

10.1

Создание массивов

00:00–00:19

Введение

В этом видео мы познакомимся с массивами. Они нужны для хранения множества элементов, чисел, строк и объектов. Массив может содержать в себе заранее известное число одинаковых элементов.

00:19–03:14

Массив примитивов

Как создаются массивы?

Пишется тип переменной, дальше ставятся квадратные скобки, после которых пишется название переменной. Например, в массиве будет храниться количество комнат на каждом этаже здания — назовём этот массив `roomCounts`. Здесь мы должны написать `new int` и в квадратных скобках обозначить количество элементов, которые есть в этом массиве.

```
int[] roomCounts = new int[];
```

Например, в здании шесть этажей. Нумерация, так же как и символов в строках, начинается с нуля.

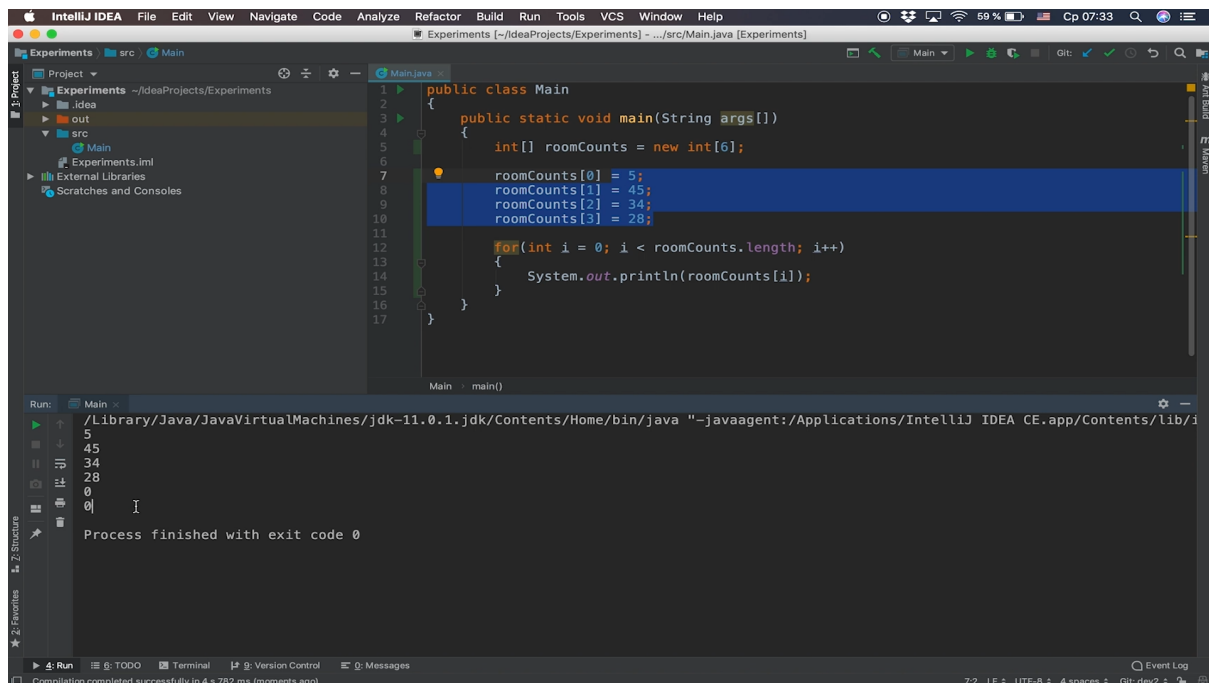
У нас есть массив, и теперь мы можем его заполнять, например, следующим образом (в квадратных скобках обозначен номер этажа, после знака равенства обозначено количество комнат на этаже):

```
room Counts [0] = 5;  
room Counts [1] = 45;  
room Counts [2] = 34;  
room Counts [3] = 28;
```

Дальнейшие элементы, четвёртый и пятый этаж, можно не заполнять, поскольку в данном случае они будут заполнены нулями: если мы создаём массив примитивов, а не объектов, то элементы заполняются автоматически значениями по умолчанию.

Напишем цикл `for`, так же от нуля до длины `length`. У массива есть свойства `length` — это, в данном случае, уже не метод, как было в строках, это именно свойства.

Распечатаем этот массив и посмотрим, что получится. Мы видим, что незаполненные элементы имеют значение 0.



