Eliminator++

v1.0

Généré par Doxygen 1.8.5

Dimanche Novembre 10 2013 18 :11 :51

Table des matières

Chapitre 1

Documentation Eliminator++

1.1 Introduction

Le cas d'utilisation étudié est le déplacement d'un personnage.

1.1.1 Outils de développement :

– C++/Qt5.1

Chapitre 2

Index hiérarchique

2.1 Hiérarchie des classes

Cette liste d'héritage est classée approximativement par ordre alphabétique :

Chapitre 3

Index des classes

3.1 Liste des classes

Liste des classes, structures, unions et interfaces avec une brève description :

Chapitre 4

Index des fichiers

4.1 Liste des fichiers

Liste de tous les fichiers documentés avec une brève description :

Chapitre 5

Documentation des classes

5.1 Référence de la classe Entity

La classe Entity hérite de QGraphicsItem.

Contient la définition d'une Entity.

`#include <entity.h>`

Graphe d'héritage de Entity :

Fonctions membres publiques

- Entity (qreal x, qreal y, int width, int height)
  Constructeur Entity.
- int getWidth () const
  Getter de Entity.
- int getHeight () const
  Getter de Entity.
- qreal getX () const
  Getter de Entity.
- qreal getY () const
  Getter de Entity.
- virtual ∼Entity ()
  Desctucteur ∼Entity.

Attributs privés

- int eWidth
  Largeur de l'entity.
- int eHeight
  Hauteur de l'entity.

5.1.1 Description détaillée

La classe Entity hérite de QGraphicsItem.

Contient la définition d'une Entity.

Auteur

Guillaume Rasolo

Version

1.0

QGraphicsItem est une classe interne de Qt.

Définition à la ligne 22 du fichier entity.h.

5.1.2 Documentation des constructeurs et destructeur

5.1.2.1 Entity : :Entity (qreal *x,* qreal *y,* int *width,* int *height*)

Constructeur Entity.

Paramètres

x	Coordonnée x qui sera donner à l'entity.
y	Coordonnée y qui sera donner à l'entity.
width	Largeur qui sera donner à l'entity.
height	Hauteur qui sera donner à l'entity.

On définie une entity, avec une position et une taille.

Définition à la ligne 10 du fichier entity.cpp.

5.1.3 Documentation des fonctions membres

5.1.3.1 int Entity : :getHeight ( ) const

Getter de Entity.

Renvoie

La hauteur de l'entity.

Définition à la ligne 22 du fichier entity.cpp.

5.1.3.2 int Entity : :getWidth ( ) const

Getter de Entity.

Renvoie

La largeur de l'entity.

Définition à la ligne 17 du fichier entity.cpp.

5.1.3.3 qreal Entity : :getX ( ) const

Getter de Entity.

Renvoie

La coordonnée x de l'entity.

Définition à la ligne 27 du fichier entity.cpp.

5.1.3.4 qreal Entity : :getY ( ) const

Getter de Entity.

Renvoie

La coordonnée y de l'entity.

Définition à la ligne 32 du fichier entity.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :
– entity.h
– entity.cpp

5.2 Référence de la classe GameScene

La classe GameScene hérite de QGraphicsScene.

GameScene s'occupe des objets.

`#include <gamescene.h>`

Graphe d'héritage de GameScene :

Connecteurs publics

– void advance ()
fonction advance

Fonctions membres publiques

– GameScene ()
Constructeur GameScene.

Fonctions membres protégées

– void mousePressEvent (QGraphicsSceneMouseEvent ∗event)
mousePressEvent
– void mouseReleaseEvent (QGraphicsSceneMouseEvent ∗event)
mouseReleaseEvent

Attributs privés

– Entity ∗ player
Pointeur d'un object Entity.

5.2.1 Description détaillée

La classe GameScene hérite de QGraphicsScene.

GameScene s'occupe des objets.

