

Space Sheep Xtreme Turbo Penguin Edition

Généré par Doxygen 1.8.4

Dimanche Juin 2 2013 19 :39 :59

Table des matières

1	Liste des bogues	1
2	Index hiérarchique	3
2.1	Hiérarchie des classes	3
3	Index des classes	5
3.1	Liste des classes	5
4	Index des fichiers	7
4.1	Liste des fichiers	7
5	Documentation des classes	9
5.1	Référence de la classe Bonus	9
5.1.1	Description détaillée	10
5.1.2	Documentation des constructeurs et destructeur	10
5.1.2.1	Bonus	10
5.1.3	Documentation des fonctions membres	10
5.1.3.1	getType	10
5.2	Référence de la structure BonusEffect	10
5.2.1	Description détaillée	10
5.3	Référence de la classe Boss	11
5.3.1	Description détaillée	12
5.3.2	Documentation des constructeurs et destructeur	12
5.3.2.1	Boss	12
5.3.3	Documentation des fonctions membres	12
5.3.3.1	changePattern	12
5.3.3.2	getShootPattern	13
5.3.3.3	move	13
5.4	Référence de la classe Ennemi	13
5.4.1	Description détaillée	15
5.4.2	Documentation des constructeurs et destructeur	15
5.4.2.1	Ennemi	15
5.4.3	Documentation des fonctions membres	15

5.4.3.1	getScore	15
5.4.3.2	getShootPattern	16
5.4.3.3	getType	16
5.4.3.4	isShot	16
5.4.3.5	shoot	17
5.5	Référence de la classe GameModel	18
5.5.1	Description détaillée	19
5.5.2	Documentation des constructeurs et destructeur	19
5.5.2.1	GameModel	19
5.5.3	Documentation des fonctions membres	19
5.5.3.1	addBonus	19
5.5.3.2	addEnemy	19
5.5.3.3	addTir	19
5.5.3.4	changeNiveau	20
5.5.3.5	gameOver	20
5.5.3.6	getBossType	21
5.5.3.7	getClock	21
5.5.3.8	getDeletedItems	22
5.5.3.9	getDifficulty	22
5.5.3.10	getEnemyTypes	22
5.5.3.11	getLevelID	22
5.5.3.12	getNextLevel	22
5.5.3.13	getPlayerShip	23
5.5.3.14	getScore	23
5.5.3.15	getShipPos	23
5.5.3.16	getStringLevelID	23
5.5.3.17	getStringScore	24
5.5.3.18	getTirTypes	24
5.5.3.19	getVectorBonuses	24
5.5.3.20	getVectorEnemies	24
5.5.3.21	getVectorTir	25
5.5.3.22	nextStep	25
5.5.3.23	removeEnemy	25
5.5.3.24	removeEnemy	26
5.5.3.25	saveScore	26
5.5.3.26	setNextLevel	26
5.5.3.27	setScore	26
5.6	Référence de la classe GameView	26
5.6.1	Description détaillée	27
5.6.2	Documentation des constructeurs et destructeur	27

5.6.2.1	GameView	27
5.6.3	Documentation des fonctions membres	28
5.6.3.1	explosion	28
5.6.3.2	options	28
5.6.3.3	setModel	28
5.6.3.4	synchronize	29
5.6.3.5	transition	30
5.6.3.6	treatEvents	31
5.6.3.7	viewHighscores	32
5.7	Référence de la classe GraphicElement	32
5.7.1	Description détaillée	33
5.7.2	Documentation des constructeurs et destructeur	34
5.7.2.1	GraphicElement	34
5.7.3	Documentation des fonctions membres	34
5.7.3.1	draw	34
5.7.3.2	getVisible	34
5.7.3.3	resize	34
5.7.3.4	setPosition	34
5.7.3.5	setPosition	35
5.7.3.6	setVisible	35
5.8	Référence de la classe Horloge	35
5.8.1	Description détaillée	35
5.8.2	Documentation des fonctions membres	36
5.8.2.1	getStatus	36
5.8.2.2	getTime	36
5.8.2.3	stop	36
5.9	Référence de la classe Interface	36
5.9.1	Description détaillée	37
5.9.2	Documentation des fonctions membres	37
5.9.2.1	drawHUD	37
5.9.2.2	update	37
5.9.2.3	updateBossPV	38
5.10	Référence de la classe Joueur	39
5.10.1	Description détaillée	40
5.10.2	Documentation des constructeurs et destructeur	40
5.10.2.1	Joueur	40
5.10.3	Documentation des fonctions membres	41
5.10.3.1	addHP	41
5.10.3.2	addPower	41
5.10.3.3	getBombs	42

5.10.3.4	getNumberOfShoots	42
5.10.3.5	getPower	42
5.10.3.6	getStringBombs	42
5.10.3.7	getStringPower	42
5.10.3.8	getVies	42
5.10.3.9	isHitByEnemy	43
5.10.3.10	isShot	44
5.10.3.11	shoot	44
5.10.3.12	useBomb	45
5.11	Référence de la classe MovableElement	46
5.11.1	Description détaillée	47
5.11.2	Documentation des constructeurs et destructeur	47
5.11.2.1	MovableElement	47
5.11.3	Documentation des fonctions membres	47
5.11.3.1	getDY	47
5.11.3.2	getH	47
5.11.3.3	getW	48
5.11.3.4	getX	48
5.11.3.5	getY	48
5.11.3.6	move	48
5.11.4	Documentation des données membres	48
5.11.4.1	my_dx	48
5.11.4.2	my_dy	48
5.11.4.3	my_h	48
5.11.4.4	my_w	49
5.11.4.5	my_x	49
5.11.4.6	my_y	49
5.12	Référence de la classe Niveau	49
5.12.1	Description détaillée	49
5.12.2	Documentation des fonctions membres	49
5.12.2.1	getID	49
5.12.2.2	getMap	50
5.12.2.3	initNiveau	50
5.13	Référence de la structure ShootPattern	50
5.13.1	Description détaillée	50
5.14	Référence de la classe Tir	50
5.14.1	Description détaillée	51
5.14.2	Documentation des constructeurs et destructeur	52
5.14.2.1	Tir	52
5.14.2.2	Tir	52

5.14.3	Documentation des fonctions membres	52
5.14.3.1	getPlayer	52
5.14.3.2	getType	52
5.15	Référence de la classe TypeBonus	52
5.15.1	Description détaillée	53
5.15.2	Documentation des constructeurs et destructeur	53
5.15.2.1	TypeBonus	53
5.15.3	Documentation des fonctions membres	53
5.15.3.1	getEffect	53
5.15.3.2	getFrequency	53
5.15.3.3	getScale	54
5.15.3.4	getValue	54
5.16	Référence de la classe TypeBoss	54
5.16.1	Description détaillée	55
5.16.2	Documentation des constructeurs et destructeur	55
5.16.2.1	TypeBoss	55
5.16.3	Documentation des fonctions membres	55
5.16.3.1	getChangePatternDelay	56
5.17	Référence de la classe TypeEnnemi	56
5.17.1	Description détaillée	57
5.17.2	Documentation des constructeurs et destructeur	57
5.17.2.1	TypeEnnemi	57
5.17.3	Documentation des fonctions membres	57
5.17.3.1	getFilename	57
5.17.3.2	getH	57
5.17.3.3	getHP	57
5.17.3.4	getScore	58
5.17.3.5	getShootingSpeed	58
5.17.3.6	getShootPattern	58
5.17.3.7	getSpeed	58
5.17.3.8	getTypeTir	58
5.17.3.9	getW	59
5.18	Référence de la classe TypeTir	59
5.18.1	Description détaillée	59
5.18.2	Documentation des constructeurs et destructeur	59
5.18.2.1	TypeTir	59
5.18.3	Documentation des fonctions membres	60
5.18.3.1	getDamages	60
5.18.3.2	getFilename	60
5.18.3.3	getH	60

5.18.3.4	getSpeed	60
5.18.3.5	getW	60
5.19	Référence de la classe Vaisseau	61
5.19.1	Description détaillée	62
5.19.2	Documentation des constructeurs et destructeur	62
5.19.2.1	Vaisseau	62
5.19.3	Documentation des fonctions membres	62
5.19.3.1	getAlive	62
5.19.3.2	getClock	63
5.19.3.3	getHP	63
5.19.3.4	getMaxHP	63
5.19.3.5	getShootingSpeed	63
5.19.3.6	getTypeTir	63
5.19.3.7	isShot	63
5.19.3.8	setAlive	64
5.19.3.9	setHP	64
5.19.3.10	setTypeTir	64
5.19.3.11	shoot	64
5.19.4	Documentation des données membres	64
5.19.4.1	my_clock	64
5.19.4.2	my_shootingSpeed	64
5.19.4.3	my_typeTir	65
6	Documentation des fichiers	67
6.1	Référence du fichier Bonus.h	67
6.1.1	Description détaillée	68
6.2	Référence du fichier Boss.h	68
6.2.1	Description détaillée	70
6.3	Référence du fichier constantes.h	70
6.3.1	Description détaillée	71
6.3.2	Documentation des macros	71
6.3.2.1	BACKTOMENU	71
6.3.2.2	BOMBES	71
6.3.2.3	DEAD	71
6.3.2.4	DIFFICULTY	71
6.3.2.5	ENEMY_SCALE	71
6.3.2.6	ENTERNAME	72
6.3.2.7	FINISHLEVEL	72
6.3.2.8	GAMEOVER	72
6.3.2.9	HARD	72

6.3.2.10	HIGHSCORES	72
6.3.2.11	JOUER	72
6.3.2.12	LANGUAGE	72
6.3.2.13	LIVESNUMBER	72
6.3.2.14	MODEL_HEIGHT	72
6.3.2.15	MODEL_WIDTH	72
6.3.2.16	MOVABLE_ELEMENT_SPEED	72
6.3.2.17	MYLANGUE	72
6.3.2.18	NB_BONUS_TYPES	73
6.3.2.19	NB_ENNEMY_TYPES	73
6.3.2.20	NB_STRINGS	73
6.3.2.21	NB_TIRS_TYPES	73
6.3.2.22	NIVEAU	73
6.3.2.23	NORMAL	73
6.3.2.24	NOSCORE	73
6.3.2.25	PAUSE	73
6.3.2.26	PUISSANCE	73
6.3.2.27	PV	73
6.3.2.28	QUIT	73
6.3.2.29	RESUME	73
6.3.2.30	SETTINGS	74
6.3.2.31	SHEEP_HEIGHT	74
6.3.2.32	SHEEP_SCALE	74
6.3.2.33	SHEEP_SPEED	74
6.3.2.34	SHEEP_WIDTH	74
6.3.2.35	SOUNDLEVEL	74
6.3.2.36	SOUNDPACK	74
6.3.2.37	VERYHARD	74
6.3.2.38	VIES	74
6.3.2.39	VIEW_HEIGHT	74
6.3.2.40	VIEW_WIDTH	74
6.4	Référence du fichier Ennemi.h	74
6.4.1	Description détaillée	76
6.5	Référence du fichier GameModel.h	76
6.5.1	Description détaillée	77
6.6	Référence du fichier GameView.h	77
6.6.1	Description détaillée	78
6.7	Référence du fichier GraphicElement.h	78
6.7.1	Description détaillée	79
6.8	Référence du fichier Horloge.h	79

6.8.1	Description détaillée	80
6.9	Référence du fichier Interface.h	80
6.9.1	Description détaillée	81
6.10	Référence du fichier Joueur.h	82
6.10.1	Description détaillée	83
6.11	Référence du fichier MovableElement.h	83
6.11.1	Description détaillée	84
6.12	Référence du fichier Niveau.h	85
6.12.1	Description détaillée	86
6.13	Référence du fichier Tir.h	86
6.13.1	Description détaillée	87
6.14	Référence du fichier TypeBonus.h	87
6.14.1	Description détaillée	89
6.14.2	Documentation du type de l'énumération	89
6.14.2.1	BonusEffect	89
6.15	Référence du fichier TypeBoss.h	89
6.15.1	Description détaillée	90
6.16	Référence du fichier TypeEnnemi.h	91
6.16.1	Description détaillée	92
6.16.2	Documentation du type de l'énumération	92
6.16.2.1	ShootPattern	93
6.17	Référence du fichier TypeTir.h	93
6.17.1	Description détaillée	94
6.18	Référence du fichier Vaisseau.h	95
6.18.1	Description détaillée	96

Index

97

Chapitre 1

Liste des bogues

Membre **GameView** : **:synchronize** ()

Une technique plus optimisée a été trouvée pour cette méthode, mais elle entraîne parfois des erreurs de segmentation, pour une raison inconnue

Membre **GameView** : **:viewHighscores** (bool &quit)

Pour une raison inconnue, un Event : :Closed se produit à l'ouverture du menu. Pour pallier à ça, il a fallu empêcher de fermer le programme "normalement" une fois sur cet écran (l'utilisation de la touche Escape est toujours possible cependant)

Chapitre 2

Index hiérarchique

2.1 Hiérarchie des classes

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

BonusEffect	10
GameModel	18
GameView	26
Horloge	35
Interface	36
MovableElement	46
Bonus	9
Tir	50
Vaisseau	61
Ennemi	13
Boss	11
Joueur	39
Niveau	49
ShootPattern	50
Sprite	
GraphicElement	32
TypeBonus	52
TypeEnnemi	56
TypeBoss	54
TypeTir	59

Chapitre 3

Index des classes

3.1 Liste des classes

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

Bonus	Classe représentant les bonii à ramasser	9
BonusEffect	Catégories de bonus	10
Boss	Classe représentant les boss de fin de niveau	11
Ennemi	Classe représentant les ennemis du joueur	13
GameModel	Classe représentant le Modèle du jeu	18
GameView	Classe représentant la Vue du jeu	26
GraphicElement	Classe représentant les éléments à afficher	32
Horloge	Classe représentant le temps dans le jeu	35
Interface	Classe représentant le HUD (affichage tête haute)	36
Joueur	Classe représentant le vaisseau contrôlé	39
MovableElement	Classe représentant les éléments mobiles	46
Niveau	Classe représentant les niveaux du jeu	49
ShootPattern	Catégories de patterns	50
Tir	Classe représentant les tirs des Vaisseau	50
TypeBonus	Classe représentant les différents types de Bonus	52
TypeBoss	Classe représentant les différents types de Boss	54
TypeEnnemi	Classe représentant les différents types d' Ennemi	56
TypeTir	Classe représentant les différents types de Tir	59
Vaisseau	Classe représentant les vaisseaux	61

Chapitre 4

Index des fichiers

4.1 Liste des fichiers

Liste de tous les fichiers documentés avec une brève description :

Bonus.h	Déclaration de la class Bonus	67
Boss.h	Déclaration de la class Boss	68
constantes.h	Déclaration des constantes utilisées	70
Ennemi.h	Déclaration de la class Ennemi	74
GameModel.h	Déclaration de la classe GameModel	76
GameView.h	Déclaration de la classe GameView	77
GraphicElement.h	Déclaration de la classe GraphicElement	78
Horloge.h	Déclaration de la classe Horloge	79
Interface.h	Déclaration de la classe Interface	80
Joueur.h	Déclaration de la classe Joueur	82
MovableElement.h	Déclaration de la classe MovableElement	83
Niveau.h	Déclaration de la classe Niveau	85
Tir.h	Déclaration de la classe Tir	86
TypeBonus.h	Déclaration de la classe TypeBonus et de l'énumération BonusEffect	87
TypeBoss.h	Déclaration de la classe TypeBoss	89
TypeEnnemi.h	Déclaration de la classe TypeEnnemi et de l'énumération ShootPattern	91
TypeTir.h	Déclaration de la classe TypeTir	93
Vaisseau.h	Déclaration de la classe Vaisseau	95

Chapitre 5

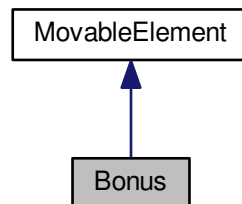
Documentation des classes

5.1 Référence de la classe Bonus

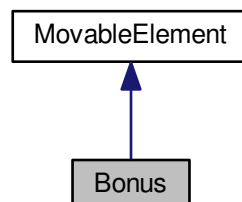
Classe représentant les bonii à ramasser.

```
#include <Bonus.h>
```

Graphe d'héritage de Bonus :



Graphe de collaboration de Bonus :



Fonctions membres publiques

- [Bonus](#) (int x, int y, [TypeBonus](#) *type)
Constructeur paramétré
- [TypeBonus](#) * [getType](#) () const
Accesseur.

Additional Inherited Members

5.1.1 Description détaillée

Classe représentant les bonii à ramasser.

Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

5.1.2 Documentation des constructeurs et destructeur

5.1.2.1 [Bonus](#) : :[Bonus](#) (int x, int y, [TypeBonus](#) * type)

Constructeur paramétré

Paramètres

<i>x</i>	: Abscisse du bonus
<i>y</i>	: Ordonnée du bonus
<i>type</i>	: Type du bonus

5.1.3 Documentation des fonctions membres

5.1.3.1 [TypeBonus](#) * [Bonus](#) : :[getType](#) () const

Accesseur.

Accesseur de my_type

Renvoie

: Le type du [Bonus](#)

La documentation de cette classe a été générée à partir des fichiers suivants :

- [Bonus.h](#)
- [Bonus.cc](#)

5.2 Référence de la structure BonusEffect

Catégories de bonus.

```
#include <TypeBonus.h>
```

5.2.1 Description détaillée

Catégories de bonus.

La documentation de cette structure a été générée à partir du fichier suivant :

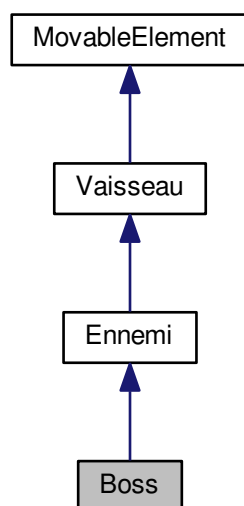
- [TypeBonus.h](#)

5.3 Référence de la classe Boss

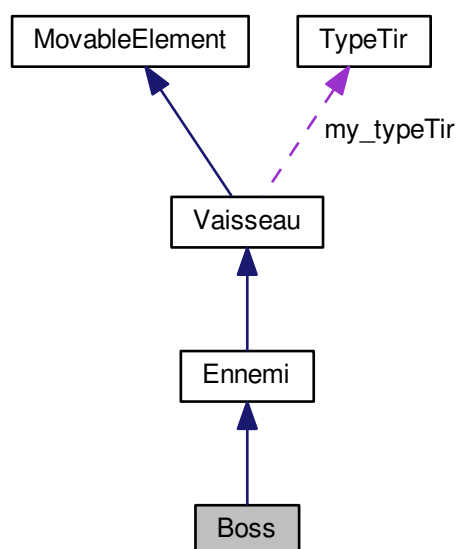
Classe représentant les boss de fin de niveau.

```
#include <Boss.h>
```

Graphe d'héritage de Boss :



Graphe de collaboration de Boss :



Fonctions membres publiques

- `Boss (TypeBoss *type)`
Constructeur paramétré
- `ShootPattern getShootPattern () const`
Accesseur.
- `void changePattern ()`
Changer le pattern.
- `void move ()`
Déplacer le boss.

