

DELTAFORCE
CPSC2720PROJECT

Generated by Doxygen 1.6.1

Tue Apr 19 00:40:37 2016

Contents

1	Bug List	1
2	Class Index	3
2.1	Class Hierarchy	3
3	Class Index	5
3.1	Class List	5
4	File Index	7
4.1	File List	7
5	Class Documentation	9
5.1	Allegro Class Reference	9
5.1.1	Constructor & Destructor Documentation	10
5.1.1.1	Allegro	10
5.1.1.2	~Allegro	10
5.1.2	Member Function Documentation	10
5.1.2.1	collision	10
5.1.2.2	collision1	11
5.1.2.3	createWindow	11
5.1.2.4	gameLoop	11
5.1.2.5	init	12
5.1.3	Member Data Documentation	12
5.1.3.1	BAbitmap	12
5.1.3.2	boss	12
5.1.3.3	display	12
5.1.3.4	enemy	12
5.1.3.5	event_queue	12
5.1.3.6	EXbitmap	12
5.1.3.7	font	12

5.1.3.8	keyboard	12
5.1.3.9	LObitmap	12
5.1.3.10	looping	12
5.1.3.11	player	12
5.1.3.12	redraw	13
5.1.3.13	timer	13
5.2	Background Struct Reference	14
5.2.1	Member Data Documentation	14
5.2.1.1	dirX	14
5.2.1.2	dirY	14
5.2.1.3	height	14
5.2.1.4	image	14
5.2.1.5	velX	14
5.2.1.6	velY	14
5.2.1.7	width	14
5.2.1.8	x	15
5.2.1.9	y	15
5.3	bullet Class Reference	16
5.3.1	Constructor & Destructor Documentation	16
5.3.1.1	bullet	16
5.3.1.2	~bullet	17
5.3.2	Member Function Documentation	17
5.3.2.1	addBullet	17
5.3.2.2	drawB	17
5.3.2.3	getBulletMap	17
5.3.2.4	setBullet	18
5.3.3	Member Data Documentation	18
5.3.3.1	Bbitmap	18
5.3.3.2	Blist	18
5.4	bulletType Class Reference	19
5.4.1	Constructor & Destructor Documentation	19
5.4.1.1	bulletType	19
5.4.2	Member Function Documentation	19
5.4.2.1	getbX	19
5.4.2.2	getbY	20
5.4.2.3	setbX	20

5.4.2.4	setbY	20
5.4.3	Member Data Documentation	20
5.4.3.1	bX	20
5.4.3.2	bY	20
5.5	Enemy Class Reference	21
5.5.1	Constructor & Destructor Documentation	21
5.5.1.1	Enemy	21
5.5.1.2	~Enemy	21
5.5.2	Member Function Documentation	22
5.5.2.1	moveboss	22
5.5.2.2	moveEnemy	22
5.5.3	Member Data Documentation	22
5.5.3.1	health	22
5.6	EnemyOne Class Reference	23
5.6.1	Constructor & Destructor Documentation	23
5.6.1.1	EnemyOne	23
5.6.1.2	~EnemyOne	24
5.6.2	Member Function Documentation	24
5.6.2.1	addEnemy	24
5.6.2.2	drawE	24
5.6.2.3	getEnemy	24
5.6.2.4	setEnemy	25
5.6.3	Member Data Documentation	25
5.6.3.1	Ebitmap	25
5.6.3.2	Ebitmap2	25
5.6.3.3	Elist	25
5.7	Enemytype Class Reference	26
5.7.1	Constructor & Destructor Documentation	26
5.7.1.1	Enemytype	26
5.7.2	Member Function Documentation	27
5.7.2.1	getMS	27
5.7.2.2	getX	27
5.7.2.3	getY	27
5.7.2.4	setX	27
5.7.2.5	setY	28
5.7.3	Member Data Documentation	28

5.7.3.1	moveSpeed	28
5.7.3.2	x	28
5.7.3.3	y	28
5.8	Keyboard Class Reference	29
5.8.1	Constructor & Destructor Documentation	29
5.8.1.1	Keyboard	29
5.8.1.2	~Keyboard	29
5.8.2	Member Data Documentation	29
5.8.2.1	key	29
5.9	Plane Class Reference	30
5.9.1	Constructor & Destructor Documentation	30
5.9.1.1	Plane	30
5.9.1.2	~Plane	31
5.9.2	Member Function Documentation	31
5.9.2.1	draw	31
5.9.2.2	getBitmap	31
5.9.2.3	getX	31
5.9.2.4	getY	32
5.9.2.5	initialPlane	32
5.9.2.6	setBitmap	32
5.9.3	Member Data Documentation	32
5.9.3.1	bitmap	32
5.9.3.2	x	32
5.9.3.3	y	32
5.10	Player Class Reference	33
5.10.1	Constructor & Destructor Documentation	33
5.10.1.1	Player	33
5.10.1.2	~Player	34
5.10.2	Member Function Documentation	34
5.10.2.1	doLogic	34
5.10.2.2	moveBullet	34
5.10.3	Member Data Documentation	34
5.10.3.1	bMoveSpeed	34
5.10.3.2	health	34
5.10.3.3	moveSpeed	34
6	File Documentation	35

6.1	Allegro.cc File Reference	35
6.1.1	Detailed Description	36
6.1.2	Function Documentation	36
6.1.2.1	DrawBackground	36
6.1.2.2	InitBackground	36
6.1.2.3	UpdateBackground	37
6.1.3	Variable Documentation	37
6.1.3.1	BG	37
6.1.3.2	bgImage	37
6.1.3.3	boom	37
6.1.3.4	FG	37
6.1.3.5	fgImage	37
6.1.3.6	HEIGHT	37
6.1.3.7	MG	37
6.1.3.8	mgImage	37
6.1.3.9	shot	37
6.1.3.10	song	37
6.1.3.11	songInstance	37
6.1.3.12	WIDTH	37
6.2	Allegro.h File Reference	38
6.2.1	Detailed Description	38
6.2.2	Enumeration Type Documentation	38
6.2.2.1	States	38
6.3	bullet.cc File Reference	39
6.3.1	Detailed Description	39
6.4	bullet.h File Reference	40
6.4.1	Detailed Description	40
6.5	Enemy.cc File Reference	41
6.5.1	Detailed Description	41
6.6	Enemy.h File Reference	42
6.6.1	Detailed Description	42
6.7	EnemyOne.cc File Reference	43
6.7.1	Detailed Description	43
6.8	EnemyOne.h File Reference	44
6.8.1	Detailed Description	44
6.9	Keyboard.cc File Reference	45

