# Java 泛型

泛型（Generics ）是把类型参数化，运用于类、接口、方法中，可以通过执行泛型类型调用 分配一个类型，将用分配的具体类型替换泛型类型。然后，所分配的类型将用于限制容器内使用的值，这样就无需进行类型转换，还可以在编译时提供更强的类型检查。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | E：元素  K：键  N：数字  T：类型  V：值  S、U、V 等：多参数情况中的第 2、3、4 个类型  ? 表示不确定的java类型（无限制通配符类型） |

我们总结一下：

* 如果频繁支持读取数据，不要求写数据，使用<? extends T>。即生产者 使用 <? extends T>
* 如果频繁支持写入数据，不特别要求读数据，使用<? super T>。即消费者 使用 <? super T>
* 如果都需要支持，使用<T>。

**对此总结成一句话：泛型类型在逻辑上看以看成是多个不同的类型，实际上都是相同的基本类型。**

# 4. 泛型的使用

泛型有三种使用方式，分别为：泛型类、泛型接口、泛型方法

## 4.3 泛型类

泛型类型用于类的定义中，被称为泛型类。通过泛型可以完成对一组类的操作对外开放相同的接口。最典型的就是各种容器类，如：List、Set、Map。

泛型类的最基本写法（这么看可能会有点晕，会在下面的例子中详解）：

class 类名称 <泛型标识：可以随便写任意标识号，标识指定的泛型的类型>{

private 泛型标识 /\*（成员变量类型）\*/ var;

.....

}

}