Adapter Pattern

デザインパターン勉強会

#dezapatan

@cafenero_777

Adapter pattern

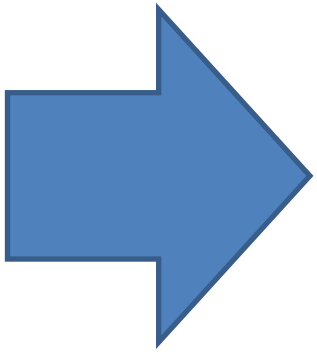
- 世の中は都合が合わないことだらけである。
  - JsonとphpSerializeの違い
  - WindowsとMacの違い
  - MysqlとoracleのDB違い
  - うちで使ってる炊飯器と君の家で使ってる炊飯器の違い

“都合が合わない事”の共通項

- やりたいことは同じ
- 方法が“著しく”違う

どうしよう？

- “方法”を“やりたいこと”で抽象化する
  - 抽象化したインターフェスに対して“やりたいこと”を投げる
  - 実際の“方法”はカプセル化される

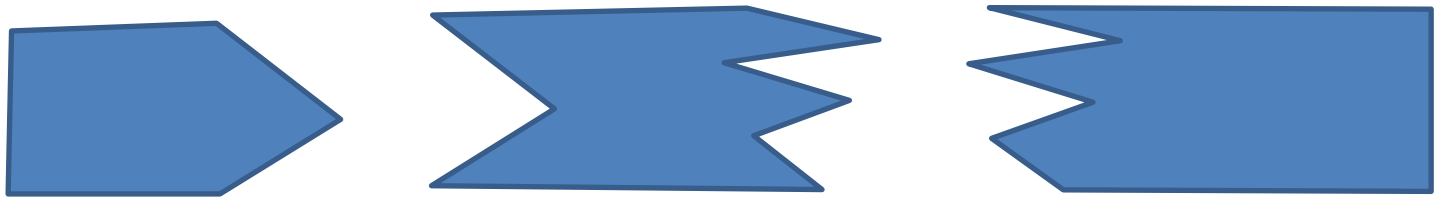


Adapter Pattern

Adapter Patternの定義

- Adapterパターンは、クラスのインターフェースをクライアントが期待する別のインターフェースに変換します。
- アダプタは、互換性のないインターフェースのためにそのままでは連携できないクラスを連携させます。

