Statki

Wygenerowano przez Doxygen 1.9.5

Rozdział 1

Indeks hierarchiczny

1.1 Hierarchia klas

Ta lista dziedziczenia posortowana jest z grubsza, choć nie całkowicie, alfabetycznie:

Rozdział 2

Indeks klas

2.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

Rozdział 3

Indeks plików

3.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

Rozdział 4

Dokumentacja klas

4.1 Dokumentacja klasy Board

`#include <board.h>`

Metody publiczne

- Board ()
- ~Board ()
- char getCellValue (unsigned int x, unsigned int y) const
- bool placeShip (unsigned int startX, unsigned int startY, unsigned int shipLength, bool isVertical)
- bool isCellAvailable (unsigned int y, unsigned int x) const
- bool shootAt (unsigned int x, unsigned int y)
- bool isShipSunk (unsigned int x, unsigned int y)
- bool areAllShipsSunk ()
- unsigned int getWidth () const
- unsigned int getHeight () const

4.1.1 Opis szczegółowy

Klasa reprezentująca planszę do gry w statki.

Klasa ta zawiera prywatne składowe reprezentujące właściwości planszy do gry w statki, takie jak szerokość, wysokość, dwuwymiarowa tablica zawierająca zawartość poszczególnych komórek oraz wektor przechowujący statki umieszczone na planszy.

4.1.2 Dokumentacja konstruktora i destruktora

4.1.2.1 Board()

```
Board::Board ( )
```

Konstruktor domyślny.

Inicjalizuje planszę o wymiarach zdefiniowanych w pliku konfiguracyjnym config.h i przygotowuje ją do gry, wypełniając wszystkie komórki znakiem '.'.

4.1.2.2 ∼Board()

```
Board::∼Board ( )
```

Destruktor.

Zwalnia zaalokowaną pamięć po planszy oraz statkach.

4.1.3 Dokumentacja funkcji składowych

4.1.3.1 areAllShipsSunk()

```
bool Board::areAllShipsSunk ( )
```

Sprawdź, czy wszystkie statki na planszy zostały zatopione.

Funkcja ta sprawdza, czy na planszy nie ma już żadnych aktywnych statków, czyli czy wszystkie statki zostały zatopione.

Zwraca

Wartość logiczna informująca o tym, czy wszystkie statki zostały zatopione. Zwraca true, jeśli wszystkie statki są zatopione, a false w przeciwnym razie.

4.1.3.2 getCellValue()

```
char Board::getCellValue (
            unsigned int x,
            unsigned int y ) const
```

Pobierz wartość komórki o określonych współrzędnych.

Funkcja pobiera wartość komórki znajdującej się na określonych współrzędnych (x, y) z planszy gry.

Parametry

x	Współrzędna pozioma x komórki.
y	Współrzędna pionowa y komórki.

Zwraca

Znak reprezentujący zawartość określonej komórki. Jeśli współrzędne są poza zakresem, zwracany jest znak spacji.

4.1.3.3 getHeight()

```
unsigned int Board::getHeight ( ) const
```

Getter - Pobierz wysokość planszy.

Funkcja ta zwraca wysokość planszy gry.

Zwraca

Wartość całkowita unsigned int reprezentująca wysokość planszy.

4.1.3.4 getWidth()

```
unsigned int Board::getWidth ( ) const
```

Getter - Pobierz szerokość planszy.

Funkcja ta zwraca szerokość planszy gry.

Zwraca

Wartość całkowita unsigned int reprezentująca szerokość planszy.

4.1.3.5 isCellAvailable()

```
bool Board::isCellAvailable (
            unsigned int y,
            unsigned int x ) const
```

Sprawdź dostępność komórki na planszy gry.

Funkcja ta sprawdza, czy określona komórka na planszy gry (o współrzędnych y, x) jest dostępna do umieszczenia statku. Komórka jest uznawana za dostępną, jeśli jej wartość to '.' oraz żadna z sąsiednich komórek nie zawiera statku ('S').

Parametry

y	Współrzędna pionowa komórki.
x	Współrzędna pozioma komórki.

Zwraca

Wartość logiczna informująca o dostępności komórki. Zwraca true, jeśli komórka jest dostępna, a false w przeciwnym razie.

4.1.3.6 isShipSunk()

```
bool Board::isShipSunk (
            unsigned int x,
            unsigned int y )
```

Sprawdź, czy statek na planszy jest zatopiony.

Funkcja ta sprawdza, czy statek na planszy, którego początkowe współrzędne to (x, y), jest zatopiony, czyli czy wszystkie jego części zostały trafione.

Parametry

x	Współrzędna pozioma początku statku.
y	Współrzędna pionowa początku statku.