6.9.1 Detailed Description	45
6.10 Keyboard.h File Reference	46
6.10.1 Detailed Description	46
6.10.2 Enumeration Type Documentation	46
6.10.2.1 keys	46
6.11 main.cc File Reference	47
6.11.1 Detailed Description	47
6.11.2 Function Documentation	47
6.11.2.1 main	47
6.12 Plane.cc File Reference	48
6.12.1 Detailed Description	48
6.13 Plane.h File Reference	49
6.13.1 Detailed Description	49
6.14 Player.cc File Reference	50
6.14.1 Detailed Description	50
6.15 Player.h File Reference	51
6.15.1 Detailed Description	51

Chapter 1

Bug List

File [Allegro.cc](#) No known bugs.

File [Allegro.h](#) No known bugs.

File [bullet.cc](#) No known bugs.

File [bullet.h](#) No known bugs.

File [Enemy.cc](#) No known bugs.

File [Enemy.h](#) No known bugs.

File [EnemyOne.cc](#) No known bugs.

File [EnemyOne.h](#) No known bugs.

File [Keyboard.cc](#) No known bugs.

File [Keyboard.h](#) No known bugs.

File [main.cc](#) No known bugs.

File [Plane.cc](#) No known bugs.

File [Plane.h](#) No known bugs.

File [Player.cc](#) No known bugs.

File [Player.h](#) No known bugs.

Chapter 2

Class Index

2.1 Class Hierarchy

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

Allegro	9
Background	14
bulletType	19
EnemyOne	23
Enemy	21
Enemytype	26
Keyboard	29
Plane	30
bullet	16
Player	33

Chapter 3

Class Index

3.1 Class List

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

Allegro	9
Background	14
bullet	16
bulletType	19
Enemy	21
EnemyOne	23
Enemytype	26
Keyboard	29
Plane	30
Player	33

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Allegro.cc (Implementation of the Allegro class)	35
Allegro.h (Definition of the Allegro class)	38
bullet.cc (Implementation of the bullet class)	39
bullet.h (Definition of the bulleyType and bullet class (which inherits from the Plane class). This contains the public and private member variables and functions of the bullet class)	40
Enemy.cc (Implementation of the Enemy class)	41
Enemy.h (Definition of the Enemy class)	42
EnemyOne.cc (Implementation of the Enemytype and EnemyOne classes)	43
EnemyOne.h (Definition of the Enemytype and EnemyOne classes)	44
Keyboard.cc (Implementation of the Keyboard class)	45
Keyboard.h (Definition of the Keyboard Class)	46
main.cc (This is the main function that calls the functions and initiates the gameplay)	47
Plane.cc (Implementation of the Plane class)	48
Plane.h (Definition of the Plane class)	49
Player.cc (Implementation of the Player class)	50
Player.h (Definition of the Player class which inherits from the bullet class)	51

Chapter 5

Class Documentation

5.1 Allegro Class Reference

```
#include <Allegro.h>
```

Public Member Functions

- [Allegro \(\)](#)
Constructor : Creates the various object and object states that will be used in this game.
- [~Allegro \(\)](#)
Destructor : Destroys the various objects used in the game.
- [int init \(\)](#)
Initial function.
- [int createWindow \(float FPS, int w, int h\)](#)
Creates the display window using the width and height and initializes all the devices, objects, images, keyboard, audio, background and timer that are used in this game.
- [void gameLoop \(\)](#)
This function defines the game itself. It loads the start image, the losing image, the game sounds, and the conditions of play, losing and collision.
- [bool collision \(Enemy *, int, int, int, int, int, int\)](#)
Defines the collision between the enemy and the player.
- [void collision1 \(Enemy *aa, Player *bb, int a, int b, int c, int d\)](#)
Defines the collision between the [bullet](#) and the enemy.

Private Attributes

- ALLEGRO_DISPLAY * [display](#)
- ALLEGRO_TIMER * [timer](#)

- ALLEGRO_EVENT_QUEUE * [event_queue](#)
- ALLEGRO_FONT * [font](#)
- [Keyboard](#) [keyboard](#)
- [Player](#) * [player](#)
- [Enemy](#) * [enemy](#)
- ALLEGRO_BITMAP * [BBitmap](#)
- ALLEGRO_BITMAP * [LObitmap](#)
- ALLEGRO_BITMAP * [EXbitmap](#)
- [Enemy](#) * [boss](#)
- bool [looping](#)
- bool [redraw](#)

5.1.1 Constructor & Destructor Documentation

5.1.1.1 Allegro::Allegro ()

Constructor : Creates the various object and object states that will be used in this game.

Parameters:

No parameters

Returns:

No return value

5.1.1.2 Allegro::~~Allegro ()

Destructor : Destroys the various objects used in the game.

Parameters:

No parameters

Returns:

No return value

5.1.2 Member Function Documentation

5.1.2.1 bool Allegro::collision (Enemy * aa, int x1, int y1, int a, int b, int c, int d)

Defines the collision between the enemy and the player.

Parameters:

*Enemy** : takes in the enemy object

int : x coordinate

int : y coordinate

int : value used to calculate the boundary between the enemy and the player

int : value used to calculate the boundary between the enemy and the player

int : value used to calculate the boundary between the enemy and the player

int : value used to calculate the boundary between the enemy and the player

Returns:

boolean value: returns true if there is a collision between the player and the enemy and false if there isn't

5.1.2.2 Allegro::collision1 (Enemy * aa, Player * bb, int a, int b, int c, int d)

Defines the collision between the [bullet](#) and the enemy.

