Warcaby

Dokumentacja Projektu

Wprowadzenie do Dokumentacji, Opis Klas, Diagramy, Testy jednostkowe

2015

Jarosław Biskupski, Krzysztof Matysik, Jakub Buziałkowski Michał Miszewski, Kamil Pałagan

Warcaby

2015-05-28

Zawartość

[1 Wstęp - Wprowadzenie do dokumentacji 3](#_Toc420577223)

[1.1 Definicje i skróty 3](#_Toc420577224)

[1.1.1 Rozgrywka - jest to główna funkcjonalność aplikacji, która odpowiada za obsługę zdarzeń związanych z przebiegiem tury gry w warcaby pomiędzy dwoma użytkownikami 3](#_Toc420577225)

[1.1.2 User-friendly - przyjazny użytkownikowi 3](#_Toc420577226)

[1.1.3 GUI - graficzny interfejs użytkownik 3](#_Toc420577227)

[1.2 Charakterystyka użytkowników (Aktorów) 3](#_Toc420577228)

[1.2.1 Gracz - Końcowy odbiorca projektu, typ użytkownika grający w trybie samouczka lub pierwszy gracz w trybie - turowym 3](#_Toc420577229)

[1.2.2 Gracz II - drugi gracz uczestniczący w trybie wieloosobowym - turowym 3](#_Toc420577230)

[1.3 Ogólne możliwości aplikacji 3](#_Toc420577231)

[1.3.1 Przeprowadzenie rozgrywki dwuosobowej na jednym urządzeniu (system turowy) 3](#_Toc420577232)

[1.3.2 Odtwarzanie prostych dźwięków związanych z ruchami pionków na szachownicy 3](#_Toc420577233)

[1.3.3 Odtwarzanie muzyki w tle 3](#_Toc420577234)

[1.3.4 Samouczek przedstawiający zasady gry w warcaby 3](#_Toc420577235)

[1.3.5 Możliwość wyboru stylu graficznego szachownicy 3](#_Toc420577236)

[1.3.6 Możliwość wyboru stylu graficznego pionków 3](#_Toc420577237)

[1.3.7 Zmiana perspektywy widoku 3](#_Toc420577238)

[1.4 Ogólne ograniczenia. 3](#_Toc420577239)

[1.4.1 Użytkownik musi posiadać urządzenie z systemem Android w wersji co najmniej 4.0.3 3](#_Toc420577240)

[1.4.2 Urządzenie musi być wyposażone w interfejs bluetooth w wersji 2.0 lub wyższej 3](#_Toc420577241)

[1.4.3 Urządzenie musi posiadać minimum 512 MB pamięci operacyjnej oraz procesor taktowany z częstotliwością 800 MHz lub wyższej 3](#_Toc420577242)

[1.4.4 Aplikacja potrzebuje około 100MB pamięci wewnętrznej 3](#_Toc420577243)

[2 Dokumentacja struktur danych 4](#_Toc420577244)

[2.1 Dokumentacja klasy Board 4](#_Toc420577245)

[2.1.1 Metody publiczne 4](#_Toc420577246)

[2.1.2 Pola danych 4](#_Toc420577247)

[2.1.3 Opis szczegółowy 4](#_Toc420577248)

[2.1.4 Dokumentacja funkcji składowych 4](#_Toc420577249)

[2.2 Dokumentacja klasy ButtonsControl 6](#_Toc420577250)

[2.2.1 Metody publiczne 6](#_Toc420577251)

[2.2.2 Pola danych 6](#_Toc420577252)

[2.2.3 Opis szczegółowy 6](#_Toc420577253)

[2.2.4 Dokumentacja funkcji składowych 6](#_Toc420577254)

[2.3 Dokumentacja klasy CameraPerspective 7](#_Toc420577255)

[2.3.1 Metody publiczne 7](#_Toc420577256)

[2.3.2 Pola danych 7](#_Toc420577257)

[2.3.3 Opis szczegółowy 7](#_Toc420577258)

[2.3.4 Dokumentacja funkcji składowych 7](#_Toc420577259)

[2.4 Dokumentacja klasy Cursor 7](#_Toc420577260)

[2.4.1 Dodatkowe Dziedziczone Składowe 8](#_Toc420577261)

[2.5 Dokumentacja klasy Field 8](#_Toc420577262)

[2.5.1 Metody publiczne 8](#_Toc420577263)

[2.5.2 Pola danych 8](#_Toc420577264)

[2.5.3 Opis szczegółowy 8](#_Toc420577265)

[2.5.4 Dokumentacja pól 8](#_Toc420577266)

[2.6 Dokumentacja struktury FieldCords 9](#_Toc420577267)

[2.6.1 Pola danych 9](#_Toc420577268)

[2.7 Dokumentacja klasy Gameplay 9](#_Toc420577269)

[2.7.1 Metody publiczne 9](#_Toc420577270)

[2.7.2 Pola danych 9](#_Toc420577271)

[2.7.3 Opis szczegółowy 9](#_Toc420577272)

[2.7.4 Dokumentacja funkcji składowych 10](#_Toc420577273)

[2.7.5 Dokumentacja pól 11](#_Toc420577274)

[2.8 Dokumentacja klasy HintWindow 11](#_Toc420577275)

[2.8.1 Opis szczegółowy 11](#_Toc420577276)

[2.9 Dokumentacja klasy Menu 11](#_Toc420577277)

[2.9.1 Metody publiczne 11](#_Toc420577278)

[2.9.2 Pola danych 12](#_Toc420577279)

[2.9.3 Opis szczegółowy 12](#_Toc420577280)

[2.9.4 Dokumentacja funkcji składowych 12](#_Toc420577281)

