

Winter Camp



Co/ding
<Club>
BY EPITECH

INTRODUCTION

Du haut d'un royaume enneigé, un jeune garçon rêveur et curieux, Tony, adorait passer ses journées à explorer les forêts et les montagnes environnantes. Un jour, il entendit un bruit étrange provenant des arbres et découvrit une petite fée en train de pleurer. La fée lui expliqua qu'elle avait perdu sa baguette magique et qu'elle ne pouvait plus rentrer chez elle. Tony décida de l'aider. Après un long périple, il se retrouva devant une grotte et aperçu la baguette magique. Il ne lui resta plus qu'une chose à faire, traverser la grotte gelée...

L'objectif de cet atelier est la réalisation d'un plateformer en Lua en utilisant le logiciel Pico8.
Pour cela vous allez devoir vous familiariser avec tout les outils disponible sur le logiciel et ainsi réaliser:

- vos propres sprites et assets
- créer votre environnement de jeux
- ajouter des features annexes



INSTALLATION

Téléchargez le .zip

DÉCOUVERTE DE PICO-8

Ouvrez Pico-8 et écrivez la commande `load exemple_plateformer.p8`, ensuite libre à vous d'explorer et modifier ce que vous souhaitez à travers le fichier.

Pour télécharger de nouveaux sprites et assets: (<https://itch.io/game-assets/tag-pico-8>).
une fois que vous vous êtes familiarisé appuyez sur echap et écrivez sur la console `reboot`
c'est maintenant à vous de créer votre propre jeu.



pensez à régulièrement sauvegarder votre travail à l'aide de `save`

DESIGN DE VOTRE PERSONNAGE

L'objectif : réalisation du design de votre personnage dans le deuxième onglet du logiciel.

Commençons par une phase statique de votre personnage qui prendra 2 sprites.

Ensuite faites son déplacement en 4 sprites.

Réaliser une animation de saut en 3 sprites.

Et pour finir une animation de slide en 1 sprite.

Pour les plus motivés, libre à vous de réaliser des sprites annexes comme des mobs ainsi que des PNJs.

Si vous n'avez pas d'inspiration ou de temps, vous pouvez utiliser ces assets si dessous:

```
static 1: [gfx]08080a8aaa0008acfc00aff8fa0aa0200a00f7f7f00f07770f000bbb00000f0f000 [/
gfx]
static 2: [gfx]08080a8aaa0008acfc00afffa00a0200a00f7f7f000f777f0000bbb00000f0f000 [/
gfx]
course 1: [gfx]0808000a8aaa0aa8aaafaaaaf7cf000fff8000770000077f0000bbbff00f0050000 [/
gfx]
course 2: [gfx]0808000a8aaa00a8aaafaaaaf7cf000fff8000770000077f0000bbb0ff00f500000 [/
gfx]
course 3: [gfx]0808000a8aaa0a8aaafaaaaf7cf000fff8000770000077f0000bbbff00500f0000 [/
gfx]
course 4: [gfx]0808000a8aaa00a8aaafaaaaf7cf000fff8000770000077f0000bbb0ff005f00000 [/
gfx]
saut 1 : [gfx]0808000a8aaa00a8aaaf0aaaf7cf000fff8aa07700000ff70000bbb00000f500000 [/
gfx]
saut 2 : [gfx]0808a008aaa0aa8aaff00aaa7cf000fff8000770000007fff0000bbb000000f5000 [/
gfx]
slide : [gfx]0808000aaa000000aaa00a8aaa0008acfc00aaff8f0f0aa277f000077b0000f1bf50 [/
gfx]
```



CRÉATION DE LA MAP

Pour la création de votre environnement de jeu, vous allez aussi devoir avoir des assets.

Conseil : créez les vous même. Mais si vous le souhaitez un pack d'asset est lui aussi disponible ci-dessous.

```
bloc 1: [gfx]0808bbbbbbbbbbbbbbbbbbbbbbbb33b33bb333b343b344344434444a4444944445444 [/
gfx]
bloc 2: [gfx]0808444444449444444444544a444444444444444444224454442e244444422444 [/
gfx]
```

Une fois que vous avez vos blocs, donnez leur les flags 0 et 1 et ensuite allez dans l'onglet numéro 3 du logiciel afin de réaliser votre map.



SETUP LES VARIABLES

Dans l'onglet code de pico 8 commençons par créer la fonction "init"

```
function _init()  
  
end
```

En mettant nos variables de jeu directement dans notre fonction principale, nous pourrions aisément reset notre jeu en appelant "init".

Vous allez voir que cela sera pratique pour la suite.

