Projekt z Programowania Obiektowego (Java) Politechnika Świętokrzyska Wydział Elektrotechniki, Automatyki i Informatyki	
Temat: **Gra w statki**	Członkowie zespołu: **Dominik Niewadził**
Rok studiów: 2	Grupa dziekańska: 2ID12B

1) Opis projektu wraz informacjami o technologiach, bibliotekach użytych w projekcie

Projekt gry w statki jest zbudowany z 2 katalogów gdzie w każdym katalogu znajduje się 6 plików typu Java. Jest on napisany w środowisku programistycznym Eclipse. Do utworzenia gry skorzystałem z następujących pakietów:
1) java.io.*
2) java.util.LinkedList
3) java.awt.BasicStroke
4) java.awt.Color
Powyższe pakiety zostały wykorzystane do plików w pierwszym katalogu o nazwie „game”.
1) java.net.Socket
2) java.io.*;
3) java.util.Iterator
4) java.io.IOException
Te pakiety zostały wykorzystane do plików w drugim katalogu o nazwie „network”.

2) Funkcjonalności projektu

Gra w statki jest grą, która ma umilić czas wolny użytkownika, ale również może ana rozwijać zmysł strategiczny. Planowanie rozmieszczenia statków na własnej planszy i rozpracowanie pozycji statków przeciwnika wymaga od gracza inteligentnego myślenia ale też i zdania się na własną intuicje.

3) Uruchomienie oraz obsługi projektu

Aby projekt został uruchomiony należy odpalić plik z rozszerzeniem JAR. Wówczas pojawi ekran gry, gdzie gracz będzie miał wybór rodzaju gry.

4) informacje na temat stworzonych klas, metod, funkcji z opisem ich podstawowej funkcjonalności

a) ElementyPlanszy - prosta klasa mająca na celu opis wszystkich możliwości pól na planszy gry;
b) Konfiguracja - klasa odpowiadająca za konfiguracje gry przez serwer. Za każdym razem gdy użytkownik zdecyduje się na grę przez serwer. Klasa nadaje nazwę hostowi oraz przydziela numer do portu i zapisuje to do pliku „statki.ini" gdzie jest to odczytywane.
c) Plansza - zaimplementowana na bazie tablicy 15x15. Jest ona wyzerowana i przypisano jej nazwę PUSTE_POLE. Następna część kodu ma za zadanie ustawienie statków tak aby nie wychodziły one za plansze oraz aby się ze sobą nie stykały. Kolejny fragment dotyczy strzału i jego wyniku. Po oddaniu strzału wybrane pole jest odpowiednio zaznaczane: PUDŁO lub TRAFIONY. W przypadku tego drugiego sprawdzane jest czy statek został zatopiony. Została tu też zaimplementowana możliwość korzystania z myszki. Każdy ze stanów znajdujący się na planszy dostał własny kolor.
d) RodzajeGraczy – kolejna prosta klasa mająca na celu przedstawienie uczestników gry.
e) Statki- jedna z najważniejszych klas projektu. Pierwsze linijki kodu tworzą przyciski i obszary do których użytkownik będzie miał dostęp. Następna funkcja odpowiada za wysyłanie wiadomości pomiędzy graczami. Kolejna funkcja tworzy informacje o ilości statków użytkownika i przeciwnika oraz o ilości statków zatopionych. Następna funkcja ma za zadanie wczytanie wysłanej wiadomości pomiędzy graczami. Kolejna funkcja dotyczy obszaru gdzie można wysyłać wiadomości. Wskazuje jej rozmiary i położenie. Następna funkcja ma za zadanie odebranie wiadomości wysłanej przez któregoś z graczy. Kolejna funkcja ma za zadanie wyświetlić tekst pod planszą jako wskazówkę co gracz musi teraz zrobić aby rozpocząć grę. Następna funkcja odpowiada za inicjacje nowej gry. Następne dwie funkcje inicjują klienta oraz serwer a trzecia ustala adres. Następnie są ustalane szczegóły połączenia czyli gdzie przycisk będzie się znajdował oraz odpowiednie komunikaty. Kolejnym elementem jest przycisk startu odpowiadający za rozpoczęcie gry. Po zakończonej partii plansza jest resetowana i potem można ponownie ustawiać statki. Po zakończeniu „meczu" połączenie jest zerowane. Kolejne funkcje odpowiadają za położenie plansz gracza i przeciwnika oraz przycisku do zmiany położenia statków.
f) WynikStrzału- prosta klasa zawierająca wynik oddania strzału przez gracza i przeciwnika
g)Client- klasa ma na celu utworzenie klienta do „walki" z innym użytkownikiem
h)Connection- funkcja odpowiada za utworzenie połączenia pomiędzy użytkownikami
i)GameEvent- klasa ma za zadanie odnotować wszystkie zdarzenia jakie miały miejsce podczas działania programu
j)Serwer- klasa odpowiadająca za prawidłowe funkcjonowanie serwera. Dba ona aby serwer nie zakłócał pracy programu.
k)WaitForClients- klasa odpowiada za oczekiwanie na połączenie gracza z klientem