Space Defender

Generated by Doxygen 1.13.2

Chapter 1

Todo List

Class Bomb

Add Bomb to the game

Member Bomb::fireWeapon (SpaceShip &shooter) override

Add this function

Member formatPlayerInfo (const Player &player)

Make better, width depends on characters

Class Laser

Add Laser to the game

Member Laser::fireWeapon (SpaceShip &shooter) override

Add this function

Member ScreenHighscore::draw (SpaceDefender &window) override

Add highscores to the screen that is read from a json file

Member ScreenSettings::draw (SpaceDefender &window) override

Add settings to the screen

Member SpaceDefender::checkCollision (std::unique_ptr< SpaceShipEnemy > &ship, std::unique_ptr< Weapon > &bullet)

make the fuction general and happen, ok with takeing in unique pointer adresses

Will the weapons be deleted? Since we are putting bullet in there?

Member SpaceDefender::enemySwarmMovement ()

Make the speed better in regards to double vs int, since most functions use int for the positions

Member SpaceDefender::setScreen (std::unique_ptr< Screen > newScreen)

Add reset function

Member SpaceDefender::SpaceDefender (TDT4102::Point position={100, 50}, int width=600, int height=650, const std::string &title="Space Defender")

Fix how enemies spawn

Need dynamic

Member SpaceShipEnemy::movements (SpaceDefender &window) override

Fix the movement of the enemy

Member SpaceShipEnemy::shooting (SpaceDefender &window) override

Fix shooting of the enemy

Method to find lowest ship in each column

One of said ships fire randomly

Firerate approximatly same as playership

Class SpaceShipPlayer

firerate in regards to different weapomn types

Member SpaceShipPlayer::shooting (SpaceDefender &window) override

Only bullets are fired, consider making it more general. For example having a set weapon type

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 5

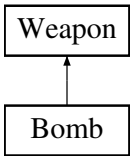
Class Documentation

5.1 Bomb Class Reference

Class for Bomb. Inherits from Weapon.

`#include <Weapon.h>`

Inheritance diagram for Bomb:



Public Member Functions

- **Bomb** (int speed, int damage)
- void fireWeapon (SpaceShip &shooter) override

Public Member Functions inherited from Weapon

- Weapon (int speed, int damage)
- virtual ~Weapon ()=default
- int getSpeed ()
- int getDamage ()
- virtual void move ()
- virtual void draw (SpaceDefender &window)=0
- int **getPositionX** () const
- int **getPositionY** () const
- void **setWeaponSpeed** (int newSpeed)
- virtual int getRadius ()=0

Additional Inherited Members

Protected Attributes inherited from Weapon

- int **speed**
- int **damage**
- int **xProjectile**
- int **yProjectile**

5.1.1 Detailed Description

Class for Bomb. Inherits from Weapon.

Todo Add Bomb to the game

5.1.2 Member Function Documentation

5.1.2.1 fireWeapon()

```
void Bomb::fireWeapon (
            SpaceShip & shooter)  [override], [virtual]
```

Todo Add this function

Parameters

shooter	

Implements Weapon.

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

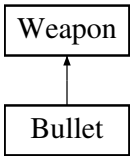
- Weapon.h
- Weapon.cpp

5.2 Bullet Class Reference

Class for Bullet. Inherits from Weapon.

```
#include <Weapon.h>
```

Inheritance diagram for Bullet:



Public Member Functions

- **Bullet** (int speed, int damage)
- void fireWeapon (SpaceShip &shooter) override

  Fire the weapon.
- void draw (SpaceDefender &window) override

  Draw the bullet on the screen.
- int getRadius () override

Public Member Functions inherited from Weapon

- Weapon (int speed, int damage)
- virtual ~Weapon ()=default
- int getSpeed ()
- int getDamage ()
- virtual void move ()
- int **getPositionX** () const
- int **getPositionY** () const
- void **setWeaponSpeed** (int newSpeed)

Additional Inherited Members

Protected Attributes inherited from Weapon

- int **speed**
- int **damage**
- int **xProjectile**
- int **yProjectile**

5.2.1 Detailed Description

Class for Bullet. Inherits from Weapon.

Parameters

radius	Radius of the bullet

5.2.2 Member Function Documentation

5.2.2.1 draw()

```
void Bullet::draw (
            SpaceDefender & window)  [override], [virtual]
```

Draw the bullet on the screen.

Parameters

window	SpaceDefender object
radius	Radius of the bullet
location	Location of the bullet. Updated in Bullet::fireWeapon(SpaceDefender& window)

Implements Weapon.

5.2.2.2 fireWeapon()

```
void Bullet::fireWeapon (
            SpaceShip & shooter)  [override], [virtual]
```

Fire the weapon.

Sets the position of the projectile to the position of the player. The movement of the projectile is handled in Bullet::move().

Parameters

window	SpaceDefender object
xProjectile	Position of the projectile in the x-axis
yProjectile	Position of the projectile in the y-axis

Implements Weapon.

5.2.2.3 getRadius()

```
int Bullet::getRadius ()  [inline], [override], [virtual]
```

Setter for speed

Implements Weapon.