Additional Inherited Members

5.3.1 Description détaillée

Classe représentant les boss de fin de niveau.

Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

5.3.2 Documentation des constructeurs et destructeur

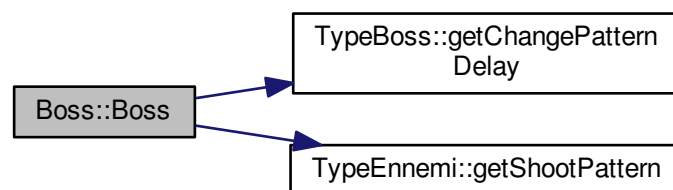
5.3.2.1 `Boss : :Boss (TypeBoss * type)`

Constructeur paramétré

Paramètres

<i>type</i>	: Type du boss
-------------	----------------

Voici le graphe d'appel pour cette fonction :



5.3.3 Documentation des fonctions membres

5.3.3.1 `void Boss : :changePattern ()`

Changer le pattern.

Méthode permettant de changer le pattern actuellement utilisé

5.3.3.2 ShootPattern Boss : :getShootPattern () const [virtual]

Accesseur.

Accesseur de my_actualShootPattern

Renvoie

: Le pattern actuellement utilisé

Réimplémentée à partir de [Ennemi](#).

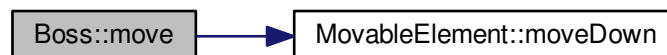
5.3.3.3 void Boss : :move () [virtual]

Déplacer le boss.

Méthode permettant de déplacer le [Boss](#)

Réimplémentée à partir de [MovableElement](#).

Voici le graphe d'appel pour cette fonction :



La documentation de cette classe a été générée à partir des fichiers suivants :

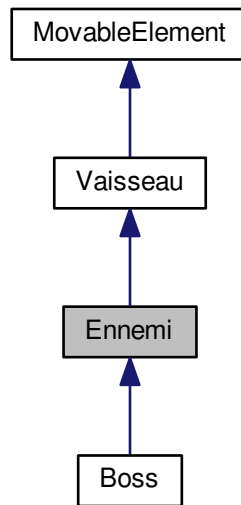
- [Boss.h](#)
- [Boss.cc](#)

5.4 Référence de la classe Ennemi

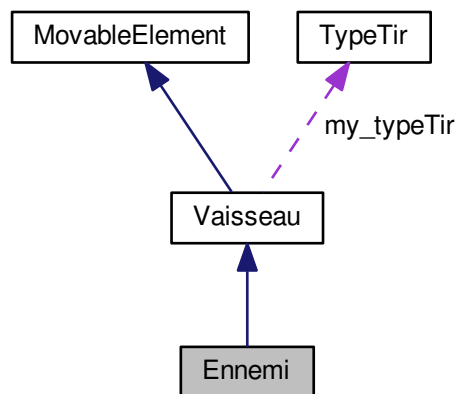
Classe représentant les ennemis du joueur.

```
#include <Ennemi.h>
```

Graphe d'héritage de Ennemi :



Graphe de collaboration de Ennemi :



Fonctions membres publiques

- [Ennemi](#) ()
Constructeur par défaut.
- [Ennemi](#) ([TypeEnnemi](#) *type)
Constructeur paramétré
- virtual [~Ennemi](#) ()
Destructeur.
- int [getScore](#) () const

- *Accesseur.*
– `TypeEnnemi * getType () const`
- *Accesseur.*
– `virtual ShootPattern getShootPattern () const`
- *Accesseur.*
– `void shoot (GameModel *model)`
- *Faire tirer l'ennemi.*
– `std::vector< Tir * >::iterator isShot (GameModel *model) const`
- *Indiquer le tir touchant l'ennemi.*
– `void move ()`
– *Déplacer l'ennemi.*

Additional Inherited Members

5.4.1 Description détaillée

Classe représentant les ennemis du joueur.

Auteur

Théo CHASSAIGNE
 Quentin HARSCOËT

5.4.2 Documentation des constructeurs et destructeur

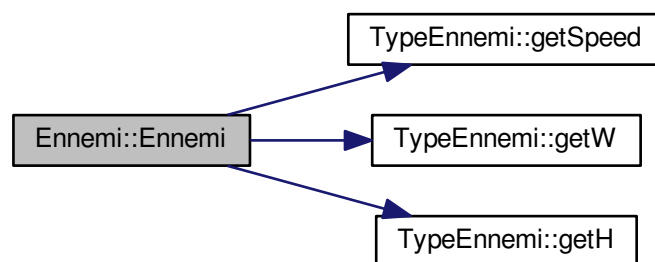
5.4.2.1 Ennemi::Ennemi (TypeEnnemi * type)

Constructeur paramétré

Paramètres

<i>type</i>	: Type d'ennemi
-------------	-----------------

Voici le graphe d'appel pour cette fonction :



5.4.3 Documentation des fonctions membres

5.4.3.1 int Ennemi::getScore () const

Accesseur.

Accesseur de `my_type.my_score`

Renvoie

: Les points rapportés par l'ennemi

Voici le graphe d'appel pour cette fonction :

**5.4.3.2 ShootPattern Ennemi : :getShootPattern () const [virtual]**

Accesseur.

Accesseur de `my_type.my_shootPattern`

Renvoie

: Le pattern utilisé par l'ennemi

Réimplémentée dans [Boss](#).

Voici le graphe d'appel pour cette fonction :

**5.4.3.3 TypeEnnemi * Ennemi : :getType () const**

Accesseur.

Accesseur de `my_type`

Renvoie

: Le type d'ennemi

5.4.3.4 vector< Tir * > : :iterator Ennemi : :isShot (GameModel * model) const [virtual]

Indiquer le tir touchant l'ennemi.

Paramètres

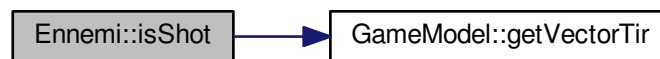
<i>model</i>	: Modèle du jeu
--------------	-----------------

Renvoie

: Le tir touchant l'ennemi

Implémente [Vaisseau](#).

Voici le graphe d'appel pour cette fonction :



5.4.3.5 void Ennemi::shoot (GameModel * model) [virtual]

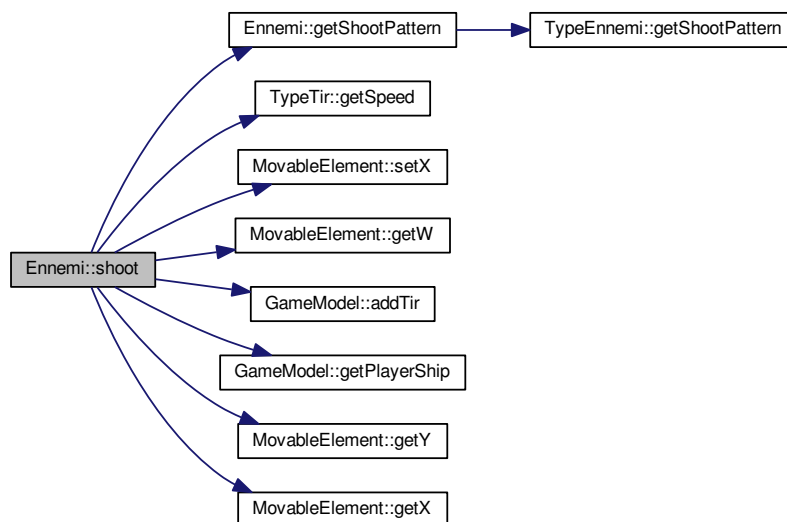
Faire tirer l'ennemi.

Paramètres

<i>model</i>	: Modèle du jeu
--------------	-----------------

Implémente [Vaisseau](#).

Voici le graphe d'appel pour cette fonction :



La documentation de cette classe a été générée à partir des fichiers suivants :

- [Ennemi.h](#)
- [Ennemi.cc](#)

5.5 Référence de la classe GameModel

Classe représentant le Modèle du jeu.

```
#include <GameModel.h>
```

Fonctions membres publiques

- [GameModel](#) (int viesMax, int initialLives, int difficulty)
Constructeur paramétré
- [~GameModel](#) ()
Destructeur.
- void [nextStep](#) ()
Passer à "l'étape" suivante.
- void [addEnemy](#) ([TypeEnemy](#) *type)
Ajouter un [Ennemi](#).
- void [addTir](#) ([Tir](#) *t)
Ajouter un [Tir](#).
- void [addBonus](#) (int x, int y)
Ajouter un [Bonus](#).
- void [removeEnemy](#) ([Ennemi](#) *ennemi)
Supprimer un [Ennemi](#).
- void [removeEnemy](#) (std::vector< [Ennemi](#) * >::iterator it)
Supprimer un [Ennemi](#).
- void [saveScore](#) (string name)
Sauvegarder le score.
- void [changeNiveau](#) ()
Passer au [Niveau](#) suivant.
- void [gameOver](#) ()
Tuer le [Joueur](#).
- void [getShipPos](#) (int &x, int &y) const
Accesseur.
- bool [getNextLevel](#) () const
Accesseur.
- int [getLevelID](#) () const
Accesseur.
- int [getScore](#) () const
Accesseur.
- int [getDifficulty](#) () const
Accesseur.
- string [getStringLevelID](#) () const
Accesseur.
- string [getStringScore](#) () const
Accesseur.
- [TypeEnemy](#) *const * [getEnemyTypes](#) () const
Accesseur.
- [TypeTir](#) *const * [getTirTypes](#) () const
Accesseur.
- [TypeBoss](#) * [getBossType](#) () const
Accesseur.
- [Joueur](#) * [getPlayerShip](#) () const
Accesseur.
- vector< [Ennemi](#) * > * [getVectorEnnemis](#) ()
Accesseur.
- vector< [Tir](#) * > * [getVectorTir](#) ()
Accesseur.
- const vector< [Bonus](#) * > & [getVectorBonuses](#) ()
Accesseur.
- vector< [MovableElement](#) * > * [getDeletedItems](#) ()
Accesseur.
- [Horloge](#) * [getClock](#) ()
Accesseur.
- void [setScore](#) (int score)
Définir le score.
- void [setNextLevel](#) (bool nextLvl)
Définir si l'on passe au niveau suivant.

5.5.1 Description détaillée

Classe représentant le Modèle du jeu.

Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

5.5.2 Documentation des constructeurs et destructeur

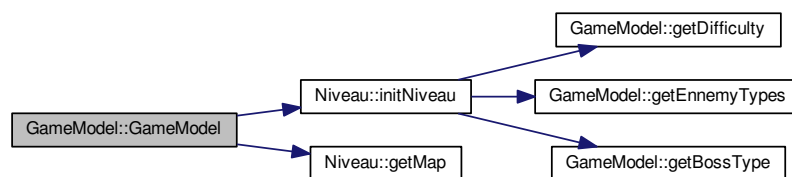
5.5.2.1 GameModel : :GameModel (int viesMax, int initialLives, int difficulty)

Constructeur paramétré

Paramètres

<i>viesMax</i>	: Nombre maximum de vies du Joueur
<i>initialLives</i>	: Nombre initial de vies du Joueur
<i>difficulty</i>	: Difficulté du jeu

Voici le graphe d'appel pour cette fonction :



5.5.3 Documentation des fonctions membres

5.5.3.1 void GameModel : :addBonus (int x, int y)

Ajouter un [Bonus](#).

Paramètres

<i>x</i>	: Abscisse du Bonus à ajouter
<i>y</i>	: Ordonnée du Bonus à ajouter

5.5.3.2 void GameModel : :addEnemy (TypeEnnemi * type)

Ajouter un [Ennemi](#).

Paramètres

<i>type</i>	: Type de l' Ennemi à ajouter
-------------	---

5.5.3.3 void GameModel : :addTir (Tir * t)

Ajouter un [Tir](#).

Paramètres

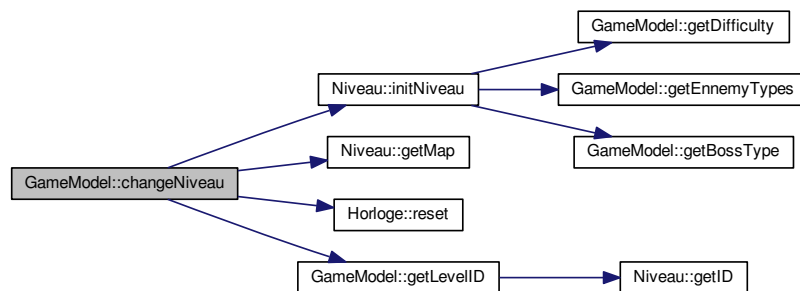
<i>t</i>	: Tir à ajouter
----------	---------------------------------

5.5.3.4 void GameModel : :changeNiveau ()

Passer au [Niveau](#) suivant.

Vide le vector des [Tir](#), réinitialise la Clock et redéfinit le [Boss](#)

Voici le graphe d'appel pour cette fonction :

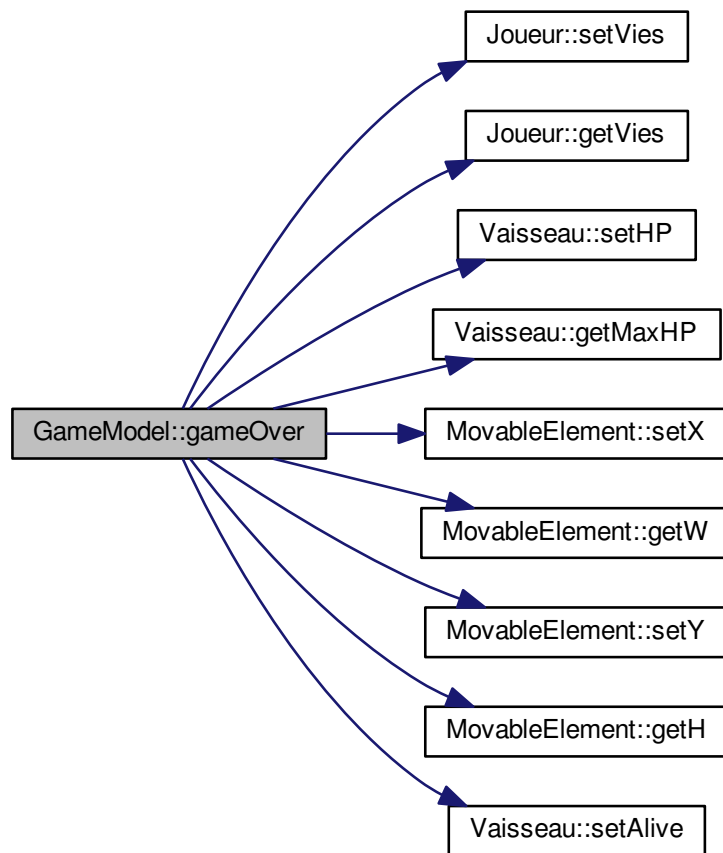


5.5.3.5 void GameModel : :gameOver ()

Tuer le [Joueur](#).

Méthode permettant de gérer la mort du [Joueur](#)

Voici le graphe d'appel pour cette fonction :



5.5.3.6 TypeBoss * GameModel : :getBossType () const

Accesseur.

Accesseur de `my_boss_types`

Renvoie

: Le tableau contenant les différents [TypeBoss](#)

5.5.3.7 Horloge * GameModel : :getClock ()

Accesseur.

Accesseur de `my_clock`

Renvoie

: L'[Horloge](#) du jeu

5.5.3.8 `vector< MovableElement * > * GameModel : :getDeletedItems ()`

Accesseur.

Accesseur de `my_tirs`

Renvoie

: Le vecteur contenant les objets à supprimer

5.5.3.9 `int GameModel : :getDifficulty () const`

Accesseur.

Accesseur de `my_difficulty`

Renvoie

: La difficulté du jeu

5.5.3.10 `TypeEnnemi *const * GameModel : :getEnemyTypes () const`

Accesseur.

Accesseur de `my_ennemi_types`

Renvoie

: Le tableau contenant les différents [TypeEnnemi](#)

5.5.3.11 `int GameModel : :getLevelID () const`

Accesseur.

Accesseur de `niveau.my_id`

Renvoie

: L'identifiant du [Niveau](#) en cours

Voici le graphe d'appel pour cette fonction :



5.5.3.12 `bool GameModel : :getNextLevel () const`

Accesseur.

Accesseur de `my_nextLevel`

Renvoie

: Si l'on passe ou non au niveau suivant (**true** si oui, **sinon**)

5.5.3.13 `Joueur * GameModel : :getPlayerShip () const`

Accesseur.

Accesseur de my_ship

Renvoie

: Le `Joueur`

5.5.3.14 `int GameModel : :getScore () const`

Accesseur.

Accesseur de my_score

Renvoie

: Le score actuel

5.5.3.15 `void GameModel : :getShipPos (int & x, int & y) const`

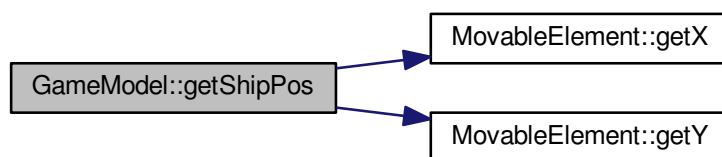
Accesseur.

Méthode récupérant la position du `Joueur`

Paramètres

<code>x</code>	: (Sortie seulement) Abscisse du <code>Joueur</code>
<code>y</code>	: (Sortie seulement) Ordonnée du <code>Joueur</code>

Voici le graphe d'appel pour cette fonction :



5.5.3.16 `string GameModel : :getStringLevelID () const`

Accesseur.

Accesseur de niveau.my_id

Renvoie

: L'identifiant du [Niveau](#) en cours, sous forme de chaîne de caractères

Voici le graphe d'appel pour cette fonction :

**5.5.3.17 string GameModel : :getStringScore () const**

Accesseur.

Accesseur de my_score

Renvoie

: Le score actuel, sous forme de chaîne de caractères

5.5.3.18 TypeTir *const * GameModel : :getTirTypes () const

Accesseur.

Accesseur de my_tirs_types

Renvoie

: Le tableau contenant les différents [TypeTir](#)

5.5.3.19 const vector< Bonus * > & GameModel : :getVectorBonuses ()

Accesseur.

Accesseur de my_bonuses

Renvoie

: Le vecteur contenant les [Bonus](#) présents

5.5.3.20 vector< Ennemi * > * GameModel : :getVectorEnnemis ()

Accesseur.

Accesseur de my_ennemis

Renvoie

: Le vecteur contenant les [Ennemi](#) présents

5.5.3.21 `vector< Tir * > * GameModel : :getVectorTir ()`

Accesseur.

