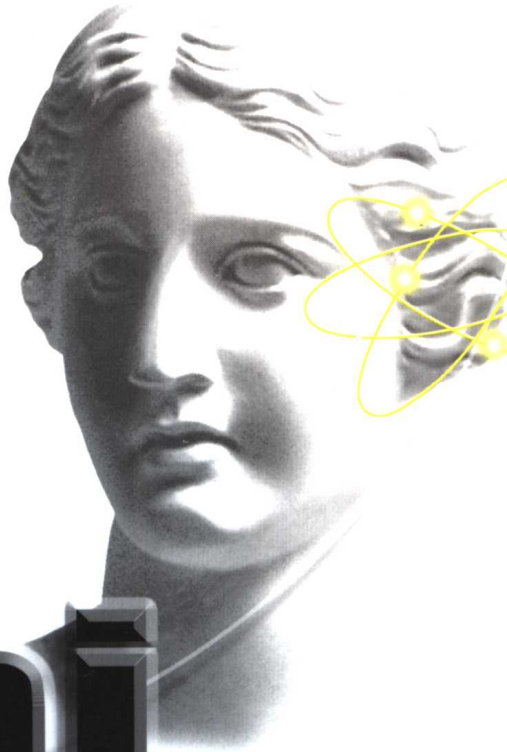
HZ BOOKS

Delphi

Borland
核心技术丛书



Delphi 面向对象编程思想

刘艺 著

从 Delphi RAD 快手成长为 OOP 高手的必读秘笈

附赠

机械工业出版社
China Machine Press

Borland 核心技术丛书

Delphi 面向对象编程思想

刘 艺 著

机械工业出版社
China Machine Press

这是一本纯粹讨论 Delphi 面向对象编程的力作。本书以精通 Delphi 面向对象编程为目的，深入浅出地讲解了 Delphi 面向对象的概念和实质、方法和经验、思想和实践；详尽讨论了 Delphi 建立在虚方法、抽象方法、对象接口等动态绑定机制上和向上转型、向下转型、接口转型等类型转换机制上的面向对象高级技巧；并深入研究了通过封装从而实现界面和业务对象的分离，从界面和业务分离逐步实现分布式多层体系结构，进而实现界面和业务应用的跨平台的企业级解决方案。本书还提供了 VCL 的内幕资料和研究心得。

全书使用 Delphi 7 附带的 ModelMaker 实现 UML 对象建模，并附有大量 Delphi 源代码实例，方便读者研究学习。

本书适用于有一定 Delphi 基础，并希望掌握面向对象编程思想和方法，进一步提升水平的软件开发人员。同样，已经掌握面向对象编程的 Java 和C++ 程序员通过本书亦能快速掌握 Delphi 编程。本书还适合大专院校用于基于 Object Pascal/Delphi 的面向对象编程教学。

图书在版编目（CIP）数据

Delphi 面向对象编程思想/刘艺著．－北京：机械工业出版社，2003.9
（Borland 核心技术丛书）
ISBN 7-111-12772-2

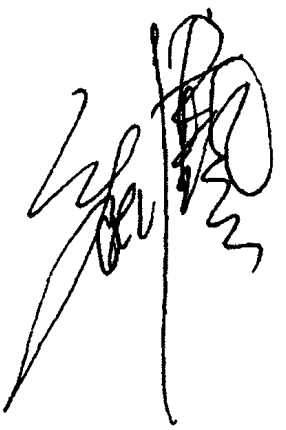
Ⅰ.D… Ⅱ.刘… Ⅲ.软件工具－程序设计 Ⅳ.TP311.56

中国版本图书馆 CIP 数据核字（2003）第 068440 号

机械工业出版社（北京市西城区百万庄大街 22 号 邮政编码 100037）
责任编辑：李云静
北京昌平奔腾印刷厂印刷·新华书店北京发行所发行
2003 年 9 月第 1 版第 1 次印刷
787mm×1092mm 1/16·30.75 印张
印数：0 001-4 000 册
定价：55.00 元（附光盘）

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线电话：（010）68326294

这是一本纯粹讨论 Delphi 面向对象编程的书，面向对象不是本书的时髦点缀，而是这本书的核心和全部。



作者简介：

刘艺，海军工程大学信息与电气学院副教授、美国Borland公司授予的Delphi产品专家、计算机技术作家。著有《Delphi 6企业级解决方案及应用剖析》、《Delphi第三方控件使用大全》等10多部计算机专著，出版重点大学计算机教材2部。其主持的多个科研项目在军内获奖。