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

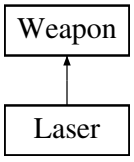
- Weapon.h
- Weapon.cpp

5.3 Laser Class Reference

Class for Laser. Inherits from Weapon.

```
#include <Weapon.h>
```

Inheritance diagram for Laser:



Public Member Functions

- **Laser** (int speed, int damage)
- void fireWeapon (SpaceShip &shooter) override

Public Member Functions inherited from Weapon

- Weapon (int speed, int damage)
- virtual ~Weapon ()=default
- int getSpeed ()
- int getDamage ()
- virtual void move ()
- virtual void draw (SpaceDefender &window)=0
- int **getPositionX** () const
- int **getPositionY** () const
- void **setWeaponSpeed** (int newSpeed)
- virtual int getRadius ()=0

Additional Inherited Members

Protected Attributes inherited from Weapon

- int **speed**
- int **damage**
- int **xProjectile**
- int **yProjectile**

5.3.1 Detailed Description

Class for Laser. Inherits from Weapon.

Todo Add Laser to the game

5.3.2 Member Function Documentation

5.3.2.1 fireWeapon()

```
void Laser::fireWeapon (
            SpaceShip & shooter)  [override], [virtual]
```

Todo Add this function

Parameters

shooter	

Implements Weapon.

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

- Weapon.h
- Weapon.cpp

5.4 Player Struct Reference

An object representing a player in regards to highscores.

`#include <Screen.h>`

Public Attributes

- std::string **rank**
- std::string **name**
- int **score**
- int **round**

5.4.1 Detailed Description

An object representing a player in regards to highscores.

Parameters

rank	The rank of the player
name	The name of the player
score	The score of the player
round	The round of the player

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

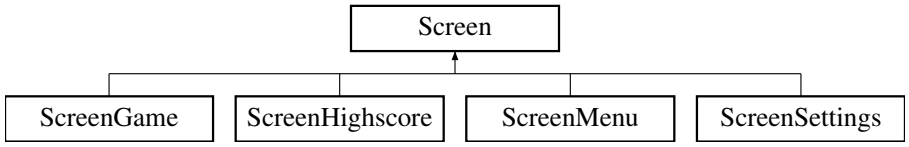
- Screen.h

5.5 Screen Class Reference

Abstract base class for different screens.

`#include <Screen.h>`

Inheritance diagram for Screen:



Public Member Functions

- virtual ~Screen ()=default
- virtual void draw (SpaceDefender &window)=0

5.5.1 Detailed Description

Abstract base class for different screens.

Parameters

window	SpaceDefender object

5.5.2 Constructor & Destructor Documentation

5.5.2.1 ~Screen()

```
virtual Screen::~Screen ()  [virtual], [default]
```

Virtual destructor to ensure proper cleanup

5.5.3 Member Function Documentation

5.5.3.1 draw()

```
virtual void Screen::draw (
            SpaceDefender & window)  [pure virtual]
```

Pure virtual function. Is supposed to draw the screen

Implemented in ScreenGame, ScreenHighscore, ScreenMenu, and ScreenSettings.

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

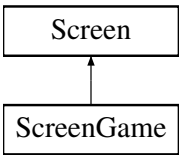
- Screen.h

5.6 ScreenGame Class Reference

The game screen.

```
#include <Screen.h>
```

Inheritance diagram for ScreenGame:



Public Member Functions

- void draw (SpaceDefender &window) override

  Draws the screencontent of the Game.

Public Member Functions inherited from Screen

- virtual ~Screen ()=default

5.6.1 Detailed Description

The game screen.

5.6.2 Member Function Documentation

5.6.2.1 draw()

```
void ScreenGame::draw (
            SpaceDefender & window)  [override], [virtual]
```

Draws the screencontent of the Game.

Draws and updates, enemie ships, the player ship and fired weapons. Itrerates through all the fired weapons and checks for collisions when iterating through all the enemy ships It is an iterator for the fired weapons and itEnemy is an iterator for the enemy ships that is an unique pointer to the enemy ship object if they collide the iterator unique pointer is deleated and the object is then deleated automatically from the vector to move along in the loop we must update the iterator either by deleating it or incrementing it draws every object in window, player ship, enemie ships and fired weapons are deleated from window if hit and helth <= 0

Parameters

window	SpaceDefender object

Implements Screen.

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

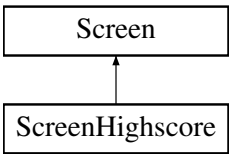
- Screen.h
- Screen.cpp

5.7 ScreenHighscore Class Reference

The highscore screen.

`#include <Screen.h>`

Inheritance diagram for ScreenHighscore:



Public Member Functions

- void draw (SpaceDefender &window) override

  Draws the screencontent of the Highscore.

Public Member Functions inherited from Screen

- virtual ~Screen ()=default

5.7.1 Detailed Description

The highscore screen.

5.7.2 Member Function Documentation

5.7.2.1 draw()

```
void ScreenHighscore::draw (
            SpaceDefender & window)  [override], [virtual]
```

Draws the screencontent of the Highscore.

Draws the highscore screen, and show the back button.

Todo Add highscores to the screen that is read from a json file

Parameters

window	SpaceDefender object
file	The json file variable which contains the highscores

Implements Screen.