Accesseur de my_tirs

Renvoie

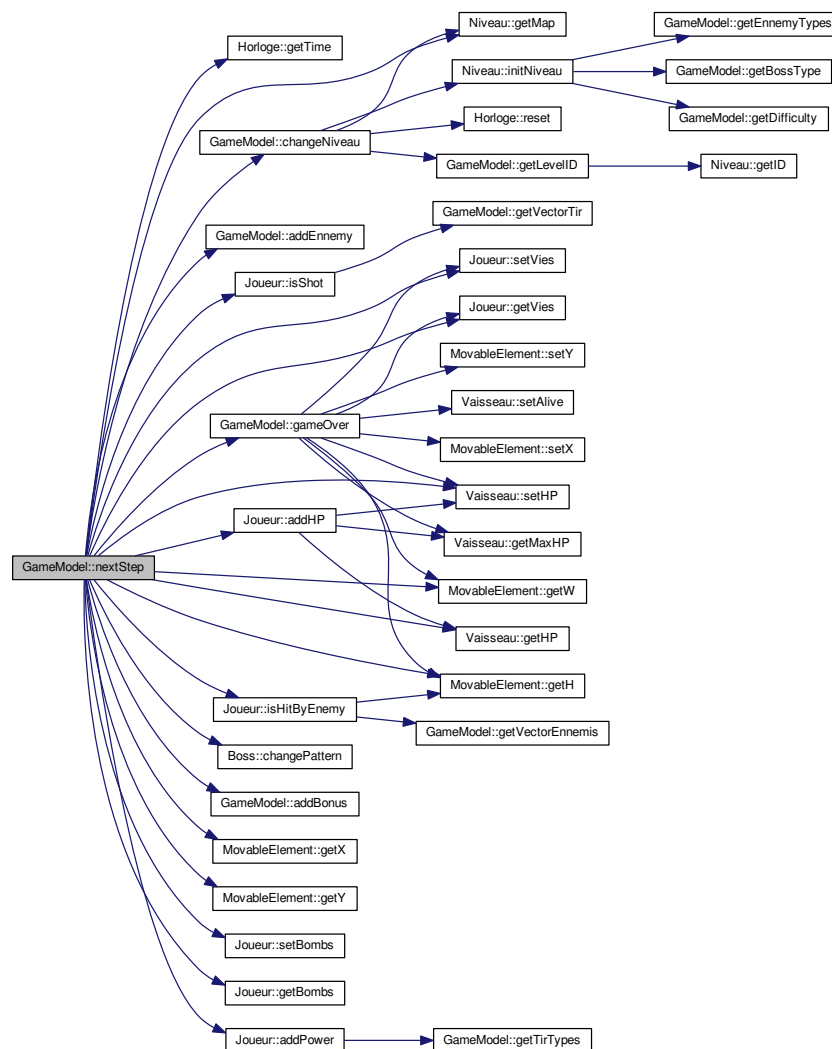
: Le vecteur contenant les [Tir](#) présents

5.5.3.22 `void GameModel : :nextStep ()`

Passer à "l'étape" suivante.

Méthode permettant de faire progresser le jeu

Voici le graphe d'appel pour cette fonction :

5.5.3.23 `void GameModel : :removeEnemy (Ennemi * ennemi)`

Supprimer un [Ennemi](#).

Paramètres

<i>ennemi</i>	: Adresse de l' Ennemi à supprimer
---------------	--

5.5.3.24 void GameModel : :removeEnemy (std : :vector< [Ennemi](#) * > : :iterator *it*)

Supprimer un [Ennemi](#).

Paramètres

<i>it</i>	: Itérateur vers l' Ennemi à supprimer
-----------	--

5.5.3.25 void GameModel : :saveScore (string *name*)

Sauvegarder le score.

Paramètres

<i>name</i>	: Pseudo à afficher
-------------	---------------------

5.5.3.26 void GameModel : :setNextLevel (bool *nextLvl*)

Définir si l'on passe au niveau suivant.

Paramètres

<i>nextLvl</i>	: true si l'on passe au niveau suivant, false sinon
----------------	---

5.5.3.27 void GameModel : :setScore (int *score*)

Définir le score.

Paramètres

<i>score</i>	: Nouveau score
--------------	-----------------

La documentation de cette classe a été générée à partir des fichiers suivants :

- [GameModel.h](#)
- GameModel.cc

5.6 Référence de la classe GameView

Classe représentant la Vue du jeu.

```
#include <GameView.h>
```

Fonctions membres publiques

- [GameView](#) (int w, int h, int bpp)
Constructeur paramétré
- [~GameView](#) ()
Destructeur.
- void [setModel](#) ([GameModel](#) *model)
Définir le Modèle.
- void [updateLangage](#) ()
Mettre à jour la langue.

- void `updateVolume` ()
Mettre à jour le volume sonore.
- void `updateSound` ()
Mettre à jour les sons utilisés.
- void `bgscroll` ()
Faire défiler l'arrière-plan.
- void `explosion` (GraphicElement &elem, Ennemi *adr)
Dessiner une explosion.
- void `draw` ()
Dessiner tous les éléments dans la Vue.
- void `Display` ()
Afficher les éléments dans une fenêtre.
- void `transition` (std::string text, float t)
Dessiner une transition.
- void `synchronize` ()
Synchroniser la Vue avec le Modèle.
- bool `treatEvents` (bool &quit)
Traite les événements de la SFML.
- int `pause` ()
Dessiner l'écran de pause.
- void `intro` ()
Dessiner l'introduction du jeu.
- int `menu` ()
Dessiner le menu principal.
- void `options` (bool &quit, int &difficulty, int &initialLives)
Dessiner le menu des options.
- void `viewHighscores` (bool &quit)
Dessiner le menu des meilleurs scores.
- void `askName` ()
Dessiner la demande de son pseudo à l'utilisateur.

5.6.1 Description détaillée

Classe représentant la Vue du jeu.

Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

5.6.2 Documentation des constructeurs et destructeur

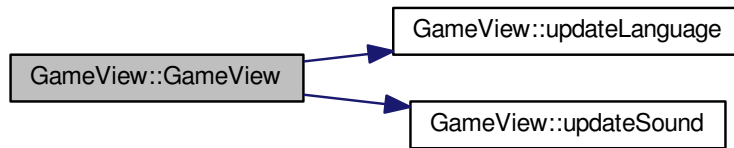
5.6.2.1 GameView : :GameView (int w, int h, int bpp)

Constructeur paramétré

Paramètres

<i>w</i>	: Largeur de la Vue
<i>h</i>	: Hauteur de la Vue
<i>bpp</i>	: Bits par pixel (profondeur des couleurs)

Voici le graphe d'appel pour cette fonction :



5.6.3 Documentation des fonctions membres

5.6.3.1 void GameView : :explosion (GraphicElement & elem, Ennemi * adr)

Dessiner une explosion.

Paramètres

<i>elem</i>	: Sprite de l' Ennemi explosant
<i>adr</i>	: Ennemi explosant

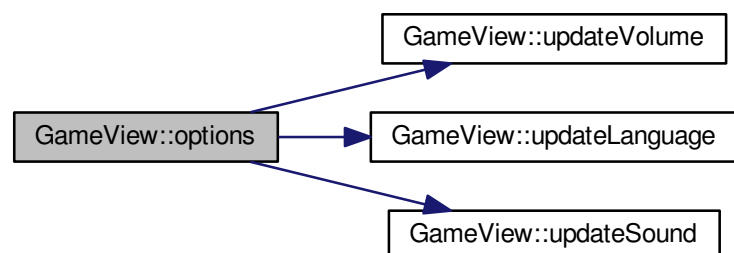
5.6.3.2 void GameView : :options (bool & quit, int & difficulty, int & initialLives)

Dessiner le menu des options.

Paramètres

<i>quit</i>	: (Sortie seulement) État du programme
<i>difficulty</i>	: (Sortie seulement) Difficulté du jeu
<i>initialLives</i>	: (Sortie seulement) Nombre initial de vies du Joueur

Voici le graphe d'appel pour cette fonction :



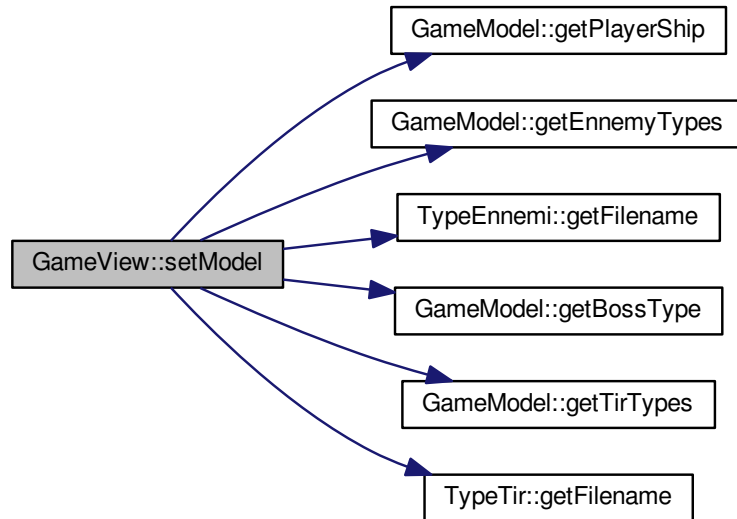
5.6.3.3 void GameView : :setModel (GameModel * model)

Définir le Modèle.

Paramètres

<i>model</i>	: GameModel à utiliser
--------------	--

Voici le graphe d'appel pour cette fonction :

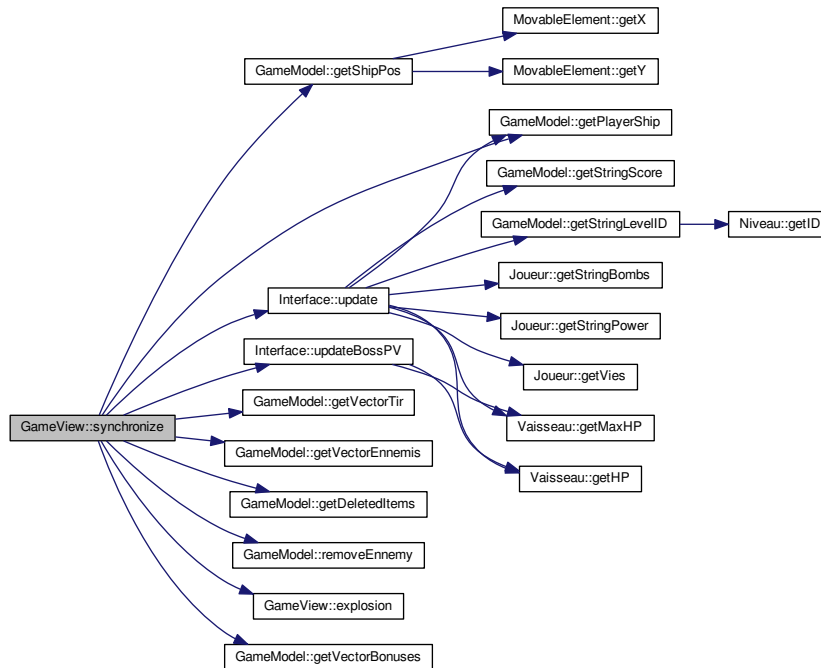


5.6.3.4 void GameView : :synchronize ()

Synchroniser la Vue avec le Modèle.

Bogue Une technique plus optimisée a été trouvée pour cette méthode, mais elle entraîne parfois des erreurs de segmentation, pour une raison inconnue

Voici le graphe d'appel pour cette fonction :



5.6.3.5 void GameView : :transition (std : :string text, float t)

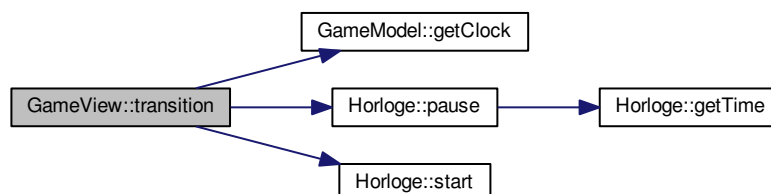
Dessiner une transition.

Méthode permettant d'afficher un texte et un compte à rebours après la mort du [Joueur](#) ou avant le début d'un [Niveau](#) par exemple

Paramètres

<i>text</i>	: Texte de la transition
<i>t</i>	: Durée de la transition (en secondes)

Voici le graphe d'appel pour cette fonction :



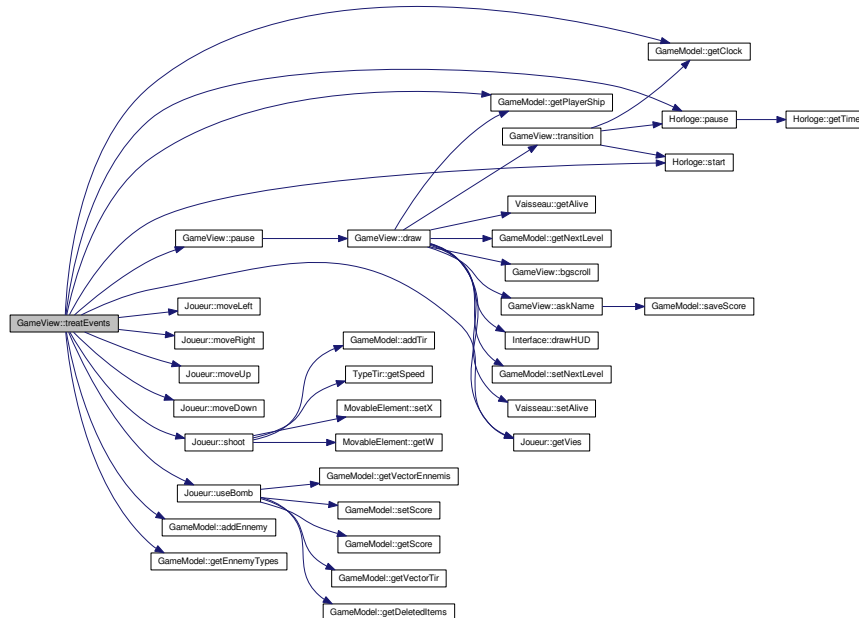
5.6.3.6 bool GameView : :treatEvents (bool & *quit*)

Traite les événements de la SFML.

Paramètres

<i>quit</i>	: (Sortie seulement) État du programme
-------------	--

Voici le graphe d'appel pour cette fonction :



5.6.3.7 void GameView : :viewHighscores (bool & quit)

Dessiner le menu des meilleurs scores.

Paramètres

<i>quit</i>	: (Sortie seulement) État du programme
-------------	--

Bogue Pour une raison inconnue, un Event : :Closed se produit à l'ouverture du menu. Pour pallier à ça, il a fallu empêcher de fermer le programme "normalement" une fois sur cet écran (l'utilisation de la touche Escape est toujours possible cependant)

La documentation de cette classe a été générée à partir des fichiers suivants :

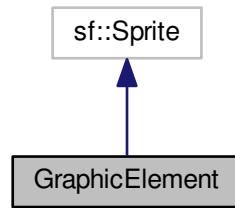
- [GameView.h](#)
- [GameView.cc](#)

5.7 Référence de la classe GraphicElement

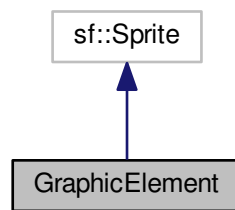
Classe représentant les éléments à afficher.

```
#include <GraphicElement.h>
```

Graphe d'héritage de `GraphicElement` :



Graphe de collaboration de `GraphicElement` :



Fonctions membres publiques

- `GraphicElement` ()
Constructeur par défaut.
- `GraphicElement` (sf : :Image *image, int x, int y, int w, int h)
Constructeur paramétré
- virtual `~GraphicElement` ()
Destructeur.
- virtual void `draw` (sf : :RenderWindow *_window)
Dessiner l'élément à afficher.
- void `setPosition` (const sf : :Vector2f &pos)
Modifier la position de l'élément à afficher.
- void `setPosition` (int x, int y)
Modifier la position de l'élément à afficher.
- void `resize` (int w, int h)
Modifier les dimensions de l'élément à afficher.
- bool `getVisible` () const
Accesseur.
- void `setVisible` (bool visible)
Définir la visibilité de l'élément à afficher.

5.7.1 Description détaillée

Classe représentant les éléments à afficher.

Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

5.7.2 Documentation des constructeurs et destructeur

5.7.2.1 `GraphicElement : :GraphicElement (sf : :Image * image, int x, int y, int w, int h)`

Constructeur paramétré

Paramètres

<i>image</i>	: Image à charger
<i>x</i>	: Abscisse de l'emplacement de l'élément à afficher dans la Vue
<i>y</i>	: Ordonnée de l'emplacement de l'élément à afficher dans la Vue
<i>w</i>	: Largeur de l'image à charger
<i>h</i>	: Hauteur de l'image à charger

5.7.3 Documentation des fonctions membres

5.7.3.1 `void GraphicElement : :draw (sf : :RenderWindow * _window) [virtual]`

Dessiner l'élément à afficher.

Méthode permettant de dessiner l'élément à afficher dans une fenêtre

Paramètres

<i>_window</i>	: Fenêtre où dessiner l'élément à afficher
----------------	--

5.7.3.2 `bool GraphicElement : :getVisible () const`

Accesseur.

Accesseur de `_visible`

Renvoie

: La visibilité actuelle de l'élément

5.7.3.3 `void GraphicElement : :resize (int w, int h)`

Modifier les dimensions de l'élément à afficher.

Paramètres

<i>w</i>	: Nouvelle largeur de l'élément à afficher
<i>h</i>	: Nouvelle hauteur de l'élément à afficher

5.7.3.4 `void GraphicElement : :setPosition (const sf : :Vector2f & pos)`

Modifier la position de l'élément à afficher.

Paramètres

<i>pos</i>	: Vecteur contenant les nouvelles coordonnées de l'élément à afficher
------------	---

5.7.3.5 void GraphicElement : :setPosition (int x, int y)

Modifier la position de l'élément à afficher.

Paramètres

<i>x</i>	: Nouvelle abscisse de l'élément à afficher
<i>y</i>	: Nouvelle ordonnée de l'élément à afficher

5.7.3.6 void GraphicElement : :setVisible (bool visible)

Définir la visibilité de l'élément à afficher.

Paramètres

<i>visible</i>	: Nouvelle visibilité de l'élément à afficher
----------------	---

La documentation de cette classe a été générée à partir des fichiers suivants :

- [GraphicElement.h](#)
- GraphicElement.cc

5.8 Référence de la classe Horloge

Classe représentant le temps dans le jeu.

```
#include <Horloge.h>
```

Fonctions membres publiques

- [Horloge](#) ()
Constructeur par défaut.
- float [getTime](#) ()
Retourner le temps écoulé
- bool [getStatus](#) () const
Accesseur.
- void [start](#) ()
Démarrer l'horloge.
- void [pause](#) ()
Mettre l'horloge en pause.
- void [stop](#) ()
Arrêter l'horloge.
- void [reset](#) ()
Réinitialiser l'horloge.

