Thème : informatique embarquée et objets connectés

TI-83 Premium CE
EDITION PYTHON

Classe enseignée : 2nde SNT

Interrupteur distant

F. GIROD

Compétences visées

Un des objectifs de l'enseignement de SNT est de développer et de coder des scripts PYTHON afin d'apporter une réponse à une problématique précise. A travers le thème "**informatique embarquée et objets connectés**", nous pouvons notamment travailler les compétences suivantes dans l'activité proposée :

- Coder des scripts simples d'acquisition de données.
- Gérer des entrées/sorties à travers les ports utilisés par le système.
- Écrire et développer des algorithmes pour résoudre une problématique.
- Identifier des algorithmes de contrôle des comportements physiques à travers les données des capteurs.

Situation déclenchante

Comment allumer une lampe (sa lampe de chevet par exemple) sans appuyer parfaitement sur l'interrupteur ? (si, au matin, on n'est pas très bien réveillé !)

Ici, une **vidéo** illustrant ce projet.

Problématique

Comment reproduire un allumage à distance d'une lampe ?

1. Identifier un capteur possible ;
2. Identifier un actionneur possible ;
3. Élaborer un algorithme modélisant le fonctionnement d'une lampe qui s'allume à distance ;
4. Mettre cet algorithme en application.

Thème : informatique embarquée et objets connectés

TI-83 Premium CE
EDITION PYTHON

Classe enseignée : 2nde SNT

Fiche méthode

F. GIROD

Matériel disponible

Les élèves disposent, en plus de leur TI-83 Premium CE :

- d'un TI-Innovator HUB ;
- d'un capteur de distance à ultrasons.

Déroulement possible du projet

En amont du projet :

Les propositions suivantes permettent de préparer le projet et de le rendre possible sur 2 séances d'1h30. Elles peuvent aussi être préparées par l'enseignant sous forme de fiches et/ou d'un exposé au tableau.

- Un groupe d'élèves peut préparer une fiche ou présenter un rapide exposé illustrant les principaux points pour maîtriser la programmation en PYTHON spécifique au **TI-Innovator HUB**, en s'appuyant sur les **'10 mn de code'** (Unité 6).
- Un groupe d'élèves peut expliquer le **mode de codage RGB** pour les couleurs.
- Un groupe d'élèves peut décrire le **TI-Innovator HUB**.

Différentes évolutions possibles pour le projet :

Les propositions suivantes donnent des pistes pour gérer les différences de vitesse d'exécution des élèves.
On peut aussi considérer ces différentes étapes comme une manière de séquencer le projet pour ceux qui en auraient besoin.

- 1ère étape : on allume une lumière en passant sa main devant le capteur à ultrasons.
- 2ème étape : on allume et on éteint une lumière en passant sa main devant le capteur à ultrasons ; création d'un compteur et discrimination selon sa parité.
- 3ème étape : on adapte l'intensité lumineuse de la lampe selon la luminosité ambiante.

A la suite du projet :

Synthétiser ce travail par rapport au cours de SNT :

- le(s) capteur(s) et le(s) actionneur(s) mis en jeu ;
- l'algorithme qui gère les données d'entrée ;
- le langage de programmation utilisé.

Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus

Fiche méthode

F. GIROD

Proposition de résolutions

Choix du capteur :

- Capteur de distance à ultrasons : en passant sa main à proximité, la distance mesurée passe en dessous d'un seuil qui entraîne le passage d'un mode à l'autre. C'est ce capteur qui a été choisi ici.
- Capteur de luminosité (brightness) : en cachant ce capteur, la luminosité passe en dessous d'un seuil qui entraîne le passage d'un mode à l'autre. On peut l'utiliser si on ne dispose pas de capteur à ultrasons ou si on veut proposer une alternative.

Choix de l'actionneur :

- LED RGB du HUB : elle pourra délivrer une lumière blanche plus ou moins intense.

Etapes de résolution

L'importation de ces bibliothèques est explicitée dans la remarque qui suit cette partie.

```
EDITEUR : LAMPED
LIGNE DU SCRIPT 0007
# Projets STEM Hub
from ti_system import *
from time import *

import color
from ranger import *
Fns… | a A # | Outils | Exéc | Script
```

La fonction **f1** va simuler un allumage automatique de la lumière à distance :

On initialise un compteur stocké dans la variable **c** qui comptera le nombre de fois où l'on passe la main à moins de 10 cm du capteur de distance.
La variable **n** donnera la parité de **c** (par **n=c%2**, reste de la division euclidienne de **c** par 2).
Ces deux variables sont initialisées à 0.

On crée la variable **m** qui représente l'entrée du ranger (capteur de distance) relié au port d'entrée **IN 1** ; les instructions se trouvent dans le module *ranger* importé précédemment.

```
EDITEUR : LAMPED
LIGNE DU SCRIPT 0018
def f1():
    c=0
    n=0
    m=ranger("IN 1")
    while not escape():
        d=m.measurement()
        sleep(0.5)
        if d<0.1:
            c+=1
        n=c%2
        if n==0:
            color.rgb(0,0,0)
        else:
            color.rgb(255,255,255)
Fns… | a A # | Outils | Exéc | Script
```

