```
namespace CreateCSharpCOM
{

    //[Guid("7D9ECD57-93B4-47AD-9ADC-7B4AA0EB6338")]
    //[ComVisible(true)]
    //public interface ITransaction
    //{
    //    void Connect(string connectString);

    //    void Disconnect();

    //    string GetVersion();

    //    int Record(Invoice invoice);
    //}
    //[Guid("086EC9B2-EA82-4DDD-8EC3-A8D27829CC97")]
    //[ComVisible(true)]
    //[ClassInterface(ClassInterfaceType.None)]
    //[Description("模拟事务记录")]
    //public class SimTransaction : ITransaction
    //{
    //    public void Connect(string connectString)
    //    {
    //    }

    //    public void Disconnect()
    //    {
    //    }

    //    public string GetVersion()
    //    {
    //        return "1.0";
    //    }
    //    public int Record(Invoice invoice)
    //    {
    //        return 0;
    //    }
    //}
    //[Guid("154BD6A6-5AB8-4d7d-A343-0A68AB79470B")]
    [Guid("8366D2B6-0275-4C9A-A8C2-7A9B4389BAD1")]
    public interface MyCom_Interface
    {
        [DispId(1)]
        int Add(int a, int b);
    }
    //[Guid("D11FEA37-AC57-4d39-9522-E49C4F9826BB"),
InterfaceType(ComInterfaceType.InterfaceIsIDispatch)]
    //public interface MyCom_Events
    //{

    //}

    [Guid("2E3C7BAD-1051-4622-9C4C-215182C6BF58"), ClassInterface(ClassInterfaceType.None),
ComSourceInterfaces(typeof(MyCom_Events))]
    //[Guid("FE406D41-05B6-4183-9082-BA4037ECEA37")]
```

```
    public class MyCom : MyCom_Interface
    {
        public int Add(int a, int b)
        {
            return a + b;
        }
    }
}

namespace CreateCSharpDLL
{
    public class DLL
    {
        public static int Multiply(int a,int b)
        {
            return a * b;
        }

        public static int Div(int a, int b)
        {
            return a / b;
        }
    }
}

extern "C" _declspec(dllexport) int _stdcall test01(int a, int b);
extern "C" _declspec(dllexport) int _stdcall test02(int a, int b);
#include "CreateDLL.h"
#include "stdafx.h"

int _stdcall test01(int a, int b) {
    return a + b;
}

int _stdcall test02(int a, int b) {
    return a - b;
}

namespace CSharpCallDLL
{
    /// <summary>
    /// Page1.xaml 的交互逻辑
    /// </summary>
    public partial class Page1 : Page
    {
        //C#调用C++的DLL
        [DllImport(@"..\..\..\Release\CreateDLL.dll", EntryPoint = "test01", SetLastError =
true, CharSet = CharSet.Ansi, ExactSpelling = false, CallingConvention =
CallingConvention.StdCall)]
        public static extern int Add(int a, int b);

        [DllImport(@"..\..\..\Release\CreateDLL.dll", EntryPoint = "test02", SetLastError =
true, CharSet = CharSet.Ansi, ExactSpelling = false, CallingConvention =
CallingConvention.StdCall)]
        public static extern int Sub(int a, int b);
```

```
string path = @"SOFTWARE\Microsoft\Windows\CurrentVersion\Themes";
string key = "SetupVersion";
public Page1()
{
    InitializeComponent();
    path_text.Text = path;
}

private void Register_Click(object sender, RoutedEventArgs e)
{

    StringBuilder lpData = new StringBuilder();
    lpData.Capacity = 30;//设置最大容量，也就是缓冲区最大长度
    //原始值是10
    RegisterUtil.GetBitRegistryKey("HKEY_LOCAL_MACHINE", path, key, ref lpData);
    version.Text = lpData.ToString();
}

/*更改启动版本号*/
private void Change_Click(object sender, RoutedEventArgs e)
{

    string value = version.Text;
    RegisterUtil.SetBitRegistryKey("HKEY_LOCAL_MACHINE", path, "SetupVersion", value);
    MessageBox.Show("修改成功");
}

/*在注册表中创建新的键值对*/
private void Create_Click(object sender, RoutedEventArgs e)
{
    if (value_text.Text == "")
    {
        MessageBox.Show("请输入创建的内容");
        return;
    }
    string notepadpath = @"SOFTWARE\Microsoft\Windows\CurrentVersion\"+value_text.Text;
    if(RegisterUtil.CreateRegistryKey("HKEY_LOCAL_MACHINE", notepadpath) == -1)
    {
        return;
    }
    MessageBox.Show("创建成功");
}



private void Sub_Button_Click(object sender, RoutedEventArgs e)
{
    int a;
    int b;
    int c;
    try
    {
        a = int.Parse(textbox1.Text);
```

```
            b = int.Parse(textbox2.Text);
            c = Sub(a, b);
            textbox3.Text = c.ToString();
        }
        catch (System.FormatException)
        {
            MessageBox.Show("请输入正确的运算数");
        }
        //string str = System.Environment.CurrentDirectory;
        //textbox3.Text = c.ToString();
    }

    private void Add_Button_Click(object sender, RoutedEventArgs e)
    {
        int a;
        int b;
        int c;
        try
        {
            a = int.Parse(textbox4.Text);
            b = int.Parse(textbox5.Text);
            c = Add(a, b);
            textbox6.Text = c.ToString();
        }
        catch (System.FormatException)
        {
            MessageBox.Show("请输入正确的运算数");
        }
    }

    private void Multi_Button_Click(object sender, RoutedEventArgs e)
    {
        int a;
        int b;
        //C#调用C#生成的DLL
        Assembly assembly =
Assembly.LoadFrom(@"..\..\..\CreateCSharpDLL\bin\Release\CreateCSharpDLL.dll");
        //path_text.Text = System.Environment.CurrentDirectory;
        try
        {
            a = int.Parse(textbox7.Text);
            b = int.Parse(textbox8.Text);
            foreach(Type t in assembly.GetTypes())
            {
                if (t.IsClass && !t.IsAbstract)
                {
                    MethodInfo Multiply = t.GetMethod("Multiply");
                    object o = Activator.CreateInstance(t);
                    Object[] paramaters = new Object[2];
                    paramaters[0] = a;
                    paramaters[1] = b;
                    textbox9.Text = Multiply.Invoke(o, paramaters).ToString();
                }
            }
        }
```

```
            catch (System.FormatException)
            {
                MessageBox.Show("请输入正确的运算数");
            }
        }

        private void Div_Button_Click(object sender, RoutedEventArgs e)
        {
            int a;
            int b;
            //C#调用C#生成的DLL
            // Assembly assembly =
Assembly.LoadFrom(@"C:\Users\dell\Desktop\Windows\CreateCSharpDLL\bin\Release\CreateCSharpDLL.dl
l");
            Assembly assembly =
Assembly.LoadFrom(@"..\..\..\CreateCSharpDLL\bin\Release\CreateCSharpDLL.dll");
            try
            {
                a = int.Parse(textbox10.Text);
                b = int.Parse(textbox11.Text);
                foreach (Type t in assembly.GetTypes())
                {
                    if (t.IsClass && !t.IsAbstract)
                    {
                        MethodInfo Div = t.GetMethod("Div");
                        object o = Activator.CreateInstance(t);
                        Object[] paramaters = new Object[2];
                        paramaters[0] = a;
                        paramaters[1] = b;
                        textbox12.Text = Div.Invoke(o, paramaters).ToString();
                    }
                }
            }
            catch (System.FormatException)
            {
                MessageBox.Show("请输入正确的运算数");
            }
        }
    }

}
namespace CSharpCallDLL
{
    /// <summary>
    /// Page2.xaml 的交互逻辑
    /// </summary>
    public partial class Page3 : Page
    {
        public MsWord.Document oDoc;
        public string doc_file_name = Directory.GetCurrentDirectory() + @"\content.doc";
        public MsWord.Application oWordApplic;
        public int curSectionNum;
        public MsWord.Range curRange;
        MsWord.Selection currentSelection;
```

```
public Page3()
{
    InitializeComponent();
}


public static MyCom.MyCom_Interface CreateTransaction()
{
