**Переменные-ссылки**

[Java Syntax](https://javarush.ru/quests/QUEST_JAVA_SYNTAX)

[Уровень 2](https://javarush.ru/quests/lectures?quest=QUEST_JAVA_SYNTAX&level=2), Лекция 6

— Привет, Амиго, это снова я – Элли. Прости, что постоянно так говорю, так просто принято на Земле в 31 веке. Я хочу рассказать тебе об особенностях переменных-ссылок и передаче в функции(методы) аргументов-ссылок.

— Я готов.

— Отлично, тогда слушай. Переменные-ссылки – это переменные всех типов, кроме примитивных. Такие переменные содержат в себе только адрес объекта (ссылку на объект).

— Переменные примитивных типов хранят в себе значения, а переменные типов-классов хранят ссылку на объекты этого же класса, ну или хранят null. Я все правильно говорю?

— Абсолютно верно.

— Так что же это такое — эта ссылка?

— Объект и ссылка на него связаны, примерно, как человек и его телефонный номер. Телефонный номер не является человеком, но номер можно использовать, чтобы звонить человеку, спрашивать у него какую-то информацию, руководить им или давать команды. Ссылка тоже используется для взаимодействия с объектом. Все объекты взаимодействуют друг с другом при помощи ссылок.

— Это как если бы люди общались друг с другом по телефону?

— Именно. При присваивании примитивного объекта, его значение копируется (дублируется). При присваивании же ссылочной переменной, копируется только адрес объекта (телефонный номер), сам же объект при этом не копируется.

— Ок, понял.

— Ссылка даёт ещё одно преимущество: можно передать ссылку на объект в какой-нибудь метод, и этот метод будет в состоянии модифицировать (изменять) наш объект используя ссылку на него, вызывая его методы и обращаясь к данным внутри объекта.

|  |  |
| --- | --- |
| Пример 1 Тут значение m и n не меняется.  public class References  {  public static void main (String[] args)  {  int m = 5;  int n = 6;  System.out.println("M=" +m+ " N=" + n);  swap(m, n);  System.out.println("M=" +m+ " N=" + n);  }  private static void swap(int a, int b)  {  int c = a;  a = b;  b = c;  }  } | И вот почему. Данный код аналогичен коду слева  public class References  {  public static void main (String[] args)  {  int m = 5;  int n = 6;  System.out.println("M=" +m+ " N=" + n);  int a = m, b = n;  int c = a;  a = b;  b = c;  System.out.println("M=" +m+ " N=" + n);  }  } |

— Переменным a и b присваиваются только значения 5(m) и 6(n) соответственно, a и b ничего не знают про (никак не влияют на) m и n.

— Честно говоря, я понял, что я ничего не понял. Можно еще пару примеров?

— В случае со ссылочным типом, можно было сделать так:

|  |  |
| --- | --- |
| Пример 2 Тут данные объектов меняются  public class Primitives  {  public static void main(String[] args)  {  Student olga = new Student();  olga.name = "Olga";  olga.age = 21;  Student vera = new Student();  vera.name = "Veronika";  vera.age = 15;  System.out.println("Olga is " + olga.age);  System.out.println("Vera is " + vera.age);  ageSwap(olga, vera);  System.out.println("Olga is " + olga.age);  System.out.println("Vera is " + vera.age);  }  private static void ageSwap(Student a,  Student b)  {  int c = a.age;  a.age = b.age;  b.age = c;  }  static class Student  {  String name;  int age;  }  } | И вот почему. Данный код аналогичен коду слева  public class Primitives  {  public static void main(String[] args)  {  Student olga = new Student();  olga.name = "Olga";  olga.age = 21;  Student vera = new Student();  vera.name = "Veronika";  vera.age = 15;  System.out.println("Olga is " + olga.age);  System.out.println("Vera is " + vera.age);  Student a = olga, b = vera;  int c = a.age;  a.age = b.age;  b.age = c;  System.out.println("Olga is " + olga.age);  System.out.println("Vera is " + vera.age);  }  static class Student  {  String name;  int age;  }  } |

— Переменным a и b присваиваются ссылки на olga и vera соответственно, a и b меняют значения внутри объектов olga и vera.

— Так классы внутри других классов ещё можно объявлять? Круто!

— Но в остальном не очень-то и понятно.

— Не все сразу.