5.8.1 Description détaillée

Classe représentant le temps dans le jeu.

Remplace la classe Clock de la SFML 1.6, afin notamment d'implémenter une pause

Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

5.8.2 Documentation des fonctions membres

5.8.2.1 bool Horloge : :getStatus () const

Accesseur.

Accesseur de m_paused

Renvoie

: L'état de l'horloge (**true** si l'horloge est en pause, **false** sinon)

5.8.2.2 float Horloge : :getTime ()

Retourner le temps écoulé

Renvoie

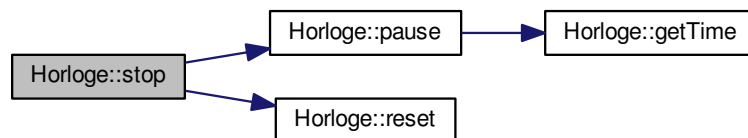
: Le temps écoulé depuis le démarrage de l'horloge

5.8.2.3 void Horloge : :stop ()

Arrêter l'horloge.

Met l'horloge en pause et la réinitialise

Voici le graphe d'appel pour cette fonction :



La documentation de cette classe a été générée à partir des fichiers suivants :

- [Horloge.h](#)
- Horloge.cc

5.9 Référence de la classe Interface

Classe représentant le HUD (affichage tête haute)

```
#include <Interface.h>
```

Fonctions membres publiques

- [Interface](#) ()
Constructeur par défaut.
- [~Interface](#) ()
Destructeur.
- void [drawHUD](#) (sf : :RenderWindow *_window)

- Dessiner le HUD.*
- void `update` (`GameModel` *`model`, string `strings`[])
Mettre à jour le HUD.
- void `updateBossPV` (`Boss` *`boss`)
Mettre à jour la barre de vie du boss.

5.9.1 Description détaillée

Classe représentant le HUD (affichage tête haute)

Auteur

Quentin HARSCOËT

5.9.2 Documentation des fonctions membres

5.9.2.1 void Interface : :drawHUD (sf : :RenderWindow * _window)

Dessiner le HUD.

Méthode permettant de dessiner le HUD dans une fenêtre

Paramètres

<code>_window</code>	: Fenêtre où dessiner le HUD
----------------------	------------------------------

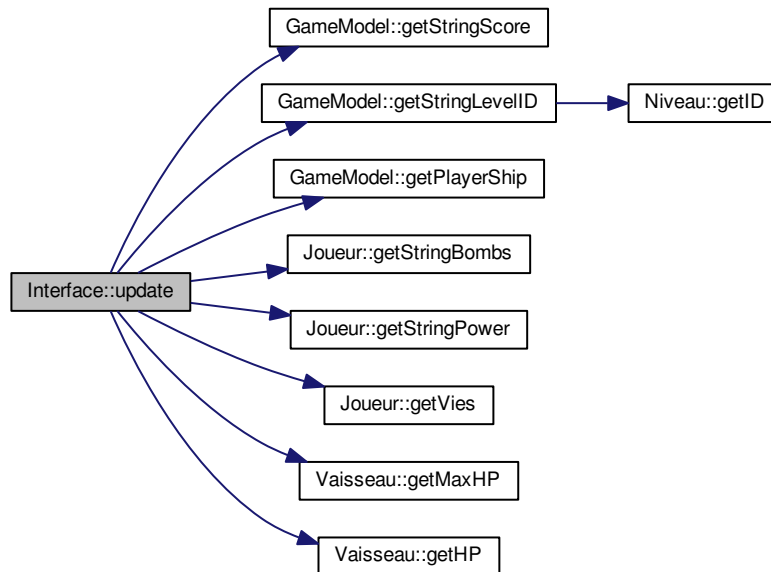
5.9.2.2 void Interface : :update (GameModel * model, string strings[])

Mettre à jour le HUD.

Paramètres

<code>model</code>	: Modèle du jeu
<code>strings[]</code>	: Tableau contenant les textes du HUD (utilisé pour la traduction)

Voici le graphe d'appel pour cette fonction :



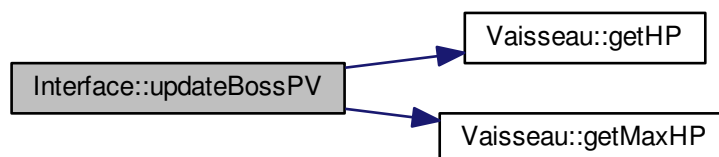
5.9.2.3 void Interface : :updateBossPV (Boss * boss)

Mettre à jour la barre de vie du boss.

Paramètres

<i>boss</i>	: Boss affronté
-------------	---------------------------------

Voici le graphe d'appel pour cette fonction :



La documentation de cette classe a été générée à partir des fichiers suivants :

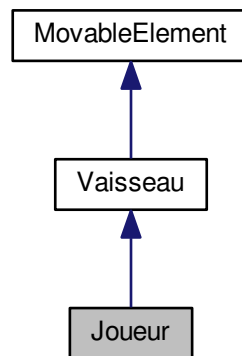
- [Interface.h](#)
- [Interface.cc](#)

5.10 Référence de la classe Joueur

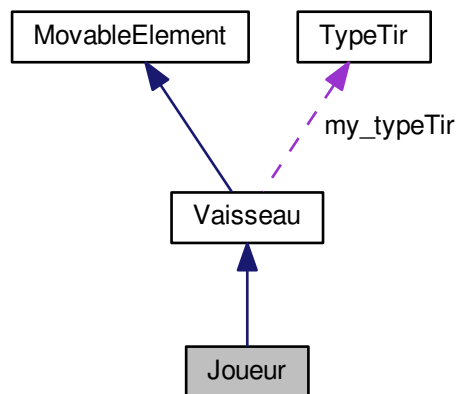
Classe représentant le vaisseau contrôlé

```
#include <Joueur.h>
```

Graphe d'héritage de Joueur :



Graphe de collaboration de Joueur :



Fonctions membres publiques

- **Joueur** (int power, **TypeTir** *typeTir, int vies, int pvMax, int bombs, float shootingSpeed)
Constructeur paramétré
- void **moveLeft** ()
Déplacer le vaisseau contrôlé vers la gauche.
- void **moveRight** ()
Déplacer le vaisseau contrôlé vers la droite.

- void `moveDown` ()
Déplacer le vaisseau contrôlé vers le bas.
- void `moveUp` ()
Déplacer le vaisseau contrôlé vers le haut.
- void `shoot` (`GameModel *model`)
Faire tirer le vaisseau contrôlé
- std::vector< `Tir` * >::iterator `isShot` (`GameModel *model`) const
Indiquer le tir touchant le vaisseau.
- bool `isHitByEnemy` (`GameModel *model`)
Indiquer si le vaisseau contrôlé est touché
- void `useBomb` (`GameModel *model`)
Utiliser une bombe.
- void `addPower` (int power, `GameModel *model`)
Ajouter de la puissance.
- void `addHP` (int hp)
Ajouter des points de vie.
- int `getVies` () const
Accesseur.
- int `getPower` () const
Accesseur.
- int `getBombs` () const
Accesseur.
- int `getNumberOfShoots` () const
Accesseur.
- std::string `getStringBombs` () const
Accesseur.
- std::string `getStringPower` () const
Accesseur.
- void `setVies` (int vies)
Définir le nombre de vies du vaisseau contrôlé
- void `setBombs` (int bombs)
Définir la quantité de bombes du vaisseau contrôlé
- void `setPower` (int i)
Définir la puissance du vaisseau contrôlé
- void `setNumberOfShoots` (int n)
Définir le nombre de tirs simultanés du vaisseau contrôlé

Additional Inherited Members

5.10.1 Description détaillée

Classe représentant le vaisseau contrôlé

Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

5.10.2 Documentation des constructeurs et destructeur

5.10.2.1 Joueur : :Joueur (int power, TypeTir * typeTir, int vies, int pvMax, int bombs, float shootingSpeed)

Constructeur paramétré

Paramètres

<i>power</i>	: Puissance initiale du vaisseau contrôlé
<i>typeTir</i>	: Type de tir initial du vaisseau contrôlé

<i>vies</i>	: Nombre initial de vies du vaisseau contrôlé
<i>pvMax</i>	: Nombre maximum de points de vie du vaisseau contrôlé
<i>bombs</i>	: Nombre initial de bombes du vaisseau contrôlé
<i>shootingSpeed</i>	: Vitesse de tir initiale du vaisseau contrôlé

5.10.3 Documentation des fonctions membres

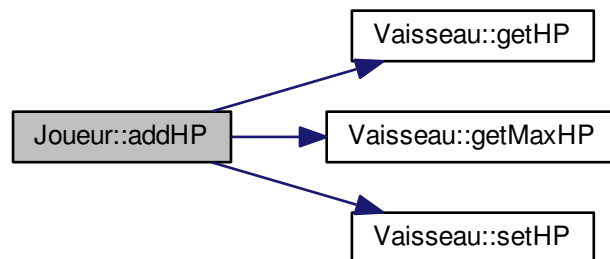
5.10.3.1 void Joueur : :addHP (int *hp*)

Ajouter des points de vie.

Paramètres

<i>hp</i>	: Points de vie à ajouter
-----------	---------------------------

Voici le graphe d'appel pour cette fonction :



5.10.3.2 void Joueur : :addPower (int *power*, GameModel * *model*)

Ajouter de la puissance.

Paramètres

<i>power</i>	: Puissance à ajouter
<i>model</i>	: Modèle du jeu

Voici le graphe d'appel pour cette fonction :



5.10.3.3 int Joueur : :getBombs () const

Accesseur.

Accesseur de my_bombs

Renvoie

: La quantité de bombes actuelle du vaisseau contrôlé

5.10.3.4 int Joueur : :getNumberOfShoots () const

Accesseur.

Accesseur de my_numberOfShoots

Renvoie

: Le nombre de tirs simultanés du vaisseau contrôlé (dépend de la puissance)

5.10.3.5 int Joueur : :getPower () const

Accesseur.

Accesseur de my_power

Renvoie

: La puissance actuelle du vaisseau contrôlé

5.10.3.6 string Joueur : :getStringBombs () const

Accesseur.

Accesseur de my_bombs

Renvoie

: La quantité de bombes actuelle du vaisseau contrôlé, en chaîne de caractères

5.10.3.7 string Joueur : :getStringPower () const

Accesseur.

Accesseur de my_power

Renvoie

: La puissance actuelle du vaisseau contrôlé, en chaîne de caractères

5.10.3.8 int Joueur : :getVies () const

Accesseur.

Accesseur de my_vies

Renvoie

: Le nombre de vies actuel du vaisseau contrôlé

5.10.3.9 bool Joueur : :isHitByEnemy (GameModel * *model*)

Indiquer si le vaisseau contrôlé est touché

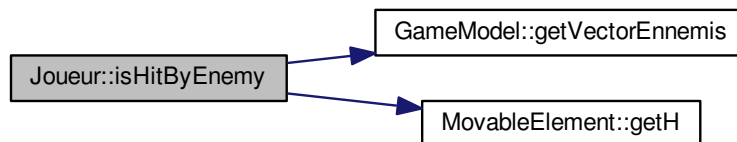
Paramètres

<i>model</i>	: Modèle du jeu
--------------	-----------------

Renvoie

: L'état du vaisseau contrôlé (**true** si le vaisseau contrôlé est touché, **false** sinon)

Voici le graphe d'appel pour cette fonction :



5.10.3.10 `vector< Tir * > :iterator Joueur : isShot (GameModel * model) const` [virtual]

Indiquer le tir touchant le vaisseau.

Paramètres

<i>model</i>	: Modèle du jeu
--------------	-----------------

Renvoie

: Le tir touchant le vaisseau contrôlé

Implémente [Vaisseau](#).

Voici le graphe d'appel pour cette fonction :



5.10.3.11 `void Joueur : shoot (GameModel * model)` [virtual]

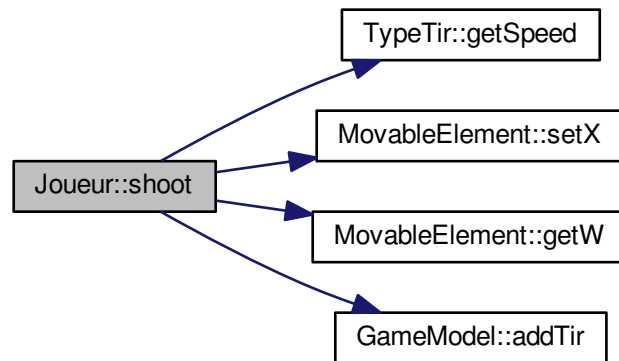
Faire tirer le vaisseau contrôlé

Paramètres

<i>model</i>	: Modèle du jeu
--------------	-----------------

Implémente [Vaisseau](#).

Voici le graphe d'appel pour cette fonction :



5.10.3.12 void Joueur : :useBomb (GameModel * model)

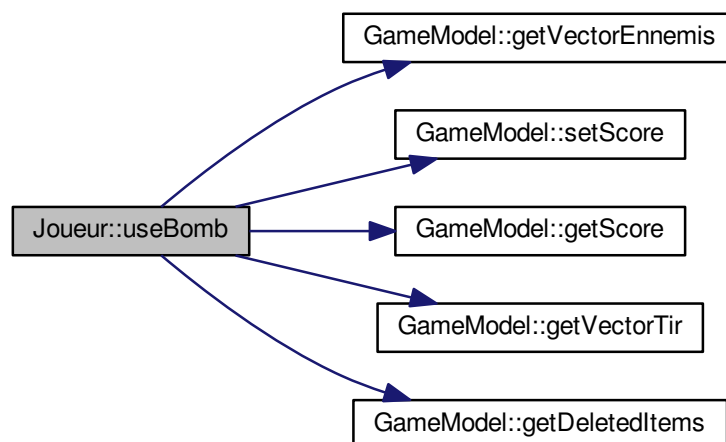
Utiliser une bombe.

Méthode permettant de détruire tous les ennemis à l'écran

Paramètres

<i>model</i>	: Modèle du jeu
--------------	-----------------

Voici le graphe d'appel pour cette fonction :



La documentation de cette classe a été générée à partir des fichiers suivants :

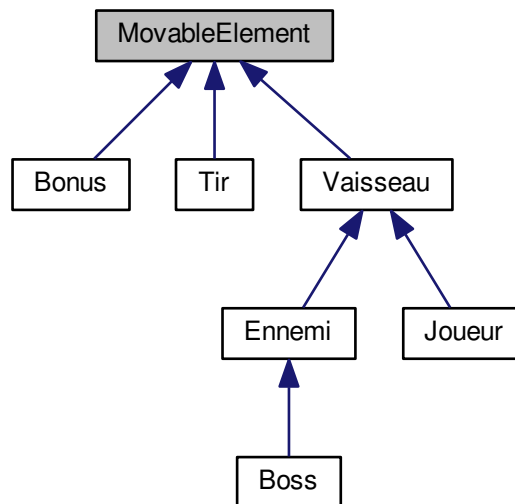
- [Joueur.h](#)
- [Joueur.cc](#)

5.11 Référence de la classe MovableElement

Classe représentant les éléments mobiles.

```
#include <MovableElement.h>
```

Graphe d'héritage de MovableElement :



Fonctions membres publiques

- [MovableElement](#) ()
Constructeur par défaut.
- [MovableElement](#) (int x, int y, int dx, int dy, int w, int h)
Constructeur paramétré
- virtual [~MovableElement](#) ()
Destructeur.
- virtual void [moveLeft](#) ()
Déplacer l'élément mobile vers la gauche.
- virtual void [moveRight](#) ()
Déplacer l'élément mobile vers la droite.
- virtual void [moveDown](#) ()
Déplacer l'élément mobile vers le bas.
- virtual void [moveUp](#) ()
Déplacer l'élément mobile vers le haut.
- virtual void [move](#) ()
Déplacer l'élément mobile.
- int [getX](#) () const
Accesseur.
- int [getY](#) () const
Accesseur.
- int [getDY](#) () const

- Accesseur.*
- int `getW` () const
- Accesseur.*
- int `getH` () const
- Accesseur.*
- void `setX` (int x)
- Définir l'abscisse de l'élément mobile.*
- void `setY` (int y)
- Définir l'ordonnée de l'élément mobile.*

Attributs protégés

- int `my_x`
- int `my_y`
- int `my_dx`
- int `my_dy`
- int `my_w`
- int `my_h`

5.11.1 Description détaillée

Classe représentant les éléments mobiles.

Auteur

Théo CHASSAIGNE
 Quentin HARSCOËT

5.11.2 Documentation des constructeurs et destructeur

5.11.2.1 MovableElement : :MovableElement (int x, int y, int dx, int dy, int w, int h)

Constructeur paramétré

Paramètres

<code>x</code>	: Abscisse de l'emplacement de l'élément mobile dans le Modèle
<code>y</code>	: Ordonnée de l'emplacement de l'élément mobile dans le Modèle
<code>dx</code>	: Vitesse (abscisse) de l'élément mobile
<code>dy</code>	: Vitesse (ordonnée) de l'élément mobile
<code>w</code>	: Largeur de l'élément mobile
<code>h</code>	: Hauteur de l'élément mobile

5.11.3 Documentation des fonctions membres

5.11.3.1 int MovableElement : :getDY () const

Accesseur.

Accesseur de `my_dy`

Renvoie

: La vitesse (ordonnée) de l'élément mobile

5.11.3.2 int MovableElement : :getH () const

Accesseur.

Accesseur de `my_h`

Renvoie

: La hauteur de l'élément mobile de l'élément mobile

5.11.3.3 int MovableElement : :getW () const

Accesseur.

Accesseur de my_w

Renvoie

: La largeur de l'élément mobile de l'élément mobile

5.11.3.4 int MovableElement : :getX () const

Accesseur.

Accesseur de my_x

Renvoie

: L'abscisse de l'élément mobile

5.11.3.5 int MovableElement : :getY () const

Accesseur.

Accesseur de my_y

Renvoie

: L'ordonnée de l'élément mobile

5.11.3.6 void MovableElement : :move () [virtual]

Déplacer l'élément mobile.

Méthode permettant de déplacer l'élément mobile en abscisse et en ordonnée

Réimplémentée dans [Ennemi](#), et [Boss](#).