Auteur

Guillaume Rasolo

Version

1.0

QGraphicsScene est une classe interne de Qt.

Définition à la ligne 26 du fichier gamescene.h.

5.2.2 Documentation des constructeurs et destructeur

5.2.2.1 GameScene : :GameScene ( )

Constructeur GameScene.

Constructeur par défaut, qui initialisera la scène.

Définition à la ligne 12 du fichier gamescene.cpp.

5.2.3 Documentation des fonctions membres

5.2.3.1 void GameScene : :advance ( ) [slot]

fonction advance

la fonction advance() est appelée, lorsque l'on reçoi le signal timeout du timer pour rafraîchir la scène.

Définition à la ligne 39 du fichier gamescene.cpp.

5.2.3.2 void GameScene : :mousePressEvent (QGraphicsSceneMouseEvent * *event*) [protected]

mousePressEvent

Paramètres

event	Événement souris.

Définition à la ligne 23 du fichier gamescene.cpp.

5.2.3.3 void GameScene : :mouseReleaseEvent (QGraphicsSceneMouseEvent * *event*) [protected]

mouseReleaseEvent

Paramètres

event	Événement souris.

Définition à la ligne 33 du fichier gamescene.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :
– gamescene.h
– gamescene.cpp

5.3 Référence de la classe GameView

La classe GameView hérite de QGraphicsView.

GameView s'occupe de la vue d'affichage d'une scène.

`#include <gameview.h>`

Graphe d'héritage de GameView :

Fonctions membres publiques

– GameView ()
  Constructeur GameView par défaut.
– GameView (int width, int height)

Constructeur GameView.
- QSize getSize () const
  Getter de GameView.
- int getWidth () const
  Getter de GameView.
- int getHeight () const
  Getter de GameView.
- qreal getZoomView () const
  Getter de GameView.

Attributs privés

- int viewWidth
  Largeur de la view.
- int viewHeight
  Hauteur de la view.
- qreal zoomView
  Zoom de la view.

5.3.1 Description détaillée

La classe GameView hérite de QGraphicsView.

GameView s'occupe de la vue d'affichage d'une scène.

Auteur

Guilaume Rasolo

Version

1.0

QGraphicsView est une classe interne de Qt.

Définition à la ligne 38 du fichier gameview.h.

5.3.2 Documentation des constructeurs et destructeur

5.3.2.1 GameView : :GameView ( )

Constructeur GameView par défaut.

Constructeur par défaut, qui initialisera la view avec les paramètres par défaut.

Définition à la ligne 10 du fichier gameview.cpp.

5.3.2.2 GameView : :GameView (int *width*, int *height*)

Constructeur GameView.

Paramètres

width	Une largeur qui sera donner à la view.
height	Une hauteur qui sera donner à la view.

Définition à la ligne 28 du fichier gameview.cpp.

5.3.3 Documentation des fonctions membres

5.3.3.1 int GameView : :getHeight ( ) const

Getter de GameView.

Renvoie

la Hauteur de la view.

Définition à la ligne 59 du fichier gameview.cpp.

5.3.3.2 QSize GameView : :getSize ( ) const

Getter de GameView.

Renvoie

La dimension de la view.

Définition à la ligne 47 du fichier gameview.cpp.

5.3.3.3 int GameView : :getWidth ( ) const

Getter de GameView.

Renvoie

La largeur de la view.

Définition à la ligne 53 du fichier gameview.cpp.

5.3.3.4 qreal GameView : :getZoomView ( ) const

Getter de GameView.

Renvoie

La valeur du Zoom appliquer à la view.

Définition à la ligne 65 du fichier gameview.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :
– gameview.h
– gameview.cpp

5.4 Référence de la classe GameWindow

La classe GameWindow hérite de QMainWindow.

GameWindow s'occupe de l'affichage.

```
#include <gamewindow.h>
```

Graphe d'héritage de GameWindow :

Fonctions membres publiques

– GameWindow (QWidget ∗parent=0)
Constructeur GameWindow.

