**《Windows应用程序原理》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | |  | | |  |  |
|  | |  | | |  |  |
| **实验名称** | 实验一 编写Windows窗口应用程序实现多行多列文本输出 | | | | | |
| **实验时间** | 2024.3.26 | | **实验地点** | DS3304 | | |
| **实验性质** | | | **□验证性 ☑设计性 □综合性** | | | |
| 一、实验目的  1.熟悉Windows窗口应用程序开发环境和工具。  2.学习如何创建窗口应用程序界面。  3.掌握在窗口应用程序中实现文本输出的基本技能。  4.理解如何管理和显示多行多列文本内容。  5.培养对Windows应用程序开发的兴趣和技能。 | | | | | | |
| 二、实验项目内容  1. 设计Windows窗口应用程序，在窗口中输出多行多列文本；  2. 支持上下滚动条；  3. 能根据窗口大小调整文本内容 | | | | | | |
| 1. 实验过程或算法（源程序）   1.SYSMETS.H  #define NUMLINES ((int) (sizeof sysmetrics / sizeof sysmetrics [0]))  struct  {  int iIndex;  TCHAR\* szLabel;  TCHAR\* szDesc;  }  sysmetrics[] =  {  SM\_CXSCREEN, TEXT("SM\_CXSCREEN"),  TEXT("Screen width in pixels"),  SM\_CYSCREEN, TEXT("SM\_CYSCREEN"),  TEXT("Screen height in pixels"),  SM\_CXVSCROLL, TEXT("SM\_CXVSCROLL"),  TEXT("Vertical scroll width"),  SM\_CYHSCROLL, TEXT("SM\_CYHSCROLL"),  TEXT("Horizontal scroll height"),  SM\_CYCAPTION, TEXT("SM\_CYCAPTION"),  TEXT("Caption bar height"),  SM\_CXBORDER, TEXT("SM\_CXBORDER"),  TEXT("Window border width"),  SM\_CYBORDER, TEXT("SM\_CYBORDER"),  TEXT("Window border height"),  SM\_CXFIXEDFRAME, TEXT("SM\_CXFIXEDFRAME"),  TEXT("Dialog window frame width"),  SM\_CYFIXEDFRAME, TEXT("SM\_CYFIXEDFRAME"),  TEXT("Dialog window frame height"),  SM\_CYVTHUMB, TEXT("SM\_CYVTHUMB"),  TEXT("Vertical scroll thumb height"),  SM\_CXHTHUMB, TEXT("SM\_CXHTHUMB"),  TEXT("Horizontal scroll thumb width"),  SM\_CXICON, TEXT("SM\_CXICON"),  TEXT("Icon width"),  SM\_CYICON, TEXT("SM\_CYICON"),  TEXT("Icon height"),  SM\_CXCURSOR, TEXT("SM\_CXCURSOR"),  TEXT("Cursor width"),  SM\_CYCURSOR, TEXT("SM\_CYCURSOR"),  TEXT("Cursor height"),  SM\_CYMENU, TEXT("SM\_CYMENU"),  TEXT("Menu bar height"),  SM\_CXFULLSCREEN, TEXT("SM\_CXFULLSCREEN"),  TEXT("Full screen client area width"),  SM\_CYFULLSCREEN, TEXT("SM\_CYFULLSCREEN"),  TEXT("Full screen client area height"),  SM\_CYKANJIWINDOW, TEXT("SM\_CYKANJIWINDOW"),  TEXT("Kanji window height"),  SM\_MOUSEPRESENT, TEXT("SM\_MOUSEPRESENT"),  TEXT("Mouse present flag"),  SM\_CYVSCROLL, TEXT("SM\_CYVSCROLL"),  TEXT("Vertical scroll arrow height"),  SM\_CXHSCROLL, TEXT("SM\_CXHSCROLL"),  TEXT("Horizontal scroll arrow width"),  SM\_DEBUG, TEXT("SM\_DEBUG"),  TEXT("Debug version flag"),  SM\_SWAPBUTTON, TEXT("SM\_SWAPBUTTON"),  