计算机实用技术系列丛书(二)

附磁盘

Borland C++

鼠标的应用

蔡明志　编著

学苑出版社

计算机实用技术系列丛书(二)

Borland C++鼠标的应用

蔡明志　编著
天　奥　改编
熊可宜　审校

学苑出版社
1994

(京)新登字 151 号

内 容 摘 要

本书是以面向范例编写完成的。主要介绍了如何利用鼠标来控制我们编写的应用程序。在本书后面建立了一套简单的绘图系统，此绘图系统完全是根据本书所建立的各鼠标功能函数来制作的，所以非常易于掌握。

本书附有大量范例程序，以供广大读者参考。

欲购本书的用户请直接与北京 8721 信箱联系，邮码：100080，电话：2562329。

版 权 声 明

计算机实用技术系列丛书(二)

Borland C++鼠标的应用

编　　著：蔡明志
改　　编：天　奥
审　　校：熊可宜
责任编辑：甄国宪
出版发行：学苑出版社　　**邮政编码**：100036
社　　址：北京市海淀区万寿路西街 11 号
排　　版：北京天奥科技公司激光照排中心
印　　刷：双青印刷厂
开　　本：787×1092　1/16
印　　张：25.875　　**字数**：580 千字
印　　数：1～5000 册
版　　次：1994 年 5 月北京第 1 版第 1 次
ISBN7—5077—0760—1/TP・7
本册定价：59.00 元(含盘)

学苑版图书印、装错误可随时退换

序　言

鼠标(Mouse)是目前很流行的一个名词，各种绘图系统或CAD计算机辅助设计、排版系统……等等都是以鼠标为主要的操作工具，因为它能提供我们更方便、更人性化的方法来控制程序的执行，而省去了过去以键盘来操作软件的不方便。近年来，各种应用软件都具备了鼠标操作的支持，如Windows、MS－DOS DOSSHELL、PCTOOLS、NORTON Utilities和Commands、BORLAND C＋＋等等，甚至新出炉各式各样的游戏都具备了鼠标操作的功能，建立了功能完整的用户界面。尤其是声光效果要求稍高的计算机软件市场中，使用鼠标来作为用户与计算机之间的接口工具更是形势发展的趋向。

本书的宗旨是带领读者走进计算机鼠标的世界，不但为读者介绍如何来控制鼠标，利用它来为我们做事，还提供一些鼠标功能函数，让读者很方便地使用，并且以范例程序为导向，教导读者怎样将这些鼠标功能函数实际应用于一般的应用程序中，最后，我们建立了一套简单的绘图系统，当然，这套绘图系统的流程控制完全是根据本书所建立的各鼠标功能函数来制作的。

蔡明志

前　言

相信读者都已经知道本书讨论的重点是鼠标，也就是讨论如何利用鼠标来控制我们自己所编写的应用程序，所以算是一本作为提高的书籍，也许大部分读者都有编写 Borland C++语言的能力，所以在本书各章节中讨论各个范例程序或工具函数时，都假设读者都已经很熟悉如何编写 Borland C++语言的的程序，因此关于程序结构的部分不再多加说明，只说明程序各部分所代表的意义。在此建议那些对 Borland C++还不太熟悉的读者最好能回头去看看 Borland C++程序设计的书籍，才不致在阅读本书时感到吃力。

为了要让读者能够实际体会如何将鼠标功能加入我们编写的应用程序中，所以本书是以面向范例编写完成的。也就是说，在讨论大部分鼠标功能时，我们都会举出范例程序来说明如何将该功能加入程序中，以及使用时机。除此之外，在第二章我们还会先举一个未加入任何鼠标功能的范例程序，等到第三章讨论完各项鼠标功能之后，在第四章便会将第三章讨论的各鼠标的功能实际加入第二章所举的范例程序内，以让读者一步步地了解如何建造出一个具有鼠标功能的应用程序。

欢迎加入希望用户协会

当今的计算机技术发展迅猛，应用领域繁多，如何准确、有效、地为您提供服务，已成为我们迫切关心的问题。为此，我们决定创立希望用户协会，其宗旨在于加强与用户的联络，了解用户的各种信息与需求，为用户提供更完善更周到的服务。

本协会的成员将定期获得各种软件、资料与培训等讯息，优先得到有关技术服务。请您认真填写后面的会员登记表，其中的信息将用电脑进行管理。

作为美国Borland软件公司在中国的总代理，北京希望电脑公司经Borland公司授权，对其产品dBASE Ⅳ 2.0 进行系统级汉化。另外，对Borland C++ & AF 3.1、 Turbo C++ for DOS 3.0、Turbo C++ for Windows、Turbo C++ Visual Edition for Windows四个产品进行了本地化。预计94年3月推出以上产品。

