Java言語で学ぶ デザインパターン入門

12/5 関嶋研B4 坂本亘

15章、16章

1

Facadeパターン(ファサード)

- プログラムが複雑になっていき、一つのクラスが肥大化していったり、複数のクラス同士が相互にやりとりを行うと、全体を把握しにくくなり、見通しが悪くなる。
- そこで、全体を把握してる窓口となるクラスを新たに設る

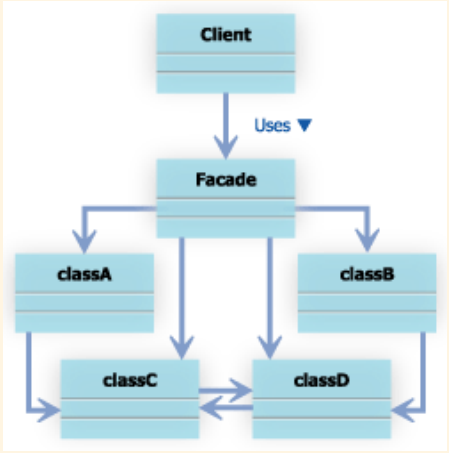
2

Facadeパターン(ファサード)

- Facadeは「建物の正面」という意味
- もっというと、シンプルな正面玄関
- 正面玄関を入ると複雑な機能があったりするが、よび出す側は、シンプルな正面玄関だけを意識すればいい

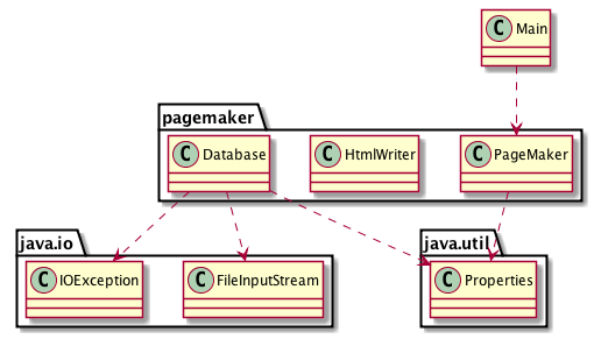
3

クラス図



4

サンプルプログラムのクラス図



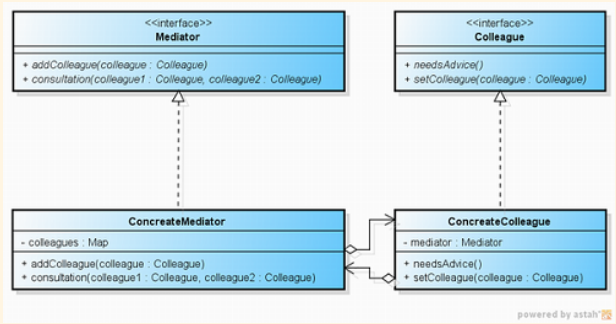
Mediator(メディエイター)

- mediatorは仲介者のこと
- さっきは窓口から各機能への一方向だったが、今度は各機能から窓口(仲介者)への双方向になる

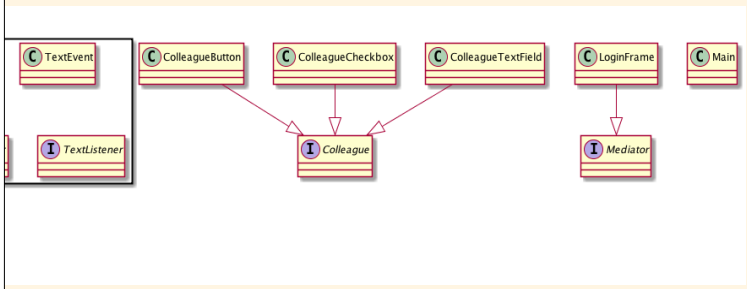
使いたいとき

- 複数のクラスのそれぞれの状態で、全体の動作を変えたいとき
- それぞれのクラスに場合分けを書いていたら、しんどい
- そこで、それぞれのクラスは状態が変わったら、新しいあるべき状態を仲介役に尋ねる
- GUIで効果的

クラス図



サンプルコードのクラス図



練習問題(Facade)

- pagemakerパッケージの外から利用できるのはPageMakerクラスだけにしたいので、DatabaseクラスとHtmlWriterクラスをpagemakerパッケージの外から利用できないようにせよ
- maildata.txtに含まれるユーザーのメールアドレスのリンク集を作成するmakeLinkPageメソッドをPageMakerクラスに追加せよ
  呼び出しは、PageMaker.makeLinkPage(“linkpage.html”)となる

練習問題(Mediator)

- ユーザーログインのときユーザー名とパスワードの両方が4文字以上の場合のみOKボタンが有効になるという仕様を満たすようにせよ
- ColleagueButton、ColleagueTextField、ColleagueChckboxみんなmediatorというフィールドと、setMediatorを持っているが、シンプルにするために、ColleagueインターフェースにmediatorというフィールドをいれてsetMediatorメソッドを実装できるか