# AME程序设计

## 还要添加的功能

8、 B03-A1基坑各测斜点在基坑开挖结束时的最大侧向位移（如果软件无该功能给我excel表也行）

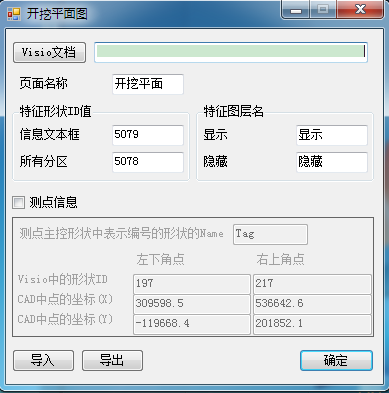
例子：

9、B03-A1基坑围护墙最大侧向位移与最大地面沉降之间的关系图（如果软件无该功能给我excel表也行）

例子：

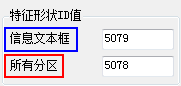
# 绘图与数据结构

## 开挖平面图



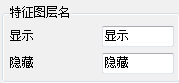
### Visio绘图规范

#### 形状的层次



* 对于开挖平面图而言，将开挖深度越深的分块图形放置在越上层，即自然地面的形状放置在最下层，而开挖基坑底的分块形状放置在最顶层；
* 对于自然地面标高以下的所有分块形状，在绘制完成后，将其统一放置在一个组合形状中，并记录此形状的ID值（红色方框）。
* 为了在页面中显示开挖工况对应的施工日期，在页面中应该一个用来记录施工时期的形状，并记录其ID值（蓝色方框）

#### 图层的布置



为了进行图形的滚动，即相应分块的显示与隐藏，必须设置一个“显示”图层和一个“隐藏”图层，并记录这两个图层的名称，可以不用手动设置其可见性，在程序中会自动进行设置。

#### 绘制测点



为了通过坐标来定义不同的监测点位在Visio页面中的位置，必须绘制两个定位图形，并记录这两个图形的ID值，然后设置其在施工中对应的坐标。

在绘制Visio图形时，为了根据不同的测点类型（如测斜、立柱、水位等）绘制不同的图形，必须在文档模具中绘制相应的模板，并在每一个主控形状中，添加一个子形状，这些子形状以相同的名称进行标记，以显示此测点的点位信息（如CX01），并记录下此名称。也就是说，此主控形状中，不能只有一个形状，而且其中有一个形状的名称必须是用来标记测点编辑的。





除了用户自己去设置主控形状，也可以选择程序中提供的模具中的主控形状。

### Excel数据表的格式与内容要求

#### 监测点编号与对应坐标——监测点位

监测点位的工作表中，需要记录如下图所示的5项信息。其中，前三列用来索引每一个监测点，后面两列表示此测点的坐标。

要注意的是，“监测项目”这一项中的值，必须在Visio的模具中有相应名称的主控形状，不然在绘图时会因为找不到形状而报错。

每一个测点的在现在的坐标值可以通过监测单位提供，也可以从CAD图纸中进行批量提取，在此不作详细说明。



在数据的排列顺序上，第一列与第二列可以交换顺序，但是这两列的数据不能出现穿插，即必须进行排序；而第三列数据在其所在的集合中，不能出现重复。

正确的方式如下左图，错误的方式如下右图。

#### 开挖分块——图形的滚动

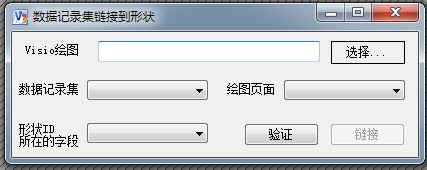


要实现图形的滚动，必须记录Visio绘图中的每一个分块的图形的ID值，以及此分块开挖完成的时间，并汇总为上图中的两列。

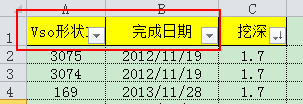
至于第三列的分块的挖深信息，并不是必需的，但是可以将其作为每一个分块的形状数据，显示在Visio中的分块形状上。