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

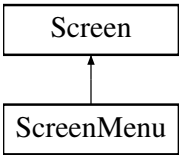
- Screen.h
- Screen.cpp

5.8 ScreenMenu Class Reference

The menu screen.

```
#include <Screen.h>
```

Inheritance diagram for ScreenMenu:



Public Member Functions

- void draw (SpaceDefender &window) override

  Draws the screencontent of the Menu.

Public Member Functions inherited from Screen

- virtual ~Screen ()=default

5.8.1 Detailed Description

The menu screen.

5.8.2 Member Function Documentation

5.8.2.1 draw()

```
void ScreenMenu::draw (
            SpaceDefender & window)  [override], [virtual]
```

Draws the screencontent of the Menu.

Draws the menu screen where it shows all the buttons exept the back button.

Parameters

window	SpaceDefender object

Implements Screen.

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

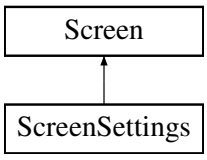
- Screen.h
- Screen.cpp

5.9 ScreenSettings Class Reference

The settings screen.

```
#include <Screen.h>
```

Inheritance diagram for ScreenSettings:



Public Member Functions

- void draw (SpaceDefender &window) override

  Draws the screencontent of the Settings.

Public Member Functions inherited from Screen

- virtual ~Screen ()=default

5.9.1 Detailed Description

The settings screen.

5.9.2 Member Function Documentation

5.9.2.1 draw()

```
void ScreenSettings::draw (
            SpaceDefender & window)  [override], [virtual]
```

Draws the screencontent of the Settings.

Draws the settings screen, where you can change the game settings

Todo Add settings to the screen

Parameters

window	SpaceDefender object

Implements Screen.

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

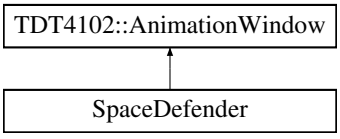
- Screen.h
- Screen.cpp

5.10 SpaceDefender Class Reference

The main game class which runs the game. Uses AnimationWindow as base class.

```
#include <SpaceDefender.h>
```

Inheritance diagram for SpaceDefender:



Public Member Functions

- SpaceDefender (TDT4102::Point position={100, 50}, int width=600, int height=650, const std::string &title="Space Defender")

  Construct a new SpaceDefender::SpaceDefender object.
- void setScreen (std::unique_ptr< Screen > newScreen)

  Set the current screen.
- void run ()

  Game loop that runs the game until the window is closed.
- void findShipToKill ()

  Finds the ship to kill.
- void enemySwarmMovement ()

  Moves the enemy ships swarm as a whole.
- bool checkCollision (std::unique_ptr< SpaceShipEnemy > &ship, std::unique_ptr< Weapon > &bullet)

  takes in a pointer to an enemy ship and a pointer to a weapon and checks if they have collided

Public Attributes

- double **enemySpeed** = 1.0
- int **enemyDirection** = 1
- int **enemyDropDistance** = 7
- int **enemiesDropCounter** = 0
- int **enemyShipCount** = 50
- std::chrono::steady_clock::time_point **lastShotTimeAlien**
- const std::chrono::milliseconds **fireRate** = std::chrono::milliseconds(1000)
- TDT4102::Button **StartGameBtn**
- TDT4102::Button **HighscoresBtn**
- TDT4102::Button **SettingsBtn**
- TDT4102::Button **EndGameBtn**
- TDT4102::Button **GoToMenuBtn**
- SpaceShipPlayer **playerShip**
- std::vector< std::unique_ptr< SpaceShipEnemy > > **enemyShips**
- std::vector< std::unique_ptr< Weapon > > **firedWeapons**

5.10.1 Detailed Description

The main game class which runs the game. Uses AnimationWindow as base class.

Parameters

currentScreen	Pointer to the current screen
btnWidth	Width of the buttons. Relative to window width
btnHeight	Height of the buttons. Relative to window height
playerShip	PlayerShip object
enemyShips	Vector of EnemyShip objects
firedWeapons	Vector of Weapon objects
enemySpeed	Speed of the enemy ships swarm
enemyDirection	Direction of the enemy ships swarm, 1 = right, -1 = left

5.10.2 Constructor & Destructor Documentation

5.10.2.1 SpaceDefender()

```
SpaceDefender::SpaceDefender (
            TDT4102::Point position = {100, 50},
            int width = 600,
            int height = 650,
            const std::string & title = "Space Defender")
```

Construct a new SpaceDefender::SpaceDefender object.

Parameters

position	Position of where the window starts in upper left corner
width	The width of the window
height	The height of the window
title	The title of the window
numEnemiesHeight	The number of enemies in the height of the window
numEnemiesWidth	The number of enemies in the width of the window

Todo Fix how enemies spawn

Todo Need dynamic

5.10.3 Member Function Documentation

5.10.3.1 checkCollision()

```
bool SpaceDefender::checkCollision (
            std::unique_ptr< SpaceShipEnemy > & itEnemy,
            std::unique_ptr< Weapon > & it)
```

takes in a pointer to an enemy ship and a pointer to a weapon and checks if they have collided

Todo Will the weapons be deleted? Since we are putting bullet in there?

the unique pointer should be deleated if they have collided

Parameters

itEnemy	the pointer to the enemy ship
it	the pointer to the weapon class often a bullet

Returns

true if they have collided, false if not

Todo make the fuction general and happen, ok with takeing in unique pointer adresses

5.10.3.2 enemySwarmMovement()

```
void SpaceDefender::enemySwarmMovement ()
```

Moves the enemy ships swarm as a whole.

