Programmer un jeu vidéo avec Pyxel : 6/6

Ajouter des images

Nous repartirons dans ce tutoriel du script réalisé précédemment : $tuto\_pyxel\_5.py$
Faire Enregistrer Sous pour renommer à présent ce script en $tuto\_pyxel\_6.py$

1. Pour démarrer…

Rappels vus lors du tutoriel précédent :

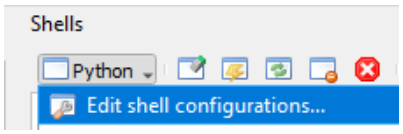
- Quelle instruction pyxel dessine un cercle ? ……
  pyxel.circb(x_centre, y_centre, rayon, n° couleur)……
- Quel mécanisme permet d'avoir un « cycle » ? …l'utilisation du modulo…… Donner un exemple : …couleur = 8 + explosion[2]%3……
- Comment étaient animées les explosions ? ……le diamètre des cercles augmentait de 1 à chaque update jusqu'à 12……

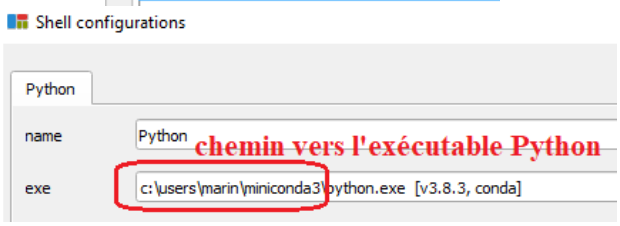
On voudrait maintenant ajouter des images pour matérialiser le vaisseau, les ennemis… Ce sera plus réaliste que des carrés !

2. Ouvrir l'éditeur Pyxel pour les images

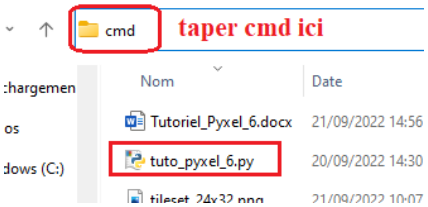
Nous allons voir qu'il est possible d'utiliser un utilitaire Pyxel pour créer ou éditer des images, ou plus généralement des ressources (sons…)
On lance cet éditeur depuis une fenêtre de commandes.
Pour cela :

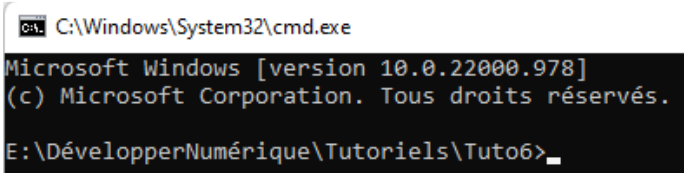
- Dans Pyzo, **ouvrir la configuration du Shell** à droite =>
- **Copier le chemin d'accès vers l'exécutable Python**. Ici, c'est c:\users\miniconda3\
- Ouvrir un explorateur de fichiers, se rendre dans le dossier où est votre script $tuto\_pyxel\_6.py$, puis **lancer une fenêtre de commandes en tapant *cmd*** dans la barre de navigation.





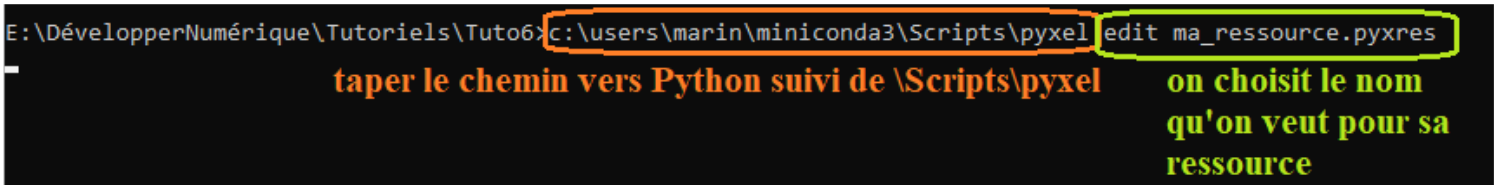
Une invite de commandes s'ouvre alors :



