# 多语言使用指南

## 多语言方案

框架采用[GNU Gettext + Poedit](https://www.cnblogs.com/hont/p/5129806.html)的方式解决实现多语言管理。

1. **关于i18n**

i18n其来源是英文单词 internationalization的首末字符i和n，18为中间的字符数是“国际化”的简称。i10n为资源本地化，全称为Localization，因为首字母I与末字母N之间共10个字母，又称为I10N。资源本地化就是要让这个软件产品使用当地的环境，如语言、文化、使用习惯等一般涉及到具体国家的语言文件，以i10n来表示；而多国语言相关的解决方案以i18n表示。

1. **GNU Gettext**

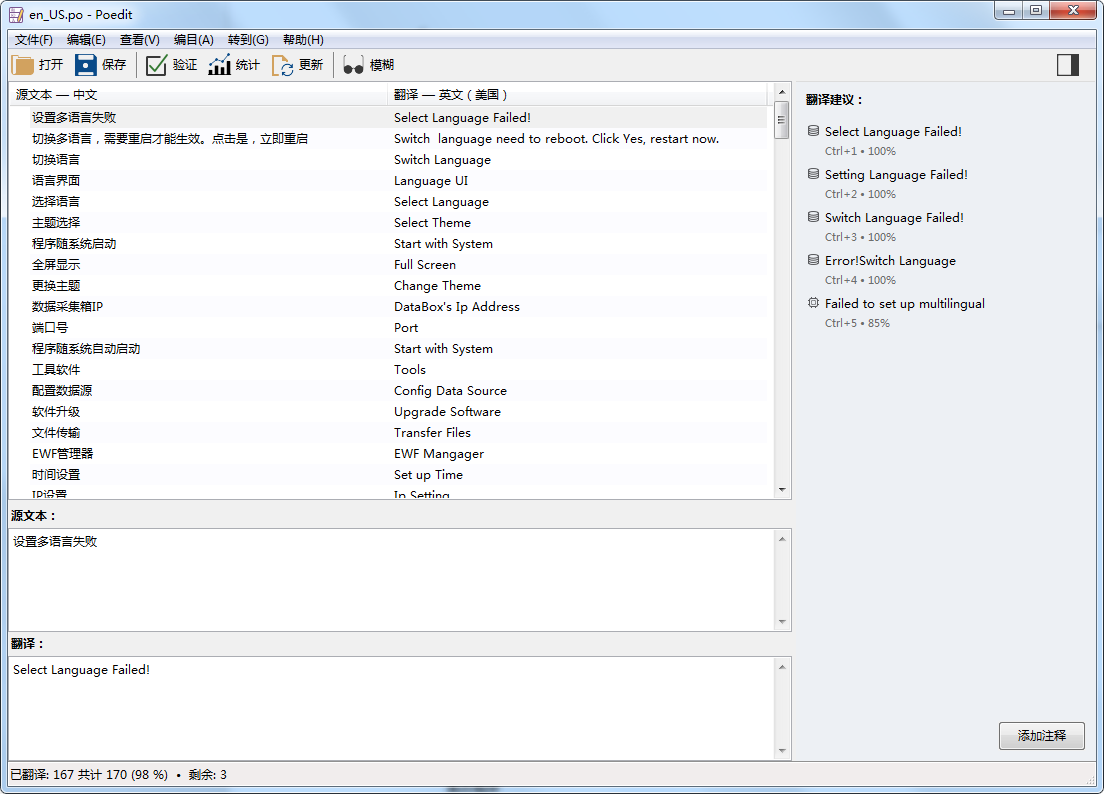
GNU Gettext是软件国际化的解决方案之一，类似的还有IBM的ICU4C。但ICU4C比较庞大。GNU Gettext最早被使用在Linux上，现在已经支持全面Windows平台。

1. **Poedit工具**

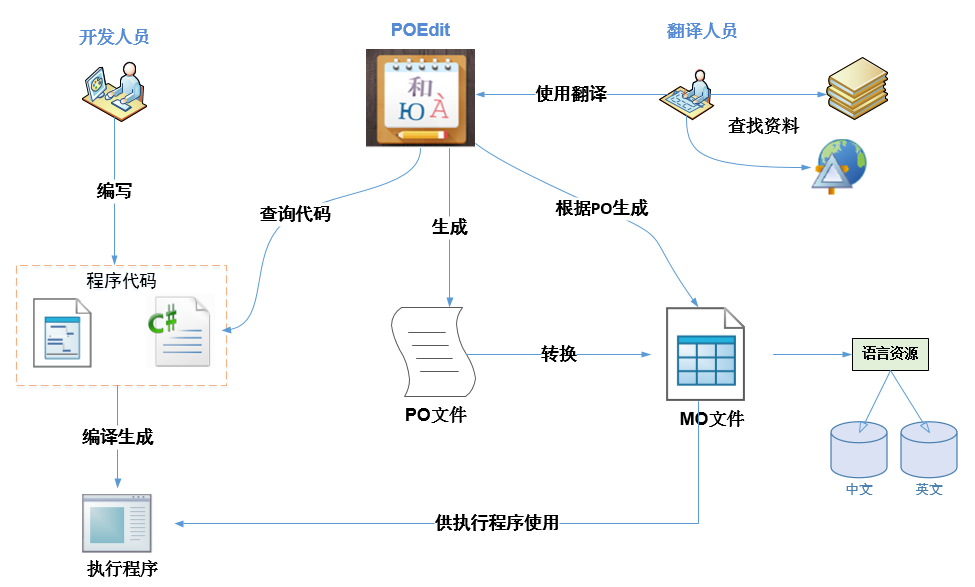
Poedit依赖GNU Gettext，实现了图形化的操作。和普通翻译相比，你不需要再手动输入一次Key，Poedit可以自动进入源文件根据一定的规则提取key。



例如可以将\_(“”)中包含的内容搜索出来形成Key，并可通过Poedit界面工具进行翻译，同时Poedit还提供辅助翻译功能。



1. **编码与翻译分离**

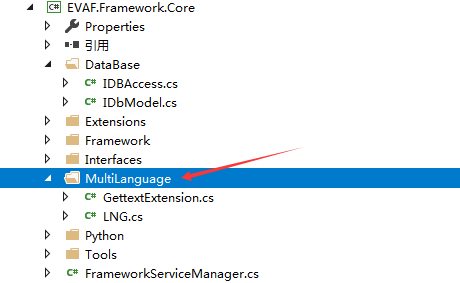


[GNU Gettext + Poedit](https://www.cnblogs.com/hont/p/5129806.html)解决方案真正意义上实现了编码与翻译分离，翻译过程中不需要修改源代码，只需要将MO结果文件修改到程序指定位置，程序启动时将自动获取新的翻译信息。开发人员通过VS代码编写工具编写代码，提供给Poedit工具进行代码查询，并生成执行程序；Poedit对代码进行遍历查询，按照一定规则搜索需要翻译的代码，形成PO文件；PO文件以Key-Value的形式展示给翻译人员，翻译人员只需要使用Poedit对指定的PO文件进行翻译，将Key对应的Value填写正确的翻译；最后通过Poedit翻译生成MO文件，再将MO文件提供给执行程序使用。

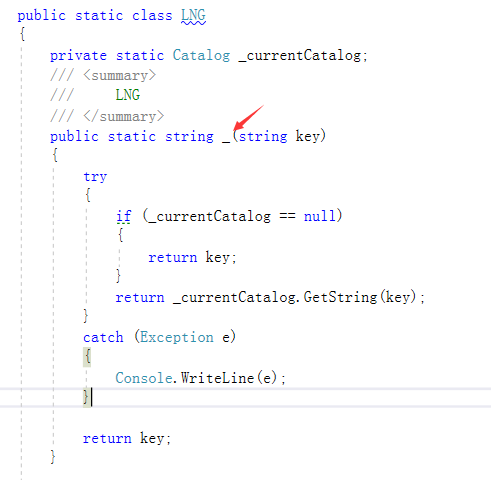
## 多语言编码

### 多语言帮助工具类

多语言帮助工具存放在EVAF.Framework.Core中，包括LNG类和GettextExtension类。



其中LNG类重写了下划线“\_”操作，用于在cs类中代码翻译的帮助，GettextExtension类用于对xaml中代码翻译的帮助。



### XAML中代码翻译写法

1. 确定翻译位置；

如下图，需要翻译的位置为窗口标题“测试窗口”，测试按钮“发送”；



1. 修改XAML中代码。

当前代码如下：

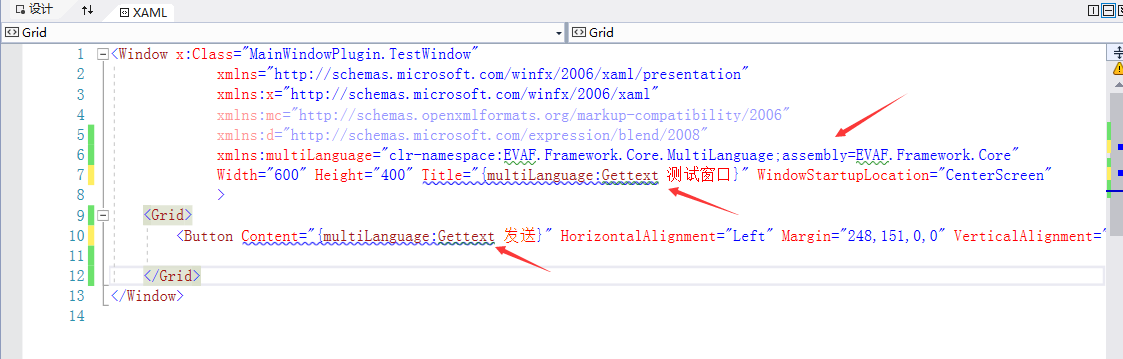


增加引用：

xmlns:multiLanguage="clr-namespace:EVAF.Framework.Core.MultiLanguage;assembly=EVAF.Framework.Core"