Updates the position of each enemy ship in the swarm. If any enemy reaches the screen edge, the swarm reverses direction. Every 4th edge hit causes the swarm to drop down vertically. The swarm speed increases as the number of remaining enemies decreases.

Parameters

enemiesShouldDrop	Bool that determines if the swarm should drop when reaching the edge 4 times.
deltaX	Precompute horizontal movement to avoid repeated computation

Todo Make the speed better in regards to double vs int, since most functions use int for the positions

5.10.3.3 findShipToKill()

```
void SpaceDefender::findShipToKill ()
```

Finds the ship to kill.

This function determines which enemy ship should fire a shot in the game. It ensures that only the front-line ships (i.e., the lowest ships in each column) are eligible to shoot. It then selects one of them at random to perform the shooting action, based on a timing condition.

Parameters

now	The current time
lowestShipsMap	The map of the lowest ships in each column

5.10.3.4 run()

```
void SpaceDefender::run ()
```

Game loop that runs the game until the window is closed.

Updates the game state and draws the current screen

Parameters

currentScreen	Draws the current screen as long as its not a null pointer

5.10.3.5 setScreen()

```
void SpaceDefender::setScreen (
            std::unique_ptr< Screen > newScreen)
```

Set the current screen.

Replaces the current screen with std::move(newScreen) of the unique_ptr<Screen>

Parameters

newScreen	The new screen that we want to point to
currentScreen	The current screen that the pointer points to

Todo Add reset function

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

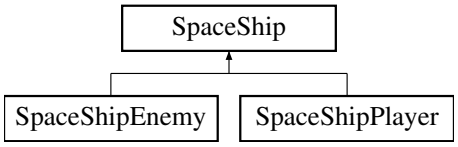
- SpaceDefender.h
- SpaceDefender.cpp

5.11 SpaceShip Class Reference

Abstract base class for different spaceships.

`#include <SpaceShip.h>`

Inheritance diagram for SpaceShip:



Public Member Functions

- SpaceShip (int startX, int startY, int startHealth)

  Constructor that initializes x, y and health.
- virtual ~SpaceShip ()=default
- virtual void movements (SpaceDefender &window)=0
- virtual void shooting (SpaceDefender &window)=0
- void healthReduction (const int &a=1)

  Reduces the Health of the ship by amount standard 1.
- int getHealth () const
- int getPositionX () const

  Getter for position in x-axis.
- int getPositionY () const
- int getShipHeight () const
- int getShipWidth () const
- void setShipSpeed (const double &newSpeed)

  Set the Ship Speed object.
- void updatePosition (const int &movementX, const int &movementY=0)

  Setter for x-position.

Protected Attributes

- int **x**
- int **y**
- int **health**
- const int **shipHeight** = 20
- const int **shipWidth** = 20
- double **shipSpeed**

5.11.1 Detailed Description

Abstract base class for different spaceships.

Parameters

window	SpaceDefender object
x	Position in the x-axis
y	Position in the y-axis
health	Health of the ship
shipHeight	Height of the ship
shipWidth	Width of the ship
shipSpeed	Speed of the ship

5.11.2 Constructor & Destructor Documentation

5.11.2.1 SpaceShip()

```
SpaceShip::SpaceShip (
            int startX,
            int startY,
            int startHealth)  [inline]
```

Constructor that initializes x, y and health.

Parameters

startX	
startY	
startHealth	

5.11.2.2 ~SpaceShip()

```
virtual SpaceShip::~SpaceShip ()  [virtual], [default]
```

Virtual destructor to ensure proper cleanup

5.11.3 Member Function Documentation

5.11.3.1 getHealth()

```
int SpaceShip::getHealth () const  [inline]
```

Getter for health

5.11.3.2 getPositionX()

```
int SpaceShip::getPositionX () const  [inline]
```

Getter for position in x-axis.

Returns

int

5.11.3.3 getPositionY()

```
int SpaceShip::getPositionY () const  [inline]
```

Getter for position in y-axis

5.11.3.4 getShipHeight()

```
int SpaceShip::getShipHeight () const  [inline]
```

Getter for ship height

5.11.3.5 getShipWidth()

```
int SpaceShip::getShipWidth () const  [inline]
```

Getter for ship width

5.11.3.6 healthReduction()

```
void SpaceShip::healthReduction (
            const int & a = 1)  [inline]
```

Reduces the Health of the ship by amount standard 1.

Parameters

a	How much health is reduced

5.11.3.7 movements()

```
virtual void SpaceShip::movements (
            SpaceDefender & window)  [pure virtual]
```

Pure virtual function. Is supposed to move the spaceship

Implemented in SpaceShipEnemy, and SpaceShipPlayer.

5.11.3.8 setShipSpeed()

```
void SpaceShip::setShipSpeed (
            const double & newSpeed)  [inline]
```

Set the Ship Speed object.

Parameters

newSpeed	The new speed of the spaceship

5.11.3.9 shooting()

```
virtual void SpaceShip::shooting (
            SpaceDefender & window)  [pure virtual]
```

Pure virtual function. Is supposed do shooting

Implemented in SpaceShipEnemy, and SpaceShipPlayer.

