16

Inventaire et objet à ramasser

Dans les jeux vidéo en général et dans les jeux de type RPG tout particulièrement, un système d'inventaire est indispensable. Il permet de savoir ce dont le joueur dispose (argent, équipement, objets, etc.). Dans ce chapitre, nous allons voir comment créer un système d'inventaire élémentaire en permettant au joueur de ramasser une clé. Nous en profiterons pour créer une interface minimaliste affichant le nombre de clés que le joueur possède. Pour suivre ce chapitre, copiez le projet 18 et appelez le dossier copié `19-Inventaire`. Ouvrez ce dossier pour travailler dedans.

16.1. Création de l'interface utilisateur

L'interface utilisateur (UI ou GUI) correspond aux éléments qui sont affichés à l'écran par-dessus le jeu. L'exemple le plus courant est certainement la barre de vie du personnage. Dans notre cas, nous allons créer une interface basique afin d'afficher notre inventaire. Dans le cadre de ce livre, le seul objet que le personnage pourra ramasser sera une petite clé qui permettra d'ouvrir la maison. Nous afficherons donc à l'écran une icône de clé avec la quantité de clés que possède le joueur.

Commencez par télécharger une image de clé. J'ai trouvé une liste de quatre clés sur OpenGameArt (image créée par BizmasterStudios) : ce sera parfait pour notre projet. Téléchargez l'image, renommez-la en `keys.png` et placez-la dans votre projet.

Créez ensuite un nouveau script `gui.lua`. Ce script contiendra les fonctions permettant l'affichage des éléments d'interface. Nous aurons besoin d'inclure l'image de la clé dans notre projet afin de pouvoir l'afficher. Notre clé se trouve sur la première map, nous la chargerons donc dans ce fichier. Ouvrez le fichier `map1.lua`.

Créez une variable pour charger l'image et créez une variable qui découpera l'image sous forme de quad afin de ne dessiner qu'une seule clé. Nous avons déjà vu ces fonctions ensemble. Pour le quad, j'affiche la dernière clé de l'image.

```
map1.keyset = love.graphics.newImage("keys.png")
map1.keyquad = love.graphics.newQuad(3*tileSize, 0*tileSize, tileSize,
 tileSize, map1.keyset :getDimensions())
```

Il nous faudra une autre variable. Cette variable contiendra le nombre de clés ramassées. Je vous propose de créer un script `inventaire.lua` qui regroupera les informations sur les objets possédés par le joueur. Pour le moment, ce script `inventaire.lua` ne contiendra que la ligne : `key = 0`. Il s'agit du nombre de clés que le joueur possède. Nous allons laisser ce script comme cela mais si vous avez besoin (pour vos futurs projets ou idées) de gérer plus d'objets, vous pourrez créer ici les variables.

Maintenant que vous avez vos variables, retournez dans le script `gui.lua` que nous venons de mettre en place et créez une fonction `DrawGui()`. Dans cette fonction, affichez la clé en haut à gauche de l'écran et affichez un texte qui correspondra au nombre de clés que le joueur possède. Il s'agit là aussi d'un bout de code que vous devez déjà maîtriser.

```
function DrawGui()
  love.graphics.draw(map1.keyset, map1.keyquad, 10, 10)
  love.graphics.print(key,55,6)
end
```

Pour afficher notre interface, nous devons nous rendre dans le fichier `main.lua`. Premièrement, pensez à inclure les deux nouveaux fichiers (`gui` et `inventaire`) que nous venons de créer. Au total vous devriez avoir six inclusions dans le fichier `main` :

```
require "collision" -- Pour accéder à la fonction de collision
require "settings" -- Nos options
require "inventaire" -- Inventaire
require "gui" -- GUI
require "allmaps" -- Tous les fichiers map
flux = require "flux" -- Outil flux pour les tweens
```

Si vous laissez votre projet tel quel, le texte que vous affichez à l'écran sera bien trop petit. Je vous propose d'ajouter la ligne suivante dans la fonction `load` du fichier `main` : `love.graphics.setFont(love.graphics.newFont(32))`. Elle permettra de définir une taille de texte de 32. Maintenant, pour afficher votre interface, appelez la fonction `DrawGui()` dans la fonction `draw` du fichier `main`. Normalement tout devrait fonctionner.

Figure 16.1 : *L'interface utilisateur*

Notre script fonctionne parfaitement mais l'interface n'est pas très lisible à cause de l'arrière-plan. Je vous propose de modifier la fonction `DrawGui` et d'ajouter un fond sombre derrière l'interface pour la rendre plus lisible :

```
function DrawGui()
  love.graphics.setColor(0, 0, 0, 0.5)
  love.graphics.rectangle("fill", 0, 0, 90, 52)
  love.graphics.setColor(1, 1, 1)
  love.graphics.draw(map1.keyset, map1.keyquad, 10, 10)
  love.graphics.print(key,55,6)
end
```

Ce qui nous permettra d'obtenir le résultat suivant :

Figure 16.2 : Un fond sombre pour l'UI

C'est tout ce que nous allons faire au niveau de l'interface utilisateur. Avec cette technique vous pourrez créer une barre de vie, une barre d'énergie, une barre de compétences, etc. Je vous invite à vous exercer et ajouter d'autres éléments d'interface qui vous seraient utiles.

16.2. Création d'un objet à ramasser

Retournez dans le fichier `map1.lua`. Ici, nous allons afficher la clé (l'objet à ramasser) à l'écran. Pour cela, créez une fonction `draw` (de la même façon que nous avons créé la fonction `update`) et ajoutez le code permettant de dessiner la clé à l'écran :

```
function map1.draw()
  love.graphics.draw(map1.keyset, map1.keyquad, 27*tileSize, 1*tileSize)
end
```

J'affiche donc la clé en haut à droite de la map 1 (voir Figure 16.3).