目的・範囲: *Objective & Scope*	Adapter パターン
分類名: *Classification Name*	
著作者: *Author*	Phone Myint Thu
実施日: *Enforcement day*	2014年6月7日
バージョン: *Version*	
初版発行日: *Original Release*	
現行版発行日: *Current Release*	
キーワード: *Key Words*	
背景情報: *Background*	

◇目的
Adapter パターン（アダプター・パターン）とは、GoF(Gang of Four;
4人のギャングたち)によって定義されたデザインパターンの1つである。Adapter
パターンを用いると、既存のクラスに対して修正を加えることなく、インタフェースを変更することができる。Adapter
パターンを実現するための手法として継承を利用した手法と委譲を利用した手法が存在する。それぞれについて以下の節で説明する。

◇効果
継承を利用したAdapterは、利用したいクラスのサブクラスを作成し、そのサブクラスに対して必要なインタフェースを実装することで実現される。

◇背景
委譲を利用したAdapterは、利用したいクラスのインスタンスを生成し、
そのインスタンスを他クラスから利用することで実現される。

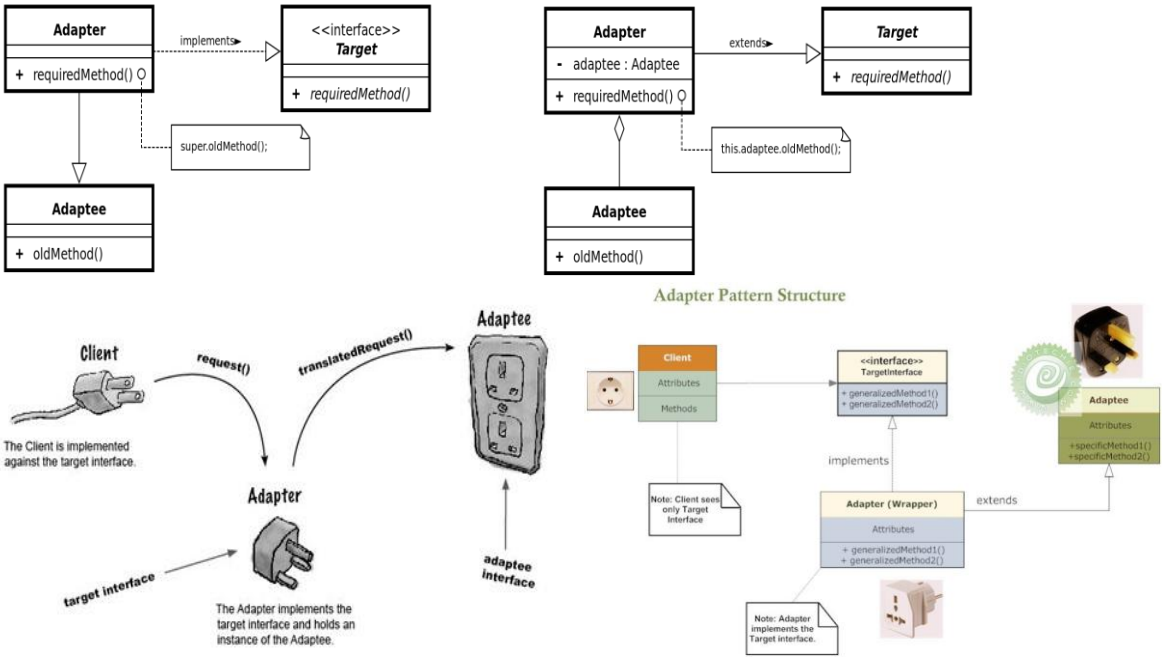
◇Adapterパターンの実際のコードと考え方
下記の例において、Productクラスは既存のクラスであり修正できないものとする。
ここで、Productクラスを利用したい開発者がいて、
その開発者はgetPriceというメソッドでProductの値段を取得したいとする。
この場合、ProductAdapterというAdapterを作成することで、既存クラス(Product)クラスを修正することなく、 異なるインタフェースを持たせることができる。
このように、既存クラスを修正することなく、異なるインタフェースを持たせるということが、Adapter パターンの役割である。

```
class ProductPrice

end

class Product
    def initialize()
    end

    def getCost()
          return @cost
     end
end

class ProductAdapter
   def getPrice()
          return self.getCost()
    end
end
```

◇問題点の改善

アダプター･パターンを実装する場合は、わかりやすくするために例DAOToProviderAdapterは、[インターフェイス]アダプタにクラス名[アダプタクラス名]を使用しています。
これは、パラメータとしてadapteeクラス変数とコンストラクタメソッドを持つ必要があります。このパラメータは、[インターフェイス]のアダプタに[AdapteeClassName]のインスタンスメンバに渡されます。

◇Adapterパターンのまとめ



◇注意
多くの場合、コードのシステムが相互に作用する必要があります。これは最初から設計されている場合、問題はない。
1つのシステムは、抽象化を定義し、他のシステムは、そのインターフェイスの実装を追加します。

抽象化を使用して、システムがその要件を変更したときに問題が発生します。実装の変更は、そのインタフェースは現在壊れている場合。
◇総括
ソフトウェアエンジニアリングでは、アダプタパターンは、
既存のクラスのインタフェースが別のインターフェイスから使用できるようにするソフトウェアの設計パターンである
これは、多くの場合、既存のクラスはそれらのソースコードを変更せずに他の人と動作させるために使用される。

用語	説明

※引用文献e-words、WikiPedia