んーなんというか、必読でしょう。ちょっと難解ですが。

パターンカタログ

生成に関するパターン

■ ABSTRACT FACTORY

目的

関連し合うオブジェクト群を、具象クラスを明確にせずに生成するためのインターフェースを提供する

別名

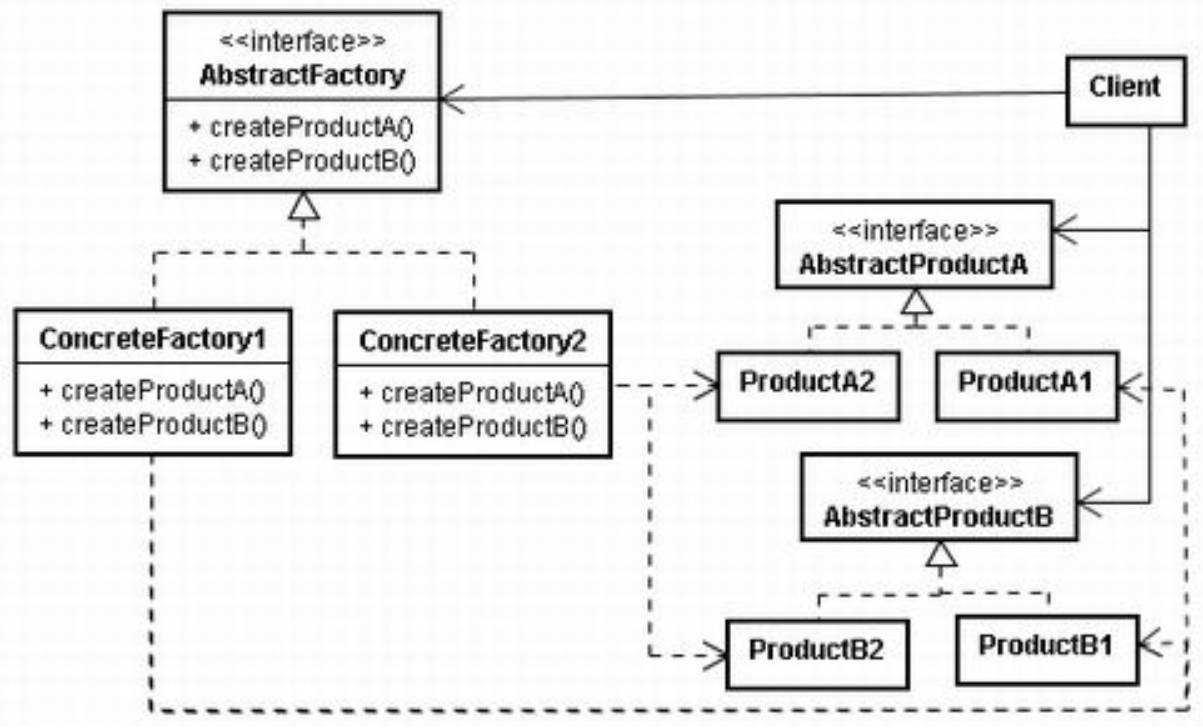
Kit

構成要素

- AbstractFactory
  - AbstactProduct オブジェクトを生成するオペレーションのインターフェースを宣言
- ConcreteFactory
  - ConcreteProduct オブジェクトを生成するオペレーションを実装
- AbstractProduct
  - 部品ごとにインターフェースを宣言
- ConcreteProduct
  - 対応する ConcreteFactory オブジェクトで生成される部品オブジェクトを定義
  - AbstractProduct のインターフェースを実装
- Client
  - AbstractFactory と、AbstractProduct で宣言されたインターフェースのみを利用

