HOPE

（共五册）

# Turbo Pascal 6.0
# Turbo Vision 指南

其它四种书是

- 用户指南
- 库函数参考指南
- 程序员指南
- 面向对象程序设计参考手册与程序设计技巧

晓青 编译

中国科学院希望高级电脑技术公司

4

# Turbo Pascal 6.0

# Turbo Vision 指南

晓 青 编译

中国科学院希望高级电脑技术公司

一九九一年四月

■ 北京市新闻出版局
准印证号：3314—90314
■ 订购单位：北京 8721信箱资料部
■ 电　　话：2562329
■ 电　　传：01—2561057
■ 电　　挂：0755
■ 地　　址：海淀影剧院北侧
■ 乘　　车：320、332、302路海淀黄庄下车
■ 办公地点：公司大楼 101房间

# 前 言

Turbo 序列软件问世以后即风靡全球，一直在高级语言编译和调试软件中占主导地位．继 Turbo Pascal 5.5 之后，Borland 公司于一九九零年底又推出了最新版本的 Pascal 编译调试器—Turbo Pascal 6.0．

一种新版本软件的诞生，总伴随着对原版本功能和性能的较大改进，Turbo Pascal 6.0 也不例外．对于初用 Turbo Pascal 的用户而言，完全不必重温 Pascal 的早期版本．内容翔实、刻划细腻、循序渐近的联机示教和各种介绍资料会使您学来如浴春风，得心应手．而 Turbo Pascal 的老用户也许会对新增加的功能更感兴趣．除了与早期版本完全兼容之外，Turbo Pascal 6.0 还提供了一种全新的集成开发环境(IDE)，它支持鼠标、多文件编辑和多重叠窗口，并配备了增强型调试工具和功能完备的嵌入式汇编器．Turbo Pascal 6.0 还支持 286 代码生成、远程过程调用，并能等同对待过程与函数．

为了适应程序设计风格的新潮流，Turbo Pascal 6.0 还在 5.5 版的基础上扩展了面向对象的程序设计(OOP)，允许使用私有域和私有方法，并支持在未知源级代码的情况下扩展目标代码的功能．此外，面向对象的应用工具 Turbo Vision 也是 Turbo Pascal 6.0 的一大新特色，它可以为您构造窗口应用程序，实现用户自己的 IDE．

为了满足广大用户的需要，我们根据有关资料编译了这套丛书．全套丛书包括《Turbo Pascal 6.0 用户指南》、《Turbo Pascal 6.0 程序员指南》、《Turbo Pascal 6.0 库函数参考指南》和《Turbo Vision 指南》．各书都具有自说明性，并侧重于 Turbo Pascal 的某个方面．

内容准确、版面清晰是我们所追求的目标．为了使读者更好地把握原文思想，我们对书中经常出现的术语进行了讨论、统一，尽可能准确地反映原编者的本意．相信这套丛书将成为读者的有力工具，我们将因此而感到欣慰．

本套丛书的出版得到了中科院希望高级电脑公司资料部秦人华同志的大力支持，在此表示真诚的谢意．

在本套丛书的编译过程中，虽然我们作出了很大的努力，尽量使文笔通顺、概念清晰，但仍存在不少错误，恳请广大读者谅解并指正．

编译者

一九九一年二月

于北京

# 目 录

## 第三部分　Turbo Vision 参考

# 引言

本书完整介绍了一种开发应用程序的全新工具箱—Turbo Vision. 本书不仅介绍Turbo Vision能做些什么，是怎么做的，而且详尽阐述了为什么要这么做. 只要稍微化点时间读一读Turbo Vision的基本原理，你就会发现Turbo Vision确实是一个省时、省力、高效的软件开发工具. 你可以用Turbo Vision来开发十分复杂的、有连贯一致用户界面的、交互式的应用软件，所化的时间比你能想象的还要少的多.

为什么要推出Turbo Vision?

我们在Borland公司开发了一系列窗口、会话和菜单式程序之后，就决定把所有这些功能组合成为一个可以重复使用的工具箱，用面向对象的程序设计理论作指导，Turbo Vision隆重推出了.

Turbo Vision的效果如何？结论当然是肯定的. 我们用Turbo Vision化了极短的时间就写出了一个新的Turbo Pascal集成开发环境（IDE），你当然可以用Turbo Vision来开发你自己的应用软件了.

有了Turbo Vision和面向对象的程序设计理论这两件新武器，你不必重新设计程序骨架，你需要做的只是"继承".

如果你想开发高效的、灵活的、有连贯一致用户界面的基于字符的应用软件，那么Turbo Vision就是你最合适的工具.

