链滴

Java 数组

作者：chenyuan995

原文链接：https://ld246.com/article/1587545537973

来源网站：链滴

数组定义和访问

数组的定义

- 格式一：

```
数组存储的数据类型[] 数组名字 = new 数组存储的数据类型[长度];
```

- 数组定义格式详解：
  - 数组存储的数据类型： 创建的数组容器可以存储什么数据类型。
  - [] : 表示数组。
  - 数组名字：为定义的数组起个变量名，满足标识符规范，可以使用名字操作数组。
  - new：关键字，创建数组使用的关键字。
  - 数组存储的数据类型： 创建的数组容器可以存储什么数据类型。
  - [长度]：数组的长度，表示数组容器中可以存储多少个元素。
  - 注意：数组有定长特性，长度一旦指定，不可更改。
    - 和水杯道理相同，买了一个2升的水杯，总容量就是2升，不能多也不能少。

- 举例：

定义可以存储3个整数的数组容器，代码如下：

```
int[] arr = new int[3];
```

- 格式二：

```
数据类型[] 数组名 = new 数据类型[]{元素1,元素2,元素3...};
```

- 举例：

定义存储1，2，3，4，5整数的数组容器。

```
int[] arr = new int[]{1,2,3,4,5};
```

- 格式三：

```
数据类型[] 数组名 = {元素1,元素2,元素3...};
```

- 举例：

定义存储1，2，3，4，5整数的数组容器

```
int[] arr = {1,2,3,4,5};
```

数组的访问

- 索引： 每一个存储到数组的元素，都会自动的拥有一个编号，从0开始，这个自动编号称为数组索引(index)，可以通过数组的索引访问到数组中的元素。
- 格式：

```
数组名[索引]
```

- 数组的长度属性： 每个数组都具有长度，而且是固定的，Java中赋予了数组的一个属性，可以获取数组的 长度，语句为： 数组名.length ，属性length的执行结果是数组的长度，int类型结果。由次以推断出，数 组的最大索引值为 数组名.length-1 。
- 索引访问数组中的元素：
  - 数组名[索引]=数值，为数组中的元素赋值
  - 变量=数组名[索引]，获取出数组中的元素

```
public static void main(String[] args) {
    //定义存储int类型数组，赋值元素1，2，3，4，5
    int[] arr = {1,2,3,4,5};
    //为0索引元素赋值为6
    arr[0] = 6;
    //获取数组0索引上的元素
    int i = arr[0];
    System.out.println(i);
    //直接输出数组0索引元素
    System.out.println(arr[0]);
}
```

数组原理内存图

Java虚拟机的内存划分

- JVM的内存划分：

Java的内存需要划分成为5个部分：

1. 栈（Stack）：存放的都是方法中的局部变量。方法的运行一定要在栈当中运行。
   局部变量：方法的参数，或者是方法{}内部的变量
   作用域：一旦超出作用域，立刻从栈内存当中消失。
2. 堆（Heap）：凡是new出来的东西，都在堆当中。
   堆内存里面的东西都有一个地址值：16进制
   堆内存里面的数据，都有默认值。规则：
   - 如果是整数　默认为0
   - 如果是浮点数　默认为0.0
   - 如果是字符　默认为'\u0000'
   - 如果是布尔　默认为false
   - 如果是引用类型　默认为null
3. 方法区（Method Area）：存储.class相关信息，包含方法的信息。
4. 本地方法栈（Native Method Stack）：与操作系统相关。
5. 寄存器（pc Register）：与CPU相关。

区域名称	作用
寄存器	给CPU使用，和我们开发无关。
本地方法栈	JVM在使用操作系统功能的时候使用，和我们开发无关。
方法区	存储可以运行的class文件。
堆内存	存储对象或者数组，new来创建的，都存在堆内存。
方法栈	方法运行时使用的内存，比如main方法行，进入方法栈中执行。

