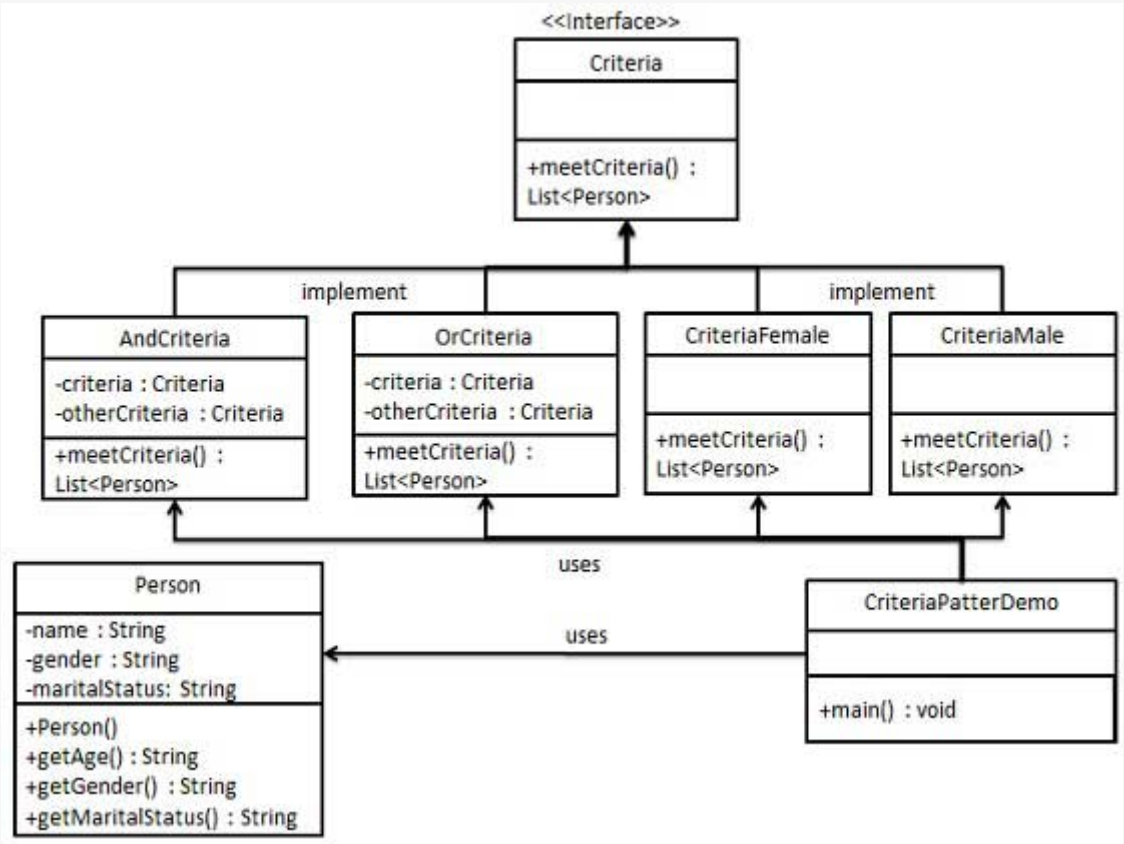
桥接模式 组合模式

# 过滤器模式

过滤器模式（Filter Pattern）或标准模式（Criteria Pattern）是一种设计模式，这种模式允许开发人员使用不同的标准来过滤一组对象，通过逻辑运算以解耦的方式把它们连接起来。这种类型的设计模式属于结构型模式，它结合多个标准来获得单一标准。

## 实现

我们将创建一个 *Person* 对象、*Criteria* 接口和实现了该接口的实体类，来过滤 *Person* 对象的列表。*CriteriaPatternDemo*，我们的演示类使用 *Criteria* 对象，基于各种标准和它们的结合来过滤 *Person* 对象的列表。




### 步骤 1

创建一个类，在该类上应用标准。

**Person.java**

```
public class Person {
private String name;
private String gender;
private String maritalStatus;
public Person(String name,String gender,String maritalStatus){
this.name = name;
this.gender = gender;
this.maritalStatus = maritalStatus;
}
public String getName() {
```

```
return name;
}
public String getGender() {
return gender;
}
public String getMaritalStatus() {
return maritalStatus;
}
}
```

## 步骤 2

为标准（Criteria）创建一个接口。

**Criteria.java**

```
import java.util.List;
public interface Criteria {
public List<Person> meetCriteria(List<Person> persons);
}
```

## 步骤 3

创建实现了 *Criteria* 接口的实体类。

**CriteriaMale.java**

```
import java.util.ArrayList;
import java.util.List;
public class CriteriaMale implements Criteria {
@Override
public List<Person> meetCriteria(List<Person> persons) {
List<Person> malePersons = new ArrayList<Person>();
for (Person person : persons) {
if(person.getGender().equalsIgnoreCase("MALE")){
malePersons.add(person);
}
}
return malePersons;
}
}
```

**CriteriaFemale.java**

```
import java.util.ArrayList;
import java.util.List;
public class CriteriaFemale implements Criteria {
@Override
public List<Person> meetCriteria(List<Person> persons) {
List<Person> femalePersons = new ArrayList<Person>();
for (Person person : persons) {
if(person.getGender().equalsIgnoreCase("FEMALE")){
femalePersons.add(person);
}
}
```

```
return femalePersons;
}
}
```

