Grupa ćwicz.	*Grupa lab.*	*Zespół.*	*Data wykonania.*	*Data odbioru*
Nr ćwicz./ wersja	*Temat ćwiczenia.* Gra Kółko i Krzyżyk – wersja klient/serwer			
Imiona i nazwiska. Patryk Kurzeja				*Ocena i uwagi*

1. Opis projektu :

*Temat : Gra Kółko i Krzyżyk – wersja klient/serwer
*Język programowania : Java
*Środowisko : IntelliJ IDEA 2016

Wykonany projekt to gra w Kółko i Krzyżyk działająca na dwóch komputerach w sieci lokalnej. Gra napisana jest w Javie, która posiada przydatne biblioteki służące np. do rysowania czy połączeń klient-serwer.

2. Kompilacja i uruchomienie :

Aby zagrać dwie osoby muszą uruchomić ten sam plik z grą : **tictactoe.jar**
Dla ułatwienia stworzyłem plik **run.bat**, który odpowiada za wydanie poleceń :
java -jar tictactoe.jar co w efekcie uruchamia grę wraz z konsolą. W konsoli pojawiają się tylko dodatkowe komunikaty o tym, że przeciwnik wykonał swój ruch.

JAR (ang. Java archive) – archiwum w formacie ZIP używane do strukturalizacji i kompresji plików klas języka Java oraz powiązanych z nimi metadanych.
Archiwum JAR składa się z pliku manifestu umieszczonego w ścieżce META-INF/MANIFEST.MF, który informuje o sposobie użycia i przeznaczeniu archiwum. Archiwum JAR, o ile posiada wyszczególnioną klasę główną, może stanowić osobną aplikację.

Osoba, która uruchomi grę jako pierwsza staję się serwerem. Podczas uruchomienia musi ustalić port, na którym będzie toczyć się rozgrywka. Druga osoba podczas uruchomienia podaje adres IP osoby-serwera oraz ustalony wcześniej port. Jeżeli dane zostały poprawnie wprowadzone i połączenie zostało nawiązane to ukazuje się okno gry i gracze mogą kontynuować rozgrywkę.

Gra została napisana w środowisku IntelliJ IDEA 2016 w języku Java.

Kompilacja od zera :

*Wymagania do kompilacji : Środowisko do programowania w języku Java, preferuję IntelliJ Idea.

1. Stworzenie nowego projektu Java w IDE
2. Dodanie nowego pakietu w folderze src i nazwanie go : **TicTacToe**
3. Dodanie pliku klasy Java : **TicTacToe.java** i napisanie kodu
4. Umieszczenie folderu res, w którym zawarte są obrazki w formacie .png potrzebne do działania gry (tablica, kółko, krzyżyk)
5. Konfiguracja struktury projektu w celu utworzenia artefaktów (gotowego pliku .jar do uruchomienia gry)
6. Zbudowanie projektu
7. Umieszczenie pliku wsadowego run.bat w jednym folderze z plikiem **tictactoe.jar**

```
import java.awt.*;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.awt.image.BufferedImage;
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.IOException;
import java.net.InetAddress;
import java.net.ServerSocket;
import java.net.Socket;
import javax.imageio.ImageIO;
import javax.swing.*;
```

Użyte biblioteki

Java.awt - AWT (The Abstract Window Toolkit) – jedna ze standardowych bibliotek graficznych środowiska Java. Jest zależna od stosowanego systemu operacyjnego. Obecnie obok innych bibliotek graficznych: Swing oraz Java2D jest częścią frameworka Java Foundation Classes (JFC).

Java.io.DataInputStream / DataOutputStream – Klasy, które udostępniają nam 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.

Java.io.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.

Java.net - jest pakietem zawierającym klasy służące do realizacji operacji wejścia / wyjścia w sieci Internet na niskim poziomie.

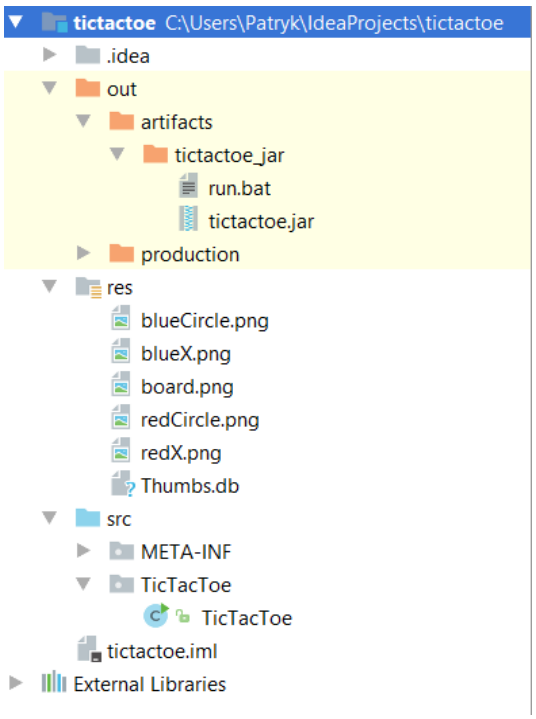
Javax.imageio.ImageIO - klasa do obsługi plików graficznych