Zwraca

Wartość logiczna informująca o tym, czy statek jest zatopiony. Zwraca true, jeśli statek jest zatopiony, a false w przeciwnym razie.

4.1.3.7 placeShip()

```
bool Board::placeShip (
            unsigned int startX,
            unsigned int startY,
            unsigned int shipLength,
            bool isVertical )
```

Umieść statek na planszy gry.

Funkcja ta umieszcza statek na planszy gry, poczynając od określonych współrzędnych (startX, startY) i o określonej długości (shipLength). Statek może być umieszczony pionowo lub poziomo w zależności od wartości parametru isVertical.

Parametry

startX	Początkowa współrzędna pozioma umieszczenia statku.
startY	Początkowa współrzędna pionowa umieszczenia statku.
shipLength	Długość statku do umieszczenia.
isVertical	Flaga określająca orientację statku (true - pionowy, false - poziomy).

Zwraca

Wartość logiczna informująca o powodzeniu umieszczenia statku. Zwraca true, jeśli umieszczenie było możliwe, a false w przeciwnym razie.

4.1.3.8 shootAt()

```
bool Board::shootAt (
            unsigned int x,
            unsigned int y )
```

Wystrzel w określoną komórkę na planszy gry.

Funkcja ta wykonuje strzał w określoną komórkę o współrzędnych (x, y) na planszy gry. W zależności od zawartości komórki, aktualizuje stan planszy oznaczając trafienie statku ('X') lub brak trafienia ('M').

Parametry

x	Współrzędna pozioma celu strzału.
y	Współrzędna pionowa celu strzału.

Zwraca

Wartość logiczna informująca o rezultacie strzału. Zwraca true, jeśli strzał trafił statek, a false w przeciwnym razie.

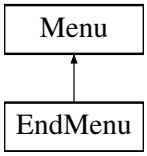
Dokumentacja dla tej klasy została wygenerowana z plików:

- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/board.h
- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/board.cpp

4.2 Dokumentacja klasy EndMenu

```
#include <endMenu.h>
```

Diagram dziedziczenia dla EndMenu



Metody publiczne

- EndMenu (float width, float height)
- void draw (sf::RenderWindow &window) override
- void setWinnerDetails (const std::string &winnerName)
- void positionExitButton ()

Dodatkowe Dziedziczone Składowe

4.2.1 Opis szczegółowy

Klasa reprezentująca menu końcowe gry.

Klasa dziedziczy po klasie Menu, zawiera informacje o zwycięzcy gry oraz opcję wyjścia.

4.2.2 Dokumentacja konstruktora i destruktora

4.2.2.1 EndMenu()

```
EndMenu::EndMenu (
            float width,
            float height )
```

Konstruktor klasy EndMenu.

Parametry

width	Szerokość okna gry.
height	Wysokość okna gry.

4.2.3 Dokumentacja funkcji składowych

4.2.3.1 draw()

```
void EndMenu::draw (
            sf::RenderWindow & window )  [override], [virtual]
```

Rysuj menu końcowe na oknie gry.

Metoda implementuje rysowanie menu końcowego na oknie gry, wyświetlając informacje o zwycięzcy oraz opcję wyjścia.

Parametry

window	Okno gry, na którym rysowane jest menu końcowe.

Reimplementowana z Menu.

4.2.3.2 positionExitButton()

```
void EndMenu::positionExitButton ( )
```

Pozycjonuj przycisk wyjścia w menu końcowym.

Metoda ustawia pozycję przycisku wyjścia w menu końcowym, tak aby był odpowiednio umiejscowiony pod informacjami o zwycięzcy.

4.2.3.3 setWinnerDetails()

```
void EndMenu::setWinnerDetails (
            const std::string & winnerName )
```

Ustaw szczegóły zwycięzcy gry.

Metoda ustawia informacje o zwycięzcy gry, które zostaną wyświetlone w menu końcowym.

Parametry

winnerName	Nazwa zwycięzcy gry.

Dokumentacja dla tej klasy została wygenerowana z plików:

- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/endMenu.h
- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/endMenu.cpp

4.3 Dokumentacja klasy Game

```
#include <game.h>
```

Metody publiczne

- Game ()
- void run ()

4.3.1 Opis szczegółowy

Klasa reprezentująca główny obiekt gry.

Klasa ta zarządza główną pętlą gry, obsługuje zdarzenia, renderuje obrazy i kontroluje przejścia między różnymi stanami gry.

4.3.2 Dokumentacja konstruktora i destruktora

4.3.2.1 Game()

```
Game::Game ( )
```

Konstruktor klasy Game.

4.3.3 Dokumentacja funkcji składowych

4.3.3.1 run()

```
void Game::run ( )
```

Uruchomienie gry.