5.11.4 Documentation des données membres**5.11.4.1 int MovableElement : :my_dx [protected]**

Vitesse (abscisse) de l'élément mobile

5.11.4.2 int MovableElement : :my_dy [protected]

Vitesse (ordonnée) de l'élément mobile

5.11.4.3 int MovableElement : :my_h [protected]

Hauteur de l'élément mobile

5.11.4.4 `int MovableElement : :my_w` `[protected]`

Largeur de l'élément mobile

5.11.4.5 `int MovableElement : :my_x` `[protected]`

Abscisse de l'élément mobile

5.11.4.6 `int MovableElement : :my_y` `[protected]`

Ordonnée de l'élément mobile

La documentation de cette classe a été générée à partir des fichiers suivants :

- [MovableElement.h](#)
- [MovableElement.cc](#)

5.12 Référence de la classe Niveau

Classe représentant les niveaux du jeu.

```
#include <Niveau.h>
```

Fonctions membres publiques

- [Niveau](#) ()
Constructeur par défaut.
- void [initNiveau](#) ([GameModel](#) *model)
Initialise le niveau.
- const std : :multimap< float,
[TypeEnnemi](#) * > * [getMap](#) () const
Accesseur.
- int [getID](#) () const
Accesseur.

5.12.1 Description détaillée

Classe représentant les niveaux du jeu.

Auteur

Quentin HARSCOËT

5.12.2 Documentation des fonctions membres

5.12.2.1 `int Niveau : :getID ()` const

Accesseur.

Accesseur de idNiveau

Renvoie

: L'identifiant du niveau

5.12.2.2 `const multimap< float, TypeEnnemi * > * Niveau : :getMap () const`

Accesseur.

Accesseur de ennemis

Renvoie

: L'adresse de la multimap contenant les instant où les ennemis apparaissent, ainsi que les types de ces derniers

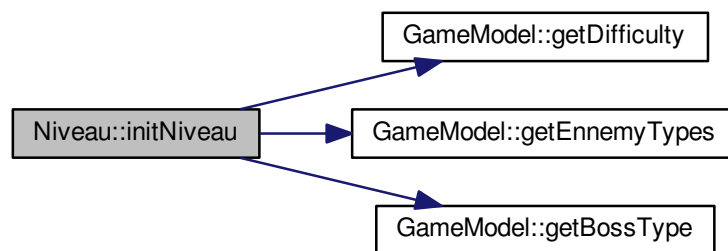
5.12.2.3 `void Niveau : :initNiveau (GameModel * model)`

Initialise le niveau.

Paramètres

<i>model</i>	: Modèle du jeu
--------------	-----------------

Voici le graphe d'appel pour cette fonction :



La documentation de cette classe a été générée à partir des fichiers suivants :

- [Niveau.h](#)
- [Niveau.cc](#)

5.13 Référence de la structure ShootPattern

Catégories de patterns.

```
#include <TypeEnnemi.h>
```

5.13.1 Description détaillée

Catégories de patterns.

La documentation de cette structure a été générée à partir du fichier suivant :

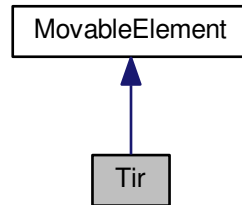
- [TypeEnnemi.h](#)

5.14 Référence de la classe Tir

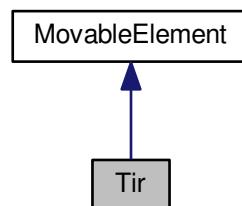
Classe représentant les tirs des [Vaisseau](#).

```
#include <Tir.h>
```

Graphe d'héritage de Tir :



Graphe de collaboration de Tir :



Fonctions membres publiques

- [Tir](#) (int x, int y, bool fromPlayer, [TypeTir](#) *type)
Constructeur paramétré
- [Tir](#) (int x, int y, int dx, int dy, bool fromPlayer, [TypeTir](#) *type)
Constructeur paramétré
- virtual [~Tir](#) ()
Destructeur.
- bool [getPlayer](#) () const
Accesseur.
- [TypeTir](#) * [getType](#) () const
Accesseur.

Additional Inherited Members

5.14.1 Description détaillée

Classe représentant les tirs des [Vaisseau](#).

Auteur

Quentin HARSCOËT

5.14.2 Documentation des constructeurs et destructeur

5.14.2.1 Tir : Tir (int x, int y, bool fromPlayer, TypeTir * type)

Constructeur paramétré

Paramètres

<i>x</i>	: Abscisse de l'emplacement du tir dans le Modèle
<i>y</i>	: Ordonnée de l'emplacement du tir dans le Modèle
<i>fromPlayer</i>	: Origine du tir (true si le tir vient du joueur, false sinon)
<i>type</i>	: Type du tir

5.14.2.2 Tir : Tir (int x, int y, int dx, int dy, bool fromPlayer, TypeTir * type)

Constructeur paramétré

Paramètres

<i>x</i>	: Abscisse de l'emplacement du tir dans le Modèle
<i>y</i>	: Ordonnée de l'emplacement du tir dans le Modèle
<i>dx</i>	: Vitesse (abscisse) du tir
<i>dy</i>	: Vitesse (ordonnée) du tir
<i>fromPlayer</i>	: Origine du tir (true si le tir vient du joueur, false sinon)
<i>type</i>	: Type du tir

5.14.3 Documentation des fonctions membres

5.14.3.1 bool Tir : getPlayer () const

Accesseur.

Accesseur de my_fromPlayer

Renvoie

: L'origine du tir (**true** si le tir vient du joueur, **false** sinon)

5.14.3.2 TypeTir * Tir : getType () const

Accesseur.

Accesseur de my_type

Renvoie

: Le type du tir

La documentation de cette classe a été générée à partir des fichiers suivants :

- [Tir.h](#)
- Tir.cc

5.15 Référence de la classe TypeBonus

Classe représentant les différents types de [Bonus](#).

```
#include <TypeBonus.h>
```

Fonctions membres publiques

- **TypeBonus** (**BonusEffect** effect, int value, float frequency, float scale)
Constructeur paramétré
- **BonusEffect** **getEffect** () const
Accesseur.
- int **getValue** () const
Accesseur.
- float **getFrequency** () const
Accesseur.
- float **getScale** () const
Accesseur.

5.15.1 Description détaillée

Classe représentant les différents types de **Bonus**.

Auteur

Quentin HARSCOËT

5.15.2 Documentation des constructeurs et destructeur

5.15.2.1 TypeBonus : :TypeBonus (**BonusEffect** effect, int value, float frequency, float scale)

Constructeur paramétré

Paramètres

<i>effect</i>	: Effet/Catégorie du Bonus
<i>value</i>	: Valeur du Bonus (le gain apporté)
<i>frequency</i>	: Fréquence/probabilité d'apparition du bonus, comprise entre 0 et 1
<i>scale</i>	: Taille du Bonus

5.15.3 Documentation des fonctions membres

5.15.3.1 **BonusEffect** TypeBonus : :getEffect () const

Accesseur.

Accesseur de my_effect

Renvoie

: L'effet/catégorie du **Bonus**

5.15.3.2 float TypeBonus : :getFrequency () const

Accesseur.

Accesseur de my_frequency

Renvoie

: La fréquence/probabilité d'apparition du **Bonus**

5.15.3.3 float TypeBonus : :getScale () const

Accesseur.

Accesseur de my_scale

Renvoie

: La taille du [Bonus](#)

5.15.3.4 int TypeBonus : :getValue () const

Accesseur.

Accesseur de my_value

Renvoie

: La valeur du [Bonus](#) (le gain apporté)

La documentation de cette classe a été générée à partir des fichiers suivants :

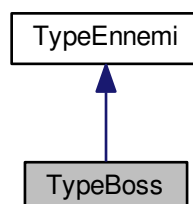
- [TypeBonus.h](#)
- TypeBonus.cc

5.16 Référence de la classe TypeBoss

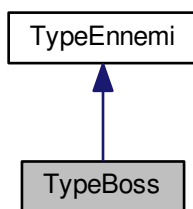
Classe représentant les différents types de [Boss](#).

```
#include <TypeBoss.h>
```

Graphe d'héritage de TypeBoss :



Graphe de collaboration de TypeBoss :



Fonctions membres publiques

- **TypeBoss** (int score, int speed, float shootingSpeed, int hp, int w, int h, **TypeTir** *typeTir, **ShootPattern** shootPattern, std::string filename, float changePatternDelay)
Constructeur paramétré
- float **getChangePatternDelay** () const
Accesseur.

5.16.1 Description détaillée

Classe représentant les différents types de **Boss**.

Auteur

Quentin HARSCOËT

5.16.2 Documentation des constructeurs et destructeur

5.16.2.1 **TypeBoss** : **TypeBoss** (int score, int speed, float shootingSpeed, int hp, int w, int h, **TypeTir** * typeTir, **ShootPattern** shootPattern, std::string filename, float changePatternDelay)

Constructeur paramétré

Paramètres

score	: Points rapportés par le Boss
speed	: Vitesse du Boss
shootingSpeed	: Vitesse de tir du Boss
hp	: Points de vie du Boss
w	: Largeur du Boss
h	: Hauteur du Boss
typeTir	: Type de tir du Boss
shootPattern	: Pattern de tir du Boss
filename	: Nom du fichier du sprite du Boss
changePattern-Delay	: Intervalle de temps entre deux changements de pattern du Boss

5.16.3 Documentation des fonctions membres

5.16.3.1 float TypeBoss : :getChangePatternDelay () const

Accesseur.

Accesseur de my_changePatternDelay

Renvoie

: L'intervalle de temps entre deux changements de pattern du [Boss](#)

La documentation de cette classe a été générée à partir des fichiers suivants :

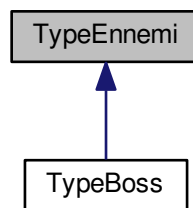
- [TypeBoss.h](#)
- [TypeBoss.cc](#)

5.17 Référence de la classe TypeEnnemi

Classe représentant les différents types d'[Ennemi](#).

```
#include <TypeEnnemi.h>
```

Graphe d'héritage de TypeEnnemi :



Fonctions membres publiques

- [TypeEnnemi](#) ()
Constructeur par défaut.
- [TypeEnnemi](#) (int score, int speed, float shootingSpeed, int hp, int w, int h, [TypeTir](#) *typeTir, [ShootPattern](#) shootPattern, std : :string filename)
Constructeur paramétré
- virtual [~TypeEnnemi](#) ()
Destructeur.
- int [getScore](#) () const
Accesseur.
- int [getSpeed](#) () const
Accesseur.
- float [getShootingSpeed](#) () const
Accesseur.
- int [getHP](#) () const
Accesseur.
- int [getW](#) () const
Accesseur.
- int [getH](#) () const
Accesseur.
- std : :string [getFilename](#) () const
Accesseur.
- [TypeTir](#) * [getTypeTir](#) () const
Accesseur.

- `ShootPattern getShootPattern ()` const
Accesseur.

5.17.1 Description détaillée

Classe représentant les différents types d'[Ennemi](#).

Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

5.17.2 Documentation des constructeurs et destructeur

5.17.2.1 `TypeEnnemi : :TypeEnnemi (int score, int speed, float shootingSpeed, int hp, int w, int h, TypeTir * typeTir, ShootPattern shootPattern, std : :string filename)`

Constructeur paramétré

Paramètres

<i>score</i>	: Points rapportés par l' Ennemi
<i>speed</i>	: Vitesse de l' Ennemi
<i>shootingSpeed</i>	: Vitesse de tir de l' Ennemi
<i>hp</i>	: Points de vie de l' Ennemi
<i>w</i>	: Largeur de l' Ennemi
<i>h</i>	: Hauteur de l' Ennemi
<i>typeTir</i>	: Type de tir de l' Ennemi
<i>shootPattern</i>	: Pattern de tir de l' Ennemi
<i>filename</i>	: Nom du fichier du sprite de l' Ennemi

5.17.3 Documentation des fonctions membres

5.17.3.1 `string TypeEnnemi : :getFilename ()` const

Accesseur.

Accesseur de `my_filename`

Renvoie

: La nom du fichier du sprite de l'[Ennemi](#)

5.17.3.2 `int TypeEnnemi : :getH ()` const

Accesseur.

Accesseur de `my_h`

Renvoie

: La hauteur de l'[Ennemi](#)

5.17.3.3 `int TypeEnnemi : :getHP ()` const

Accesseur.

Accesseur de `my_hp`

Renvoie

: Les points de vie de l'[Ennemi](#)

5.17.3.4 `int TypeEnnemi : :getScore () const`

Accesseur.

Accesseur de `my_score`

Renvoie

: Les points rapportés par l'[Ennemi](#)

5.17.3.5 `float TypeEnnemi : :getShootingSpeed () const`

Accesseur.

Accesseur de `my_shootingSpeed`

Renvoie

: La vitesse de tir de l'[Ennemi](#)

5.17.3.6 `ShootPattern TypeEnnemi : :getShootPattern () const`

Accesseur.

Accesseur de `my_shootPattern`

Renvoie

: Le pattern de tir de l'[Ennemi](#)

5.17.3.7 `int TypeEnnemi : :getSpeed () const`

Accesseur.

Accesseur de `my_speed`

Renvoie

: La vitesse de l'[Ennemi](#)

5.17.3.8 `TypeTir * TypeEnnemi : :getTypeTir () const`

Accesseur.

Accesseur de `my_typeTir`

Renvoie

: Le type de tir de l'[Ennemi](#)

5.17.3.9 int TypeEnnemi : :getW () const

Accesseur.

Accesseur de my_w

Renvoie

: La largeur de l'[Ennemi](#)

La documentation de cette classe a été générée à partir des fichiers suivants :

- [TypeEnnemi.h](#)
- TypeEnnemi.cc

5.18 Référence de la classe TypeTir

Classe représentant les différents types de [Tir](#).

```
#include <TypeTir.h>
```

Fonctions membres publiques

- [TypeTir](#) (int w, int h, int damages, int speed, std : :string filename)
Constructeur paramétré
- std : :string [getFilename](#) () const
Accesseur.
- int [getH](#) () const
Accesseur.
- int [getW](#) () const
Accesseur.
- int [getDamages](#) () const
Accesseur.
- int [getSpeed](#) () const
Accesseur.

5.18.1 Description détaillée

Classe représentant les différents types de [Tir](#).

Auteur

Quentin HARSCOËT

5.18.2 Documentation des constructeurs et destructeur

5.18.2.1 [TypeTir](#) : :TypeTir (int w, int h, int damages, int speed, std : :string filename)

Constructeur paramétré

Paramètres

<i>w</i>	: Largeur du Tir
<i>h</i>	: Hauteur du Tir
<i>damages</i>	: Dommages infligés par le Tir
<i>speed</i>	: Vitesse du Tir

<i>filename</i>	: Nom du fichier du sprite du Tir
-----------------	---

5.18.3 Documentation des fonctions membres

5.18.3.1 `int TypeTir::getDamages () const`

Accesseur.

Accesseur de `my_w`

Renvoie

: Les dommages infligés par le [Tir](#)

5.18.3.2 `string TypeTir::getFilename () const`

Accesseur.

Accesseur de `my_filename`

Renvoie

: Le nom du fichier du sprite du [Tir](#)

5.18.3.3 `int TypeTir::getH () const`

Accesseur.

Accesseur de `my_h`

Renvoie

: La hauteur du [Tir](#)

5.18.3.4 `int TypeTir::getSpeed () const`

Accesseur.

Accesseur de `my_w`

Renvoie

: La vitesse du [Tir](#)

5.18.3.5 `int TypeTir::getW () const`

Accesseur.

Accesseur de `my_w`

Renvoie

: La largeur du [Tir](#)

La documentation de cette classe a été générée à partir des fichiers suivants :

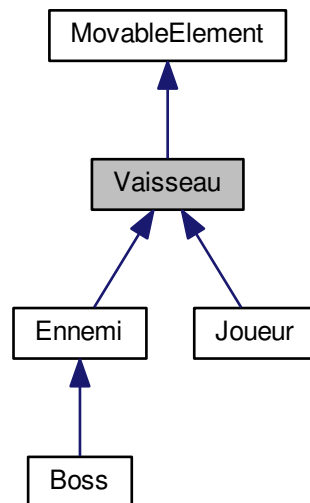
- [TypeTir.h](#)
- [TypeTir.cc](#)

5.19 Référence de la classe Vaisseau

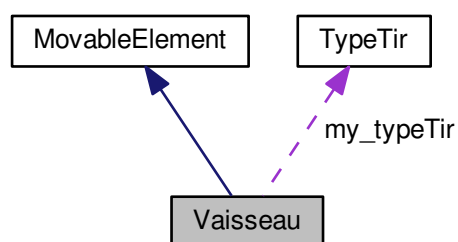
Classe représentant les vaisseaux.

```
#include <Vaisseau.h>
```

Graphe d'héritage de Vaisseau :



Graphe de collaboration de Vaisseau :



Fonctions membres publiques

- **Vaisseau** (int hp, **TypeTir** *type, float shootingSpeed)
Constructeur paramétré
- virtual **~Vaisseau** ()
Destructeur.
- int **getHP** () const
Accesseur.

- int `getMaxHP` () const
Accesseur.
- `TypeTir` * `getTypeTir` () const
Accesseur.
- bool `getAlive` () const
Accesseur.
- float `getShootingSpeed` () const
Accesseur.
- sf : :Clock * `getClock` ()
Accesseur.
- void `setHP` (int i)
Définir les points de vie du `Vaisseau`.
- void `setTypeTir` (`TypeTir` *type)
Définir le type de tir du `Vaisseau`.
- void `setAlive` (bool b)
Définir l'état du `Vaisseau`.
- virtual void `shoot` (`GameModel` *model)=0
Faire tirer le `Vaisseau` (méthode virtuelle pure)
- virtual std : :vector< `Tir` * >
: :iterator `isShot` (`GameModel` *model) const =0
Indiquer le tir touchant le `Vaisseau`.

Attributs protégés

- `TypeTir` * `my_typeTir`
- sf : :Clock `my_clock`
- float `my_shootingSpeed`

5.19.1 Description détaillée

Classe représentant les vaisseaux.

Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

5.19.2 Documentation des constructeurs et destructeur

5.19.2.1 `Vaisseau` : :`Vaisseau` (int *hp*, `TypeTir` * *type*, float *shootingSpeed*)

Constructeur paramétré

Paramètres

<i>hp</i>	: Points de vie initiaux du <code>Vaisseau</code>
<i>type</i>	: Type de tir initial du <code>Vaisseau</code>
<i>shootingSpeed</i>	: Cadence de tir initiale du <code>Vaisseau</code>

5.19.3 Documentation des fonctions membres

5.19.3.1 bool `Vaisseau` : :`getAlive` () const

Accesseur.

Accesseur de `my_isAlive`

Renvoie

: L'état actuel du `Vaisseau` (**true** si le `Vaisseau` n'est pas détruit, **false** sinon)

5.19.3.2 `sf : :Clock * Vaisseau : :getClock ()`

Accesseur.

Accesseur de my_clock

Renvoie

: La Clock interne au [Vaisseau](#)

5.19.3.3 `int Vaisseau : :getHP () const`

Accesseur.

Accesseur de my_hp

Renvoie

: Les points de vie actuels du [Vaisseau](#)

5.19.3.4 `int Vaisseau : :getMaxHP () const`

Accesseur.

