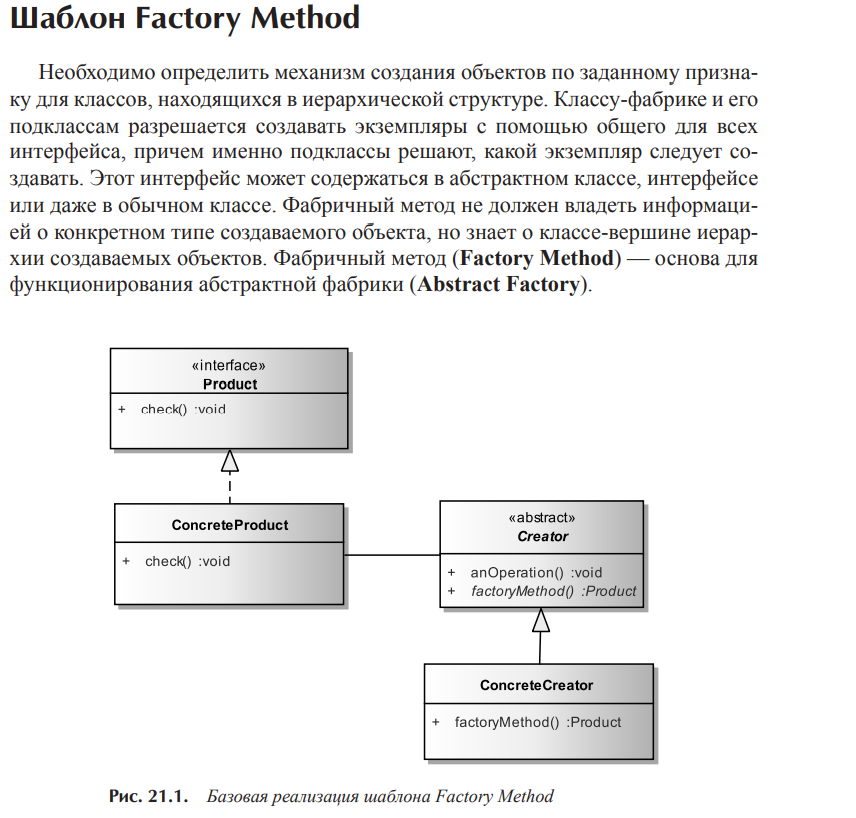
**Design Patterns**

1. Зачем нужны паттерны? Привести примеры из проекта.
2. Какие паттерны вы знаете и как их применяет Java SE.
3. Factory Method, Builder



Factory Pattern or Factory Method Pattern говорит **«просто** **определите интерфейс или абстрактный класс для создания объекта, но позвольте подклассам решить, какой класс создать экземпляр».** Другими словами, подклассы отвечают за создание экземпляра класса.

Этот шаблон также известен как **виртуальный конструктор.**

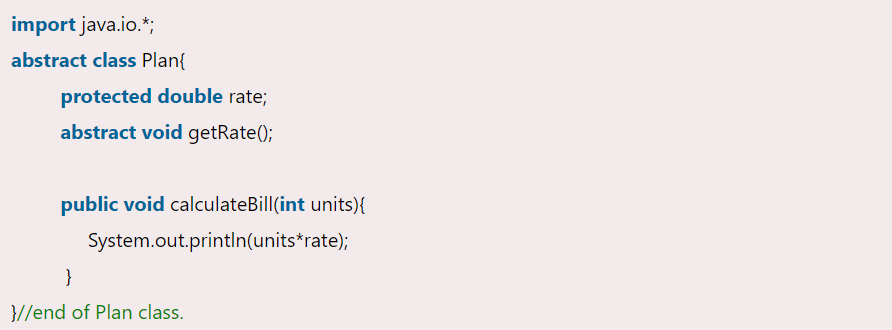
Преимущества Factory Method Pattern:

* Позволяет подклассам выбирать тип создаваемых объектов.
* Он способствует **слабой связи,** устраняя необходимость привязки классов приложения к коду. Это означает, что код взаимодействует исключительно с интерфейсом или абстрактным классом, так что он будет работать с любыми классами, которые реализуют этот интерфейс или расширяют этот абстрактный класс.

