Pong v.1

DIVERSITY by EPITECH

I – Introduction

Le Lua a été créé en 1993 au Brésil afin de pouvoir être embarqué au sein d'autres applications par trois étudiants d'une université de Rio de Janeiro.

Il est utilisé pour faire de l'embarqué mais surtout du réseau et du jeu vidéo.
Il a notamment été utilisé dans Roblox, World of Warcraft, Garrysmod et bien d'autres.
Il est également utilisé dans des moteurs de jeu vidéo tels que CryENGINE.

Pour ce qui est de Pico 8 c'est une machine virtuelle de type « Fantasy Console » qui donne la possibilité de développer, partager et jouer à des petits jeux vidéo.
L'affichage est limité à 16 couleurs et donne donc facilement aux jeux créés un aspect retro.

PICO-8

II – Consignes

1. **Le nom du repo est : pong_cc**

2. **En cas de questions, n'hésitez pas à vous entraider et si vous êtes toujours bloqué, un cobra se fera une joie de vous aider.**

3. **Vous allez utiliser le langage de programmation « Lua » sur la machine virtuelle « Pico 8 ».**

4. **Si l'installation de pico-8 ne se déroule pas bien, recommencez et demandez de l'aide à un cobra.**

III – On se fait un pong ?

Vous êtes dans les années 70 à l'aube d'une nouvelle ère, le monde de l'informatique commence à voir le jour.

Des ordinateurs sont reliés à Aparnet, l'ancêtre d'internet et des langages tels que le C apparaissent.

Vous voulez vous former au métier de développeur de jeux vidéo et un ami, Nolan Bushnell vous contacte afin de réaliser un projet top secret pour vous initier. Il vous demande de réaliser " le jeu le plus simple", un ping pong en vue aérienne.

IV – Prise en main de Pico 8

Une fois PICO-8 lancé, tu commences en Command Mode.

À partir de cet écran, tu peux appuyer sur la touche "ESC" de ton clavier pour rentrer en Editor Mode, ou tu pourras créer tes jeux.

Quand tu voudras lancer ton jeu, il te suffira d'appuyer sur "ESC" pour revenir en Command Mode, puis d'écrire RUN.

Pour revenir à nouveau en Command Mode, il suffit d'appuyer sur ESC.

Tu peux aussi sauvegarder ton jeu en écrivant SAVE GAME-NAME.
Pour charger ton jeu à ton retour, tu peux écrire LOAD GAME-NAME.

Si tu oublies le nom de ton jeu, tu peux écrire DIR et PICO-8 te donnera une liste de tous les jeux qu'il connait.

Pour ce jeu, tu pourras le sauvegarder en écrivant SAVE PONG.

Maintenant que tu connais les bases, commençons à créer notre jeu !

V – Une raquette mobile

Faisons bouger une raquette !

Appuie sur ESC pour rentrer en Editor Mode et copie le morceau de code en dessous.
C'est utile d'appuyer sur ESPACE au début des lignes à l'intérieur des blocks IF et FUNCTION, car cela rend le code beaucoup plus lisible. Les commentaires sont également utiles. Ce sont des lignes commençant par un "-", ils permettent d'ajouter des notes dans le code afin de pouvoir laisser des informations. L'ordinateur ignore ces lignes.

```
padx = 53
pady = 122
padw = 24
padh = 4


function movepaddle()
  if btn(0) then
    padx -= 3
  elseif btn(1) then
    padx += 3
  end
end
```

```
function _update()
  movepaddle()
end

function _draw()
  rectfill(0, 0, 128, 128, 3)
  rectfill(padx, pady, padx+padw, pady+padh, 15)
end
```

Appuie sur la touche "ESC", puis tape RUN. En appuyant sur la flèche droite ou gauche, la raquette sur l'écran devrait bouger.

Regarde la fonction « *movepaddle* ». Cela nous permettra de

comprendre plus facilement quel code fait quoi plus tard.

