**№ 4Применение порождающих паттернов проектирования**

**Паттерн** представляет определенный способ построения программного кода для решения часто встречающихся проблем проектирования. В данном случае предполагается, что есть некоторый набор общих формализованных проблем, которые довольно часто встречаются, и паттерны предоставляют ряд принципов для решения этих проблем.

**Что же дает нам применение паттернов?** При написании программ мы можем формализовать проблему в виде классов и объектов и связей между ними. И применить один из существующих паттернов для ее решения. В итоге нам не надо ничего придумывать. У нас уже есть готовый шаблон, и нам только надо его применить в конкретной программе.

1.Какие типы паттернов бывают?

- **Порождающие паттерны -** процесс порождения классов и объектов, отвечают за создание объектов и позволяют системе быть независимой от типов этих самых объектов и от процесса порождения.

Среди них выделяются следующие:

* **Абстрактная фабрика (Abstract Factory)**
* **Строитель (Builder)-**  отвечает за процесс поэтапного создания объекта.
* **Фабричный метод (Factory Method)-** работает с полиморфизмом.
* **Прототип (Prototype)-** о клонировании.
* **Одиночка (Singleton)-** контролирует создание лишь одного экземпляра класса.

- **структурные паттерны** - рассматривает, как классы и объекты образуют более крупные структуры - более сложные по характеру классы и объекты. К таким шаблонам относятся:

* **Адаптер (Adapter)**
* **Мост (Bridge)**
* **Компоновщик (Composite)**
* **Декоратор (Decorator)**
* **Фасад (Facade)**
* **Приспособленец (Flyweight)**
* **Заместитель (Proxy)**

**поведенческими** - они определяют алгоритмы и взаимодействие между классами и объектами, то есть их поведение. Среди подобных шаблонов можно выделить следующие:

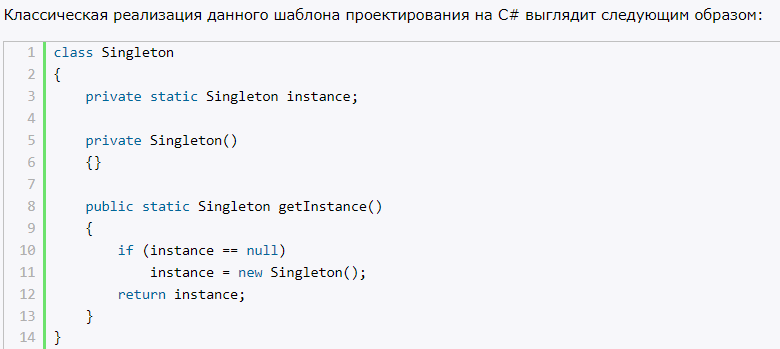
* **Цепочка обязанностей (Chain of responsibility)**
* **Команда (Command)**
* **Интерпретатор (Interpreter)**
* **Итератор (Iterator)**
* **Посредник (Mediator)**
* **Хранитель (Memento)**
* **Наблюдатель (Observer)**
* **Состояние (State)**
* **Стратегия (Strategy)**
* **Шаблонный метод (Template method)**
* **Посетитель (Visitor)**

2. Нарисуете диаграмму классов и поясните принцип работы паттерна Singleton. Назовите условия применения.

**Singleton-**  - порождающий паттерн, который гарантирует, что для определенного класса будет создан только один объект, а также предоставит к этому объекту точку доступа.

Когда надо использовать Синглтон? Когда необходимо, чтобы для класса существовал только один экземпляр

Синглтон позволяет создать объект только при его необходимости. Если объект не нужен, то он не будет создан.



В классе определяется статическая переменная - ссылка на конкретный экземпляр данного объекта и приватный конструктор. В статическом методе getInstance() этот конструктор вызывается для создания объекта, если, конечно, объект отсутствует и равен null.

