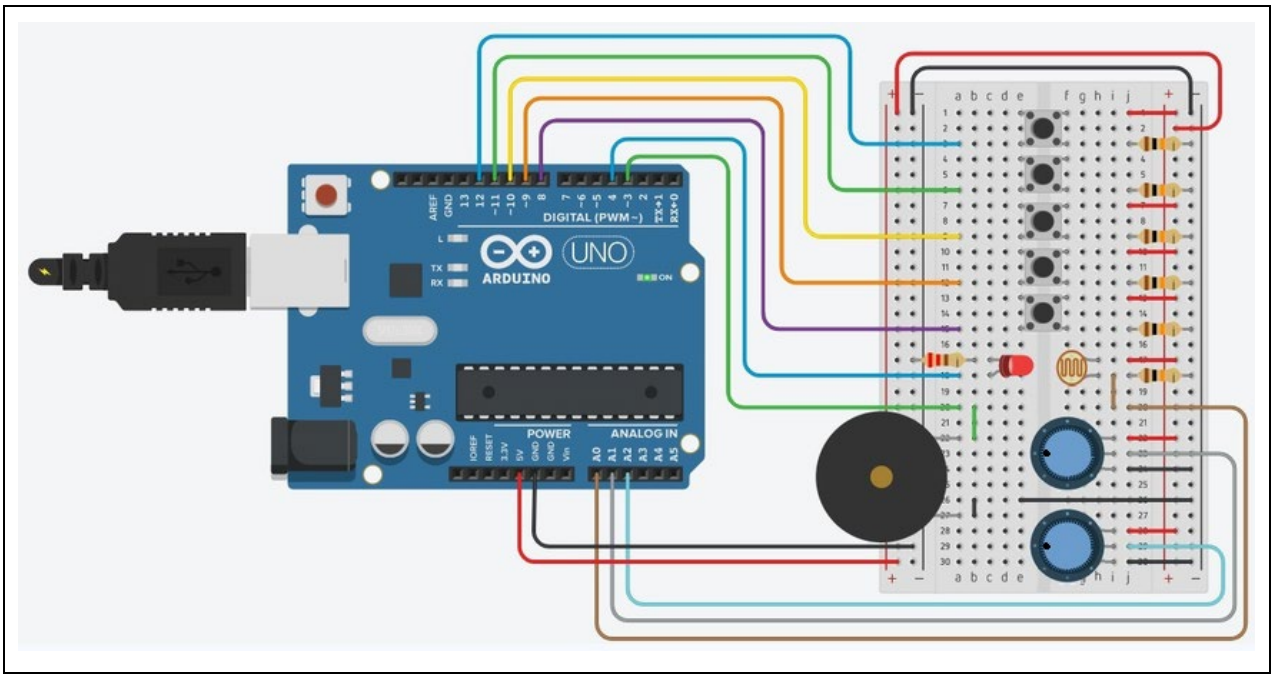
Dans cette activité, nous allons voir qu'il est possible d'émettre un son avec un Arduino et modifier le premier programme qui permet de faire clignoter une DEL, pour commander la production d'un signal sonore ("un beep"), émis par un buzzer ou un petit haut-parleur, synchrone avec le clignotement de la diode.



Après avoir cliqué sur la prise USB, un menu permettant de régler la fréquence d'émission du "beep" et son rapport cyclique (rapport de la durée d'émission et de la durée de silence) est affiché.

Si le mode de fonctionnement est le "contrôle de l'Arduino", la DEL du circuit réel et la DEL sur l'écran clignote et un "beep" est émis à la fréquence et au rapport cyclique sélectionnés.

En mode "simulation", **ARDUINO LAB** utilise le lecteur audio du système pour l'émission du signal sonore.
Il est possible que le clignotement de la diode et l'émission sonore soient légèrement désynchronisés.

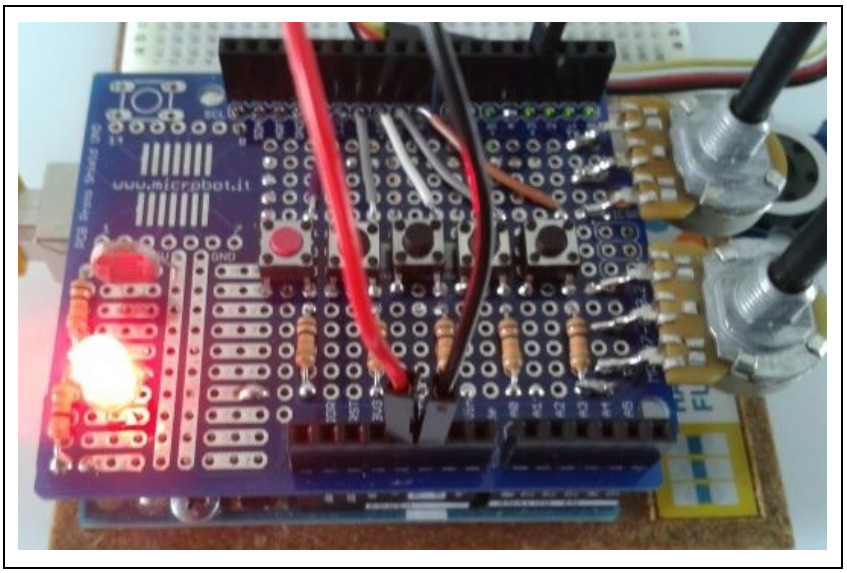


ARDUINO LAB – « La DEL clignote et un "beep" est émis »

A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

Le code pourra être modifié pour voir l'influence des variables (fréquence de l'onde sonore, durée d'émission, durée de silence).

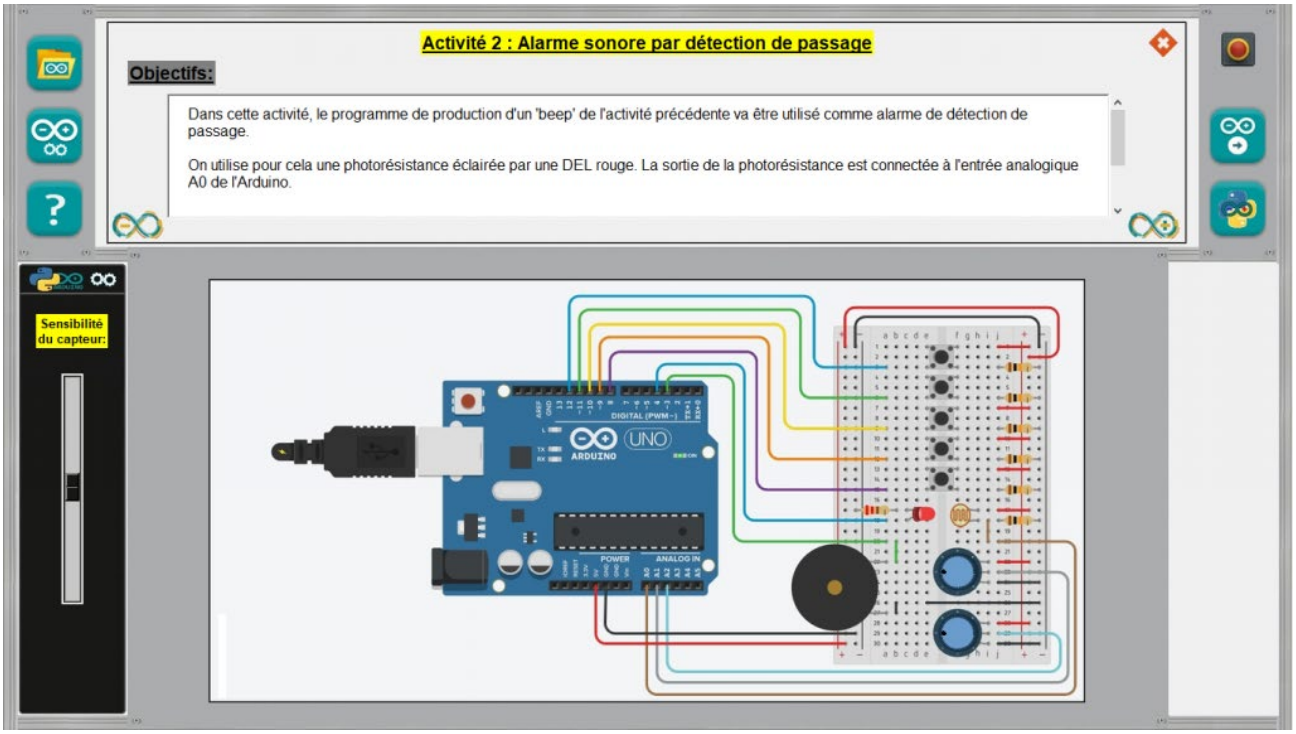
- Activité 2 : Alarme sonore par détection de passage



Dans cette activité, le programme de production d'un "beep" de l'activité précédente va être utilisé comme alarme de détection de passage.

On utilise pour cela une photorésistance éclairée par une DEL rouge. La sortie de la photorésistance est connectée à l'entrée analogique **A0** de l'Arduino.
La valeur de la broche **A0** est alors proportionnelle à l'intensité lumineuse reçue par la photorésistance.

En présence d'un obstacle entre la DEL et la photorésistance, la tension mesurée au niveau de la broche A0 diminue et quand celle-ci est inférieure à un seuil (la sensibilité du capteur définie initialement), l'alarme sonore est déclenchée.

