### **Spring Frameworkとは？**

Java開発において利用できるフレームワークの **集合体** のこと。  
※「Spring」と縮めて呼ばれているものは、Spring Frameworkのことを指している。

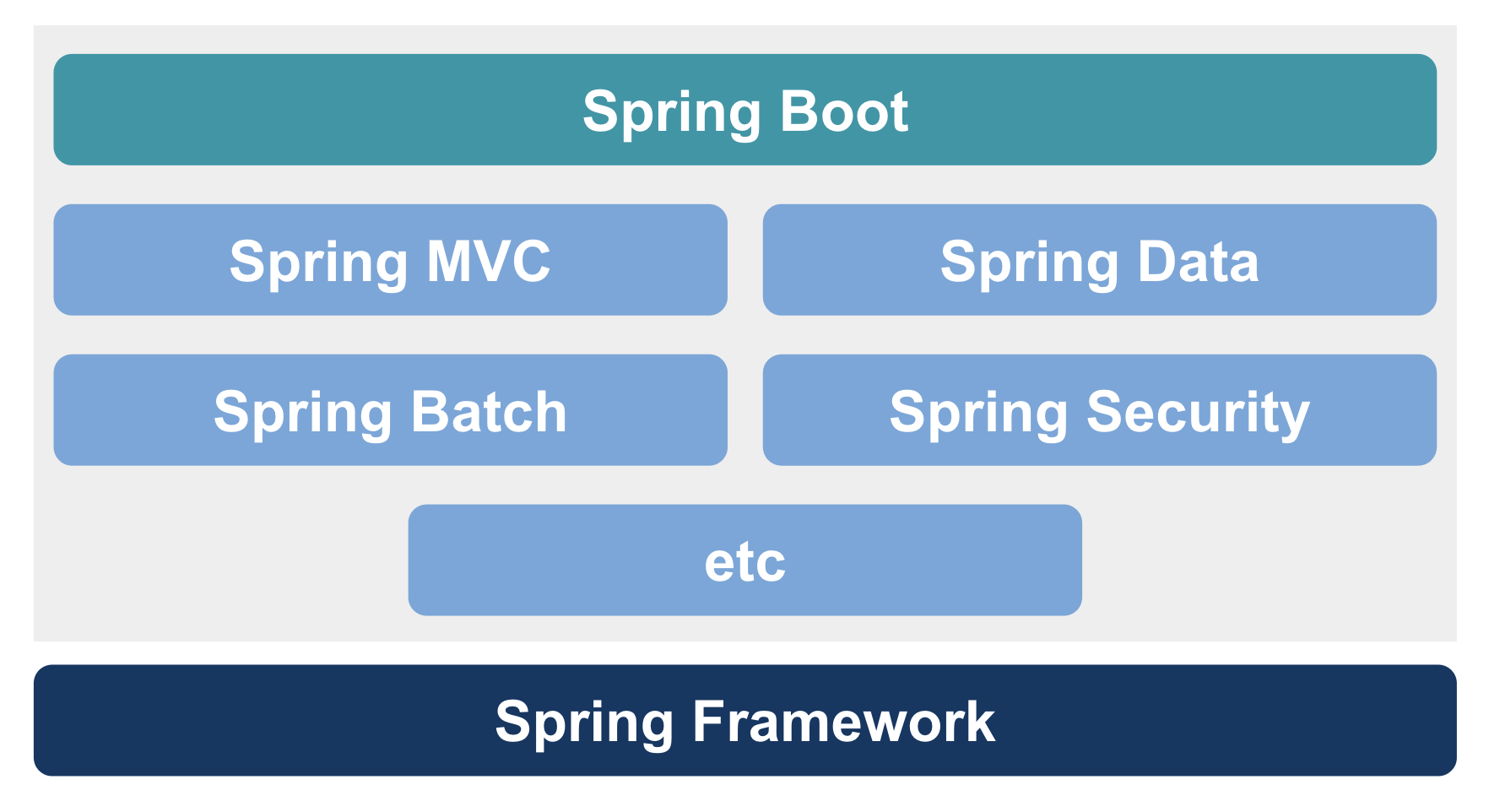
Spring Frameworkは開発を効率化するための様々な機能を提供しており、それぞれの機能ごとにフレームワークが存在している。  
そのため実現したいシステムに応じて使用するフレームワークを選ぶことができる。

| **フレームワーク** | **提供する機能の概要** |
| --- | --- |
| Spring MVC | Webアプリケーションを簡単に作成する機能を提供 |
| Spring Data | データアクセスに関する機能提供 |
| Spring Batch | バッチ処理機能を提供 |
| Spring Security | 認証認可の機能を提供 |

