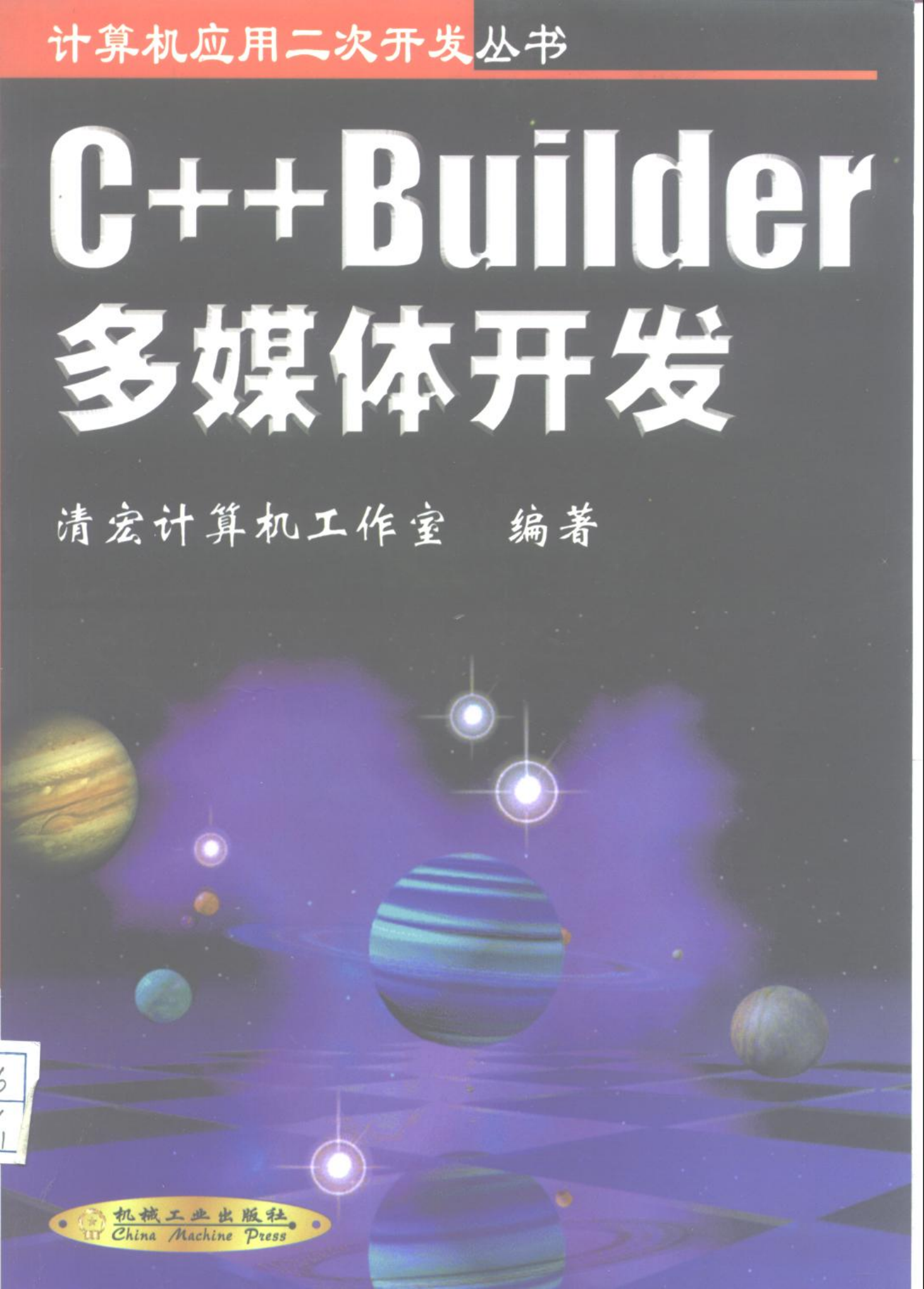
计算机应用二次开发丛书

C++Builder 多媒体开发

清宏计算机工作室 编著



机械工业出版社
China Machine Press

计算机应用二次开发丛书

C++Builder 多媒体开发

清宏计算机工作室　编著

机 械 工 业 出 版 社

本书全面系统地介绍了在 C++Builder 中进行图像处理和多媒体编程的技术。内容包括绘图编程、图像处理、利用 TeeChart 制作图表、运用 GDI 函数、其他图像控件的使用、自编一简单的地理信息系统、OpenGL、DirectDraw 技术和多媒体。书中内容详实、新颖，实例丰富，给出了大量的程序原代码，这些程序几乎包括了图形、图像处理的各个方面。

