Une image contenant shoji, bâtiment, joueur

Description générée automatiquement

**CASSE-BRIQUES**

Broquet Rayan

EMT-DIVTEC

Rapport technique

DOCUMENTATION INTÉGRÉE AU CODE SOURCE

Table des matières

[Description du projet 2](#_Toc62465493)

[Convention de nommage 2](#_Toc62465494)

[Les variables 2](#_Toc62465495)

[Les booléens 2](#_Toc62465496)

[Les Constantes 2](#_Toc62465497)

[Les fonctions 2](#_Toc62465498)

[Eléments techniques 2](#_Toc62465499)

[Les signals 2](#_Toc62465500)

[Les slots 4](#_Toc62465501)

[Initialisation du jeu 4](#_Toc62465502)

# Description du projet

Mon projet est un simple casse-briques fait en 2D qui gère les collisions, les vies du joueur et la balle qui s’y déplace pour détruire les blocs. Un menu pour commencer, redémarrer ou quitter le jeu est présent dans le projet.

# Convention de nommage

## Les variables

Les variables ont 2 types de nommages, sois ***m\_maVariable*** qui représente une variable membre au sein du programme, sois le deuxième type de variable qui sont les pointers, exemple : ***m\_pMaVariable*** qui représente un pointer comme un Sprite ou un GameScene.  
Après les différents préfixes, commencent par une majuscule et sont camélisées  
Ordre des préfixes :  
‘***m\_***’, et finalement ‘***p***’

## Les booléens

Les booléens sont nommés comme les variables membres et peuvent avoir le mot ***is*** devant le nom si ça représente un état du jeu, exemple : bool m\_isDead = false;

## Les Constantes

Les constantes sont définies en haut de page, par exemple dans le fichier GameCore.cpp et sont écrites en majuscule suivies d’un « \_ » pour séparer les mots, exemple : const int BRICK\_SIZE = 15;

## Les fonctions

Les fonctions sont définies comme une variable lambda, elle a son nom séparé par une majuscule entre chaque mot, exemple : void createSceneWin();

# Eléments techniques

## Les signals

J’utilise des signals durant le déroulement de mon jeu.

Une image contenant texte

Description générée automatiquementCe signal me permet de vérifier que le pad ne sort pas de l’écran lors que le joueur joue avec la souris et en le replacement à la bordure si il essaie d’en sortir.

Le code ci-dessous me permet de gérer les clics de l’utilisateur sur mes sprites boutons sur l’écran du menu. Tout le reste du code est expliqué via les commentaires.



Une image contenant texte

Description générée automatiquement

Le code suivant gère si le joueur appuie sur une touche, exemple si l’utilisateur appuie sur la touche Echap, il est emmené au menu du jeu.

## Les slots

J’utilise un slot pour faire la connexion entre les blocs détruits et ma variable qui compte le nombre de blocs. Pour se faire, j’utilise la fonction onSpriteDestroyed() définit dans GameCore.h

## Initialisation du jeu

Voici comment je créer mes différentes scènes et autres méthodes utile au fonctionnement de mon jeu, ce qu’il est important à retenir est le ***m\_pGameCanvas->startTick();*** qui va démarrer le tick (la cadence) du jeu pour permettre à la boule et aux différents éléments de fonctionner et de se déplacer, sans cela, le jeu resterait immobile et rien ne se passerait.