Parameters:

*Enemy** : takes in the enemy object

*Player** : takes in a player object(which inherits from a bullet object)

int : value used to calculate the boundary between the enemy and the [bullet](#)

int : value used to calculate the boundary between the enemy and the [bullet](#)

int : value used to calculate the boundary between the enemy and the [bullet](#)

int : value used to calculate the boundary between the enemy and the [bullet](#)

Returns:

No return value

5.1.2.3 int Allegro::createWindow (float FPS, int w, int h)

Creates the display window using the width and height and initializes all the devices, objects, images, keyboard, audio, background and timer that are used in this game.

Parameters:

w : the width of the display

h : the height of the display

FPS

Returns:

int : returns 0 if the display is open, returns -1 if the display stops

5.1.2.4 void Allegro::gameLoop ()

This function defines the game itself. It loads the start image, the losing image, the game sounds, and the conditions of play, losing and collision.

Parameters:

No parameters

Returns:

No return value

5.1.2.5 `int Allegro::init ()`

Initial function.

Parameters:

No parameters

Returns:

`int` : 0 or -1

5.1.3 Member Data Documentation

5.1.3.1 `ALLEGRO_BITMAP* Allegro::BAbitmap [private]`

Image of the start page of the game

5.1.3.2 `Enemy* Allegro::boss [private]`

The larger enemy

5.1.3.3 `ALLEGRO_DISPLAY* Allegro::display [private]`

display window

5.1.3.4 `Enemy* Allegro::enemy [private]`

The smaller enemy

5.1.3.5 `ALLEGRO_EVENT_QUEUE* Allegro::event_queue [private]`

5.1.3.6 `ALLEGRO_BITMAP* Allegro::EXbitmap [private]`

Image of an explosion

5.1.3.7 `ALLEGRO_FONT* Allegro::font [private]`

5.1.3.8 `Keyboard Allegro::keyboard [private]`

5.1.3.9 `ALLEGRO_BITMAP* Allegro::LObitmap [private]`

Image of the page displayed when the user gets killed in the game

5.1.3.10 `bool Allegro::looping [private]`

5.1.3.11 `Player* Allegro::player [private]`

The player object

5.1.3.12 `bool Allegro::redraw [private]`

5.1.3.13 `ALLEGRO_TIMER* Allegro::timer [private]`

game timer

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

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

5.2 Background Struct Reference

```
#include <Allegro.h>
```

Public Attributes

- float [x](#)
- float [y](#)
- float [velX](#)
- float [velY](#)
- int [dirX](#)
- int [dirY](#)
- int [width](#)
- int [height](#)
- ALLEGRO_BITMAP * [image](#)

5.2.1 Member Data Documentation

5.2.1.1 int Background::dirX

Direction of the background on the x axis

5.2.1.2 int Background::dirY

Direction of the background on the y axis

5.2.1.3 int Background::height

height of the background image

5.2.1.4 ALLEGRO_BITMAP* Background::image

The background image

5.2.1.5 float Background::velX

Velocity of the background moving in the X direction

5.2.1.6 float Background::velY

Velocity of the background moving in the Y direction

5.2.1.7 int Background::width

width of the background image

5.2.1.8 float Background::x

x coordinate

5.2.1.9 float Background::y

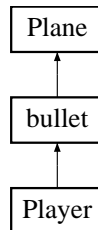
y coordinate

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

- [Allegro.h](#)

5.3 bullet Class Reference

`#include <bullet.h>`Inheritance diagram for `bullet::`



Public Member Functions

- `bullet ()`
Constructor.
- `~bullet ()`
Destructor.
- `void addBullet (int, int)`
Creates the `bullet` and sets it at the x and y coordinates given with the two int parameters.
- `void setBullet (std::string fileName)`
Loads the bitmap image of the `bullet`.
- `ALLEGRO_BITMAP * getBulletMap ()`
getter function to get the bitmat image of the `bullet`
- `void drawB ()`
Draw function that actually draws the `bullet` image.

Public Attributes

- `list< bulletType * > Blist`

Protected Attributes

- `ALLEGRO_BITMAP * Bbitmap`

5.3.1 Constructor & Destructor Documentation

5.3.1.1 `bullet::bullet ()`

Constructor.

Parameters:

No parameters

Returns:

No return value

5.3.1.2 bullet::~~bullet ()

Destructor.

Parameters:

No parameters

Returns:

No return value

5.3.2 Member Function Documentation**5.3.2.1 void bullet::addBullet (int *e*, int *f*)**

Creates the [bullet](#) and sets it at the x and y coordinates given with the two int parameters.

Parameters:

int : representing the x coordinate of the [bullet](#)

int : representing the y coordinate of the [bullet](#)

Returns:

No return value

5.3.2.2 bullet::drawB ()

Draw function that actually draws the [bullet](#) image.

Parameters:

No parameters

Returns:

No return value

5.3.2.3 ALLEGRO_BITMAP * bullet::getBulletMap ()

getter function to get the bitmat image of the [bullet](#)

Parameters:

No parameters

Returns:

Bitmap image of the [bullet](#)

5.3.2.4 `bullet::setBullet (std::string fileName)`

Loads the bitmap image of the [bullet](#).

Parameters:

fileName : the file name/path that contains the image of the [bullet](#)

Returns:

No return value

5.3.3 Member Data Documentation**5.3.3.1 `ALLEGRO_BITMAP* bullet::Bbitmap` `[protected]`**

Creates a Bitmap object that will hold the [bullet](#) image

5.3.3.2 `list<bulletType*> bullet::Blist`

Creates a list of the bullets

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

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

5.4 bulletType Class Reference

```
#include <bullet.h>
```

Public Member Functions

- `bulletType` (int a, int b)
Constructor.
- `int getbX ()`
Gets the x coordinate of the `bullet`.
- `void setbY` (int a)
Sets the Y coordinate of the `bullet` to the value of the int parameter.
- `void setbX` (int b)
Sets the X coordinate of the `bullet` to the value of the int parameter.
- `int getbY ()`
Gets the y coordinate of the `bullet`.

Private Attributes

- `int bX`
- `int bY`

5.4.1 Constructor & Destructor Documentation

5.4.1.1 `bulletType::bulletType (int a, int b) [inline]`

Constructor.