結果

- 具象クラスの局所化
- 部品の集合を用意に変更できる
- 部品間の無矛盾せいを促進
- 新たな種類の部品に対応することは困難



■ BUILDER

目的

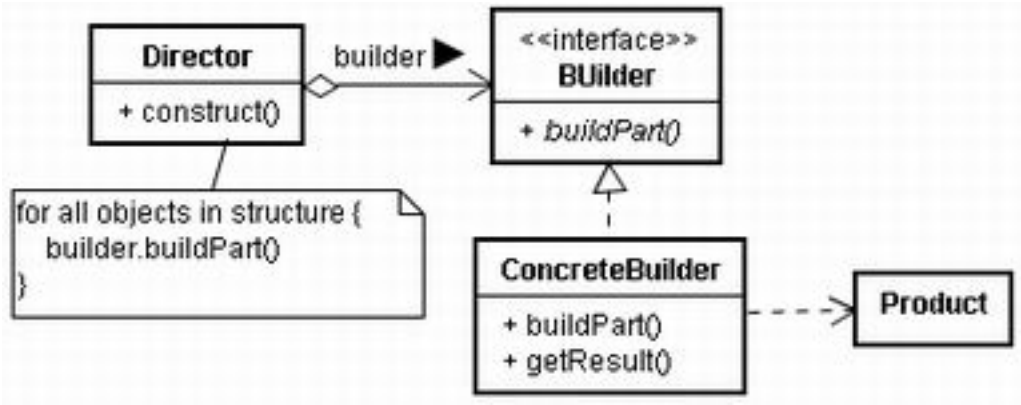
複合オブジェクトについて、作成過程を表現形式に依存しないものとし、同じ作成過程で異なる表現形式のオブジェクトを生成できるようになる

構成要素

- Builder
  - Product オブジェクトの構成要素を生成するための抽象化されたインターフェースを規定
- ConcreteBuilder
  - Builder クラスのインターフェースを実装することで、Product オブジェクトの構成要素の生成や組み合わせを行う
- Director
  - Builder クラスのインターフェースを使って、オブジェクトを生成する
- Product
  - 多くの構成要素からなる複合オブジェクトをあらわす。ConcreteBuilder クラスは、Product オブジェクトの内部表現を作成し、また組み立ての過程を定義
  - 構成要素を定義するクラス、構成要素を Product オブジェクトに組み合わせていくためのインターフェースを含む

結果

- Product オブジェクトの内部表現の変更が可能になる
- 生成や表現のためのコードを局所化する
- Product オブジェクトの生成過程をより細かくコントロールできるようになる



■ FACTORY METHOD

目的

オブジェクトを生成するときのインターフェースのみを規定し、実際にどのクラスをインスタンス化するかはサブクラスに任せる

別名

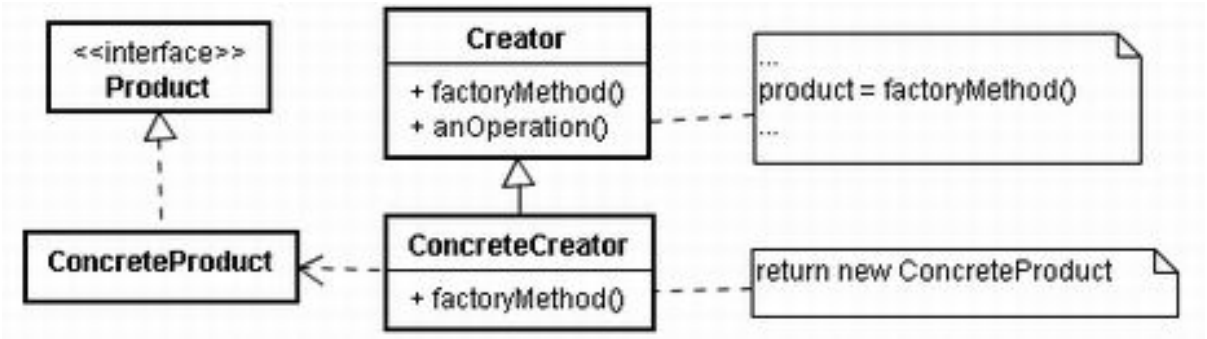
Virtual Constructor

構成要素

- Product
  - factory method が生成するオブジェクトのインターフェースを定義
- ConcreteProduct
  - Product クラスのインターフェースを実装
- Creator
  - Product 型のオブジェクトを返す factory method を宣言する
- ConcreteCreator
  - ConcreteProduct クラスのインスタンスを返すように、factory method をオーバーライド

結果

- オブジェクトを拡張する場合の手がかりをサブクラスに提供
- パラレルなクラス階層をつなぐ



■ XXXXXXX

目的

別名

構成要素

-
  -
-
  -
-
  -

結果