# 接口设计

## 内部接口

用户登录：

参数类型：

用户名 String

用户密码 ：String

返回值类型 ：bool

功能描述 ：用户登录接口实现用户的登录，如果用户名和用户密码可以进行对应，则登录成功，返回true。否则返回false

用户注册：

参数类型：

用户名 String

用户密码 ：String

返回值类型 ：bool

功能描述 ：用户注册接口实现用户的注册，如果用户名之前没有重复，则注册成功，返回true。否则返回false

图片分割：

参数类型：

图片内容 String

用户名称 ：String

返回值类型 ：String

功能描述 ：图片分割实现了将待处理的图片实现分割，完成证件照制作的相关功能。该接口可以将待处理的图片以及处理完成的图片全部保存在服务器中，并且可以将图片元信息保存至数据库中。返回值是处理完成的图片的图片名。

图片美化：

参数类型：

图片内容 String

用户名称 ：String

返回值类型 ：String

功能描述 ：图片美化实现了将待处理的图片实现美化，完成证件照制作的相关功能。该接口可以将待处理的图片以及处理完成的图片全部保存在服务器中，并且可以将图片元信息保存至数据库中。返回值是处理完成的图片的图片名。

public class AlgorithmReq {

public static String ImageToBase64(String imagePath) {

InputStream in = null;

byte[] data = null;

try {

in = new FileInputStream(imagePath);

data = new byte[in.available()];

in.read(data);

} catch (IOException e) {

e.printStackTrace();

} finally {

try{

if(in != null){

in.close();

}

}catch (IOException e){

e.printStackTrace();

}

}

Encoder encoder = Base64.getEncoder();

return encoder.encodeToString(data);

}

public static String doPost() {

String imagePath = "lib/time.jpg";

String imgStr = ImageToBase64(imagePath);

OutputStreamWriter out = null;

BufferedReader in = null;

StringBuilder result = new StringBuilder();

HttpURLConnection conn = null;

try{

URL url = new URL("https://openapi.mtlab.meitu.com/v3/makeup?api\_key=xxx&api\_secret=xxx");

conn = (HttpURLConnection) url.openConnection();

conn.setRequestMethod("POST");

conn.setDoOutput(true);

conn.setDoInput(true);

conn.setConnectTimeout(30000);

conn.setReadTimeout(10000);

conn.setRequestProperty("Content-Type", "application/json");

out = new OutputStreamWriter(conn.getOutputStream());

String jsonStr = "{\"parameter\":{\"beautyAlpha\":70},\"extra\":{},\"media\_info\_list\":[{\"media\_data\":\""+ imgStr +"\",\"media\_profiles\":{\"media\_data\_type\":\"jpg\"}}]}";

out.write(jsonStr);

out.flush();

out.close();

if (200 == conn.getResponseCode()){

in = new BufferedReader(new InputStreamReader(conn.getInputStream(), "UTF-8"));

}else{

System.out.println("ResponseCode is an error code:" + conn.getResponseCode());

in = new BufferedReader(new InputStreamReader(conn.getErrorStream(), "UTF-8"));

}

String line;

while ((line = in.readLine()) != null){

result.append(line);

System.out.println(line);

}

}catch (Exception e){

e.printStackTrace();

}finally {

try{

if(out != null){

out.close();

}

if(in != null){

in.close();

}

}catch (IOException ioe){

ioe.printStackTrace();

}

}

return result.toString();

}

public static void main(String[] args) {

AlgorithmReq.doPost();

}

}

## 外部接口

### 接口说明

### 该系统通过Restful API的方式从深度学习模块取得处理后的图片的相关信息。

## 接口描述：识别人体的轮廓范围，与背景进行分离，适用于拍照背景替换、照片合成、身体特效等场景。输入正常人像图片，返回分割后的**二值结果图、灰度图、透明背景的人像图（png格式）**；**并输出画面中的人数、人体坐标信息，可基于此对图片进行过滤、筛选**，如筛选出大于x人的图片进行特殊处理。

## 错误返回格式

若请求错误，服务器将返回的JSON文本包含以下参数：

* **error\_code**：错误码。
* **error\_msg**：错误描述信息，帮助理解和解决发生的错误。

## 请求说明

**请求示例**

HTTP 方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/image-classify/v1/body\_seg

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token，参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/x-www-form-urlencoded |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 是否必选 | 类型 | 可选值范围 | 说明 |
| --- | --- | --- | --- | --- |
| image | 是 | string | - | 图像数据，base64编码后进行urlencode，要求base64编码和urlencode后大小不超过4M。图片的base64编码是不包含图片头的，如(data:image/jpg;base64,)，支持图片格式：jpg、bmp、png，最短边至少50px，最长边最大4096px |
| type | 否 | string | labelmap,scoremap,foreground | 可以通过设置type参数，自主设置返回哪些结果图，避免造成带宽的浪费 1）可选值说明： labelmap - 二值图像，需二次处理方能查看分割效果 scoremap - 人像前景灰度图 foreground - 人像前景抠图，透明背景 2）type 参数值可以是可选值的组合，用逗号分隔；如果无此参数默认输出全部3类结果图 |

### 调用方式

内部接口调用：

通过传入图片的base64编码，调用该外部接口，针对返回值进行下一步的处理

将返回值的某些额外例如，检测到的人体框数目，人体框信息，人体区域的高度，也可以进行相关的处理。最终内部接口返回处理后的图片名，方便前端进行调用。

实例：

String imgParam = URLEncoder.*encode*(imgStr, "UTF-8");  
  
String param = "image=" + imgParam;  
  
// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。  
String accessToken = AuthService.*getAuth*();  
/\*

public static String getAuth(String ak, String sk) {  
 // 获取token地址  
 String authHost = "https://aip.baidubce.com/oauth/2.0/token?";  
 String getAccessTokenUrl = authHost  
 // 1. grant\_type为固定参数  
 + "grant\_type=client\_credentials"  
 // 2. 官网获取的 API Key  
 + "&client\_id=" + ak  
 // 3. 官网获取的 Secret Key  
 + "&client\_secret=" + sk;  
 try {  
 URL realUrl = new URL(getAccessTokenUrl);  
 // 打开和URL之间的连接  
 HttpURLConnection connection = (HttpURLConnection) realUrl.openConnection();  
 connection.setRequestMethod("GET");  
 connection.connect();  
 // 获取所有响应头字段  
 Map<String, List<String>> map = connection.getHeaderFields();  
 // 遍历所有的响应头字段  
 for (String key : map.keySet()) {  
 System.*err*.println(key + "--->" + map.get(key));  
 }  
 // 定义 BufferedReader输入流来读取URL的响应  
 BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));  
 String result = "";  
 String line;  
 while ((line = in.readLine()) != null) {  
 result += line;  
 }  
 */\*\*  
 \* 返回结果示例  
 \*/* System.*err*.println("result:" + result);  
 JSONObject jsonObject = new JSONObject(result);  
 String access\_token = jsonObject.getString("access\_token");  
 return access\_token;  
 } catch (Exception e) {  
 System.*err*.printf("获取token失败！");  
 e.printStackTrace(System.*err*);  
 }  
 return null;  
}

\*/  
String result = HttpUtil.*post*(url, accessToken, param);

/\* public static String post(String requestUrl, String accessToken, String contentType, String params, String encoding)  
 throws Exception {  
 String url = requestUrl + "?access\_token=" + accessToken;  
 return HttpUtil.*postGeneralUrl*(url, contentType, params, encoding);  
}

\*/  
JsonObject obj = GsonUtils.*fromJson*(result, JsonObject.class);  
  
JsonElement foreground = obj.get("foreground");  
  
Renew.*GenerateImage*(imgStr,FileUtil.*GetFrontEndPath*(fileName));  
String base64NewImg = GsonUtils.*toJson*(foreground);  
Renew.*GenerateImage*(base64NewImg, FileUtil.*GetFrontEndPath*("modify-"+ fileName));  
return "modify-" + fileName;