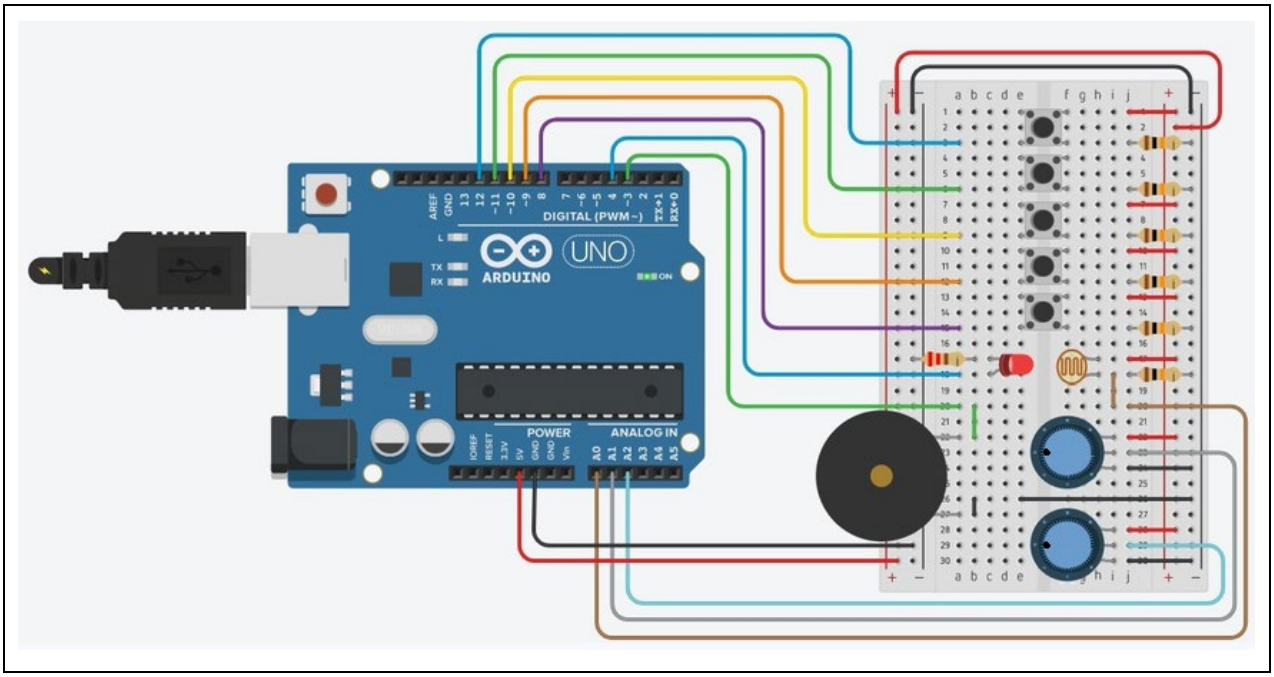


Après avoir cliqué sur le connecteur USB, un menu permettant de régler la sensibilité du capteur est affiché (sauf en mode "simulation") et les DELs réelle et virtuelle s'allument.

Si le mode de fonctionnement est le "contrôle de l'Arduino", en présence d'un obstacle entre la DEL et la photorésistance réelles déclenche l'alarme si la sensibilité de déclenchement a été bien réglée.

De même, l'alarme est déclenchée lors du passage de la souris entre la DEL et la photorésistance virtuelles.

Il s'agit donc bien d'une alarme sonore par détection de passage.

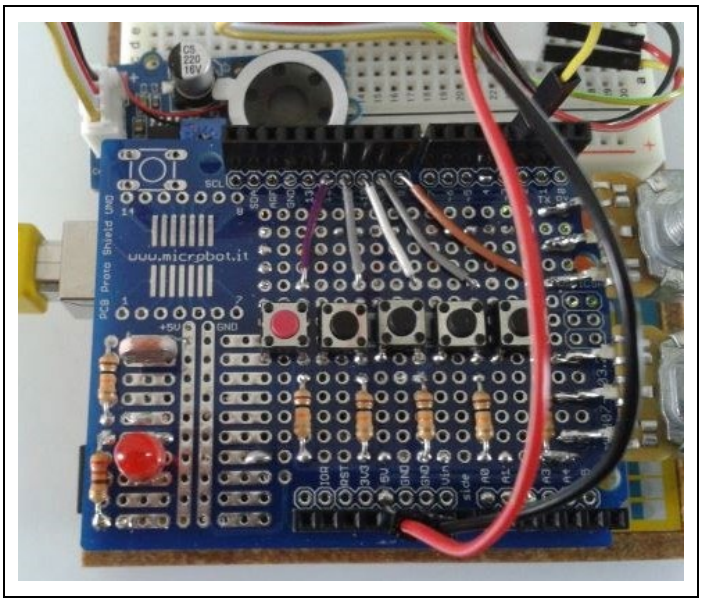


ARDUINO LAB – « L'alarme est déclenchée en cas d'obstacle entre la DEL et la photorésistance »

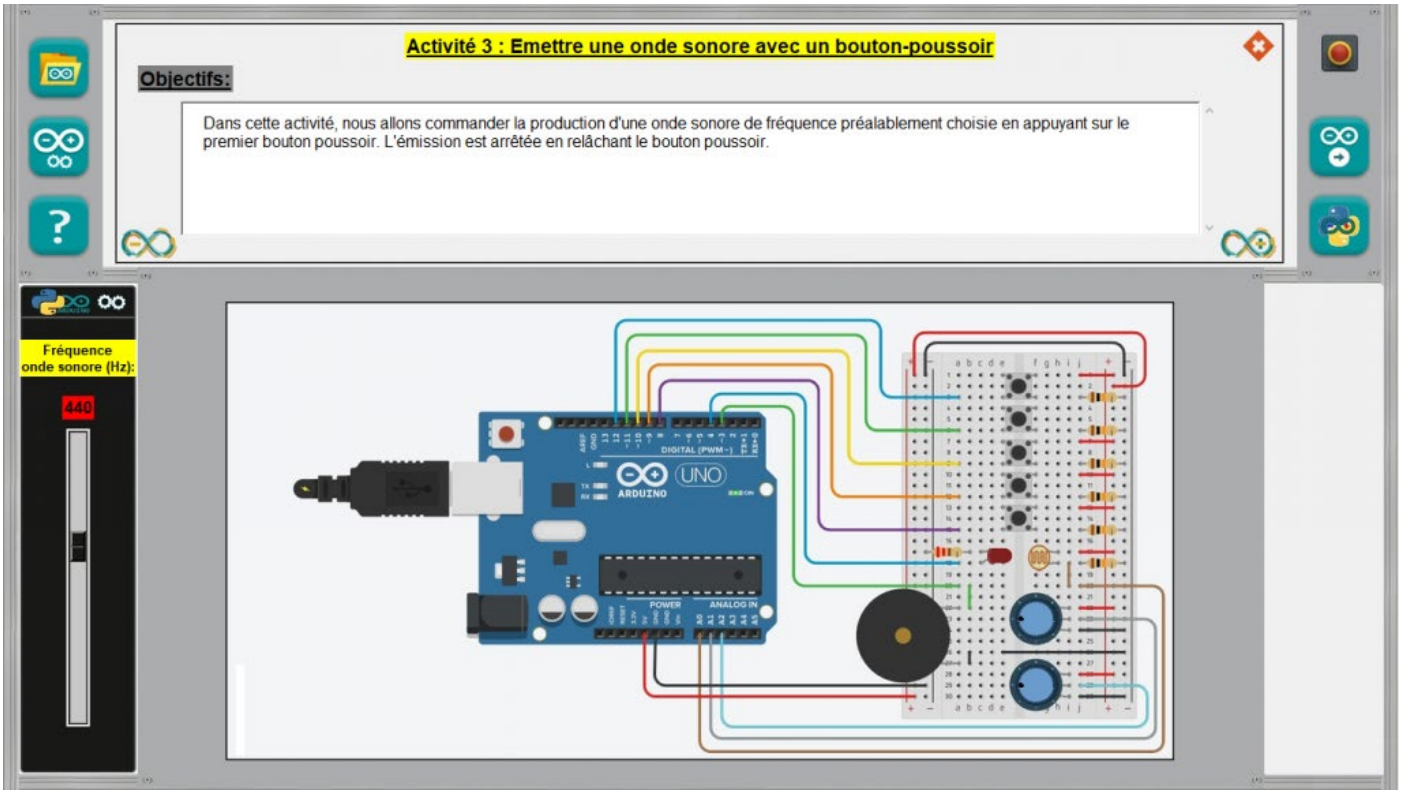
A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

Le code pourra être modifié pour voir l'influence des variables (sensibilité du capteur, fréquence de l'onde sonore, durée d'émission, durée de silence).

- Activité 3 : Emettre une onde sonore avec un bouton-poussoir



Dans cette activité, nous allons commander la production d'une onde sonore de fréquence préalablement choisie en appuyant sur le premier bouton poussoir. L'émission est arrêtée en relâchant le bouton poussoir.



ARDUINO LAB – Fenêtre « Projet 3 – Activité 3 »

Après avoir cliqué sur le connecteur USB, un menu permettant de régler la fréquence (en Hz) de l'onde sonore est affiché.

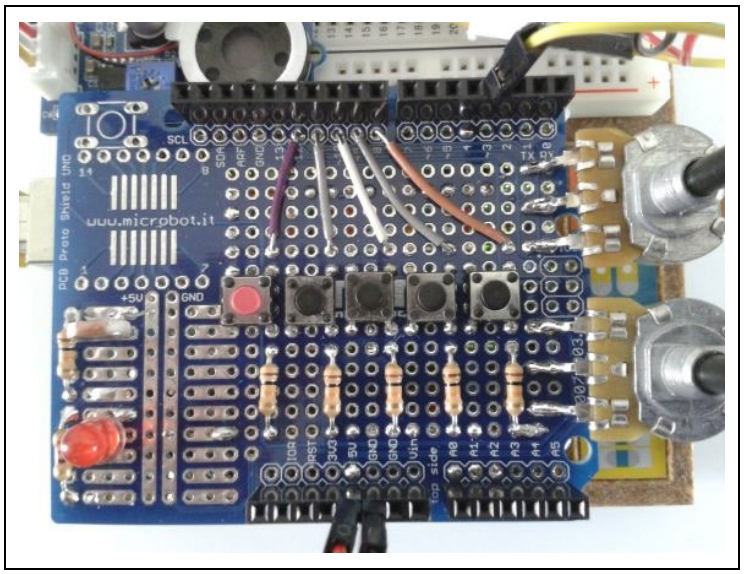
Si le mode de fonctionnement est le "contrôle de l'Arduino", un appui sur le premier bouton poussoir réel ou virtuel déclenche l'émission d'une onde sonore de fréquence égale à la valeur sélectionnée et affichée (de 100 à 1000 Hz par pas de 1 Hz).

En mode "simulation", ARDUINO LAB utilise le lecteur audio du système pour l'émission du signal sonore. Les fréquences d'ondes disponibles sont celles des notes de musique de A2 (110 Hz) à C8 (4186 Hz).

A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

Le code pourra être modifié pour voir l'influence des variables (fréquence onde sonore en Hz).

