[GoF] Abstract/Concrete Factory

Leonardo Bizzoni

June 23, 2024

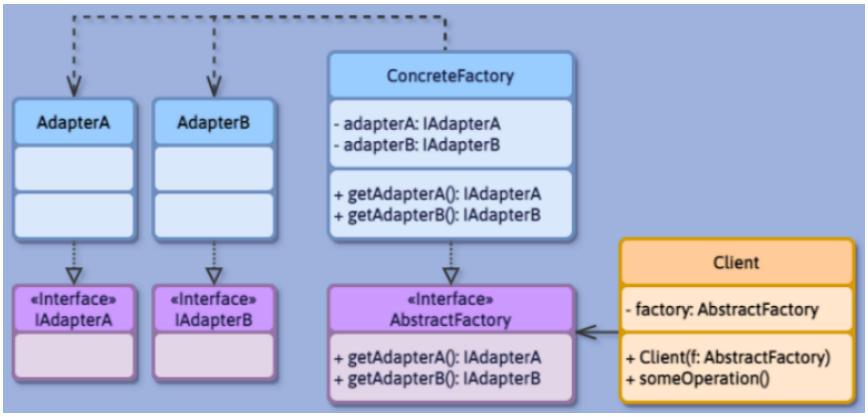
Non si tratta di un pattern GoF, ma è comunque un pattern molto diffuso.

1 Problema

Chi deve essere responsabile della creazione di oggetti quando ci sono delle considerazioni speciali, come una logica di creazione complessa, quando si desidera separare le responsabilità di creazione per una coesione migliore, e così via?

2 Soluzione

Crea un oggetto pure fabrication che gestisce la creazione.



2.1 Esempio

```
enum BurgerType { Beef, Veggie }
```

```
trait Burger {
  fn prepare(&self);
}

struct Restaurant;
impl Restaurant {
  fn orderBurger(&self, request: BurgerType) -> Box<dyn Burger> {
    let factory = SimpleBurgerFactory::new();

    let burger = factory.createBurger(request);
    burger.prepare();
    burger
  }
}

struct SimpleBurgerFactory;
impl SimpleBurgerFactory {
  fn new() -> Self {
    Self { }
  }

  fn createBurger(&self, of_type: BurgerType) -> Box<dyn Burger> {
    match of_type {
      BurgerType::Beef => Box::new(BeefBurger),
      BurgerType::Veggie => Box::new(VeggieBurger),
    }
  }
}

struct BeefBurger;
impl Burger for BeefBurger {
  fn prepare(&self) {
    // Implementation
  }
}

struct VeggieBurger;
impl Burger for VeggieBurger {
  fn prepare(&self) {
```

```
    // Implementation
  }
}
```