

Space Defender

Generated by Doxygen 1.13.2

1 Todo List	1
2 Hierarchical 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 Bomb Class Reference	9
5.1.1 Detailed Description	10
5.1.2 Member Function Documentation	10
5.1.2.1 fireWeapon()	10
5.2 Bullet Class Reference	10
5.2.1 Detailed Description	11
5.2.2 Member Function Documentation	11
5.2.2.1 draw()	11
5.2.2.2 fireWeapon()	12
5.2.2.3 getRadius()	12
5.3 Laser Class Reference	12
5.3.1 Detailed Description	13
5.3.2 Member Function Documentation	13
5.3.2.1 fireWeapon()	13
5.4 Player Struct Reference	14
5.4.1 Detailed Description	14
5.5 Screen Class Reference	14
5.5.1 Detailed Description	14
5.5.2 Constructor & Destructor Documentation	15
5.5.2.1 ~Screen()	15
5.5.3 Member Function Documentation	15
5.5.3.1 draw()	15
5.6 ScreenGame Class Reference	15
5.6.1 Detailed Description	16
5.6.2 Member Function Documentation	16
5.6.2.1 draw()	16
5.7 ScreenHighscore Class Reference	16
5.7.1 Detailed Description	17
5.7.2 Member Function Documentation	17
5.7.2.1 draw()	17
5.8 ScreenMenu Class Reference	17
5.8.1 Detailed Description	18

5.8.2 Member Function Documentation	18
5.8.2.1 draw()	18
5.9 ScreenSettings Class Reference	18
5.9.1 Detailed Description	19
5.9.2 Member Function Documentation	19
5.9.2.1 draw()	19
5.10 SpaceDefender Class Reference	19
5.10.1 Detailed Description	20
5.10.2 Constructor & Destructor Documentation	21
5.10.2.1 SpaceDefender()	21
5.10.3 Member Function Documentation	21
5.10.3.1 checkCollision()	21
5.10.3.2 enemySwarmMovement()	22
5.10.3.3 findShipToKill()	23
5.10.3.4 run()	23
5.10.3.5 setScreen()	23
5.11 SpaceShip Class Reference	24
5.11.1 Detailed Description	25
5.11.2 Constructor & Destructor Documentation	25
5.11.2.1 SpaceShip()	25
5.11.2.2 ~SpaceShip()	25
5.11.3 Member Function Documentation	25
5.11.3.1 getHealth()	25
5.11.3.2 getPositionX()	26
5.11.3.3 getPositionY()	26
5.11.3.4 getShipHeight()	26
5.11.3.5 getShipWidth()	26
5.11.3.6 healthReduction()	26
5.11.3.7 movements()	26
5.11.3.8 setShipSpeed()	26
5.11.3.9 shooting()	27
5.11.3.10 updatePosition()	27
5.12 SpaceShipEnemy Class Reference	27
5.12.1 Detailed Description	28
5.12.2 Member Function Documentation	29
5.12.2.1 movements()	29
5.12.2.2 shooting()	29
5.13 SpaceShipPlayer Class Reference	30
5.13.1 Detailed Description	31
5.13.2 Member Function Documentation	31
5.13.2.1 movements()	31
5.13.2.2 shooting()	31

5.14 Weapon Class Reference	32
5.14.1 Detailed Description	32
5.14.2 Constructor & Destructor Documentation	33
5.14.2.1 Weapon()	33
5.14.2.2 ~Weapon()	33
5.14.3 Member Function Documentation	33
5.14.3.1 draw()	33
5.14.3.2 fireWeapon()	33
5.14.3.3 getDamage()	34
5.14.3.4 getRadius()	34
5.14.3.5 getSpeed()	34
5.14.3.6 move()	34
6 File Documentation	35
6.1 main.cpp File Reference	35
6.1.1 Detailed Description	35
6.1.2 Function Documentation	35
6.1.2.1 main()	35
6.2 Screen.cpp File Reference	36
6.2.1 Detailed Description	36
6.2.2 Function Documentation	36
6.2.2.1 formatPlayerInfo()	36
6.3 Screen.h File Reference	37
6.3.1 Detailed Description	38
6.4 Screen.h	38
6.5 SpaceDefender.cpp File Reference	39
6.5.1 Detailed Description	39
6.6 SpaceDefender.h File Reference	39
6.6.1 Detailed Description	40
6.7 SpaceDefender.h	40
6.8 SpaceShip.cpp File Reference	41
6.8.1 Detailed Description	41
6.9 SpaceShip.h File Reference	41
6.9.1 Detailed Description	42
6.10 SpaceShip.h	42
6.11 Weapon.cpp File Reference	43
6.11.1 Detailed Description	43
6.12 Weapon.h File Reference	44
6.12.1 Detailed Description	44
6.13 Weapon.h	44

