**WMI开发日志及随想**

-------------------------------8.20----------------------------------

1.遇到的问题

（1） delete [] m\_tx （m\_tx = new cDateTimeCtrl [3]）报错？

定位：CExternParent.cpp::OnClose()->228l

分析：将他放在析构函数执行前，不报错，所以可能是他自身调用了其他的类，而那个类处于执行状态，或者说指向了其他地方

解决：在父窗口关闭时，向子窗口发消息，让他delete变量

（2）

2.之后的计划

（1） 现在窗口基本显示没问题了，今后主要是窗口控件的逻辑判定，以怎样的形式进行判定，当我操作控件时，要知道操作的是哪一个控件，他的作用域是哪个范围，并且还要方便后期数据加入时的处理。

（2） 每一个窗口初始化或者从数据库获取数据的时候，对于有CDateTimeCtrl的“窗口”，怎么初始化时间静态文本的文本。

（3） 数据库的操作和保存的对象，不同用户又该怎么处理，服务器与本地数据库之间的联系又是什么。

（4）

3.今天的感想

今天是第一次写日志，难免有些不习惯，一步一步的慢慢来吧，养成良好的习惯。谁叫你之前不努力！哼！

还有就是，你有多粗心，一个表格改了多少次，不要拿着数据就跑，认真分析，先记录哪些有问题，一个一个的排查。

-------------------------------8.23----------------------------------

1. 遇到的问题

无。

2. 之后的计划

（1） 对于控件的操作对象，打算把list进行重绘，一方面也是学习并了解控件的重绘流程，另一方面，将list重绘弄到一个dll里面，以后用到的时候还可以复用，尽量弄的丰富一点。这样就可以把对list的外界操控，转换到list里面，降低一些逻辑难度。

（2） 对于需要初始化的“时间”“总计”文本框，在查询点击后，获取CDateTimeCtrl的数据，还有就是要拦截“enter”消息，当编辑time后，“用户”可能会点击“enter”，但在窗口中“enter”默认是执行窗口完毕并退出，所以这个消息需要拦截并定义。

3.今天的感想

（1） 以后需要改一下自己的逻辑思维，比如“return true”和“return false”，“true”（常定义为非0值）并不意味着程序运行“正确”，而是返回一个“真值”，然后继续传递程序值，继续执行函数；“false”也不意味着程序运行“失败”，而是返回一个“假值”，然后终止程序。

（1.1）“true”和“false”就好比是“装一碗水”（函数功能），最终最好的状态就是刚好“装一碗水”，不多也不少，水多一点（>0）或者水少一点（<0）(true)对于“装一碗水”来说都不是最好的状态，但是也是可以的，而水刚刚好（== 0）（false）就是最好的状态。

（1.2）或者理解为，前一个函数需要某个函数执行并返回判断值“bool”的时候，询问该函数，“是否继续执行？”，“true”-->“yes please！”。“false”->“no thanks！”。最好的例子就是sqlite判断回调函数是否继续执行或者sql语句是否执行成功，sqlite自己定义了一个bool值，“SQLITE\_OK”，字面上感觉是表示“true”（仅我自己原本的思维），然后在调用时，如果CALLBACK返回“true”就会导致查询或者其他输出语句，只输出结果的第一行，如果去查看SQLITE\_OK的定义你会发现，他定义的值为“0”,也就是“false”。

（1.3) 但另一方面，也可以“true”表示函数执行成功，“是否执行成功？”，反之false表示失败，这只是逻辑上或者实际需求的不同。

（1.4）所以该怎么办？我也不知道，多了解一些正确的书写规则吧，将自己的代码写的更加的规范！（经过询问群里面的大佬得知：“**既然是bool当然true是正常，不能用int去理解bool，如果返回是int,就说明返回值有很多含义，不只是成功和失败2种结果，比如错误码，如果成功，也可能是记录数**”）在很多情况下，int型返回值，往往0表示没有错，而bool，true表示没错

（2）之后写日志和随想，不能只有结束，还要有开始，比如今天的计划是什么，完成到什么程度，遇到了什么问题，怎么解决的，对于这种解决方式为什么没有想到，这解决方法又可能存在哪些局限性，对于这种解决方法有什么发散性的思维或者想法，然后之后的计划，最后今天的感想。要不弄一个日志exe？可以试着了解并制作一个。