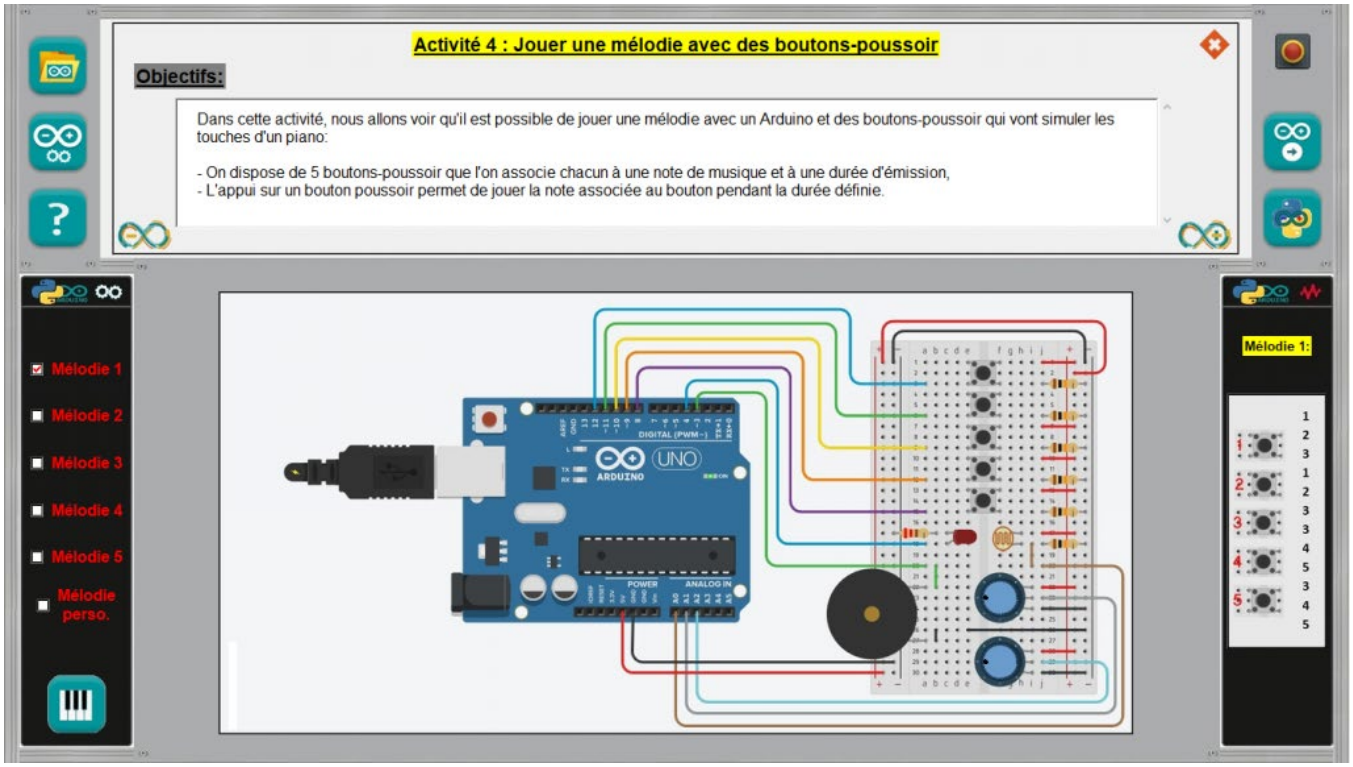
- Activité 4 : Jouer une mélodie avec des boutons poussoir



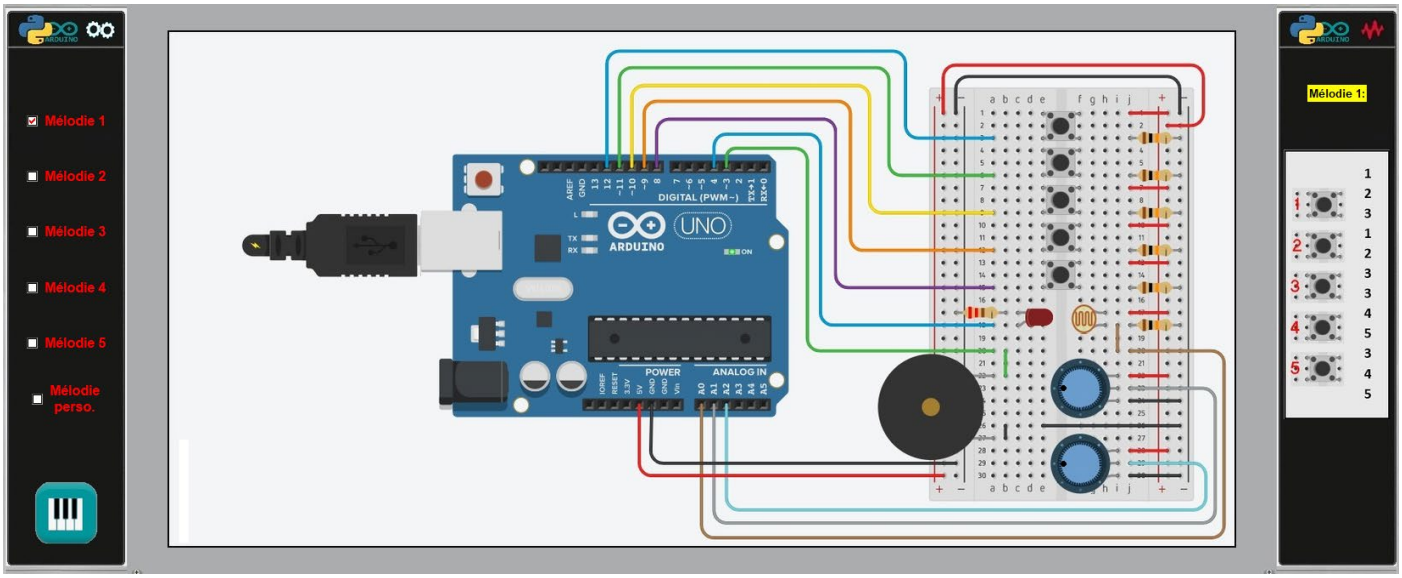
Dans cette activité, nous allons voir qu'il est possible de jouer une mélodie avec un Arduino et des boutons poussoir qui vont simuler les touches d'un piano :

- On dispose de 5 boutons poussoir que l'on associe chacun à une note de musique (une onde sonore de fréquence déterminée en Hz - voir tableau des fréquences des notes de musique) et à une durée d'émission,

- L'appui sur un bouton poussoir permet de jouer la note associée au bouton pendant la durée définie.



Après avoir cliqué sur le connecteur USB, deux menus permettant de choisir une mélodie et montrant la partition de la mélodie choisie sont affichés de chaque côté du circuit :

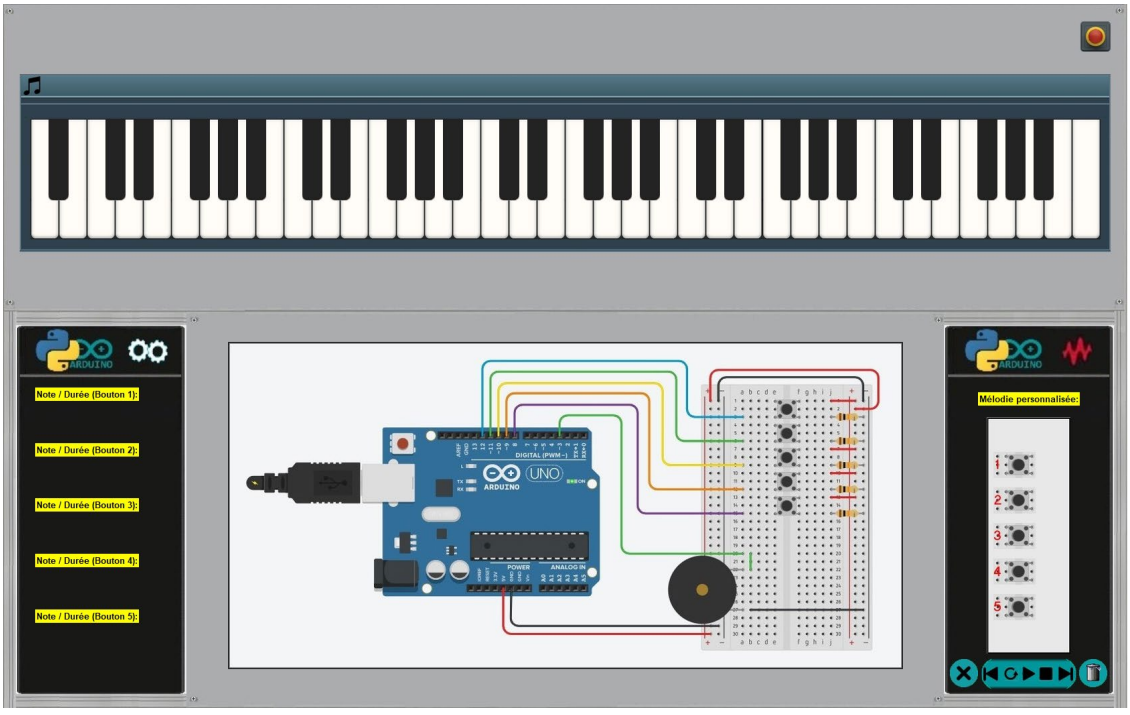


Si le mode de fonctionnement est le "contrôle de l'Arduino", la mélodie est jouée aussi bien en appuyant sur les boutons poussoir réels que virtuels, en suivant la partition (liste dans l'ordre des boutons à appuyer)

En mode "simulation", ARDUINO LAB utilise le lecteur audio du système pour jouer les notes. Le rythme de la mélodie est plus lent.

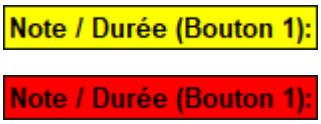
Vous disposez de 5 mélodies préenregistrées et de la possibilité de créer une mélodie personnalisée en cliquant sur le bouton :

En cliquant sur ce bouton, une nouvelle fenêtre apparait :

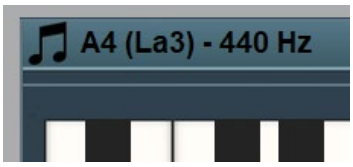


Il est alors possible de jouer une mélodie directement en cliquant sur les touches du piano (notes de musique de A2 à C8) ou de concevoir une mélodie selon ce procédé :

- dans le menu de gauche, cliquer sur un encadré en fond jaune représentant chacun une note associée à un bouton, le fond de l'encadré devient alors rouge,



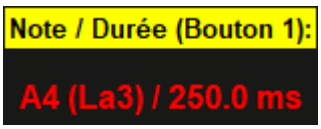
- ensuite, cliquer sur une touche de piano correspondant à la note souhaitée pour le bouton sélectionné (La note et sa fréquence sont affichées quand une touche du piano est survolée avec la souris),



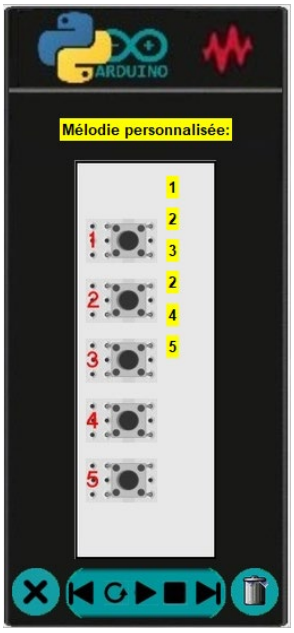
- un encadré représentant différents types de note (ronde, blanche, noire, croche, double croche) apparait alors, cliquer sur le type de note souhaité afin de régler la durée pendant laquelle la note sera jouée,