Javax.swing - biblioteka graficzna używana w języku programowania Java, upubliczniona w lipcu roku 1997. Jest nowszą, ulepszoną wersją biblioteki AWT.

```
public TicTacToe() -> inicjalizacja połączenia i interfejsu graficznego
{...}

public void run() -> obsługa przebiegu gry
{...}

private void rendering(Graphics g) -> rysowanie interfejsu
{...}

private void checkForWin() -> sprawdza czy wygraliśmy
{...}

private void checkForOpponentWin() -> sprawdza czy wygrał przeciwnik
{...}

private void checkForDraw() -> sprawdza czy remis
{...}

private void checkForErrors() -> sprawdza czy są błędy połączenia
{...}

private void waitForServerRequest() -> obsługa połączenia
{...}

private boolean connect() -> obsługa socketu
{...}

private void startServer() -> inicjalizacja serwera
{...}

private void loadTextures() -> wczytywanie tekstur
{...}

/unused/
public static void main(String[] args) { TicTacToe ticTacToe = new TicTacToe(); }

private class Painter extends JPanel implements MouseListener {...} -> obsługa myszy
```

Funkcje odpowiedzialne za poprawne działanie gry

Gra składa się z wielu funkcji, które pozwalają na jej poprawne działanie.

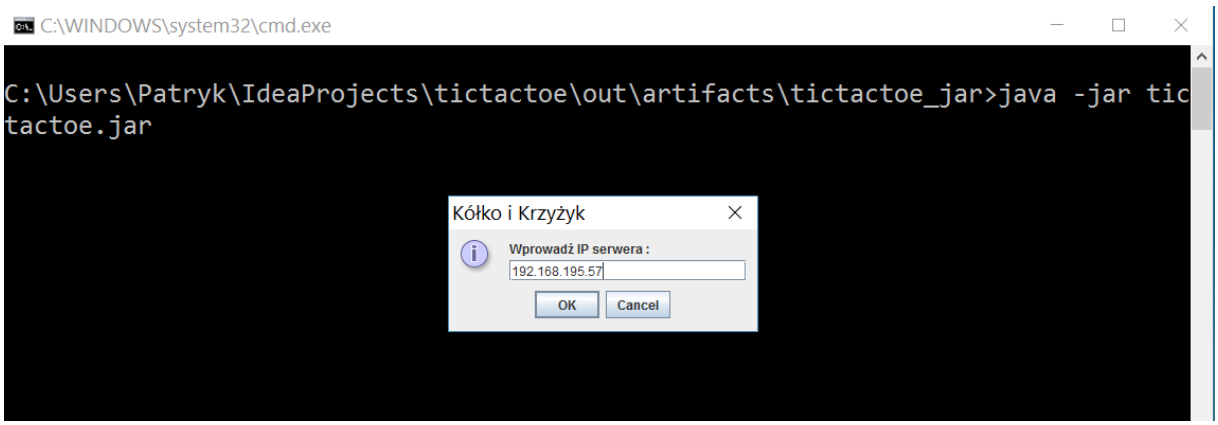


Struktura projektu

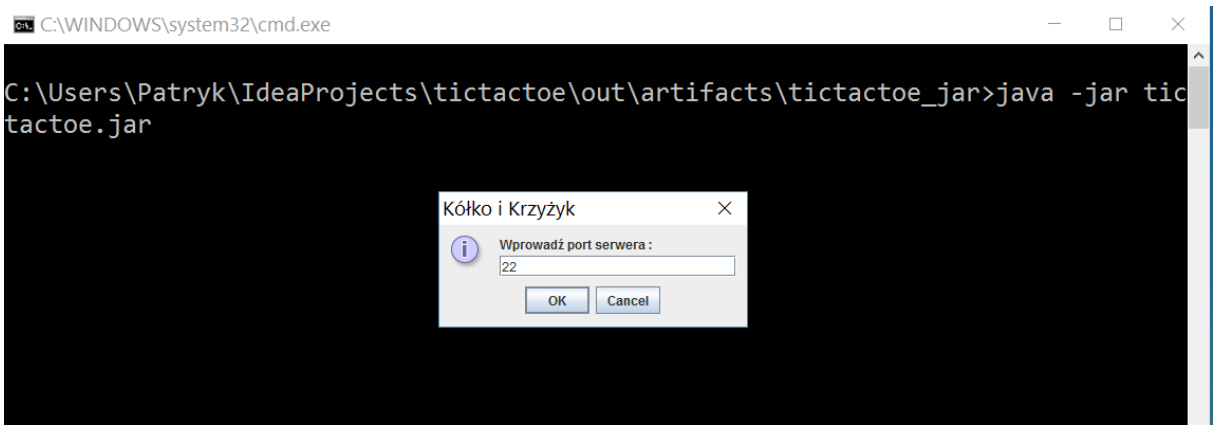
3. Opis działania programu :