本书可供从事图形和游戏软件开发人员阅读。

图书在版编目（CIP）数据

C++Builder 多媒体开发/清宏计算机工作室编著.
-北京 :机械工业出版社,1999.12
（计算机应用二次开发丛书）
ISBN 7-111-07692-3

Ⅰ.C… Ⅱ.清… Ⅲ.①C 语言-软件工具，Builder-程序设计
②多媒体软件开发 Ⅳ.TP311.56

中国版本图书馆 CIP 数据核字（1999）第 69136 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）
责任编辑：边 萌 封面设计：姚 毅
责任印制：路 琳
北京市密云县印刷厂印刷 新华书店北京发行所发行
2000 年 1 月第 1 版第 1 次印刷
787mm×1092mm 1/16 • 29 印张 • 704 千字
0 001— 5000 册
定价：45.00 元

丛书前言

随着计算机技术的飞速发展，计算机应用及开发迅速得到推广，多种编程开发语言不断为人们掌握和使用。当前，有一大批从事计算机应用开发者对于某一种或某些开发工具的掌握已经达到了一定水准，但他们要进一步提高开发能力，就迫切地需要高层次的开发参考书。为了满足这些计算机开发人员的需求，进一步提高他们的技术水平，推动我国计算机事业的发展，我们特地组织编写了这套《计算机应用二次开发丛书》，适于中高级计算机开发人员学习。

本套丛书主要以常用的开发工具为基础，吸收成功的开发经验和技巧，结合开发原理，向读者介绍了如何有效地利用这些基本开发工具，开发实用性较强的应用程序。使读者不仅知道该怎样做，还知道为什么这样做，达到触类旁通、举一反三的目的。

本丛书重点明确、图文并茂、实用性强，每章除了讲解开发原理之外，还附有大量实例，这些实例绝大多数都是作者在开发工作中的实际成果或其中的一部分，因此，适用性非常强。

由于本丛书的读者定位比较高，适于已经具备一定开发能力的读者阅读。因此在使用这套书之前，要求读者对相应的开发工具已经初步了解。

在这套书中，我们首先推出了《C++ Builder 数据库开发》、《C++ Builder 多媒体开发》、《Delphi 数据库开发》和《AutoCAD 工程 二次开发》4 本书。此后，我们还会相继推出其他图书，以满足广大计算机开发人员的需求。

为了把这套丛书编写得更好，我们真诚地希望广大读者提出宝贵意见和建议。

编 者 的 话

图形和多媒体技术是当今计算机行业中发展最为迅速、最具有吸引力的领域之一，也是二次开发平台最为注重的、不可缺少的部分。本书共分九章，深入、全面、系统地介绍了C++Builder中的图形编程和多媒体技术。

第 1 章“绘图编程”重点介绍了 TCanvas 对象，它封装了应用程序在图形输出方面所需要的大多数 GDI 对象和绘图函数，使得图形处理工作变得异常简单。在详细介绍 TCanvas 的常用属性和方法之后，本章综合利用 TCanvas 开发了一个简单的绘图程序，对于想深入了解 C++面向对象编程思想的读者是一个极好的实例。

第 2 章“图像处理”全面介绍了 C++Builder 中图形对象对图像处理的支持，主要包括比较 TImage 控件与 TPaintBox 控件的异同、如何处理 Windows 中的重绘问题、元文件、位图像素操作、绘制透明位图等。最后，综合本章所学知识，开发了一个简单的图像处理程序。该程序可以作为开发专业图像处理的起点。

第 3 章“利用 TeeChart 制作图表”介绍了如何利用 TeeChart 的几个控件，制作精美的图表。主要内容有：利用 TChart 和 TDBChart 创建图表；利用 TQRChart 创建图表；Series 的类型；动态建立 Series；在图表上绘图；图表的缩放和滚动；图表的保存和打印等。

