目录

[Thinking in Java 2](#_Toc494637989)

[一、 对象入门 2](#_Toc494637990)

[1.1 基本概念 2](#_Toc494637991)

[1.2 类边界 2](#_Toc494637992)

[1.3 成员对象 2](#_Toc494637993)

[1.4 抽象类与接口 4](#_Toc494637994)

[1.5 单根结构 5](#_Toc494637995)

[1.6 程序设计方法 6](#_Toc494637996)

[二、 一切都是对象 6](#_Toc494637997)

[2.1 用句柄操纵对象 6](#_Toc494637998)

[2.2 数据保存 6](#_Toc494637999)

[2.3 类 7](#_Toc494638000)

[2.4 Static关键字 7](#_Toc494638001)

[2.5 javadoc 7](#_Toc494638002)

[三、 控制程序流程 7](#_Toc494638003)

[常用类 8](#_Toc494638004)

[一、 String类 8](#_Toc494638005)

[二、 ArrayList类 8](#_Toc494638006)

[1. ArrayList的使用 8](#_Toc494638007)

[三、 HashMap类 8](#_Toc494638008)

[四、 HashSet类 8](#_Toc494638009)

[五、 Iterable<T>类 8](#_Toc494638010)

# Thinking in Java

Fourth Edition. Bruce Eckel.

## 对象入门

### 基本概念

1. 面向对象编程（Object Oriented Programming, OOP）:OOP的一条基本原则是计算机程序是由单个能够起到子程序作用的单元或对象组合而成。

*OOP达到了软件工程的三个主要目标：重用性、灵活性和扩展性。*

1. 对象（Object）：问题空间中的元素以及它们在方案空间的表示物。
2. 接口（Interface）：泛指实体把自己提供给外界的一种抽象化物（可以为另以实体），用以由内部操作分离出外部沟通方法，使其能被内部修改而不影响其他实体与其交互的方式。
3. 集合：是一个用来放置对象的储藏所。
4. 继承器（Iterator）：将集合抽象成一个简单的序列。允许遍历该序列，而不用关心基础接口是什么——换言之，不管它是一个矢量、一个链接列表、一个堆栈或者其他。

Eg:

Light lt = new Light(); lt.on(); 通过简单地声明一个名字lt，为Light对象创建了一个句柄。然后用new关键字新建类型为Light的一个对象，再用等号将其赋给句柄。为了向对象发送一条消息，列出句柄名lt，再用一个句点符号.把它同消息名称on连接起来。

***句柄****：一个句柄是指使用的一个唯一的整数值，即一个4字节（64位程序中为8字节）长的数值，来标识应用程序中的不同对象和同类中的不同的实例。应用程序能够通过句柄访问相应的对象的信息，但是句柄不是指针，程序不能利用句柄来直接阅读文件中的信息。*

***程式化编程****（Procedural Programming）：*

### 类边界

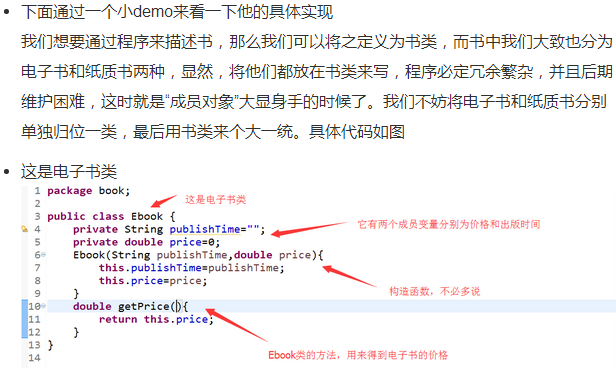
三个显式（明确）关键字以及一个隐式（暗示）关键字，public, private, protected以及friendly。若未明确指定其他关键字，则默认为后者。

