

## Projet POA

Généré par Doxygen 1.8.13



# Table des matières

<b>1</b>	<b>Index hiérarchique</b>	<b>1</b>
1.1	Hiérarchie des classes . . . . .	1
<b>2</b>	<b>Index des classes</b>	<b>3</b>
2.1	Liste des classes . . . . .	3
<b>3</b>	<b>Index des fichiers</b>	<b>5</b>
3.1	Liste des fichiers . . . . .	5
<b>4</b>	<b>Documentation des classes</b>	<b>7</b>
4.1	Référence de la structure Box . . . . .	7
4.1.1	Documentation des données membres . . . . .	7
4.1.1.1	_ntex . . . . .	7
4.1.1.2	_x . . . . .	7
4.1.1.3	_y . . . . .	7
4.2	Référence de la classe Chasseur . . . . .	8
4.2.1	Documentation des constructeurs et destructeur . . . . .	8
4.2.1.1	Chasseur() . . . . .	9
4.2.2	Documentation des fonctions membres . . . . .	9
4.2.2.1	die() . . . . .	9
4.2.2.2	fire() . . . . .	9
4.2.2.3	hurt() . . . . .	9
4.2.2.4	move() . . . . .	9
4.2.2.5	move_aux() . . . . .	10
4.2.2.6	process_fireball() . . . . .	10

4.2.2.7	<code>right_click()</code>	10
4.2.2.8	<code>update()</code>	10
4.2.3	Documentation des données membres	11
4.2.3.1	<code>_hunter_fire</code>	11
4.2.3.2	<code>_hunter_hit</code>	11
4.2.3.3	<code>_pv</code>	11
4.2.3.4	<code>_wall_hit</code>	11
4.2.3.5	<code>l</code>	11
4.2.3.6	<code>perte_precision</code>	12
4.3	Référence de la structure <code>coord</code>	12
4.3.1	Description détaillée	12
4.3.2	Documentation des données membres	12
4.3.2.1	<code>x</code>	12
4.3.2.2	<code>y</code>	12
4.4	Référence de la classe <code>Environnement</code>	13
4.4.1	Documentation des constructeurs et destructeur	13
4.4.1.1	<code>~Environnement()</code>	13
4.4.2	Documentation des fonctions membres	13
4.4.2.1	<code>data()</code>	14
4.4.2.2	<code>height()</code>	14
4.4.2.3	<code>init()</code>	14
4.4.2.4	<code>make_fireballs()</code>	14
4.4.2.5	<code>reconfigure()</code>	14
4.4.2.6	<code>wall_texture()</code>	14
4.4.2.7	<code>width()</code>	15
4.4.3	Documentation des données membres	15
4.4.3.1	<code>_boxes</code>	15
4.4.3.2	<code>_guards</code>	15
4.4.3.3	<code>_nboxes</code>	15
4.4.3.4	<code>_nguards</code>	15

4.4.3.5	<code>_npicts</code>	15
4.4.3.6	<code>_nwall</code>	15
4.4.3.7	<code>_picts</code>	16
4.4.3.8	<code>_treasor</code>	16
4.4.3.9	<code>_walls</code>	16
4.4.3.10	<code>modele_dir</code>	16
4.4.3.11	<code>scale</code>	16
4.4.3.12	<code>texture_dir</code>	16
4.5	Référence de la classe <code>FireBall</code>	16
4.5.1	Documentation des constructeurs et destructeur	17
4.5.1.1	<code>FireBall()</code>	17
4.5.2	Documentation des fonctions membres	17
4.5.2.1	<code>display()</code>	17
4.5.2.2	<code>explode()</code>	18
4.5.2.3	<code>get_x()</code>	18
4.5.2.4	<code>get_y()</code>	18
4.5.2.5	<code>init()</code>	18
4.5.2.6	<code>move_step()</code>	18
4.5.2.7	<code>set_orig_size()</code>	18
4.5.3	Documentation des données membres	18
4.5.3.1	<code>_angle</code>	19
4.5.3.2	<code>_angle2</code>	19
4.5.3.3	<code>_azimuth</code>	19
4.5.3.4	<code>_fire_list</code>	19
4.5.3.5	<code>_fire_texture</code>	19
4.5.3.6	<code>_move_angle</code>	19
4.5.3.7	<code>_orig_size</code>	19
4.5.3.8	<code>_owner</code>	19
4.5.3.9	<code>_size</code>	20
4.5.3.10	<code>_state</code>	20

4.5.3.11	<code>_x</code>	20
4.5.3.12	<code>_y</code>	20
4.5.3.13	<code>_z</code>	20
4.6	Référence de la classe Gardien	20
4.6.1	Description détaillée	21
4.6.2	Documentation des constructeurs et destructeur	22
4.6.2.1	<code>Gardien()</code>	22
4.6.2.2	<code>~Gardien()</code>	22
4.6.3	Documentation des fonctions membres	22
4.6.3.1	<code>avancer()</code>	22
4.6.3.2	<code>can_see_player()</code>	22
4.6.3.3	<code>die()</code>	22
4.6.3.4	<code>fire()</code>	23
4.6.3.5	<code>hurt()</code>	23
4.6.3.6	<code>is_legit_move()</code>	23
4.6.3.7	<code>move()</code>	23
4.6.3.8	<code>move_to_treasure()</code>	24
4.6.3.9	<code>process_fireball()</code>	24
4.6.3.10	<code>try_move()</code>	24
4.6.3.11	<code>update()</code>	24
4.6.4	Documentation des données membres	24
4.6.4.1	<code>_pv</code>	25
4.6.4.2	<code>BASE_PV</code>	25
4.6.4.3	<code>dead</code>	25
4.6.4.4	<code>fired</code>	25
4.6.4.5	<code>l</code>	25
4.6.4.6	<code>perte_precision</code>	25
4.6.4.7	<code>reload</code>	26
4.6.4.8	<code>RELOAD_TIME</code>	26
4.6.4.9	<code>reloading</code>	26

4.7	Référence de la classe Labyrinthe . . . . .	26
4.7.1	Documentation des constructeurs et destructeur . . . . .	27
4.7.1.1	Labyrinthe() [1/2] . . . . .	27
4.7.1.2	Labyrinthe() [2/2] . . . . .	27
4.7.2	Documentation des fonctions membres . . . . .	27
4.7.2.1	build_boxes() . . . . .	27
4.7.2.2	build_guards() . . . . .	28
4.7.2.3	build_text() . . . . .	28
4.7.2.4	build_treasure() . . . . .	28
4.7.2.5	build_walls() . . . . .	29
4.7.2.6	data() . . . . .	29
4.7.2.7	dist_of_treasure() . . . . .	29
4.7.2.8	fill_dist() . . . . .	30
4.7.2.9	height() . . . . .	30
4.7.2.10	hurt_gardien_at() . . . . .	30
4.7.2.11	hurt_joueur() . . . . .	30
4.7.2.12	iamdying() . . . . .	31
4.7.2.13	init_data() . . . . .	31
4.7.2.14	init_vector_dist() . . . . .	31
4.7.2.15	nb_alive() . . . . .	31
4.7.2.16	set_data() . . . . .	31
4.7.2.17	width() . . . . .	32
4.7.3	Documentation des données membres . . . . .	32
4.7.3.1	_data . . . . .	32
4.7.3.2	_dist_vect . . . . .	32
4.7.3.3	_nb_alive . . . . .	32
4.7.3.4	lab_height . . . . .	33
4.7.3.5	lab_width . . . . .	33
4.8	Référence de la classe Mover . . . . .	33
4.8.1	Documentation des constructeurs et destructeur . . . . .	34

4.8.1.1	Mover()	34
4.8.1.2	~Mover()	34
4.8.2	Documentation des fonctions membres	34
4.8.2.1	fire()	34
4.8.2.2	init()	34
4.8.2.3	move()	34
4.8.2.4	process_fireball()	35
4.8.2.5	rester_au_sol()	35
4.8.2.6	right_click()	35
4.8.2.7	tomber()	35
4.8.2.8	update()	35
4.8.3	Documentation des données membres	35
4.8.3.1	_angle	35
4.8.3.2	_fb	36
4.8.3.3	_l	36
4.8.3.4	_model	36
4.8.3.5	_x	36
4.8.3.6	_y	36
4.9	Référence de la structure node	36
4.9.1	Documentation des données membres	36
4.9.1.1	x	37
4.9.1.2	y	37
4.10	Référence de la structure node_comparator	37
4.10.1	Documentation des fonctions membres	37
4.10.1.1	operator()	37
4.11	Référence de la classe Sound	37
4.11.1	Documentation des constructeurs et destructeur	38
4.11.1.1	Sound()	38
4.11.1.2	~Sound()	38
4.11.2	Documentation des fonctions membres	38



4.11.2.1	init()	38
4.11.2.2	play()	38
4.11.3	Documentation des données membres	38
4.11.3.1	_channel	39
4.11.3.2	_nsounds	39
4.11.3.3	_sound	39
4.11.3.4	_system	39
4.12	Référence de la structure Wall	39
4.12.1	Documentation des données membres	39
4.12.1.1	_ntex	39
4.12.1.2	_x1	40
4.12.1.3	_x2	40
4.12.1.4	_y1	40
4.12.1.5	_y2	40
<b>5</b>	<b>Documentation des fichiers</b>	<b>41</b>
5.1	Référence du fichier Chasseur.cc	41
5.2	Référence du fichier Chasseur.h	41
5.3	Référence du fichier Environnement.h	41
5.3.1	Documentation des macros	42
5.3.1.1	EMPTY	42
5.3.2	Documentation des fonctions	42
5.3.2.1	message()	42
5.3.2.2	partie_terminee()	42
5.4	Référence du fichier FireBall.h	42
5.4.1	Documentation des macros	43
5.4.1.1	M_PI	43
5.4.2	Documentation du type de l'énumération	43
5.4.2.1	FBstat	43
5.5	Référence du fichier Gardien.cc	43
5.6	Référence du fichier Gardien.h	43

