

Wzorce projektowe

Laboratorium 5. Obserwator + stan

Obserwator

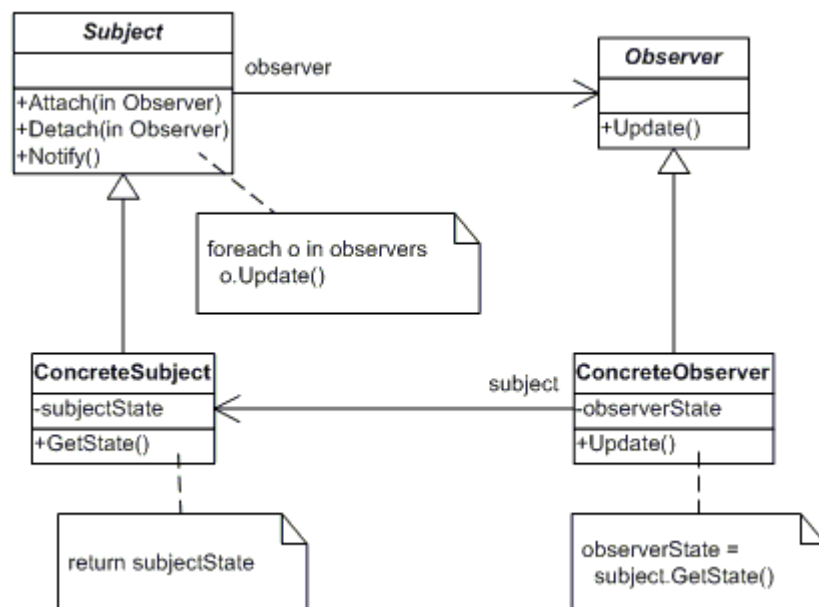
Ogólnie

Definiuje zależności typu jeden-do-wiele. Dzięki temu w momencie zmiany stanu jednego obiektu, wszystkie obiekty zależne są o tym powiadamiane i aktualizowane automatycznie.

Przykłady

- Prezentowanie danych liczbowych. Użytkownik edytuje tabelę zawierającą wyniki a graficzne przedstawienie na wykresach różnego typu jest aktualizowane automatycznie.
- Projektowanie obiektu w monecie gdy nie znana jest jeszcze lista obiektów, które będą od niego zależać lub lista ta może być zmienna. Wykorzystanie obserwatora pozwala na dodawanie nowych obiektów zależnych bez potrzeby zmieniania obiektu, od którego zależą.

Struktura



- **Subject** – obiekt, od którego stanu mogą zależeć inne,
- **ConcreteSubject** – implementacja obiektu, od którego stanu zależą inne,
- **Observer** – obiekt zależący od innego,
- **ConcreteObserver** – implementacja obiektu zależnego.

Przykładowa implementacja w języku Java:

```
public abstract class Subject {
    private List<Observer> observers;

    public Subject() {
        this.observers = new ArrayList<Observer>();
    }

    public void attach(Observer observer) {
        observers.add(observer);
    }

    public void detach(Observer observer) {
        observers.remove(observer);
    }

    public void notify() {
        for (Observer o : observers) {
            o.update(this);
        }
    }
}
```

```
public abstract class Observer {
    public abstract void update(Subject subject);
}
```

```
public class ConcreteObserver extends Observer {
    @Override
    public void update(Subject subject) {
        System.out.println("state updated");
    }
}
```

Stan

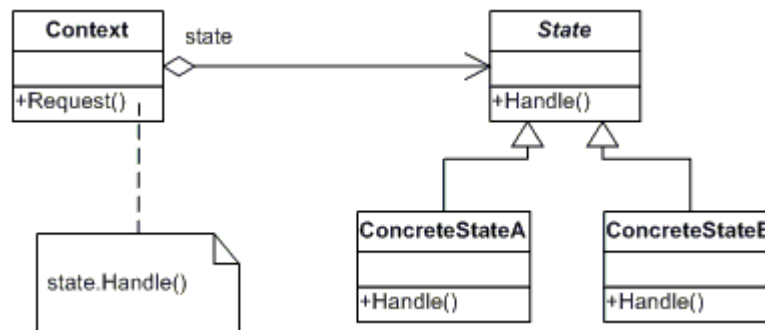
Ogólnie

Pozwala na zmianę zachowania obiektu w momencie zmiany jego stanu.

Przykłady

- Klasa reprezentująca połączenie. Połączenie może być w kilku stanach (*połączony*, *niepołączony*, *nasłuchujący*). W momencie gdy na obiekcie połączenia wywoływana jest jakaś metoda, jej działanie będzie różne w zależności od stanu w jakim znajduje się obiekt.

Struktura



Przykładowa implementacja w języku Java:

```
public abstract class State {
    public abstract void handle(Context context);
}
```

```
public class ConcreteStateA extends State {
    @Override
    public void handle(Context context) {
        context.setState(new ConcreteStateB());
    }
}
```

```
public class ConcreteStateB extends State {
    @Override
    public void handle(Context context) {
        context.setState(new ConcreteStateA());
    }
}
```

```
public class Context {
    private State state;

    public Context() {
        this.state = new ConcreteStateA();
    }

    public State getState() {
        return this.state;
    }

    public void setState(State state) {
        this.state = state;
    }

    public void request() {
        state.handle(this);
    }
}
```

Zadanie

Należy zaimplementować klasę, której obiekty będą mogły się znajdować w trzech różnych stanach a następnie trzy różne klasy typu obserwator prezentujące w różny sposób aktualny stan obiektów. Rodzaje stanów dostępnych dla obiektów oraz sposobu przedstawienia ich przez obserwatorów zostaną podane przez prowadzącego.