[2.10 Dokumentacja klasy MenuBoard 13](#_Toc420577282)

[2.10.1 Pola danych 13](#_Toc420577283)

[2.10.2 Opis szczegółowy 13](#_Toc420577284)

[2.10.3 Dokumentacja pól 13](#_Toc420577285)

[2.11 Dokumentacja klasy Options 13](#_Toc420577286)

[2.11.1 Metody publiczne 13](#_Toc420577287)

[2.11.2 Pola danych 14](#_Toc420577288)

[2.11.3 Opis szczegółowy 14](#_Toc420577289)

[2.11.4 Dokumentacja pól 14](#_Toc420577290)

[2.12 Dokumentacja klasy Pawn 14](#_Toc420577291)

[2.12.1 Metody publiczne 14](#_Toc420577292)

[2.12.2 Pola danych 15](#_Toc420577293)

[2.12.3 Opis szczegółowy 15](#_Toc420577294)

[2.12.4 Dokumentacja funkcji składowych 15](#_Toc420577295)

[2.12.5 Dokumentacja pól 15](#_Toc420577296)

[2.13 Dokumentacja klasy Player 16](#_Toc420577297)

[2.13.1 Metody publiczne 16](#_Toc420577298)

[2.13.2 Pola danych 16](#_Toc420577299)

[2.13.3 Opis szczegółowy 16](#_Toc420577300)

[2.13.4 Dokumentacja funkcji składowych 16](#_Toc420577301)

[2.14 Dokumentacja klasy Tutorial 17](#_Toc420577302)

[2.14.1 Metody publiczne 17](#_Toc420577303)

[2.14.2 Opis szczegółowy 17](#_Toc420577304)

[2.14.3 Dokumentacja funkcji składowych 18](#_Toc420577305)

[3 Podsumowanie 19](#_Toc420577306)

[3.1 Aplikacja działa stabilnie i poprawnie oraz spełnia wszystkie wymagania funkcjonalne oraz nie funkcjonalne - rozgrywka jest satysfakcjonująca 19](#_Toc420577307)

[3.2 Apliakacja zostałą umieszczona w sklepie Google Play w celu jej dystrybucji 19](#_Toc420577308)

[3.3 Aplikacja działa na systemach Android 4.0.3 i nowszych 19](#_Toc420577309)

[3.4 Aplikacja została wykonana w środowisku Unity , wykorzystywany był język programowania C# 19](#_Toc420577310)

[3.5 Testy zostały przeprowadzone pomyślnie, napotkane błędy usunięte, 19](#_Toc420577311)

[3.6 Odpowiednie diagramy i tabele zostały dodane w załączniku w celu zwiększenia ich czytelności 19](#_Toc420577312)

[4 Załączniki 20](#_Toc420577313)

[4.1 Diagram Klas 20](#_Toc420577314)

[4.2 Schematy blokowe użyte podczas projektowania aplikacji: 21](#_Toc420577315)

[4.3 Diagram Przepływu danych: 22](#_Toc420577316)

[4.4 Tabele testów: 23](#_Toc420577317)

# Wstęp - Wprowadzenie do dokumentacji

Dokemunt ten zawiera dokumentację do projektu "Warcaby", którego celem było stworzenie aplikacji - gry na system Android 4.0.3 i nowszy. Aplikacja spełnia wszystkie wymogi , które były zawarte w "System Requirements Secification for "Warcaby".

## Definicje i skróty

### Rozgrywka - jest to główna funkcjonalność aplikacji, która odpowiada za obsługę zdarzeń związanych z przebiegiem tury gry w warcaby pomiędzy dwoma użytkownikami

### User-friendly - przyjazny użytkownikowi

### GUI - graficzny interfejs użytkownik

## Charakterystyka użytkowników (Aktorów)

### Gracz - Końcowy odbiorca projektu, typ użytkownika grający w trybie samouczka lub pierwszy gracz w trybie - turowym

### Gracz II - drugi gracz uczestniczący w trybie wieloosobowym - turowym

## Ogólne możliwości aplikacji

### Przeprowadzenie rozgrywki dwuosobowej na jednym urządzeniu (system turowy)

### Odtwarzanie prostych dźwięków związanych z ruchami pionków na szachownicy

### Odtwarzanie muzyki w tle

### Samouczek przedstawiający zasady gry w warcaby

### Możliwość wyboru stylu graficznego szachownicy

### Możliwość wyboru stylu graficznego pionków

### Zmiana perspektywy widoku

## Ogólne ograniczenia.

### Użytkownik musi posiadać urządzenie z systemem Android w wersji co najmniej 4.0.3

### Urządzenie musi być wyposażone w interfejs bluetooth w wersji 2.0 lub wyższej

### Urządzenie musi posiadać minimum 512 MB pamięci operacyjnej oraz procesor taktowany z częstotliwością 800 MHz lub wyższej

### Aplikacja potrzebuje około 100MB pamięci wewnętrznej

# Dokumentacja struktur danych

## 2.1 Dokumentacja klasy Board

Klasa Planszy

Dziedziczy MonoBehaviour.