    MyCom.MyCom_Interface iMycom = null;
    try
    {
        //Guid guid = Parameters.Get().transaction;
        Guid guid = new Guid("2E3C7BAD-1051-4622-9C4C-215182C6BF58");
        Type transactionType = Type.GetTypeFromCLSID(guid);
        object transaction = Activator.CreateInstance(transactionType);
        iMycom = transaction as MyCom.MyCom_Interface;
        //m.Connect(Parameters.Get().transactionString);
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
    return iMycom;
}


private void DIY_COM_Button_Click(object sender, RoutedEventArgs e)
{
    //ITransaction it = CreateTransaction();
    listbox.Items.Add("正在调用自定义 COM");
    // MyCom.MyCom mycom = new MyCom.MyCom();
    MyCom.MyCom_Interface iMycom = CreateTransaction();
    try
    {
        int a = int.Parse(textbox1.Text);
        int b = int.Parse(textbox2.Text);
        int c = iMycom.Add(a, b);
        listbox.Items.Add(a.ToString()+"+"+b.ToString()+"="+c.ToString());
    }
    catch(Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}

private void insertAbstract()
{
    if (File.Exists(doc_file_name))
    {
        File.Delete(doc_file_name);
    }
    //创建文档
    oWordApplic = new MsWord.Application();
```

```
object missing = Missing.Value;//对用不上的参数占位
                                        //创建 word 文档小节
object curTxt;
curSectionNum = 1;
oDoc = oWordApplic.Documents.Add(ref missing, ref missing, ref missing, ref missing);
oDoc.Activate();
this.Dispatcher.Invoke(() => listbox.Items.Add("正在输出文档小节"));
object section_nextPage = MsWord.WdBreakType.wdSectionBreakNextPage;
object page_break = MsWord.WdBreakType.wdPageBreak;
for (int si = 0; si < 6; si++)
{
    oDoc.Paragraphs[1].Range.InsertParagraphAfter();
    oDoc.Paragraphs[1].Range.InsertBreak(ref section_nextPage);
}
//写入摘要
this.Dispatcher.Invoke(() => listbox.Items.Add("正在生成摘要"));
curSectionNum = 1;
curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range;
curRange.Select();
string one_str, key_word;
StreamReader file_absract = new StreamReader("abstract.txt");
oWordApplic.Options.Overtype = false;
currentSelection = oWordApplic.Selection;
if (currentSelection.Type == MsWord.WdSelectionType.wdSelectionNormal)
{
    one_str = file_absract.ReadLine();
    key_word = file_absract.ReadLine();//读入关键字
    currentSelection.TypeText(one_str);
    currentSelection.TypeParagraph();//添加段落标记
    currentSelection.TypeText("摘要");//添加摘要
    currentSelection.TypeParagraph();//添加段落标记
    one_str = file_absract.ReadLine();//读入段落文本
    while (one_str != null)
    {
        currentSelection.TypeText(one_str);
        currentSelection.TypeParagraph();//添加段落标记
        one_str = file_absract.ReadLine();
    }
    currentSelection.TypeText("关键字: ");
    currentSelection.TypeText(key_word);
    currentSelection.TypeParagraph();
}
file_absract.Close();
//设置摘要标题格式
curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range;
curTxt = curRange.Paragraphs[1].Range.Text;
curRange.Font.Name = "宋体";
curRange.Font.Size = 16;
curRange.Paragraphs[1].Alignment = MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
//设置摘要格式
curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[2].Range;
curRange.Select();
curRange.Font.Name = "黑体";
```

```
            curRange.Font.Size = 14;
            curRange.Paragraphs[1].Alignment = MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
            //摘要正文
            oDoc.Sections[curSectionNum].Range.Paragraphs[1].Alignment =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
            for (int i = 3; i < oDoc.Sections[curSectionNum].Range.Paragraphs.Count; i++)
            {
                curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[i].Range;
                curRange.Font.Name = "宋体";
                curRange.Font.Size = 10;
                //设置段落为多倍行距
                oDoc.Sections[curSectionNum].Range.Paragraphs[i].LineSpacingRule =
MsWord.WdLineSpacing.wdLineSpaceMultiple;
                oDoc.Sections[curSectionNum].Range.Paragraphs[i].LineSpacing = 15f;
                //首行缩进 2 字符
                oDoc.Sections[curSectionNum].Range.Paragraphs[i].IndentFirstLineCharWidth(2);

            }
            curRange = curRange.Paragraphs[curRange.Paragraphs.Count].Range;
            curRange.Font.Bold = 1;
        }

        private void insertCatalog()
        {
            //插入目录
            curSectionNum = 2;
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range;
            curRange.Select();
            //插入目录的参数
            object useheading_styles = true;
            object upperheading_level = 1;
            object lowerheding_level = 3;
            object useelds = 1;
            object tableid = 1;
            object RightAlignPageNumbers = true;
            object IncludePageNumbers = true;
            currentSelection = oWordApplic.Selection;
            currentSelection.TypeText("目录");
            currentSelection.TypeParagraph();
            currentSelection.Select();
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[2].Range;
            curRange.Collapse();
            oDoc.TablesOfContents.Add(curRange, ref useheading_styles, ref upperheading_level, ref
lowerheding_level,
                ref useelds, ref tableid, ref RightAlignPageNumbers, ref IncludePageNumbers);
            oDoc.Sections[curSectionNum].Range.Paragraphs[1].Alignment =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
            // curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range;
            oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range.Font.Bold = 1;
            oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range.Font.Name = "黑体";
            oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range.Font.Size = 16;
        }
```

```
        private void insertMainBody()
        {
            string one_str = "";
            //插入正文
            curSectionNum = 3;
            oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range.Select();
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range;
            this.Dispatcher.Invoke(() => listbox.Items.Add("正在设置标题样式"));

            object wdFontSizeIndex;
            wdFontSizeIndex = 14;//14 是标题一，一级标题
            //oWordApplic.ActiveDocument.Styles.get_Item(ref wdFontSizeIndex).ParagraphFormat.Alignment
= MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
            oWordApplic.ActiveDocument.Styles.get_Item(ref wdFontSizeIndex).Font.Name = "黑体";
            oWordApplic.ActiveDocument.Styles.get_Item(ref wdFontSizeIndex).Font.Size = 16;
            wdFontSizeIndex = 15;//15 是标题二，一级标题
            oWordApplic.ActiveDocument.Styles.get_Item(ref wdFontSizeIndex).Font.Name = "黑体";
            oWordApplic.ActiveDocument.Styles.get_Item(ref wdFontSizeIndex).Font.Size = 15;
            object Style1 = MsWord.WdBuiltinStyle.wdStyleHeading1;
            object Style2 = MsWord.WdBuiltinStyle.wdStyleHeading2;
            oDoc.Sections[curSectionNum].Range.Select();
            currentSelection = oWordApplic.Selection;

            //读入第一章文本信息
            StreamReader file_content = new StreamReader("content.txt");
            one_str = file_content.ReadLine();//一级标题
            currentSelection.TypeText(one_str);
            currentSelection.TypeParagraph();
            one_str = file_content.ReadLine();//二级标题
            currentSelection.TypeText(one_str);
            currentSelection.TypeParagraph();
            one_str = file_content.ReadLine();//正文
            while (one_str != null)
            {
                currentSelection.TypeText(one_str);
                currentSelection.TypeParagraph();