. Ronde = 1000 ms
. Blanche = 500 ms
. Noire = 250 ms
. Croche = 125 ms
. Double croche = 67,5 ms

- La note et sa durée sont alors affichées sous l'encadré (redevenu en fond jaune) précédemment sélectionné :



-Dans le menu "Mélodie personnalisée", composer votre mélodie en cliquant sur les boutons du menu. La mélodie est affichée au fur et à mesure,



- La mélodie créée est supprimée en cliquant sur :

,

- La dernière note de la mélodie est effacée en cliquant sur :

,

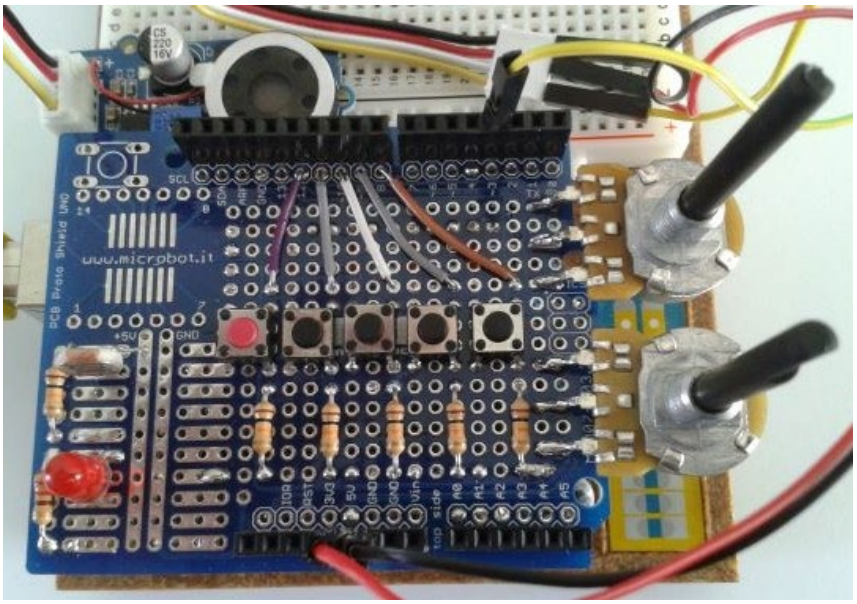
- la barre de lecture  permet de jouer la mélodie personnalisée (de façon répétée en option) ou celles préenregistrées en cliquant sur :  ou  pour sélectionner une mélodie.

La fenêtre "Mélodie personnalisée" est fermée en cliquant sur : 

A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

Le code pourra être modifié pour voir l'influence des variables (fréquence des notes associées aux boutons en Hz, durée de la note)

Activité 5 : Régler la fréquence d'une onde sonore avec deux potentiomètres



Dans cette dernière activité, l'appui sur le premier bouton-poussoir produit une onde sonore dont la fréquence est réglée à l'aide de 2 potentiomètres :

- le premier potentiomètre permet un réglage rapide de la fréquence entre 0 et 4080 Hz,
- le deuxième potentiomètre effectue un réglage fin de la fréquence sur une plage de 255 Hz,
- l'émission sonore est arrêtée en appuyant de nouveau sur le bouton poussoir.

Le potentiomètre de réglage rapide est connecté sur la broche **A1** de l'Arduino. La tension de cette broche varie donc entre **0** et **5** V (voir le principe de fonctionnement du potentiomètre) en fonction de la position du curseur du potentiomètre. La lecture de la valeur de la broche **A1** convertie par le convertisseur analogique numérique de l'Arduino donne donc un nombre entier entre **0** et **1023**.

Ce nombre est divisé par 4 de façon à obtenir un nombre entier compris entre **0** et **255** qui sera convertie en nombre binaire (sur 8 bits) :

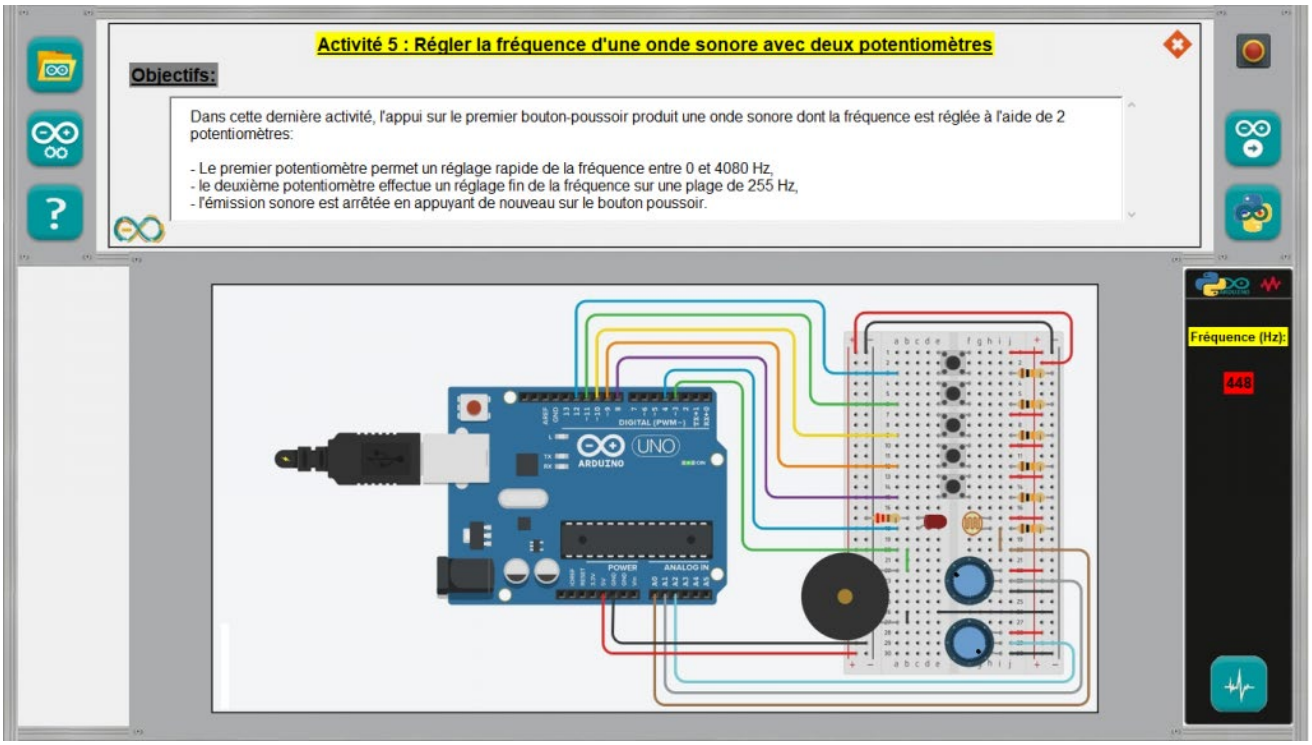
0 en décimal = **00000000** en binaire
255 en décimal = **11111111** en binaire

Ce nombre binaire sur 8 bits est convertie en nombre binaire sur 12 bits en ajoutant 4 bits de poids faibles, **0000**, à sa fin. On obtient donc un nombre binaire (sur 12 bits) compris entre **000000000000** et **111111110000**, soit en décimal, un nombre entier entre **0** et **4080.**

Le potentiomètre de réglage fin est connecté sur la broche **A2** de l'Arduino. Selon le même principe que précédemment, la lecture de la broche **A2** donne une valeur comprise entre **0** et **1023**.

Ce nombre est également divisé par 4 et convertie en nombre binaire sur 12 bits. On obtient donc un nombre binaire compris entre **0000000000** et **00001111111111** (entre 0 et 255 en décimal).

La conversion en décimal de l'addition des deux nombres binaires (issus de A1 et A2) nous donnent la valeur de la fréquence en Hz de l'onde sonore, soit entre **0** et **4335** Hz avec un pas de réglage de 1 Hz.



ARDUINO LAB – Fenêtre « Projet 3 – Activité 5 »

Après avoir cliqué sur le connecteur USB, un menu permettant de visualiser la fréquence (en Hz) de l'onde sonore est affiché.

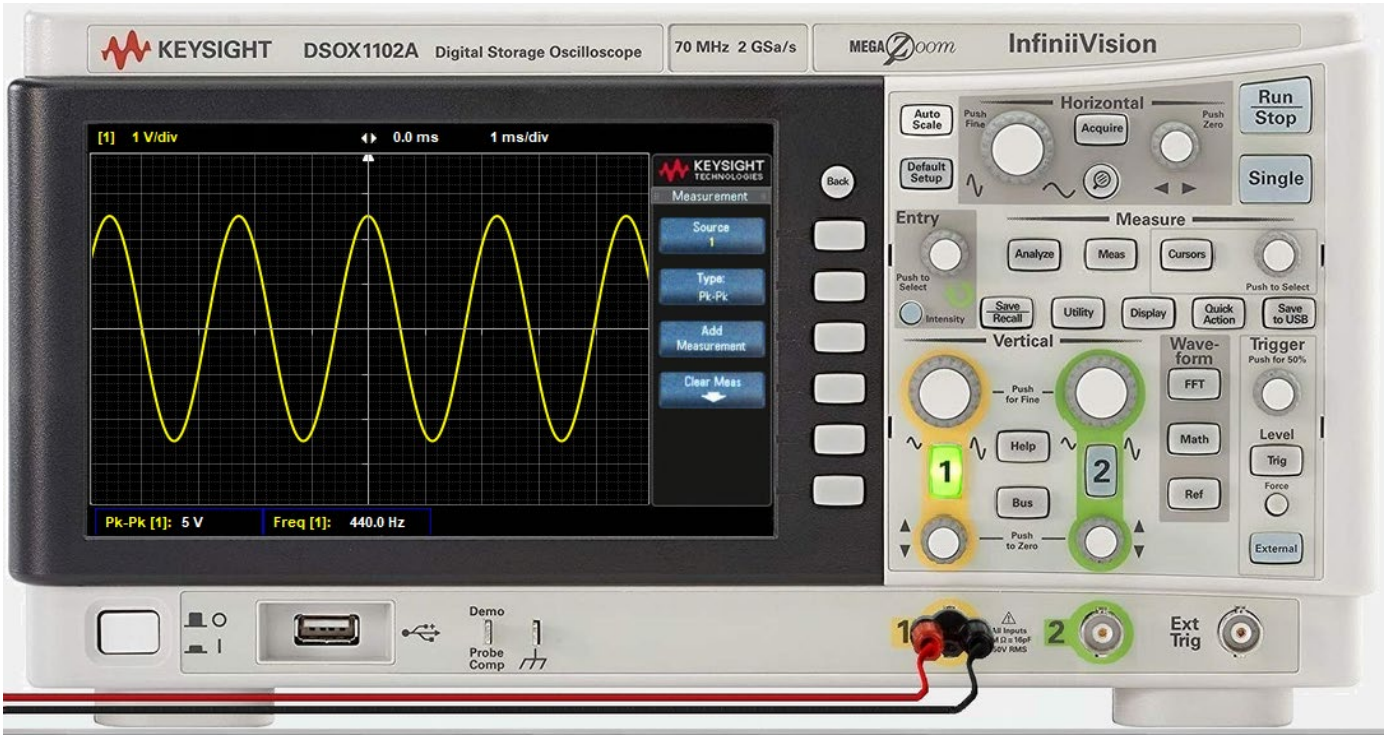
Si le mode de fonctionnement est le "contrôle de l'Arduino", un appui sur le premier bouton poussoir réel ou virtuel déclenche l'émission de l'onde sonore. Le réglage de la fréquence est fait aussi bien avec les potentiomètres réels qu'avec les virtuels (par utilisation de la molette de la souris quand celle-ci est au-dessus du potentiomètre)

En mode "simulation", ARDUINO LAB utilise le lecteur audio du système d'exploitation pour l'émission du signal sonore. Les fréquences d'ondes disponibles sont celles des notes de musique de A2 (110 Hz) à C8 (4186 Hz).