### Metody publiczne

List< GameObject > **getDarkFieldsList** ()

*Zwraca listę ciemnych pól*

**FieldCords** **TranslateCords** (string sID)

*Funkcja zamienia ID pola ze stringa do postaci współrzędnych*

void **setFieldState** (FieldState fStat, string idField)

*Ustawia status pola*

FieldState **getFieldState** (string idField)

*zwraca status pola*

Vector3 **getFieldCenterCoordinate** (string idField)

*zwraca współrzędne środka pola*

int **getPawnIDonField** (string idField)

string **getFieldIDinLine** (string startFieldID, string middleFieldID)

*Zwraca id pola w lini prostej, jeśli pole jest zbyt blisko krawędzi planszy to zwraca napis brak\_pola*

bool **isItFront** (string sFieldID, string eFieldID, int pID)

*Sprawdza czy pole sąsiadujące z pionkiem jest z przodu*

List< string > **getSurroundingFields** (string idField)

*Funkcja zwraca id-ki pól sąsiadujących z polem o podanym id jako argument*

List< string > **getSurroundingFieldsWithEnemies** (string idField, int pID)

List< string > **getFrontFields** (string idField, int pID)

bool **searchForCaptures** (**Pawn** cPawn)

bool **areThereCaptures** (**Pawn** cPawn)

### Pola danych

GameObject **Field\_bright**

GameObject **Field\_dark**

GameObject[][] **Fields**

### Opis szczegółowy

Klasa Planszy

### Dokumentacja funkcji składowych

#### bool Board.areThereCaptures (Pawn *cPawn*)

Odczytujemy id pól znajdujących się w sąsiedztwie pionka, na których stoi wróg

Sprawdzamy czy pola za pionkiem wroga w lini prostej są wolne

Najpierw szukamy id pola za pionkiem wroga

Trzeba uzyskać id pionka wroga do bicia

ID pionka trzeba przetłumaczyć na index

Sprawdzamy czy to pole jest puste

#### List<GameObject> Board.getDarkFieldsList ()

Zwraca listę ciemnych pól

#### Vector3 Board.getFieldCenterCoordinate (string *idField*)

zwraca współrzędne środka pola

Konwertujemy id pola

Zwracamy pozycję pola jako Vector3

#### string Board.getFieldIDinLine (string *startFieldID*, string *middleFieldID*)

Zwraca id pola w lini prostej, jeśli pole jest zbyt blisko krawędzi planszy to zwraca napis brak\_pola

#### FieldState Board.getFieldState (string *idField*)

zwraca status pola

zwracamy status pola

#### int Board.getPawnIDonField (string *idField*)

Konwertujemy ID pola

#### List<string> Board.getSurroundingFields (string *idField*)

Funkcja zwraca id-ki pól sąsiadujących z polem o podanym id jako argument

#### List<string> Board.getSurroundingFieldsWithEnemies (string *idField*, int *pID*)

Sprawdzamy czy coś stoi na polu

Sprawdzamy czy to pionek wroga

#### bool Board.isItFront (string *sFieldID*, string *eFieldID*, int *pID*)

Sprawdza czy pole sąsiadujące z pionkiem jest z przodu

#### bool Board.searchForCaptures (Pawn *cPawn*)

Sprawdzamy czy pola za pionkiem wroga w lini prostej są wolne

Najpierw szukamy id pola za pionkiem wroga

Trzeba uzyskać id pionka wroga do bicia

ID pionka trzeba przetłumaczyć na index

Zabezpieczenie przed sytuacją kiedy nie ma możłiwości bicia pionków, wtedy musi nastąpić zmiana tury

Sprawdzamy czy to pole jest puste

Jeśli tak to podświetlamy

Zapamiętujemy id pionka ktory wykonuje bicie

#### void Board.setFieldState (FieldState *fStat*, string *idField*)

Ustawia status pola

Konwertujemy ID pola

Ustawiamy stan dla pola podany w argumencie

#### FieldCords Board.TranslateCords (string *sID*)

Funkcja zamienia ID pola ze stringa do postaci współrzędnych

#### Dokumentacja dla tej klasy została wygenerowana z pliku:

gameplay/Board.cs

## Dokumentacja klasy ButtonsControl

Klasa Buttons.Control

Dziedziczy **Menu**.

### Metody publiczne

void **PlayClickSound** ()

*Odtwarzanie dzwięku*

### Pola danych

AudioClip **click**

##### *Parametry:*

|  |  |
| --- | --- |
| *click* | Dzwięku "clicku" n |

### Opis szczegółowy

Klasa Buttons.Control

### Dokumentacja funkcji składowych

#### void ButtonsControl.PlayClickSound ()

Odtwarzanie dzwięku

#### Dokumentacja dla tej klasy została wygenerowana z pliku:

ButtonsControl.cs

## Dokumentacja klasy CameraPerspective

Klasa Odpoweidzialna za ustawienia kamery

Dziedziczy MonoBehaviour.

### Metody publiczne

void **CameraZoomIn** ()

void **CameraZoomOut** ()

void **setTopView** ()

void **setFreeView** ()

### Pola danych

GameObject **buttontopview**

Camera **camera**

float **ZoomSpeed**

### Opis szczegółowy

Klasa Odpoweidzialna za ustawienia kamery

### Dokumentacja funkcji składowych

#### void CameraPerspective.setFreeView ()

Ustawienie elementów interfejsu

Przywrócenie kamery do poprzedniego widoku

#### void CameraPerspective.setTopView ()

Ustawienie elementów interfejsu

Zapamiętanie pozycji kamery

Ustawienie kamery nad planszą

#### Dokumentacja dla tej klasy została wygenerowana z pliku:

gameplay/CameraPerspective.cs

## Dokumentacja klasy Cursor

Dziedziczy **Gameplay**.

### Dodatkowe Dziedziczone Składowe

Dokumentacja dla tej klasy została wygenerowana z pliku:

gameplay/Cursor.cs

## Dokumentacja klasy Field

Klasa Pola Gry

Dziedziczy MonoBehaviour.