                one_str = file_content.ReadLine();//正文
            }
            file_content.Close();
            //段落对齐方式
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range;
            curRange.set_Style(ref Style1);
            oDoc.Sections[curSectionNum].Range.Paragraphs[1].Alignment                                    =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[2].Range;
            curRange.set_Style(ref Style2);
            //第一章正文格式
            for (int i = 3; i < oDoc.Sections[curSectionNum].Range.Paragraphs.Count; i++)
            {
                curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[i].Range;
                curRange.Font.Name = "宋体";
                curRange.Font.Size = 12;
                //设置段落为多倍行距
```

```
                oDoc.Sections[curSectionNum].Range.Paragraphs[i].LineSpacingRule                    =
MsWord.WdLineSpacing.wdLineSpaceMultiple;
                oDoc.Sections[curSectionNum].Range.Paragraphs[i].LineSpacing = 15f;
                //首行缩进 2 字符
                oDoc.Sections[curSectionNum].Range.Paragraphs[i].IndentFirstLineCharWidth(2);

            }
        }

        private void insertTable()
        {
            object Style1 = MsWord.WdBuiltinStyle.wdStyleHeading1;
            this.Dispatcher.Invoke(() => listbox.Items.Add("正在插入第二章内容"));
            curSectionNum = 4;
            oDoc.Sections[curSectionNum].Range.Select();
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range;
            currentSelection = oWordApplic.Selection;
            currentSelection.TypeText("2 表格");
            currentSelection.TypeParagraph();
            currentSelection.TypeText("表格示例");
            currentSelection.TypeParagraph();
            currentSelection.TypeParagraph();
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[3].Range;
            oDoc.Sections[curSectionNum].Range.Paragraphs[3].Range.Select();
            currentSelection = oWordApplic.Selection;
            MsWord.Table oTable;
            oTable = curRange.Tables.Add(curRange, 5, 3);
            oTable.Range.Paragraphs.Alignment = MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
            oTable.Range.Font.Name = "宋体";
            oTable.Range.Font.Size = 16;
            oTable.Range.Cells.VerticalAlignment                                                    =
MsWord.WdCellVerticalAlignment.wdCellAlignVerticalCenter;
            oTable.Range.Rows.Alignment = MsWord.WdRowAlignment.wdAlignRowCenter;
            oTable.Columns[1].Width = 80;
            oTable.Columns[2].Width = 180;
            oTable.Columns[3].Width = 80;
            oTable.Cell(1, 1).Range.Text = "字段";
            oTable.Cell(1, 2).Range.Text = "描述";
            oTable.Cell(1, 3).Range.Text = "数据类型";
            oTable.Cell(2, 1).Range.Text = "ProductID";
            oTable.Cell(2, 2).Range.Text = "产品标识";
            oTable.Cell(2, 3).Range.Text = "字符串";
            oTable.Borders.InsideLineStyle = MsWord.WdLineStyle.wdLineStyleSingle;
            oTable.Borders.OutsideLineStyle = MsWord.WdLineStyle.wdLineStyleSingle;
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range;
            curRange.set_Style(ref Style1);
            curRange.ParagraphFormat.Alignment                                                      =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
        }

        private void insertImage()
        {
            object Style1 = MsWord.WdBuiltinStyle.wdStyleHeading1;
```

```
            this.Dispatcher.Invoke(() => listbox.Items.Add("正在插入第三章内容"));
            curSectionNum = 5;
            oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range.Select();
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range;
            currentSelection = oWordApplic.Selection;
            currentSelection.TypeText("3 图片");
            currentSelection.TypeParagraph();
            currentSelection.TypeText("图片示例");
            currentSelection.TypeParagraph();
            //listbox.Items.Add(System.Environment.CurrentDirectory.ToString());

currentSelection.InlineShapes.AddPicture(System.Environment.CurrentDirectory.ToString()+"\\WHU.jpg");
            curRange.set_Style(Style1);
            curRange.ParagraphFormat.Alignment                                                    =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
        }

        private void insertArtWord()
        {
            object missing = Missing.Value;
            this.Dispatcher.Invoke(() => listbox.Items.Add("正在插入第四章内容(艺术字)"));
            object Style1 = MsWord.WdBuiltinStyle.wdStyleHeading1;
            curSectionNum = 6;
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range;
            oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range.Select();
            currentSelection = oWordApplic.Selection;
            currentSelection.TypeText("4 艺术字");
            currentSelection.TypeParagraph();
            currentSelection.TypeText("艺术字示例");
            currentSelection.TypeParagraph();
            curRange.set_Style(Style1);
            curRange.ParagraphFormat.Alignment                                                    =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[3].Range;
            oDoc.Shapes.AddTextEffect(Microsoft.Office.Core.MsoPresetTextEffect.msoTextEffect17, "在最美
的大学成就最美的人生", "Arial Black", 24, Microsoft.Office.Core.MsoTriState.msoTrue,
Microsoft.Office.Core.MsoTriState.msoTrue, (float)120.0, (float)20.0, curRange);
            curRange.ParagraphFormat.Alignment                                                    =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
        }



        private void insertReference()
        {
            this.Dispatcher.Invoke(() => listbox.Items.Add("正在生成参考文献"));
            object Style1 = MsWord.WdBuiltinStyle.wdStyleHeading1;
            curSectionNum = 7;
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range;
            oDoc.Sections[curSectionNum].Range.Paragraphs[1].Range.Select();
            currentSelection = oWordApplic.Selection;
            currentSelection.TypeText("参考文献");
            currentSelection.TypeParagraph();
```

```
            curRange.ParagraphFormat.Alignment                                          =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
            curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[2].Range;
            oDoc.Sections[curSectionNum].Range.Paragraphs[2].Range.Select();
            currentSelection = oWordApplic.Selection;
            StreamReader file_reference = new StreamReader("reference.txt");
            string one_str = file_reference.ReadLine();
            while (one_str != null)
            {
                currentSelection.TypeText(one_str);
                currentSelection.TypeParagraph();
                one_str = file_reference.ReadLine();
            }
            file_reference.Close();
            for (int i = 1; i < oDoc.Sections[curSectionNum].Range.Paragraphs.Count; i++)
            {
                curRange = oDoc.Sections[curSectionNum].Range.Paragraphs[i].Range;
                if (i == 1)
                {
                    curRange.set_Style(Style1);
                    curRange.Paragraphs.Alignment                                       =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
                }
                else
                {
                    curRange.Font.Name = "宋体";
                    curRange.Font.Size = 10;
                    curRange.Paragraphs.Alignment                                       =