※「フレームワーク」とは  
開発を簡単にするための「骨組み」のこと。  
フレームワークを使うことで全てのプログラムを一から作る必要がないため、開発効率が上がる。

### **Spring Bootとは？**

Spring Frameworkに含まれるフレームワークの一つで、 **Spring Frameworkの機能を使いやすくするため** のフレームワーク。



## **SpringBootの特徴**

ここからはSpringBoot（正しくはSpring Framework）のコア機能である **「DI」** と **「AOP」** について説明します。  
結論、この二つの機能は **「システムの開発効率・保守性を高める」** ためのものです。

### **①DI**

「Dependency Injection」の略で、「依存性の注入」と訳される。  
この機能を「DIコンテナ」が担っている。

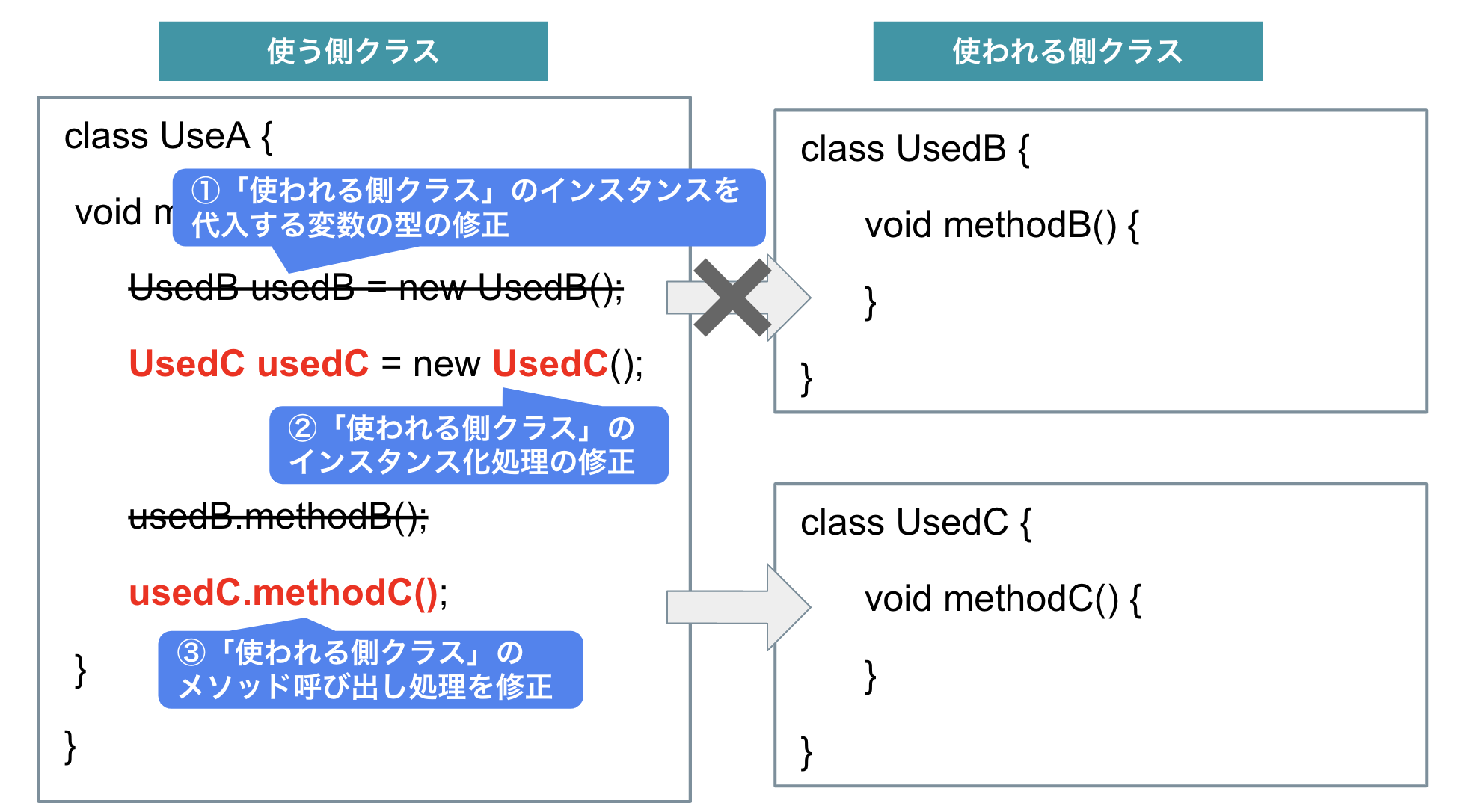
#### **「依存性」とは？**

「依存性」とは、 **あるクラス（＝使う側）が他のクラス（＝使われる側）を利用している** こと。

Javaには以下2種類の依存がある。  
この2種類の依存では、「使われる側クラス」に変更があった場合、 **「使う側クラス」の修正が必要** になる。

##### **1. クラス依存**

「使う側クラス」で「使われる側クラス」をインスタンス化し、メソッドを呼び出すこと。

**【「使われる側クラス」に修正が入った場合】**①「使われる側クラス」のインスタンスを代入する変数の型の修正  
②「使われる側クラス」のインスタンス化処理の修正  
③「使われる側クラス」のメソッド呼び出し処理を修正  


