翻译工具说明：

翻译档案格式说明：

档案中按照section来区分不同的模块的handle，我发现有如下的特点：

1)handle出现在相同的section中两次及以上会使用第一次出现的作为翻译结果。

<section name=”ASIA”>

<handle =”XXX”>

<visual 1…

<handle =”XXX”>

<visual 2…

</section>

以上情况，XXX的翻译就是1…

2)section 可以重复，重复的section并不会影响到翻译档的使用，同名的handle翻译依旧如规则1)的方式获取翻译。

3)同一个handle出现在不同的section，这时两个handle之间没有相互关联。

基于以上，程序中在判定两个handle是否等价时，总是要判定是否是位于同一个section下。

使用方法：

1. 将Merge\_Tool.pl，find\_lost\_string\_handles.pl放于$DIR(表示任意一个目录)下。
2. 在$DIR下新建filesToUpdate目录，将要翻译的档案(例如2xx5从美国那边拿过来的原始翻译档案)全部放置于该目录下；然后再在$DIR目录下新建Ref目录，该目录用于放置翻译参考文档(例如从2565中拿出来的翻译档案)，将asia.str置于$DIR目录下，运行Merge\_Tool.pl档案即可更新原始翻译档案(追加规则见下文)。
3. find\_lost\_string\_handle.pl是由mint写的初始版修改而来，用于检查update之后的翻译档案是否完全包含了asia.str中的handle(是一个辅助检查的工具)，如果检查到有没有翻译的handle，会在$DIR目录下生成对应的log文件，其中记录未被翻译的handle。  
   Merge\_Tool.pl 的运行逻辑：
4. 首先会读入asia.str中的handle和它的string。
5. 尝试到参考翻译档案中去查找对应handle的翻译，如果找到了就用参考档案中的翻译结果来翻译该handle，如果没有，则使用特殊字串表示其翻译结果，以便查看。
6. 更新原始翻译档，如果一个handle在原始档和参考档中都有翻译，则使用原始档案中的翻译作为其结果，如果该handle对于原始档案来说是一个新的handle，那么就将其追加在原始翻译档案的结尾，追加的顺序：
7. 放入有参考翻译的handle
8. 没有参考翻译的handle

内部的排序使用handle的字典序排列

------以上过程会使用filesToUpdate作为标准，也就是说如果一个handle在两个翻译档案中均有翻译，则会使用fileToUpdate中的为准。

用于标记没有翻译的标记：\_NOT\_Trans\_YET\_，其后会紧跟英文原文

例如：

<entry handle="2D\_LANDMARK">

<strings>

<string default="true">

<visual>\_NOT\_Trans\_YET\_:2D Landmark</visual>

</string>

</strings>

<usage visual="true"/>

</entry>

翻译冲突：

CompareTwoTranslateFile.pl为了解决两个翻译档中不同翻译结果：执行起来非常慢，原因是文件IO太多造成的，解决办法是可以把翻译档按照一定格式存入到内存中，然后再做比较，但调试起来麻烦。

在同样目录下，执行CompareTwoTranslateFile.pl会得到compare.log

log的格式如下

Compare XXX and YYY

<section\_Name><Handle\_name><translation\_A><translation\_B>

….

….

Compare AAA and BBB

意思是一个handle在XXX档中的翻译时translation\_A，在YYY中的翻译是translation\_B，

这样一来就可以方便的比较到底使用哪个翻译档案中的翻译

需要注意的是：一个handle可以有很多种不同的翻译，程序没有办法处理这种情况，也就是说如果一个handle的翻译实际是一样的，只是翻译出现的顺序不一样，程序依旧会认为同一个handle有两个不同的翻译。这个可以在log文件中看到