

Bombás játék

Generated by Doxygen 1.8.5

Wed Oct 30 2013 01:15:45

Contents

1	Hierarchical Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	Class Documentation	5
3.1	Animation Class Reference	5
3.1.1	Constructor & Destructor Documentation	5
3.1.1.1	Animation	5
3.1.1.2	Animation	5
3.1.2	Member Function Documentation	5
3.1.2.1	draw	5
3.1.2.2	getRect	6
3.2	AnimFile Class Reference	6
3.2.1	Constructor & Destructor Documentation	6
3.2.1.1	AnimFile	6
3.2.1.2	AnimFile	6
3.2.2	Member Function Documentation	6
3.2.2.1	frame	6
3.3	AnimObject Class Reference	6
3.3.1	Constructor & Destructor Documentation	7
3.3.1.1	AnimObject	7
3.3.1.2	AnimObject	7
3.3.2	Member Function Documentation	7
3.3.2.1	draw	7
3.4	Bomb Class Reference	7
3.4.1	Constructor & Destructor Documentation	8
3.4.1.1	Bomb	8
3.4.1.2	Bomb	8
3.4.2	Member Function Documentation	8
3.4.2.1	boom	8

3.4.2.2	boom	8
3.4.2.3	getOwner	8
3.4.2.4	getPoint	9
3.4.2.5	getSize	9
3.4.2.6	getX	9
3.4.2.7	getY	9
3.4.2.8	state	9
3.5	Client Class Reference	9
3.6	Config Struct Reference	9
3.6.1	Member Data Documentation	10
3.6.1.1	CFG_BOMBTIME	10
3.6.1.2	CFG_DEFAULT_SPEED	10
3.6.1.3	CFG_FAST_SPEED	10
3.6.1.4	CFG_FLAMETIME	10
3.6.1.5	CFG_PORT	10
3.6.1.6	CFG_RESPAWN_MONSTER	10
3.6.1.7	CFG_RESPAWN_PLAYER	10
3.6.1.8	CFG_SHOW_SCOREBOARD	10
3.6.1.9	CFG_SLOW_SPEED	10
3.6.1.10	CFG_VIRUSTIME	10
3.7	Control Class Reference	11
3.7.1	Constructor & Destructor Documentation	11
3.7.1.1	Control	11
3.7.2	Member Function Documentation	11
3.7.2.1	getDelay	11
3.7.2.2	getPut	11
3.7.2.3	getX	11
3.7.2.4	getY	11
3.7.2.5	px	11
3.7.2.6	py	12
3.7.2.7	setDelay	13
3.7.2.8	setPut	13
3.7.2.9	setX	13
3.7.2.10	setY	13
3.8	Flame Class Reference	13
3.8.1	Constructor & Destructor Documentation	13
3.8.1.1	Flame	13
3.8.1.2	Flame	14
3.8.2	Member Function Documentation	15
3.8.2.1	expire	15

3.8.2.2	getDirection	15
3.8.2.3	getOwner	15
3.8.2.4	getPoint	15
3.8.2.5	getX	15
3.8.2.6	getY	15
3.8.2.7	state	15
3.8.3	Friends And Related Function Documentation	15
3.8.3.1	operator==	15
3.9	Frame Struct Reference	16
3.9.1	Constructor & Destructor Documentation	16
3.9.1.1	Frame	16
3.9.2	Member Data Documentation	16
3.9.2.1	obj	16
3.9.2.2	time	16
3.10	Game Class Reference	16
3.11	GameConfig Class Reference	16
3.11.1	Constructor & Destructor Documentation	17
3.11.1.1	GameConfig	17
3.11.2	Member Data Documentation	17
3.11.2.1	gen	17
3.11.2.2	init	17
3.11.2.3	nBomb	17
3.11.2.4	nFlame	17
3.11.2.5	nVirus	17
3.11.2.6	sb	17
3.11.2.7	sx	17
3.11.2.8	sy	18
3.12	GameManager Class Reference	18
3.12.1	Constructor & Destructor Documentation	18
3.12.1.1	GameManager	18
3.12.2	Member Function Documentation	18
3.12.2.1	checkAfter	18
3.12.2.2	checkBefore	18
3.12.2.3	checkState	18
3.12.2.4	getRound	18
3.12.2.5	getRun	19
3.12.2.6	newRound	19
3.12.2.7	prepareTable	19
3.12.2.8	waitRound	19
3.13	GameTable Class Reference	19

3.13.1	Constructor & Destructor Documentation	19
3.13.1.1	GameTable	19
3.13.1.2	~GameTable	20
3.13.2	Member Function Documentation	20
3.13.2.1	bind	20
3.13.2.2	buildWalls	20
3.13.2.3	countMonster	20
3.13.2.4	execute	20
3.13.2.5	generateBase	20
3.13.2.6	generateMap	20
3.13.2.7	getConfig	20
3.13.2.8	getMovement	20
3.13.2.9	getPlayers	20
3.13.2.10	incScore	21
3.13.2.11	loadMap	21
3.13.2.12	move	21
3.13.2.13	setConfig	21
3.13.2.14	setPlayers	21
3.13.2.15	setStart	21
3.13.2.16	spawnMonster	21
3.13.2.17	spawnPlayer	22
3.13.2.18	spawnWalls	22
3.14	Goods Class Reference	22
3.14.1	Constructor & Destructor Documentation	22
3.14.1.1	Goods	22
3.14.1.2	Goods	22
3.14.2	Member Function Documentation	23
3.14.2.1	effect	23
3.14.2.2	getPoint	23
3.14.2.3	getType	23
3.14.2.4	getX	23
3.14.2.5	getY	23
3.14.2.6	state	23
3.15	Ground Class Reference	23
3.15.1	Constructor & Destructor Documentation	24
3.15.1.1	Ground	24
3.15.1.2	Ground	24
3.15.2	Member Function Documentation	24
3.15.2.1	operator=	24
3.15.2.2	state	24

3.15.3 Friends And Related Function Documentation	24
3.15.3.1 operator!=	24
3.15.3.2 operator<<	24
3.15.3.3 operator==	24
3.15.3.4 operator>>	24
3.16 Input Class Reference	25
3.16.1 Constructor & Destructor Documentation	25
3.16.1.1 Input	25
3.16.1.2 ~Input	25
3.16.2 Member Function Documentation	25
3.16.2.1 getNJoy	25
3.16.2.2 getNPlayers	25
3.16.2.3 getRun	25
3.16.2.4 getRunSoft	25
3.16.2.5 operator()	25
3.16.2.6 poll	26
3.16.2.7 setDelay	26
3.16.2.8 setMode	26
3.16.2.9 setRun	26
3.16.2.10 setView	26
3.16.2.11 updateJoySticks	26
3.17 Logger Class Reference	26
3.17.1 Detailed Description	26
3.17.2 Member Function Documentation	26
3.17.2.1 error	26
3.17.2.2 info	27
3.17.2.3 setLevel	27
3.18 Matrix< T > Class Template Reference	27
3.18.1 Constructor & Destructor Documentation	27
3.18.1.1 Matrix	27
3.18.1.2 Matrix	27
3.18.2 Member Function Documentation	28
3.18.2.1 count	28
3.18.2.2 find	28
3.18.2.3 getM	28
3.18.2.4 getN	28
3.18.2.5 operator()	28
3.18.2.6 operator()	28
3.18.2.7 operator()	28
3.18.2.8 operator()	28

3.18.2.9	setSize	28
3.18.2.10	valid	28
3.18.2.11	valid	28
3.18.3	Friends And Related Function Documentation	29
3.18.3.1	operator<<	29
3.18.3.2	operator>>	29
3.19	Menu Class Reference	29
3.19.1	Constructor & Destructor Documentation	29
3.19.1.1	Menu	29
3.19.2	Member Function Documentation	29
3.19.2.1	check	29
3.19.2.2	down	29
3.19.2.3	getMode	29
3.19.2.4	mainLoop	30
3.19.2.5	up	30
3.20	MenuItem Class Reference	30
3.20.1	Constructor & Destructor Documentation	30
3.20.1.1	MenuItem	30
3.20.2	Member Data Documentation	30
3.20.2.1	mode	30
3.20.2.2	next	30
3.20.2.3	txt	30
3.21	Monster Class Reference	31
3.21.1	Detailed Description	31
3.21.2	Constructor & Destructor Documentation	31
3.21.2.1	Monster	31
3.21.3	Member Function Documentation	31
3.21.3.1	alive	31
3.21.3.2	change	31
3.21.3.3	collide	31
3.21.3.4	die	32
3.21.3.5	getPoint	32
3.21.3.6	getPointInt	32
3.21.3.7	getWall	32
3.21.3.8	getX	32
3.21.3.9	getY	32
3.21.3.10	isActive	32
3.21.3.11	move	32
3.21.3.12	setDX	32
3.21.3.13	setDY	32

3.21.3.14 setX	32
3.21.3.15 setY	33
3.21.3.16 state	33
3.22 Multi Class Reference	33
3.22.1 Constructor & Destructor Documentation	33
3.22.1.1 Multi	33
3.22.1.2 Multi	33
3.22.2 Member Function Documentation	33
3.22.2.1 draw	33
3.23 Network Class Reference	34
3.23.1 Constructor & Destructor Documentation	34
3.23.1.1 Network	34
3.23.1.2 ~Network	34
3.23.2 Member Function Documentation	34
3.23.2.1 getInternetList	34
3.23.2.2 mainLoop	34
3.23.2.3 startListen	34
3.23.2.4 stopListen	35
3.24 NetworkEvent Struct Reference	35
3.24.1 Member Data Documentation	35
3.24.1.1 e	35
3.24.1.2 id	35
3.24.1.3 x	35
3.24.1.4 y	35
3.25 Object Class Reference	35
3.25.1 Detailed Description	36
3.25.2 Constructor & Destructor Documentation	36
3.25.2.1 ~Object	36
3.25.3 Member Function Documentation	36
3.25.3.1 bind	36
3.25.3.2 draw	36
3.26 Player Class Reference	36
3.26.1 Detailed Description	37
3.26.2 Constructor & Destructor Documentation	38
3.26.2.1 Player	38
3.26.2.2 ~Player	38
3.26.3 Member Function Documentation	38
3.26.3.1 alive	38
3.26.3.2 decBombs	38
3.26.3.3 decKill	38

3.26.3.4	decKill	38
3.26.3.5	die	38
3.26.3.6	getBombs	38
3.26.3.7	getBombX	38
3.26.3.8	getBombY	38
3.26.3.9	getControl	38
3.26.3.10	getDie	39
3.26.3.11	getId	39
3.26.3.12	getKill	39
3.26.3.13	getKilled	39
3.26.3.14	getLife	39
3.26.3.15	getPoint	39
3.26.3.16	getPrevX	39
3.26.3.17	getPrevY	39
3.26.3.18	getPut	39
3.26.3.19	getScore	39
3.26.3.20	getShowKills	39
3.26.3.21	getSize	39
3.26.3.22	getSkin	40
3.26.3.23	getX	40
3.26.3.24	getY	40
3.26.3.25	incBombs	40
3.26.3.26	incKill	40
3.26.3.27	incKill	40
3.26.3.28	incKilled	40
3.26.3.29	incScore	40
3.26.3.30	incSize	40
3.26.3.31	infect	40
3.26.3.32	infect	40
3.26.3.33	init	41
3.26.3.34	move	41
3.26.3.35	move	41
3.26.3.36	novirus	41
3.26.3.37	setBombX	41
3.26.3.38	setBombY	41
3.26.3.39	setControl	41
3.26.3.40	setDie	41
3.26.3.41	setId	41
3.26.3.42	setKill	41
3.26.3.43	setKilled	42