– Les commandes utilisées

Fonction	Utilisation
update()	appelée 30 fois par secondes, c'est ici qu'on peut mettre à jour le jeu.
draw()	Elle est appelée après update(). C'est ici qu'on dessine sur l'écran.
btn(b)	Vérifie si l'utilisateur a appuyé sur un bouton. « b » peut avoir ces valeurs. - 0 = gauche - 1 = droite - 2 = haut - 3 = bas - 4 = Z - 5 = X

VI – Rajoutes une balle

Appuie deux fois sur ESC pour revenir en Editor Mode. Rajoutes des variables en haut du fichier pour savoir où positionner la balle :

```
ballx = 64
bally = 64
ballsize = 3
ballxdir = 5
ballydir = 3
padx = 53
pady = 122
padw = 24
padh = 4
```

Rajoutes ensuite cette fonction DRAW au fond du fichier :

```
function _draw()
  -- nettoye l'écran
  rectfill(0, 0, 128, 128, 3)
  -- Dessine la raquette
  rectfill(padx, pady, padx+padw, pady+padh, 15)
  -- Dessine la balle
  circfill(ballx, bally, ballsize, 15)
end
```

– Les commandes utilisées

Fonction	Utilisation
circfill(x, y, size, col)	Dessine un cercle centré sur x et y.

VII – Une balle à l'arrêt est une mauvaise balle

Appuie sur ESC jusqu'à revenir en Editor Mode, puis rajoute une nouvelle fonction avant la fonction UPDATE.
Puis assure-toi de l'appeler dans la fonction UPDATE comme ceci :

```
function moveball()
  ballx += ballxdir
  bally += ballydir
end

function _update()
  movepaddle()
  moveball()
end
```

Si tu RUN ton programme, tu devrais avoir une balle qui s'envole en haut à droite de l'écran.

VIII – Garde-le sur le terrain

La balle doit rebondir sur les côtés de l'écran. Il suffit de vérifier les coordonnées de la balle.

Rappelle-toi que le coin en haut à gauche à comme coordonnées (0, 0) et que le coin en bas à droite est situé en (128, 128).

Pour faire rebondir la balle sur un côté, il suffit d'inverser le signe de la direction de la balle.

Si la vitesse est supérieure à 0, la balle bouge vers la droite, si elle est inférieure à 0, la balle bouge vers la gauche.

```
function bounceball()
  --gauche
  if ballx < ballsize then
    ballxdir =- ballxdir
    sfx(0)
  end
  --droite
  if ballx > 128 - ballsize then
    ballxdir =- ballxdir
    sfx(0)
  end
  --haut
  if bally < ballsize then
    ballydir =- ballydir
    sfx(0)
  end
end
```

```
function bounceball()
  --gauche
  if ballx < ballsize then
    ballxdir =- ballxdir
    sfx(0)
  end
  --droite
  if ballx > 128 - ballsize then
    ballxdir =- ballxdir
    sfx(0)
  end
  --haut
  if bally < ballsize then
    ballydir =- ballydir
    sfx(0)
  end
end
```

Et appelle-la depuis la méthode update() :

RUN ce que tu as, et tu devrais avoir une balle qui rebondit sur l'écran jusqu'à tomber en dehors du bas de l'écran.

- Les commandes utilisées

Fonction	Utilisation
sfx(number)	Joue un son

Pour créer un son, clique sur l'icône comme indiqué ci-dessous.

C’est ici que tu vas créer le son que tu vas entendre !
Le numéro entre parenthèses correspond au 00 dans l’image ci-dessous.

IX – Frappe cette balle

On doit vérifier si l'abscisse de la balle est comprise dans la longueur de la raquette et si la balle est rentrée dans la raquette.

