**Sprawozdanie Projektowe**

***Przedmiot:***

* ***Programowanie w języku Java (projekt)***

***Temat:***

* ***Gra Statki***

***Zespół:***

* ***Dominik Grudzień***
* ***Patryk Grzywacz***
* ***Adrian Pełka***

***Informacje:***

* ***Rok: 2***
* ***Kierunek: Informatyka***
* ***Grupa: 2ID12A***

**Szczegółowy Opis Projektu**

***Język:***

* **Obiektowa JavaSE-15**

***Technologia:***

* **Silnik LIBGDX w wersji 1.9.14 na podstawie frameworka LWJGL**
* **Framework JUNIT5 w wersji 5.6.0 do testu jednostkowego**
* **GRADLE Project Manager w wersji 6.7.1**
* **Do tworzenia użyto IDE: VSCODE z rozszerzeniami umożliwiającymi pisanie projektów w Javie, kontrolowanie zdalnym repozytorium GitHub i innymi kluczowymi technologiami dla projektu .**

***Biblioteki:***

* **Standardowe biblioteki języka Java wersji 15**
* **Biblioteki LIBGDX – do tworzenia aplikacji desktopowej wraz z całą zawartością.**
* **Biblioteki Junit Jupiter oraz Junit Platform – do przeprowadzenia testu jednostkowego.**
* **Własne napisane biblioteki jak np. do prowadzenia rozgrywki, kontrolowania AI przeciwnika czy też manipulowania obiektami gry.**

***Funkcjonalności:***

* **Możliwość samodzielnego rozmieszczenia na planszy statków różnych wielkości i ich obrotu, lub też skorzystanie z przycisku do automatyzacji tych działań .**
* **Rozgrywanie gry w statki wersji papierowej wg. klasycznych zasad przeniesione w system graficzny 2D.**
* **Prowadzenie bitwy w otoczeniu specjalnych efektów graficznych i audio na żywo z wymagającym przeciwnikiem komputerowym.**
* **Bonusy w postaci Combo ,które zwiększa w czasie trwania nabijane punkty za trafienia i zniszczenia.**
* **Możliwość zapisu i odczytu wyników najlepszych graczy.**
* **Interaktywne menu gry za pomocą myszki.**
* **Klimatyczna muzyka.**

***Uruchamianie oraz obsługa projektu:***

* **Na systemie Windows:**
  + **Proste uruchomienie pliku ShipsGameV\_1.2.exe**
  + **Lub poprzez uruchomienie ShipsDist.jar poprzez terminal komendą java -jar ShipsDist.jar**
  + **Ewentualnie można też stworzyć projekt w dowolnym IDE poprzez Gradle i przekopiować cały folder projektu z plikami źródłowymi, assetami , ustawieniami Gradle itd. do właściwego folderu nowego projektu i skompilowanie go.**
* **Teoretycznie na innych systemach typu Linux:**
  + **Poprzez uruchomienie ShipsDist.jar poprzez terminal komendą java -jar ShipsDist.jar**

