Sprawozdanie końcowe: `Kwantowe Kółko i Krzyżyk`

Michał Żdanuk

Wydział Elektryczny
Politechnika Warszawska
Maj 2022

1 Cel dokumentu

1.1 Przeznaczenie dokumentu

Dokument powstał, by podsumować wszelkie prace związane z realizacją projektu indywidualnego (gry konsolowej poszerzonej o możliwość gry z interfejsem graficznym) - "Kwantowe Kółko i Krzyżyk". Gra została napisana w języku JAVA (openjdk version 17.0.2 2022-01-18), kładąc szczególny nacisk na zasady programowania obiektowego (tzn. abstrakcji, hermetyzacji). Sprawozdanie zawiera kompleksowy opis sposobu realizacji projektu oraz wnioski wyciągnięte po skończeniu prac.

1.2 Organizacja dokumentu

Dokument zawiera:

- wprowadzenie, które wyjaśnia co było celem projektu
- krótki opis modyfikacji wprowadzonych w projekcie względem pierwotnych założeń
- zwięzły opis aktualnego stanu projektu
- opis metod testowania
- podsumowanie całego projektu

1.3 Dokumenty powiązane

Dokument ten związany jest z utworzonymi na początku projektu (pierwszy tydzień prac tzn. 02.03.2022 - 09.03.2022) dokumentami: **specyfikacją implementacyjną oraz specyfikacją funkcjonalną**. W tym dokumencie będę odnosić się do nich, opisując jakie założenia udało się zrealizować, problemy napotkane przy realizacji projektu, a także wszelkie modyfikacje względem wstępnych założeń.

2 Wprowadzenie

Celem projektu było napisanie gry w języku JAVA, która pozwoliłaby dwóm graczom (przy jednym urządzeniu) zagrać w znacznie bardziej zaawansowaną wersję wszystkim dobrze znanej gry Kółko i Krzyżyk. Co więcej, gra posiada opcję trybu gry "single" pozwalającą zagrać przeciwko bot'owi wykonującemu losowe, dozwolone ruchy. Poza aspektem technicznym zależało mi na tym, by gra była miła dla oka. W związku z tym dołożyłem starań i udało mi się wykonać wersję graficzną (okienkową). Ograniczenia czasowe nie pozwoliły na "idealne dopieszczenie" wszystkich aspektów, jednakże gra spełnia wszystkie przyjęte wymagania, kod jest poprawny, a aplikację udało się wzbogacić o dodatkowe funkcjonalności.

3 Modyfikacje

W toku pisania projektu przeszedł on kilka modyfikacji. Zmieniła się przede wszystkich hierarchia klas. Grę wzbogacono o wersję graficzną. W tym celu korzystałem z biblioteki Swing.

3.1 Hierarchia klas

Zaprojektowanie odpowiedniej hierarchii klas okazało się zadaniem znacznie trudniejszym, niż początkowo mnie się wydawało. Względem tego, co przedstawiłem w **Specyfikacji implementacyjnej** w sekcji 4 (Opis zdefiniowanych klas) oraz 5 (Diagram klas) wprowadziłem następujące zmiany:

- zrezygnowanie z klasy Player - jedyne informacje jakie potrzebuję o graczu to tylko określenie, który aktualnie wykonuje ruch - można to rozwiązać po prostu jedną zmienną
- wprowadzenie klasy GUI - uznałem za bardzo ważne wzbogacenie gry o interfejs graficzny, zbudowana jest ona w oparciu o bibliotekę Swing
- wprowadzenie klasy Bot - która ma metody do obsługi ruchów przez komputer (wcześniej fragmenty kodu odpowiadające za to występowały wewnątrz klas Game oraz GUI)

Poza wymienionymi zmianami większość mechanizmów gry została przeniesiona z klasy Game do klasy Board, tak by tylko jedna klasa zawierała logikę ruchów. Zmianie uległ również sposób wyświetlania planszy w wersji terminalowej gry.

