软·件·工·程·师·典·藏

超量代码·实用工具

Visual Basic 技术方案宝典

■ 明日科技 刘彬彬 高春艳 安剑 编著

人民邮电出版社
POSTS & TELECOM PRESS

Visual Basic 技术方案宝典

■ 明日科技 刘彬彬 高春艳 安剑 编著

人民邮电出版社
北京

图书在版编目（CIP）数据

Visual Basic 技术方案宝典 / 刘彬彬，高春艳，安剑编著. —北京：人民邮电出版社，2008.7
（软件工程师典藏）
ISBN 978-7-115-18042-1

Ⅰ. V… Ⅱ. ①刘…②高…③安… Ⅲ. BASIC 语言—程序设计 Ⅳ. TP312

中国版本图书馆 CIP 数据核字（2008）第 061849 号

内 容 提 要

本书从用户在利用 Visual Basic 进行软件开发中必须掌握的核心技术入手，通过各种实用方案深入介绍各种核心技术在实际开发中的应用。全书分为 10 章，分别介绍了模式与程序模块化、用户界面设计、数据库操作技术、数据查询技术、决策分析、报表打印技术、安全策略、邮件发送、打包发行、帮助文件等。通过本书，读者不但可以学习相关技术的各种核心应用，更能触类旁通，学以致用，掌握 Visual Basic 应用开发的精髓。

本书附有配套光盘。光盘提供了书中所有方案实例的源代码，所有实例都经过精心调试，在 Windows XP/2003 下测试通过，保证能够正常运行。

本书内容翔实，突出技术功能属性，具有非常强的实用性。本书适合于各级软件开发人员学习使用，也可供大、中专院校师生学习参考用书。

软件工程师典藏

Visual Basic 技术方案宝典

◆ 编　　著　明日科技　刘彬彬　高春艳　安　剑
责任编辑　屈艳莲

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
三河市海波印务有限公司印刷

◆ 开本：787×1092　1/16
印张：40.75
字数：1 111 千字　　2008 年 7 月第 1 版
印数：1–4 000 册　　2008 年 7 月河北第 1 次印刷