Metoda ta uruchamia główną pętlę gry, obsługując zdarzenia i renderując obrazy.

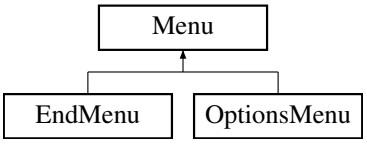
Dokumentacja dla tej klasy została wygenerowana z plików:

- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/game.h
- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/game.cpp

4.4 Dokumentacja klasy Menu

```
#include <menu.h>
```

Diagram dziedziczenia dla Menu



Metody publiczne

- Menu (float width, float height)
- ~Menu ()=default
- virtual void draw (sf::RenderWindow &window)
- virtual void MoveLeft ()
- virtual void MoveRight ()
- virtual void MoveUp ()
- virtual void MoveDown ()
- virtual int GetPressedItem () const
- void updateMenuSize (float width, float height)

Atrybuty publiczne

- sf::Text menu [3]

Metody chronione

- virtual void positionMenuItems ()

Atrybuty chronione

- sf::Font font
- float screenWidth
- float screenHeight
- int selectedItemIndex

4.4.1 Dokumentacja konstruktora i destruktora

4.4.1.1 Menu()

```
Menu::Menu (
            float width,
            float height )
```

Konstruktor klasy Menu.

Parametry

width	Szerokość okna gry.
height	Wysokość okna gry.

4.4.1.2 ~Menu()

```
Menu::~Menu ( )  [default]
```

Destruktor klasy Menu (domyślny).

4.4.2 Dokumentacja funkcji składowych

4.4.2.1 draw()

```
void Menu::draw (
            sf::RenderWindow & window )  [virtual]
```

Metoda rysująca menu na oknie.

Parametry

window	Referencja do obiektu sf::RenderWindow, na którym rysowane jest menu.

Reimplementowana w EndMenu i OptionsMenu.

4.4.2.2 GetPressedItem()

```
int Menu::GetPressedItem ( ) const  [virtual]
```

Metoda zwracająca indeks wybranego elementu w menu.

Zwraca

Indeks wybranego elementu w menu.

Reimplementowana w OptionsMenu.

4.4.2.3 MoveDown()

```
virtual void Menu::MoveDown ( )  [inline], [virtual]
```

Metoda obsługująca ruch w dół w menu (pusta implementacja w bazowej klasie).

Reimplementowana w OptionsMenu.

4.4.2.4 MoveLeft()

```
void Menu::MoveLeft ( )  [virtual]
```

Metoda obsługująca ruch w lewo w menu.

4.4.2.5 MoveRight()

```
void Menu::MoveRight ( )  [virtual]
```

Metoda obsługująca ruch w prawo w menu.

4.4.2.6 MoveUp()

```
virtual void Menu::MoveUp ( )  [inline], [virtual]
```

Metoda obsługująca ruch w górę w menu (pusta implementacja w bazowej klasie).

Reimplementowana w OptionsMenu.

4.4.2.7 positionMenuItems()

```
void Menu::positionMenuItems ( )  [protected], [virtual]
```

Metoda ustawiająca pozycje elementów menu na podstawie aktualnych rozmiarów okna.

4.4.2.8 updateMenuSize()

```
void Menu::updateMenuSize (
            float width,
            float height )
```

Metoda aktualizująca rozmiar menu z uwzględnieniem zmiany rozdzielczości okna.

Parametry

width	Nowa szerokość okna.
height	Nowa wysokość okna.

4.4.3 Dokumentacja atrybutów składowych

4.4.3.1 font

```
sf::Font Menu::font  [protected]
```

Obiekt czcionki używanej do tekstu w menu.

4.4.3.2 menu

```
sf::Text Menu::menu[3]
```

4.4.3.3 screenHeight

```
float Menu::screenHeight  [protected]
```

Wysokość okna gry.

4.4.3.4 screenWidth

```
float Menu::screenWidth  [protected]
```

Szerokość okna gry.

4.4.3.5 selectedItemIndex

```
int Menu::selectedItemIndex  [protected]
```

Indeks aktualnie zaznaczonego elementu w menu.

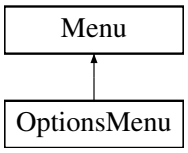
Dokumentacja dla tej klasy została wygenerowana z plików:

- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/menu.h
- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/menu.cpp

4.5 Dokumentacja klasy OptionsMenu

`#include <optionsMenu.h>`

Diagram dziedziczenia dla OptionsMenu



Metody publiczne

- OptionsMenu (float width, float height)
- ~OptionsMenu ()=default
- int GetPressedItem () const override
- void draw (sf::RenderWindow &window) override
- void MoveUp () override
- void MoveDown () override
- void toggleFullscreen ()
- void updateSizeAndPosition (float newWidth, float newHeight)
- bool isFullscreen () const
- Resolution getSelectedResolution () const

Dodatkowe Dziedziczone Składowe

4.5.1 Opis szczegółowy

Klasa reprezentująca menu opcji w grze.