Parameters:

- a* X coordinate
- b* Y coordinate

Returns:

No return value

5.4.2 Member Function Documentation

5.4.2.1 `int bulletType::getbX () [inline]`

Gets the x coordinate of the `bullet`.

Parameters:

No parameters

Returns:

Integer value

5.4.2.2 int bulletType::getbY () [inline]

Gets the y coordinate of the [bullet](#).

Parameters:

No parameters

Returns:

Integer value

5.4.2.3 void bulletType::setbX (int *b*) [inline]

Sets the X coordinate of the [bullet](#) to the value of the int parameter.

Parameters:

b : int value of the new X coordinate of the [bullet](#)

Returns:

No return value

5.4.2.4 void bulletType::setbY (int *a*) [inline]

Sets the Y coordinate of the [bullet](#) to the value of the int parameter.

Parameters:

a : int value of the new Y coordinate of the [bullet](#)

Returns:

No return value

5.4.3 Member Data Documentation**5.4.3.1 int bulletType::bX [private]**

X coordinate of the [bullet](#)

5.4.3.2 int bulletType::bY [private]

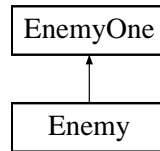
Y coordinate of the [bullet](#)

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

- [bullet.h](#)

5.5 Enemy Class Reference

#include <Enemy.h>Inheritance diagram for Enemy::



Public Member Functions

- `Enemy ()`
Constructor.
- `~Enemy ()`
Destructor.
- `void moveEnemy ()`
Defines the movement of the first enemy type to move from right to left by subtracting the 'moveSpeed' from the x coordinate. if enemy goes out of the screen on the left, it is reset to the right side.
- `void moveboss ()`
Defines the movement of the second enemy type to move from right to left by subtracting the 'moveSpeed' from the x coordinate.

Protected Attributes

- `int health`

5.5.1 Constructor & Destructor Documentation

5.5.1.1 Enemy::Enemy ()

Constructor.

Parameters:

No parameters

Returns:

No return value

5.5.1.2 Enemy::~~Enemy ()

Destructor.

Parameters:

No parameters

Returns:

No return value

5.5.2 Member Function Documentation

5.5.2.1 void Enemy::moveboss ()

Defines the movement of the second enemy type to move from right to left by subtracting the 'moveSpeed' from the x coordinate.

Parameters:

No parameters

Returns:

No return value

5.5.2.2 void Enemy::moveEnemy ()

Defines the movement of the first enemy type to move from right to left by subtracting the 'moveSpeed' from the x coordinate. if enemy goes out of the screen on the left, it is reset to the right side.

Parameters:

No parameters

Returns:

No return value

5.5.3 Member Data Documentation

5.5.3.1 int Enemy::health [protected]

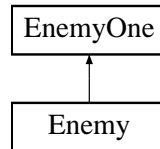
the health of the enemy which is set to 0 by the constructor

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

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

5.6 EnemyOne Class Reference

`#include <EnemyOne.h>`Inheritance diagram for EnemyOne::



Public Member Functions

- [EnemyOne](#) ()
Constructor : Sets the images for the two enemy types to NULL before they are actually initialized.
- virtual [~EnemyOne](#) ()
Destructor : Destroys any enemy type created dynamically.
- void [addEnemy](#) (int, int)
Creates an [Enemy](#) object dynamically at the coordinates given by b & s which represent the x and y coordinates respectively.
- void [setEnemy](#) (std::string name)
Loads the image name in the parameter list onto the Ebitmap object.
- ALLEGRO_BITMAP * [getEnemy](#) ()
Getter function to get the set [Enemy](#) image.
- void [drawE](#) ()
A draw function that draws the image loaded onto the enemy bitmap.

Public Attributes

- list< [Enemytype](#) * > [Elist](#)

Protected Attributes

- ALLEGRO_BITMAP * [Ebitmap](#)
- ALLEGRO_BITMAP * [Ebitmap2](#)

5.6.1 Constructor & Destructor Documentation

5.6.1.1 EnemyOne::EnemyOne ()

Constructor : Sets the images for the two enemy types to NULL before they are actually initialized.

Parameters:

No parameters

Returns:

No return value

5.6.1.2 EnemyOne::~~EnemyOne () [virtual]

Destructor : Destroys any enemy type created dynamically.

Parameters:

No parameters

Returns:

No return value

5.6.2 Member Function Documentation**5.6.2.1 void EnemyOne::addEnemy (int *b*, int *s*)**

Creates an [Enemy](#) object dynamically at the coordinates given by *b* & *s* which represent the x and y coordinates respectively.

Parameters:

b : x coordinate

s : y coordinate

Returns:

No return value

5.6.2.2 void EnemyOne::drawE ()

A draw function that draws the image loaded onto the enemy bitmap.

Parameters:

No parameters

Returns:

No return value

5.6.2.3 ALLEGRO_BITMAP * EnemyOne::getEnemy ()

Getter function to get the set [Enemy](#) image.

Parameters:

No parameters

Returns:

ALLEGRO_BITMAP type : the enemy image

5.6.2.4 void EnemyOne::setEnemy (std::string *name*)

Loads the image name in the parameter list onto the Ebitmap object.

Parameters:

name : the name of the image as stored in the folder

Returns:

No return value

5.6.3 Member Data Documentation**5.6.3.1 ALLEGRO_BITMAP* EnemyOne::Ebitmap [protected]**

The first enemy type

5.6.3.2 ALLEGRO_BITMAP* EnemyOne::Ebitmap2 [protected]

The second enemy type

5.6.3.3 list<Enemytype*> EnemyOne::Elist

A list that allows for the creation of multiple enemies dynamically

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

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

5.7 Enemytype Class Reference

```
#include <EnemyOne.h>
```

Public Member Functions

- `int getX ()`
Getter function for the X coordinate of the enemy.
- `int getY ()`
Getter function for the Y coordinate of the enemy.
- `void setX (int a)`
Setter function which sets the x coordinate of the enemy.
- `void setY (int b)`
Setter function which sets the Y coordinate of the enemy.
- `int getMS ()`
a getter function which gets the set speed of the enemy
- `Enemytype (int a, int s)`
Constructor : Creates the enemy to come from the right side of the screen(the x coordinate is automatically set to 640) and sets the speed at which they move.