MsWord.WdParagraphAlignment.wdAlignParagraphJustify;
                }
                //listbox.Items.Add(i.ToString());
            }
        }

        private void insertHeader()
        {
            this.Dispatcher.Invoke(() => listbox.Items.Add("正在设置页码"));
            //for (curSectionNum=1; curSectionNum < 7; curSectionNum++)
            //{
            //    //进入页脚视图
            //    oDoc.Sections[curSectionNum].Range.Select();
            //
oDoc.Sections[curSectionNum].Headers[MsWord.WdHeaderFooterIndex.wdHeaderFooterPrimary].Range.Border
s[MsWord.WdBorderType.wdBorderBottom].LineStyle = MsWord.WdLineStyle.wdLineStyleNone;
            //                                  oWordApplic.ActiveWindow.ActivePane.View.SeekView       =
MsWord.WdSeekView.wdSeekCurrentPageFooter;
            //                              oWordApplic.Selection.HeaderFooter.PageNumbers.NumberStyle  =
MsWord.WdPageNumberStyle.wdPageNumberStyleUppercaseRoman;
            //    oWordApplic.Selection.HeaderFooter.PageNumbers.RestartNumberingAtSection = true;
            //    oWordApplic.Selection.HeaderFooter.PageNumbers.StartingNumber = 1;
            //    oWordApplic.ActiveWindow.ActivePane.Selection.InsertAfter(curSectionNum.ToString());
            //                                      oWordApplic.Selection.ParagraphFormat.Alignment     =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
```

```
                //                                  oWordApplic.ActiveWindow.ActivePane.View.SeekView           =
MsWord.WdSeekView.wdSeekMainDocument;
                //}

                curSectionNum = 1;
                oDoc.Sections[curSectionNum].Range.Select();
                //进入页脚视图

oDoc.Sections[curSectionNum].Headers[MsWord.WdHeaderFooterIndex.wdHeaderFooterPrimary].Range.Border
s[MsWord.WdBorderType.wdBorderBottom].LineStyle = MsWord.WdLineStyle.wdLineStyleNone;
                oWordApplic.ActiveWindow.ActivePane.View.SeekView                                               =
MsWord.WdSeekView.wdSeekCurrentPageFooter;
                oWordApplic.Selection.HeaderFooter.PageNumbers.NumberStyle                                      =
MsWord.WdPageNumberStyle.wdPageNumberStyleUppercaseRoman;
                oWordApplic.Selection.HeaderFooter.PageNumbers.RestartNumberingAtSection = true;
                oWordApplic.Selection.HeaderFooter.PageNumbers.StartingNumber = 1;
                oWordApplic.ActiveWindow.ActivePane.Selection.InsertAfter("1");
                //切换到文档
                oWordApplic.ActiveWindow.ActivePane.View.SeekView                                               =
MsWord.WdSeekView.wdSeekMainDocument;
                this.Dispatcher.Invoke(() => listbox.Items.Add("正在设置第 2 节的目录页眉页脚内容"));


                curSectionNum = 2;
                oDoc.Sections[curSectionNum].Range.Select();
                //进入页脚视图

oDoc.Sections[curSectionNum].Headers[MsWord.WdHeaderFooterIndex.wdHeaderFooterPrimary].Range.Border
s[MsWord.WdBorderType.wdBorderBottom].LineStyle = MsWord.WdLineStyle.wdLineStyleNone;
                oWordApplic.ActiveWindow.ActivePane.View.SeekView                                               =
MsWord.WdSeekView.wdSeekCurrentPageFooter;
                oWordApplic.Selection.HeaderFooter.PageNumbers.NumberStyle                                      =
MsWord.WdPageNumberStyle.wdPageNumberStyleUppercaseRoman;
                oWordApplic.Selection.HeaderFooter.PageNumbers.RestartNumberingAtSection = true;
                oWordApplic.Selection.HeaderFooter.PageNumbers.StartingNumber = 1;
                oWordApplic.ActiveWindow.ActivePane.Selection.InsertAfter("2");
                //切换到文档
                oWordApplic.ActiveWindow.ActivePane.View.SeekView                                               =
MsWord.WdSeekView.wdSeekMainDocument;
                this.Dispatcher.Invoke(() => listbox.Items.Add("正在设置第 3 节的目录页眉页脚内容"));


                //curSectionNum = 3;
                //oDoc.Sections[curSectionNum].Range.Select();
                ////进入页脚视图

//oDoc.Sections[curSectionNum].Headers[MsWord.WdHeaderFooterIndex.wdHeaderFooterPrimary].Range.Bord
ers[MsWord.WdBorderType.wdBorderBottom].LineStyle = MsWord.WdLineStyle.wdLineStyleNone;
                //oWordApplic.ActiveWindow.ActivePane.View.SeekView                                             =
MsWord.WdSeekView.wdSeekCurrentPageFooter;
                //oWordApplic.Selection.HeaderFooter.PageNumbers.NumberStyle                                    =
MsWord.WdPageNumberStyle.wdPageNumberStyleUppercaseRoman;
                //oWordApplic.Selection.HeaderFooter.PageNumbers.RestartNumberingAtSection = true;
```

```
            //oWordApplic.Selection.HeaderFooter.PageNumbers.StartingNumber = 1;
            ////切换到文档
            //oWordApplic.ActiveWindow.ActivePane.View.SeekView                                    =
MsWord.WdSeekView.wdSeekMainDocument;
            //listbox.Items.Add("正在设置第 4 节的目录页眉页脚内容");
            //curSectionNum = 4;
            //oDoc.Sections[curSectionNum].Range.Select();
            ////进入页脚视图

//oDoc.Sections[curSectionNum].Headers[MsWord.WdHeaderFooterIndex.wdHeaderFooterPrimary].Range.Bord
ers[MsWord.WdBorderType.wdBorderBottom].LineStyle = MsWord.WdLineStyle.wdLineStyleNone;
            //oWordApplic.ActiveWindow.ActivePane.View.SeekView                                    =
MsWord.WdSeekView.wdSeekCurrentPageFooter;
            //oWordApplic.Selection.HeaderFooter.PageNumbers.NumberStyle                           =
MsWord.WdPageNumberStyle.wdPageNumberStyleUppercaseRoman;
            //oWordApplic.Selection.HeaderFooter.PageNumbers.RestartNumberingAtSection = true;
            //oWordApplic.Selection.HeaderFooter.PageNumbers.StartingNumber = 1;
            ////切换到文档
            //oWordApplic.ActiveWindow.ActivePane.View.SeekView                                    =
MsWord.WdSeekView.wdSeekMainDocument;
            //listbox.Items.Add("正在设置第 4 节的目录页眉页脚内容");
            ////添加页码域
            //object fieldpage = MsWord.WdFieldType.wdFieldPage;
            //oWordApplic.Selection.Fields.Add(oWordApplic.Selection.Range, fieldpage);
            //oWordApplic.Selection.ParagraphFormat.Alignment                                      =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;

        }


        public void InsertFooter()
        {

            foreach (MsWord.Section wordSection in oWordApplic.ActiveDocument.Sections)
            {
                MsWord.Range                                footerRange                              =
wordSection.Footers[MsWord.WdHeaderFooterIndex.wdHeaderFooterPrimary].Range;
                //footerRange.Font.ColorIndex = MsWord.WdColorIndex.wdDarkRed;
                footerRange.Font.Size = 15;
                footerRange.Fields.Add(footerRange, MsWord.WdFieldType.wdFieldPage);
                //footerRange.Text = "页脚  页脚";
                footerRange.ParagraphFormat.Alignment                                            =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
            }

            foreach (MsWord.Section section in oWordApplic.ActiveDocument.Sections)
            {
                MsWord.Range                                headerRange                              =
section.Headers[MsWord.WdHeaderFooterIndex.wdHeaderFooterPrimary].Range;
                headerRange.Fields.Add(headerRange, MsWord.WdFieldType.wdFieldPage);
                headerRange.ParagraphFormat.Alignment                                            =
MsWord.WdParagraphAlignment.wdAlignParagraphCenter;