Il est possible de visualiser la simulation de l'observation de l'acquisition de l'onde sonore par un microphone sur un oscilloscope en cliquant sur ce bouton :

Une nouvelle fenêtre est alors affichée :

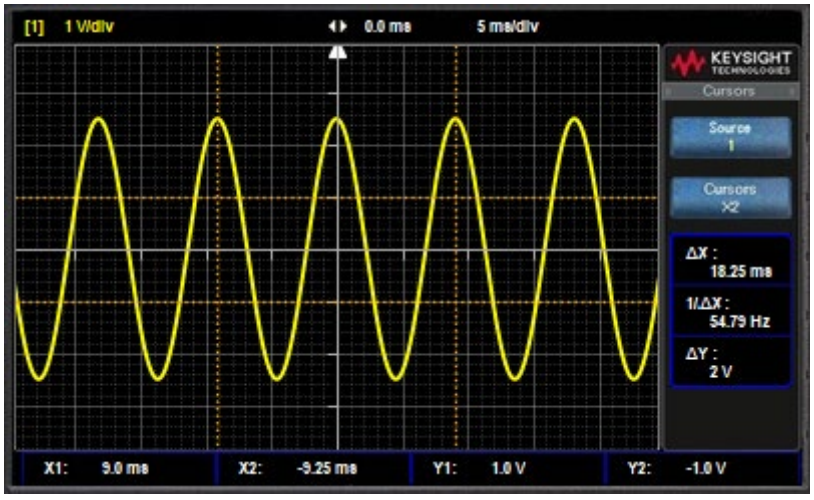
Après avoir cliqué sur l'interrupteur On/Off de l'oscilloscope, l'onde sonore, de fréquence F en Hz, acquise (virtuellement) par le microphone est affichée :



Tous les réglages classiques d'un oscilloscope (balayage horizontal, sensibilité verticale, affichage des mesures, curseurs, …) sont simulés et donc modifiables (soit par clic sur les boutons, soit par utilisation de la molette de la souris quand celle-ci est au-dessus d'un bouton de réglage) permettant ainsi l'apprentissage de son utilisation.

Les valeurs d'amplitude et de fréquence sont affichées en appuyant sur : 

ou en utilisant les curseurs en cliquant sur : 



En mode "contrôle de l'Arduino", la modification de la fréquence de l'onde sonore par les potentiomètres est visible sur l'oscilloscope.

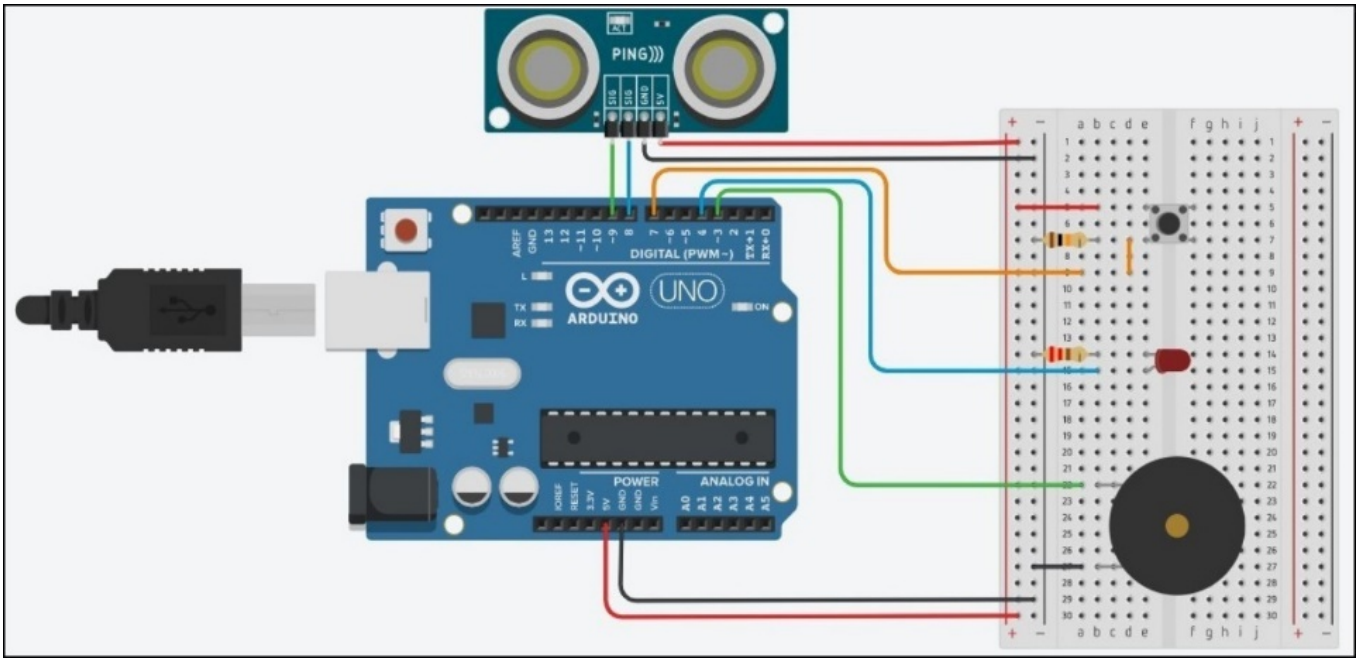
Enfin, le haut-parleur se déplace pour visualiser l'influence, sur l'amplitude du signal, de la distance entre la source sonore (le haut-parleur) et le capteur d'acquisition (le microphone).

La fenêtre "Acquisition / Observation de l'onde sonore " est fermée en cliquant sur : 

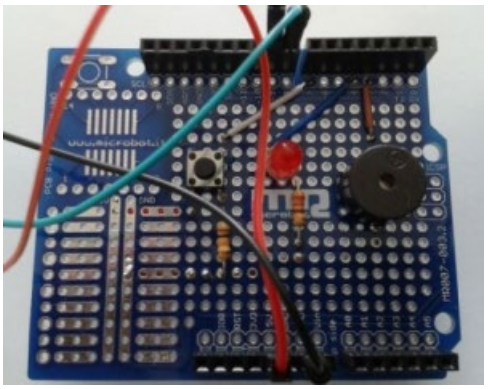
A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

2.5.4 Ondes ultrasonores : Vitesse & Distances

Nous avons vu qu'il était possible, avec un Arduino, de produire des ondes sonores, caractérisées par leur fréquence. Nous allons maintenant, nous intéresser à la vitesse de propagation des ondes sonores.



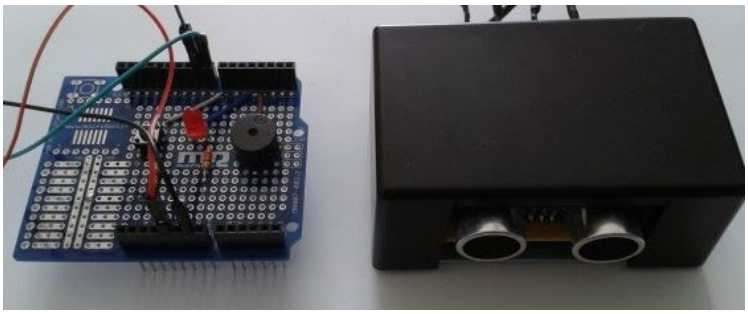
- Liste des composants :



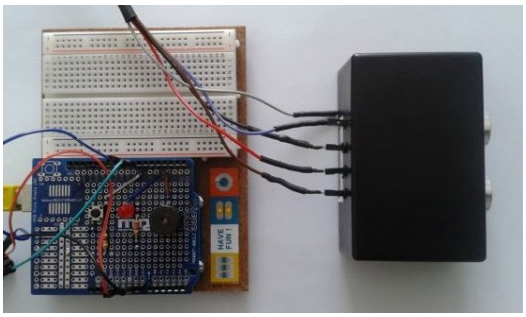
. 1 capteur ultrasonique (par exemple, le HC-SR04)
. 1 DEL Rouge
. 1 résistance de 220 Ω
. 1 résistance de 10 kΩ
. 1 bouton poussoir
. 1 haut-parleur (ou piezzo)
. 1 plaque d'essai
. Fils de connexion

- Protocole de communication (Mode "Contrôle de l'Arduino") :

