|  |
| --- |
| DivTec |
| Rapport technique |
| Documentation technique du projet |

|  |
| --- |
| Nussbaumer Thibaud  25/10/2019 |

Table des matières

[1 Introduction 2](#_Toc26192110)

[2 Description du projet 2](#_Toc26192111)

[3 Convention de nommage 2](#_Toc26192112)

[4 Éléments du code nécessitant une explication 3](#_Toc26192113)

# Introduction

Ce document a pour but de réunir les différentes informations techniques tels que la description du projet, la convention de nommage, les fonctions que j’ai créées ainsi que des explications pour pouvoir améliorer, corriger et distribuer mon projet.

# Description du projet

Ce projet fait suite à celui des portes ouvertes, le but étant de créer un jeu 2d orienté objet (dans le langage que l’on veut) d’une période allant du 30 septembre jusqu’au 21 décembre.

Ce projet nous permet de mettre en pratique nos connaissances ainsi que notre manière de se débrouiller.

J’ai choisi de le faire avec la bibliothèque GameFramework qui a été développée par M. Conus. Une documentation technique sous format html sera créée grâce à Doxygen (plugin pour faire de la doc). Je travaille avec l’IDE Qt Creator.

# Convention de nommage

La convention de nommage utilisée est semblable si ce n’est la même que celle qu’on apprend lors des modules et des ateliers C++ avec M. Conus.

* Les commentaires
  + Écrits en français
* Les fonctions
  + Nommées en anglais
  + Possèdent un commentaire explicatif lors de la définition
* Les variables
  + En anglais
  + Après les différents préfixes, commencent par une majuscule et sont camélisées
* Les variables membres
  + Elles sont préfixées par ‘m\_’
* Les pointeurs
  + Ils sont préfixés par la lettre ‘p’
* Les éléments QSound
  + Ils sont terminés par le mot ‘Sound’
* Les variables représentant un passage par référence
  + Elles sont préfixées par la lettre ‘r’
* Ordre des préfixes
  + ‘m\_’, ‘r’ et finalement ‘p’
* Les constantes
  + Ecrites entièrement en majuscules avec un tiret bas entre chaque mot
* Les types énumérés
  + Ils sont suffixés de ‘\_e’
* Les structures
  + Elles sont suffixées de ‘\_s’

# Éléments du code nécessitant une explication

## Problème lors du débogage

Quand je voulais déboguer, des fichiers continuaient de charger alors que le programme était lancé, cela affichait une erreur et ensuite mon programme crashait. Pour y remédier M. Conus à instancier un ennemi dans le menu afin que tous les fichiers soient présents, on peut retrouver cette instanciation entre les balises « #ifdef QT\_DEBUG » et « #endif ».

//! Crée la scène Menu et y ajoute différent éléments

void GameCore::**setupSceneMenu**(){

// Création et disposition de la scène Menu

// Crée la scène Menu

m\_pSceneMenu = m\_pGameCanvas->createScene(0, 0, SCENE\_WIDTH, SCENE\_HEIGHT);

// Affichage du titre Menu

m\_pSceneMenu->createText(QPointF(SCENE\_WIDTH-400,SCENE\_HEIGHT/2-200), "Menu", 70);

// Affichage des différentes options du menu

m\_pMenuItems[0] = m\_pSceneMenu->createText(QPointF(SCENE\_WIDTH-400,SCENE\_HEIGHT/2), "Jouer", 50, Qt::*red*);

m\_pMenuItems[1] = m\_pSceneMenu->createText(QPointF(SCENE\_WIDTH-400,SCENE\_HEIGHT/2 + 50), "Contrôles", 50, Qt::*white*);

m\_pMenuItems[2] = m\_pSceneMenu->createText(QPointF(SCENE\_WIDTH-400,SCENE\_HEIGHT/2 + 100), "Quitter", 50, Qt::*white*);

m\_menuChoosenItem = 0;

#ifdef QT\_DEBUG

// Bricolage pour forcer le chargement de tous les plugins dès le lancement

// de l'application. Si on ne fait pas ça, impossible de débogguer l'app, car

// c'est au moment de la création des ennemis que les formats d'images sont chargés,

// ce qui est trop tard.

Enemy\* pEnemy = new Enemy;

pEnemy->setPos(std::rand() % (SCENE\_WIDTH-pEnemy->width()) + 1,100);

pEnemy->setZValue(1);

m\_pSceneMenu->addSpriteToScene(pEnemy);

#endif

}