### 形状数据的链接

这是一个小的插件功能，用来根据每一个形状的ID值，来进行自动的数据链接。



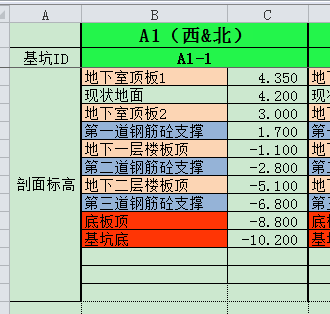
在使用此功能前，先在Excel表格中记录下要进行数据链接的形状的ID值，以及对应的数据信息，然后使用Visio的“将数据链接到形状”功能将其手动导入到Visio的数据库中，生成一个数据记录集。

然后使用此程序，将数据记录集按形状的名称链接到对应的形状上。借此即可以实现数据的实时刷新，以及将数据信息作为数据图形显示在图形中。

## 测斜曲线

### 构件与对应标高的数据格式



应用简化表示法

|  |  |  |
| --- | --- | --- |
| 构件 | 英文 | 简化 |
| 地面 | Ground | GRD |
| 第一道钢筋砼支撑 | Reinforced concrete struts | S1:RC |
| 第二道砼支撑 | concrete struts | S2:C |
| 第三道钢支撑 | Steel struts | S3:S |
| 第四道支撑（支撑类型不明） | Unknown strut type | S4:U |
| 地下室顶板（即地下一层顶板） | Floor | F0 |
| 地下一层楼板顶（即地下一层脚底） | Floor | F1 |
| 底板顶 | 拼音di ban ding | DBD |
| 基坑底 | Bottom | BTM |

### 监测数据格式

## 其他监测曲线

# 代码整理

## FSO嵌入WindowsForm

|  |
| --- |
| Imports Microsoft.Office.Interop.Visio  Imports Microsoft.Office.Interop  Imports Microsoft.Office.Interop.Excel  Public Class FrmExcavation  Dim vsdoc As Visio.Document  Private Sub Button1\_Click(ByVal sender As System.Object, \_  ByVal e As System.EventArgs) Handles Button1.Click  Dim strFileName As String  'Find the Office document.  With OpenFileDialog1  .FileName = ""  .ShowDialog()  strFileName = .FileName  End With  If strFileName.Length Then  AxFramerControl1.Open(strFileName)  '------------------------------------------------------ 获取Visio文档的对象。  vsdoc = AxFramerControl1.ActiveDocument  With vsdoc.Application  With .Windows(1)  .ShowScrollBars = False  .ShowRulers = False  .ShowPageTabs = False  .ScrollLock = True  .ViewFit = VisWindowFit.visFitWidth  .Windows.ItemFromID(VisWinTypes.visWinIDShapeSearch).Visible = True  End With  .ShowStatusBar = False  If vsdoc IsNot Nothing Then .ShowToolbar = False  .Settings.ShowShapeSearchPane = True  .DoCmd(VisUICmds.visCmdShapesWindow)  End With  End If  End Sub  Private Sub Form1\_activate(sender As Object, e As EventArgs) Handles Me.Activated  If vsdoc IsNot Nothing Then  vsdoc.Application.ShowToolbar = False  End If  End Sub  Private Sub Form1\_Load(ByVal sender As Object, ByVal e As \_  System.EventArgs) Handles MyBase.Load  Button1.Text = "Browse"  With OpenFileDialog1  .Filter = "Office Documents " & \_  "(\*.doc, \*.xls, \*.vsd)|\*.doc;\*.xls;\*.vsd"  .FilterIndex = 1  End With  End Sub  Private Sub Form1\_Closing(ByVal sender As Object, ByVal e As \_  System.ComponentModel.CancelEventArgs) Handles MyBase.Closing  vsdoc.Close()  vsdoc = Nothing  End Sub  End Class |

## Excel

### 对象尺寸的设置

如果某对象包含于某父对象中，则一定要先设置其父对象的尺寸，再设置子对象的尺寸，因为在设置父对象的尺寸时，子对象的尺寸会进行缩放。比如设置在Chart中的图例的尺寸前，一定要先设置Chart的尺寸。

另外，在设置Application的尺寸时，可以在Application.Visible=False时对窗口的尺寸进行设置，但是要注意，设置窗口尺寸时一定要确保Application的WindowState不能为xlMaximized（可以为xlMinimized）。