*Uruchomianie wymaga jedynie plików **tictactoe.jar** oraz **run.bat**.
*Wymagania do uruchomienia : Zainstalowana Java na komputerze - serwer oraz na komputerze - klient

1. Uruchomienie pliku **run.bat (wersja z konsolą)** lub **tictactoe.jar**
2. Osoba która będzie serwerem uruchamia grę pierwsza
3. Zostawia ona pole do wprowadzania IP puste
4. W następnym kroku wprowadza port
5. Pojawia się okno do gry z napisem „Oczekiwanie na gracza”
6. Druga osoba uruchamia grę w taki sam sposób
7. Wprowadza IP osoby będącej serwerem
8. Również wprowadza jej port
9. Gra rozpoczyna się
10. Gracze wykonują ruchy naprzemiennie
11. Po skończonej grze należy ją wyłączyć
12. W celu ponownej gry załączamy ją ponownie

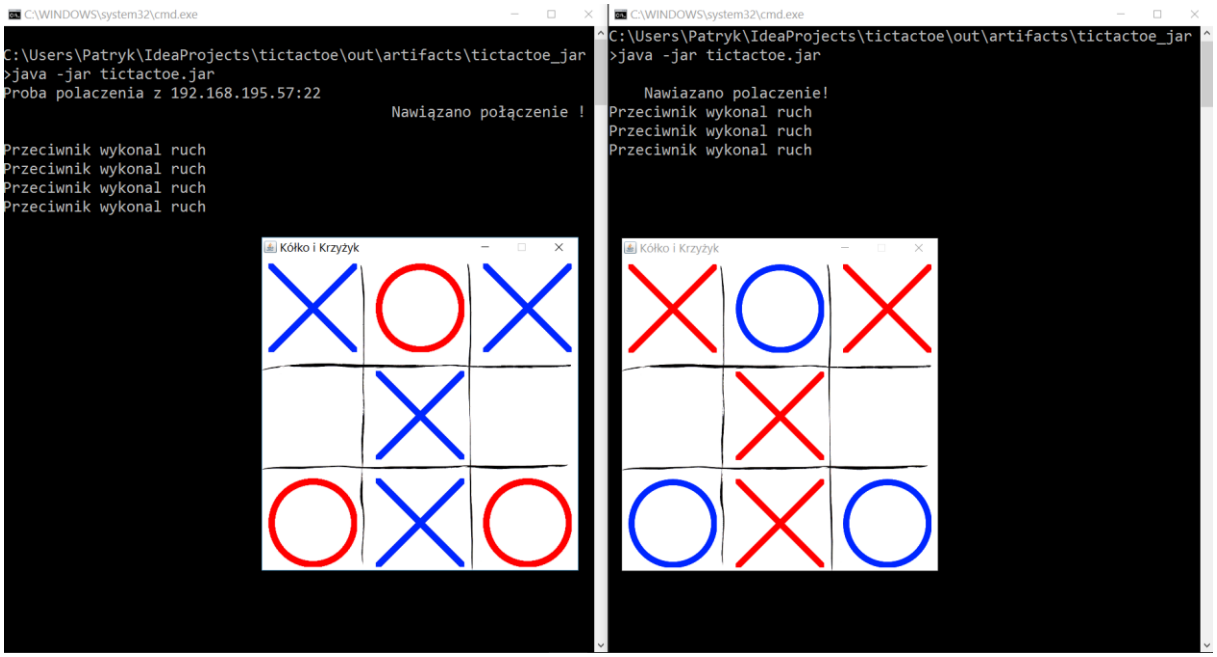


Okno początkowe



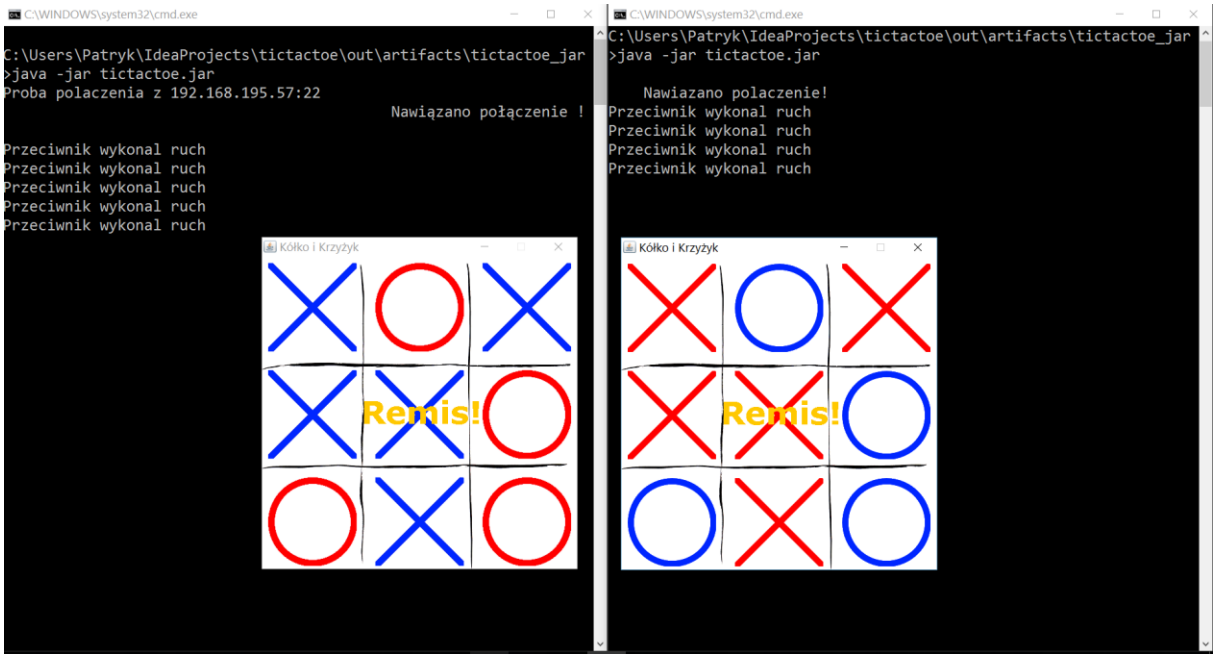
Okno początkowe

W oknie początkowym gracz musi wprowadzić adres IP osoby będącej „serwerem” oraz port tej osoby i po zatwierdzeniu pojawia się okno z grą. Gracze mogą kontynuować rozgrywkę aż do wygranej, przegranej lub remisu.



Gra w trakcie działania (klient został połączony z serwerem)

Po wykonaniu ruchu widzimy rezultaty na bieżąco u siebie jak i u przeciwnika. Gracze posiadają osobne kolory dla rozróżnienia (niebieski i czerwony). Dodatkowo gdy przeciwnik wykona ruch, gracz otrzymuje informację o tym w konsoli gry.



Okno gry, która dobiegła końca

Gdy gra dobiegnie końca wyświetla się stosowny komunikat u obydwu graczy.

