Enum

Перечисление - это список именованных констант.

Но в Java перечисления имеют более сложный функционал, чем в других языках программирования. Они могут иметь конструкторы, методы и переменные экземпляра.

Перечисления создаются с использованием ключевого слова enum. Создадим перечисление семейства кошачьих:

```
enum Cat {
    Leopard, Puma, Lion, Tiger, Manul
}
```

Идентификаторы в фигурных скобках называются константами перечисления.

Каждый из них явно объявлен как открытый статический финальный член класса Cat. Объявив перечисление, вы можете создавать переменные этого типа. Но делать это нужно без оператора new, а в упрощенном виде. Объявим переменную manul перечислимого типа Cat.

```
Cat manul = Cat.Manul;
```

Перечислимые константы можно проверить на равенство:

```
if (manul == Cat.Manul) {
   String result = "Погладь кота, ...";
}
```

Также их можно применять в конструкции switch, где у операторов case используются константы из перечисления enum. При этом имена констант используются без имени типа перечисления.

```
enum Cat {
        Leopard, Puma, Lion, Tiger, Manul
}

public void onClick() {
        Cat cat;
        cat = Cat.Manul;
        String result = "";

        switch (cat) {
        case Leopard:
                result = "Я леопард";
                break;
        case Puma:
                result = "Я пума";
                break;
                }
        System.out.printlm(result);
}
```

Метод values()

Автоматически предопределённый метод для перечисления values() возвращает массив, содержащий список констант перечисления.

```
enum Cat {
        Leopard, Puma, Lion, Tiger, Manul
}

public void onClick() {
        Cat[] allcats = Cat.values();

        for(Cat cat : allcats) {
            System.out.println(cat);
        }
}
```

Для примера использовалась дополнительная переменная allcats, которой присваивается ссылка на массив перечислимых значений. Можно обойтись без дополнительной переменной.

```
for(Cat cat : Cat.values()) {
   System.out.println(cat);
}
```

Метод valueOf(String string)

Автоматически предопределённый метод для перечисления valueOf() возвращает константу перечисления, значение которой соответствует строке, переданной в параметре

```
enum Cat {
        Leopard, Puma, Lion, Tiger, Manul
}

public void onClick(View v) {
        Cat cat;
        cat = Cat.valueOf("Puma");

        textViewInfo.setText(cat.toString());
}
```

Так как перечисление в Java - это тип класса, то вы можете использовать конструкторы, добавлять переменных экземпляров, методы и интерфейсы. Следует сказать, что в Android сначала не рекомендовалось использовать перечисления из-за большого потребления памяти. Сейчас вроде это ограничение сняли, но тем не менее пока перечисления не так широко используются в приложениях для мобильных устройств, поэтому подробно разбирать все возможности перечисления не будем.

Добавление методов

У Вас есть возможность добавлять собственные методы как в enum-класс, так и в его элементы:

```
enum Direction {
   UP, DOWN;

   public Direction opposite() { return this == UP ? DOWN : UP; }
}
```

То же, но с полиморфизмом:

```
enum Direction {
   UP {
       public Direction opposite() { return DOWN; }
   },
   DOWN {
       public Direction opposite() { return UP; }
   };

   public abstract Direction opposite();
}
```

Наследование

С помощью enum в Java можно реализовать иерархию классов, объекты которой создаются в единственном экземпляре и доступны статически. При этом элементы enum могут содержать собственные конструкторы.

```
enum Type {
    INT(true) {
        public Object parse(String string) { return Integer.valueOf(string); }
    },
    INTEGER(false) {
        public Object parse(String string) { return Integer.valueOf(string); }
    },
    STRING(false) {
        public Object parse(String string) { return string; }
    };

    boolean primitive;
    Type(boolean primitive) { this.primitive = primitive; }

    public boolean isPrimitive() { return primitive; }
    public abstract Object parse(String string);
}
```

Метод ordinal()

У перечислений есть несколько удобных методов. Например, вы можете получить значение, которое указывает позицию константы в списке констант перечисления (порядковое значение или ordinal value), с помощью метода ordinal(). Порядковые значения начинаются с нуля.

```
for(Cat cat : Cat.values()) {
        System.out.println(cat + " Его порядковое значение" + cat.ordinal());
}
```

Задание

Создать тип перечисления планет солнечной системы. Каждая планета характеризуется массой и радиусом. И для каждой планеты возможно рассчитать силу притяжения на поверхности (G*mass/R^2, где G - гравитационная постоянная).