修改Title为Title="{multiLanguage:Gettext 测试窗口}"，修改Button的Content为Content="{multiLanguage:Gettext 发送}"。

其中：multiLanguage:Gettext为翻译标识，其后内容为需要翻译的内容。



注：multiLanguage:Gettext中multiLanguage为引用的缩写，在本例子中为multiLanguage。该缩写智能修改为LNG和multiLanguage，为方便后面的工具使用。

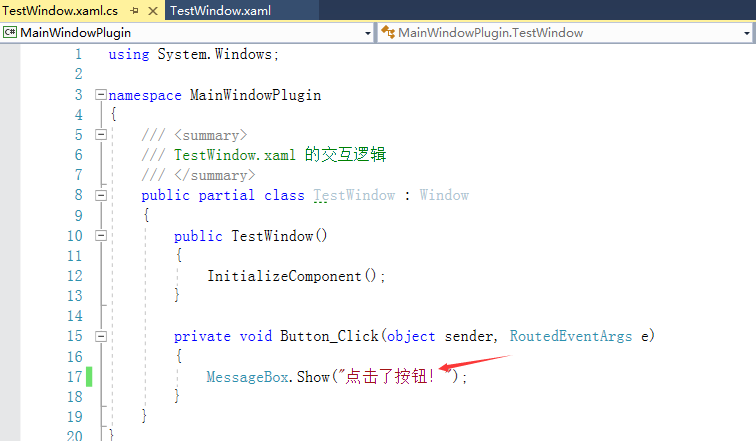
如果需要修改为LNG，则需将引用修改为：

xmlns:LNG="clr-namespace:EVAF.Framework.Core.MultiLanguage;assembly=EVAF.Framework.Core"