-------------------------------8.24----------------------------------

1.今天的计划

了解ListCtrl控件的自绘

2.了解的结果

能找到的资料，内容都是关于控件自身的背景重新设置，并没有和其他控件进行联系，主要就是一个关键的结构体（详细见附件1.）

typedef struct tagDRAWITEMSTRUCT

{

UINT CtlType; //指定了控件的类型

UINT CtlID; //指定了自绘控件的ID值，而对于菜单项则不需要使用该成员

UINT itemID; //表示菜单项ID，也可以表示列表框或者组合框中某项的索引值。

UINT itemAction; //指定绘制行为，

UINT itemState; //指定了当前绘制操作完成后，所绘项的可见状态。

HWND hwndItem; //自绘控件的窗口句柄

HDC hDC; //指定了绘制操作所使用的设备环境。

RECT rcItem; //指定了将被绘制的矩形区域。

ULONG\_PTR itemData;//传递给控件的值

}

对象也都注释了，在自绘中，就是对这个结构体的某个值或者某些组合值进行判断，然后再进行相应的操作，结构体的属性值是WM\_MESSAGE传递值给子类，然后子类（在定义OWNERDRAW或者DRAWITEM的窗口风格下）反射消息给派生类，其他的的消息在派生类中也可以重载。

3.今天的感想

每当了解一个新的操作，总是痛苦的，网上的文档鱼龙混杂，编译环境又不同，造成很多地方与实际不同，虽然思维不变。还有就是一个新的操作，总是伴随着一些新的、不了解的类或结构体什么的，哎，怪自己太懒。

很多时候自己的想法还是可行的，但是一定要注意细节！

-------------------------------8.26----------------------------------

1. 之后的时间安排

|  |  |
| --- | --- |
| 多线程 | 时间：一个月。主要是如何高效的运行程序 |
| excel | 时间：两周。涉及对com的调用，暂时把现阶段所需要的功能先封装起来，之后有时间的话就继续添加后续的功能，一步一步的完善 |
| word | 时间：两周。还没有了解过，但是可以通过excel宏来辅助设计，先简单的满足现阶段要求，后续继续完善，也要封装起来。 |
| socket | 时间：两周。缓存池、通信传输的数据等处理。 |
| sql | 时间：一学期。这个作为辅助，sqlite那个已经封装一部分了，但是不规范，有时间把他弄的规范一点。然后是sqlServer,语句和sqlite差别不大，只是调用方式可能不同，分开封装。 |
| 日志 | 这个简单，后期的时候，先把功能操作分个类，然后在现有函数结尾添加代码，然后存到数据库即可。 |
| 控件自绘 | 各种控件，有时间的话就弄，全部封装起来，尽量的把所有可以改的全部改了，且可以实现动态自定义 |

-------------------------------8.30----------------------------------

1. 今天的任务

完成CDialogAdd类的设计。（尽量尝试自绘控件）

1. 永远在列表最后一行添加一个“+”，表示增加列操作，在最初，只要鼠标点击列表框列表区域，就显示一个编辑框，然后“+”隐藏，也就是说，上一行数据添加完后，“+”才显示。
2. 当鼠标移动时，且移动区域在list内部、在list最后一行前面，每移动到新的一列，就在该列前面显示一个“x”，表示删除该列。当某行处于编辑状态时不显示“x”。
3. 留出给服务器通信的接口，并返回所添加的值，添加到父窗口的list里面。
4. 如果有单据编号，这个要自动添加，且该编号从服务器返回，保持唯一性。
5. 其余属性，非空，前提是有数据填入的情况下，如果为空给出提示，并显示矩形框，跳转到那个地方。

列表纵向获取文本，如果文本为“”，就设置当前点击的行为改空白文本行

1. 当按下退格（backspace）时，且当前为非编辑状态，显示编辑框在选中区域，对该区域进行编辑，因为非空的原因。
2. 遇到的问题
3. 怎么覆盖原本存在的“+”图片？
4. 怎么知道当前行，内容全部编辑，且非空？
5. 怎么去掉底层点击后的遮罩？

m\_Ls.SetItemState(lvhti.iItem, 0, LVIS\_SELECTED);

m\_Ls.SetSelectionMark(-1);