If App.WindowState = Excel.XlWindowState.xlMaximized Then

App.WindowState = Excel.XlWindowState.xlNormal

End If

App.Height = .ChartHeight + .MarginOut\_Height

App.Width = .ChartWidth + .MarginOut\_Width

## Visio

### 在Visio中将导入的外部数据链接到形状

|  |
| --- |
| Sub 将外部数据链接到形状()  Dim data As DataRecordset  Dim IDs() As Long  Dim RowID As Variant  Dim shp As Shape    Set data = ActiveDocument.DataRecordsets(1)  IDs = data.GetDataRowIDs("")  For Each RowID In IDs  Set shp = ActivePage.Shapes.ItemFromID(data.GetRowData(RowID)(0))  shp.LinkToData 1, RowID  Next  End Sub |

### 根据形状数据来控制形状的显示与否

|  |
| --- |
| Sub 挖土分区的显示与隐藏()  Dim excelApp As Excel.Application  Dim wkbkdatabase As Excel.Workbook  Dim shtRegion As Excel.Worksheet  Set excelApp = CreateObject("Excel.application")  excelApp.Visible = True  Set wkbkdatabase = excelApp.Workbooks.Open("F:\基坑数据\程序编写\文件测试\database.xlsb", False, True)  Set shtRegion = wkbkdatabase.Worksheets("开挖分块")  Dim dic As Object  Set dic = CreateObject("scripting.dictionary")  dic.comparemode = vbTextCompare    Dim arrKeys As Variant  Dim arrValues As Variant  With shtRegion  arrKeys = .Range(.Cells(2, 1), .Cells(.UsedRange.Rows.Count, 1)).Value  arrValues = .Range(.Cells(2, 5), .Cells(.UsedRange.Rows.Count, 5)).Value  End With    Dim today As Date  Dim shp As Shape  Dim shpAllRegion As Shape  Dim layerVsb As Layer  Dim layerInVsb As Layer  Set shpAllRegion = ActivePage.Shapes.ItemFromID(5041)  'shp.Cells("layermember").Formula = ""  Set layerVsb = ActivePage.Layers.Item("显示")  Set layerInVsb = ActivePage.Layers.Item("隐藏")  layerInVsb.CellsC(visLayerVisible) = 0  layerVsb.CellsC(visLayerVisible) = 1  layerInVsb.Add shpAllRegion, 0  For today = #11/19/2012# To #7/2/2014# Step 30  For i = LBound(arrValues, 1) To UBound(arrValues, 1)  Set shp = ActivePage.Shapes.ItemFromID(arrKeys(i, 1))  If today >= arrValues(i, 1) Then 'show  layerVsb.Add shp, 0  Else 'hide  layerVsb.Add shp, 1  layerVsb.Remove shp, 0  End If  Next  Next  End Sub |

## Word

## 控件的移动

自定义一个控件，它可以在鼠标的拖动下进行移动。

|  |
| --- |
| Imports System.Drawing.Drawing2D  Public Class Pic\_Member  Inherits Windows.Forms.PictureBox  #Region " --- 移动图片"  ''' <summary>  ''' 控件是否正处于移动状态中  ''' </summary>  ''' <remarks></remarks>  Private blnOnMove As Boolean = False  ''' <summary>  ''' 开始移动时的鼠标基准点  ''' </summary>  ''' <remarks></remarks>  Private RefMousePoint As Point  ''' <summary>  ''' 开始移动时的控件左上角基准点  ''' </summary>  ''' <remarks></remarks>  Private LeftUpPoint As Point  Private Sub Pic\_Member\_MouseDown(sender As Object, e As MouseEventArgs) Handles Me.MouseDown  Me.blnOnMove = True  Me.RefMousePoint = New Point(MousePosition.X, MousePosition.Y)  Me.LeftUpPoint = New Point(Me.Left, Me.Top)  End Sub  Private Sub Pic\_Member\_MouseMove(sender As Object, e As MouseEventArgs) Handles Me.MouseMove  If Me.blnOnMove = True Then  Dim NewTop As Integer = Me.LeftUpPoint.Y + (MousePosition.Y - RefMousePoint.Y)  Dim NewLeft As Integer = Me.LeftUpPoint.X + (MousePosition.X - RefMousePoint.X)  If NewTop >= 0 Then  Me.Top = NewTop  Else  Me.Top = 0  End If  '  If NewLeft >= 0 Then  Me.Left = NewLeft  Else  Me.Left = 0  End If  End If  End Sub  Private Sub Pic\_Member\_MouseUp(sender As Object, e As MouseEventArgs) Handles Me.MouseUp  Me.blnOnMove = False  End Sub  #End Region  End Class |