5.11.3.10 updatePosition()

```
void SpaceShip::updatePosition (
            const int & movementX,
            const int & movementY = 0)  [inline]
```

Setter for x-position.

Parameters

movementX	The movement in x-axis
movementY	The movement in y-axis

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

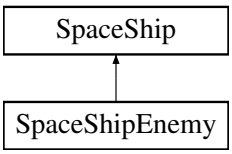
- SpaceShip.h

5.12 SpaceShipEnemy Class Reference

Class for enemy ship.

```
#include <SpaceShip.h>
```

Inheritance diagram for SpaceShipEnemy:



Public Member Functions

- **SpaceShipEnemy** (int startX, int startY)
- void movements (SpaceDefender &window) override

  Moves the spaceship of the enemy.
- void shooting (SpaceDefender &window) override

  Fires the weapon at some interval.

Public Member Functions inherited from SpaceShip

- SpaceShip (int startX, int startY, int startHealth)

  Constructor that initializes x, y and health.
- virtual ~SpaceShip ()=default
- void healthReduction (const int &a=1)

  Reduces the Health of the ship by amount standard 1.
- int getHealth () const
- int getPositionX () const

  Getter for position in x-axis.
- int getPositionY () const
- int getShipHeight () const
- int getShipWidth () const
- void setShipSpeed (const double &newSpeed)

  Set the Ship Speed object.
- void updatePosition (const int &movementX, const int &movementY=0)

  Setter for x-position.

Public Attributes

- TDT4102::Image **alienImage**

Additional Inherited Members

Protected Attributes inherited from SpaceShip

- int **x**
- int **y**
- int **health**
- const int **shipHeight** = 20
- const int **shipWidth** = 20
- double **shipSpeed**

5.12.1 Detailed Description

Class for enemy ship.

Parameters

alienImage	Image of the alienship

5.12.2 Member Function Documentation

5.12.2.1 movements()

```
void SpaceShipEnemy::movements (
            SpaceDefender & window)  [override], [virtual]
```

Moves the spaceship of the enemy.

Make a bullet, fire the and stores it in a vector

Parameters

window	SpaceDefender object

Todo Fix the movement of the enemy

Implements SpaceShip.

5.12.2.2 shooting()

```
void SpaceShipEnemy::shooting (
            SpaceDefender & window)  [override], [virtual]
```

Fires the weapon at some interval.

A random enenmy fires a bullet and stores it in a vector

Parameters

window	SpaceDefender object

Todo Fix shooting of the enemy

Method to find lowest ship in each column

One of said ships fire randomly

Firerate approximatly same as playership

Implements SpaceShip.

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

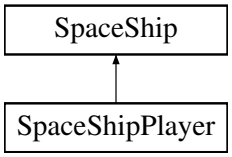
- SpaceShip.h
- SpaceShip.cpp

5.13 SpaceShipPlayer Class Reference

Class for player ship.

`#include <SpaceShip.h>`

Inheritance diagram for SpaceShipPlayer:



Public Member Functions

- **SpaceShipPlayer** (int startX, int startY)
- void movements (SpaceDefender &window) override

  Move the spaceship in x-axis using the arrow keys.
- void shooting (SpaceDefender &window) override

  Fires the weapon if the space key is pressed.

Public Member Functions inherited from SpaceShip

- SpaceShip (int startX, int startY, int startHealth)

  Constructor that initializes x, y and health.
- virtual ~SpaceShip ()=default
- void healthReduction (const int &a=1)

  Reduces the Health of the ship by amount standard 1.
- int getHealth () const
- int getPositionX () const

  Getter for position in x-axis.
- int getPositionY () const
- int getShipHeight () const
- int getShipWidth () const
- void setShipSpeed (const double &newSpeed)

  Set the Ship Speed object.
- void updatePosition (const int &movementX, const int &movementY=0)

  Setter for x-position.

Public Attributes

- TDT4102::Image **playerImage**

Additional Inherited Members

Protected Attributes inherited from SpaceShip

- int **x**
- int **y**
- int **health**
- const int **shipHeight** = 20
- const int **shipWidth** = 20
- double **shipSpeed**

5.13.1 Detailed Description

Class for player ship.

Parameters

playerImage	Image of the playership
lastShotTime	The time of the last
fireRate	How often the weapon can be fired

Todo firerate in regards to different weapomn types

5.13.2 Member Function Documentation

5.13.2.1 movements()

```
void SpaceShipPlayer::movements (
            SpaceDefender & window)  [override], [virtual]
```

Move the spaceship in x-axis using the arrow keys.

Parameters

window	SpaceDefender object
x	Position of the spaceship in the x-axis

Implements SpaceShip.

5.13.2.2 shooting()

```
void SpaceShipPlayer::shooting (
            SpaceDefender & window)  [override], [virtual]
```

Fires the weapon if the space key is pressed.

Fires a bullet, fire the and stores it in a vector

Parameters

window	SpaceDefender object
newBullet	Creates a new Bullet
new	The time at the point when the function is called

Todo Only bullets are fired, consider making it more general. For example having a set weapon type

Implements SpaceShip.