Accesseur de my_pvMax

Renvoie

: Les points de vie maximum du [Vaisseau](#)

5.19.3.5 `float Vaisseau : :getShootingSpeed () const`

Accesseur.

Accesseur de my_shootingSpeed

Renvoie

: La cadence de tir actuelle du [Vaisseau](#)

5.19.3.6 `TypeTir * Vaisseau : :getTypeTir () const`

Accesseur.

Accesseur de my_typeTir

Renvoie

: Le type de tir actuel du [Vaisseau](#)

5.19.3.7 `virtual std : :vector<Tir*> : :iterator Vaisseau : :isShot (GameModel * model) const` [pure virtual]

Indiquer le tir touchant le [Vaisseau](#).

Paramètres

<i>model</i>	: Modèle du jeu (méthode virtuelle pure)
--------------	--

Renvoie

: Le tir touchant le vaisseau contrôlé

Implémenté dans [Ennemi](#), et [Joueur](#).

5.19.3.8 void Vaisseau : :setAlive (bool *b*)

Définir l'état du [Vaisseau](#).

Paramètres

<i>b</i>	: Nouvel état du Vaisseau (true si le Vaisseau n'est pas détruit, false sinon)
----------	---

5.19.3.9 void Vaisseau : :setHP (int *i*)

Définir les points de vie du [Vaisseau](#).

Paramètres

<i>i</i>	: Nouveaux points de vie du Vaisseau
----------	--

5.19.3.10 void Vaisseau : :setTypeTir (TypeTir * *type*)

Définir le type de tir du [Vaisseau](#).

Paramètres

<i>type</i>	: Nouveau type de tir du Vaisseau
-------------	---

5.19.3.11 virtual void Vaisseau : :shoot (GameModel * *model*) [pure virtual]

Faire tirer le [Vaisseau](#) (méthode virtuelle pure)

Paramètres

<i>model</i>	: Modèle du jeu
--------------	-----------------

Implémenté dans [Ennemi](#), et [Joueur](#).

5.19.4 Documentation des données membres

5.19.4.1 sf : :Clock Vaisseau : :my_clock [protected]

Pendule interne au [Vaisseau](#), pour y gérer le temps

5.19.4.2 float Vaisseau : :my_shootingSpeed [protected]

Cadence de tir, en secondes (un tir toutes les x secondes)

5.19.4.3 `TypeTir*` Vaisseau : `:my_typeTir` [protected]

Pointeur vers le type des tirs du [Vaisseau](#)

La documentation de cette classe a été générée à partir des fichiers suivants :

- [Vaisseau.h](#)
- [Vaisseau.cc](#)

Chapitre 6

Documentation des fichiers

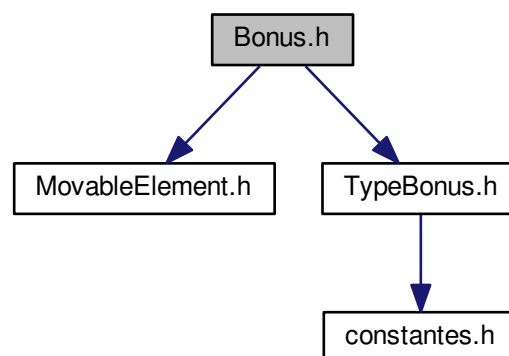
6.1 Référence du fichier Bonus.h

Déclaration de la class [Bonus](#).

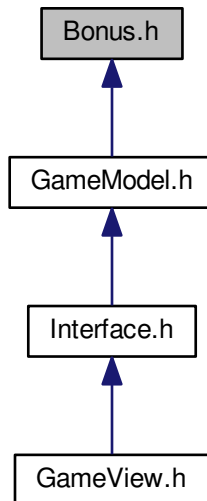
```
#include "MovableElement.h"
```

```
#include "TypeBonus.h"
```

Graphe des dépendances par inclusion de Bonus.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class [Bonus](#)
Classe représentant les bonii à ramasser.

6.1.1 Description détaillée

Déclaration de la class [Bonus](#).

Auteur

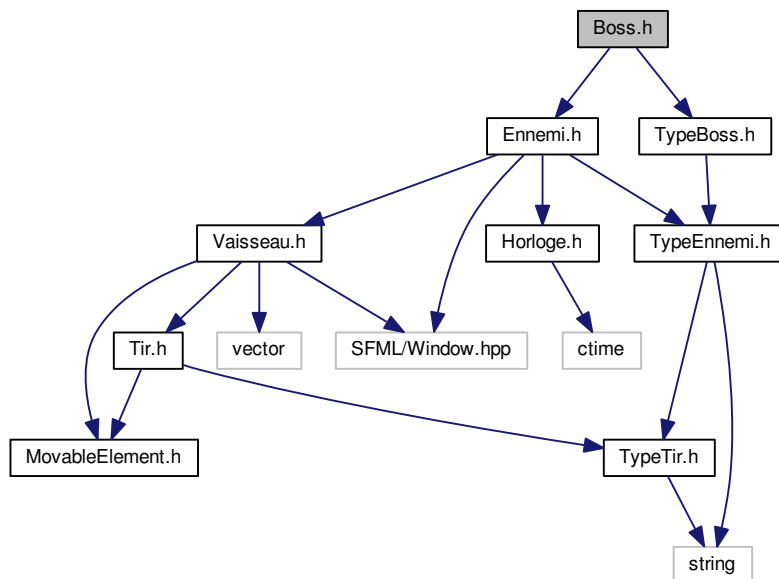
Théo CHASSAIGNE
Quentin HARSCOËT

6.2 Référence du fichier Boss.h

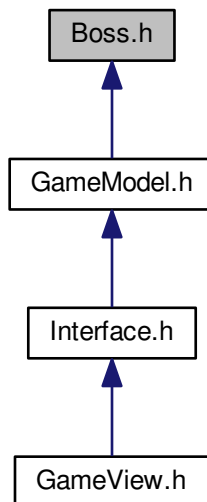
Déclaration de la class [Boss](#).

```
#include "Ennemi.h"  
#include "TypeBoss.h"
```

Graphe des dépendances par inclusion de Boss.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class `Boss`
Classe représentant les boss de fin de niveau.

6.2.1 Description détaillée

Déclaration de la class [Boss](#).

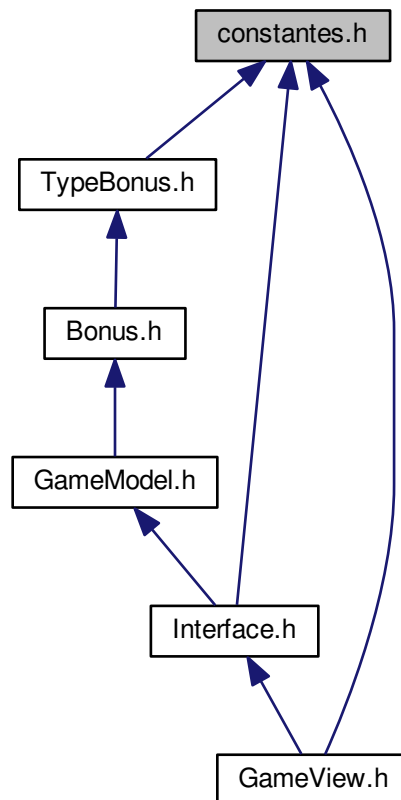
Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

6.3 Référence du fichier constantes.h

Déclaration des constantes utilisées.

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Macros

```
- #define MODEL_WIDTH 768
- #define MODEL_HEIGHT 768
- #define VIEW_WIDTH 768
- #define VIEW_HEIGHT 768
- #define SHEEP_WIDTH 200
- #define SHEEP_HEIGHT 309
- #define SHEEP_SCALE 0.2
- #define ENEMY_SCALE 0.3
- #define SHEEP_SPEED 10
```

```
– #define MOVABLE_ELEMENT_SPEED 10
– #define NB_ENNEMY_TYPES 3
– #define NB_TIRS_TYPES 7
– #define NB_BONUS_TYPES 5
– #define NB_STRINGS 26
– #define VIES 0
– #define PV 1
– #define NIVEAU 2
– #define BOMBES 3
– #define PUISSANCE 4
– #define JOUER 5
– #define HIGHSCORES 6
– #define SETTINGS 7
– #define QUIT 8
– #define PAUSE 9
– #define RESUME 10
– #define BACKTOMENU 11
– #define NOSCORE 12
– #define ENTERNAME 13
– #define FINISHLEVEL 14
– #define GAMEOVER 15
– #define DEAD 16
– #define SOUNDLEVEL 17
– #define DIFFICULTY 18
– #define LIVESNUMBER 19
– #define LANGUAGE 20
– #define NORMAL 21
– #define HARD 22
– #define VERYHARD 23
– #define MYLANGUE 24
– #define SOUNDPACK 25
```

6.3.1 Description détaillée

Déclaration des constantes utilisées.

Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

6.3.2 Documentation des macros

6.3.2.1 #define BACKTOMENU 11

Index de la chaîne "Retourner au Menu"

6.3.2.2 #define BOMBES 3

Index de la chaîne "Bombes"

6.3.2.3 #define DEAD 16

Index de la chaîne "Vous avez perdu une vie ! "

6.3.2.4 #define DIFFICULTY 18

Index de la chaîne "Difficulté"

6.3.2.5 #define ENEMY_SCALE 0.3

Echelle des [Ennemi](#)

6.3.2.6 #define ENTERNAME 13

Index de la chaîne "Entrez votre nom"

6.3.2.7 #define FINISHLEVEL 14

Index de la chaîne "Vous avez fini un niveau !"

6.3.2.8 #define GAMEOVER 15

Index de la chaîne "Partie Terminée !"

6.3.2.9 #define HARD 22

Index de la chaîne "Difficile"

6.3.2.10 #define HIGHSCORES 6

Index de la chaîne "Meilleurs scores"

6.3.2.11 #define JOUER 5

Index de la chaîne "Jouer"

6.3.2.12 #define LANGUAGE 20

Index de la chaîne "Langue"

6.3.2.13 #define LIVESNUMBER 19

Index de la chaîne "Nombre de vies"

6.3.2.14 #define MODEL_HEIGHT 768

Hauteur du Modèle

6.3.2.15 #define MODEL_WIDTH 768

Largeur du Modèle

6.3.2.16 #define MOVABLE_ELEMENT_SPEED 10

Vitesse par défaut des [MovableElement](#)

6.3.2.17 #define MYLANGUE 24

Index de la chaîne indiquant la langue

6.3.2.18 `#define NB_BONUS_TYPES 5`

Nombre de [TypeBonus](#)

6.3.2.19 `#define NB_ENNEMY_TYPES 3`

Nombre de [TypeEnnemi](#)

6.3.2.20 `#define NB_STRINGS 26`

Nombre d'entrées du tableau contenant les textes du jeu

6.3.2.21 `#define NB_TIRS_TYPES 7`

Nombre de [TypeTir](#)

6.3.2.22 `#define NIVEAU 2`

Index de la chaîne "Niveau"

6.3.2.23 `#define NORMAL 21`

Index de la chaîne "Normal"

6.3.2.24 `#define NOSCORE 12`

Index de la chaîne "Pas de score"

6.3.2.25 `#define PAUSE 9`

Index de la chaîne "Pause"

6.3.2.26 `#define PUISSANCE 4`

Index de la chaîne "Puissance"

6.3.2.27 `#define PV 1`

Index de la chaîne "PV"

6.3.2.28 `#define QUIT 8`

Index de la chaîne "Quitter"

6.3.2.29 `#define RESUME 10`

Index de la chaîne "Continuer"

6.3.2.30 `#define SETTINGS 7`

Index de la chaîne "Paramètres"

6.3.2.31 `#define SHEEP_HEIGHT 309`

Hauteur du [Joueur](#)

6.3.2.32 `#define SHEEP_SCALE 0.2`

Echelle du [Joueur](#)

6.3.2.33 `#define SHEEP_SPEED 10`

Vitesse du [Joueur](#)

6.3.2.34 `#define SHEEP_WIDTH 200`

Largeur du [Joueur](#)

6.3.2.35 `#define SOUNDLEVEL 17`

Index de la chaîne "Volume sonore"

6.3.2.36 `#define SOUNDPACK 25`

Index de la chaîne "Pack de son"

6.3.2.37 `#define VERYHARD 23`

Index de la chaîne "Très difficile"

6.3.2.38 `#define VIES 0`

Index de la chaîne "Vies"

6.3.2.39 `#define VIEW_HEIGHT 768`

Hauteur de la Vue

6.3.2.40 `#define VIEW_WIDTH 768`

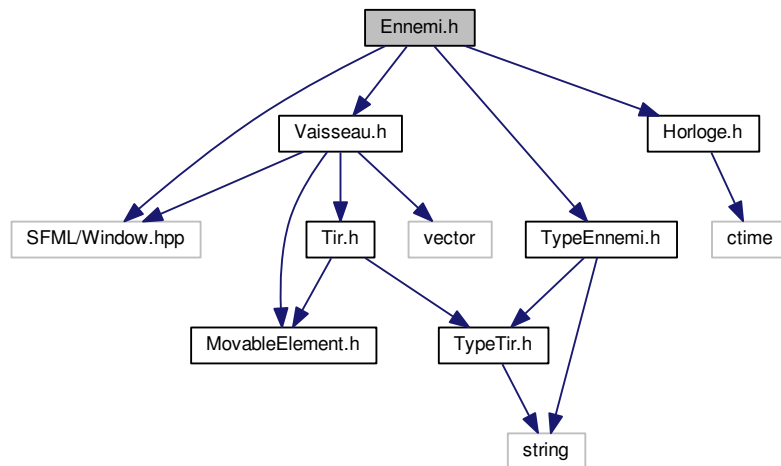
Largeur de la Vue

6.4 Référence du fichier `Ennemi.h`

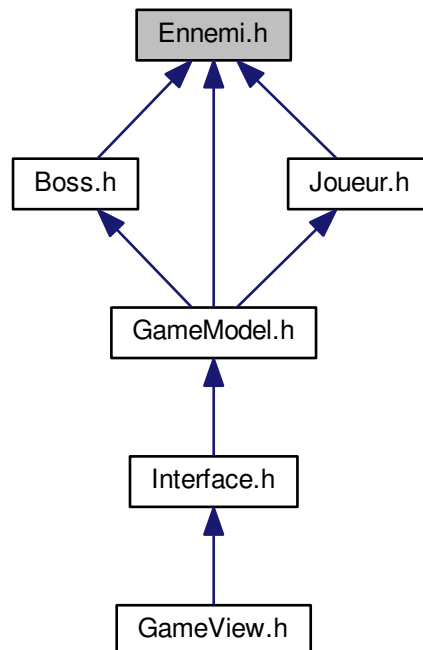
Déclaration de la class [Ennemi](#).

```
#include <SFML/Window.hpp>
#include "Vaisseau.h"
#include "TypeEnnemi.h"
#include "Horloge.h"
```

Graphe des dépendances par inclusion de Ennemi.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class [Ennemi](#)
Classe représentant les ennemis du joueur.

6.4.1 Description détaillée

Déclaration de la class [Ennemi](#).

Auteur

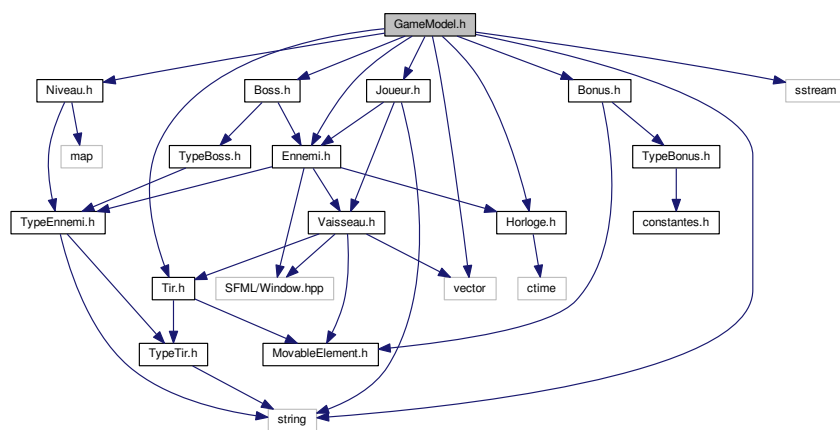
Théo CHASSAIGNE
Quentin HARSCOËT

6.5 Référence du fichier GameModel.h

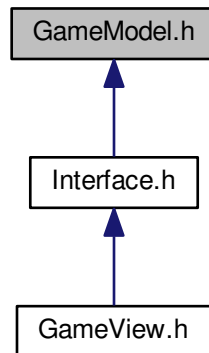
Déclaration de la classe [GameModel](#).

```
#include "Tir.h"
#include "Ennemi.h"
#include "Boss.h"
#include "Joueur.h"
#include "Bonus.h"
#include "Niveau.h"
#include "Horloge.h"
#include <vector>
#include <string>
#include <sstream>
```

Graphe des dépendances par inclusion de GameModel.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class [GameModel](#)
Classe représentant le Modèle du jeu.

6.5.1 Description détaillée

Déclaration de la classe [GameModel](#).

Auteur

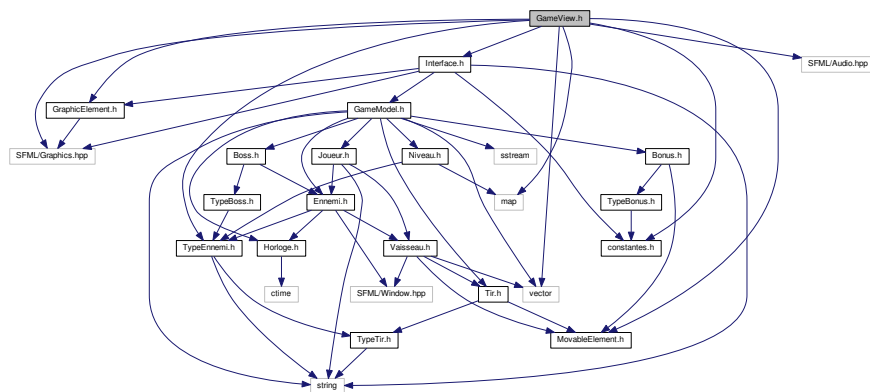
Théo CHASSAIGNE
Quentin HARSCOËT

6.6 Référence du fichier GameView.h

Déclaration de la classe [GameView](#).

```
#include <SFML/Graphics.hpp>
#include <SFML/Audio.hpp>
#include <map>
#include <vector>
#include "GraphicElement.h"
#include "MovableElement.h"
#include "TypeEnnemi.h"
#include "constantes.h"
#include "Interface.h"
```

Graphe des dépendances par inclusion de `GameView.h` :



Classes

- class `GameView`
Classe représentant la Vue du jeu.

6.6.1 Description détaillée

Déclaration de la classe `GameView`.

