notepad--

制作者 Doxygen 1.8.13

Contents

Chapter 1

说明文档

基于MFC的文本编辑器

目标

1. 不使用MFC文本框控件，实现一个简单的文本编辑器
2. 最低要求如下：
    - 带有图形界面
    - 能够比较方便的给不同的字符设置不同的字体、颜色和字号
    - 可以设置字与字之间的间距
3. 另外还实现了以下功能
    - 保存、打开自定义格式的文件
    - 设置对齐方式，包括左对齐、居中对齐、右对齐、分散对齐
    - 设置行间距

使用方法

1. 常规操作与普通文本编辑器相同
2. 设置字体、颜色、字号：
    - 选中想要改变的文本内容
    - 点击编辑菜单栏，点击字体，将会弹出选择字体的窗口
    - 选择想要设置的格式即可
3. 设置对齐方式：
    - 如果只需要设置单行的格式，请直接将光标移到该行
    - 如果需要设置多行的格式，请选中多行
    - 点击编辑菜单栏，点击对齐方式
    - 在弹出的右侧菜单栏中选择需要的对齐方式
4. 设置行间距、字间距：
    - 选中需要设置的文本内容

- 点击编辑菜单栏，点击段落
- 在弹出的窗口中输入需要设置的行间距以及字间距
- 注意单位为像素

5. 打开与保存
   - 首先注意，本文本编辑器只支持自定义格式的（后缀名为\.orz）文件，其他格式的文件一律无法打开
   - 打开：点击文件菜单栏，点击打开，在文件浏览器中选择想要打开的文件，点击确定即可
   - 保存：点击文件菜单栏，点击保存，在文件浏览器中选择想要保存的目录，点击确定即可

构件库

详情请参见Class List

UI界面的交互实现方法说明

1. 设置字体字号颜色
   通过CFontDialog窗口类，创建选择字体的窗口
   用户选择字体以及颜色后，获取字体信息LOGFONT以及颜色信息REFCOLOR
   通过set_select_font与set_select_col函数将字体以及颜色信息传递给后端

2. 设置字间距行间距
   通过M_PARA_DIA对话框，获取用户设置的字间距与行间距的值
   通过set_select_lspace与set_select_cspace函数将字间距与行间距传给后端

3. 保存、打开文件
   通过CFileDialog类，返回用户需要保存或者打开文件的路径
   通过#与#函数将路径作为字符串传给后端

4. 设置对齐方式
   直接通过消息响应函数，调用set_select_align函数,设置对齐方式

后端（kernal）实现方法说明

1. 抽象
   - 把画在屏幕上的字符看作一个矩形
   - 用链表从前往后储存字符

2. 计算绘制信息
   - 遍历文本链表，进行预处理
   - 遍历文本链表，把链表分成很多行

  - 当某行的字符宽度之和(tot_width)将要超过页面宽度(pagewidth)时，开一个新行
  - 当遇到换行符时，开一格新行

- 从上往下遍历每一行，根据相关信息计算该行内的n个字符矩形的位置
  - 如果该行是默认对齐方式或者左对齐，那么x坐标从0开始，每个矩形宽度加上字间距，从左往右计算
  - 如果该行是右对齐，那么x坐标从pagewidth-tot_width开始，每个矩形宽度加上字间距，从左往右计算
  - 如果对齐方式是居中对齐，那么x坐标从(pagewidth-tot_width)/2开始，每个矩形宽度加上字间距，从左往右计算
  - 如果对齐方式是分散对齐，那么x坐标从0开始，每个矩形宽度加上字间距和delta=(pagewidth-tot_width)/n

UI及交互部分遇到的问题及解决方案

1. 绘图过程闪烁的问题
   问题描述：使用过程中每当需要屏幕刷新的时候，画面会不停的闪烁
   可能原因：每次调用OnPaint函数的时候都会先将屏幕清空，由于绘制屏幕仍然需要一定的时间，所以每次重绘都会出现闪烁的情况
   解决方案：使用双缓冲绘图，分为两个部分：
   (1)重载BOOL CChildView::OnEraseBkgnd(CDC* pDC)函数，使得画面不会被擦除
   (2)先在使用内存DC上画图，画好后直接调用BitBlt函数将画面传给物理DC

2. 滚动条位置以及大小的问题
   问题描述：滚动条大小以及位置随着窗口大小改变而随机变动
   可能原因：由于滚动条控件需要手动设置大小以及位置，所以当改变窗口大小的时候会出现滚动条位置偏移的情况
   解决方案：设置两个变量：m_client_cy记录页面长度，scrolledpix记录页面相对于窗口上端的偏移量
   每次重绘的时候通过这两个量来重新计算滚动条的大小以及位置

3. 解析用户传入的键盘消息
   问题描述：根据官方文档，键盘消息的对应关系很复杂，需要解析传入的参数才能获得字符信息
   解决方案：使用void CChildView::OnChar(UINT nChar, UINT nRepCnt, UINT nFlags)的消息响应函数
   系统将键盘消息处理后返回用户输入的字符

4. 连按消息的问题
   问题描述：通常连按一个按键，被认为是连续输入字符
   解决方案：在void CChildView::OnKeyDown(UINT nChar, UINT nRepCnt, UINT nFlags)函数中
   nRepCnt参数是用于标记连按消息的次数的，循环相应的次数即可实现连续输入

5. 获取字体的像素大小的问题
   问题描述：LOGFONT结构体的成员lfHeight与lfWidth不是像素高度
   解决方案：通过公式pixHeight = -(lfHeight * 72) / GetDeviceCaps(hDC,LOGPIXELSY)转换为像素高度
   参考资料：MSDN上关于LOGFONT的说明

后端（**kernal**）遇到的问题及解决方案

1. 空行的问题
   问题描述：链表中换行符只用作开启一行的标识符，如果用户连续键入两个换行，第二个换行会变成"空行"而无法在屏幕上打出来
   解决方案：在预处理中，先遍历链表，在相邻两个换行符之间插入一个`S.width=0`，`size=-1`的字符，避免"空行"的产生

2. 使用"打开"命令后单击屏幕会崩溃的问题
   问题描述：添加"打开"模块后，用户在界面上单击会导致崩溃
   问题原因：执行"打开"命令后，后端的选中信息会丢失，而UI当作用户"完成"了一次选中操作调用了`end_select`函数，导致内存无效访问
   解决方案：在对选中区域执行任何操作前，先检查是否存在选中区域

3. "打开"命令对于某些字体无法执行的问题
   问题描述：当打开包含字体名称(`lfFaceName`)为多个单词的文件时，"打开"操作会失败
   问题原因：字体名称是`wchar_t`类型的,最初我用的scanf("%ls")来读取字体名称，这样就只能读到字体名称的第一个单词，而且我没有找到针对`wchar_t`数组的getline函数。
   解决方案：手写针对`wchar_t`的`si_getline`函数

测试用例

1. `align.orz`：测试四种对齐方式
2. `keepcalm.orz`：测试设置字体、字号、颜色
3. `linechaspace.orz`：测试设置行间距、字间距

总结

1. 程序具有一定的可使用性，实现了基本目标，在数据规模不太大的情况下可以达到较好效果
2. 当数据规模较大时（超过50000字符），会出现明显的卡顿现象，所以程序仍然有很大改进空间

   - 可以通过每次只计算、绘制当前屏幕附近的字符来解决这个问题

作者信息

- kernal部分作者：李仁杰
  - 学院：清华大学软件学院
  - 班级：软件62
  - 学号：2016013271
  - e-mail: ShadowIterator@hotmail.com
  - 电话：18801005736