3.2 Wersja graficzna gry

W trakcie trwania projektu uznałem za obowiązek wzbogacić grę w wersję graficzną. Gra w wersji konsolowej nie jest zbyt intuicyjna dla zwykłego gracza i nie sprawia tyle przyjemności co ta w wersji graficznej. Do realizacji tej wersji gry skorzystałem z bardzo prostej biblioteki, jaką jest Swing. Stwierdziłem, że do wykonania tej gry najlepszym rozwiązaniem będzie skorzystanie z prostego i intuicyjnego narzędzia, jakim jest biblioteka Swing. Biblioteka jest w pełni darmowa, by korzystać z niej wystarczy wyłącznie importować interesujące nas odpowiednie struktury.

4 Aktualny stan projektu

Udało się zrealizować wszystkie wymagania - do poprawy pozostała jedynie drobna optymalizacja kodu. Przeprowadzone zostały testy jednostkowe najbardziej istotnych metod. Po zakończeniu semestru i sesji egzaminacyjnej planowane jest rozwinięcie gry o możliwość gry z botem o wyższym stopniu trudności. Poniżej zaprezentuję instrukcję jak uruchomić grę we własnym środowisku oraz krótki opis korzystania z aplikacji.

4.1 Uruchomienie projektu w Visual Studio Code

W celu sprawnego wdrożenia aplikacji na własne urządzenie zalecam skorzystać z poniższej instrukcji.
Instrukcja wdrożenia aplikacji na własnym urządzeniu (na systemie Windows):

- należy upewnić się, że na komputerze w VSC zainstalowane jest pakiet rozszerzeń **Extension Pack for Java**
- W konsoli przechodzimy do interesującej nas lokalizacji, tworzymy folder np. mkdir tic-tac-toe
- przechodzimy do katalogu
- klonujemy zawartość repozytorium poleceniem:
  git clone https://github.com/MichalZdanuk/Quantum-TicTacToe.git
- uruchamiamy Visual Studio Code, klikamy przycisk "Open Folder", wskazujemy na ścieżkę do katalogu quantum-tic-tac-toe ze sklonowanego repozytorium w naszym folderze
  (przykładowo: `D:User:DesktoptttQuantum-TicTacToequantum-tic-tac-toe`)
  i wciskamy "Choose folder",
- by gre uruchomić w wersji konsolowej wybieramy klasę Game.java, następnie wciskamy **CTRL+F5**
- by gre uruchomić w wersji graficznej (okienkowej) wybieramy klasę GUI.java, następnie wciskamy **CTRL+F5**

4.2 Uruchomienie pliku .jar

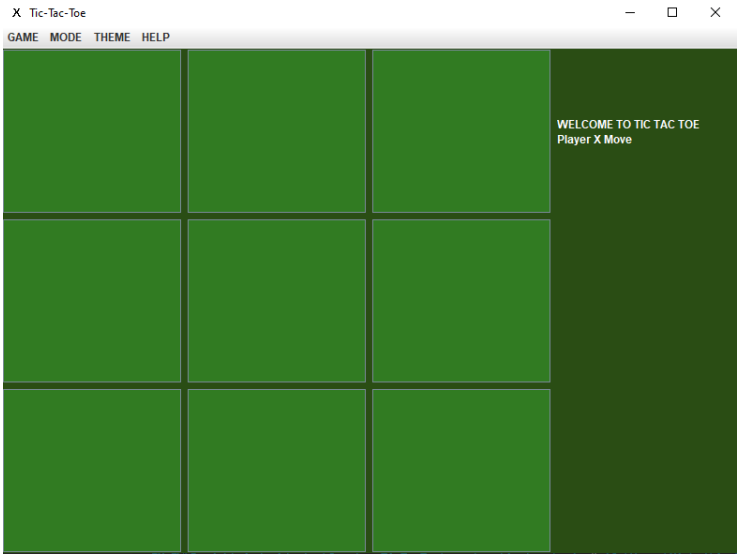
W repozytorium w folderze quantum-tic-tac-toe umieściłem wykonywalne archiwum .jar. Można je uruchomić, klikając dwukrotnie na plik i wybierając (*Otwórz za pomocą...*) uruchomienie za pomocą aplikacji Javy.
Program można również uruchomić przez konsolę za pomocą polecenia: **java -jar ttt.jar** lub wersja konsolowa **java -jar ttt-console.jar**

4.3 Uruchomienie pliku .exe

W repozytorium w folderze quantum-tic-tac-toe znajduje się także plik ttt.exe. Należy utworzyć jego skrót, klikając na niego prawym przyciskiem myszy "*Utwórz skrót*" i wybrać interesującą nas lokalizację. Teraz grę możemy uruchomić podwójnym kliknięciem myszki na ikonkę gry.

