Projet Technologique « Battlefield Mario » : Fiche n°4

Désormais vous connaissez les bases de la bibliothèque SDL pour créer un jeu de plates-formes avec un personnage qui se déplace, saute et tire des projectiles en évitant ou éliminant des ennemis sur une carte pré-définie.

1 Les débuts de Mario

Commencez par faire une copie des sources du jeu que vous avez précédemment créé, Bird. Sur cette copie, supprimez tous les éléments spécifiques au jeu précédent et créez un nouveau module `mario.h`, `mario.c` qui implémentera les comportements de notre personnage principal. Profitez-en pour intégrer le sprite de Mario plutôt que celui de l'oiseau. Pensez-bien que celui-ci n'a pas (encore) d'ailes. Donc sa position de départ sera en bas à gauche de l'écran et de la carte. Mario doit pouvoir se déplacer à gauche et à droite dans un premier temps.

2 Une carte du jeu simple

Définissez le module `map.h`, `map.c` qui permet de manipuler la carte du jeu.La figure 1 présente un exemple d'interface minimale de ce module.

Vous pouvez considérer que la carte est un simple tableau à deux dimensions. Chaque case du tableau représente un objet ou une valeur par défaut si la carte ne contient pas d'objets à cet endroit.

Remarquez que les objets n'ont pas tous la même propriété. Certains sont solides (on ne peut jamais les traverser), d'autres sont semi-solides (on peut les traverser vers le haut, mais ils se comportent comme un plancher lorsque l'on tombe dessus), d'autres encore sont comme de l'air : on peut les traverser.

```c
#ifndef MAP_IS_DEF
#define MAP_IS_DEF

  enum {
  MAP_OBJECT_SOLID,
  MAP_OBJECT_SEMI_SOLID,
  MAP_OBJECT_AIR,
  MAP_OBJECT_NUM
  };

  ...

  void map_new(unsigned width, unsigned height);
  void map_allocate(unsigned width, unsigned height);
  void map_set(int map_object, int x, int y);
  int map_get(int x, int y);
  void map_object_add(const char* path, int nb_sprites, int type);
  ...

#endif
```

FIGURE 1 – Interface du module map.h

- `void map_allocate(unsigned width, unsigned height)` renvoie une carte sans aucun objet ;
- `void map_set(int map_object, int x, int y)` place un objet `map_object` aux coordonnées `(x,y)` ;
- `int map_get(int x, int y)` renvoie l'objet présents aux coordonnées `(x,y)` ;
- `void map_object_add(const char* path, int nb_sprites, int type)` ajoute un objet dont la représentation graphique est dans le fichier `path`, le nombre de sprites associés ainsi qu'une ou plusieurs propriétés (eg. `MAP_OBJECT_SOLID`).

Vous pouvez ainsi définir la fonction `void map_new(unsigned width, unsigned height)` qui crée une carte 2D avec un sol solide (*ground.png*), une taille `width * height` (potentiellement plus grande que la taille de la fenêtre) et des murs (*wall.png*) à chaque extrémité.

3 Qui ne saute pas n'est pas Mario

Gérez l'appui sur la flèche du haut du clavier comme un saut de Mario (élevation puis chute). Associez également la barre Espace à l'envoi d'un laser généré depuis Mario et dans la direction de son regard.

4 Gestion des collisions, le retour

Pour éviter que Mario ne sorte de la carte du jeu, il faut faire en sorte qu'il ne puisse pas traverser les objets solides (murs, plates-formes, ...). Vous pourriez être tentés de réutiliser la détection

de collision entre objets mis en place lors de la fiche précédente. Étant donné qu'il s'agit d'un objet de la carte elle-même, il est préférable de réaliser une détection par anticipation pour ce genre de cas. Ainsi, lors d'un déplacement de Mario, évaluez la distance par rapport à un mur pour le stopper uniquement quand il est accolé au mur (au pixel près).

5 Les propriétés des objets de la carte

La manipulation des propriétés des objets permet d'employer un grand nombre de textures différentes tout en testant un nombre réduit de caractéristiques pendant le déroulement du jeu, pour la gestion des collisions notamment. Actuellement trois propriétés définissent les objets :

— `MAP_OBJECT_SOLID` : des objets que Mario ne peut traverser ;
— `MAP_OBJECT_SEMI_SOLID` : des objets que Mario peut traverser temporairement comme des plateformes ;
— `MAP_OBJECT_AIR` : des objets que Mario peut traverser (fleurs du décor, objets collectables, ...).

5.1 Nouvelles propriétés, ...

Ajoutez les propriétés suivantes :

— `MAP_OBJECT_LIQUID`
— `MAP_OBJECT_COLLECTIBLE`
— `MAP_OBJECT_DESTRUCTIBLE`
— `MAP_OBJECT_EXPLOSIVE`
— `MAP_OBJECT_TRANSPARENT`

5.2 ... nouveaux objets

Modifiez la carte pour inclure deux nouveaux objets :

1. des fleurs, uniquement destinées à égayer le paysage (*flower.png*), qui auront comme propriété d'être de type `MAP_OBJECT_AIR`;
2. des pièces à collecter, qui sont des objets animés (le fichier *coin.png* est composé de 16 sprites) qui ont comme propriétés `MAP_OBJECT_AIR` et `MAP_OBJECT_COLLECTIBLE`.