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:

TDT4102::AnimationWindow	
SpaceDefender	19
Player	14
Screen	14
ScreenGame	15
ScreenHighscore	16
ScreenMenu	17
ScreenSettings	18
SpaceShip	24
SpaceShipEnemy	27
SpaceShipPlayer	30
Weapon	32
Bomb	9
Bullet	10
Laser	12

Chapter 3

Class Index

3.1 Class List

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

Bomb	Class for Bomb . Inherits from Weapon	9
Bullet	Class for Bullet . Inherits from Weapon	10
Laser	Class for Laser . Inherits from Weapon	12
Player	An object representing a player in regards to highscores	14
Screen	Abstract base class for different screens	14
ScreenGame	The game screen	15
ScreenHighscore	The highscore screen	16
ScreenMenu	The menu screen	17
ScreenSettings	The settings screen	18
SpaceDefender	The main game class which runs the game. Uses AnimationWindow as base class	19
SpaceShip	Abstract base class for different spaceships	24
SpaceShipEnemy	Class for enemy ship	27
SpaceShipPlayer	Class for player ship	30
Weapon	Abstract base class for different weapons	32

Chapter 4

File Index

4.1 File List

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

main.cpp	Main file	35
Screen.cpp	The cpp file for the Screen class	36
Screen.h	The header file for the Screen class	37
SpaceDefender.cpp	The cpp file for the SpaceDefender class	39
SpaceDefender.h	The header file for the SpaceDefender class	39
SpaceShip.cpp	The cpp file for the SpaceShip class	41
SpaceShip.h	The header file for the SpaceShip class	41
Weapon.cpp	The cpp file for the Weapon class	43
Weapon.h	The header file for the Weapon class	44

Chapter 5

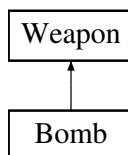
Class Documentation

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

<i>shooter</i>	
----------------	--

Implements [Weapon](#).

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

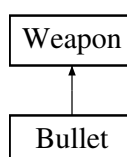
- [Weapon.h](#)
- [Weapon.cpp](#)

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

<i>radius</i>	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

<i>window</i>	SpaceDefender object
<i>radius</i>	Radius of the bullet
<i>location</i>	Location of the bullet. Updated in <code>Bullet::fireWeapon(SpaceDefender& window)</code>

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

<i>window</i>	SpaceDefender object
<i>xProjectile</i>	Position of the projectile in the x-axis
<i>yProjectile</i>	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:

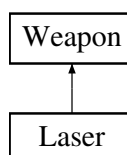
- [Weapon.h](#)
- [Weapon.cpp](#)

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

<i>shooter</i>	
----------------	--

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

<i>rank</i>	The rank of the player
<i>name</i>	The name of the player
<i>score</i>	The score of the player
<i>round</i>	The round of the player

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

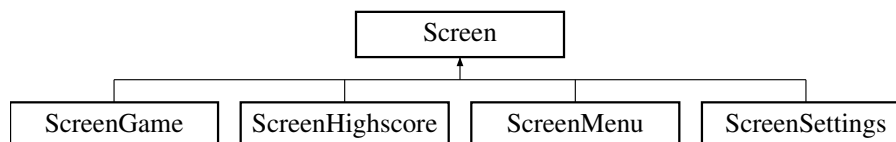
- [Screen.h](#)

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

<i>window</i>	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:

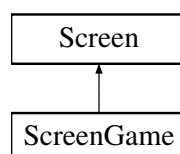
- [Screen.h](#)

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. Iterates 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 a unique pointer to the enemy ship object. If they collide, the iterator unique pointer is deleted and the object is then deleted automatically from the vector. To move along in the loop, we must update the iterator either by deleting it or incrementing it. It draws every object in window, player ship, enemy ships, and fired weapons are deleted from window if hit and health ≤ 0.

Parameters

<i>window</i>	SpaceDefender object
---------------	--------------------------------------

Implements [Screen](#).

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

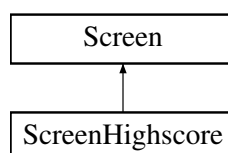
- [Screen.h](#)
- [Screen.cpp](#)

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

<i>window</i>	SpaceDefender object
<i>file</i>	The json file variable which contains the highscores

Implements [Screen](#).

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

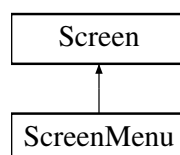
- [Screen.h](#)
- [Screen.cpp](#)

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

<i>window</i>	SpaceDefender object
---------------	--------------------------------------

Implements [Screen](#).

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

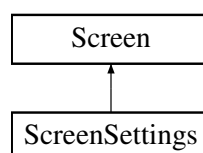
- [Screen.h](#)
- [Screen.cpp](#)

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

<i>window</i>	SpaceDefender object
---------------	--------------------------------------

Implements [Screen](#).

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

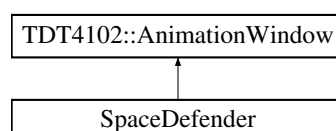
- [Screen.h](#)
- [Screen.cpp](#)

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.
- void [writeScores](#) (const std::string &filename)
- void [addHearts](#) ()
draws the hearts for the player on the screen
- 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

- TDT4102::Image **heart** = TDT4102::Image("bilder/pixel-heart.png")
- double **enemySpeed** = 1.0
- int **enemyDirection** = 1
- int **enemyDropDistance** = 7
- int **enemiesDropCounter** = 0
- int **enemyShipCount** = 50
- int **score** = 0
- 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

<i>currentScreen</i>	Pointer to the current screen
<i>btnWidth</i>	Width of the buttons. Relative to window width
<i>btnHeight</i>	Height of the buttons. Relative to window height
<i>playerShip</i>	PlayerShip object
<i>enemyShips</i>	Vector of EnemyShip objects
<i>firedWeapons</i>	Vector of Weapon objects
<i>enemySpeed</i>	Speed of the enemy ships swarm
<i>enemyDirection</i>	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

<i>position</i>	Position of where the window starts in upper left corner
<i>width</i>	The width of the window
<i>height</i>	The height of the window
<i>title</i>	The title of the window
<i>numEnemiesHeight</i>	The number of enemies in the height of the window
<i>numEnemiesWidth</i>	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 deleted if they have collided

Parameters

<i>itEnemy</i>	the pointer to the enemy ship
<i>it</i>	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 addresses

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

<i>enemiesShouldDrop</i>	Bool that determines if the swarm should drop when reaching the edge 4 times.
<i>deltaX</i>	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

<i>now</i>	The current time
<i>lowestShipsMap</i>	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

<i>currentScreen</i>	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

<i>newScreen</i>	The new screen that we want to point to
<i>currentScreen</i>	The current screen that the pointer points to

Todo Add reset function

5.10.3.6 writeScores()

```
void SpaceDefender::writeScores (
    const std::string & filename)
```

Parameters

filename	
----------	--

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

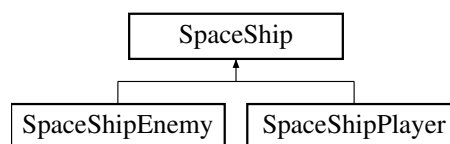
- [SpaceDefender.h](#)
- [SpaceDefender.cpp](#)

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

<i>window</i>	SpaceDefender object
<i>x</i>	Position in the x-axis
<i>y</i>	Position in the y-axis
<i>health</i>	Health of the ship
<i>shipHeight</i>	Height of the ship
<i>shipWidth</i>	Width of the ship
<i>shipSpeed</i>	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

<i>startX</i>	
<i>startY</i>	
<i>startHealth</i>	

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

<i>a</i>	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

<i>newSpeed</i>	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

<i>movementX</i>	The movement in x-axis
<i>movementY</i>	The movement in y-axis

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

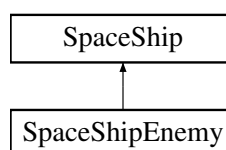
- [SpaceShip.h](#)

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

<i>alienImage</i>	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

<i>window</i>	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 enemy fires a bullet and stores it in a vector

Parameters

<i>window</i>	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:

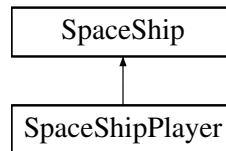
- [SpaceShip.h](#)
- [SpaceShip.cpp](#)

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

<i>playerImage</i>	Image of the playership
<i>lastShotTime</i>	The time of the last
<i>fireRate</i>	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

<i>window</i>	SpaceDefender object
<i>x</i>	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

<i>window</i>	SpaceDefender object
<i>newBullet</i>	Creates a new Bullet
<i>new</i>	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:

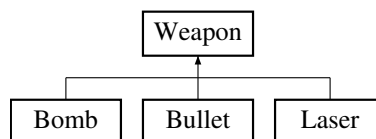
- [SpaceShip.h](#)
- [SpaceShip.cpp](#)

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

<i>speed</i>	Speed of the projectile
<i>damage</i>	Damage the projectile does
<i>xProjectile</i>	Position of the projectile in the x-axis
<i>yProjectile</i>	Position of the projectile in the y-axis
<i>window</i>	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

<i>game</i>	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>
#include "nlohmann/json.hpp"
```

Typedefs

- using **json** = nlohmann::json

Functions

- std::vector< [Player](#) > [readScores](#) (const std::string &filename)
Reads the highscores from a json file.
- std::string [formatPlayerInfo](#) (const [Player](#) &player)
Function to format the text that is drawn for each highscore.

6.2.1 Detailed Description

The cpp file for the [Screen](#) class.

Author

Tor Gunnar Ravatn Hammer (tor.ravatn@gmail.com)
Gabriel Anton Norheim (gabriel.norheim@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)
```

Function to format the text that is drawn for each highscore.

Parameters

<i>player</i>	A single input of the Player struct
---------------	---

Returns

std::string of the formatted player info

Todo Make better, width depends on characters

6.2.2.2 readScores()

```
std::vector< Player > readScores (  
    const std::string & filename)
```

Reads the highscores from a json file.

Parameters

<i>filename</i>	The name of the json file to be read
-----------------	--------------------------------------

Returns

std::vector<Player>

6.3 Screen.h File Reference

The header file for the [Screen](#) class.

```
#include <string>  
#include <chrono>
```

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

Copyright (c) 2025

6.4 Screen.h

[Go to the documentation of this file.](#)

```
00001
00011 #pragma once
00012 #include <string>
00013 #include <chrono>
00014
00015
00025 struct Player {
00026     std::string rank;
00027     std::string name;
00028     int score;
00029     int round;
00030 };
00031
00032 //std::vector<Player> readScores(const std::string& filename = "highscores.json");
00033
00034 class SpaceDefender; // Forward declaration of SpaceDefender to avoid circular dependency
00035
00041 class Screen {
```

```
00042 public:
00043     virtual ~Screen() = default;
00044     virtual void draw(SpaceDefender& window) = 0;
00045 };
00046
00047
00052 class ScreenMenu : public Screen {
00053 public:
00054     void draw(SpaceDefender& window) override;
00055 };
00056
00057
00062 class ScreenGame : public Screen {
00063 public:
00064     void draw(SpaceDefender& window) override;
00065 };
00066
00067
00072 class ScreenHighscore : public Screen {
00073 public:
00074     void draw(SpaceDefender& window) override;
00075 };
00076
00077
00082 class ScreenSettings : public Screen {
00083 public:
00084     void draw(SpaceDefender& window) override;
00085 };
```

6.5 SpaceDefender.cpp File Reference

The cpp file for the [SpaceDefender](#) class.

```
#include "SpaceDefender.h"
#include <iostream>
#include <random>
#include <fstream>
#include "nlohmann/json.hpp"
```

Functions

- `std::string getOrdinal (int number)`
Get the Ordinal object.

6.5.1 Detailed Description

The cpp file for the [SpaceDefender](#) class.

Author

Tor Gunnar Ravatn Hammer (tor.ravatn@gmail.com)
Gabriel Anton Norheim (gabriel.norheim@gmail.com)

Version

1.0

Date

2025-04-01

Copyright

Copyright (c) 2025

6.5.2 Function Documentation

6.5.2.1 getOrdinal()

```
std::string getOrdinal (
    int number)
```

Get the Ordinal object.

Parameters

<i>number</i>	
---------------	--

Returns

std::string

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)
Gabriel Anton Norheim (gabriel.norheim@gmail.com)

Version

1.0

Date

2025-04-01

Copyright

Copyright (c) 2025

6.7 SpaceDefender.h

[Go to the documentation of this file.](#)

```

00001
00012 #pragma once
00013
00014 #include "AnimationWindow.h"
00015 #include "widgets/Button.h"
00016
00017 #include "Screen.h"
00018 #include "SpaceShip.h"
00019 #include "Weapon.h"
00020
00034 class SpaceDefender : public TDT4102::AnimationWindow {
00035 private:
00036     std::unique_ptr<Screen> currentScreen;
00037     unsigned int btnWidth;
00038     unsigned int btnHeight;
00039
00040     // Callback functions for buttons
00041     void cb_endGame() {close();}
00042     void cb_startGame() {setScreen(std::make_unique<ScreenGame>());}
00043     void cb_showHighscores() {setScreen(std::make_unique<ScreenHighscore>());}
00044     void cb_settings() {setScreen(std::make_unique<ScreenSettings>());}
00045     void cb_menu() {setScreen(std::make_unique<ScreenMenu>()); }
00046
00047 public:
00048     SpaceDefender(TDT4102::Point position = {100, 50}, int width = 600, int height = 650, const
std::string& title = "Space Defender");
00049     void setScreen(std::unique_ptr<Screen> newScreen);
00050     void run();
00051     // Helper functions
00052     void findShipToKill();
00053     void enemySwarmMovement();
00054     void writeScores(const std::string& filename);
00055     void addHearts();
00056     TDT4102::Image heart = TDT4102::Image("bilder/pixel-heart.png");
00057     double enemySpeed = 1.0; // Pixels per movement step
00058     int enemyDirection = 1; // 1 for right, -1 for left
00059     int enemyDropDistance = 7; // Drop when hitting screen edge
00060     int enemiesDropCounter = 0;
00061     int enemyShipCount = 50;
00062     int score = 0;
00063
00064     std::chrono::steady_clock::time_point lastShotTimeAlien;
00065     const std::chrono::milliseconds fireRate = std::chrono::milliseconds(1000);
00066
00067     // Buttons
00068     TDT4102::Button StartGameBtn;
00069     TDT4102::Button HighscoresBtn;
00070     TDT4102::Button SettingsBtn;
00071     TDT4102::Button EndGameBtn;
00072     TDT4102::Button GoToMenuBtn;
00073
00074     // Spaceships
00075     SpaceShipPlayer playerShip;
00076     std::vector<std::unique_ptr<SpaceShipEnemy> enemyShips; //canged to unique
00077
00078     // Weapons
00079     std::vector<std::unique_ptr<Weapon> firedWeapons;
00080
00081     //Check collision
00082     bool checkCollision(std::unique_ptr<SpaceShipEnemy>& ship, std::unique_ptr<Weapon>& bullet);
00083 };

```

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)
Gabriel Anton Norheim (gabriel.norheim@gmail.com)

Version

1.0

Date

2025-04-01

Copyright

Copyright (c) 2025

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)
Gabriel Anton Norheim (gabriel.norheim@gmail.com)

Version

1.0

Date

2025-04-01

Copyright

Copyright (c) 2025

6.10 SpaceShip.h

[Go to the documentation of this file.](#)

```

00001
00012 #pragma once
00013 #include <chrono>
00014 #include "subprojects/animationwindow/include/Image.h" //for image type
00015
00016 class SpaceDefender; // Forward declaration of SpaceDefender to avoid circular dependency
00017
00030 class SpaceShip {
00031 public:
00038     SpaceShip(int startX, int startY, int startHealth) : x(startX), y(startY), health(startHealth) {}
00039     virtual ~SpaceShip() = default;
00040     virtual void movements(SpaceDefender& window) = 0;
00041     virtual void shooting(SpaceDefender& window) = 0;
00046     void healthReduction(const int& a = 1) { health -= a; }
00047     int getHealth() const {return health;}
00052     int getPositionX() const {return x;}
00053     int getPositionY() const {return y;}
00054     int getShipHeight() const {return shipHeight;}
00055     int getShipWidth() const {return shipWidth;}
00060     void setShipSpeed(const double &newSpeed) {shipSpeed = newSpeed;}
00066     void updatePosition(const int &movementX, const int &movementY = 0) {x += movementX; y +=
movementY;}
00067
00068 protected:
00069     int x;
00070     int y;
00071     int health;
00072     const int shipHeight = 20;
00073     const int shipWidth = 20;
00074     double shipSpeed;
00075 };
00076
00085 class SpaceShipPlayer : public SpaceShip {
00086 public:
00087     SpaceShipPlayer(int startX, int startY) :
00088         SpaceShip(startX, startY, 3),
00089         playerImage("bilder/ShipSprite.png") {
00090             setShipSpeed(10);
00091         }
00092     void movements(SpaceDefender& window) override;
00093     void shooting(SpaceDefender& window) override;
00094     TDT4102::Image playerImage;
00095 private:
00096     std::chrono::steady_clock::time_point lastShotTime;
00097     const std::chrono::milliseconds fireRate = std::chrono::milliseconds(500);
00098 };
00099
00105 class SpaceShipEnemy : public SpaceShip {
00106 public:
00107     SpaceShipEnemy(int startX, int startY) :
00108         SpaceShip(startX, startY, 1),
00109         alienImage("bilder/aillenHead.png") {}
00110     void movements(SpaceDefender& window) override;
00111     void shooting(SpaceDefender& window) override;
00112     TDT4102::Image alienImage;
00113     //bool operator==(const SpaceShipEnemy& other) const { return x == other.x && y == other.y; }
00114 };

```

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

Copyright (c) 2025

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 };

```