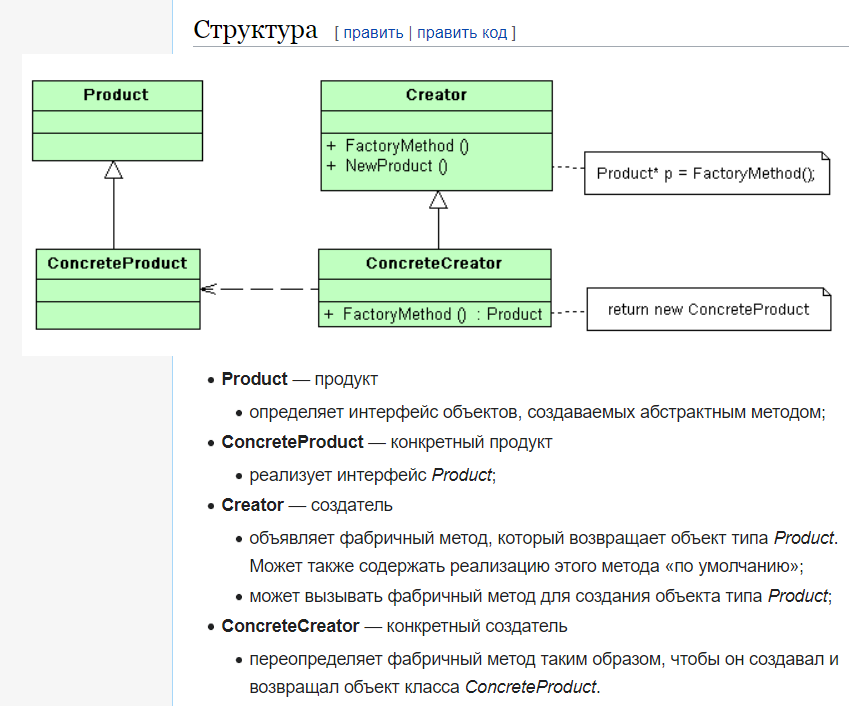
3. Как сделать потокобезопасную, с отложенной реализацией версию Singleton.

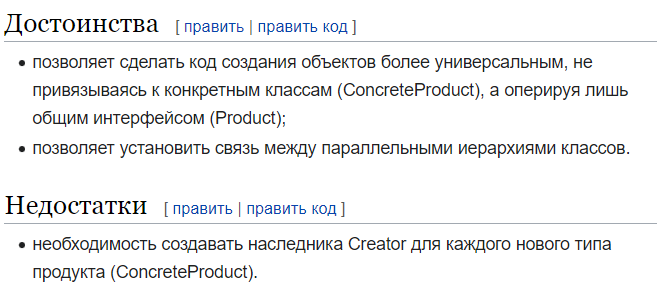
4. Нарисуете диаграмму классов и поясните принцип работы паттерна Factory Method. Назовите условия применения

Фабричный метод (Factory Method) - это паттерн, который определяет интерфейс для создания объектов некоторого класса, но непосредственное решение о том, объект какого класса создавать происходит в подклассах. То есть паттерн предполагает, что базовый класс делегирует создание объектов классам-наследникам.

### Когда надо применять паттерн

* Когда заранее неизвестно, объекты каких типов необходимо создавать
* Когда система должна быть независимой от процесса создания новых объектов и расширяемой: в нее можно легко вводить новые классы, объекты которых система должна создавать.
* Когда создание новых объектов необходимо делегировать из базового класса классам наследникам





5. Нарисуете диаграмму классов и поясните принцип работы паттерна Abstract Factory. Назовите условия применения

(Abstract Factory) предоставляет интерфейс для создания семейств взаимосвязанных объектов с определенными интерфейсами без указания конкретных типов данных объектов.