数组在内存中的存储

- 一个数组内存图

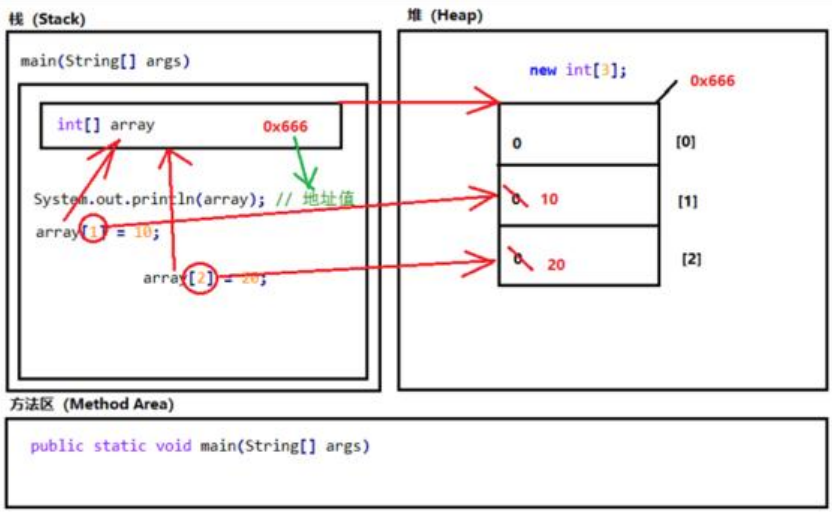
```
public class Demo01ArrayOne {
    public static void main(String[] args) {
        int[] array = new int[3]; // 动态初始化
        System.out.println(array); // 地址值
        System.out.println(array[0]); // 0
        System.out.println(array[1]); // 0
        System.out.println(array[2]); // 0
        System.out.println("==============");
        // 改变数组当中元素的内容
        array[1] = 10;
        array[2] = 20;
        System.out.println(array); // 地址值
        System.out.println(array[0]); // 0
        System.out.println(array[1]); // 10
        System.out.println(array[2]); // 20
    }
}
```

```java
public class Demo01ArrayOne {

    public static void main(String[] args) {
        int[] array = new int[3]; // 动态初始化
        System.out.println(array); // 地址值
        System.out.println(array[0]); // 0
        System.out.println(array[1]); // 0
        System.out.println(array[2]); // 0
        System.out.println("==============");

        // 改变数组当中元素的内容
        array[1] = 10;
        array[2] = 20;
        System.out.println(array); // 地址值
        System.out.println(array[0]); // 0
        System.out.println(array[1]); // 10
        System.out.println(array[2]); // 20
    }

}
```

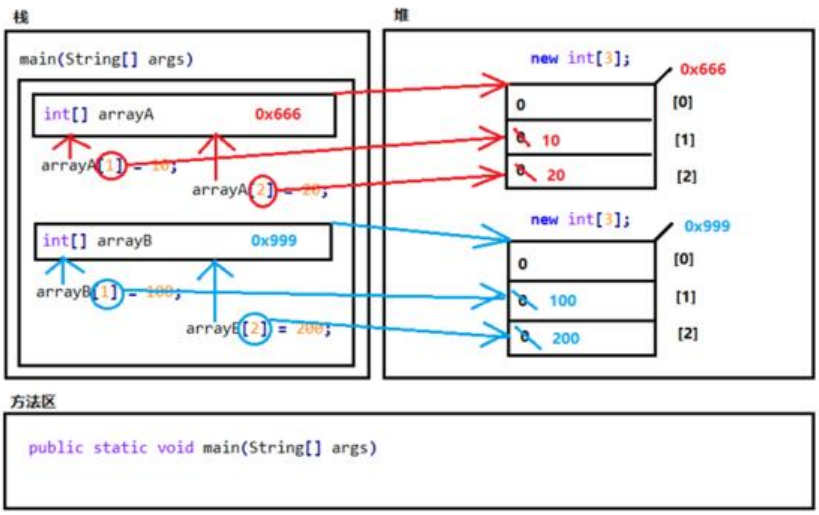


- 两个数组内存图

```java
public class Demo02ArrayTwo {
    public static void main(String[] args) {
        int[] arrayA = new int[3];
        System.out.println(arrayA); // 地址值
        System.out.println(arrayA[0]); // 0
        System.out.println(arrayA[1]); // 0
        System.out.println(arrayA[2]); // 0
        System.out.println("==============");
        arrayA[1] = 10;
        arrayA[2] = 20;
        System.out.println(arrayA); // 地址值
        System.out.println(arrayA[0]); // 0
        System.out.println(arrayA[1]); // 10
        System.out.println(arrayA[2]); // 20
        System.out.println("==============");
        int[] arrayB = new int[3];
        System.out.println(arrayB); // 地址值
        System.out.println(arrayB[0]); // 0
        System.out.println(arrayB[1]); // 0
        System.out.println(arrayB[2]); // 0
        System.out.println("==============");
        arrayB[1] = 100;
        arrayB[2] = 200;
        System.out.println(arrayB); // 地址值
        System.out.println(arrayB[0]); // 0
        System.out.println(arrayB[1]); // 100
        System.out.println(arrayB[2]); // 200
    }
}
```

```java
public class Demo02ArrayTwo {
    public static void main(String[] args) {
        int[] arrayA = new int[3];
        System.out.println(arrayA); // 地址值
        System.out.println(arrayA[0]); // 0
        System.out.println(arrayA[1]); // 0
        System.out.println(arrayA[2]); // 0
        System.out.println("==============");
        arrayA[1] = 10;
        arrayA[2] = 20;
        System.out.println(arrayA); // 地址值
        System.out.println(arrayA[0]); // 0
        System.out.println(arrayA[1]); // 10
        System.out.println(arrayA[2]); // 20
        System.out.println("==============");

        int[] arrayB = new int[3];
        System.out.println(arrayB); // 地址值
        System.out.println(arrayB[0]); // 0
        System.out.println(arrayB[1]); // 0
        System.out.println(arrayB[2]); // 0
        System.out.println("==============");
        arrayB[1] = 100;
        arrayB[2] = 200;
        System.out.println(arrayB); // 地址值
        System.out.println(arrayB[0]); // 0
        System.out.println(arrayB[1]); // 100
        System.out.println(arrayB[2]); // 200
    }
}
```