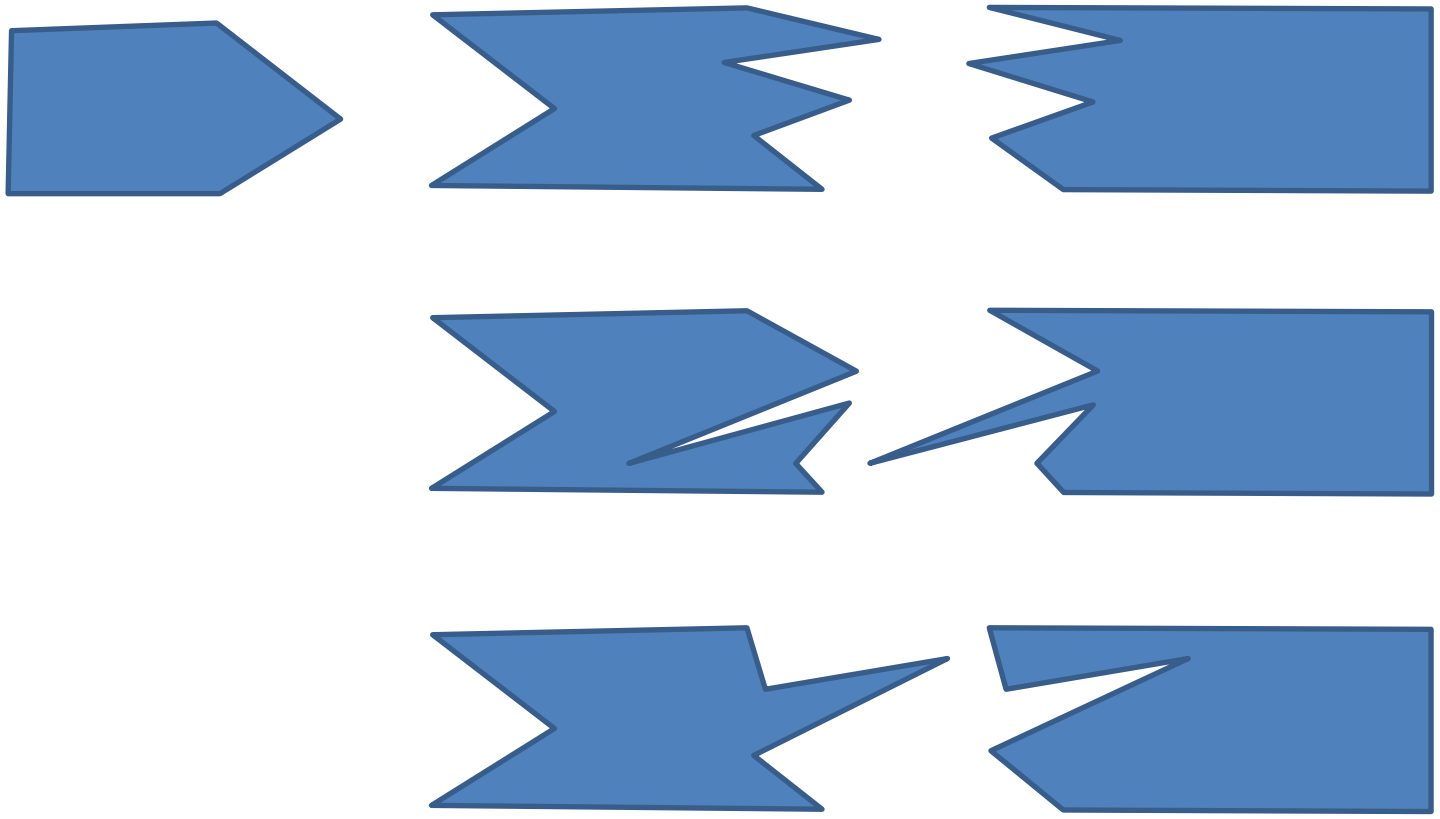
イメージ



クライアント　　アダプタ　　（変更できない）クラス
アダプティ

そのままでは互換性がないが、
アダプタを使うことにより解決
クライアントを変更する必要なし！

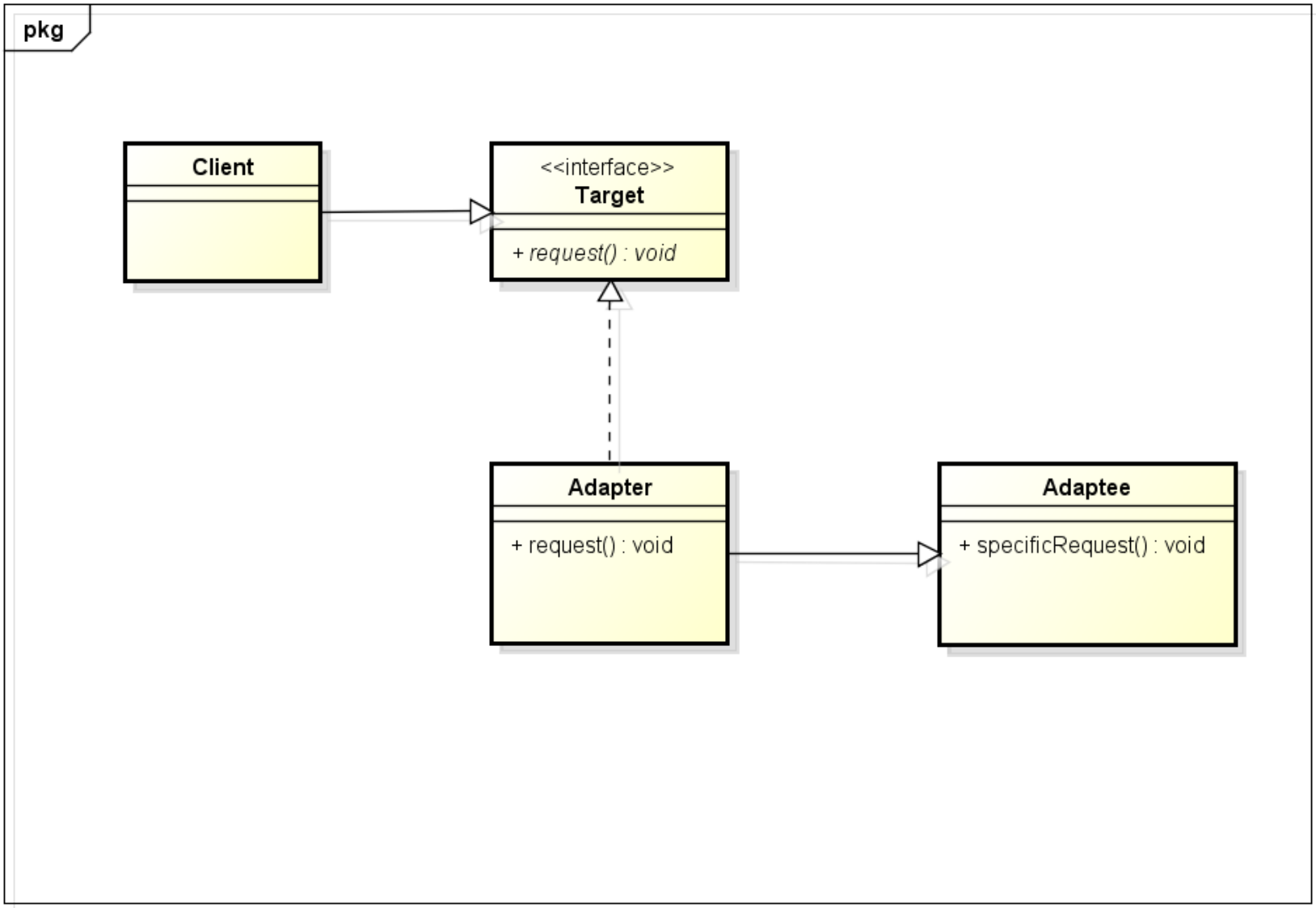
アダプティが増えたら？



クライアント　　アダプタ　　（変更できない）クラス
アダプティ

クライアントは変更する必要なし！

クラス図で書くと



今回のAdapter Patternストーリー

- 各マリモにIPアドレスを割り振ろう

192.168.1.1
255.255.255.0

192.168.1.2
255.255.255.0

192.168.1.254
255.255.255.0

おや、マリモの様子が・・・

- 各マリモにIPアドレスを割り振ろう
- 懸念点:
  - 各種ベンダーによって設定方法が異なる。
  - Cisco社のマリモ
  - Linuxのマリモ
  - Hoge社のマリモ
  - Etc,etc,etc,,