3.26.3.44	setLife	42
3.26.3.45	setPut	42
3.26.3.46	setScore	42
3.26.3.47	setShowKills	42
3.26.3.48	setSkin	42
3.26.3.49	setStart	42
3.26.3.50	setStart	42
3.26.3.51	setX	42
3.26.3.52	setY	42
3.26.3.53	state	42
3.26.3.54	win	43
3.27	Point< T > Class Template Reference	43
3.27.1	Constructor & Destructor Documentation	43
3.27.1.1	Point	43
3.27.1.2	Point	43
3.27.2	Member Function Documentation	43
3.27.2.1	getX	43
3.27.2.2	getY	43
3.27.2.3	operator=	44
3.27.2.4	setX	45
3.27.2.5	setY	45
3.27.3	Friends And Related Function Documentation	45
3.27.3.1	distance	45
3.27.3.2	operator==	45
3.27.3.3	Way	45
3.28	Rect Struct Reference	45
3.28.1	Detailed Description	46
3.28.2	Constructor & Destructor Documentation	46
3.28.2.1	Rect	46
3.28.2.2	Rect	46
3.28.3	Member Data Documentation	46
3.28.3.1	h	46
3.28.3.2	w	46
3.28.3.3	x	46
3.28.3.4	y	46
3.29	ScoreBoard Class Reference	46
3.29.1	Constructor & Destructor Documentation	47
3.29.1.1	ScoreBoard	47
3.29.2	Member Function Documentation	48
3.29.2.1	mainLoop	48

3.30	Select Class Reference	48
3.30.1	Detailed Description	48
3.30.2	Constructor & Destructor Documentation	48
3.30.2.1	Select	48
3.30.3	Member Function Documentation	48
3.30.3.1	getN	48
3.30.3.2	getRun	49
3.30.3.3	mainLoop	49
3.31	Server Class Reference	49
3.31.1	Constructor & Destructor Documentation	49
3.31.1.1	Server	49
3.31.1.2	~Server	49
3.31.2	Member Function Documentation	49
3.31.2.1	mainLoop	49
3.32	Sprite Class Reference	49
3.32.1	Detailed Description	50
3.32.2	Constructor & Destructor Documentation	50
3.32.2.1	Sprite	50
3.32.2.2	Sprite	50
3.32.2.3	~Sprite	50
3.32.3	Member Function Documentation	50
3.32.3.1	bind	50
3.32.3.2	draw	50
3.33	Text Class Reference	51
3.33.1	Detailed Description	51
3.33.2	Constructor & Destructor Documentation	51
3.33.2.1	Text	51
3.33.2.2	Text	51
3.33.3	Member Function Documentation	52
3.33.3.1	draw	52
3.34	Translate Class Reference	52
3.35	Utf8 Class Reference	52
3.35.1	Detailed Description	52
3.35.2	Constructor & Destructor Documentation	52
3.35.2.1	Utf8	52
3.35.3	Member Function Documentation	53
3.35.3.1	isBr	53
3.35.3.2	notNull	53
3.35.3.3	notSpace	53
3.35.4	Friends And Related Function Documentation	53

3.35.4.1	operator<	53
3.35.4.2	operator<<	53
3.35.4.3	operator==	53
3.35.4.4	operator>	53
3.35.4.5	operator>>	53
3.36	View Class Reference	53
3.36.1	Detailed Description	54
3.36.2	Constructor & Destructor Documentation	54
3.36.2.1	View	54
3.36.2.2	~View	54
3.36.3	Member Function Documentation	54
3.36.3.1	draw	54
3.36.3.2	draw	54
3.36.3.3	drawAnim	55
3.36.3.4	frameBegin	55
3.36.3.5	frameDelay	55
3.36.3.6	frameEnd	55
3.36.3.7	getH	55
3.36.3.8	getNow	55
3.36.3.9	getW	55
3.36.3.10	load	55
3.36.3.11	loadPackage	55
3.36.3.12	resize_event	56
3.36.3.13	Screenshot	56
3.36.3.14	setNow	56
3.36.3.15	setTickInterval	56
3.36.3.16	swap	56
3.36.3.17	Toggle	56
3.36.3.18	toggleFullScreen	56

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Animation	5
AnimFile	6
Bomb	7
Client	9
Config	9
Control	11
Flame	13
Frame	16
Game	16
GameConfig	16
GameManager	18
GameTable	19
Goods	22
Ground	23
Input	25
Logger	26
Matrix< T >	27
Matrix< Ground >	27
Matrix< Utf8 >	27
Menu	29
MenuItem	30
Monster	31
Network	34
NetworkEvent	35
Object	35
AnimObject	6
Multi	33
Sprite	49
Text	51
Player	36
Point< T >	43
Point< float >	43
Point< int >	43
Rect	45
ScoreBoard	46
Select	48
Server	49
Translate	52

Utf8	52
View	53

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Animation	5
AnimFile	6
AnimObject	6
Bomb	7
Client	9
Config	9
Control	11
Flame	13
Frame	16
Game	16
GameConfig	16
GameManager	18
GameTable	19
Goods	22
Ground	23
Input	25
Logger	26
Matrix< T >	27
Menu	29
MenuItem	30
Monster	31
Multi	33
Network	34
NetworkEvent	35
Object	35
Player	36
Point< T >	43
Rect	45
ScoreBoard	46
Select	48
Server	49
Sprite	49
Text	51
Translate	52
Utf8	52
View	53

Chapter 3

Class Documentation

3.1 Animation Class Reference

Public Member Functions

- [Animation](#) ()
- [Animation](#) ([Rect](#) *irect*, [TIME](#) *istart*, [AnimFile](#) **ianim*, [std::string](#) *iparam*="")
- [std::string](#) [draw](#) ([TIME](#) *time*)
- [Rect](#) [getRect](#) ()

3.1.1 Constructor & Destructor Documentation

3.1.1.1 [Animation::Animation](#) () [[inline](#)]

Default constructor

3.1.1.2 [Animation::Animation](#) ([Rect](#) *irect*, [TIME](#) *istart*, [AnimFile](#) * *ianim*, [std::string](#) *iparam* = " ") [[inline](#)]

Constructor

Parameters

<i>irect</i>	rectangle where the animation will be shown
<i>istart</i>	start time of animation
<i>ianim</i>	previously loaded animation file
<i>iparam</i>	drawing parameters

3.1.2 Member Function Documentation

3.1.2.1 [std::string](#) [Animation::draw](#) ([TIME](#) *time*) [[inline](#)]

Draw animation frame regarding time

Parameters

<i>time</i>	current frame time
-------------	--------------------

Returns

graphic string

3.1.2.2 Rect Animation::getRect () [inline]

Rectangle where animation is drawn

The documentation for this class was generated from the following file:

- view/animation.h

3.2 AnimFile Class Reference

Public Member Functions

- [AnimFile](#) ()
- [AnimFile](#) (std::string filename)
- std::string [frame](#) (TIME time)

3.2.1 Constructor & Destructor Documentation

3.2.1.1 AnimFile::AnimFile () [inline]

Default constructor

3.2.1.2 AnimFile::AnimFile (std::string filename)

Constructor, loads animation phases

Parameters

<i>filename</i>	name of animation file (.anm)
-----------------	-------------------------------

3.2.2 Member Function Documentation

3.2.2.1 std::string AnimFile::frame (TIME time)

Calculate the current animation phase to show depending on current frame time

Parameters

<i>time</i>	frame time
-------------	------------

Returns

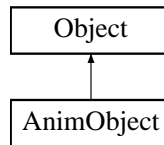
graphic name (full draw path)

The documentation for this class was generated from the following files:

- view/animfile.h
- view/animfile.cpp

3.3 AnimObject Class Reference

Inheritance diagram for AnimObject:



Public Member Functions

- [AnimObject](#) ()
- [AnimObject](#) (std::deque< [Animation](#) > *ianim, std::map< std::string, [AnimFile](#) > *ianims, std::string iname, TIME *inow)
- void [draw](#) (float x, float y, float w, float h, std::string param="")

3.3.1 Constructor & Destructor Documentation

3.3.1.1 [AnimObject::AnimObject](#) () [inline]

Default constructor

3.3.1.2 [AnimObject::AnimObject](#) (std::deque< [Animation](#) > * *ianim*, std::map< std::string, [AnimFile](#) > * *ianims*, std::string *iname*, TIME * *inow*) [inline]

Constructor

Parameters

<i>ianim</i>	deque of animations from view
<i>ianims</i>	animation files
<i>iname</i>	name of animation
<i>inow</i>	pointer to current frame time

3.3.2 Member Function Documentation

3.3.2.1 void [AnimObject::draw](#) (float *x*, float *y*, float *w*, float *h*, std::string *param* = " ") [inline], [virtual]

Draw animation

Parameters

<i>x</i>	coordinate
<i>y</i>	coordinate
<i>w</i>	width
<i>h</i>	height
<i>param</i>	graphic parameter

Reimplemented from [Object](#).

The documentation for this class was generated from the following file:

- view/animobject.h

3.4 Bomb Class Reference

Public Member Functions

- [Bomb](#) ()
- [Bomb](#) (TIME *iactivated*, [Point](#)< int > *ipoint*, [Player](#) **iplayer*)
- void [boom](#) ()
- bool [boom](#) (TIME *inow*)
- std::string [state](#) ()
- int [getX](#) ()
- int [getY](#) ()
- [Point](#)< int > [getPoint](#) ()
- int [getSize](#) ()
- int [getOwner](#) ()

3.4.1 Constructor & Destructor Documentation

3.4.1.1 [Bomb](#)::[Bomb](#) () `[inline]`

Default constructor

3.4.1.2 [Bomb](#)::[Bomb](#) (TIME *iactivated*, [Point](#)< int > *ipoint*, [Player](#) * *iplayer*) `[inline]`

Constructor

Parameters

<i>iactivated</i>	when should it become ignited
<i>ipoint</i>	place of bomb
<i>iplayer</i>	Player who put it

3.4.2 Member Function Documentation

3.4.2.1 void [Bomb](#)::[boom](#) () `[inline]`

[Bomb](#) ignites by another flame, update player's number of bombs

3.4.2.2 bool [Bomb](#)::[boom](#) (TIME *inow*) `[inline]`

[Bomb](#) ignites by time

Parameters

<i>inow</i>	current frame time
-------------	--------------------

Returns

ignites or not

3.4.2.3 int [Bomb](#)::[getOwner](#) () `[inline]`

Get owner

Returns

player ID

3.4.2.4 `Point<int> Bomb::getPoint ()` `[inline]`

Get place of bomb

3.4.2.5 `int Bomb::getSize ()` `[inline]`

Get flame size

3.4.2.6 `int Bomb::getX ()` `[inline]`

Get X coordinate

3.4.2.7 `int Bomb::getY ()` `[inline]`

Get Y coordinate

3.4.2.8 `std::string Bomb::state ()` `[inline]`

[Animation](#) state

Returns

graphic string

The documentation for this class was generated from the following file:

