面向对象的思考过程

（原书第二版）

The Object-Oriented Thought Process
(Second Edition)

[美]Matt Weisfeld 著
杨会珍 尹清辽 等译

中国水利水电出版社
www.waterpub.com.cn

面向对象的思考过程

（原书第二版）

[美] Matt Weisfeld 著

杨会珍 尹清辽 等译

中国水利水电出版社

内 容 提 要

面向对象的程序设计（OOP）是现代程序设计语言的基本概念，本书深入探讨了如何以面向对象的方式来进行思考。全书共分为 15 章，内容包括：面向对象的基本概念，如何以对象的方式进行思考，高级的面向对象概念，类的剖析，类设计，继承和组合，框架和重用，创建对象，使用 UML 创建对象模型，持久对象，可移植的数据，分布式对象和企业，设计模式等。

本书是当代程序员的基础理论读物，适合于大学本科计算机专业的学生以及想要学习面向对象技术的程序员。

图书在版编目（CIP）数据

面向对象的思考过程：第 2 版 /（美）韦森菲尔德（Weisfeld,M.）著；杨会珍等译. —北京：中国水利水电出版社，2004

书名原文：The Object-Oriented Thought Process

ISBN 7-5084-2291-0

Ⅰ.面… Ⅱ.①韦…②杨… Ⅲ.面向对象语言—程序设计 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2004）第 077127 号

书　　名	**面向对象的思考过程（原书第二版）**
作　　者	［美］ Matt Weisfeld 著
译　　者	杨会珍 尹清辽 等译
出版 发行	中国水利水电出版社（北京市三里河路 6 号 100044）
	网址：www.waterpub.com.cn
	E-mail：mchannel@263.net（万水）
	sales@waterpub.com.cn
	电话：（010）63202266（总机） 68331835（营销中心） 82562819（万水）
经　　售	全国各地新华书店和相关出版物销售网点
排　　版	北京万水电子信息有限公司
印　　刷	北京北医印刷厂
规　　格	787mm×1092mm 16 开本 13.25 印张 296 千字
版　　次	2004 年 8 月第 1 版 2004 年 8 月第 1 次印刷
印　　数	0001—5000 册
定　　价	26.00 元

译者序

随着 Java、C#.NET、Visual Basic .NET 这些面向对象语言越来越流行，对面向对象（OO）概念与编程思想的理解也变得越来越重要。很多老程序员正从传统的程序设计语言（如 C、Visual Basic）转向面向对象的程序设计语言（如 Visual Basic .NET、Java、C#），还有一些没有严格遵守 OO 惯例的 C++程序员，也正在逐步地遵循 OO 编程思想，而对于不断涌现的新程序员来说，当今面向对象程序设计是一种主流，因此他们刚入门时就应该牢固掌握 OO 编程的概念。本书就是这样一本书，帮助这些程序员掌握 OO 编程的概念，树立 OO 编程的思考方式。

本书是学习面向对象程序设计的基础读物，它对面向对象的概念做了精辟的讲述。这些概念基本覆盖了面向对象程序设计的各个方面，包括类、对象、消息、方法、属性、继承、封装、多态、公共接口/私有实现、抽象方法/类、聚合/关联、构造函数/析构函数、异常、重载、访问器方法和接口等。书中还提供了一些基本的面向对象程序设计的原则，在类设计和软件开发方面都给出了一些实例。本书在介绍 OO 概念时使用 UML 图来描述，通过 Java 代码示例来实现。本书讲解清晰而集中，并向读者提出了一些忠告，能让读者在很短的时间内洞悉面向对象的奥秘，因而非常值得一读。

本书共分为 15 章，内容包括：面向对象的基本概念，如类、对象、继承、封装、多态等；如何以对象的方式进行思考；高级的面向对象概念，如构造函数、错误处理、作用域、重载、多重继承等；类的剖析；类设计指导；使用对象进行设计；继承和组合；框架和重用：利用接口和抽象类进行设计；创建对象；使用 UML 创建对象模型；持久对象：串行化和关系数据库；对象和 XML：可移植的数据；对象和 Internet；分布式对象和企业；设计模式等。另外，书中每章还给出了大量的参考资料，供读者深入学习 OO 的细节内容。