- UI及交互部分作者：洪方舟
  - 学院：清华大学软件学院
  - 班级：软件62
  - 学号： 2016013259
  - e-mail: hongfz16@163.com
  - 电话：15062727188

Chapter 2

继承关系索引

2.1 类继承关系

此继承关系列表按字典顺序粗略的排序:

Chapter 3

类索引

3.1 类列表

这里列出了所有类、结构、联合以及接口定义等，并附带简要说明:

Chapter 4

文件索引

4.1 文件列表

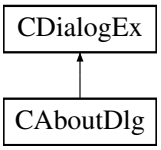
这里列出了所有文档化的文件，并附带简要说明:

Chapter 5

类说明

5.1 CAboutDlg类 参考

类 CAboutDlg 继承关系图:



Protected 成员函数

- virtual void **DoDataExchange** (CDataExchange *pDX)

该类的文档由以下文件生成:

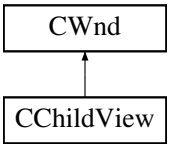
- Notepad/Notepad.cpp

5.2 CChildView类 参考

子视窗类CChildView
子视窗是程序中客户区部分，即文本编辑的画布部分 -此类继承自CWnd

```
#include <ChildView.h>
```

类 CChildView 继承关系图:



Public 成员函数

- CChildView ()

  *CChildView*类构造函数
- virtual ~CChildView ()

  *CChildView*类析构函数
- void m_paintText (CDC &dc)

  绘制所有文字以及背景色的函数
- void **m_paintText** (CPaintDC &dc)
- void m_paintCur (CDC &dc)

  绘制光标
- void **m_paintCur** (CPaintDC &dc)
- afx_msg void OnChar (UINT nChar, UINT nRepCnt, UINT nFlags)

  响应发送文字消息的函数
  响应*ON_WM_CHAR*消息
- bool is_input (UINT nChar)

  判断是否为正常字符范围的输入
- void m_changed ()

  当文本内容根据用户的操作有变化的时候调用
  操作如下
- afx_msg void OnLButtonDown (UINT nFlags, CPoint point)

  响应鼠标左键按下消息的函数
  当程序收到*ON_WM_LBUTTONDOWN*消息时调用该函数
  主要处理两种情况

  1. 当没有文字被选中的时候，将光标位置调整为当前点击位置，进入选择状态
  2. 没有文字被选中的时候,不作任何动作
- afx_msg void OnLButtonUp (UINT nFlags, CPoint point)

  响应鼠标左键抬起消息的函数
  当程序收到*ON_WM_LBUTTONUP*消息时调用该函数
  主要处理两种情况

  1. 当没有文字被选中的时候，将光标位置调整为当前抬起的位置，结束选中状态。如果鼠标左键在按下与抬起之间有移动，则会有文字被选中
  2. 当有文字被选中的时候，将光标位置更新，并且取消选中状态
- afx_msg void OnMouseMove (UINT nFlags, CPoint point)

  响应鼠标移动消息的函数
  当程序收到*ON_WM_MOUSEMOVE*消息时调用该函数
  进入函数之后需要鼠标左键被按下的*flag*为*true*才会做出操作
  操作主要为更新选择文字过程中的光标位置以及画面
- afx_msg void OnKeyDown (UINT nChar, UINT nRepCnt, UINT nFlags)

  键盘按下消息响应函数
  此函数内部对*nChar*有判断，也就是说此函数只处理按下上下左右键的消息
- afx_msg void OnKeyUp (UINT nChar, UINT nRepCnt, UINT nFlags)

  键盘抬起消息响应函数
  此函数内部对*nChar*有判断，也就是说此函数只处理按下上下左右键的消息
- afx_msg BOOL OnEraseBkgnd (CDC *pDC)

  重载擦除屏幕函数，使它的返回值为*false*，即不擦除，配合双缓冲绘图使用
- void curchanged ()

  重新计算页面长度以及滚动条位置

Public 属性

- SITEXT * m_text
  自定义的文本文字类成员
- CMainFrame * mainframep
  指向主窗口的指针
- bool need_recompute
  需要重新计算页面布局的*flag*

Protected 成员函数

- virtual BOOL PreCreateWindow (CREATESTRUCT &cs)
  创建窗口的预处理函数
- afx_msg void OnPaint ()
  绘图函数
  当程序接受到*ON_WM_PAINT*的消息的时候就会调用该函数
  该函数所做工作包括

5.2.1 详细描述

子视窗类CChildView
子视窗是程序中客户区部分，即文本编辑的画布部分 -此类继承自CWnd

5.2.2 成员函数说明

5.2.2.1 is_input()

```
bool CChildView::is_input (
            UINT nChar )  [inline]
```

判断是否为正常字符范围的输入

参数

in	*nChar*	

参见

OnChar

返回值

true	是正常范围的字符
false	不是正常范围的字符

5.2.2.2 m_changed()

```
void CChildView::m_changed ( )  [inline]
```

当文本内容根据用户的操作有变化的时候调用
操作如下

- 设置画面变化的flag(即text_changed_f)为true
- 调用Invalidate(false)函数向程序发送重绘消息

5.2.2.3 m_paintCur()

```
void CChildView::m_paintCur (
            CDC & dc )
```

绘制光标

参数

in	*dc*	当前绘图区的句柄

注解

仅在OnPaint函数中调用

5.2.2.4 m_paintText()

```
void CChildView::m_paintText (
            CDC & dc )
```

绘制所有文字以及背景色的函数

参数

in	*dc*	当前绘图区的句柄

注解

仅在OnPaint函数中调用

5.2.2.5 OnChar()

```
void CChildView::OnChar (
            UINT nChar,
            UINT nRepCnt,
            UINT nFlags )
```

响应发送文字消息的函数
响应ON_WM_CHAR消息

参数

in	*nChar*	用户发送给系统的字符，为UINT型，需要手动转化为wchar_t型
in	*nRepCnt*	重复发送同一个消息（用户按下一个键不松手）的次数
in	*nFlags*	各个符号位的具体作用参见MSDN官方文档

注解

请不要手动调用该函数

5.2.2.6 OnKeyDown()

```
void CChildView::OnKeyDown (
            UINT nChar,
            UINT nRepCnt,
            UINT nFlags )
```

键盘按下消息响应函数
此函数内部对nChar有判断，也就是说此函数只处理按下上下左右键的消息

参数

in	*nChar*	用户按下的键值
in	*nRepCnt*	用户按下该键的重复次数（用户按住该键）
in	*nFlags*	各个符号位的具体作用参见MSDN官方文档

注解

本函数只处理将光标上下左右移动的键盘消息，并且本函数只在用户按住一个键重复次数超过一的时候有效

5.2.2.7 OnKeyUp()

```
void CChildView::OnKeyUp (
            UINT nChar,
            UINT nRepCnt,
            UINT nFlags )
```