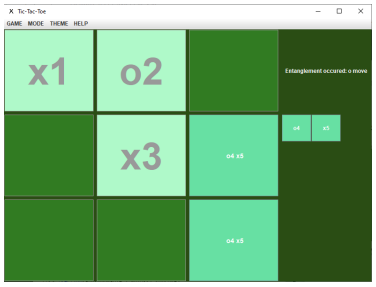
4.4 Instrukcja obsługi

Po wykonaniu instrukcji wdrożeniowej powinniśmy ujrzeć okienko główne gry.

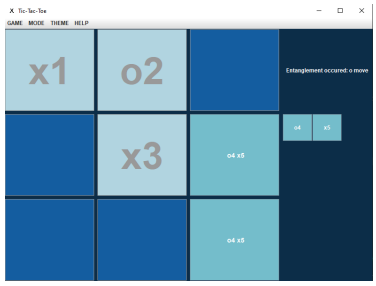


Rysunek 1: zrzut ekranu prezentujący okno główne gry "Kwantowe Kółko i Krzyżyk"

Do dyspozycji mamy cztery przyciski w pasku głównym, górnej części ekranu. Każdy przycisk zawiera rozwijaną listę przycisków odpowiadających za odpowiednie funkcje. Przycisk **"GAME"** zawiera rozwijamy przycisk restart - resetujący całą rozgrywkę, wracając do stanu początkowego przy włączaniu gry. Przycisk **"MODE"** posiada dwa tryby "single" - włączenie gry przeciwko botowi oraz "multi" - umożliwiający grę dwóch graczy. Domyślnie gra, przy włączaniu ustawiana jest na tryb "multi". Przycisk **"THEME"** zawiera cztery przyciski "blue", "red", "green", "orange" umożliwiające ustawienie motywu kolorystycznego grafiki okienka. Domyślnie przy uruchomieniu włączony jest motyw "green". Ostatni przycisk **"HELP"** zawiera opcję "rules", która po wciśnięciu otworzy dodatkowe okienko informacyjne z zasadami gry.



(a) motyw zielony

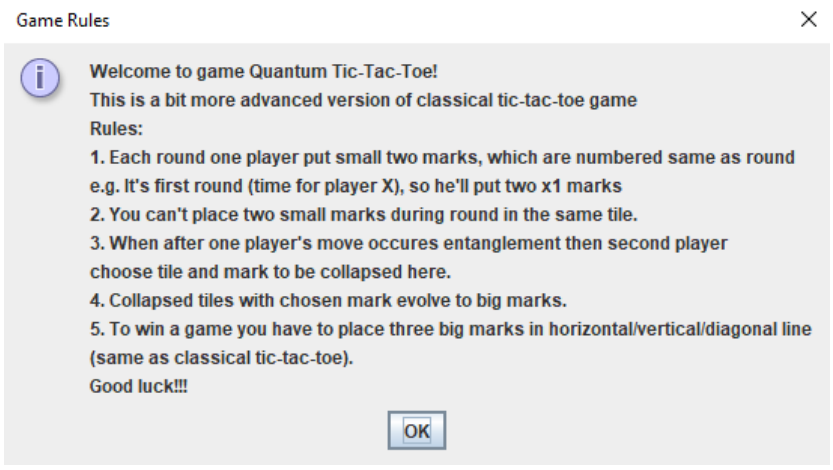


(b) motyw niebieski

(c) motyw czerwony

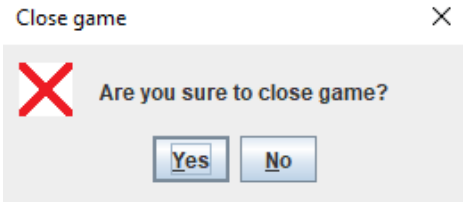
(d) motyw pomarańczowy

Rysunek 2: Zrzut ekranu prezentujący warianty kolorystyczne gry



Rysunek 3: zrzut ekranu prezentujący wyskakujące okno z zasadami gry

Po wciśnięciu ikony X w prawym górnym oknie wyskoczy okno informacyjne z zapytaniem, czy na pewno chcemy wyłączyć gre.