### CS类中代码翻译写法

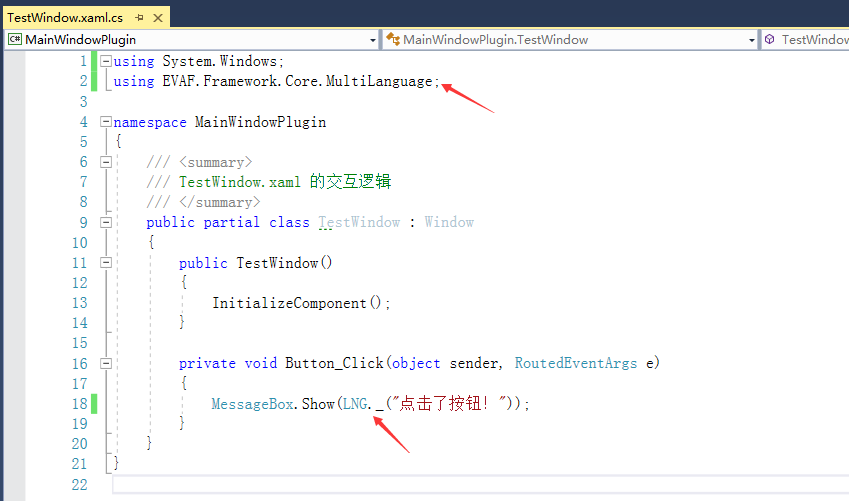
1. 确定翻译位置；

下图中需要对弹出消息进行翻译。



1. 修改CS类中代码。

需引用EVAF.Framework.Core.MultiLanguage，同时，将MessageBox中需要弹出的代码加上LNG.\_(" ")即可。

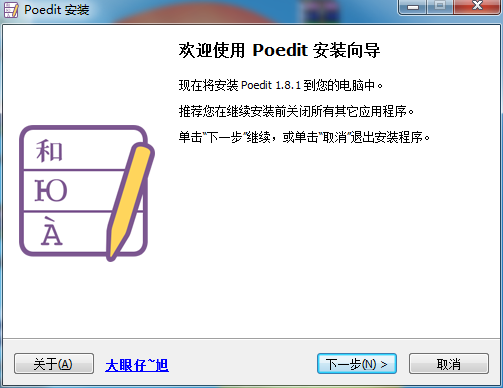


## 多语言翻译

### 安装POEdit翻译工具

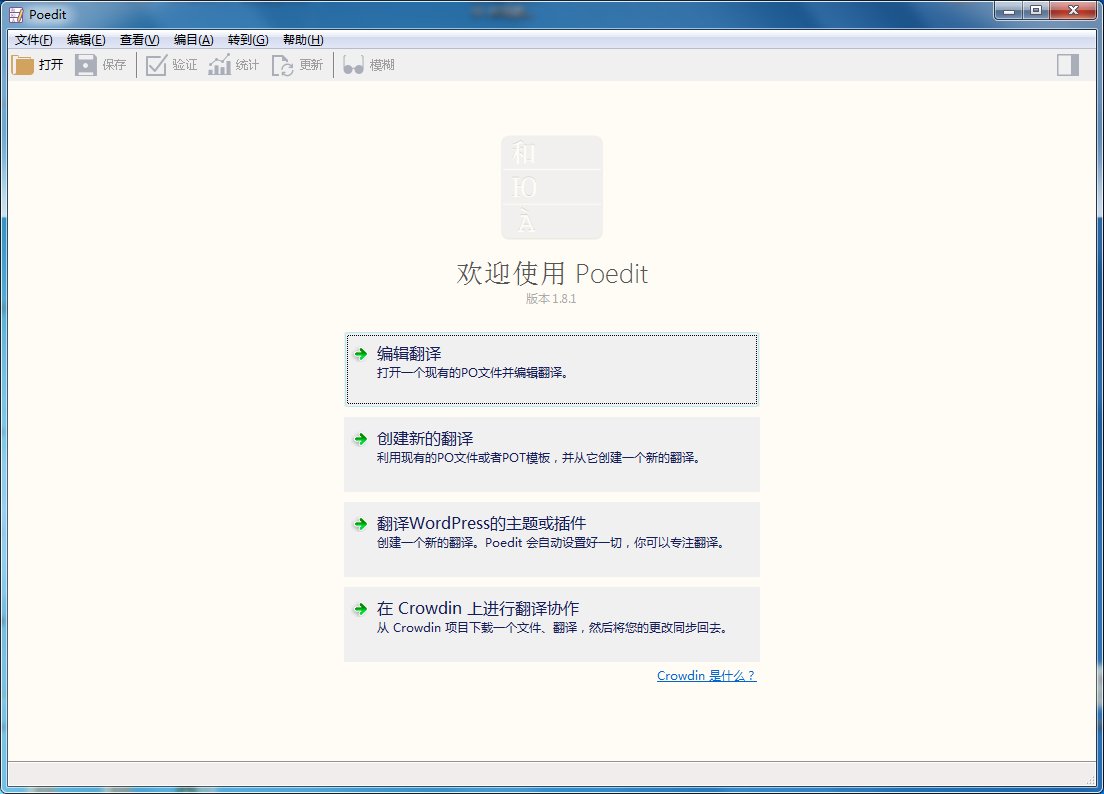


安装十分简单，根据提示，点击下一步即可。



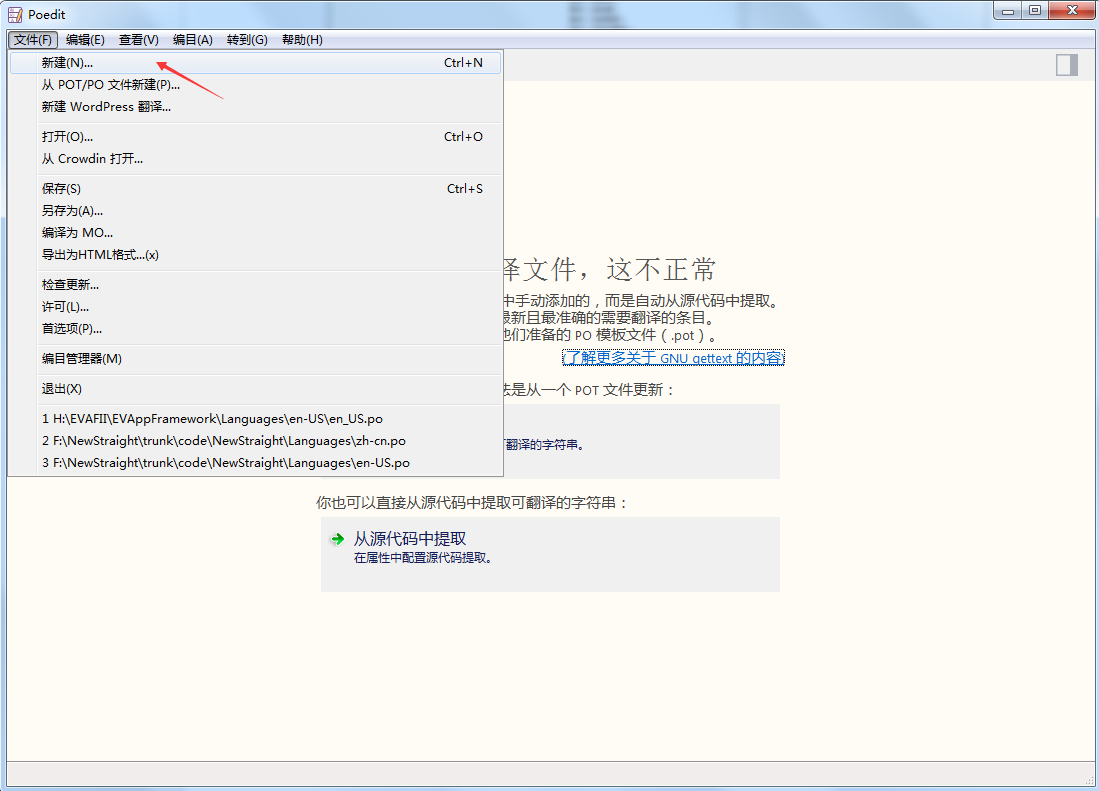