## 置顶VB中的对话框InputBox

用Visual Basic.NET在Excel的事件中创建一个InputBox时，这个对话框是以Modal的形式弹出的，而且它的Modal对象是Excel界面，而不是VB的主程序界面。

但是有一个问题，这个对话框初始时并不是默认出现在Excel界面的前面的，它有可能被Excel界面挡住。而挡住后，不能通过将Excel移动或缩小来显示出InputBox对话框，因为Excel界面被Modal而不能操作了。

此时就要想办法，让InputBox对话框在出现时就置顶。

解决代码如下：

|  |
| --- |
| Dim strPoint As String = ""  Dim strInputBoxTitle As String = "Speed Mode"  Dim thd As New System.Threading.Thread(  Sub()  strPoint = InputBox("内容", strInputBoxTitle)  End Sub)  Thread.Sleep(100)  thd.Start()  Dim hwnd As IntPtr = myAPI.FindWindow("#32770", vbNullString)  Dim bln As Long = myAPI.SetForegroundWindow(hwnd)  thd.Join() |

其主要思路为：用API函数FindWindow获取这个弹出的InputBox的句柄（InputBox对话框为Windows的标准对话框，其类名为“#32770”），然后用API函数SetForegroundWindow将其设置为置顶窗口。

但是还有一个问题，如果直接弹出一个InputBox对话框的话，进程会等待用户在InputBox对话框中输入完成并点确定（此时InputBox对话框会关闭）后才继续执行后面的代码，而此时再用FindWindow已经找不到这个窗口了。

所以，可以新开一个线程，通过异步操作的方式来解决这个问题。具体代码见上面。

说明一下，上面的一个关键处理：Thread.Sleep(100)。它是为了解决系统操作的时间差问题。即有可能而且确实出现了在FindWindow函数的执行出现在InputBox对话框之前的问题。所以在这里先让执行此事件方法的线程暂停0.1秒，以等待对话框出现，然后再进行FindWindow，这样就不会出现找不到对话框的问题了。

但是这里还有一个问题，Thread.Sleep(100)的目的如上所述，但是在测试时发现，将它放在thd.Start()的效果比放在thd.Start()的后面的效果要好。但是先暂停主线程，再启动对话框，这种方法好像并不能实现在和线程中等待对话框出现的效果，而实际测试中却实现了——有待进一步学习与讨论。

## 在VB.NET中响应Excel中的Resize事件

参考程序：[..\..\Visual Basic.NET\VB2010 Project\Excel的Resize事件\Excel的Resize事件.sln](../../Visual%20Basic.NET/VB2010%20Project/Excel的Resize事件/Excel的Resize事件.sln)

# 代码规范化

## 代码的注释

## 变量命名的习惯

大多数人习惯于采用类似公共变量的方式命名这些私有变量，在它们前面加上小写的前缀字母m或者加上下划线区分所关联的属性（m前缀是VB6中继承下来的，意思是：“模块级变量”）。举一个例子：

|  |
| --- |
| Private \_ContactName As String  Public Property ContactName()  Get  Return \_ContactName  End Get  Set(ByVal Value)  \_ContactName = Value  End Set  End Property |

## 创建低耦合度的方法

在定义的方法中，尽量少使用类中的全局变量，而将这些变量以输入形参的形式定义。

对于完全封装的方法，可以在方法名前面加上一个“S\_”的前缀，表示Shared或者C#中的Static关键字。

## 为用户开启的线程命名

这是出于调试的方便。

Dim thd As New Thread(AddressOf Methods)

Thd.name = "线程启动的代码位置或者线程的特征。"

Thd.start()