Effective_Java Study

Item 34 by 알파

목차

Effective_Java Study Item 34

Def : An *enum type* is a special data type that enables for a variable to be a set of predefined constants.

EX. 월요일, 화요일 등 요일, 동서남북 등 방향

My Def : 미리 정의된 상수들을 모아놓은 객체!

코드로는 어떻게?

Effective_Java Study Item 34

```
public enum Days {
    MONDAY, TUESDAY, WEDNESDAY,
    THURSDAY, FRIDAY, SATURDAY,
    SUNDAY;
}
```

생성자, 인스턴스 변수, 클래스 변수

```
public enum Days {
    MONDAY, TUESDAY, WEDNESDAY,
    THURSDAY, FRIDAY, SATURDAY,
    SUNDAY;

    public Days() {};
}
```

?

Effective_Java Study Item 34

```java
public enum Days {
    MONDAY, TUESDAY, WEDNESDAY,
    THURSDAY, FRIDAY, SATURDAY,
    SUNDAY;

    private Days() {};
}
```

!

Predefined가 아니기 때문!

Enum으로
할 수 있는 것

Effective_Java Study Item 34

```java
public enum Days {
    MONDAY( order: 1), TUESDAY( order: 2), WEDNESDAY( order: 3),
    THURSDAY( order: 4), FRIDAY( order: 5), SATURDAY( order: 6),
    SUNDAY( order: 7);

    private static final String NAME = "WEEK";
    private final int order;

    Days(int order) {
        this.order = order;
    }
}
```

인스턴스 메소드, 클래스 메소드

Enum으로
할 수 있는 것

Effective_Java Study Item 34

```java
public enum Days {
    MONDAY( order: 1), TUESDAY( order: 2), WEDNESDAY( order: 3),
    THURSDAY( order: 4), FRIDAY( order: 5), SATURDAY( order: 6),
    SUNDAY( order: 7);

    private static final String NAME = "WEEK";
    private final int order;

    Days(int order) {
        this.order = order;
    }

    public int getOrder() {
        return this.order;
    }

    public static String getNAME() {
        return NAME;
    }
}
```

내장된 메소드..?

Effective_Java Study Item 34

```java
public class Main {

    public static void main(String[] args) {
        Days[] days = Days.values();
    }
}
```

Effective_Java Study Item 34

```java
public class Main {

    public static void main(String[] args) {
        System.out.println(Days.valueOf("MONDAY"));
    }
}
```

```
Main
"C:\Program Files (x86)\Java\jdk1.8.0_311\bin\java.exe" ...
MONDAY

Process finished with exit code 0
```

Enum으로
할 수 있는 것

Effective_Java Study Item 34

```java
public class Main {

    public static void main(String[] args) {
        System.out.println(Days.MONDAY.ordinal());
    }
}
```

```
Main ×
"C:\Program Files (x86)\Java\jdk1.8.0_311\bin\java.exe" ...
0

Process finished with exit code 0
```

Effective_Java Study Item 34

```java
public class Main {

    public static void main(String[] args) {
        System.out.println(Days.MONDAY.compareTo(Days.FRIDAY));
        System.out.println(Days.FRIDAY.compareTo(Days.MONDAY));
        System.out.println(Days.TUESDAY.compareTo(Days.TUESDAY));
    }
}
```

```
Main
"C:\Program Files (x86)\Java\jdk1.8.0_311\bin\java.exe" ...
-4
4
0
```