### Metody publiczne

void **setHighlighted** ()

void **unsetHighlighted** ()

### Pola danych

string **idField**

*Identyfikator pola*

FieldState **fState**

*Stan pola*

int **pawnId**

*Identyfikator pionka który stoi na polu*

### Opis szczegółowy

Klasa Pola Gry

### Dokumentacja pól

#### FieldState Field.fState

Stan pola

#### string Field.idField

Identyfikator pola

#### int Field.pawnId

Identyfikator pionka który stoi na polu

#### Dokumentacja dla tej klasy została wygenerowana z pliku:

gameplay/Field.cs

## Dokumentacja struktury FieldCords

### Pola danych

int **Y**

int **X**

Dokumentacja dla tej struktury została wygenerowana z pliku:

gameplay/Board.cs

## Dokumentacja klasy Gameplay

Klasa **Gameplay**

Dziedziczy MonoBehaviour.

Dziedziczona przez **Cursor**.

### Metody publiczne

void **onTurnStart** (int pID)

*Funkcja musi być wywoływana na początku każdUj tury aby sprawdzić czy są ruchy i bicia dla danego gracza*

GameObject **getEnemyPlayer** ()

*Zwraca gracza który posiada prawo ruchu*

void **changeTurn** ()

*Funkcja zmieniająca gracza*

void **showResults** (string Winner)

void **backToMainMenu** ()

### Pola danych

int **whoseTurnID**

*Elementy funkcjonalne*

bool **isThereCapture**

bool **isThereMove**

GameObject **panelGameEnd**

*Elementy GUI*

GameObject **panelGameStart**

Text **textWinner**

Image **wPawnS**

GameObject **Field\_bright**

GameObject **Field\_dark**

Mesh **boardMesh**

Material **boardMaterial**

### Opis szczegółowy

Klasa **Gameplay**

### Dokumentacja funkcji składowych

#### void Gameplay.backToMainMenu ()

Sprawdzamy platformę i zamykamy program

#### void Gameplay.changeTurn ()

Funkcja zmieniająca gracza

#### GameObject Gameplay.getEnemyPlayer ()

Zwraca gracza który posiada prawo ruchu

#### void Gameplay.onTurnStart (int *pID*)

Funkcja musi być wywoływana na początku każdUj tury aby sprawdzić czy są ruchy i bicia dla danego gracza

Ustawiamy wartość określającą czy jest jakieś bicie na fałsz

Ustawiamy wartość określającą czy jest jakiś ruch na fałsz

Pobieramy listę pionków gracza

Trzeba sprawdzić każdy pionek pod względem możliwości wykonania ruchu bądź bicia

Pobieramy komponent skryptu dla aktualnie sprawdzanego pionka

Sprawdzamy czy pionek nie został ubity

Trzeba pobrać id pola na którym jest pionek

Tworzymy listę która będzie przechowywać id-ki sąsiadujących pól

W tej pętli sprawdzane będą wszystkie pola sąsiadujące, czy na którymś jest pionek wroga

Jeśli pole jest nie puste

Jeśli pionek wroga

Sprawdzamy czy w lini prostej za tym polem jest wolne pole

Trzeba sprawdzić czy pole za jest wolne dlatego pobieramy status pola końcowego

sprawdzamy czy pole końcowe jest puste

Jeśli pole za pionkiem wroga jest puste to mamy bicie

Jeśli pole graniczne jest puste

Sprawdzamy czy pole graniczne jest z przodu

jeśli pole jest z przodu to ustawiamy MOVE\_ALLOWED

a pionek moze wykonać ruch jeśli pionek ma bicie to nie może wykonać ruchu

Sprawdzamy czy są jakieś ruchy lub bicia

Jeśli nie ma możliwych ruchów lub bić to koniec gry i remis

#### void Gameplay.showResults (string *Winner*)

Wyświetlenie remisu

Wyświetlenie informacji o tym który gracz wygrał

### Dokumentacja pól

#### GameObject Gameplay.panelGameEnd

Elementy GUI

#### int Gameplay.whoseTurnID

Elementy funkcjonalne

#### Dokumentacja dla tej klasy została wygenerowana z pliku:

gameplay/Gameplay.cs

## Dokumentacja klasy HintWindow

Klasa **HintWindow**

Dziedziczy MonoBehaviour.

### Opis szczegółowy

Klasa **HintWindow**

Dokumentacja dla tej klasy została wygenerowana z pliku:

Tutorial/HintWindow.cs

## Dokumentacja klasy Menu

Klasa **Menu** Gównego

Dziedziczy MonoBehaviour.

Dziedziczona przez **ButtonsControl**.

### Metody publiczne

void **ExitApplication** ()

*Sprawdzenie platformy i zamknięcie aplikacji*

void **CreateGameOnOneDevice** ()

*Przejście do nowej sceny - utworzenie rozgrywki na jednym urządzeniu*

void **CreateGame** ()

*Przejście do nowej sceny - utworzenie rozgrywki*

void **JoinGame** ()

*Przejście do nowej sceny - dołączenie do rozgrywkui*

void **StartTutorial** ()

*Przejście do nowej sceny - samouczka*

void **changeMusicVolume** ()

void **changeSoundsVolume** ()

void **changeGraphicQuality** ()

void **changeShadows** ()

void **Info** ()

*Pokazujemy panel z informacjami*

void **InfoExit** ()

*Ukrywamy panel z informacjami*

### Pola danych

GameObject **mainCanva**

GameObject **options**

InputField **inputGameName**

ToggleGroup **toggleColorGroup**

### Opis szczegółowy

Klasa **Menu** Gównego

### Dokumentacja funkcji składowych

#### void Menu.CreateGame ()

Przejście do nowej sceny - utworzenie rozgrywki

#### void Menu.CreateGameOnOneDevice ()

Przejście do nowej sceny - utworzenie rozgrywki na jednym urządzeniu

#### void Menu.ExitApplication ()

Sprawdzenie platformy i zamknięcie aplikacji

#### void Menu.Info ()

Pokazujemy panel z informacjami

#### void Menu.InfoExit ()

Ukrywamy panel z informacjami

#### void Menu.JoinGame ()

Przejście do nowej sceny - dołączenie do rozgrywkui

#### void Menu.StartTutorial ()

Przejście do nowej sceny - samouczka

#### Dokumentacja dla tej klasy została wygenerowana z pliku:

Menu.cs

## Dokumentacja klasy MenuBoard

Klasu **MenuBoard** Odpowiada za kontrolę planszy w menu głównym

Dziedziczy MonoBehaviour.

