CSCI 4448/5448, Object-Oriented Analysis and Design
Team: Abiriham Bitew and Nathan Mukooba

Project 6 Update

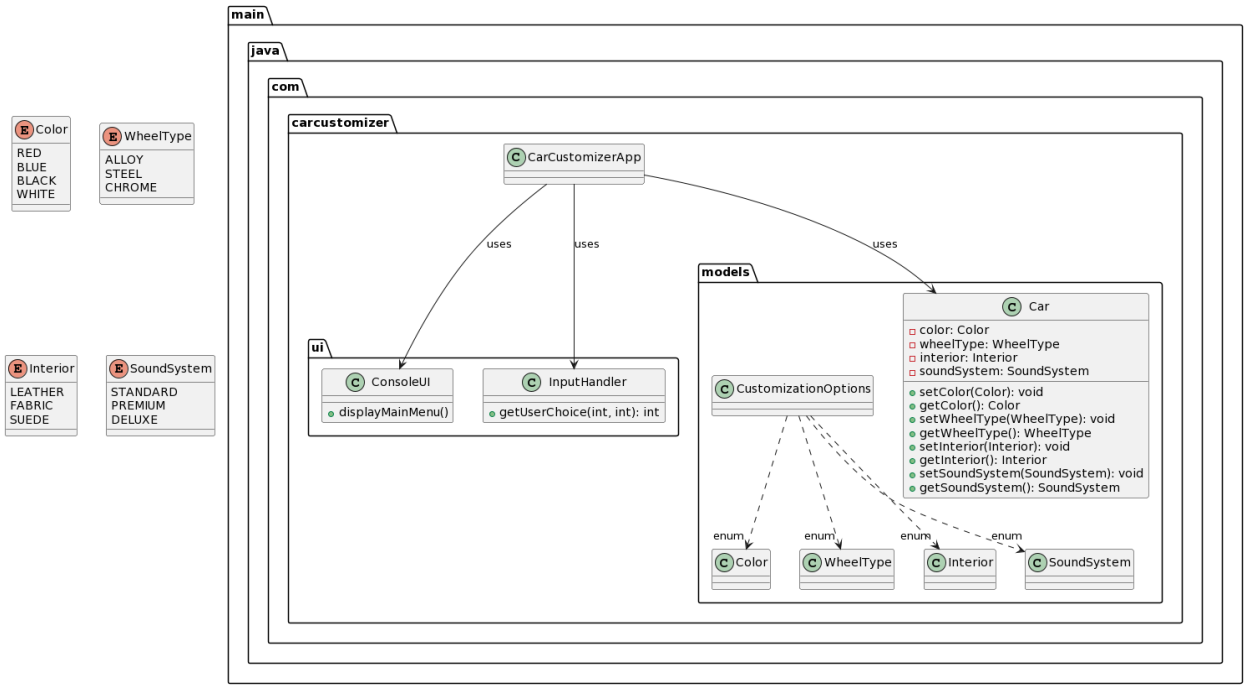
Status Summary

1. **Work Done**
    - The First Sprint:
        - **[Nathan Mukooba]:** Implemented a console-based UI for Users to view Car Models
        - **[Abiriham Bitew]:** Implemented a console-based UI for Users to View Car Models
        - **[Nathan and Abiriham]:** Used OOP Principles to Create Appropriate Structure in UI and Data Parsing or Car Model
2. **Changes or Issues Encountered**
    - Shift from Project 5 Design
        - COVID Incident caused a delay in programming
        - Shifted to Console-Based Application
    - New Plan:
        - We maintain real-time updates with the drawback of visuals from a web application
3. **Patterns**

    Currently, our project has not incorporated design patterns. However, we intend to integrate them in the next phase of development. Our plan includes the implementation of:

    - **Factory Pattern** to facilitate the creation of Car objects with diverse configurations. Additionally, we aim to utilize the
    - **Builder Pattern** which will be particularly beneficial for constructing complex Car objects through a step-by-step process, is especially useful in scenarios involving extensive customization.
    - **Strategy Pattern** to handle dynamic variations in the customization logic.
    - **Observer Pattern**: Additionally, we plan to implement the Observer Pattern to manage event-driven updates and real-time feedback in the application.

Class Diagram:



Next Steps:

As we proceed with the CarCustomizer Console Simulator project, our ongoing focus revolves around further development and enhancements within the command-line interface framework. Here's a comprehensive outline of our next steps:

1. **Implementation of Design Patterns:**

We are dedicated to incorporating essential design patterns to fortify the architecture and maintainability of our CLI-based application. Our immediate objectives involve integrating:

- **Factory Pattern:** Facilitating the creation of various Car objects with distinct configurations.
- **Builder Pattern:** Streamlining complex car customization processes through structured, step-by-step methodologies.
- **Strategy Pattern:** Managing dynamic variations in customization logic, ensuring flexibility and scalability.

- **Observer Pattern**: Implementing event-driven updates and real-time feedback management in the application.

2. **Visual Representation of Customized Cars:**

In addition to the ongoing CLI development, we are committed to enhancing the user experience. We plan to integrate a feature that enables users to visualize their customized cars. This will be achieved through:

- ASCII art representation or an external application generating visual representations based on user customization data.

3. **Reflection of the New Plan**

The CarCustomizer Console Simulator transitions to a Java-based CLI program. Ongoing developments focus on refining user interaction and backend systems, as detailed previously.