Borland 软件系列清单：

1. Borland C++ & AF 3.1
2. Borland C++ 3.1
3. Turbo C++ for DOS 3.0
4. Turbo C++ for Windows
5. Paradox for DOS 4.0
6. Paradox for Windows
7. Pdox Engine & Database Framework 3.0
8. Turbo Pascal for DOS 7.0
9. Turbo Pascal for Windows 1.5
10. dBASE Ⅳ 2.0
11. dBASE Ⅳ Compiler
12. Object Vision 2.1

软件与培训

北京希望电脑公司即将与Borland公司密切合作，经销Borland多种软件产品的同时，提供配套的函授与培训服务，以满足用户的不同要求。欢迎用户索取软件与培训资料。

联系方法

有关资料事宜，请与朱红小姐联系；有关软件事宜，请与周东先生联系；有关函授与培训事宜，请与宋明华先生联系。 联系方法如下：

通信地址： 北京 8721 信箱 邮政编码： 100080

联系电话： (01)2562329 (01)2541992 (01)2579874

传真电话： (01)2561057

目　录

第一章　关于本书

下面是本书各章所讨论的重点：

第一章　关于本书

第二章　未使用鼠标的一个范例 GP1.CPP

先举一个未使用任何鼠标功能的的范例程序。虽然程序有点长，但请读者耐心地看完，因为本程序中会讨论许多关于屏幕菜单控制的技巧，如颜色的控制、反白卷动棒的制作等等。而且在第四章中我们会为这个程序加进鼠标的功能，以教导读者如何实际地应用各种鼠标功能。

第三章　鼠标功能的介绍

介绍鼠标所提供的各种功能，并提供根据这些功能所建立的函数，并且会举一些小例子来说明如何利用这些函数，以及它们的使用时机。

第四章　将范例程序 GP1.CPP 加入鼠标功能

将第三章所提供的各鼠标功能函数加进第二章所举的范例程序 GP1.CPP 中，使 GP1.CPP 具有鼠标控制的功能，以教导读者如何实际地应用各种鼠标功能。

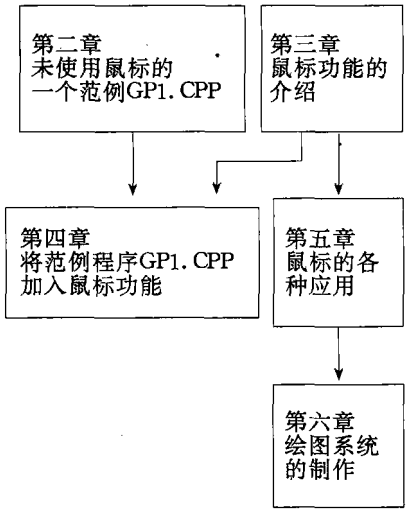
第五章　鼠标的各种应用

本章中我们会教导读者如何编写一些常见的应用程序，如 Double—click、PULL 窗口、延迟效果及利用鼠标来画曲线、直线、矩形等等。

第六章　绘图系统的制作

绘合以上各章所介绍的各种功能及应用之后，在本章中我们将会介绍如何建立一个以鼠标为驱动工具的绘图系统，以作为本书的总结。

以上各章的流程如下图所示：



最后建议读者在学习了各种鼠标功能或未曾学习过的各种知识之后，最好能亲自上机实

际操作练习，因为只能看懂并没有太大的效果，真正利用自己的双手去完成一项工作才算是自己的东西。

第二章　未使用鼠标的一个范例

在第一章的简介中我们曾说到在第二章会先举一个未使用鼠标来驱动的范例程序，等到第四章再将这个程序引进鼠标的控制功能，以让读者能亲身体会如何编写具有鼠标功能的应用程序，所以在本章中我们先来看看这个还没有加进鼠标功能的范例程序。

本书算是一本 C++语言之后的提高应用书籍，所以对于 C++语言的程序编写法以及一些细节部分我们就不再多做解释；除此之外，我们还假设读者对 Borland C++所提供的绘图系统有所了解，由于鼠标应用最直接的联想就是在绘图上，因此在本书中我们会将常常见到关于绘图的程序，所以希望那些还没有接触过 Borland C++绘图功能的读者能先参考有关的书籍，才不致在阅读本书时感到吃力，不过，对于那些具有关键性及比较重要的绘图函数，我们仍会在遇到它时稍作解释。现在就让我们来看看本章所要举的例子。

2.1　范例程序 GP1.CPP

下面是范例程序 GP1.CPP 的列表.

程序列表(本程序所在路径和文件名为 A:\CH2\GP1.CPP)：

```
#include <graphics.h>
#include <stdlib.h>
#include <alloc.h>
#include <conio.h>
#define   F1              -59
#define   F2              -60
#define   F3              -61
#define   UP_ARROW        -72
#define   DOWN_ARROW      -80
#define   ENTER            13

class GraphOne{
   private:
      int   MaxX, MaxY;
      int   XL,   YL;
      char   Ch;
      void *Buf;

      int   FuncNo  ;
      int   FuncY[3];

```

```
22          int    Graph    ;
23          int    Times    ;
24          int    Status ;
25    public:
26        void Run();
27        void End();
28
29          int    GetKey();
30          void PutRec(int);
31          void PutBar(int, int);
32          void UpScroll();
33          void DownScroll();
34
35          void DrawFrame();
36          void GetBuf();
37
38          void Circle(int);
39          void Rectangle(int);
40          void Bar(int);
41          void Line(int);
42          void Pie(int);
43          void CtrlProgram();
44    };
45
46
47    void main()
48    {
49     int    gd=DETECT, gm;
50     GraphOne G;
51        initgraph(&gd, &gm, "c:\\borlandc\\bgi");
52        G. CtrlProgram();
53     }
54
55    void GraphOne::CtrlProgram()
56        {
57          FuncNo    = 0;
58          FuncY[0] =    3 ;
59          FuncY[1] = 133 ;
60          FuncY[2] = 223 ;
61
62          Graph    = 0;
63          Times    = 0;
64          Status = 0;
```

```
65      GetBuf();
66      DrawFrame();
67
68      PutBar(0, 0);
69      PutBar(1, 0);
70      PutBar(2, 0);
71
72      FuncNo = 0;
73      PutRec(FuncNo);
74
75      while (1)
76      {    Ch = GetKey();
77           switch (Ch)
78              {       case F1 :
79                                PutRec(FuncNo);
80                                FuncNo = 0;
81                                PutRec(FuncNo);
82                                break;
83
84                      case F2 :
85                                PutRec(FuncNo);
86                                FuncNo = 1;
87                                PutRec(FuncNo);
88                                break;
89
90                      case F3 :
91                                  PutRec(FuncNo);
92                                  FuncNo = 2;
93                                  PutRec(FuncNo);
94                                  break;
95
96                       case UP_ARROW :
97                                  UpScroll();
98                                  break;
99
100                     case DOWN_ARROW :
101                               DownScroll();
102                               break;
103
104                     case ENTER :
105                               if (FuncNo == 2)
106                                  {   if (Status==1)
107                                         End();
```

```
                                        else
                                        {   PutBar(2, 0);
                                            Run();
                                            PutBar(2, 0);
                                        }
                                    }
                                break;

            }  //  switch
        }  //  while loop
    }


    void GraphOne::Run()
    {   int   times;

        times = 10 + 20 * Times;
        switch (Graph)
        {
            case 0 : Circle(times);
                     break;

            case 1 : Rectangle(times);
                     break;

            case 2 : Bar(times);
                     break;

            case 3 : Line(times);
                     break;

            case 4 : Pie(times);
                     break;
        }
    }


    void GraphOne::End()
    {
        free(Buf);
        closegraph();
        exit(0);
    }
```

```
151
152
153  int GraphOne::GetKey()
154  {   int   Ch;
155
156      Ch = getch();
157      if (Ch == 0) Ch = -- getch();
158      return(Ch);
159   }
160
161
162  void GraphOne::PutRec(int Func)
163  {
164      setcolor(LIGHTGREEN);
165      setwritemode(XOR_PUT);
166      rectangle(8, FuncY[Func]+3, 95, FuncY[Func]+27);
167      setwritemode(COPY_PUT);
168  }
169
170
171  void GraphOne::PutBar(int Func, int Num)
172  {
173      switch (Func)
174      {
175              case   0 : putimage(4, FuncY[0]+30+20 * Num+1,
176                                  Buf, XOR_PUT);
177                     break;
178
179              case   1 : putimage(4, FuncY[1]+30+20 * Num+1,
180                                  Buf, XOR_PUT);
181                     break;
182  {
183              case   2 : putimage(4, FuncY[2]+30+20 * Num+1,
184                                  Buf, XOR_PUT);
185                        break;
186        }
187  }
188
189
190  void GraphOne::UpScroll()
191  {
192      switch (FuncNo)
193         {
```

```
194                 case 0 : PutBar(0, Graph);
195                             Graph = (Graph+5-1) % 5;
196                             PutBar(0, Graph);
197                             break;
198
199                 case 1 : PutBar(1, Times);
200                             Times = (Times+3-1) % 3;
201                            PutBar(1, Times);
202                             break;
203
204             case 2 : PutBar(2, Status);
205                         Status = 1 - Status;
206                         PutBar(2, Status);
207                         break;
208         }
209 }
210
211 {
212 void GraphOne::DownScroll()
213 {
214     switch (FuncNo)
215       {
216         case 0 : PutBar(0, Graph);
217                     Graph = (Graph+1) % 5;
218                     PutBar(0, Graph);
219                     break;
220
221         case 1 : PutBar(1, Times);
222                     Times = (Times+1) % 3;
223                     PutBar(1, Times);
224                     break;
225
226         case 2 : PutBar(2, Status);
227                     Status = 1 - Status;
228                     PutBar(2, Status);
229                     break;
230       }
231 }
232
233
234 void GraphOne::DrawFrame()
235 {   int   i;
236     char   graph[5][10] =   {   " Circle   ",
```

```
237                                    "Rectangle",
238                                    "   Bar      ",
239                                     "   Line     ",
240                                     "   Pie      ",
241                                  };
242     char    times[3][10] =    {    " 10 times",
243                                    " 30 times",
244                                    " 50 times"
245                                  };
246     char    status[2][10] = {    "   RUN",
247                                    "   EXIT"
248                                  };
249
250     MaxX = getmaxx();
251     MaxY = getmaxy();
252     XL = MaxX-4-101+1;
253     YL = MaxY-4-4+1;
254
255     //    Set Background color = BLUE.
256     setpalette(0, BLUE);
257     setpalette(1, BLACK);
258     cleardevice();
259
260     //    Draw frame of the screen.
261     setcolor(YELLOW);
262     setlinestyle(0, 1, 3);
262     rectangle(0, 0, MaxX, MaxY);
264     setlinestyle(0, 1, 1);
275     rectangle(3, 3, MaxX-3, MaxY-3);
266
267     line(100, 3, 100, MaxY-3);
268
269     //    Draw the F1 function menu.
270
271     setcolor(LIGHTGREEN);
272     outtextxy(16, FuncY[0]+5,   "F1 ------- ");
273     outtextxy(16, FuncY[0]+15, "   Graph");
274     setcolor(YELLOW);
275     line(3, FuncY[0]+30, 100, FuncY[0]+30);
276     for (i=0; i<5; i++)
277       {
278             setcolor(LIGHTRED);
279        outtextxy(16, FuncY[0]+30+20*i+5, graph[i]);
```

```
280                 setcolor(YELLOW);
281                 line(3, FuncY[0]+30+20*(i+1),
282                         100, FuncY[0]+30+20*(i+1));
283             }
284
285         //    Draw the F2 function menu.
286         setcolor(LIGHTGREEN);
287         outtextxy(16, FuncY[1]+5,   "F2 ------- ");
288         outtextxy(16, FuncY[1]+15, "   times");
289         setcolor(YELLOW);
290         line(3, FuncY[1]+30, 100, FuncY[1]+30);
291         for (i=0; i<3; i++)
292           {
293             setcolor(LIGHTRED);
294             outtextxy(16, FuncY[1]+30+20*i+5, times[i]);
295             setcolor(YELLOW);
296                   line(3, FuncY[1]+30+20*(i+1),
297                     100, FuncY[1]+30+20*(i+1));
298           }
299
300         //    Draw the F3 function menu.
301
302         setcolor(LIGHTGREEN);
303         outtextxy(16, FuncY[2]+5,   "F3 ------- ");
304         outtextxy(16, FuncY[2]+15, " Status");
305         setcolor(YELLOW);
306         line(3, FuncY[2]+30, 100, FuncY[2]+30);
307         for (i=0; i<2; i++)
308           {
309             setcolor(LIGHTRED);
310             outtextxy(16, FuncY[2]+30+20*i+5, status[i]);
311             setcolor(YELLOW);
312             line(3, FuncY[2]+30+20*(i+1),
313                     100, FuncY[2]+30+20*(i+1));
314     }
315
316         setcolor(LIGHTGREEN);
317         outtextxy(10, MaxY-20, "Mouse   Test");
318     }
319
320
321
322     void GraphOne::GetBuf()
```