### 新建PO文件

1. 打开POEdit程序；

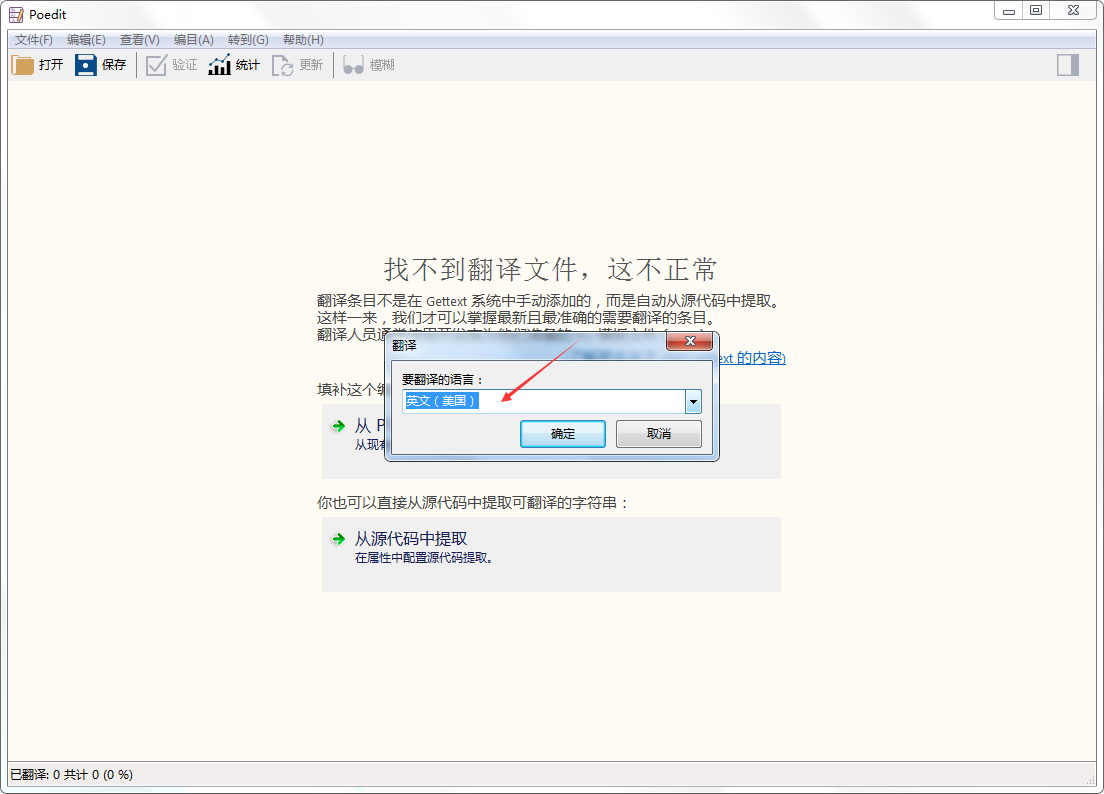


1. 新建PO文件；

点击文件->新建。

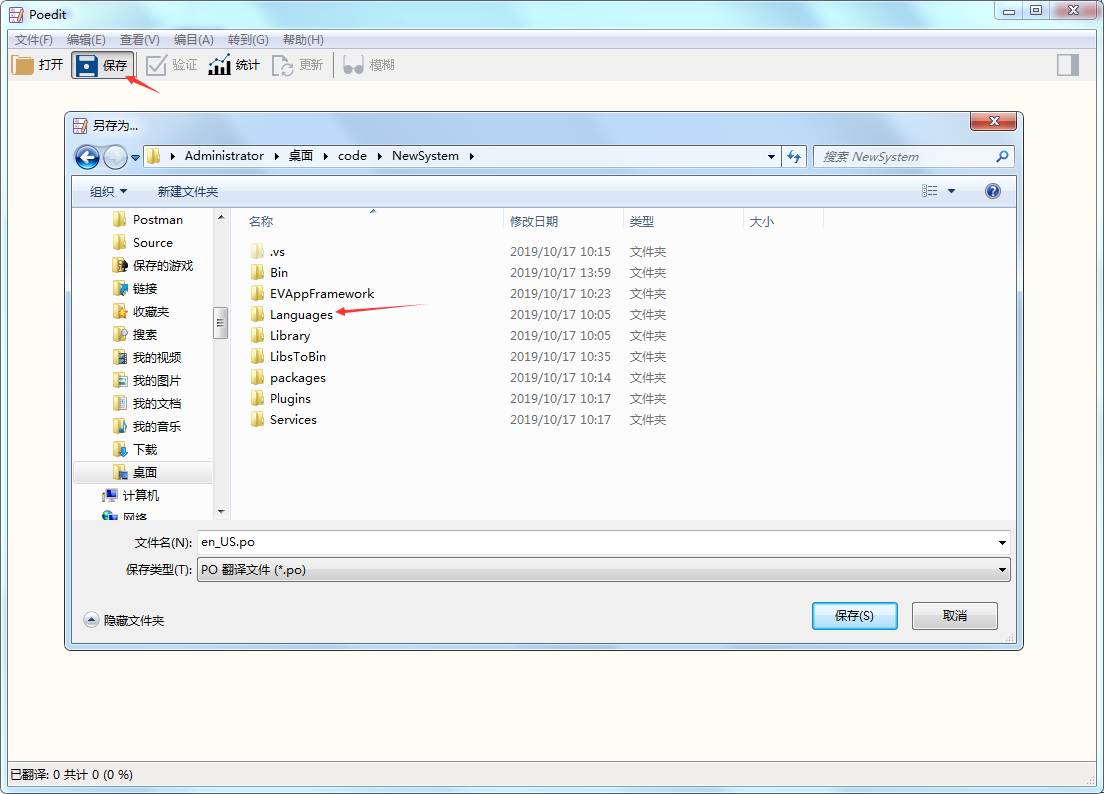


选择英语（美国）。

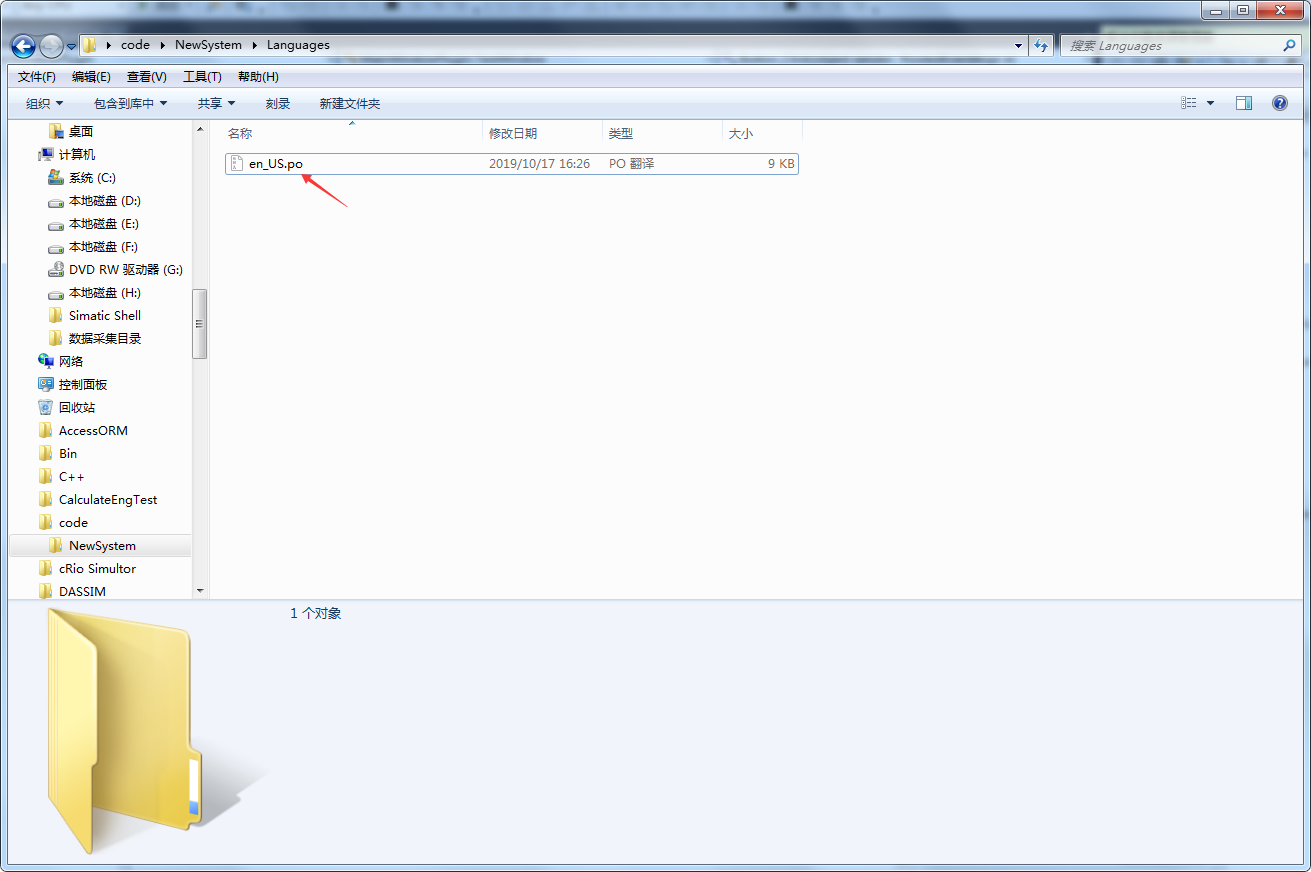


1. 点击保存，生成PO文件；

点击保存，可将PO文件生成在工程目录的Languages目录下（主要方便管理），也可放在其他自定义位置。



点击保存后，在Languages文件夹中包含一个en\_US.po文件（可双击打开）。

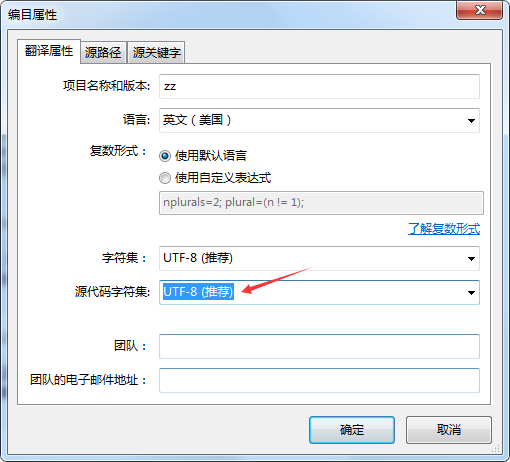