4.5.2 Dokumentacja konstruktora i destruktora

4.5.2.1 OptionsMenu()

```
OptionsMenu::OptionsMenu (
            float width,
            float height )
```

Konstruktor klasy OptionsMenu.

Parametry

width	Szerokość okna gry.
height	Wysokość okna gry.

4.5.2.2 ~OptionsMenu()

```
OptionsMenu::~OptionsMenu ( )  [default]
```

Destruktor klasy OptionsMenu (domyślny).

4.5.3 Dokumentacja funkcji składowych

4.5.3.1 draw()

```
void OptionsMenu::draw (
            sf::RenderWindow & window )  [override], [virtual]
```

Metoda rysująca menu opcji na oknie.

Parametry

window	Referencja do obiektu sf::RenderWindow, na którym rysowane jest menu opcji.

Reimplementowana z Menu.

4.5.3.2 GetPressedItem()

```
int OptionsMenu::GetPressedItem ( ) const  [override], [virtual]
```

Metoda zwracająca indeks wybranego elementu w menu opcji.

Zwraca

Indeks wybranego elementu w menu opcji.

Reimplementowana z Menu.

4.5.3.3 getSelectedResolution()

```
Resolution OptionsMenu::getSelectedResolution ( ) const
```

Metoda zwracająca wybraną rozdzielczość w menu opcji.

Zwraca

Obiekt Resolution reprezentujący wybraną rozdzielczość.

4.5.3.4 isFullscreen()

```
bool OptionsMenu::isFullscreen ( ) const
```

Metoda sprawdzająca, czy tryb pełnoekranowy jest włączony.

Zwraca

Wartość logiczna informująca, czy tryb pełnoekranowy jest włączony.

4.5.3.5 MoveDown()

```
void OptionsMenu::MoveDown ( )  [override], [virtual]
```

Metoda obsługująca ruch w dół w menu opcji.

Reimplementowana z Menu.

4.5.3.6 MoveUp()

```
void OptionsMenu::MoveUp ( )  [override], [virtual]
```

Metoda obsługująca ruch w górę w menu opcji.

Reimplementowana z Menu.

4.5.3.7 toggleFullscreen()

```
void OptionsMenu::toggleFullscreen ( )
```

Metoda przełączająca tryb pełnoekranowy.

4.5.3.8 updateSizeAndPosition()

```
void OptionsMenu::updateSizeAndPosition (
            float newWidth,
            float newHeight )
```

Metoda aktualizująca rozmiar i pozycję menu opcji po zmianie rozdzielczości okna.

Parametry

newWidth	Nowa szerokość okna.
newHeight	Nowa wysokość okna.

Dokumentacja dla tej klasy została wygenerowana z plików:

- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/optionsMenu.h
- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/optionsMenu.cpp

4.6 Dokumentacja klasy Player

`#include <player.h>`

Typy publiczne

- enum class PlayerSetupState { SETTING_NICKNAME , PLACING_SHIPS , DONE }
- enum class PlayerGameState { PLACING_SHIPS , ATTACKING , WAITING , GAME_OVER }

Metody publiczne

- Player (const std::string &id, const std::string &nickname, Board &board)
- void handleSetupEvents (const sf::Event &event, Board &board)
- void renderSetup (sf::RenderWindow &window, const sf::Font &font, const std::string &playerId)
- bool allShipsPlaced () const
- void setNickname (const std::string &newNickname)
- void setSetupState (PlayerSetupState state)
- PlayerSetupState getSetupState () const
- bool handleGameEvents (const sf::Event &event, Player &otherPlayer, int &currentPlayer)
- void renderGame (sf::RenderWindow &window, const sf::Font &font, const Player &enemyPlayer)
- void setGameState (PlayerGameState state)
- bool allShipsSunk () const
- PlayerGameState getGameState () const
- std::string getName () const
- void moveAimUp ()
- void moveAimDown ()
- void moveAimLeft ()
- void moveAimRight ()
- std::pair< int, int > getAim () const
- Board & getBoard ()

4.6.1 Dokumentacja składowych wyliczanych

4.6.1.1 PlayerGameState

`enum class Player::PlayerGameState [strong]`

Enumeracja reprezentująca różne stany gry dla gracza.