键盘抬起消息响应函数
此函数内部对nChar有判断，也就是说此函数只处理按下上下左右键的消息

参数

in	*nChar*	用户按下后抬起的键值
in	*nRepCnt*	用处不明
in	*nFlags*	各个符号位的具体作用参见MSDN官方文档

注解

本函数只处理将光标上下左右移动的键盘消息

5.2.2.8 OnLButtonDown()

```
void CChildView::OnLButtonDown (
            UINT nFlags,
            CPoint point )
```

响应鼠标左键按下消息的函数
当程序收到ON_WM_LBUTTONDOWN消息时调用该函数
主要处理两种情况

1. 当没有文字被选中的时候，将光标位置调整为当前点击位置，进入选择状态
2. 没有文字被选中的时候,不作任何动作

参数

in	*nFlags*	各个符号位的具体作用参见MSDN官方文档
in	*point*	返回鼠标左键按下时的屏幕坐标

注解

返回的坐标值是相对于客户区左上角而言的

5.2.2.9 OnLButtonUp()

```
void CChildView::OnLButtonUp (
            UINT nFlags,
            CPoint point )
```

响应鼠标左键抬起消息的函数
当程序收到ON_WM_LBUTTONUP消息时调用该函数
主要处理两种情况

1. 当没有文字被选中的时候，将光标位置调整为当前抬起的位置，结束选中状态。如果鼠标左键在按下与抬起之间有移动，则会有文字被选中
2. 当有文字被选中的时候，将光标位置更新，并且取消选中状态

参数

in	*nFlags*	各个符号位的具体作用参见MSDN官方文档
in	*point*	返回鼠标左键抬起时的屏幕坐标

注解

返回的坐标值是相对于客户区左上角而言的

5.2.2.10 OnMouseMove()

```
void CChildView::OnMouseMove (
            UINT nFlags,
            CPoint point )
```

响应鼠标移动消息的函数
当程序收到ON_WM_MOUSEMOVE消息时调用该函数
进入函数之后需要鼠标左键被按下的flag为true才会做出操作
操作主要为更新选择文字过程中的光标位置以及画面

参数

in	*nFlags*	各个符号位的具体作用参见MSDN官方文档
in	*point*	返回鼠标移动时的屏幕坐标

注解

返回的坐标值是相对于客户区左上角而言的

5.2.2.11 OnPaint()

```
void CChildView::OnPaint ( )  [protected]
```

绘图函数
当程序接受到ON_WM_PAINT的消息的时候就会调用该函数
该函数所做工作包括

- 重新计算所有文字的位置
- 重新绘制画面，包括所有文字以及背景色
- 将需要重绘的flag(即text_changed_f)置为flase

注解

请不要手动调用该函数，如需重绘画面请用Invalidate(false);函数

5.2.3 类成员变量说明

5.2.3.1 m_text

```
SITEXT* CChildView::m_text
```

自定义的文本文字类成员

参见

class SITEXT

该类的文档由以下文件生成:

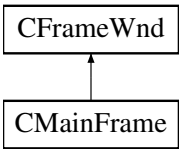
- Notepad/ChildView.h
- Notepad/ChildView.cpp

5.3 CMainFrame类 参考

程序主框架类CMainFrame类
主框架包括子视图类以及滚动条 继承自CFrameWnd类

```
#include <MainFrm.h>
```

类 CMainFrame 继承关系图:



Public 成员函数

- CMainFrame ()

  *CMainFrame*类构造函数
  初始化部分成员变量
- virtual BOOL PreCreateWindow (CREATESTRUCT &cs)

  绘图预处理函数,如果需要修改窗口样式则改写该函数
- virtual BOOL OnCmdMsg (UINT nID, int nCode, void *pExtra, AFX_CMDHANDLERINFO *pHandlerInfo)

  主消息循环函数,无需修改
- virtual ~CMainFrame ()

  默认析构函数
- void UpdateClientRect ()

  当窗口大小被调整之后调用的函数
- afx_msg void OnSize (UINT nType, int cx, int cy)

  当窗口大小被调整之后调用的消息响应函数
  当窗口大小被调整,共有两个地方需要调整
- afx_msg void OnVScroll (UINT nSBCode, UINT nPos, CScrollBar *pScrollBar)

  改变滚动块的位置时调用的函数
  调整两个地方
- void UpdateScrollBarPos ()

  当改变窗口大小时,调整滚动块的位置
  通过记录页面相对于上边界的偏移量来调整

Public 属性

- CChildView m_wndView

  子视图类成员变量
- CRect m_client_rect

  用于存储客户区的大小的变量
- int m_client_cy

  用于存储子视图的*y*方向的长度
- int scrolledpix

  用于记录页面相对于上边界偏移量
- int maincx

  记录当前客户区的宽度
- int maincy

  记录当前客户区的高度

Protected 成员函数

- afx_msg int OnCreate (LPCREATESTRUCT lpCreateStruct)

  创建主窗口时调用的函数
  创建了两个实例
- afx_msg void OnSetFocus (CWnd *pOldWnd)

  当窗口被聚焦的时候调用,无需修改

Protected 属性

- CScrollBar m_scrollBar
  滚动条类成员

5.3.1 详细描述

程序主框架类CMainFrame类
主框架包括子视图类以及滚动条 继承自CFrameWnd类

5.3.2 构造及析构函数说明

5.3.2.1 CMainFrame()

```
CMainFrame::CMainFrame ( )
```

CMainFrame类构造函数
初始化部分成员变量

- m_client_cy = 10000
- scrolledpix = 0

5.3.3 成员函数说明

5.3.3.1 OnCreate()

```
int CMainFrame::OnCreate (
            LPCREATESTRUCT lpCreateStruct )  [protected]
```

创建主窗口时调用的函数
创建了两个实例

- CChildView m_wndView
  参见
  CChildView
- CScrollBar m_scrollBar 竖直滚动条实例

参数

in	*lpCreateStruct*	至于这个是什么不重要,不需要自己调用这个函数

5.3.3.2 OnSize()

```
void CMainFrame::OnSize (
            UINT nType,
            int cx,
            int cy )
```

当窗口大小被调整之后调用的消息响应函数
当窗口大小被调整,共有两个地方需要调整

- 子窗口的大小
  通过调用UpdateClientRect函数

  参见

  UpdateClientRect

- 滚动条的一系列调整
  滚动条的长度调整
  滚动条上拖动块的位置调整

参数

in	*nType*	参阅MSDN官方文档
in	*cx*	由系统传入当前窗口x方向的长度
in	*cy*	有系统传入当前窗口y方向的长度

注解

无需自己调用,当手动改变窗口大小的时候,系统会传入ON_WM_SIZE消息,此时会自动调用该函数

5.3.3.3 OnVScroll()

```
void CMainFrame::OnVScroll (
            UINT nSBCode,
            UINT nPos,
            CScrollBar * pScrollBar )
```

改变滚动块的位置时调用的函数
调整两个地方

- 子视图的位置
- 滚动块的位置更新

参数

in	*nSBCode*	参阅MSDN官方文档
in	*nPos*	参阅MSDN官方文档
in	*pScrollBar*	当前被滑动的滚动条的指针

5.3.3.4 UpdateClientRect()

```
void CMainFrame::UpdateClientRect ( )
```

当窗口大小被调整之后调用的函数

参数

in	*cx*	当前窗口x方向长度
in	*cy*	当前窗口y方向长度

注解

该函数仅在OnSize函数中被调用,因为传入的参数需要用到OnSize函数的传入值

参见

OnSize

5.3.3.5 UpdateScrollBarPos()

```
void CMainFrame::UpdateScrollBarPos ( )
```

当改变窗口大小时,调整滚动块的位置
通过记录页面相对于上边界的偏移量来调整

注解

此函数仅在OnSize函数中调用

参见

OnSize重置滚动条位置

该类的文档由以下文件生成:

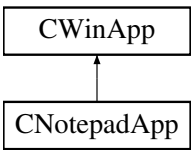
- Notepad/MainFrm.h
- Notepad/MainFrm.cpp

5.4 CNotepadApp类 参考

程序大类CNotepadApp
程序启动时创建,初始化函数中创建其他的对象
在初始化过程中创建菜单,主窗口
继承自CWinApp

```
#include <Notepad.h>
```