                headerRange.Font.ColorIndex = MsWord.WdColorIndex.wdDarkRed;
```

```
            headerRange.Text = "武汉大学";
            headerRange.Font.Size = 20;
        }

    }

    private void WORD_COM()
    {
        //MsWord.Application oWordApplic;
        //MsWord.Document oDoc;
        //string doc_file_name = Directory.GetCurrentDirectory() + @"\content.doc";
        //所有 Word 对象的 COM 方法调用必须在异常处理代码块中
        try
        {
            insertAbstract();
            insertCatalog();
            insertMainBody();
            insertTable();
            insertImage();
            insertArtWord();
            insertReference();
            InsertFooter();
            //以指定文件名保存文档
            object format = MsWord.WdSaveFormat.wdFormatDocument;
            object path = doc_file_name;
            this.Dispatcher.Invoke(() => listbox.Items.Add("正在保存 Word 文档"));
            //listbox.Items.Add("正在保存 Word 文档");
            oDoc.SaveAs(ref path, ref format);
            oDoc.Close();
            this.Dispatcher.Invoke(() => listbox.Items.Add("正在释放 COM 资源"));
            System.Runtime.InteropServices.Marshal.ReleaseComObject(oDoc);
            oDoc = null;
            oWordApplic.Quit();
            System.Runtime.InteropServices.Marshal.ReleaseComObject(oWordApplic);
            oWordApplic = null;
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
        finally
        {
            this.Dispatcher.Invoke(() => listbox.Items.Add("正在结束 Word 进程"));
            System.Diagnostics.Process[] AllProcess = System.Diagnostics.Process.GetProcesses();
            for (int j = 0; j < AllProcess.Length; j++)
            {
                if (String.Compare(AllProcess[j].ProcessName, "WINWORD") == 0)
                {
                    if (AllProcess[j].Responding && !AllProcess[j].HasExited)
                    {
                        AllProcess[j].Kill();
                    }
                }
```

```
            }
        }
    }

    private void WORD_COM_Button_Click(object sender, RoutedEventArgs e)
    {
        Thread thread = new Thread(WORD_COM);
        thread.Start();
    }


    private void EXCEL_COM_Button_Click(object sender, RoutedEventArgs e)
    {
        MsExcel.Application oExcApp;
        MsExcel.Workbook oExcBook;
        try
        {
            oExcApp = new MsExcel.Application();
            oExcBook = oExcApp.Workbooks.Add(true);
            MsExcel.Worksheet worksheet1 = (MsExcel.Worksheet)oExcBook.Worksheets["sheet1"];
            worksheet1.Activate();
            oExcApp.Visible = false;
            oExcApp.DisplayAlerts = false;
            MsExcel.Range range1 = worksheet1.get_Range("B1", "H1");
            range1.Columns.ColumnWidth = 8;
            range1.Columns.RowHeight = 20;
            range1.Merge(false);
            range1.VerticalAlignment = MsExcel.XlVAlign.xlVAlignCenter;
            range1.HorizontalAlignment = MsExcel.XlHAlign.xlHAlignCenter;
            range1.Borders.LineStyle = MsExcel.XlLineStyle.xlContinuous;
            //range1.Font.Color = System.Drawing.ColorTranslator
            range1.Font.Size = 20;
            range1.Font.Bold = true;
            worksheet1.Cells[1, 2] = "任务分布";
            worksheet1.Cells[3, 1] = "任务编号";
            worksheet1.Cells[3, 2] = "维度";
            worksheet1.Cells[3, 3] = "经度";
            worksheet1.Columns[1].ColumnWidth = 12;
            StreamReader sw = new StreamReader("list.csv");
            string a_str;
            string[] str_list;
            int i = 4;
            a_str = sw.ReadLine();
            this.Dispatcher.Invoke(() => listbox.Items.Add("正在写入数据"));
            while (a_str != null)
            {
                str_list = a_str.Split("\t".ToArray());
                worksheet1.Cells[i, 1] = str_list[0];
                worksheet1.Cells[i, 2] = str_list[1];
                worksheet1.Cells[i, 3] = str_list[2];
                i++;
                a_str = sw.ReadLine();
```

```
                }
                sw.Close();


                //MsExcel.Chart chart = new MsExcel.Chart();
                //Microsoft.Office.Interop.Excel.Axis xAxis =
chart.Axes(Microsoft.Office.Interop.Excel.XlAxisType.xlCategory,
Microsoft.Office.Interop.Excel.XlAxisGroup.xlPrimary);
                //Microsoft.Office.Interop.Excel.Axis yAxis =
chart.Axes(Microsoft.Office.Interop.Excel.XlAxisType.xlValue,
Microsoft.Office.Interop.Excel.XlAxisGroup.xlPrimary);
                //Microsoft.Office.Interop.Excel.Axis zAxis = null;
                //xAxis.HasTitle = true;
                //xAxis.AxisTitle.Text = "X 轴标题";
                //yAxis.HasTitle = true;
                //yAxis.AxisTitle.Text = "Y 轴标题";
                this.Dispatcher.Invoke(() => listbox.Items.Add("正在插入散点图"));
                MsExcel.Shape chartshape = worksheet1.Shapes.AddChart2(Type.Missing,
MsExcel.XlChartType.xlXYScatter, 200, 130, 380, 250, Type.Missing);
                MsExcel.Range range = worksheet1.get_Range("B8:C8", Type.Missing);
                chartshape.Chart.SetSourceData(range, Type.Missing);

                //MsExcel.Shape chartshape = worksheet1.Shapes.AddChart2(Type.Missing, chart, 200, 130,
380, 250, Type.Missing);
                //chartshape.Chart.Axes();

                //chartshape.Chart.Series[0].CategoryLabels = worksheet1.Range["B2:B10"];
                //chartshape.Chart.SeriesCollection.range = worksheet1.Range["C2:C10"];
                //chartshape.Chart.SetSourceData(range, Microsoft.Office.Interop.Excel.XlRowCol.xlRows);
                object file_name = Directory.GetCurrentDirectory() + @"\one.xlsx";
                oExcBook.Close(true, file_name, null);
                this.Dispatcher.Invoke(() => listbox.Items.Add("数据写入完成"));
            }
            catch(Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }

    }
}

namespace CSharpCallDLL
{
    /// <summary>
    /// Page4.xaml 的交互逻辑
    /// </summary>
    public partial class Page4 : Page
    {
        [DllImport("user32.dll", EntryPoint = "FindWindow")]
        private extern static IntPtr FindWindow(string lpClassName, string lpWindowName);
        [DllImport("user32.dll ", EntryPoint = "SendMessage")]
        private static extern IntPtr SendMessage(IntPtr hWnd, int Msg, IntPtr wParam, IntPtr lParam);
```

```
/*异步重定向*/
public static StreamWriter cmdStreamInput = null;
public static StringBuilder cmdOutput = null;
public const int WM_COPYDATA = 0x004A;
public Process process1 = null;
/*客户端服务器通信*/
public Thread clientThread = null;//客户端线程
public Thread serverThread = null;//服务器端线程
public Process clientProcess = null;
public Process serverProcess = null;
public NamedPipeServerStream pipServer = null;
public NamedPipeClientStream pipClient = null;
public StreamWriter sw = null;
public StreamReader sr = null;
public string message;
public bool isconnect;

/*生产者消费者问题*/
public int buffer_size;
public int num;//产品数目
public Semaphore full;
public Semaphore empty;
public Mutex mutex;//互斥访问缓冲区
public bool isStop = false;
public Page4()
{
    InitializeComponent();
    HwndSource hWndSource;
    //WindowInteropHelper wih = new WindowInteropHelper(Application.Current.MainWindow);
    IntPtr WINDOW_HANDLER = FindWindow(null, "Demo");
    hWndSource = HwndSource.FromHwnd(WINDOW_HANDLER);
    //添加处理程序
    hWndSource.AddHook(MainWindowProc);
}


private IntPtr MainWindowProc(IntPtr hwnd, int msg, IntPtr wParam, IntPtr lParam, ref bool handled)
{
    //Console.WriteLine(WM_COPYDATA);
    Console.WriteLine(wParam+"xxx");
    Console.WriteLine(msg);
    switch (msg)
    {
        //???