Wartości wyliczeń

PLACING_SHIPS	Gracz rozmieszcza statki na planszy.
ATTACKING	Gracz wykonuje atak na przeciwnika.
WAITING	Gracz oczekuje na ruch przeciwnika.
GAME_OVER	Gra zakończona.

4.6.1.2 PlayerSetupState

```
enum class Player::PlayerSetupState  [strong]
```

Enumeracja reprezentująca różne stany dla procesu konfiguracji gracza.

Wartości wyliczeń

SETTING_NICKNAME	Gracz ustawia swój nick.
PLACING_SHIPS	Gracz rozmieszcza statki na planszy.
DONE	Proces konfiguracji zakończony.

4.6.2 Dokumentacja konstruktora i destruktora

4.6.2.1 Player()

```
Player::Player (
            const std::string & id,
            const std::string & nickname,
            Board & board )
```

Konstruktor klasy Player.

Inicjalizuje obiekt klasy Player przy użyciu podanych parametrów i ustawia początkowe wartości zmiennych członkowskich.

Parametry

id	Unikalny identyfikator gracza.
nickname	Nick gracza.
boardRef	Referencja do obiektu klasy Board, na której gra gracz.

4.6.3 Dokumentacja funkcji składowych

4.6.3.1 allShipsPlaced()

```
bool Player::allShipsPlaced ( ) const  [inline]
```

Sprawdza, czy wszystkie statki zostały umieszczone na planszy.

Zwraca

True, jeśli wszystkie statki zostały umieszczone; w przeciwnym razie false.

4.6.3.2 allShipsSunk()

```
bool Player::allShipsSunk ( ) const
```

Sprawdza, czy wszystkie statki zostały zatopione.

Zwraca

True, jeśli wszystkie statki zostały zatopione, false w przeciwnym razie.

4.6.3.3 getAim()

```
std::pair< int, int > Player::getAim ( ) const
```

Pobiera aktualne współrzędne celownika.

Zwraca

Współrzędne celownika jako para intów.

4.6.3.4 getBoard()

```
Board & Player::getBoard ( )
```

Pobiera planszę gry.

Zwraca

Referencja do planszy gry.

4.6.3.5 getGameState()

```
Player::PlayerGameState Player::getGameState ( ) const
```

Pobiera aktualny stan gry.

Zwraca

Aktualny stan gry.

4.6.3.6 getName()

```
std::string Player::getName ( ) const
```

Pobiera nazwę gracza.

Zwraca

Nazwa gracza.

4.6.3.7 getSetupState()

```
Player::PlayerSetupState Player::getSetupState ( ) const
```

Zwraca aktualny stan konfiguracji gracza.

Zwraca

Aktualny stan konfiguracji gracza.

4.6.3.8 handleGameEvents()

```
bool Player::handleGameEvents (
            const sf::Event & event,
            Player & otherPlayer,
            int & currentPlayer )
```

Obsługuje zdarzenia związane z fazą gry atakującą.

Parametry

event	Zdarzenie SFML.
otherPlayer	Przeciwnik.
currentPlayer	Aktualny numer gracza.

Zwraca

True, jeśli tura zakończona; w przeciwnym razie false.

4.6.3.9 handleSetupEvents()

```
void Player::handleSetupEvents (
            const sf::Event & event,
            Board & board )
```

Obsługuje zdarzenia związane z konfiguracją gracza.

W zależności od aktualnego stanu konfiguracji, reaguje na zdarzenia takie jak wprowadzanie tekstu czy klawisze, umożliwiając odpowiednie ustawienia gracza.

Parametry

event	Zdarzenie SFML, które ma być obsłużone.
board	Referencja do obiektu planszy gry.

4.6.3.10 moveAimDown()

```
void Player::moveAimDown ( )
```

Przesuwa celownik w dół.

4.6.3.11 moveAimLeft()

```
void Player::moveAimLeft ( )
```

Przesuwa celownik w lewo.

4.6.3.12 moveAimRight()

```
void Player::moveAimRight ( )
```

Przesuwa celownik w prawo.

4.6.3.13 moveAimUp()

```
void Player::moveAimUp ( )
```

Przesuwa celownik w górę.

4.6.3.14 renderGame()

```
void Player::renderGame (
            sf::RenderWindow & window,
            const sf::Font & font,
            const Player & enemyPlayer )
```

Renderuje interfejs gry.

Parametry

window	Okno renderingu SFML.
font	Czcionka do użycia.
enemyPlayer	Przeciwnik.

4.6.3.15 renderSetup()

```
void Player::renderSetup (
            sf::RenderWindow & window,
            const sf::Font & font,
            const std::string & playerId )
```

Renderuje interfejs konfiguracji gracza.

