yaahp 算法库使用起步

Step 1 解压算法库打包zip文件;

Step 2 打开Visual Studio, 创建一个新的Windows窗体应用程序项目(或其他项目类型, 此文档以此为例), 如图1所示.

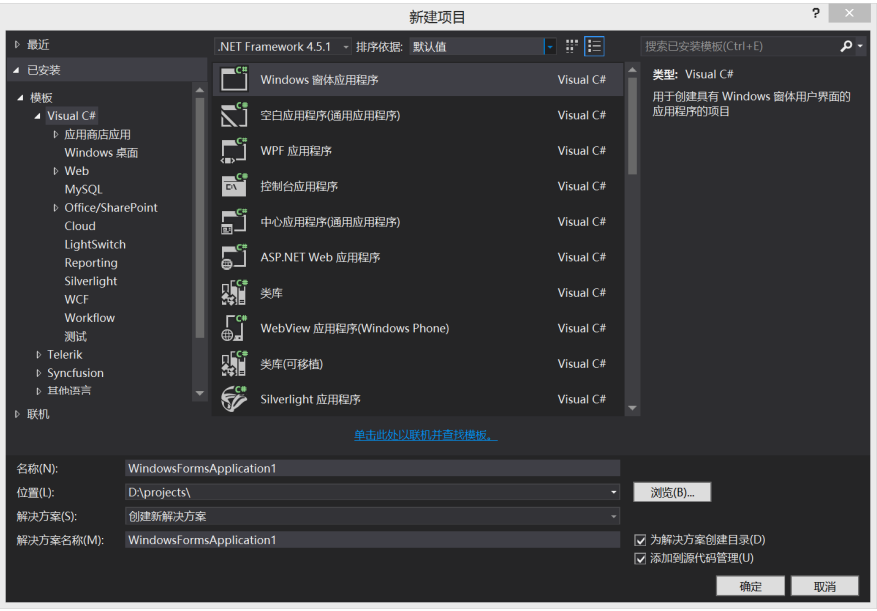

图1 创建个新的Windows窗体应用程序项目

Step 3 从yaahplibs下的x64(如果是64位系统)/x86(如果是32位系统)目录下拷贝”yaahplibs.dll", "psoahp.dll", “License.dll”以及"yaahplib.license"四个文件到新创建的项目目录中;

Step 4 Visual Studio中, 在新建项目的引用上右键, 选择”添加引用…”, 在打开的窗口中点击右下角的”浏览…”按钮, 找到yaahplibs.dll, 点击确定添加引用.

Step 5 打开Windows的”我的电脑”/“文件资源管理器”, 找到刚才拷贝的”yaahplib.license”文件, 将其拖动到Visual Studio中当前项目”解决方案资源管理器”中的工程里. 在”解决方案资源管理器”中选中刚添加进来的”yaahplib.license”文件, 查看它的属性, 并将”复制到输出目录”属性改为”始终复制”, 如图2所示.

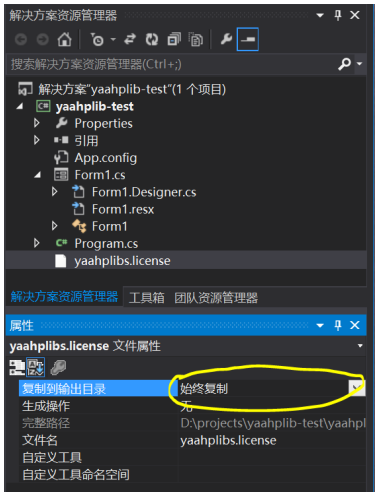

图2 添加”yaahplib.license”并修改为”始终复制”

Step 6 添加了yaahplibs.dll引用以及yaahplib.license文件的项目资源管理器如图3所示.