The screenshot shows the IntelliJ IDEA IDE with a project named 'Experiments'. The source file 'Main.java' is open, showing the following code:

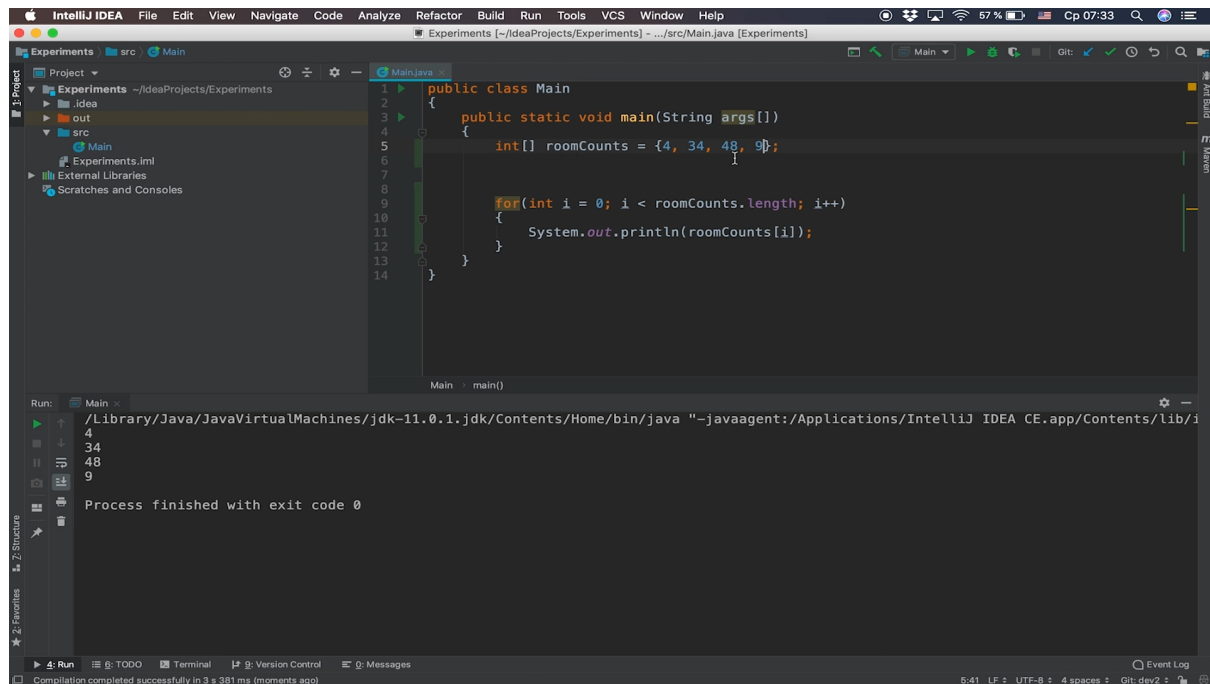
```
1 public class Main
2 {
3     public static void main(String args[])
4     {
5         int[] roomCounts = new int[6];
6
7         roomCounts[0] = 5;
8         roomCounts[1] = 45;
9         roomCounts[2] = 34;
10        roomCounts[3] = 28;
11
12        for(int i = 0; i < roomCounts.length; i++)
13        {
14            System.out.println(roomCounts[i]);
15        }
16    }
17 }
```

The 'Run' tab at the bottom shows the output of the program:

```
5
45
34
28
0
0
Process finished with exit code 0
```

The status bar at the bottom indicates 'Compilation completed successfully in 4 s 782 ms (moments ago)'.

Есть и другой, более удобный способ заполнения массива.



```
public class Main
{
    public static void main(String args[])
    {
        int[] roomCounts = {4, 34, 48, 9};

        for(int i = 0; i < roomCounts.length; i++)
        {
            System.out.println(roomCounts[i]);
        }
    }
}
```

Run: Main

/Library/Java/JavaVirtualMachines/jdk-11.0.1.jdk/Contents/Home/bin/java "-javaagent:/Applications/IntelliJ IDEA CE.app/Contents/lib/i