类 CNotepadApp 继承关系图:



Public 成员函数

- CNotepadApp ()
  默认构造函数
- virtual BOOL InitInstance ()
  初始化函数,实例化*CMainFrame*类,并且将*mainp*指针指向它
- virtual int ExitInstance ()
  退出程序调用函数
- afx_msg void OnFont ()
  选择字体消息响应函数
- afx_msg void OnPara ()
  选择行间距字间距消息响应函数
- afx_msg void **OnCut** ()
- afx_msg void **OnCopy** ()
- afx_msg void **OnPaste** ()
- afx_msg void OnAppAbout ()
  "关于"窗口的消息相应函数
- afx_msg void OnAlignLeft ()
  左对齐的消息响应函数
- afx_msg void OnAlignCenter ()
  居中的消息响应函数农户
- afx_msg void OnAlignRight ()
  右对齐的消息响应函数
- afx_msg void OnAlignDistribute ()
  分散对齐的消息响应函数
- afx_msg void OnOpen ()
  打开文件的消息响应函数
- afx_msg void OnClose ()
  关闭文件的消息响应函数
- void change_align (int flag)
  改变对齐方式的函数

Public 属性

- CMainFrame * mainp
  指向主窗口的指针
- SIRANGE m_cutBoard
  剪贴板,实际上是两个指向文字节点的指针组成的结构体

5.4.1 详细描述

程序大类CNotepadApp
程序启动时创建,初始化函数中创建其他的对象
在初始化过程中创建菜单,主窗口
继承自CWinApp

5.4.2 成员函数说明

5.4.2.1 OnFont()

```
void CNotepadApp::OnFont ( )
```

选择字体消息响应函数

- 实例化MFC字体选择对话框
- 将用户选择的字体传入SITEXT实例中

参见

SITEXT

注解

该函数响应ID_FONT消息,当用户点击菜单栏中"字体"一项时发送该消息

5.4.2.2 OnPara()

```
void CNotepadApp::OnPara ( )
```

选择行间距字间距消息响应函数

- 实例化选择行间距字间距的对话框
- 将用户设置的行间距以及字间距传入SITEXT实例中

参见

SITEXT

注解

该函数响应ID_PARA消息,当用户点击菜单栏中"段落"一项时发送该消息

该类的文档由以下文件生成:

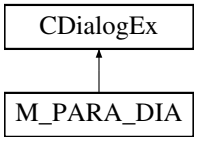
- Notepad/Notepad.h
- Notepad/Notepad.cpp

5.5 M_PARA_DIA类 参考

设置行间距以及字间距的对话框类M_PARA_DIA
该对话框用于用户设置行间距以及字间距
如果用户选择了一段文字，则输入的行距字间距将被用于选中的文字
如果用户没有选择文字，那么输入的行间距字间距将被应用于之后将会打出来的文字上

`#include <M_PARA_DIA.h>`

类 M_PARA_DIA 继承关系图:



Public 成员函数

- M_PARA_DIA (CWnd *pParent=NULL)
  默认构造函数
- virtual ~M_PARA_DIA ()
  默认析构函数
- afx_msg void **OnEnChangeEdit2** ()
- afx_msg void **OnEnChangeEdit3** ()
- afx_msg void **OnEnUpdateLine** ()
- afx_msg void **OnEnUpdateCharac** ()
- afx_msg void OnBnClickedOk ()
  当按下*OK*按钮的消息响应函数

Public 属性

- int m_linespace = 0
  行间距，默认值为*0*
- int m_charaspace = 0
  字间距，默认值为*0*

Protected 成员函数

- virtual void DoDataExchange (CDataExchange *pDX)
  数据交换函数

5.5.1 详细描述

设置行间距以及字间距的对话框类M_PARA_DIA
该对话框用于用户设置行间距以及字间距
如果用户选择了一段文字，则输入的行距字间距将被用于选中的文字
如果用户没有选择文字，那么输入的行间距字间距将被应用于之后将会打出来的文字上

该类的文档由以下文件生成:

- Notepad/M_PARA_DIA.h
- Notepad/M_PARA_DIA.cpp

5.6 SICHAR_INFO类 参考

表示一个字符的可以"设置"的信息

`#include <kernal.h>`

Public 成员函数

- SICHAR_INFO ()
  构造函数*1(*默认*)*
- SICHAR_INFO (const SICHAR_INFO &tinfo)
  构造函数*2*
- void set_fontpc (SIFONT_P tfontpc)
  设置*fontpc*成员*(*重载*:1)*
- void set_fontpc (SIFONT &tfont)
  设置*fontpc*成员*(*重载*:2)*
- void set_color (COLORERF tcolor)
  设置*color*成员
- void set_size (CHARSIZE tsize)
  设置*size*成员
- void set_cspace (CHARSPACE tcspace)
  设置*cspace*成员
- void set_lspace (LINESPACE tlspace)
  设置*lspace*成员
- SIFONT_P get_fontpc ()
  得到字体信息
- COLORERF get_color ()
  得到颜色信息
- CHARSIZE get_size ()
  得到字体大小
- CHARSPACE get_cspace ()
  得到字间距
- LINESPACE get_lspace ()
  得到行间距
- void print_info ()
  输出信息
- void read_info ()
  读入信息
- SICHAR_INFO & operator= (const SICHAR_INFO &tinfo)
  重载赋值函数

Public 属性

- SIFONT_P fontpc
  字体信息,详见*MSDN*
- COLORERF color
  颜色信息
- COLORERF bgcolor
  背景颜色信息
- CHARSIZE size
  字体大小，这里用作标识符
- CHARSPACE cspace
  字间距
- LINESPACE lspace
  行间距
- SIALIGN align
  对齐方式

友元

- class **SICHARNODE**

5.6.1 详细描述

表示一个字符的可以"设置"的信息

5.6.2 成员函数说明

5.6.2.1 get_color()

```
COLORERF SICHAR_INFO::get_color ( )  [inline]
```

得到颜色信息

返回值

一个*COLORREF*类型的变量，表示颜色	

5.6.2.2 get_cspace()

```
CHARSPACE SICHAR_INFO::get_cspace ( )  [inline]
```

得到字间距

返回值

一个*CHARSPACE*类型的变量，表示字间距	

5.6.2.3 get_fontpc()

```
SIFONT_P SICHAR_INFO::get_fontpc ( )  [inline]
```

得到字体信息

返回值

一个指向*SIFONT*的指针，表示字体信息	

5.6.2.4 get_lspace()

```
LINESPACE SICHAR_INFO::get_lspace ( )  [inline]
```

得到行间距

返回值

一个*LINESPACE*类型的变量，表示行间距	

5.6.2.5 get_size()

```
CHARSIZE SICHAR_INFO::get_size ( )  [inline]
```

得到字体大小

返回值

一个*CHARSIZE*类型的变量，表示字体大小	

5.6.2.6 operator=()

```
SICHAR_INFO & SICHAR_INFO::operator= (
            const SICHAR_INFO & tinfo )
```

重载赋值函数

注解

这个函数中，fontpc是复制指针指向的内容而不是指针本身

参数

in	*this	左操作数
in	tinfo	右操作数

返回值

*this	

5.6.2.7 set_color()

```
void SICHAR_INFO::set_color (
            COLORERF tcolor )  [inline]
```

设置color成员

参见

color

5.6.2.8 set_cspace()

```
void SICHAR_INFO::set_cspace (
            CHARSPACE tcspace )  [inline]
```

