



Пример 1: Программа "Lamp"

- Цель примера

Продемонстрировать создание и использование простого класса в Java. В качестве примера будет рассмотрен класс – лампа `Lamp`.

- Поэтапное создание программы

Язык Java относится к объектно-ориентированным, что позволяет решать самые сложные задачи путем их деления на более маленькие – при помощи создания классов и описания взаимодействий между ними.

Все объекты имеют две характеристики:

- Состояние (state) – то, каким качествами обладает объект;
- Поведение (behaviour) – то, какие действия умеет выполнять объект;

!! Примечание: Несмотря на характеристики, в программировании объекты не всегда имеют состояние и поведение одновременно. Часто встречаются объекты, у которых есть только поведение, без состояния (stateless).

i Информация: Поведение объекта также называют методами (method) или функциями (function).

Прежде чем создавать конкретный объект, нужно определить класс, поскольку класс является чертежом для объекта.

Класс – это элемент, который описывает общие свойства и поведение, которые характерны чему-то. Например, к свойствам машины можно отнести марку, тип кузова и цвет.

Объект – это конкретный экземпляр (instance) класса с определенными значениями этих свойств. Например, синий седан Audi.

Очевидно, что на основе одного класса можно создать сколько угодно объектов с разными свойствами.

!! Примечание: Поведение, которое описано в классе, будет работать во всех объектах одинаково. Исключение: если в действии используется состояние самого объекта.

– Шаг 1.

Важно определиться с тем, какими свойствами и действиями будет обладать класс `Lamp`. В самом простом виде, лампа может выглядеть следующим образом:

- Свойства:
 - Лампа может быть включена или выключена;
- Методы:
 - Лампу можно включить или выключить;

Обратите внимание, что методы будут влиять на состояние.

- Шаг 2.

Создадим класс `Lamp` :

```
public class Lamp {  
  
}
```


Метод `main()` в данном случае не нужен, поскольку этот класс не будет входной точкой в программу.


- Шаг 3.

Для описания состояния лампы (включена или выключена) удобно использовать примитивный логический тип `boolean` который отвечает на вопрос `isOn` . Если значение `true` , то лампа включена, иначе `false` – выключена.

```
public class Lamp {  
  
    public boolean isOn;  
  
}
```

Добавив всего одну строку мы объявили свойство у лампы.

 **Важно:** Ключевые слова `private` и `public` будут рассматриваться более подробно в дальнейшем. На данном этапе это не принципиально.

 **Информация:** Свойства класса также называют его полями (instance variable).

- Шаг 4.

Для описания поведения лампы (включить и выключить) подходит использование методов, которые изменяют состояние объекта, но не возвращают результат:

```
public class Lamp {  
  
    public boolean isOn;  
  
    public void turnOn() {  
        this.isOn = true;  
    }  
  
    public void turnOff() {  
        this.isOn = false;  
    }  
  
}
```

Внутри метода переменной присваивается значение переменной в соответствии с логикой, которая описана выше.

Обратите внимание на ключевое слово `this` при обращении к переменной: оно позволяет уточнить, что эта переменная является свойством этого класса. В данном случае оно избыточно и его можно не использовать, но оно важно в том случае, если имена свойств и параметров метода конфликтуют.

Ключевое слово `void` в сигнатуре метода означает, что результата от работы метода не ожидается.

i **Информация:** Сигнатурой (заголовком) метода называют комбинацию из его имени и параметров.

- Шаг 5.

Для работы с классом `Lamp`, а именно созданием объектов, необходимо создать класс `LampDemo` для демонстрационных целей с методом `main()`:

```
public class LampDemo {  
  
    public static void main(String[] args) {  
  
    }  
  
}
```

- Шаг 6.

Любой класс можно рассматривать как произвольный ссылочный тип данных. Таким образом, любой объект – это переменная типа вашего класса, в данном случае типа лампы. Для создания нового объекта используется специальное ключевое слово `new`:

```
public class LampDemo {  
  
    public static void main(String[] args) {  
  
        Lamp lampOne = new Lamp();  
        Lamp lampTwo = new Lamp();  
  
    }  
  
}
```

Мы создали две переменные `lampOne` и `lampTwo` типа `Lamp`.

- Шаг 7.

Для того, чтобы обратиться к конкретному состоянию конкретного объекта, используют оператор `.`. Например, чтобы узнать, включены ли лампы `lampOne` и `lampTwo`, можно вывести свойство `isOn` в консоль:

```

public class LampDemo {

    public static void main(String[] args) {

        Lamp lampOne = new Lamp();
        Lamp lampTwo = new Lamp();

        System.out.println("lampOne.isOn = " + lampOne.isOn);
        System.out.println("lampTwo.isOn = " + lampTwo.isOn);

    }

}

```

Результатом работы программы будет:

```

lampOne.isOn = false lampTwo.isOn = false

```

Как видно, обе лампы по-умолчанию выключены, потому что неопределенное значение примитивного типа `boolean` равняется `false`.

- Шаг 8.

Для использования метода конкретной лампы используется тот же оператор `.`. Включим первую лампу `lampOne` и посмотрим на состояние обеих ламп:

```

public class LampDemo {

    public static void main(String[] args) {

        Lamp lampOne = new Lamp();
        Lamp lampTwo = new Lamp();

        System.out.println("lampOne.isOn = " + lampOne.isOn);
        System.out.println("lampTwo.isOn = " + lampTwo.isOn);

        System.out.println("Turning on lampOne!");
        lampOne.turnOn();

        System.out.println("lampOne.isOn = " + lampOne.isOn);
        System.out.println("lampTwo.isOn = " + lampTwo.isOn);

    }

}

```

Результат работы программы:

```

lampOne.isOn = false lampTwo.isOn = false Turning on lampOne! lampOne.isOn = true lampTwo.isOn = false

```

Состояние первой лампы `lampOne` изменилось на `true`, а второй лампы `lampTwo` осталось прежним. Это говорит о том, что объекты изолированы друг от друга и состояние одного объекта не влияет на состояние другого.

• Рекомендации:

- Запустить программу и сравнить результаты;

- Попробовать изменить состояние обеих ламп включая и выключая их и проверить их свойства;
- Создать еще один экземпляр лампы и изучить ее свойства и действия;