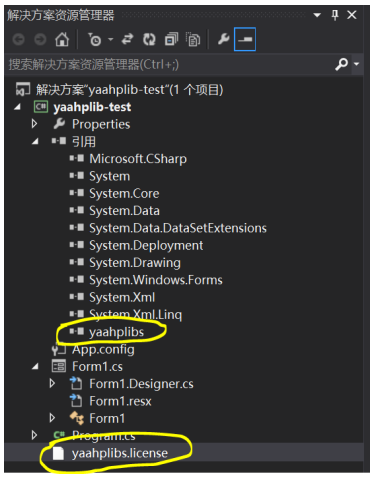

图3 添加yaahplibs.dll引用和"yaahplib.license"后的解决方案资源管理器

Step 7 双击"Form1.cs"打开窗体设计器, 在上面摆放两个按钮, 文本设定为"Load"和"Matrix Count", 如图4 所示.

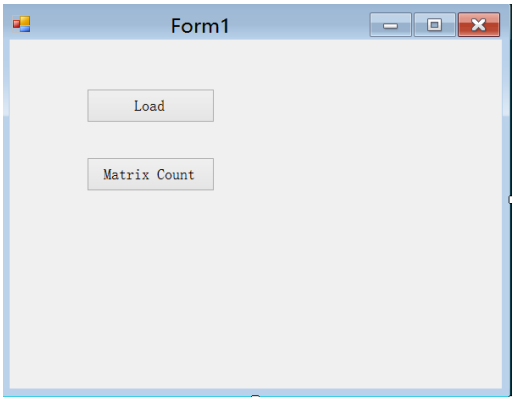

图4 窗体设计器

Step 8 双击"Load"按钮, 创建该按钮按下调用的代码, 并用代码编辑器打开Form1.cs, 在文件头部添加yaahplibs的引用:

```
using yaahplibs;
```

Step 9 向类Form1中添加一个私有的变量:

```
private AhpCalc _ahpcalc = null;
```

Step 10 然后将"Load"按钮的响应函数"button1_Click"改为:

```
private void button1_Click(object sender, EventArgs e)
{
    try
    {
        _ahpcalc = new AhpCalc(@"D:\bridge.single.xml");
        MessageBox.Show("Data loaded from file!");
    }
    catch (Exception ex)
    {
        MessageBox.Show("Data load failed!\nDetail:\n" + ex.Message);
    }
}
```

上面的代码从D:\载入数据, 所以将算法库中的”bridge.single.xml”拷贝到D:\, 或者拷贝到其他位置并相应地修改上面代码中的”D:\bridge.single.xml".

Step 11 打开Form1.cs的窗体设计器, 双击”Matrix Count”按钮, 创建该按钮按下调用的代码, 并用代码编辑器打开Form1.cs, 将”Matrix Count”按钮的响应函数”button1_Click”改为:

```
private void button2_Click(object sender, EventArgs e)
{
    if (_ahpcalc != null)
        MessageBox.Show("Matrices count: " + _ahpcalc.MatricesCount);
}
```

Step 12 “生成” -> “生成解决方案”, 然后执行程序, 就可以看到主窗口了. 点击”Load”按钮, 如果数据文件位置正确(Step 10指定及拷贝), 会弹出消息框提示载入数据成功. 然后点击”Matrix Count”按钮, 将会弹出消息框显示判断矩阵数量.

Step 13 在此基础上, 参考example中的算法库示例, 就可以添加其他功能了.

注意

psoahp.dll是C语言编写的非受管DLL, 需要安装Microsoft VC 2015 运行时(算法库打包文件中的vc-runtime下), 并且区分32位和64位. 如果没有安装VC2015运行时或使用了不正确的dll文件版本, 将会发生错误(图5的错误提示信息).

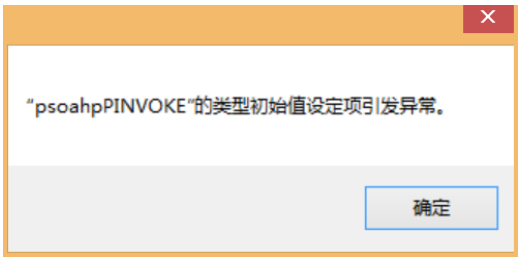

图5 psoahp.dll调用错误

此外, Visual Studio中, Debug模式启动的程序即使在64位Windows系统上仍然以32位执行, 但在Debug模式下可能会出现无论32位还是64位的psoahp.dll都提示上述错误, 这种情况需要修改主程序的编译输出设定才能解决. 在Release模式下使用与系统匹配的psoahp.dll不会出现这个问题.