Attributs privés

– GameView ∗ gameView
  La view qui contiendra la scène.
– GameScene ∗ gameScene
  La scène qui contiendra les objects.
– QTimer time
  Le timer qui s'occupera du rafraîchissement de la scène.

5.4.1 Description détaillée

La classe GameWindow hérite de QMainWindow.

GameWindow s'occupe de l'affichage.

Auteur

Guilaume Rasolo

Version

1.0

QMainWindow est une classe interne de Qt.

Définition à la ligne 27 du fichier gamewindow.h.

5.4.2 Documentation des constructeurs et destructeur

5.4.2.1 GameWindow : :GameWindow (QWidget ∗ *parent* = 0)

Constructeur GameWindow.

Paramètres

parent	Pointeur vers un QWidget parent, initialiser à 0.

Le constructeur s'occupera de mettre en liaison la scène avec la view. Il gére aussi la dimension de la fenêtre, le titre et le timer.

Définition à la ligne 13 du fichier gamewindow.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :
– gamewindow.h
– gamewindow.cpp

5.5 Référence de la classe Player

La classe Player hérite de Entity.

Contient la définition d'un player.

```
#include <player.h>
```

Graphe d'héritage de Player :

Fonctions membres publiques

– Player ()
  Constructeur Player par défaut.

- Player (qreal x, qreal y, int width, int height, QString crustomPathSprite=NULL)
  Constructeur Player.
- void paint (QPainter ∗painter, const QStyleOptionGraphicsItem ∗option, QWidget ∗widget)
  Permet l'affichage du player avec ces animations correspondantes.
- void advance (int phase)
  Permet le rafraîchissement de l'objet.
- QRectF boundingRect () const
  boundingRect
- QPainterPath shape () const
  shape
- ∼Player ()
  Destructeur ∼Player.

Fonctions membres protégées

- void keyPressEvent (QKeyEvent ∗event)
  keyPressEvent
- void keyReleaseEvent (QKeyEvent ∗event)
  keyReleaseEvent

Fonctions membres privées

- void move (int xa, int ya)
  Gère le déplacement par rapport aux coordonnées x et y données.

Attributs privés

- SpriteImgMove ∗ pSpriteMove
  Pointeur sur SpriteImgMove qui gère l'affichage d'une animation.
- int dirMove
  Valeur de la direction du mouvement à faire.
- int animFrame
  Valeur de la frame d'animation à faire.
- int animDelta
  Valeur de la frame par second.
- int animTime
  Vitesse que prendra une animation de déplacement dans le temps.

5.5.1 Description détaillée

La classe Player hérite de Entity.

Contient la définition d'un player.

Auteur

Guillaume Rasolo

Version

1.0

Définition à la ligne 44 du fichier player.h.

5.5.2 Documentation des constructeurs et destructeur

5.5.2.1 Player : :Player ( )

Constructeur Player par défaut.

On initialise le player avec les paramètres par défaut.

Définition à la ligne 18 du fichier player.cpp.

5.5.2.2 Player : :Player (qreal *x,* qreal *y,* int *width,* int *height,* QString *crustomPathSprite* = NULL)

Constructeur Player.

Paramètres

x	Coordonnée x qui sera donner au player.
y	Coordonnée y qui sera donner au player.
width	Largeur qui sera donner au player.
height	Hauteur qui sera donner au player.
crustomPath-Sprite	Chemin du sprite qui correspond au player. Initialiser à NULL.

Définition à la ligne 39 du fichier player.cpp.

5.5.3 Documentation des fonctions membres

5.5.3.1 void Player : :advance (int *phase*)

Permet le rafraîchissement de l'objet.

Advance() est appelée que lorsque la scène lui dit de se rafraîchir.

Paramètres

phase	

Définition à la ligne 76 du fichier player.cpp.