第 4 章“应用 GDI 函数”，并不是对前面章节的重复，而是运用 GDI 函数来处理图形控件不能直接处理的问题。诸如通过 TCanvas 提供的 Handle 属性访问设备描述表、映像模式、调色板处理、区域等一些高级话题。

第 5 章“其他图像控件的使用”介绍了如何使用其他一些图像控件，扩充 C++Builder 提供的图像处理控件的功能，减轻程序员编程的负担。主要介绍图像编辑、图像管理、图像扫描和图像缩略图等控件的功能及其使用。

第 6 章“自编一简单地理信息系统”中的地理信息系统是近年来迅速发展起来的一门新兴技术学科，简单地讲，它就是实现信息可视化的过程。本章通过一些实例，从读、写文件到显示图像、操作像素等一系列过程，说明如何利用 TImage 等控件自编一简单的地理信息系统。

第 7 章“OpenGL”详细介绍了在 C++Builder 中使用 OpenGL 绘制具有彩色及光照图形的基本内容、图形的特殊效果处理以及由位图生成三维图的方法等内容。

第 8 章“DirectDraw 技术”主要介绍在 C++Builder 中如何利用 DirectDraw 技术，快速绘制高性能的 Windows 图形。

第 9 章“多媒体”，这章本应该另写一分册来包含多媒体的内容，但多媒体与单纯的图形技术之间的界限是非常模糊的。本章大部分内容实际上是一些图形技术的灵活运用，例如图形的特殊显示效果、动画的制作等。另外，本章还涉及到利用 TMediaPlayer 控件播放 MIDI 和 WAVE 等声音文件和 AVI 视频文件等内容。

本书不但对 C++Builder 中图形对象做了详细介绍，而且通过大量实例，结合作者实际开发工作中的经验，对用户可能遇到的一些问题进行了深入透彻的分析，希望能给读者带来一些既切合实际又有用的知识。

作者水平有限，书中难免有错误和遗漏的地方，希望广大读者能谅解和批评。

编 者

目　录

丛书前言
编者的话
第 1 章　绘图编程.....1
1.1　功能强大的 TCanvas.....1
1.2　TColor.....1
1.3　TCanvas 的属性.....2
1.3.1　画笔属性.....2
1.3.2　画刷属性.....3
1.3.3　字体属性.....4
1.3.4　Pixels 属性.....6
1.3.5　CopyMode 属性.....6
1.3.6　PenPos 属性.....7
1.3.7　ClipRect 属性.....7
1.4　TShape.....7
1.5　TCanvas 方法的使用.....12
1.5.1　TCanvas 的画线方法.....12
1.5.2　绘制填充图形.....15
1.5.3　文本输出函数.....18
1.5.4　图形拷贝方法.....20
1.6　绘图程序开发.....23
1.6.1　响应鼠标事件.....23
1.6.2　构造绘图类.....26
1.6.3　切换快捷按钮.....31
1.6.4　绘图功能的实现.....32
1.7　本章小结.....44
第 2 章　图像处理.....45
2.1　图像概述.....45
2.2　TPicture.....46
2.2.1　TPicture 对象的主要方法.....46
2.2.2　TPicture 对象的主要属性.....47
2.3　TImage.....48

2.3.1 装入和保存图像.....48
2.3.2 设置图像属性.....49
2.3.3 图像浏览器.....50
2.3.4 在 TImage 控件上绘图.....51
2.4 TImageList.....55
2.5 TPaintBox.....57
2.6 TGraphic 对象.....61
2.7 元文件.....63
2.8 位图对象.....70
2.8.1 在位图上绘图.....70
2.8.2 透明位图.....71
2.8.3 位图调色板.....72
2.8.4 位图像素操作.....75
2.9 图形数据交换.....80
2.10 图像处理程序的开发.....82
2.10.1 Image 程序的工作原理.....82
2.10.2 获得图像像素值.....83
2.10.3 图像处理功能的实现.....84
2.11 本章小结.....90
第 3 章 利用 TeeChart 制作图表.....91
3.1 TeeChart 快速入门.....91
3.1.1 C++Builder 中 TeeChart 控件.....91
3.1.2 使用 TChart 或 TDBChart 创建图表.....91
3.1.3 使用 TQRChart 创建图表.....94
3.2 操纵数据序列.....95
3.2.1 数据序列的类型.....95
3.2.2 动态创建、删除、操纵序列.....102
3.3 处理图表.....103
3.3.1 在图表上绘图.....103
3.3.2 图表的缩放和滚动.....107
3.3.3 单击事件.....112
3.3.4 图表的保存.....114
3.3.5 图表的打印.....115
3.4 本章小结.....117
第 4 章 应用 GDI 函数.....118
4.1 设备描述表.....118
4.2 GDI 坐标系统、映像模式.....121
4.2.1 GDI 坐标系统、映像模式概述.....121