5.7	Référence du fichier laby2.c . . . . .	44
5.8	Référence du fichier Labyrinthe.cc . . . . .	44
5.8.1	Documentation des fonctions . . . . .	44
5.8.1.1	add_to_wall_list() . . . . .	44
5.8.1.2	build_map() . . . . .	45
5.8.1.3	first_char() . . . . .	45
5.8.1.4	is_blank() . . . . .	45
5.8.1.5	parse_map() . . . . .	46
5.8.1.6	parse_wall_extr() . . . . .	46
5.9	Référence du fichier Labyrinthe.h . . . . .	46
5.9.1	Documentation des fonctions . . . . .	47
5.9.1.1	add_to_wall_list() . . . . .	48
5.9.1.2	build_map() . . . . .	48
5.9.1.3	first_char() . . . . .	48
5.9.1.4	is_blank() . . . . .	48
5.9.1.5	parse_map() . . . . .	49
5.9.1.6	parse_wall_extr() . . . . .	49
5.9.2	Documentation des variables . . . . .	50
5.9.2.1	BOX . . . . .	50
5.9.2.2	GARDE . . . . .	50
5.9.2.3	HORIZONTAL . . . . .	50
5.9.2.4	JOUEUR . . . . .	50
5.9.2.5	TREASURE . . . . .	50
5.9.2.6	VERTICAL . . . . .	51
5.9.2.7	WALL . . . . .	51
5.10	Référence du fichier Mover.h . . . . .	51
5.11	Référence du fichier old_gardien.cc . . . . .	51
5.11.1	Documentation des fonctions . . . . .	52
5.11.1.1	build_solution() . . . . .	52
5.11.1.2	estimation_dist() . . . . .	52
5.11.1.3	get() . . . . .	52
5.11.1.4	get_shortest_path() . . . . .	52
5.11.1.5	operator==( ) . . . . .	52
5.11.2	Documentation des variables . . . . .	52
5.11.2.1	MAX . . . . .	53
5.11.2.2	MIN . . . . .	53
5.12	Référence du fichier Sound.h . . . . .	53

# Chapitre 1

## Index hiérarchique

### 1.1 Hiérarchie des classes

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

Box . . . . .	7
coord . . . . .	12
Environnement . . . . .	13
Labyrinthe . . . . .	26
FireBall . . . . .	16
Mover . . . . .	33
Chasseur . . . . .	8
Gardien . . . . .	20
node . . . . .	36
node_comparator . . . . .	37
Sound . . . . .	37
Wall . . . . .	39



## Chapitre 2

# Index des classes

### 2.1 Liste des classes

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

Box	7
Chasseur	8
coord	
Structure représentant une coordonnée du labyrinthe	12
Environnement	13
FireBall	16
Gardien	
Classe implémentant les gardes	20
Labyrinthe	26
Mover	33
node	36
node_comparator	37
Sound	37
Wall	39



## Chapitre 3

# Index des fichiers

### 3.1 Liste des fichiers

Liste de tous les fichiers avec une brève description :

Chasseur.cc . . . . .	41
Chasseur.h . . . . .	41
Environnement.h . . . . .	41
FireBall.h . . . . .	42
Gardien.cc . . . . .	43
Gardien.h . . . . .	43
laby2.c . . . . .	44
Labyrinthe.cc . . . . .	44
Labyrinthe.h . . . . .	46
Mover.h . . . . .	51
old_gardien.cc . . . . .	51
Sound.h . . . . .	53





## Chapitre 4

# Documentation des classes

### 4.1 Référence de la structure Box

```
#include <Environnement.h>
```

#### Attributs publics

- int [\\_x](#)
- int [\\_y](#)
- int [\\_ntex](#)

#### 4.1.1 Documentation des données membres

##### 4.1.1.1 [\\_ntex](#)

```
int Box::_ntex
```

##### 4.1.1.2 [\\_x](#)

```
int Box::_x
```

##### 4.1.1.3 [\\_y](#)

```
int Box::_y
```

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

- [Environnement.h](#)

## 4.2 Référence de la classe Chasseur

```
#include <Chasseur.h>
```

Graphe d'héritage de Chasseur :

Graphe de collaboration de Chasseur :

### Fonctions membres publiques

- [Chasseur](#) ([Labyrinthe](#) \*l)  
*constructeur de chasseur, en prenant en param un pointeur de [Labyrinthe](#)*
- void [hurt](#) ()  
*fonction indiquant au chasseur qu'il a été touché*
- bool [move](#) (double dx, double dy)  
*fonction de mouvement du chasseur*
- void [update](#) (void)
- bool [process\\_fireball](#) (float dx, float dy)  
*fonction permettant le déplacement des boules de feu*
- void [fire](#) (int angle\_vertical)  
*fonction de tir. Crée une boule de feu et initie son mouvement*
- void [right\\_click](#) (bool shift, bool control)

### Attributs publics statiques

- static [Sound](#) \* [\\_hunter\\_fire](#)  
*bruit de l'arme du chasseur.*
- static [Sound](#) \* [\\_hunter\\_hit](#)  
*cri du chasseur touché.*
- static [Sound](#) \* [\\_wall\\_hit](#)  
*on a tapé un mur.*

### Fonctions membres privées

- bool [move\\_aux](#) (double dx, double dy)  
*accepte ou non un déplacement*
- void [die](#) ()  
*fonction indiquant au chasseur qu'il est mort*

### Attributs privés

- int [\\_pv](#) = 10  
*nombre de points de vie du chasseur*
- [Labyrinthe](#) \* l  
*[Labyrinthe](#) correspondant à l'environnement.*
- int [perte\\_precision](#) = 0  
*perte de précision du chasseur*

### Membres hérités additionnels

#### 4.2.1 Documentation des constructeurs et destructeur

#### 4.2.1.1 Chasseur()

```
Chasseur::Chasseur (
    Labyrinthe * l )
```

constructeur de chasseur, en prenant en param un pointeur de [Labyrinthe](#)

### 4.2.2 Documentation des fonctions membres

#### 4.2.2.1 die()

```
void Chasseur::die ( ) [private]
```

fonction indiquant au chasseur qu'il est mort

Va simplement entraîner la fin de la partie

#### 4.2.2.2 fire()

```
void Chasseur::fire (
    int angle_vertical ) [virtual]
```

fonction de tir. Crée une boule de feu et initie son mouvement

##### Paramètres

<i>angle_vertical</i>	l'angle vertical
-----------------------	------------------

Implémente [Mover](#).

#### 4.2.2.3 hurt()

```
void Chasseur::hurt ( )
```

fonction indiquant au chasseur qu'il a été touché

Va baisser son nombre de points de vie et sa précision

#### 4.2.2.4 move()

```
bool Chasseur::move (
    double dx,
    double dy ) [inline], [virtual]
```

fonction de mouvement du chasseur

Implémente [Mover](#).

#### 4.2.2.5 move\_aux()

```
bool Chasseur::move_aux (
    double dx,
    double dy ) [private]
```

accepte ou non un déplacement

#### 4.2.2.6 process\_fireball()

```
bool Chasseur::process_fireball (
    float dx,
    float dy ) [virtual]
```

fonction permettant le déplacement des boules de feu

##### Paramètres

<i>dx</i>	le déplacement de la boule de feu sur l'axe x
<i>dy</i>	le déplacement de la boule de feu sur l'axe y

##### Renvoie

true si la boule de feu n'a pas explosé, false si la boule de feu a explosé (par exemple à la rencontre d'un mur)

Implémente [Mover](#).

#### 4.2.2.7 right\_click()

```
void Chasseur::right_click (
    bool shift,
    bool control ) [virtual]
```

Réimplémentée à partir de [Mover](#).

#### 4.2.2.8 update()

```
void Chasseur::update (
    void ) [inline], [virtual]
```

Implémente [Mover](#).

### 4.2.3 Documentation des données membres

#### 4.2.3.1 `_hunter_fire`

```
Sound * Chasseur::_hunter_fire [static]
```

bruit de l'arme du chasseur.

#### 4.2.3.2 `_hunter_hit`

```
Sound * Chasseur::_hunter_hit [static]
```

cri du chasseur touché.

#### 4.2.3.3 `_pv`

```
int Chasseur::_pv = 10 [private]
```

nombre de points de vie du chasseur

#### 4.2.3.4 `_wall_hit`

```
Sound * Chasseur::_wall_hit [static]
```

on a tapé un mur.

#### 4.2.3.5 `l`

```
Labyrinthe* Chasseur::l [private]
```

`Labyrinthe` correspondant à l'environnement.

#### 4.2.3.6 perte\_precision

```
int Chasseur::perte_precision = 0 [private]
```

perte de précision du chasseur

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

- [Chasseur.h](#)
- [Chasseur.cc](#)
- [Labyrinthe.cc](#)

### 4.3 Référence de la structure coord

Structure représentant une coordonnée du labyrinthe.

```
#include <Labyrinthe.h>
```

#### Attributs publics

- int [x](#)
- int [y](#)

#### 4.3.1 Description détaillée

Structure représentant une coordonnée du labyrinthe.