素直に実装してみる

- 各ベンダーマリモクラスを直接操作
  - ベンダークラスのメソッドを直接たたく

素直に実装した問題点

- ユーザがすべてのベンダー仕様を把握しなければならない

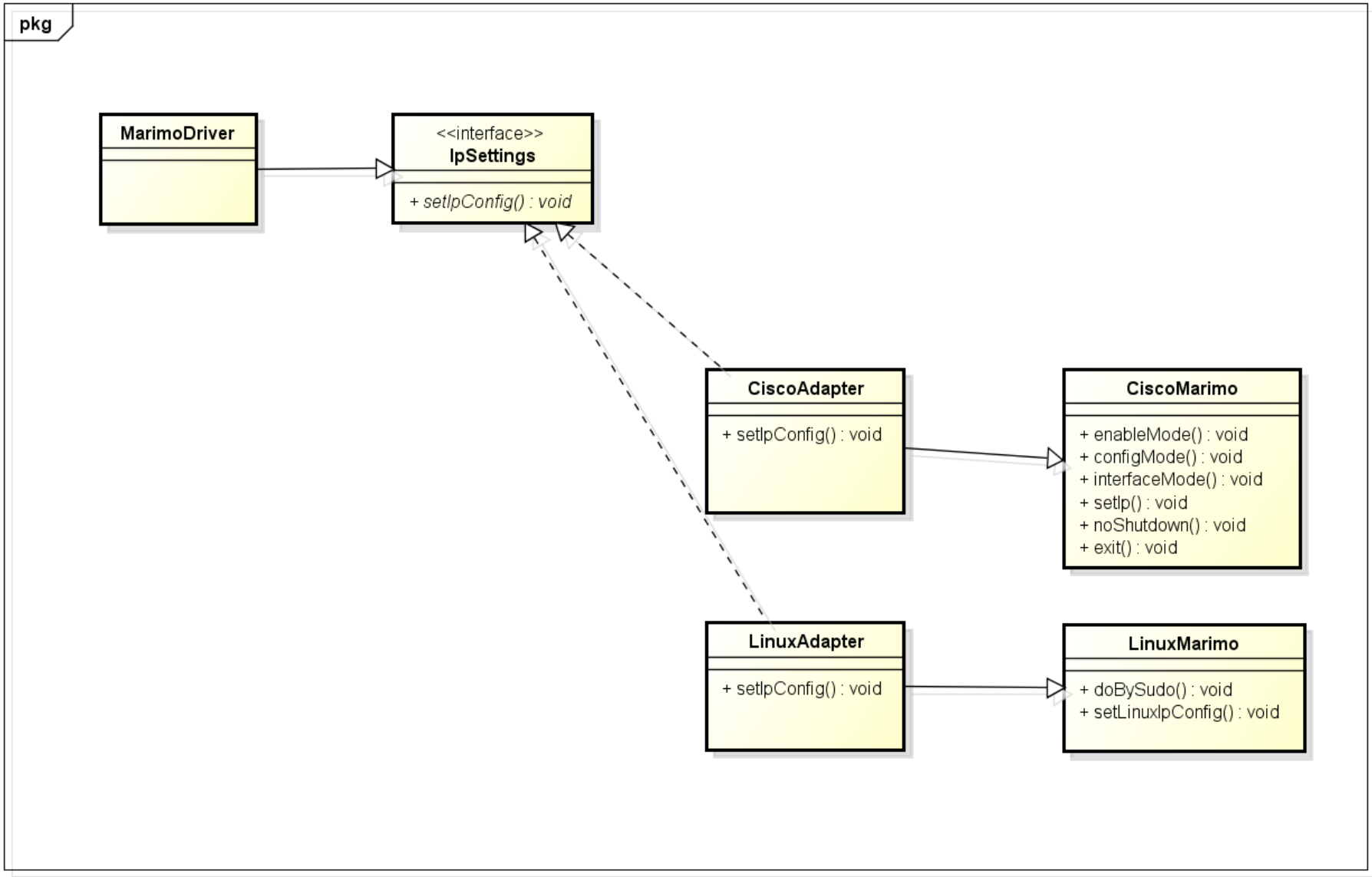
```
CiscoMarimo cMarimo = new CiscoMarimo();
cMarimo.enableMode();
cMarimo. configMode();
cMarimo.interfaceMode();
cMarimo.setIP(ip, subnet, interfaceName);
cMarimo.noShutdown();
cMarimo.exit();
```

- ベンダー仕様が“少しでも”変わるとすべてのクライアントを変更する必要がある
- ベンダー仕様が増えると、クライアント側で対応する必要がある
- 正直しんどい

Adapter Patternを使う

- ユーザはインターフェースに対して操作を行う。

クラス図



改善点

- ユーザはインターフェース（アダプタの共通仕様）を知るだけでよい

```
IpSettings cAdaptor = new CiscoAdaptor(new CiscoMarimo());
cAdaptor.setIpConfig(ip, subnet, interfaceName);
```

- ユーザはアダプティ（各ベンダー仕様）を気にしなくて良い

- 楽ちん

Web業界でよく見たことがあるような・・・

- もしかして: ***API***

忘れてはいけないこと

- Adapterクラス作成に必要なたった3つの事
  - 全てのアダプティ仕様を網羅して
  - 各種アダプタクラスを作り
  - インターフェースを策定する

- Adapter Patternを使うときは“ありがとう”と言う。

おしまい