Programmer un jeu vidéo avec Pyxel : 1/6

Doc officielle de Pyxel : https://github.com/kitao/pyxel

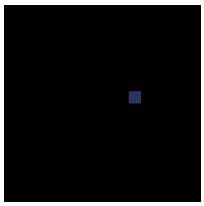
Prérequis : avoir fait en groupe le jeu du serpent, qui montre rapidement dans un exemple « tout fait » les fonctionnalités principales. Dans ce tutoriel, nous allons explorer un peu plus en détail les mécanismes.

1. Pour démarrer…

Rappels vus lors du jeu du serpent :

- Quelle première ligne doit-on obligatoirement écrire au début du script pour pouvoir utiliser le module pyxel ? ……**import pyxel**……
- Que faire si ce module n'est pas installé ? ……**pip install pyxel**……
- Quelles sont les deux fonctions prédéfinies appelée automatiquement par Pyxel dans une boucle infinie ? ……**draw et update**……
- Que fait l'instruction `pyxel.init(128, 128, title="Mon super jeu…")` ? ……**crée une fenêtre de taille 128*128 pixels**……
- Quelle instruction écrire à la fin du script pour lancer le jeu ? …… **pyxel.run(update, draw)** …….

2. Déplacer un carré avec les touches de direction



a. Créer la structure du code et le vaisseau

Dans cette partie, nous allons représenter un vaisseau spatial par un simple carré de couleur, qui pourra être déplacé à l'aide des flèches du clavier.

Deux variables globales (définies au niveau principal du code et non pas dans les fonctions) serviront à repérer la position : $vaisseau\_x$ et $vaisseau\_y$

- Où est l'origine du repère pour les coordonnées ? …**en haut à gauche**……..
- On définit la position initiale du vaisseau par les deux instructions ci-contre
  Où est le vaisseau au début du jeu ? ……**au centre de la fenêtre**……

```
vaisseau_x = 60
vaisseau_y = 60
```

- A quel endroit écrire cette instruction ? …**niveau global du programme (pas dans les fonctions)**
- Quelle instruction écrire pour dessiner un carré bleu de côté 8 pixels à cette position ?
  ……**pyxel.rect(vaisseau_x, vaisseau_y, 8,8,1)**……
- A quel endroit écrire cette instruction ? ……**dans la fonction draw()**…….
- Il est préférable d'effacer l'écran et de le remplir de noir avant de dessiner le vaisseau. Quelle instruction écrire pour cela ? ……**pyxel.cls(0)**……..

Jalon 1 : le script doit être structuré avec : les variables globales, les deux fonctions fondamentales du jeu (l'une ne contient rien, juste l'instruction $pass$), l'instruction de lancement, et quand on lance le script, on doit voir le vaisseau.

b. Gérer les déplacements du vaisseau

Dans cette partie, nous allons compléter la fonction update() qui est appelée 30 fois par seconde par Pyxel au sein de la boucle infinie du jeu

```
def update():
    """mise à jour des variables (30 fois par seconde)"""

    global vaisseau_x, vaisseau_y

    # mise à jour de la position du vaisseau
    vaisseau_x, vaisseau_y = vaisseau_deplacement(vaisseau_x, vaisseau_y)
```

- A quoi sert la ligne $\boldsymbol{global\ vaisseau\_x, vaisseau\_y}$? **il faut que la fonction update() ait le droit de modifier ces variables : on les déclare donc en global**…….
- Les coordonnées du vaisseau sont mises à jour en une seule ligne, qui affecte simultanément les deux variables, comme lorsqu'on écrit $(\boldsymbol{a}, \boldsymbol{b}) = (\mathbf{2}, \mathbf{0})$
  On appelle cela **l'affectation par tuple**
  Un tuple est un objet contenant plusieurs valeurs, souvent entre parenthèses, mais ce n'est pas obligatoire, on peut aussi écrire $\boldsymbol{a}, \boldsymbol{b} = \mathbf{2}, \mathbf{0}$
  On leur affecte visiblement ce qui est renvoyé par la fonction $\boldsymbol{vaisseau\_deplacement}$.
  Quel est forcément le type de l'objet renvoyé par cette fonction ? …**un tuple à 2 éléments**..

- On s'intéresse maintenant à la fonction $\boldsymbol{vaisseau\_deplacement}$ dont voici un canevas

```
def vaisseau_deplacement(x, y):
    """déplacement avec les touches de directions"""

    if pyxel.btn(pyxel.KEY_RIGHT):
        if (x < 120) :
            x = x + 1
    elif pyxel.btn(pyxel.KEY_LEFT):
        if (x > 0) :
            x = x - 1
    elif pyxel.btn(pyxel.KEY_DOWN):
        if (y < 120) :
            y = y + 1
    elif pyxel.btn(pyxel.KEY_UP):
        if (y > 0) :
            y = y - 1
    return x, y
```

- Quel est le type des ***paramètres formels*** de cette fonction *(ceux qui figurent dans la parenthèse)* ? ……**des entiers**…….
- Lors de son utilisation, quels ***paramètres effectifs*** lui seront passés ? *(ceux avec lesquels on fait réellement « fonctionner » la fonction lors de l'exécution du script*) ……**vaisseau_x et vaisseau_y**……
- Cette fonction gère les interactions avec le joueur : quel type d'interactions ? **appui sur flèches du clavier**…….
- A quoi sert la ligne de test $\boldsymbol{if\ x < 120}$: **vérifier que le vaisseau n'est pas sorti de la fenêtre sur la droite**……
- Que doit renvoyer la fonction en sortie pour qu'on puisse l'utiliser comme prévu dans la fonction $\boldsymbol{update}()$? …**le tuple (x,y) des nouvelles coordonnées**…..
- Compléter les autres cas figurant dans le script, et intégrer ce code dans votre script de jeu.

Jalon 2 : quand on lance le script, on doit cette fois pouvoir déplacer le vaisseau avec les flèches du clavier, dans les 4 directions.

c. Perfectionnement

- Remplacer les $\boldsymbol{elif}$ par des simples $\boldsymbol{if}$: quelle est la différence ? Expliquer …… ……**on peut appuyer sur 2 touches à la fois**……..
- Rajouter une instruction qui remet le vaisseau au centre si on appuie sur la touche Espace **if pyxel.btn(pyxel.KEY_SPACE): x,y = (60,60)**
- Que se passe-t-il si on appuie à un moment quelconque sur la touche Echap ?..**quitte le jeu**.