1. 点击编目->属性；



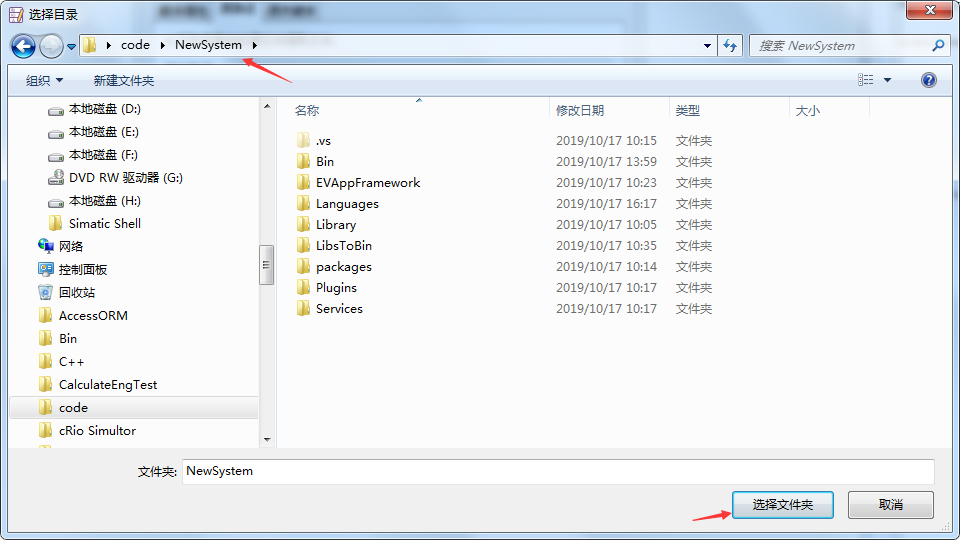
1. 翻译属性设置；

翻译属性用于设置当前源代码的字符集，设置源代码字符集为UTF-8。



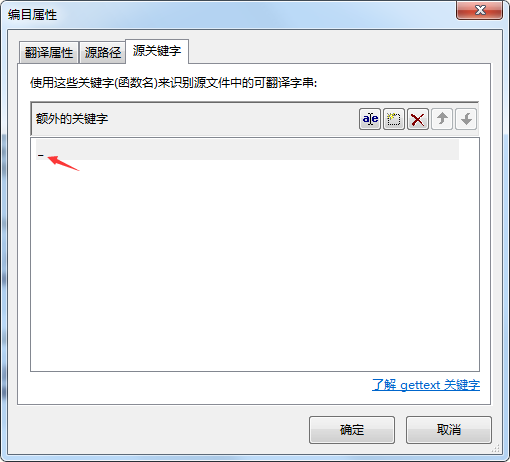
1. 源路径设置；

源路径用于设置源代码工程目录，便于POEdit查找需要翻译的软件。

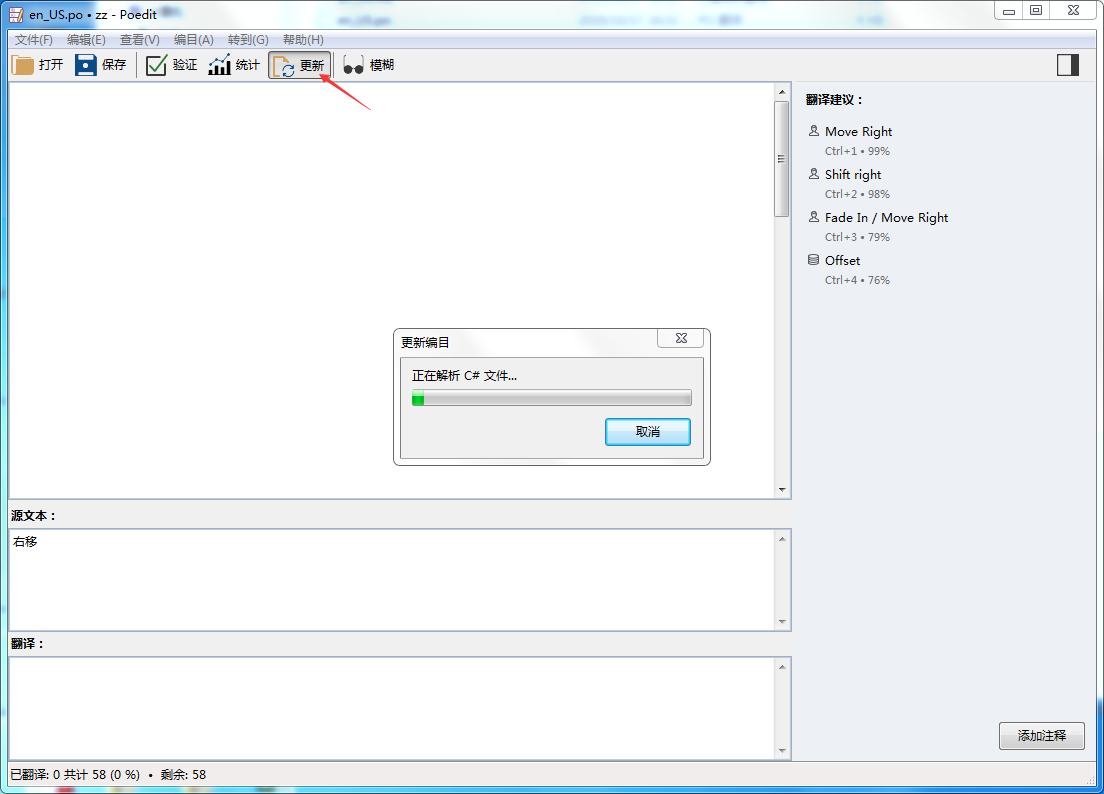


1. 源关键字设置；

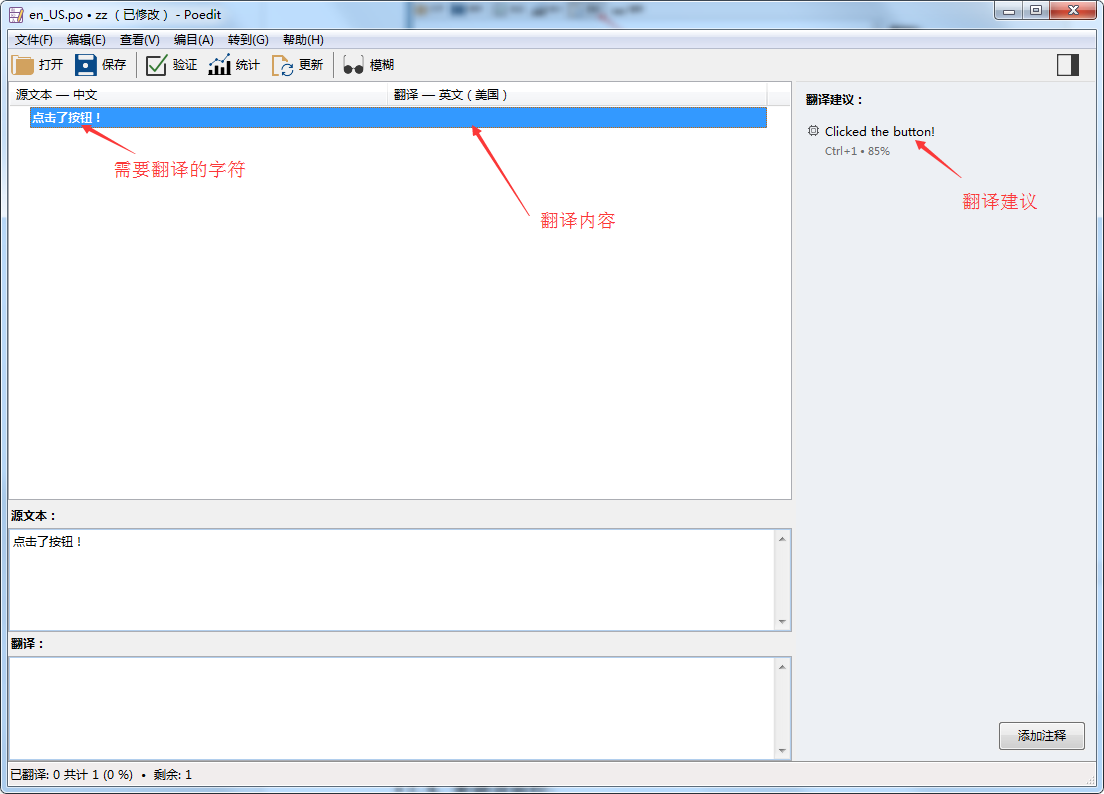
源关键字用于匹配需要翻译的字符串，我们采用下划线、括号与双引号的方式\_(“”)，因此输入额外关键字为下划线“\_”。



