Kółko krzyżyk

*Single player, multiplayer na 1 telefonie, multiplayer na 2 telefonach z użyciem serwera Heroku + bazy danych PostgreeSql.*

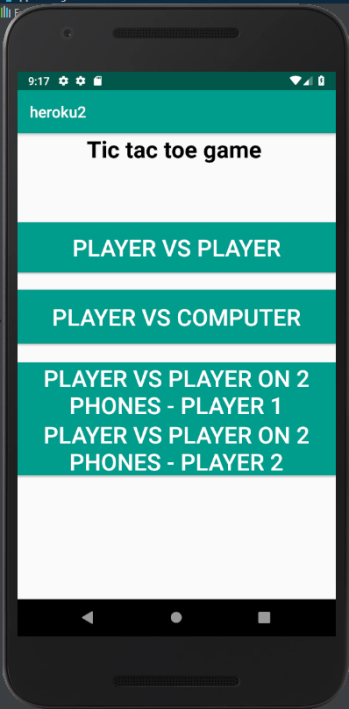
Jak używać aplikacji?

Po włączeniu aplikacji mamy możliwość wyboru trzech trybów gry:

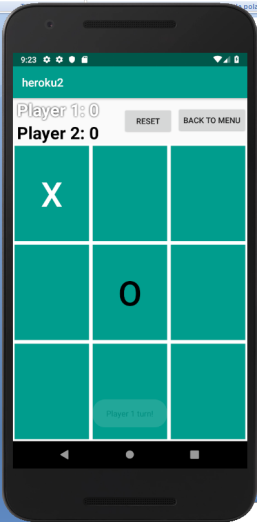
- Single player,

- Multi-player,

- Multiplayer na 2 telefony – wymaga połączenia z Internetem.



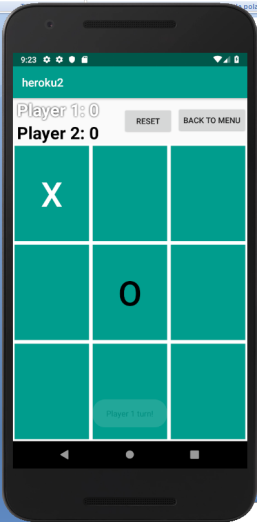
- Po kliknięciu trybu Player vs Player zostaje wyświetlona plansza do gry. Pierwszy gracz „Player 1” jest krzyżykiem a drugi „Player 2” jest kółkiem.



Po wygraniu gry przez jednego z dwóch graczy zostaje mu przypisany punkt, jeśli natomiast gra zakończy się remisem wtedy żaden z graczy nie otrzymuje punktu.

Do dyspozycji mamy również przycisk reset, który czyści plansze oraz resetuje punkty gry oraz przycisk „Back to menu”, który wraca do menu głównego.

- Po kliknięciu trybu Player vs Computer zostaje wyświetlona plansza do gry. Pierwszy gracz „Player 1” jest krzyżykiem a drugi „Computer” jest kółkiem.

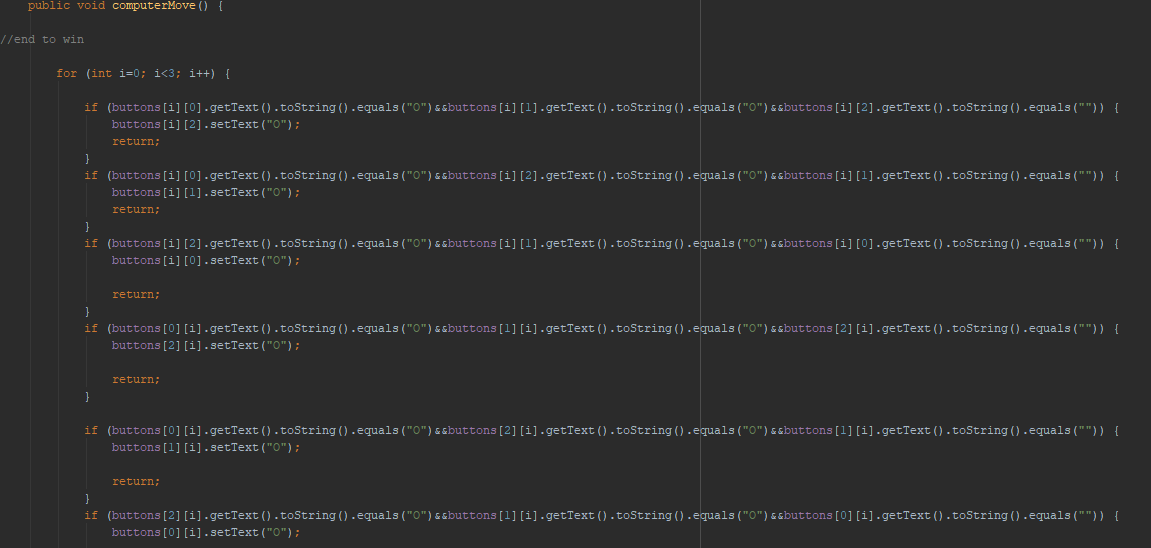


Po wygraniu gry przez gracza lub AI zostaje mu przypisany punkt, jeśli natomiast gra zakończy się remisem wtedy żadnej z graczy nie otrzymuje punktu.

Do dyspozycji mamy również przycisk reset, który czyści plansze oraz resetuje punkty gry oraz przycisk „Back to menu”, który wraca do menu głównego.

Komputer wykorzystuje algorytm brute-force do wyszukania najlepszego możliwego ruchu.

Zachowanie komputera reguluje metoda wykorzystująca algorytm brute-force:



Widzimy tutaj jego wycinek, komputer wybiera najlepszy z możliwych ruchów. Z komputerem można wygrać tylko w jednej sytuacji (specjalna luka w AI).