TEXT("Mouse buttons swapped flag"),  SM\_CXMIN, TEXT("SM\_CXMIN"),  TEXT("Minimum window width"),  SM\_CYMIN, TEXT("SM\_CYMIN"),  TEXT("Minimum window height"),  SM\_CXSIZE, TEXT("SM\_CXSIZE"),  TEXT("Min/Max/Close button width"),  SM\_CYSIZE, TEXT("SM\_CYSIZE"),  TEXT("Min/Max/Close button height"),  SM\_CXSIZEFRAME, TEXT("SM\_CXSIZEFRAME"),  TEXT("Window sizing frame width"),  SM\_CYSIZEFRAME, TEXT("SM\_CYSIZEFRAME"),  TEXT("Window sizing frame height"),  SM\_CXMINTRACK, TEXT("SM\_CXMINTRACK"),  TEXT("Minimum window tracking width"),  SM\_CYMINTRACK, TEXT("SM\_CYMINTRACK"),  TEXT("Minimum window tracking height"),  SM\_CXDOUBLECLK, TEXT("SM\_CXDOUBLECLK"),  TEXT("Double click x tolerance"),  SM\_CYDOUBLECLK, TEXT("SM\_CYDOUBLECLK"),  TEXT("Double click y tolerance"),  SM\_CXICONSPACING, TEXT("SM\_CXICONSPACING"),  TEXT("Horizontal icon spacing"),  SM\_CYICONSPACING, TEXT("SM\_CYICONSPACING"),  TEXT("Vertical icon spacing"),  SM\_MENUDROPALIGNMENT, TEXT("SM\_MENUDROPALIGNMENT"),  TEXT("Left or right menu drop"),  SM\_PENWINDOWS, TEXT("SM\_PENWINDOWS"),  TEXT("Pen extensions installed"),  SM\_DBCSENABLED, TEXT("SM\_DBCSENABLED"),  TEXT("Double-Byte Char Set enabled"),  SM\_CMOUSEBUTTONS, TEXT("SM\_CMOUSEBUTTONS"),  TEXT("Number of mouse buttons"),  SM\_SECURE, TEXT("SM\_SECURE"),  TEXT("Security present flag"),  SM\_CXEDGE, TEXT("SM\_CXEDGE"),  TEXT("3-D border width"),  SM\_CYEDGE, TEXT("SM\_CYEDGE"),  TEXT("3-D border height"),  SM\_CXMINSPACING, TEXT("SM\_CXMINSPACING"),  TEXT("Minimized window spacing width"),  SM\_CYMINSPACING, TEXT("SM\_CYMINSPACING"),  TEXT("Minimized window spacing height"),  SM\_CXSMICON, TEXT("SM\_CXSMICON"),  TEXT("Small icon width"),  SM\_CYSMICON, TEXT("SM\_CYSMICON"),  TEXT("Small icon height"),  SM\_CYSMCAPTION, TEXT("SM\_CYSMCAPTION"),  TEXT("Small caption height"),  SM\_CXSMSIZE, TEXT("SM\_CXSMSIZE"),  TEXT("Small caption button width"),  SM\_CYSMSIZE, TEXT("SM\_CYSMSIZE"),  TEXT("Small caption button height"),  SM\_CXMENUSIZE, TEXT("SM\_CXMENUSIZE"),  TEXT("Menu bar button width"),  SM\_CYMENUSIZE, TEXT("SM\_CYMENUSIZE"),  TEXT("Menu bar button height"),  SM\_ARRANGE, TEXT("SM\_ARRANGE"),  TEXT("How minimized windows arranged"),  SM\_CXMINIMIZED, TEXT("SM\_CXMINIMIZED"),  TEXT("Minimized window width"),  SM\_CYMINIMIZED, TEXT("SM\_CYMINIMIZED"),  