设置cspace成员

参见

cspace

5.6.2.9 set_fontpc() [1/2]

```
void SICHAR_INFO::set_fontpc (
            SIFONT_P tfontpc )  [inline]
```

设置fontpc成员(重载:1)

参见

fontpc

5.6.2.10 set_fontpc() [2/2]

```
void SICHAR_INFO::set_fontpc (
            SIFONT & tfont )  [inline]
```

设置fontpc成员(重载:2)

参见

fontpc

5.6.2.11 set_lspace()

```
void SICHAR_INFO::set_lspace (
            LINESPACE tlspace )  [inline]
```

设置lspace成员

参见

lspace

5.6.2.12 set_size()

```
void SICHAR_INFO::set_size (
            CHARSIZE tsize )  [inline]
```

设置size成员

参见

size

5.6.3 类成员变量说明

5.6.3.1 fontpc

```
SIFONT_P SICHAR_INFO::fontpc
```

字体信息,详见MSDN

参见

SIFONT_P

该类的文档由以下文件生成:

- Notepad/kernal.h
- Notepad/kernal.cpp

5.7 SICHARNODE类 参考

表示文本链表的一个节点

```
#include <kernal.h>
```

Public 成员函数

- SICHARNODE (SICHAR_T tch, SICHARNODE *tprevp=NULL, SICHARNODE *tnextp=NULL, SICHAR_INFO_P tchar_infop=NULL, SIDRAW_INFO_P tdraw_infop=NULL)
  构造函数*1*
- SICHARNODE (SICHAR_T tch, int tswidth, int tsheight)
  构造函数*2*
- void calc_S_from_font ()
  从当前的字体计算出绘制信息中的*S*的大小
- void set_fontpc (SIFONT_P tfontpc)
  设置*char_infop*中的*fontpc*并重新计算*draw_infop*中的*S*的大小
- void **set_fontpc** (SIFONT &tfont)
- void **set_color** (COLORERF tcolor)
- void **set_size** (CHARSIZE tsize)
- void **set_draw_infop_S** (const SIRECT &TS)
- void **set_draw_infop_L** (const SIRECT &TL)
- void **set_draw_infop_P** (const SIPOINT &TP)
- void **set_align** (SIALIGN align)
- void ins_prev (SICHARNODE *p)
  在当前字符的前面插入一个节点
- void ins_next (SICHARNODE *p)

在当前字符的后面插入一个节点

- void ins_prev (SICHARNODE *ps, SICHARNODE *pe)
  在当前字符的前面插入一段节点
- void ins_next (SICHARNODE *ps, SICHARNODE *pe)
  在当前字符的后面插入一段节点
- void ins_prev (const SIRANGE &range)
  等价于`ins_prev(range.sp,range.ep)`
- void ins_next (const SIRANGE &range)
  等价于`ins_next(range.sp,range.ep)`
- const SICHAR_INFO_P get_char_infop ()
  得到一个*char_infop*的*const*副本
- const SIDRAW_INFO_P get_draw_infop ()
  得到一个*draw_infop*的*const*副本
- void print_info ()
  输出信息
- void read_info ()
  输入信息

Public 属性

- SICHAR_T ch
  字符
- SICHAR_INFO_P char_infop
  字体信息
  一个指向*SICHAR_INFO*类型的指针，储存该字符的字体信息
- SIDRAW_INFO_P draw_infop
  绘制信息
  一个指向*SIDRAW_INFO*类型的指针，储存该字符的绘制信息
- SICHARNODE * prevp
  指向链表前一个节点的指针
- SICHARNODE * nextp
  指向链表后一个节点的指针

友元

- class **SITEXT**
- void del (SICHARNODE *p)
  从链表中删除一个节点
- void del (SICHARNODE *ps, SICHARNODE *pe)
  从链表中删除一段节点
- void del (const SIRANGE &range)

5.7.1 详细描述

表示文本链表的一个节点

5.7.2 成员函数说明

5.7.2.1 calc_S_from_font()

```
void SICHARNODE::calc_S_from_font ( )
```

从当前的字体计算出绘制信息中的S的大小

参见

S
fontpc

5.7.2.2 get_char_infop()

```
const SICHAR_INFO_P SICHARNODE::get_char_infop ( )  [inline]
```

得到一个char_infop的const副本

参见

char_infop

5.7.2.3 get_draw_infop()

```
const SIDRAW_INFO_P SICHARNODE::get_draw_infop ( )  [inline]
```

得到一个draw_infop的const副本

参见

draw_infop

5.7.2.4 ins_next() [1/3]

```
void SICHARNODE::ins_next (
            SICHARNODE * p )
```

在当前字符的后面插入一个节点

参数

in	*p*	指向被插入节点的指针

注解

该函数会把p指向的节点的字体信息改为和p的前一个节点相同

5.7.2.5 ins_next() [2/3]

```
void SICHARNODE::ins_next (
            SICHARNODE * ps,
            SICHARNODE * pe )
```

在当前字符的后面插入一段节点

参数

in	*ps*	指向被插入节点段的头指针
in	*pe*	指向被插入节点段的尾指针

注解

该函数不会改变被插入节点段的字体信息
要求这段节点必须事先已经用链表的结构组织好了

5.7.2.6 ins_next() [3/3]

```
void SICHARNODE::ins_next (
            const SIRANGE & range )  [inline]
```

等价于`ins_next(range.sp,range.ep)`

参见

ins_next(SICHARNODE* ps, SICHARNODE* pe)
SIRANGE

5.7.2.7 ins_prev() [1/3]

```
void SICHARNODE::ins_prev (
            SICHARNODE * p )
```

在当前字符的前面插入一个节点

参数

in	*p*	指向被插入节点的指针

注解

该函数会把p指向的节点的字体信息改为和p的前一个节点相同

5.7.2.8 ins_prev() [2/3]

```
void SICHARNODE::ins_prev (
            SICHARNODE * ps,
            SICHARNODE * pe )
```

在当前字符的前面插入一段节点

参数

in	*ps*	指向被插入节点段的头指针
in	*pe*	指向被插入节点段的尾指针

注解

该函数不会改变被插入节点段的字体信息
要求这段节点必须事先已经用链表的结构组织好了

5.7.2.9 ins_prev() [3/3]

```
void SICHARNODE::ins_prev (
            const SIRANGE & range )  [inline]
```

等价于ins_prev(range.sp,range.ep)

参见

ins_prev(SICHARNODE* ps, SICHARNODE* pe)
SIRANGE

5.7.3 友元及相关函数文档

5.7.3.1 del [1/3]

```
void del (
            SICHARNODE * p )  [friend]
```

从链表中删除一个节点

参数

in	*p*	待删除的节点

注解

p不能是链表的头节点或尾节点

5.7.3.2 del [2/3]

```
void del (
            SICHARNODE * ps,
            SICHARNODE * pe )  [friend]
```

从链表中删除一段节点

参数

in	*ps*	指向被删除节点段的头节点
in	*pe*	指向被删除节点段的尾节点

注解

该节点段不能包含链表的头节点或尾节点

5.7.3.3 del [3/3]

```
void del (
            const SIRANGE & range )  [friend]
```

参见

del

5.7.4 类成员变量说明

5.7.4.1 ch

```
SICHAR_T SICHARNODE::ch
```

字符

参见

SICHAR_T

5.7.4.2 char_infop

```
SICHAR_INFO_P SICHARNODE::char_infop
```

字体信息
一个指向SICHAR_INFO类型的指针，储存该字符的字体信息

参见

class SICHAR_INFO

5.7.4.3 draw_infop

```
SIDRAW_INFO_P SICHARNODE::draw_infop
```

绘制信息
一个指向SIDRAW_INFO类型的指针，储存该字符的绘制信息

参见

class SIDRAW_INFO

该类的文档由以下文件生成:

- Notepad/kernal.h
- Notepad/kernal.cpp

5.8 SIDRAW_INFO类 参考

表示一个字符的与绘制相关的信息
把每个字符抽象成一个屏幕上的矩形，每两个矩形之间没有缝隙

```
#include <kernal.h>
```

Public 成员函数

- SIDRAW_INFO ()
  构造函数*1(*默认*)*
- SIDRAW_INFO (const SIRECT &TS, const SIRECT &TL, const SIPOINT &TPOS)
  构造函数*2*
- void set_S (const SIRECT &)
  设置*S*的值*(*重载*:1)*
- void set_S (int, int)
  设置*S*的值*(*重载*:2)*
- void set_L (const SIRECT &)
  设置*L*的值
- void set_POS (const SIPOINT &)
  设置*POS*的值
- const SIRECT & get_S ()
  获得*S*的值
- SIRECT & get_L ()
  获得*L*的值
- SIPOINT & get_POS ()
  获得*POS*的值
- void print_info ()
  输出信息
- void read_info ()
  读入信息

Public 属性

- SIRECT S
  一个字符初始矩形的形状
  影响它的属性是

- SIRECT L
  一个字符画在屏幕上的形状
  影响它的属性是

- SIPOINT **POS**

友元

- class **SICHARNODE**

5.8.1 详细描述

表示一个字符的与绘制相关的信息
把每个字符抽象成一个屏幕上的矩形，每两个矩形之间没有缝隙

5.8.2 类成员变量说明

5.8.2.1 L

```
SIRECT SIDRAW_INFO::L
```

一个字符画在屏幕上的形状
影响它的属性是

- 字体
- 字号
- 字间距
- 行间距
- 该字符当前所在行高度最高的字符的高度

参见

struct SIRECT

5.8.2.2 S

```
SIRECT SIDRAW_INFO::S
```

一个字符初始矩形的形状
影响它的属性是

- 字体
- 字号

参见

struct SIRECT

该类的文档由以下文件生成:

- Notepad/kernal.h
- Notepad/kernal.cpp

5.9 SIPOINT结构体 参考

点/向量结构体
用于表示一个字符(的左上角)在屏幕上的位置

```
#include <kernal.h>
```

Public 成员函数

- SIPOINT (int tx=0, int ty=0)
  构造函数
- void print_info ()
  输出信息
- void read_info ()
  读入信息

Public 属性

- int x
  *x*坐标*(px)*
- int y
  *y*坐标*(px)*

友元

- SIPOINT operator+ (const SIPOINT &A, const SIPOINT &B)
- SIPOINT operator- (const SIPOINT &A, const SIPOINT &B)
- bool operator< (const SIPOINT &A, const SIPOINT &B)
  定义两个点的小于关系
- bool operator> (const SIPOINT &A, const SIPOINT &B)

5.9.1 详细描述

点/向量结构体
用于表示一个字符(的左上角)在屏幕上的位置

5.9.2 友元及相关函数文档

5.9.2.1 operator+

```
SIPOINT operator+ (
            const SIPOINT & A,
            const SIPOINT & B )  [friend]
```

参数

B	向量加法

5.9.2.2 operator-

```
SIPOINT operator- (
            const SIPOINT & A,
            const SIPOINT & B )  [friend]
```

参数

B	向量减法

5.9.2.3 operator<

```
bool operator< (
            const SIPOINT & A,
            const SIPOINT & B )  [friend]
```

定义两个点的小于关系

参数

in	*A*	左操作数
in	*B*	右操作数

返回值

true	A的x坐标更小 或 AB的x坐标相等且A的y坐标更小
false	B的x坐标更小 或 AB的x坐标相等且B的y坐标更小

5.9.2.4 operator>

```
bool operator> (
            const SIPOINT & A,
            const SIPOINT & B )  [friend]
```

参数

B	定义两个点的大于关系，与小于关系相反

参见

operator <

该结构体的文档由以下文件生成:

- Notepad/kernal.h
- Notepad/kernal.cpp

5.10 SIRANGE结构体 参考

表示文本链表中的一段节点

`#include <kernal.h>`

Public 成员函数

- SIRANGE (SICHARNODE_P tsp=NULL, SICHARNODE_P tep=NULL)
  构造函数(默认)
- void _clear ()
  清空函数

Public 属性

- SICHARNODE_P sp
  指向这段节点的头指针
- SICHARNODE_P ep
  指向这段节点的尾指针

5.10.1 详细描述

表示文本链表中的一段节点

5.10.2 成员函数说明

5.10.2.1 _clear()

```
void SIRANGE::_clear ( )  [inline]
```

清空函数

注解

这个函数只是把sp和ep的值赋为NULL，并不会清空链表中的节点

5.10.3 类成员变量说明

5.10.3.1 ep

```
SICHARNODE_P SIRANGE::ep
```

指向这段节点的尾指针

参见

SICHARNODE_P

5.10.3.2 sp

```
SICHARNODE_P SIRANGE::sp
```

指向这段节点的头指针

参见

SICHARNODE_P

该结构体的文档由以下文件生成:

- Notepad/kernal.h

5.11 SIRECT结构体 参考

矩形结构体
用于表示一个字符占据的屏幕空间大小

```
#include <kernal.h>
```

Public 成员函数

- SIRECT (int twidth=20, int theight=20)

  构造函数*1*

- SIRECT (const SIRECT &trect)

  构造函数*2*

- void print_info ()

  输出信息

- void read_info ()

  读入信息

Public 属性

- int width

  宽度*(px)*

- int height

  高度*(px)*

5.11.1 详细描述

矩形结构体
用于表示一个字符占据的屏幕空间大小

该结构体的文档由以下文件生成:

- Notepad/kernal.h
- Notepad/kernal.cpp

5.12 SITEXT类 参考

文本链表
把文本从头到尾抽象成一个链表

```
#include <kernal.h>
```

Public 成员函数

- set false when repaint () is called bool text_changed_f
  文本改变标志
- void _init ()
  初始化方法
- void _destroy ()
  清除数据并释放内存
- void draw_line_from_left (SICHARNODE_P ps, SICHARNODE_P pe, int sx, int y, int line_height, int deltax)
  行计算函数
  用于计算一行字符的绘制信息
- void proc_line (SICHARNODE_P ps, SICHARNODE_P pe, int n, int y, int line_height, int tot_weight, SIALIGN align)
  行处理函数
  根据不同对齐方式用于计算一行字符的绘制信息
- void pre_proc ()
  预处理函数
  对文本做预处理*(*主要是添加分段符*)*
- void proc_text ()
  文本处理函数
- std::vector< SILINE >::iterator point_to_line (const SIPOINT &P)
  求包含屏幕上某一点的那一行
- void set_line_align (const SILINE &line, SIALIGN align)
  设置某一行的对齐方式
- void save ()
  保存文本
- void open ()
  打开文本
- void ins_char (SICHAR_T)
  在光标前方插入一个字符
- void ins_char (SICHAR_T, int, int)
  在光标前方插入一个字符，指定*S*的大小
- void del_char (bool backwards)
  删除字符
- void start_select ()
  进入选择状态
- void end_select ()
  结束选择状态
- void del_select ()
  删除被选中的字符
- void replace_select (const SIRANGE &)
  替换被选中的字符*(*重载*1)*
- void replace_select (SICHARNODE_P ps, SICHARNODE_P pe)
  替换被选中的字符*(*重载*2)*
- void cancel_select ()
  取消选中
- void mov_select (SICURSORP)
  把选中字符移到光标区域
- void mov_cursorp (SICURSORP)