Private Attributes

- `int x`
- `int y`
- `int moveSpeed`

5.7.1 Constructor & Destructor Documentation

5.7.1.1 Enemytype::Enemytype (int a, int s) [inline]

Constructor : Creates the enemy to come from the right side of the screen(the x coordinate is automatically set to 640) and sets the speed at which they move.

Parameters:

- a*,: the y coordinate of the enemy
s,: the speed at which the enemies move

Returns:

No return value

5.7.2 Member Function Documentation

5.7.2.1 `int Enemytype::getMS () [inline]`

a getter function which gets the set speed of the enemy

Parameters:

No parameters

Returns:

moveSpeed : the speed of the enemy

5.7.2.2 `int Enemytype::getX () [inline]`

Getter function for the X coordinate of the enemy.

Parameters:

No parameters

Returns:

No return value

5.7.2.3 `int Enemytype::getY () [inline]`

Getter function for the Y coordinate of the enemy.

Parameters:

No parameters

Returns:

int : the y coordinate of the enemy

5.7.2.4 `void Enemytype::setX (int a) [inline]`

Setter function which sets the x coordinate of the enemy.

Parameters:

a : the value which the X coordinate of the enemy will be set to

Returns:

No return value

5.7.2.5 void Enemytype::setY (int *b*) [inline]

Setter function which sets the Y coordinate of the enemy.

Parameters:

b : the value which the Y coordinate of the enemy will be set to

Returns:

No return value

5.7.3 Member Data Documentation

5.7.3.1 int Enemytype::moveSpeed [private]

the speed at which the enemy object is moving

5.7.3.2 int Enemytype::x [private]

the x coordinate of the enemy object

5.7.3.3 int Enemytype::y [private]

the y coordinate of the enemy object

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

- [EnemyOne.h](#)

5.8 Keyboard Class Reference

```
#include <Keyboard.h>
```

Public Member Functions

- [Keyboard \(\)](#)
- [~Keyboard \(\)](#)

Public Attributes

- bool [key](#) [6]

5.8.1 Constructor & Destructor Documentation

5.8.1.1 Keyboard::Keyboard ()

5.8.1.2 Keyboard::~~Keyboard ()

5.8.2 Member Data Documentation

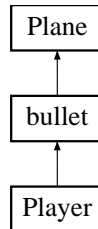
5.8.2.1 bool Keyboard::key[6]

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

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

5.9 Plane Class Reference

#include <Plane.h>Inheritance diagram for Plane::



Public Member Functions

- [Plane \(\)](#)
Constructor.
- virtual [~Plane \(\)](#)
Destructor.
- int [getX \(\)](#)
this function gets the X coordinate of the object
- int [getY \(\)](#)
this function gets the Y coordinate of the object
- void [setBitmap](#) (std::string filePath)
- ALLEGRO_BITMAP * [getBitmap](#) ()
This function is used to get the object loaded onto the bitmap object by the function 'setBitmap'.
- void [initialPlane](#) ()
This function set the starting point of the player by setting both the x and y coordinates to 10 and 240 respectively.
- void [draw](#) ()
this function draws the loaded bitmap to the screen on the point provided by the x and y coordinates provided

Protected Attributes

- ALLEGRO_BITMAP * [bitmap](#)
- int [x](#)
- int [y](#)

5.9.1 Constructor & Destructor Documentation

5.9.1.1 Plane::Plane ()

Constructor.

Parameters:

This function doesn't take any parameters

Returns:

No return value

5.9.1.2 Plane::~~Plane () [virtual]

Destructor.

Parameters:

this function doesn't take any parameters

Returns:

no return value

5.9.2 Member Function Documentation**5.9.2.1 void Plane::draw ()**

this function draws the loaded bitmap to the screen on the point provided by the x and y coordinates provided

Parameters:

takes no parameters

Returns:

No return value

5.9.2.2 ALLEGRO_BITMAP * Plane::getBitmap ()

This function is used to get the object loaded onto the bitmap object by the function 'setBitmap'.

Parameters:

This function takes in no parameters

Returns:

This function returns a bitmap object

5.9.2.3 int Plane::getX ()

this function gets the X coordinate of the object

Parameters:

this function takes no parameters

Returns:

this function returns the X coordinate of the object

5.9.2.4 int Plane::getY ()

this function gets the Y coordinate of the object

Parameters:

this function takes no parameters

Returns:

this function returns the Y coordinate of the object

5.9.2.5 void Plane::initialPlane ()

This function set the starting point of the player by setting both the x and y coordinates to 10 and 240 respectively.

Parameters:

This function takes in no parameters

Returns:

No return value

5.9.2.6 void Plane::setBitmap (std::string filePath)**5.9.3 Member Data Documentation****5.9.3.1 ALLEGRO_BITMAP* Plane::bitmap [protected]**

The picture of the player

5.9.3.2 int Plane::x [protected]

x coordinate

5.9.3.3 int Plane::y [protected]

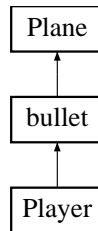
y coordinate

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

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

5.10 Player Class Reference

`#include <Player.h>`Inheritance diagram for Player::



Public Member Functions

- [Player \(\)](#)
Constructor.
- [~Player \(\)](#)
Destructor.
- void [doLogic \(Keyboard keyboard\)](#)
Defines the movement key controls for the player object so it can be moved using a keyboard.
- void [moveBullet \(\)](#)
This function initializes the movement of the [bullet](#) through the display.

Private Attributes

- int [health](#)
- int [moveSpeed](#)
- int [bMoveSpeed](#)

5.10.1 Constructor & Destructor Documentation

5.10.1.1 Player::Player ()

Constructor.

Parameters:

No parameters

Returns:

No return value

5.10.1.2 `Player::~~Player ()`

Destructor.

Parameters:

No parameters

Returns:

No return value

5.10.2 Member Function Documentation

5.10.2.1 `Void Player::doLogic (Keyboard keyboard)`

Defines the movement key controls for the player object so it can be moved using a keyboard.