本书是对基本 OO 概念的总体介绍，是当代程序员的基础理论读物。本书适合于大学本科计算机专业的学生作为程序设计理论的教材，也适合于想要学习面向对象技术的程序员，包括程序设计人员、开发人员、项目管理人员以及任何想要总体了解面向对象的人。阅读本书将为阅读其他讲述更高级的 OO 主题的图书打下坚实的基础。

本书由杨会珍、尹清辽翻译，另外还要感谢孙美娟、张文起、于旭升、盛海燕、谢君英为本书翻译所做的工作。本书代码录入和初排工作由杨晓梅完成。译者在翻译过程中尽心尽力，但错误仍在所难免，恳请读者指正。

译者

2004 年 7 月

前　言

顾名思义，本书是关于面向对象（OO）的思考过程的。很明显，选择本书的主题和书名都是重要的决定，然而，这些决定并不简单。许多图书讲述面向对象的一种或者另一种层次。若干畅销图书讲述以下主题：OO 分析、OO 设计、OO 程序设计、设计模式、OO 数据库、统一建模语言（UML）、各种 OO 程序设计语言以及许多与 OO 程序设计有关的其他主题。

然而，在研究以上所有图书时，许多人忘记各个主题都是建立在下面这个单一基础之上：如何以 OO 的方式进行思考。遗憾的是，许多软件开发专家常常还没有花适当的时间和精力来真正了解这些图书中的概念，就开始钻研它们。

我认为学习 OO 概念不是通过学习某种具体的开发方法或者某组工具就可以实现的。简而言之，以 OO 的方式完成工作是一种思考方法。本书完全和 OO 思考过程有关。

从 OO 思考过程中分离出方法和工具并不容易。许多人都是通过其中一种方法或者工具来了解 OO 概念的。许多 C 程序员是在直接转移到 C++时首次了解到面向对象，在此之前甚至没有间接接触这个方法。一些软件专业人员是通过演示文稿（包含使用 UML 的对象模型）而首次接触面向对象的，在此之前他们也没有直接接触过 OO 概念。

理解学习 OO 概念和使用支持该范例的方法和工具之间的显著区别是很重要的。在“What *the UML Is—and Isn't*”一文中，作者 Craig Larman 说：“遗憾的是，软件工程和 UML 图表语言领域中，获得阅读和编写 UML 符号的技能在某些时候似乎和获得面向对象分析和设计的技能同等重要。当然，事实并非如此，后者比前者更重要。因此，我建议寻找以下培训和培训材料：其中面向对象的分析和设计要比 UML 符号或者某种电脑辅助软件工程工具的用途更为重要。”

尽管学习一门建模语言是一个很重要的步骤，但是先学习 OO 技能更重要。在学习 OO 概念之前先学习 UML，类似于在不了解任何电子技术时学习如何阅读电气图。

程序设计语言存在相同的问题。前面说过，许多 C 程序员还没有直接接触到 OO 概念，就通过转移到 C++而进入面向对象领域。很多情况下，声称自己是 C++程序员的开发人员只是使用 C++编辑器的 C 程序员。

随着 Java、C#.NET、Visual Basic .NET 之类的面向对象语言越来越流行，这个问题越来越重要。有很多 Visual Basic 程序员必须转移到 Visual Basic .NET 中。类似地，许多可能没有遵守严格的 OO 惯例的 C++程序员，需要转移到 Java 或者 C#中，在使用这些语言时，他们就别无选择，只能以 OO 方式进行思考。

早期的 Visual Basic 版本不是 OO 的。C 不是 OO 的，而开发 C++时考虑了与 C 的向后兼容性。因此，仅仅使用 C++编辑器编写 C 的句子构造，而抛弃了 C++所有的 OO 特性，这是非常可能的。更坏的情况下，程序员可以使用足够的 OO 特性，刚好使某个程序不能

使用 OO 进行理解，而非 OO 程序员也可以这样。

因此，在开始 OO 开发的时候，首先学习基本的 OO 概念是至关重要的。我们应该抵挡住直接跨入某种程序设计语言（如 C++、C#或者 Java）或者某种建模语言（如 UML）的诱惑，而花时间来学习面向对象的思考过程。

本书是向程序员介绍面向对象技术的概念性图书。它的读者对象当然包括想要跨越到面向对象技术的结构化程序设计人员。因此，书中包括的一些资料实际上是结构化的技术和面向对象技术之间的桥梁。第 6 章就是该方法的一个好例子，该章添加了结构化程序设计人员所熟悉的技术。重要的是理解面向对象和结构化实践并非是相互排斥的。结构化技术的使用在 OO 设计中随处可见（仅仅考虑 for 循环或者 if 语句）。