Aby zagrać ponownie należy uruchomić grę od nowa plikiem **run.bat** lub **tictactoe.jar**

4. Podsumowanie i wnioski :

Język Java okazał się dobrym językiem do napisania gry Kółko i Krzyżyk gdyż zawiera wiele gotowych i przydanych bibliotek służących do rysowania interfejsu graficznego gry czy połączeń przez Socket. Bez problemu można tworzyć różne gry i aplikacje działające przez sieć i posiadające przy tym interfejs graficzny. Biblioteki graficzne **Java.awt** oraz **Javax.swing**, która jest kontynuacją awt dają dużo możliwości podczas tworzenia aplikacji okienkowych.

Widoczne w grze obiekty czyli kółka, krzyżyki czy kratka mogą być łatwo zmienione na etapie kompilacji gdyż są zawarte osobno w plikach graficznych a program wczytuje je w momencie działania.

W grze sterujemy wyłącznie myszką klikając na wybrany fragment kratki a program interpretuje nasz wybór i w dalszych krokach dokonuje decyzji zgodnie z logiką i zasadami gry.

Posiada ona obsługę błędów na wypadek gdyby gracz podał zły Adres IP lub port co uniemożliwiłoby poprawne połączenie się z drugą osobą.
Aby doszło do połączenia należy wcześniej upewnić się czy dany port nie jest blokowany przez Firewall i w razie potrzeby zmienić go lub odblokować.

Gra napisana przeze mnie jest dobrym przykładem na wykorzystanie komunikacji poprzez sockety. Można w nią grać w obrębie sieci lokalnej na dwóch komputerach w czasie rzeczywistym.