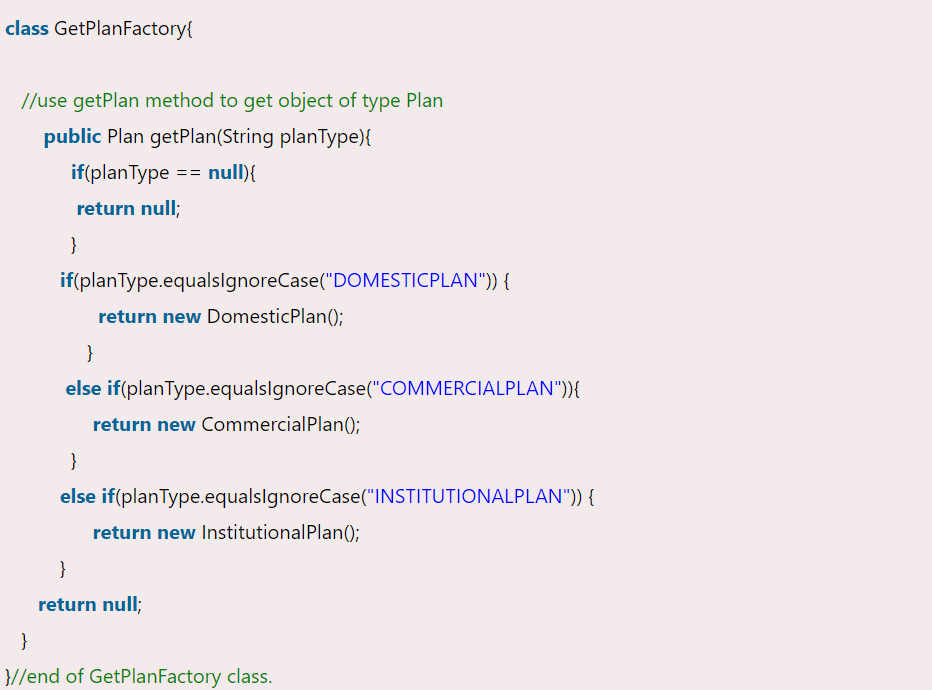
Использование шаблона проектирования фабрики

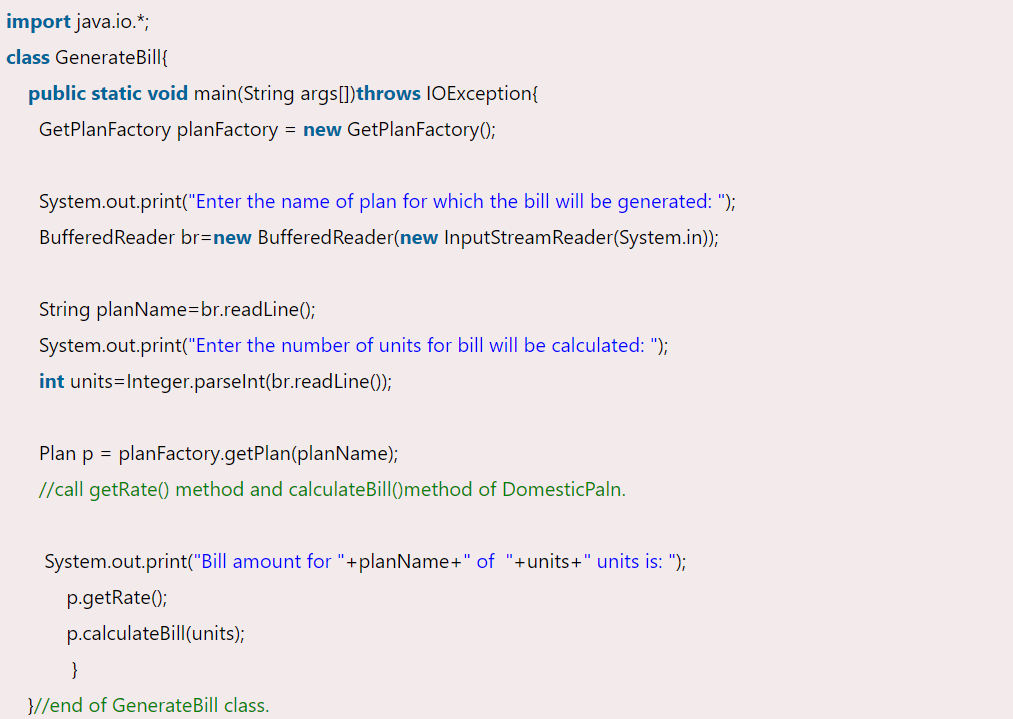
* Когда класс не знает, какие подклассы потребуется создать
* Если класс хочет, чтобы его подклассы определяли создаваемые объекты.
* Когда родительские классы выбирают создание объектов для своих подклассов.

