Immutable Objects

Докладчик : Андрей

Определение

- Неизменяемый класс – это класс, состояние которого не может быть изменено после создания. Здесь состоянием объекта по существу считаются значения, хранимые в экземпляре класса, будь то примитивные типы или ссылочные типы.

Как создать immutable object

1) Не предоставляйте сеттеры или методы, которые изменяют поля или объекты, ссылающиеся на поля.

   - сеттеры, которые изменяют поля объекта должны возвращать новый объект

2) Сделайте все поля final и private.

3) Не разрешайте подклассам переопределять методы. Самый простой способ это сделать – объявить класс как final. Более сложный подход – это сделать private constructor.

4) Если поля объекта(immutable) содержат ссылки на mutable objects, не разрешайте изменять эти объекты:

   - не предоставляйте методы, что изменяют mutable objects;

   - в конструкторе immutable объекта не используйте прямые ссылки на передаваемые mutable объекты. Нужно сделать копии передаваемых объектов и сохранить ссылки на них.

   - обязательно всегда возвращайте клонированную копию поля и никогда не возвращайте экземпляр реального объекта.

Примеры

- String
- Wrappers: Integer, Byte, Double, Boolean…

```
public final class ImmutableStudent {

    private final String name;
    private final Age age;

    public ImmutableStudent(String name, Age age) {
        this.name = name;
        this.age = age;
    }

    public String getName() { return name; }
    public Age getAge() { return age; }
    public void setName(String name)
            { this.name = name;}
    public void setAge(Age age) {this.age = age; }
}
```

```
    public class Age {
        private int year;
        public int getYear() { return year; }
        public void setYear(int year) { this.year = year; }
    }
```

```
public static void main(String s[]) {
ImmutableStudent immutableStudent =
                        new ImmutableStudent("Ivan", new Age());
immutableStudent.setName("Dima");
System.out.println(immutableStudent.getName());}
```

1) Ivan
2) Dima
3) Ошибка компиляции
4) Ошибка во время выполнения

```
public final class ImmutableStudent {

    private final String name;
    private final Age age;

    public ImmutableStudent(String name, Age age) {
        this.name = name;
        this.age = age;
    }

    public String getName() { return name; }
    public Age getAge() { return age; }
    public void setName(String name)
            { this.name = name;}
    public void setAge(Age age) {this.age = age; }
}
```

```
    public class Age {
        private int year;
        public int getYear() { return year; }
        public void setYear(int year) { this.year = year; }
    }
```

```
  public static void main(String s[]) {
ImmutableStudent immutableStudent =
                        new ImmutableStudent("Ivan", new Age());
immutableStudent.setName("Dima");
System.out.println(immutableStudent.getName());}
```

```
public final class ImmutableStudent {

    private final String name;
    private final Age age;

    public ImmutableStudent(String name, Age age) {
        this.name = name;
        this.age = age;
    }

    public String getName() { return name; }
    public Age getAge() { return age; }
    public ImmutableStudent setName(String name)
      { return new ImmutableStudent(name, this.age); }
   public ImmutableStudent setAge(Age age)
      { return new ImmutableStudent(this.name, age); }
}
```

```
    public class Age {
        private int year;
        public int getYear() { return year; }
        public void setYear(int year) { this.year = year; }
    }
```

```
  public static void main(String s[]) {
ImmutableStudent immutableStudent =
                              new ImmutableStudent("Ivan", new Age());
immutableStudent = immutableStudent.setName("Dima");
System.out.println(immutableStudent.getName());
```

```
public final class ImmutableStudent {

    private final String name;
    private final Age age;

    public ImmutableStudent(String name, Age age) {
        this.name = name;
        this.age = age;
    }

    public String getName() { return name; }
    public Age getAge() { return age; }
}
```

```
public class Age {
    private int year;
    public int getYear() { return year; }
    public void setYear(int year) { this.year = year; }}
```

```
public static void main(String s[]) {

    Age age = new Age();

    age.setYear(1992);

    ImmutableStudent student = new ImmutableStudent("Alex", age);

    System.out.println("Alex year = " + student.getAge().getYear());
    age.setYear(1993);
    System.out.println("Alex year = " + student.getAge().getYear());
  }
}
```

