Thème : informatique embarquée et objets connectés

TI-83 Premium CE
EDITION PYTHON

Classe enseignée : 2nde SNT

Radar pédagogique

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 signaler à un conducteur d'un véhicule qu'il dépasse la vitesse autorisée ?

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

Problématique

Comment reproduire le fonctionnement d'un radar pédagogique ?

1. Identifier un capteur possible ;
2. Identifier un actionneur ou des actionneurs possibles ;
3. Élaborer un algorithme modélisant la mesure d'une vitesse ;
4. Mettre cet algorithme en application.

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.

Thème : informatique embarquée et objets connectés

TI-83 Premium CE
EDITION PYTHON

Classe enseignée : 2nde SNT

Fiche méthode

F. GIROD

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 mesure la distance entre le véhicule et le capteur toutes les t secondes.
- 2ème étape : on utilise ces données pour déterminer la distance parcourue pendant ce laps de temps.
- 3ème étape : on calcule la vitesse moyenne du véhicule durant ce laps de temps.
- 4ème étape : on signale par un code couleur que la vitesse est acceptable ou non.

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.

Proposition de résolutions

Choix du capteur :

- Capteur de distance à ultrasons : on mesure la distance entre le capteur et le véhicule entre deux temps donnés, ce qui permet de calculer la vitesse moyenne sur cette durée.

Choix de / des actionneurs :

- LED RGB du HUB : elle pourra délivrer une lumière de la couleur de son choix : c'est cet actionneur qui est choisi ici ; rouge si la vitesse est trop importante, vert si la vitesse est conforme.
- Haut-parleur du HUB : il émet un son dont on définit la fréquence d'émission et la durée ; on choisit ici d'émettre un bip si la vitesse autorisée est dépassée.

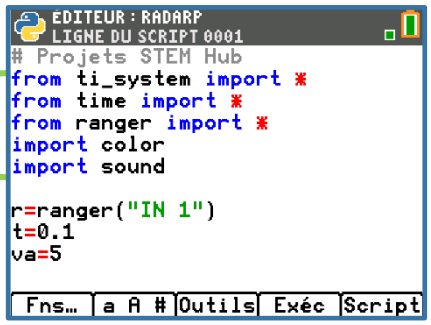
Pour profiter de tutoriels vidéos, Flasher le QRCode ou cliquer dessus !

Etapes de résolution

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

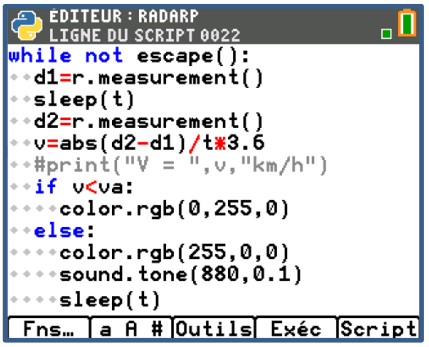
La variable **r** représente l'entrée du ranger (capteur de distance) relié au port d'entrée **IN 1** ; les instructions se trouvent dans le module *Ranger* importé précédemment.
On crée les variables **t** et **va** qui pourront être modifiées par l'utilisateur.

- **t** est la durée entre deux mesures de distance ;
- **va** est la limite de vitesse que l'on s'autorise.



- `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).
  - On effectue deux mesures : c'est pourquoi on a créé deux variables **d1** et **d2**.
  - Un intervalle de temps **t** sépare ces deux mesures. **sleep** se trouve dans la bibliothèque *time*.
- **v** est la vitesse moyenne sur la durée **t** :
  - On utilise la fonction **abs** (valeur absolue) pour pouvoir donner la vitesse aussi bien quand on se rapproche que quand on s'éloigne du capteur de distance.
  - la vitesse est en $km.h^{-1}$ (du fait de la multiplication par 3,6)
- Selon la valeur de **v**, on aura deux cas de figure :
  - vitesse autorisée : on allume une lumière verte par **color.rgb(0,255,0)** (la fonction **color.rgb** se trouve dans la bibliothèque *color* importée en préambule).
  - vitesse dépassée : on allume une lumière rouge par **color.rgb(255,0,0)** et on émet un bip aigu (fréquence de 880 Hz toutes les 0,1 seconde).

  A noter que la fonction **sound.tone** se situe dans la bibliothèque *sound* importée en préambule et que 0,1 est la durée minimale pour laquelle cette fonction est opérationnelle.



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

Thème : informatique embarquée et objets connectés

TI-83 Premium CE
EDITION PYTHON

Classe enseignée : 2nde SNT

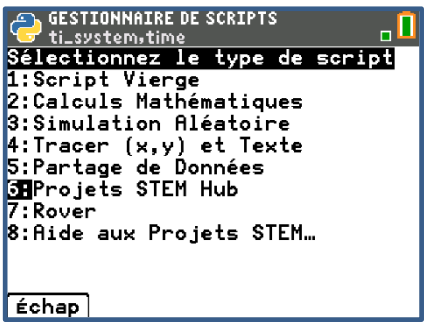
Fiche méthode

F. GIROD

Remarques

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

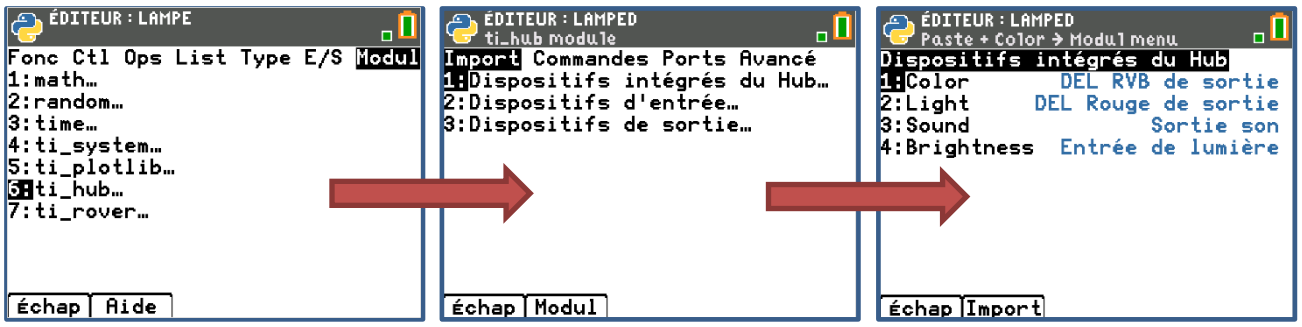
- Elles peuvent être importées une à une en allant les chercher dans le menu **Modul** ;
- Elles peuvent s'implémenter en choisissant le type de programme « **Projets STEM Hub** » (choix 6 des types de programmes proposés) à la création du nouveau programme.



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 si 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.

On procède de même pour importer la bibliothèque *Ranger* qui s'obtiendra en choisissant '**dispositifs d'entrée**' à l'étape 2 de la procédure précédente.

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