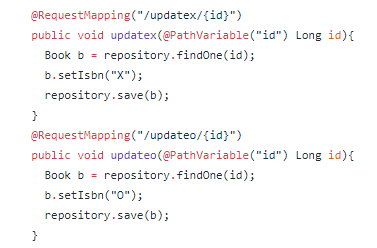
- W trybie Player vs Player on 2 Phones Online mamy do wyboru dwa przyciski, które decydują, który gracz zostaje kółkiem a który krzyżykiem. Player 1 – krzyżyk, Player 2 – kółko.

Ten tryb działa dzięki aplikacji na serwerze Heroku. Na Heroku jest zainstalowana baza danych, w której są przechowane są wybrane pozycje graczy, przykładowo: Pole 1-„X”, Pole 2-„O”.

Połączenie aplikacji klienckiej z serwerem jest zrealizowane za pomocą Springa + Retrofita 2.

**Przykładowe interesujące rozwiązania zastosowane w aplikacji:**

Wpisywanie pozycji gracza do bazy danych (Spring):



Gdzie,

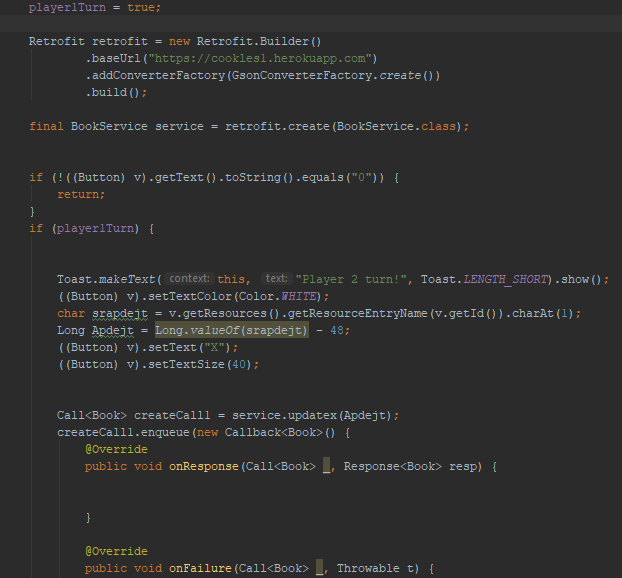
Book b = poszczególne pole w tabeli.

ISBN – komórka, w której zapisujemy „X” lub „O”.

setIsbn – wpisywanie wartości na daną pozycje.

repository.save(b) – zapisuje zmiany w bazie.

Komunikacja aplikacji z mapowaniem Springa za pomocą Retrofit 2:

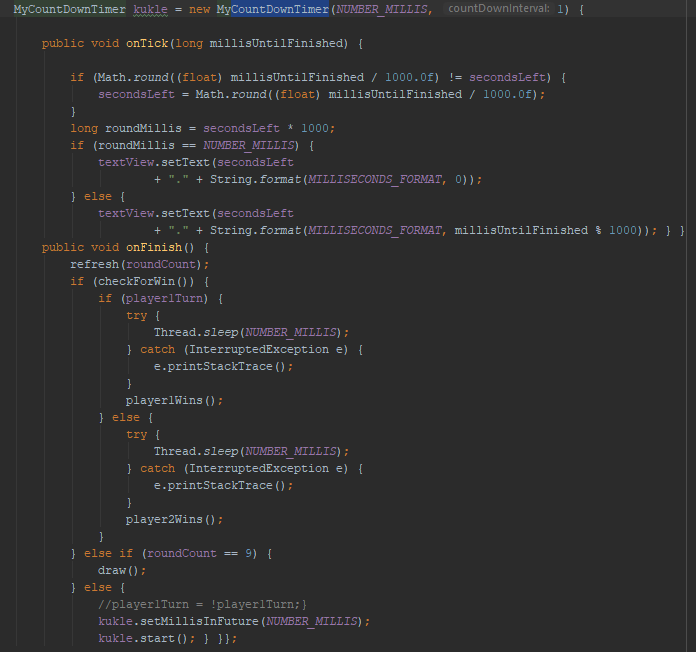


Na zrzucie ekranu został przedstawiony fragment kodu odpowiedzialny za zapisanie pozycji gracza „X” w bazie danych (na serwerze) oraz lokalnie na kliencie.

Początkowo inicjalizujemy Retrofita, później tworzymy serwis, na którym będziemy operować (zapewnia nam dostęp do mapowań Springa).

Zmienna spradejt wyszukuje ID buttona, na który kliknął użytkownik, następnie konwertujemy wartość na Long (id w bazie jest typu long), ustawiamy tekst buttona i tworzymy zapytanie poleceniem Create Call. Z racji, że komunikacja z Heroku jest asynchroniczna, używamy metody „enqueue”, która czeka na odpowiedź serwera.

Przekazywanie ruchów jednego gracza do drugiego zostało wykonane autorsko z pomocą wbudowanej funkcji androida „CountDownTimer”, do której został dorobiony setter na potrzeby naszej aplikacji. Licznik ma za zadanie, co sekundę uruchamiać funkcje Refresh, która z kolei pobiera wszystkie pozycje graczy z bazy danych i wyświetla je na ekranie (zmieniając tekst buttonów i sprawdzając czy przeciwnik nie wygrał).



Podsumowanie wykorzystanych technologii:

Cały projekt został wykonany w Android Studio, wykorzystano Framework Spring (komunikacja z bazą), Retrofit 2 (komunikacja front endu z Springiem), logika aplikacji została napisana w Javie.

Wykonali: Krystian Drozd, Adrian Bury.