1) Alex year = 1992
Alex year = 1993
2) Alex year = 1992
Alex year = 1992
3) Ошибка компиляции
4) Ошибка во время выполнения

```
"C:\Program Files\Java\jdk-10.0.2\bin\java.exe"
Alex year = 1992
Alex year = 1993

Process finished with exit code 0
```

```
public ImmutableStudent(String name, Age age) {
    this.name = name;
    Age cloneAge = new Age();
    cloneAge.setYear(age.getYear());
    this.age = cloneAge;
}
```

```
public final class ImmutableStudent {

    private final String name;
    private final Age age;

    public ImmutableStudent(String name, Age age) {
        this.name = name;
        Age cloneAge = new Age();
        cloneAge.setYear(age.getYear());
        this.age = cloneAge;}

    public String getName() { return name; }
    public Age getAge() { return age; }
}
```

```
public class Age {
    private int year;
    public int getYear() { return year; }
    public void setYear(int year) { this.year = year; }}
```

```
public static void main(String s[]) {

        Age age = new Age();

        age.setYear(1992);

        ImmutableStudent student = new ImmutableStudent("Alex", age);

        System.out.println("Alex year = " + student.getAge().getYear());
        age.setYear(1993);
        System.out.println("Alex year = " + student.getAge().getYear());
    }
}
```

```
"C:\Program Files\Java\jdk-10.0.2\bin\java.exe"
Alex year = 1992
Alex year = 1992

Process finished with exit code 0
```

```
public final class ImmutableStudent {

    private final String name;
    private final Age age;

    public ImmutableStudent(String name, Age age) {
        this.name = name;
        Age cloneAge = new Age();
        cloneAge.setYear(age.getYear());
        this.age = cloneAge;}

    public String getName() { return name; }
    public Age getAge() { return age; }
}
```

```
public class Age {
    private int year;
    public int getYear() { return year; }
    public void setYear(int year) { this.year = year; }}
```

```
public static void main(String s[]) {

        Age age = new Age();
        age.setYear(1992);

        ImmutableStudent student = new ImmutableStudent("Alex", age);

        System.out.println("Alex year  = " + student.getAge().getYear());
        student.getAge().setYear(1993);
        System.out.println("Alex year  = " + student.getAge().getYear());
    }
}
```

1) Alex year = 1992
Alex year = 1993
2) Alex year = 1992
Alex year = 1992
3) Ошибка компиляции
4) Ошибка во время выполнения

```
"C:\Program Files\Java\jdk-10.0.2\bin\java.exe"
Alex year  = 1992
Alex year  = 1993

Process finished with exit code 0
```

```
public Age getAge() {
    Age cloneAge = new Age();
    cloneAge.setYear(this.age.getYear());
    return cloneAge;
}
```

```
public final class ImmutableStudent {

    private final String name;
    private final Age age;

    public ImmutableStudent(String name, Age age) {
        this.name = name;
        Age cloneAge = new Age();
        cloneAge.setYear(age.getYear());
        this.age = cloneAge;}

    public String getName() { return name; }
    public Age getAge() {
        Age cloneAge = new Age();
        cloneAge.setYear(this.age.getYear());
        return cloneAge;}
}
```

```
public class Main {

    public static void main(String s[]) {

        Age age = new Age();
        age.setYear(1992);

        ImmutableStudent student = new ImmutableStudent("Alex", age);

        System.out.println("Alex year  = " + student.getAge().getYear());
        student.getAge().setYear(1993);
        System.out.println("Alex year  = " + student.getAge().getYear());
    }
}
```

```
"C:\Program Files\Java\jdk-10.0.2\bin\java.exe"
Alex year  = 1992
Alex year  = 1992

Process finished with exit code 0
```

```
public final class ImmutableStudent {
    private final String name;
    private final Age age;

    public ImmutableStudent(String name, Age age) {
        this.name = name;
        Age cloneAge = new Age();
        cloneAge.setYear(age.getYear());
        this.age = cloneAge;}

    public String getName() { return name; }
    public Age getAge() {
        Age cloneAge = new Age();
        cloneAge.setYear(this.age.getYear());
        return cloneAge;}
public ImmutableStudent setName(String name)
       { return new ImmutableStudent(name, this.age); }
   public ImmutableStudent setAge(Age age)
       { return new ImmutableStudent(this.name, age); }}
```

```
public class Age {
    private int year;
    public int getYear() { return year; }
    public void setYear(int year) { this.year = year; }}
```

Вывод

- Неизменяемые классы предоставляют преимущества при правильном использовании в многопоточной среде. Единственным недостатком является то, что они потребляют больше памяти, чем традиционный класс, так как при каждой их модификации в памяти создается новый объект ... но разработчик не должен переоценивать потребление памяти, поскольку оно незначительно по сравнению с преимуществами, предоставляемыми этими объектами.
- Объект является неизменным, если он может представить только одно состояние другим объектам независимо от того, как и когда они вызывают его методы. Если это так, то это потокобезопасно.

Ссылки

- https://docs.oracle.com/javase/tutorial/essential/concurrency/imstrat.html
- https://dzone.com/articles/how-to-create-an-immutable-class-in-java
- https://javarush.ru/groups/posts/765-java-core-voprosih-k-sobesedovaniju-ch-1