- `bomb.h`

3.5 Client Class Reference

Public Member Functions

- void `mainLoop ()`

The documentation for this class was generated from the following files:

- `client.h`
- `client.cpp`

3.6 Config Struct Reference

Public Attributes

- int `CFG_FLAMETIME`
- int `CFG_BOMBTIME`
- float `CFG_DEFAULT_SPEED`
- float `CFG_SLOW_SPEED`
- float `CFG_FAST_SPEED`
- int `CFG_VIRUSTIME`
- int `CFG_RESPAWN_PLAYER`
- int `CFG_RESPAWN_MONSTER`
- int `CFG_SHOW_SCOREBOARD`
- unsigned int `CFG_PORT`

3.6.1 Member Data Documentation

3.6.1.1 int Config::CFG_BOMBTIME

[Bomb](#) ticks for that time

3.6.1.2 float Config::CFG_DEFAULT_SPEED

Default speed factor

3.6.1.3 float Config::CFG_FAST_SPEED

Fast virus speed factor

3.6.1.4 int Config::CFG_FLAMETIME

[Flame](#) lasts for that time after ignited

3.6.1.5 unsigned int Config::CFG_PORT

Port for multiplayer

3.6.1.6 int Config::CFG_RESPAWN_MONSTER

[Monster](#) spawning time interval

3.6.1.7 int Config::CFG_RESPAWN_PLAYER

[Player](#) respawn time

3.6.1.8 int Config::CFG_SHOW_SCOREBOARD

Show scoreboard for given time

3.6.1.9 float Config::CFG_SLOW_SPEED

Slow virus speed factor

3.6.1.10 int Config::CFG_VIRUSTIME

Virus time

The documentation for this struct was generated from the following files:

- config.h
- config.cpp

3.7 Control Class Reference

Public Member Functions

- [Control](#) ()
- float [px](#) (float ix, float speed)
- float [py](#) (float iy, float speed)
- void [setX](#) (float x)
- void [setY](#) (float y)
- void [setDelay](#) (float d)
- void [setPut](#) (bool p)
- float [getX](#) ()
- float [getY](#) ()
- float [getDelay](#) ()
- bool [getPut](#) ()

3.7.1 Constructor & Destructor Documentation

3.7.1.1 `Control::Control ()` [\[inline\]](#)

Constructor

3.7.2 Member Function Documentation

3.7.2.1 `float Control::getDelay ()` [\[inline\]](#)

Get time difference between frames

3.7.2.2 `bool Control::getPut ()` [\[inline\]](#)

Get if the player wants to put bomb

3.7.2.3 `float Control::getX ()` [\[inline\]](#)

Get X vector

3.7.2.4 `float Control::getY ()` [\[inline\]](#)

Get Y vector

3.7.2.5 `float Control::px (float ix, float speed)` [\[inline\]](#)

Get new X coordinate regarding current position, speed and frame time

Parameters

<i>ix</i>	current X position
<i>speed</i>	speed factor

Returns

new X position

3.7.2.6 `float Control::py (float iy, float speed)` `[inline]`

Get new Y coordinate regarding current position, speed and frame time

Parameters

<i>iy</i>	current Y position
<i>speed</i>	speed factor

Returns

new Y position

3.7.2.7 `void Control::setDelay (float d)` `[inline]`

Set time difference between frames

3.7.2.8 `void Control::setPut (bool p)` `[inline]`

Set if the player wants to put bomb

3.7.2.9 `void Control::setX (float x)` `[inline]`

Set X vector

3.7.2.10 `void Control::setY (float y)` `[inline]`

Set Y vector

The documentation for this class was generated from the following file:

- control.h

3.8 Flame Class Reference

Public Member Functions

- [Flame](#) ()
- [Flame](#) (int x, int y, TIME itime, int idirection, int iOwner)
- int [getX](#) ()
- int [getY](#) ()
- [Point](#)< int > [getPoint](#) ()
- int [getDirection](#) ()
- int [getOwner](#) ()
- std::string [state](#) ()
- bool [expire](#) (TIME now)

Friends

- bool [operator==](#) ([Flame](#) &a, [Point](#)< int > &b)

3.8.1 Constructor & Destructor Documentation

3.8.1.1 `Flame::Flame ()` `[inline]`

Default constructor

3.8.1.2 Flame::Flame (int *x*, int *y*, TIME *itime*, int *idirection*, int *iOwner*)

Constructor, initialize values

Parameters

<i>x</i>	coordinate
<i>y</i>	coordinate
<i>itime</i>	flame expire time
<i>idirection</i>	direction, see getDirection()
<i>iOwner</i>	player ID

3.8.2 Member Function Documentation

3.8.2.1 bool Flame::expire (TIME *now*)

Whether flame is expired or not

Parameters

<i>now</i>	current frame time
------------	--------------------

3.8.2.2 int Flame::getDirection ()

Get direction defined.

Returns

direction

3.8.2.3 int Flame::getOwner ()

Return owner of flame

3.8.2.4 Point< int > Flame::getPoint ()

Get point

3.8.2.5 int Flame::getX ()

Get X coordinate

3.8.2.6 int Flame::getY ()

Get Y coordinate

3.8.2.7 std::string Flame::state ()

Get animation state, returns draw string

3.8.3 Friends And Related Function Documentation

3.8.3.1 bool operator== (Flame & *a*, Point< int > & *b*) [friend]

Equality operator

The documentation for this class was generated from the following files:

- flame.h
- flame.cpp

3.9 Frame Struct Reference

Public Member Functions

- [Frame](#) (std::string iobj, TIME itime)

Public Attributes

- TIME [time](#)
- std::string [obj](#)

3.9.1 Constructor & Destructor Documentation

3.9.1.1 `Frame::Frame (std::string iobj, TIME itime) [inline]`

Constructor

Parameters

<i>iobj</i>	graphic to show
<i>itime</i>	delay after previous

3.9.2 Member Data Documentation

3.9.2.1 `std::string Frame::obj`

Graphic to show

3.9.2.2 `TIME Frame::time`

Delay after previous frame

The documentation for this struct was generated from the following file:

- view/animfile.h

3.10 Game Class Reference

The documentation for this class was generated from the following files:

- game.h
- game.cpp

3.11 GameConfig Class Reference

Public Member Functions

- [GameConfig](#) ()

Public Attributes

- int [sx](#)
- int [sy](#)
- int [sb](#)
- bool [gen](#)
- bool [init](#)
- int [nVirus](#)
- int [nBomb](#)
- int [nFlame](#)

3.11.1 Constructor & Destructor Documentation

3.11.1.1 GameConfig::GameConfig () [\[inline\]](#)

Default constructor

3.11.2 Member Data Documentation

3.11.2.1 bool GameConfig::gen

Generate map

3.11.2.2 bool GameConfig::init

Initialize map

3.11.2.3 int GameConfig::nBomb

Number of bomb size boosters on map

3.11.2.4 int GameConfig::nFlame

Number of flame size boosters on map

3.11.2.5 int GameConfig::nVirus

Number of viruses on map

3.11.2.6 int GameConfig::sb

A squares size on map

3.11.2.7 int GameConfig::sx

Number of columns on map

3.11.2.8 int GameConfig::sy

Number of rows on map

The documentation for this class was generated from the following file:

- gameconfig.h

3.12 GameManager Class Reference

Public Member Functions

- [GameManager](#) (GAME_MODE_TYPE iGameMode, [View](#) &iview, [Input](#) &iInput, [GameTable](#) &iTable)
- void [prepareTable](#) ()
- void [newRound](#) ()
- void [checkState](#) ()
- void [checkBefore](#) ()
- void [checkAfter](#) ()
- void [waitRound](#) (TIME t=1000)
- bool [getRun](#) ()
- bool [getRound](#) ()

3.12.1 Constructor & Destructor Documentation

3.12.1.1 GameManager::GameManager (GAME_MODE_TYPE iGameMode, View & iview, Input & iInput, GameTable & iTable)

Constructor

Parameters

<i>iGameMode</i>	game mode
<i>iview</i>	View class
<i>iInput</i>	Input class
<i>iTable</i>	Table class

3.12.2 Member Function Documentation

3.12.2.1 void GameManager::checkAfter ()

Check after rounds

3.12.2.2 void GameManager::checkBefore ()

Check after frames

3.12.2.3 void GameManager::checkState ()

Check state of game (score of players) and stop if necessary

3.12.2.4 bool GameManager::getRound ()

End of round

3.12.2.5 bool GameManager::getRun ()

End of this game

3.12.2.6 void GameManager::newRound ()

Start new round, reset table

3.12.2.7 void GameManager::prepareTable ()

Prepare game table

3.12.2.8 void GameManager::waitRound (TIME $t = 1000$)

Wait for round start

The documentation for this class was generated from the following files:

- gamemanager.h
- gamemanager.cpp

3.13 GameTable Class Reference

Public Member Functions

- [GameTable](#) ([View](#) &iView)
- [~GameTable](#) ()
- void [setPlayers](#) (int n)
- std::string [loadMap](#) (std::string file)
- void [generateMap](#) ()
- void [generateBase](#) ()
- void [setStart](#) (int x, int y, int i)
- void [spawnPlayer](#) (int i)
- void [spawnMonster](#) (int i=-1, int x=-1, int y=-1)
- void [spawnWalls](#) (int limit)
- void [buildWalls](#) ()
- void [move](#) ()
- void [setConfig](#) ([GameConfig](#) &conf)
- [GameConfig](#) & [getConfig](#) ()
- const std::vector< [Player](#) > & [getPlayers](#) () const
- void [incScore](#) (int pld)
- int [countMonster](#) () const
- void [bind](#) (int i, [Control](#) *c)
- void [execute](#) ([NetworkEvent](#) event)
- [NetworkEvent](#) [getMovement](#) ()

3.13.1 Constructor & Destructor Documentation

3.13.1.1 GameTable::GameTable (View & iView)

Constructor

Parameters

<i>iView</i>	View class
--------------	----------------------------

3.13.1.2 `GameTable::~~GameTable () [inline]`

Destructor

3.13.2 Member Function Documentation

3.13.2.1 `void GameTable::bind (int i, Control * c) [inline]`

Bind player to [Control](#) class

Parameters

<i>i</i>	player ID
<i>c</i>	Control class

3.13.2.2 `void GameTable::buildWalls ()`

If the wall is build, then put it on the map (after animation)

3.13.2.3 `int GameTable::countMonster () const [inline]`

Count alive monsters

3.13.2.4 `void GameTable::execute (NetworkEvent event)`

Execute remote command

3.13.2.5 `void GameTable::generateBase ()`

Generate an empty map

3.13.2.6 `void GameTable::generateMap ()`

Generate a random map

3.13.2.7 `GameConfig& GameTable::getConfig () [inline]`

Get configuration of map

3.13.2.8 `NetworkEvent GameTable::getMovement ()`

Get movement of player for network

3.13.2.9 `const std::vector<Player>& GameTable::getPlayers () const [inline]`

Get players array (const)

3.13.2.10 void GameTable::incScore (int *pld*) [inline]

Increase score of player

Parameters

<i>pld</i>	player ID
------------	-----------

3.13.2.11 std::string GameTable::loadMap (std::string *file*)

Load map from file

Parameters

<i>file</i>	file name
-------------	-----------

Returns

name of the next map

3.13.2.12 void GameTable::move ()

Move things on map and check collision

3.13.2.13 void GameTable::setConfig (GameConfig & *conf*)

Set configuration for map

3.13.2.14 void GameTable::setPlayers (int *n*)

Set number of players

3.13.2.15 void GameTable::setStart (int *x*, int *y*, int *i*)

Set start position of player

Parameters

<i>x</i>	coordinate x
<i>y</i>	coordinate y
<i>i</i>	player ID

3.13.2.16 void GameTable::spawnMonster (int *i* = -1, int *x* = -1, int *y* = -1)

Spawn monster. Call it with -1 to get random position or type.