移动光标*(重载1)*
把光标移到参数指定的节点处

- void mov_cursorp (SIDIRECT)

  移动光标*(重载2)*
  把光标往上/下/左/右移动一格
- void mov_cursorp (const SIPOINT &)
- void set_default_font (SIFONT &tfont)

  设置默认字体*(重载1)*
- void set_default_font (SIFONT_P tfontp)

  设置默认字体*(重载2)*
- void set_select_font (SIFONT_P tfontpc)

  设置选中区域的字体*(重载1)*
- void set_select_font (SIFONT &tfont)

  设置选中区域的字体*(重载2)*
- void **set_curfont** (SIFONT_P tcurfontpc)
- void **set_curfont** (SIFONT &tcurfont)
- void set_select_color (COLORERF tcolor)

  设置选中区域的颜色
- void set_select_lspace (LINESPACE tlspace)

  设置选中区域的行距
- void set_select_cspace (CHARSPACE tcspace)

  设置选中区域的字间距
- void set_pagewidth (PAGEWIDTH tpagewidth)

  设置页面宽度
- void set_select_align (SIALIGN align)

  设置选中区域的对齐方式
- void set_cursorp_align (SIALIGN align)

  设置光标所在行的对其方式
- SIALIGN get_range_align (SICHARNODE_P ps, SICHARNODE_P pe)

  得到一段节点的对齐方式*(重载1)*
- SIALIGN get_range_align (const SIRANGE &range)

  得到一段节点的对齐方式*(重载2)*
- void set_save_path (string tsave_path)

  设置保存路径
- void set_open_path (string topen_path)

  设置打开路径
- SICURSORP point_to_cursorp (const SIPOINT &P)

  把屏幕上的点转换为光标位置
- void repaint ()

  重新计算所有信息
- void print_info ()

  输出信息
- void read_info ()

  读入信息
- void anticolor (SICHARNODE_P ps, SICHARNODE_P pe)

  把一段字符反色

Public 属性

- string save_path
  保存路径
- string open_path
  打开路径
- SICHARNODE_P headp
  文本链表头指针
- SICHARNODE_P tailp
  文本链表尾指针
- SICURSORP cursorp
  光标位置
  一个指向链表中一个节点的指针，用来表示光标目前所在的位置
- SISELECT select
  当前选中的区间
- PAGEWIDTH pagewidth
  页面宽度，用于计算字符的位置
- SIFONT_P **curfontpc**
- SIFONT default_font
  初始的默认字体
- bool **inselect**
- int **fwdnum**
- std::vector< SILINE > vlinep
  按顺序储存屏幕上从上到下的每一行
- std::vector< SIPARAGRAPH > vparap
  按顺序储存屏幕上从上到下每一段
- bool **set_curfontp_f**

静态 Public 属性

- static const PAGEWIDTH **DEFAULT_PAGEWIDTH** = 110
- static const SIDIRECT **DLEFT** = 0
- static const SIDIRECT **DRIGHT** = 1
- static const SIDIRECT **DUP** = 2
- static const SIDIRECT **DDOWN** = 3
- static const SIALIGN **ANORMAL** = 0
- static const SIALIGN **ALEFT** = 1
- static const SIALIGN **ARIGHT** = 2
- static const SIALIGN **ACENTER** = 3
- static const SIALIGN **ADISTRIBUTED** = 4

友元

- bool point_on_line (SILINE L, const SIPOINT &P)
  判断一个屏幕上的点是否在某一行内

- bool point_on_char_col (SICHARNODE_P np, const SIPOINT &P)
  判断一个屏幕上的点是否在某一个字符所处的列内

5.12.1 详细描述

文本链表
把文本从头到尾抽象成一个链表

5.12.2 成员函数说明

5.12.2.1 cancel_select()

```
void SITEXT::cancel_select ( )  [inline]
```

取消选中

注解

该函数把选中区域赋为NULL，不会对文本做出更改

5.12.2.2 del_char()

```
void SITEXT::del_char (
            bool backwards = true )
```

删除字符

参数

in	*backwards*	表示删除的方向 -true 删除前一个字符 -false 删除后一个字符

5.12.2.3 draw_line_from_left()

```
void SITEXT::draw_line_from_left (
            SICHARNODE_P ps,
            SICHARNODE_P pe,
            int sx,
            int y,
```

```
            int line_height,
            int deltax )
```

行计算函数
用于计算一行字符的绘制信息

参数

in	*ps*	指向该行头节点的指针
in	*pe*	指向该行尾节点的指针
in	*sx*	该行左上角的x坐标
in	*y*	该行左上角的y坐标
in	*line_height*	行高
in	*delta*	每个字符除了自身的宽度外的附加宽度(只在分散对齐时不为0)

5.12.2.4 get_range_align() [1/2]

```
SIALIGN SITEXT::get_range_align (
            SICHARNODE_P ps,
            SICHARNODE_P pe )
```

得到一段节点的对齐方式(重载1)

参数

in	该节点段的头节点	
in	该节点段的尾节点	

注解

如果这段节点有多种非默认的对齐方式，选择最靠前的节点的对齐方式

5.12.2.5 get_range_align() [2/2]

```
SIALIGN SITEXT::get_range_align (
            const SIRANGE & range )  [inline]
```

得到一段节点的对齐方式(重载2)

参见

get_range_align

5.12.2.6 mov_cursorp() [1/3]

```
void SITEXT::mov_cursorp (
            SICURSORP tcursorp )  [inline]
```

移动光标(重载1)
把光标移到参数指定的节点处

参数

in	*tcursorp*	移动的目标节点

5.12.2.7 mov_cursorp() [2/3]

```
void SITEXT::mov_cursorp (
            SIDIRECT tdir )
```

移动光标(重载2)
把光标往上/下/左/右移动一格

参数

in	*tdir*	移动方向

5.12.2.8 mov_cursorp() [3/3]

```
void SITEXT::mov_cursorp (
            const SIPOINT & P )  [inline]
```

@移动光标(重载3)
把光标移到屏幕上一点给定的位置

参数

in	*P*	目标位置

5.12.2.9 proc_line()

```
void SITEXT::proc_line (
            SICHARNODE_P ps,
            SICHARNODE_P pe,
```

```
            int n,
            int y,
            int line_height,
            int tot_weight,
            SIALIGN align )
```

行处理函数
根据不同对齐方式用于计算一行字符的绘制信息

参数

in	*ps*	指向该行头节点的指针
in	*pe*	指向该行尾节点的指针
in	*n*	该行的字符总数(不包括换行符)
in	*y*	该行左上角的坐标
in	*line_height*	行高
in	*tot_width*	该行字符自身的宽度总和
in	*align*	该行的对齐方式 -ANORMAL 0 默认对齐方式(左对齐) -ALEFT 1 左对齐 -ARIGHT 2 右对齐 -ACENTER 3 居中对齐 -ADISTRIBUTED 4 分散对齐

5.12.3 友元及相关函数文档

5.12.3.1 point_on_char_col

```
bool point_on_char_col (
            SICHARNODE_P np,
            const SIPOINT & P )  [friend]
```