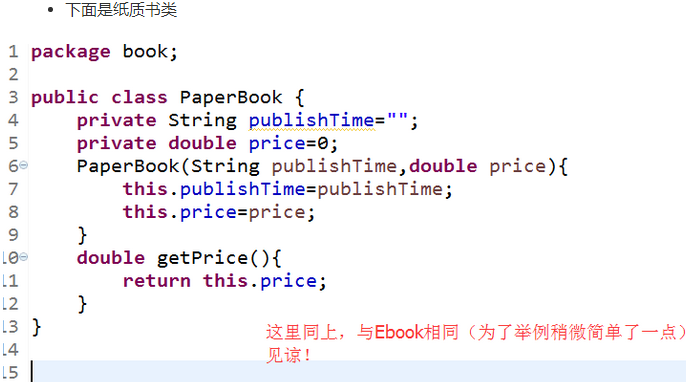
* Public：任何人均可使用
* Private：除自己、类型的创建者以及那个类型的内部函数成员，其他任何人都不能访问后续的定义信息。
* Friendly：涉及“包装”或“封装”的概念——即Java用来构建库的方法。意味着它只能在这个包装的范围内使用（所以这一访问级别有时也叫做“包装访问”）。
* Protected：与private相似，只是一个继承的类可以访问受保护的成员，但不能访问私有成员。

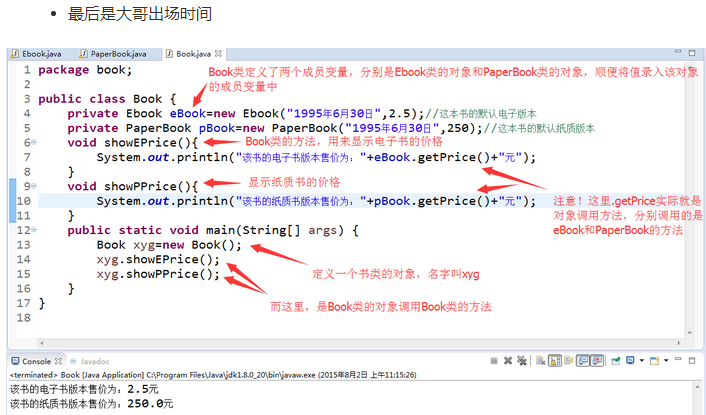
### 成员对象

Java中最简单的复用某个类的方式就是直接使用该类的对象，此外也可以将那个类的一个对象置于某个新类中，即“创建一个成员对象”。新的类可以由任意数量、任意类型的其他对象以任意可以实现新的类中想要的功能的方式所组成。如，包含关系，“一辆车包含了一个变速箱”。

如，A类的成员变量的类型是B类，那么A类的成员变量的属性就是B类的属性，那么该成员变量也可以调用与之所对应类型B类的方法。见下例[[1]](#footnote-1)。







*心得：刚从面向过程的学习过度到面向对象，对于类定义的成员变量不是基础类型如int。而是一个自己定义的类，在用的时候一时转不过弯来。可以这样理解，把int型也当成一个类，double型也当成一个类，诚然，他们都有包裹类，int型对应Integer类，double型对应Double类，在用法上类和类是一样的。把基础类也当成我们定义的类来想象能有所帮助。*

*如，int i = 10; Integer i = new Integer(10); //这里和我们定义的类都一样，他可以调用Integer的方法。*

### 抽象类与接口[[2]](#footnote-2)

#### 抽象类

抽象方法是一种特殊的方法：它只有声明，而没有具体的实现。>>abstact void fun(); 抽象方法必须用abstract关键字进行修饰。如果一个类含有抽象方法，则称该类为抽象类，而抽象类必须在类前用abstract关键字修饰。

抽象类就是为了继承而存在的。对于一个父类，如果它的某个方法在父类中实现出来没有任何意义，必须根据子类的实际需求来进行不同的实现，那么就可以将这个方法声明为abstract方法，此时这个类就称为了abstract类了。

包含抽象方法的类称为抽象类，但并不意味着抽象类中只能有抽象方法，它和普通类一样，同样可以拥有成员变量和普通的成员方法。注意抽象类与普通类的主要有三点区别：

1. 抽象方法必须为public或者protected（因为如果为Private，则不能被子类继承，子类便无法实现该方法），缺省情况下默认为public。
2. 抽象类不能用来创建对象。
3. 如果一个类继承于一个抽象类，则子类必须实现父类的抽象方法。如果子类没有实现父类的抽象方法，则必须将子类也定义为抽象类。

在其他方面，抽象类和普通类并没有区别。

#### 接口

接口，泛指供别人调用的方法或者函数。它是对行为的抽象。接口中可以含有变量和方法。但是，接口中的变量会被隐式的指定为public static final变量，而方法会被隐式的制定我public abstract方法，并且接口中所有的方法不能有具体的实现，也就是说，接口中的方法必须是抽象方法。从这里可以隐约看出接口和抽象类的区别，接口是一种极度抽象的类型，它比抽象类更加抽象，并且在一般情况下不在接口中定义变量。