***Informacje na temat klas oraz metod:***

* **Klasa DesktopLauncher – Główna klasa programu.**
  + **Statyczna metoda main() – Główna metoda odpowiedzialna za połączenie silnika z projektem Java i utworzenie aplikacji.**
* **Klasa Main – Główna klasa operująca oknami gry.**
  + **Metoda create() – Metoda odpowiedzialna za stworzenie okna gry i ustawienie ekranu na Menu Główne.**
  + **Metoda dispose() – Metoda odpowiedzialna za niszczenie elementów silnika libgdx.**
* **Klasa MenuScreen – Klasa zawierająca główne menu gry.**
  + **MenuScreen(Main) – Konstruktor obiektu klasy MenuScreen.**
  + **Metoda update(float) – Metoda odpowiedzialna za odświeżanie operacji i wyglądu w menu gry.**
  + **Metoda render(float) – Metoda odpowiedzialna za renderowanie okna menu gry.**
  + **Metoda show() – Metoda odpowiedzialna za tworzenie, ustawianie i ładowanie elementów w głównym menu gry.**
  + **Metoda pause() – Metoda obsługująca pauzę aktywności okna w menu.**
  + **Metoda resume() – Metoda obsługująca wznowienie aktywności okna w menu.**
  + **Metoda resize(int , int) – Metoda obsługująca zmianę rozmiaru okna w menu.**
  + **Metoda hide() - Metoda obsługująca ukrycie okna w menu.**
  + **Metoda dispose() - Metoda obsługująca niszczenie elementów silnika libgdx oraz menu głównego.**
* **Klasa MenuGlobalElements – Klasa zawierająca metody i pola wykorzystywane przez okna menu.**
  + **MenuGlobalElements(Game) – Konstruktor obiektu klasy tworzący assety menu i ustawiający inne atrybuty menu.**
  + **Metoda moveMenu(float) – Metoda odpowiedzialna za animację tła w menu gry.**
  + **Metoda disposeMenu() – Metoda odpowiedzialna za niszczenie elementów wykorzystywanych przez menu.**
* **Klasa HelpScreen – Klasa okna pomocy w głównym menu.**
  + **HelpScreen(Main) – Konstruktor obiektu okna.**
  + **Metoda update(float) – Metoda odpowiedzialna za odświeżanie operacji i wyglądu w oknie pomocy menu.**
  + **Metoda render(float) – Metoda odpowiedzialna za renderowanie elementów okna pomocy.**
  + **Metoda show() – Metoda odpowiedzialna za tworzenie, ustawianie i ładowanie elementów w oknie pomocy menu.**
  + **Metoda pause() – Metoda obsługująca pauzę aktywności okna w oknie pomocy menu.**
  + **Metoda resume() – Metoda obsługująca wznowienie aktywności okna w oknie pomocy menu.**
  + **Metoda resize(int , int) – Metoda obsługująca zmianę rozmiaru okna w oknie pomocy menu.**
  + **Metoda hide() - Metoda obsługująca ukrycie okna w oknie pomocy menu.**
  + **Metoda dispose() - Metoda obsługująca niszczenie elementów silnika libgdx oraz okna pomocy menu.**
* **Klasa OptionScreen – Klasa opcji w menu gry.**
  + **OptionScreen(Main) – Konstruktor obiektu okna.**
  + **Metoda update(float) – Metoda odpowiedzialna za odświeżanie operacji i wyglądu w oknie opcji menu.**
  + **Metoda render(float) – Metoda odpowiedzialna za renderowanie elementów okna opcji.**
  + **Metoda show() – Metoda odpowiedzialna za tworzenie, ustawianie i ładowanie elementów w oknie opcji menu.**
  + **Metoda pause() – Metoda obsługująca pauzę aktywności okna w oknie opcji menu.**
  + **Metoda resume() – Metoda obsługująca wznowienie aktywności okna w oknie opcji menu.**
  + **Metoda resize(int , int) – Metoda obsługująca zmianę rozmiaru okna w oknie opcji menu.**
  + **Metoda hide() - Metoda obsługująca ukrycie okna w oknie opcji menu.**
  + **Metoda dispose() - Metoda obsługująca niszczenie elementów silnika libgdx oraz okna opcji menu.**
  + **Metoda saveSettings() – Metoda odpowiedzialna za zapis ustawień z okna opcji do pliku tekstowego settings.txt wykorzystywanego w całej aplikacji.**
* **Klasa ScoreScreen – Klasa okna wyników w menu.**
  + **ScoreScreen(Main) – Konstruktor obiektu okna oraz obiektu klasy Scores do wyświetlania wyników.**
  + **Metoda update(float) – Metoda odpowiedzialna za odświeżanie operacji i wyglądu w oknie wyników menu.**
  + **Metoda render(float) – Metoda odpowiedzialna za renderowanie elementów okna wyników.**
  + **Metoda show() – Metoda odpowiedzialna za tworzenie, ustawianie i ładowanie elementów w oknie wyników menu.**
  + **Metoda pause() – Metoda obsługująca pauzę aktywności okna w oknie wyników menu.**
  + **Metoda resume() – Metoda obsługująca wznowienie aktywności okna w oknie wyników menu.**
  + **Metoda resize(int , int) – Metoda obsługująca zmianę rozmiaru okna w oknie wyników menu.**
  + **Metoda hide() - Metoda obsługująca ukrycie okna w oknie wyników menu.**