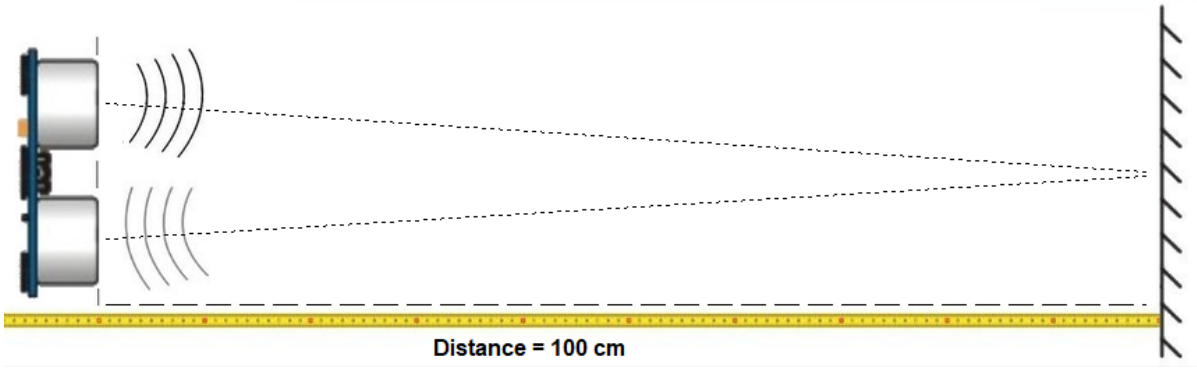
. Firmata Express

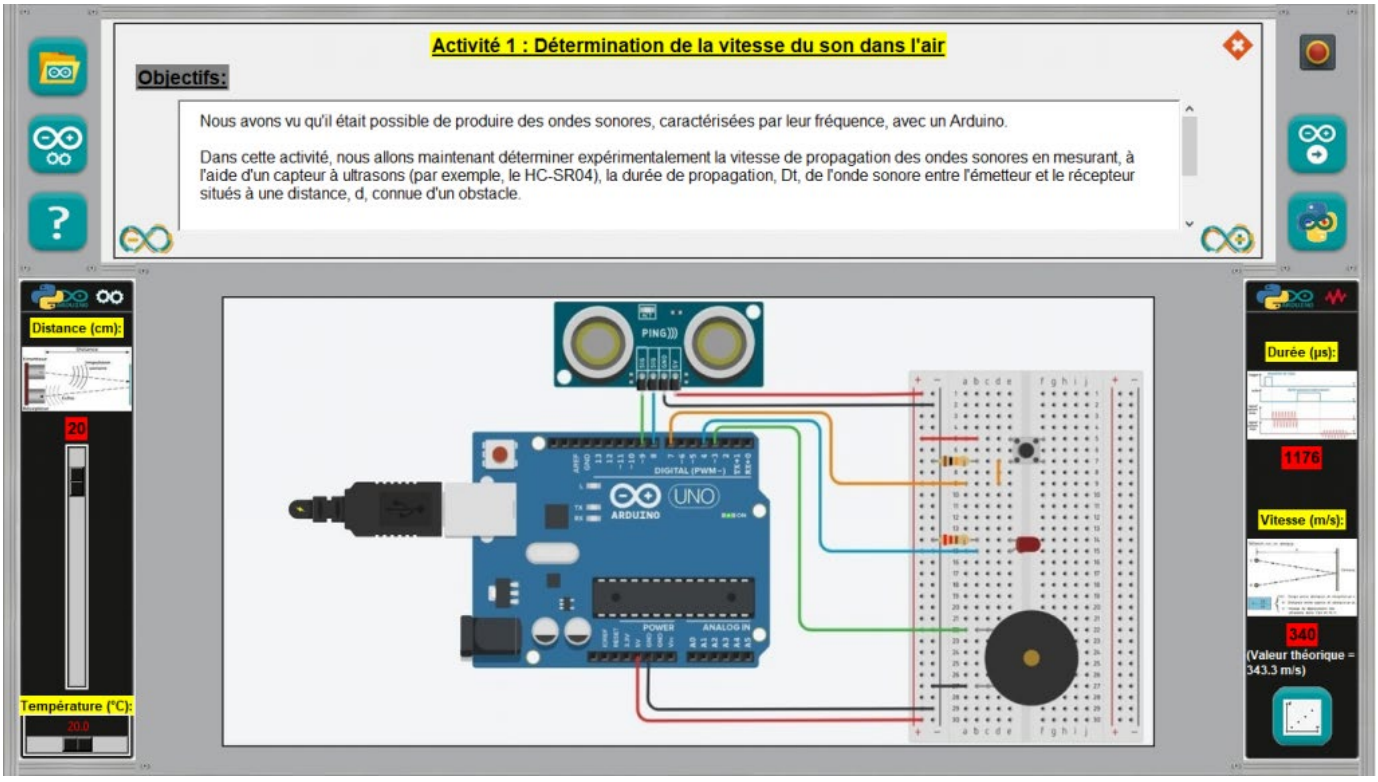


- Activité 1 : Détermination de la vitesse du son dans l'air



Dans cette activité, nous allons déterminer expérimentalement la vitesse de propagation des ondes sonores en mesurant, à l'aide d'un capteur à ultrasons (par exemple, le HC-SR04), la durée de propagation, Dt, de l'onde sonore entre l'émetteur et le récepteur situés à une distance, d, connue d'un obstacle.

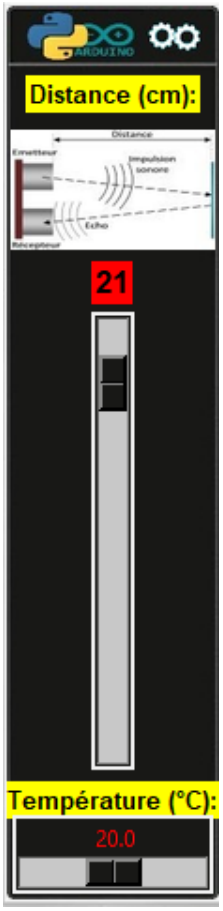




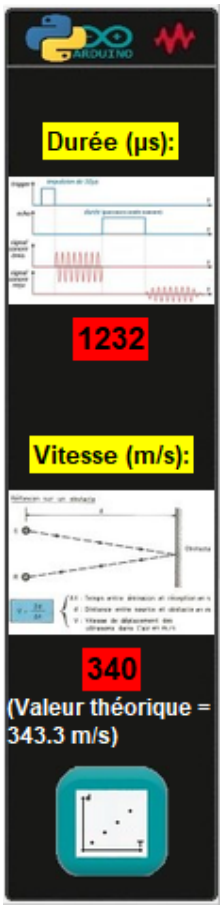
Après avoir cliqué sur la prise USB, un menu permettant de régler la distance entre le capteur et l'obstacle, et la température de l'air, est affiché.

Si le mode de fonctionnement est le "contrôle de l'Arduino", un appui sur le bouton poussoir réel ou virtuel déclenche la mesure de la durée de propagation de l'onde ultrasonore. Celle-ci est affichée, dans le menu "Résultats".
La vitesse de propagation en m/s est alors calculée et affichée. La vitesse théorique du son dans l'air en fonction de la température indiquée est également affichée afin de pouvoir la comparer à la valeur déterminée expérimentalement.

En mode "simulation", ARDUINO LAB utilise la valeur théorique de la vitesse de propagation des sons dans l'air à la température indiquée par l'utilisateur et affiche la durée de propagation de l'onde calculée avec cette valeur.



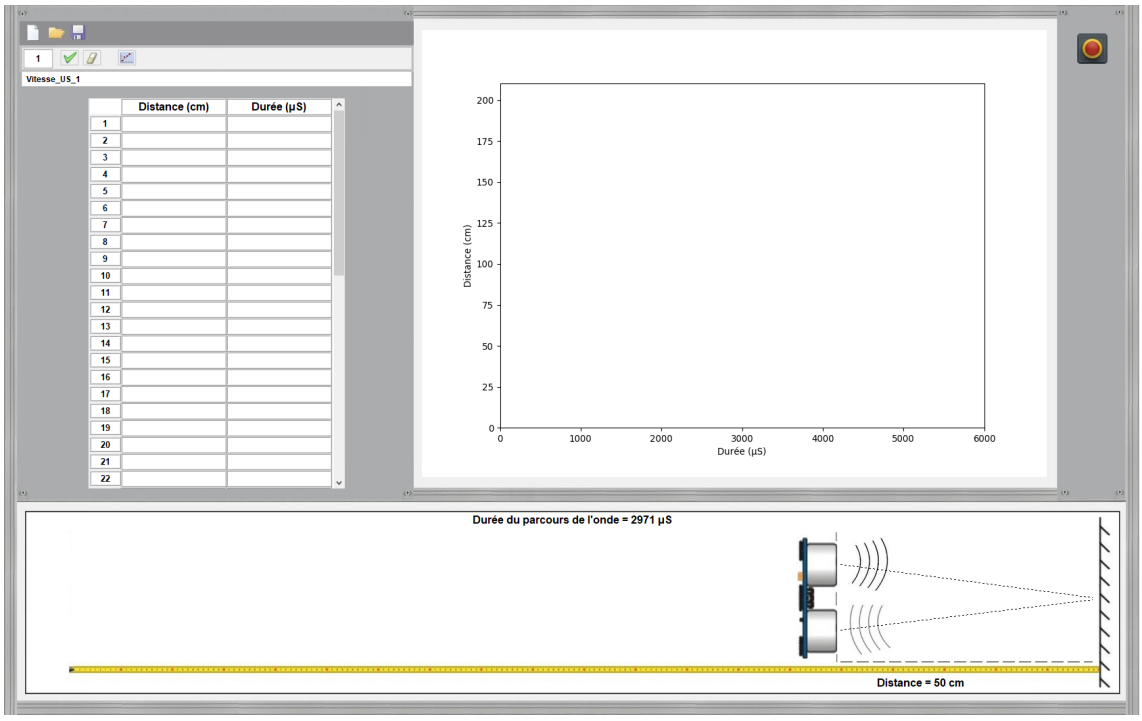
Menu "Réglages"



Menu "Résultats"

Il est possible d'enregistrer plusieurs durées de propagation pour différentes distances et de représenter graphiquement la distance en fonction de la durée afin de déterminer une vitesse moyenne du son dans l'air en cliquant sur :

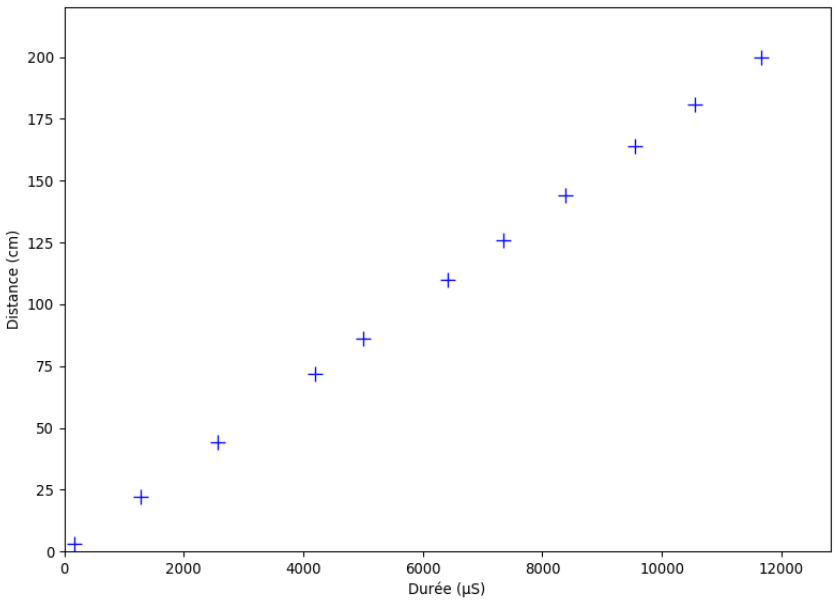
Une nouvelle fenêtre est alors affichée :



En mode "contrôle de l'Arduino", il suffit de positionner le capteur ultrasonique virtuel à la même distance que le capteur réel, la durée de propagation de l'onde est alors affichée. Puis afin d'enregistrer les valeurs, de cliquer sur :  (pour effacer un point, cliquer sur : )

Les valeurs de distance et de durée de propagation sont affichées dans le tableur et le graphe est tracé automatiquement.