作者个人网站：http://www.liu-yi.net

前　言

第一次知道 Delphi 并不是因为 Borland 公司的 Delphi 软件，而是在柏拉图的经典著作《柏拉图对话录》“申辩篇”[⊖]中读到了这个单词：

A friend of mine … went to Delphi and boldly asked the oracle to tell him whether … there was anyone wiser than I was, and the Pythian prophetess answered that there was no one wiser …

Delphi 和苏格拉底的智慧

Delphi 是雅典西北方一座美丽而神圣的小城，传说古时众神之父宙斯测量大地，而 Delphi 正好是世界的中央。《柏拉图对话录》记载了苏格拉底的一位朋友前往 Delphi 向预言女神 Pythian 询问谁是最智慧的人。Pythian 说没有人比苏格拉底更智慧。苏格拉底深感不解，因为他发现自己身边有很多政治家、诗人、哲学家和艺术家。难道这些“专家”和“权威”不是更有智慧吗？自己比起他们差多了。

苏格拉底一一拜访了这些“专家”和“权威”，却发现他们往往自以为是，自欺欺人，对自己不懂的东西也假装知晓。通过拜访，同时苏格拉底也发现自己在许多方面知之甚少。但苏格拉底并没有不懂装懂，他坦诚了自己的无知。这就是预言女神 Pythian 所说的真正的“智慧”。

其实这种智慧不是古希腊人最早提出来的。早在苏格拉底之前，我国的老子就已经总结过了。《道德经》中就有“知人者智，自知者明”，此之谓也。

现在，我们使用的 Delphi 已经是一个优秀的编程语言和软件开发工具了。然而，面对博大精深、不断发展的 Delphi，我们在许多方面还知之甚少。可是在学习和使用 Delphi 时，我们是否也具备了苏格拉底那种自知自明的态度和知所不知的智慧呢？

有许多选择 Delphi 的朋友，最初的想法可能是因为 Delphi 功能强大，易学易用。他们甚至 3 个月就声称精通了 Delphi，半年就敢独立开发软件。其实他们所能做的工作仅仅是控件的拖拉而已。当他们为自己的程序陶醉时，实际上是在为 Delphi 的精巧睿智和别人的控件所陶醉。这种 Delphi 程序员往往被别人戏称为“拖拉”员。而他们却俨然以 Delphi 高手自居。

也有许多放弃 Delphi 的朋友，最初的想法可能是因为觉得 Delphi 只是一个类似 VB 的 RAD 工具。在他们看来 Delphi 就是控件编程，无法像C++ 或 Java 那样真正实现 OOP。他们怀疑 Delphi 是不是具备面向对象编程语言的特性，能不能实现多态、模式等面向对象的技术。在他们眼里只有使用C++ 或 Java 的人才是真正的高手。

那么，什么是真正的高手，怎样才能成为一个高手呢？

⊖ 《The Dialogues of Plato》之“Apology”。

论“器”与“气”

首先让我们来看一看什么是武林高手，或许我们能够从中得到启发。

凡是喜爱武功或武侠小说的人都知道，修炼武功分为外练和内修两种途径。外练拳脚、兵器，拳脚的招式和兵器的好坏是关键；内修真气，练精化气、练气化神、练神还虚是根本。这两种修炼方法的最大差别在于时间和功力的函数关系上，如图 1 所示。通俗地讲，头 3 年，练外功的可以轻易打败练内功的；第 10 年，双方只能打个平手；15 年后，无论你如何练外功都不是练内功的对手。20 年后，内功高手天下无敌。其中的道理正是在于“器”和“气”的辩证关系。

