Parking System Program Documentation Using .NET 6.0

Author: Sheila Nuur Ditrie

A console application in C# and .NET 6.0 for managing a virtual parking system. Allows users to allocate parking slots, park vehicles based on registration number and color, and generate various parking reports.

Components

1. Main Program (`Program` class)
    - Entry point (Main method) of the application.
    - Uses a `Dictionary<int, Vehicle>` (`parkingLot`) to store vehicles parked in the slots and `totalLots` to track the total number of parking slots available.
2. Vehicle Class
    - Represents a vehicle with properties `RegistrationNumber`, `Color`, and `Type`.
    - Constructor initializes these properties.
3. Main Loop (`Main` method)
    - Continuously prompts for user input (`$ `).
    - Parses commands and routes them to respective methods using a `switch` statement based on the first command token.
4. Command Methods
    - **`CreateParkingLot(string[] command)`**: Creates a parking lot with the specified number of slots (`totalLots`).
    - **`ParkVehicle(string[] command)`**: Parks a vehicle with registration number, color, and type into the parking lot.
    - **`LeaveParkingLot(string[] command)`**: Frees up a parking slot based on slot number.
    - **`GetParkingStatus()`**: Displays current parking status including slot number, vehicle type, registration number, and color.
    - **`CountVehiclesByType(string[] command)`**: Counts vehicles of a specific type.
    - **`GetVehiclesByOddPlate()`**: Lists vehicles with odd-numbered registration plates.
    - **`GetVehiclesByEvenPlate(string[] command)`**: Lists vehicles with even-numbered registration plates.
    - **`GetVehiclesByColor(string[] command)`**: Lists vehicles by specified color.

- **`GetSlotsByColor(string[] command)`**: Lists slots occupied by vehicles of specified color.
- **`GetSlotByRegistrationNumber(string[] command)`**: Retrieves slot number by vehicle registration number.
- **`GetParkingReport()`**: Generates a report showing occupied slots, available slots, vehicles with odd/even registration numbers, and counts of vehicles by type and color.

5. Utility Methods
   - **`PrintErrorMessage(string message)`**: Prints error messages to the console.
   - **`IsValidVehicleType(string type)`**: Validates if the vehicle type is either "Mobil" or "Motor".

Usage Instructions

- Users interact with the program by entering commands at the `$` prompt.
- Each command performs specific actions related to managing the parking lot or retrieving information about parked vehicles.
- Error handling ensures that users receive appropriate feedback when operations fail (e.g., parking lot is full, invalid commands).

Code Explain

- The program uses collections (`Dictionary` and LINQ queries) to manage and query vehicle data efficiently.
- Command parsing (`Split` and `ToLower`) ensures case-insensitive command handling.
- Methods are designed to be modular and handle specific functionalities, promoting code reuse and maintainability.