[Java 数组和Array类](https://www.cnblogs.com/wuchangming/archive/2013/03/28/2986184.html)

**1、数组创建的方法有很多种:**

int[] anArray ;//这一步只是做了声明，内存中并没分配，这一点是和C/C++不同的地方

anArray = new int[10];//这一步数组才真正建立

int[] anArray = {1,2,3};

**2、数组与Array类的关系:**

　　数组的写法与创建方式与一般的类有很大的不同，我一直认为数组与Java基本数据类型相似。其实不是。数组的创建确实与一般的类有所不同。

　　Java中的所有数组都是通过Array类实例化。Array类没有public的构造方法，数组是通过Array的newInstance()方法进行实例化。

源代码如下：

一维数组：

public static Object newInstance(Class<?> componentType, int length)//componentType 数组的类型，length数组的长度

throws NegativeArraySizeException {

return newArray(componentType, length); //这个方法会调用底层C代码开辟内存空间

}

多维数组：

public static Object newInstance(Class<?> componentType, int... dimensions)

throws IllegalArgumentException, NegativeArraySizeException {

return multiNewArray(componentType, dimensions);

}

**3、除去各种通过编译器简单化的写法后，还原真实的数组创建，过程如下：**

Object array = Array.newInstance(int.class,2);//创建数组实例

Array.set(array, 1/\*index\*/, 1);// 给数组元素赋值

Array.get(array, 1/\*index\*/);//取值

多维数组的类似，只是set和get的时候把第二维维度的数组理解为第一维度中的数据值的数据类型即可。

如：

Object array = Array.newInstance(int.class, 2,2);

Object arrayF = Array.get(array, 0);

Array.set(arrayF, 0, 1);

**4、本人的一些猜想**

1、数组的实例化之所以不用构造函数的方法，是因为数组的类型其实是无数种。用构造方法来实例化不利于对不同类型数组的扩展。

　　2、每个数组属于被映射为 Class 对象的一个类，所有具有相同元素类型和维数的数组都共享该 Class 对象。

 　　3、Java反射技术除了可以在运行时动态地决定要创建什么类型的对象，访问哪些成员变量，方法，还可以动态地创建各种不同类型，不同维度的数组。

**\*花边知识**

1、关于Class类可参考：<http://lavasoft.blog.51cto.com/62575/15433>

2、for循环遍历数组的方法

[复制代码](javascript:void(0);)

Employee[] staff = new Employee[3];

staff[0] = new Employee("a");

staff[1] = new Employee("b");

staff[2] = new Employee("c");

for( Employer e : staff){

System.out.println( e.getName() );

}

[复制代码](javascript:void(0);)