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

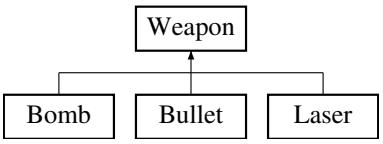
- SpaceShip.h
- SpaceShip.cpp

5.14 Weapon Class Reference

Abstract base class for different weapons.

`#include <Weapon.h>`

Inheritance diagram for Weapon:



Public Member Functions

- Weapon (int speed, int damage)
- virtual ~Weapon ()=default
- virtual void fireWeapon (SpaceShip &shooter)=0
- int getSpeed ()
- int getDamage ()
- virtual void move ()
- virtual void draw (SpaceDefender &window)=0
- int **getPositionX** () const
- int **getPositionY** () const
- void **setWeaponSpeed** (int newSpeed)
- virtual int getRadius ()=0

Protected Attributes

- int **speed**
- int **damage**
- int **xProjectile**
- int **yProjectile**

5.14.1 Detailed Description

Abstract base class for different weapons.

Parameters

speed	Speed of the projectile
damage	Damage the projectile does
xProjectile	Position of the projectile in the x-axis
yProjectile	Position of the projectile in the y-axis
window	SpaceDefender object

5.14.2 Constructor & Destructor Documentation

5.14.2.1 Weapon()

```
Weapon::Weapon (
            int speed,
            int damage)  [inline]
```

Constructor that initializes speed and damage

5.14.2.2 ~Weapon()

```
virtual Weapon::~Weapon ()  [virtual], [default]
```

Virtual destructor to ensure proper cleanup

5.14.3 Member Function Documentation

5.14.3.1 draw()

```
virtual void Weapon::draw (
            SpaceDefender & window)  [pure virtual]
```

Pure virtual function. Is supposed to draw the projectile

Implemented in Bullet.

5.14.3.2 fireWeapon()

```
virtual void Weapon::fireWeapon (
            SpaceShip & shooter)  [pure virtual]
```

Pure virtual function. Is supposed to fire the projectile, aka get the postion when fired.

Implemented in Bomb, Bullet, and Laser.

5.14.3.3 getDamage()

```
int Weapon::getDamage ()  [inline]
```

Getter for damage

5.14.3.4 getRadius()

```
virtual int Weapon::getRadius ()  [pure virtual]
```

Setter for speed

Implemented in Bullet.

5.14.3.5 getSpeed()

```
int Weapon::getSpeed ()  [inline]
```

Getter for speed

5.14.3.6 move()

```
virtual void Weapon::move ()  [inline], [virtual]
```

Move the projectile in y-axis

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

- Weapon.h

Chapter 6

File Documentation

6.1 main.cpp File Reference

Main file.

```
#include "std_lib_facilities.h"
#include "SpaceDefender.h"
#include <iostream>
#include <fstream>
```

Functions

- int main ()

  Starts the game.

6.1.1 Detailed Description

Main file.

Author

Tor Gunnar Ravatn Hammer (tor.ravatn@gmail.com)

Gabriel Anton Norheim ()

Version

1.0

Date

2025-04-01

Copyright

Copyright (c) 2025

6.1.2 Function Documentation

6.1.2.1 main()

```
int main ()
```

Starts the game.

Parameters

game	SpaceDefender object

Returns

Returns 0 on success

6.2 Screen.cpp File Reference

The cpp file for the Screen class.

```
#include <std_lib_facilities.h>
#include "Screen.h"
#include "SpaceDefender.h"
#include <iostream>
#include <vector>
#include <fstream>
#include <sstream>
#include <iomanip>
```

Functions

- std::string formatPlayerInfo (const Player &player)
  Reads the highscores from a json file.

6.2.1 Detailed Description

The cpp file for the Screen class.

Author

Tor Gunnar Ravatn Hammer (tor.ravatn@gmail.com)

Version

1.0

Date

2025-04-01

Copyright

Copyright (c) 2025

6.2.2 Function Documentation

6.2.2.1 formatPlayerInfo()

```
std::string formatPlayerInfo (
            const Player & player)
```

Reads the highscores from a json file.

Parameters

filename	The name of the json file to be read

Returns

std::vector<Player>

Function to format the text that is drawn for each highscore

Parameters

player	A single input of the Player struct

Returns

std::string of the formatted player info

Todo Make better, width depends on characters

6.3 Screen.h File Reference

The header file for the Screen class.

```
#include <string>
```

Classes

- struct Player
  An object representing a player in regards to highscores.
- class Screen
  Abstract base class for different screens.
- class ScreenMenu
  The menu screen.
- class ScreenGame
  The game screen.
- class ScreenHighscore
  The highscore screen.
- class ScreenSettings
  The settings screen.

6.3.1 Detailed Description

The header file for the Screen class.

