Birzeit University
Linux Lab ENCS313
Python Project

Description: in this project, you will automate the creation of a system manual for Python commands using Python programming language. The script will generate 20 structured XML files for each command, containing:

1. **Command Description**: Retrieve command information using subprocess.Popen.
2. **Version History of the Command**: Use subprocess.run for version extraction.
3. **Example**: Handle potential errors when retrieving examples using exception handling.
4. **Related Commands**: Extract related commands using subprocess.Popen and subprocess.PIPE.
5. **Syntax and Usage Patterns:** Provide a clear syntax for the command, including required and optional parameters.
6. **Online Documentation Links**: Include links to official Python documentation or relevant online resources for further information.

You must read Python commands from an input file and generate corresponding manuals. The input file should contain one Python command per line.

Part 2: Verification

Objective: Verify the correctness of the generated content. Read the existing document and check the command, example. etc has not changed. If there was a change, you must describe this change.

Part 3: Continuous Improvement and Extension

Objective:

1- Command Recommendation System:

Implement a system suggesting Python commands based on commands names and commands functionalities. The recommendations must appear after each search of a command.

2- Search Functionality:

Add a search functionality to the manual, allowing users to find information quickly. The search feature must be independent and the user can choose to perform search at any time.

Part 4: OOP Integration

You will use these guidelines to structure your project using Object-Oriented Programming principles to enhance modularity, encapsulation, and reusability.

1. ‘CommandManualGenerator’ Class

The CommandManualGenerator class serves as the main orchestrator for manual generation. It encapsulates the logic for reading commands from an input file and generating manuals for each command. This class promotes separation of concerns by handling the overall process of manual creation.

2. ‘CommandManual’ Class

The CommandManual class represents an individual manual for a specific Python command. It encapsulates various aspects of the manual, including command description, version history, examples, related commands, syntax and usage patterns, and online documentation links. This class is responsible for interacting with subprocesses to gather necessary information and formatting the data into a structured XML file.

Each instance of the CommandManual class is associated with a single Python command, providing a clean and encapsulated way to manage and organize the information related to that command.

3. Use any other OOP principles or structure that you see necessary.

BONUS:

Part 5: GUI Integration

Objective: Develop a graphical user interface (GUI) for the Python Command Manual Generator to enhance user interaction and provide a more intuitive experience.

Key Features:

1. **Command Selection:** Implement a user-friendly interface for users to press on a button which reads the commands we want to generate manuals for from *commands.txt* file.

2. **Manual generation**: Create a dedicated section in the GUI to generate the manual for *one selected command.*

3. **Manual Visualization:** Create a dedicated section in the GUI to press on to open the manual for *one selected command.*

4. **Verification Interface:** Provide a verification section where users can run the selected command directly from the GUI and view the results to verify correctness. (Implementing Part 2)

5. **Search Functionality**: Use your GUI to enable users to search for specific commands using commands names. (Implementing Part 3)

6. **Command Recommendation Panel**: Use your GUI to enable users for getting command recommendations, displaying suggestions based on command's name and command's usage. (Implementing Part 3)

XML file example:

```
<Manuals>
<CommandManual>
<CommandName>...</CommandName>
<CommandDescription>...</CommandDescription>
<VersionHistory>...</VersionHistory>
<Example>...</Example>
<RelatedCommands>...</RelatedCommands>
<SyntaxAndUsage>...</SyntaxAndUsage>
<DocumentationLinks>...</DocumentationLinks>
<CommandManual/>
</Manuals>
```

- <Manuals>: The root element containing all command manuals.
- <CommandManual>: Represents an individual manual for a Python command. Contains sub-elements for different pieces of information related to the command.
- <CommandName>: The name of the Python command.
- <CommandDescription>: Description of the command.
- <VersionHistory>: Version history information.
- <Example>: Example of using the command.
- <RelatedCommands>: Information about related commands.
- <SyntaxAndUsage>: Syntax and usage patterns.
- <DocumentationLinks>: Links to online documentation.

What to submit?

- You need to submit your python file (.py).
- You need to submit a technical report (.PDF) describing your code and including run examples no longer than 10 pages.

Where to submit?

- Send your project files as a reply to the message.

Good luck!