Parameters:

This function takes in a [Keyboard](#) object

Returns:

No return value

5.10.2.2 `void Player::moveBullet ()`

This function initializes the movement of the [bullet](#) through the display.

Parameters:

This function takes no parameters

Returns:

No return value

5.10.3 Member Data Documentation

5.10.3.1 `int Player::bMoveSpeed [private]`

speed of the [bullet](#)

5.10.3.2 `int Player::health [private]`

5.10.3.3 `int Player::moveSpeed [private]`

Rate at which the player can be moved

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

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

Chapter 6

File Documentation

6.1 Allegro.cc File Reference

Implementation of the [Allegro](#) class. `#include "Allegro.h"`
`#include <stdio.h>`

Functions

- void [InitBackground](#) ([Background](#) &back, float x, float y, float velx, float vely, int width, int height, int dirX, int dirY, ALLEGRO_BITMAP *image)

This function initializes the various background images : acts like a constructor for the background struct.

- void [UpdateBackground](#) ([Background](#) &back)

Initializes the speed at which the background moves.

- void [DrawBackground](#) ([Background](#) &back)

Draws the background and makes sure it is always cycles through.

Variables

- const int [WIDTH](#) = 640
- const int [HEIGHT](#) = 480
- [Background](#) BG
- [Background](#) MG
- [Background](#) FG
- ALLEGRO_BITMAP * [bgImage](#) = NULL
- ALLEGRO_BITMAP * [mgImage](#) = NULL
- ALLEGRO_BITMAP * [fgImage](#) = NULL
- ALLEGRO_SAMPLE * [shot](#) = NULL
- ALLEGRO_SAMPLE * [boom](#) = NULL
- ALLEGRO_SAMPLE * [song](#) = NULL
- ALLEGRO_SAMPLE_INSTANCE * [songInstance](#) = NULL

6.1.1 Detailed Description

Implementation of the [Allegro](#) class. This contains the implementation of member variables and functions of the [Allegro](#) class

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi.

Bug

No known bugs.

6.1.2 Function Documentation

6.1.2.1 void DrawBackground (Background & *back*)

Draws the background and makes sure it is always cycles through.

Parameters:

&back : [Background](#) object with the properties of the background struct

Returns:

No return value

6.1.2.2 void InitBackground (Background & *back*, float *x*, float *y*, float *velx*, float *vely*, int *width*, int *height*, int *dirX*, int *dirY*, ALLEGRO_BITMAP * *image*)

This function initializes the various background images : acts like a constructor for the background struct.

Parameters:

&back : a background object with the properties of the background struct

x : x coordinate

y : y coordinate

velx : velocity of the background moving in the x direction

vely : velocity of the background moving in the y direction

width : width of the background

height : height of the background

dirX : direction of x

dirY : direction of y

ALLEGRO_BITMAP : the background image

Returns:

No return value

6.1.2.3 void UpdateBackground (Background & *back*)

Initializes the speed at which the background moves.

Parameters:

&back : [Background](#) object with the properties of the background struct

Returns:

No return value

6.1.3 Variable Documentation

6.1.3.1 Background BG

6.1.3.2 ALLEGRO_BITMAP* bgImage = NULL

6.1.3.3 ALLEGRO_SAMPLE* boom = NULL

6.1.3.4 Background FG

6.1.3.5 ALLEGRO_BITMAP* fgImage = NULL

6.1.3.6 const int HEIGHT = 480

6.1.3.7 Background MG

6.1.3.8 ALLEGRO_BITMAP* mgImage = NULL

6.1.3.9 ALLEGRO_SAMPLE* shot = NULL

6.1.3.10 ALLEGRO_SAMPLE* song = NULL

6.1.3.11 ALLEGRO_SAMPLE_INSTANCE* songInstance = NULL

6.1.3.12 const int WIDTH = 640

6.2 Allegro.h File Reference

Definition of the [Allegro](#) class. `#include <allegro5/allegro.h>`

```
#include <allegro5/allegro_image.h>
#include <allegro5/allegro_primitives.h>
#include <allegro5/allegro_font.h>
#include <allegro5/allegro_ttf.h>
#include <allegro5/allegro_audio.h>
#include <allegro5/allegro_acodec.h>
#include "Player.h"
#include "Keyboard.h"
#include "Enemy.h"
```

Classes

- class [Allegro](#)
- struct [Background](#)

Enumerations

- enum [States](#) { [TITLE](#), [PLAY](#), [LOST](#) }

6.2.1 Detailed Description

Definition of the [Allegro](#) class. This contains the public and private member variables and functions of the [Allegro](#) class

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi.

Bug

No known bugs.

6.2.2 Enumeration Type Documentation

6.2.2.1 enum States

Enumerator:

TITLE

PLAY

LOST

6.3 bullet.cc File Reference

Implementation of the [bullet](#) class. `#include "bullet.h"`

6.3.1 Detailed Description

Implementation of the [bullet](#) class. This contains the implementation of member variables and functions of the [bullet](#) class

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi.

Bug

No known bugs.

6.4 bullet.h File Reference

Definition of the bulleyType and [bullet](#) class (which inherits from the [Plane](#) class). This contains the public and private member variables and functions of the [bullet](#) class. `#include <string>`

```
#include <allegro5/allegro.h>
#include <allegro5/allegro_image.h>
#include <allegro5/allegro_primitives.h>
#include "Plane.h"
#include <list>
```

Classes

- class [bulletType](#)
- class [bullet](#)

6.4.1 Detailed Description

Definition of the bulleyType and [bullet](#) class (which inherits from the [Plane](#) class). This contains the public and private member variables and functions of the [bullet](#) class.

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi.

Bug

No known bugs.

6.5 Enemy.cc File Reference

Implementation of the [Enemy](#) class. `#include "Enemy.h"`

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

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

```
#include <time.h>
```

6.5.1 Detailed Description

Implementation of the [Enemy](#) class. This contains the implementation of member variables and functions of the [Enemy](#) class

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi.

Bug

No known bugs.