TEXT("Minimized window height"),  SM\_CXMAXTRACK, TEXT("SM\_CXMAXTRACK"),  TEXT("Maximum dragable width"),  SM\_CYMAXTRACK, TEXT("SM\_CYMAXTRACK"),  TEXT("Maximum dragable height"),  SM\_CXMAXIMIZED, TEXT("SM\_CXMAXIMIZED"),  TEXT("Width of maximized window"),  SM\_CYMAXIMIZED, TEXT("SM\_CYMAXIMIZED"),  TEXT("Height of maximized window"),  SM\_NETWORK, TEXT("SM\_NETWORK"),  TEXT("Network present flag"),  SM\_CLEANBOOT, TEXT("SM\_CLEANBOOT"),  TEXT("How system was booted"),  SM\_CXDRAG, TEXT("SM\_CXDRAG"),  TEXT("Avoid drag x tolerance"),  SM\_CYDRAG, TEXT("SM\_CYDRAG"),  TEXT("Avoid drag y tolerance"),  SM\_SHOWSOUNDS, TEXT("SM\_SHOWSOUNDS"),  TEXT("Present sounds visually"),  SM\_CXMENUCHECK, TEXT("SM\_CXMENUCHECK"),  TEXT("Menu check-mark width"),  SM\_CYMENUCHECK, TEXT("SM\_CYMENUCHECK"),  TEXT("Menu check-mark height"),  SM\_SLOWMACHINE, TEXT("SM\_SLOWMACHINE"),  TEXT("Slow processor flag"),  SM\_MIDEASTENABLED, TEXT("SM\_MIDEASTENABLED"),  TEXT("Hebrew and Arabic enabled flag"),  SM\_MOUSEWHEELPRESENT, TEXT("SM\_MOUSEWHEELPRESENT"),  TEXT("Mouse wheel present flag"),  SM\_XVIRTUALSCREEN, TEXT("SM\_XVIRTUALSCREEN"),  TEXT("Virtual screen x origin"),  SM\_YVIRTUALSCREEN, TEXT("SM\_YVIRTUALSCREEN"),  TEXT("Virtual screen y origin"),  SM\_CXVIRTUALSCREEN, TEXT("SM\_CXVIRTUALSCREEN"),  TEXT("Virtual screen width"),  SM\_CYVIRTUALSCREEN, TEXT("SM\_CYVIRTUALSCREEN"),  TEXT("Virtual screen height"),  SM\_CMONITORS, TEXT("SM\_CMONITORS"),  TEXT("Number of monitors"),  SM\_SAMEDISPLAYFORMAT, TEXT("SM\_SAMEDISPLAYFORMAT"),  TEXT("Same color format flag")  };  2.sysmets3  #define WINVER 0x0500  #include <windows.h>  #include "SYSMETS.H"  LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam);  int WINAPI WinMain(  HINSTANCE hInstance,  HINSTANCE hPrevInstance,  LPSTR IpszCmdLine,  int nCmdShow  )//注册窗口类  {  static TCHAR szAppName[] = TEXT("sysmets3");  WNDCLASS wndclass;  HWND hWnd;  MSG msg;  wndclass.style = CS\_HREDRAW | CS\_VREDRAW;//窗口高度或者宽度发生变化时，窗口根据窗口大小重新绘制  wndclass.lpfnWndProc = WndProc;//窗口回调函数，记录窗口进行的操作并返回操作导致的结果  wndclass.cbClsExtra = 0;//窗口类的额外空间  wndclass.cbWndExtra = 0;//窗口实例的额外空间  wndclass.hInstance = hInstance;//窗口实例的句柄  wndclass.hIcon = LoadIcon(NULL, IDI\_APPLICATION);//  wndclass.hCursor = LoadCursor(NULL, IDC\_ARROW);//鼠标光标句柄  wndclass.hbrBackground = (HBRUSH)GetStockObject(WHITE\_BRUSH);//窗口背景颜色句柄  