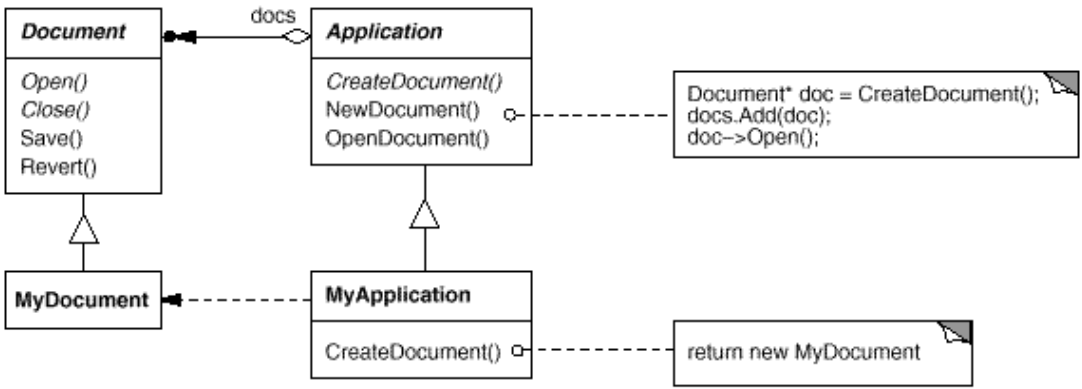
Factory Method (Metoda wytwórcza)
(Wirtualny Konstruktor)

Cel:

Zdefiniowanie interfejsu do tworzenia obiektów i pozwolenie klasom potomnym na decydowanie jakiej klasy obiekt utworzyć.

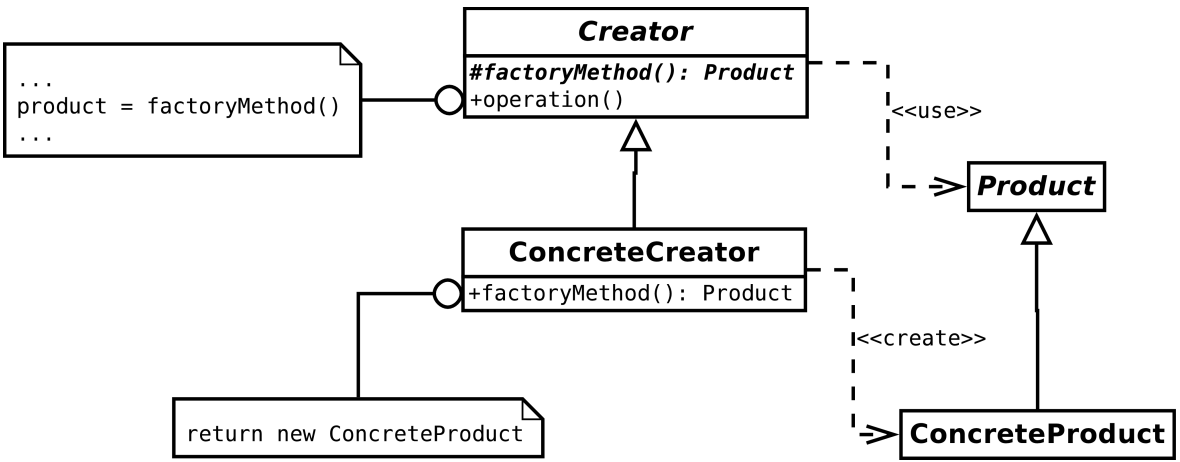
Przykład:



Zastosowanie:

- gdy klasa nie jest w stanie przewidzieć jakiej klasy obiekty ma tworzyć.
- gdy klasa chce, by to jej klasy potomne decydowały jakie obiekty będą tworzone.
- gdy klasa składa część odpowiedzialności na jedną z kilku pomocniczych podklas i chcemy wiedzieć której z tych pomocniczych klas użyć.

Struktura:



Składniki:

- Product – definiuje interfejs obiektu tworzonego przez fabrykę.
- ConcreteProduct – implementuje ten interfejs.
- Creator – deklaruje metodę wytwórczą zwracającą obiekt klasy Product. Może również udostępniać domyślną implementację tej metody, a także wywoływać ją w celu tworzenia produktów.
- ConcreteCreator – implementuje metodę wytwórczą, zwracając instancję konkretnego produktu.

Zależności:

Kreator zostawia swym klasom potomnym zdefiniowanie Factory Method i zwrócenie instancji właściwego Produktu.

Konsekwencje:

1. Kod operuje jedynie interfejsem Produktu, nie zależy od jakiegoś specyficznego ConcreteProduct.
2. Hermetyzacja procesu tworzenia obiektu – użyteczne, gdy jest to złożoną operacją.
3. Zostawia miejsce na ewentualne rozszerzanie Produktu lub Fabryki.
4. Wiąże równoległe hierarchie klas – metoda wytwórcza może być wywoływana nie tylko przez samą fabrykę, ale i zewnętrznego klienta, który dzięki temu uzyskuje właściwy produkt dla danej fabryki.

5. Potencjalny minus Factory Method – może okazać się, że klient musi dziedziczyć z klasy Kreatora wyłącznie po to, aby stworzyć własny ConcreteProduct.

Implementacja

1. Dwa podstawowe warianty: (1) klasa Kreatora jest abstrakcyjna i nie zapewnia implementacji dla metody fabrykującej, (2) Kreator jest konkretną klasą i udostępnia domyślną implementację metody fabrykującej. W pierwszym wypadku musimy dziedziczyć, w drugim Factory Method istnieje właściwie tylko dla przyszłej rozszerzalności.
2. Sparametryzowana metoda wytwórcza. Jedna metoda do tworzenia wielu rodzajów produktów. Przyjmuje ona parametr identyfikujący rodzaj obiektu, jaki ma być stworzony. Wszystkie obiekty tworzone przez Factory Method mają wspólny interfejs Produktu. Dziedziczenie z takiego Kreatora pozwala wybiórczo zmienić, rozszerzyć lub dodać gatunki tworzonych Produktów.

```
class Creator {
    Product create(ProductId id) {
        if (id == MINE)  return new MyProduct();
        if (id == YOURS) return new YourProduct();
        // ...
        return null;
    }
}
```

Przykładowy kod:

```
abstract class Shape {
     public abstract String draw();
}
class TriangleShape extends Shape {
     public String draw() { return "triangle"; }
}
class CircleShape extends Shape {
     public String draw() { return "circle"; }
}
class SquareShape extends Shape {
     public String draw() { return "square"; }
}
```

```
class FiguresFactory {
     public enum ShapeType {
          Triangle,
          Square,
          Circle
     }
     public static Shape createFigure(ShapeType type) {
          switch(type) {
               case Triangle: return new TriangleShape();
               case Circle:   return new CircleShape();
               case Square:   return new SquareShape();
          }
          throw new IllegalArgumentException();
     }
}
class Client {
     public static void main (String args[]) {
          Shape figure = FigureFactory.createFigure(
                              FiguresFactory.ShapeType.Circle);
          figure.draw();
        }
    }
}
```

Powiązania:

- Abstract Factory – przeważnie implementowana z użyciem Factory Methods.
- Factory methods są zazwyczaj wołane z użyciem Template Methods (np. metoda NewDocument z przykładu)