**如果一个非抽象类遵循了某个接口，就必须实现该接口中的所有方法。**对于遵循某个接口的抽象类，可以不实现该接口中的抽象方法。

#### 抽象类和接口的区别

1. 语法层面上的区别
2. 抽象类可以提供成员方法的实现细节，而接口中只能存在public abstract方法。
3. 抽象类中的成员变量可以是各种类型的，而接口中的成员变量只能是public static final类型的。
4. 接口中不能含有静态代码块以及静态方法，而抽象类可以有静态代码块和静态方法。
5. **一个类只能继承一个抽象类，而一个类却可以实现多个接口。**
6. 设计层面上的区别
7. 抽象类是对一种事物的抽象，即对类抽象，而接口是对行为的抽象。抽象类是对整个类整体进行抽象，包括属性、行为，但是接口却是对类局部（行为）进行抽象。

举个例子，飞机和鸟是不同类的事物，但是它们都有一个共性，就是都会飞。那么在设计的时候，可以将飞机设计为一个类Airplane，将鸟设计为一个类Bird，但是不能将飞行这个特性也设计为类，因此它只是一个行为特性，并不是对一类事物的抽象描述。此时可以将飞行设计为一个接口Fly，包含方法fly()，然后Airplane和Bird分别根据自己的需要实现Fly这个接口。然后至于有不同种类的飞机，比如战斗机、民用飞机等直接继承Airplane即可，对于鸟也是类似的，不同种类的鸟直接继承Bird类即可。从这里可以看出，**继承是一个"是不是"的关系，而接口实现则是 "有没有"的关系。**如果一个类继承了某个抽象类，则子类必定是抽象类的种类，而接口实现则是有没有、具备不具备的关系，比如鸟是否能飞（或者是否具备飞行这个特点），能飞行则可以实现这个接口，不能飞行就不实现这个接口。

1. 设计层面不同，抽象类作为很多子类的父类，它是一种模板式设计。而接口是一种行为规范，它是一种辐射式设计。什么是模板式设计？最简单例子，大家都用过ppt里面的模板，如果用模板A设计了ppt B和ppt C，ppt B和ppt C公共的部分就是模板A了，如果它们的公共部分需要改动，则只需要改动模板A就可以了，不需要重新对ppt B和ppt C进行改动。而辐射式设计，比如某个电梯都装了某种报警器，一旦要更新报警器，就必须全部更新。也就是说对于抽象类，如果需要添加新的方法，可以直接在抽象类中添加具体的实现，子类可以不进行变更；而对于接口则不行，如果接口进行了变更，则所有实现这个接口的类都必须进行相应的改动。

### 单根结构

**所有类最终都从单独一个基础类继承，即“Object”。**单根结构中的所有对象（比如所有Java对象）都可以保证拥有一些特定的功能。在自己的系统中，我们知道对每个对象都能进行一些基本操作。一个单根结构，加上所有对象都在内存堆中创建，可以极大简化参数的传递。

***脚本语言****：又称为扩建的语言，或者动态语言，是一种编程语言，用来控制软件应用程序。常见的有JavaScript、VBScript、Perl、PHP、Python、Ruby。特点是语法简单，一般以文本形式保存，并且不需要编译成目标程序，在调用的时候直接解释。比如JavaScript，只需要用记事本新建一个HTML文件，在里面加上一段脚本，在浏览器打开HTML文件时会自动调用JavaScript脚本。总之就是用记事本写出程序，不需要编译环境生成目标程序的基本上就是脚本语言。*

### 程序设计方法

在程序设计中，时刻提醒自己：

1. 对象是什么？（怎样将自己的项目分隔成一系列单独的组件？）
2. 它们的接口是什么？（需要将什么消息发给每一个对象？）
3. 阶段0：你拟出一个计划

可能需要决定一些必要的附加处理结构。设定每个阶段的目标。

1. 阶段1：要制作什么？

核心问题：决定采用什么系统？

解决问题：采用“假如…，系统该怎么做？”。

尽可能总结出自己系统的一套完整的“使用条件”或者“应用场合”。该工作就是想让系统完成的核心任务。接着用几个简单的段落对自己的系统做出描述，然后围绕它们进行扩充，添加一些“名词”和“动词”。“名词”就是对象，而“动词”就是要整合到对象接口中的“方法”。