5.5.3.2 QRectF Player : :boundingRect ( ) const

boundingRect

Renvoie

La dimension du rectangle.

Définition à la ligne 122 du fichier player.cpp.

5.5.3.3 void Player : :keyPressEvent (QKeyEvent * *event*) [protected]

keyPressEvent

Paramètres

event	Événement clavier

Définition à la ligne 172 du fichier player.cpp.

5.5.3.4 void Player : :keyReleaseEvent (QKeyEvent * *event*) [protected]

keyReleaseEvent

Paramètres

event	Événement clavier

Définition à la ligne 191 du fichier player.cpp.

5.5.3.5 void Player : :move (int *xa,* int *ya*) [private]

Gère le déplacement par rapport aux coordonnées x et y données.

Paramètres

xa	Coordonnée x de déplacement.
ya	Coordonnée y de déplacement.

Définition à la ligne 140 du fichier player.cpp.

5.5.3.6 void Player : :paint (QPainter ∗ *painter,* const QStyleOptionGraphicsItem ∗ *option,* QWidget ∗ *widget*)

Permet l'affichage du player avec ces animations correspondantes.

Paramètres

painter	Pointeur du painter de la fenêtre qui servira a dessiner.
option	
widget	

Définition à la ligne 67 du fichier player.cpp.

5.5.3.7 QPainterPath Player : :shape ( ) const

shape

Renvoie

La forme de l'objet player comme un QPainterPath en coordonnées locales.

La fonction est très utilisée pour les collisions.

Définition à la ligne 127 du fichier player.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :
– player.h
– player.cpp

5.6 Référence de la classe SpriteImgMove

La classe SpriteImgMove.

`#include <spriteimgmove.h>`

Fonctions membres publiques

– SpriteImgMove (QString pathSprite)
Constructeur SpriteImgMove.
– ∼SpriteImgMove ()
Destructeur ∼SpriteImgMove.
– void render (qreal x, qreal y, int width, int height, QPainter ∗painter, int dirMove, int animFrame)
Permet de dessiner l'animation du sprite du personnage.
– bool getIsWalking () const

SpriteImgMove.
- void setIsWalking (bool value)
  setIsWalking
- bool getIsLookingUp () const
  getIsLookingUp
- void setIsLookingUp (bool value)
  setIsLookingUp
- bool getIsLookingLeft () const
  getIsLookingLeft
- void setIsLookingLeft (bool value)
  setIsLookingLeft
- bool getIsLookingRight () const
  getIsLookingRight
- void setIsLookingRight (bool value)
  setIsLookingRight
- bool getIsLookingDown () const
  getIsLookingDown
- void setIsLookingDown (bool value)
  setIsLookingDown

Attributs privés

- QPixmap sprite
  Contient le sprite du personnage.
- bool isWalking
  Définie si le personnage bouge.
- bool isLookingUp
  Définie si le personnage regard en haut.
- bool isLookingLeft
  Définie si le personnage regard à gauche.
- bool isLookingRight
  Définie si le personnage regard à droite.
- bool isLookingDown
  Définie si le personnage regard en bas.
- const int pImgDown [3][2] = {{ 0, 0 }, { 1, 0 }, { 2, 0 }}
  On définie une matrice d'animation du déplacement en bas.
- const int pImgLeft [3][2] = {{ 0, 1 }, { 1, 1 }, { 2, 1 }}
  On définie une matrice d'animation du déplacement à gauche.
- const int pImgRight [3][2] = {{ 0, 2 }, { 1, 2 }, { 2, 2 }}
  On définie une matrice d'animation du déplacement à droite.
- const int pImgUp [3][2] = {{ 0, 3 }, { 1, 3 }, { 2, 3 }}
  On définie une matrice d'animation du déplacement en haut.

5.6.1 Description détaillée

La classe SpriteImgMove.