  + **Metoda dispose() - Metoda obsługująca niszczenie elementów silnika libgdx oraz okna wyników menu.**
* **Klasa Scores – Klasa, której rolą jest tworzenie pliku scores.txt oraz przechowywanie wszystkich wyników.**
  + **Klasa Node – Klasa zagnieżdzona przechowująca jeden rekord wyników**
    - **Node(String , float , float , float) – Konstruktor obiektu rekordu z danymi przekazanymi mu przez parametr.**
  + **Klasa SortByScore – Klasa zagnieżdzona pozwalająca przeciążyć komparator do listy typu ArrayList.**
    - **Metoda compare(Node , Node) – Przeciążona metoda z Comparatora ,która pozwala sortować klasie Scores rekordy wg. wyników.**
  + **Scores() – Konstruktor obiektu klasy Scores, tworzący listę typu ArrayList i wypełniający ją rekordami.**
  + **Metoda loadScores() – Metoda ta tworzy plik z wynikami jeśli nie istnieje lub pobiera dane z pliku i umieszcza je w rekordach listy a następnie sortuje listę.**
  + **Metoda drawScores(SpriteBatch , BitmapFont , float , float) – Metoda odpowiedzialna za rysowanie rekordów z listy na ekranie.**
* **Klasa Animator…**
* **Klasa BoomEffect – Klasa odpowiedzialna za tworzenie efektów trafień lub nietrafień oraz zniszczeń.**
  + **BoomEffect(Sound , Texture) – Konstruktor tworzący efekt trafienia.**
  + **BoomEffect(Sound , Texture , Vector2 , float) – Konstruktor tworzący efekt nietrafienia.**
  + **BoomEffect(Sound , Texture , boolean) – Konstruktor tworzący efekt zniszczenia.**
  + **Metoda getPos() – Metoda zwracająca pozycję efektu.**
  + **Metoda setPos(Vector2f) – Metoda ustawiająca pozycję efektu.**
  + **Metoda setPos(Vector2f , int , int) – Metoda tworząca i ustawiająca wiele efektów zniszczeń.**
  + **Metoda updateAnimation() – Metoda do aktualizacji animacji efektu.**
  + **Metoda updateAnimation(boolean) – Metoda do aktualizacji animacji efektów znisczeń.**
  + **Metoda drawEffect(SpriteBatch) – Metoda do rysowania efektu na ekranie.**
  + **Metoda drawEffect(SpriteBatch , boolean) – Metoda do rysowania efektów zniszczeń na ekranie.**
  + **Metoda playSound(float) – Metoda do włączania dźwięku efektu.**
  + **Metoda playSound(boolean , float) – Metoda do włączania dźwięku efektów znisczeń.**
  + **Metoda resetAnimation() – Metoda do resetowania animacji efektu/efektów do stanu początkowego.**
* **Klasa ShootParticleEffect – Klasa przechowująca wszystkie efekty strzałów pojedynczego statku.**
  + **ShootParticleEffect(Texture , float , float , Vector2 , int) – Konstruktor obiektu**
  + **Metoda setPositions(GameObject) – Zadaniem tej metody jest poprawne ulokowanie efektów w zależności od pozycji i rotacji statku i wieżyczek.**
  + **Metoda updateAnimation(GameObject) – Metoda do aktualizowania animacji wszystkich efektów.**
  + **Metoda drawAnimation(SpriteBatch) – Metoda do rysowania efektów na ekranie.**
  + **Metoda resetAnimation() – Metoda do resetowania animacji do stanu początkowego.**
* **Interface Constant – Interfejs przechowujący kilka stałych do obliczeń i ustawiania logiki gry.**
* **Klasa Score – Klasa przechowująca wyniki gracza i komputera w czasie bitwy**
  + **Score(int) – Konstruktor obiektu klasy Score nadający identyfikator .**
  + **Metoda setPlayerName(String) – Metoda nadająca nazwę do obiektu.**
  + **Metoda drawInfo(BitmapFont , SpriteBatch , float , float, int , int , int , Texture[]) – Metoda do rysowania na ekranie elementów informacyjnych z obiektu.**
  + **Metoda update(int[][]) – Metoda do aktualizacji danych odnośnie strzałów.**
  + **Metoda updateTime(float) – Metoda aktualizująca czas trwania tur właściciela obiektu.**
  + **Metoda addPointsForHit() – Metoda przyznająca punkty za trafienie.**
  + **Metoda addPointsForDestroy(int) – Metoda przyznająca punkty za zniszczenie statku.**
  + **Metoda increaseCombo() – Metoda zwiększająca combo za każde trafienie pod rząd.**
  + **Metoda zeroCombo() – Metoda zerująca combo.**
  + **Metoda getPlayerName() – Metoda zwracająca nazwę właściciela obiektu.**
  + **Metoda getTimeElapsed() – Metoda zwracająca czas trwania tur.**
  + **Metoda getAccuracyRatio() – Metoda zwracająca celność strzałów.**
  + **Metoda getCombo() – Metoda zwracająca combo.**
  + **Metoda getShipsDestroyed() - Metoda zwracająca ilość zniszczonych statków.**
  + **Metoda getShotsMissed() – Metoda zwracająca ilość chybionych strzałów.**
  + **Metoda getIdNumber() – Metoda zwracająca identyfikator.**