그런데 정적 팩토리 메소드를 곁들인

Enum으로
할 수 있는 것

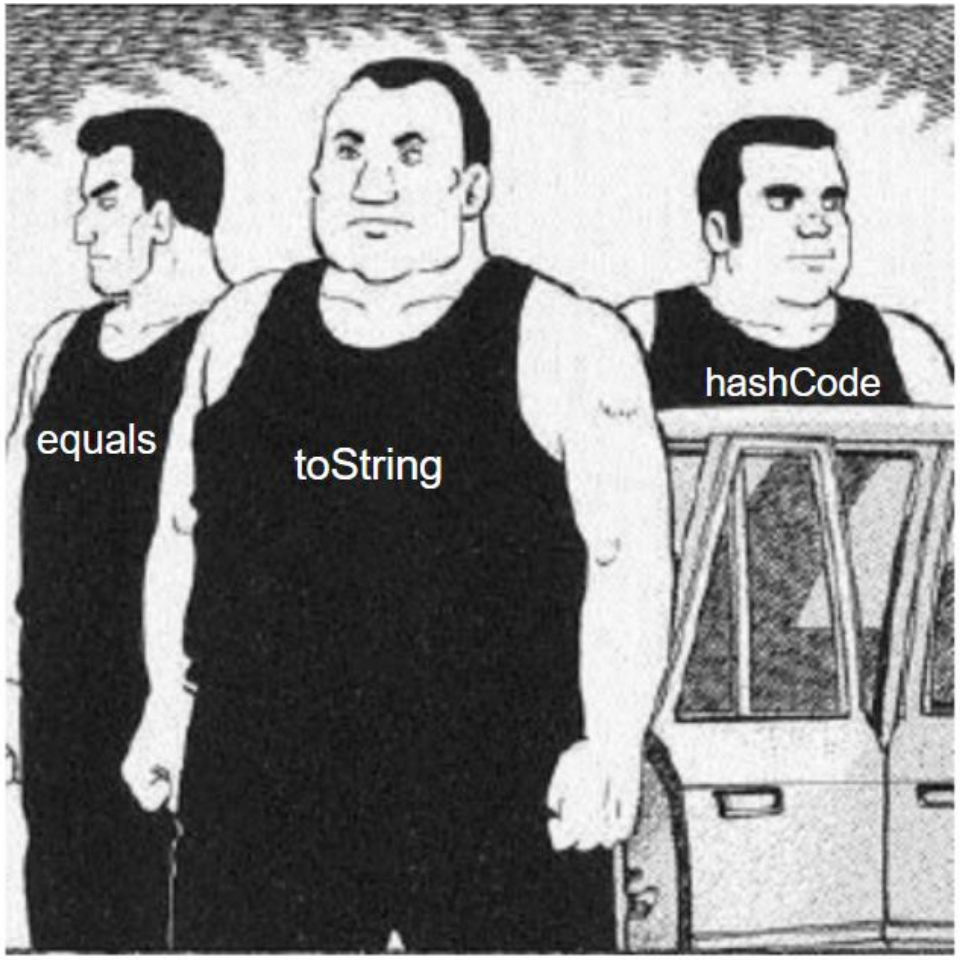
Effective_Java Study Item 34

```java
public enum Days {
    MONDAY( order: 1), TUESDAY( order: 2), WEDNESDAY( order: 3),
    THURSDAY( order: 4), FRIDAY( order: 5), SATURDAY( order: 6),
    SUNDAY( order: 7);

    private static final String NAME = "WEEK";
    private final int order;

    Days(int order) {
        this.order = order;
    }

    public int getOrder() {
        return this.order;
    }

    public static String getNAME() {
        return NAME;
    }

    public static Days from(int order) {
        return Arrays.stream(Days.values()).filter(eachDay -> eachDay.order == order).findAny().get();
    }
}
```

3형제

Enum으로
할 수 있는 것

Effective_Java Study Item 34



equals, hashCode는 Override 할 수 없다,

그러나 toString은 가능하다!

그런데 두 상수간 비교는 어떻게?

Effective_Java Study Item 34

```java
public class Main {

    public static void main(String[] args) {
        System.out.println(Days.from(3) == Days.WEDNESDAY);
    }
}
```

```
Main ×
"C:\Program Files (x86)\Java\jdk1.8.0_311\bin\java.exe" ...
true

Process finished with exit code 0
```

Pigeonhole Principle

ETC…

Effective_Java Study Item 34

```
public class MyClass {

    //...

    enum MyEnum {

        //...
    }
}
```

Enum으로
할 수 있는 것

Effective_Java Study Item 34

```
public class MyClass {

    //...

}

enum MyEnum {

    //...

}
```

Enum으로
할 수 있는 것

Effective_Java Study Item 34

```
public enum MyEnum {
    ONE, TWO;

    enum InsideMyEnum {
        THREE, FOUR
    }
}
```

그냥 상수 클래스 하나 박아넣으면 안돼?