        case WM_COPYDATA:
            //textboxresult.Text = cmdOutput.ToString();
            Console.WriteLine("哈哈哈");
            break;
        default:
            Console.WriteLine(msg);
            if (cmdOutput != null)
            {
```

```
                textboxresult.Text = cmdOutput.ToString();
            }
            break;
        }
        return hwnd;
    }
    /*同步调用 Ping 重定向*/
    private void SynPing_Button_Click(object sender, RoutedEventArgs e)
    {
        process1 = new Process();
        process1.StartInfo.FileName = "cmd.exe";
        // 是否使用外壳程序
        process1.StartInfo.UseShellExecute = false;
        // 是否在新窗口中启动该进程的值
        process1.StartInfo.CreateNoWindow = true;
        // 重定向输入流
        process1.StartInfo.RedirectStandardInput = true;
        // 重定向输出流
        process1.StartInfo.RedirectStandardOutput = true;
        //使 ping 命令执行九次
        string strCmd = "ping www.whu.edu.cn -n  9";
        process1.Start();
        process1.StandardInput.WriteLine(strCmd);
        process1.StandardInput.WriteLine("exit");
        // 获取输出信息
        textboxresult.Text = process1.StandardOutput.ReadToEnd();
        process1.WaitForExit();
        process1.Close();
    }

    /*异步调用重定向*/
    private void startCmd(string strCmd)
    {
        process1 = new Process();
        process1.StartInfo.FileName = "cmd.exe";
        ///process.StartInfo.Arguments = strCmd;
        // 是否使用外壳程序
        process1.StartInfo.UseShellExecute = false;
        // 是否在新窗口中启动该进程的值
        process1.StartInfo.CreateNoWindow = true;
        // 重定向输入流
        process1.StartInfo.RedirectStandardInput = true;
        // 重定向输出流
        process1.StartInfo.RedirectStandardOutput = true;
        cmdOutput = new StringBuilder("");
        // 注册接受到数据的事件回调函数
        process1.OutputDataReceived += new DataReceivedEventHandler(strOutputHandler);
        //process1.OutputDataReceived += new DataReceivedEventHandler(send);
        process1.EnableRaisingEvents = true;
        process1.Start();
        //Console.WriteLine(process1.ProcessName);
        process1.StandardInput.WriteLine(strCmd);
        process1.StandardInput.WriteLine("exit");
```

```
        process1.BeginOutputReadLine();
        process1.Close();
    }

    private void send()
    {
        IntPtr WINDOW_HANDLER = FindWindow(null, "Demo");
        //Console.WriteLine(WINDOW_HANDLER);
        //Process[] ps = Process.GetProcesses();
        SendMessage(WINDOW_HANDLER, WM_COPYDATA, (IntPtr)0, (IntPtr)0);
    }

    private void ASynPing_Button_Click(object sender, RoutedEventArgs e)
    {
        string strCmd = "ping www.whu.edu.cn -n    9";
        startCmd(strCmd);
    }

    private    void strOutputHandler(object sendingProcess,
      DataReceivedEventArgs outLine){
        cmdOutput.AppendLine(outLine.Data);

        //textboxresult.Text = cmdOutput.ToString();
        //this.Dispatcher.Invoke(() => textboxresult.Text = cmdOutput.ToString());
        // send();
        //通过 FindWindow API 的方式找到目标进程句柄，然后发送消息
        IntPtr WINDOW_HANDLER = FindWindow(null, "Demo");


        //Console.WriteLine(WINDOW_HANDLER);
        //Process[] ps = Process.GetProcesses();
        SendMessage(WINDOW_HANDLER, WM_COPYDATA, (IntPtr)0, (IntPtr)0);
        //Console.WriteLine(WM_COPYDATA);
    }

    private void ASynGetMac_Button_Click(object sender, RoutedEventArgs e)
    {
        if (process1 != null)
        {
            process1.StartInfo.RedirectStandardOutput = false;
        }
        send();
        string strCmd = "getmac";
        startCmd(strCmd);
    }

    private void ASynShutDown_Button_Click(object sender, RoutedEventArgs e)
    {
        string strCmd = "shutdown";
        startCmd(strCmd);
    }
```

```
/*模拟服务器客户端通信问题*/
private void Connect_Button_Click(object sender, RoutedEventArgs e)
{
    // isconnect = true;
     clientThread = new Thread(clientRun);
     clientThread.IsBackground = true;
     clientThread.Start();
}
private void NewServer_Button_Click(object sender, RoutedEventArgs e)
{
     simulateCS();
}
private void DisConnect_Button_Click(object sender, RoutedEventArgs e)
{
     //sw.Close();
     //sr.Close();
     //pipClient.Close();
     //pipServer.Close();
     //isconnect = false;
     //sw = null;
     //sr = null;
     //pipClient = null;
     //pipServer = null;
     serverProcess.Close();
     textserver.AppendText("已经断开连接...\n");
}

//private void simulateCS()
//{
//      //客户端写，服务端读
//      clientThread = new Thread(clientRun);
//      clientThread.IsBackground = true;
//      serverThread = new Thread(serverRun);
//      serverThread.IsBackground = true;
//      clientThread.Start();
//      serverThread.Start();
//}
private void simulateCS()
{
     //客户端写，服务端读
     serverProcess = new Process();
     serverProcess.StartInfo.FileName = @"..\..\..\ServerWindow\bin\Release\ServerWindow.exe";
     serverProcess.Start();
}

private void clientRun()
{
     try
     {
          pipClient = new NamedPipeClientStream(".", "testpipe", PipeDirection.InOut);
          pipClient.Connect();
          try
          {
```

```
                sw = new StreamWriter(pipClient);
                sw.AutoFlush = true;
            }
            catch (IOException e)
            {
                MessageBox.Show(e.Message);
            }
        }catch (System.IO.IOException)
        {
            MessageBox.Show("已经建立了连接不能重复建立");
        }
    }


    private void serverRun()
    {
        try
        {
            pipServer = new NamedPipeServerStream("testpipe", PipeDirection.InOut);
            this.Dispatcher.BeginInvoke(new Action(() => textserver.AppendText("等待客户端连接\n")));
            pipServer.WaitForConnection();
            this.Dispatcher.BeginInvoke(new Action(() => textserver.AppendText("连接成功\n")));
            sr = new StreamReader(pipServer);
            while (isconnect)
            {
                message = sr.ReadLine();
                this.Dispatcher.BeginInvoke(new Action(() => textserver.AppendText(message + "\n")));
            }
        }
        catch (System.IO.IOException)
        {
            MessageBox.Show("已经建立了连接不能重复建立");
        }

    }
    private void Send_Button_Click(object sender, RoutedEventArgs e)
    {
        try
        {
            sw.WriteLine(textclient.Text);
        }
        catch (Exception)
        {
            MessageBox.Show("请先进行客户端和服务器的连接");
        }
    }


    /*模拟生产者消费者问题*/
    private void PC_Button_Click(object sender, RoutedEventArgs e)
    {
        //if (process1 != null)
        //{
```

```
        //      //process1.Close();
        //      //process1 = null;
        //      cmdOutput = new StringBuilder("");
        //}
        textboxresult.Text = "";
        //textboxresult
        isStop = false;
        //设置缓冲区大小
        //buffer_size = 5;
        try
        {
            buffer_size = int.Parse(textbuffer.Text);
        }catch(Exception ex)
        {
            MessageBox.Show(ex.Message);
            return;
        }
        //产品数目
        num = 0;
        //初始 full = 0;empty=buffer_size
        full = new Semaphore(0, buffer_size);
        empty = new Semaphore(buffer_size, buffer_size);
        mutex = new Mutex();
        try
        {
            int producer_num = int.Parse(textproducer.Text);
            Thread[] producerThread = new Thread[producer_num];
            int consumer_num = int.Parse(textconsumer.Text);
            Thread[] consumerThread = new Thread[consumer_num];
            for (int i = 0; i < producer_num; i++)
            {
                producerThread[i] = new Thread(new ParameterizedThreadStart(producerRun));