W zależności od aktualnego stanu konfiguracji, renderuje odpowiednie elementy interfejsu graficznego, takie jak pole do wprowadzania nicku, informacje o rozmieszczaniu statków, planszę do rozmieszczania statków itp.

Parametry

window	Okno SFML, na którym ma być renderowany interfejs.
font	Czcionka używana do renderowania tekstu.
playerId	Identyfikator gracza.

4.6.3.16 setGameState()

```
void Player::setGameState (
            PlayerGameState state )
```

Ustawia stan gry.

Parametry

state	Nowy stan gry.

4.6.3.17 setNickname()

```
void Player::setNickname (
            const std::string & newNickname )
```

Ustawia nowy nick gracza.

Parametry

newNickname	Nowy nick gracza.

4.6.3.18 setSetupState()

```
void Player::setSetupState (
            PlayerSetupState state )
```

Ustawia nowy stan konfiguracji gracza.

Parametry

state	Nowy stan konfiguracji gracza.

Dokumentacja dla tej klasy została wygenerowana z plików:

- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/player.h
- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/player.cpp

4.7 Dokumentacja struktury Resolution

```
#include <config.h>
```

Atrybuty publiczne

- unsigned int width
- unsigned int height

4.7.1 Opis szczegółowy

Struktura reprezentująca rozdzielczość, składającą się z szerokości i wysokości.

4.7.2 Dokumentacja atrybutów składowych

4.7.2.1 height

```
unsigned int Resolution::height
```

Wysokość

4.7.2.2 width

```
unsigned int Resolution::width
```

Szerokość

Dokumentacja dla tej struktury została wygenerowana z pliku:

- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/config.h

4.8 Dokumentacja klasy Ship

```
#include <ship.h>
```

Metody publiczne

- Ship (unsigned int startX, unsigned int startY, unsigned int length, bool isVertical)

Przyjaciele

- class Board

4.8.1 Opis szczegółowy

Klasa reprezentująca statek.

4.8.2 Dokumentacja konstruktora i destruktora

4.8.2.1 Ship()

```
Ship::Ship (
            unsigned int x,
            unsigned int y,
            unsigned int len,
            bool vertical )
```

Konstruktor inicjalizujący obiekt Ship.

Parametry

startX	Początkowa pozycja X statku.
startY	Początkowa pozycja Y statku.
length	Długość statku.
isVertical	Określa, czy statek jest ustawiony pionowo (true) czy poziomo (false).

Parametry

x	Początkowa pozycja X statku.
y	Początkowa pozycja Y statku.
len	Długość statku.
vertical	Określa, czy statek jest ustawiony pionowo (true) czy poziomo (false).

4.8.3 Dokumentacja przyjaciół i funkcji związanych

4.8.3.1 Board

```
friend class Board  [friend]
```

Dokumentacja dla tej klasy została wygenerowana z plików:

- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/ship.h
- C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/ship.cpp

Rozdział 5

Dokumentacja plików

5.1 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/board.cpp

```
#include "board.h"
```

5.2 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/board.h

```
#include "config.h"
#include "Ship.h"
#include <vector>
#include <iostream>
```

Komponenty

- class Board

5.3 board.h

Idź do dokumentacji tego pliku.

```
1 #pragma once
2 #include "config.h"
3 #include "Ship.h"
4 #include <vector>
5 #include <iostream>
6
14 class Board {
15 public:
16
23     Board();
24
30     ~Board();
31
43     char getCellValue(unsigned int x, unsigned int y) const;
44
```

```
60      bool placeShip(unsigned int startX, unsigned int startY, unsigned int shipLength, bool isVertical);
61
75      bool isCellAvailable(unsigned int y, unsigned int x) const;
76
90      bool shootAt(unsigned int x, unsigned int y);
91
104      bool isShipSunk(unsigned int x, unsigned int y);
105
115      bool areAllShipsSunk();
116
124      unsigned int getWidth() const;
125
133      unsigned int getHeight() const;
134
135 private:
136      unsigned int width;
137      unsigned int height;
138      char** boardArray;
139      std::vector<Ship> ships;
140 };
141
142
```

5.4 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/config.h

Komponenty

- struct Resolution

Zmienne

- const Resolution boardSize = { 10, 10 }

5.4.1 Dokumentacja zmiennych

5.4.1.1 boardSize

```
const Resolution boardSize = { 10, 10 }
```

Stała reprezentująca rozmiar planszy do gry w statki.

Stała ta zawiera obiekt struktury Resolution określający szerokość i wysokość planszy. Domyślnie ustawiona na rozmiar 10x10.