Author

Tor Gunnar Ravatn Hammer (tor.ravatn@gmail.com)

Version

1.0

Date

2025-04-01

Copyright

6.4 Screen.h

Go to the documentation of this file.

```
00001
00011 #pragma once
00012 #include <string>
00013
00014
00024 struct Player {
00025     std::string rank;
00026     std::string name;
00027     int score;
00028     int round;
00029 };
00030
00031 //std::vector<Player> readScores(const std::string& filename = "highscores.json");
00032
00033 class SpaceDefender;  // Forward declaration of SpaceDefender to avoid circular dependency
00034
00040 class Screen {
00041 public:
00042     virtual ~Screen() = default;
00043     virtual void draw(SpaceDefender& window) = 0;
00044 };
00045
00046
00051 class ScreenMenu : public Screen {
00052 public:
00053     void draw(SpaceDefender& window) override;
00054 };
00055
00056
00061 class ScreenGame : public Screen {
00062 public:
00063     void draw(SpaceDefender& window) override;
00064 };
00065
00066
00071 class ScreenHighscore : public Screen {
00072 public:
00073     void draw(SpaceDefender& window) override;
00074 };
00075
00076
00081 class ScreenSettings : public Screen {
00082 public:
00083     void draw(SpaceDefender& window) override;
00084 };
```

6.5 SpaceDefender.cpp File Reference

The cpp file for the SpaceDefender class.

```
#include "SpaceDefender.h"
#include <iostream>
#include <random>
```

6.5.1 Detailed Description

The cpp file for the SpaceDefender class.

Author

Tor Gunnar Ravatn Hammer (tor.ravatn@gmail.com)

Gabriel Anton Norheim ()

Version

1.0

Date

2025-04-01

Copyright

6.6 SpaceDefender.h File Reference

The header file for the SpaceDefender class.

```
#include "AnimationWindow.h"
#include "widgets/Button.h"
#include "Screen.h"
#include "SpaceShip.h"
#include "Weapon.h"
```

Classes

- class SpaceDefender

  The main game class which runs the game. Uses AnimationWindow as base class.

6.6.1 Detailed Description

The header file for the SpaceDefender class.

Author

Tor Gunnar Ravatn Hammer (tor.ravatn@gmail.com)

Version

1.0

Date

2025-04-01

Copyright

6.7 SpaceDefender.h

Go to the documentation of this file.

```
00001
00011 #pragma once
00012
00013 #include "AnimationWindow.h"
00014 #include "widgets/Button.h"
00015
00016 #include "Screen.h"
00017 #include "SpaceShip.h"
00018 #include "Weapon.h"
00019
00033 class SpaceDefender : public TDT4102::AnimationWindow {
00034 private:
00035     std::unique_ptr<Screen> currentScreen;
00036     unsigned int btnWidth;
00037     unsigned int btnHeight;
00038
00039     // Callback functions for buttons
00040     void cb_endGame()        {close();}
00041     void cb_startGame()      {setScreen(std::make_unique<ScreenGame>());}
00042     void cb_showHighscores() {setScreen(std::make_unique<ScreenHighscore>());}
00043     void cb_settings()       {setScreen(std::make_unique<ScreenSettings>());}
00044     void cb_menu()           {setScreen(std::make_unique<ScreenMenu>()); }
00045
00046 public:
00047     SpaceDefender(TDT4102::Point position = {100, 50}, int width = 600, int height = 650, const
      std::string& title = "Space Defender");
00048     void setScreen(std::unique_ptr<Screen> newScreen);
00049     void run();
00050     // Helper functions
00051     void findShipToKill();
00052     void enemySwarmMovement();
00053     double enemySpeed = 1.0;                // Pixels per movement step
00054     int enemyDirection = 1;             // 1 for right, -1 for left
00055     int enemyDropDistance = 7;          // Drop when hitting screen edge
00056     int enemiesDropCounter = 0;
00057     int enemyShipCount = 50;
00058
00059     std::chrono::steady_clock::time_point lastShotTimeAlien;
00060     const std::chrono::milliseconds fireRate = std::chrono::milliseconds(1000);
00061
00062     // Buttons
00063     TDT4102::Button StartGameBtn;
00064     TDT4102::Button HighscoresBtn;
00065     TDT4102::Button SettingsBtn;
00066     TDT4102::Button EndGameBtn;
```

```
00067     TDT4102::Button GoToMenuBtn;
00068
00069     // Spaceships
00070     SpaceShipPlayer playerShip;
00071     std::vector<std::unique_ptr<SpaceShipEnemy» enemyShips; //canged to unique
00072
00073     // Weapons
00074     std::vector<std::unique_ptr<Weapon» firedWeapons;
00075
00076     //Check collision
00077     bool checkCollision(std::unique_ptr<SpaceShipEnemy>& ship, std::unique_ptr<Weapon>& bullet);
00078 };
```

6.8 SpaceShip.cpp File Reference

The cpp file for the SpaceShip class.

```
#include "SpaceShip.h"
#include "SpaceDefender.h"
#include <iostream>
```

6.8.1 Detailed Description

The cpp file for the SpaceShip class.

Author

Tor Gunnar Ravatn Hammer (tor.ravatn@gmail.com)

Version

1.0

Date

2025-04-01

Copyright

6.9 SpaceShip.h File Reference

The header file for the SpaceShip class.

```
#include <chrono>
#include "subprojects/animationwindow/include/Image.h"
```

Classes

- class SpaceShip

  Abstract base class for different spaceships.
- class SpaceShipPlayer

  Class for player ship.
- class SpaceShipEnemy

  Class for enemy ship.

6.9.1 Detailed Description

The header file for the SpaceShip class.