                producerThread[i].IsBackground = true;
                producerThread[i].Start(i+1);
            }

            for (int j = 0; j < consumer_num; j++)
            {
                consumerThread[j] = new Thread(new ParameterizedThreadStart(consumerRun));
                consumerThread[j].IsBackground = true;
                consumerThread[j].Start(j+1);
            }
        }catch(Exception ex)
        {
            MessageBox.Show(ex.Message);
        }

    }

    private void Stop_Button_Click(object sender, RoutedEventArgs e)
    {
        isStop = true;
    }
```

```
        private void producerRun(object id)
        {
            while (!isStop)
            {
                empty.WaitOne();
                mutex.WaitOne();
                //生产
                this.Dispatcher.Invoke(() => { textboxresult.AppendText("Product:" + num + " Producer" +
id.ToString() + " produce\n");});
                num++;
                mutex.ReleaseMutex();
                full.Release();
            }
        }

        private void consumerRun(object id)
        {
            while (!isStop)
            {
                full.WaitOne();
                mutex.WaitOne();
                //消费
                this.Dispatcher.Invoke(() => { textboxresult.AppendText("Product:" + num + " Consumer" +
id.ToString() + " consume\n");});
                num--;
                mutex.ReleaseMutex();
                empty.Release();
            }
        }
    }
}

namespace CSharpCallDLL
{
    /// <summary>
    /// Page5.xaml 的交互逻辑
    /// </summary>
    ///
    /*Handler:处理事件的函数,参数包括：1.发送者2.参数
     EventArgs:事件的参数类。包含事件信息
         */
    public class FireEventArgs:EventArgs{

        public string room;
        public int ferocity;
        public FireEventArgs(string room,int ferocity)
        {
            this.room = room;
            this.ferocity = ferocity;
        }
    }

    //将火情处理定义为FireEventHandler 代理(delegate) 类型，这个代理声明的事件的参数列表
    public delegate void FireEventHandler(object sender, FireEventArgs fe);
```

```
    public class FireAlarm
    {
        //定义FireEvent 为FireEventHandler delegate 事件(event) 类型.
        public event FireEventHandler FireEvent;
        //激活事件的方法，创建了FireEventArgs 对象，发起事件，并将事件参数对象传递过去
        public void ActivateFireAlarm(string room, int ferocity)
        {
            FireEventArgs fireArgs = new FireEventArgs(room, ferocity);
            //执行对象事件处理函数指针，必须保证处理函数要和声明代理时的参数列表相同
            FireEvent(this, fireArgs);
        }

    }

    public struct COPYDATASTRUCT
    {
        public IntPtr dwData;
        public int cbData;
        [MarshalAs(UnmanagedType.LPStr)]
        public string lpData;
    }
    public partial class Page5 : Page
    {

        [DllImport("user32.dll", EntryPoint = "FindWindow")]
        private extern static IntPtr FindWindow(string lpClassName, string lpWindowName);
        [DllImport("user32.dll ", EntryPoint = "SendMessage")]
        private static extern IntPtr SendMessage(IntPtr hWnd, int Msg, IntPtr wParam, ref
COPYDATASTRUCT lParam);

        public const int RECEIVE_DATA = 0x05A;
        public string Message = null;

        public FireAlarm fireAlarm;

        public Page5()
        {
            InitializeComponent();
            fireAlarm = new FireAlarm();
            fireAlarm.FireEvent += new FireEventHandler(ExtinguishFire);
        }
        private void ExtinguishFire(object sender, FireEventArgs fe)
        {
            messagebox.Text = "火灾发生地点：" + fe.room + "\n火灾级别：" + fe.ferocity + "\n";
            if (fe.ferocity >= 0 && fe.ferocity <= 3)
            {
                messagebox.AppendText("火灾级别一般，请使用灭火器扑灭");

            }else if(fe.ferocity >= 4 && fe.ferocity <= 6)
            {
                messagebox.AppendText("火灾级别中等，请控制火情然后报警");
            }
            else
            {
```

```
                messagebox.AppendText("火灾较严重，请尽快报警，并注意自身安全");
            }
        }


        private void New_Button_Click(object sender, RoutedEventArgs e)
        {
            Window receivewindow = new ReceiveWindow();
            receivewindow.Show();
        }

        private void Send_Button_Click(object sender, RoutedEventArgs e)
        {
            /*初始化消息发送结构体*/
            COPYDATASTRUCT cOPYDATASTRUCT;
            cOPYDATASTRUCT.dwData = (IntPtr)0;
            cOPYDATASTRUCT.lpData = messagebox.Text;
            cOPYDATASTRUCT.cbData = 0;
            Message = messagebox.Text;
            IntPtr WINDOW_HANDLER = FindWindow(null, "ReceiveWindow");
            SendMessage(WINDOW_HANDLER, RECEIVE_DATA, (IntPtr)0,ref cOPYDATASTRUCT);
        }

        private void Event_Button_Click(object sender, RoutedEventArgs e)
        {
            try
            {
                fireAlarm.ActivateFireAlarm(textroom.Text, int.Parse(textclass.Text));
            }catch(Exception)
            {
                MessageBox.Show("请输入正确的火灾发生地点和严重程度");
            }
        }

        private void NewForm_Button_Click(object sender, RoutedEventArgs e)
        {
            SendForm sendForm = new SendForm();
            sendForm.Show();
            ReceiveForm receiveForm = new ReceiveForm();
            receiveForm.Show();
        }
    }
}
```

```
namespace CSharpCallDLL
{
    /// <summary>
    /// Page6.xaml 的交互逻辑
    /// </summary>
    public partial class Page6 : Page
    {
        public DataTable dt;
        public DataBaseConnect databaseconnect;
        // static string DataSource = @"C:\Users\dell\Desktop\Windows\Grade.xls";
        static string DataSource = "Grade.xls";
        int columnIndex = 0;
        public Page6()
        {
            InitializeComponent();
            this.datagrid.CellEditEnding += Datagrid_CellEditEnding;
            this.datagrid.BeginningEdit += Datagrid_BeginningEdit;
        }

        private void Datagrid_BeginningEdit(object sender, DataGridBeginningEditEventArgs e)
        {
            columnIndex = datagrid.CurrentCell.Column.DisplayIndex;
        }

        private string getPgsqlAttribute()
        {
            string result = null;
            switch (columnIndex)
            {
                case 0:
                    result = "sno";
                    break;
                case 1:
                    result = "sname";
                    break;
                case 2:
                    result = "ssex";
                    break;
                case 3:
                    result = "sage";
                    break;
                case 4:
                    result = "sdept";
                    break;
            }
            return result;
        }

        private string getExcelAttribute()
        {
            string result = null;
            switch (columnIndex)
            {
                case 0:
                    result = "学号";
```

```
                break;
            case 1:
                result = "姓名";
                break;
            case 2:
                result = "数学";
                break;
            case 3:
                result = "物理";
                break;
            case 4:
                result = "历史";
                break;
            case 5:
                result = "政治";
                break;
            case 6:
                result = "体育";
                break;
            case 7:
                result = "艺术";
                break;
        }
        return result;
    }
    private void Datagrid_CellEditEnding(object sender, DataGridCellEditEndingEventArgs e)
    {

        int columnnum = datagrid.Columns.Count;
        DataRowView SelectedElement = (DataRowView)datagrid.SelectedItem;
