

Le logiciel pour démarrer la programmation et la création de jeux vidéo

Il y a quelques années, m'est venue l'envie de **créer des jeux vidéo**. Le problème c'est que je n'avais jamais réellement programmé et je ne savais pas dessiner. Étant très motivé, j'ai vaguement essayé Unity mais le logiciel était déjà très complet et je me sentais un peu perdu, j'avais beau suivre les séries de tutoriels, impossible d'aller plus loin qu'un cube blanc qui bouge de gauche à droite ! Ce que je veux vous présenter ici, c'est **le logiciel** que j'aurais voulu avoir lorsque je voulais me lancer dans le jeu vidéo. Un logiciel **simple**, avec **tous les outils fournis** (pas besoin d'installer d'autres logiciels pour faire la map ou la musique par exemple).

Cet outil, c'est **PICO-8**. Pour avoir une idée des jeux réalisables, vous pouvez jouer gratuitement aux cartouches proposées par des membres de la communauté [ici](#).



Les points qui seront abordés dans ce document :

1. Comment utiliser le shell PICO-8 ?
2. Les bases de la programmation en Lua
3. Comment afficher un pixel à l'écran ?
4. Comment déplacer un personnage à l'écran ?
5. Comment faire un jeu sur PICO-8 ?

Présentation générale

Si je devais traduire la description de PICO-8 disponible sur le [site officiel](#), ça donnerait ça :

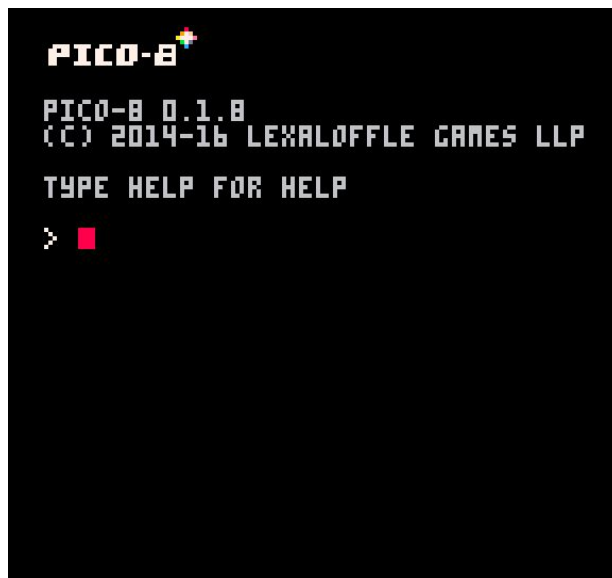
PICO-8 est une console “imaginaire” qui permet de créer, partager et jouer à de petits jeux/logiciels. Quand vous la démarrez, la machine vous accueille avec un shell afin de taper des commandes Lua et fournit des outils pour créer ses propres cartouches.

La console vous met à disposition un éditeur de **scripts**, de **sprites**, de **maps**, d'**effets sonores** et de **musiques**, de quoi faire **un jeu complet sans quitter l'éditeur** ! Vous pouvez voir ci-dessous 4 screenshots tirés d'un jeu réalisé sur PICO-8.



Comment utiliser le shell PICO-8 ?

Ce document s'adressant à des débutants, je vais expliquer rapidement ce qu'est un shell. Vous avez déjà dû voir ce "truc" sur linux, un grand rectangle noir dans lequel on peut écrire des lignes compliquées : c'est un shell. Je dois avouer que ça fait un peu peur au début, sachez que pour PICO-8, il y a **grand maximum 5 commandes à retenir**. A la fin de cette page vous saurez tout. Démarrons PICO-8 !



```
PICO-8
PICO-8 0.1.0
(C) 2014-16 LEXALOFFLE GAMES LLP
TYPE HELP FOR HELP
>
```

Lorsque vous démarrez PICO-8, vous pouvez utiliser "LS" pour **LiSter les dossiers** et commencer en **créant un dossier** nommé "test" en utilisant :

MKDIR TEST

PICO-8 ne fait pas la différence entre majuscule et minuscule mais prenez l'habitude d'écrire en minuscule. En effet, certaines lettres majuscules permettent d'écrire des caractères spéciaux donc retenez-bien, **que des minuscules !** :)



MKDIR signifie "MaKe DIRectory" donc littéralement "créer un dossier". Ensuite, on va pouvoir Changer de Dossier (CD), on va donc utiliser :

CD TEST

Voilà vous avez fait la moitié des étapes pour commencer à créer votre jeu !

