AKADEMIA GÓRNICZO-HUTNICZA

Dokumentacja do projektu

Command line interpreter

z przedmiotu

Języki programowania obiektowego

Elektronika 2 rok

Jakub Konior

czwartek 14:40

prowadzący: Rafał Frączek

21.01.2024

1. Opis projektu

Projekt ten to system, który umożliwia interakcję z komputerem poprzez wprowadzanie i wykonywanie poleceń. Celem tego projektu jest stworzenie inteligentnego narzędzia, które może zrozumieć polecenia użytkownika w formie zdań lub fraz, interpretować ich znaczenie i odpowiednio reagować na te polecenia.

2. Project description

This project is a system that allows you to interact with a computer by entering and executing commands using natural language. The goal of this project is to create an intelligent tool that can understand user commands in the form of sentences or phrases, interpret their meaning and respond appropriately to these commands.

3. Instrukcja użytkownika

Aplikacja działa podobnie do konsoli w systemach Windows, jednakże ma mniej poleceń, które są też mniej rozbudowane. Po włączeniu aplikacji ukazuje się okienko z aktualną ścieżką. Program czeka na wporwadzenie przez użytkownika polecenia i odpowiednio go interpretuje. Jest 6 różnych poleceń:

CD Wyświetla nazwę lub zmienia bieżący katalog.

MKDIR Tworzy katalog.

RMDIR Usuwa katalog.

DIR Wyświetla listę plików i podkatalogów w katalogu.

COPY Kopiuje jeden lub więcej plików do innej lokalizacji.

HELP Zawiera informacje pomocy dotyczące poleceń systemu Windows.

4. Kompilacja

Wystarczy standardowa kompilacja. Program działa tylko w systemie Windows.

5. Pliki źródłowe

Projekt składa się z następujących plików źródłowych:

- *CommandLineInterpreter.cpp* – główny plik projektu odpowiedzialny za sterowanie danymi wejściowymi,
- *Command.h*, *Command.cpp* – klasa odpowiedzialna za przechowywanie informacji o poleceniach,
- *Parser.h*, *Parser.cpp* – klasa odpowiedzialna za interpretacje danych wejściowych,

6. Zależności

W projekcie wykorzystano następujące dodatkowe biblioteki:

- filesystem – Biblioteka Filesystem zapewnia narzędzia do wykonywania operacji na systemach plików i ich komponentach, takich jak ścieżki, zwykłe pliki i katalogi. Strona internetowa: https://en.cppreference.com/w/cpp/filesystem

7. Opis klas

W projekcie utworzono następujące klasy:

1. Klasa Command reprezentująca polecenie posiada metody:
    - ICommand:

Jest to interfejs definiujący jedną czysto wirtualną funkcję execute().Jest abstrakcyjną klasą bazową dla wszystkich poleceń.

- PrintPath:

Klasa dziedzicząca po ICommand. Implementuje funkcję execute(), która wypisuje ścieżkę bieżącego katalogu na standardowe wyjście.

- ChangeDirectory:

Klasa dziedzicząca po ICommand. Implementuje funkcję execute(), która zmienia bieżący katalog na określony.

- PrintDirectory:

Klasa dziedzicząca po ICommand. Implementuje funkcję execute(), która wypisuje zawartość bieżącego katalogu na standardowe wyjście.

- CreateDirectory:

Klasa dziedzicząca po ICommand. Implementuje funkcję execute(), która tworzy nowy katalog o określonej nazwie.

- CopyFile:

Klasa dziedzicząca po ICommand. Implementuje funkcję execute(), która kopiuje plik z określoną ścieżką źródłową do określonej ścieżki docelowej.

- RemoveDirectory:

Klasa dziedzicząca po ICommand. Implementuje funkcję execute(), która usuwa katalog o określonej nazwie.

- Help:

Klasa dziedzicząca po ICommand. Implementuje funkcję execute(), która wyświetla pomoc dotyczącą określonej komendy.

2. Klasa Parser interpretująca polecenia posiada metody:

Parser(const string& command): Konstruktor, który przyjmuje komendę jako argument i inicjuje pole fcommand tą wartością.

void setCommand(string command): Metoda pozwalająca na ustawienie nowej wartości dla pola fcommand.

string getCommand() const: Metoda zwracająca aktualną wartość pola fcommand.

string getKeyWord(string command): Metoda przyjmująca komendę i zwracająca jej pierwsze słowo kluczowe (czyli "słowo" do pierwszej spacji).

string getFirstContent(string command): Metoda przyjmująca komendę i zwracająca jej pierwszą zawartość (wszystko po pierwszym słowie kluczowym).

string getSecondContent(string command): Metoda przyjmująca komendę i zwracająca jej drugą zawartość (wszystko po drugim słowie kluczowym).

bool isACommand(string command): Metoda sprawdzająca, czy podana komenda jest uznawana za polecenie. Może to być przydatne do różnicowania komend od innych ciągów znaków.

8. Zasoby

- brak

9. Dalszy rozwój i ulepszenia

Dodanie większej ilości poleceń.

Dodanie możliwości korzystania z przełączników.

Lepsze formatowanie wyświetlanych komunikatów.

10. Inne

brak