5.5 config.h

Idź do dokumentacji tego pliku.

```
1 #pragma once
2
6 struct Resolution {
7     unsigned int width;
8     unsigned int height;
9 };
10
17 const Resolution boardSize = { 10, 10 };
```

5.6 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/endMenu.cpp

```
#include "endMenu.h"
```

5.7 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/endMenu.h

```
#include "Menu.h"
```

Komponenty

- class EndMenu

5.8 endMenu.h

Idź do dokumentacji tego pliku.

```
1 #pragma once
2 #include "Menu.h"
3
9 class EndMenu : public Menu {
10 public:
17     EndMenu(float width, float height);
18
27     void draw(sf::RenderWindow& window) override;
28
36     void setWinnerDetails(const std::string& winnerName);
37
44     void positionExitButton();
45
46 private:
47     sf::Text winnerDetails;
48     sf::Text menuItems[1];
49 };
```

5.9 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/game.cpp

```
#include "Game.h"
```

5.10 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/game.h

```
#include "menu.h"
#include "optionsMenu.h"
#include "endMenu.h"
#include "board.h"
#include "player.h"
#include <SFML/Graphics.hpp>
#include <iostream>
```

Komponenty

- class Game

Wyliczenia

- enum class GameState {
  MENU , OPTIONS , SETUP , PLAY ,
  END }

5.10.1 Dokumentacja typów wyliczanych

5.10.1.1 GameState

```
enum class GameState  [strong]
```

Stan gry.

Enumeracja określająca różne stany gry, takie jak menu, opcje, ustawianie planszy, rozgrywka i zakończenie gry.

Wartości wyliczeń

MENU	Stan menu.
OPTIONS	Stan opcji.
SETUP	Stan ustawiania planszy.
PLAY	Stan rozgrywki.
END	Stan zakończenia gry.

5.11 game.h

Idź do dokumentacji tego pliku.

```
1 #pragma once
2 #include "menu.h"
3 #include "optionsMenu.h"
4 #include "endMenu.h"
5 #include "board.h"
6 #include "player.h"
7 #include <SFML/Graphics.hpp>
8 #include <iostream>
9
15 enum class GameState {
16     MENU,
17     OPTIONS,
18     SETUP,
19     PLAY,
20     END
21 };
22
28 class Game {
29 public:
33     Game();
34
40     void run();
```

```
41
42 private:
50     void handleMenuEvents(const sf::Event& event);
51
59     void handleOptionsEvents(const sf::Event& event);
60
68     void handlePlayEvents(const sf::Event& event);
69
77     void handleEndMenuEvents(const sf::Event& event);
78
86     void handleSetupEvents(const sf::Event& event);
87
93     void processEvents();
94
100     void render();
101
107     void changeResolution();
108
114     void toggleFullscreen();
115
121     void updateUIForResolution();
122
128     void checkEndGameCondition();
129
130     sf::RenderWindow window;
131     sf::Font font;
132     GameState gameState;
133     Menu menu;
134     OptionsMenu optionsMenu;
135     EndMenu endMenu;
136     Player player1, player2;
137     Board board1, board2;
138     sf::Vector2u originalResolution;
139     int currentPlayer = 1;
140     bool settingUpPlayer1 = true;
141 };
142
```

5.12 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/main.cpp

```
#include <SFML/Graphics.hpp>
#include <iostream>
#include "Menu.h"
#include "Game.h"
```

Funkcje

- int main ()

5.12.1 Dokumentacja funkcji

5.12.1.1 main()

```
int main ( )
```

5.13 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/menu.cpp

```
#include "Menu.h"
```

5.14 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/menu.h

```
#include <iostream>
#include <SFML/Graphics.hpp>
```

Komponenty

- class Menu

5.15 menu.h

Idź do dokumentacji tego pliku.

```
1 #pragma once
2 #include <iostream>
3 #include <SFML/Graphics.hpp>
4
5 class Menu {
6 public:
13      Menu(float width, float height);
14
18      ~Menu() = default;
19
25      virtual void draw(sf::RenderWindow& window);
26
30      virtual void MoveLeft();
31
35      virtual void MoveRight();
36
40      virtual void MoveUp() {};
41
45      virtual void MoveDown() {};
46
52      virtual int GetPressedItem() const;
53
60      void updateMenuSize(float width, float height);
61      sf::Text menu[3];
62
63 private:
64      sf::Sprite logo;
65      sf::Texture logoTexture;
66
67 protected:
68      sf::Font font;
69      float screenWidth;
70      float screenHeight;
71      int selectedItemIndex;
72
76      virtual void positionMenuItems();
77 };
```

5.16 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/optionsMenu.cpp

```
#include "optionsMenu.h"
```

5.17 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/optionsMenu.h

```
#include "menu.h"
#include "config.h"
#include <SFML/Graphics.hpp>
#include <iostream>
#include <vector>
```