4.2.2 设置映像模式 122
4.2.3 坐标变换 125
4.3 利用 GDI 函数扩展 VCL 绘图功能 125
4.3.1 创建画刷 125
4.3.2 旋转字体 125
4.4 调色板 127
4.4.1 理解调色板 127
4.4.2 系统调色板与逻辑调色板 127
4.4.3 使用逻辑调色板 127
4.5 区域对象 131
4.5.1 区域的创建 131
4.5.2 区域的操作 132
4.5.3 区域的绘制 135
4.6 图形资源 138
4.6.1 建立图形资源 138
4.6.2 装载图形资源 139
4.6.3 使用光标资源 142
4.7 本章小结 143
第 5 章 其他图像控件的使用 144
5.1 TImgEdit 控件 145
5.1.1 TImgEdit 控件的功能 145
5.1.2 TImgEdit 控件的属性、方法和事件 146
5.1.3 应用举例 149
5.2 TImgAnnTool 控件 156
5.3 TImgScan 控件 163
5.3.1 TImgScan 控件概述 163
5.3.2 TImgScan 控件的属性、方法和事件 163
5.3.3 TImgScan 控件的使用 165
5.4 TImgAdmin 控件 167
5.4.1 TImgAdmin 控件的属性和方法 167
5.4.2 TImgAdmin 控件的使用 168
5.5 TImgThumbnail 控件 172
5.5.1 TImgThumbnail 控件概述 172
5.5.2 TImgThumbnail 控件的属性和方法 173
5.5.3 TImgThumbnail 控件的使用 174
5.6 图像控件的综合使用 177
5.7 本章小结 187
第 6 章 自编一简单的地理信息系统 188

6.1 地理信息系统概述.....188
6.1.1 地理信息系统的发展.....188
6.1.2 GIS 的空间数据结构.....189
6.2 系统的总体规划.....190
6.3 矢量图像的显示.....192
6.4 栅格图像的显示.....198
6.5 图像的叠加.....207
6.5.1 栅格图像和矢量图像的叠加.....207
6.5.2 矢量图像和矢量图像的叠加.....215
6.6 图像的代数运算.....218
6.7 图像的平滑和锐化.....225
6.8 制作调色板.....235
6.9 其他一些功能.....241
6.9.1 再分类.....241
6.9.2 文件描述.....248
6.9.3 保存为其他图像格式.....250
6.9.4 系统的片头.....251
6.9.5 关于模块.....253
6.10 本章小结.....256
第 7 章 OpenGL.....257
7.1 OpenGL 概述.....257
7.1.1 OpenGL 的发展.....257
7.1.2 有关 OpenGL 的概念.....257
7.1.3 与 OpenGL 相关的库函数.....260
7.1.4 C++ Builder 中利用 OpenGL 的步骤.....261
7.2 绘制几何体.....273
7.2.1 点、线、多边形的绘制.....273
7.2.2 控制点、线的属性.....281
7.2.3 曲线、曲面的绘制.....286
7.3 利用显示列表.....296
7.3.1 为何要利用显示列表.....296
7.3.2 创建并调用显示列表.....297
7.4 使用 OpenGL 应用程序库.....302
7.5 OpenGL 中图形的变换.....310
7.5.1 变换基础.....310
7.5.2 几何变换.....311
7.5.3 投影变换.....315
7.5.4 视口变换.....317