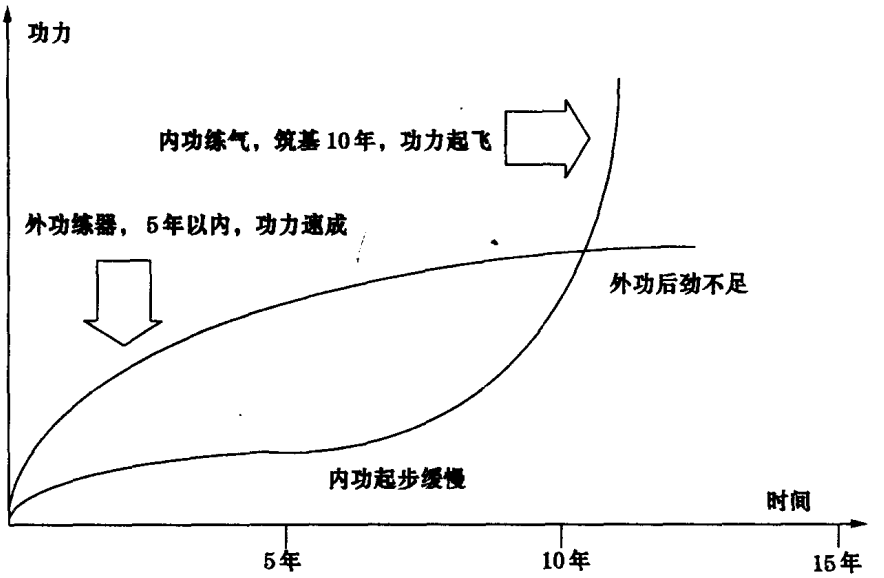


图 1　两种修炼方法的时间和功力函数图

“器”为有形之物，刀枪剑戟皆为有形之器。外练武功，离不开这些有形之器，练功即为练兵器。所以使枪弄剑的武林高手往往依赖于兵器的好坏。

“气”为无形之质，一切智慧、法力、思想皆为无形之质。内修气功，离不开这些无形之质，练功即为练气。所以真正的武林高手往往无需依赖于特定的兵器，对于他们而言，任何有形之物皆可随气所运，拟为兵器。一折扇，一拂尘在其手中威力不亚于名枪好剑，是得气所致也。

在软件开发中，编程工具是“器”，编程思想是“气”。

在编程之器中，易用之器如 VB、PB；难用之器如汇编、C ++ ；古老之器如 FORTRAN、COBOL；时尚之器如 Java、C # 。

在编程思想中，又有面向对象和面向过程之分，它们既是世界观又是方法论。前者反映的是人对客观对象的思维方式，后者反映的是机器对指令的思维方式。在软件开发的不断实践中，前者的优越性已经得到不断的体现和证实。

掌握面向对象的编程思想如同获得练气的真谛。它的重要性往往胜过了对编程语言的选择!

有人即使选择了面向对象的利器，也无法成为真正的高手，因为他看重的是“器”的好坏，忽略了“气”的修炼。

实际上“一个系统或语言是不是面向对象的并不重要，重要的是怎样才能是面向对象的以及用什么办法实现相关的好处”（《面向对象方法：原理与实践》机械工业出版社 2003 年 3 月出版）。

练器虽易，但难成高手。练气虽好，但见效缓慢，寂寞难耐，非一般常人可以明心见性，直取大道。所以很多武林高手都是先练器、后练气；内外兼修，终成正果。

对于初入武林的新手，他们需要借助兵器的威力，以补内力之不足，器的好坏往往很重要。但随着武功的增加，内力的勃发，对器的依赖性应该减小。《神雕侠侣》中杨过练剑，起步初学时好使利剑，谙悉剑法后喜用钝剑，内功纯熟后树枝亦当剑。所以对于真正的高手而言，剑器的好坏往往并不重要。

同样，软件高手的成长也有这样的过程。初学编程需要选择好的语言，这样可以取得事半功倍的效果，同时激发学习兴趣，增强信心。一旦熟悉了一种语言之后，应以此为契机进而掌握面向对象编程思想。这时你熟悉的不再是语言本身的语法、函数、类库，而是绑定、多态、模式等思维方法，然后触类旁通，再学其他面向对象语言也不难。苦练内功，勇于实践，最终成功的真正的软件高手，是不受编程语言限制的。他们可能比较熟悉一种开发工具，但那也只是承载他们大道无形之气的器具。他们虚心好学，善于总结。他们的思想、方法、模式甚至哲学，既超越了编程语言，又可以指导编程语言的实践。

Delphi 为软件高手的成长提供了内外兼修的捷径。学练 Delphi，既可用其 RAD 之长，控件之利，初学起步，迅速击败对手；学练 Delphi，也能以其 OOP 之能，VCL 之强，培根固本，成就不败之功。

