武汉纺织大学

Web应用开发课程设计

**在线翻译**

**学 院： 数学与计算机学院**

**班 级： 物联网11802**

**姓 名： 徐珺鸿**

**学 号： 1804240930**

**指导老师： 聂刚**

**成 绩：**

**完成日期： 2020年12月17日**

目 录

[1 需求分析 1](#_Toc28054)

[1.1前端页面 1](#_Toc901)

[1.2主要功能 1](#_Toc20276)

[1.3背景分析 1](#_Toc28583)

[2 系统设计 1](#_Toc16378)

[2.1用户用例图 1](#_Toc17772)

[2.2 UML时序图（Sequence Diagram） 2](#_Toc28549)

[2.3 UML活动图（Activity Diagram） 3](#_Toc10467)

[2.3.1 前端布局 3](#_Toc24829)

[2.3.2 ajax请求 4](#_Toc10881)

[2.3.3servlet参数获取 4](#_Toc23855)

[2.3.4 数据处理 5](#_Toc10595)

[2.3.5 响应数据 5](#_Toc8767)

[3 系统实现 6](#_Toc16558)

[3.1 项目结构 6](#_Toc9737)

[3.2 配置文件 6](#_Toc24821)

[3.2.1 Tomcat V8.5 6](#_Toc10346)

[3.2.2 JRE System Library 7](#_Toc32206)

[3.2.3 gson-2.8.6.jar 8](#_Toc32015)

[3.3 前端布局文件translate.html 8](#_Toc14580)

[3.4前端响应文件function.js 9](#_Toc28794)

[3.5请求文件HttpGet.java 9](#_Toc32692)

[3.6 API设置文件ApiKey.java 13](#_Toc29805)

[3.7 API处理文件TransAPI.java 14](#_Toc16220)

[3.8 加密文件MD5.java 15](#_Toc30535)

[3.9 编码设置文件UnicodeToZh.java 18](#_Toc23423)

[3.10控制器文件TranslateServlet.java 18](#_Toc12219)

[4 系统测试 20](#_Toc15818)

[4.1 空页面 20](#_Toc2822)

[4.2 自动检测 20](#_Toc13832)

[4.3 中-英 21](#_Toc21015)

[4.4 中-法 21](#_Toc26394)

[4.5 中-德 21](#_Toc27440)

[4.6 英-文 21](#_Toc22910)

[4.7 系统测试总结 21](#_Toc4887)

[5 系统总结 21](#_Toc15413)

[5.1参考资料 21](#_Toc21710)

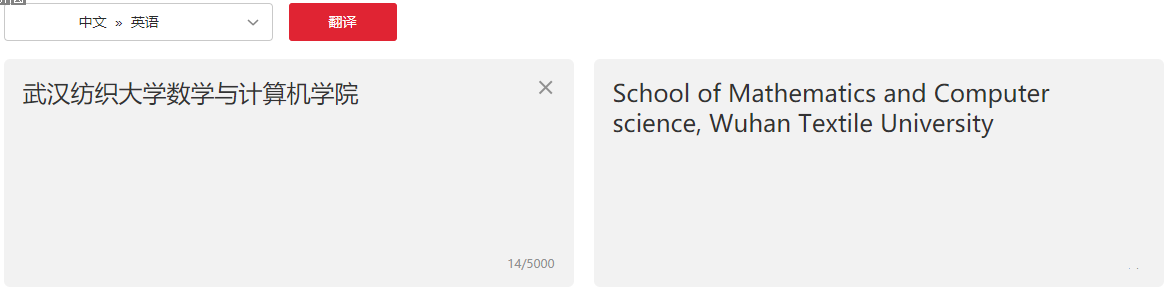
[5.2系统总结 21](#_Toc19994)

[5.3课程总结 22](#_Toc23584)

[5.4自我评价 22](#_Toc15038)

[5.5展望未来 22](#_Toc24965)