Author

Tor Gunnar Ravatn Hammer (tor.ravatn@gmail.com)

Version

1.0

Date

2025-04-01

Copyright

6.10 SpaceShip.h

Go to the documentation of this file.

```
00001
00011 #pragma once
00012 #include <chrono>
00013 #include "subprojects/animationwindow/include/Image.h" //for image type
00014
00015 class SpaceDefender;  // Forward declaration of SpaceDefender to avoid circular dependency
00016
00029 class SpaceShip {
00030 public:
00037     SpaceShip(int startX, int startY, int startHealth) : x(startX), y(startY), health(startHealth) {}
00038     virtual ~SpaceShip() = default;
00039     virtual void movements(SpaceDefender& window) = 0;
00040     virtual void shooting(SpaceDefender& window) = 0;
00045     void healthReduction(const int& a = 1) { health -= a; }
00046     int getHealth() const {return health;}
00051     int getPositionX() const {return x;}
00052     int getPositionY() const {return y;}
00053     int getShipHeight() const {return shipHeight;}
00054     int getShipWidth() const {return shipWidth;}
00059     void setShipSpeed(const double &newSpeed) {shipSpeed = newSpeed;}
00065     void updatePosition(const int &movementX, const int &movementY = 0) {x += movementX; y +=
      movementY;}
00066
00067 protected:
00068     int x;
00069     int y;
00070     int health;
00071     const int shipHeight = 20;
```

```
00072     const int shipWidth = 20;
00073     double shipSpeed;
00074 };
00075
00084 class SpaceShipPlayer : public SpaceShip {
00085     public:
00086         SpaceShipPlayer(int startX, int startY) :
00087             SpaceShip(startX, startY, 3),
00088             playerImage("bilder/ShipSprite.png") {
00089             setShipSpeed(10);
00090         }
00091         void movements(SpaceDefender& window) override;
00092         void shooting(SpaceDefender& window) override;
00093         TDT4102::Image playerImage;
00094     private:
00095         std::chrono::steady_clock::time_point lastShotTime;
00096         const std::chrono::milliseconds fireRate = std::chrono::milliseconds(500);
00097 };
00098
00104 class SpaceShipEnemy : public SpaceShip {
00105     public:
00106         SpaceShipEnemy(int startX, int startY) :
00107             SpaceShip(startX, startY, 1),
00108             alienImage("bilder/ailienHead.png") {}
00109         void movements(SpaceDefender& window) override;
00110         void shooting(SpaceDefender& window) override;
00111         TDT4102::Image alienImage;
00112         //bool operator==(const SpaceShipEnemy& other) const { return x == other.x && y == other.y; }
00113 };
```

6.11 Weapon.cpp File Reference

The cpp file for the Weapon class.

```
#include "Weapon.h"
#include "SpaceDefender.h"
#include "SpaceShip.h"
```

6.11.1 Detailed Description

The cpp file for the Weapon class.

Author

Tor Gunnar Ravatn Hammer (tor.ravatn@gmail.com)

Version

1.0

Date

2025-04-01

Copyright

6.12 Weapon.h File Reference

The header file for the Weapon class.

Classes

- class Weapon

  Abstract base class for different weapons.
- class Bullet

  Class for Bullet. Inherits from Weapon.
- class Bomb

  Class for Bomb. Inherits from Weapon.
- class Laser

  Class for Laser. Inherits from Weapon.

6.12.1 Detailed Description

The header file for the Weapon class.

Author

Tor Gunnar Ravatn Hammer (tor.ravatn@gmail.com)

Version

1.0

Date

2025-04-01

Copyright

Copyright (c) 2025

6.13 Weapon.h

Go to the documentation of this file.

```
00001
00011 #pragma once
00012 class SpaceShip;
00013 class SpaceDefender;  // Forward declaration of SpaceDefender to avoid circular dependency
00014
00024 class Weapon {
00025 public:
00026     Weapon(int speed, int damage) : speed(speed), damage(damage) {}
00027     virtual ~Weapon() = default;
00028     virtual void fireWeapon(SpaceShip& shooter) = 0;
00029     int getSpeed() {return speed;}
00030     int getDamage() {return damage;}
00031     virtual void move() {yProjectile -=speed;}
00032     virtual void draw(SpaceDefender& window) = 0;
00033     int getPositionX() const {return xProjectile;};
```

```
00034     int getPositionY() const {return yProjectile;};
00035     void setWeaponSpeed(int newSpeed) {speed = newSpeed;};
00036     virtual int getRadius() = 0; //**< Pure virtual function. Is supposed to get the radius of the
      projectile */
00037 protected:
00038     int speed;
00039     int damage;
00040     int xProjectile;
00041     int yProjectile;
00042 };
00043
00049 class Bullet : public Weapon {
00050 public:
00051     Bullet(int speed, int damage) : Weapon(speed, damage) {}
00052     void fireWeapon(SpaceShip& shooter) override;
00053     void draw(SpaceDefender& window) override;
00054     int getRadius() override {return radius;} //**< Getter for radius */
00055 private:
00056     int radius = 5;
00057 };
00058
00064 class Bomb : public Weapon {
00065 public:
00066     Bomb(int speed, int damage) : Weapon(speed, damage) {}
00067     void fireWeapon(SpaceShip& shooter) override;
00068 };
00069
00075 class Laser : public Weapon {
00076 public:
00077     Laser(int speed, int damage) : Weapon(speed, damage) {}
00078     void fireWeapon(SpaceShip& shooter) override;
00079 };
```