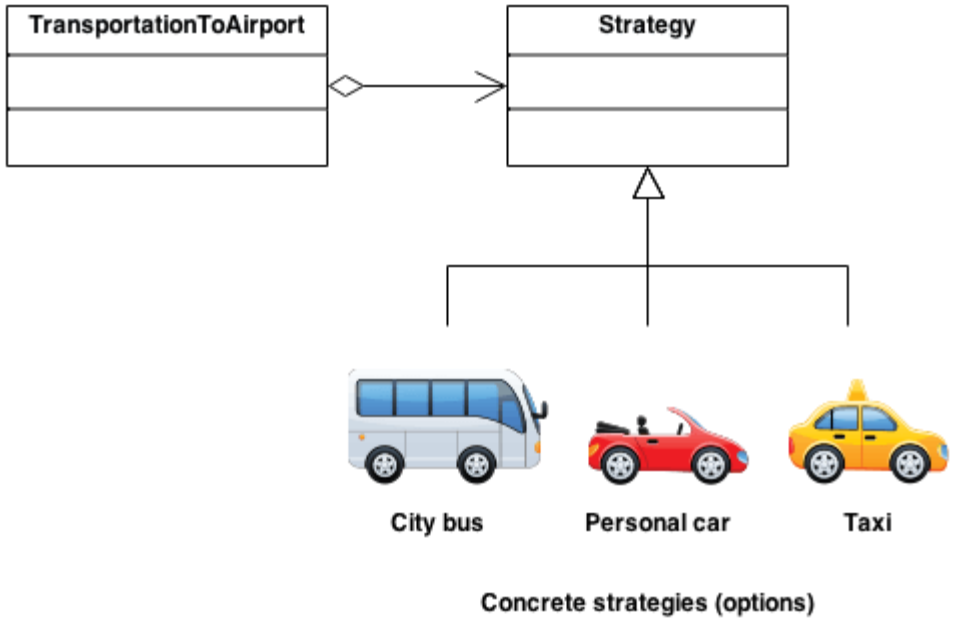
بررسی الگوهای طراحی– Strategy

توسط : محمد حسین زارع
@zare88r : twitter
telegram
Join Us : @JavaLandCH

مفاهیم

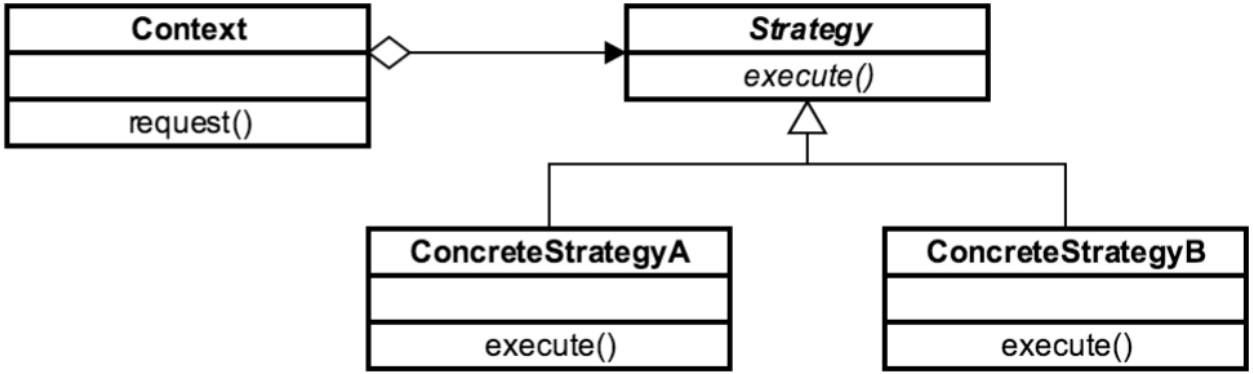
- حذف عبارات شرطی
- Encapsulate کردن رفتار در کلاس های مختلف
- کلاینت نیاز به دانستن استراتژی های مختلف دارد
- کلاینت استراتژی اش را انتخاب میکند
- نمونه ها
  - java.util.Comparator

مفاهيم


TransportationToAirport
Strategy
City bus
Personal car
Taxi
Concrete strategies (options)

طراحی

- Context
- Abstract Strategy – Concrete Strategy

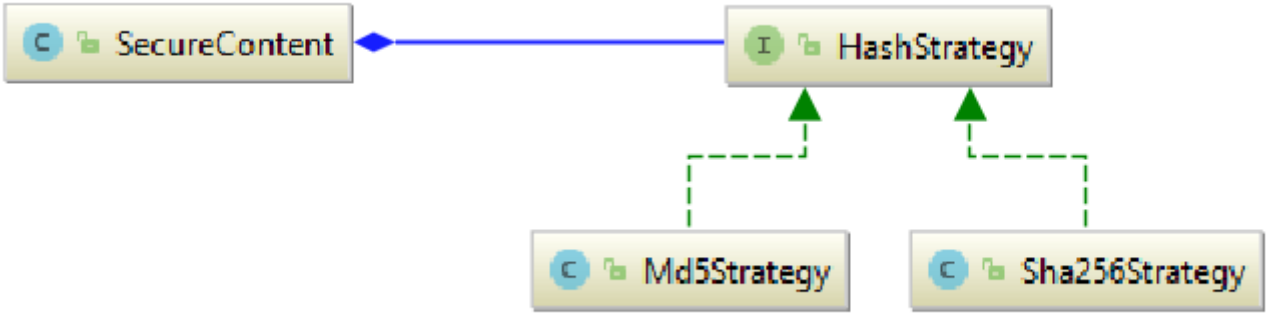


مثال 1

```
class Person{  int age; String name;}
List<Person> people= Arrays.asList(new Person(12 , "Jack")
 , new Person(14 , "Joe"));

Collections.sort(people, new Comparator<Person>() {
    @Override
    public int compare(Person o1, Person o2) {
        if(o1.getAge() > o2.getAge()) {
            return 1;
        }
        if(o1.getAge() < o2.getAge()) {
            return -1;
        }
        return 0;
    }
});
```

یک نمونه


SecureContent
HashStrategy
Md5Strategy
Sha256Strategy

یک نمونه

```
public class SecureContent {

    private String raw;

    public SecureContent(String raw) {
        this.raw = raw;
    }

    public byte[] hashContent(HashStrategy hashStrategy){
        return hashStrategy.hash(raw);
    }
}
```

یک نمونه

```
public interface HashStrategy {
    byte[] hash(String raw);
}


public class Md5Strategy implements HashStrategy {
    public byte[] hash(String raw) {
        return DigestUtils.md5(raw);
    }
}


public class Sha256Strategy implements HashStrategy {
    public byte[] hash(String raw) {
        return DigestUtils.sha256(raw);
    }
}
```

جمع بندی

- جداسازی الگوریتم ها
- کلاینت از استراتژی های مختلف اطلاع دارد
- به ازای هر استراتژی یک کلاس باید ساخته شود