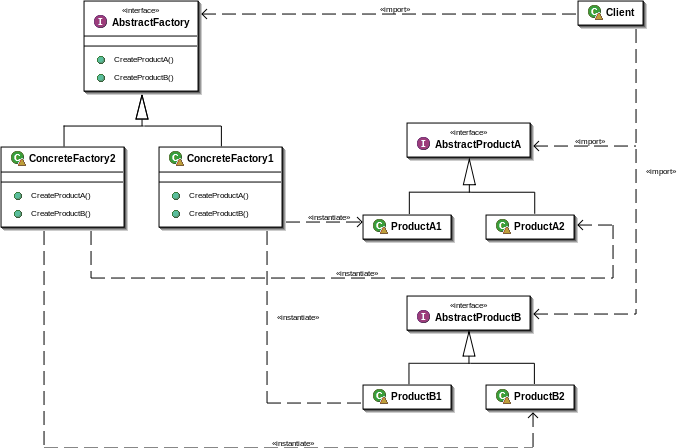
Абстрактная фабрика (Abstract Factory)

Паттерн, порождающие объекты. Предоставляет интерфейс для создания семейств взаимосвязанных или взаимозависимых объектов, не специфицируя из конкретных классов.

**Применение**

Используйте данный паттерн, когда:

* система не должна зависеть от того, как создаются, компонуются и представляются входящие в нее объекты;
* входящие в семейство взаимосвязанные объекты должны использоваться вместе и вам необходимо обеспечить выполнение этого ограничения;
* система должна конфигурироваться одним из семейств составляющих ее объектов;
* вы хотите предоставить библиотеку объектов, раскрывая только их интерфейсы, но не реализацию.



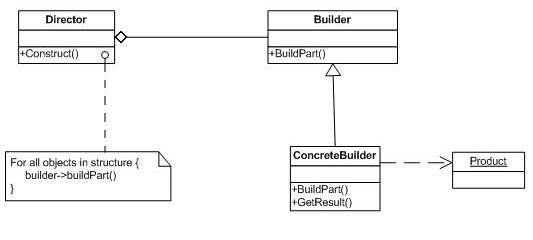
Строитель (Builder)

Отделяет конструирование сложного объекта от его представления, так что в результате одного и того же процесса конструирования могут получаться разные представления.

## Применимость

Используйте паттерн строитель, когда:

* алгоритм сложного объекта не должно зависеть от того, из каких частей состоят объект и как они стыкуются между собой;
* процесс конструирования должен обеспечивать различные представления конструируемого объекта.



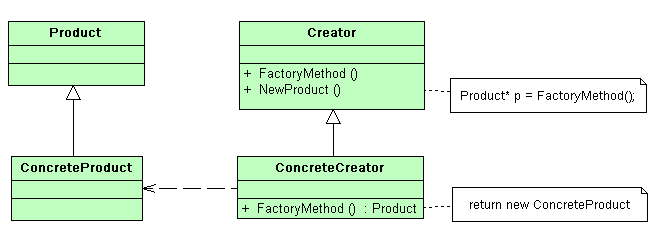
Фабричный метод (Factory method)

Определяет интерфейс объекта, но оставляет подклассам решение о том, какой класс инстанцировать. Фабричный метод позволяет классу делегировать инстанцирование подклассам.

## Применимость

Используйте паттерн фабричный метод, когда:

* классе заранее неизвестно, объекты каких классов ему нужно создавать;
* класс спроектирован так, чтобы объекты, которые он создает, специфицировались подклассами;
* класс делегирует свои обязанности одному из нескольких вспомогательных подклассов, и вы планируете локализовать знание о том, какой класс принимает эти обязанности на себя.



Одиночка (Singleton)

Гарантирует, что класс имеет только один экземпляр и предоставляет глобальную точку доступа к нему. Иными словами, при использовании паттерна одиночка гарантируется единственность экземпляра класса.

## Применение

Используется тогда, когда:

* должен быть ровно один экземпляр некоторого класса, легко доступный всем клиентам;
* единственный экземпляр должен расширяться путем порождения подклассов, и клиентам нужно имет044C возможность работать с расширенным экземпляром без модификации своего кода.

Прототип (Prototype)

Задает виды создаваемых объектов с помощью экземпляра-прототипа и создает новые объекты путем копирования этого прототипа.

## Применимость

Используйте паттерн **прототип**, когда система не должна зависеть от того, как в ней создаются, компонуются и представляются продукты:

* инстанцируемые классы определяются во время выполнения, например, с помощью динамической загрузки;
* для того, чтобы избежать построения иерархий классов или фабрик, параллельных иерархии классов продуктов;
* экземпляры класса могут находиться в одном из не очень большого числа различных состояний. Может оказаться удобнее установить соответствующее число прототипов и клонировать их, а не инстанцировать каждый раз класс вручную в подходящем состоянии.