Parameters

<i>i</i>	type of monster
<i>x</i>	coordinate x

<i>y</i>	coordinate y
----------	--------------

3.13.2.17 void GameTable::spawnPlayer (int *i*)

Spawn player on map

Parameters

<i>i</i>	player ID
----------	-----------

3.13.2.18 void GameTable::spawnWalls (int *limit*)

Start building maximum given number of walls on map

Parameters

<i>limit</i>	maximum number of walls
--------------	-------------------------

The documentation for this class was generated from the following files:

- gametable.h
- gametable.cpp

3.14 Goods Class Reference

Public Member Functions

- [Goods](#) ()
- [Goods](#) ([Point](#)< int > *ipoint*, GOODS *itype*)
- int [getX](#) ()
- int [getY](#) ()
- [Point](#)< int > [getPoint](#) ()
- GOODS [getType](#) ()
- std::string [state](#) ()
- std::string [effect](#) ([Player](#) *p)

3.14.1 Constructor & Destructor Documentation

3.14.1.1 Goods::Goods () [[inline](#)]

Default constructor

3.14.1.2 Goods::Goods ([Point](#)< int > *ipoint*, GOODS *itype*) [[inline](#)]

Constructor

Parameters

<i>ipoint</i>	place of good
---------------	---------------

<i>itype</i>	type of good (virus, bomb+, flame+)
--------------	-------------------------------------

3.14.2 Member Function Documentation

3.14.2.1 `std::string Goods::effect (Player * p)` `[inline]`

Apply effect on player

Parameters

<i>p</i>	pointer to player
----------	-------------------

Returns

text to show

3.14.2.2 `Point<int> Goods::getPoint ()` `[inline]`

Get point

3.14.2.3 `GOODS Goods::getType ()` `[inline]`

Get type of good

3.14.2.4 `int Goods::getX ()` `[inline]`

Get X coordinate

3.14.2.5 `int Goods::getY ()` `[inline]`

Get Y coordinate

3.14.2.6 `std::string Goods::state ()` `[inline]`

Get graphic file name of good

The documentation for this class was generated from the following file:

- goods.h

3.15 Ground Class Reference

Public Member Functions

- [Ground](#) ()
- [Ground](#) (GROUND_TYPE ig)
- `std::string` [state](#) ()
- [Ground](#) & [operator=](#) (const GROUND_TYPE &i)

Friends

- `bool operator== (const Ground &a, const Ground &b)`
- `bool operator!= (const Ground &a, const Ground &b)`
- `std::istream & operator>> (std::istream &input, Ground &ground)`
- `std::ostream & operator<< (std::ostream &output, const Ground &ground)`

3.15.1 Constructor & Destructor Documentation

3.15.1.1 `Ground::Ground ()` `[inline]`

Default constructor

3.15.1.2 `Ground::Ground (GROUND_TYPE ig)` `[inline]`

Constructor

Parameters

<i>ig</i>	ground type
-----------	-------------

3.15.2 Member Function Documentation

3.15.2.1 `Ground& Ground::operator= (const GROUND_TYPE & i)` `[inline]`

Equality operator

3.15.2.2 `std::string Ground::state ()` `[inline]`

Graphic file name of current ground

3.15.3 Friends And Related Function Documentation

3.15.3.1 `bool operator!= (const Ground & a, const Ground & b)` `[friend]`

Check not equals

3.15.3.2 `std::ostream& operator<< (std::ostream & output, const Ground & ground)` `[friend]`

Output stream operator

3.15.3.3 `bool operator== (const Ground & a, const Ground & b)` `[friend]`

Check equality

3.15.3.4 `std::istream& operator>> (std::istream & input, Ground & ground)` `[friend]`

`Input` stream operator

The documentation for this class was generated from the following file:

- `ground.h`

3.16 Input Class Reference

Public Member Functions

- [Input](#) ()
- [~Input](#) ()
- void [updateJoySticks](#) ()
- bool [getRun](#) ()
- bool [getRunSoft](#) ()
- int [getNJoy](#) ()
- int [getNPlayers](#) ()
- void [setRun](#) (bool irun)
- void [setView](#) ([View](#) *iView)
- void [setMode](#) (bool m)
- [Control](#) * [operator\(\)](#) (int i)
- void [poll](#) (TIME delay)
- void [setDelay](#) (TIME delay)

3.16.1 Constructor & Destructor Documentation

3.16.1.1 `Input::Input ()`

Constructor, initialize controls

3.16.1.2 `Input::~~Input ()`

Destructor, release joysticks

3.16.2 Member Function Documentation

3.16.2.1 `int Input::getNJoy ()` `[inline]`

Get number of joysticks

3.16.2.2 `int Input::getNPlayers ()` `[inline]`

Get number of players

3.16.2.3 `bool Input::getRun ()` `[inline]`

Program should run

3.16.2.4 `bool Input::getRunSoft ()` `[inline]`

Program should step back one level

3.16.2.5 `Control* Input::operator() (int i)` `[inline]`

Get [Control](#) class of selected ID

3.16.2.6 void Input::poll (TIME *delay*)

Poll events with SDL

3.16.2.7 void Input::setDelay (TIME *delay*)

Set delay between frames

3.16.2.8 void Input::setMode (bool *m*) [inline]

Set mode: 0:menu, 1:in-game

3.16.2.9 void Input::setRun (bool *irun*) [inline]

Set run

3.16.2.10 void Input::setView (View * *iView*) [inline]

Set [View](#) class

3.16.2.11 void Input::updateJoySticks ()

Update joysticks, search for newly connected

The documentation for this class was generated from the following files:

- input.h
- input.cpp

3.17 Logger Class Reference

```
#include <logger.h>
```

Static Public Member Functions

- static void [setLevel](#) (const int *_level*)
- static void [info](#) (const std::string &*str*)
- static void [error](#) (const std::string &*str*)

3.17.1 Detailed Description

[Logger](#) class

3.17.2 Member Function Documentation

3.17.2.1 static void Logger::error (const std::string & *str*) [inline],[static]

Trace error

3.17.2.2 `static void Logger::info (const std::string & str)` `[inline]`, `[static]`

Trace info

3.17.2.3 `static void Logger::setLevel (const int _level)` `[inline]`, `[static]`

Set error level

The documentation for this class was generated from the following file:

- `logger.h`

3.18 Matrix< T > Class Template Reference

Public Member Functions

- `Point< int > find (T &s)`
- `Matrix ()`
- `Matrix (int in, int im)`
- `int getN ()`
- `int getM ()`
- `void setSize (int in, int im)`
- `bool valid (Point< int > p)`
- `bool valid (int x, int y)`
- `unsigned int count (const T &e)`
- `T & operator() (Point< int > p)`
- `T & operator() (int i, int j)`
- `const T & operator() (Point< int > p) const`
- `const T & operator() (int i, int j) const`

Friends

- `std::istream & operator>> (std::istream &input, Matrix< T > &p)`
- `std::ostream & operator<< (std::ostream &output, const Matrix< T > &p)`

3.18.1 Constructor & Destructor Documentation

3.18.1.1 `template<typename T> Matrix< T >::Matrix ()` `[inline]`

Default constructor

3.18.1.2 `template<typename T> Matrix< T >::Matrix (int in, int im)` `[inline]`

Constructor

Parameters

<i>in</i>	number of rows
<i>im</i>	number of columns

3.18.2 Member Function Documentation

3.18.2.1 `template<typename T> unsigned int Matrix< T >::count (const T & e)` `[inline]`

Count an element's number of occurrences in matrix

3.18.2.2 `template<typename T> Point<int> Matrix< T >::find (T & s)` `[inline]`

Find the first occurrence of element in matrix

3.18.2.3 `template<typename T> int Matrix< T >::getM ()` `[inline]`

Number of columns

3.18.2.4 `template<typename T> int Matrix< T >::getN ()` `[inline]`

Number of rows

3.18.2.5 `template<typename T> T& Matrix< T >::operator() (Point< int > p)` `[inline]`

Get element at point

3.18.2.6 `template<typename T> T& Matrix< T >::operator() (int i, int j)` `[inline]`

Get element at coordinates

3.18.2.7 `template<typename T> const T& Matrix< T >::operator() (Point< int > p) const` `[inline]`

Get element at point (const)

3.18.2.8 `template<typename T> const T& Matrix< T >::operator() (int i, int j) const` `[inline]`

Get element at coordinates (const)

3.18.2.9 `template<typename T> void Matrix< T >::setSize (int in, int im)` `[inline]`

Set size of matrix

3.18.2.10 `template<typename T> bool Matrix< T >::valid (Point< int > p)` `[inline]`

The given point is in a valid range or not

3.18.2.11 `template<typename T> bool Matrix< T >::valid (int x, int y)` `[inline]`

The given coordinates is in a valid range or not

3.18.3 Friends And Related Function Documentation

3.18.3.1 `template<typename T> std::ostream& operator<< (std::ostream & output, const Matrix< T > & p)`
`[friend]`

Output stream operator

3.18.3.2 `template<typename T> std::istream& operator>> (std::istream & input, Matrix< T > & p)` `[friend]`

Input stream operator

The documentation for this class was generated from the following file:

- `view/matrix.h`

3.19 Menu Class Reference

Public Member Functions

- `Menu (View &iView, Input &iInput)`
- `GAME_MODE_TYPE getMode ()`
- `bool mainLoop ()`
- `void down ()`
- `void up ()`
- `void check ()`

3.19.1 Constructor & Destructor Documentation

3.19.1.1 `Menu::Menu (View & iView, Input & iInput)`

Constructor

Parameters

<i>iView</i>	View class
<i>iInput</i>	Input class

3.19.2 Member Function Documentation

3.19.2.1 `void Menu::check ()`

Check time between the last change of menu items and now, step to next if necessary

3.19.2.2 `void Menu::down ()`

Go down one item

3.19.2.3 `GAME_MODE_TYPE Menu::getMode ()`

Get selected game mode

3.19.2.4 `bool Menu::mainLoop ()`

Main loop, draws on screen and checks input

3.19.2.5 `void Menu::up ()`

Go up one item

The documentation for this class was generated from the following files:

- `menu.h`
- `menu.cpp`

3.20 Menuitem Class Reference

Public Member Functions

- [Menuitem](#) (`std::string inp`, `int inext`, `GAME_MODE_TYPE imode`)

Public Attributes

- `int` [next](#)
- `GAME_MODE_TYPE` [mode](#)
- `std::string` [txt](#)

3.20.1 Constructor & Destructor Documentation

3.20.1.1 `Menuitem::Menuitem (std::string inp, int inext, GAME_MODE_TYPE imode)` `[inline]`

Constructor

3.20.2 Member Data Documentation

3.20.2.1 `GAME_MODE_TYPE Menuitem::mode`

[Game](#) mode

3.20.2.2 `int Menuitem::next`

Next menu item when selected

3.20.2.3 `std::string Menuitem::txt`

[Text](#) to show

The documentation for this class was generated from the following file:

- `menu.h`

3.21 Monster Class Reference

```
#include <monster.h>
```

Public Member Functions

- [Monster](#) (int itype, int isb, [Point](#)< float > ipoint)
- bool [isActive](#) ()
- bool [collide](#) ()
- std::string [die](#) ()
- bool [alive](#) ()
- void [move](#) (TIME now)
- bool [change](#) ()
- void [setX](#) (float x)
- void [setY](#) (float y)
- void [setDX](#) (float dx)
- void [setDY](#) (float dy)
- [Point](#)< float > [getPoint](#) ()
- [Point](#)< int > [getPointInt](#) ()
- float [getX](#) ()
- float [getY](#) ()
- std::string [state](#) ()
- bool [getWall](#) ()

3.21.1 Detailed Description

This class contains a monster

3.21.2 Constructor & Destructor Documentation

3.21.2.1 [Monster::Monster](#) (int *itype*, int *isb*, [Point](#)< float > *ipoint*)

Constructor

Parameters

<i>itype</i>	type of monster
<i>isb</i>	size of a brick on map, necessary for random way changes
<i>ipoint</i>	starting point

3.21.3 Member Function Documentation

3.21.3.1 [bool Monster::alive](#) ()

Is alive

3.21.3.2 [bool Monster::change](#) ()

Change direction

3.21.3.3 [bool Monster::collide](#) ()

Is colliding with wall

3.21.3.4 `std::string Monster::die ()`

Dies

Returns

animation of dying

3.21.3.5 `Point< float > Monster::getPoint ()`

Get exact point on screen

3.21.3.6 `Point< int > Monster::getPointInt ()`

Get rounded point, which rectangle is the nearest

3.21.3.7 `bool Monster::getWall ()`

Return that the monster can walk through walls or not

3.21.3.8 `float Monster::getX ()`

Get exact X coordinate

3.21.3.9 `float Monster::getY ()`

Get exact Y coordinate

3.21.3.10 `bool Monster::isActive ()`

Not yet active, false for a few seconds when the class is created

3.21.3.11 `void Monster::move (TIME now)`

Move the monster, with calculating the difference between current and previous frame time

Parameters

<i>now</i>	current frame time
------------	--------------------

3.21.3.12 `void Monster::setDX (float dx)`

Set direction X

3.21.3.13 `void Monster::setDY (float dy)`

Set direction Y

3.21.3.14 `void Monster::setX (float x)`

Set X coordinate

3.21.3.15 void Monster::setY (float y)

Set Y coordinate

3.21.3.16 std::string Monster::state ()

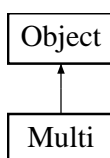
Get animation state

The documentation for this class was generated from the following files:

- monster.h
- monster.cpp

3.22 Multi Class Reference

Inheritance diagram for Multi:



Public Member Functions

- [Multi](#) ()
- [Multi](#) (std::string fileName, std::map< std::string, [Object](#) * > *igraphs)
- void [draw](#) (float x, float y, float w, float h, std::string param)

3.22.1 Constructor & Destructor Documentation

3.22.1.1 Multi::Multi () [inline]

Default constructor

3.22.1.2 Multi::Multi (std::string *fileName*, std::map< std::string, [Object](#) * > * *igraphs*) [inline]

Constructor

Parameters

<i>fileName</i>	name of multi file (.mlt)
<i>igraphs</i>	list of loaded graphics from View class

3.22.2 Member Function Documentation

3.22.2.1 void Multi::draw (float x, float y, float w, float h, std::string *param*) [virtual]

Draw a specific slice of a [Sprite](#)

Parameters

<i>x</i>	coordinate
<i>y</i>	coordinate
<i>w</i>	width
<i>h</i>	height
<i>param</i>	full string param, determines which slice need to be drawn

Reimplemented from [Object](#).

The documentation for this class was generated from the following files:

- view/multi.h
- view/multi.cpp

3.23 Network Class Reference

Public Member Functions

- [Network](#) ()
- [~Network](#) ()
- void [getInternetList](#) ()
- void [startListen](#) ()
- void [stopListen](#) ()
- void [mainLoop](#) ()

3.23.1 Constructor & Destructor Documentation

3.23.1.1 [Network::Network](#) ()

Constructor

3.23.1.2 [Network::~~Network](#) ()

Destructor

3.23.2 Member Function Documentation

3.23.2.1 [void Network::getInternetList](#) ()

Get list of Internet servers

3.23.2.2 [void Network::mainLoop](#) ()

Main loop, for doing the work

3.23.2.3 [void Network::startListen](#) ()

Start server

3.23.2.4 void Network::stopListen ()

Stop server

The documentation for this class was generated from the following files:

- network.h
- network.cpp

3.24 NetworkEvent Struct Reference

Public Attributes

- EVENT_TYPE [e](#)
- int [id](#)
- float [x](#)
- float [y](#)

3.24.1 Member Data Documentation

3.24.1.1 EVENT_TYPE NetworkEvent::e

Type of event

3.24.1.2 int NetworkEvent::id

ID of player

3.24.1.3 float NetworkEvent::x

Coordinate X

3.24.1.4 float NetworkEvent::y

Coordinate Y

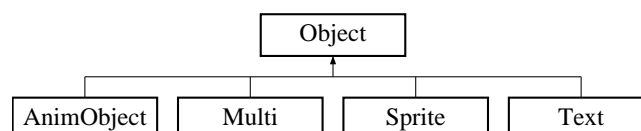
The documentation for this struct was generated from the following file:

- networkevent.h

3.25 Object Class Reference

```
#include <object.h>
```

Inheritance diagram for Object:



Public Member Functions

- virtual void [draw](#) (float x, float y, float w, float h, std::string param="")
- virtual void [bind](#) ()
- virtual [~Object](#) ()

3.25.1 Detailed Description

Base class of drawable objects

3.25.2 Constructor & Destructor Documentation

3.25.2.1 virtual [Object::~Object](#) () `[inline],[virtual]`

Destructor

3.25.3 Member Function Documentation

3.25.3.1 virtual void [Object::bind](#) () `[inline],[virtual]`

Bind graphic to draw texture

Reimplemented in [Sprite](#).

3.25.3.2 virtual void [Object::draw](#) (float x, float y, float w, float h, std::string *param* = " ") `[inline],[virtual]`

Drawing object on screen

Parameters

<i>x</i>	coordinate
<i>y</i>	coordinate
<i>w</i>	width
<i>h</i>	height
<i>param</i>	full drawing string

Reimplemented in [Sprite](#), [Text](#), [AnimObject](#), and [Multi](#).

The documentation for this class was generated from the following file:

- view/object.h

3.26 Player Class Reference

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

Public Member Functions

- [Player](#) ()
- [~Player](#) ()
- void [init](#) ()
- void [novirus](#) ()
- void [move](#) (TIME inow)
- void [move](#) ([Point](#)< float > p)

- int `infect` ()
- int `infect` (int type)
- void `die` ()
- void `win` ()
- std::string `state` ()
- bool `getPut` ()
- int `getId` () const
- `Control` * `getControl` ()
- `Point`< float > `getPoint` ()
- float `getX` ()
- float `getY` ()
- float `getPrevX` ()
- float `getPrevY` ()
- int `getBombX` ()
- int `getBombY` ()
- int `getSize` ()
- int `getBombs` ()
- int `getLife` ()
- bool `alive` () const
- int `getScore` () const
- int `getKill` () const
- int `getKilled` () const
- std::string `getSkin` () const
- TIME `getDie` () const
- TIME `getShowKills` () const
- void `setId` (int i)
- void `setControl` (`Control` *icontrol)
- void `setX` (float x)
- void `setY` (float y)
- void `setBombX` (int x)
- void `setBombY` (int y)
- void `incSize` ()
- void `incBombs` ()
- void `decBombs` ()
- void `setLife` (int l)
- void `setPut` (bool p)
- void `setScore` (int s)
- void `incScore` ()
- void `setKill` (int k)
- void `incKill` ()
- void `incKill` (TIME iShowKills)
- void `decKill` ()
- void `decKill` (TIME iShowKills)
- void `setShowKills` (TIME iShowKills)
- void `setKilled` (int k)
- void `incKilled` ()
- void `setSkin` (std::string iSkin)
- void `setDie` (TIME idie)
- void `setStart` (`Point`< float > p)
- void `setStart` ()

3.26.1 Detailed Description

This class contains all information about a player

3.26.2 Constructor & Destructor Documentation

3.26.2.1 `Player::Player ()`

Constructor, initialize player

3.26.2.2 `Player::~~Player ()`

Destructor

3.26.3 Member Function Documentation

3.26.3.1 `bool Player::alive () const` [inline]

Return if the player is alive or not

3.26.3.2 `void Player::decBombs ()` [inline]

Decrease number of bombs

3.26.3.3 `void Player::decKill ()` [inline]

Decrease kills (when suicide)

3.26.3.4 `void Player::decKill (TIME iShowKills)` [inline]

Decrease kills (when suicide) with showing the statistics

Parameters

<i>iShowKills</i>	frame time when it should disappear
-------------------	-------------------------------------

3.26.3.5 `void Player::die ()`

[Player](#) dies

3.26.3.6 `int Player::getBombs ()` [inline]

Get number of bombs

3.26.3.7 `int Player::getBombX ()` [inline]

Get the X coordinate where the player put bomb

3.26.3.8 `int Player::getBombY ()` [inline]

Get the Y coordinate where the player put bomb

3.26.3.9 `Control* Player::getControl ()` [inline]

Returns the [Control](#) class of player

3.26.3.10 `TIME Player::getDie () const [inline]`

Return the frame time when player died

3.26.3.11 `int Player::getId () const [inline]`

Return the ID of the player

3.26.3.12 `int Player::getKill () const [inline]`

Get the number of kills

3.26.3.13 `int Player::getKilled () const [inline]`

Get the number of get killed

3.26.3.14 `int Player::getLife () [inline]`

Get life of player

3.26.3.15 `Point<float> Player::getPoint () [inline]`

Get position of player

3.26.3.16 `float Player::getPrevX () [inline]`

Get previous X coordinate of player

3.26.3.17 `float Player::getPrevY () [inline]`

Get previous Y coordinate of player

3.26.3.18 `bool Player::getPut ()`

Return that the player puts bomb or not

3.26.3.19 `int Player::getScore () const [inline]`

Get score of player

3.26.3.20 `TIME Player::getShowKills () const [inline]`

Return when showing kill statistic should disappear

3.26.3.21 `int Player::getSize () [inline]`

Get flame size

3.26.3.22 `std::string Player::getSkin () const` `[inline]`

Get skin of player (prefix for graphic files)

3.26.3.23 `float Player::getX ()` `[inline]`

Get X coordinate of player

3.26.3.24 `float Player::getY ()` `[inline]`

Get Y coordinate of player

3.26.3.25 `void Player::incBombs ()` `[inline]`

Increase number of bombs

3.26.3.26 `void Player::incKill ()` `[inline]`

Increase kills

3.26.3.27 `void Player::incKill (TIME iShowKills)` `[inline]`

Increase kills with showing the statistics

Parameters

<i>iShowKills</i>	frame time when it should disappear
-------------------	-------------------------------------

3.26.3.28 `void Player::incKilled ()` `[inline]`

Increase killed by other

3.26.3.29 `void Player::incScore ()` `[inline]`

Increase score

3.26.3.30 `void Player::incSize ()` `[inline]`

Increase flame size

3.26.3.31 `int Player::infect ()`

Infect player with random virus

3.26.3.32 `int Player::infect (int type)`

Infect player with specific virus

3.26.3.33 void Player::init ()

Set values to default, including position and anything that effected the player in previous rounds

3.26.3.34 void Player::move (TIME *inow*)

Move the player to the direction given by [Control](#) class, with calculating the time difference, so the frame rate doesn't matter. For local players.

