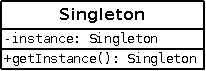
1. Какие типы паттернов бывают?

Порождающие - абстрагируют процесс порождения классов и объектов.  
поведения – опр. алгоритмы взаимод. между классами и объектами  
структурные – рассматривает, как классы и объекты образуют более крупные структуры – более сложные по характеру классы и объекты

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



Используется когда необходимо, чтобы для класса существовал только 1 экземпляр. Позволяет создать объект только при его необходимости. Если объект не нужен, то он не будет создан.

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

\*Чтобы избежать одновременного доступа к коду из разных потоков критическая секция заключается в блок lock.

\*Lazy-реализация - данные иниц. только перед непосред использованием.

\*Реализация через класс Lazy<T>

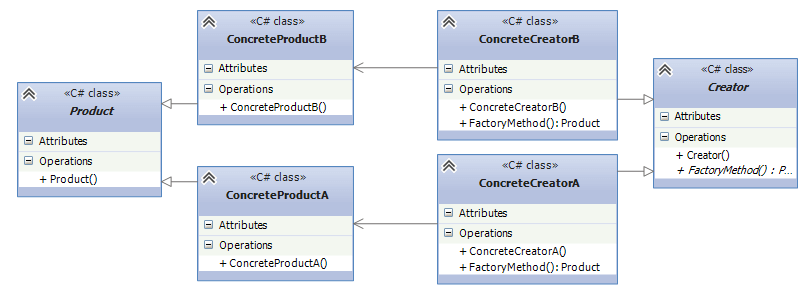
4. Нарисуете диаграмму классов и поясните принцип Factory Method. Назовите усл примен

Определяет интерфейс для создания объектов некоторого класса, но непосредственное решение о том, объект какого класса создавать происходит в подклассах.

**Когда:** - заранее известно объекты каких типов необх.создавать

-система дб независимой от процесса создания новых объектов и расширяемой

- создание новых объектов необходимо делегировать из базового класса наследникам.

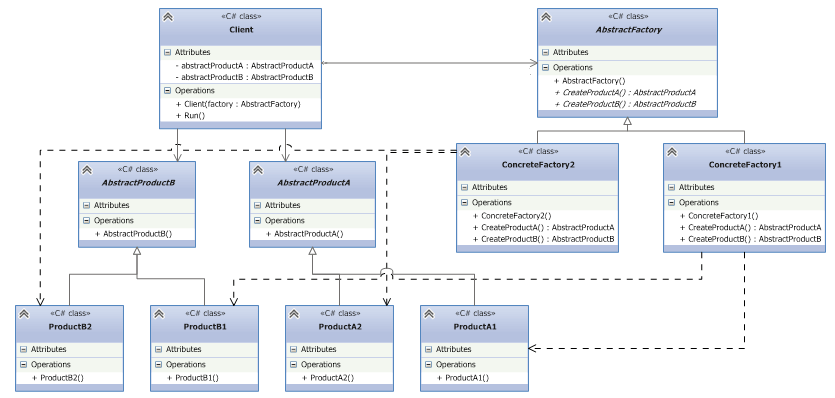


5. Нарисуете диаграмму классов и поясните принцип Abstract Factory + условия применения

Предоставляет интерфейс для создания семейств взаимосвязанных объектов с определенными интерфейсами без указания конкретных типов данных объектов.

**Когда:** - система не д.зависеть от способа создания и компоновки новых объектов

-создаваемые объекты д.использоваться вместе и явл.взаимосвязанными.



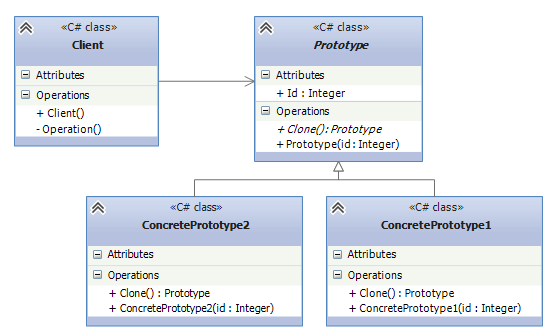
6. Нарисуете диаграмму классов и поясните принцип работы Prototype+ условия применения

Позвол.созд.объекты на осн.ранее созданных объектов-прототипов.(клонирование)

**Когда:** -конкр. тип создаваемого объекта д. определяться динамически во время выполнения

-нежелательно создание отдельной иерархии классов фабрик создания объектов-продуктов из параллельной иерархии классов

- клонирование – более предпочтительный варик нежели создание и инициализирование с помощью конструктора.