### Когда использовать абстрактную фабрику

* Когда система не должна зависеть от способа создания и компоновки новых объектов
* Когда создаваемые объекты должны использоваться вместе и являются взаимосвязанными

6. Нарисуете диаграмму классов и поясните принцип работы паттерна Prototype. Назовите условия применения

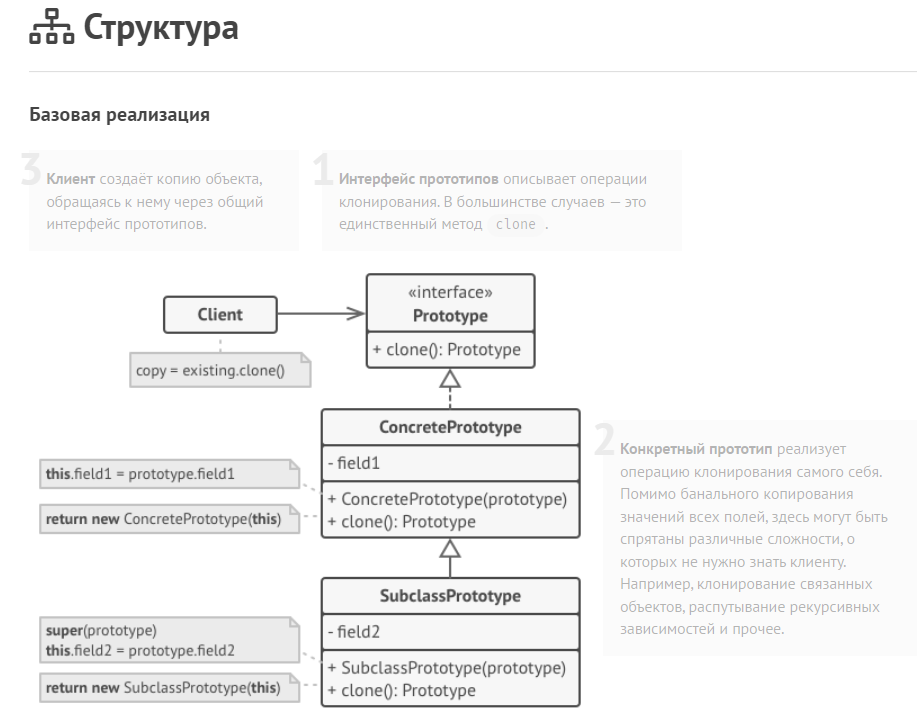
**Прототип** — это порождающий паттерн, который позволяет копировать объекты любой сложности без привязки к их конкретным классам.

Все классы—Прототипы имеют общий интерфейс. Поэтому вы можете копировать объекты, не обращая внимания на их конкретные типы и всегда быть уверены, что получите точную копию. Клонирование совершается самим объектом-прототипом, что позволяет ему скопировать значения всех полей, даже приватных.

Паттерн Прототип (Prototype) позволяет создавать объекты на основе уже ранее созданных объектов-прототипов. То есть по сути данный паттерн предлагает технику клонирования объектов.

Когда использовать Прототип?

* Когда конкретный тип создаваемого объекта должен определяться динамически во время выполнения
* Когда нежелательно создание отдельной иерархии классов фабрик для создания объектов-продуктов из параллельной иерархии классов (как это делается, например, при использовании паттерна Абстрактная фабрика)
* Когда клонирование объекта является более предпочтительным вариантом нежели его создание и инициализация с помощью конструктора. Особенно когда известно, что объект может принимать небольшое ограниченное число возможных состояний.



7. Нарисуете диаграмму классов и поясните принцип работы паттерна Builder. Назовите условия применения

Строитель позволяет отделить процесс создания сложного объекта от его реализации. При этом, результатом одних и тех же операций могут быть различные объекты.

Строитель (Builder) - шаблон проектирования, который инкапсулирует создание объекта и позволяет разделить его на различные этапы.

### Когда использовать паттерн Строитель?

* 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

8. Что такое Пул объектов (Object pool), в чем суть паттерна?

Объектный **пул** (англ. **object pool**) — порождающий **паттерн** (шаблон) проектирования, набор инициализированных и готовых киспользованию **объектов**. Когда системе требуется **объект**, он не создаётся, а берётся из **пула**. Когда объектбольше не нужен, он не уничтожается, а возвращается в **пул**.

9. Какое основное назначение паттерна отложенная инициализация (Lazy initialization)?

**Основное назначение отложенной инициализации** -отложить создание ресурсоемких объектов до того момента, когда это действительно необходимо.

Для лабы пометки

2)Settings

6-7

Впф-

Приемущества