Auteur

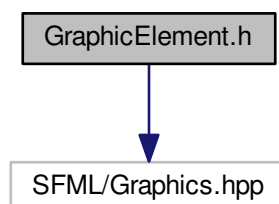
Théo CHASSAIGNE
Quentin HARSCOËT

6.7 Référence du fichier `GraphicElement.h`

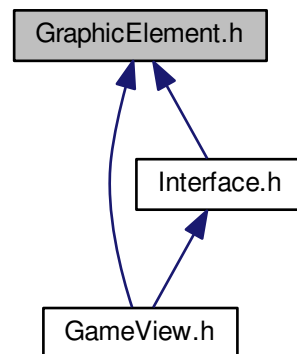
Déclaration de la classe `GraphicElement`.

```
#include <SFML/Graphics.hpp>
```

Graphe des dépendances par inclusion de `GraphicElement.h` :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class [GraphicElement](#)
Classe représentant les éléments à afficher.

6.7.1 Description détaillée

Déclaration de la classe [GraphicElement](#).

Auteur

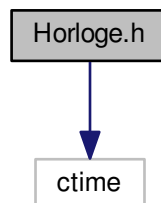
Théo CHASSAIGNE
Quentin HARSCOËT

6.8 Référence du fichier Horloge.h

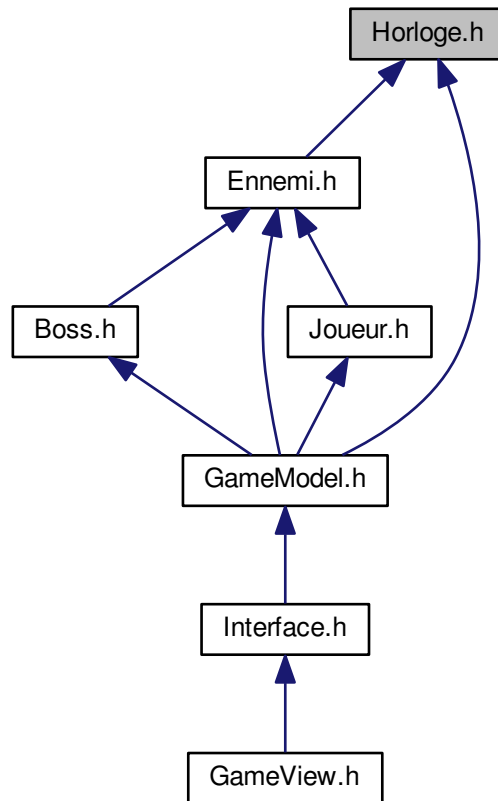
Déclaration de la classe [Horloge](#).

```
#include <ctime>
```

Graphe des dépendances par inclusion de Horloge.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class [Horloge](#)
Classe représentant le temps dans le jeu.

6.8.1 Description détaillée

Déclaration de la classe [Horloge](#).

Auteur

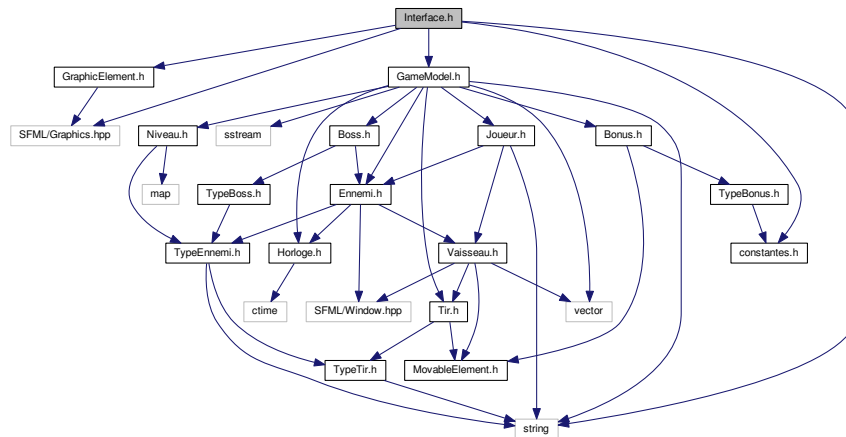
Théo CHASSAIGNE
Quentin HARSCOËT

6.9 Référence du fichier Interface.h

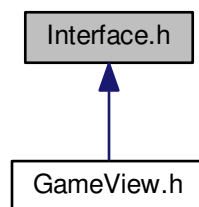
Déclaration de la classe [Interface](#).

```
#include <SFML/Graphics.hpp>
#include <string>
#include "GameModel.h"
#include "GraphicElement.h"
#include "constantes.h"
```

Graphe des dépendances par inclusion de Interface.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class [Interface](#)
Classe représentant le HUD (affichage tête haute)

6.9.1 Description détaillée

Déclaration de la classe [Interface](#).

Auteur

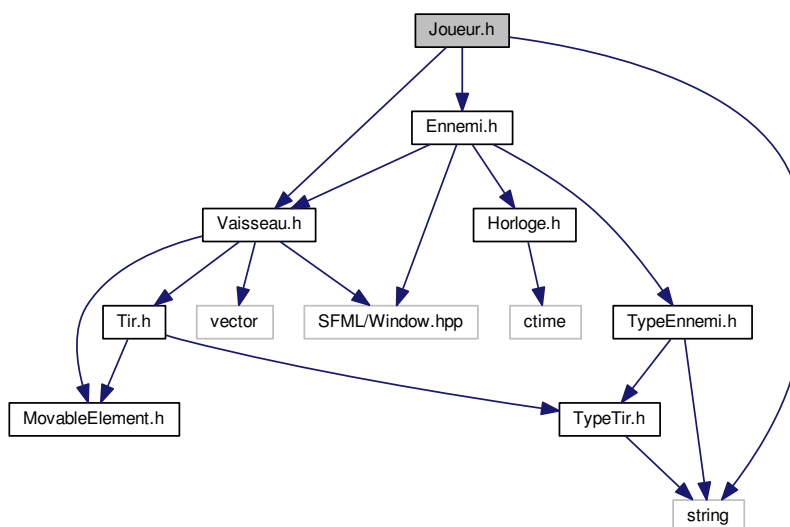
Quentin HARSCOËT

6.10 Référence du fichier Joueur.h

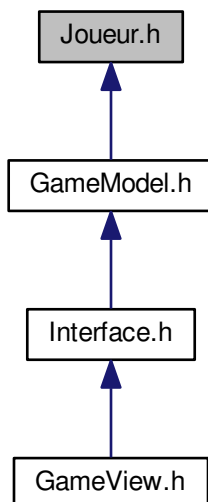
Déclaration de la classe [Joueur](#).

```
#include "Vaisseau.h"  
#include "Ennemi.h"  
#include <string>
```

Graphe des dépendances par inclusion de Joueur.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class [Joueur](#)
Classe représentant le vaisseau contrôlé

6.10.1 Description détaillée

Déclaration de la classe [Joueur](#).

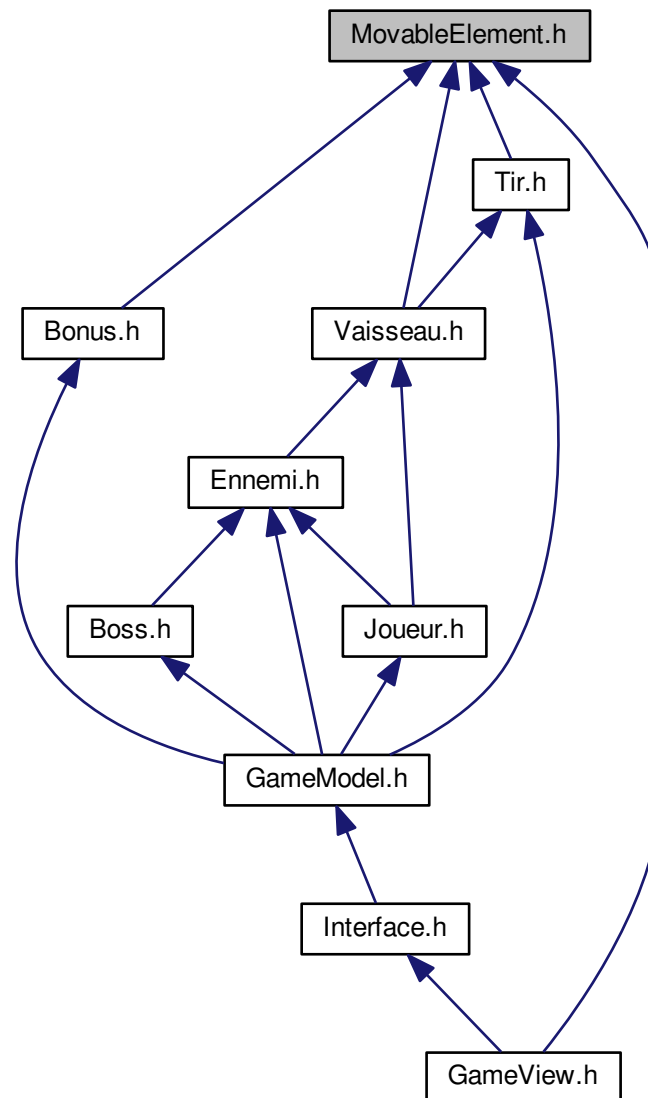
Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

6.11 Référence du fichier MovableElement.h

Déclaration de la classe [MovableElement](#).

Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class [MovableElement](#)
Classe représentant les éléments mobiles.

6.11.1 Description détaillée

Déclaration de la classe [MovableElement](#).

Auteur

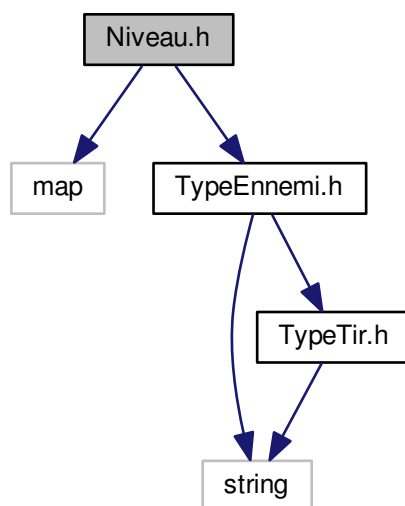
Théo CHASSAIGNE
Quentin HARSCOËT

6.12 Référence du fichier Niveau.h

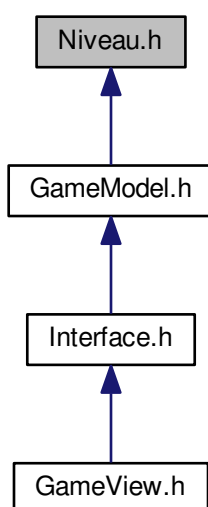
Déclaration de la classe [Niveau](#).

```
#include <map>
#include "TypeEnnemi.h"
```

Graphe des dépendances par inclusion de Niveau.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class [Niveau](#)
Classe représentant les niveaux du jeu.

6.12.1 Description détaillée

Déclaration de la classe [Niveau](#).

Auteur

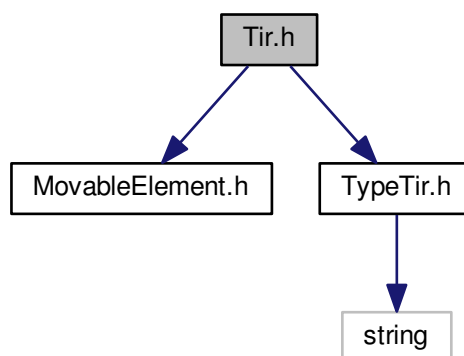
Quentin HARSCOËT

6.13 Référence du fichier Tir.h

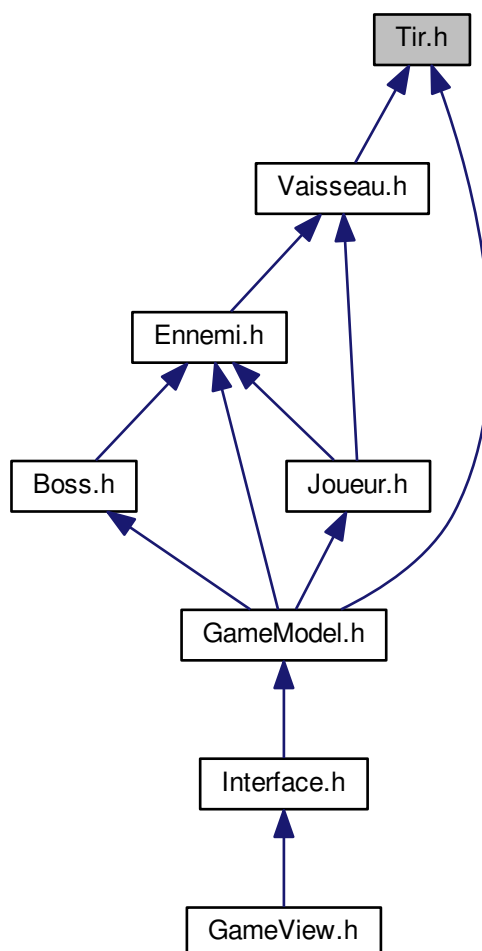
Déclaration de la classe [Tir](#).

```
#include "MovableElement.h"  
#include "TypeTir.h"
```

Graphe des dépendances par inclusion de Tir.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class [Tir](#)
Classe représentant les tirs des [Vaisseau](#).

6.13.1 Description détaillée

Déclaration de la classe [Tir](#).

Auteur

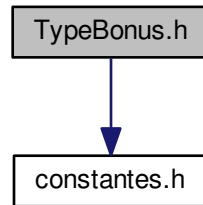
Quentin HARSCOËT

6.14 Référence du fichier TypeBonus.h

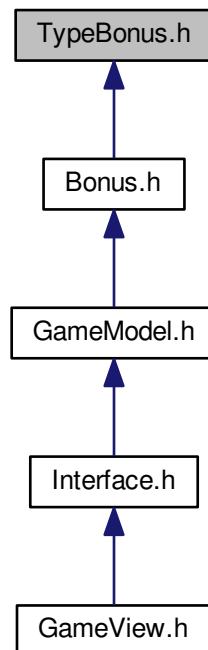
Déclaration de la classe [TypeBonus](#) et de l'énumération [BonusEffect](#).

```
#include "constantes.h"
```

Graphe des dépendances par inclusion de TypeBonus.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class `TypeBonus`
Classe représentant les différents types de `Bonus`.

Énumérations

- enum `BonusEffect` { `POWER`, `LIFE`, `BOMB`, `HP` }

6.14.1 Description détaillée

Déclaration de la classe [TypeBonus](#) et de l'énumération [BonusEffect](#).

Auteur

Quentin HARSCOËT

6.14.2 Documentation du type de l'énumération

6.14.2.1 enum [BonusEffect](#)

Valeurs énumérées

POWER Gain de puissance

LIFE Vie supplémentaire

BOMB Bombe supplémentaire

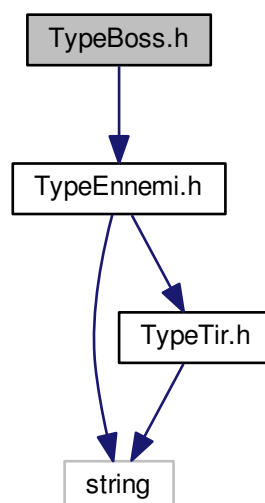
HP Gain de points de vie

6.15 Référence du fichier TypeBoss.h

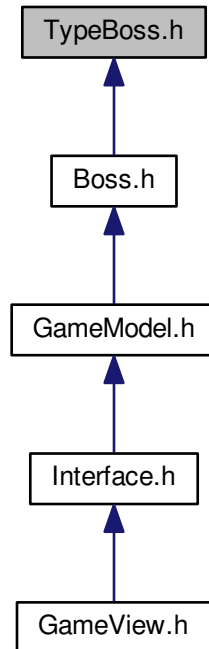
Déclaration de la classe [TypeBoss](#).

```
#include "TypeEnnemi.h"
```

Graphe des dépendances par inclusion de TypeBoss.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class `TypeBoss`
Classe représentant les différents types de `Boss`.

6.15.1 Description détaillée

Déclaration de la classe `TypeBoss`.

Auteur

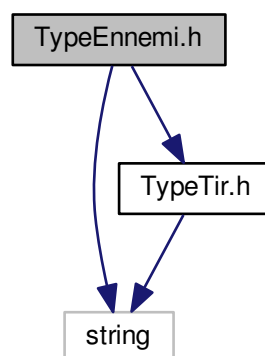
Quentin HARSCOËT

6.16 Référence du fichier TypeEnnemi.h

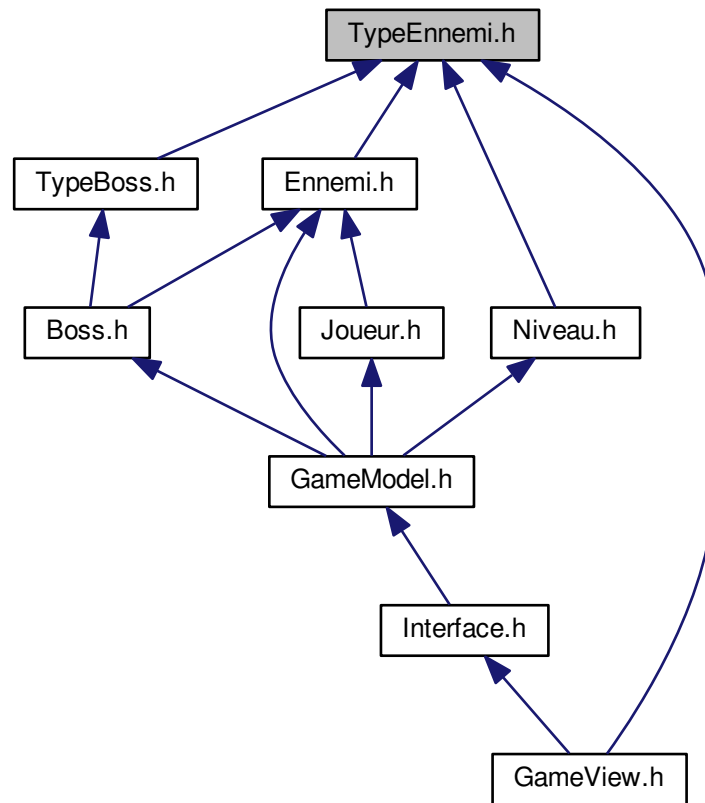
Déclaration de la classe [TypeEnnemi](#) et de l'énumération [ShootPattern](#).

```
#include <string>
#include "TypeTir.h"
```

Graphe des dépendances par inclusion de TypeEnnemi.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class `TypeEnnemi`
Classe représentant les différents types d'`Ennemi`.

Énumérations

- enum `ShootPattern` { `SIMPLE`, `M_TRIPLE`, `TOPLAYER`, `HALFCIRCLE` }

6.16.1 Description détaillée

Déclaration de la classe `TypeEnnemi` et de l'énumération `ShootPattern`.