1. 确定后，点击更新；



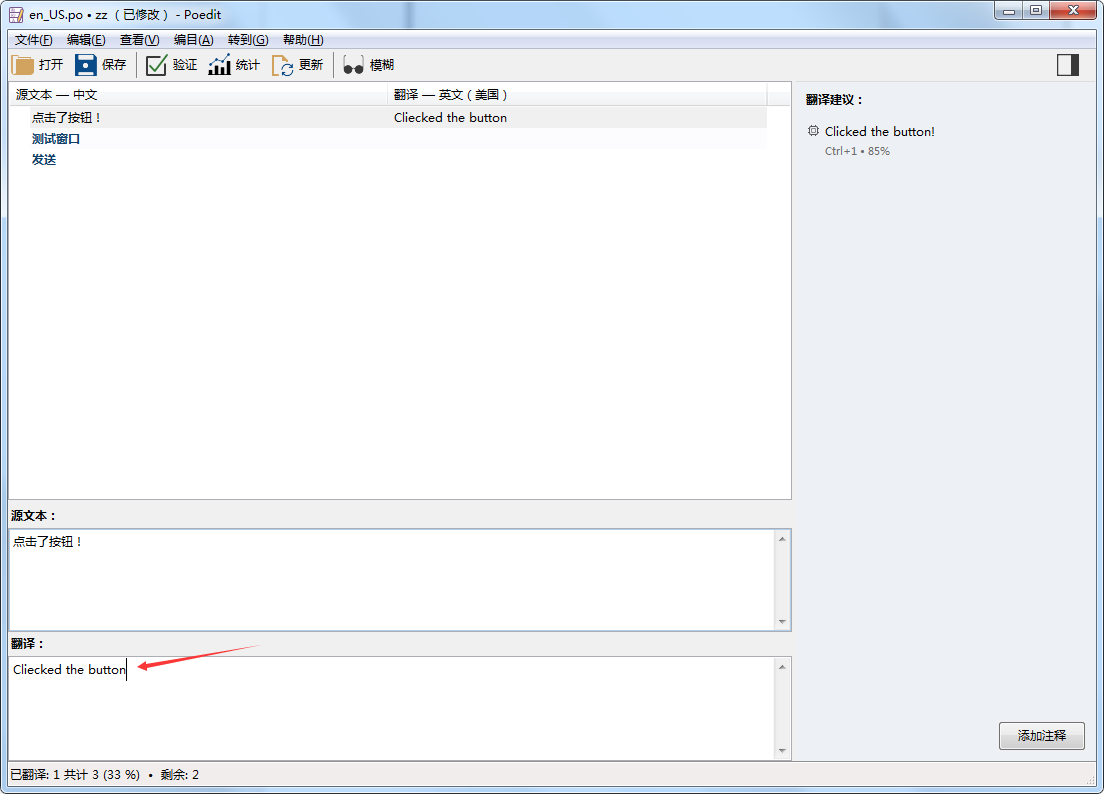
结果如下：



### 多语言翻译

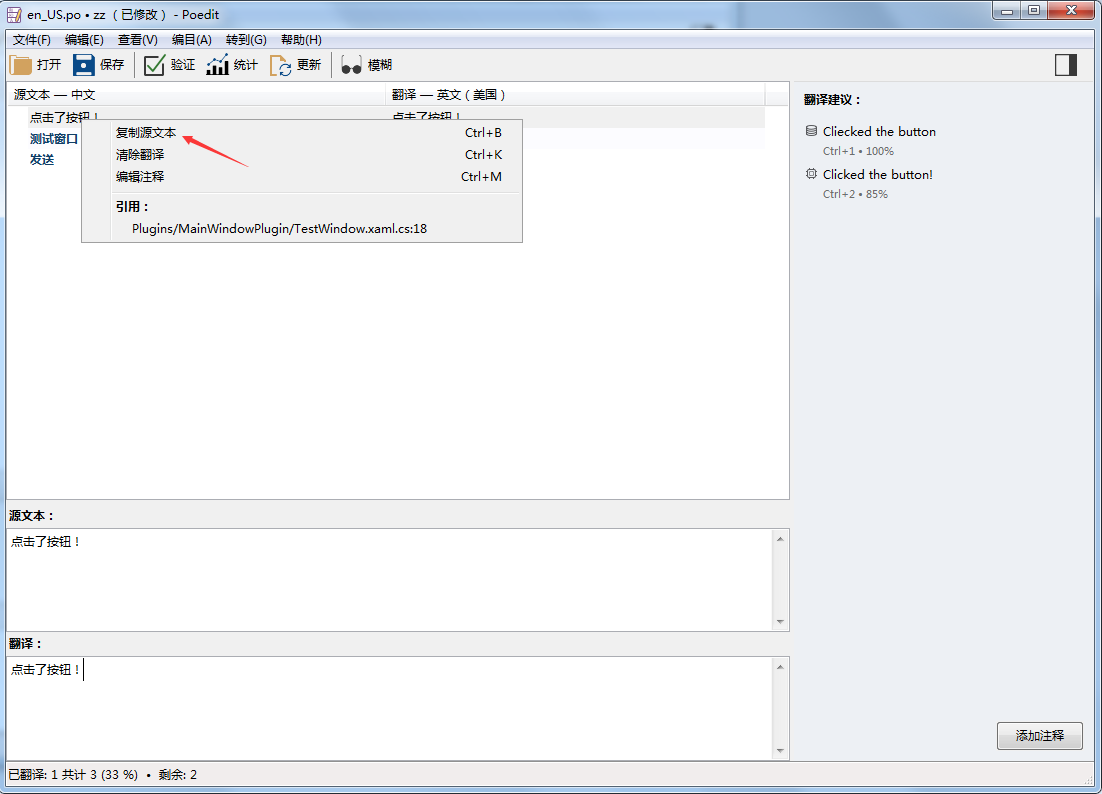
1. 单个翻译

选中需要翻译的项，在翻译列表中直接输入字符进行翻译，同时上方显示译文信息。

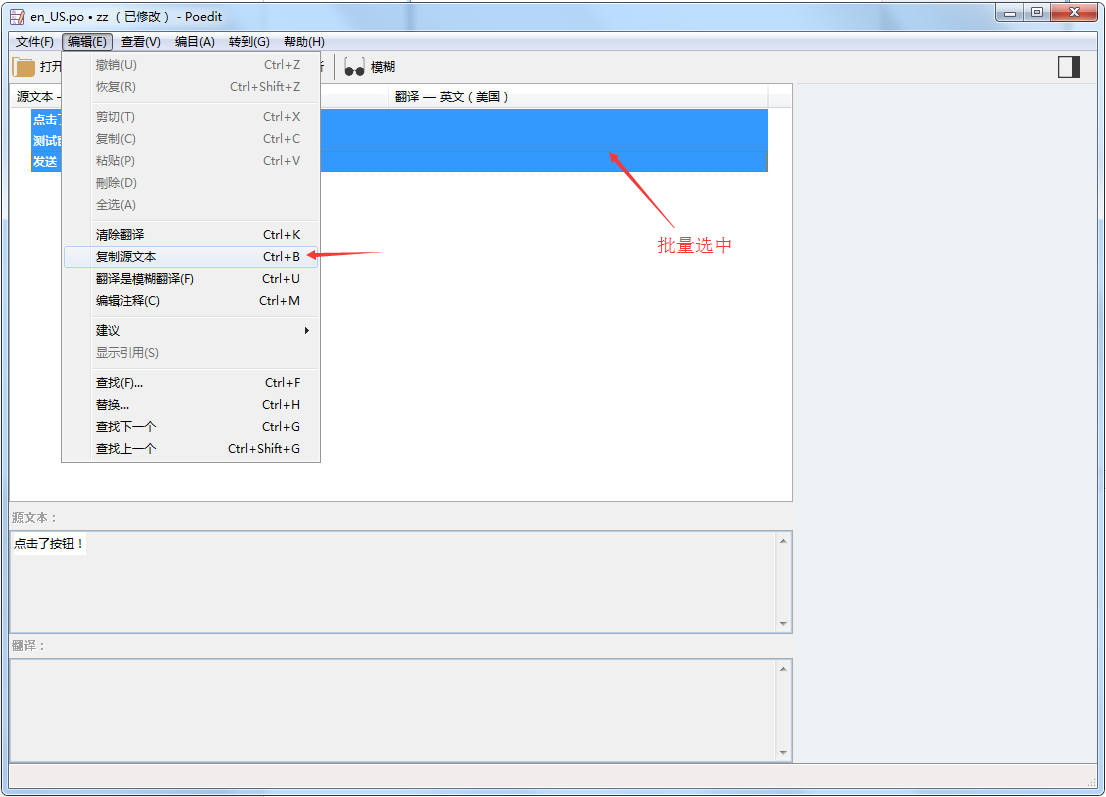


1. 复制原文

对于原文和译文相同的情况，可右键点击“复制源文本”。

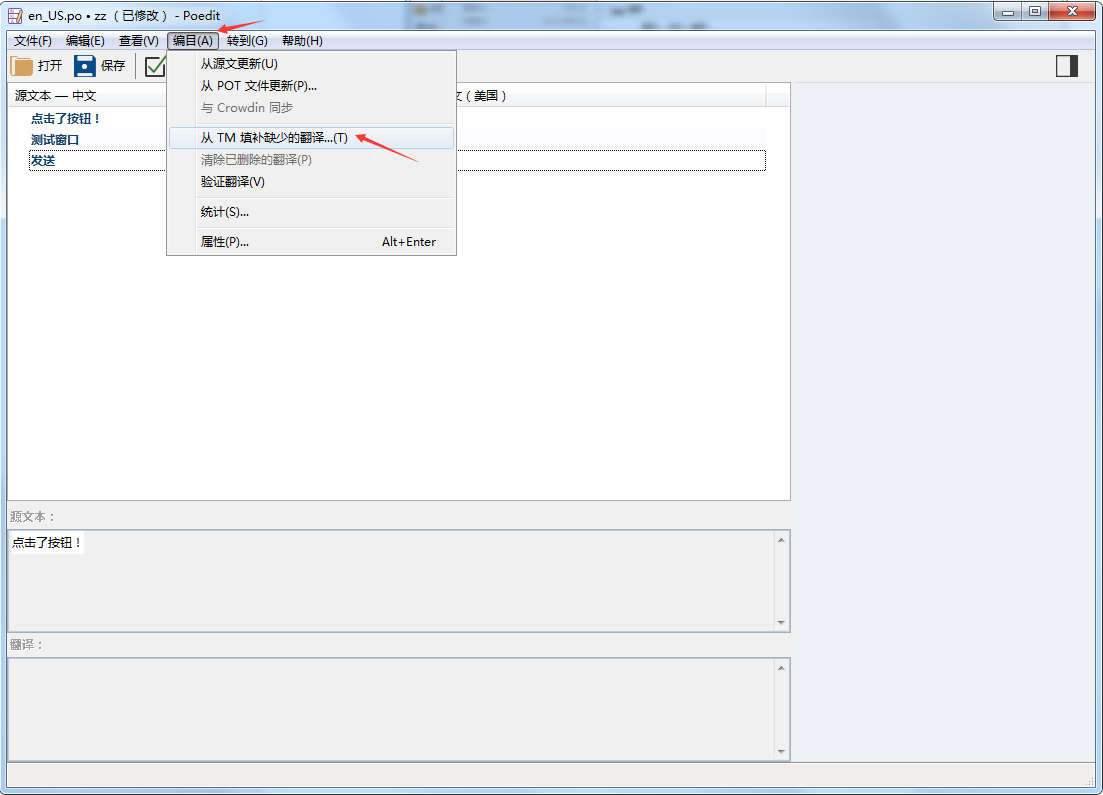


下图为批量复制：

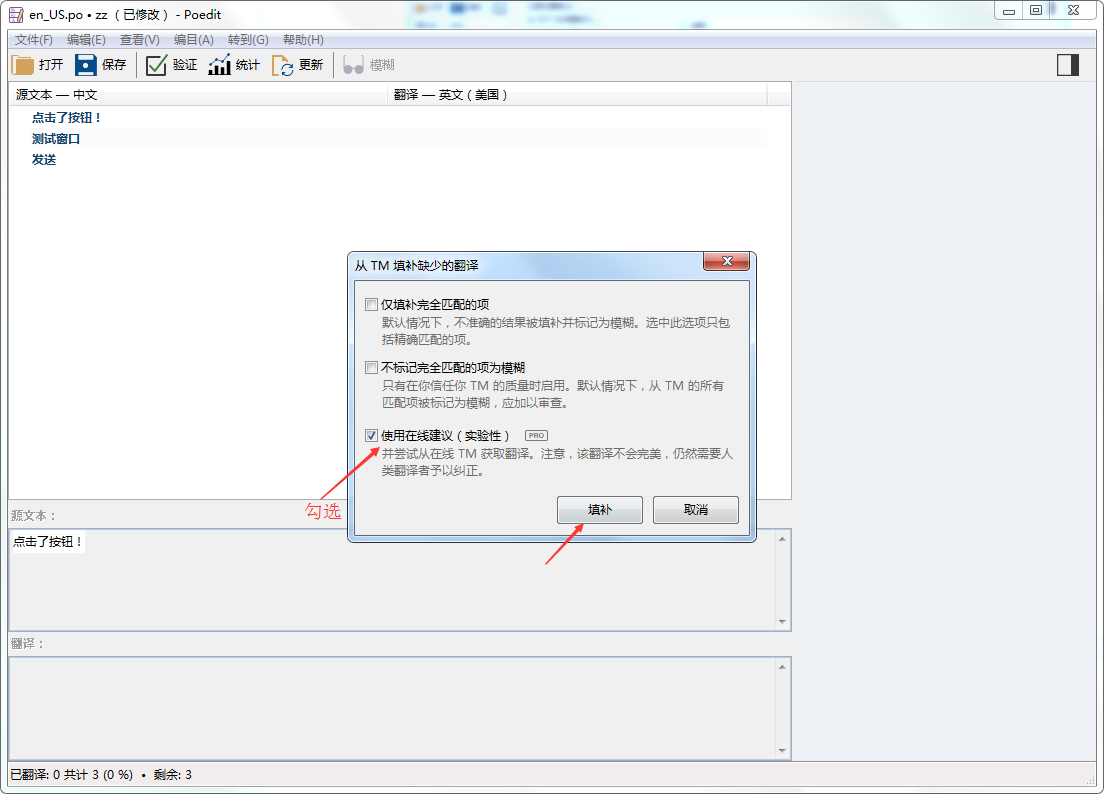