##### 

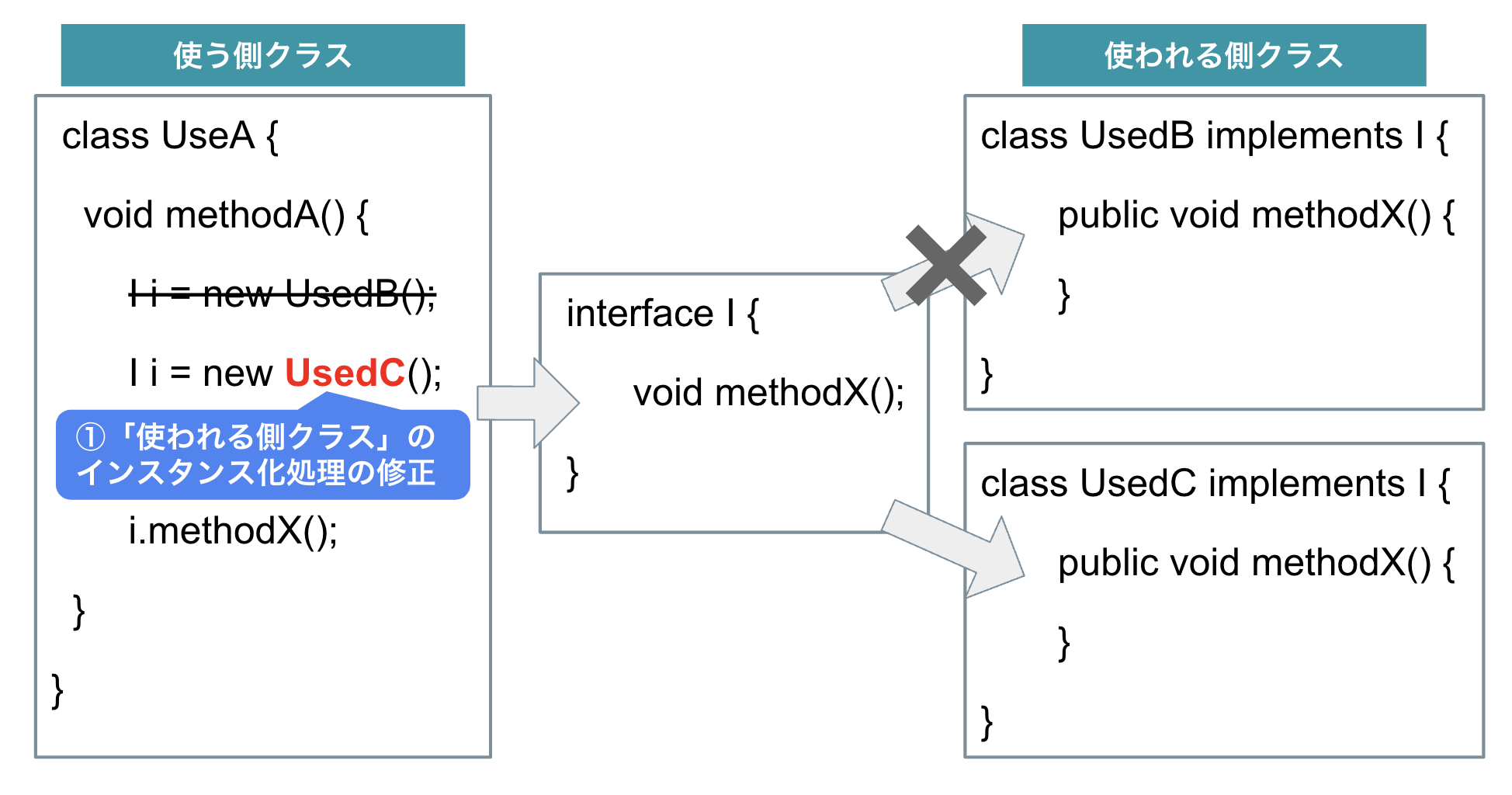
##### 

##### 

##### 

##### **2. インターフェース依存**

「使う側クラス」で「使われる側クラス」をインターフェース型でインスタンス化し、メソッドを呼び出すこと。

**【「使われる側クラス」に修正が入った場合】**①「使われる側クラス」のインスタンス化処理の修正  


このように、Javaには2種類の依存があるが、「インターフェース依存」は「クラス依存」よりも修正箇所が少なくなり、「**低依存**」な状態にできるというメリットがある。

しかし、「インターフェース依存」でも「使う側クラス」の修正は必要であり、  
「使う側クラス」が複数ある場合、修正漏れが起こる可能性が出てしまう。  
この問題を解消してくれるのがDIの機能。