威震四海的华山剑派曾分为“剑宗”和“气宗”，前者只练器，讲招式；后者兼练气，重筑基。学习 Delphi 好比修炼华山剑法，走 RAD 之路是“剑宗”，从 OOP 之道是“气宗”。前者喜用控件，看中奇技淫巧；后者好为对象，热衷方法模式。前者追求速成，后者志存高远。

我认为，无论是为 RAD 而选择 Delphi 还是因 OOP 而放弃 Delphi 的朋友都没有真正了解 Delphi。Delphi 是一个不错的 RAD 开发软件，可是不学 OOP，不深入 VCL 就很难成为真正的高手。同样，Delphi 是一个地道的 OOP 编程工具，结合 Delphi 强大的 RAD 和高效的编译器，可以比其他 OOP 语言有更多的优势和更高的效率。如果能打破门户之见，“器”与“气”同练，内外兼修，我相信 Delphi 程序员不难从一个 RAD 快手成长为 OOP 高手，最终笑傲江湖，纵横四海。

面向对象编程思想和大道无形之气

前面我简单讨论了“器”与“气”的辩证关系。在编程中，修持内功、提高内力的关键之一在于掌握面向对象编程思想。实际上，我更认为面向对象编程思想才最合大道无形之真气的

妙处。

为什么这么讲？“古之大化者，乃与无形俱生”（《鬼谷子》反应第二），气的奥妙首先在于它的“大化”。大化者，天地之大造化也，集一切创造和变化之能。面向对象的编程思想具备了这种特质。

老子曰“无名天地之始，有名万物之母”，无名是无以名状，无法定义的意思。所谓面向对象的思维方式，最奥妙之处在于如何从“无名”中识别和定义对象，如何从“有名”中构造和使用对象。

对于软件开发人员来说，认识客观实体的过程、对用户需求进行分析和设计的过程，就是发现和界定对象的过程，是从无名到有名的过程。然而这里的对象又不同于面向过程中的变量或函数，对象是由类创建的，类是概念的抽象，是可以定义的“有名”，是对象之母。

于是乎，太极生两仪，两仪生八卦，通过类的继承和派生，万物始生，系统构成。

即使作为面向对象编程工具的“器”，也体现和承载了面向对象的编程思想之“气”。

号称“万古丹经王”的内功练气经典之作《周易参同契》开篇第一句话就是“乾坤者，易之门户，众卦之父母。坎离匡郭，运毂正轴，牝牡四卦，以为橐籥。”

从软件开发的角度来理解，面向对象的编程工具虽然提供了构建无穷种软件系统的可能性，但这种无限性却是建立在自身类库的有限框架之中。无论是 Delphi 的 VCL，还是 Java 的类库，或是 .NET 框架，无一不是建立在这样一个类似于周易八卦的架构之中。“易有太极，太极生两仪，两仪生四象，四象生八卦”，这样的结构完美无瑕，有着无穷的创造力。

太极是 Delphi 中的 TObject，它是构建系统的原子，并是所有类的祖先，它具有所有类的基本特征。在 Delphi 的编程世界中，根类 TObject 生持久对象类 TPersistent，持久对象类 TPersistent 生组件对象类 TComponent，进而为开发应用程序提供了丰富的控件和强大的功能。

然而类库的结构框架不仅仅是给我们可以作为“器”用的组件，更重要的是这种结构通过类之间的关系和相关作用，实现了“气”的构造和变化，体现出面向对象编程思想的精髓。为我们创建自己的系统提供了绝好的示范。

气的奥妙其次在于它“生于无形”。无形意味着它的自由性、开放性、适应性。在面向对象编程思想中处处充满着“气”的这种大道无形的智慧。

比如，面向编程对象中的多态性使程序员可以撰写更加通用的、更加开放的程序。程序员可以为 Vehicle 对象编写一个纯虚抽象方法 stop ()，这样的通用 stop ()方法与驾驶什么车无关。程序员可以让派生类去操心如何完成 stop ()方法，而继续在更高的抽象级别上编写自己的通用过程。即使 Car 对象的 stop ()方法与 Bicycle 对象的 stop ()方法完全不一样，程序员也可以使用 Vehicle.stop ()。多态性可以让创建的对象自动知道哪一个合适的方法将被调用。这就使程序具备了“气”的开放性和适应性。