# 1 需求分析



前端页面选择语言，输入文字，发送给服务器端，服务器端调用翻译API接口，将翻译结果传递至前端页面进行显示。

## 1.1前端页面

前端页面需要显示：

1. 语言选择下拉框
2. 翻译按钮
3. 左输入区
4. 右翻译区

## 1.2主要功能

前端页面选择语言，输入文字，发送给服务器端，服务器端响应请求调用翻译API接口，将翻译结果传递至前端页面进行显示。

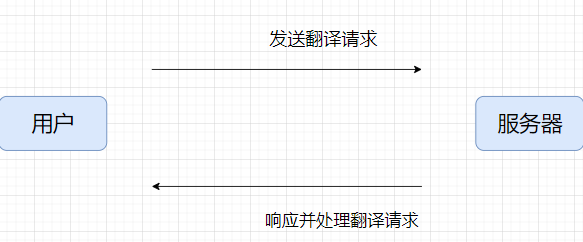
## 1.3背景分析

通用翻译API是百度翻译依托领先的自然语言处理技术推出的在线文本翻译服务，可支持中、英、日、韩等200+语言互译，100+语种自动检测。  
通用翻译API旨在为用户提供低廉、方便的机器翻译服务解决方案。  
您只需调用通用翻译API，传入待翻译内容，并指定要翻译的源语言（支持语种自动检测）和目标语言，即可获得相应的结果。通用翻译API可广泛应用于APP、网站及客户端，提供实时优质的多语言翻译服务。

