**ОТЧЕТ**

по лабораторной работе №1

по дисциплине «Объектно-ориентированный анализ и проектирование»

**Строитель (Builder)**

Назначение: порождающий паттерн, который позволяет создавать сложные объекты пошагово. Строитель даёт возможность использовать один и тот же код строительства для получения разных представлений объектов.

* Когда процесс создания нового объекта не должен зависеть от того, из каких частей этот объект состоит и как эти части связаны между собой.
* Когда необходимо обеспечить получение различных вариаций объекта в процессе его создания

Задача: Информационная система предназначена для «сборки» комплектации автомобиля. В зависимости от марки и модели комплектация может включать следующие категории элементов: экстерьер, интерьер, комфорт, безопасность, мультимедиа и т.д. **Реализовать «сборку» различных комплектаций для выбранной модели автомобиля и подготовку сравнительной таблицы опций в формате html**. При выполнении задания использовать реальный каталог выбранного автопроизводителя.

**ПРОГРАММА БЕЗ ПАТТЕРНА:**

1. **Классы MainWindow и Car.** Класс **Car** является **основным проблемным классом** содержащим конструктор из пяти string полей:  
   exterior, interior, comfort, safety, multimedia.
2. Обработчики событий **QCombobox**: Эти методы вызываются при изменении выбранных элементов в комбо-боксах, отвечающих за выбор машины и её комплектации соответственно. При выборе элемента обновляется список доступных моделей/комплектаций в следующем комбо-боксе.

**Проблема** **подхода** – большое количество классов, которые придётся создать для каждой реализации.

**ПРОГРАММА С ПАТТЕРНОМ СТРОИТЕЛЬ:**

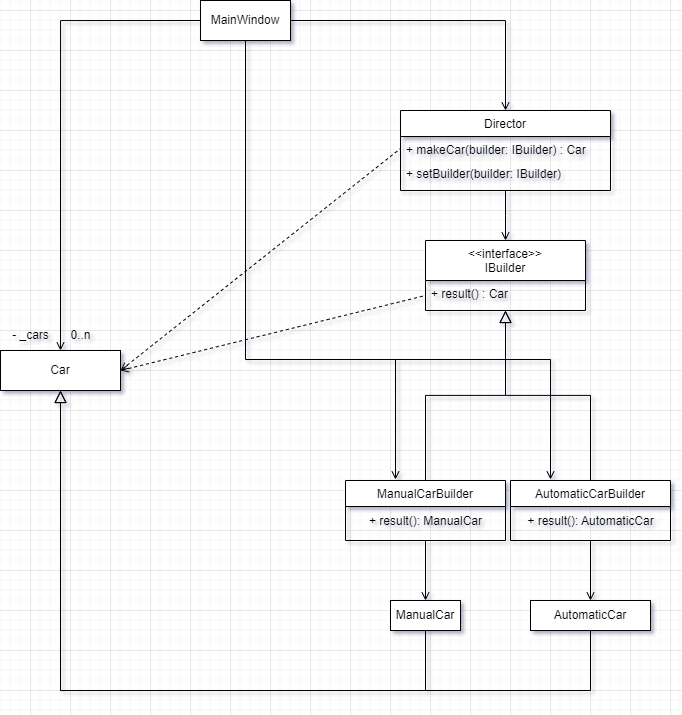
Решение: Через паттерн мы можем вынести конструирование объекта за пределы его собственного класса, поручив это дело отдельным объектам – строителям.

В контексте данной задачи, **строитель** будет **отвечать за сборку комплектаций автомобиля**, а **директор будет возвращать результат процесса.** Каждый конкретный строитель реализует свой собственный способ построения класса.

2. **IBuilder** (Строитель автомобиля): Это абстрактный класс, который определяет основные шаги для создания автомобиля. Он содержит список опций options, которые будут добавляться при построении. Метод GetOptions() возвращает список всех опций.

3. **ManualCarBuilder**: Этот класс наследуется от IBuilder и реализует методы для создания конкретной модели автомобиля BMW. Каждый метод setExterior(), setInterior(), setComfort(), setSafety(), setMultimedia(), setDriveMode().

4. **Director (Директор)**: Этот класс отвечает за построение конечного продукта (автомобиля). Он принимает экземпляр IBuilder и использует его для получения результатов работы строителя.



1. Паттерн Строитель позволяет **разделить процесс** создания сложного объекта **на отдельные шаги** и **упрощает его конструирование**.
2. Классы Строителя позволяют легко **добавлять или удалять** значения полей класса.
3. Использование отдельного класса Строителя для конструирования объекта **упрощает и улучшает читаемость кода**, так как каждый метод Строителя отвечает за определенный шаг в процессе сборки объекта.