On fait ça en utilisant le mot AND dans PICO-8, comme si on disait "et" en français. Rajoute une fonction « BOUNCEPADDLE » après la fonction « BOUNCEBALL » :

```
function bouncepaddle()
  if ballx >= padx and ballx <= padx + padw and bally > pady then
    sfx(0)
    ballydir =- ballydir
  end
end
```

Si tu veux, tu peux ajouter un nouveau son à jouer quand la balle frappe la raquette.

N'oublie pas d'appeler cette fonction dans update.

Si tu "RUN" ce que tu as actuellement, tu devrais pouvoir faire bouger la balle en la frappant avec la raquette (même si elle va disparaitre une fois passée le bas de l'écran).

X – Peut-on ravoir notre balle, s'il vous plait ?

Quand la balle disparaît de l'écran, on doit la remettre au centre de l'écran (on doit aussi perdre une vie, mais on s'en occupera plus tard)

Rajoute une nouvelle fonction après MOVEBALL :

```
function loseball()
  if bally > 128 then
    sfx(3)
    bally = 24
  end
end
```

N'oublie pas d'appeler cette fonction dans update et de créer un son pour l'occasion.

Quand tu tapes RUN, tu dois avoir la meilleure partie du jeu !
Maintenant il est temps de passer au…

XI – Score !

Maintenant que nous avons un jeu, on se doit d'avoir un système de meilleur score !
On aura besoin d'une autre variable en haut de notre fichier :

```
ballx = 64
bally = 64
ballsize = 3
ballxdir = 5
ballydir = 3
padx = 53
pady = 122
padw = 24
padh = 4
score = 0
```

Et, à chaque rebond sur la raquette, on augmentera le score :

```
function bouncepaddle()
  if ballx >= padx and ballx <= padx + padw and bally > pady then
    sfx(0)
    ballydir =- ballydir
    score += 10 -- augmente le score
  end
end
```

Ensuite on peut afficher le score dans la fonction DRAW :

```
function _draw()
  -- nettoye l'écran
  rectfill(0, 0, 128, 128, 3)
  --dessine le score
  print(score, 12, 6, 15)
  -- Dessine la raquette
  rectfill(padx, pady, padx+padw, pady+padh, 15)
  -- Dessine la balle
  circfill(ballx, bally, ballsize, 15)
end
```

RUN, et voilà !