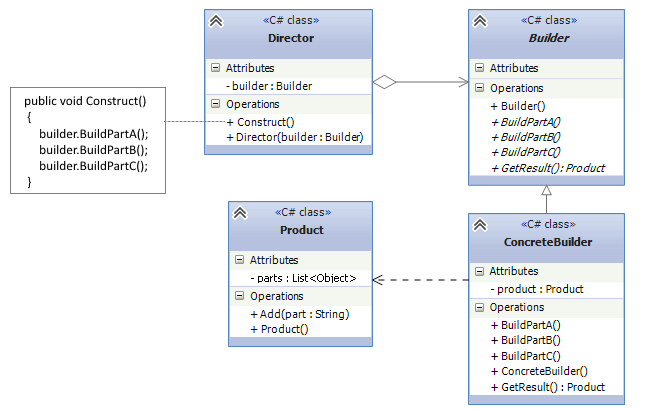


7. Нарисуете диаграмму классов и поясните принцип работ Builder + условия применения

Инкапсулирует создание объекта и позволяет разделить его на различные этапы

**Когда:** - процесс создания нового объекта не должен зависеть от того, из каких частей этот объект состоит и как эти части связаны между собой

- необх. обеспечить получение различных вариаций объекта в процессе его создания



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

Это порождающий паттерн, набор инициализированных и готовых к использованию объектов. Когда системе нужен объект – не создается, а берется из пула. Когда объект больше не нужен – не уничтожается, а возвращается в пул.

Повышает производительность.

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

Прием, когда некоторая ресурсоемкая операция (создание объекта, вычисление значения) выполняется непосредственно перед тем, как будет использован ее результат.  
инициализация вып. по требованию, а не заблаговременно.

Частный случай – создание объекта в момент обращения к нему – один из порождающих паттернов. Используется в сочетании с фабричным методом, одиночкой и заместителем.

+ускоряется нач.инициал.

+иниц.только тогда, когда необх

-задержка при 1-ом обращ. к объекту

-невозможно явным образом задать порядок инициализации объектов.

ABSTRACT FACTORY:

* Абстрактные классы **AbstractProductA** и **AbstractProductB** определяют интерфейс для классов, объекты которых будут создаваться в программе.
* Конкретные классы **ProductA1 / ProductA2** и **ProductB1 / ProductB2**представляют конкретную реализацию абстрактных классов
* Абстрактный класс фабрики **AbstractFactory** определяет методы для создания объектов. Причем методы возвращают абстрактные продукты, а не их конкретные реализации.
* Конкретные классы фабрик **ConcreteFactory1** и **ConcreteFactory2** реализуют абстрактные методы базового класса и непосредственно определяют какие конкретные продукты использовать
* Класс клиента **Client** использует класс фабрики для создания объектов. При этом он использует исключительно абстрактный класс фабрики AbstractFactory и абстрактные классы продуктов AbstractProductA и AbstractProductB и никак не зависит от их конкретных реализаций

FACTORY METHOD

* Абстрактный класс **Product** определяет интерфейс класса, объекты которого надо создавать.
* Конкретные классы **ConcreteProductA** и **ConcreteProductB** представляют реализацию класса Product. Таких классов может быть множество
* Абстрактный класс **Creator** определяет абстрактный фабричный метод FactoryMethod(), который возвращает объект Product.
* Конкретные классы **ConcreteCreatorA** и **ConcreteCreatorB** - наследники класса Creator, определяющие свою реализацию метода FactoryMethod(). Причем метод FactoryMethod() каждого отдельного класса-создателя возвращает определенный конкретный тип продукта. Для каждого конкретного класса продукта определяется свой конкретный класс создателя.

Таким образом, класс Creator делегирует создание объекта Product своим наследникам. А классы ConcreteCreatorA и ConcreteCreatorB могут самостоятельно выбирать какой конкретный тип продукта им создавать.

PROTOTYPE

* **Prototype**: определяет интерфейс для клонирования самого себя, который, как правило, представляет метод Clone()
* **ConcretePrototype1** и **ConcretePrototype2**: конкретные реализации прототипа. Реализуют метод Clone()
* **Client**: создает объекты прототипов с помощью метода Clone()

BULDER

* **Product**: представляет объект, который должен быть создан. В данном случае все части объекта заключены в списке parts.
* **Builder**: определяет интерфейс для создания различных частей объекта Product
* **ConcreteBuilder**: конкретная реализация Buildera. Создает объект Product и определяет интерфейс для доступа к нему
* **Director**: распорядитель - создает объект, используя объекты Builder