# Java编程思想第4版笔记

## 对象入门

1. 所有东西都是对象。可将对象想象成一种新型变量；它保存着数据，但可要 求它对自身进行操作。理论 上讲，可从要解决的问题身上提出所有概念性的组件，然后在程序中将其表达为一个对象。
2. 程序是一大堆对象的组合；通过消息传递，各对象知道自己该做些什么。为 了向对象发出请求，需向那个对象“发送一条消息”。更具体地讲，可将消息想象为一个调用请求，它调用的是从属于目标对象的一个 子例程或函数。
3. 每个对象都有自己的存储空间，可容纳其他对象。或者说，通过封装现有对象，可制作出新型对象。所 以，尽管对象的概念非常简单，但在程序中却可达到任意高的复杂程度。
4. 每个对象都有一种类型。根据语法，每个对象都是某个“类”的一个“实例”。其中，“类”（Class） 是“类型”（Type）的同义词。一个类最重要的特征就是“能将什么消息发给它？”。
5. 同一类所有对象都能接收相同的消息。这实际是别有含义的一种说法，大家不久便能理解。由于类型为 “圆”（Circle）的一个对象也属于类型为“形状”（Shape）的一个对象，所以一个圆完全能接收形状消 息。这意味着可让程序代码统一指挥“形状”，令其自动控制所有符合“形状”描述的对象，其中自然包括 “圆”。这一特性称为对象的“可替换性”，是OOP 最重要的概念之一。

Java 采用三个显式（明确）关键字以及一个隐式（暗示）关键字来设置类边界：public，private， protected 以及暗示性的friendly

继承

使用继承时，相当于创建了一个新类。这个新类不仅包含了现有类型的所有成员（尽管private 成员被隐藏 起来，且不能访问），但更重要的是，它复制了基础类的接口。也就是说，可向基础类的对象发送的所有消 息亦可原样发给衍生类的对象。根据可以发送的消息，我们能知道类的类型。这意味着衍生类具有与基础类 相同的类型！为真正理解面向对象程序设计的含义，首先必须认识到这种类型的等价关系。

### 1.7 对象的创建和存在时间

1. C++认为程序的执行效率是最重要的一个问题，所以它允许程序员 作出选择。为获得最快的运行速度，存储以及存在时间可在编写程序时决定，只需将对象放置在堆栈（有时 也叫作自动或定域变量）或者静态存储区域即可。这样便为存储空间的分配和释放提供了一个优先级。某些 情况下，这种优先级的控制是非常有价值的。然而，我们同时也牺牲了灵活性，因为在编写程序时，必须知 道对象的准确的数量、存在时间、以及类型。如果要解决的是一个较常规的问题，如计算机辅助设计、仓储 管理或者空中交通控制，这一方法就显得太局限了。

1. 所以在内存堆里分配存储空间的时间比在堆栈里创建 的时间长得多（在堆栈里创建存储空间一般只需要一个简单的指令，将堆栈指针向下或向下移动即可）。由 于动态创建方法使对象本来就倾向于复杂，所以查找存储空间以及释放它所需的额外开销不会为对象的创建 造成明显的影响。除此以外，更大的灵活性对于常规编程问题的解决是至关重要的。

### 1 .7.1 集合与继承器

1. 既然如此，怎样才能知道那些对象要求多少空间呢？事先上根本无法提前知道，除非进入 运行期。

### 1.7.3 集合库与方便使用集合

1. 下溯造型与模板／通用性

Circle（圆）属于 Shape（几何形状）的一种类型，所以上溯造型是安全的。但 我们不知道一个Object 到底是 Circle 还是Shape，所以很难保证下溯造型的安全进行，除非确切地知道自 己要操作的是什么。

Java 比C++简单，但付出的代价是效率以及一定程度的灵活性

### 1.9 多线程

所以同一时间只能有一个线程使用特定的内存空间。 为达到这个目的，需要使用synchronized 关键字。其他类型的资源必须由程序员明确锁定，这通常要求程序 员创建一个对象，用它代表一把锁，所有线程在访问那个资源时都必须检查这把锁。

## 一切都是对象

### 2 . 8 . 1 注释文档

1. 提取注释的工具叫作javadoc。它采用了部分来自Java 编译器的技术，查找我们置入程序的特殊注释标记。它不仅提取由这些标记指示的信息，也将毗邻注释的类名或方法名提取出来
2. javadoc 输出的是一个HTML 文件，可用自己的Web 浏览器查看。该工具允许我们创建和管理单个源文件，并生动生成有用的文档。由于有了jvadoc，所以我们能够用标准的方法创建文档。而且由于它非常方便，所以我们能轻松获得所有Java 库的文档。

## 第三章

## 初始化和清除

### 4 . 2 方法过载

另一项因素强迫方法名出现过载情况：构建器。由于构建器的名字由类名决定，所以只能有一个构建器名称。

### 4 . 2 . 3 返回值过载

因为我们关心的不是返回值，而是方法调用的其他效果