ISBN 978-7-115-18042-1/TP

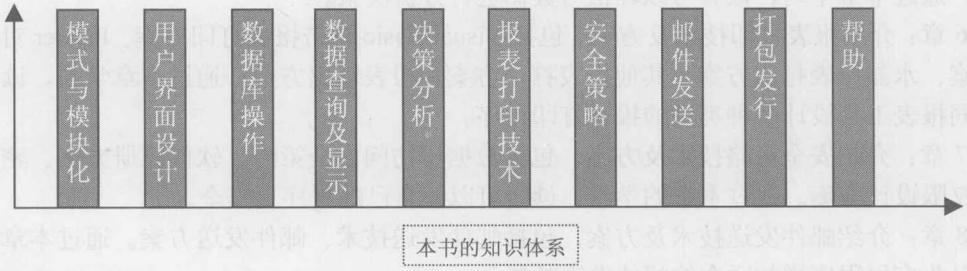
定价：75.00 元（附光盘）

读者服务热线：(010)67132692　印装质量热线：(010)67129223
反盗版热线：(010)67171154

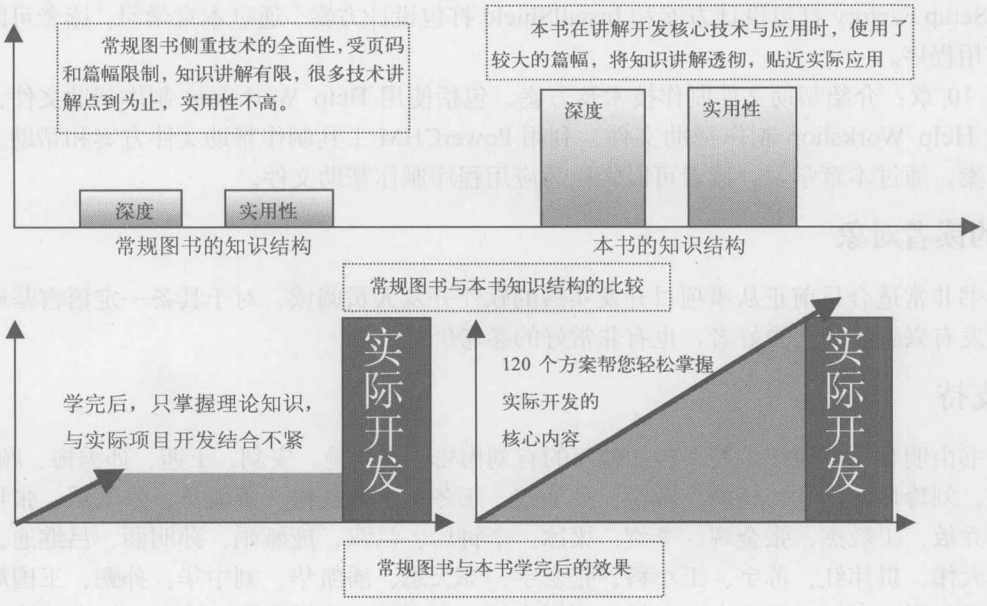
前 言

Visual Basic 是 Microsoft 公司推出的采用事件驱动方式的可视化编程语言。因为具有开发速度快，语法简单易学，开发环境简捷灵活的特点，深受广大开发人员的喜欢，已成为使用最广泛的程序开发工具之一。无论是开发中小型数据库，还是开发多媒体软件、数据采集控制、网络应用程序，Visual Basic 都是快速完成开发任务的最佳开发工具之一。

目前，虽然介绍 Visual Basic 的图书数不胜数，但绝大多数是为初级用户介绍基础知识的入门书籍。真正从应用角度讲解 Visual Basic 实际开发与应用的图书则珍稀难觅。本书从实际开发必须掌握的核心技术出发，将实际开发中与该技术相关的应用整理成方案，通过各种实际开发方案讲透各种技术的具体应用，真正授人以渔，满足实际开发者的需求。读者学完本书后，可以顺利进行实际项目的开发。



下面列出了常规图书和本套图书在知识结构和学习效果上的比较。



本书内容

本书主要从开发者必须掌握的核心技术入手，深入探讨开发中必须掌握的技术和技术的各

种应用方案，使读者不但可以了解知识的来龙去脉，更能学会利用学到的知识进行实际开发。本书共分 10 章，各章主要内容简介如下。

第 1 章：介绍开发模式及相关方案，主要包括设计模式、窗体模式设计方案、程序模块化设计方案等。学好本章，可以选择合适的开发模式，从而使程序设计更合理，代码更清晰，更容易维护。

第 2 章：介绍界面设计技术及相关方案，包括用户界面设计技术、主界面设计方案、用户操作界面设计方案、信息提示界面设计方案和界面美化方案。通过本章学习，读者可以透彻掌握界面的相关技能，设计界面友好的应用程序。

第 3 章：介绍数据库操作技术及方案，包括数据库开发技术、数据库访问方案、ADO 操作数据库方案、SQL 语句操作数据库方案、二进制数据处理方案、数据库维护方案和数据库转换方案。通过本章的学习，读者可以轻松实现对数据库的各种操作。

第 4 章：介绍数据查询、显示技术及方案。包括数据查询技术、简单查询方案、中级查询方案、高级查询方案和数据显示方案。通过本章学习，读者可以非常容易地处理数据库查询与显示问题，并能设计较复杂的查询。

第 5 章：介绍决策分析技术及方案。包括图表分析方案、透视表分析方案和透视图决策分析方案。通过本章学习，读者可以轻松对数据进行分析决策。

第 6 章：介绍报表打印技术及方案。包括 Visual Basic 自带报表打印方案、Printer 对象编程打印方案、水晶报表打印方案、其他报表打印方案和报表输出方案。通过本章学习，读者可以通过不同报表工具设计各种类型的报表打印程序。

第 7 章：介绍安全策略技术及方案，包括数据库访问安全策略、软件注册方案、密码验证和用户权限设计方案。通过本章的学习，读者可以使自己的程序更安全。

第 8 章：介绍邮件发送技术及方案。包括邮件发送技术、邮件发送方案。通过本章学习，读者可以为应用程序增加适合的邮件发送功能。

第 9 章：介绍程序打包技术及方案。包括打包设计分析、Visual Basic 自带的打包工具打包方案、Setup Factory 打包设计方案和 InstallShield 打包设计方案。通过本章学习，读者可以轻松打包应用程序。