20 世纪 80 年代后期，在我的第一堂 Smalltalk 课上，教师告诉全班同学，新的 OO 范例是一种全新的思考方式。他接着说到："虽然我们差不多都是非常优秀的程序员，但是其中的 10%～20%永远都不能真正以 OO 的方式完成工作。"如果这个陈述确实是正确的，也主要是因为一些人从来没有真正地花时间进行范例转换并学习基础的 OO 概念。

第二版的新内容

正如我常说的，我认为第一版主要是一本概念性的图书。尽管第二版仍然坚持这个目标，但我还添加了若干应用主题，它们非常适合于面向对象的概念。这些应用包括以下几点：

- 对象建模
- 对象持久性
- XML
- 对象和 Internet
- 企业
- 设计模式

从本质上说，讲述这些主题的章节仍然是概念性的。然而，其中一些章节包括 Java 代码，它们展示了如何实现这些概念。

读者对象

本书是对基本的 OO 概念的总体介绍。适合的读者包括设计人员、开发人员、项目管理人员以及任何想要总体了解面向对象的人。阅读这本书将为阅读其他讲述更高级的 OO 主题的图书打下坚实的基础。

在这些主题更高级的书中，我最喜欢的图书之一仍然是 Stephen Gilbert 和 Bill McCarty 所著的《*Object-Oriented Design in Java*》。我确实非常喜欢该书中的方法，并且在我教过的 OO 概念课堂上，将它用作教科书。在本书中我经常引用到《*Object-Oriented Design in Java*》，并且推荐您学完本书之后学习这本书。

其他我认为非常有用的图书包括 Scott Meyers 所著的《*Effective C++*》、Stephen R. Schach 所著的《*Classical and Object-Oriented Software Engineering*》、Bruce Eckel 所著的《*Thinking in C++*》、Martin Fowler 所著的《*UML Distilled*》以及 Peter Coad 和 Mark Mayfield 所著的《*Java Design*》。

在公司和学校向程序员讲授介绍性的 Java 时，我很快就发现其中很多程序员很容易掌握 Java 语法。然而，他们很难了解该语言的 OO 本质。

本书的范围

到目前为止很明显，我坚信在跨入某个程序设计语言或者建模语言之前先熟悉面向对象的思考过程是很有益的。本书处处都是 Java 代码例子以及 UML 图，然而，您不必了解 Java 或者 UML 来阅读它们。既然我说过先学习概念，为什么这里有这么多 Java 代码和 UML 图呢？首先，用它们来阐述 OO 概念都很棒。其次，它们对 OO 过程都很重要，并且我们应该对其进行介绍性的讲解。重点并没有放在 Java 或者 UML 上，而是使用它们作为辅助工具，以了解基本概念。

本书中的 Java 例子说明循环和函数之类的概念。然而，理解这些代码本身并不是理解这些概念的先决条件，手头拥有讲述 Java 语法的书可能会有所帮助。

强调一下，本书不讲述 Java 或者 UML，讲述它们都需要几卷的篇幅。我希望本书能激起您学习其他 OO 主题（如 OO 分析、面向对象的设计以及 OO 程序设计）的愿望。

本书约定

本书使用以下排版约定：

- 代码行、命令、语句以及其他与代码相关的术语是以等宽字体（monospace）出现的。
- 代表实际应该键入的内容的占位符是以斜体等宽字体（*italic monospace*）出现的。应该键入的文本是以粗体等宽字体（**bold monospace**）出现的。
- 贯穿本书，有一些特殊的说明性内容，例如：

注意

注意显示关于该讨论的有趣信息，它是一点见解或者对新技术的启示。

提示

提示向您提供建议或者展示完成任务的更容易的方式。

警告

警告提醒您某个可能的问题并向您提供如何避免该问题的建议。

本书中使用的源代码

您可以从 http://www.samspublishing.com 下载本书中讨论到的所有源代码和例子。在“search”窗口，简单地键入本书的 ISBN（0672326116），按下 Enter 健，然后就出现带有指向源代码链接的页面。