Auteur

Théo CHASSAIGNE
 Quentin HARSCOËT

6.16.2 Documentation du type de l'énumération

6.16.2.1 enum ShootPattern

Valeurs énumérées

SIMPLE Tir simple en ligne droite

M_TRIPLE Tir triple, "en triangle"

TOPLAYER Tir dirigé vers le Joueur

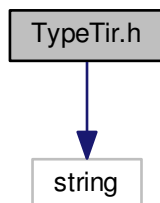
HALFCIRCLE Tir en demi-cercle, devant le Vaisseau

6.17 Référence du fichier TypeTir.h

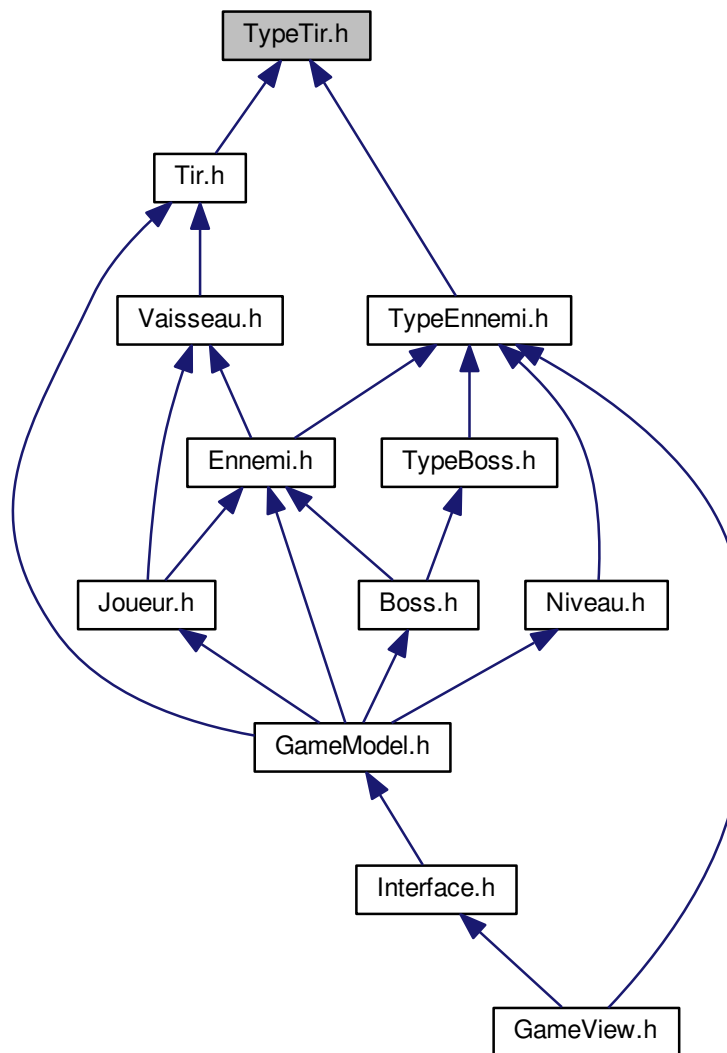
Déclaration de la classe [TypeTir](#).

```
#include <string>
```

Graphe des dépendances par inclusion de TypeTir.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class `TypeTir`
Classe représentant les différents types de `Tir`.

6.17.1 Description détaillée

Déclaration de la classe `TypeTir`.

Auteur

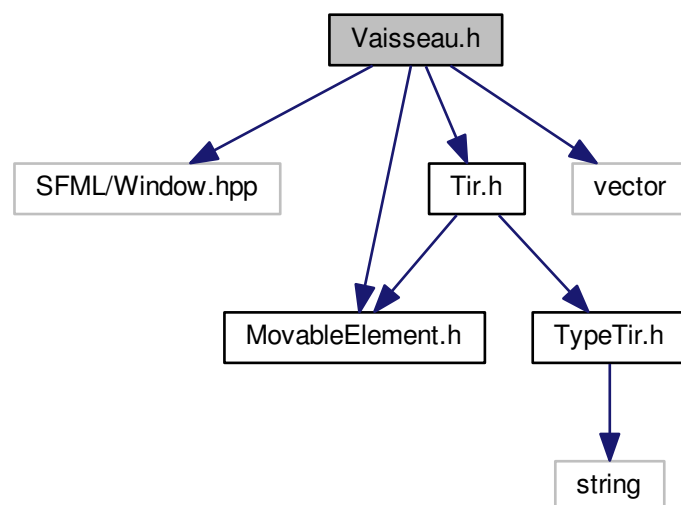
Quentin HARSCOËT

6.18 Référence du fichier Vaisseau.h

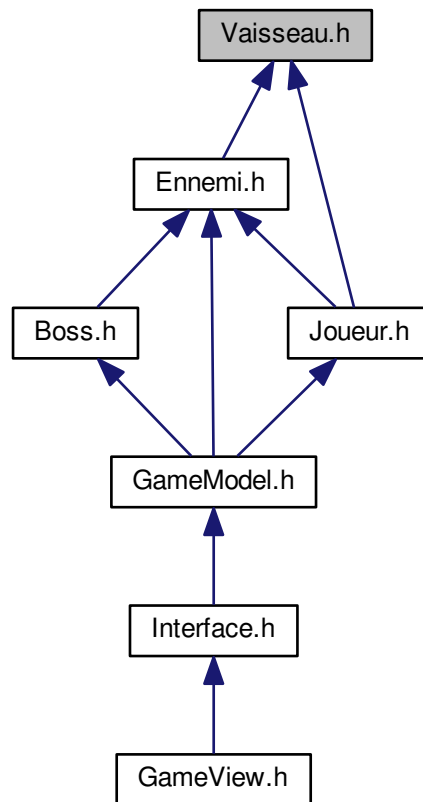
Déclaration de la classe [Vaisseau](#).

```
#include <SFML/Window.hpp>
#include "MovableElement.h"
#include "Tir.h"
#include <vector>
```

Graphe des dépendances par inclusion de Vaisseau.h :



Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :



Classes

- class [Vaisseau](#)
Classe représentant les vaisseaux.

6.18.1 Description détaillée

Déclaration de la classe [Vaisseau](#).

Auteur

Théo CHASSAIGNE
Quentin HARSCOËT

Index

addBonus
 GameModel, 19
addEnemy
 GameModel, 19
addHP
 Joueur, 41
addPower
 Joueur, 41
addTir
 GameModel, 19

BOMB
 TypeBonus.h, 89

BACKTOMENU
 constantes.h, 71

BOMBES
 constantes.h, 71

Bonus, 9
 Bonus, 10
 getType, 10

Bonus.h, 67

BonusEffect, 10
 TypeBonus.h, 89

Boss, 11
 Boss, 12
 changePattern, 12
 getShootPattern, 12
 move, 13

Boss.h, 68

changeNiveau
 GameModel, 20

changePattern
 Boss, 12

constantes.h, 70
 BACKTOMENU, 71
 BOMBES, 71
 DEAD, 71
 DIFFICULTY, 71
 ENEMY_SCALE, 71
 ENTERNAME, 71
 FINISHLEVEL, 72
 GAMEOVER, 72
 HARD, 72
 HIGHSCORES, 72
 JOUER, 72
 LANGUAGE, 72
 LIVESNUMBER, 72
 MODEL_HEIGHT, 72
 MODEL_WIDTH, 72

MYLANGUE, 72
NB_BONUS_TYPES, 72
NB_ENNEMY_TYPES, 73
NB_STRINGS, 73
NB_TIRS_TYPES, 73
NIVEAU, 73
NORMAL, 73
NOSCORE, 73
PAUSE, 73
PUISSANCE, 73
PV, 73
QUIT, 73
RESUME, 73
SETTINGS, 73
SHEEP_HEIGHT, 74
SHEEP_SCALE, 74
SHEEP_SPEED, 74
SHEEP_WIDTH, 74
SOUNDLEVEL, 74
SOUNDPACK, 74
VERYHARD, 74
VIES, 74
VIEW_HEIGHT, 74
VIEW_WIDTH, 74

DEAD
 constantes.h, 71

DIFFICULTY
 constantes.h, 71

draw
 GraphicElement, 34

drawHUD
 Interface, 37

ENEMY_SCALE
 constantes.h, 71

ENTERNAME
 constantes.h, 71

Ennemi, 13
 Ennemi, 15
 getScore, 15
 getShootPattern, 16
 getType, 16
 isShot, 16
 shoot, 17

Ennemi.h, 74
explosion
 GameView, 28

FINISHLEVEL

- constantes.h, 72
- GAMEOVER
 - constantes.h, 72
- GameModel, 18
 - addBonus, 19
 - addEnemy, 19
 - addTir, 19
 - changeNiveau, 20
 - GameModel, 19
 - gameOver, 20
 - GameModel, 19
 - getBossType, 21
 - getClock, 21
 - getDeletedItems, 21
 - getDifficulty, 22
 - getEnemyTypes, 22
 - getLevelID, 22
 - getNextLevel, 22
 - getPlayerShip, 23
 - getScore, 23
 - getShipPos, 23
 - getStringLevelID, 23
 - getStringScore, 24
 - getTirTypes, 24
 - getVectorBonuses, 24
 - getVectorEnemies, 24
 - getVectorTir, 24
 - nextStep, 25
 - removeEnemy, 25, 26
 - saveScore, 26
 - setNextLevel, 26
 - setScore, 26
- GameModel.h, 76
- gameOver
 - GameModel, 20
- GameView, 26
 - explosion, 28
 - GameView, 27
 - GameView, 27
 - options, 28
 - setModel, 28
 - synchronize, 29
 - transition, 30
 - treatEvents, 30
 - viewHighscores, 32
- GameView.h, 77
- getAlive
 - Vaisseau, 62
- getBombs
 - Joueur, 41
- getBossType
 - GameModel, 21
- getChangePatternDelay
 - TypeBoss, 55
- getClock
 - GameModel, 21
 - Vaisseau, 62
- getDY
 - MovableElement, 47
- getDamages
 - TypeTir, 60
- getDeletedItems
 - GameModel, 21
- getDifficulty
 - GameModel, 22
- getEffect
 - TypeBonus, 53
- getEnemyTypes
 - GameModel, 22
- getFilename
 - TypeEnnemi, 57
 - TypeTir, 60
- getFrequency
 - TypeBonus, 53
- getH
 - MovableElement, 47
 - TypeEnnemi, 57
 - TypeTir, 60
- getHP
 - TypeEnnemi, 57
 - Vaisseau, 63
- getID
 - Niveau, 49
- getLevelID
 - GameModel, 22
- getMap
 - Niveau, 49
- getMaxHP
 - Vaisseau, 63
- getNextLevel
 - GameModel, 22
- getNumberOfShoots
 - Joueur, 42
- getPlayer
 - Tir, 52
- getPlayerShip
 - GameModel, 23
- getPower
 - Joueur, 42
- getScale
 - TypeBonus, 53
- getScore
 - Ennemi, 15
 - GameModel, 23
 - TypeEnnemi, 58
- getShipPos
 - GameModel, 23
- getShootPattern
 - Boss, 12
 - Ennemi, 16
 - TypeEnnemi, 58
- getShootingSpeed
 - TypeEnnemi, 58
 - Vaisseau, 63
- getSpeed
 - TypeEnnemi, 58

- TypeTir, 60
- getStatus
 - Horloge, 36
- getStringBombs
 - Joueur, 42
- getStringLevelID
 - GameModel, 23
- getStringPower
 - Joueur, 42
- getStringScore
 - GameModel, 24
- getTime
 - Horloge, 36
- getTirTypes
 - GameModel, 24
- getType
 - Bonus, 10
 - Ennemi, 16
 - Tir, 52
- getTypeTir
 - TypeEnnemi, 58
 - Vaisseau, 63
- getValue
 - TypeBonus, 54
- getVectorBonuses
 - GameModel, 24
- getVectorEnnemis
 - GameModel, 24
- getVectorTir
 - GameModel, 24
- getVies
 - Joueur, 42
- getVisible
 - GraphicElement, 34
- getW
 - MovableElement, 48
 - TypeEnnemi, 58
 - TypeTir, 60
- getX
 - MovableElement, 48
- getY
 - MovableElement, 48
- GraphicElement, 32
 - draw, 34
 - getVisible, 34
 - GraphicElement, 34
 - GraphicElement, 34
 - resize, 34
 - setPosition, 34, 35
 - setVisible, 35
- GraphicElement.h, 78
- HALFCIRCLE
 - TypeEnnemi.h, 93
- HP
 - TypeBonus.h, 89
- HARD
 - constantes.h, 72
- HIGHSCORES
 - constantes.h, 72
- Horloge, 35
 - getStatus, 36
 - getTime, 36
 - stop, 36
- Horloge.h, 79
- initNiveau
 - Niveau, 50
- Interface, 36
 - drawHUD, 37
 - update, 37
 - updateBossPV, 38
- Interface.h, 80
- isHitByEnemy
 - Joueur, 42
- isShot
 - Ennemi, 16
 - Joueur, 44
 - Vaisseau, 63
- JOUER
 - constantes.h, 72
- Joueur, 39
 - addHP, 41
 - addPower, 41
 - getBombs, 41
 - getNumberOfShoots, 42
 - getPower, 42
 - getStringBombs, 42
 - getStringPower, 42
 - getVies, 42
 - isHitByEnemy, 42
 - isShot, 44
 - Joueur, 40
 - shoot, 44
 - useBomb, 45
- Joueur.h, 82
- LIFE
 - TypeBonus.h, 89
- LANGUAGE
 - constantes.h, 72
- LIVESNUMBER
 - constantes.h, 72
- M_TRIPLE
 - TypeEnnemi.h, 93
- MODEL_HEIGHT
 - constantes.h, 72
- MODEL_WIDTH
 - constantes.h, 72
- MYLANGUE
 - constantes.h, 72
- MovableElement, 46
 - getDY, 47
 - getH, 47
 - getW, 48
 - getX, 48

- getY, 48
- MovableElement, 47
- MovableElement, 47
- move, 48
- my_dx, 48
- my_dy, 48
- my_h, 48
- my_w, 48
- my_x, 49
- my_y, 49
- MovableElement.h, 83
- move
 - Boss, 13
 - MovableElement, 48
- my_clock
 - Vaisseau, 64
- my_dx
 - MovableElement, 48
- my_dy
 - MovableElement, 48
- my_h
 - MovableElement, 48
- my_shootingSpeed
 - Vaisseau, 64
- my_typeTir
 - Vaisseau, 64
- my_w
 - MovableElement, 48
- my_x
 - MovableElement, 49
- my_y
 - MovableElement, 49
- NB_BONUS_TYPES
 - constantes.h, 72
- NB_ENNEMY_TYPES
 - constantes.h, 73
- NB_STRINGS
 - constantes.h, 73
- NB_TIRS_TYPES
 - constantes.h, 73
- NIVEAU
 - constantes.h, 73
- NORMAL
 - constantes.h, 73
- NOSCORE
 - constantes.h, 73
- nextStep
 - GameModel, 25
- Niveau, 49
 - getID, 49
 - getMap, 49
 - initNiveau, 50
- Niveau.h, 85
- options
 - GameView, 28
- POWER
 - TypeBonus.h, 89
- PAUSE
 - constantes.h, 73
- PUISSANCE
 - constantes.h, 73
- PV
 - constantes.h, 73
- QUIT
 - constantes.h, 73
- RESUME
 - constantes.h, 73
- removeEnemy
 - GameModel, 25, 26
- resize
 - GraphicElement, 34
- SIMPLE
 - TypeEnnemi.h, 93
- SETTINGS
 - constantes.h, 73
- SHEEP_HEIGHT
 - constantes.h, 74
- SHEEP_SCALE
 - constantes.h, 74
- SHEEP_SPEED
 - constantes.h, 74
- SHEEP_WIDTH
 - constantes.h, 74
- SOUNDLEVEL
 - constantes.h, 74
- SOUNDPACK
 - constantes.h, 74
- saveScore
 - GameModel, 26
- setAlive
 - Vaisseau, 64
- setHP
 - Vaisseau, 64
- setModel
 - GameView, 28
- setNextLevel
 - GameModel, 26
- setPosition
 - GraphicElement, 34, 35
- setScore
 - GameModel, 26
- setTypeTir
 - Vaisseau, 64
- setVisible
 - GraphicElement, 35
- shoot
 - Ennemi, 17
 - Joueur, 44
 - Vaisseau, 64
- ShootPattern, 50
 - TypeEnnemi.h, 92
- stop

- Horloge, 36
- synchronize
 - GameView, 29
- TOPLAYER
 - TypeEnnemi.h, 93
- Tir, 50
 - getPlayer, 52
 - getType, 52
 - Tir, 52
- Tir.h, 86
- transition
 - GameView, 30
- treatEvents
 - GameView, 30
- TypeBonus.h
 - BOMB, 89
 - HP, 89
 - LIFE, 89
 - POWER, 89
- TypeEnnemi.h
 - HALFCIRCLE, 93
 - M_TRIPLE, 93
 - SIMPLE, 93
 - TOPLAYER, 93
- TypeBonus, 52
 - getEffect, 53
 - getFrequency, 53
 - getScale, 53
 - getValue, 54
 - TypeBonus, 53
 - TypeBonus, 53
- TypeBonus.h, 87
 - BonusEffect, 89
- TypeBoss, 54
 - getChangePatternDelay, 55
 - TypeBoss, 55
 - TypeBoss, 55
- TypeBoss.h, 89
- TypeEnnemi, 56
 - getFilename, 57
 - getH, 57
 - getHP, 57
 - getScore, 58
 - getShootPattern, 58
 - getShootingSpeed, 58
 - getSpeed, 58
 - getTypeTir, 58
 - getW, 58
 - TypeEnnemi, 57
 - TypeEnnemi, 57
- TypeEnnemi.h, 91
 - ShootPattern, 92
- TypeTir, 59
 - getDamages, 60
 - getFilename, 60
 - getH, 60
 - getSpeed, 60
 - getW, 60
 - TypeTir, 59
 - TypeTir, 59
 - TypeTir.h, 93
- update
 - Interface, 37
- updateBossPV
 - Interface, 38
- useBomb
 - Joueur, 45
- VERYHARD
 - constantes.h, 74
- VIES
 - constantes.h, 74
- VIEW_HEIGHT
 - constantes.h, 74
- VIEW_WIDTH
 - constantes.h, 74
- Vaisseau, 61
 - getAlive, 62
 - getClock, 62
 - getHP, 63
 - getMaxHP, 63
 - getShootingSpeed, 63
 - getTypeTir, 63
 - isShot, 63
 - my_clock, 64
 - my_shootingSpeed, 64
 - my_typeTir, 64
 - setAlive, 64
 - setHP, 64
 - setTypeTir, 64
 - shoot, 64
 - Vaisseau, 62
- Vaisseau.h, 95
- viewHighscores
 - GameView, 32