Documentation for File Operations Console Application

Overview

The File Operations console application is a utility program designed to manage and manipulate files and directories through a command-line interface (CLI). It provides users with a set of commands to create, delete, read, write, and navigate through the file system. The application ensures an interactive and structured approach to performing file operations while maintaining user-friendly error handling and feedback.

Features and Functionalities

File Commands

1. Create File

- **Command:** `create-file`
- Prompts the user for a file name and creates the file in the current directory.
- Handles invalid input and ensures the file name is correct.
- Utilizes `FileCommand.Create()` method.

2. Delete File

- **Command:** `delete-file`
- Prompts the user for a file name and deletes the specified file from the current directory.
- Utilizes `FileCommand.Delete()` method.

3. Write to File

- **Command:** `write-file`
- Allows the user to write content to a specified file in the current directory.
- Supports appending to or overwriting the file content.
- Utilizes `FileCommand.Write()` method.

4. Read File

- **Command:** `read-file`
- Prompts the user for a file name and displays its content.
- Utilizes `FileCommand.Read()` method.

Directory Commands

1. Create Directory

- **Command:** `create-directory`
- Prompts the user for a directory name and creates it in the current directory.
- Utilizes `DirectoryCommand.Create()` method.

2. Delete Directory

- **Command:** `delete-directory`
- Prompts the user for a directory name and deletes the specified directory.
- Utilizes `DirectoryCommand.Delete()` method.

3. List Directory

- **Command:** `list-directory`
- Lists all files and subdirectories in the current directory.
- Utilizes `DirectoryCommand.List()` method.

4. Navigate Directories

- **Command:** `goto`
- Navigates to a specified directory or the parent directory.
- Maintains the current path for subsequent operations.
- Utilizes `DirectoryCommand.GetParent()` and `DirectoryCommand.GetDirectory()` methods.

Additional Commands

1. Clear Console

- **Command:** `clear`
- Clears the console window for better readability.

2. Show Help

- **Command:** `help`
- Displays a list of available commands and their functionalities.

3. Exit Application

- **Command:** `exit`
- Terminates the application.

Application Flow

1. **Initialization:**
   - The application starts by displaying a message prompting the user to type `help` for a list of commands.
   - The current working directory is initialized to the program's root path.
2. **Command Execution:**
   - The application waits for user input and processes the command.
   - Depending on the command, it invokes the corresponding method to execute the operation.
3. **Error Handling:**
   - Handles invalid commands by displaying an error message and suggesting `help`.
   - Ensures invalid inputs (e.g., empty file names) are rejected, prompting the user to retry.

Key Components

Main Class: `Program`

Properties:

- `_programPath`: Stores the root path of the application.
- `currentPath`: Tracks the current working directory.

Methods:

- `Main`: Application entry point. Implements the main loop for handling user input.
- `ShowCommands`: Displays available commands.
- `CreateFile`, `DeleteFile`, `WriteFile`, `ReadFile`: Methods for file operations.
- `CreateDirectory`, `DeleteDirectory`, `ListDirectory`, `GoTo`: Methods for directory operations.

Supporting Components

`FileCommand` and `DirectoryCommand`

- Encapsulate file and directory operations such as creating, deleting, and navigating.

`Message` Helper Class

- Provides utility methods for writing messages to the console and waiting for user responses.

Usage Example

Create a File:

```
> create-file
Type the name of the file.
> example.txt
File created successfully.
```

Write to a File:

```
> write-file
Type the name of the file.
> example.txt
Content: Hello, world!
Type 'true' - to append, 'false' - to write from start
> false
File written successfully.
```

Navigate Directories:

```
> goto
Type the name of the directory. Type '.' or empty to go to the parent directory
> subfolder
Directory changed to: C:\Users\LENOVO\Desktop\FileOperations\subfolder
```

List Directory Contents:

```
> list-directory
Contents of the current directory:
- example.txt
- subfolder
```