练气功中讲究“上德无为，不以察求。下德为之，其用不休”（引自《周易参同契》）。

在面向对象编程思想中，“上德”是晚绑定的纯虚抽象方法，是以不变应万变的对象接口，它是对事物的高度抽象，是形而上学。“上德无为”是说在抽象层次，通过无为来体现编程“虚”的一面，因为这时还无法确定实际使用的真正对象（可能是 Car 对象也可能是 Bicycle 对

象，更可能是以后发明的新交通工具对象），“不以察求”要求我们跳出具体需求的约束，不去考虑具体的实现代码。所以纯虚抽象方法或对象接口中是没有任何代码实现的。

在面向对象编程思想中，“下德”是对纯虚抽象方法的覆盖，是对对象接口的实现。“下德为之”，提供了真正的代码实现。“其用不休”，满足了不断变化的需求。

多态性使得程序员在以后不费多大力气就可以派生对象，实现程序。假设程序员在为 Car 和 Bicycle 构建应用程序，并不知道还存在 Truck，但这并不要紧。程序员可以为继承 Vehicle 类的 Car 和 Bicycle 类撰写覆盖 stop ()的方法。这样，在程序中只要把创建的 Car 和 Bicycle 对象转型为 Vehicle 的类型，使用 Vehicle 的 stop ()方法，就可以让 Car 和 Bicycle 对象动态绑定符合自己要求的 stop ()方法。即使后来新增了 Truck 对象，仍然是调用 Vehicle 的 stop ()方法，并不需要改动程序。

“物有自然，事有合离。有近而不可见，有远而可知。近而不可见者，不察其辞也；远而可知者，反往以验来也。”（《鬼谷子》抵戏第四）

虽然客观事物复杂，用户需求是变化的，但是其中也有一定的内在规律。

近而不可见者，只看眼前的具体功能实现，不察事物的一般发展规律，心中只有孤立的数据和机械的流程，一旦“事有合离”，则措手不及，难以应对。这样的编程是静态的、机械的、难以维护和扩展的。

远而可知者，善于发现规律，重视代码重用，眼中尽是有机的对象和和谐的关系，即使需求改变，也能从容应对，游刃有余。这样的编程是动态的、灵活的、可维护和可扩展的。

Paul Kimmel 在《Delphi 6 应用开发指南》中说“以非面向对象的方法去使用面向对象工具是一个错误。使用 Delphi 编写结构化程序可以很快地到达 beta 版……你的程序可能永远脱离不了 beta 版。迅速得到错误的答案，仍然是错误的。”

同样是使用 Delphi，如果没有面向对象的编程思想，好比“不察其辞”，最终仍然是“近而不可见”，难以开发出优秀的系统。惟有潜心苦练，悉心总结，掌握面向对象编程思想的精髓，才能运“气”自如，“反往以验来也”，最终达到“远而可知”的境界。

关于本书

从第一个真正面向对象的语言 Smalltalk（1972 年）出现至今已经有 30 多年的历史了。然而书店中充斥着的面向对象编程的书籍大都是C ++ 和 Java，似乎面向对象的语言仅有这两种，而实际上真正的面向对象语言却有 4 个基本分支，近 20 种之多。由于 Delphi 面向对象编程的书籍很少，不少程序员为了学习 OOP，不得不放弃 Delphi。这真是 Delphi 的一大悲哀。当我读到 Bruce Eckel 的《Thinking in Java》，就感叹过为什么就没有这样的 Delphi 大作呢？

其实，Delphi 系出名门，它是 Borland 公司在 Object Pascal 基础上开发的。现在，Borland 公司从 Delphi 7 开始使用 Delphi 语言来取代 Object Pascal 叫法[⊖]。Delphi 在 OOP 方面实际上并不比 C ++ 和 Java 逊色，这一点读者可以参见本书附录 B “面向对象编程语言比较：Java、C ++ 和

⊖ 参见 Delphi 帮助文件 “Delphi Language Reference”。在本书中笔者也将 Object Pascal 不加区别地称为 Delphi。