Pour créer votre cartouche, vous allez utiliser "SAVE PONG" pour **créer un fichier**. PICO-8 va directement lui mettre l'extension .P8 donc on va se retrouver avec un fichier qui s'appelle PONG.P8. Pour l'ouvrir on va faire "LOAD PONG.P8" et voilà, **votre cartouche est dans la console** ! Pour accéder à l'éditeur, appuyez sur la touche échap. Vous êtes fin prêt à démarrer votre jeu.

Petit récapitulatif des commandes que l'on a effectuées :

```
PICO-8   
PICO-8 0.1.8  
(C) 2014-16 LEXALOFFLE GAMES LLP  
TYPE HELP FOR HELP  
  
> MKDIR TEST  
> CD TEST  
/TEST/  
> LS  
DIRECTORY: /TEST/  
> SAVE PONG  
SAVED PONG.P8  
> LOAD PONG.P8  
LOADED PONG.P8 (0 CHARS)  
> 
```

Une fois dans l'éditeur, faites Ctrl+S pour sauvegarder, Echap pour retourner dans le shell et utilisez la commande "RUN" pour lancer votre jeu. (malheureusement, rien ne se passera pour l'instant mais d'ici 5 minutes ça devrait changer ! :)).

Quelques petites astuces :

- Plein écran : <ALT> + <ENTER>
- Quitter PICO-8 : SHUTDOWN
- Redémarrer PICO-8 : REBOOT

Les bases de la programmation en Lua

Vous devez tout d'abord savoir une chose, si vous n'avez **jamais codé** et si vous voulez avoir de **très bonnes bases** pour commencer PICO-8, je ne pourrai que vous conseiller de d'abord faire le **tutoriel Javascript sur Codecademy** (en deux soirées c'est plié, voir une si vous êtes très bon ;)). Vous allez comprendre ce que sont les **variables**, les **conditions**, les **boucles** et les **fonctions**. Vous pouvez quand même suivre le document sans avoir fait ce tutoriel car les notions ne sont pas très compliquées. Sachez que la syntaxe Lua est encore plus simple que la syntaxe Javascript !

Si vous avez **déjà quelques bases en programmation** ou que vous avez fini le tutoriel Javascript (félicitations !), vous devez maintenant comprendre comment s'écrit le Lua.

→ Pour définir une variable :

- ◆ HP = 100
- ◆ NAME = "Bobby"
- ◆ ALIVE = TRUE
- ◆ INVENTORY = {"sword", "shield"}

On a donc respectivement utilisé un nombre, une chaîne de caractères (entourée de guillemets), un booléen (TRUE ou FALSE) et un tableau. Sachez que pour accéder à la première valeur d'une table il faut utiliser INVENTORY[1] et non INVENTORY[0], c'est une spécificité du Lua car dans la plupart des langages on va utiliser l'index 0 pour la première valeur.

→ Les conditions

A screenshot of a PICO-8 editor window. The editor has a black background with a red border. The code is written in a monospaced font with syntax highlighting: keywords like IF, THEN, ELSE, END, and PRINT are in red; strings are in green; and numbers and variables are in blue. The code defines a health variable HP and checks its value. There is a comment in French. The status bar at the bottom shows 'LINE 22/33' and '36/8192' in red.

```
HP = 50
IF (HP > 0) THEN
  PRINT("ALIVE")
END
--[[ VA AFFICHER ALIVE SI LA
VARIABLE HP EST SUPERIEURE
A ZERO ]]--

HP = 0
IF (HP > 0) THEN
  PRINT("ALIVE")
ELSE
  PRINT("DEAD")
END

IF (HP <= 0) PRINT ("DEAD")
```

Si votre IF est sur plusieurs lignes, n'oubliez pas "THEN" en fin de ligne et "END" à la fin.

→ Les boucles

```
FOR X=1,5 DO  
  PRINT(X)  
END
```

va afficher les chiffres de 1 à 5

→ Les fonctions

```
FUNCTION SAY_HELLO(NAME)  
  PRINT("HELLO "..NAME.." !")  
END  
SAY_HELLO("BLOBBY")
```

ici on appelle la fonction SAY_HELLO qui prend comme paramètre un nom et qui affiche "Hello blobby !" dans le cas où "blobby" est donné en paramètre. Vous remarquerez que pour insérer une variable au sein d'une chaîne de caractères, on utilise deux points consécutifs(..).

Voilà vous avez **les bases** pour faire du Lua ! Si vous n'êtes pas encore très à l'aise, vous pouvez continuer à manipuler les conditions, les boucles et les fonctions en affichant dans le shell avec PRINT(). On va maintenant utiliser les fonctions qui sont mises à disposition par PICO-8 pour afficher des pixels à l'écran ! Ci-dessous un autre petit exemple de programme avec des fonctions et des conditions.