判断一个屏幕上的点是否在某一个字符所处的列内

参数

in	*np*	指向被判断字符的节点指针
in	*P*	被判断的点

返回值

true	该点在该字符所在的列内
false	该点不在该字符所在列内

注解

该函数判断的只有列，不是判断点是否在字符上

5.12.3.2 point_on_line

```
bool point_on_line (
            SILINE L,
            const SIPOINT & P )  [friend]
```

判断一个屏幕上的点是否在某一行内

参数

in	*L*	被判断的行
in	*P*	被判断的点

返回值

true	该点在这一行内
false	该点不在这一行内

5.12.4 类成员变量说明

5.12.4.1 cursorp

```
SICURSORP SITEXT::cursorp
```

光标位置
一个指向链表中一个节点的指针，用来表示光标目前所在的位置

注解

该指针指向的节点代表屏幕上显示的光标的后一个字符

参见

SICURSORP

5.12.4.2 default_font

```
SIFONT SITEXT::default_font
```

初始的默认字体

参见

SIFONT

5.12.4.3 headp

```
SICHARNODE_P SITEXT::headp
```

文本链表头指针

参见

SICHARNODE_P

5.12.4.4 pagewidth

```
PAGEWIDTH SITEXT::pagewidth
```

页面宽度，用于计算字符的位置

参见

PAGEWIDTH

5.12.4.5 select

```
SISELECT SITEXT::select
```

当前选中的区间

参见

SISELECT

5.12.4.6 tailp

```
SICHARNODE_P SITEXT::tailp
```

文本链表尾指针

参见

SICHARNODE_P

5.12.4.7 vlinep

```
std::vector<SILINE> SITEXT::vlinep
```

按顺序储存屏幕上从上到下的每一行

参见

SILINE

5.12.4.8 vparap

```
std::vector<SIPARAGRAPH> SITEXT::vparap
```

按顺序储存屏幕上从上到下每一段

参见

SIPARAGRAPH

该类的文档由以下文件生成:

- Notepad/kernal.h
- Notepad/kernal.cpp

Chapter 6

文件说明

6.1 Notepad/ChildView.h 文件参考

声明子视窗类CChildView

```
#include "stdafx.h"
#include "kernal.h"
```

类

- class CChildView

  子视窗类*CChildView*
  子视窗是程序中客户区部分，即文本编辑的画布部分 -此类继承自*CWnd*

6.1.1 详细描述

声明子视窗类CChildView

作者

洪方舟 hongfz16@163.com

版本

1.0

日期

2017.5.25

6.2 Notepad/kernal.h 文件参考

记事本的内核

```
#include "stdafx.h"
#include <string>
#include <vector>
```

类

- struct SIRECT
  矩形结构体
  用于表示一个字符占据的屏幕空间大小
- struct SIPOINT
  点/向量结构体
  用于表示一个字符(的左上角)在屏幕上的位置
- struct SIRANGE
  表示文本链表中的一段节点
- class SIDRAW_INFO
  表示一个字符的与绘制相关的信息
  把每个字符抽象成一个屏幕上的矩形，每两个矩形之间没有缝隙
- class SICHAR_INFO
  表示一个字符的可以"设置"的信息
- class SICHARNODE
  表示文本链表的一个节点
- class SITEXT
  文本链表
  把文本从头到尾抽象成一个链表

类型定义

- typedef int **SIDIRECT**
- typedef int **SIALIGN**
- typedef int **PAGEWIDTH**
- typedef int **COLORERF**
- typedef int **CHARSIZE**
- typedef int **CHARSPACE**
- typedef int **LINESPACE**
- typedef char **SICHAR_T**
- typedef LOGFONT **SIFONT**
- typedef SIFONT * **SIFONT_P**
- typedef SIPOINT **SIVECTOR**
- typedef SICHAR_INFO * **SICHAR_INFO_P**
- typedef SIDRAW_INFO * **SIDRAW_INFO_P**
- typedef SICHARNODE * **SICHARNODE_P**
- typedef SIRANGE **SISELECT**
- typedef SIRANGE **SIPARAGRAPH**
- typedef SIRANGE **SILINE**
- typedef SICHARNODE_P **SICURSORP**

函数

- template<class T >
  void **exchange** (T &a, T &b)
- bool **isenter** (SICHAR_T ch)
- int **Max** (int a, int b)
- COLORERF anticolor_ext (COLORERF c)
  反色函数

- int **ABS** (int a)
- bool point_on_line (SILINE L, const SIPOINT &P)
- bool point_on_char_col (SICHARNODE_P np, const SIPOINT &P)

6.2.1 详细描述

记事本的内核

作者

李仁杰 ShadowIterator@hotmail.com

版本

1.0

日期

2015.6.5

6.2.2 函数说明

6.2.2.1 anticolor_ext()

```
COLORERF anticolor_ext (
            COLORERF c )  [inline]
```

反色函数

参数

in	c	当前颜色

返回值

反色后的颜色	

6.2.2.2 point_on_char_col()

```
bool point_on_char_col (
            SICHARNODE_P np,
            const SIPOINT & P )  [inline]
```

参数

in	*np*	指向被判断字符的节点指针
in	*P*	被判断的点

返回值

true	该点在该字符所在的列内
false	该点不在该字符所在列内

注解

该函数判断的只有列，不是判断点是否在字符上

6.2.2.3 point_on_line()

```
bool point_on_line (
            SILINE L,
            const SIPOINT & P )  [inline]
```

参数

in	*L*	被判断的行
in	*P*	被判断的点

返回值

true	该点在这一行内
false	该点不在这一行内

6.3 Notepad/lib.h 文件参考

声明全局变量：窗口HDC

```
#include "stdafx.h"
```

变量

- HDC **globalhdc**

6.3.1 详细描述

声明全局变量：窗口HDC

作者

洪方舟 hongfz16@163.com

版本

1.0

日期

2017.6.8

6.4 Notepad/M_PARA_DIA.h 文件参考

声明一个用于设置行间距以及字间距的对话框类M_PARA_DIA

```
#include "stdafx.h"
#include "resource.h"
```

类

- class M_PARA_DIA

  设置行间距以及字间距的对话框类*M_PARA_DIA*
  该对话框用于用户设置行间距以及字间距
  如果用户选择了一段文字，则输入的行距字间距将被用于选中的文字
  如果用户没有选择文字，那么输入的行间距字间距将被应用于之后将会打出来的文字上

6.4.1 详细描述

声明一个用于设置行间距以及字间距的对话框类M_PARA_DIA

作者

洪方舟 hongfz16@163.com

版本

1.0

日期

2017.5.25

6.5 Notepad/MainFrm.h 文件参考

声明程序主框架类CMainframe类

```
#include "stdafx.h"
#include "ChildView.h"
#include "M_PARA_DIA.h"
```

类

- class CMainFrame
  程序主框架类*CMainFrame*类
  主框架包括子视图类以及滚动条 继承自*CFrameWnd*类

6.5.1 详细描述

声明程序主框架类CMainframe类

作者

洪方舟 hongfz16@163.com

版本

1.0

日期

2017.5.25

6.6 Notepad/Notepad.h 文件参考

声明程序大类CNotepadApp

```
#include "stdafx.h"
#include "MainFrm.h"
#include "resource.h"
#include <string>
```

类

- class CNotepadApp

  程序大类*CNotepadApp*
  程序启动时创建,初始化函数中创建其他的对象
  在初始化过程中创建菜单,主窗口
  继承自*CWinApp*

变量

- CNotepadApp **theApp**

6.6.1 详细描述

声明程序大类CNotepadApp

作者

洪方舟 hongfz16@163.com

版本

1.0

日期

2017.5.25

Index