Auteur

Guillaume Rasolo

Version

1.0

La classe s'occupe que de gérer les animations de déplacement, garce aux informations qu'il reçoit de la classe qui l'appelle.

Définition à la ligne 23 du fichier spriteimgmove.h.

5.6.2 Documentation des constructeurs et destructeur

5.6.2.1 SpriteImgMove : :SpriteImgMove (QString *pathSprite*)

Constructeur SpriteImgMove.

Paramètres

pathSprite	Chemin du sprite qui correspond au personnage.

Définition à la ligne 12 du fichier spriteimgmove.cpp.

5.6.3 Documentation des fonctions membres

5.6.3.1 bool SpriteImgMove : :getIsLookingDown ( ) const

getIsLookingDown

Renvoie

Définition à la ligne 113 du fichier spriteimgmove.cpp.

5.6.3.2 bool SpriteImgMove : :getIsLookingLeft ( ) const

getIsLookingLeft

Renvoie

Définition à la ligne 93 du fichier spriteimgmove.cpp.

5.6.3.3 bool SpriteImgMove : :getIsLookingRight ( ) const

getIsLookingRight

Renvoie

Définition à la ligne 103 du fichier spriteimgmove.cpp.

5.6.3.4 bool SpriteImgMove : :getIsLookingUp ( ) const

getIsLookingUp

Renvoie

Définition à la ligne 83 du fichier spriteimgmove.cpp.

5.6.3.5 bool SpriteImgMove : :getIsWalking ( ) const

SpriteImgMove.

Renvoie

Définition à la ligne 73 du fichier spriteimgmove.cpp.

5.6.3.6 void SpriteImgMove : :render (qreal *x,* qreal *y,* int *width,* int *height,* QPainter ∗ *painter,* int *dirMove,* int *animFrame*)

Permet de dessiner l'animation du sprite du personnage.

Paramètres

x	Coordonnée x de la position.
y	Coordonnée y de la position.
width	Largeur du rectangle du sprite qui doit être dessiné.
height	Hauteur du rectangle du sprite qui doit être dessiné.
painter	Pointeur du painter de la fenêtre qui servira à dessiner.
dirMove	Valeur de la direction du mouvement qui sera donnée.
animFrame	Valeur de la frame d'animation qui sera donnée.

Définition à la ligne 25 du fichier spriteimgmove.cpp.

5.6.3.7 void SpriteImgMove : :setIsLookingDown (bool *value*)

setIsLookingDown

Paramètres

value	

Définition à la ligne 118 du fichier spriteimgmove.cpp.

5.6.3.8 void SpriteImgMove : :setIsLookingLeft (bool *value*)

setIsLookingLeft

Paramètres

value	

Définition à la ligne 98 du fichier spriteimgmove.cpp.

5.6.3.9 void SpriteImgMove : :setIsLookingRight (bool *value*)

setIsLookingRight

Paramètres

value	

Définition à la ligne 108 du fichier spriteimgmove.cpp.

5.6.3.10 void SpriteImgMove : :setIsLookingUp (bool *value*)

setIsLookingUp

Paramètres

value	

Définition à la ligne 88 du fichier spriteimgmove.cpp.

5.6.3.11 void SpriteImgMove : :setIsWalking (bool *value*)

setIsWalking

Paramètres

value	

Définition à la ligne 78 du fichier spriteimgmove.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :
– spriteimgmove.h
– spriteimgmove.cpp

Chapitre 6

Documentation des fichiers

6.1 Référence du fichier directionmove.h

Contient l'enum DirectionMove.

Énumérations

– enum DirectionMove { DIR_DOWN_MOVING, DIR_LEFT_MOVING, DIR_UP_MOVING, DIR_RIGHT_MOVING }
The DirectionMove enum.

6.1.1 Description détaillée

Contient l'enum DirectionMove.

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Définition dans le fichier directionmove.h.