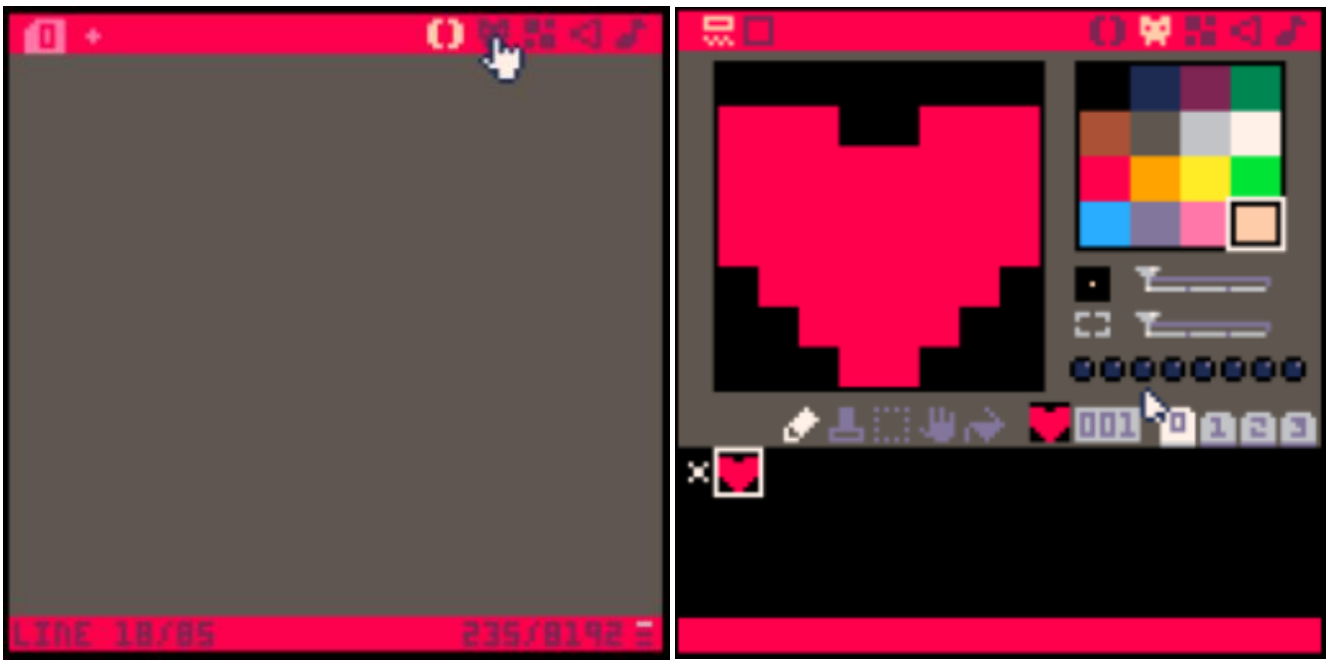
- Les commandes utilisées

Fonction	Utilisation
print(message, x, y, col)	**Cette fonction écrit un message à l'écran. X et Y sont les coordonnées du coin bas gauche de la première lettre**

XII – Cœur

Le prochain morceau du puzzle est le système de vie du joueur.

On aura besoin d'un sprite (une petite image) pour montrer un cœur. Ouvre donc le Sprite Editor dans PICO-8 :



Rappelle-toi le numéro du sprite que tu viens de créer, par exemple 004.Tu peux maintenant rajouter une variable en haut du fichier :

```
ballx = 64
bally = 64
ballsize = 3
ballxdir = 5
ballydir = 3
padx = 53
pady = 122
padw = 24
padh = 4
score = 0
lives = 3
```

Ainsi que ce code dans DRAW :

```
function _draw()
  -- nettoye l'écran
  rectfill(0, 0, 128, 128, 3)
  --dessine le score
  print(score, 12, 6, 15)
  --dessine le sprite de vie
  for i = 1, lives do
    spr(001, 100, 4)
  end
  --dessine la vie
  print(lives, 110, 6, 15)
  -- Dessine la raquette
  rectfill(padx, pady, padx+padw, pady+padh, 15)
  -- Dessine la balle
  circfill(ballx, bally, ballsize, 15)
end
```

- Les commandes utilisées

Fonction	Utilisation
spr(number, x, y)	**Dessine un sprite sur l'écran avec son coin en haut à gauche aux coordonnées X et Y.**

Les dernières fonctionnalités que l'on doit ajouter sont la possibilité de perdre une vie et la fin du jeu une fois que notre vie est à zéro. On doit rendre notre fonction "LOSEBALL" un peu plus compliquée :

```
function loseball()
  if bally > 128 then
    if lives > 0 then
      sfx(3)
      bally = 24
      lives -= 1
    end
    else if lives == 0 then
      ballydir = 0
      ballxdir = 0
      bally = 64
      sfx(5)
    end
  end
end
```

Tu peux aussi ajouter un autre son pour la fin de la partie.

XIII – Conclusion

Félicitations, Nolan est ravi de voir que tu as réussi à réaliser ce magnifique jeu. Toutefois il souhaiterait que tu y ajoutes quelques bonus avant de le commercialiser. Voici ses retours...

XIX – Bonus

Et si on y ajoutait du multijoueur ? Faire quelques parties avec un ami serait cool.

Ça manque de son je trouve. Rajoutes quelques bruitages.

Il n'y a pas assez de couleurs. Essaie de faire en sorte qu'il y ait plus de couleurs, voir même que la raquette change de couleur à chaque contact avec la balle.

Et une IA aussi ça serait pas mal.