Rysunek 4: zrzut ekranu prezentujący wyskakujące okno z zapytaniem o zamknięcie gry

5 Metody testowania programu

Testy przeprowadzone zostały za pomocą biblioteki JUnit. Przetestowałem najbardziej krytyczne metody. Głównym sposobem weryfikacji poprawności działania mechanizmów gry okazały się testy statystyczne (po napisaniu większych fragmentów kodu) oraz dynamiczne - zwłaszcza pod koniec projektu i przy tworzeniu interfejsu graficznego.

5.1 Testy jednostkowe

Do przeprowadzenia testów jednostokowych wykorzystałem bibliotekę JUnit. Przetestowałem klasę Board.java , która jest najbardziej złożona - odpowiada za całą logikę "plątania" i "zapadania się" (patrz **specyfikacja funkcjonalna**), klasę Tile.java, by sprawdzić poprawną konfigurację stanu pola po wykonaniu operacji oraz Bot.java w celu weryfikacji poprawności generacji ruchów wykonywanych przez bot'a. Przetestowane zostały również przypadki podania błędnych danych - przypadek, gdy gramy w wersji konsolowej.

6 Podsumowanie i wnioski końcowe

Pracę nad projektem rozpocząłem 2 marca 2022 roku - po spotkaniu organizacyjnym z opiekunem projektu, a zakończyłem 19 maja. Okres niespełna trzech miesięcy to niedużo, jednak zdążyłem rozwinąć swoje umiejętności związane z programowaniem obiektowym oraz zdobyć kolejne doświadczenie praktyczne pod kątem prowadzenia małego projektu. Pogłębiłem swoją wiedzę o języku Java.

6.1 Sukcesy

Szczególnie zadowolony jestem z tego, że moja gra **"Kwantowe Kółko i Krzyżyk"** została wzbogacona o możliwość gry z interfejsem graficznym. Po ukończeniu podstawowej wersji graficznej gry, z tygodnia na tydzień wzbogacałem ją o kolejne małe dodatki. Tym sposobem gra została wzbogacona o, chociażby kilka motywów kolorystycznych, czy wyskakujące okienko informacyjne z zasadami gry.

6.2 Samokrytyka i wnioski

Moja praca mogła być niewątpliwie bardziej wydajna i sprawniejsza. Cenną nauczką na przyszłość okazało się, że na okres planowania architektury rozwiązania i stworzenie specyfikacji implementacyjnej i funkcjonalnej należy poświęcić więcej czasu (w moim przypadku na ten etap poświęciłem tylko tydzień). Bardzo szybko wyszły wszelkie niedokładności specyfikacji i należało na nowo przemyśleć architekturę. Powstały kod jest poprawny, jednakże należałoby poświęcić jeszcze czas na dodatkową refaktoryzację (szczególnie "nieelegancko" wykonana została metoda sprawdzająca wygranego gry, przeglądając po kolei wszystkie możliwe kombinacje).
Zdalne repozytorium GIT'a było bardzo przydatnym narzędziem, które pozwoliło mi na kontrolowanie wprowadzanych zmian i stworzyło przestrzeń do przechowywania kopii kodu.
Testy automatyczne są bardzo ważne. Jako założenie obrałem, że będą one powstawały na bieżąco przy tworzeniu nowych funkcjonalności. Niestety powstały one pod koniec projektu, gdy cała funkcjonalność była gotowa. Realizacji testów w trakcie tworzenia funkcjonalności pomogłaby w sprawniejszym wykrywaniu błędów.

Podsumowując, projekt był bardzo ciekawy i pożyteczny, a pisanie go sprawiło przyjemność i satysfakcję.