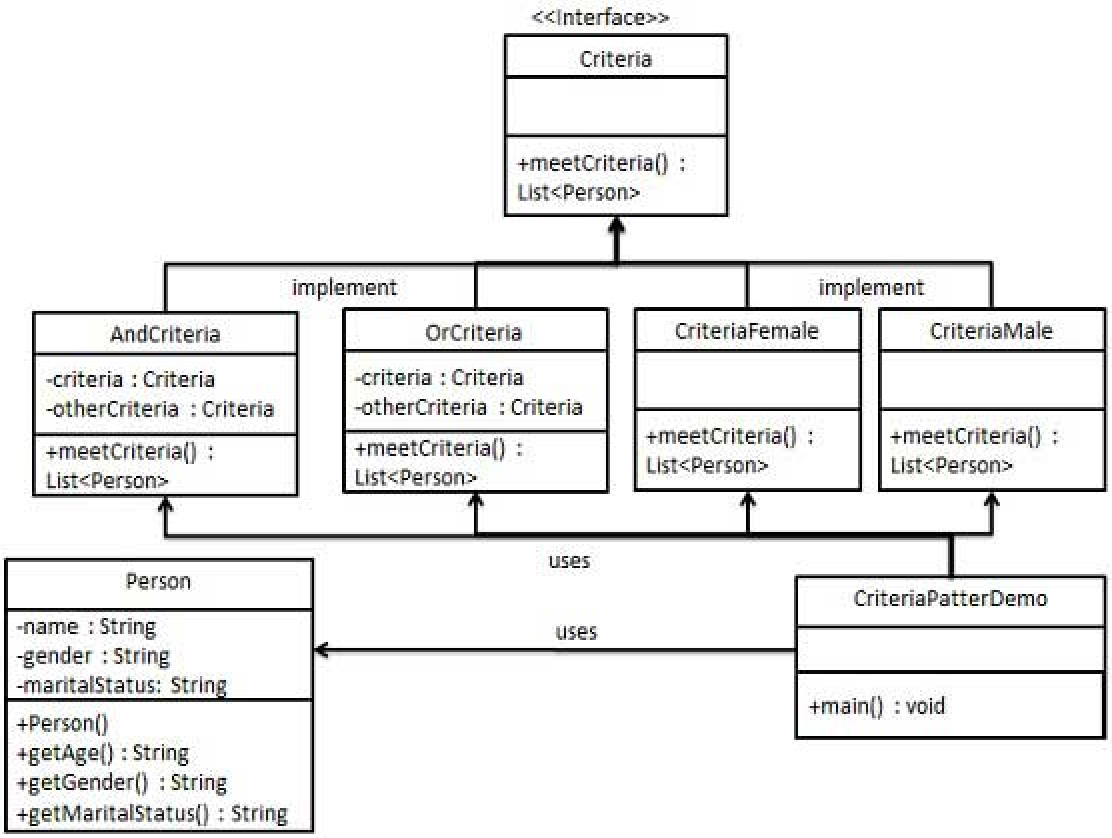
Design Patterns - Filter Pattern

Filter pattern or Criteria pattern is a design pattern that enables developers to filter a set of objects using different criteria and chaining them in a decoupled way through logical operations. This type of design pattern comes under structural pattern as this pattern combines multiple criteria to obtain single criteria.

Implementation

We're going to create a *Person* object, *Criteria* interface and concrete classes implementing this interface to filter list of *Person* objects. *CriteriaPatternDemo*, our demo class uses *Criteria* objects to filter List of *Person* objects based on various criteria and their combinations.



Step 1

Create a class on which criteria is to be applied.

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;
   }
}
```

Step 2

Create an interface for Criteria.

Criteria.java

```
import java.util.List;

public interface Criteria {
   public List<Person> meetCriteria(List<Person> persons);
}
```

Step 3

Create concrete classes implementing the *Criteria* interface.

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;
   }
}
```

Step4

Use different Criteria and their combination to filter out persons.

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() + ", Gen
      }
   }
}
```

Step 5

Verify the output.

```
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 ]
```