

Dokumentacja Projektu

Tytuł: Cube Checkers 3D  
Wersja projektu: 1.0

Przedmiot:Programowanie  
Grupa projektowa: 11 (sl02)

Prowadzący:  
Dr Marek Jaszuk

Wykonawcy:  
Oleh Korsunskyi  
Nazarii Korniienko  
Krzysztof Pudysz

Rzeszów, 2019

TREŚĆ PROJEKTU

[0. Całkowite wymagania projektowe 3](#_Toc11830694)

[1. Założenie projektowe (wymagania biznesowe) 4](#_Toc11830695)

[2. Lista wymagań z podziałem na wymagania 5](#_Toc11830696)

[2.1 Wymagania funkcjonalne 5](#_Toc11830697)

[2.2 Wymagania niefunkcjonalne 5](#_Toc11830698)

[3. Diagram przypadków użycia z wyróżnieniem aktorów 6](#_Toc11830699)

[4. Harmonogram realizacji projektu na diagramie Gantta 7](#_Toc11830700)

[5. Opis techniczny projektu 8](#_Toc11830701)

[5.1 CheckersBoard 8](#_Toc11830702)

[5.2 Piece 9](#_Toc11830703)

[6. Prezentacja warstwy użytkowej projektu 10](#_Toc11830704)

[7. System kontroli wersji 11](#_Toc11830705)

[8. Materiały źródłowe 11](#_Toc11830706)

[9. Dokumentacja w Html Doxygen 12](#_Toc11830707)

# 0. Całkowite wymagania projektowe

1. Założenie projektowe (wymagania biznesowe).
2. Specyfikacja wymagań – lista wymagań z podziałem   
   na wymagania funkcjonalne i niefunkcjonalne.
3. Diagram przypadków użycia z wyróżnieniem aktorów projektu oraz  
   przypadków użycia odzwierciedlających wymagania funkcjonalne.
4. Harmonogram realizacji projektu typu diagram Gantta.  
   W przypadku projektu realizowanego zespołowo należy  
   przypisać poszczególne zadania do członków zespołu.
5. Opis techniczny projektu – w jaki sposób projekt  
   został zrealizowany (struktura kodu programu).
6. Prezentacja warstwy użytkowej projektu   
   (widoki ekranu z opisem ich użytkowania).
7. Projekt powinien być realizowany z wykorzystaniem wybranego   
   systemu kontroli wersji. Należy podać link do repozytorium.
8. Materiały źródłowe – wskazanie literatury i materiałów źródłowych  
   wykorzystanych przy realizacji projektu (w tym tutoriale internetowe).
9. Do kodu programu powinna być stworzona dokumentacja poprzez  
   system komentarzy dokumentujących. Załącznikiem dokumentacji   
   projektu powinna być dokumentacja kodu w postaci Html Doxygen.  
   Wygenerowaną dokumentację należy udostępnić poprzez dysk   
   internetowy podając link w niniejszym dokumencie.

# 1. Założenie projektowe (wymagania biznesowe)

Głównym założeniem naszego projektu jest tworzenie gry komputerowej Warcaby 3D. Program będzie przypominać zwykłe Warcaby Klasyczne z planszą, która składa się z 64 pól, ale z nowym, wymyślonym przez nas designem. Gra będzie przeznaczona dla dwóch użytkowników, którzy potrafią grać na jednym komputerze. Design figur będzie w stylu geometrycznym, a mówiąc dokładniej, poszczególne figury będą sześcianami.

Gra zostanie zbudowana na silniku Unity, który idealnie pasuje do tworzenia prostych gier 3D. Poszczególne modele 3D, które będą wykorzystywane w tym projekcie, zostaną zrobione w programie 3D MAX. Programem dla napisania kodu jest Visual Studio, który łatwo się łączy z Unity. Cały projekt będzie napisany w języku programowania C#. Harmonogram (diagram Gantta) zostanie zrealizowany w środowisku Hygger. Projekt będzie realizowany na podstawie systemu kontroli wersji – GitHub.

Głównym celem tego projektu jest zrozumienie poszczególnych kroków tworzenia aplikacji desktopowych w oparciu o silnik Unity, znalezienie i rozwiązanie problemów, które pojawią się w czasie tworzenia gry. Dodatkowym celem jest nauczyć się pracować w zespole i pomagać sobie nawzajem.

# 2. Lista wymagań z podziałem na wymagania

## 2.1 Wymagania funkcjonalne

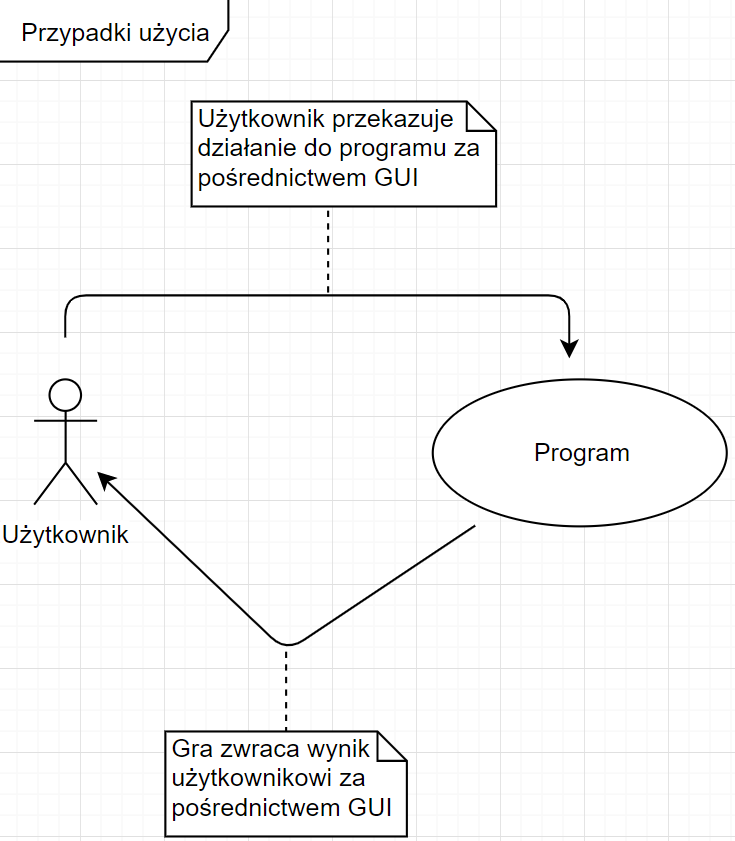
* Program ma udostępniać całą funkcjonalność gry Warcaby Klasyczne (tzn. przestrzegać wszystkich zasad tej gry).
* Program ma być wykonany jako gra 3D i umożliwiać spojrzenie na planszę pod różnym kątem, kąt spojrzenia musi być ustalony z góry przed rozpoczęciem gry, i nie może zostać zmieniony podczas rozgrywki.
* Program ma zapewniać kolejność ruchów każdego z użytkowników.
* Jeżeli jakaś strona zwycięża, to program ma zabraniać robienie jakichkolwiek ruchów na planszy.
* Użytkownik musi mieć możliwość przesuwania figur za pomocą myszy komputerowej na dostępne pola, ale jednocześnie program ma zabraniać użytkownikowi robienie nieprawidłowych ruchów zgodnie z zasadami gry.

## 2.2 Wymagania niefunkcjonalne

* Program ma być wykonany w stylu figur geometrycznych.
* Plansza i figury mają być przedstawione w dwóch kolorach.
* Program ma powstać na silniku Unity3D.
* Interfejsem programu ma być GUI.
* Program jest przeznaczony dla dwóch użytkownik.
* Program ma działać na systemie Windows.
* Dodatkowe oprogramowanie - .Net Framework 4.6.1 lub nowszy.

# 3. Diagram przypadków użycia z wyróżnieniem aktorów

Use case to opis zbioru sekwencji działań systemu oraz wariantów takich sekwencji, w wyniku których uzyskuje się obserwowalny wynik, który ma pewną wartość dla niektórych z uczestników procesu.

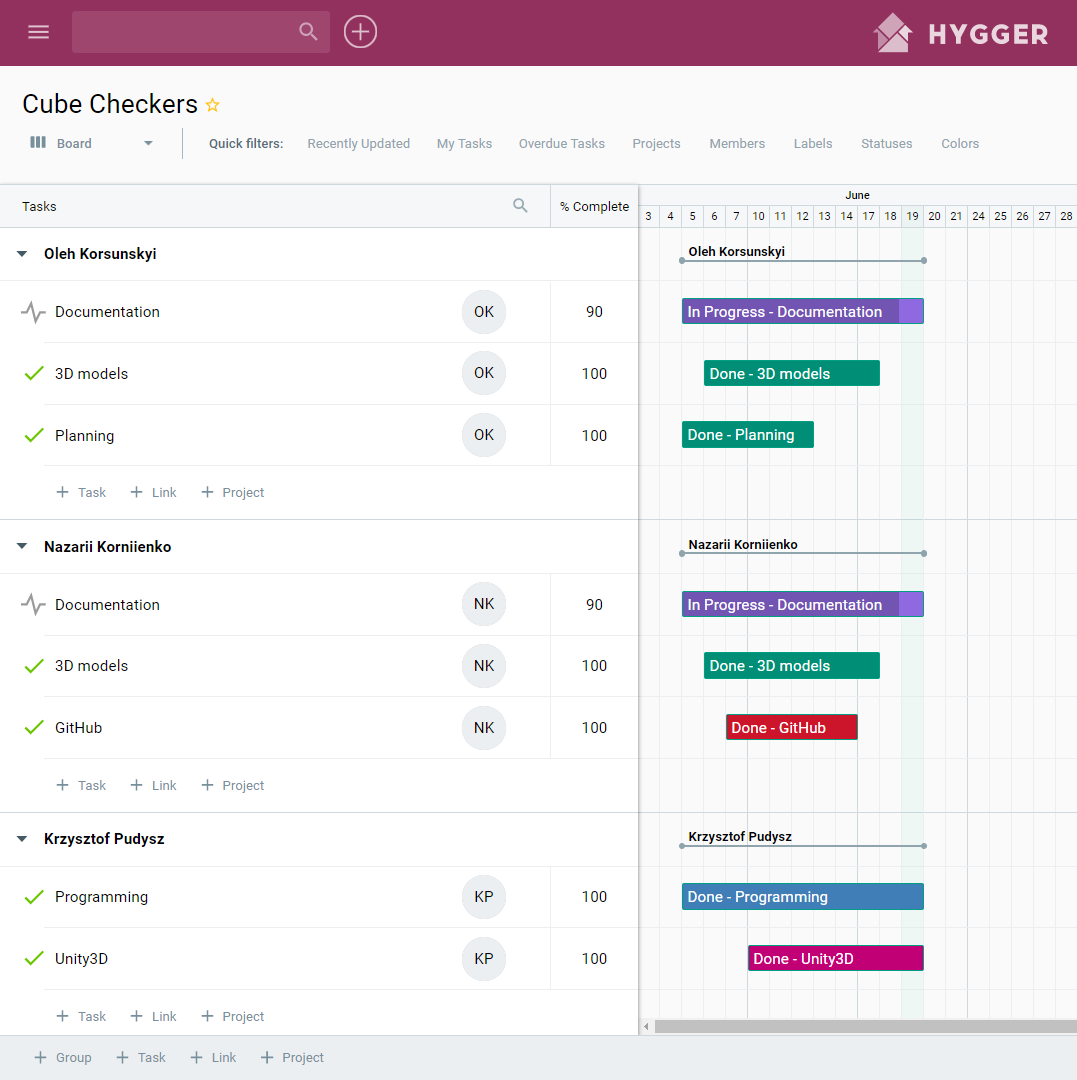


# 4. Harmonogram realizacji projektu na diagramie Gantta

Wykres Gantta to wizualny sposób wyświetlania zaplanowanych zadań.

Grafika pozioma jest szeroko stosowana do planowania projektów o dowolnej wielkości w różnych branżach i branżach. Jest to wygodny sposób na pokazanie, jakie prace mają być wykonane w określonym dniu i czasie.

Wykresy Gantta pomagają również zespołom i kierownikom projektów monitorować daty rozpoczęcia i zakończenia każdego projektu. Wszystko w jednym miejscu.



# 5. Opis techniczny projektu

Projekt składa się z dwóch klas, CheckersBoard oraz Piece. Klasa CheckersBoard składa się z 16 metod, natomiast klasa Piece z 2 metod.

## 5.1 CheckersBoard

Pierwsza metoda to Start. Znajdują się tu wszystkie informacje odnośnie tego co ma się stać po włączeniu programu.

Druga metoda to Update. Znajdują się tu informacje na temat przesuwania figur na planszy oraz możliwość zaznaczania figury i przesunięcia jej w inne miejsce. Ta funkcja również zmienia turę graczy.

Trzecia metoda to UpdateMouseOver. Ta metoda śledzi położenie kursora myszy za pomocą RaycastHit. Z punktu, w którym znajduje się kursor myszy jest wysyłana wiązka laserowa i po kolizji z planszą, na którą jest naniesiony BoxColider odczytywana jest wartość, gdzie znajduje się kursor myszy.

Czwarta metoda to UpdatePieceDrag. Ta metoda również używa RaycastHit, tym razem do przenoszenia figur. Bez tej metody przemieszczanie figur za kursorem myszy byłoby niemożliwe.

Piąta metoda to SelectPiece. Przyjmuje argumenty x, y i jest to położenie figur na planszy. Plansza składa się z pól X od 0 do 8 leżących w poziomie oraz z pól Y od 0 do 8 leżących w pionie. Sprawdzane tu jest czy gracz nie próbuje wyciągnąć jakiejś figury za dozwolone pola planszy oraz czy gracz musi ruszyć jakąś figurą, która ma zbić inną. Sprawia to, że nie może wykonać ruchu innymi figurami.

Szósta metoda to TryMove. Znajdują się w niej głównie informacje na temat ruchu figur. Jeżeli figura nie ruszyła się, to ma wrócić na swoje dawne miejsce. Sprawdza też czy dany ruch jest dozwolony oraz czy jakaś figura nie została zbita podczas ruchu.

Siódma metoda to EndTurn. Sprawdzane jest w niej czy jakaś figura została zbita i czy może wykonać kolejny ruch, jeżeli nie to kończy turę gracza. Również tutaj odbywają się promocje po dojściu figury na koniec planszy – figury mogą poruszać się wtedy również w tył.

Ósma metoda to CheckVictory. Sprawdza czy któryś gracz wygrał.

Dziewiąta metoda to Victory. Jest pomocnicza do poprzedniej metody, wypisuje na ekranie który gracz wygrał rozgrywkę.

Dziesiąta i jedenasta metoda to ScanForPossibleMove. Jedna z metod jest przeciążona. To tutaj dodawane są do listy figury, które muszą wykonać ruch.

Dwunasta metoda GenerateBoard tworzy figury na planszy. Najpierw dla białej drużyny a następnie dla czarnej.

Trzynasta metoda to GeneratePiece robi to samo co dwunasta metoda z tą różnicą, że tworzy modele figur w miejscu, gdzie mają się znajdować.

Czternasta metoda to MovePiece. Na podstawie informacji co to za figura, ustawia ją w odpowiednim położeniu tak, aby na początku gry znalazła się na swoim polu.

Piętnasta metoda to Highlight. Podświetla figury, które muszą wykonać ruch, co sprawia, że gracz łatwiej może zobaczyć figury którymi ma się poruszyć

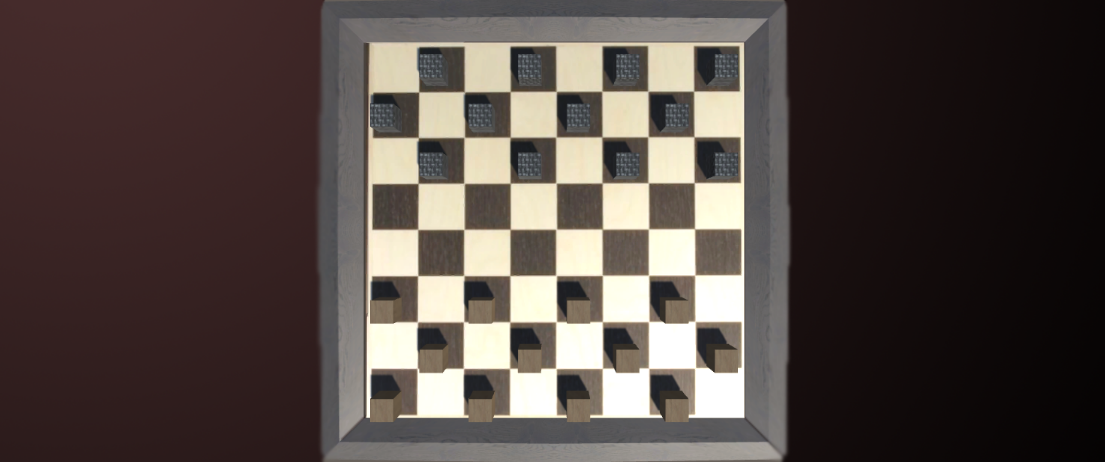
Szesnasta metoda to Alert. Powiadamia gracza o tym czyja jest tura.

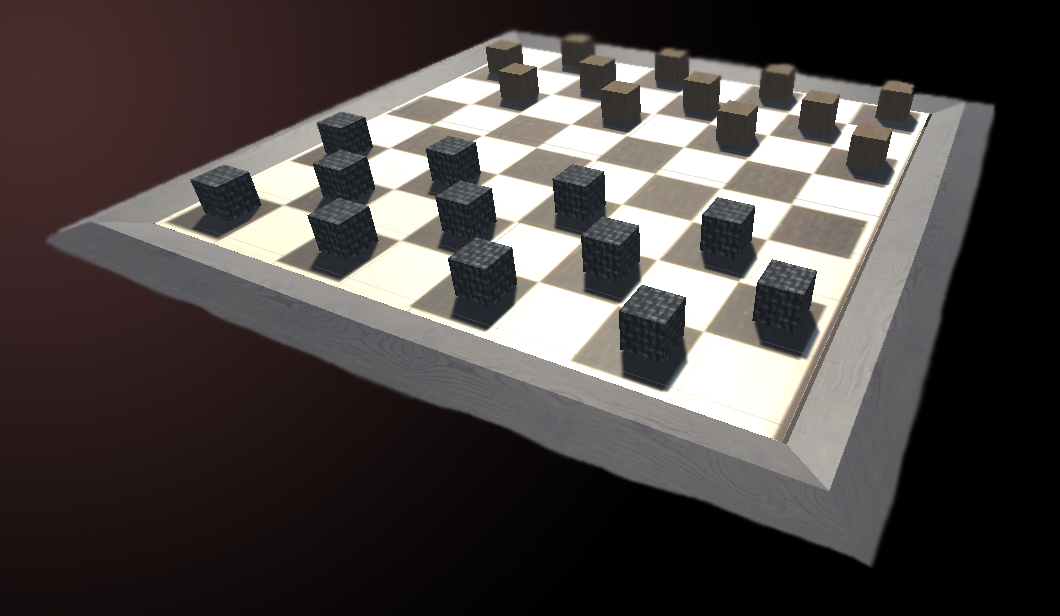
## 5.2 Piece

Pierwsza metoda to IsForceToMove. Dzieli planszę na cztery części w których sprawdzane są warunki czy figura obok nas nie jest tego samego koloru oraz czy miejsce, w które chcemy się przenieść po zbiciu figury jest wolne.

Druga metoda to ValidMove. Znajduje się w niej upewnienie, że gracz nie może przenieść figury na inną figurę oraz warunki czy ruch, który został wykonany jest dozwolony czy też nie.

# 6. Prezentacja warstwy użytkowej projektu





Użytkownik może przestawiać figury za pomocą myszy komputerowej, ma nacisnąć na figurę i przeciągnąć w dostępne pole.

# 7. System kontroli wersji

[https://github.com/uitmers/checkers](https://github.com/uitmers/checkers.git)

# 8. Materiały źródłowe

1. G. B. Jeremy: „PROJEKTOWANIE GIER PRZY UŻYCIU ŚRODOWISKA UNITY I JĘZYKA C#. OD POMYSŁU DO GOTOWEJ GRY” (Wydawnictwo Helion);
2. <https://www.youtube.com/playlist?list=PLLH3mUGkfFCVXrGLRxfhst7pffE9o2SQO> (tutorial na YouTube).

# 9. Dokumentacja w Html Doxygen

-