Delphi”。

为此，我一直打算写一本 Delphi 面向对象编程的书，总结自己在 Delphi 面向对象编程方面的学习体会和实践经验。然而这是一项难度不小的工作，全书从构思到下笔花费了很长的时间，直到今年 6 月才算正式完稿。刚巧今年也是 Borland 创建 20 周年，作为 Borland 产品 Delphi 的用户，拙作的出版也算是对此的纪念。

这是一本纯粹讨论 Delphi 面向对象编程的书，面向对象不是本书的时髦点缀，而是这本书的核心和全部。

本书自第 1 章“建立面向对象的新思维”开始就试图从面向对象编程的历史和现状入手，阐述面向对象编程思想的起源发展和基本观念，以及面向对象建模方法和 UML 的应用。这一章是为了帮助读者建立面向对象的基本概念，了解面向对象的思维方法。

第 2 章“Delphi 对象模型”介绍了 Delphi 面向对象编程的基础知识及其对象模型结构体系。

第 3 章“理解对象”从对象的本质、生死、关系三方面深入讨论了对象的内部机制、生命周期、相互作用，为读者了解和掌握对象打下了基础。

第 4 章“使用对象”讲解了在 Delphi 面向对象编程中如何高效使用对象。这里重点讨论了界面对象、组件对象、对象集和对象参数的使用方法和技巧，并对 VCL 组件使用和开发中的常见问题进行了深入思考。

第 5 章“深入多态”介绍了多态的概念及其在编程中的应用。其中通过大量的实例讲解了重载和覆盖、虚方法与动态方法、抽象类和抽象方法、类的类型转换等重要概念和思维方法。

第 6 章“剖析接口”全面介绍了对象接口的编程知识和应用技巧。阐述了接口在实现动态绑定和多重继承方面的重要作用，演示了接口在面向对象编程中的实际用法。

第 7 章“研究封装”阐明了封装在面向对象编程中的重要意义和应用原则，并分别从逻辑上的封装和物理上的封装来进一步讨论封装的实现方法和应用技巧。

第 8 章“实现界面和业务的分离”将面向对象编程应用到一个新的高度。这一章通过界面和业务分离的演化实例，讲解了如何利用面向对象的设计将一个桌面程序进化到分布式多层系统。并结合 Delphi 的最新 Web 技术，介绍了如何用 Web Service 封装业务对象，用 Web Form 封装界面对象，用新技术封装旧对象，从而实现跨平台的应用。

最后本书第 9 章和第 10 章“深入浅出 VCL”，研究了 VCL 的内部机制，并剖析了 VCL 重要类系的对象用法，为渴望深入提高编程水平的读者提供了参考。

从本书的组成结构上看可以划分成五大部分。

第一部分，全书的前两章是 Delphi 面向对象编程的入门。已经掌握面向对象基本概念并有 Delphi 编程经验的读者可以跳过这两章。

第二部分，第 3、4 章是 Delphi 面向对象编程的关键。不掌握对象的实质，就无法使用好对象。

第三部分，第 5、6 章是 Delphi 面向对象编程的深入。面向对象的高级技巧无一不是建立在虚方法、抽象方法、对象接口等动态绑定机制上和向上转型、向下转型、接口转型等类型转换机制上的。

第四部分，第 7、8 章是 Delphi 面向对象编程的应用。为了实现程序的可维护性、可扩展性和可重用性，封装已经成为面向对象编程的重要思想之一。通过封装从而实现界面和业务对象的分离，从界面和业务分离逐步实现分布式多层体系结构，进而实现界面和业务应用的跨平台。这里演示了基于面向对象编程思想的从一般应用程序到企业级应用程序的解决方案。

第五部分，最后的第 9 章和第 10 章是 Delphi 面向对象编程的参考。熟悉 VCL、学习 VCL 对精通 Delphi 十分有帮助。鉴于目前 VCL 内幕资料的缺乏，因而这一部分所提供的内容可能比较有限，但却是很难得的。

准确地讲，这本书不是写给“高手”的，而是写给那些想从 RAD 向 OOP 转变的程序员的，以及希望通过 Delphi 来学习 OOP 的朋友的。我觉得作为一本比较实用的中级 Delphi 技术书，比较合适。所以在全书的行文中，力求通俗易懂，图文并茂，并精心编写了大量的示例程序（随书光盘源代码超过 50MB），供读者研习。这本书的核心是 OOP，而不是针对 Delphi 的所有方面。阅读本书需要有一定的 Delphi 基础，书中涉及到一些专门的知识（如：COM + 等），还需读者进一步参阅相关书籍。

