POLITECHNIKA ŚWIĘTOKRZYSKA

Programowanie Obiektowe (JAVA) – Projekt

Grupa dziekańska: 2ID15B

Temat projektu:	Autorzy:
Kółko i krzyżyk – wersja sieciowa	Piotr Jaśkiewicz Kamil Żak

1.Ogólny opis projektu

Nasz projekt został napisany w języku Java z wykorzystaniem biblioteki definiowania graficznego Swing.

W projekcie wykorzystaliśmy biblioteki BufferedImage, DataInputStream, DataOutputStream, IOException, InetAddress, ServerSocket, Socket. Poszczególne biblioteki:

Java BufferedImage klasa jest podklasą klasy obrazu. Jest on używany do przenoszenia i manipulowania danymi obrazu.

DataInputStream / DataOutputStream:
Z pomocą w przetwarzaniu danych binarnych przychodzą nam klasy DataInputStream/DataOutputStream. Udostępniają one metody, dzięki którym nie musimy w naszym programie operować jedynie na danych, którymi posługuje się komputer, ale także na typach prostych takich, jak int, double itp.

IOException jest wyjątkiem z grupy wyjątków osbługiwanych (checked exceptions), co oznacza, że dziedziczy bezpośredno po klasie *Exception* i mamy obowiązek jego obsługi, gdy jakaś metoda deklaruje go w swojej sygnaturze.

InetAddress Klasa reprezentuje adres IP. Klasa ta nie definiuje publicznych konstruktorów, a obiekty tworzone s ˛a jako wynik wywołania metod statycznych (metod klasy).

ServerSocket – klasa reprezentująca gniazdo oczekujące na przychodzące żądania połączeń.

Socket – klasa reprezentująca gniazdo służące do nawiązywania połączenia, wysyłania I odbierania danych,

2.Funkcjonalnosc projektu

Projekt napisany przez nas to nic innego jak uatrakcyjniona wersja klasycznej gry w kółko i krzyżyk. Zamiast standardowego kółka i krzyżyka mamy tutaj logo Batmana i Jokera. Podczas uruchamiania podajemy adres IP i port który chcemy wykorzystać, w przypadku gdy nie ma utworzonego serwera pod podanymi parametrami, aplikacja tworzy serwer i przechodzi do nasłuchu. Gdy serwer jest już utworzony, klient łączy się z serwerem i rozpoczyna się rozgrywka. Gra kończy się gdy ktoś z grających ułoży ze swoich symboli linie prostą składającą się z 3 symboli.

3. Informacje na temat stworzonych klas i metod wraz z opisem

Podstawową klasa jest klasa publiczna TicTacToe, w której zaimplementowane są metody i zmienne.
Dodatkowo wykorzystaliśmy tutaj klasę Painter odpowiadającą za rysowanie interfejsu graficznego.

W klasie TicTacToe po deklaracji zmiennych napisany jest konstruktor TicTacToe, który odwołuje się do prepareConField(); i showLogin();. Metoda prepareConField(); odpowiada za stworzenie okna służącego do połączenia, a showLogin(); wypełnia to okno odpowiednimi

polami na IP a także numer portu. Całość jest opatrzona odpowiednimi informacjami, a także posiada button „połącz”, po wciśnięciu którego wywoływana jest metoda startTic();.

startTic(); to metoda która na początku wywołuje metodę loadImages(); a następnie tworzy obiekt painter. W tej metodzie sprawdzane jest również czy connect zwraca wartość prawdziwą, a w przeciwnym wypadku wywoływana jest metoda initializeServer();. Następnie tworzone jest okno z właściwą grą i nowy wątek, który jest rozpoczynany.

Kolejna metoda to run(); która wciąż wywołuje metodę tick(); a także rysuje od nowa painter. W tej metodzie zawarta jest także if w petli while, który sprawdza czy circle i accepted zwracają false, po czym wywołują metode listenForServerRequest();.

Następna metoda to render(), która przyjmuje obiekt Graphics g i rysuje tło, plansze, a także zależnie od tego czy gramy jako Batman czy Joker, odpowiednie logo po prawo informujące nas o tym jaką postacią gramy. Dalej w zależności od kto wykonał ruch, rysuje odpowiednie symbole na planszy. Sprawdza także czy ktoś już wygrał i jeżeli ten if zwraca prawdę rysuje odpowiednio linię, która informuje nas które pola wypełnione symbolami wygrywają. Następnie w tej metodzie w zależności czy ktoś już wygrał, może zostać wypisany komunikat o wygranie przez nas czy też przeciwnika. W przypadku remisu, odpowiedni warunek jest sprawdzany i wypisywany na planszy jest odpowiedni komunikat.

Następna metoda tick(); wywoływana w run(); sprawdza czy możemy się komunikować z przeciwnikiem. Po czym w if’ie sprawdza czy to nie jest nasz ruch i czy komunikacja jest możliwa i odczytuje z zmiennej dis pozycje do zapisania na planszy. Po wszystkim ustawia zmienna yourTurn na True, co oznacza, że to nasz ruch.

Kolejne metody to checkForWin();, checkForEnemyWin(); i checkForTie(); sprawdzające kolejno czy wygraliśmy, czy wygrał przeciwnik i czy doszło do remisu. Metody te ustawiają odpowiednie zmienne które wykorzystywane są w innych fragmentach programu(won, enemyWon, tie).

listenForServerRequest(); ustawia socket na wartość null. Po czym w try catch socket ustawiany jest na zaakceptowanie serverSocket. W kolejnym kroku dis i dos przyjmują wartości z strumieni, odpowiednio DataInputStream i DataOutputStream, a zmienna accepted ustawiania jest na wartość logiczna true.

Kolejna metoda to connect(); odpowiadająca za połączenie na wcześniej podanych ip i porcie.

initializeServer(); odpowiada za stworzenie serwera i ustawienie yourTurn na true, a circle na false.

Następnie mamy metode loadImages(); ładującą do zmiennych wykorzystywane w obrazie pliki .png

Metoda main ustawia nam wygląd okna na systemowy, a także tworzy nowy obiekt TicTacToe z klasy TicTacToe.

Następna metoda to painter(); ustawia kolor tła i wywołuje metody addMouseListener odpowiadającą za obsługę myszki, setFocusable ustawia możliwość posiadania focusa przez komponent, a requestFocus ustawia ten focus.

paintComponent przyjmuje Graphics g i wywołuje metodę render.

mouseClicked sprawdza który z graczy może wykonywać aktualnie ruch, próbuje zapisać wybraną pozycje na planszy i sprawdza czy ktoś wygrał lub czy doszło do remisu.

4.Praca włożona w tworzenie projektu

Napisany przez nas projekt powstawał wspólnie. Trudno oszacować ilość pracy włożonej przez każdego z członków zespołu indywidualnie, gdyż z założenia projekt miał być projektem zespołowym, i dążyliśmy do tego aby praca każdego z nas była porównywalna. Podczas powstawania projektu każdy z nas miał jakieś pomysły a także wkład w pisanie kodu.