### CriteriaSingle.java

```
import java.util.ArrayList;
import java.util.List;
public class CriteriaSingle implements Criteria {
@Override
public List<Person> meetCriteria(List<Person> persons) {
List<Person> singlePersons = new ArrayList<Person>();
for (Person person : persons) {
if(person.getMaritalStatus().equalsIgnoreCase("SINGLE")){
singlePersons.add(person);
}
}
return singlePersons;
}
}
```

### AndCriteria.java

```
import java.util.List;
public class AndCriteria implements Criteria {
private Criteria criteria;
private Criteria otherCriteria;
public AndCriteria(Criteria criteria, Criteria otherCriteria) {
this.criteria = criteria;
this.otherCriteria = otherCriteria;
}
@Override
public List<Person> meetCriteria(List<Person> persons) {
List<Person> firstCriteriaPersons = criteria.meetCriteria(persons);
return otherCriteria.meetCriteria(firstCriteriaPersons);
}
}
```

### OrCriteria.java

```
import java.util.List;
public class OrCriteria implements Criteria {
private Criteria criteria;
private Criteria otherCriteria;
public OrCriteria(Criteria criteria, Criteria otherCriteria) {
this.criteria = criteria;
this.otherCriteria = otherCriteria;
}
@Override
public List<Person> meetCriteria(List<Person> persons) {
List<Person> firstCriteriaItems = criteria.meetCriteria(persons);
List<Person> otherCriteriaItems = otherCriteria.meetCriteria(persons);
for (Person person : otherCriteriaItems) {
if(!firstCriteriaItems.contains(person)){
```

```
firstCriteriaItems.add(person);
}
}
return firstCriteriaItems;
}
}
```

## 步骤4

使用不同的标准（Criteria）和它们的结合来过滤 *Person* 对象的列表。

**CriteriaPatternDemo.java**

```
import java.util.ArrayList;
import java.util.List;
public class CriteriaPatternDemo {
public static void main(String[] args) {
List<Person> persons = new ArrayList<Person>();
persons.add(new Person("Robert","Male", "Single"));
persons.add(new Person("John","Male", "Married"));
persons.add(new Person("Laura","Female", "Married"));
persons.add(new Person("Diana","Female", "Single"));
persons.add(new Person("Mike","Male", "Single"));
persons.add(new Person("Bobby","Male", "Single"));
Criteria male = new CriteriaMale();
Criteria female = new CriteriaFemale();
Criteria single = new CriteriaSingle();
Criteria singleMale = new AndCriteria(single, male);
Criteria singleOrFemale = new OrCriteria(single, female);
System.out.println("Males: ");
printPersons(male.meetCriteria(persons));
System.out.println("\nFemales: ");
printPersons(female.meetCriteria(persons));
System.out.println("\nSingle Males: ");
printPersons(singleMale.meetCriteria(persons));
System.out.println("\nSingle Or Females: ");
printPersons(singleOrFemale.meetCriteria(persons));
}
public static void printPersons(List<Person> persons){
for (Person person : persons) {
System.out.println("Person : [ Name : " + person.getName()
+", Gender : " + person.getGender()
+", Marital Status : " + person.getMaritalStatus()
+" ]");
}
}
}
```

## 步骤 5

执行程序，输出结果：

```
Males:
Person : [ Name : Robert, Gender : Male, Marital Status : Single ]
```

```
Person : [ Name : John, Gender : Male, Marital Status : Married ]
Person : [ Name : Mike, Gender : Male, Marital Status : Single ]
Person : [ Name : Bobby, Gender : Male, Marital Status : Single ]

Females:
Person : [ Name : Laura, Gender : Female, Marital Status : Married ]
Person : [ Name : Diana, Gender : Female, Marital Status : Single ]

Single Males:
Person : [ Name : Robert, Gender : Male, Marital Status : Single ]
Person : [ Name : Mike, Gender : Male, Marital Status : Single ]
Person : [ Name : Bobby, Gender : Male, Marital Status : Single ]

Single Or Females:
Person : [ Name : Robert, Gender : Male, Marital Status : Single ]
Person : [ Name : Diana, Gender : Female, Marital Status : Single ]
Person : [ Name : Mike, Gender : Male, Marital Status : Single ]
Person : [ Name : Bobby, Gender : Male, Marital Status : Single ]
Person : [ Name : Laura, Gender : Female, Marital Status : Married ]
```

← 桥接模式 组合模式 →

## 1 篇笔记

写笔记

过滤模式的实现在java8里面有典型的应用方法就是分组操作，可以根据指定的指标进行分组筛选。

```
Map<Integer, List<Person >> groupMap = persons.stream().collect(Collectors.groupingBy(Per
son::getGender));
groupMap.forEach((k, v) -> {
    System.out.println(k);
    v.forEach(System.out::println);
});
```

得到的结果形式就是：

- **k**：是分组的指标，上面代码中的 gender
- **v**：是一个list的集合对象，就是 personList

dream_on_sakura_rain 7个月前 (09-01)