# **2 系统设计**

## 2.1用户用例图

系统分为用户，服务器两个角色，各角色的用户用例如下图所示：



## 2.2 UML时序图（Sequence Diagram）

我的项目结构包括前端布局，ajax异步请求，参数获取，数据处理，数据响应输出5个部分。

具体内容工作如下：

1. **前端布局**

控制基本页面显示，包括输入样式与输出样式

1. **ajax异步请求**

onclick事件触发后，发送请求，作为前端与服务器的交互

1. **servlet参数获取**

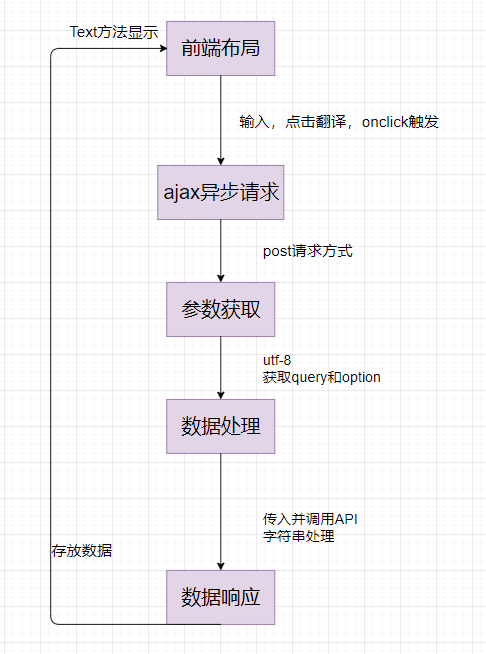
设置字符集，获取前端的输入与选择

1. **数据处理**

利用API开始对字符串进行翻译

1. **数据响应**

处理后把数据获取并显示到前端

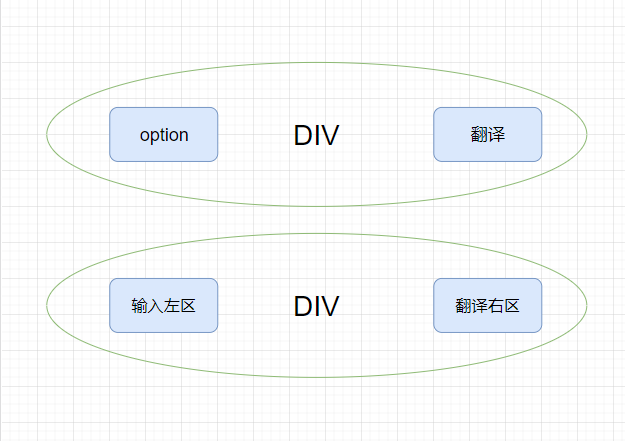


## 2.3 UML活动图（Activity Diagram）

### 2.3.1 前端布局

前端页面包括四个小部分，总共用了两个div，第一个div为选择下拉框和翻译按钮，第二个div为输入左区和翻译右区。

结构图如下：

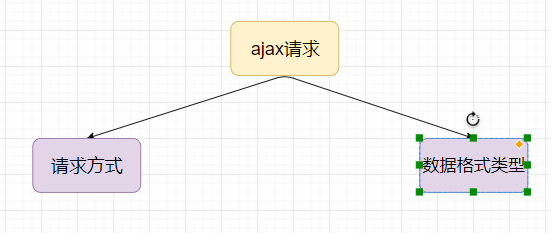


### 2.3.2 ajax请求

onclick事件触发后，发送请求，作为前端与服务器的交互

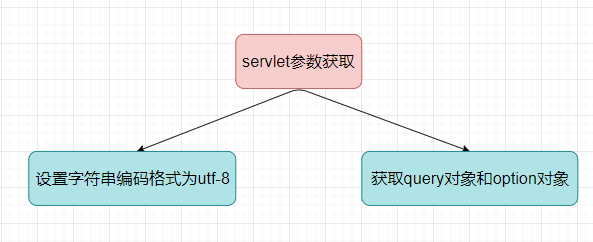
这里请求方式选择为：post

数据格式设置为：json



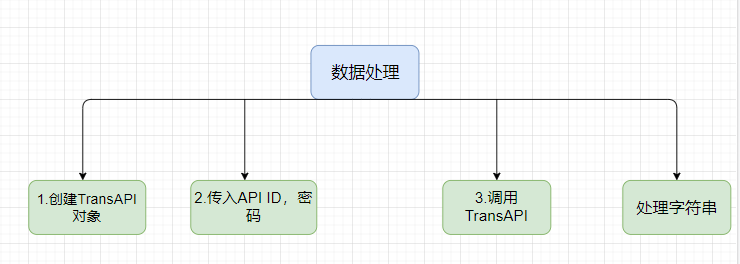
### 2.3.3servlet参数获取

这部分我设置字符集编码格式为utf-8，并获取前端的输入与选择



### 2.3.4 数据处理

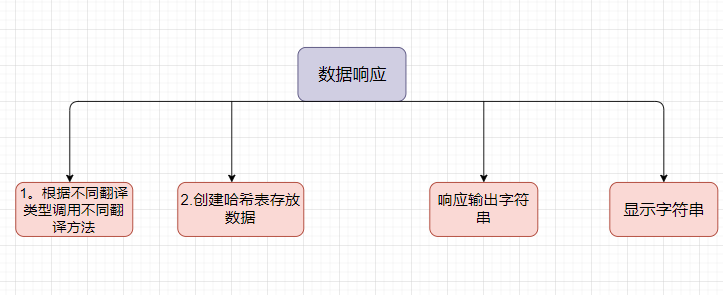
利用API开始对字符串进行翻译，包括创建TransAPI对象，传入API ID密码，调用TransAPI方法，处理字符串



### 2.3.5 响应数据

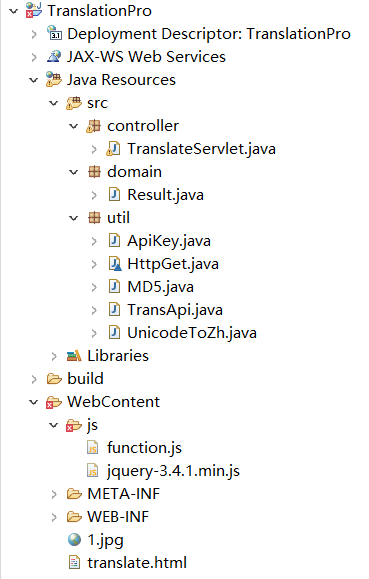
这部分功能是处理后把数据获取并显示到前端

包括根据option的翻译形式调用不同翻译方法，创建哈希表存放数据，输出字符串，最后jquery获取show对象，text方法显示



# **3 系统实现**

## 3.1 项目结构



备注：不知道为什么，项目目录js上会有小红叉，但是js文件都没问题。

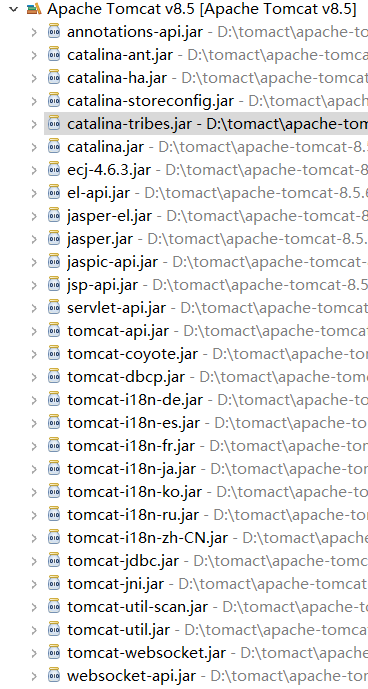
老师可将整个项目下载运行，是可以运行的，小红叉我就不知道是什么原因了

## 3.2 配置文件

### 3.2.1 Tomcat V8.5

Tomcat是一个免费的[开放源代码](https://www.baidu.com/s?wd=%E5%BC%80%E6%94%BE%E6%BA%90%E4%BB%A3%E7%A0%81&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YLrywbnyRsPjTYryPWnHmL0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3En1cknjD4rHDL)的Web 应用服务器，属于轻量级应用服务器，在中小型系统和并发访问用户不是很多的场合下被普遍使用，是开发和调试JSP 程序的首选。另外，Tomcat和IIS等Web服务器一样，具有处理HTML页面的功能，另外它还是一个Servlet和JSP容器，独立的Servlet容器是Tomcat的默认模式。

