

图形窗口界面平台的研究与实现

王延乔

提 要 叙述了 DOS 下如何实现图形窗口、菜单及下拉式菜单的生成、驱动与管理,提出了其构造的基本结构,解决了中西文环境下的全汉化、鼠标与键盘同时驱动、菜单单位图显示、辅助工具支持等问题,实现了 DOS 图形方式下界面平台的仿 Windows 化操作.

关键词 图形窗口; 菜单; 全汉化; 界面平台
中图法分类号 TP311.56

0 引 言

随着计算机技术的迅猛发展,PC 已进入了 Windows 和 Win 95 时代;但在许多情况下 DOS 仍有它的一席之地,同时,如何在对环境要求最低的情况下,提供良好的用户界面平台也是一个很困难的问题,很好地解决这个问题将有助于软件开发的效益与质量,更好地创建有特色和轻松的编程工具.因此,如何构造界面核心结构是体现界面的灵活性、可扩充性、功能覆盖、事件驱动、代码最简等方面的关键所在.本文将从窗口基本构成出发,给出最基本的结构代码,讨论建立在该基础上的图形窗口界面平台各种功能的实现与驱动.

1 基本结构分析

1.1 窗口结构与构造函数

窗口结构是构成图形界面的基础,对它的定义与控制决定了系统菜单与功能驱动的关键要素,以下是用 C 语言描述的窗口具体结构.

```
typedef struct g_w_node
{
    int left,top,right,bottom; /* 窗口位置与区域 */
    int htu,wtl,htb,wtr; /* 窗口上下左右边区域的宽度 */
    char FDX; /* 窗口阴影宽度 */
    int tfc,tbkc; /* 窗口标题颜色和边区域背景颜色 */
}
```

收稿日期: 1996-09-25

作者王延乔,男,工程师,上海师范大学地理系,上海,200234

```

int msgfc,msgbkc;          /* 窗口区域的文字颜色和背景颜色 */
int winmode;              /* 决定窗口边框类型、是否调用位图及位图与文字说明的
                           位置关系 */
char framemode,framewide; /* 决定窗口菜单中各选项立体投影状态的投影宽度 */
int frametopcolor,framebottomcolor; /* 决定窗口菜单中各选项立体投影状态的上下
                                       边框颜色 */

char state;               /* 表明当前窗口的打开或关闭状态 */
char tp;                  /* 决定当前窗口标题所在窗口上的位置 */
char * title;             /* 当前窗口标题字符串 */
char _huge * buffer;     /* 指向窗口缓冲区的指针 */
char active;              /* 决定是否使用窗口缓冲区保护窗口区域 */
int cx,cy;                /* 内部存放相对于当前窗口光标所在的位置 */
char CHH,CHW;            /* 决定在窗口中输出汉字的字高和字宽 */
g_Rect EBox;             /* 窗口左上角退出框的矩形范围 */

```

```
} g_WINDOW;
```

窗口结构定义完毕,便可产生相应的结构函数,部分基本函数如下:

① 创建窗口

```
g_WINDOW *g_Window_Create(int left,int top,int right,int bottom,
                           int htu,int htb,int wtl,int wtr,char FDX,int winmode,
                           char tp,int tfc,int tbkc,char * title,int msgfc,int msgbkc);
```

② 打开窗口

```
int g_Window_Open(g_WINDOW * gw);
```

③ 关闭与清除窗口

```
int g_Window_Close(g_WINDOW * gw);
int g_Window_Kill(g_WINDOW * gw);
```

④ 在窗口中输出与添加汉字

```
int g_Window_Text(g_WINDOW * gw,char * str);
int g_Window_String(g_WINDOW * gw,char * p);
```

1.2 菜单结构和构造函数

菜单意味着选择,对菜单中的每个条目既要考虑它的键盘和鼠标选择方式,即热键、功能键、鼠标双击或单击有效选择等,还要考虑菜单的图标显示、热键显示方式、条目的激活状态以及菜单移动等因素;另外,当待选的条目数大于显示可选的条目数时,还须考虑采用滚动杆选择,因此,在菜单结构的创建过程中还要包含滚动杆驱动结构,具体叙述如下:

```
typedef struct g_scrollbar
```

```
{
    g_Rect frame,uparr,dnarr,thumb,jump;
                                     /* 组成滚动杆的各部件矩形区域 */
    char state;                       /* 决定滚动杆处于关闭还是打开状态 */
    char vorh;                         /* 决定是垂直还是水平方向的滚动杆 */
    char active;                      /* 决定滚动杆处于激活还是非激活状态 */
    char mode;                        /* 决定滚动杆区域是否设定缓冲区 */
    int min,max,cur;                  /* 滚动杆数值变化的最大最小值和当前设定值 */
    int size;                         /* 滚动杆上的滚动块跳变步长 */
    char * buffer1,* buffer2;        /* 指向滚动杆与滚动块区域的缓冲区指针 */

```

```

} g_SCROLLBAR;

typedef struct g_m_node
{
    g_WINDOW *gw; /* 指向菜单窗口区的窗口指针 */
    char *mstr; /* 指向菜单选择条目字符串指针,各条目字符串用'|'来分割 */
    char *icon; /* 指向对应菜单选择各条目所配合的位图文件名字符串指针 */
    char *hotkey; /* 指向存放菜单中各条目选项的热键字符指针 */
    char *itemon; /* 指向菜单中各条目激活状态的指针 */
    char hkm; /* 设定热键的表示方式,如一般设定、下划线设定或取消热键设定等 */
    int hkcolor; /* 设定热键的颜色 */
    int a,b; /* 设定菜单的纵横条目数 */
    int x,y; /* 菜单上光标的纵横坐标 */
    int bar; /* 菜单光标的长度 */
    char msw; /* 设定菜单各条目项光标边框的宽度 */
    char move; /* 决定菜单是否可移动 */
    int now; /* 记录着菜单上当前光标所在的选项 */
    int total; /* 记录菜单中的条目总数 */
    g_SCROLLBAR *gb; /* 指向菜单中滚动杆的指针 */
} g_MENU;

```

在以上两个结构定义完毕后,有关菜单的部分结构函数设计如下:

- ① `g_MENU *g_Menu_Create(int left,int top,int a,int b,
int htu,int htb,int wtl,int wtr,char FDX,int winmode,
char tp,int tlc,int tbc,char *title,
int msgfc,int msgbkc,char *mstr);`
- ② `int g_Menu_Open(g_MENU *gm);`
- ③ `int g_Menu_Close(g_MENU *gm);`
- ④ `int g_Menu_Kill(g_MENU *gm);`
- ⑤ `int g_Mouse_Menu_Drive(g_MENU *gm,int inkey,int mode);`

另外,对于滚动杆结构函数的设计同样有创建、打开、关闭、清除等功能函数外,还有如下功能函数:

- ① `int g_Erase_Thumb(g_SCROLLBAR *gb);`
- ② `int g_Draw_Thumb(g_SCROLLBAR *gb);`
- ③ `int g_Track_ScrollBar(g_SCROLLBAR *gb,g_Point pt,int task);`
- ④ `int g_Set_Thumb(g_SCROLLBAR *gb,int cur);`
- ⑤ `int g_Adjust_Thumb(g_SCROLLBAR *gb,int n);`

1.3 下拉式菜单的结构和构造函数

设计与构造下拉式菜单是进行较大程序开发与管理的的主要目的,同时也是程序规范化和提高开发效率的有效手段.下拉式菜单的功能是要支持多层菜单、菜单切换与搜索、功能键设定、菜单各条目的响应等,其中在DOS常规内存有限的情况下如何保存多层菜单的每一幅图形区域、键盘与鼠标同时驱动和界面响应等都是必须解决的难题.本平台实现了下拉式菜单所能提供的应有功能,并已应用在商品化软件中,下拉式菜单的结构如下:

```

typedef struct g_p_node
{
    int id;                /* 每张下拉式菜单的标识码 */
    int pull;             /* 主菜单使用,表示次要菜单必须拉下或不拉下 */
    g_MENU * gm;         /* 指向每张菜单的结构指针 */
    struct g_p_node * father; /* 以下两个结构组成了下拉式菜单的上下级关系 */
    struct g_p_node * son[MAXSON+1];
    char funkey[MAXSON+1]; /* 存放功能键识别的数组 */
} g_PULL;

```

有了以上这个结构定义,则有关菜单的部分结构函数设计如下:

- ① g_PULL * g_Pull_Root(int id,int left,int top,

int htu,int htb,int wtl,int wtr,char FDX,int winmode,

char tp,int tfc,int tbkc,char * title,

int msgfc,int msgbkc,char * mstr);
- ② g_PULL * g_Pull_Create(int id,int left,int top,

int htu,int htb,int wtl,int wtr,char FDX,int winmode,

char tp,int tfc,int tbkc,char * title,

int msgfc,int msgbkc,char * mstr);
- ③ int g_Pull_Link(g_PULL * parent,int num,g_PULL * child);
- ④ int g_Pull_Init();
- ⑤ int g_Pull_Hotkey(g_PULL * p,int item,int key);
- ⑥ int g_Pull_Drive(int inkey,int mode);
- ⑦ int g_Pull_Close(int level);
- ⑧ int g_Pull_Kill();
- ⑨ int g_Pull_Funkey(g_PULL * p,int item,int key);
- ⑩ int g_Treat_Funkey(int key);

2 图形界面平台的实现

在上述的研究中,建立了图形界面平台所必需的窗口与菜单管理工具,但从实际应用的角度出发,系统还必须具有许多配套的实用工具才能满足各种实际需要;因此,本系统不但提供了磁盘文件选择器、创建新文件窗口、磁盘文件检查、出错信息提示窗口、通用信息提示与阅读器等常用功能函数,而且还提供了多项条目选择器、数据自动选择输入器、鼠标器形状及函数库、全屏幕保护、通用执行外部程序函数、16和24点阵汉字的放大旋转及变换方向输出函数、Windows位图的读取与配套显示等辅助工具,使DOS下的编程变得格外简洁明快,代码精练.其中全屏幕保护等有关功能的实现,使能同时开启多个大窗口和进行系统大程序的开发变得非常方便,真正做到了鼠标一点到底的用户界面操作.

3 结 论

DOS下图形界面的研究是一项在有限环境下发挥最大功效和提高编程效率以及其他非

DOS 环境(如工业控制)界面设计的基础方法研究,因此,本图形界面平台的成功达到了预期的设计目标,并在商品化软件的设计应用中,在稳定性、可靠性、安全性等方面得到了验证。

参 考 文 献

- 1 高利佳编著. Turbo C 实例解析. 北京: 海洋出版社, 1993
- 2 王军政编著. Turbo C 2.0 实用高级编程技巧. 北京: 科海培训中心, 1992
- 3 袁一萍. 改变鼠标光标形状的方法. 计算机世界月刊, 1992, 68(6): 29~30
- 4 沈激. 应用 OOP 技术实现汉字显示控制. 计算机世界月刊, 1993, 83(9): 18~21
- 5 陈捍东编译, Turbo C 高级程序员编程指南. 北京: 希望公司, 1991

The Research and Realization of Graphic Window Platform

Wang Yanqiao

(Department of Geography)

Abstract This article describes how to realize graphic windows, menus, pull-down menus as well as their generating, driving and management under DOS. The basic configuratin is provided. Several problems regarding Chinese character displaying under the west and Chinese circumstance, mouse and keyboard driving at the same time, menu bitmap displaying, assistant tool support and so on have been solved. The interface platform imitating Microsoft Windows is implemented in the graphic mode under DOS.

Key words graphic window; menu; total Chinese mode; interface platform