1. 阶段2：如何构建？

完成一套设计方案，并解释其中包含的各类对象在外观上是什么样子，以及相互间是如何沟通的。此时可考虑一种特殊的图标工具：“统一建模语言”（UML）。

***统一建模语言（UML）****：《信息系统设计与分析》上所教授的图表。*

1. 阶段3：开始创建
2. 阶段4：校订

整个开发周期还没有结束。但是第一次做的东西并不完善，所以需为自己留下一个深入学习、认知的空间，再回头做一些改变。

***设计范式：参考本书第16章，待补充。***

## 一切都是对象

本章将探讨Java程序的基本组件，并体会为什么说Java乃至Java程序内的一切都是对象。

### 用句柄操纵对象

注意，尽管将一切都“看作”对象，但操作的标识符实际是指向一个对象的**“句柄”（handle）**。可以拥有一个句柄，但不表示必须有一个对象同它连接。

>>String s; 如果想容纳一个词或句子，这里创建的只是句柄，并不是对象。

>>String s = “asdf”; 初始化了一个字符串

### 数据保存

对象句柄保存在堆栈中。驻留于常规RAM（随机访问存储器）区域，但可通过它的“堆栈指针”获得处理的直接支持。

对象保存在堆中，一种常规用途的内存池（也在RAM区域）。

*随机存取存储器（****RAM****）：又称“随机存储器”，****主存（内存）****。是与CPU直接交换数据的内部存储器。可以随时读写，而且速度很快，通常作为操作系统或其他正在运行中的程序的临时数据存储媒介。*

### 类

类：数据成员（字段） + 成员函数（方法）

>>DataOnly d = new DataOnly();

>>d.i = 48; //**对象句柄.成员**，可以将值赋给数据成员

数据成员是一种对象（通过它的句柄与其通信），可以为任何类型。它可以是主类型（并不是句柄）之一。

方法：完成某事的途径。 //**对象名.方法名（自变量1，自变量2，自变量3…）**

### Static关键字

作用：

1. 只想用一个存储区域来保存一个特定的数据——无论要创建多少个对象，公用同样的内存区域。
2. 需要一个特殊的方法，它没有与这个类的任何对象关联。

**Eg\*\***

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | class StaticTest**{**  Static int i **=** 47**;**  **}**  StaticTest st1 **=** **new** StaticTest**();**  StaticTest st2 **=** **new** StaticTest**();**  StaticTest**.**i **++;** |

尽管我们制作了两个StaticTest对象，但它们仍然只占据StaticTest.i 的一个存储空间，这两个对象都共享同样的i。无论st1.i 或 st2.i 都有同样的值47，因为它们引用的是同样的内存区域。可以直接使用st2.i，也可以使用StaticTest.i。++运算符使变量增值，此时，无论st1.i 或 st2.i 的值都是48.

### javadoc

将代码同文档链接起来。使用特殊的注释语法，以便提取注释。Javadoc输出的是一个HTML文件，可用自己的web浏览器进行查看。具体的Javadoc语法请自行搜索，TIJ书上也有。

在eclipse提取文档的操作步骤：Project 🡪 Generate Javadoc 🡪 Javadoc command: jdk\bin\javadoc.exe; private; doclet: Destination 🡪 Finish.

## 控制程序流程

# 常用类

## String类

|  |  |
| --- | --- |
| 1  2 | str**.**charAt**(**int index**)** //返回index位置  str**.**length**();** |

## ArrayList类

ArrayList即动态数组，好处：动态的增加和减少元素；实现了ICollection和List的接口；灵活的设置数组的大小。

### ArrayList的使用

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | ArrayList**<**Integer**>** alist **=** **new** ArrayList**<**Integer**>();** //声明  alist**.**add**(**int num**);** //添加元素  alist**.**get**(**int index**);** //index位置上的元素  alist**.**size**();** //长度  String**[]** slist **=** **(**String**[])**alist**.**toArray**(new** String**[**size**]);** //ArrayList转String[]  Object**[]** olist **=** alist**.**toArray**();** //ArrayList转数组  alist**.**contains**(**x**);** //返回boolean，是否包含  alist**.**isEmpty**();** //返回boolean  alist**.**indexOf**(**Object o**);** //o所在的位置 |

## HashMap类

## HashSet类

## Iterable<T>类

1. http://www.jianshu.com/p/aed30c8c14d9 [↑](#footnote-ref-1)
2. http://www.cnblogs.com/dolphin0520/p/3811437.html [↑](#footnote-ref-2)