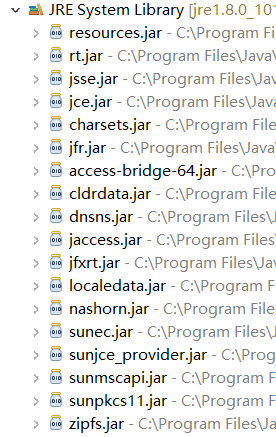
包括如下jar包：



### 3.2.2 JRE System Library

JRE是Java Runtime Environment的缩写，是Java程序的运行环境。既然是运行，当然要包含JVM，也就是所谓的Java虚拟机，还有所以的Java类库的class文件，都在目录下，并且都打包成了jar。

包括如下jar包：



### 3.2.3 gson-2.8.6.jar

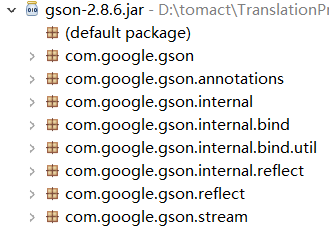
从服务器端接收数据的时候，那些数据必须以浏览器能够理解的格式来发送。

服务器端的编程语言只能以如下 3 种格式返回数据： HTML XML JSON

JSON是一种简单的数据格式，比xml更轻巧。

该配置文件的作用就是实现JSON 对象和 字符串间 转换,

包括如下部分



## 3.3 前端布局文件translate.html

该文件通过设置body和div把整个页面划分成了四小块，分别是选择下拉框，翻译键，左输入区，右翻译区，并且翻译按钮设置了onclick响应事件，点击后会触发js中的tanslation方法

代码如下：

<!DOCTYPE html>

<html>

<head>

<meta charset=*"UTF-8"*>

<title>Insert title here</title>

</head>

<body>

<div class=*"head"*>

<span><select id=*"text"*>

<option value =*"auto"*>自动检测</option>

<option value =*"zh-en"*>中文>>英文</option>

<option value=*"zh-jp"*>中文>>日文</option>

<option value=*"zh-kor"*>中文>>韩文</option>

<option value =*"zh-fra"*>中文>>法文</option>

<option value =*"zh-ru"*>中文>>俄文</option>

<option value=*"zh-th"*>中文>>泰文</option>

<option value=*"en-zh"*>英文>>中文</option>

<option value=*"zh-de"*>中文>>德文</option>

</select></span>

<span><input type=*"button"* value=*"翻译"* style="background: *red*" onclick="translation()"></span>

</div>

<div class=*"body"*>

<span><textarea rows=*"20"* cols=*"100"* id=*"input"* placeholder=*"请输入翻译的内容"*></textarea></span>

<span><textarea rows=*"20"* cols=*"100"* id=*"show"*></textarea></span>

</div>

</body>

<script type=*"text/javascript"* src=*"js/jquery-3.4.1.min.js"*></script>

<script type=*"text/javascript"* src=*"js/function.js"*></script>

</html>

## 3.4前端响应文件function.js

HTML文件中设置了onclick事件，点击瞬间，调用js中的translation方法。并通过ajax进行post请求，之前已经导入json包，这里进行了数据格式规定。最后如果返回成功，就调用text方法显示

**function** translation(){

$.ajax({

type:"post",

url:"translate.do",

//Content-Type:application/x-www-form-urlencoded;

data:{query:$("#input").val(),option:$("#text option:selected").val()},

dataType:"json",

success:**function**(response){

//-------------------------------------------------------------------------------------------

$("#show").text(response.answer);

}

});

## 3.5请求文件HttpGet.java

## 该部分首先需要设置SSL套接字，而后创建url对象，设置响应超时，而后读取服务器数据，并对字符串进行url编码，具体实现如下：

**class** HttpGet {

**protected** **static** **final** **int** ***SOCKET\_TIMEOUT*** = 10000; // 10S

**protected** **static** **final** String ***GET*** = "GET";

**public** **static** String get(String host, Map<String, String> params) {

**try** {

// 设置SSLContext

SSLContext sslcontext = SSLContext.*getInstance*("TLS");

sslcontext.init(**null**, **new** TrustManager[] { *myX509TrustManager* }, **null**);

String sendUrl = *getUrlWithQueryString*(host, params);

URLEncoder.*encode*(sendUrl, "utf-8");

// System.out.println("URL:" + sendUrl);