Q. 상수는 타입이 안전한가?

Enum을
써야하는 이유

Effective_Java Study Item 34

```java
public class MyClass {
    private static final int MALE_ALPHA = 1;
    private static final int FEMALE_BETA = 2;
    private static final int MALE_KID = 3;
    private static final int FEMALE_KID = 4;

    private static void MALE_CONDITION(int maleConst) {
        if (maleConst == 1) {
            System.out.println("2000년 이전 출생 남성입니다.");
        }
        else {
            System.out.println("2000년 이후 출생 남성입니다.");
        }
    }

    private static void someMethod() {
        System.out.println(MALE_ALPHA == FEMALE_BETA);
    }
}
```

Enum을
써야하는 이유

Effective_Java Study Item 34

```java
public class MyClass {

    private static void MALE_CONDITION(MaleEnum maleConst) {
        if (maleConst.getNumber() == 1) {
            System.out.println("2000년 이전 출생 남성입니다.");
        }
        else {
            System.out.println("2000년 이후 출생 남성입니다.");
        }
    }
}
```

```java
public enum MaleEnum {

    BEFORE_2000_MALE( number: 1),
    AFTER_2000_MALE( number: 3);

    private final int number;

    MaleEnum(int number) { this.number = number; }

    public int getNumber() { return this.number; }
}
```

A. Enum으로 빼준다면 관련없는 값을 넣을 수 없다.
즉, 타입 안전을 보장할 수 있다.

Q. 클래스가 내부의 상수만 관련있는 로직을 관리하는 게 클래스의 책임 원칙에 부합하는가?

Enum을
써야하는 이유

Effective_Java Study Item 34

```java
public enum Planet {

    MERCURY( mass: 3.302e+23,  radius: 2.439e6),
    VENUS( mass: 4.869e+24,  radius: 6.052e6),
    EARTH( mass: 5.975e+24,  radius: 6.378e6),
    MARS( mass: 6.419e+23,  radius: 3.393e6);

    private static final double G = 6.67300E-11;
    private final double mass;
    private final double radius;
    private final double surfaceGravity;

    Planet(double mass, double radius) {
        this.mass = mass;
        this.radius = radius;
        this.surfaceGravity = G * this.mass / (this.radius * this.radius);
    }

    public double surfaceWeight(double mass) {
        return this.mass * this.surfaceGravity;
    }
}
```

A. Enum을 사용하면 관련 있는 상수, 필드, 메소드를 Enum에서만 관리할 수 있다

Q. 상수끼리 다른 로직을 처리해야 할 때 이것이 가능한가?

Enum을
써야하는 이유

Effective_Java Study Item 34

```java
import static com.company.PayRollDay.PayType.WEEKDAY;
import static com.company.PayRollDay.PayType.WEEKEND;

public enum PayRollDay {
    MONDAY(WEEKDAY), TUESDAY(WEEKDAY), WEDNESDAY(WEEKDAY),
    THURSDAY(WEEKDAY), FRIDAY(WEEKDAY), SATURDAY(WEEKEND),
    SUNDAY(WEEKEND);

    private final PayType payType;

    PayRollDay(PayType payType) {
        this.payType = payType;
    }

    int pay(int minutesWorked, int payRate) {
        return this.payType.pay(minutesWorked, payRate);
    }

    enum PayType {
        WEEKDAY {
            int overtimePay(int minutesWorked, int payRate) {
                if (minutesWorked <= MINS_PER_SHIFT) {
                    return 0;
                }
                return (minutesWorked - MINS_PER_SHIFT) * payRate / 2;
            }
        },
        WEEKEND {
            int overtimePay(int minutesWorked, int payRate) {
                return minutesWorked * payRate / 2;
            }
        };

        private static final int MINS_PER_SHIFT = 480;
        abstract int overtimePay(int minutesWorked, int payRate);

        public int pay(int minutesWorked, int payRate) {
            int basePay = minutesWorked * payRate;
            return basePay + overtimePay(minutesWorked, payRate);
        }

    }
}
```

A. Enum을 사용하면 상수간 다른 동작을 효율적으로 수행할 수 있다

그러나....

상수들 간의 관계가 Enum을 이룰만큼 큰 연관이 없고, 단순한 값으로만 사용된다면 굳이 Enum으로 뺄 이유는 없지 않을까?

E.O.D

Item 34 by 알파