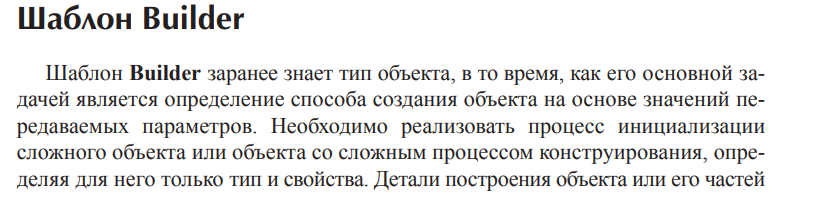
Пример:

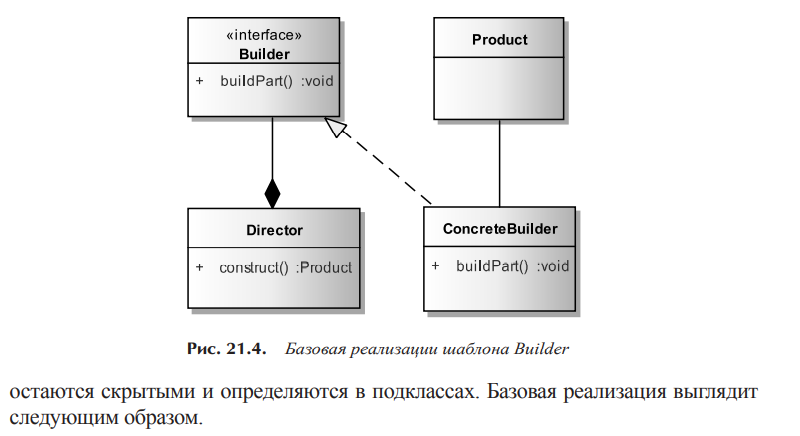


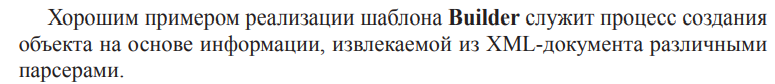


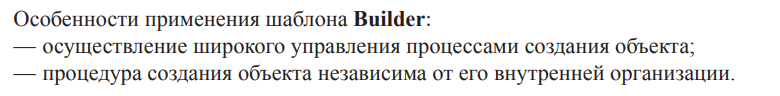












1. Singleton (class Runtime)
2. Как сделать чтобы в Singleton не работала двойная блокировка?
3. Как у Singleton создать второй объект? И как воспрепятствовать этому.
4. Prototype
5. Command
6. Composite
7. Chain of responsibility (Filter, closing io stream, closing connection)
8. State (Thread.State)
9. Iterator (Enumeration, Iterator, ListIterator)
10. Proxy
11. Observer (Listener-s)
12. Wrapper
13. Immutable
14. MVC
15. DAO vs. Repository