Parameters

<i>inow</i>	current frame time
-------------	--------------------

3.26.3.35 void Player::move (Point< float > *p*)

Move the player to a specific point. For remote players.

Parameters

<i>p</i>	point where it moves
----------	----------------------

3.26.3.36 void Player::novirus ()

Reset virus attributes

3.26.3.37 void Player::setBombX (int *x*) [inline]

Set X coordinate for the last bomb put

3.26.3.38 void Player::setBombY (int *y*) [inline]

Set Y coordinate for the last bomb put

3.26.3.39 void Player::setControl (Control * *icontrol*) [inline]

Set [Control](#) class for player, only one can be active per player

3.26.3.40 void Player::setDie (TIME *idie*) [inline]

Set frame time when the player died

3.26.3.41 void Player::setId (int *i*) [inline]

Set id of player

3.26.3.42 void Player::setKill (int *k*) [inline]

Set kills

3.26.3.43 void Player::setKilled (int *k*) [inline]

Set killed by other

3.26.3.44 void Player::setLife (int *l*) [inline]

Set life

3.26.3.45 void Player::setPut (bool *p*) [inline]

Set put

3.26.3.46 void Player::setScore (int *s*) [inline]

Set score

3.26.3.47 void Player::setShowKills (TIME *iShowKills*) [inline]

Set end time of showing statistics

3.26.3.48 void Player::setSkin (std::string *iSkin*) [inline]

Set skin of player

3.26.3.49 void Player::setStart (Point< float > *p*) [inline]

Set the start point of player

3.26.3.50 void Player::setStart () [inline]

Position the player to the start point already set

3.26.3.51 void Player::setX (float *x*) [inline]

Set X coordinate

3.26.3.52 void Player::setY (float *y*) [inline]

Set Y coordinate

3.26.3.53 std::string Player::state ()

Returns the current animation phase of player

Returns

graphic name

3.26.3.54 void Player::win ()

Player wins

The documentation for this class was generated from the following files:

- player.h
- player.cpp

3.27 Point< T > Class Template Reference

Public Member Functions

- [Point](#) ()
- [Point](#) (T ix, T iy)
- T [getX](#) () const
- T [getY](#) () const
- void [setX](#) (T ix)
- void [setY](#) (T iy)
- [Point](#)< T > & [operator=](#) (const [Point](#)< T > &a)

Friends

- bool [operator==](#) (const [Point](#)< T > &a, const [Point](#)< T > &b)
- T [distance](#) (const [Point](#)< T > &a, const [Point](#)< T > &b)
- WAY [Way](#) (const [Point](#)< T > &a, const [Point](#)< T > &b)

3.27.1 Constructor & Destructor Documentation

3.27.1.1 template<typename T> [Point](#)< T >::Point () [inline]

Default constructor

3.27.1.2 template<typename T> [Point](#)< T >::Point (T ix, T iy) [inline]

Constructor

Parameters

<i>ix</i>	coordinate x
<i>iy</i>	coordinate y

3.27.2 Member Function Documentation

3.27.2.1 template<typename T> T [Point](#)< T >::getX () const [inline]

Getter for X coordinate

3.27.2.2 template<typename T> T [Point](#)< T >::getY () const [inline]

Getter for Y coordinate

3.27.2.3 `template<typename T> Point<T>& Point< T >::operator= (const Point< T > & a)` `[inline]`

Equals operator

Parameters

<i>a</i>	input
----------	-------

Returns

the class

3.27.2.4 `template<typename T> void Point< T >::setX (T ix) [inline]`

Setter for X coordinate

3.27.2.5 `template<typename T> void Point< T >::setY (T iy) [inline]`

Setter for Y coordinate

3.27.3 Friends And Related Function Documentation

3.27.3.1 `template<typename T> T distance (const Point< T > & a, const Point< T > & b) [friend]`

Distance between two points

3.27.3.2 `template<typename T> bool operator== (const Point< T > & a, const Point< T > & b) [friend]`

Equality check

3.27.3.3 `template<typename T> WAY Way (const Point< T > & a, const Point< T > & b) [friend]`

A special function, calculates the direction of a vector

The documentation for this class was generated from the following file:

- point.h

3.28 Rect Struct Reference

```
#include <rect.h>
```

Public Member Functions

- [Rect](#) ()
- [Rect](#) (float ix, float iy, float iw, float ih)

Public Attributes

- float [x](#)
- float [y](#)
- float [w](#)
- float [h](#)

3.28.1 Detailed Description

This class stores a rectangle with float coordinates and dimensions

3.28.2 Constructor & Destructor Documentation

3.28.2.1 Rect::Rect () [inline]

Default constructor

3.28.2.2 Rect::Rect (float *ix*, float *iy*, float *iw*, float *ih*) [inline]

Constructor

Parameters

<i>ix</i>	coordinate
<i>iy</i>	coordinate
<i>iw</i>	width
<i>ih</i>	height

3.28.3 Member Data Documentation

3.28.3.1 float Rect::h

Height

3.28.3.2 float Rect::w

Width

3.28.3.3 float Rect::x

Coordinate X

3.28.3.4 float Rect::y

Coordinate Y

The documentation for this struct was generated from the following file:

- view/rect.h

3.29 ScoreBoard Class Reference

Public Member Functions

- [ScoreBoard](#) ([View](#) &iView, [Input](#) &iInput, [GameTable](#) &iTable)
- bool [mainLoop](#) ()

3.29.1 Constructor & Destructor Documentation

3.29.1.1 ScoreBoard::ScoreBoard (View & *iView*, Input & *ilnput*, GameTable & *iTable*)

Constructor

Parameters

<i>iView</i>	View class
<i>iInput</i>	Input class
<i>iTable</i>	Table class

3.29.2 Member Function Documentation

3.29.2.1 bool ScoreBoard::mainLoop ()

Draws the scoreboard on screen after a round

Returns

is not finished

The documentation for this class was generated from the following files:

- scoreboard.h
- scoreboard.cpp

3.30 Select Class Reference

```
#include <select.h>
```

Public Member Functions

- [Select](#) ([View](#) &iview, [Input](#) &iinput, [GameTable](#) &itable)
- bool [mainLoop](#) ()
- bool [getRun](#) ()
- int [getN](#) ()

3.30.1 Detailed Description

This class allows the users to select controllers before starting the game

3.30.2 Constructor & Destructor Documentation

3.30.2.1 Select::Select ([View](#) & *iview*, [Input](#) & *iinput*, [GameTable](#) & *itable*)

Constructor

Parameters

<i>iview</i>	View class
<i>iinput</i>	Input class
<i>itable</i>	Table class

3.30.3 Member Function Documentation

3.30.3.1 int Select::getN () [inline]

Getter for number of players

3.30.3.2 `bool Select::getRun () [inline]`

Getter for run

3.30.3.3 `bool Select::mainLoop ()`

The main loop, shows the menu for selecting input

Returns

selecting is not finished

The documentation for this class was generated from the following files:

- `select.h`
- `select.cpp`

3.31 Server Class Reference

Public Member Functions

- [Server \(\)](#)
- [~Server \(\)](#)
- void [mainLoop \(\)](#)

3.31.1 Constructor & Destructor Documentation

3.31.1.1 `Server::Server ()`

Constructor

3.31.1.2 `Server::~~Server ()`

Destructor

3.31.2 Member Function Documentation

3.31.2.1 `void Server::mainLoop ()`

Main loop of server, does everything

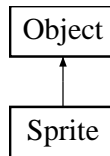
The documentation for this class was generated from the following files:

- `server.h`
- `server.cpp`

3.32 Sprite Class Reference

```
#include <sprite.h>
```

Inheritance diagram for Sprite:



Public Member Functions

- [Sprite](#) ()
- [Sprite](#) (std::string fileName)
- [~Sprite](#) ()
- void [draw](#) (float x, float y, float w, float h, std::string param="")
- void [bind](#) ()

3.32.1 Detailed Description

Class for handle single drawable objects.

3.32.2 Constructor & Destructor Documentation

3.32.2.1 `Sprite::Sprite ()` `[inline]`

Default constructor

3.32.2.2 `Sprite::Sprite (std::string fileName)` `[inline]`

Constructor

Parameters

<i>fileName</i>	name of graphic file (.png)
-----------------	-----------------------------

3.32.2.3 `Sprite::~~Sprite ()`

Destructor, free memory

3.32.3 Member Function Documentation

3.32.3.1 `void Sprite::bind ()` `[inline]`, `[virtual]`

Bind graphic for drawing

Reimplemented from [Object](#).

3.32.3.2 `void Sprite::draw (float x, float y, float w, float h, std::string param = " ")` `[virtual]`

Draw sprite on screen

Parameters

<i>x</i>	coordinate
<i>y</i>	coordinate
<i>w</i>	width
<i>h</i>	height
<i>param</i>	full draw string

Reimplemented from [Object](#).

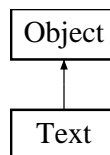
The documentation for this class was generated from the following files:

- view/sprite.h
- view/sprite.cpp

3.33 Text Class Reference

```
#include <text.h>
```

Inheritance diagram for Text:



Public Member Functions

- [Text](#) ()
- [Text](#) (std::string fileName, std::map< std::string, [Object](#) * > *igraphs)
- void [draw](#) (float x, float y, float w, float h, std::string param)

3.33.1 Detailed Description

Class for text handling using [Multi](#) class

3.33.2 Constructor & Destructor Documentation

3.33.2.1 Text::Text () [inline]

Default constructor

3.33.2.2 Text::Text (std::string *fileName*, std::map< std::string, [Object](#) * > * *igraphs*) [inline]

Constructor, load resources

Parameters

<i>fileName</i>	name of txh file
-----------------	------------------

<i>igraphs</i>	graphics from view class
----------------	--------------------------

3.33.3 Member Function Documentation

3.33.3.1 `void Text::draw (float x, float y, float w, float h, std::string param) [virtual]`

Draw text (font.txh::my_text)

Reimplemented from [Object](#).