7.6 光照处理 322
7.6.1 光照处理概述 322
7.6.2 光照处理的一般步骤 322
7.6.3 光源 329
7.6.4 材料 336
7.7 混合、反走样和雾化 346
7.7.1 混合 346
7.7.2 反走样 351
7.7.3 雾化 351
7.8 OpenGL 的一个具体应用 351
7.9 本章小结 362
第 8 章 DirectDraw 技术 363
8.1 DirectDraw 概述 363
8.1.1 DirectX 技术 363
8.1.2 DirectDraw 简述 364
8.1.3 DirectDraw 和组件对象模型 365
8.2 基本概念和术语 366
8.2.1 表面 366
8.2.2 位块传输 Blt 367
8.2.3 全屏模式和窗口模式 367
8.2.4 DirectDraw 对象 368
8.3 DirectDraw 应用程序的步骤 369
8.3.1 创建 DirectDraw 对象 375
8.3.2 设置程序的工作模式 377
8.3.3 设置显示模式 377
8.3.4 创建可翻转的表面 386
8.3.5 在表面上绘制图形 387
8.3.6 翻转表面 388
8.3.7 释放 DirectDraw 对象 389
8.4 DirectDrawSurface 接口 390
8.5 在 DirectDraw 中使用 GDI 394
8.5.1 绘制多边形 394
8.5.2 显示位图 397
8.6 DirectDraw 程序的调试 402
8.6.1 全屏模式给调试带来的困难 402
8.6.2 解决办法——远程调试 402
8.7 本章小结 404
第 9 章 多媒体 405

9.1 多媒体的概念.....405
9.2 图形显示特殊效果.....405
9.2.1 渐变图形.....405
9.2.2 爆炸效果.....409
9.2.3 窗口填充效果.....412
9.2.4 淡入\淡出效果.....418
9.3 制作动画.....427
9.3.1 制作帧动画.....427
9.3.2 制作精灵动画.....429
9.4 文字的特殊效果.....437
9.4.1 滚动字幕.....437
9.4.2 卡拉 OK 字幕效果.....439
9.5 TMediaPlayer 控件及其应用.....441
9.5.1 TMediaPlayer 控件的属性.....441
9.5.2 TMediaPlayer 控件的方法.....442
9.5.3 TMediaPlayer 控件的响应事件.....443
9.5.4 播放声音文件.....443
9.5.5 播放视频文件.....446
9.5.6 制作 CD 播放器.....448
9.6 本章小结.....451

第1章 绘图编程

Windows 是一个图形操作系统，所以传统的 Windows 程序设计方法在处理有关图形设计时，多是遵循自 Windows 诞生以来的方法，即图形设备接口 GDI(Graphics Devices Interface)来进行图形编程的。这对于早期在 DOS 下进行图形编程的程序员来说，通过图形设备接口的绘图方式进行图形编程确实减少了许多麻烦。但是对于完全没有任何编程经验的初学者来说，庞大复杂的 GDI 绘图系统以及 GDI 对象的建立、使用和删除等各种操作仍是一个难以逾越的学习障碍。C++Builder 中的绘图方式避免这种复杂的 GDI 绘图方式，使在 Windows 中进行图形编程变得更加简单、直观。

本章中主要介绍了 TCanvas 的属性和方法，最后通过建立一个绘图应用程序来说明 TCanvas 的具体应用。

1.1 功能强大的 TCanvas

在 C++Builder 中提供了一个 TCanvas 对象，它封装了应用程序在图形输出方面所需要的大多数 GDI 对象和绘图命令，使得工作更加简单。尽管仍可以在 C++Builder 中直接对 GDI 编程，但大多数应用可以使用 TCanvas 的属性和方法来产生图形输出，而不必直接与 GDI 打交道。TCanvas 是一个用于绘图的表面，它相当于在 GDI 中的 DC，即设备描述表。在这个区域上，程序可实现各种绘图功能，很多图形控件（如 TImage、TPaintBox）的画布(Canvas)属性都是一个 TCanvas 对象。使用 TCanvas 非常方便，如要在窗体上画一红色边框的矩形，下列几行代码就可以实现。

```
Form->Canvas->Brush->Style=bsClear;
Form->Canvas->Pen->Color=clRed;
Form->Canvas-> Rectangle(20, 20, 200, 200);
```

当然，也可以这样写：

```
Canvas->Brush->Style=bsClear;
Canvas->Pen->Color=clRed;
Canvas-> Rectangle(20, 20, 200, 200);
```

1.2 TColor

在 Windows 程序中，颜色是根据红、绿、蓝三种颜色的饱和度来定义的，这种模型称为 RGB 模型。任何颜色都是由红、绿、蓝三种基本色的不同组合而组成，因此每种颜色都可以用红、绿、蓝基本色来表示。Red、Green、Blue 用来表示基本色构成的三个分量，它们的取值为 0~255，最小值 0 表示没有该色，最大值 255 表示最高的饱和度。三元组合（0，0，0）