URL uri = **new** URL(sendUrl); // 创建URL对象

HttpURLConnection conn = (HttpURLConnection) uri.openConnection();

**if** (conn **instanceof** HttpsURLConnection) {

((HttpsURLConnection) conn).setSSLSocketFactory(sslcontext.getSocketFactory());

}

conn.setConnectTimeout(***SOCKET\_TIMEOUT***); // 设置响应超时

conn.setRequestMethod(***GET***);

**int** statusCode = conn.getResponseCode();

**if** (statusCode != HttpURLConnection.***HTTP\_OK***) {

System.***out***.println("Http错误码：" + statusCode);

}

// 读取服务器的数据

InputStream is = conn.getInputStream();

BufferedReader br = **new** BufferedReader(**new** InputStreamReader(is,"utf-8"));

StringBuilder builder = **new** StringBuilder();

String line = **null**;

**while** ((line = br.readLine()) != **null**) {

builder.append(line);

}

String text = builder.toString();

*close*(br); // 关闭数据流

*close*(is); // 关闭数据流

conn.disconnect(); // 断开连接

**return** text;

} **catch** (MalformedURLException e) {

e.printStackTrace();

} **catch** (IOException e) {

e.printStackTrace();

} **catch** (KeyManagementException e) {

e.printStackTrace();

} **catch** (NoSuchAlgorithmException e) {

e.printStackTrace();

}

**return** **null**;

}

**public** **static** String getUrlWithQueryString(String url, Map<String, String> params) {

**if** (params == **null**) {

**return** url;

}

StringBuilder builder = **new** StringBuilder(url);

**if** (url.contains("?")) {

builder.append("&");

} **else** {

builder.append("?");

}

**int** i = 0;

**for** (String key : params.keySet()) {

String value = params.get(key);

**if** (value == **null**) { // 过滤空的key

**continue**;

}

**if** (i != 0) {

builder.append('&');

}

builder.append(key);

builder.append('=');

builder.append(*encode*(value));

i++;

}

**return** builder.toString();

}

**protected** **static** **void** close(Closeable closeable) {

**if** (closeable != **null**) {

**try** {

closeable.close();

} **catch** (IOException e) {

e.printStackTrace();

}

}

}

/\*\*

\* 对输入的字符串进行URL编码, 即转换为%20这种形式

\*

\* **@param** input 原文

\* **@return** URL编码. 如果编码失败, 则返回原文

\*/

**public** **static** String encode(String input) {

**if** (input == **null**) {

**return** "";

}

**try** {

**return** URLEncoder.*encode*(input, "utf-8");

} **catch** (UnsupportedEncodingException e) {

e.printStackTrace();

}

**return** input;

}

**private** **static** TrustManager *myX509TrustManager* = **new** X509TrustManager() {

@Override

**public** X509Certificate[] getAcceptedIssuers() {

**return** **null**;

}

@Override

**public** **void** checkServerTrusted(X509Certificate[] chain, String authType) **throws** CertificateException {

}

@Override

**public** **void** checkClientTrusted(X509Certificate[] chain, String authType) **throws** CertificateException {

}

};

}

## 3.6 API设置文件ApiKey.java

首先我们需要去百度官网注册和申请API，步骤如下：





填完申请表后我们就获得一个ID和key，该java文件作用就是传入API ID和KEY

**package** util;

**public** **class** ApiKey {

**private** **static** **final** String ***APP\_ID*** = "20201217000649707";

**private** **static** **final** String ***SECURITY\_KEY*** = "Len5h23uuA0z3Kpn4k3Z";

**public** **static** String getAppId() {

**return** ***APP\_ID***;

}

**public** **static** String getSecurityKey() {

**return** ***SECURITY\_KEY***;

}

}

## 

## 3.7 API处理文件TransAPI.java

## 该文件首先获取了上一个文件传入的APIID和key，而后对用户输入和选择内容进行初步处理，option就是下拉框选择，from是翻译前的语种，to是翻译后的语种，appid就是百度翻译给的API ID，把这些传进来的参数放入一个map容器进行处理，这里我设了一个和时间随机数，以保证在不同时刻启动时都能新的响应，让每一次响应唯一。具体代码如下：

**public** **class** TransApi {

**private** **static** **final** String ***TRANS\_API\_HOST*** = "http://api.fanyi.baidu.com/api/trans/vip/translate";

**private** String appid;

**private** String securityKey;

**public** TransApi(String appid, String securityKey) {

**this**.appid = appid;

**this**.securityKey = securityKey;

}

**public** String getTransResult(String query, String from, String to) {

Map<String, String> params = buildParams(query, from, to);