6.1.2 Documentation du type de l'énumération

6.1.2.1 enum DirectionMove

The DirectionMove enum.

DirectionMove est une série de constantes prédéfinie pour définir la la direction de déplacement.

Valeurs énumérées

DIR_DOWN_MOVING Déplacement vers le bas.

DIR_LEFT_MOVING Déplacement vers la gauche.

DIR_UP_MOVING Déplacement vers le haut.

DIR_RIGHT_MOVING Déplacement vers la droite.

Définition à la ligne 18 du fichier directionmove.h.

6.2 Référence du fichier entity.cpp

```
#include "entity.h"
```

6.2.1 Description détaillée

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Définition dans le fichier entity.cpp.

6.3 Référence du fichier entity.h

Contient la définition d'une Entity.

```
#include <QGraphicsItem>
```

Classes

– class Entity
La classe Entity hérite de QGraphicsItem.
Contient la définition d'une Entity.

6.3.1 Description détaillée

Contient la définition d'une Entity.

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Une entity est un objet qui a la possibilité de se déplacer dans une scène.

Définition dans le fichier entity.h.

6.4 Référence du fichier gamescene.cpp

```
#include "player.h"
#include "entity.h"
#include "gamescene.h"
```

6.4.1 Description détaillée

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Définition dans le fichier gamescene.cpp.

6.5 Référence du fichier gamescene.h

GameScene s'occupe des objects.

```
#include <QGraphicsScene>
```

Classes

– class GameScene
La classe GameScene hérite de QGraphicsScene.
GameScene s'occupe des objets.

6.5.1 Description détaillée

GameScene s'occupe des objects.

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Elle s'occupe du comportement des objets quelles contient.

Le rafraichissement, leur position par rapport à la scène et l'envoie des évènements destinés aux objets y sont gérés.

Définition dans le fichier gamescene.h.

6.6 Référence du fichier gameview.cpp

```
#include "gameview.h"
```

6.6.1 Description détaillée

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Définition dans le fichier gameview.cpp.

6.7 Référence du fichier gameview.h

GameView s'occupe de la vue d'affichage d'une scène.

```
#include <QGraphicsView>
```

Classes

– class GameView
La classe GameView hérite de QGraphicsView.
GameView s'occupe de la vue d'affichage d'une scène.

Macros

– #define DEFAULT_WIDTH 1280
Correspond à la largeur par défaut que prendra la fenêtre.
– #define DEFAULT_HEIGHT 720
Correspond à la hauteur par défaut que prendra la fenêtre.
– #define DEFAULT_ZOOM 2
Correspond au zoom par défaut qui sera appliquer à la vue d'affichage.

6.7.1 Description détaillée

GameView s'occupe de la vue d'affichage d'une scène.

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Elle a pour but de positionner/configurer l'affichage d'une scène.

Définition dans le fichier gameview.h.

6.7.2 Documentation des macros

6.7.2.1 #define DEFAULT_HEIGHT 720

Correspond à la hauteur par défaut que prendra la fenêtre.

Initialiser a 720

Définition à la ligne 23 du fichier gameview.h.

6.7.2.2 #define DEFAULT_WIDTH 1280

Correspond à la largeur par défaut que prendra la fenêtre.

Initialiser a 1280

Définition à la ligne 18 du fichier gameview.h.

6.7.2.3 #define DEFAULT_ZOOM 2

Correspond au zoom par défaut qui sera appliquer à la vue d'affichage.

Initialiser a 2

Définition à la ligne 28 du fichier gameview.h.

6.8 Référence du fichier gamewindow.cpp

```
#include "gamescene.h"
#include "gameview.h"
#include "gamewindow.h"
```

6.8.1 Description détaillée

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Définition dans le fichier gamewindow.cpp.

6.9 Référence du fichier gamewindow.h

GameWindow s'occupe de l'affichage.

```
#include <QTimer>
#include <QMainWindow>
```

Classes

– class GameWindow
La classe GameWindow hérite de QMainWindow.
GameWindow s'occupe de l'affichage.