## Turbo Vision是什么?

Turbo Vision是一个窗口式的、面向对象的程序骨架，你可以在这种程序骨架的基础上构作你的应用程序，而不必一次又一次的重复编写程序的基本骨架，从而为你节省下大量的宝贵时间.

Turbo Vision是一个完整的面向对象的函数库，包括如下一些内容:

- 多层次的、可定义大小的、覆盖式窗口
- 下拉菜单
- 鼠标支持
- 会话框
- 内部颜色设置
- 按钮、滚行条、输入窗口、检查方框及无线按钮
- 标准的击键输入处理和鼠标器输入处理
- 其他

用Turbo Vision开发的应用软件在外观和感觉上具有上述这些现代程序的技术特性和功能特性，而你的开发工作却是轻松又轻松的.

## 预备知识

首先你需要了解面向对象程序设计的原理．用 Turbo Vision 开发应用软件扩展了面向对象的程序设计技术，包括继承（inheritance）和多态性（polymorphism），这方面的内容在《用户指南》第四章"面向对象的程序设计"中有详尽阐述．

其次你还要熟悉指针类型和动态内存分配，几乎所有 Turbo Vision 的对象都是通过堆进行动态内存分配的．你可能还需要回顾一下 New 函数的扩展语法，New 函数的扩展语法允许构造方法的蕴含（inclusion）作为 New 函数的参数，大多数 Turbo Vision 的对象是通过这种方式创建的．

## 本书的内容

Turbo Vision 是一个全新的工具箱，许多程序开发人员可能还不太熟悉它所采用的一些技术，因此本书中有大量的解释材料，还有一个完整的参考手册．

本书共分成三个部分:

- 第一部分向你介绍Turbo Vision的基本的背景性原理，并教给你用Turbo Vision开发应用程序的全过程．
- 第二部分详细介绍Turbo Vision的所有的基本组成部分，包括对Turbo Vision对象层次成分的解释和开发高质量的应用程序的建议．
- 第三部分是Turbo Vision的所有对象和其他组成部分的一个总的参考手册．

# 第一部分　Turbo Vision 入门

# 第一章　继承程序骨架

先请你回答一个问题：你新近开发完成的那个应用程序，“肉”占多少？“骨”又是占多少？

所谓应用程序的“肉”是指调用计算、数据库操作等处理的部分，而“骨”是指支撑菜单、编辑区域、出错信息窗口、鼠标支持等处理的部分。大该和大多数程序一样，你的那个程序的“骨”部分化的时间和“肉”部分化的时间一样多，甚至更多。尽管这类程序的基本部分适用于任何类似的应用程序，但出于传统，程序开发人员在开发一个软件时总是重写新的只有少许差异的编辑器、菜单管理、事件（event）处理等部分。

你知道应该尽量避免重复劳动一编写同样的程序骨架，这里给你提供的工具就可以帮助你做到这一点，你不必重复劳动，你只需“继承”。

## 窗口程序的骨架

Turbo Vision 是事件触发（event-driven）的窗口程序的骨架，Turbo Vision 提供强有力的灵活多变的“骨架”，而不附任何“肉”，你要做的只是利用 Turbo Vision 面向对象程序设计的扩展特性赋之以血肉。Turbo Vision 提供给你一个应用程序骨架对象（skeleton application object）—TApplication，你可以创建一个 TApplication 对象的子对象（不妨称之为 MyApplication）作为对 TAaplication 对象的调用，然后再把完成你自己的任务所需要的部分加进 MyApplication。

从程序设计的最顶层意义上来说，这就是你要做的所有工作了。你的应用程序形式可能是这样的：

```
begin
    MyApplication.Init;          { 初始化 }
    MyApplication.Run;           { 运行 }
    MyApplication.Done;      { 退出 }
end.
```

## 开发应用软件的一种全新的观点

你可能使用过过程／函数库，初看起来，Turbo Vision 和传统的函数库没什么两样。实质上，在市场上你可以买到能提供菜单、窗口，鼠标支持等功能的函数库。但在表面上的相似性底下，Turbo Vision 和传统的函数库有本质上的区别，记住这一点将有助于避免概念理解上的障碍。

首先，你应该提醒自己你现在是处在对象的世界里．传统的结构化程序设计理论告诉我们，当某件工具，比如菜单管理，不完全符合我们的需要时，我们必须修改它的代码，直至完全符合需要．在进入和修改一个工具的源代码的过程中要想折返回来是很困难的，除非你确切地知道代码的原本样子．进一步说，在修改一个调试正确的源代码（特别是不是你编写的程序）过程中，很可能同时播下一些十分讨厌的程序错误（bug），而且这些程序错误从你修改的代码段向外扩散．

有了 Turbo Vision 你再也不必去修改实际的源代码．你只是通过扩充的方式来"剪裁修改"Turbo Vision，而 APP.TPU 内部的 TApplication 程序骨架是一成不变的．你可以通过衍生一个新对象类型来作增加，通过用你自己写的新的方法替代继承的方法来作修改．

Turbo Vision 是一个层次结构，而不是杂乱无章的．只要你想动用工具箱中的一件工具，你就要拿起整个箱子．Turbo Vision 工具箱中的构件（component）有一个简单的脉络，它们可以通过许多精巧的联锁方式密切合作．你不要试图简单地"拿起"鼠标支持就用，这种"拿起"应该比从程序稿纸上照录你自己的鼠标支持程序要有一些工作量．

这里再提醒一下，用 Turbo Vision 开发应用程序有两个基本原理，一是完全使用面向对象的程序设计技术，二是完整地领会 Turbo Vision 的特定概念．换句话说，要根据 Turbo Vision 的本来意义，按照 Turbo Vision 的规则，恰当地使用 Turbo Vision．我们开发 Turbo Vision 的主旨在于提供给用户一个可以信赖的经过全面测试运行的工具箱，这个工具箱可以帮助用户摆脱繁重的、不必要的重复劳动．你想从中受益吗？不妨现在就开始让 Turbo Vision 为你服务．

## Turbo Vision 应用程序的组成部分

在了解 Turbo Vision 应用程序是如何工作的之前先让我们看一看 Turbo Vision 工具箱中都有些什么—Turbo Vision 给你提供了哪些工具来开发你自己的应用程序．

### 名词

Turbo Vision 应用程序是视图、事件和哑对象协同合作的结果．

#### 视图（Views）

视图是程序的屏幕可见部分—所有这些部分都是对象．在 Turbo Vision 的上下文中，屏幕可见的就是视图．显示区域、显示区域标题、窗口边框、滚行条、菜单条和会话框等等都是视图．视图可以组合成更复杂的程序单元如窗口和会话框，这些视图的组合称作对象组（group）．虽然这些视图是一个个单个的视图，但是它们协同工作．概念上可以把对象组当作视图．①

视图都是矩形的，包括单个的字符、单个水平行或单个竖直列．

---

①视图见第四章．

**事件（Events）**

事件是指程序必须作出响应的情况．事件由键盘、鼠标器或 Turbo Vision 的其他部分产生．例如一次击键是一个事件，按一下鼠标键也是一个事件．Turbo Vision 应用程序的骨架部分负责把事件排成一个队列，然后交事件处理器顺序处理．TApplication 对象作为应用程序的主体部分，含有一个事件处理器．通过下面将要解释的一种机制，不该由 TApplication 处理的事件将依次递交给程序的其他视图，直到被某个视图接受或发生"废弃事件"错误．①

例如，"F1"键调用求助功能，除非每个视图都有自己的独立的求助系统（例如一个下上下有关的求助系统），"F1"键事件是通过主程序的事件处理器处理的．相反，一般的字母、数字键或行编辑键事件是由当前视图也就是正和用户进行信息交流的视图处理的．

**哑对象（Mute objects）**

程序中的非视图对象均为哑对象．称其为"哑"是因为它们没有屏幕输出．哑对象做些计算、取接外部设备等方面的工作以及一般性的应用工作．哑对象只有借助视图才能向屏幕发送信息．这个概念对 Turbo Vision 应用程序的层次结构是很重要的．记住：只有视图才能取接屏幕．

**注意：** 哑对象仍然可以用Turbo Pascal的write或writeln语句向屏幕发送信息，但哑变量擅自屏幕发送的信息会扰乱 Turbo Vision 向屏幕发送的信息，而且这种擅自发送的"叛乱"信息会被 Turbo Vision 擦掉（如在移动窗口或改变窗口大小时）．

## 共同的外观和感觉

由于 Turbo Vision 被设计成通过标准化的、合理的途径设计屏幕输出，用 Turbo Vision 开发的应用程序在外观和感觉上是相似的．这种外观和感觉是多年的经验和使用性能测试的结果，Turbo 语言都有这种外观和感觉．应用程序有着共同的易于理解的外观对程序用户和程序开发人员有着不同的好处：不管程序内部多么神秘，程序的使用方法是熟悉的，因而易于学习．所有这方面的内容将在第四章"视图"中作详尽介绍．

图 1.1 是一组普通的对象，可能成为 Turbo Vision 应用程序的组成部分．工作屏示区是阴影背景，有别于屏幕的其它部分，和 Turbo Vision 的其他部分一样，工作屏示区也是对象．屏幕顶部的菜单条（menu bar）和底部的状态行（status line）也是对象．菜单条上的词条代表菜单，按动热键或者在词条上按鼠标键可以下拉出相应的菜单．②

状态行上的正文由你来决定，但一般用来显示程序运行的出前状态，提示当前可用的热键或者当前可用的命令．

下拉出一个菜单之后，可以用鼠标器或方向键移动加亮横条．当键入 Enter 键或按下鼠标器左键时，加亮横条处的菜单项就被选中了．选中一个菜单项即向程序的某些部分发

---

①事件见第五章．

②所有这些名词在第四章"视图"中介绍．

送一命令.

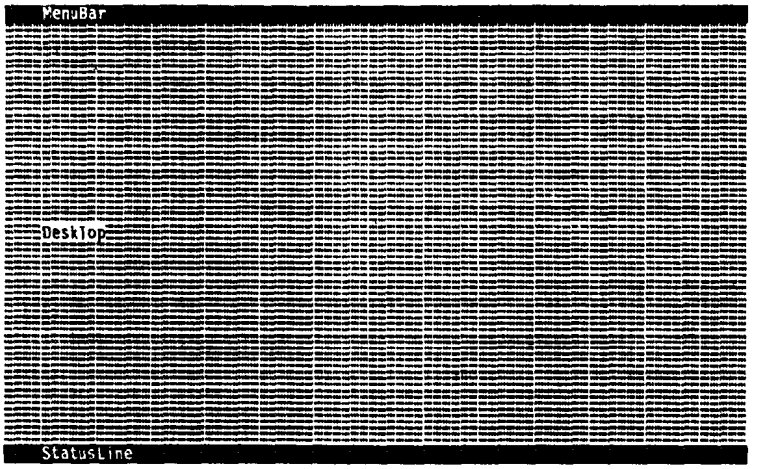


图 1.1 Turbo Vision 屏幕对象

用户通过一个或多个窗口、会话框和程序进行对话，窗口和会话框可以通过按鼠标键或击键输入来打开或关闭. Turbo Vision 提供了大量的各种各样的窗口机制来键入和显示信息. 窗口内里可以设计成可滚行，以作为一个窗口显示诸如正文文件等大量的信息. 移动窗口底部和窗口右部的滚行条可以改变窗口在欲显示信息上的相对位置. 滚行条显示出窗口在整个欲显示信息上的相对位置.

会话框通常都设有按钮，即可以在其上按鼠标键的加亮词条（或者先按 Tab 键移至该词条然后按空格键）. 按钮被选中时，加亮词条看起来好象下移了，同时发送一条指令给程序.

## "Hello, World!"的 Turbo Vision 风格

演示怎样使用一种新的程序设计语言或用户接口工具箱的通常方法是用该种语言或工具箱编写一段显示正文"Hello, World!"的程序. 这段程序一般只需要在屏幕上显示正文"Hello, World!"并返回 DOS.

Turbo Vision 给我们提供了一种完全不同的办法在屏幕上显示"Hello, World!".

传统的"Hello, World!"程序不是交互式的（它只会"说"，不会"听"），而 Turbo Vision 首先是一个开发交互式程序的工具.

这个最简单的 Turbo Vision 程序比用 begin 和 end 复合起来的 writeln 语句要复杂的多. 和传统的"Hello, World!"程序相比较，Turbo Vision 的 HELLO.PAS 程序要做如下一些事情:①

- 把工作屏示区设置成半灰度（halftone）模式
- 在屏幕顶部和底部分别显示菜单条和状态行
- 创建一个击键输入和鼠标器输入的事件处理器

① "Hello, World!"程序源代码见磁盘文件HELLO.PAS.

■ 创建一个幕后菜单对象和菜单条相联结
■ 创建一个也是幕后的会话框
■ 把会话框和菜单相联结
■ 等待你通过鼠标器或键盘发出指令。

**注意：** 在这里没有一件事情是在屏幕上显示正文的，正文是准备好了的，但在后台，要用相应的命令调出来，这一点在学习 Turbo Vision 过程中是要切记的。用 Turbo Vision 编程的实质在于设计一个视图，并且教给它接到指令时该做些什么。Turbo Vision一作为骨架一只关心从你的视图接收正确的指令，而你只关心视图接收到击键输入、鼠标器输入或菜单命令时要做些什么。

响应用户命令做某些有意义工作的代码是程序的"肉"一作为"肉"的代码包含在你创建的视图对象中。

## 运行 HELLO.PAS

在仔细分析程序 HELLO.PAS 之前，不妨把程序装上，编译后运行一下。

运行时，程序首先清屏，然后设置一个象图 1.2 中的工作屏示区。没有任何窗口被打开，只有一个菜单项出现在屏幕顶行的菜单条上：命令"Hello"，注意"Hello"中的字母"H"的颜色和"ello"的颜色是不同的。状态行有一段正文"Alt-X Exit"。

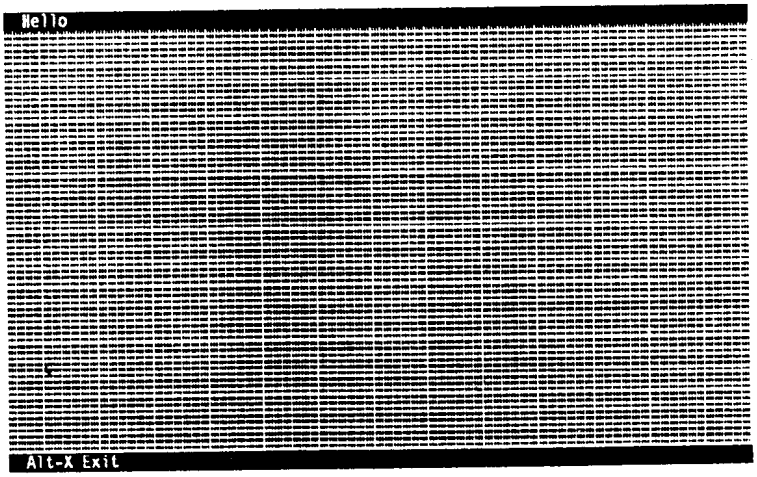


图 1.2 HELLO.PAS 初始屏幕

现在是介绍两条在任何用户环境中进行程序开发的通用规则的时候了：第一条是，永远不能让用户不知道下一步该怎么做；第二条是，让用户在任何时候都可以返回到上一步。一开始"Hello"程序提供两个明确选择：或者选取菜单项"Hello"，或者按"Alt-X"退出。

## 下拉一个菜单

记住选取菜单项"Hello"有三种方式：

■ 把鼠标器指针移至"Hello"，按动左键。

■ 按"F10"键把光标移到菜单条上，"Hello"被加亮，按Enter键.

■ 按"Alt-H"，"H"是菜单命令"Hello"中被加亮的那个字符.

用这三种方式中的任何一种作选取之后，一个下拉菜单出现在菜单项"Hello"之下. 作为一个 Turbo Vision 程序设计人员，你肯定熟悉这种下拉菜单，Turbo Vision 集成开发环境也是这样下拉菜单的. 你已经看出 Turbo Vision 采用了 Turbo Pascal 集成环境中的所有惯例，说到底 Turbo Pascal 集成开发环境是用 Turbo Vision 开发的.

你可以用键盘或鼠标器选取一个菜单项. 加亮横条可以用箭头键在菜单中上下移动. 当你要选取的菜单项处在加亮横条上时，你可以按 Enter 键来选取. 用鼠标器选取菜单项则比较有意思：当鼠标器指针指着加亮横条时在按住左键不放的同时移动鼠标器，则加亮横条相应在菜单中移动，当加亮横条处的菜单项正是你要选取的菜单项时松开左键则该菜单项就被选中了.

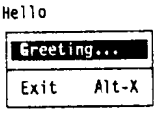


图 1.3 HELLO.PAS 高层菜单

## 会话框

当你选取了"Greeting…"菜单项后，屏幕上出现一个矩形窗口称作会话框，见图 1.4. 会话框出现在屏幕的中间，但你可以把它移动到屏幕上的任何位置，方法是这样的：把鼠标器指针移至会话框顶上，在按住左键不放的同时移动鼠标器，一旦松开左钮，会话框就移到当前位置.①

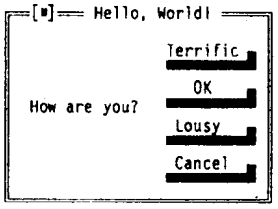


图 1.4 "Hello，World!"会话框

该会话框有个标题"Hello，World!"，左上角还有一个关窗图标，在这个图标上按鼠标键可以关闭会话框. 会话框中有一句简单的话："How are you?"，这是没有交互能力的"静态正文"（static text）. 换句话说，"静态正文"只用来标记事件，在"静态正文"上按鼠标键什么也不会发生.

## 按钮

"Hello，World!"会话框右部的四个矩形条是该会话框最有意思的部分，它们被称作按钮，是控制（control）的实例. 所以被叫做控制是因为它们看起来象电子仪器上的控

①菜单项后的省略号（…）表示该菜单项将启动一个会话框.