#### 

#### 

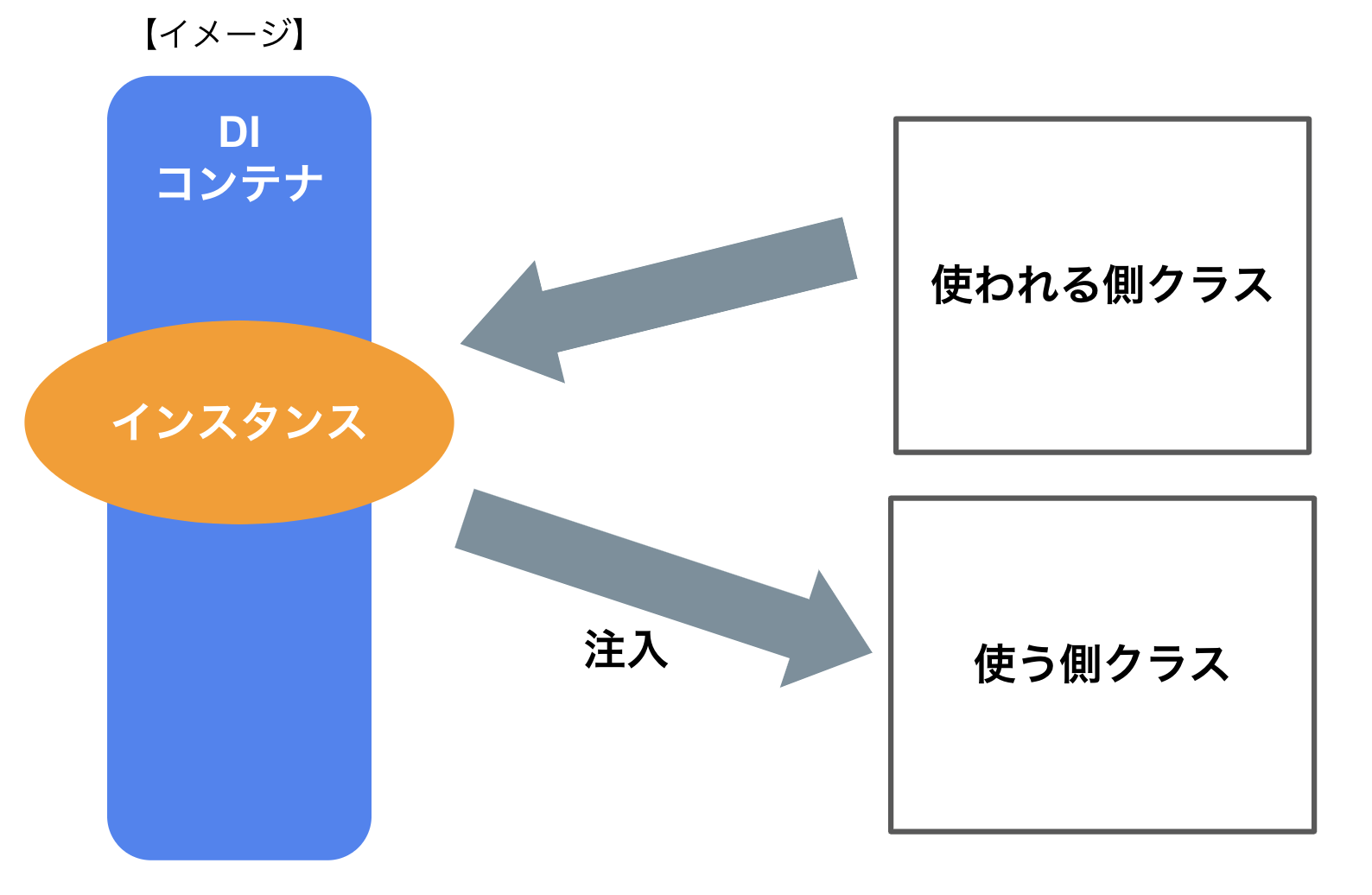
#### 

#### 

#### 

#### **DIコンテナの役割**

DIコンテナは **「使われる側クラス」をインスタンス化し、そのインスタンスを「使う側クラス」に注入** する。  
「クラス依存」と「インターフェース依存」に置いて、インスタンス生成はnewキーワードを用いて「使う側クラス」で行っているが、このインスタンス生成を代わりに行ってくれるのがDIコンテナ。  
※生成されたインスタンスは「Bean」と呼ばれる