6.9.1 Description détaillée

GameWindow s'occupe de l'affichage.

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Elle a pour but de connecter une scène avec une view, pour afficher les éléments dans la fenêtre.

Définition dans le fichier gamewindow.h.

6.10 Référence du fichier main.cpp

Le main gére le lancement du programme. C'est ici que l'application lancera l'interface graphique.

```
#include <QApplication>
#include "gamewindow.h"
```

Fonctions

– int main (int argc, char ∗argv[])
Entrée du programme.

6.10.1 Description détaillée

Le main gére le lancement du programme. C'est ici que l'application lancera l'interface graphique.

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Définition dans le fichier main.cpp.

6.10.2 Documentation des fonctions

6.10.2.1 int main (int *argc*, char * *argv[]*)

Entrée du programme.

Paramètres

argc	Correspond au nombre d'arguments.
argv	Correspond aux arguments donnés.

Renvoie

Une boucle infini sur a.exec().
a.exec() correspond est l'application graphique qui s'occupera de lancer notre programme.

Définition à la ligne 33 du fichier main.cpp.

6.11 Référence du fichier player.cpp

```
#include <QGraphicsScene>
#include <QPainter>
#include <QKeyEvent>
#include "player.h"
#include "directionmove.h"
#include "spriteimgmove.h"
```

6.11.1 Description détaillée

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Définition dans le fichier player.cpp.

6.12 Référence du fichier player.h

Contient la définition d'un Player.

```
#include "entity.h"
```

Classes

– class Player
La classe Player hérite de Entity.
Contient la définition d'un player.

Macros

– #define DEFAULT_P_WIDTH 32
Correspond à la largeur par défaut que prendra le player.
– #define DEFAULT_P_HEIGHT 32
Correspond à la hauteur par défaut que prendra le player.
– #define DEFAULT_ANIME_TIME 15
Correspond à la vitesse que prendra une animation de déplacement dans le temps.
– #define DEFAULT_ANIME_FRAME 2
Correspond au frame d'animation du player.

6.12.1 Description détaillée

Contient la définition d'un Player.

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Un player est une Entity qui se déplace dans une scène grace aux évènements clavier.

Définition dans le fichier player.h.

6.12.2 Documentation des macros

6.12.2.1 #define DEFAULT_ANIME_FRAME 2

Correspond au frame d'animation du player.

Initialiser à 2.

Définition à la ligne 34 du fichier player.h.

6.12.2.2 #define DEFAULT_ANIME_TIME 15

Correspond à la vitesse que prendra une animation de déplacement dans le temps.

Initialiser à 15.

Définition à la ligne 29 du fichier player.h.

6.12.2.3 #define DEFAULT_P_HEIGHT 32

Correspond à la hauteur par défaut que prendra le player.

Initialiser à 32.

Définition à la ligne 24 du fichier player.h.

6.12.2.4 #define DEFAULT_P_WIDTH 32

Correspond à la largeur par défaut que prendra le player.

Initialiser à 32.

Définition à la ligne 19 du fichier player.h.

6.13 Référence du fichier spriteimgmove.cpp

```
#include "spriteimgmove.h"
#include "directionmove.h"
```

6.13.1 Description détaillée

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

Définition dans le fichier spriteimgmove.cpp.

6.14 Référence du fichier spriteimgmove.h

Contient la déscription des animations.

```
#include <QPainter>
#include <QPixmap>
```

Classes

– class SpriteImgMove
La classe SpriteImgMove.

6.14.1 Description détaillée

Contient la déscription des animations.

Auteur

Guillaume Rasolo

Date

08/11/2013

Version

1.0

La classe SpriteImgMove permet de gérer les animations de déplacement.

Définition dans le fichier spriteimgmove.h.

Index