### Pola danych

float **rotationSpeed**

### Opis szczegółowy

Klasu **MenuBoard** Odpowiada za kontrolę planszy w menu głównym

### Dokumentacja pól

#### float MenuBoard.rotationSpeed

#### Dokumentacja dla tej klasy została wygenerowana z pliku:

MenuBoard.cs

## Dokumentacja klasy Options

**Options** - Klasa zażądzająca opcjami gry

Dziedziczy MonoBehaviour.

### Metody publiczne

void **backToDefaultSettings** ()

void **setGameplayName** (string name)

string **getGameplayName** ()

void **setPawnColor** (PawnsColors color)

PawnsColors **getPawnColor** ()

void **setMusicVolume** (float musicVol)

float **getMusicVolume** ()

void **setSoundVolume** (float soundVol)

float **getSoundVolume** ()

void **setGraphicsQuality** (int gQ)

int **getGraphicsQuality** ()

### Pola danych

GameMode **gMode**

float **musicVolume**

*Opcje dźwięku*

float **soundsVolume**

GameObject **Music**

GameObject **Sounds**

int **graphicsQuality**

*Opcje grafiki*

int **boardStyle**

int **pawnStyle**

string **gameplayName**

PawnsColors **pawnColor**

const int **gamePort** = 25000

### Opis szczegółowy

**Options** - Klasa zażądzająca opcjami gry

### Dokumentacja pól

#### int Options.graphicsQuality

Opcje grafiki

#### float Options.musicVolume

Opcje dźwięku

#### Dokumentacja dla tej klasy została wygenerowana z pliku:

Options.cs

## Dokumentacja klasy Pawn

Klasa Pionka

Dziedziczy MonoBehaviour.

### Metody publiczne

void **unHighlightPawn** (GameObject pawn)

*Funkcja odswietlająca pionka*

void **move** (string fieldId)

*Funkcja rusza pionka*

void **putOut** ()

### Pola danych

GameObject **shadow**

int **pawnID**

*Identyfikator pionka*

string **fieldID**

*Identyfikator pola na którym stoi pionek*

PawnState **pState**

*Stan pionka*

bool **isSelected**

*Czy pionek jest zaznaczony*

### Opis szczegółowy

Klasa Pionka

### Dokumentacja funkcji składowych

#### void Pawn.move (string *fieldId*)

Funkcja rusza pionka

Szukamy pozycji na którą postawimy pionek

Przenosimy ten pionek na pozycję dest

#### void Pawn.unHighlightPawn (GameObject *pawn*)

Funkcja odswietlająca pionka

### Dokumentacja pól

#### string Pawn.fieldID

Identyfikator pola na którym stoi pionek

#### bool Pawn.isSelected

Czy pionek jest zaznaczony

#### int Pawn.pawnID

Identyfikator pionka

#### PawnState Pawn.pState

Stan pionka

#### Dokumentacja dla tej klasy została wygenerowana z pliku:

gameplay/Pawn.cs

## Dokumentacja klasy Player

Klasa Gracza

Dziedziczy MonoBehaviour.

### Metody publiczne

void **createPawns** ()

void **makeMove** (string idField)

*Funkcja wykonująca fizyczny ruch*

void **makeCapture** (string idOfDestField, int idOfCapturedPawn)

*Funkcja wykonująca bicie*

void **changeTurn** (**Pawn** pawn2Move)

void **waitForTurn** ()

### Pola danych

int **playerID**

GameObject[] **Pawns**

int **pawnToMove**

### Opis szczegółowy

Klasa Gracza

### Dokumentacja funkcji składowych

#### void Player.changeTurn (Pawn *pawn2Move*)

Trzeba jeszcze odznaczyć pionka po wykonanym ruchu

Zmiana tury

Reszte pól trzeba odznaczyć

#### void Player.createPawns ()

Tworzymy obiekt pionka z prefaba

Dodajemy skrypt dla pionka

Przypisujemy nazwę dla pionka

Skalujemy obiekt

Ustawiamy pionka na polu Przypisujemy ID Pionka dla pola

Dodajemy Box collider

Obracamy pionek o 45 stopni

Przypisujemy pionki dla playera oraz numery id dla pionków, oraz numery id pól na których stoją pionki

Zmieniamy stan pola na którym stawiamy pionka:

Dodajemy tag dla pionka

#### void Player.makeCapture (string *idOfDestField*, int *idOfCapturedPawn*)

Funkcja wykonująca bicie

Ruszamy pionkiem i ustawiamy mu nowe id pola na które ruszył

Zmieniamy status pola z którego ruszyliśmy

Zmieniamy status pola z którego ruszyliśmy

Trzeba zlikwidować pionka wroga podczas likwidacji trzeba także zmienić parametry pola z którego zbity jest pionek oraz parametry bitego pionka

Zerujemy wartość id pionka który stał na polu

Odkładamy pionek na bok

#### void Player.makeMove (string *idField*)

Funkcja wykonująca fizyczny ruch

Ruszamy pionkiem i ustawiamy mu nowe id pola na które ruszył

zmieniamy status pola na którego postawiliśmy pionka

Zmieniamy status pola z którego ruszyliśmy

#### Dokumentacja dla tej klasy została wygenerowana z pliku:

gameplay/Player.cs

## Dokumentacja klasy Tutorial

Klasa samouczka

Dziedziczy MonoBehaviour.

### Metody publiczne

void **BackToMenu** ()

*Funkcja powrotu do* ***Menu*** *Głównego*

### Opis szczegółowy

Klasa samouczka

### Dokumentacja funkcji składowych

#### void Tutorial.BackToMenu ()

Funkcja powrotu do **Menu** Głównego

#### Dokumentacja dla tej klasy została wygenerowana z pliku:

Tutorial/Tutorial.cs

# Podsumowanie

## Aplikacja działa stabilnie i poprawnie oraz spełnia wszystkie wymagania funkcjonalne oraz nie funkcjonalne - rozgrywka jest satysfakcjonująca

## Apliakacja zostałą umieszczona w sklepie Google Play w celu jej dystrybucji

## Aplikacja działa na systemach Android 4.0.3 i nowszych

## Aplikacja została wykonana w środowisku Unity , wykorzystywany był język programowania C#

## Testy zostały przeprowadzone pomyślnie, napotkane błędy usunięte,

## Odpowiednie diagramy i tabele zostały dodane w załączniku w celu zwiększenia ich czytelności

# Indeks

areThereCaptures

Board, 6

backToMainMenu

Gameplay, 12

BackToMenu

Tutorial, 20

Board, 6

areThereCaptures, 6

getDarkFieldsList, 7

getFieldCenterCoordinate, 7

getFieldIDinLine, 7

getFieldState, 7

getPawnIDonField, 7

getSurroundingFields, 7

getSurroundingFieldsWithEnemies, 7

isItFront, 7

searchForCaptures, 7

setFieldState, 8

TranslateCords, 8

ButtonsControl, 8

PlayClickSound, 8

CameraPerspective, 9

setFreeView, 9

setTopView, 9

changeTurn

Gameplay, 12

Player, 18

CreateGame

Menu, 14

CreateGameOnOneDevice

Menu, 14

createPawns

Player, 18

Cursor, 9

ExitApplication

Menu, 14

Field, 10

fState, 10

idField, 10

pawnId, 10

FieldCords, 11

fieldID

Pawn, 17

fState

Field, 10

Gameplay, 11

backToMainMenu, 12

changeTurn, 12

getEnemyPlayer, 12

onTurnStart, 12

panelGameEnd, 13

showResults, 12

whoseTurnID, 13

getDarkFieldsList

Board, 7

getEnemyPlayer

Gameplay, 12

getFieldCenterCoordinate

Board, 7

getFieldIDinLine

Board, 7

getFieldState

Board, 7

getPawnIDonField

Board, 7

getSurroundingFields

Board, 7

getSurroundingFieldsWithEnemies

Board, 7

graphicsQuality

Options, 16

HintWindow, 13

idField

Field, 10

Info

Menu, 14

InfoExit

Menu, 14

isItFront

Board, 7

isSelected

Pawn, 17

JoinGame

Menu, 14

makeCapture

Player, 19

makeMove

Player, 19

Menu, 13

CreateGame, 14

CreateGameOnOneDevice, 14

ExitApplication, 14

Info, 14

InfoExit, 14

JoinGame, 14

StartTutorial, 15

MenuBoard, 15

rotationSpeed, 15

move

Pawn, 17

musicVolume

Options, 16

onTurnStart

Gameplay, 12

Options, 15

graphicsQuality, 16

musicVolume, 16

panelGameEnd

Gameplay, 13

Pawn, 16

fieldID, 17

isSelected, 17

move, 17

pawnID, 17

pState, 18

unHighlightPawn, 17

pawnId

Field, 10

pawnID

Pawn, 17

PlayClickSound

ButtonsControl, 8

Player, 18

changeTurn, 18

createPawns, 18

makeCapture, 19

makeMove, 19

pState

Pawn, 18

rotationSpeed

MenuBoard, 15

searchForCaptures

Board, 7

setFieldState

Board, 8

setFreeView

CameraPerspective, 9

setTopView

CameraPerspective, 9

showResults

Gameplay, 12

StartTutorial

Menu, 15

TranslateCords

Board, 8

Tutorial, 19

BackToMenu, 20

unHighlightPawn

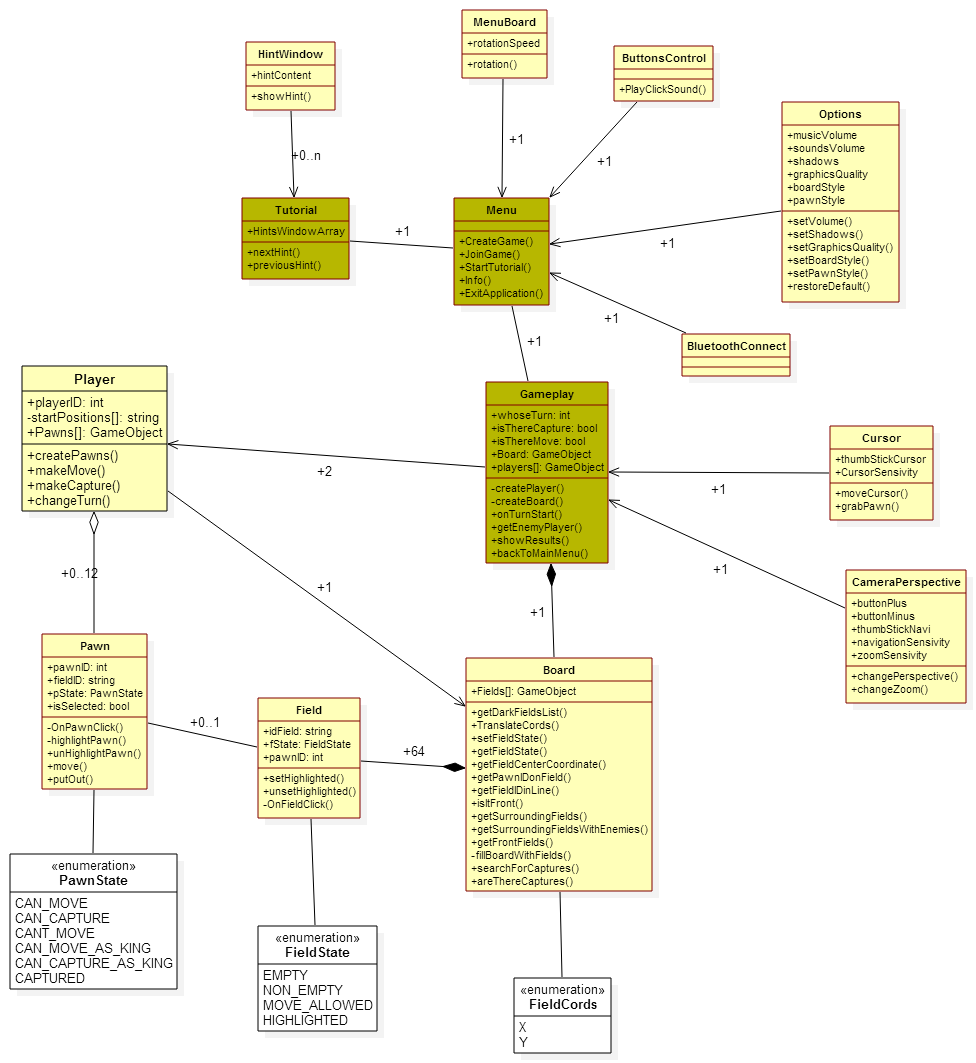
Pawn, 17

whoseTurnID

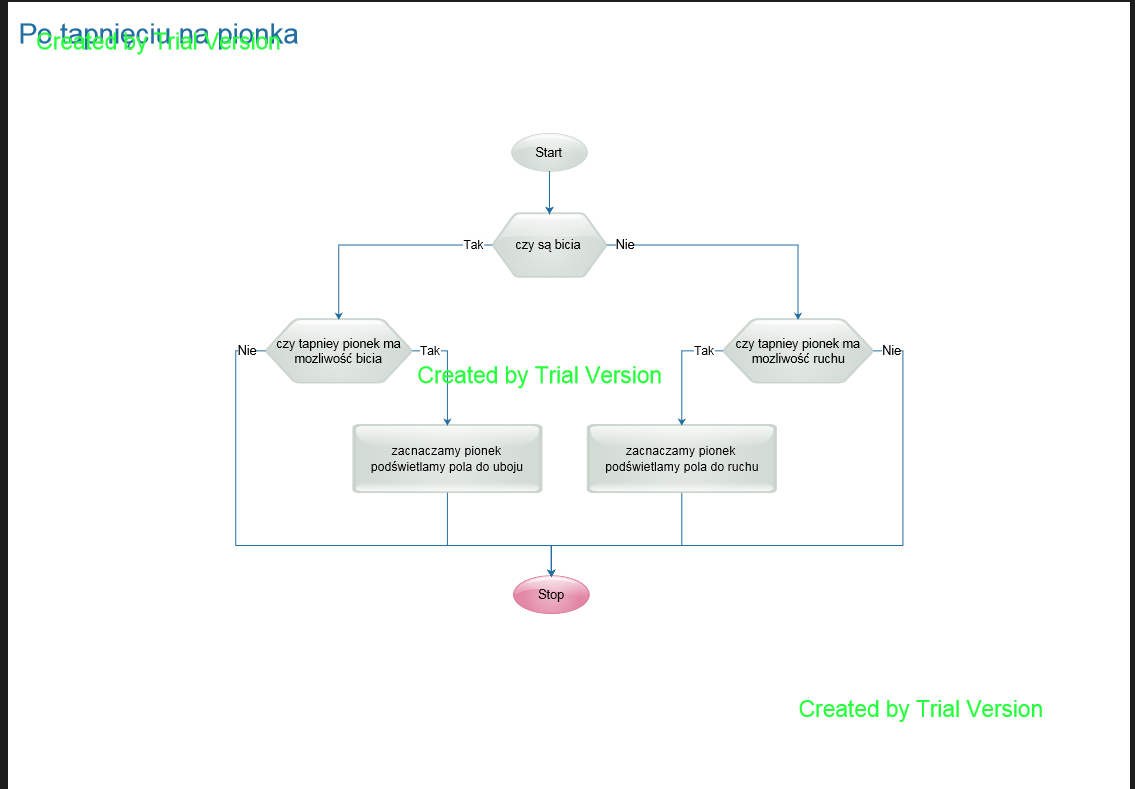
Gameplay, 13

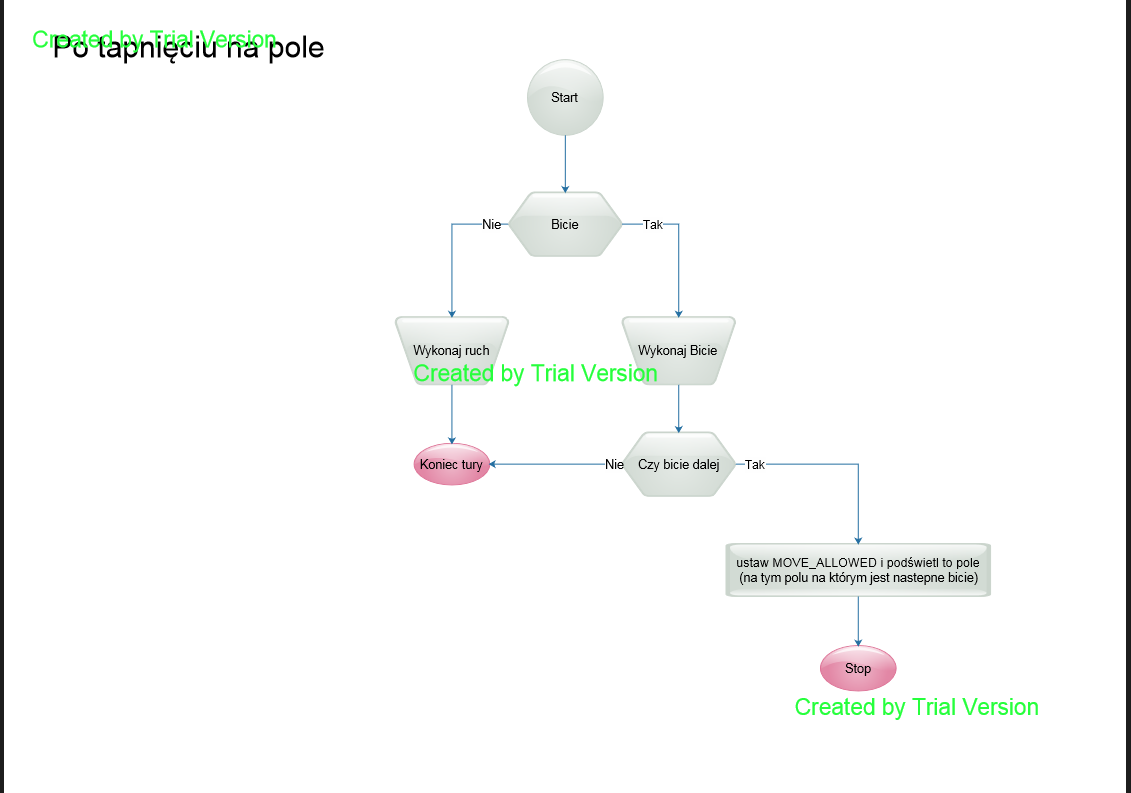
# Załączniki

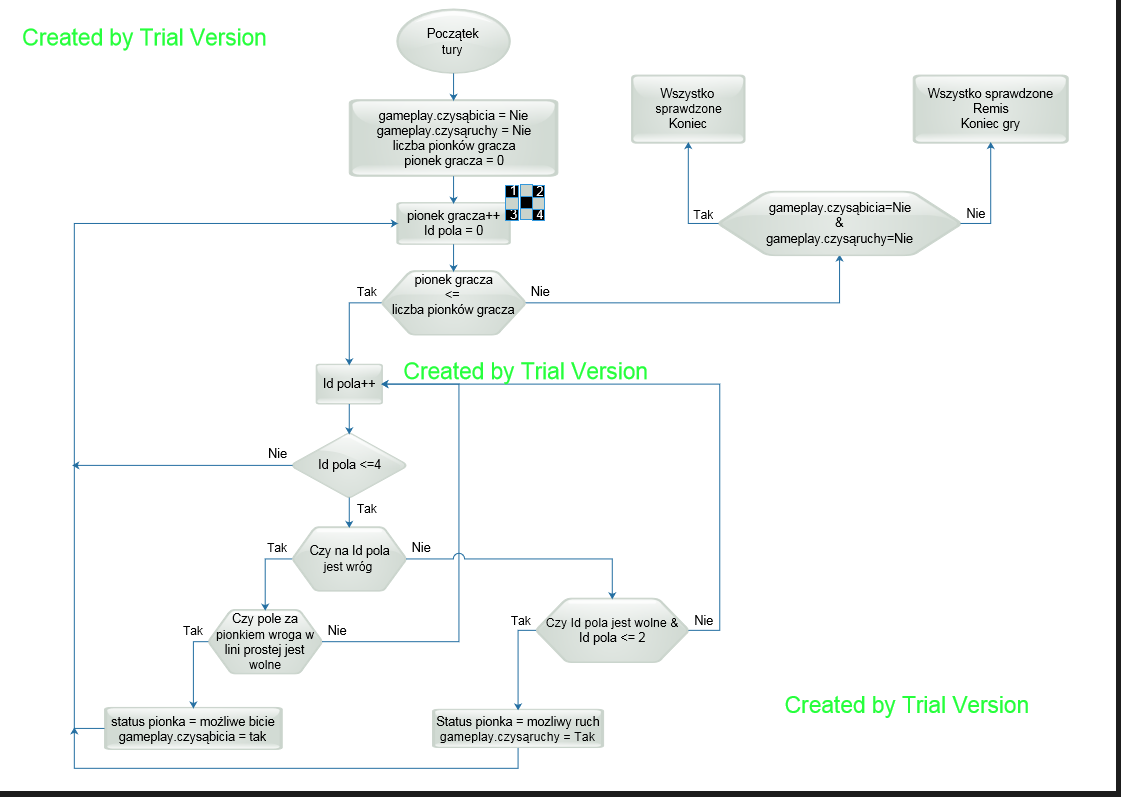
## Diagram Klas



## Schematy blokowe użyte podczas projektowania aplikacji:







## Diagram Przepływu danych:

## Tabele testów: