1．扫描码和虚拟码  
　　扫描码对应着键盘上的不同键，每一个键被按下或释放时，都会产生一个唯一的扫描码作为本身的标识。扫描码依赖于具体的硬件设备，即当相同的键被按下或释放时，在不同的机器上可能产生不同的扫描码。在程序中通常使用由Windows系统定义的与具体设备无关的虚拟码。在击键产生扫描码的同时，键盘驱动程序KEYBOARD.DRV截取键的扫描码，然后将其翻译成对应的虚拟码，再将扫描码和虚拟码一齐编码形成键盘消息。所以，最后发送到消息队列的键盘消息中，既包含了扫描码又包含了虚拟码。  
　　经常使用的虚拟码在WINDOWS．H文件中定义，常用虚拟码的数值、常量符号和含义如表所示。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 取值 | 常量符号 | 含义 |  | 取值 | 常量符号 | 含义 |
| 01 | VK\_LBUTTON | 鼠标左键 |  | 2A | -- | OEM保留 |
| 02 | VK\_RBUTTON | 鼠标右键 |  | 2B | VK\_EXECUTE | Execute键 |
| 03 | VK\_CANCEL | Break中断键 |  | 2C | VK\_SNAPSHOT | Print |
| 04 | VK\_MBUTTON | 鼠标中键 |  | 2D | VK\_INSERT | Insert键 |
| 05-07 | -- | 未定义 |  | 2E | VK\_DELETE | Delete键 |
| 08 | VK\_BACK | (BackSpace)键 |  | 2F | VK\_HELP | Help键 |
| 09 | VK\_TAB | Tab键 |  | 30-39 | VK\_0-VK\_9 | 数字键0-9 |
| 0A-0B | -- | 未定义 |  | 3A-40 | -- | 未定义 |
| 0C | VK\_CLEAR | Clear键 |  | 41-5A | VK\_A-VK\_Z | 字母键A-Z |
| 0D | VK\_RETURN | Enter键 |  | 5B-5F | -- | 未定义 |
| 0E-0F | -- | 未定义 |  | 60-69 | VK\_NUMPAD0-  VK\_NUMPAD9 | 小键盘数字键0-9 |
| 10 | VK\_SHIFT | Shift键 |  | 6A | VK\_MULTIPLY | \*(乘号)键 |
| 11 | VK\_CONTROL | Ctrl键 |  | 6B | VK\_ADD | +(加号)键 |
| 12 | VK\_MENU | Alt键 |  | 6C | VK\_SEPAPATOR | 分隔符键 |
| 13 | VK\_PAUSE | Pause键 |  | 6E | VK\_SUBTRACT | -(减号)键 |
| 14 | VK\_CAPTIAL | CapsLock键 |  | 6F | VK\_DECIMAL | .(小数点)键 |
| 15-19 | -- | 汉字系统保留 |  | 70-87 | VK\_DIVIDE | /(除号)键 |
| 1A | -- | 未定义 |  | 88-8F | VK\_F1-VK\_F24 | F1-F24功能键 |
| 1B | VK\_ESCAPE | Esc键 |  | 90 | VK\_NUMBERLOCK | Number |
| 1C-1F | -- | 汉字系统保留 |  | 91 | VK\_SCROLL | Scroll |
| 20 | VK\_SPACE | 空格键 |  | 92-B9 | -- | 未定义 |
| 21 | VK\_PRIOR | PageUp键 |  | BA-C0 | -- | OEM保留 |
| 22 | VK\_NEXT | PageDown键 |  | C1-DA | -- | 未定义 |
| 23 | VK\_END | End键 |  | DB\_E4 | -- | OEM保留 |
| 24 | VK\_HOME | Home键 |  | E5 | -- | 未定义 |
| 25 | VK\_LEFT | ←(Left |  | E6 | -- | OEM保留 |
| 26 | VK\_UP | ↑(Up |  | E7-E8 | -- | 未定义 |
| 27 | VK\_RIGHT | →(Right |  | E9-F5 | -- | OEM保留 |
| 28 | VK\_DOWN | ↓(Down |  | F6-FE | -- | 未定义 |
| 29 | VK\_SELECT | Select键 |  |

