|  |  |
| --- | --- |
|  | **AKADEMIA GÓRNICZO-HUTNICZA** |

Dokumentacja do projektu

**Connect Four**

z przedmiotu

**Języki programowania**

Elektronika, 1. rok

*Kacper Filipek*

czwartek 14:40

prowadzący: Rafał Frączek

tutaj wpisać datę wykonania dokumentacji

# Opis projektu

Projekt jest realizacją gry w „Czwórki” (ang. „Connect Four”) w C++, z interfejsem TUI (t.j. interfejs używający znaków specjalnych i kolorów do rysowania interfejsu użytkownika w konsoli).

# Project description

The project is a realisation of the game “Connect Four” made in C++ with TUI interface (an interface using special characters for drawing UI in the terminal)

# Instrukcja użytkownika

W tym punkcie należy umieścić instrukcję użytkowania programu. Może być to na przykład opis poszczególnych menu w programie. W przypadku gry należy opisać zasady gry. Opcjonalnie można wstawić zrzuty ekranu. Jeśli uruchomienie programu wymaga wykonania jakiś niestandardowych lub dodatkowych czynności (na przykład uruchomienie serwera baz danych itp.) to należy zamieścić tę informację.

# Kompilacja

Program został napisany na systemy operacyjne z rodziny Linux, chociaż powinien on działać na Windowsie, ponieważ kod nie używa żadnych zależnych od platformy plików nagłówkowych. Program używa systemu CMake do budowania projektu. Można skompilować go na dwa sposoby:

* Sposób 1:

1. Wejść do folderu build/
2. Wykonać polecenie cmake ..

* Sposób 2:

1. Z katalogu głównego wywołać skrypt ./bld.sh. Ze względu na fakt, że skrypt wywołuje program make, może on nie działać na Windowsie.

Po zbudowaniu powienien się plik wykonywalny o ./build/connect-four. Z uwagi na fakt, że ścieżki do zasobów są wpisane w programie relatywnie do głównego katalogu, to program wykonywalny powinien z niego wywoływany.

# Pliki źródłowe

W tym punkcie należy opisać wszystkie pliki źródłowe (.cpp, .h) w projekcie. Należy podać nazwę każdego pliku oraz informację o tym co się w nim znajduje. Na przykład:

Projekt składa się z następujących plików źródłowych:

* *board.h*, *board.cpp* – deklaracja oraz implementacja klasy Board,
* *game.h*, game*.cpp* – deklaracja oraz implementacja klasy Game,
* *menu.h*, menu*.cpp* – deklaracja oraz implementacja klasy Menu.
* *extras.h*, extras*.cpp* – deklaracja oraz implementacja funkcji pomocniczych.
* *color.h* – definicje procesora nazw kolorów do użycia w funkcjach biblioteki ncurses.
* *main.cpp* – główny plik z implementacją funkcji main.

# Zależności

W projekcie wykorzystano następujące dodatkowe biblioteki:

* ncurses – biblioteka do sterowania interakcji z emulatorem terminala, pozwala tworzyć zaawansowane interfejsy konsolowe: <https://invisible-island.net/ncurses/>

# Opis klas

W tym punkcie należy umieścić opis wszystkich stworzonych w projekcie klas. Należy podać do czego służy dana klasa oraz informację o jej publicznych metodach. Opcjonalnie można załączyć fragmenty kodu źródłowego. Na przykład:

W projekcie utworzono następujące klasy:

* BankAccount – reprezentuje konto w systemie bankowym.
  + float get\_balance(void) – zwraca bieżący stan konta,
  + void withdraw(float ammount) – pobiera z konta sumę ammount,
* Globals – klasa zawierająca publiczne oraz statyczne metody i zmienne.
  + void save(const string& file\_path) – zapisuje bieżący stan programu,
  + int num\_accounts – całkowita liczba kont w systemie.
* Utils – klasa zawierająca rożne przydatne funkcje.
  + bool contains(const string& str, const string& key) – zwraca true gdy str zawiera key.
  + vector<string> split(const string& str, const string& delim) – dzieli str na składniki rozdzielone przez delim.

# Zasoby

Jeśli projekt wykorzystuje jakieś dodatkowe zasoby jak na przykład pliki z danymi tekstowymi, pliki obrazów itp. to w tym punkcie należy je wyszczególnić. W przypadku plików tekstowych konieczne jest opisanie struktury takiego pliku. Jeśli w projekcie nie ma żadnych zasobów to piszemy słowo „brak”. Na przykład:

W projekcie wykorzystywane są następujące pliki zasobów:

* assets/ – katalog zawierający dodatkowe zasoby do gry. Struktura katalogu:
  + logo1.txt, logo2.txt – pliki zawierające tekstowe logo pojawiające się na ekranie startowym.

# Dalszy rozwój i ulepszenia

W tym punkcie podajemy możliwe dalsze kierunki rozwoju projektu. Można podać na przykład kolejne funkcjonalności, które mogłyby być zaimplementowane w projekcie.

# Inne

W tym punkcie podajemy opcjonalnie inne informacje nieuwzględnione w poprzednich punktach. W przypadku braku dodatkowych informacji wpisujemy słowo „brak”.