#### 4.3.2 Documentation des données membres

##### 4.3.2.1 x

```
int coord::x
```

##### 4.3.2.2 y

```
int coord::y
```

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

- [Labyrinthe.h](#)

## 4.4 Référence de la classe Environnement

```
#include <Environnement.h>
```

Graphe d'héritage de Environnement :

Graphe de collaboration de Environnement :

### Fonctions membres publiques

- virtual int `width` ()=0
- virtual int `height` ()=0
- virtual char `data` (int i, int j)=0
- virtual `~Environnement` ()
- void `reconfigure` ()
- int `wall_texture` (char \*)
- void `make_fireballs` (void)

### Fonctions membres publiques statiques

- static `Environnement` \* `init` (char \*filename)

### Attributs publics

- `Wall` \* `_walls`
- int `_nwall`
- `Wall` \* `_picts`
- int `_npicts`
- `Box` \* `_boxes`
- int `_nboxes`
- `Box` `_treasure`
- `Mover` \*\* `_guards`
- int `_nguards`

### Attributs publics statiques

- static const int `scale`
- static const char \* `texture_dir`
- static const char \* `modele_dir`

#### 4.4.1 Documentation des constructeurs et destructeur

##### 4.4.1.1 `~Environnement()`

```
virtual Environnement::~~Environnement ( ) [inline], [virtual]
```

#### 4.4.2 Documentation des fonctions membres

#### 4.4.2.1 data()

```
virtual char Environnement::data (
    int i,
    int j ) [pure virtual]
```

Implémenté dans [Labyrinthe](#).

#### 4.4.2.2 height()

```
virtual int Environnement::height ( ) [pure virtual]
```

Implémenté dans [Labyrinthe](#).

#### 4.4.2.3 init()

```
Environnement * Environnement::init (
    char * filename ) [static]
```

#### 4.4.2.4 make\_fireballs()

```
void Environnement::make_fireballs (
    void )
```

#### 4.4.2.5 reconfigure()

```
void Environnement::reconfigure ( )
```

#### 4.4.2.6 wall\_texture()

```
int Environnement::wall_texture (
    char * )
```



#### 4.4.2.7 width()

```
virtual int Environnement::width ( ) [pure virtual]
```

Implémenté dans [Labyrinthe](#).

### 4.4.3 Documentation des données membres

#### 4.4.3.1 \_boxes

```
Box* Environnement::_boxes
```

#### 4.4.3.2 \_guards

```
Mover** Environnement::_guards
```

#### 4.4.3.3 \_nboxes

```
int Environnement::_nboxes
```

#### 4.4.3.4 \_nguards

```
int Environnement::_nguards
```

#### 4.4.3.5 \_npicts

```
int Environnement::_npicts
```

#### 4.4.3.6 \_nwall

```
int Environnement::_nwall
```

#### 4.4.3.7 `_picts`

```
Wall* Environnement::_picts
```

#### 4.4.3.8 `_treasor`

```
Box Environnement::_treasor
```

#### 4.4.3.9 `_walls`

```
Wall* Environnement::_walls
```

#### 4.4.3.10 `modele_dir`

```
const char* Environnement::modele_dir [static]
```

#### 4.4.3.11 `scale`

```
const int Environnement::scale [static]
```

#### 4.4.3.12 `texture_dir`

```
const char* Environnement::texture_dir [static]
```

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

- [Environnement.h](#)
- [Labyrinthe.cc](#)

## 4.5 Référence de la classe FireBall

```
#include <FireBall.h>
```

Graphe de collaboration de FireBall :

### Fonctions membres publiques

- `FireBall` (float size, unsigned int tex, `Mover` \*)
- void `display` ()
- void `move_step` ()
- void `init` (float x, float y, float z, int angle\_vertical, int angle\_horizontal)
- void `set_orig_size` (float size)
- float `get_x` ()
- float `get_y` ()

### Fonctions membres privées

- void `explode` ()

### Attributs privés

- float `_x`
- float `_y`
- float `_z`
- float `_size`
- float `_orig_size`
- int `_angle`
- int `_angle2`
- unsigned int `_fire_texture`
- int `_move_angle`
- int `_azimuth`
- `Mover` \* `_owner`
- `FBstat` `_state`

### Attributs privés statiques

- static unsigned int `_fire_list`

## 4.5.1 Documentation des constructeurs et destructeur

### 4.5.1.1 `FireBall()`

```
FireBall::FireBall (
    float size,
    unsigned int tex,
    Mover * )
```

## 4.5.2 Documentation des fonctions membres

### 4.5.2.1 `display()`

```
void FireBall::display ( )
```

#### 4.5.2.2 explode()

```
void FireBall::explode ( ) [private]
```

#### 4.5.2.3 get\_x()

```
float FireBall::get_x ( ) [inline]
```

#### 4.5.2.4 get\_y()

```
float FireBall::get_y ( ) [inline]
```

#### 4.5.2.5 init()

```
void FireBall::init (
    float x,
    float y,
    float z,
    int angle_vertical,
    int angle_horizontal )
```

#### 4.5.2.6 move\_step()

```
void FireBall::move_step ( )
```

#### 4.5.2.7 set\_orig\_size()

```
void FireBall::set_orig_size (
    float size ) [inline]
```

### 4.5.3 Documentation des données membres

#### 4.5.3.1 \_angle

```
int FireBall::_angle [private]
```

#### 4.5.3.2 \_angle2

```
int FireBall::_angle2 [private]
```

#### 4.5.3.3 \_azimuth

```
int FireBall::_azimuth [private]
```

#### 4.5.3.4 \_fire\_list

```
unsigned int FireBall::_fire_list [static], [private]
```

#### 4.5.3.5 \_fire\_texture

```
unsigned int FireBall::_fire_texture [private]
```

#### 4.5.3.6 \_move\_angle

```
int FireBall::_move_angle [private]
```

#### 4.5.3.7 \_orig\_size

```
float FireBall::_orig_size [private]
```

#### 4.5.3.8 \_owner

```
Mover* FireBall::_owner [private]
```

#### 4.5.3.9 `_size`

```
float FireBall::_size [private]
```

#### 4.5.3.10 `_state`

```
FBstat FireBall::_state [private]
```

#### 4.5.3.11 `_x`

```
float FireBall::_x [private]
```

#### 4.5.3.12 `_y`

```
float FireBall::_y [private]
```

#### 4.5.3.13 `_z`

```
float FireBall::_z [private]
```

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

— [FireBall.h](#)

## 4.6 Référence de la classe Gardien

Classe implémentant les gardes.

```
#include <Gardien.h>
```

Graphe d'héritage de Gardien :

Graphe de collaboration de Gardien :

## Fonctions membres publiques

- `Gardien` (`Labyrinthe *laby`, `const char *modele`)
- `~Gardien` ()
- `void update` (void)
- `void hurt` ()  
*fonction indiquant au gardien qu'il a été touché par une boule de feu provenant du joueur. Baissera sa précision et son nombre de points de vie*
- `bool move` (double `dx`, double `dy`)  
*Tente d'accomplir le mouvement en paramètre. Meme principe que `try_move`, mais va faire en sorte de glisser sur les murs.*
- `void fire` (int `angle_vertical`)  
*initialisera une boule de feu, qui sera envoyée*
- `bool process_fireball` (float `dx`, float `dy`)  
*Gère le déplacement de la boule de feu venant du garde.*

## Fonctions membres privées

- `bool can_see_player` ()  
*indique si le gardien peut voir le joueur (ie s'il n'y a ni mur ni obstacle entre les deux).*
- `bool is_legit_move` (double `dx`, double `dy`)  
*teste si le mouvement de coordonnées (`dx`, `dy`) est acceptable, id est que ça implique pas de marcher au travers d'un mur*
- `bool try_move` (double `dx`, double `dy`)  
*tente d'accomplir le mouvement en paramètre*
- `bool avancer` ()  
*fait marcher le gardien vers l'avant*
- `bool move_to_treasure` ()  
*fait en sorte que le gardien aille vers le trésor*
- `void die` ()  
*fonction appelée lors de la mort du gardien*

## Attributs privés

- `bool fired` = false  
*Indique si une boule de feu a déjà été tirée. Si c'est le cas, il est impossible pour le gardien d'en tirer une autre.*
- `bool reloading` = false  
*indique si le gardien est en train de recharger (si c'est le cas, il ne peut pas tirer)*
- `int reload` = 0  
*indique le statut du rechargement*
- `int perte_precision` = 5  
*indique la perte de précision, en degrés, du gardien*
- `bool dead` = false  
*indique si le gardien est mort*
- `Labyrinthe *l`  
*pointeur vers le labyrinthe*
- `int _pv` = `BASE_PV`  
*Nombre de PV restants au gardien.*

## Attributs privés statiques

- `static const int RELOAD_TIME` = 100  
*Temps de rechargement pour les gardiens (en nombre d'appels de `update()`)*
- `static const int BASE_PV` = 10  
*Nombre de PV de base du gardien.*

## Membres hérités additionnels

### 4.6.1 Description détaillée

Classe implémentant les gardes.

## 4.6.2 Documentation des constructeurs et destructeur

### 4.6.2.1 Gardien()

```
Gardien::Gardien (
    Labyrinthe * laby,
    const char * modele ) [inline]
```

### 4.6.2.2 ~Gardien()

```
Gardien::~~Gardien ( ) [inline]
```

## 4.6.3 Documentation des fonctions membres

### 4.6.3.1 avancer()

```
bool Gardien::avancer ( ) [private]
```

fait marcher le gardien vers l'avant

### 4.6.3.2 can\_see\_player()

```
bool Gardien::can_see_player ( ) [private]
```

indique si le gardien peut voir le joueur (ie s'il n'y a ni mur ni obstacle entre les deux).

#### Renvoie

true si pas d'obstacle entre le garde et le joueur, false sinon

### 4.6.3.3 die()

```
void Gardien::die ( ) [private]
```

fonction appelée lors de la mort du gardien



#### 4.6.3.4 fire()

```
void Gardien::fire (
    int angle_vertical ) [virtual]
```

initialisera une boule de feu, qui sera envoyée

Implémente [Mover](#).

#### 4.6.3.5 hurt()

```
void Gardien::hurt ( )
```

fonction indiquant au gardien qu'il a été touché par une boule de feu provenant du joueur. Baissera sa précision et son nombre de points de vie

#### 4.6.3.6 is\_legit\_move()

```
bool Gardien::is_legit_move (
    double dx,
    double dy ) [private]
```

teste si le mouvement de coordonnées (dx, dy) est acceptable, id est que ça implique pas de marcher au travers d'un mur

#### 4.6.3.7 move()

```
bool Gardien::move (
    double dx,
    double dy ) [virtual]
```

Tente d'accomplir le mouvement en paramètre. Meme principe que try\_move, mais va faire en sorte de glisser sur les murs.

##### Paramètres

<i>dx</i>	mouvement en x tenté par le gardien
<i>dy</i>	mouvement en y tenté par le gardien

##### Renvoie

true si le mouvement a été accompli, false sinon

Implémente [Mover](#).

#### 4.6.3.8 move\_to\_treasure()

```
bool Gardien::move_to_treasure ( ) [private]
```

fait en sorte que le gardien aille vers le trésor

#### 4.6.3.9 process\_fireball()

```
bool Gardien::process_fireball (
    float dx,
    float dy ) [virtual]
```

Gère le déplacement de la boule de feu venant du garde.

##### Paramètres

<i>dx</i>	le déplacement sur la coordonnée x
<i>dy</i>	le déplacement sur la coordonnée y

Implémente [Mover](#).

#### 4.6.3.10 try\_move()

```
bool Gardien::try_move (
    double dx,
    double dy ) [private]
```

tente d'accomplir le mouvement en paramètre

#### 4.6.3.11 update()

```
void Gardien::update (
    void ) [virtual]
```

Implémente [Mover](#).

### 4.6.4 Documentation des données membres

#### 4.6.4.1 \_pv

```
int Gardien::_pv = BASE_PV [private]
```

Nombre de PV restants au gardien.

#### 4.6.4.2 BASE\_PV

```
const int Gardien::BASE_PV = 10 [static], [private]
```

Nombre de PV de base du gardien.

#### 4.6.4.3 dead

```
bool Gardien::dead = false [private]
```

indique si le gardien est mort

#### 4.6.4.4 fired

```
bool Gardien::fired = false [private]
```

Indique si une boule de feu a déjà été tirée. Si c'est le cas, il est impossible pour le gardien d'en tirer une autre.

#### 4.6.4.5 l

```
Labyrinthe* Gardien::l [private]
```

pointeur vers le labyrinthe

#### 4.6.4.6 perte\_precision

```
int Gardien::perte_precision = 5 [private]
```

indique la perte de précision, en degrés, du gardien

#### 4.6.4.7 reload

```
int Gardien::reload = 0 [private]
```

indique le statut du rechargement

#### 4.6.4.8 RELOAD\_TIME

```
const int Gardien::RELOAD_TIME = 100 [static], [private]
```

Temps de rechargement pour les gardiens (en nombre d'appels de [update\(\)](#))

#### 4.6.4.9 reloading

```
bool Gardien::reloading = false [private]
```

indique si le gardien est en train de recharger (si c'est le cas, il ne peut pas tirer)

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

- [Gardien.h](#)
- [Gardien.cc](#)
- [old\\_gardien.cc](#)

## 4.7 Référence de la classe Labyrinthe

```
#include <Labyrinthe.h>
```

Graphe d'héritage de Labyrinthe :

Graphe de collaboration de Labyrinthe :

### Fonctions membres publiques

- [Labyrinthe](#) ()
- [Labyrinthe](#) (char \*)
- int [width](#) ()  
*retourne la largeur du labyrinthe.*
- int [height](#) ()  
*retourne la longueur du labyrinthe.*
- char [data](#) (int i, int j)  
*Informe sur l'état d'occupation de la case (i, j).*
- void [iamdying](#) ()  
*indique au labyrinthe qu'un des gardes est mort*
- int [nb\\_alive](#) ()  
*Indique le nombre de gardiens encore en vie dans le labyrinthe.*
- void [hurt\\_gardien\\_at](#) (int x, int y)  
*indique que une instance de Fireball du joueur a explosé à l'endroit où était placé un (ou plusieurs) des gardiens. Va donc "Blessier" le gardien via la méthode [Gardien::hurt\(\)](#)*
- void [hurt\\_joueur](#) ()  
*indique au labyrinthe que le joueur a été touché par une boule de feu provenant d'un des gardiens*
- void [set\\_data](#) (int x, int y, char value)  
*change la valeur d'une case de la matrice \_data, par exemple lors du déplacement d'un gardien*
- int [dist\\_of\\_treasure](#) (int x, int y)  
*rend la distance de la case courante*

## Fonctions membres privées

- void `init_data` ()  
*initialise `_data[][]`, en fonction de `lab_with` et `lab_height`*
- void `fill_dist` (int x, int y)  
*remplit la matrice de distance au trésor, récursivement, en partant de la case de coordonnées (x, y) qu'on sait déjà remplir*
- void `build_guards` (list< `coord` > guards, const `coord` &player\_pos)  
*Initialise `_guards` et `_nguard` dans la classe.*
- void `build_walls` (list< pair< `coord`, `coord` >> wall\_list)  
*Initialise les murs (`_nwall` et `_walls[]`)*
- void `build_boxes` (list< `coord` > box\_list)  
*initialise les boites*
- void `build_treasure` (`coord` c)  
*initialisation du trésor*
- void `build_text` (map< char, string > text\_map, list< tuple< `coord`, char, bool >> text\_list)  
*initialisation des textures*
- void `init_vector_dist` ()  
*initialise le vecteur des distances. Doit être appelé à la fin du constructeur*

## Attributs privés

- char \*\* `_data`  
*indique si la case est libre ou occupée.*
- int `lab_width`  
*Dimension de l'axe 'x' du labyrinthe.*
- int `lab_height`  
*Dimension de l'axe 'y' du labyrinthe.*
- int `_nb_alive`  
*Indique le nombre.*
- vector< vector< int > > `_dist_vect`  
*Vecteur contenant les distances entre chaque case et le trésor.*

## Membres hérités additionnels

## 4.7.1 Documentation des constructeurs et destructeur

## 4.7.1.1 Labyrinthe() [1/2]

```
Labyrinthe::Labyrinthe ( )
```

## 4.7.1.2 Labyrinthe() [2/2]

```
Labyrinthe::Labyrinthe (
    char * path )
```

## 4.7.2 Documentation des fonctions membres

## 4.7.2.1 build\_boxes()

```
void Labyrinthe::build_boxes (
    list< coord > box_list ) [private]
```

initialise les boites

## Paramètres

<i>box_list</i>	(in) : liste des coordonnées des boîtes
-----------------	---

## 4.7.2.2 build\_guards()

```
void Labyrinthe::build_guards (
    list< coord > guards,
    const coord & player_pos ) [private]
```

Initialise `_guards` et `_nguard` dans la classe.

## Paramètres

<i>guards</i>	(in) : une liste des coordonnées des gardes
<i>player_pos</i>	(in) : La coordonnée du joueur TODO : génération de gardes différents

## 4.7.2.3 build\_text()

```
void Labyrinthe::build_text (
    map< char, string > text_map,
    list< tuple< coord, char, bool >> text_list ) [private]
```

initialisation des textures

## Paramètres

<i>text_map</i>	(in) : la structure associant un caractère représentant une texture dans le .txt du labyrinthe au nom de la texture
<i>text_list</i>	(in) : Une liste des textures, où chaque élément contient la coordonnée de la texture, le caractère représentant la texture, ainsi que l'orientation de la texture (égale à HORIZONTAL ou VERTICAL)

## 4.7.2.4 build\_treasure()

```
void Labyrinthe::build_treasure (
    coord c ) [private]
```

initialisation du trésor

## Paramètres

<i>c</i>	la coordonnée du trésor
----------	-------------------------

## 4.7.2.5 build\_walls()

```
void Labyrinthe::build_walls (
    list< pair< coord, coord >> wall_list ) [private]
```

Initialise les murs (\_nwall et \_walls[])

## Paramètres

<i>wall_list</i>	(in) : la liste des murs, un mur étant représenté par une paire de coordonnées. Pour chaque paire de coordonnées (x1 y1) (x2 y2) de la liste, on fait l'hypothèse que La première coordonnées est "plus petite" que la seconde, id est (x1 < x2    y1 < y2).
------------------	--

## 4.7.2.6 data()

```
char Labyrinthe::data (
    int i,
    int j ) [inline], [virtual]
```

Informe sur l'état d'occupation de la case (i, j).

## Renvoie

WALL si la case est un mur, BOX si la case est occupée par une boîte, GARDE si la case est occupée par un garde, JOUEUR si la case est occupée par le joueur, TREASURE si la case est occupée par le trésor

Implémente [Environnement](#).

## 4.7.2.7 dist\_of\_treasure()

```
int Labyrinthe::dist_of_treasure (
    int x,
    int y ) [inline]
```

rend la distance de la case courante

## Paramètres

<i>x</i>	la coordonnée dans l'axe des abscisses
<i>y</i>	la coordonnée dans l'axe des ordonnées

**Renvoie**

-1 si la case est un mur, 1 000 000 s'il n'existe pas de chemin entre le point de coordonnées (x,y), La distance en nombre de cases entre le point (x,y) et le trésor dans les autres cas

**4.7.2.8 fill\_dist()**

```
void Labyrinthe::fill_dist (
    int x,
    int y ) [private]
```

remplit la matrice de distance au trésor, récursivement, en partant de la case de coordonnées (x, y) qu'on sait déjà remplie

**4.7.2.9 height()**

```
int Labyrinthe::height ( ) [inline], [virtual]
```

retourne la longueur du labyrinthe.

Implémente [Environnement](#).

**4.7.2.10 hurt\_gardien\_at()**

```
void Labyrinthe::hurt_gardien_at (
    int x,
    int y )
```

indique que une instance de Fireball du joueur a explosé à l'endroit où était placé un (ou plusieurs) des gardiens. Va donc "Blesser" le gardien via la méthode [Gardien :hurt\(\)](#)

.

**Paramètres**

x	la composante dans l'axe des abscisses de la case dans laquelle la boule de feu explose y la composante dans l'axe des ordonnées de la case dans laquelle la boule de feu explose
---	---

**4.7.2.11 hurt\_joueur()**

```
void Labyrinthe::hurt_joueur ( )
```



indique au labyrinthe que le joueur a été touché par une boule de feu provenant d'un des gardiens

Va donc appeler la méthode `Chasseur : :hurt()`

#### 4.7.2.12 `iamdying()`

```
void Labyrinthe::iamdying ( ) [inline]
```

indique au labyrinthe qu'un des gardes est mort

#### 4.7.2.13 `init_data()`

```
void Labyrinthe::init_data ( ) [private]
```

initialise `_data[][]`, en fonction de `lab_with` et `lab_height`

#### 4.7.2.14 `init_vector_dist()`

```
void Labyrinthe::init_vector_dist ( ) [private]
```

initialise le vecteur des distances. Doit être appelé à la fin du constructeur

#### 4.7.2.15 `nb_alive()`

```
int Labyrinthe::nb_alive ( ) [inline]
```

Indique le nombre de gardiens encore en vie dans le labyrinthe.

**Renvoie**

le nombre de gardes encore en vie.

#### 4.7.2.16 `set_data()`

```
void Labyrinthe::set_data (
    int x,
    int y,
    char value )
```

change la valeur d'une case de la matrice `_data`, par exemple lors du déplacement d'un gardien

**Paramètres**

<i>x</i>	
<i>y</i>	
<i>value</i>	La nouvelle valeur de la case <code>_data[x][y]</code> . En théorie, compris entre EMPTY et TREASURE, mais dans les faits ce sera EMPTY, JOUEUR, ou GARDE, étant donné que seules ces entités se déplacent)

**4.7.2.17 width()**

```
int Labyrinthe::width ( ) [inline], [virtual]
```

retourne la largeur du labyrinthe.

Implémente [Environnement](#).

**4.7.3 Documentation des données membres****4.7.3.1 \_data**

```
char** Labyrinthe::_data [private]
```

indique si la case est libre ou occupée.

**4.7.3.2 \_dist\_vect**

```
vector<vector<int> > Labyrinthe::_dist_vect [private]
```

Vecteur contenant les distances entre chaque case et le trésor.

**4.7.3.3 \_nb\_alive**

```
int Labyrinthe::_nb_alive [private]
```

Indique le nombre.

## 4.7.3.4 lab\_height

```
int Labyrinthe::lab_height [private]
```

Dimension de l'axe 'y' du labyrinthe.

## 4.7.3.5 lab\_width

```
int Labyrinthe::lab_width [private]
```

Dimension de l'axe 'x' du labyrinthe.

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

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

## 4.8 Référence de la classe Mover

```
#include <Mover.h>
```

Graphe d'héritage de Mover :

Graphe de collaboration de Mover :

## Fonctions membres publiques

- [Mover](#) (int x, int y, [Labyrinthe](#) \*l, const char \*modele)
- virtual [~Mover](#) ()
- void [tomber](#) ()
- void [rester\\_au\\_sol](#) ()
- virtual void [update](#) (void)=0
- virtual bool [process\\_fireball](#) (float dx, float dy)=0
- virtual bool [move](#) (double dx, double dy)=0
- virtual void [fire](#) (int angle\_vertical)=0
- virtual void [right\\_click](#) (bool shift, bool control)

## Attributs publics

- [Environnement](#) \* \_l
- [FireBall](#) \* \_fb
- float \_x
- float \_y
- int \_angle
- void \* \_model

## Fonctions membres privées statiques

- static void \* [init](#) (const char \*)

## 4.8.1 Documentation des constructeurs et destructeur

### 4.8.1.1 Mover()

```
Mover::Mover (
    int x,
    int y,
    Labyrinthe * l,
    const char * modele ) [inline]
```

### 4.8.1.2 ~Mover()

```
virtual Mover::~~Mover ( ) [inline], [virtual]
```

## 4.8.2 Documentation des fonctions membres

### 4.8.2.1 fire()

```
virtual void Mover::fire (
    int angle_vertical ) [pure virtual]
```

Implémenté dans [Gardien](#), et [Chasseur](#).

### 4.8.2.2 init()

```
static void* Mover::init (
    const char * ) [static], [private]
```

### 4.8.2.3 move()

```
virtual bool Mover::move (
    double dx,
    double dy ) [pure virtual]
```

Implémenté dans [Gardien](#), et [Chasseur](#).

#### 4.8.2.4 process\_fireball()

```
virtual bool Mover::process_fireball (
    float dx,
    float dy ) [pure virtual]
```

Implémenté dans [Gardien](#), et [Chasseur](#).

#### 4.8.2.5 rester\_au\_sol()

```
void Mover::rester_au_sol ( )
```

#### 4.8.2.6 right\_click()

```
virtual void Mover::right_click (
    bool shift,
    bool control ) [inline], [virtual]
```

Réimplémentée dans [Chasseur](#).

#### 4.8.2.7 tomber()

```
void Mover::tomber ( )
```

#### 4.8.2.8 update()

```
virtual void Mover::update (
    void ) [pure virtual]
```

Implémenté dans [Gardien](#), et [Chasseur](#).

### 4.8.3 Documentation des données membres

#### 4.8.3.1 \_angle

```
int Mover::_angle
```

#### 4.8.3.2 `_fb`

```
FireBall* Mover::_fb
```

#### 4.8.3.3 `_l`

```
Environnement* Mover::_l
```

#### 4.8.3.4 `_model`

```
void* Mover::_model
```

#### 4.8.3.5 `_x`

```
float Mover::_x
```

#### 4.8.3.6 `_y`

```
float Mover::_y
```

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

— [Mover.h](#)

## 4.9 Référence de la structure node

```
#include <Gardien.h>
```

### Attributs publics

— `int` [x](#)  
— `int` [y](#)

#### 4.9.1 Documentation des données membres

#### 4.9.1.1 `x`

```
int node::x
```

#### 4.9.1.2 `y`

```
int node::y
```

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

— [Gardien.h](#)

## 4.10 Référence de la structure `node_comparator`

### Fonctions membres publiques

— `bool operator()` (`const node &a`, `const node &b`) `const`

#### 4.10.1 Documentation des fonctions membres

##### 4.10.1.1 `operator()`

```
bool node_comparator::operator() (
    const node & a,
    const node & b ) const [inline]
```

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

— [old\\_gardien.cc](#)

## 4.11 Référence de la classe `Sound`

```
#include <Sound.h>
```

### Fonctions membres publiques

— `Sound` (`const char *`)  
— `~Sound` ()  
— `void play` (`float volume=1.`, `float pan=0.`)

### Fonctions membres privées

— `void init` (`void`)

## Attributs privés

— FMOD : :Sound \* [\\_sound](#)

## Attributs privés statiques

— static FMOD : :System \* [\\_system](#)  
— static FMOD : :Channel \* [\\_channel](#)  
— static int [\\_nsounds](#)

## 4.11.1 Documentation des constructeurs et destructeur

### 4.11.1.1 Sound()

```
Sound::Sound (
    const char * )
```

### 4.11.1.2 ~Sound()

```
Sound::~~Sound ( )
```

## 4.11.2 Documentation des fonctions membres

### 4.11.2.1 init()

```
void Sound::init (
    void ) [private]
```

### 4.11.2.2 play()

```
void Sound::play (
    float volume = 1.,
    float pan = 0. )
```

## 4.11.3 Documentation des données membres



#### 4.11.3.1 `_channel`

```
FMOD::Channel* Sound::_channel [static], [private]
```

#### 4.11.3.2 `_nsounds`

```
int Sound::_nsounds [static], [private]
```

#### 4.11.3.3 `_sound`

```
FMOD::Sound* Sound::_sound [private]
```

#### 4.11.3.4 `_system`

```
FMOD::System* Sound::_system [static], [private]
```

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

— [Sound.h](#)

## 4.12 Référence de la structure Wall

```
#include <Environnement.h>
```

### Attributs publics

- [int \\_x1](#)
- [int \\_y1](#)
- [int \\_x2](#)
- [int \\_y2](#)
- [int \\_ntex](#)

### 4.12.1 Documentation des données membres

#### 4.12.1.1 `_ntex`

```
int Wall::_ntex
```

#### 4.12.1.2 \_x1

```
int Wall::_x1
```

#### 4.12.1.3 \_x2

```
int Wall::_x2
```

#### 4.12.1.4 \_y1

```
int Wall::_y1
```

#### 4.12.1.5 \_y2

```
int Wall::_y2
```

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

— [Environnement.h](#)

## Chapitre 5

# Documentation des fichiers

### 5.1 Référence du fichier Chasseur.cc

```
#include "Chasseur.h"
```

Graphe des dépendances par inclusion de Chasseur.cc :

### 5.2 Référence du fichier Chasseur.h

```
#include <stdio.h>
#include "Mover.h"
#include "Sound.h"
#include "Labyrinthe.h"
```

Graphe des dépendances par inclusion de Chasseur.h : Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

#### Classes

— class [Chasseur](#)

### 5.3 Référence du fichier Environnement.h

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

#### Classes

— struct [Wall](#)  
— struct [Box](#)  
— class [Environnement](#)

#### Macros

— #define [EMPTY](#) 0

## Fonctions

- void [partie\\_terminee](#) (bool win)
- void [message](#) (const char \*format,...)

### 5.3.1 Documentation des macros

#### 5.3.1.1 EMPTY

```
#define EMPTY 0
```

### 5.3.2 Documentation des fonctions

#### 5.3.2.1 message()

```
void message (  
    const char * format,  
    ... )
```

#### 5.3.2.2 partie\_terminee()

```
void partie_terminee (  
    bool win )
```

## 5.4 Référence du fichier FireBall.h

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

## Classes

- class [FireBall](#)

## Macros

- #define [M\\_PI](#) 3.1415926

## Énumérations

— enum `FBstat` { `FB_NONE`, `FB_MOVE`, `FB_EXPLODE` }

### 5.4.1 Documentation des macros

#### 5.4.1.1 M\_PI

```
#define M_PI 3.1415926
```

### 5.4.2 Documentation du type de l'énumération

#### 5.4.2.1 FBstat

```
enum FBstat
```

Valeurs énumérées

<code>FB_NONE</code>	
<code>FB_MOVE</code>	
<code>FB_EXPLODE</code>	

## 5.5 Référence du fichier Gardien.cc

```
#include "Gardien.h"
```

Graphe des dépendances par inclusion de Gardien.cc :

## 5.6 Référence du fichier Gardien.h

```
#include "Mover.h"
#include <stdlib.h>
#include <iostream>
#include <cstdlib>
#include <cmath>
#include <map>
#include <list>
#include <limits.h>
#include <set>
#include <iterator>
```

```
#include <stdexcept>
#include "Labyrinthe.h"
```

Graphe des dépendances par inclusion de Gardien.h : Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

## Classes

- struct [node](#)
- class [Gardien](#)  
*Classe implémentant les gardes.*

## 5.7 Référence du fichier laby2.c

## 5.8 Référence du fichier Labyrinthe.cc

```
#include "Labyrinthe.h"
#include "Chasseur.h"
#include "Gardien.h"
#include <sstream>
```

Graphe des dépendances par inclusion de Labyrinthe.cc :

## Fonctions

- bool [is\\_blank](#) (const string &str)  
*Décide si une ligne est considérée comme vide (commentaires inclus), id est ne contenant que des caractères espace (ou un commentaire)*
- char [first\\_char](#) (const string &str)  
*rend le premier caractère non-nul de la chaîne, ou 0 si la chaîne est vide*
- void [build\\_map](#) (char \*path, vector< string > &terrain, map< char, string > &tex\_list)
- void [parse\\_wall\\_extr](#) (const vector< string > &terrain, const [coord](#) &base\_coord, list< [coord](#) > &extr\_list, list< tuple< [coord](#), char, bool >> &text\_list)  
*fait la liste des extrémités des murs partant de la coordonnée base\_coord sur le terrain.*
- void [add\\_to\\_wall\\_list](#) (const [coord](#) &base\_coord, list< [coord](#) > other\_extr, list< pair< [coord](#), [coord](#) >> &wall\_list)  
*fusionne les extrémités et la coordonnée de base, pour créer un ou deux murs*
- void [parse\\_map](#) (const vector< string > &terrain, list< [coord](#) > &guard\_list, list< pair< [coord](#), [coord](#) >> &wall\_list, list< tuple< [coord](#), char, bool >> &text\_list, list< [coord](#) > &box\_list, [coord](#) &player\_pos, [coord](#) &treasure\_pos, int &height, int &width)  
*construit les listes*

### 5.8.1 Documentation des fonctions

#### 5.8.1.1 add\_to\_wall\_list()

```
void add_to_wall_list (
    const coord & base_coord,
    list< coord > other_extr,
    list< pair< coord, coord >> & wall_list )
```

fusionne les extrémités et la coordonnée de base, pour créer un ou deux murs

## Paramètres

<i>other_extr</i>	(in) : La liste des extrémités
<i>wall_lst</i>	(inout) : La liste des murs

## 5.8.1.2 build\_map()

```
void build_map (
    char * path,
    vector< string > & terrain,
    map< char, string > & tex_list )
```

## 5.8.1.3 first\_char()

```
char first_char (
    const string & str )
```

rend le premier caractère non-nul de la chaîne, ou 0 si la chaîne est vide

## Paramètres

<i>la</i>	str à tester
-----------	--------------

## 5.8.1.4 is\_blank()

```
bool is_blank (
    const string & str )
```

Décide si une ligne est considérée comme vide (commentaires inclus), id est ne contenant que des caractères espace (ou un commentaire)

## Paramètres

<i>str</i>	la chaîne de caractère testée
------------	-------------------------------

## Renvoie

true si la chaîne est vide (uniquement des espaces ou des commentaires), false sinon

### 5.8.1.5 parse\_map()

```
void parse_map (
    const vector< string > & terrain,
    list< coord > & guard_list,
    list< pair< coord, coord >> & wall_list,
    list< tuple< coord, char, bool >> & text_list,
    list< coord > & box_list,
    coord & player_pos,
    coord & treasure_pos,
    int & height,
    int & width )
```

construit les listes

#### Paramètres

<i>terrain</i>	(in) : Le terrain/labyrinthe
<i>guard_list</i>	(out) : La liste des coordonnées des gardes
<i>wall_list</i>	(out) : La liste des coordonnées des murs
<i>text_list</i>	(out) : La liste des textures (une texture = coordonnée, char-id de texture et un indicateur sur le sens de la texture)
<i>box_list</i>	(out) : La liste des boîtes
<i>player_pos</i>	(out) : La position du joueur
<i>treasure_pos(out)</i>	La position du trésor

### 5.8.1.6 parse\_wall\_extr()

```
void parse_wall_extr (
    const vector< string > & terrain,
    const coord & base_coord,
    list< coord > & extr_list,
    list< tuple< coord, char, bool >> & text_list )
```

fait la liste des extrémités des murs partant de la coordonnée base\_coord sur le terrain.

#### Paramètres

<i>terrain</i>	(in) : Le labyrinthe
<i>base_coord</i>	(in) : La coordonnée de départ du mur
<i>extr_list</i>	(out) : La liste des extrémités des murs
<i>text_list</i>	(inout) : La liste des coordonnées+caractères+"verticalité" des textures

## 5.9 Référence du fichier Labyrinthe.h

```
#include <stdio.h>
#include <stdlib.h>
```



```
#include <utility>
#include <list>
#include <iostream>
#include <string>
#include <vector>
#include <ios>
#include <map>
#include <fstream>
#include <unistd.h>
#include "Environnement.h"
```

Graphe des dépendances par inclusion de Labyrinthe.h : Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

## Classes

- struct [coord](#)  
*Structure représentant une coordonnée du labyrinthe.*
- class [Labyrinthe](#)

## Fonctions

- void [build\\_map](#) (const char \*path, vector< string > &terrain, map< char, string > &tex\_list)  
*fonction construisant à la fois le terrain et la liste des textures*
- void [parse\\_map](#) (const vector< string > &terrain, list< [coord](#) > &guard\_list, list< pair< [coord](#), [coord](#) >> &wall\_list, list< tuple< [coord](#), char, bool >> &text\_list, list< [coord](#) > &box\_list, [coord](#) &player\_pos, [coord](#) &treasure\_pos, int &height, int &width)  
*construit les listes*
- void [parse\\_wall\\_extr](#) (const vector< string > &terrain, const [coord](#) &base\_coord, list< [coord](#) > &extr\_list, list< tuple< [coord](#), char, bool >> &text\_list)  
*fait la liste des extrémités des murs partant de la coordonnée base\_coord sur le terrain.*
- void [add\\_to\\_wall\\_list](#) (const [coord](#) &base\_coord, list< [coord](#) > other\_extr, list< pair< [coord](#), [coord](#) >> &wall\_list)  
*fusionne les extrémités et la coordonnée de base, pour créer un ou deux murs*
- bool [is\\_blank](#) (const string &str)  
*Décide si une ligne est considérée comme vide (commentaires inclus), id est ne contenant que des caractères espace (ou un commentaire)*
- char [first\\_char](#) (const string &str)  
*rend le premier caractère non-nul de la chaîne, ou 0 si la chaîne est vide*

## Variables

- const bool [HORIZONTAL](#) = false  
*booléen permettant d'indiquer le sens des affiches*
- const bool [VERTICAL](#) = true  
*booléen permettant d'indiquer le sens des affiches*
- const char [WALL](#) = 1
- const char [BOX](#) = 2
- const char [GARDE](#) = 3
- const char [JOUEUR](#) = 4
- const char [TREASURE](#) = 5

### 5.9.1 Documentation des fonctions

### 5.9.1.1 add\_to\_wall\_list()

```
void add_to_wall_list (
    const coord & base_coord,
    list< coord > other_extr,
    list< pair< coord, coord >> & wall_lst )
```

fusionne les extrémités et la coordonnée de base, pour créer un ou deux murs

#### Paramètres

<i>other_extr</i>	(in) : La liste des extrémités
<i>wall_lst</i>	(inout) : La liste des murs

### 5.9.1.2 build\_map()

```
void build_map (
    const char * path,
    vector< string > & terrain,
    map< char, string > & tex_list )
```

fonction construisant à la fois le terrain et la liste des textures

#### Paramètres

<i>path</i>	(in) : Le chemin d'accès au fichier du labyrinthe
<i>terrain</i>	(out) : Le tableau de string représentant le labyrinthe
<i>tex_list</i>	(out) : La hashmap associant un chemin d'accès à un caractère

### 5.9.1.3 first\_char()

```
char first_char (
    const string & str )
```

rend le premier caractère non-nul de la chaîne, ou 0 si la chaîne est vide

#### Paramètres

<i>la</i>	str à tester
-----------	--------------

### 5.9.1.4 is\_blank()

```
bool is_blank (
```

```
const string & str )
```

Décide si une ligne est considérée comme vide (commentaires inclus), id est ne contenant que des caractères espace (ou un commentaire)

#### Paramètres

<i>str</i>	la chaîne de caractère testée
------------	-------------------------------

#### Renvoie

true si la chaîne est vide (uniquement des espaces ou des commentaires), false sinon

#### 5.9.1.5 parse\_map()

```
void parse_map (
    const vector< string > & terrain,
    list< coord > & guard_list,
    list< pair< coord, coord >> & wall_list,
    list< tuple< coord, char, bool >> & text_list,
    list< coord > & box_list,
    coord & player_pos,
    coord & treasure_pos,
    int & height,
    int & width )
```

construit les listes

#### Paramètres

<i>terrain</i>	(in) : Le terrain/labyrinthe
<i>guard_list</i>	(out) : La liste des coordonnées des gardes
<i>wall_list</i>	(out) : La liste des coordonnées des murs
<i>text_list</i>	(out) : La liste des textures (une texture = coordonnée, char-id de texture et un indicateur sur le sens de la texture)
<i>box_list</i>	(out) : La liste des boites
<i>player_pos</i>	(out) : La position du joueur
<i>treasure_pos(out)</i>	La position du trésor

#### 5.9.1.6 parse\_wall\_extr()

```
void parse_wall_extr (
    const vector< string > & terrain,
    const coord & base_coord,
    list< coord > & extr_list,
    list< tuple< coord, char, bool >> & text_list )
```

fait la liste des extrémités des murs partant de la coordonnée base\_coord sur le terrain.

**Paramètres**

<i>terrain</i>	(in) : Le labyrinthe
<i>base_coord</i>	(in) : La coordonnée de départ du mur
<i>extr_list</i>	(out) : La liste des extrémités des murs
<i>text_list</i>	(inout) : La liste des coordonnées+caractères+"verticalité" des textures

## 5.9.2 Documentation des variables

### 5.9.2.1 BOX

```
const char BOX = 2
```

### 5.9.2.2 GARDE

```
const char GARDE = 3
```

### 5.9.2.3 HORIZONTAL

```
const bool HORIZONTAL = false
```

booléen permettant d'indiquer le sens des affiches

### 5.9.2.4 JOUEUR

```
const char JOUEUR = 4
```

### 5.9.2.5 TREASURE

```
const char TREASURE = 5
```

### 5.9.2.6 VERTICAL

```
const bool VERTICAL = true
```

booléen permettant d'indiquer le sens des affiches

### 5.9.2.7 WALL

```
const char WALL = 1
```

## 5.10 Référence du fichier Mover.h

```
#include "FireBall.h"  
#include "Environnement.h"
```

Graphe des dépendances par inclusion de Mover.h : Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

### Classes

— class [Mover](#)

## 5.11 Référence du fichier old\_gardien.cc

```
#include "Gardien.h"
```

Graphe des dépendances par inclusion de old\_gardien.cc :

### Classes

— struct [node\\_comparator](#)

### Fonctions

- bool [operator==](#) (const [node](#) &a, const [node](#) &b)
- int [get](#) (map< [node](#), int, [node\\_comparator](#) > &m, [node](#) &key, int def)
- list< [node](#) > [build\\_solution](#) (const [node](#) &goal, map< [node](#), [node](#), [node\\_comparator](#) > &predecesseurs)
- int [estimation\\_dist](#) (const [node](#) &a, const [node](#) &b)
- list< [node](#) > [get\\_shortest\\_path](#) ([node](#) start, [node](#) goal, [Environnement](#) \*evt)

### Variables

- int [MIN](#) = -1000000
- int [MAX](#) = 1000000

## 5.11.1 Documentation des fonctions

### 5.11.1.1 build\_solution()

```
list<node> build_solution (
    const node & goal,
    map< node, node, node_comparator > & predecesseurs )
```

### 5.11.1.2 estimation\_dist()

```
int estimation_dist (
    const node & a,
    const node & b )
```

### 5.11.1.3 get()

```
int get (
    map< node, int, node_comparator > & m,
    node & key,
    int def )
```

### 5.11.1.4 get\_shortest\_path()

```
list<node> get_shortest_path (
    node start,
    node goal,
    Environnement * evt )
```

### 5.11.1.5 operator==( )

```
bool operator== (
    const node & a,
    const node & b )
```

## 5.11.2 Documentation des variables

### 5.11.2.1 MAX

```
int MAX = 1000000
```

### 5.11.2.2 MIN

```
int MIN = -1000000
```

## 5.12 Référence du fichier Sound.h

```
#include "fmod.hpp"  
#include "fmod_errors.h"
```

Graphe des dépendances par inclusion de Sound.h : Ce graphe montre quels fichiers incluent directement ou indirectement ce fichier :

### Classes

— class [Sound](#)





# Index

- FireBall, [18](#)
  - Mover, [35](#)
- FireBall, [19](#)
- FireBall, [19](#)
- Environnement, [15](#)
- Sound, [38](#)
- Labyrinthe, [32](#)
- Labyrinthe, [32](#)
- Mover, [35](#)
- FireBall, [19](#)
- FireBall, [19](#)
- Environnement, [15](#)
- Chasseur, [11](#)
- Chasseur, [11](#)
- Mover, [36](#)
- Mover, [36](#)
- FireBall, [19](#)
- Labyrinthe, [32](#)
- Environnement, [15](#)
- Environnement, [15](#)
- Sound, [39](#)
- Box, [7](#)
  - Wall, [39](#)
- Environnement, [15](#)
- FireBall, [19](#)
- FireBall, [19](#)
- Environnement, [15](#)
- Chasseur, [11](#)
- Environnement, [16](#)
- Box, [7](#)
  - FireBall, [20](#)
  - Mover, [36](#)
- Wall, [39](#)
- Wall, [40](#)
- Box, [7](#)
  - FireBall, [20](#)
  - Mover, [36](#)
- Wall, [40](#)
- Wall, [40](#)
- FireBall, [20](#)
- Environnement, [13](#)
- Gardien, [22](#)
- Mover, [34](#)
- Sound, [38](#)
- add\_to\_wall\_list

- Labyrinthe.cc, [44](#)
  - Labyrinthe.h, [47](#)
- avancer
  - Gardien, [22](#)
- BASE\_PV
  - Gardien, [25](#)
- BOX
  - Labyrinthe.h, [50](#)
- Box, [7](#)
  - \_ntex, [7](#)
  - \_x, [7](#)
  - \_y, [7](#)
- build\_boxes
  - Labyrinthe, [27](#)
- build\_guards
  - Labyrinthe, [28](#)
- build\_map
  - Labyrinthe.cc, [45](#)
  - Labyrinthe.h, [48](#)
- build\_solution
  - old\_gardien.cc, [52](#)
- build\_text
  - Labyrinthe, [28](#)
- build\_treasure
  - Labyrinthe, [28](#)
- build\_walls
  - Labyrinthe, [29](#)
- can\_see\_player
  - Gardien, [22](#)
- Chasseur, [8](#)
  - \_hunter\_fire, [11](#)
  - \_hunter\_hit, [11](#)
  - \_pv, [11](#)
  - \_wall\_hit, [11](#)
  - Chasseur, [8](#)
  - die, [9](#)
  - fire, [9](#)
  - hurt, [9](#)
  - l, [11](#)
  - move, [9](#)
  - move\_aux, [9](#)
  - perte\_precision, [11](#)
  - process\_fireball, [10](#)
  - right\_click, [10](#)
  - update, [10](#)
- Chasseur.cc, [41](#)
- Chasseur.h, [41](#)
- coord, [12](#)
  - x, [12](#)
  - y, [12](#)
- data
  - Environnement, [13](#)
  - Labyrinthe, [29](#)
- dead
  - Gardien, [25](#)
- die
  - Chasseur, [9](#)
  - Gardien, [22](#)
- display
  - FireBall, [17](#)
- dist\_of\_treasure
  - Labyrinthe, [29](#)
- EMPTY
  - Environnement.h, [42](#)
- Environnement, [13](#)
  - \_boxes, [15](#)
  - \_guards, [15](#)
  - \_nboxes, [15](#)
  - \_nguards, [15](#)
  - \_npicts, [15](#)
  - \_nwall, [15](#)
  - \_picts, [15](#)
  - \_treasure, [16](#)
  - \_walls, [16](#)
  - ~Environnement, [13](#)
  - data, [13](#)
  - height, [14](#)
  - init, [14](#)
  - make\_fireballs, [14](#)
  - modele\_dir, [16](#)
  - reconfigure, [14](#)
  - scale, [16](#)
  - texture\_dir, [16](#)
  - wall\_texture, [14](#)
  - width, [14](#)
- Environnement.h, [41](#)
  - EMPTY, [42](#)
  - message, [42](#)
  - partie\_terminee, [42](#)
- estimation\_dist
  - old\_gardien.cc, [52](#)
- explode
  - FireBall, [17](#)
- FBstat
  - FireBall.h, [43](#)
- fill\_dist
  - Labyrinthe, [30](#)
- fire
  - Chasseur, [9](#)
  - Gardien, [22](#)
  - Mover, [34](#)
- FireBall, [16](#)
  - \_angle, [18](#)
  - \_angle2, [19](#)
  - \_azimuth, [19](#)
  - \_fire\_list, [19](#)
  - \_fire\_texture, [19](#)
  - \_move\_angle, [19](#)
  - \_orig\_size, [19](#)
  - \_owner, [19](#)
  - \_size, [19](#)
  - \_state, [20](#)
  - \_x, [20](#)

- [\\_y](#), [20](#)
  - [\\_z](#), [20](#)
  - [display](#), [17](#)
  - [explode](#), [17](#)
  - [FireBall](#), [17](#)
  - [get\\_x](#), [18](#)
  - [get\\_y](#), [18](#)
  - [init](#), [18](#)
  - [move\\_step](#), [18](#)
  - [set\\_orig\\_size](#), [18](#)
- [FireBall.h](#), [42](#)
- [FBstat](#), [43](#)
- [M\\_PI](#), [43](#)
- [fired](#)
  - [Gardien](#), [25](#)
- [first\\_char](#)
  - [Labyrinthe.cc](#), [45](#)
  - [Labyrinthe.h](#), [48](#)
- [GARDE](#)
  - [Labyrinthe.h](#), [50](#)
- [Gardien](#), [20](#)
  - [\\_pv](#), [24](#)
  - [~Gardien](#), [22](#)
  - [avancer](#), [22](#)
  - [BASE\\_PV](#), [25](#)
  - [can\\_see\\_player](#), [22](#)
  - [dead](#), [25](#)
  - [die](#), [22](#)
  - [fire](#), [22](#)
  - [fired](#), [25](#)
  - [Gardien](#), [22](#)
  - [hurt](#), [23](#)
  - [is\\_legit\\_move](#), [23](#)
  - [I](#), [25](#)
  - [move](#), [23](#)
  - [move\\_to\\_treasure](#), [23](#)
  - [perte\\_precision](#), [25](#)
  - [process\\_fireball](#), [24](#)
  - [RELOAD\\_TIME](#), [26](#)
  - [reload](#), [25](#)
  - [reloading](#), [26](#)
  - [try\\_move](#), [24](#)
  - [update](#), [24](#)
- [Gardien.cc](#), [43](#)
- [Gardien.h](#), [43](#)
- [get](#)
  - [old\\_gardien.cc](#), [52](#)
- [get\\_shortest\\_path](#)
  - [old\\_gardien.cc](#), [52](#)
- [get\\_x](#)
  - [FireBall](#), [18](#)
- [get\\_y](#)
  - [FireBall](#), [18](#)
- [HORIZONTAL](#)
  - [Labyrinthe.h](#), [50](#)
- [height](#)
  - [Environnement](#), [14](#)
  - [Labyrinthe](#), [30](#)
- [hurt](#)
  - [Chasseur](#), [9](#)
  - [Gardien](#), [23](#)
- [hurt\\_gardien\\_at](#)
  - [Labyrinthe](#), [30](#)
- [hurt\\_joueur](#)
  - [Labyrinthe](#), [30](#)
- [iamdying](#)
  - [Labyrinthe](#), [31](#)
- [init](#)
  - [Environnement](#), [14](#)
  - [FireBall](#), [18](#)
  - [Mover](#), [34](#)
  - [Sound](#), [38](#)
- [init\\_data](#)
  - [Labyrinthe](#), [31](#)
- [init\\_vector\\_dist](#)
  - [Labyrinthe](#), [31](#)
- [is\\_blank](#)
  - [Labyrinthe.cc](#), [45](#)
  - [Labyrinthe.h](#), [48](#)
- [is\\_legit\\_move](#)
  - [Gardien](#), [23](#)
- [JOUEUR](#)
  - [Labyrinthe.h](#), [50](#)
- [I](#)
  - [Chasseur](#), [11](#)
  - [Gardien](#), [25](#)
- [lab\\_height](#)
  - [Labyrinthe](#), [32](#)
- [lab\\_width](#)
  - [Labyrinthe](#), [33](#)
- [laby2.c](#), [44](#)
- [Labyrinthe](#), [26](#)
  - [\\_data](#), [32](#)
  - [\\_dist\\_vect](#), [32](#)
  - [\\_nb\\_alive](#), [32](#)
  - [build\\_boxes](#), [27](#)
  - [build\\_guards](#), [28](#)
  - [build\\_text](#), [28](#)
  - [build\\_treasure](#), [28](#)
  - [build\\_walls](#), [29](#)
  - [data](#), [29](#)
  - [dist\\_of\\_treasure](#), [29](#)
  - [fill\\_dist](#), [30](#)
  - [height](#), [30](#)
  - [hurt\\_gardien\\_at](#), [30](#)
  - [hurt\\_joueur](#), [30](#)
  - [iamdying](#), [31](#)
  - [init\\_data](#), [31](#)
  - [init\\_vector\\_dist](#), [31](#)
  - [lab\\_height](#), [32](#)
  - [lab\\_width](#), [33](#)
  - [Labyrinthe](#), [27](#)
  - [nb\\_alive](#), [31](#)

- set\_data, 31
  - width, 32
- Labyrinthe.cc, 44
  - add\_to\_wall\_list, 44
  - build\_map, 45
  - first\_char, 45
  - is\_blank, 45
  - parse\_map, 45
  - parse\_wall\_extr, 46
- Labyrinthe.h, 46
  - add\_to\_wall\_list, 47
  - BOX, 50
  - build\_map, 48
  - first\_char, 48
  - GARDE, 50
  - HORIZONTAL, 50
  - is\_blank, 48
  - JOUEUR, 50
  - parse\_map, 49
  - parse\_wall\_extr, 49
  - TREASURE, 50
  - VERTICAL, 50
  - WALL, 51
- M\_PI
  - FireBall.h, 43
- MAX
  - old\_gardien.cc, 52
- MIN
  - old\_gardien.cc, 53
- make\_fireballs
  - Environnement, 14
- message
  - Environnement.h, 42
- modele\_dir
  - Environnement, 16
- move
  - Chasseur, 9
  - Gardien, 23
  - Mover, 34
- move\_aux
  - Chasseur, 9
- move\_step
  - FireBall, 18
- move\_to\_treasure
  - Gardien, 23
- Mover, 33
  - \_angle, 35
  - \_fb, 35
  - \_l, 36
  - \_model, 36
  - \_x, 36
  - \_y, 36
  - ~Mover, 34
  - fire, 34
  - init, 34
  - move, 34
  - Mover, 34
  - process\_fireball, 34
  - rester\_au\_sol, 35
  - right\_click, 35
  - tomber, 35
  - update, 35
- Mover.h, 51
- nb\_alive
  - Labyrinthe, 31
- node, 36
  - x, 36
  - y, 37
- node\_comparator, 37
  - operator(), 37
- old\_gardien.cc, 51
  - build\_solution, 52
  - estimation\_dist, 52
  - get, 52
  - get\_shortest\_path, 52
  - MAX, 52
  - MIN, 53
  - operator==, 52
- operator()
  - node\_comparator, 37
- operator==
  - old\_gardien.cc, 52
- parse\_map
  - Labyrinthe.cc, 45
  - Labyrinthe.h, 49
- parse\_wall\_extr
  - Labyrinthe.cc, 46
  - Labyrinthe.h, 49
- partie\_terminee
  - Environnement.h, 42
- perte\_precision
  - Chasseur, 11
  - Gardien, 25
- play
  - Sound, 38
- process\_fireball
  - Chasseur, 10
  - Gardien, 24
  - Mover, 34
- RELOAD\_TIME
  - Gardien, 26
- reconfigure
  - Environnement, 14
- reload
  - Gardien, 25
- reloading
  - Gardien, 26
- rester\_au\_sol
  - Mover, 35
- right\_click
  - Chasseur, 10
  - Mover, 35
- scale

- Environnement, [16](#)
- set\_data
  - Labyrinthe, [31](#)
- set\_orig\_size
  - FireBall, [18](#)
- Sound, [37](#)
  - \_channel, [38](#)
  - \_nsounds, [39](#)
  - \_sound, [39](#)
  - \_system, [39](#)
  - ~Sound, [38](#)
  - init, [38](#)
  - play, [38](#)
  - Sound, [38](#)
- Sound.h, [53](#)
- TREASURE
  - Labyrinthe.h, [50](#)
- texture\_dir
  - Environnement, [16](#)
- tomber
  - Mover, [35](#)
- try\_move
  - Gardien, [24](#)
- update
  - Chasseur, [10](#)
  - Gardien, [24](#)
  - Mover, [35](#)
- VERTICAL
  - Labyrinthe.h, [50](#)
- WALL
  - Labyrinthe.h, [51](#)
- Wall, [39](#)
  - \_ntex, [39](#)
  - \_x1, [39](#)
  - \_x2, [40](#)
  - \_y1, [40](#)
  - \_y2, [40](#)
- wall\_texture
  - Environnement, [14](#)
- width
  - Environnement, [14](#)
  - Labyrinthe, [32](#)
- x
  - coord, [12](#)
  - node, [36](#)
- y
  - coord, [12](#)
  - node, [37](#)