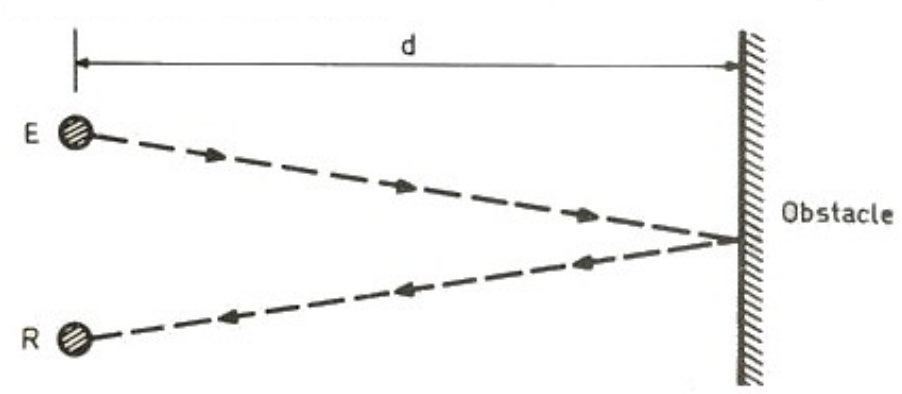
	Distance (cm)	Durée (µS)
1	3	174
2	22	1281
3	44	2563
4	72	4194
5	86	5010
6	110	6408
7	126	7340
8	144	8389
9	164	9554
10	181	10545
11	200	11652
12		
13		
14		
15		
16		
17		
18		
19		
20		
21		
22		



En mode "simulation", la durée de propagation est calculée, en utilisant la valeur théorique de la vitesse des sons dans l'air à la température indiquée et la distance choisie par l'utilisateur.

La modélisation de la distance en fonction de la durée est réalisée en appuyant sur : 

On rappelle que la vitesse de propagation de l'onde ultrasonore est :



$$V = \frac{2d}{\Delta t}$$

- Δt : Temps entre émission et réception en s
- d : Distance entre source et obstacle en m
- V : Vitesse de déplacement des ultrasons dans l'air en m/s

La modélisation de **d = f(Dt)** est alors : **y = a x** (régression linéaire)

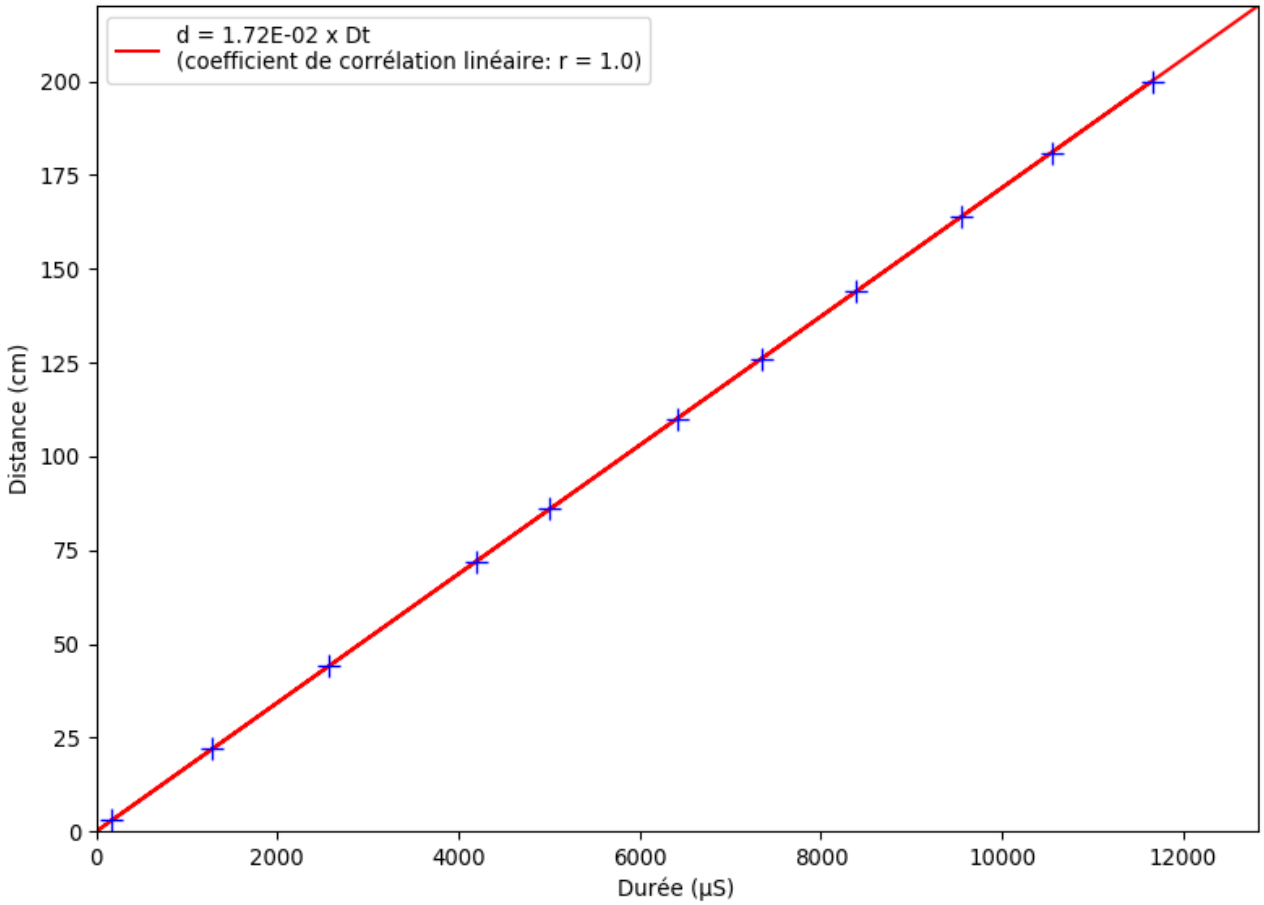
Où **a** est le coefficient directeur de la droite, avec : **a = V / 2**

On en déduit alors : **V = 2 a**

<u>Attention</u> :

Les mesures effectuées sont exprimées en cm pour la distance et en µS pour la durée. La vitesse calculée est donc en **cm/µS**. Il faut multiplier le résultat par 10000 pour obtenir une vitesse en **m/s**.

La droite de modélisation est alors affichée :



. Les données peuvent être enregistrées dans un fichier csv en cliquant sur : 

. Un fichier de mesures est ouvert en cliquant sur : 

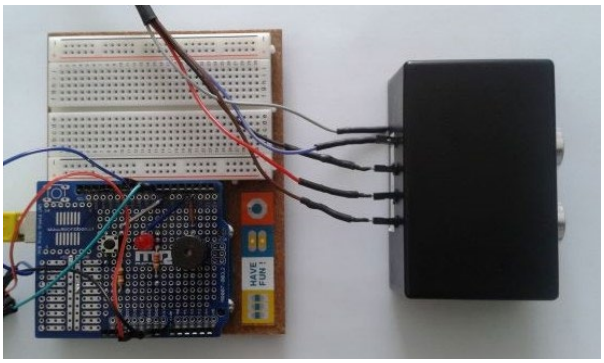
. Un nouveau fichier de mesure est créé en cliquant sur : 

La fenêtre "Détermination de la vitesse du son dans l'air" est fermée en cliquant sur : 

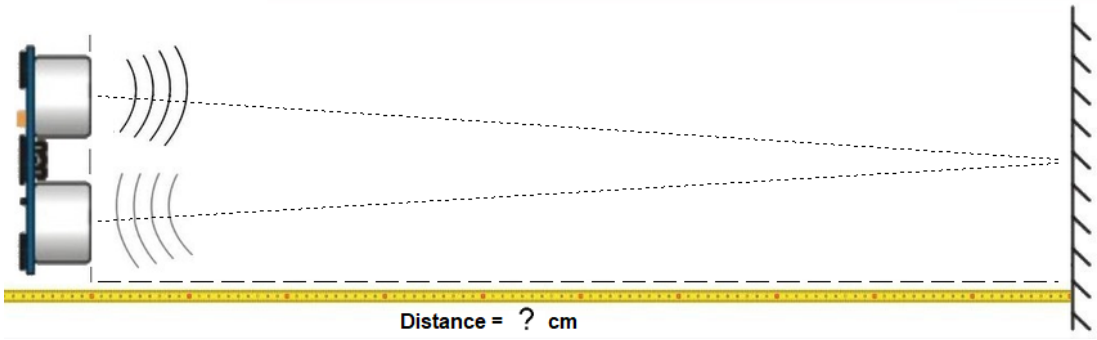
Les mesures sont arrêtées en appuyant de nouveau sur le bouton poussoir réel ou virtuel, en mode "Contrôle de l'Arduino", ou virtuel en mode "Simulation".