wndclass.lpszMenuName = NULL;  wndclass.lpszClassName = szAppName;//菜单名  RegisterClass(&wndclass);  hWnd = CreateWindow(//创建窗口  szAppName,  TEXT("get SYSTEM Metrics Version 3"),//窗口标题  WS\_OVERLAPPEDWINDOW | WS\_HSCROLL | WS\_VSCROLL,//同时加上水平和垂直的滚动条  CW\_USEDEFAULT, CW\_USEDEFAULT,//初始位置，已经设定好的常数  CW\_USEDEFAULT, CW\_USEDEFAULT,//初始尺寸  NULL, NULL, hInstance, NULL  );  ShowWindow(hWnd, nCmdShow);//显示创建好的窗口  UpdateWindow(hWnd);//刷新句柄  //关闭窗口，结束进程  while (GetMessage(&msg, NULL, 0, 0))  {  TranslateMessage(&msg);  DispatchMessage(&msg);  }  return msg.wParam;  }  LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam) {  static int cxChar, cxCaps, cyChar;  static int cxClient, cyClient;//客户区的宽度和高度  static int nxMax;//水平方向最多有多少个位置  TCHAR szBuffer[30];  HDC hDc;  PAINTSTRUCT ps;  TEXTMETRIC tm;  SCROLLINFO si;//滚条信息结构  int iPreVertPos, iPreHorzPos;//之前滑块所处垂直和水平位置  int iVertPos, iHorzPos;//滑块当前位置  int iPaintBegin, iPaintEnd;//重绘时在无效矩形内的文本的起始行号和结束行号  int i, x, y;  switch (message) {  case WM\_CREATE:  hDc = GetDC(hWnd); // 获取环境句柄  GetTextMetrics(hDc, &tm);  cxChar = tm.tmAveCharWidth;  cxCaps = (tm.tmPitchAndFamily & 1 ? 3 : 2) \* cxChar / 2;  cyChar = tm.tmExternalLeading + tm.tmHeight;  nxMax = 2 + (22 \* cxCaps + 40 \* cxChar) / cxChar;  ReleaseDC(hWnd, hDc); // 释放句柄  return 0;  case WM\_SIZE:  //获取每次重绘后的屏幕大小  cxClient = LOWORD(lParam); //获取当前窗口的大小  cyClient = HIWORD(lParam);  //设置垂直滚动条的范围和页面大小  si.cbSize = sizeof(SCROLLINFO); //为了更好的兼容版本  si.fMask = SIF\_RANGE | SIF\_PAGE;  si.nMin = 0;  si.nMax = NUMLINES - 1;  si.nPage = cyClient / cyChar;  SetScrollInfo(hWnd, SB\_VERT, &si, TRUE);//设置后滑块大小会自动改变  //不需要设置nPos，因为要保留窗口大小改变之前滑块的位置  si.cbSize = sizeof(SCROLLINFO);  si.fMask = SIF\_RANGE | SIF\_PAGE;  si.nMin = 0;  si.nMax = nxMax;  si.nPage = cxClient / cxChar;  SetScrollInfo(hWnd, SB\_HORZ, &si, TRUE);//同理设置水平滑块的大小  return 0;  case WM\_VSCROLL://获得垂直滚动条的信息  si.fMask = SIF\_ALL;  GetScrollInfo(hWnd, SB\_VERT, &si);  iPreVertPos = si.nPos;//记录之前的位置  switch (LOWORD(wParam))  {  case SB\_TOP://触顶。