#### 

#### 

#### 

#### 

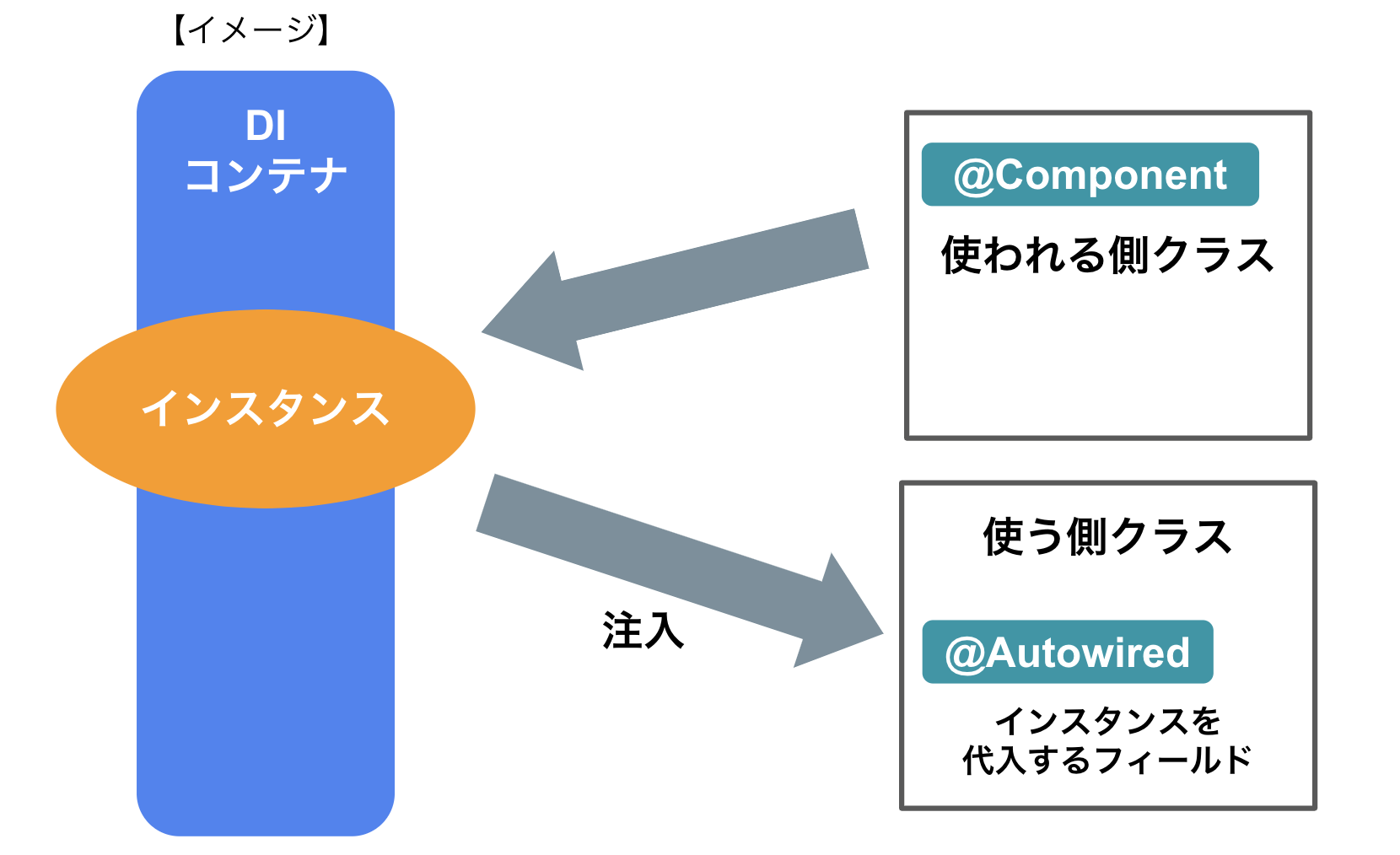
#### 

#### **DIプログラムの実装**

DIコンテナにインスタンス生成を行ってもらい、そのインスタンスを利用するために必要なのは、下記のように「**アノテーションを付与する**」ことである。

**①使われる側クラスに 「**[**@Component**](https://qiita.com/Component) **」アノテーションを付与する**　※「[@Controller](https://qiita.com/Controller)」「[@Service](https://qiita.com/Service)」「[@Repository](https://qiita.com/Repository)」もインスタンス生成アノテーション。  
**②使う側クラスで、インスタンスの参照を受け取るためのフィールドを宣言し、フィールドに 「**[**@Autowired**](https://qiita.com/Autowired)**」アノテーションを付与する**

Spring Bootは起動時に、「[@Component](https://qiita.com/Component)」など対象のアノテーションが付与されたクラスを探し、インスタンス化する。それらをDIコンテナに登録しておき、「[@Autowired](https://qiita.com/Autowired)」が付与されている箇所に注入している。



#### 

#### 

#### 

#### 

#### **DIのメリット**

* **「使う側クラス」の修正が不要になる**DIコンテナにインスタンス化を行ってもらうことで、「使われる側クラス」に修正が入った場合でも、**「使う側クラス」の修正は不要** になる。  
  「使われる側クラス」に付与しているアノテーションを修正するだけで、「使う側クラス」で使用したいインスタンスを変えることができる。  
  