The image shows a screenshot of a PICO-8 editor window. The editor has a dark background with a red header bar containing icons for file operations. The code is written in a monospaced font with syntax highlighting. The script defines a variable, two functions, and then calls them. Below the editor, a separate window shows the output of the script when run.

```
PERSO_HP = 100  
  
FUNCTION SAY_HELLO(NAME)  
  PRINT("HELLO "..NAME.." !")  
END  
  
FUNCTION ETAT()  
  IF (PERSO_HP < 100) THEN  
    PRINT("BLOBBY A MAL.")  
  END  
END  
  
SAY_HELLO("BLOBBY")  
ETAT()  
PERSO_HP = PERSO_HP - 10  
-- IDENTIQUE A PERSO_HP -= 10  
ETAT()  
  
LINE 21/21 38/8192
```

```
> RUN  
HELLO BLOBBY !  
BLOBBY A MAL.  
>
```


Comment afficher un pixel à l'écran ?

Nous allons tout d'abord voir **une des deux fonctions essentielles** à PICO-8 : `_DRAW()`. Cette fonction est appelée 30 fois par seconde et **permet d'afficher des choses à l'écran**. Prenons un exemple :

```
FUNCTION _DRAW()  
  CLS()  
  PSET(63,63,8)  
END
```

CLS, "CLear Screen", va permettre de **vider l'écran** (vous pouvez essayer de l'enlever, lorsque vous lancerez votre jeu, la commande RUN et tout ce qui la précède va rester affiché).

PSET, "Pixel SET", est une des fonctions fournies par PICO-8. Elle permet d'**afficher un pixel à l'écran** et prend 3 paramètres : X, Y, COLOR. Une première chose à comprendre c'est que **votre écran est une grille**, comme un tableur Excel par exemple. Vous avez les X qui vont de 0 à 127 de **gauche à droite** et les Y qui vont aussi de 0 à 127 mais de **haut en bas**. On a donc ici demandé d'afficher un pixel aux coordonnées (63,63) soit **presque le centre de l'écran**. La valeur 8 va correspondre à la couleur. Si vous passez dans le mode "Editeur de sprite" en utilisant le 2ème onglet en haut de PICO-8, vous pourrez voir la palette ci-contre. Les cases sont **numérotées de 0 à 15** donc 8 va correspondre au rouge. On a donc utilisé `PSET(63,63,8)`.

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

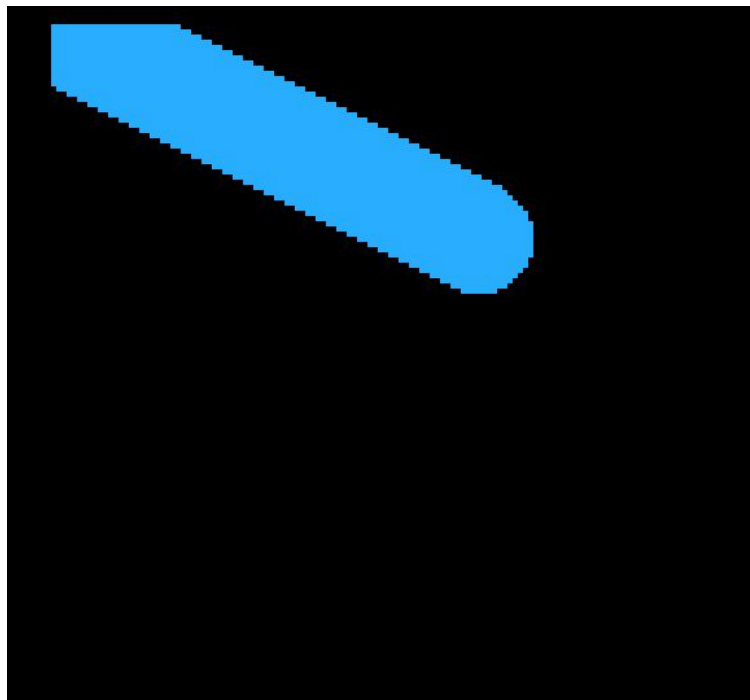
Lorsqu'on lance notre jeu en retournant sur le shell et en exécutant RUN, on voit un **écran noir** avec notre **pixel rouge** au centre.

Si vous voulez un rectangle, vous pouvez utiliser `RECT(x0,y0,x1,y1,color)` ou `RECTFILL` si vous voulez un rectangle plein.