1. 使用在线翻译（该功能为收费版）

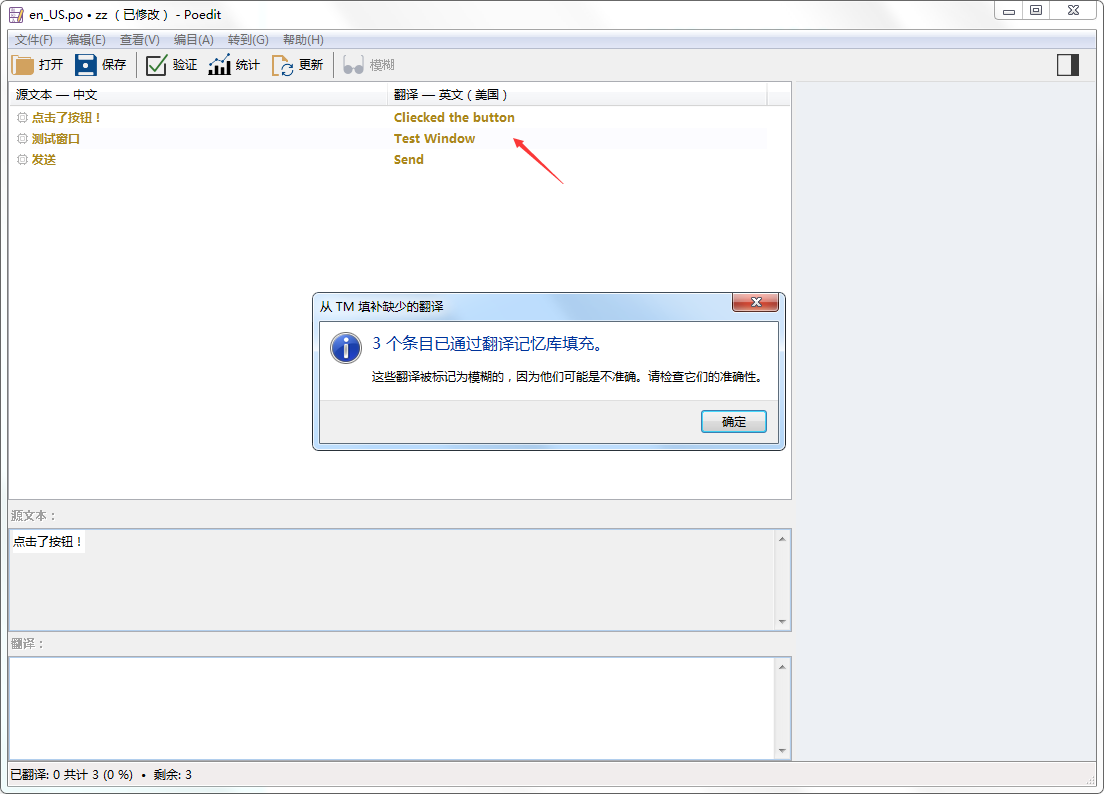
点击编目->从TM填补缺少的翻译



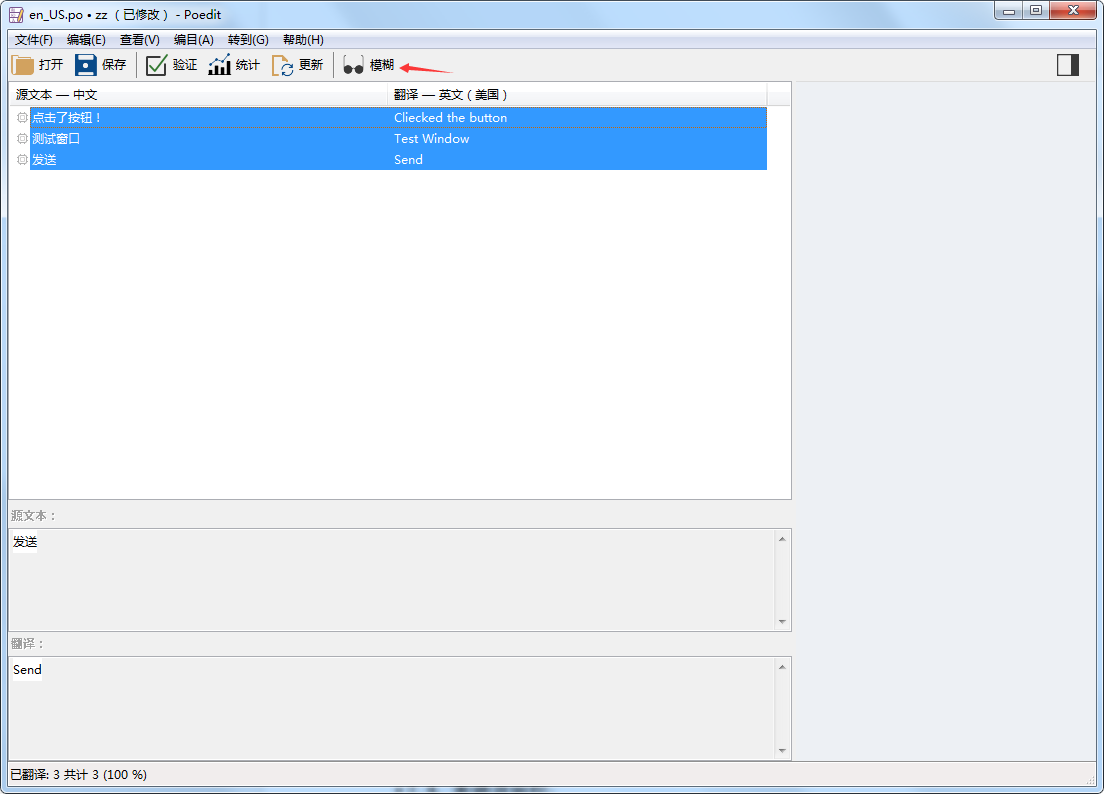
然后勾选“使用在线建议”，点击填补



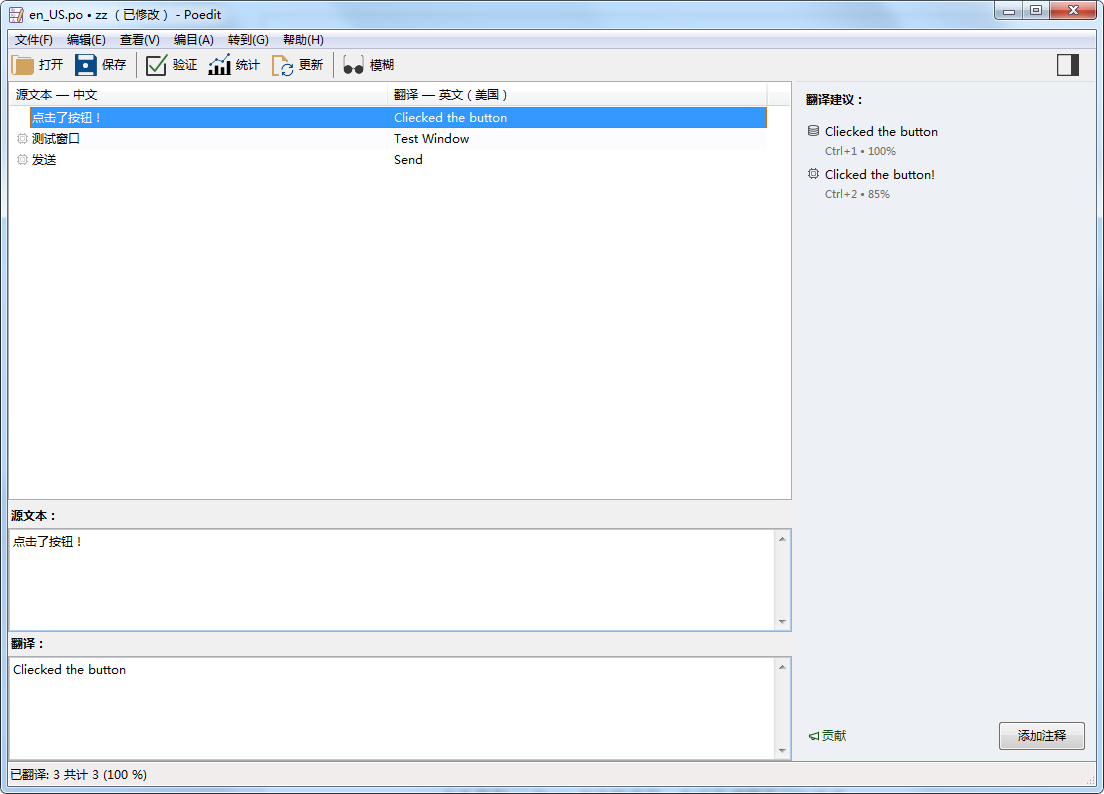
结果如下：



上图中黄色标识表示需要确认的，否则为未翻译，可全选，再点击工具栏“模糊”按钮；

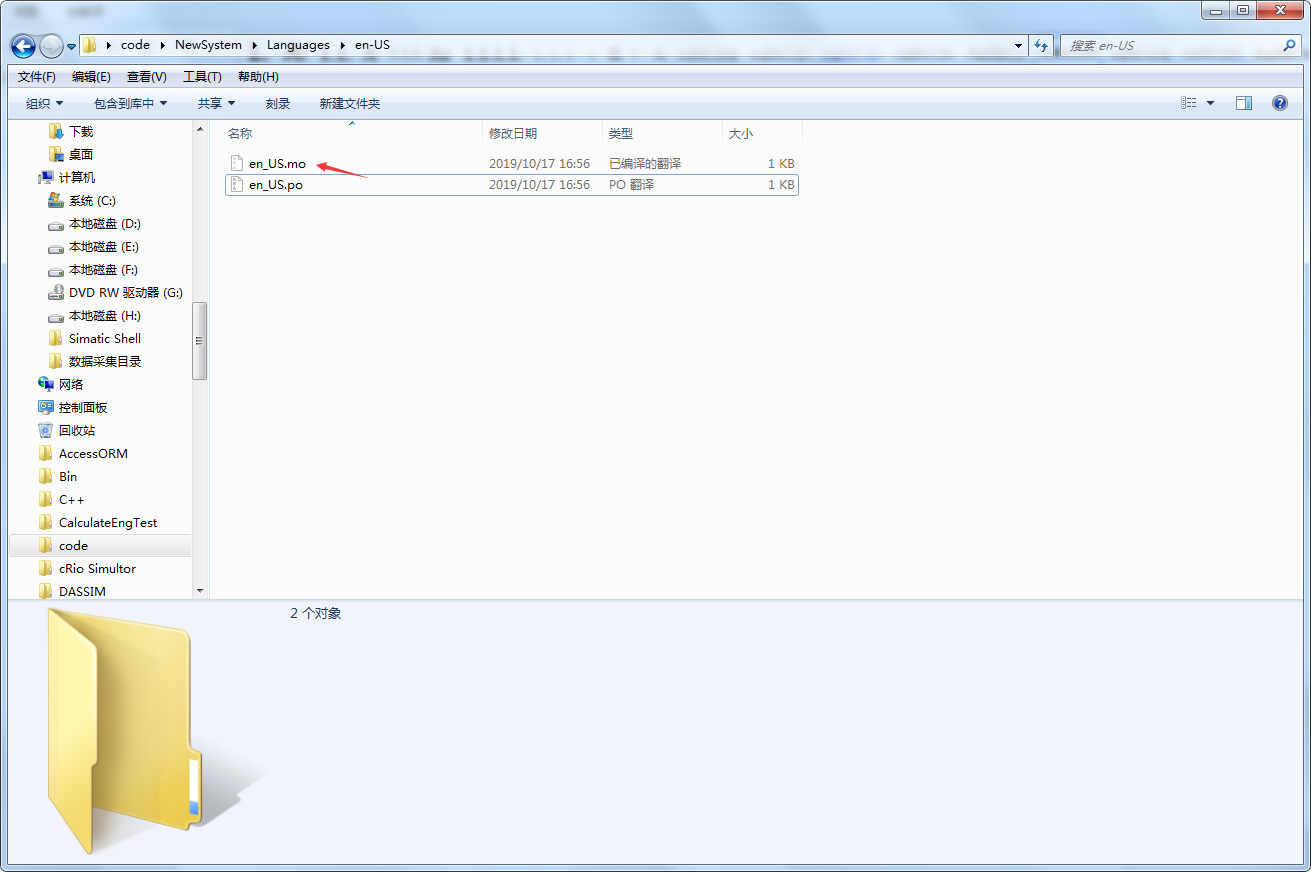


翻译结果如下：（如需要进行修改，可修改翻译内容）



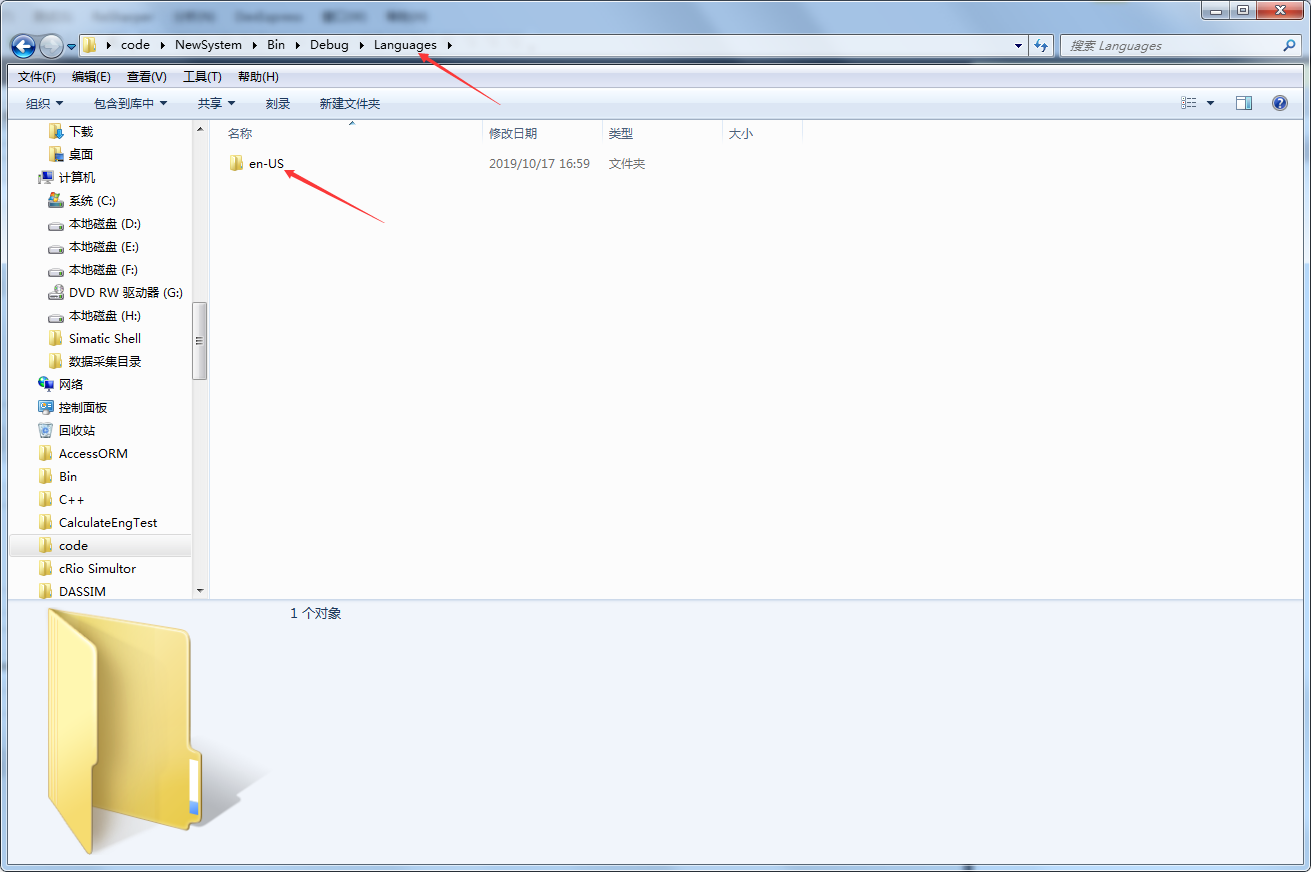