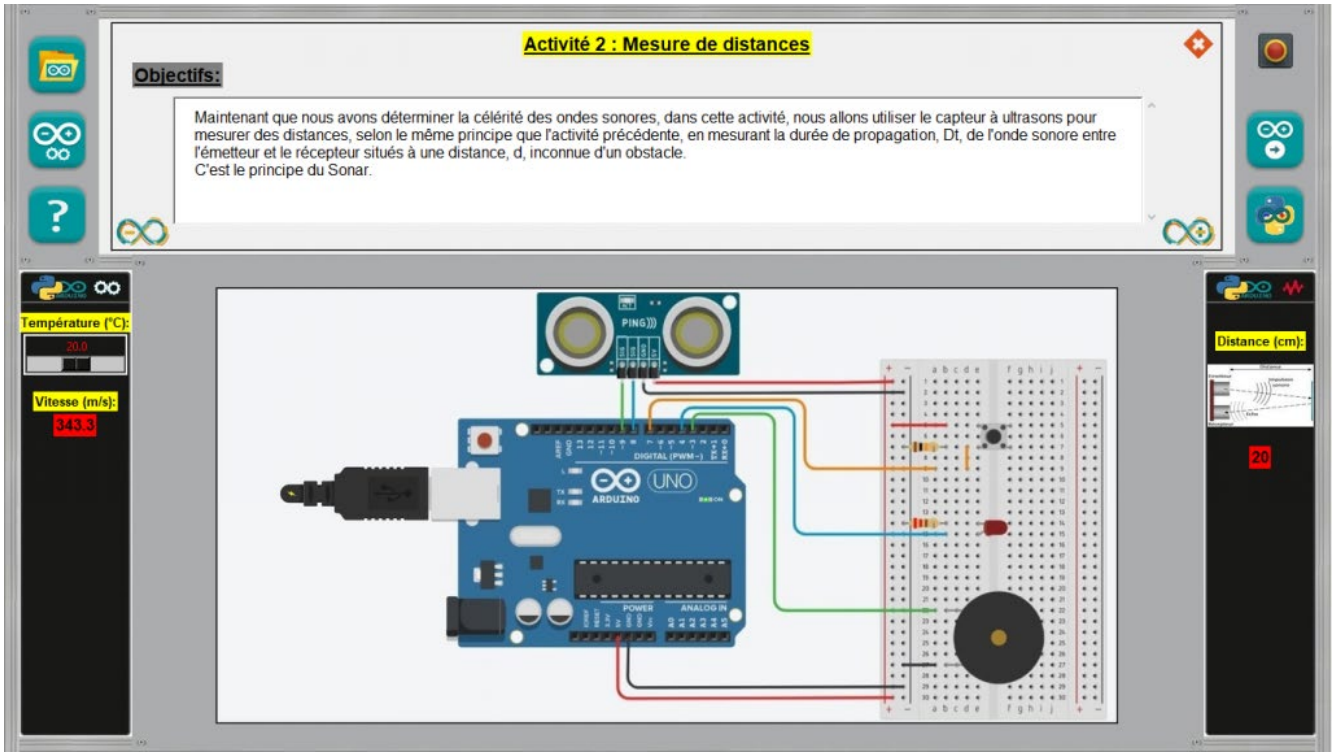
A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

- Activité 2 : Mesure de distances



Maintenant que nous avons déterminé la célérité des ondes sonores dans l'air, dans cette activité, nous allons utiliser le capteur à ultrasons pour mesurer des distances, selon le même principe que l'activité précédente, en mesurant la durée de propagation, Dt, de l'onde sonore entre l'émetteur et le récepteur situés à une distance, d, inconnue d'un obstacle. C'est le principe du Sonar.





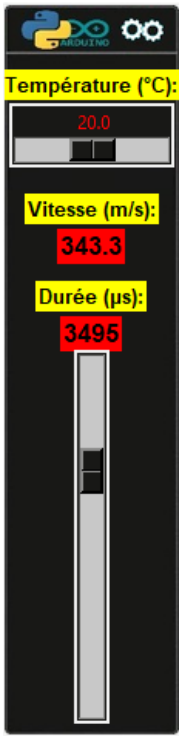
Si le mode de fonctionnement est le "contrôle de l'Arduino", après avoir cliqué sur la prise USB, un menu permettant de régler la température de l'air est affiché. La vitesse théorique du son dans l'air en fonction de la température indiquée est alors calculée et affichée.

Un appui sur le bouton poussoir réel ou virtuel déclenche la mesure de la durée de propagation, Dt, de l'onde ultrasonore.
La distance, en cm, entre le capteur et l'obstacle est alors calculée, à partir de Dt et de la célérité théorique du son dans l'air, et affichée dans le menu "Résultats".

Menu "Réglages"

Menu "Résultats"

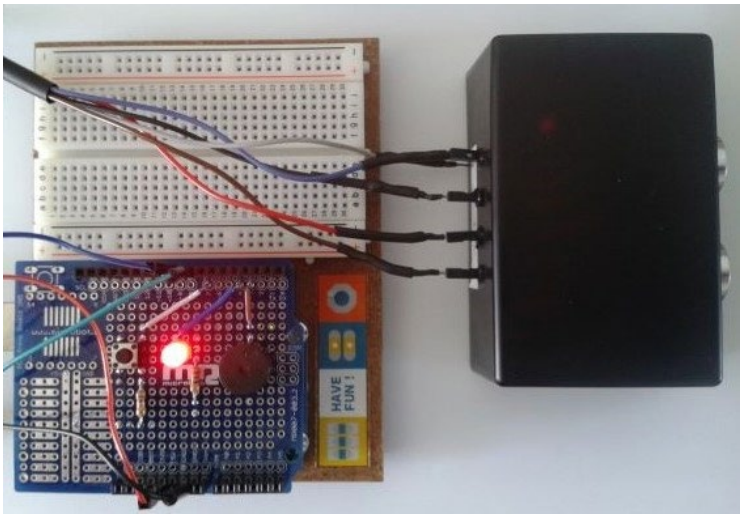
En mode "simulation", l'utilisateur doit régler la durée de propagation, Dt, des ondes ultrasonores dans le menu "Réglages" et après avoir appuyé sur le bouton poussoir virtuel, la distance, en cm, correspondant à ce Dt est calculée et affichée le menu "Résultats".



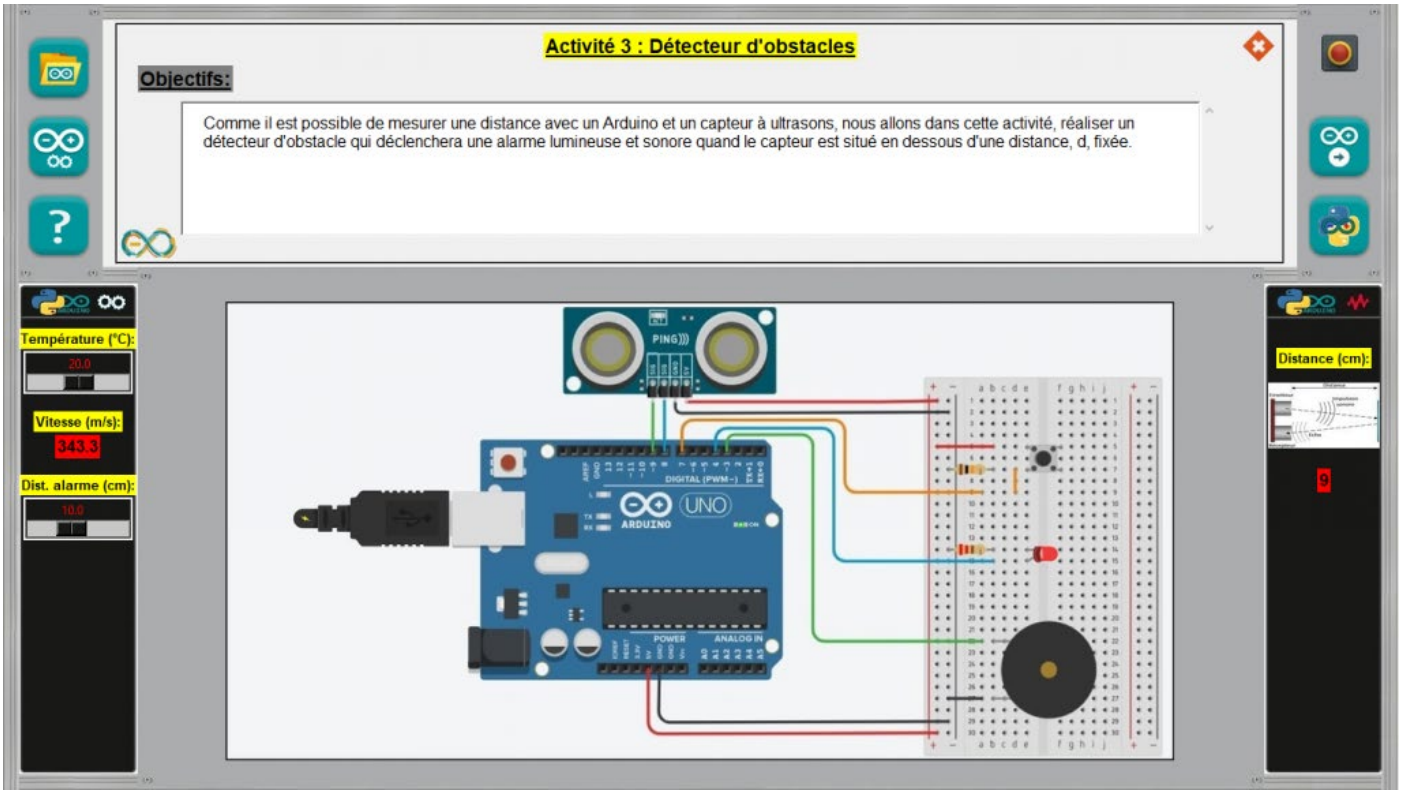
Les mesures sont arrêtées en appuyant de nouveau sur le bouton poussoir réel ou virtuel, en mode "Contrôle de l'Arduino", ou virtuel en mode "Simulation".

A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :

- Activité 3 : Détecteur d'obstacles



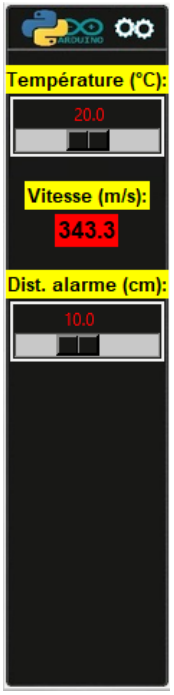
Comme il est possible de mesurer une distance avec un Arduino et un capteur à ultrasons, nous allons dans cette activité, réaliser un détecteur d'obstacle qui déclenchera une alarme lumineuse et sonore quand le capteur est situé en dessous d'une distance, d, fixée.



ARDUINO LAB – Fenêtre « Projet 4 – Activité 3 »

Si le mode de fonctionnement est le "contrôle de l'Arduino", après avoir cliqué sur la prise USB, un menu permettant de régler la température de l'air est affiché. La vitesse théorique du son dans l'air en fonction de la température indiquée est alors calculée et affichée. La distance minimale de déclenchement de l'alarme est également choisie dans ce menu.

Un appui sur le bouton poussoir réel ou virtuel déclenche la mesure de la durée de propagation, Dt, de l'onde ultrasonore.
La distance, en cm, entre le capteur et l'obstacle est alors calculée, à partir de Dt et de la célérité théorique du son dans l'air, et affichée dans le menu "Résultats".



Menu "Réglages"

Menu "Résultats"

Comme dans l'activité précédente, en mode "simulation", l'utilisateur doit régler la durée de propagation, Dt, des ondes ultrasonores dans le menu "Réglages" et après avoir appuyé sur le bouton poussoir virtuel, la distance, en cm, correspondant à ce Dt est calculée et affichée le menu "Résultats".

Dans les deux cas, si la distance calculée est inférieure à la distance minimale, l'alarme est déclenchée.

Les mesures sont arrêtées en appuyant de nouveau sur le bouton poussoir réel ou virtuel, en mode "Contrôle de l'Arduino", ou virtuel en mode "Simulation".

A tout moment, il est possible de visualiser le code et son algorithme, programmé en langage Arduino IDE ou en Python, permettant de réaliser cette activité, en cliquant sur les boutons :