用来表示黑色，因为所有颜色都没有，而三元组合（255，255，255）用来表示白色。其他颜色有各种不同的组合，如（255， 0， 0）表示纯红色，（0，255，255）表示纯青色（绿和蓝混合后得到),三种基本色的组合共有256×256×256＝16777216种。在计算机中可以用RGB宏来定义颜色，如红色可以表示为RGB（255，0，0）。

TColor类型用于定义一个对象的颜色，很多控件的颜色属性就是TColor 类型。 TColor在C++Builder中是以下面的方法说明的。

enum TColor {clMin=-0x7FFFFFFF-1, clMax=0x7FFFFFFF};

这行代码说明 TColor 的值是以十六进制进行存储的，低三位分别代表红、绿、兰三种基本色的饱和度。值00FF0000代表纯兰，0000FF00代表纯绿，000000FF代表纯红，而00000000代表黑色，00FFFFFF 代表白色。如果最高位是 0（00），则得到的颜色最接近系统调色板的颜色；如果最高位是 1（01），则得到的颜色最接近当前调色板的颜色；如果最高位是 2（02），则得到的颜色最接近当前设备描述表的逻辑调色板的颜色。

在Graphic.hpp中定义了一些常用的TColor的常量，这些常量或直接映射成系统调色板中最相近的颜色，或映射成Windows 控制面板中颜色部分的系统视频颜色。

直接映射成系统调色板中的颜色有：

clAqua、clBlack、clBlue、 clbkGray、 clFuchsoa、 clGreen、clLime、clLtGray、clMaroon、clNavy、clOlive、clPurple、clRed、clSilver、clTeal、clWhite、clYellow。

映射成Windows控制面板中颜色部分的系统视频颜色有：

clActiveCaptio，clInactiveCaption，clBackground，clMenu, clWindow，clWindowFrame, clMenuText, clWindowText, clCaptionText, clActiveBorder, clInfoBk, clBtnFace, clGrayText, clInactiveBorder, clHighlight, clAppWorkSpace, clHightlightText, clBtnShadow, cl3Dlight, clBtnText, clInactiveCaptionText, clBtnHighlight, clInfoText , cl3DdkShadow。

在应用程序中，可以利用下列三种方法设置颜色：

```
Form1->Color=clRed;
Form1->Color=TColor(RGB(255,0,0));
Form1->Color=static_cast<TColor>(0x0000FF);
```

最后一行代码使用了C++中的类型转换。有关static_cast和dynamic_cast的用法，可以查阅帮助文件。

1.3 TCanvas的属性

1.3.1 画笔属性

TCanvas的画笔（Pen）属性是一个TPen对象。在用画布画线的时候，需要设置画笔的属性。对每一个画笔均可以选择不同的宽度（Width）、颜色（Color）、线型（Style）和绘图模式（Mode）。

1. Width属性 确定线的宽度，缺省宽度为1，可以按如下方式设置画笔的宽度：

```
Canvas->Pen->Width=5;
```

按缺省的映像模式，该代码创建一个宽度为5个像素的画笔。

2. Color 属性 控制线的颜色，可以使用预定义的颜色或设置自己的颜色，下面是画笔

颜色设置的实例：

```
Canvas->Pen->Color=clBlue;
Canvas->Pen->Color= clGrayText ;
Canvas->Pen->Color=TColor(RGB(125,0,0));
```

3．Style 属性　确定线的各种类型，线的类型如表 1-1 所示。

表 1-1　画笔的线型

值	说　明
PsSolid	实心线
PsDash	由下画线组成的线段
PsDot	由点组成的线段
PsDashDot	点画线
PsDashDotDot	双点画线
PsClear	看不见的线
PsInsideFrame	实心线，如果线宽大于一个像素时则使用抖动色

