AKADEMIA GÓRNICZO-HUTNICZA

Dokumentacja do projektu

ToDo List

z przedmiotu

Języki programowania obiektowego

Elektronika i Telekomunikacja, rok 3

Miłosz Janik i Antoni Kijania

piątek 12:50

prowadzący: Rafał Frączek

21.01.2021 r.

1. Opis projektu

Program utworzony w ramach tego projektu umożliwia tworzenie i modyfikację prostych list zawierających zadania do wykonania, zapisywanych lokalnie w oddzielnych plikach tekstowych. Zapewnia on także dodatkowe funkcjonalności manipulacji danych takie jak oznaczanie zadań jako wykonanych, usuwanie plików, dodawanie nowych zdań do wcześniej utworzonych plików. Wykonany z zachowaniem ogólnie przyjętych paradygmatów programowania obiektowego w języku C++.

2. Project description

Program created in the scope of the project was designed to allow creation of simple lists containing tasks to do, saved in separate local text files. It also provides additional functionality for data manipulation such as marking tasks as done, deleting files and adding new tasks to already existing lists. Made with following generally accepted, object oriented C++ programming paradigms.

3. Instrukcja użytkownika

Po uruchomieniu programu, należy zadecydować, czy zamierzamy używać pliku zapisu notatek innego niż domyślny “todo_default.txt”. Pozwoli to na tworzenie wielu list zadań - np. do pracy i do domu.
W przypadku wybrania pliku innego niż domyślny, program poprosi o podanie nazwy wybranej listy.
Jeżeli lista domyślna lub wybrana przez użytkownika nie istnieje, program poinformuje o tym i poprosi o jej uzupełnienie - należy wprowadzać kolejno każde zadanie i zatwierdzić, wprowadzając pustą linię.
Jeżeli lista domyślna lub wybrana przez użytkownika istnieje, program wyświetli jej zawartość oraz główne menu programu.

W głównym menu programu, dostępne są 4 opcje wyboru:

a) “Dodaj zadanie” - pozwala dopisać zadania do bieżącej listy.
b) “Potwierdź ukończenie zadania” - pozwala zaznaczyć, które zadania bieżącej listy zostały już ukończone.
c) “Wyczyść listę” - usuwa bieżącą listę i jej plik, a następnie umożliwia stworzenie nowej listy o tej samej nazwie.
d) “Zakończ i wyjdź” - zamyka program. Zmiany są na bieżąco zapisywane podczas działania programu

4. Kompilacja

Nie jest wymagana kompilacja niestandardowa, działanie programu nie jest ograniczone do jednej platformy. Wymagany jest jedynie terminal ze standardowym kompilatorem np. gcc.

5. Pliki źródłowe

Program nie posiada własnych plików źródłowych. w całości zawiera się w jednym pliku o rozszerzeniu .cpp.

6. Zależności

- iostream - obsługa strumieni wejścia/wyjścia.
- fstream - obsługa strumieni do zapisu plików.
- vector - obsługa kontenera typu vector.
- iomanip - obsługa manipulatorów do formatowania danych wyjściowych.

7. Opis klas

W projekcie utworzono następujące klasy:

a) **`Item`** – reprezentuje daną czynność do wykonania

- `Item(string)` - tworzenie nowego zadania (lub wczytanie z pliku)
- `string text()` - zwraca zawartość tekstową zadania,
- `void done())` - ustawia pole _done na true - zadanie wykonane
- `bool is_done()` - zwraca stan wykonania zadania

b) **`TodoList`** - reprezentuje listę zadań do wykonania, operuje na wektorze zadań oraz własnym pliku.

- `TodoList(const char* filename))` - tworzenie nowej listy zadań
- `void set_filename(const char* filename)` - ustawianie nazwy pliku używanego przez listę
- `void create()` - czyszczenie listy i tworzenie nowej
- `void read()` - wczytanie listy z pliku (po uruchomieniu programu)
- `void show()` - wyświetlanie wektora listy zadań
- `void save()` - zapisywanie (aktualizowanie) do pliku wektora listy zadań
- `void add()` - dopisywanie nowych zadań do listy
- `void check()` - metoda ustawiająca stan wybranego zadania na wykonane
- `void destroy_file()` - metoda czyszcząca wektor zadań i usuwająca plik
- `int get_count()` - zwraca rozmiar listy

8. Zasoby

Program tworzy pliki zawierające oddzielne listy zadań do wykonania. Nazwa każdego z nich jest podawana przez użytkownika. Użytkownik ma wybór użycia domyślnej nazwy i rozszerzenia “todo_default.txt”. Możliwy jest także wybór innego rozszerzenia w trakcie podawania nazwy pliku. Każda linijka w pliku zawiera nazwę jednego zadania oraz spokrewniony z nim, oddzielony jedną spacją tekst *true/false*, mówiący o tym czy zadanie to zostało oznaczone jako wykonane, czy też nie. Uniemożliwione zostało nadanie nazwy zadania zawierającej ciągi znaków *true* lub *false*. Każdemu zadaniu, domyślnie przy tworzeniu nadawany jest stan wykonania *false*.

9. Dalszy rozwój i ulepszenia

Program można rozwinąć o wprowadzanie dodatkowych danych takich jak data przewidywanego wykonania zadania z automatycznym obliczaniem pozostałego czasu, podział zadań na kategorie w ramach danego pliku lub inne. Dodatkowe proponowane funkcjonalności to synchronizacja list zadań z chmurą, obsługa importu/eksportu list innych użytkowników, automatyczne tworzenie archiwum ukończonych zadań z opcją odzyskiwania przypadkowo usuniętych wpisów.

10. Inne

Brak.