The documentation for this class was generated from the following files:

- view/text.h
- view/text.cpp

3.34 Translate Class Reference

The documentation for this class was generated from the following file:

- translate.h

3.35 Utf8 Class Reference

```
#include <utf8.h>
```

Public Member Functions

- [Utf8](#) ()
- bool [notSpace](#) ()
- bool [notNull](#) ()
- bool [isBr](#) ()

Friends

- std::ostream & [operator<<](#) (std::ostream &output, const [Utf8](#) &u)
- std::istream & [operator>>](#) (std::istream &input, [Utf8](#) &u)
- bool [operator==](#) (const [Utf8](#) &u, const [Utf8](#) &v)
- bool [operator>](#) (const [Utf8](#) &u, const [Utf8](#) &v)
- bool [operator<](#) (const [Utf8](#) &u, const [Utf8](#) &v)

3.35.1 Detailed Description

Class for UTF-8 character handling

3.35.2 Constructor & Destructor Documentation

3.35.2.1 `Utf8::Utf8 () [inline]`

Constructor

3.35.3 Member Function Documentation

3.35.3.1 `bool Utf8::isBr () [inline]`

Character is not line-break

3.35.3.2 `bool Utf8::notNull () [inline]`

Character is not null

3.35.3.3 `bool Utf8::notSpace () [inline]`

Character is not space

3.35.4 Friends And Related Function Documentation

3.35.4.1 `bool operator< (const Utf8 & u, const Utf8 & v) [friend]`

Less than operator

3.35.4.2 `std::ostream& operator<< (std::ostream & output, const Utf8 & u) [friend]`

Output operator, don't write second byte if it is null

Returns

standard ostream

3.35.4.3 `bool operator== (const Utf8 & u, const Utf8 & v) [friend]`

Equality operator

3.35.4.4 `bool operator> (const Utf8 & u, const Utf8 & v) [friend]`

Greater than operator

3.35.4.5 `std::istream& operator>> (std::istream & input, Utf8 & u) [friend]`

[Input](#) operator, smartly determines UTF-8 characters and only read the second byte if it is valid UTF-8 character

Returns

standard istream

The documentation for this class was generated from the following file:

- `view/utf8.h`

3.36 View Class Reference

```
#include <view.h>
```

Public Member Functions

- [View](#) ()
- [~View](#) ()
- void [swap](#) ()
- bool [load](#) (std::string fileName)
- bool [loadPackage](#) (std::string fileName)
- void [draw](#) ([Rect](#) r, std::string res)
- void [draw](#) (float x, float y, float w, float h, std::string res)
- void [resize_event](#) (int &w, int &h)
- void [toggleFullScreen](#) ()
- void [drawAnim](#) ()
- TIME [getNow](#) ()
- void [setNow](#) ()
- int [getW](#) ()
- int [getH](#) ()
- void [setTickInterval](#) (int i)
- void [Screenshot](#) ()
- void [Toggle](#) ()
- void [frameBegin](#) ()
- void [frameEnd](#) ()
- TIME [frameDelay](#) ()

3.36.1 Detailed Description

Main class for handling graphics

3.36.2 Constructor & Destructor Documentation

3.36.2.1 [View::View](#) ()

Constructor, initialize screen

3.36.2.2 [View::~~View](#) ()

Destructor, unload objects

3.36.3 Member Function Documentation

3.36.3.1 void [View::draw](#) ([Rect](#) *r*, std::string *res*)

Draw on frame

Parameters

<i>r</i>	rectangle of drawing position
<i>res</i>	full drawing string

3.36.3.2 void [View::draw](#) (float *x*, float *y*, float *w*, float *h*, std::string *res*)

Draw on frame

Parameters

<i>x</i>	coordinate
<i>y</i>	coordinate
<i>w</i>	width
<i>h</i>	height
<i>res</i>	full drawing string

3.36.3.3 void View::drawAnim ()

Draw animations on frame, delete ended animations

3.36.3.4 void View::frameBegin ()

Begin of frame, actualilze time variables for current and previous frame

3.36.3.5 TIME View::frameDelay ()

Amount of sleep needed for enough frames per second

3.36.3.6 void View::frameEnd ()

End of frame, sleep for a while to save CPU time

3.36.3.7 int View::getH () [inline]

Get screen height

3.36.3.8 TIME View::getNow () [inline]

Get current frame time

3.36.3.9 int View::getW () [inline]

Get screen width

3.36.3.10 bool View::load (std::string *fileName*)

Load image, animation or text file

Parameters

<i>fileName</i>	name of file, extension determines the type
-----------------	---

Returns

success

3.36.3.11 bool View::loadPackage (std::string *fileName*)

Load resources from list

Parameters

<i>fileName</i>	name of package
-----------------	-----------------

Returns

success

3.36.3.12 void View::resize_event (int & *w*, int & *h*)

Resize callback function

Parameters

<i>w</i>	width
<i>h</i>	height

3.36.3.13 void View::Screenshot ()

Request to take screenshot at the end of frame

3.36.3.14 void View::setNow () [inline]

Set current frame time to actual time

3.36.3.15 void View::setTickInterval (int *i*) [inline]

Set refresh rate

3.36.3.16 void View::swap ()

Swap buffers, draw on screen

3.36.3.17 void View::Toggle () [inline]

Request to toggle fullscreen

3.36.3.18 void View::toggleFullScreen ()

Toggle fullscreen

The documentation for this class was generated from the following files:

- view/view.h
- view/view.cpp

Index

- ~GameTable
 - GameTable, [20](#)
- ~Input
 - Input, [25](#)
- ~Network
 - Network, [34](#)
- ~Object
 - Object, [36](#)
- ~Player
 - Player, [38](#)
- ~Server
 - Server, [49](#)
- ~Sprite
 - Sprite, [50](#)
- ~View
 - View, [54](#)
- alive
 - Monster, [31](#)
 - Player, [38](#)
- AnimFile, [6](#)
 - AnimFile, [6](#)
 - AnimFile, [6](#)
 - frame, [6](#)
- AnimObject, [6](#)
 - AnimObject, [7](#)
 - AnimObject, [7](#)
 - draw, [7](#)
- Animation, [5](#)
 - Animation, [5](#)
 - draw, [5](#)
 - getRect, [5](#)
- bind
 - GameTable, [20](#)
 - Object, [36](#)
 - Sprite, [50](#)
- Bomb, [7](#)
 - Bomb, [8](#)
 - boom, [8](#)
 - getOwner, [8](#)
 - getPoint, [8](#)
 - getSize, [9](#)
 - getX, [9](#)
 - getY, [9](#)
 - state, [9](#)
- boom
 - Bomb, [8](#)
- buildWalls
 - GameTable, [20](#)

- CFG_BOMBTIME
 - Config, [10](#)
- CFG_DEFAULT_SPEED
 - Config, [10](#)
- CFG_FAST_SPEED
 - Config, [10](#)
- CFG_FLAMETIME
 - Config, [10](#)
- CFG_PORT
 - Config, [10](#)
- CFG_RESPAWN_MONSTER
 - Config, [10](#)
- CFG_RESPAWN_PLAYER
 - Config, [10](#)
- CFG_SHOW_SCOREBOARD
 - Config, [10](#)
- CFG_SLOW_SPEED
 - Config, [10](#)
- CFG_VIRUSTIME
 - Config, [10](#)
- change
 - Monster, [31](#)
- check
 - Menu, [29](#)
- checkAfter
 - GameManager, [18](#)
- checkBefore
 - GameManager, [18](#)
- checkState
 - GameManager, [18](#)
- Client, [9](#)
- collide
 - Monster, [31](#)
- Config, [9](#)
 - CFG_BOMBTIME, [10](#)
 - CFG_DEFAULT_SPEED, [10](#)
 - CFG_FAST_SPEED, [10](#)
 - CFG_FLAMETIME, [10](#)
 - CFG_PORT, [10](#)
 - CFG_RESPAWN_MONSTER, [10](#)
 - CFG_RESPAWN_PLAYER, [10](#)
 - CFG_SHOW_SCOREBOARD, [10](#)
 - CFG_SLOW_SPEED, [10](#)
 - CFG_VIRUSTIME, [10](#)
- Control, [11](#)
 - Control, [11](#)
 - getDelay, [11](#)
 - getPut, [11](#)
 - getX, [11](#)

- getY, 11
 - px, 11
 - py, 11
 - setDelay, 13
 - setPut, 13
 - setX, 13
 - setY, 13
- count
 - Matrix, 28
- countMonster
 - GameTable, 20
- decBombs
 - Player, 38
- deckKill
 - Player, 38
- die
 - Monster, 31
 - Player, 38
- distance
 - Point, 45
- down
 - Menu, 29
- draw
 - Animation, 5
 - AnimObject, 7
 - Multi, 33
 - Object, 36
 - Sprite, 50
 - Text, 52
 - View, 54
- drawAnim
 - View, 55
- e
 - NetworkEvent, 35
- effect
 - Goods, 23
- error
 - Logger, 26
- execute
 - GameTable, 20
- expire
 - Flame, 15
- find
 - Matrix, 28
- Flame, 13
 - expire, 15
 - Flame, 13
 - getDirection, 15
 - getOwner, 15
 - getPoint, 15
 - getX, 15
 - getY, 15
 - operator==, 15
 - state, 15
- Frame, 16
 - Frame, 16
- obj, 16
 - time, 16
- frame
 - AnimFile, 6
- frameBegin
 - View, 55
- frameDelay
 - View, 55
- frameEnd
 - View, 55
- Game, 16
- GameConfig, 16
 - GameConfig, 17
 - GameConfig, 17
 - gen, 17
 - init, 17
 - nBomb, 17
 - nFlame, 17
 - nVirus, 17
 - sb, 17
 - sx, 17
 - sy, 17
- GameManager, 18
 - checkAfter, 18
 - checkBefore, 18
 - checkState, 18
 - GameManager, 18
 - GameManager, 18
 - getRound, 18
 - getRun, 18
 - newRound, 19
 - prepareTable, 19
 - waitRound, 19
- GameTable, 19
 - ~GameTable, 20
 - bind, 20
 - buildWalls, 20
 - countMonster, 20
 - execute, 20
 - GameTable, 19
 - GameTable, 19
 - generateBase, 20
 - generateMap, 20
 - getConfig, 20
 - getMovement, 20
 - getPlayers, 20
 - incScore, 20
 - loadMap, 21
 - move, 21
 - setConfig, 21
 - setPlayers, 21
 - setStart, 21
 - spawnMonster, 21
 - spawnPlayer, 22
 - spawnWalls, 22
- gen
 - GameConfig, 17
- generateBase