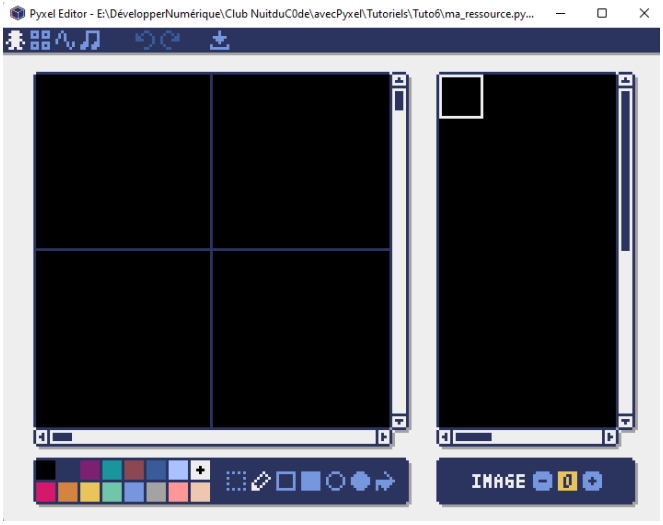


- Taper la commande $pyxel\ edit\ ma\_ressource.pyxres$

<u>Important : il faut faire précéder le mot pyxel du chemin où il se trouve : c'est le même que le chemin vers l'exécutable Python, suivi de \Scripts\</u>



L'interface de l'utilitaire Pyxel s'ouvre …



Cet éditeur permet de dessiner-soi même des images façon pixel art, c'est-à-dire très simples, donc très légères.

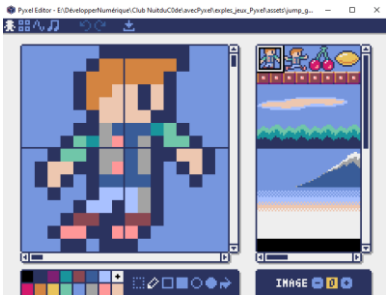
On dessine dans la partie gauche, qui « zoome » sur un carré de 16*16 pixels découpé en 4 parties.

Ce carré est en fait une petite portion de la **tilemap** (à droite), sorte d'immense nappe sur laquelle on va juxtaposer tous les dessins, et qu'on « découpera » en petits morceaux, pour venir coller les morceaux sur la fenêtre de jeu au bon endroit.

Jalon 1 : avoir réussi à ouvrir l'éditeur Pyxel !

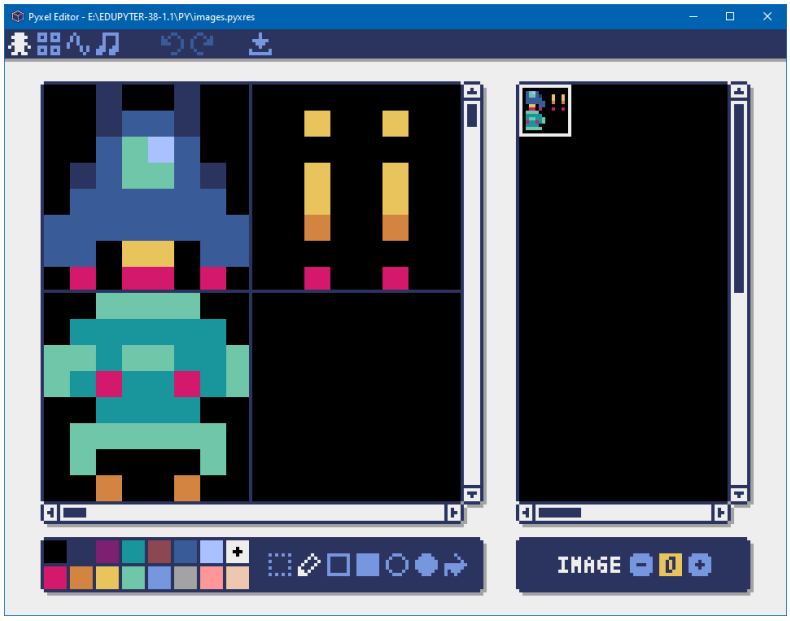
3. Dessiner avec l'éditeur Pyxel

Une tilemap (littéralement une "carte de tuile") est une grille utilisée pour créer la disposition/le fond graphique d'un jeu



On peut au choix :

- Dessiner soi-même
- Utiliser une tilemap déjà faite, au format ***.pyxres***



Dans cette partie, vous allez dessiner vous-même et reproduire le schéma suivant, qui servira pour le vaisseau (bleu foncé), les ennemis et les missiles

Vous n'oublierez pas d'enregistrer à la fin votre dessin, qui porte, rappelons-le, le nom **ma_ressource.pyxres**

Jalon 2 : avoir terminé le dessin ci-contre, et vérifié que le fichier **ma_ressource.pyxres** se trouve bien dans le même répertoire que votre code Python

4. Utiliser la tilemap pour insérer des images dans le jeu

Maintenant, nous avons une magnifique tilemap avec plusieurs images, mais nous ne les avons pas encore « blittées » (« collées ») sur l'écran !