6.6 Enemy.h File Reference

Definition of the [Enemy](#) class. `#include "EnemyOne.h"`

Classes

- class [Enemy](#)

6.6.1 Detailed Description

Definition of the [Enemy](#) class. This contains the public and private member variables and functions of the [Enemy](#) class

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi.

Bug

No known bugs.

6.7 EnemyOne.cc File Reference

Implementation of the [Enemytype](#) and [EnemyOne](#) classes. `#include "EnemyOne.h"`
`#include <allegro5/allegro_primitives.h>`

6.7.1 Detailed Description

Implementation of the [Enemytype](#) and [EnemyOne](#) classes. This contains the implementation of member variables and functions of the [Enemytype](#) and [EnemyOne](#) classes.

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi

Bug

No known bugs.

6.8 EnemyOne.h File Reference

Definition of the [Enemytype](#) and [EnemyOne](#) classes. `#include <string>`

```
#include <allegro5/allegro.h>
```

```
#include <allegro5/allegro_image.h>
```

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

```
#include <time.h>
```

```
#include <iostream>
```

```
#include <list>
```

Classes

- class [Enemytype](#)
- class [EnemyOne](#)

6.8.1 Detailed Description

Definition of the [Enemytype](#) and [EnemyOne](#) classes. This contains the public and private member variables and functions of the [Enemytype](#) and [EnemyOne](#) classes

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi

Bug

No known bugs.

6.9 Keyboard.cc File Reference

Implementation of the [Keyboard](#) class. `#include "Keyboard.h"`

6.9.1 Detailed Description

Implementation of the [Keyboard](#) class. This contains the implementation of member variables and functions of the [Keyboard](#) class

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi.

Bug

No known bugs.

6.10 Keyboard.h File Reference

Definition of the [Keyboard](#) Class.

Classes

- class [Keyboard](#)

Enumerations

- enum [keys](#) {
 [UP](#), [LEFT](#), [DOWN](#), [RIGHT](#),
 [SPACE](#), [ENTER](#) }

6.10.1 Detailed Description

Definition of the [Keyboard](#) Class. This contains the public and private member variables and functions of the [Keyboard](#) class

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi.

Bug

No known bugs.

6.10.2 Enumeration Type Documentation

6.10.2.1 enum keys

Enumerator:

UP
LEFT
DOWN
RIGHT
SPACE
ENTER

6.11 main.cc File Reference

This is the main function that calls the functions and initiates the gameplay. `#include "Allegro.h"`

Functions

- `int main ()`

6.11.1 Detailed Description

This is the main function that calls the functions and initiates the gameplay.

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi.

Bug

No known bugs.

6.11.2 Function Documentation

6.11.2.1 `int main ()`

6.12 Plane.cc File Reference

Implementation of the [Plane](#) class. `#include "Plane.h"`
`#include <allegro5/allegro_primitives.h>`

6.12.1 Detailed Description

Implementation of the [Plane](#) class. This contains the implementation of member variables and functions of the [Plane](#) class

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi.

Bug

No known bugs.

6.13 Plane.h File Reference

Definition of the [Plane](#) class. #include <string>
#include <allegro5/allegro.h>
#include <allegro5/allegro_image.h>

Classes

- class [Plane](#)

6.13.1 Detailed Description

Definition of the [Plane](#) class. This contains the public and private member variables and functions of the [Plane](#) class.

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi

Bug

No known bugs.

6.14 Player.cc File Reference

Implementation of the [Player](#) class. `#include "Player.h"`

6.14.1 Detailed Description

Implementation of the [Player](#) class. This contains the implementation of member variables and functions of the [Player](#) class

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi.

Bug

No known bugs.

6.15 Player.h File Reference

Definition of the [Player](#) class which inherits from the [bullet](#) class. `#include "Keyboard.h"`
`#include "bullet.h"`

Classes

- class [Player](#)

6.15.1 Detailed Description

Definition of the [Player](#) class which inherits from the [bullet](#) class. This contains the public and private member variables and functions of the [Player](#) class

Author:

Wang Kangning, Jefferson Sylva-Iriogbe and Yuhai Shi

Bug

No known bugs.

Index

- ~Allegro
 - Allegro, [10](#)
- ~Enemy
 - Enemy, [21](#)
- ~EnemyOne
 - EnemyOne, [24](#)
- ~Keyboard
 - Keyboard, [29](#)
- ~Plane
 - Plane, [31](#)
- ~Player
 - Player, [33](#)
- ~bullet
 - bullet, [17](#)
- addBullet
 - bullet, [17](#)
- addEnemy
 - EnemyOne, [24](#)
- Allegro, [9](#)
 - ~Allegro, [10](#)
 - Allegro, [10](#)
 - BABitmap, [12](#)
 - boss, [12](#)
 - collision, [10](#)
 - collision1, [11](#)
 - createWindow, [11](#)
 - display, [12](#)
 - enemy, [12](#)
 - event_queue, [12](#)
 - EXbitmap, [12](#)
 - font, [12](#)
 - gameLoop, [11](#)
 - init, [11](#)
 - keyboard, [12](#)
 - LObitmap, [12](#)
 - looping, [12](#)
 - player, [12](#)
 - redraw, [12](#)
 - timer, [13](#)
- Allegro.cc, [35](#)
 - BG, [37](#)
 - bgImage, [37](#)
 - boom, [37](#)
 - DrawBackground, [36](#)
 - FG, [37](#)
 - fgImage, [37](#)
 - HEIGHT, [37](#)
 - InitBackground, [36](#)
 - MG, [37](#)
 - mgImage, [37](#)
 - shot, [37](#)
 - song, [37](#)
 - songInstance, [37](#)
 - UpdateBackground, [36](#)
 - WIDTH, [37](#)
- Allegro.h, [38](#)
 - LOST, [38](#)
 - PLAY, [38](#)
 - States, [38](#)
 - TITLE, [38](#)
- BABitmap
 - Allegro, [12](#)
- Background, [14](#)
 - dirX, [14](#)
 - dirY, [14](#)
 - height, [14](#)
 - image, [14](#)
 - velX, [14](#)
 - velY, [14](#)
 - width, [14](#)
 - x, [14](#)
 - y, [15](#)
- Bbitmap
 - bullet, [18](#)
- BG
 - Allegro.cc, [37](#)
- bgImage
 - Allegro.cc, [37](#)
- bitmap
 - Plane, [32](#)
- Blist
 - bullet, [18](#)
- bMoveSpeed
 - Player, [34](#)
- boom
 - Allegro.cc, [37](#)
- boss
 - Allegro, [12](#)