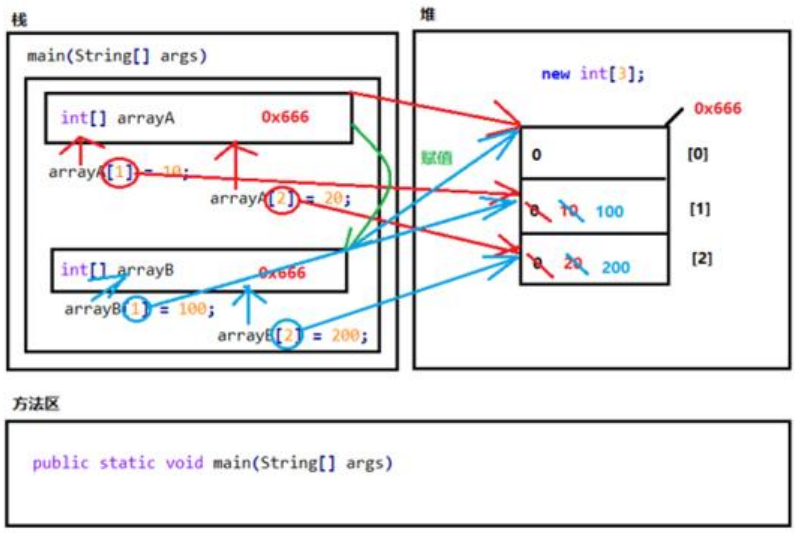


- 两个引用指向一个数组

```java
public class Demo03ArraySame {
    public static void main(String[] args) {
        int[] arrayA = new int[3];
        System.out.println(arrayA); // 地址值
        System.out.println(arrayA[0]); // 0
        System.out.println(arrayA[1]); // 0
        System.out.println(arrayA[2]); // 0
        System.out.println("==============");
        arrayA[1] = 10;
        arrayA[2] = 20;
        System.out.println(arrayA); // 地址值
        System.out.println(arrayA[0]); // 0
        System.out.println(arrayA[1]); // 10
        System.out.println(arrayA[2]); // 20
        System.out.println("==============");
        // 将arrayA数组的地址值，赋值给arrayB数组
        int[] arrayB = arrayA;
        System.out.println(arrayB); // 地址值
        System.out.println(arrayB[0]); // 0
        System.out.println(arrayB[1]); // 10
        System.out.println(arrayB[2]); // 20
        System.out.println("==============");
        arrayB[1] = 100;
        arrayB[2] = 200;
        System.out.println(arrayB); // 地址值
        System.out.println(arrayB[0]); // 0
        System.out.println(arrayB[1]); // 100
        System.out.println(arrayB[2]); // 200
    }
}
```

```java
public class Demo03ArraySame {
    public static void main(String[] args) {
        int[] arrayA = new int[3];
        System.out.println(arrayA); // 地址值
        System.out.println(arrayA[0]); // 0
        System.out.println(arrayA[1]); // 0
        System.out.println(arrayA[2]); // 0
        System.out.println("==============");
        arrayA[1] = 10;
        arrayA[2] = 20;
        System.out.println(arrayA); // 地址值
        System.out.println(arrayA[0]); // 0
        System.out.println(arrayA[1]); // 10
        System.out.println(arrayA[2]); // 20
        System.out.println("==============");

        int[] arrayB = arrayA;
        System.out.println(arrayB); // 地址值
        System.out.println(arrayB[0]); // 0
        System.out.println(arrayB[1]); // 10
        System.out.println(arrayB[2]); // 20
        System.out.println("==============");
        arrayB[1] = 100;
        arrayB[2] = 200;
        System.out.println(arrayB); // 地址值
        System.out.println(arrayB[0]); // 0
        System.out.println(arrayB[1]); // 100
        System.out.println(arrayB[2]); // 200
    }
}
```



原文链接：Java 数组