Où va-t-on écrire les instructions pour ce faire ? ……dans draw()……

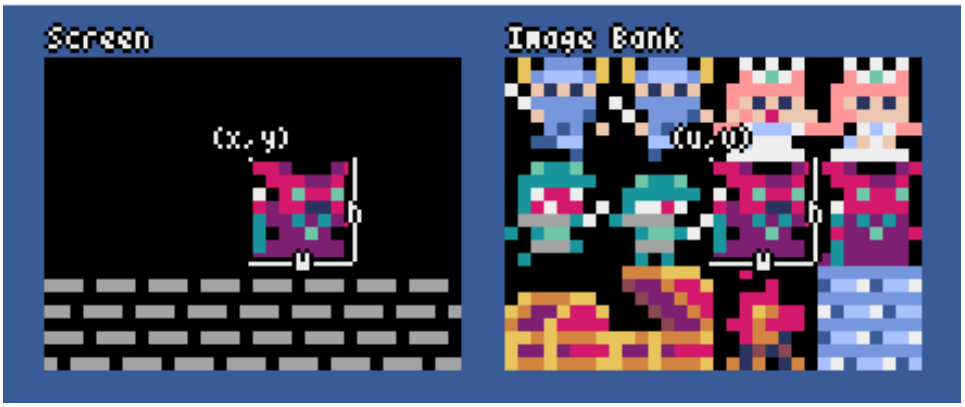
Il faudra au préalable que le programme charge la tilemap dans la mémoire de l'ordinateur, en écrivant au début du script, après le $pyxel.init()$, l'instruction

$$\boldsymbol{pyxel.load("ma\_ressource.pyxres")}$$

NB : pour que cela fonctionne, il faut que le fichier ressource soit bien dans le répertoire courant de Python ! Normalement, si vous l'avez mis au même endroit que votre script, c'est le cas. Sinon, il faut préciser le chemin, par exemple `pyxel.load("c:\users\toto\ma_ressource.pyxres")`

Une fois que votre banque d'images est chargée, vous pouvez afficher n'importe quelle image à l'aide de la commande blt(). Voici comment la commande s'utilise :

$$\boldsymbol{pyxel.blt(x, y, img, u, v, w, h)}$$



Cettecommande copie :

- à la **position** (x, y) de l'écran de jeu
- l'image de la tilemap qui est située à la **position** (u, v) et qui a pour **taille** (w, h)**.**

Elle « découpe » donc un morceau de la tilemap (morceau décrit par u, v, w, h) et vient le « coller » au point (x,y) sur l'écran

Le paramètre img correspond à l'une des trois grandes images de la banque d'images. Par défaut sa valeur est zéro (cela correspond à la tilemap qu'on voit en ouvrant l'éditeur)

Exemple : Ainsi, pour afficher en (x, y) une image de **16*16 pixels en haut à gauche** de la tilemap, on utilisera la commande …… pyxel.blt(x, y, 0, 0, 0, 16, 16)……..

Application à notre jeu

Dans le script, on va remplacer les dessins de rectangles par des blittages d'images.

- Dessin du vaisseau : on rappelle que les coordonnées du vaisseau sont vaisseau_x et vaisseau_y. Quelle instruction écrire pour blitter l'image du vaisseau au bon endroit ?
  …… pyxel.blt(vaisseau_x, vaisseau_y, 0, 0, 0, 8, 8)……

- Ecrire une boucle qui affiche de même chacun des tirs de la liste $tirs\_liste$ à ses coordonnées
  Dans la boucle for tir in tirs_liste : pyxel.blt(tir[0], tir[1], 0, 8, 0, 8, 8)

- Faire de même pour les ennemis : dans boucle : pyxel.blt(ennemi[0], ennemi[1], 0, 0, 8, 8, 8)

Jalon final : les images sont dessinées sur l'écran lors du jeu