Vous pouvez faire beaucoup de choses avec cette base. N'hésitez pas à lire le [manuel](#) dans la section Graphics (un Ctrl+F en recherchant Graphics et vous allez tomber dessus) qui vous donne **toutes les fonctions** que vous pouvez utiliser pour **afficher des choses à l'écran**. Ci-dessous vous trouverez un exemple de sphère qui bouge jusqu'à l'extrémité droite de l'écran et qui repart avec un X égal à 0.

```
X=0
Y=0
FUNCTION _DRAW()
  CIRCfill(X,Y,10,12)
  X += 1
  Y += 0.5
  IF (X >= 128) X=0
END
```

LINE 26/26 29/0192



Nous allons maintenant voir comment **déplacer un pixel** avec les **touches du clavier**.

Comment déplacer un pixel à l'écran ?

Dans le chapitre précédent, nous avons vu comment utiliser `_DRAW()`. Nous allons ici nous focaliser sur la boucle de jeu nous permettant de **savoir si le joueur a appuyé sur une touche**. Je vous présente donc la **seconde fonction essentielle** : `_UPDATE()`. Cette fonction est exécutée exactement 30 fois par seconde (là où `_DRAW()` est appelée à chaque frame visible donc parfois moins de 30 fois par seconde si le jeu a un ralentissement). La stabilité de cette fonction permet une **fluidité totale dans les contrôles**. On va tout de suite prendre un exemple et on va commencer par créer un sprite ! Allons dans le deuxième onglet et dessinons un petit personnage dans la case 001 (numéro visible au centre de l'écran). On va ensuite utiliser ce script :

```
X=63
Y=63
FUNCTION _UPDATE()
  IF (BTN(0)) X -= 1 -- 8
  IF (BTN(1)) X += 1 -- 8
END

FUNCTION _DRAW()
  CLS()
  SPR(1,X,Y)
END

LINE 24/26 35/8192
```



Ici on va donc dans la fonction `_DRAW()` afficher le **sprite 1** en (X,Y). Dans la fonction `_UPDATE()` on va vérifier avec des IF si les boutons sont actuellement enfoncés. `BTN(0)` va vérifier la flèche gauche, `BTN(1)` va vérifier la flèche droite etc...

Pour les input, vous devez retenir :

- 0 → Gauche
- 1 → Droite
- 2 → Haut
- 3 → Bas
- 4 → X
- 5 → C

Je vous laisse faire en sorte que votre personnage se déplace **de haut en bas**, si vous avez bien tout suivi, ça ne devrait pas vous poser de problème ! :)

Comment faire un jeu sur PICO-8 ?

Suite à la lecture de ce document, **vous en savez déjà beaucoup sur PICO-8** ! La chose la **plus importante** pour continuer, c'est de d'abord lire une bonne partie du manuel disponible [ici](#). Par exemple, si vous voulez parcourir un tableau vous verrez que vous pouvez utiliser la forme:

```
STORE = {"SWORD", "BOU", "AXE"}  
FOR WEAPON IN ALL(STORE) DO  
  PRINT(WEAPON)  
END
```

Ce sont des choses que vous aurez besoin par la suite donc **lisez ce manuel**. De plus vous pouvez vous inspirer des démos disponibles en utilisant INSTALL_DEMOS et en allant dans le dossier DEMOS créé à la racine de PICO-8.

J'espère que vous avez encore plus envie de **créer votre premier jeu** !

Et vu que vous êtes arrivés jusqu'ici, je vais vous donner **un conseil bonus** pour pouvoir vous lancer : **participez à des game jam** ! Ce sont des week-ends où des dizaines, centaines (voir des milliers pour certaines Jam) de personnes créent un jeu sur un thème donné. Les plus connues sont la Global Game Jam et la Ludum Dare. C'est vraiment **le coup de boost** qu'il me fallait pour enfin me lancer dans la création de jeu. Vous avez deux jours pour faire du mieux que vous pouvez, les gens savent que le résultat ne sera pas parfait car en deux jours c'est très difficile de faire un gros jeu ! Vous pouvez vous rendre [ici](#) ou [ici](#) pour jouer à mes deux premiers jeux de game jam en solo. Comme vous pouvez le voir c'est loin d'être parfait, que ce soit au niveau du gameplay ou au niveau des graphismes. Croyez en vous et lancez-vous !

Si vous avez des **questions** ou un **feedback** concernant ce document, n'hésitez pas à me contacter à l'adresse contact@creationdejeuxvideo.com.

Le blog : www.creationdejeuxvideo.com