触顶和触底都是屏幕上滑块显示的位置而不是实际位置  si.nPos = si.nMin;  break;  case SB\_BOTTOM://触底，只有在键盘接口后才有效，只用于响应键盘信息  si.nPos = si.nMax;  break;  //后续SB\_LEFT\SB\_RIGHT也一样  case SB\_LINEUP:  si.nPos--;  break;  case SB\_LINEDOWN:  si.nPos++;  break;  case SB\_PAGEUP:  si.nPos -= si.nPage;  break;  case SB\_PAGEDOWN:  si.nPos += si.nPage;  break;  case SB\_THUMBTRACK:  si.nPos = si.nTrackPos;  break;  }  si.cbSize = sizeof(SCROLLINFO);  si.fMask = SIF\_POS;  SetScrollInfo(hWnd, SB\_VERT, &si, TRUE);  GetScrollInfo(hWnd, SB\_VERT, &si);  if (si.nPos != iPreVertPos)//只有在实际位置改变的情况下对客户区内容进行滚动  {  ScrollWindow(hWnd, 0, cyChar \* (iPreVertPos - si.nPos), NULL, NULL);  //最后两个参数表示滚动的客户区范围和剪裁矩形，null表示对整个客户区进行滚动  }  return 0;  case WM\_HSCROLL://响应垂直滚条消息，同理  si.cbSize = sizeof(SCROLLINFO);  si.fMask = SIF\_ALL;  GetScrollInfo(hWnd, SB\_HORZ, &si);  iPreHorzPos = si.nPos;  switch (LOWORD(wParam))  {  case SB\_LEFT:  si.nPos = si.nMin;  break;  case SB\_RIGHT:  si.nPos = si.nMax;  break;  case SB\_LINELEFT:  si.nPos--;  break;  case SB\_LINERIGHT:  si.nPos++;  break;  case SB\_PAGELEFT:  si.nPos -= si.nPage;  break;  case SB\_PAGERIGHT:  si.nPos += si.nPage;  break;  case SB\_THUMBTRACK:  si.nPos = si.nTrackPos;  break;  }  si.cbSize = sizeof(SCROLLINFO);  si.fMask = SIF\_POS;  SetScrollInfo(hWnd, SB\_HORZ, &si, TRUE);  GetScrollInfo(hWnd, SB\_HORZ, &si);  if (iPreHorzPos != si.nPos)  {  ScrollWindow(hWnd, 0, cxChar \* (iPreHorzPos - si.nPos), 0,NULL, NULL);  }  return 0;  case WM\_PAINT:  hDc = BeginPaint(hWnd, &ps);  si.fMask = SIF\_POS;  GetScrollInfo(hWnd, SB\_VERT, &si);  iVertPos = si.nPos;  GetScrollInfo(hWnd, SB\_HORZ, &si);  iHorzPos = si.nPos;  iPaintBegin = max(0, iVertPos + ps.rcPaint.top / cyChar);//重绘的起始行号  iPaintEnd = min(NUMLINES - 1, iVertPos + ps.rcPaint.bottom / cyChar);//结束行号  for (i = iPaintBegin; i <= iPaintEnd; i++) {  //水平方向无法逐字输出，因此依赖windows对客户区外的部分裁剪  x = cxChar \* (1 - iHorzPos);//起始留一个空白，美观  y = cyChar \* (i - iVertPos);  TextOut(hDc, x, y, sysmetrics[i].szLabel, lstrlen(sysmetrics[i].szLabel));  TextOut(hDc, x + 22 \* cxCaps, y, sysmetrics[i].szDesc, lstrlen(sysmetrics[i].szDesc));  SetTextAlign(hDc, TA\_RIGHT | TA\_TOP);  TextOut(hDc, x + 22 \* cxCaps + 40 \* cxChar, y, szBuffer, wsprintf(szBuffer, TEXT("%5d"), GetSystemMetrics(sysmetrics[i].iIndex)));  SetTextAlign(hDc, TA\_LEFT | TA\_TOP);  }  EndPaint(hWnd, &ps);  return 0;  case WM\_DESTROY:  PostQuitMessage(0);  return 0;  }  return DefWindowProc(hWnd, message, wParam, lParam);  } | | | | | | |
| 1. 实验结果及分析和（或）源程序调试过程     图1 窗口弹出并输出多行多列文本  图2 滚动条上下滑动    图3 根据窗口大小调整文本信息和滚动条大小    图4 根据窗口大小改变调整滚动条    图5 窗口水平长度较小时裁剪客户区外的内容并显示垂直和水平滚动条 | | | | | | |