2．键盘消息  
　　键盘消息分为系统键消息和非系统键消息。系统键消息是指由Aft键和其他键组合而产生的按键消息。当系统键被按下时产生WM\_ SYSKEYDOWN消息，当系统键被释放时产生WM\_SYSKEYUP消息。 Aft键与其他键形成的组合键通常用于对程序菜单和系统菜单进行选择，或用于在不同的程序之间进行切换。因此，系统键消息应该交由Windows进行处理，用户所编制的程序一般不处理系统键消息，而是将这些消息交由DefWindowProc函数进行处理。如果用户想对系统键消息进行处理，应该在处理完这些消息后，再将其发送给DefWindowProc函数，使得Windows系统能够正常工作。  
　　某些击键消息可以被转换成字符消息，例如字母键、数字键等。而有些键只能产生按键消息而没有字符消息，例如 Shift键、Insert键等。消息循环中的 TranslateMessage函数可以实现从击键消息向字符消息的转化。当GetMessage函数捕获一个WM\_SYSKEYDOWN消息或 WM\_KEYDOWN消息后，TranslateMessage函数判断产生该消息的键是否能够被转换成字符消息，如果能，就将该消息转换成字符消息，再通过DispatchMessape函数将转换后的字符消息发送到消息队列中去。字符消息共有以下四种，如表所示。  
  
字符 系统字符 非系统字符  
普通字符 WM\_SYSCHAR WM\_CHAR  
死字符 WM\_SYSDEADCHAR WM\_DEADCHAR  
  
　　其中死字符是由某些特殊键盘上的按键所造成的，Windows一般忽略死字符所产生的消息。  
　　Windows的消息一般是通过一个MSG结构体变量传送给消息处理函数的。对于键盘消息， MSG结构体变量的各个域中较重要的是lParam域和 wParam域。wParam域用于保存按键的虚拟键代码或字符的ASCII码。对于非字符消息，wParam域保存按键的虚拟健代码；对于字符消息， wParam域不保存字符的ASCII码。lParam域则用于保存击键时产生的附加信息，实际上一个32位的lParam变量被分为六部分，记录了以下相关信息：重复次数、OEM扫描码、扩展键标志、关联键标志、前一击键状态和转换状态。lParam域各位的含义如表所示。  
  
位数 含义  
0-15 击键重复次数累加  
16-23 OEM扫描码  
24 是否为扩展键  
25-28 未定义  
29 是否便用关联键，及Alt键是否同时按下。  
30 前一次击键状态，0表示该键前一次状态为抬起，1表示前一次状态为按下  
31 转换状态  
  
　　按键的次序不同，产生的消息也不相同。例如，按下并释放1键，读过程依次产生如表所示三条消息。按下1键所产生的消息和wParam的取值：  
消息 message 和 wParam变量取值：  
WM\_KEYDOWN 虚拟码1 pMsg->message==WM\_KEYDOWN pMsg->wParam==VK\_1  
WM\_CHAR ASCII码“1” pMsg->message==WM\_CHAR pMsg->wParam==’1’  
WM\_KEYUP 虚拟码1 pMsg->message==WM\_KEYUP pMsg->wParam==VK\_1  
  
　　如果按下Shift键后再按下1键并释放，则依次产生如表所示的消息。按下 Shift键后按 1健所产生的消息和 wParam的取值：  
消息 message 和 wParam变量取值：  
WM\_KEYDOWN 虚拟码 VK\_SHIFT  
WM\_KEYDOWN 虚拟码 VK\_1  
WM\_CHAR ASCII码　“1”  
WM\_KEYUP 虚拟码 VK\_1  
WM\_KEYUP 虚拟码 VK\_SHIFT