线的式样可以这样设置：

```
Canvas->Pen-> Style＝psDot;
```

4．Mode　确定线的颜色与画布颜色如何相互作用，与 Windows 中的 ROP（光栅操作）代码相对应。表 1-2 说明 Mode 的取值及其作用结果。

表 1-2　画笔的模式

绘 图 模 式	作 用 结 果
PmBlack	永远是黑色
pmWhite	永远是白色
pmNop	不改变画布的颜色
pmNot	画布背景色的反色
pmCopy	由画笔的颜色控制
pmNotCopy	画笔颜色的反色
pmMergePenNot	画笔颜色和画布反色的组合
pmMaskPenNot	画布和画笔反色的组合
pmMergeNotPen	画笔颜色和画布背景色的组合
pmMaskNotPen	画布背景色和画笔反色的组合
pmMerge	画笔颜色和画布背景色不相同部分的组合
pmNotMerge	pmMerge 的反色
pmXor	画笔颜色和画布背景色取或
pmNotXor	pmXor 的反色
pmMask	画布背景色与画笔颜色取并后的组合
pmNotMask	pmMask 的反色

画笔的 pmNotXor 模式和 pmXor 模式是一个非常有用的属性，当利用 pmNotXor 模式画图以后，如果在原来的图形上再画一次，则会擦除原来的图形。使用该模式比较常见的就是利用鼠标绘图，例如画“橡皮筋”线。

1.3.2　画刷属性

Canvas 的画刷（Brush）属性是一个 TBrush 对象，它封装了标准的 Windows 刷子对象。画刷可以利用颜色和图案来填充矩形、多边形和圆等。使用画刷的 Color 属性可以设置画刷

的颜色，缺省的画刷颜色是 clWhite。可以通过下面的代码来设置画刷的颜色。

Canvas->Brush->Color = clRed;

使用画刷的 Style 属性可以确定填充图案的模式。表 1-3 列出了一些预定义的画刷模式。

表 1-3 预定义的 TBrush 的 Style 属性的模式

值	描 述
BsSolid	实心颜色
BsClear	透明
BsHorizontal	水平线
BsVertical	垂直线
BsFDiagonal	右斜线
BsBDiagonal	左斜线
BsCross	水平垂直交叉线
BsDiagCross	左右垂直交叉线

如果要画一个内部不填充的图形，可以将画刷的模式设置为 bsClear。可以利用下面的代码设置画刷的模式。

Canvas->Brush->Style = bsDiagCross;

除了使用这些预定义的画刷模式外，还可以自己创建画刷模式。这些模式以位图形式存储，或者使用已有的位图。这些自定义的位图可以通过画刷的 Bitmap 属性加载，画刷可以使用位图填充图形产生特殊效果。位图大小为 8 个像素高，8 个像素宽。如果位图大于 8×8 的像素，则只有左上角 8×8 的像素可以使用。位图画刷的使用如下：

```
//---------------------------------------------------------------------------
Graphics::TBitmap *BrushBmp = new Graphics::TBitmap;
try
 {
  BrushBmp->LoadFromFile("MyBitmap.bmp");
  Form1->Canvas->Brush->Bitmap = BrushBmp;
  Form1->Canvas->FillRect(Rect(0,0,100,100));
 }
__finally
 {
  Form1->Canvas->Brush->Bitmap = NULL;
  delete BrushBmp;
 }
```

注意使用完画刷后，必须将这个属性复位为空（NULL）,代码如下：

```
Form1->Canvas->Brush->Bitmap = NULL;
delete BrushBmp;
```

1.3.3 字体属性

Canvas 的字体（Font）属性是一个 TFont 对象。在 Windows 程序设计中，字体的处理曾经是一件很令人头痛的事情，在 C++Bulider 中，TFont 封装了 Windows 中的字体属性，

使得改变字体的属性变得异常的简单。字体的主要属性有字体颜色（Color）、字体大小（Size）、字体样式（Style）和字体名(Name)等。字体颜色属性的使用与前面画笔，画刷颜色的使用类似。

1．字体的 Name 属性　它定义了所用字型的名称，如 Arial,Times New Roman 等。可以使用系统中任何预先安装的字型，也可以使用自定义的字型。利用下列代码能够获得系统中可以使用的字型。

```
//---------------------------------------------------------------------------
void __fastcall TForm1::FormCreate(TObject *Sender)
{
  for (int i = 0; i < Screen->Fonts->Count; i++)
    ComboBox1->Items->Add(Screen->Fonts->Strings[i]);
}
//---------------------------------------------------------------------------
```

程序运行情况如图 1-1 所示。Name 属性采用标准字符串（AnsiString），可以利用下面代码设置字体的 Name 属性。

```
Canvas->Font->Name=“Arial ”;
```

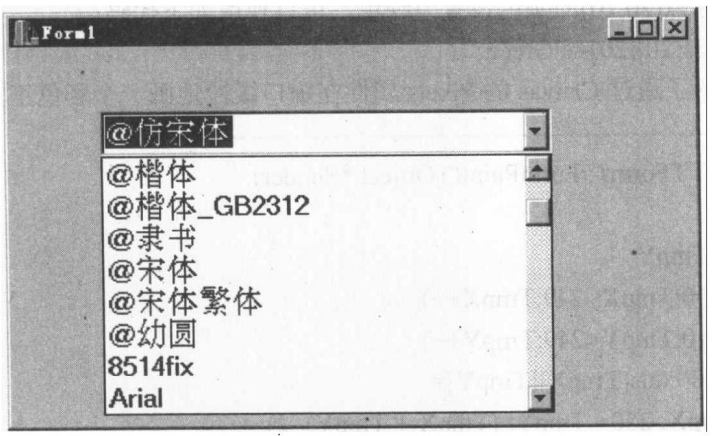


图 1-1　获得系统可使用的字体

2．字体的 Size 属性　它用来确定字体的大小，它通常指的是字型的磅值大小。与字体大小相关的属性还有 Height 属性，字体的 Height 属性是指字型不包括内部铅框的高度，用像素来表示。字体的 Size 属性和 Height 属性有如下关系：

```
Font->Size = -Font->Height * 72 / Font->PixelsPerInch
```

可以看出，当在 Height 属性中输入一个正值时，字体的 Size 属性值变为负。反之，当在 Size 属性中输入一个正值时，字体的 Height 属性变为负。利用下面代码可以设置字体的 Size 属性。

```
Canvas->Font->Size=24;
```

3. **字体的 Style 属性** 与画笔和画刷的 Style 属性的不同是，它是枚举数据类型。字体的 Style 属性是一个集合，该属性可以将字体赋值成黑体（fsBold）、斜体（ fsItalic），有下画线（fsUnderline）以及有删除线（fsStrikeOut）。可以通过 TFontStyles 数据类型访问该属性。利用下面代码可以设置字体的 Style 属性。

```
TFontStyles  FontStyle;
FontStyle<< fsBold <<fsUnderline;
Canvas->Font->Style=FontStyle;
```

也可以利用下面一行代码来代替上面三行代码。

```
Canvas->Font->Style<<fsBold <<fsUnderline;
```

这段代码将字型设置为黑体并有下画线，关于集合的运算可以查看 C++Builder 的帮助文件。

1.3.4 Pixels 属性

Canvas 的 Pixels 属性可以用来确定像素的颜色。可以利用 Pixels 属性来获得某一点的颜色值，也可以通过它来设置某一点的颜色值。

如要获得坐标点(20 , 20)的颜色值，可以使用下面的代码：

```
TColor  Color;
Color=Canvas->Pixels[20][20];
```

如果要将坐标点(20,20)的颜色设置为绿色，可以使用如下代码：

```
Canvas->Pixels[20][20]=clGreen;
```

下列代码演示了通过 Canvas 的 Pixels 属性在窗口缓慢地画一个彩色正方形。

```
//---------------------------------------------------------------------------
void __fastcall TForm1::FormPaint(TObject *Sender)
{
  int TmpX, TmpY;
  for (TmpX=0;TmpX<249;TmpX++)
  for (TmpY=0;TmpY<249;TmpY++)
    Canvas->Pixels[TmpX][TmpY]=
    RGB(TmpX, 250 - TmpY, (TmpX + TmpY)/ 2);
}
//---------------------------------------------------------------------------
```

1.3.5 CopyMode 属性

Canvas 的 CopyMode 属性确定如何将一幅图像从其他画布拷贝到它的绘制表面。CopyMode 影响着画布上图像的绘制方式。用 CopyMode 属性可以产生多种不同的效果。比如，将几幅图像叠加在一起，用不同的 CopyMode 组合多幅图像使得位图的某一部分变得透明等。CopeMode 的缺省模式为 cmSrcCopy。表 1-4 列出了 CopyMode 属性的不同值及其拷贝效果。