关于作者

Matt Weisfeld 是俄亥俄州 Cleveland 市 Cuyahoga Community College（Tri-C）的助理教授。Matt 是信息技术系的教员，讲授 C++、Java 和 C# .NET 之类的程序设计语言。在 Tri-C 任职之前，Matt 在信息技术行业工作了 20 年，获得了软件开发、项目管理、业务拓展、公司培训以及兼职教学方面的经验。Matt 获得了计算机科学的硕士学位以及项目管理的 MBA。除了《面向对象的思考过程》第一版，Matt 还出版了其他两本计算机图书，并且在 *Dr. Dobb's Journal*、*The C/C++ Users Journal*、*Software Development Magazine*、*Java Report* 之类的杂志和期刊以及国际期刊 *Project Management* 上发表过许多文章。Matt 曾出席美国和加拿大的各种会议。

致　　谢

和第一版相同，本书是许多人努力的成果。我想花一些时间来感谢尽可能多的人，因为如果没有他们，本书永远都不会问世。

首先，我想感谢我的妻子 Sharon，感谢她的所有帮助。她不仅在漫长的编写过程中给予了我支持和鼓励，而且与第一版一样，她编辑了每一章的初稿。

我还要感谢我的妈妈和其余家人，感谢他们对我的不断支持。

在编写这两版图书时，我非常喜欢与 Sam 的人员合作。和编辑 Todd Green、Seth Kerney、Matt Purcell 以及 Songlin Qiu 合作是非常令人开心的事。

感谢 Dennis Vargo、Handel Gibbings 和 Bob Reselman，他们对原稿进行了技术编辑。

最后要感谢我的女儿 Stacy 和 Stephanie 以及我的猫 Duffy，她们使我专心工作。Stacy 为原稿制作了最初的插图。

目　　录

第 1 章　面向对象的概念简介

尽管可能非常令人惊奇，但自 20 世纪 60 年代早期以来，面向对象（OO）软件开发就已经出现了。尽管在当今的软件行业中，对象已经变得相当流行，但许多软件厂商尚未涉足 OO 领域。大家都知道，软件产业有时进展缓慢。而另一点也是事实：在工作系统完好时，一定有充足的理由才能替换它们。这已经阻碍了 OO 系统的普及。有许多非 OO 遗留系统（*legacy system*，也就是已经存在的旧系统）看起来还运行良好——因此为什么要冒着潜在的危险来更改它们呢？在多数情况下，不应该改变它们，至少不要为了更改而改变它们。非 OO 代码编写的系统没有什么本质错误。不过，全新的开发理所应当考虑使用 OO 技术。

尽管在最近的十年中，OO 开发已经有稳定显著的增长，但全新的竞争有助于进一步将它推向主流。Web 的出现已经打开了全新的竞争领域，在该领域中，多数软件开发是新的，而且基本上不会受到遗留问题的阻碍。即使存在遗留的问题，也倾向于将遗留系统包装到对象包装器（wrapper）中。

对象包装器

对象包装器是内部包含结构化代码的面向对象的代码。例如，您可以编写结构化的模块，然后把它包装到对象中，使它看起来似乎是一个对象。

对象正在缓慢但稳步地进入我们的专业信息系统（IS）生活——它们是不能被忽视的。由于 Java 的成功以及 Microsoft .NET 技术的引入，对象正在变成 IS 系统的主要部分。由于 Internet 的蓬勃发展，并经过多年的打造，电子高速公路实际上变成了基于对象的高速公路。而且随着商业向 Web 方向的发展，它们也正在向着对象发展，因为用于 Web 的技术本质上通常都是 OO。

本章概述了基本的 OO 概念。讲述的主题涉及后续章节论及的很多（假如不是全部的话）主题，不过后续章节还会更加详细地探讨这些问题。

1.1　面向过程程序设计和 OO 程序设计

在我们深入研究 OO 开发的优点之前，让我们考虑一个非常基本的问题：准确地讲，什么是对象？这是一个既复杂又简单的问题。说它复杂是因为调整方向来学习全新的思维方式并不是简单的事情。说它简单是因为多数人已经以对象的方式进行思考。

例如，当你看某个人时，就会把这个人看成对象。对象是由两个术语定义的：属性和行为。人具有属性，例如眼睛颜色、年龄和身高等。人也具有行为，例如行走、谈话和呼吸等。在基本定义中，对象是同时包含数据和行为的实体。同时一词是更传统的程序设计