- bullet, 16
 - ~bullet, 17
 - addBullet, 17
 - Bbitmap, 18
 - Blist, 18
 - bullet, 16
 - drawB, 17
 - getBulletMap, 17
 - setBullet, 18
- bullet.cc, 39
- bullet.h, 40
- bulletType, 19
 - bulletType, 19
 - bX, 20
 - bY, 20
 - getbX, 19
 - getbY, 20
 - setbX, 20
 - setbY, 20
- bX
 - bulletType, 20
- bY
 - bulletType, 20
- collision
 - Allegro, 10
- collision1
 - Allegro, 11
- createWindow
 - Allegro, 11
- dirX
 - Background, 14
- dirY
 - Background, 14
- display
 - Allegro, 12
- doLogic
 - Player, 34
- DOWN
 - Keyboard.h, 46
- draw
 - Plane, 31
- drawB
 - bullet, 17
- DrawBackground
 - Allegro.cc, 36
- drawE
 - EnemyOne, 24
- Ebitmap
 - EnemyOne, 25
- Ebitmap2
 - EnemyOne, 25
- Elist
 - EnemyOne, 25
- Enemy, 21
 - ~Enemy, 21
 - Enemy, 21
 - health, 22
 - moveboss, 22
 - moveEnemy, 22
- enemy
 - Allegro, 12
- Enemy.cc, 41
- Enemy.h, 42
- EnemyOne, 23
 - ~EnemyOne, 24
 - addEnemy, 24
 - drawE, 24
 - Ebitmap, 25
 - Ebitmap2, 25
 - Elist, 25
 - EnemyOne, 23
 - getEnemy, 24
 - setEnemy, 25
- EnemyOne.cc, 43
- EnemyOne.h, 44
- Enemytype, 26
 - Enemytype, 26
 - getMS, 27
 - getX, 27
 - getY, 27
 - moveSpeed, 28
 - setX, 27
 - setY, 27
 - x, 28
 - y, 28
- ENTER
 - Keyboard.h, 46
- event_queue
 - Allegro, 12
- EXbitmap
 - Allegro, 12
- FG
 - Allegro.cc, 37
- fgImage
 - Allegro.cc, 37
- font
 - Allegro, 12
- gameLoop
 - Allegro, 11
- getBitmap
 - Plane, 31
- getBulletMap
 - bullet, 17

- getbX
 - bulletType, 19
- getbY
 - bulletType, 20
- getEnemy
 - EnemyOne, 24
- getMS
 - Enemytype, 27
- getX
 - Enemytype, 27
 - Plane, 31
- getY
 - Enemytype, 27
 - Plane, 32
- health
 - Enemy, 22
 - Player, 34
- HEIGHT
 - Allegro.cc, 37
- height
 - Background, 14
- image
 - Background, 14
- init
 - Allegro, 11
- InitBackground
 - Allegro.cc, 36
- initialPlane
 - Plane, 32
- key
 - Keyboard, 29
- Keyboard, 29
 - ~Keyboard, 29
 - key, 29
 - Keyboard, 29
- keyboard
 - Allegro, 12
- Keyboard.cc, 45
- Keyboard.h, 46
 - DOWN, 46
 - ENTER, 46
 - keys, 46
 - LEFT, 46
 - RIGHT, 46
 - SPACE, 46
 - UP, 46
- keys
 - Keyboard.h, 46
- LEFT
 - Keyboard.h, 46
- LObixmap
 - Allegro, 12
- looping
 - Allegro, 12
- LOST
 - Allegro.h, 38
- main
 - main.cc, 47
- main.cc, 47
 - main, 47
- MG
 - Allegro.cc, 37
- mgImage
 - Allegro.cc, 37
- moveboss
 - Enemy, 22
- moveBullet
 - Player, 34
- moveEnemy
 - Enemy, 22
- moveSpeed
 - Enemytype, 28
 - Player, 34
- Plane, 30
 - ~Plane, 31
 - bitmap, 32
 - draw, 31
 - getBitmap, 31
 - getX, 31
 - getY, 32
 - initialPlane, 32
 - Plane, 30
 - setBitmap, 32
 - x, 32
 - y, 32
- Plane.cc, 48
- Plane.h, 49
- PLAY
 - Allegro.h, 38
- Player, 33
 - ~Player, 33
 - bMoveSpeed, 34
 - doLogic, 34
 - health, 34
 - moveBullet, 34
 - moveSpeed, 34
 - Player, 33
- player
 - Allegro, 12
- Player.cc, 50
- Player.h, 51

- redraw
 - Allegro, [12](#)
- RIGHT
 - Keyboard.h, [46](#)
- setBitmap
 - Plane, [32](#)
- setBullet
 - bullet, [18](#)
- setbX
 - bulletType, [20](#)
- setbY
 - bulletType, [20](#)
- setEnemy
 - EnemyOne, [25](#)
- setX
 - Enemytype, [27](#)
- setY
 - Enemytype, [27](#)
- shot
 - Allegro.cc, [37](#)
- song
 - Allegro.cc, [37](#)
- songInstance
 - Allegro.cc, [37](#)
- SPACE
 - Keyboard.h, [46](#)
- States
 - Allegro.h, [38](#)
- timer
 - Allegro, [13](#)
- TITLE
 - Allegro.h, [38](#)
- UP
 - Keyboard.h, [46](#)
- UpdateBackground
 - Allegro.cc, [36](#)
- velX
 - Background, [14](#)
- velY
 - Background, [14](#)
- WIDTH
 - Allegro.cc, [37](#)
- width
 - Background, [14](#)
- x
 - Background, [14](#)
 - Enemytype, [28](#)
 - Plane, [32](#)
- y
 - Background, [15](#)
 - Enemytype, [28](#)
 - Plane, [32](#)