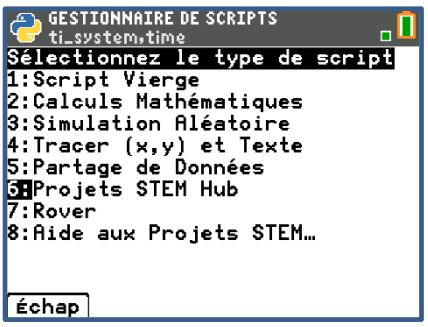
- `while not escape()`: crée une boucle 'infinie' qui sera interrompue par l'appui sur la touche **on** ; l'instruction se trouve dans le module *ti_system*.
- `d=m.measurement()` crée la variable **d** qui est la mesure donnée par **m** (le ranger branché à l'entrée 1).
- Une mesure est réalisée chaque demi-seconde.
- Si la distance mesurée par le ranger est inférieure à 0,1 m = 10 cm (en passant la main à proximité), le compteur **c** s'incrémente d'une unité.
- rappel : `c+=1` est un raccourci pour `c=c+1`
- Si **n** est pair, on éteint la led en saisissant la fonction **color.rgb(0,0,0)** ; la fonction **color.rgb** se trouve dans la bibliothèque *color* importée en préambule.
- Sinon, on saisit **color.rgb(255,255,255)**, ce qui a pour effet de mettre la led en lumière blanche à son maximum d'intensité.

Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus

Remarque

En préambule du code, on trouve l'importation de bibliothèques spécifiques au projet :

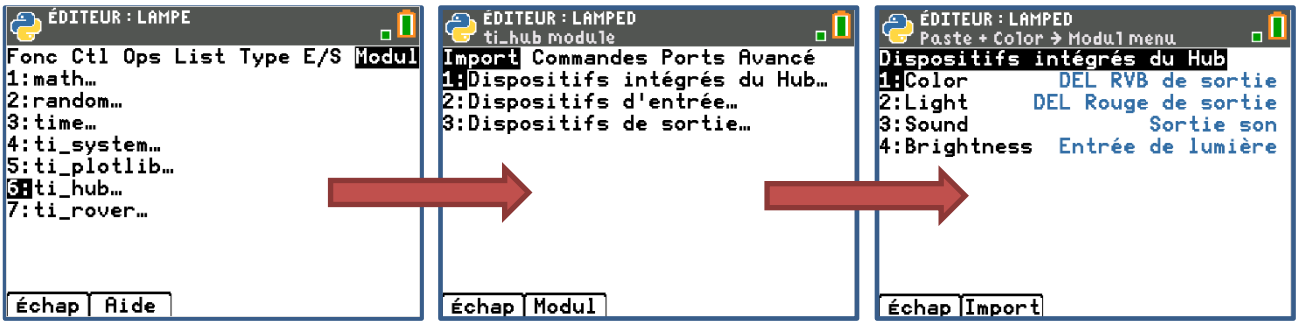
- elles peuvent être importées une à une en allant les chercher dans le menu **Modul** ;
- elles peuvent s'implémenter en choisissant le type de programme **« Projets STEM Hub »** (choix 6 des types de programmes proposés) à la création du nouveau programme.



La bibliothèque *ti_system* permet d'importer ou d'exporter des données dans les listes de la calculatrice.
Pour ce projet, elle contient l'instruction **while not escape( ) :** qui permet de lancer une boucle qui ne s'interrompt que lorsqu'on appuie sur la touche **on**.

La bibliothèque *time* permet en particulier d'utiliser la fonction **sleep** qui prend comme argument un temps de pause donné en secondes.

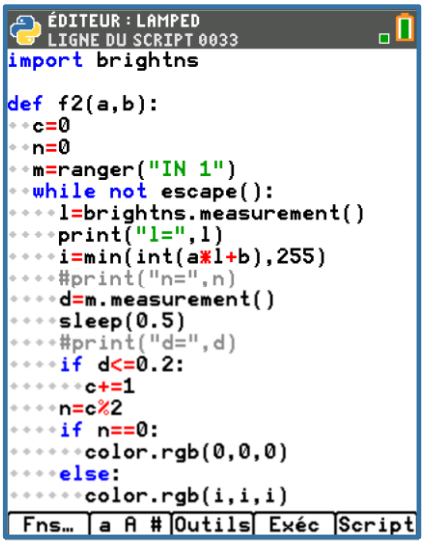
On va ensuite importer une bibliothèque spécifique au projet, à savoir la bibliothèque *color* en suivant les instructions suivantes :



Au fur et à mesure que les bibliothèques sont importées, elles s'ajoutent aux bibliothèques déjà présentes (*math*, *random*, …etc.) et permettent d'accéder aux instructions qu'elles contiennent.

Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus

Pour aller plus loin



On peut demander **une intensité lumineuse dépendant de la luminosité ambiante**. Pour cela, on mesurera la luminosité par le capteur de luminosité intégré au HUB et on transformera la valeur captée en un nombre entier compris entre 0 et 255.

C'est l'objet de la capture d'écran ci-contre qui montre la fonction **f2**.

Elle a pour paramètres **a** et **b** qui seront à choisir par l'utilisateur selon la luminosité ambiante et l'effet voulu :

- Très peu de lumière si l'ambiance est sombre ;
- Ou, au contraire, une luminosité d'autant plus grande que l'ambiance est sombre.

Une fonction affine est à déterminer selon le choix des élèves et la luminosité ambiante qu'il faudra mesurer pour avoir une idée des valeurs données qui varient d'un jour à l'autre.

Remarques

Quelques remarques sur cet approfondissement possible :

- Le module **brigthns** a été importé comme décrit précédemment comme un des dispositifs intégrés au HUB.

  Une fois intégré, il apparaît parmi les autres modules et permet de charger l'instruction **brighns.measurement**.

- On peut demander quelques affichages pour aider au bon réglage des paramètres.

  C'est par exemple ce que permet **print("l = ",l)** qui affichera la luminosité ambiante.

  Si cet affichage devient inutile, on peut le commenter en précédant l'instruction de # (qui s'obtient notamment par le raccourci "touche **2nde** suivie de la touche **3**") ; c'est ce qui a été fait dans le programme proposé.

Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus