Terminal Screen Project Progress

DATE: 9th December 2024

Challenges

Technical Challenges

- **Cursor Movement Logic:** Implementing the cursor movement functionality with ANSI escape codes was technically demanding. Ensuring accurate validation of cursor positions relative to the screen's dimensions required careful design.
- **Rendering Complexity:** Efficiently rendering the terminal screen while maintaining performance and visual clarity involved understanding ANSI escape codes and their integration with OCaml's I/O operations.
- **State Management:** Handling immutable state updates in OCaml proved challenging, particularly when dealing with a dynamically changing screen buffer. Adopting a functional approach required a significant shift from the typical imperative programming paradigms.
- **Error Handling:** Developing robust error handling mechanisms for invalid inputs, out-of-bound cursor movements, and rendering issues was crucial for creating a stable application.

Non-Technical Challenges

- **Time Constraints:** Allocating adequate time to tackle complex technical problems while balancing other personal commitments was challenging.
- **Learning Curve:** Learning and mastering ANSI escape codes and their nuances in terminal applications, coupled with the functional programming style of OCaml, required significant effort.

Collaboration

- Collaborated with a professional developer friend to discuss project architecture and terminal rendering best practices. This interaction provided valuable insights into structuring the code for better modularity and maintainability.

Project Updates

Features Completed

1. **Screen Setup Functionality**
    - The `setup_screen` function initializes the screen buffer with specified dimensions and colour mode, ensuring a consistent starting point for rendering.

2. **Drawing Characters**
    - Implemented the `draw_character` function to place characters at specified coordinates on the screen with a given colour index.
3. **Line Drawing**
    - Added the `draw_line` feature, enabling users to draw straight lines between two points with customisable characters and colours.
4. **Text Rendering**
    - The `render_text` function allows rendering strings at specified coordinates, advancing the cursor automatically for each character.
5. **Screen Display**
    - Developed the `display_screen` function to render the entire screen buffer using ANSI escape codes. This function dynamically updates the terminal view, ensuring a seamless display experience.
6. **Cursor Movement**
    - Completed the cursor movement functionality using ANSI escape codes, enabling users to reposition the cursor dynamically within the screen boundaries without affecting other elements.

Progress

Rating: 8/10
Reasoning:

- The project has achieved several foundational features, including screen setup, character and line drawing, text rendering, and cursor movement. These components provide a solid base for future enhancements.
- Despite significant progress, certain aspects, such as advanced UI/UX improvements and feature optimisations, remain pending.

Measuring Progress

Progress is measured by the following metrics:

1. Completion of planned features up to cursor movement.
2. Testing and validation of each feature for robustness and performance.
3. Maintaining adherence to the project's timeline.

Remaining Tasks:

- Implementation of advanced UI/UX features for a polished terminal interface.
- Comprehensive testing for edge cases and stress scenarios.

Completion Assessment

Given the current progress, completing the project on time appears achievable with focused effort. The foundational features are functional, and remaining tasks mainly involve refinement and enhancement.

Action Plan:

- Allocate focused time blocks for UI/UX improvements and pending features.
- Conduct rigorous testing to ensure stability and reliability.
- Regularly reassess priorities to maintain alignment with the MVP goals.

By maintaining this trajectory and addressing pending tasks with urgency, the project is well-positioned to deliver a functional and reliable terminal screen application.