Komponenty

- class OptionsMenu

5.18 optionsMenu.h

Idź do dokumentacji tego pliku.

```
1 #pragma once
2 #include "menu.h"
3 #include "config.h"
4 #include <SFML/Graphics.hpp>
5 #include <iostream>
6 #include <vector>
7
8
12 class OptionsMenu : public Menu {
13 public:
20     OptionsMenu(float width, float height);
21
25     ~OptionsMenu() = default;
26
32     int GetPressedItem() const override;
33
39     void draw(sf::RenderWindow& window) override;
40
44     void MoveUp() override;
45
49     void MoveDown() override;
50
54     void toggleFullscreen();
55
62     void updateSizeAndPosition(float newWidth, float newHeight);
63
69     bool isFullscreen() const;
70
76     Resolution getSelectedResolution() const;
77
78 private:
79     sf::Font font;
80     sf::Text menuItems[5];
81     int selectedItemIndex;
82     int currentResolutionIndex;
83     float screenWidth, screenHeight;
84     bool isFullscreenEnabled;
85
89     void positionMenuItems();
90     std::vector<Resolution> resolutions = { {800, 600}, {1280, 720},{1920, 1080} };
91
92 };
```

5.19 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/player.cpp

```
#include "Player.h"
#include "game.h"
```

5.20 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/player.h

```
#include <SFML/Graphics.hpp>
#include <iostream>
#include <string>
#include <vector>
#include "Board.h"
```

Komponenty

- class Player

5.21 player.h

Idź do dokumentacji tego pliku.

```
1 #pragma once
2 #include <SFML/Graphics.hpp>
3 #include <iostream>
4 #include <string>
5 #include <vector>
6 #include "Board.h"
7
8 class Player {
9 public:
19     Player(const std::string& id, const std::string& nickname, Board& board);
20
24     enum class PlayerSetupState {
25         SETTING_NICKNAME,
26         PLACING_SHIPS,
27         DONE
28     };
29
33     enum class PlayerGameState {
34         PLACING_SHIPS,
35         ATTACKING,
36         WAITING,
37         GAME_OVER
38     };
39
40      //metody do setupu
41
51     void handleSetupEvents(const sf::Event& event, Board& board);
52
63     void renderSetup(sf::RenderWindow& window, const sf::Font& font, const std::string& playerId);
64
65
71     bool allShipsPlaced()const { return shipsPlaced >= MAX_SHIPS; }
72
78     void setNickname(const std::string& newNickname);
79
86     void setSetupState(PlayerSetupState state);
92     PlayerSetupState getSetupState() const;
93
94      //metody do gry
103     bool handleGameEvents(const sf::Event& event, Player& otherPlayer, int& currentPlayer);
104
112     void renderGame(sf::RenderWindow& window, const sf::Font& font, const Player& enemyPlayer);
113
119     void setGameState(PlayerGameState state);
120
126     bool allShipsSunk() const;
127
133     PlayerGameState getGameState() const;
134
140     std::string getName() const;
141
142     //metody do celowania
146     void moveAimUp();
147
```

```
151     void moveAimDown();
152
156     void moveAimLeft();
157
161     void moveAimRight();
162
168     std::pair<int, int> getAim() const;
169
175     Board& getBoard();
176
177 private:
178      //czesc do setupu
179      Board& board;
180      PlayerSetupState setupState;
181      std::string nickname;
182      std::string playerId;
183      unsigned int startX, startY;
184      unsigned int shipsPlaced;
185      static const unsigned int MAX_SHIPS = 5;
186      std::vector<unsigned int> shipLengths;
187      bool isVertical;
189      //czesc do gry
190      PlayerGameState gamestate;
191      unsigned int attackX, attackY;
192      int aimX, aimY;
195      //do renderingu
204      sf::Vector2f calculatePosition(sf::RenderWindow& window, float width, float height);
205 };
206
207
```

5.22 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/ship.cpp

```
#include "Ship.h"
```

5.23 Dokumentacja pliku C:/Users/Barte/Desktop/GitHub/9c491b86-gr01-repo/Projekt/Statki/ship.h

Komponenty

- class Ship

5.24 ship.h

Idź do dokumentacji tego pliku.

```
1 #pragma once
2
6 class Ship {
7 public:
16      Ship(unsigned int startX, unsigned int startY, unsigned int length, bool isVertical);
17
18      //do dostepu do elementow planszy
19      friend class Board;
20
21 private:
25      unsigned int startX;
26
30      unsigned int startY;
31
35      unsigned int length;
36
40      bool isVertical;
41
45      unsigned int hits;
46 };
```