**return** HttpGet.*get*(***TRANS\_API\_HOST***, params);

}

**private** Map<String, String> buildParams(String query, String from, String to) {

Map<String, String> params = **new** HashMap<String, String>();

params.put("q", query);

params.put("from", from);

params.put("to", to);

params.put("appid", appid);

// 随机数

String salt = String.*valueOf*(System.*currentTimeMillis*());

params.put("salt", salt);

// 签名

String src = appid + query + salt + securityKey; // 加密前的原文

**try** {

URLEncoder.*encode*(src, "utf-8");

} **catch** (UnsupportedEncodingException e1) {

// **TODO** Auto-generated catch block

e1.printStackTrace();

}

**try** {

params.put("sign", MD5.*md5*(src));

} **catch** (UnsupportedEncodingException e) {

// **TODO** Auto-generated catch block

e.printStackTrace();

}

**return** params;

}

}

## 3.8 加密文件MD5.java

为了保证整个项目的安全性，我参考百度上网友做法，设置了MD5加密文件。MD5是将一个任意长度的字符串变换成一个128位的大整数，而且这是一个不可逆的变换算法，保证我们整个项目的安全性。具体代码如下：

/\*\*

\* MD5编码相关的类

\*

\* @author wangjingtao

\*

\*/

public class MD5 {

// 首先初始化一个字符数组，用来存放每个16进制字符

private static final char[] hexDigits = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd',

'e', 'f' };

/\*\*

\* 获得一个字符串的MD5值

\*

\* @param input 输入的字符串

\* @return 输入字符串的MD5值

\* @throws UnsupportedEncodingException

\*

\*/

public static String md5(String input) throws UnsupportedEncodingException {

if (input == null)

return null;

try {

// 拿到一个MD5转换器（如果想要SHA1参数换成”SHA1”）

MessageDigest messageDigest = MessageDigest.getInstance("MD5");

// 输入的字符串转换成字节数组

byte[] inputByteArray = input.getBytes("utf-8");

// inputByteArray是输入字符串转换得到的字节数组

messageDigest.update(inputByteArray);

// 转换并返回结果，也是字节数组，包含16个元素

byte[] resultByteArray = messageDigest.digest();

// 字符数组转换成字符串返回

return byteArrayToHex(resultByteArray);

} catch (NoSuchAlgorithmException e) {

return null;

}

}

/\*\*

\* 获取文件的MD5值

\*

\* @param file

\* @return

\*/

public static String md5(File file) {

try {

if (!file.isFile()) {

System.err.println("文件" + file.getAbsolutePath() + "不存在或者不是文件");

return null;

}

FileInputStream in = new FileInputStream(file);

String result = md5(in);

in.close();

return result;

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

return null;

}

public static String md5(InputStream in) {

try {

MessageDigest messagedigest = MessageDigest.getInstance("MD5");

byte[] buffer = new byte[1024];

int read = 0;

while ((read = in.read(buffer)) != -1) {

messagedigest.update(buffer, 0, read);

}

in.close();

String result = byteArrayToHex(messagedigest.digest());

return result;

} catch (NoSuchAlgorithmException e) {

e.printStackTrace();

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

return null;

}

private static String byteArrayToHex(byte[] byteArray) {

// new一个字符数组，这个就是用来组成结果字符串的（解释一下：一个byte是八位二进制，也就是2位十六进制字符（2的8次方等于16的2次方））

char[] resultCharArray = new char[byteArray.length \* 2];

// 遍历字节数组，通过位运算（位运算效率高），转换成字符放到字符数组中去

int index = 0;

for (byte b : byteArray) {

resultCharArray[index++] = hexDigits[b >>> 4 & 0xf];

resultCharArray[index++] = hexDigits[b & 0xf];

}

// 字符数组组合成字符串返回

return new String(resultCharArray);

}

}

## 3.9 编码设置文件UnicodeToZh.java

Unicode是一种字符编码方案，它为每种语言中的每个字符都设定了统一唯一编码，以实现跨语言、跨平台进行文本转换、处理的要求。这里我们要把其他语言翻译成中文，需要把unicode转化为中文，具体见如下代码和文字注释：

**public** **class** UnicodeToZh {

**public** **static** String unicodeToCn(String unicode) {

/\*\* 以 \ u 分割，因为java注释也能识别unicode，因此中间加了一个空格\*/

String[] strs = unicode.split("\\\\u");

String returnStr = "";

// 由于unicode字符串以 \ u 开头，因此分割出的第一个字符是""。

**for** (**int** i = 1; i < strs.length; i++) {

returnStr += (**char**) Integer.*valueOf*(strs[i], 16).intValue();

}

**return** returnStr;

}

}

## 3.10控制器文件TranslateServlet.java

在这个控制器文件里，我首先对字符集编码格式设置为了utf-8,而后对用户下拉框内容option和输入区内容query进行分析，option就是下拉框选择，from是翻译前的语种，to是翻译后的语种，answer是翻译结果。这里首先对下拉框进行分析，如果选择的是auto自动识别，那么就默认翻译为英文，如果不是自动检测，那就先用split函数对选择框的字符串进行分割，把“-”剥离出来处理，而后调用api翻译。那如果是转化成中文就要先调用unicodetozh的方法再显示，如果不是，那就可以直接json格式显示。

这里说一下Substring函数应该是从括号传的参数开始查找子串，所以括号里的参数必须保证是“：”的后一位。LastIndexOf取的是“：”最后一次出现的索引，再加上字符串的引号，所以我们需要加2，最后就是把处理后的字符串放到map容器内，用text方法显示，详细代码如下：

import java.io.IOException;

import java.io.PrintWriter;

import java.net.URLEncoder;

import java.util.HashMap;

import java.util.Map;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import com.google.gson.Gson;

import domain.Result;

import util.ApiKey;

import util.TransApi;

import util.UnicodeToZh;

@WebServlet("/translate.do")

public class TranslateServlet extends HttpServlet{

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {

// TODO Auto-generated method stub

//System.out.println("进入该servlet");

request.setCharacterEncoding("utf-8");

String q=request.getParameter("query");

URLEncoder.encode(q, "utf-8");

System.out.println(q);

String option=request.getParameter("option");

System.out.println(option);

String from="";

String to="";

String answer="";

if(option.equals("auto")) {

from="auto";

to="en";

}

else {

from=option.split("-")[0];//把分离下来的字符看成数组，左边相当于第一个元素，下标为0

to=option.split("-")[1];

}

System.out.println(from);

System.out.println(to);

TransApi api = new TransApi(ApiKey.getAppId(), ApiKey.getSecurityKey());

String jsonResult=api.getTransResult(q, from, to);

System.out.println(jsonResult);

if(option.equals("en-zh")) {

answer=UnicodeToZh.unicodeToCn(jsonResult.substring(jsonResult.lastIndexOf(":")+2,jsonResult.indexOf("\"}")));

}else {

answer = jsonResult.substring(jsonResult.lastIndexOf(":")+2,jsonResult.indexOf("\"}"));

}

Map<String,String> res=new HashMap<String, String>();

res.put("answer", answer);

String jsonStr=new Gson().toJson(res);

response.setContentType("text/html;charset=utf-8");

PrintWriter pw=response.getWriter();

pw.printf(jsonStr);

pw.flush();

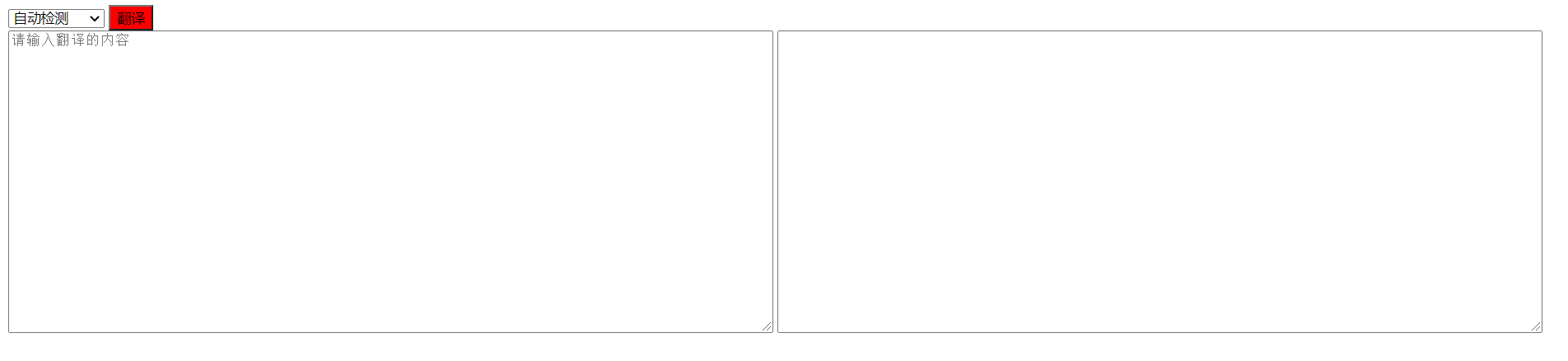
pw.close();

}

}

# **4 系统测试**

## 空页面



## 自动检测

4.1.1第一次测试



4.1.2第二次测试



4.1.3第三次测试



## 中-英



## 中-法



## 中-德



## 英-文



## 系统测试总结

综合上面的测试结果，我们发现我们的项目能够调用api实现翻译，符合实验预期与老师的要求

# **5 系统总结**

## 5.1参考资料

1. 《web应用开发》电子工业出版社
2. 《web前端开发实战》上海交通大学出版社
3. 菜鸟教程官网https://www.runoob.com/
4. 百度翻译开放平台https://api.fanyi.baidu.com/
5. 以及csdn各类来自大佬网友的帖子

衷心对以上书籍编者和百度翻译官网，菜鸟教程官网，csdn等技术人员表示感谢！

## 5.2系统总结

这次我们实现的是用服务器调用api翻译，把翻译结果显示到前端上，经测试，我们的实验结果符合预期，实验基本成功，这次实验，是第一次做这种调用api型的东西，对我来说也是全新的挑战，因为需要全新的思路，需要大量查阅资料和csdn论坛的帖子。比如如何获取API，如何处理系统返回的字符串，如何实现MD5加密都去网上csdn论坛看了教程或者是参考案例。

但是，该程序无法处理特殊字符和自动添加标点符号，比如有道官网来一句：你今天吃饭了么，翻译结果会自动加上问号，当然，实话实说，目前我们还没有能力把自动添加标点符号做出来，需要以后在工作中逐步完善自己的能力。

其次无法处理特殊字符，智能实现普通文字和字母翻译，比如韩文作为特殊字符会乱码，这个问题后面的学习中还可以慢慢探索和解决，和聂老师交流了一下，也有可能是操作系统不支持。

**由于时间关系，我这次加紧时间做后端处理，前端做的很简单，我可能后面把前端完善以后还会再交一次，到时会删除目前这个文档和项目，然后重新上交新的。**

## 5.3课程总结

聂老师是非常负责任的老师，这学期web课四节连堂，但聂老师不辞辛苦，就这么一直坚持讲下去，并且由于要在黑板上板书，聂老师经常都是站着的，非常辛苦。聂老师的课堂循序渐进，非常生动，知识结构完整，教学体系清晰，并且注重和同学们互动，再加上聂老师自己辛苦写的web应用开发教案，这学期收获很多。能够把前端和后端进行交互，和一年前只会基本前端的我来说进步巨大。

一学期结束了，但是学海无涯，学习永无止境，后面的学习中我还会继续跟着聂老师学习，探索新的知识，完善自己的技能。

## 5.4自我评价

这次实验我是达到了聂老师的要求，并且不止能做出中英文翻译，还能实现自动检测和简单的小语种翻译，总体来说还是不错的。不过目前还不能实现和翻译官网完全一致的全语种翻译，后面学习中可以往这方面深入探索，查询其它语言和中文之间编码方式转换，把我们这个项目做得更好。

目前我给自己的评分为：90分

## 5.5展望未来

我国在高新技术产业近年来飞速发展，美国的GPS要比我国的北斗导航系统早很多年，但我国后来居上，北斗导航系统在精度上已经领先GPS，这是中国制造和中国创造的魅力。

但是，不得不说，我国还有很多方面与发达国家存在差距。首先是软件系统和硬件芯片领域。目前，中国还没有完全成熟和广泛运用的手机操作系统和电脑操作系统。手机系统界被谷歌，苹果垄断。电脑系统界被微软，苹果垄断。在芯片领域，由于我国目前无法生产高精度的芯片，手机芯片被高通，苹果，三星等公司垄断，电脑芯片被英特尔，AMD，英伟达等公司垄断。

尽管我国目前正处在被发达国家卡脖子的状态，但是一批年轻的青年公司如雨后春笋般诞生，目前华为已拥有高端麒麟芯片设计能力和初步的鸿蒙系统，小米也正在研发澎湃系列芯片。

我坚信，在中华民族顽强不屈，砥砺奋进的精神鼓舞下，在全计算机IT行业工程师的共同努力下，我国的高科技领域将会逐渐迎难而上，不断进步，勇往直前。  
 而我的目标，就是成为能我国IT行业进步的无数工程师中的一个。

让中国制造惠及全球，让中国创造引领时代，这是每一个青年工程师的担当。

正如在华为鸿蒙系统发布会上华为消费者业务总裁余承东所说：

没有人能熄灭满天星光

每一位工程师，都是祖国要汇聚的星星之火！

而我，将为成为这样的星星之火努力奋斗！