第 10 章：介绍帮助文件制作技术及方案。包括使用 Help Workshop 制作帮助文件、使用 HTML Help Workshop 制作帮助文件、利用 PowerCHM 工具制作帮助文件方案和帮助文件的调用方案。通过本章学习，读者可以轻松为应用程序制作帮助文件。

本书的读者对象

本书非常适合目前正从事项目开发实践的软件开发人员阅读，对于具备一定语言基础，对程序开发有兴趣的编程爱好者，也有非常好的参考价值。

技术支持

本书由明日科技组织编写，参加编写的有刘彬彬、高春艳、安剑、王茜、孙秀梅、顾彦玲、梁晓岚、刘玲玲、刘欣、杨丽、黄锐、孙明姣、王冬雪、寇长梅、董大勇、张鹏斌、张艳、郭佳博、乔敏、王敬杰、张金辉、李贺、梁冰、李钟尉、吕双、庞娅娟、孙明丽、吕继迪、张跃廷、房大伟、贯伟红、苏宇、王小科、张宏宇、邹天思、潘凯华、刘中华、孙鹏、王国辉、马文强、尹相群、王毅、王殊宇、宋坤、刘锐宁等。由于编写水平有限，疏漏之处在所难免，请广大读者批评指正。

如果读者在使用本书时遇到问题，我们将通过明日科技网站为读者提供网上服务和支持。

读者使用本书遇到的问题，我们承诺在 1～6 个工作日给您提供及时回复。

服务网站：www.mingrisoft.com　　服务信箱：mingrisoft@mingrisoft.com

服务电话：0431-84978981/84978982

明日科技

2008 年 5 月

目 录

第 1 章

模式与程序模块化

- 设计模式概述
- 设计模式
- 窗体模式设计方案
- 程序模块化设计方案

模式和程序模块化是程序设计中比较重要的部分，在程序设计时，如果采用适当的模式就可以为程序带来模块化的效果。如果程序都是利用高内聚、低耦合的程序模块组成，对于以后的维护和移植都会带来很大的方便。因此在程序设计时，应选择正确的模式，并力求程序模块化。

1.1 设计模式概述

设计模式在设计者中是一种流行的思考设计问题的方法，是一套被反复使用、多数人知晓的、经过分类编目的、代码设计经验的总结。本节将对设计模式进行一个简单的介绍，使读者对设计模式有一个初步的认识。

1.1.1 统一建模语言（UML）

UML（Unified Modeling Language）是统一建模语言，是一种标准的图形化（或可视化）建模语言。它是一种标准的表示方法，而不是一种完整的方法学。人们可以以任何一种形式使用 UML，但是无论何种形式，其基础都是 UML 图。

UML 用面向对象的图形方法来描述任何类型的系统，它的应用领域很宽，最为常见的是建立软件系统的模型，当然也可以描述计算机软件以外的任何系统。如机械、商业系统等。总之，UML 是一个通用的标准建模语言，可以为任何具有静态或动态行为的系统建立模型。

UML 适用于以面向对象方法来描述任何类型的系统，适合于系统开发的全过程。

1.1.2 Visual Basic 中的 OOP

面向对象编程（Object Oriented Programming，OOP，面向对象程序设计）是一种计算机编程架构。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的。OOP 达到了软件工程的 3 个主要目标：重用性、灵活性和扩展性。为了实现整体运算，每个对象都能够接收信息、处理数据和向其他对象发送信息。OOP 主要有以下特性。

（1）封装（Encapsulation）

也叫做信息封装：确保组件不会以不可预期的方式改变其他组件的内部状态；只有在那些提供了内部状态改变方法的组件中，才可以访问其内部状态。每类组件都提供了一个与其他组件联系的接口，并规定了其他组件进行调用的方法。

（2）多态性（Polymorphism）

许多不同的对象有相同的名字和相同的方法。如，某一方法可以完成同样的功能，但是实现的方法不同。或者，方法的名称相同，但是参数不同。在 Visual Basic 中，一个类中不允许存在多个同名的方法，即使是参数不同。

（3）继承性（Inheritance）

继承是指可以从其他对象中继承属性和方法，这样可以从一些简单的对象开始构造更加复杂的程序。在 Visual Basic 中，仅仅支持继承性的部分特征，只能通过接口来体现继承。