        string sno = SelectedElement.Row[0].ToString();
        string value = (e.EditingElement as TextBox).Text;
        if (columnnum == 5)
        {
            string sql = "UPDATE student SET "+getPgsqlAttribute() +"='"+ value+ "' WHERE
sno ='" + sno + "'";
            databaseconnect.ExecuteCommand(sql);
            MessageBox.Show("信息修改成功");
            pgsql_refresh();
        }
        else if (columnnum == 8)
        {
            string sql = "UPDATE [Sheet1$] SET "+getExcelAttribute()+"='"+value+"'"+" WHERE
学号='" + sno + "'";
            Console.WriteLine(sql);
            databaseconnect.ExecuteCommand(1, sql);
            MessageBox.Show("信息修改成功");
            excel_refresh();
        }
        else
        {
            return;
        }
    }
```

```
        private void Excel_Click(object sender, RoutedEventArgs e)
        {
            string Provider, ExtendedProperties, HDR;
            int IMEX;
            Provider = "Microsoft.ACE.OLEDB.12.0";
            ExtendedProperties = "Excel 8.0";
            HDR = "NO";
            IMEX = 0;
            databaseconnect = new DataBaseConnect(Provider, DataSource, ExtendedProperties, HDR,
IMEX);
            excel_refresh();
        }
        public void excel_refresh()
        {
            databaseconnect.OleConn.Open();
            string sql = string.Format("SELECT * FROM [{0}] where 学号 is not null", "Sheet1$");
//查询字符串
            OleDbDataAdapter ada = new OleDbDataAdapter(sql, databaseconnect.getConString());
            DataSet set = new DataSet();
            ada.Fill(set);
            dt = set.Tables[0];
            datagrid.ItemsSource = dt.DefaultView;
            databaseconnect.OleConn.Close();
        }

        private void PostGreSQL_Click(object sender, RoutedEventArgs e)
        {
            string server, port, username, password, command, database;
            server = "localhost";
            port = "5432";
            username = "postgres";
            password = "xjy19991012";
            database = "whu";
            databaseconnect = new DataBaseConnect(server, port, password, username, database);
            command = "select * from student";
            var cmd = new NpgsqlCommand(command, databaseconnect.SqlConn);
            NpgsqlDataAdapter da = new NpgsqlDataAdapter(cmd);
            DataTable dt = new DataTable();
            da.Fill(dt);
            datagrid.ItemsSource = dt.DefaultView;
        }

        public void pgsql_refresh()
        {
            string command = "select * from student";
            var cmd = new NpgsqlCommand(command, databaseconnect.SqlConn);
            NpgsqlDataAdapter da = new NpgsqlDataAdapter(cmd);
            DataTable dt = new DataTable();
            da.Fill(dt);
            datagrid.ItemsSource = dt.DefaultView;
        }

        private void NewGrade_Click(object sender, RoutedEventArgs e)
        {
            int columnnum = datagrid.Columns.Count;
```

```
            if (columnnum == 5)
            {
                Window info_window = new InformationWindow(this);
                info_window.Show();
            }
            else if (columnnum == 8)
            {
                Window grade_window = new GradeInfoWindow(this);
                grade_window.Show();
            }
            else
            {
                return;
            }
        }

        private void DeleteGrade_Click(object sender, RoutedEventArgs e)
        {
            int columnnum = datagrid.Columns.Count;
            DataRowView SelectedElement = (DataRowView)datagrid.SelectedItem;
            string sno = SelectedElement.Row[0].ToString();
            if (columnnum == 5)
            {
                string sql = "DELETE from student where sno='" + sno + "'";
                databaseconnect.ExecuteCommand(sql);
                MessageBox.Show("数据删除成功");
                pgsql_refresh();
            }
            else if (columnnum == 8)
            {
                string sql = "UPDATE [Sheet1$] SET 姓名 = NULL,数学=NULL,物理=NULL,历史=NULL,政
治=NULL,体育=NULL,艺术=NULL,学号 = NULL WHERE 学号 ='" + sno + "'";
                databaseconnect.ExecuteCommand(1, sql);
                MessageBox.Show("数据删除成功");
                excel_refresh();
            }
            else
            {
                return;
            }
        }

        private void SearchGrade_Click(object sender, RoutedEventArgs e)
        {
            int columnnum = datagrid.Columns.Count;
            if (columnnum == 5)
            {
                try
                {
                    string sql = "select * from student where sname='" + name_text.Text + "'";
                    var cmd = new NpgsqlCommand(sql, databaseconnect.SqlConn);
                    NpgsqlDataAdapter da = new NpgsqlDataAdapter(cmd);
                    DataTable dt = new DataTable();
                    da.Fill(dt);
                    if (dt.Rows.Count == 0)
```

```
                    {
                        MessageBox.Show("查无此人");
                    }
                    else
                    {
                        datagrid.ItemsSource = dt.DefaultView;
                    }
                }
                catch(Npgsql.NpgsqlException)
                {
                    MessageBox.Show("查无此人");
                }
            }
            else if (columnnum == 8)
            {
                databaseconnect.OleConn.Open();
                string sql = string.Format("SELECT * FROM [{0}] WHERE 姓名
='"+name_text.Text+"'", "Sheet1$"); //查询字符串
                OleDbDataAdapter ada = new OleDbDataAdapter(sql,
databaseconnect.getConString());
                DataSet set = new DataSet();
                ada.Fill(set);
                dt = set.Tables[0];
                if (dt.Rows.Count == 0)
                {
                    MessageBox.Show("查无此人");
                }
                else
                {
                    datagrid.ItemsSource = dt.DefaultView;
                }
                databaseconnect.OleConn.Close();
            }
            else
            {
                return;
            }
        }
    }
}


namespace CSharpCallDLL
{
    /// <summary>
    /// ReceiveWindow.xaml 的交互逻辑
    /// </summary>
    public partial class ReceiveWindow : Window
    {
        [DllImport("user32.dll", EntryPoint = "FindWindow")]
        private extern static IntPtr FindWindow(string lpClassName, string lpWindowName);

        //public const int RECEIVE_DATA = 0x05A;
        public ReceiveWindow()
        {
```

```
            InitializeComponent();
            this.Loaded += Window_Loaded;
        }

        /*界面加载后添加钩子程序*/
        private void Window_Loaded(object sender, RoutedEventArgs e)
        {
            HwndSource hWndSource;
            IntPtr WINDOW_HANDLER = FindWindow(null, "ReceiveWindow");
            hWndSource = HwndSource.FromHwnd(WINDOW_HANDLER);
            hWndSource.AddHook(MainWindowProc);
        }


        private IntPtr MainWindowProc(IntPtr hwnd, int msg, IntPtr wParam, IntPtr lParam, ref
bool handled)
        {
            switch (msg)
            {
                case Page5.RECEIVE_DATA:
                    //COPYDATASTRUCT mystr = new COPYDATASTRUCT();
                    //Type mytype = mystr.GetType();
                    /*从指针转换到指针指向的结构体*/
                    COPYDATASTRUCT MyHookStruct = (COPYDATASTRUCT)Marshal.PtrToStructure(lParam,
typeof(COPYDATASTRUCT));
                    receivebox.Text = MyHookStruct.lpData;
                    break;
                default:
                    break;
            }
            return hwnd;
        }
    }
}
```