-------------------------------9.4---------------------------------

1.多线程编程

（1）进程间通信，可以用内存映射，CreateFileMapping(-1,read/write)(-1表示不是磁盘文件句柄，这里是通过共享数组内存地址)、MapViewOfFile。

（2）保持某些操作完成后，才继续其他线程，用CreateMutex、操作结束用ReleaseMutex释放，这样其他线程开始争抢自己的cpu时间，前提是要有WaitForSingleObject。如果需要同时等待多个信号量有效，就用WaitForMultipleObject（HANDLE\_COUNT，HANDLE，Bool，time），如果等待cout个变量都有效（或者多个线程都释放了Mutex，这也意味着这几个线程都结束了），bool值就为true，如果只是其中一个有效，就false。

（3）所有的句柄使用完后都要CloseHandle（）

2.Excel操作

（1）类向导add lib（C:\Program Files(x86)\Microsoft\Office\root\Office16）

(2) 加入\_Application，Font，Range，\_WorkBook，WorkBooks，WorkSheet，WorkSheets

(3）

\_Application app; //Excel应用程序接口

\_Workbook book;

Workbooks books;

\_Worksheet sheet;

Worksheets sheets;

Range range;

Font font;

Range cols;

CString tem,stri,strA,strB;

COleVariant covOptional((long)DISP\_E\_PARAMNOTFOUND,VT\_ERROR);

if (!app.CreateDispatch(\_T("Excel.Application")))

{

this->MessageBox(\_T("无法创建Excel应用！"));

return;

}

books = app.GetWorkbooks(); //获取工作薄集合

book = books.Add(covOptional); //添加一个工作薄

sheets = book.GetWorksheets(); //获取工作表集合

sheet = sheets.GetItem(COleVariant((short)1)); //获取第一个工作表

/////////////////////////第一列/////////////////////////////////////////////////

//设置列名

range = sheet.GetRange(COleVariant(\_T("A1")), COleVariant(\_T("A1"))); //选择工作表中A1:A1单元格区域

range.SetValue2(COleVariant(\_T("序号"))); //A1:A1中填入“序号”

//设置字体为粗体

font = range.GetFont();

font.SetBold(COleVariant((short)TRUE));

//写入数据

arr1.RemoveAll(); //清空数组（用数组存该列的所有数据）

int line=m\_ls.GetItemCount(); //查询clistctrl控件中记录数

CString str\_line;

str\_line.Format("%d",line+1);

for(int i=0;i < m\_ls.GetItemCount();i++)

{

tem=m\_ls.GetItemText(i,0);

arr1.Add(tem);

}

for(i=1;i<=line;i++) //循环将数组的内容写到Excel中

{

stri.Format("%d",i+1);

strA="A"+stri;

strB="A"+str\_line;

range = sheet.GetRange(COleVariant(\_T(strA)),COleVariant(\_T(strB)));

//确定表的范围

range.SetValue2(COleVariant(\_T(arr1[i-1])));

}

//选择整列，并设置宽度为自适应

cols = range.GetEntireColumn();

cols.AutoFit();

//后面几列处理方法相同，省略。

app.SetVisible(TRUE); //显示表格

app.SetUserControl(TRUE);

3.打印

调用CPrintDialog类，然后获取打印机的DC，初始化DOCINFO结构体变量，StartDoc、（View\_Draw）、EndPage、EndDoc。然后打印机就会在后台进行打印，特别要注意的一点是获取打印机纸张的尺寸、和像素（GetDeviceCaps），获取的像素是每英寸的像素点数量，所以在绘制的时候，数值比例要控制好。

附件1.