- GameTable, [20](#)
- generateMap
 - GameTable, [20](#)
- getBombX
 - Player, [38](#)
- getBombY
 - Player, [38](#)
- getBombs
 - Player, [38](#)
- getConfig
 - GameTable, [20](#)
- getControl
 - Player, [38](#)
- getDelay
 - Control, [11](#)
- getDie
 - Player, [38](#)
- getDirection
 - Flame, [15](#)
- getH
 - View, [55](#)
- getId
 - Player, [39](#)
- getInternetList
 - Network, [34](#)
- getKill
 - Player, [39](#)
- getKilled
 - Player, [39](#)
- getLife
 - Player, [39](#)
- getM
 - Matrix, [28](#)
- getMode
 - Menu, [29](#)
- getMovement
 - GameTable, [20](#)
- getN
 - Matrix, [28](#)
 - Select, [48](#)
- getNJoy
 - Input, [25](#)
- getNPlayers
 - Input, [25](#)
- getNow
 - View, [55](#)
- getOwner
 - Bomb, [8](#)
 - Flame, [15](#)
- getPlayers
 - GameTable, [20](#)
- getPoint
 - Bomb, [8](#)
 - Flame, [15](#)
 - Goods, [23](#)
 - Monster, [32](#)
 - Player, [39](#)
- getPointInt
 - Monster, [32](#)
- getPrevX
 - Player, [39](#)
- getPrevY
 - Player, [39](#)
- getPut
 - Control, [11](#)
 - Player, [39](#)
- getRect
 - Animation, [5](#)
- getRound
 - GameManager, [18](#)
- getRun
 - GameManager, [18](#)
 - Input, [25](#)
 - Select, [48](#)
- getRunSoft
 - Input, [25](#)
- getScore
 - Player, [39](#)
- getShowKills
 - Player, [39](#)
- getSize
 - Bomb, [9](#)
 - Player, [39](#)
- getSkin
 - Player, [39](#)
- getType
 - Goods, [23](#)
- getW
 - View, [55](#)
- getWall
 - Monster, [32](#)
- getX
 - Bomb, [9](#)
 - Control, [11](#)
 - Flame, [15](#)
 - Goods, [23](#)
 - Monster, [32](#)
 - Player, [40](#)
 - Point, [43](#)
- getY
 - Bomb, [9](#)
 - Control, [11](#)
 - Flame, [15](#)
 - Goods, [23](#)
 - Monster, [32](#)
 - Player, [40](#)
 - Point, [43](#)
- Goods, [22](#)
 - effect, [23](#)
 - getPoint, [23](#)
 - getType, [23](#)
 - getX, [23](#)
 - getY, [23](#)
 - Goods, [22](#)
 - state, [23](#)
- Ground, [23](#)

- Ground, 24
- operator<<, 24
- operator>>, 24
- operator=, 24
- operator==, 24
- state, 24
- h
 - Rect, 46
- id
 - NetworkEvent, 35
- incBombs
 - Player, 40
- incKill
 - Player, 40
- incKilled
 - Player, 40
- incScore
 - GameTable, 20
 - Player, 40
- incSize
 - Player, 40
- infect
 - Player, 40
- info
 - Logger, 26
- init
 - GameConfig, 17
 - Player, 40
- Input, 25
 - ~Input, 25
 - getNJoy, 25
 - getNPlayers, 25
 - getRun, 25
 - getRunSoft, 25
 - Input, 25
 - operator(), 25
 - poll, 25
 - setDelay, 26
 - setMode, 26
 - setRun, 26
 - setView, 26
 - updateJoySticks, 26
- isActive
 - Monster, 32
- isBr
 - Utf8, 53
- load
 - View, 55
- loadMap
 - GameTable, 21
- loadPackage
 - View, 55
- Logger, 26
 - error, 26
 - info, 26
 - setLevel, 27
- mainLoop
 - Menu, 29
 - Network, 34
 - ScoreBoard, 48
 - Select, 49
 - Server, 49
- Matrix
 - count, 28
 - find, 28
 - getM, 28
 - getN, 28
 - Matrix, 27
 - operator<<, 29
 - operator>>, 29
 - operator(), 28
 - setSize, 28
 - valid, 28
- Matrix< T >, 27
- Menu, 29
 - check, 29
 - down, 29
 - getMode, 29
 - mainLoop, 29
 - Menu, 29
 - up, 30
- MenuItem, 30
 - MenuItem, 30
 - MenuItem, 30
 - mode, 30
 - next, 30
 - txt, 30
- mode
 - MenuItem, 30
- Monster, 31
 - alive, 31
 - change, 31
 - collide, 31
 - die, 31
 - getPoint, 32
 - getPointInt, 32
 - getWall, 32
 - getX, 32
 - getY, 32
 - isActive, 32
 - Monster, 31
 - move, 32
 - setDX, 32
 - setDY, 32
 - setX, 32
 - setY, 32
 - state, 33
- move
 - GameTable, 21
 - Monster, 32
 - Player, 41
- Multi, 33
 - draw, 33
 - Multi, 33

- nBomb
 - GameConfig, 17
- nFlame
 - GameConfig, 17
- nVirus
 - GameConfig, 17
- Network, 34
 - ~Network, 34
 - getInternetList, 34
 - mainLoop, 34
 - Network, 34
 - startListen, 34
 - stopListen, 34
- NetworkEvent, 35
 - e, 35
 - id, 35
 - x, 35
 - y, 35
- newRound
 - GameManager, 19
- next
 - MenuItem, 30
- notNull
 - Utf8, 53
- notSpace
 - Utf8, 53
- novirus
 - Player, 41
- obj
 - Frame, 16
- Object, 35
 - ~Object, 36
 - bind, 36
 - draw, 36
- operator<
 - Utf8, 53
- operator<<
 - Ground, 24
 - Matrix, 29
 - Utf8, 53
- operator>
 - Utf8, 53
- operator>>
 - Ground, 24
 - Matrix, 29
 - Utf8, 53
- operator()
 - Input, 25
 - Matrix, 28
- operator=
 - Ground, 24
 - Point, 43
- operator==
 - Flame, 15
 - Ground, 24
 - Point, 45
 - Utf8, 53
- Player, 36
 - ~Player, 38
 - alive, 38
 - decBombs, 38
 - decKill, 38
 - die, 38
 - getBombX, 38
 - getBombY, 38
 - getBombs, 38
 - getControl, 38
 - getDie, 38
 - getId, 39
 - getKill, 39
 - getKilled, 39
 - getLife, 39
 - getPoint, 39
 - getPrevX, 39
 - getPrevY, 39
 - getPut, 39
 - getScore, 39
 - getShowKills, 39
 - getSize, 39
 - getSkin, 39
 - getX, 40
 - getY, 40
 - incBombs, 40
 - incKill, 40
 - incKilled, 40
 - incScore, 40
 - incSize, 40
 - infect, 40
 - init, 40
 - move, 41
 - novirus, 41
 - Player, 38
 - setBombX, 41
 - setBombY, 41
 - setControl, 41
 - setDie, 41
 - setId, 41
 - setKill, 41
 - setKilled, 41
 - setLife, 42
 - setPut, 42
 - setScore, 42
 - setShowKills, 42
 - setSkin, 42
 - setStart, 42
 - setX, 42
 - setY, 42
 - state, 42
 - win, 42
- Point
 - distance, 45
 - getX, 43
 - getY, 43
 - operator=, 43
 - operator==, 45

- Point, 43
- setX, 45
- setY, 45
- Way, 45
- Point< T >, 43
- poll
 - Input, 25
- prepareTable
 - GameManager, 19
- px
 - Control, 11
- py
 - Control, 11
- Rect, 45
 - h, 46
 - Rect, 46
 - w, 46
 - x, 46
 - y, 46
- resize_event
 - View, 56
- sb
 - GameConfig, 17
- ScoreBoard, 46
 - mainLoop, 48
 - ScoreBoard, 47
 - ScoreBoard, 47
- Screenshot
 - View, 56
- Select, 48
 - getN, 48
 - getRun, 48
 - mainLoop, 49
 - Select, 48
- Server, 49
 - ~Server, 49
 - mainLoop, 49
 - Server, 49
- setBombX
 - Player, 41
- setBombY
 - Player, 41
- setConfig
 - GameTable, 21
- setControl
 - Player, 41
- setDX
 - Monster, 32
- setDY
 - Monster, 32
- setDelay
 - Control, 13
 - Input, 26
- setDie
 - Player, 41
- setId
 - Player, 41
- setKill
 - Player, 41
- setKilled
 - Player, 41
- setLevel
 - Logger, 27
- setLife
 - Player, 42
- setMode
 - Input, 26
- setNow
 - View, 56
- setPlayers
 - GameTable, 21
- setPut
 - Control, 13
 - Player, 42
- setRun
 - Input, 26
- setScore
 - Player, 42
- setShowKills
 - Player, 42
- setSize
 - Matrix, 28
- setSkin
 - Player, 42
- setStart
 - GameTable, 21
 - Player, 42
- setTickInterval
 - View, 56
- setView
 - Input, 26
- setX
 - Control, 13
 - Monster, 32
 - Player, 42
 - Point, 45
- setY
 - Control, 13
 - Monster, 32
 - Player, 42
 - Point, 45
- spawnMonster
 - GameTable, 21
- spawnPlayer
 - GameTable, 22
- spawnWalls
 - GameTable, 22
- Sprite, 49
 - ~Sprite, 50
 - bind, 50
 - draw, 50
 - Sprite, 50
- startListen
 - Network, 34
- state

- Bomb, [9](#)
- Flame, [15](#)
- Goods, [23](#)
- Ground, [24](#)
- Monster, [33](#)
- Player, [42](#)
- stopListen
 - Network, [34](#)
- swap
 - View, [56](#)
- sx
 - GameConfig, [17](#)
- sy
 - GameConfig, [17](#)
- Text, [51](#)
 - draw, [52](#)
 - Text, [51](#)
- time
 - Frame, [16](#)
- Toggle
 - View, [56](#)
- toggleFullScreen
 - View, [56](#)
- Translate, [52](#)
- txt
 - MenuItem, [30](#)
- up
 - Menu, [30](#)
- updateJoySticks
 - Input, [26](#)
- Utf8, [52](#)
 - isBr, [53](#)
 - notNull, [53](#)
 - notSpace, [53](#)
 - operator<, [53](#)
 - operator<<, [53](#)
 - operator>, [53](#)
 - operator>>, [53](#)
 - operator==, [53](#)
 - Utf8, [52](#)
- valid
 - Matrix, [28](#)
- View, [53](#)
 - ~View, [54](#)
 - draw, [54](#)
 - drawAnim, [55](#)
 - frameBegin, [55](#)
 - frameDelay, [55](#)
 - frameEnd, [55](#)
 - getH, [55](#)
 - getNow, [55](#)
 - getW, [55](#)
 - load, [55](#)
 - loadPackage, [55](#)
 - resize_event, [56](#)
 - Screenshot, [56](#)
 - setNow, [56](#)
 - setTickInterval, [56](#)
 - swap, [56](#)
 - Toggle, [56](#)
 - toggleFullScreen, [56](#)
 - View, [54](#)
- w
 - Rect, [46](#)
- waitRound
 - GameManager, [19](#)
- Way
 - Point, [45](#)
- win
 - Player, [42](#)
- x
 - NetworkEvent, [35](#)
 - Rect, [46](#)
- y
 - NetworkEvent, [35](#)
 - Rect, [46](#)