### 生成MO文件

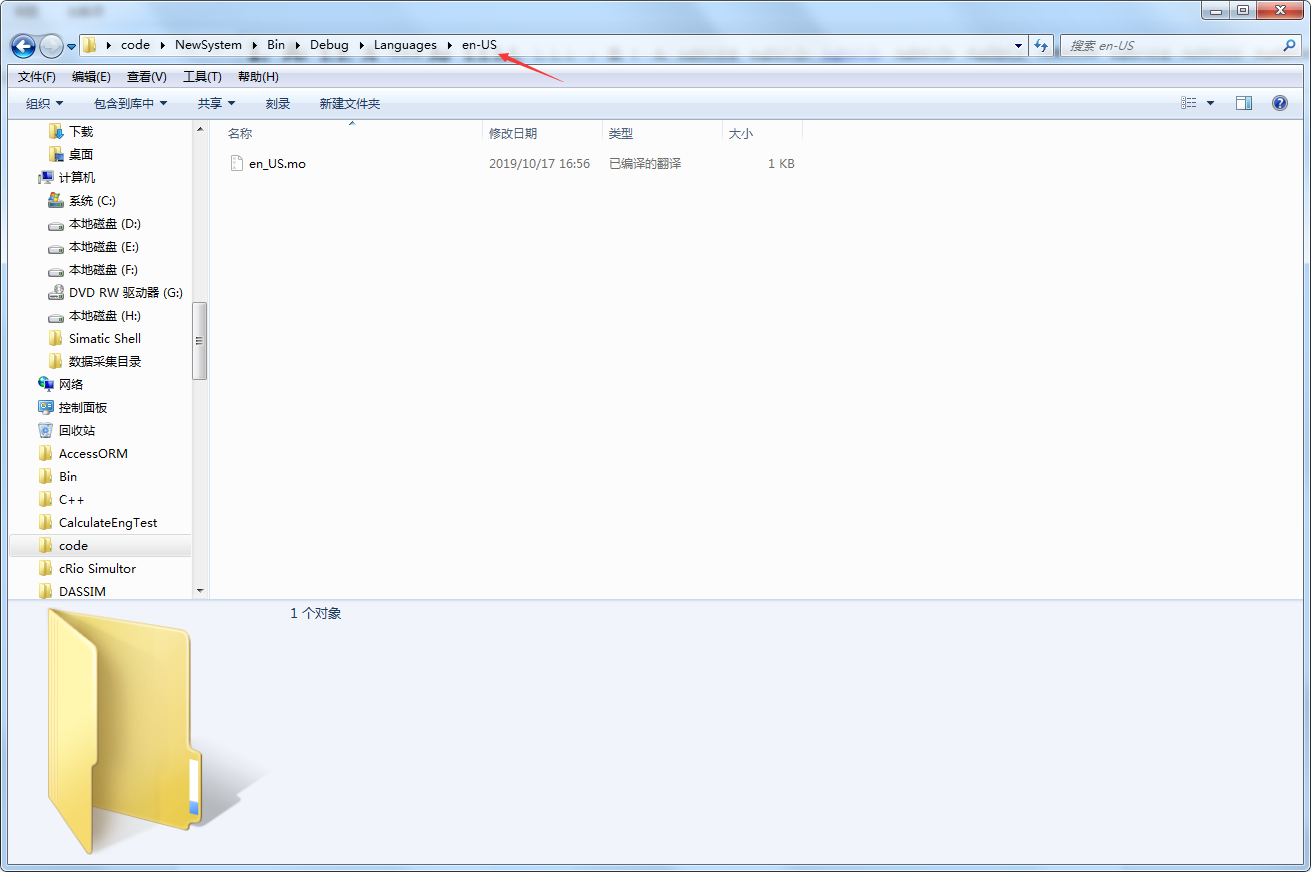
点击保存， 在PO文件路径下，自动生成同名MO文件。



## 多语言运行

1. 在Debug目录下创建一个Languages文件夹，并在Languages文件夹中创建一个“en-US”文件夹，将生成的MO文件拷贝到该文件夹中；（en-US中可以有多个MO文件，支持框架、模块为独立的MO文件）



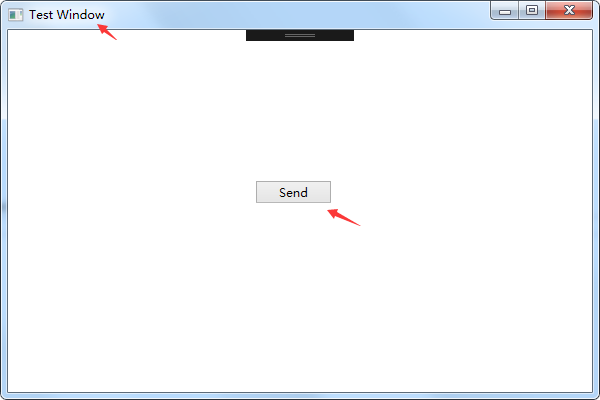


1. 在应用程序启动时设置en-us；

（如需切换显示，可再创建一个zh-cn的文件夹，用于存放中文翻译，中文翻译可直接复制源文）



1. 运行、查看效果。



点击send按钮：