1.1.3 设计模式

设计模式是一种高级软件重用技术，是一套被反复使用、多数人知晓、经过分类编目的、大量设计经验的总结，是在软件开发过程中，在特定的环境下解决问题的方法。使用设计模式可以更加简便地重用成功的设计和体系结构，使代码编制真正的工程化，设计模式是软件工程的基石，是不同对象之间交互的定义和分类。

1.1.4 设计模式基本要素

设计模式使人们可以更加简单方便地复用成功的设计和体系结构。将已证实的技术表述成设计模式也会使新系统开发者更加容易理解其设计思路。设计模式的基本要素为模式名称、问题、解决方案和效果。

- 模式名称

一个助记名称，用来描述设计模式、解决方案和效果。设计模式允许在较高的抽象层次上进行设计。基于一个模式词汇表，开发团队之间可以讨论模式并在编写文档时使用它们。模式名称可以帮助开发人员思考，便于团队成员交流设计思想及设计结果。找到合适的模式名称也是设计模式编目工作的难点之一。

- 问题

问题主要描述在何时使用设计模式。解释了设计问题和问题存在的前因后果，描述特定的设计问题，怎样用对象表示算法等。通常情况下，模式必须满足的一系列先决条件的问题。

- 解决方案

解决方案描述了设计的组成成分，它们之间的相互关系及各自的职责和协作方式。因为模式就像一个模板，可应用于多种不同场合，所以解决方案并不描述一个特定具体的设计或者是实现，而是提供设计问题的抽象描述和怎样用一个具有一般意义的元素组合（类或对象组合）来解决这个问题。

- 效果

效果描述了模式应用的效果及使用模式应权衡的问题。尽管描述设计决策时，并不是总提到模式效果，但它们对于评价设计选择和理解使用模式的代价及优势具有重要意义。软件效果大多关注对时间和空间的衡量，它们也表述了语言和实现问题。因为复用是面向对象设计的要素之一，所以模式效果包括它对系统的灵活性、扩充性或可移植性的影响，显式地列出这些效果对理解和评价这些模式很有帮助。

1.1.5 基本的设计模式

目前比较流行的设计模式有 23 种，下面简单对这 23 种设计模式进行概述。

- 抽象工厂（Abstract Factory）模式

抽象工厂模式提供一个创建一系列相关或相互依赖对象的接口，而无需指定它们具体的类。

- 适配器（Adapter）模式

把一个类的接口变换成客户端所期待的另一种接口，从而使原本因接口原因不匹配而无法一起工作的两个类能够一起工作。适配类可以根据参数返还一个合适的实例给客户端。

- 桥（Bridge）模式

桥模式将抽象化与实现分离，使得二者可以独立的变化，即将它们之间的强关联变成弱关联，也就是指在一个软件系统的抽象化和实现化之间使用组合/聚合关系而不是继承关系，从而使两者可以独立的变化。

- 建造（Builder）模式

将产品的内部表象和产品的生成过程分割开来，从而使一个建造过程生成具有不同的内部表象的产品对象。建造模式使得产品内部表象可以独立的变化，客户不必知道产品内部组成的细节。建造模式可以强制实行一种分步骤进行的建造过程。

- 响应链（Chain of Responsibility）模式

在响应链模式中，很多对象由每一个对象对其下家的引用而接起来形成一条链。请求在这个链上传递，直到链上的某一个对象决定处理此请求。客户并不知道链上的哪一个对象最终处理这个请求，系统可以在不影响客户端的情况下动态的重新组织链和分配责任。处理者有两个

选择：承担责任或者把责任推给下家。一个请求可以最终不被任何接收端对象所接受。

● 命令（Command）模式

命令模式把一个请求或者操作封装到一个对象中。命令模式把发出命令的责任和执行命令的责任分割开，委派给不同的对象。命令模式允许请求的一方和发送的一方独立开来，使得请求的一方不必知道接收请求一方的接口，更不必知道请求是怎么被接收，以及操作是否执行，何时被执行以及是怎么被执行的。系统支持命令的撤消。

● 组合（Composite）模式

组合模式将对象组织到树结构中，可以用来描述整体与部分的关系。组合模式就是一个处理对象的树结构的模式。组合模式把部分与整体的关系用树结构表示出来。组合模式使得客户端把一个个单独的成分对象和由它们复合而成的组合对象同等看待。

