

# 基于 java web service 的英汉互译系统

## 运行结果

```
Run: Main x
"D:\Program Files\Java\jdk1.8.0_212\bin\java.exe" ...
请输入待查询的单词: 你好
-----基本解释-----
你好
nǐ hǎo
hello; how are you

-----近义词汇-----
ye
chín-chín
How do you do.
Hi, there.
nihao
-----参考例句-----
你好哇, 哥儿们! |How are you doing, sport!
`你好吗?' `很好, 谢谢!' |`How are you?' `Fine, thanks.'
你好, 史蒂文斯。|Hi there, Stevens.
```

```
请输入待查询的单词: hello
-----基本解释-----
hello
'heləu, he'ləu

int. (见面打招呼或打电话用语) 喂, 哈罗
1059.mp3
-----近义词汇-----
- Empty -
-----参考例句-----
She actually condescended to say hello to me in the street today. |她今天在街上竟能屈尊跟我打招呼。
I said hello to her, but she ignored me completely! |我向她打招呼, 可她根本不理我!
Hello there, what a coincidence! |你好, 真巧啊!
```

系统基于第三方 Webservice 服务, 实现了翻译、音标 (拼音)、解释、相关词条、例句、候选词等功能

## 设计过程

- 生成 Web 服务代理类 (WSDL to Java)  
安装 JDK 8, 配置环境变量

```
{107,23} size, {121,1000} buffer
D:\Program Files\Java\jdk1.8.0_212\bin
λ wsimport.exe
缺少 WSDL_URI

用法: wsimport [options] <WSDL_URI>

其中 [options] 包括:
-b <path>          指定 jaxws/jaxb 绑定文件或附加模式
                    (每个 <path> 都必须具有自己的 -b)
-B<jaxbOption>      将此选项传递给 JAXB 模式编译器
-catalog <file>     指定用于解析外部实体引用的目录文件
                    支持 TR9401, XCatalog 和 OASIS XML 目录格式。
-d <directory>      指定放置生成的输出文件的位置
-encoding <encoding> 指定源文件所使用的字符编码
-extension          允许供应商扩展 - 不按规范
                    指定功能。使用扩展可能会导致应用程序不可移植或
                    无法与其他实现进行互操作
-help              显示帮助
-httpproxy:<host>:<port> 指定 HTTP 代理服务器 (端口默认为 8080)
-keep              保留生成的文件
```

1. 从在线的 webservice 服务中找一个感兴趣的接口: [http://www.webxml.com.cn/zh\\_cn/web\\_services.aspx?offset=1](http://www.webxml.com.cn/zh_cn/web_services.aspx?offset=1)
2. 在这里, 我选择使用第一个中英文互译接口, 使用 wsimport 工具, 生成 Web 服务代理类

```
1 wsimport -keep -p wsproxy -d E:\workspace\javafile\JavaWebService\src
  http://fy.webxml.com.cn/webservices/EnglishChinese.asmx?wsdl
```

3. 命令执行时会报错, 我们需要首先把 xml 进行修改

用浏览器下载 xml 文件, 然后用文本编辑器打开,  
用

```
1 <s:any minOccurs="2" maxOccurs="2"/>
```

替代

```
1 <s:element ref="s:schema" />
2 <s:any />
```

```
12 </s:element>
13 <s:element name="TranslatorResponse">
14   <s:complexType>
15     <s:sequence>
16       <s:element minOccurs="0" maxOccurs="1" name="TranslatorResult">
17         <s:complexType>
18           <s:sequence>
19             <s:any minOccurs="2" maxOccurs="2"/>
20           </s:sequence>
21         </s:complexType>
22       </s:element>
23     </s:sequence>
24   </s:complexType>
25 </s:element>
```

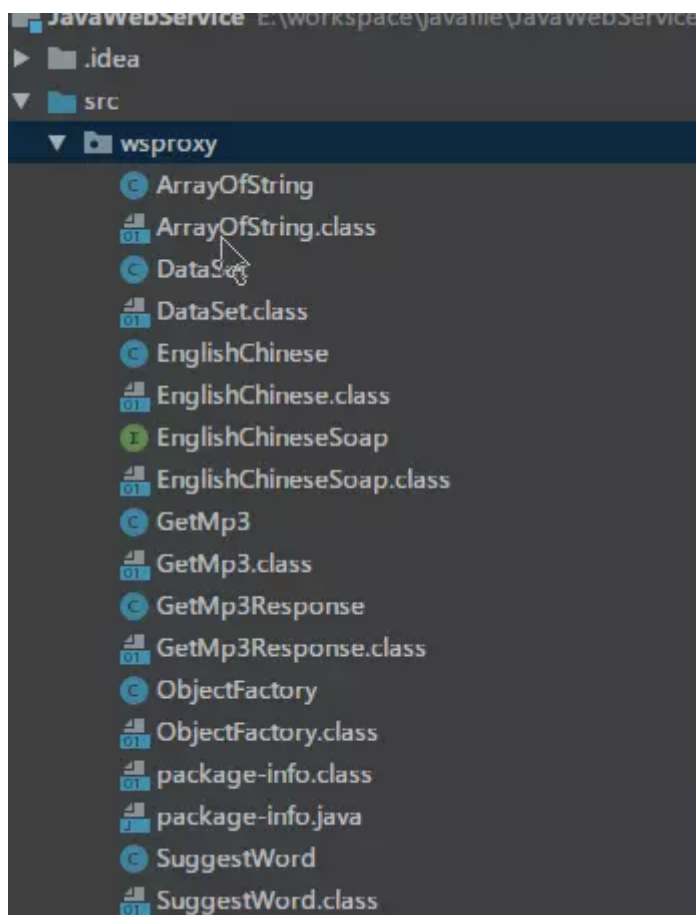
#### 4. 用本地文件来替换 xml 网络地址:

```
1 | wsimport -keep -p wsproxy -d E:\workspace\javafile\JavaWebService\src  
   | E:\workspace\javafile\JavaWebService\EnglishChinese.xml
```

编译成功:

```
D:\Program Files\Java\jdk1.8.0_212\bin  
A wsimport -keep -p wsproxy -d E:\workspace\javafile\JavaWebService\src E:\workspace\javafile\JavaWebService\EnglishChinese.xml  
正在解析 WSDL...  
  
[WARNING] 忽略 SOAP 端口 "EnglishChineseSoap12": 它使用非标准 SOAP 1.2 绑定。  
必须指定 "-extension" 选项以使用此绑定。  
file:/E:/workspace/javafile/JavaWebService/EnglishChinese.xml 的第 549 行  
  
[WARNING] 忽略端口 "EnglishChineseHttpGet": 未指定 SOAP 地址。请尝试运行带 -extension 开关的 wsimport。  
file:/E:/workspace/javafile/JavaWebService/EnglishChinese.xml 的第 552 行  
  
[WARNING] 忽略端口 "EnglishChineseHttpPost": 未指定 SOAP 地址。请尝试运行带 -extension 开关的 wsimport。  
file:/E:/workspace/javafile/JavaWebService/EnglishChinese.xml 的第 555 行  
  
正在生成代码...  
  
正在编译代码...
```

可以看到已经为我们生成了对应的代码



#### • 编写客户端代码

编译后为我们产生了 EnglishChinese 类, 调用该类, 便可以直接使用 WebService 提供的服务  
具体的实现见源代码