Dokumentacja deweloperska

Rozproszona platforma do tworzenia gier planszowych, przeznaczona na system Android.

*Niniejsze opracowanie powstało w trakcie i jako rezultat za­jęć dy­dak­tycz­nych z przedmiotu wymienionego na stronie tytułowej, pro­wa­dzo­nych w Akademii Górniczo-Hutniczej w Krakowie (AGH) przez oso­bę (oso­by) wy­mie­nioną (wymienione) po słowach "Pro­wa­dzą­cy zajęcia" i nie może być wy­korzystywane w jakikolwiek sposób i do jakichkolwiek ce­lów, w ca­ło­ści lub części, w szczególności pub­li­ko­wa­ne w ja­ki­kol­wiek spo­sób i w jakiejkolwiek formie, bez uzy­ska­nia uprzed­niej, pi­sem­nej zgody tej oso­by (tych osób) lub odpowiednich władz AGH.****Copyright © 2013 Akademia Górniczo-Hutnicza (AGH) w Krakowie***

**Spis treści**

1. Zakres dokumentu 3

2. Zastosowane rozwiązania implementacyjne 4

2.1. Obsługa tur 4

2.2. Logika Gry 4

2.3. Połączenie z innymi graczami 4

2.4. Grafika 5

2.5. Wykonywanie tury przez gracza 5

2.6. Zapisywanie i wczytywanie gier 5

2.7. Menu 6

3. Bibliografia 7

Indeks ważniejszych terminów 8

Spis ilustracji i tabel 8

# Zakres dokumentu

Niniejszy dokument opisuje przyjęte rozwiązania implementacyjne platformy. Znajduje się w nim opis aktualnego stan implementacji jak i informacje potrzebne do dalszego rozwijania projektu.

# Zastosowane rozwiązania implementacyjne

W rozdziale znajdują się informację na temat wysokopoziomowej architektury rozwiązania , skupiając się na informacjach które byłyby przydatne w trakcie rozwijania projektu. Podzielony jest on na kilka części w których są omawiane jak poszczególne składowe platformy były implementowane.

Oczywiście do dokładniejszego poznania działania platformy należy zapoznać się z dokumentacją techniczną zawierającą dokładniejsze informacje na temat składowych klas i pakietów kodu źródłowego platformy.

## Obsługa tur

Obsługa tur znajduje się w pakiecie game w klasie Game. Po rozpoczęciu gry metodą start gra wchodzi w pętle której warunek zakończenia jest badany przez odczyt wartości zwracanej przez metodę gameEnded sprawdzającą obecny stan rozgrywki. Z obecnego stanu rozgrywki State są wczytywane parametry takie jak:

- lista wszystkich aktywnych graczy ( getAllPlayers)

- obecny kontekst rozgrywki ( getCurrentPlayer )

- obecny gracz ( getCurrentPlayer)

Na obiekcie stanu przeprowadzana jest tura, po wcześniejszym odczytaniu wyników tury od wszystkich graczy biorących w rozgrywce. To od implementującego grę zależy czy tylko jeden gracz będzie mógł wykonać akcję w turze czy też nie. Po odczytaniu wyników tury w postaci obiektu typu TurnResult ten wynik jest przekazany do obecnego stanu poprzez wywołanie na nim metody makeTurn, która wykonuje odpowiednie akcje, zmienia model rozgrywki ( plansze, ułożenie pionków) i być może (jeżeli to konieczne) stan rozgrywki. Obsługa tur na platformie jest obecnie zaimplementowana i nie wymaga zmian.

## Logika Gry

Logikę gry na platformie implementuje się poprzez utworzenie odpowiedniej ilości stanów typu State z których jeden jest stanem początkowym. Obiekt State implementuje wzorzec projektowy Stan gdzie poprzez metodę changeState z interfejsu StateChangable możliwa jest zmiana obecnego stanu. Z obiektu Stan

są wczytywane wszystkie informację potrzebne do przeprowadzenia tury.

## Połączenie z innymi graczami

Obecnie połączenia z pokojami , innymi graczami odbywają się poprzez interfejs Java RMI. Pokoje w momencie utworzenia gry czekają na graczy. W wypadku połączenia z pokojem gracz podaje parametry swojego połączenia poprzez proxy RemoteTurnHandler, na którym to później są wywoływane metody zdalne.

Jak do tej pory cały pakiet network związany z połączeniem bezprzewodowym ,jak już to było wspomniane, jest napisany w Java RMI , który jest przeznaczony do obsługi połączeń internetowych na komputerach klasy PC.

By móc spełnić cel projektu czyli implementację połączenia z innymi graczami poprzez Bluetooth należałoby cały pakiet network przepisać, szczególną uwagę poświęcając klasie RemoteTurnHandler z wykorzystaniem dostępnego środowiska middleware ( np Ice które zawiera implementację RMI na telefony z systemem Android) , bądź z wykorzystaniem niskopoziomowego dostępu do interfejsu Bluetooth.

## Grafika

Na obecną chwilę platforma zrzuca odpowiedzialność na twórcę gry . Wyświetlanie grafiki odbywa się w metodzie klasy TurnHandler makeTurn która przyjmuje kontekst obecnej rozgrywki w którym są wszelkie informacje na temat obecnego jej stanu.

Implementacja grafiki na platformie wymagałaby zaprojektowania interfejsu dostępu do metod związanych z grafiką w obiekcie klasy TurnHandler z poziomu metody makeTurn nie zmniejszając elastyczności rdzenia platformy.

## Wykonywanie tury przez gracza

W chwili obecnej odpowiedzialność za wczytanie akcji wykonanych przez gracza jest w pełni odpowiedzialny twórca gry. Implementacja mechanizmów ułatwiających byłaby związana podobnie jak w przypadku grafiki z metodą makeTurn klasy TurnHandler i jej odpowiednim dowiązaniem z grafiką wyświetlaną dla odpowiednich akcji.

## Zapisywanie i wczytywanie gier

Obecny stan implementacji nie uwzględnia zarówno zapisów jak i odczytów stanów rozgrywki. Implementacja tej funkcjonalności wymagałaby ustawienia bazy danych na każdej kopii gry podczas jej instalacji. Następnie należałoby rozszerzyć metodę start klasy Game o obsługę żadania zapisania stanu rozgrywki który by w tej sytuacji po połączeniu z bazą danych zapisywałby jej stan. Wczytanie stanu

rozgrywki mogłoby być zrealizowane jako dodatkową metodę start klasy Game z parametrem będącym wczytywanym stanem rozgrywki który odpowiednio zainicjalizowany stanowiłby rozpoczęcie nowej rozgrywki. Należałoby klasę TurnResult rozszerzyć o możliwość komunikowania klasy Game o żądanie stanu zapisu. Oczywiście GUI też musiałoby wspierać tę opcję.

## Menu

W chwili obecnej platforma nie posiada żadnego zunifikowanego menu dla tworzonych gier , musi być on obsłużony przez twórcę. Podczas rozwijania projektu wprowadzenie tego mechanizmu wymagałoby napisania go w pełni.

# Bibliografia

1. Brett D. McLaughlin, Gary Police & David West: *Object Oriented Analysis and Design*, O’Reilly, 2007

Indeks ważniejszych terminów

**Nie można odnaleźć żadnych haseł indeksu.**

1. Spis ilustracji i tabel

Ilustracje

**Nie można odnaleźć pozycji dla spisu ilustracji.**

Tabele