● 装饰（Decorator）模式

装饰模式以对客户端透明的方式扩展对象的功能，是继承关系的一个替代方案，提供比继承更多的灵活性。动态地给一个对象增加功能，这些功能可以再动态地撤消。增加由一些基本功能的排列组合而产生的非常大量的功能。

● 伪（Facade）模式

为子系统中的一组接口提供一个一致的界面，伪模式定义了一个高层接口，这个接口使得这一子系统更加容易使用。

● 工厂方法（Factory Method）模式

核心工厂类不再负责所有产品的创建，而是将具体创建的工作交给子类去做，成为一个抽象工厂角色，仅负责给出具体工厂类必须实现的接口，而不接触哪一个产品类应当被实例化这种细节。

● 轻量（Flyweight）模式

轻量模式以共享的方式高效的支持大量的细粒度对象。

● 解释器（Interpreter）模式

给定一个语言后，解释器模式可以定义出其文法的一种表示，并同时提供一个解释器。客户端可以使用这个解释器来解释这个语言中的句子。解释器模式将描述怎样在有了一个简单的文法后，使用模式设计解释这些语句。在解释器模式里面提到的语言是指任何解释器对象能够解释的任何组合。在解释器模式中需要定义一个代表文法的命令类的等级结构，也就是一系列的组合规则。每一个命令对象都有一个解释方法，代表对命令对象的解释。命令对象的等级结构中对象的任何排列组合都是一种语言。

● 迭代（Iterator）模式

迭代子模式可以顺序访问一个聚集中的元素而不必暴露聚集的内部表象。多个对象聚在一起形成的总体称之为聚集，聚集对象是能够包容一组对象的容器对象。迭代子模式将迭代逻辑封装到一个独立的子对象中，从而与聚集本身隔开。迭代子模式简化了聚集的界面。每一个聚集对象都可以有一个或一个以上的迭代子对象，每一个迭代子对象的迭代状态可以是彼此独立的。迭代算法可以独立于聚集角色变化。

● 调停者（Mediator）模式

调停者模式包装了一系列对象相互作用的方式，使得这些对象不必相互明显作用，从而使它们可以松散偶合。当某些对象之间的作用发生改变时，不会立即影响其他的一些对象之间的作用，保证这些作用可以彼此独立的变化。调停者模式将多对多的相互作用转化为一对多的相互作用。调停者模式将对象的行为和协作抽象化，把对象在小尺度的行为上与其他对象的相互作用分开处理。

● 记事（Memento）模式

备忘录对象是一个用来存储另外一个对象内部状态快照的对象。备忘录模式的用意是在不破坏封装的条件下，将一个对象的状态捉住，并外部化，存储起来，从而可以在将来合适的时候把这个对象还原到存储起来的状态。

- 观察者（Observer）模式

观察者模式定义了一种一对多的依赖关系，让多个观察者对象同时监听某一个主题对象。这个主题对象在状态上发生变化时，会通知所有观察者对象，使它们能够自动更新自己。

- 原型（Prototype）模式

用原型实例指定创建对象的种类，并且通过拷贝这个原型来创建新的对象。

- 代理（Proxy）模式

代理模式给某一个对象提供一个代理对象，并由代理对象控制对源对象的引用。代理就是一个人或一个机构代表另一个人或者一个机构采取行动。某些情况下，客户不想或者不能够直接引用一个对象，代理对象可以在客户和目标对象之间起到中介的作用。客户端分辨不出代理主题对象与真实主题对象。代理模式可以并不知道真正的被代理对象，而仅仅持有一个被代理对象的接口，这时候代理对象不能够创建被代理对象，被代理对象必须有系统的其他角色代为创建并传入。

- 单一类（Singleton）模式

保证一个类仅有一个实例，并提供一个访问它的全局访问点。

- 状态（State）模式

状态模式允许一个对象在其内部状态改变的时候改变行为。这个对象看上去像是改变了它的类一样。状态模式把所研究的对象的行为包装在不同的状态对象里，每一个状态对象都属于一个抽象状态类的一个子类。状态模式的意图是让一个对象在其内部状态改变的时候，其行为也随之改变。状态模式需要对每一个系统可能取得的状态创立一个状态类的子类。当系统的状态变化时，系统便改变所选的子类。

- 策略（Strategy）模式

策略模式针对一组算法，将每一个算法封装到具有共同接口的独立的类中，从而使得它们可以相互替换。策略模式使算法可以在不影响客户端的情况下发生变化。策略模式把行为和环境分开。环境类负责维持和查询行为类，各种算法在具体的策略类中提供。由于算法和环境独立开来，算法的增减、修改都不会影响到环境和客户端。

- 模板方法（Template Method）模式

定义一个操作中的算法的骨架，而将一些步骤延迟到子类中。模板方法模式使得子类可以不改变一个算法的结构即可重定义该算法的某些特定步骤。

- 访问者（Visitor）模式

访问者模式的目的是封装一些施加于某种数据结构元素之上的操作。一旦这些操作需要修改，接受这个操作的数据结构可以保持不变。访问者模式适用于数据结构相对不确定的系统，它把数据结构和作用于结构上的操作之间的耦合解脱开，使得操作集合可以相对自由的演化。访问者模式使得增加新的操作变得很容易，就是增加一个新的访问者类。访问者模式将有关的行为集中到一个访问者对象中，而不是分散到一个个的节点类中。当使用访问者模式时，要将尽可能多的对象浏览逻辑放在访问者类中，而不是放到它的子类中。访问者模式可以跨过几个类的等级结构访问属于不同的等级结构的成员类。

1.2 设计模式

GoF 的“设计模式”是第一次将设计模式提升到理论高度，并将其规范化。在本节中将对

几种常见的设计模式进行详细的介绍，并利用 Visual Basic 进行实现。

1.2.1 工厂模式

工厂模式是将具有共同接口的类进行实例化。工厂模式可以动态决定将某一个类实例化。工厂模式分为以下 3 种形态：

- 简单工厂（Simple Factory）模式；
- 工厂方法（Factory Method）模式；
- 抽象工厂（Abstract Factory）模式。

1．方案分析

工厂模式包括常用的简单工厂、工厂方法和抽象工厂 3 种形态。通过定义一个通用的接口来创建对象。无论使用哪一种工厂模式目的都相同，那就是“把对象的创建和对象的使用过程分离使其可以自由变动，而不会相互影响”。

本方案中采用的是抽象工厂模式。抽象工厂模式是对简单工厂模式的更高级的抽象，是所有形态的工厂模式中最为抽象和最具一般性的一种形态。它可以从几个相关的类对象中返回一个类对象。举一个生活中的例子来说明工厂模式。如在肯德基可以买到鸡翅，在麦当劳也可以买到鸡翅，这里肯德基和麦当劳合在一起，即快餐店，就是一个抽象工厂。

又如在植物园中有季生、一年生和多年生的植物，而季生、一年生和多年生的植物又分别有喜阴凉和喜阳光的植物，喜阴的植物要种在阴凉处，喜阳的植物要种在阳光充裕的地方。根据上述需求，可以将类的 UML 图设计为如图 1.1 所示的效果。

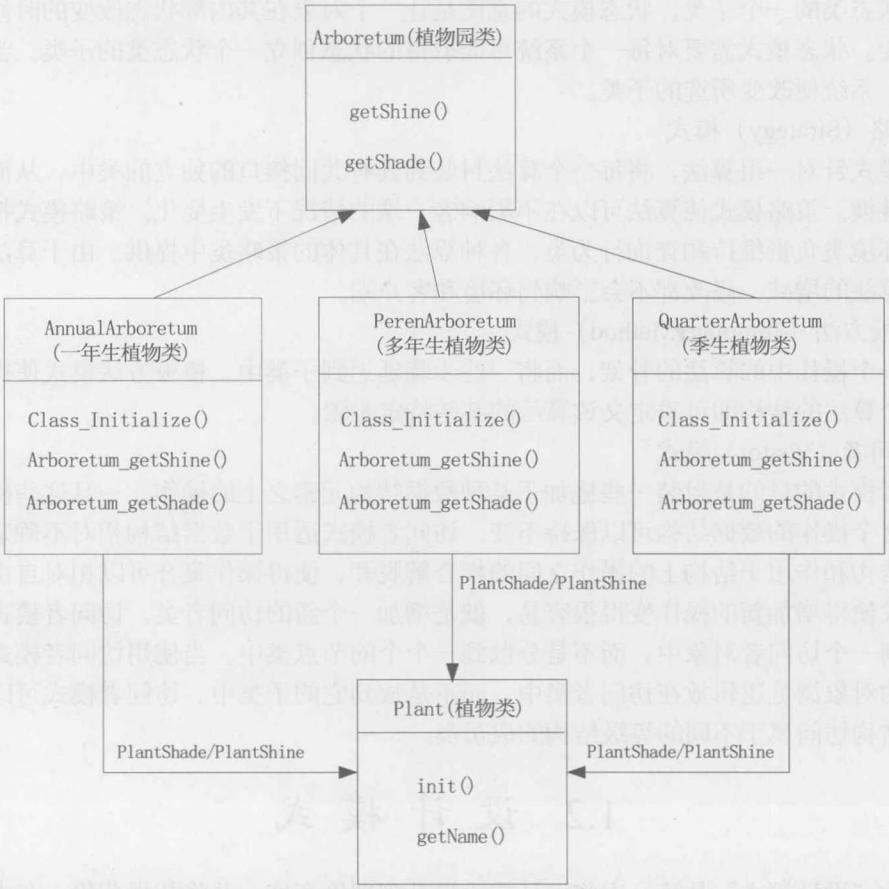


图 1.1 抽象工厂模式中类的层次结构

2．实施过程

下面以在植物园中如何种植植物的例子详细阐述抽象工厂模式的实施过程，实现的效果如图 1.2 所示。

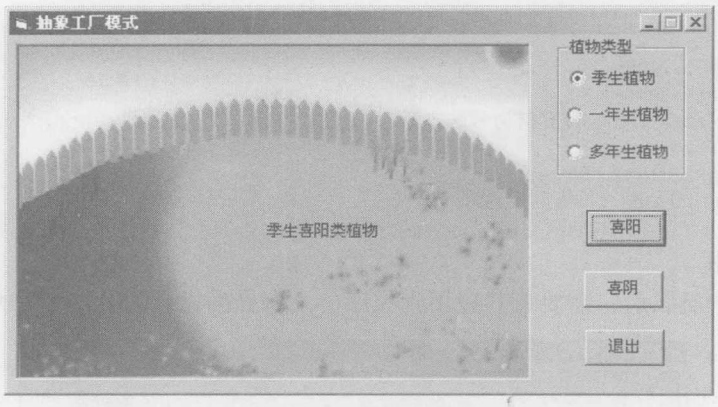


图 1.2　抽象工厂类的效果图

实例位置：光盘\mr\1\1.2\1.2.1\01

（1）Arboretum 类（植物园类）

在植物园中有些植物喜欢阳光，有些植物喜欢阴凉，在这里通过 Arboretum 类来实现判断某一种植物是喜阳还是喜阴的功能，具体的代码如下：

```
Public Function getShine() As Plant     '喜阳植物
End Function
Public Function getShade() As Plant     '喜阴植物
End Function
```

Arboretum 类事实上就是一个用于实现继承的接口类，即这里所说的抽象工厂。因为在 Arboretum 类中定义了一个具体类的方法，同时也返回一个具体的类。

（2）PerenArboretum 类（一年生植物类）

以一年生植物为例，介绍如何利用抽象工厂的方法返回植物喜阴和喜阳的情况。在使用中，可以将具体的植物名称填写到具体的位置，这里为了说明问题，以“一年生喜阴植物”和“一年生喜阳植物”来代替具体的植物名称，具体的实现代码如下：

```
' 一年生的植物类
Implements Arboretum
Private PlantShade As Plant, PlantShine As Plant
Private Sub Class_Initialize()
    Set PlantShade = New Plant
    PlantShade.init "一年生喜阴植物"
    Set PlantShine = New Plant
    PlantShine.init "一年生喜阳植物"
End Sub
Private Function Arboretum_getShine() As Plant
    Set Arboretum_getShine = PlantShine
End Function
Private Function Arboretum_getShade() As Plant
    Set Arboretum_getShade = PlantShade
End Function
```

除了一年生植物以外，还有多年生植物（PerenArboretum 类）和季生植物（QuarterArboretum 类），它们都属于具体的工厂，就像前面说的麦当劳或肯德基。它们都实现了父类中的方法，同时每个具体的工厂都可以返回一个 Plant 对象。Plant 对象可以返回具体工厂的植物名称。