Pour commencer, voici un exemple de fonction init que je vous conseille d'utiliser pour le début et qui sera modifiable selon vos envies par la suite.

```
function _init()  
  player={  
    sp=1,  
    x=59,  
    y=59,  
    w=8,  
    h=8,  
    flp=false,  
    dx=0,  
    dy=0,  
    max_dx=2,  
    max_dy=3,  
    acc=0.5,  
    boost=4,  
    anim=0,  
    running=false,  
    jumping=false,  
    falling=false,  
    sliding=false,  
    landed=false  
  }  
  
  gravity=0.3  
  friction=0.85  
  
  --simple camera  
  cam_x=0  
  
  --map limits  
  map_start=0  
  map_end=1024  
end
```



LES FONCTIONS UPDATE & DRAW

La fonction “update” va mettre à jour ce que vous allez voir sur votre jeu.

La fonction “draw” va afficher votre map et votre sprite.

Dans la fonction “update” vous allez gérer votre caméra.

Dans l'exemple ci-dessous la caméra ne prend en compte que l'axe des abscisses donc à vous de la modifier afin qu'elle prenne aussi l'axe des ordonnées.

La fonction “draw” ci-dessous choisit la couleur de votre fond via `cls()` (ici le fond sera noir), affiche votre map via `map()` et dessine votre sprite avec `spr()`.

```
function _update()
    player_update()
    player_animate()

    --simple camera
    cam_x=player.x-64+(player.w/2)
    if cam_x<map_start then
        cam_x=map_start
    end
    if cam_x>map_end-128 then
        cam_x=map_end-128
    end
    camera(cam_x,0)
end

function _draw()
    cls()
    map(0,0)
    spr(player.sp,player.x,player.y,1,1,player.flp)
end
```

Maintenant vous avez la base du fonctionnement de votre jeu.

LES COLLISIONS

Les collisions sont la plus grosse partie de notre jeu, vous allez découvrir la notion des flags sur vos sprites et assets.

Les flags sont des moyens de donner de conditions de comportements à tous vos assets ce qui nous sera très utiles pour la gestion de collision entre notre map et notre player



Pour cela je vais juste vous demander de copier/coller ce bout de code.

```
--collisions

function collide_map(obj, aim, flag)
  --obj = table needs x,y,w,h
  --aim = left, right, up, down

  local x=obj.x  local y=obj.y
  local w=obj.w  local h=obj.h

  local x1=0  local y1=0
  local x2=0  local y2=0

  if aim=="left" then
    x1=x-1  y1=y
    x2=x    y2=y+h-1

  elseif aim=="right" then
    x1=x+w-1  y1=y
    x2=x+w    y2=y+h-1

  elseif aim=="up" then
    x1=x+2  y1=y-1
    x2=x+w-3  y2=y

  elseif aim=="down" then
    x1=x+2  y1=y+h
    x2=x+w-3  y2=y+h
  end

  --pixels to tiles
  x1/=8  y1/=8
  x2/=8  y2/=8

  if fget(mget(x1,y1), flag)
  or fget(mget(x1,y2), flag)
  or fget(mget(x2,y1), flag)
  or fget(mget(x2,y2), flag) then
    return true
  else
    return false
  end
end

end
```




LES DÉPLACEMENTS

Pour finir vous allez mettre les déplacements de votre personnage dans la fonction ci dessous:

```
function player_update()
```

```
end
```

l'objectif ici est d'utiliser les touches directionnelles de votre clavier pour le déplacer ainsi que la touche 'X' pour le faire sauter.

Pour cela pensez à utiliser les paramètres que vous avez donnés dans la fonction "init"

Une fois les déplacements faits, ajouter ce morceau de code juste après, il va permettre de vérifier si votre personnage collisionne avec le décor

```
--check collision up and down
if player.dy>0 then
    player.falling=true
    player.landed=false
    player.jumping=false

    player.dy=limit_speed(player.dy,player.max_dy)

    if collide_map(player,"down",0) then
        player.landed=true
        player.falling=false
        player.dy=0
        player.y-=((player.y+player.h+1)%8)-1
    end
elseif player.dy<0 then
    player.jumping=true
    if collide_map(player,"up",1) then
        player.dy=0
    end
end

--check collision left and right
if player.dx<0 then

    player.dx=limit_speed(player.dx,player.max_dx)

    if collide_map(player,"left",1) then
        player.dx=0
    end
elseif player.dx>0 then

    player.dx=limit_speed(player.dx,player.max_dx)

    if collide_map(player,"right",1) then
        player.dx=0
    end
end
```

Maintenant il ne vous reste plus qu'à animer votre personnage dans la fonction ci dessous:

```
function player_animate()
```

```
end
```

Pour finir ajoutez tout à la fin cette fonction qui consistera à limiter la vitesse maximale de votre personnage:



```
function limit_speed(num,maximum)
  return mid(-maximum,num,maximum)
end
```

VOILA VOUS AVEZ LA BASE D'UN PLATEFORMER !

N'hésitez pas à ajouter ce que vous voulez comme feature afin d'aller plus loin si le coeur vous en dit.

Merci d'avoir suivi cet atelier !