4
34
48
9

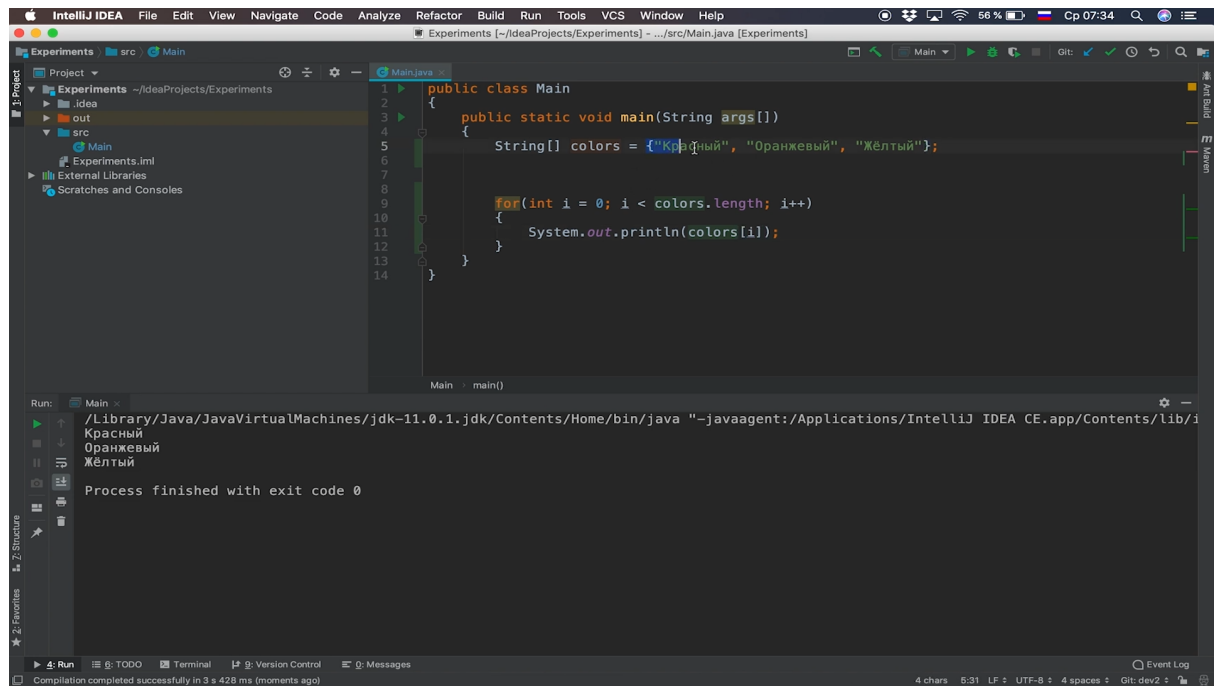
Process finished with exit code 0

В фигурных скобках можно через запятую прописать значения, которыми необходимо заполнить массив. Если не прописать в фигурных скобках нули для последних двух этажей, код инициализируется и в консоль они не будут выведены (см. скрин выше).

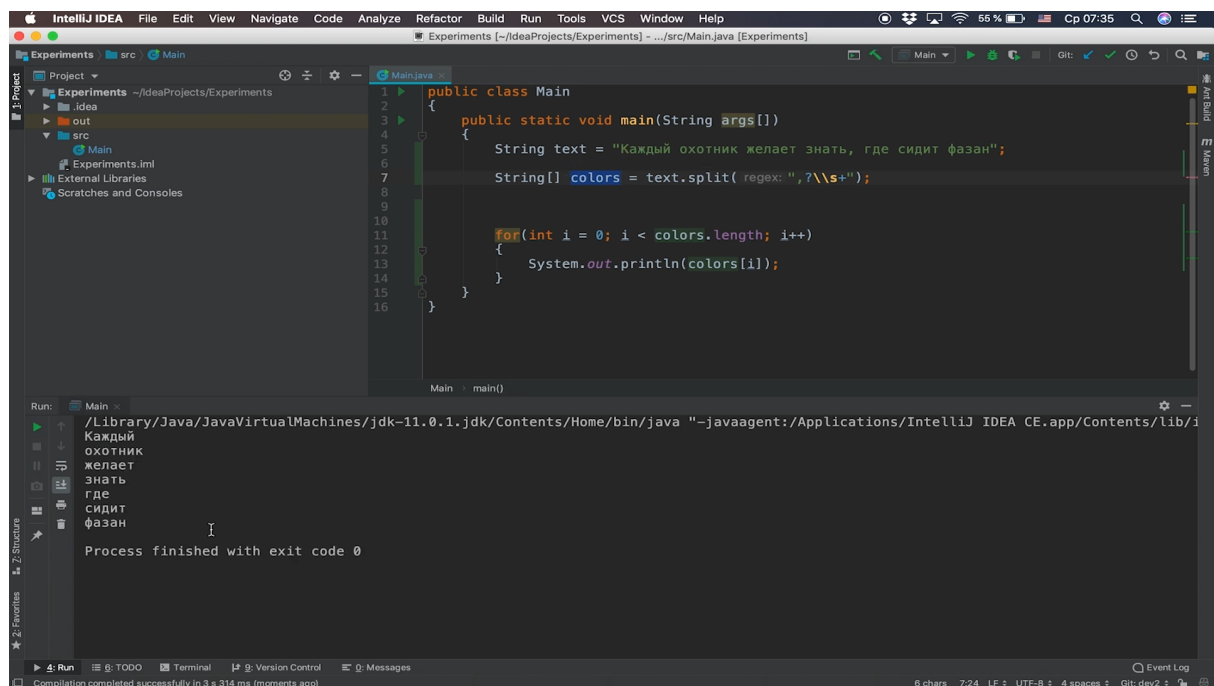
03:14

Массив объектов

Создадим массив объектов. Допустим, это массив строк — пусть это будут цвета радуги. Можно написать его так же, как мы делали с массивами примитивов — вписать в фигурных скобках через запятую значения, только сейчас это будут не числа, а слова в кавычках. И точно так же можно будет напечатать здесь элементы массива:



Массивы можно получать из методов. Например, мы можем написать строку с радугой и разделить строку на отдельные подстроки.



Это тоже массив, и он имеет заданное количество элементов.