可能会有一些“高手”对本书失望。我觉得自己不适合写“高手”阅读的书，因为我就不是高手，我觉得自己永远是新手。和其他新手不同的是，我使用 Delphi 的时间更长一点，经验和阅历稍多一点。所以，若发现本书有不妥之处敬请读者指正，不尽如人意之处希望多多包涵。

网友 xzh2000 讲得好：“一本书的生命很重要，如果作者能花心血经常修改补充，才能成就经典!”的确一本好书需要经过多次反复修订才能成为经典之作，所以我愿意聆听所有读者的宝贵建议，并希望这本书能够不断修订再版。

光盘内容

本书光盘包含了书中绝大多数示例代码。

光盘内容按章编排。其运行环境为 Windows 98 以上的 Windows 操作系统，完全安装需要不少于 70MB 的硬盘空间。

光盘提供了完整的 Delphi 项目文件，可用 Delphi 直接打开。所有项目文件和源程序均在 Delphi 7 上调试通过。

致谢

其实要想获得 Delphi 预言女神 Pythian 所说的“智慧”并不是一件简单的事。随着我们不断地学习，不断地获得经验，我们的感觉通常是自己知道的越来越多。但实际上，因为我们知识水平的提高和认识能力的增加，反而更能发现自己不知道的领域，意识到自己的无知。

每当拙作完稿之际，我总是发现自己仅仅写了一些皮毛。虽然通过自己的体会和经验解决了一些问题，但又有更多的问题涌现，有待研究。于是自己像一个新手一样又要重新开始学习。

当然，一本书的写作离不开众人的智慧。在这本书的写作中我参考了大量的资料，包括

Delphi 以外的很多资料。在书后的参考文献中我列出了主要的参考资料，并向这些资料的作者表示敬意。

在此特别要感谢的是胡冰乐先生，他无私地将自己撰写和整理的 VCL 相关资料提供给我参考。

参加本书资料整理、审稿校对、文字录入工作的人员有：段立、李启元、罗勇、吴苗、王剑、刘卫华、尹亚兰、洪蕾、吴英等。为本书撰写提供其他帮助的还有：王永斌、周安栋、屈晓旭、曹旭峰，在此一并表示感谢。

最后感谢 Borland 北京代表处王玉红女士的大力支持。

刘　艺

http://www.liu-yi.net

E-mail:my_reader@sina.com

2003 年 6 月 12 日于南京

目　录

第 1 章　建立面向对象的新思维

1.1　导论

1.1.1　历史背景

软件开发的过程就是人们使用各种计算机语言将自身关心的现实世界映射到计算机世界的过程。

数字计算机的先驱——第一台加法机，是 1642 年由法国科学家、数学家兼哲学家布莱斯·帕斯卡（Blaise Pascal）设计的。这个装置使用了一系列有 10 个齿的轮子，每个齿代表从 0 到 9 的一个数字。轮子互相连接，从而通过按照正确的齿数向前移动轮子，就可以将数字彼此相加。后来 Pascal 这个伟大的名字被用来为一种广泛使用的计算机语言命名，Pascal 语言经过不断发展，注入了面向对象、可视化、RAD（Rapid Application Devolopment）等最具活力的要素，就变成了现在我们所用的 Delphi。

1．开端

当代计算机的数学理论基础是由计算机的开山鼻祖——大名鼎鼎的图灵于 1937 年提出的图灵机模型。随后不到 10 年，电子计算机就诞生了（1945）。第一台电子计算机 ENIAC（见图 1-1）含有 18 000 个真空管，具有每分钟几百次的运算速度，但是最初程序是通过导线传送到处理器内的，必须由人工更改。它当时的主要任务之一就是用于导弹弹道轨迹的计算。当时的软件开发（如果可以称之为软件开发的话）与现在的大不相同。为了算一道题，要有人事先把完成加减乘除等各类运算的部件像搭积木那样搭起来，如果换一道题，则要把这些部件分解开来，再根据新的要求重新搭建，与现在相比效率极低。

图 1-1　第一台电子计算机 ENIAC