DRAWITEMSTRUCT结构的定义如下：

typedef struct tagDRAWITEMSTRUCT

{

UINT CtlType; //指定了控件的类型

UINT CtlID; //指定了自绘控件的ID值，而对于菜单项则不需要使用该成员

UINT itemID; //表示菜单项ID，也可以表示列表框或者组合框中某项的索引值。

UINT itemAction; //指定绘制行为，

UINT itemState; //指定了当前绘制操作完成后，所绘项的可见状态。

HWND hwndItem; //自绘控件的窗口句柄

HDC hDC; //指定了绘制操作所使用的设备环境。

RECT rcItem; //指定了将被绘制的矩形区域。

ULONG\_PTR itemData; //传递给控件的值

} DRAWITEMSTRUCT, NEAR \*PDRAWITEMSTRUCT, FAR \*LPDRAWITEMSTRUCT;

CtlType：指定了控件的类型，其取值如下表所示。

ODT\_BUTTON：按钮控件 ODT\_COMBOBOX：组合框控件 ODT\_LISTBOX：列表框控件

ODT\_LISTVIEW：列表视图控件 ODT\_MENU：菜单项 ODT\_STATIC：静态文本控件

ODT\_TAB：Tab控件

CtlID：指定了自绘控件的ID值，而对于菜单项则不需要使用该成员

itemID：表示菜单项ID，也可以表示列表框或者组合框中某项的索引值。对于一个空的列表框或组合框，该成员的值为–1。这时应用程序只绘制焦点矩形（该矩形的坐标由rcItem 成员给出）虽然此时控件中没有需要显示的项，但是绘制焦点矩形还是很有必要的，因为这样做能够提示用户该控件是否具有输入焦点。当然也可以设置itemAction 成员为合适值，使得无需绘制焦点。

itemAction：指定绘制行为，其取值可以为下表中所示值的一个或者多个的联合。

ODA\_DRAWENTIRE：当整个控件都需要被绘制时，设置该值

ODA\_FOCUS：如果控件需要在获得或失去焦点时被绘制，则设置该值。此时应该检查itemState：成员，以确定控件是否具有输入焦点。

ODA\_SELECT：如果控件需要在选中状态改变时被绘制，则设置该值。此时应该检查itemState：成员，以确定控件是否处于选中状态。

itemState：指定了当前绘制操作完成后，所绘项的可见状态。

例如，如果菜单项应该被灰色显示，则可以指定ODS\_GRAYED状态标志。其取值可以为下表中所示值的一个或者多个的联合。

ODS\_CHECKED：如果菜单项将被选中，则可设置该值。该值只对菜单项有用。

ODS\_COMBOBOXEDIT：在自绘组合框控件中只绘制选择区域。

ODS\_DEFAULT：默认值。

ODS\_DISABLED：如果控件将被禁止，则设置该值。

ODS\_FOCUS：如果控件需要输入焦点，则设置该值。

ODS\_GRAYED：如果控件需要被灰色显示，则设置该值。该值只在绘制菜单时使用。

ODS\_HOTLIGHT：Windows 98 / Me, Windows 2000 / XP: 如果鼠标指针位于控件之上，则设置该值，这时控件会显示高亮颜色。

ODS\_INACTIVE：Windows 98 / Me, Windows 2000 / XP : 表示没有激活的菜单项。

ODS\_NOACCEL：Windows 2000 / XP : 控件是否有快速键盘。

ODS\_NOFOCUSRECT：Windows 2000 / XP : 不绘制捕获焦点的效果。

ODS\_SELECTED：选中的菜单项。

hwndItem：指定了组合框、列表框和按钮等自绘控件的窗口句柄；如果自绘的对象时菜单项，则表示包含该菜单项的菜单句柄。

hDC：指定了绘制操作所使用的设备环境。

rcItem：指定了将被绘制的矩形区域。这个矩形区域就是上面hDC的作用范围。系统会自动裁剪组合框、列表框或按钮等控件的自绘制区域以外的部分。也就是说rcItem中的坐标点（0，0）指的就是控件的左上角。但是系统不裁剪菜单项，所以在绘制菜单项的时候，必须先通过一定的换算得到该菜单项的位置，以保证绘制操作在我们希望的区域中进行。

itemData：对于菜单项，该成员的取值可以是由CMenu::AppendMenu、CMenu::InsertMenu或者CMenu::ModifyMenu 等函数传递给菜单的值。对于列表框或这组合框，该成员的值可以为由ComboBox::AddString、CComboBox::InsertString、CListBox::AddString或者CListBox::InsertString等传递给控件的值。如果ctlType 的取值是ODT\_BUTTON或者ODT\_STATIC, itemData的取值为0。

附件2.

赠品核算、自用核算、拆并、无进价采购、etc。