用户地址服务V1.2

0. 版本信息

版本	时间
V1.0	2022-04-20
V1.1	2022-05-09
V1.2	2022-08-24

1. 场景一：用户地址核验

> 验证用户手机号与地址是否匹配，适合用户地址真伪的核验

1.1 接口信息

接口地址： https://leoapi.zt-express.com/checkUserAddress

请求类型：POST

接口功能：通过手机号和详细地址判断用户是否匹配

签名公式：在文档第3节

1.2 请求头信息

HTTP请求Header头参数：

```
x-zop-ns：leo-api
Content-Type：application/json
x-timestamp：当前时间戳（必须传递当前调用方操作时的13位时间戳，为了重放攻击，时间戳有效时间是一小时）
x-sign：签名（计算方法下面说明）
```

1.3 请求参数

参数名称	参数类型	必填	备注
`mobile`	String	是	11位手机号
`address`	String	是	详细地址， 如：上海上海市嘉定区XXX路XX小区4栋101室
`addressLevel`	Integer	否	地址精确等级 *（中通内部根据详细地址做地址翻译，根据传入地址的详细层度能解析出省、市、区、镇、村、路、终端等信息，请根据实际业务场景控制精确层度，能提高识别率）* 默认值：0 枚举值如下： 0（全匹配） 1 一级地址（省） 2 二级地址（市） 3 三级地址（区） 4 四级地址（镇） ~~5 五级地址（村）~~ ~~6 六级地址（路）~~

```
{
  "mobile": "17701824572",
  "address": "上海上海市嘉定区江桥镇金沙江西路1555弄西郊商务20号楼9层",
  "addressLevel": 4 // 匹配到镇级别（address详细地址中要包含镇信息）
}
```

> 注意： mobile 手机号参数必须为11位的标准手机号，不用加 86 等额外信息 address 详细地址信息，目前必须包含省市区信息（后续升级地址标准化接口能自动补全） addressLevel 地址精确等级，若不传为全地址匹配（建议按照业务等级传精确等级，提高识别率）

1.4 响应结果

```
{
  "result": 1, // 结果信息 （具体取值如下列表）
  "message": "匹配成功", // 提示信息
  "statusCode": "SYS000", // 状态码code，用来排查问题
  "status": true // 请求是否出错
}
```

通过返回的result参数能判断数据匹配情况，具体参考如下：

- 1 匹配成功
- 0 无匹配信息
- -1 无数据 （2022年12月2日新增）
- 99 未知信息无法确定

签名错误的提示：

```
{
  "message": "签名错误",
  "result": null,
  "status": false,
  "statusCode": "S211"
}
```

参数错误的提示：

```
{
  "message": "addressLevel参数错误，范围为0-6，请检查",
  "result": null,
  "status": false,
  "statusCode": "S400"
}
```

1.5 请求示例

```
curl 'https://leoapi.zt-express.com/checkUserAddress' \
  -H 'Connection: keep-alive' \
  -H 'x-zop-ns: leo-api' \
  -H 'sec-ch-ua-mobile: ?0' \
  -H 'User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/99.0.4844.51 Safari/537.36' \
  -H 'Content-Type: application/json' \
  -H 'Accept: */*' \
  -H 'x-timestamp: 1650461840201' \
  -H 'x-sign: 18xYLZGnyjFyvPDIKq6kIw==' \
  -H 'sec-ch-ua-platform: "macOS"' \
  -H 'sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="99", "Google Chrome";v="99"' \
  -H 'Origin: chrome-extension://ieoejemkppmjcdfbnfphhpbfmallhfnc' \
  -H 'Sec-Fetch-Site: none' \
  -H 'Sec-Fetch-Mode: cors' \
  -H 'Sec-Fetch-Dest: empty' \
  -H 'Accept-Language: zh-CN,zh;q=0.9' \
```

```
  -H 'Cookie:
wyandyy=eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE2NTA0ODQ4MDAsImlhdCI6MTY1MDQxOT
YyMCwiaXNzIjoiY29tLnp0by5jb25uZWN0IiwibmJmIjoxNjUwNDE5NjIwLCJzZXNzaW9uaWQiOiJzNDFWdWNCT
UVleXlZUUJRVnEwaTBBfDE2NTA0MTk2MjAiLCJ1dWlkIjoiRXZGem4zT3lXeG1CZXI1WkoxdU9kdyJ9.t4VeWil
WCFC4MsmaQd9xaJG1CKz3gOsiWfBLSoT6opFnn9nq79OvqVZKI42PFFEe--
wwO6SrubU9rBg5UaeJGm2Tha38iFmbx7hHZplLXyaOFku8A6EscG2ZMXhGKQ5BBFw8aUDSyYyR-
KtzxE4VyKqIjDIYl56kmOfWaZ-
iHMwq9kRp3clM0Mtoi50ii0I7mxLtcfirrUwoc05fv4Gz8WbQ2IJidXesfIr0ziONf3_euJkv9XJtl88l1-
PDAMUn3p8Bt4T5Guqj3KvLrd3xdMB2ckvmPD4rVfY_dI_ukEPXWmWTEKg-
FPa2GigixPk4aSiMmYVVQxegJGH4wjjAsw;
wyzdzjxhdnh=eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE2NTE2MjkyMjAsImlhdCI6MTY1MD
QxOTYyMCwiaXNzIjoiY29tLnp0by5jb25uZWN0IiwibmJmIjoxNjUwNDE5NjIwLCJ1dWlkIjoiRXZGem4zT3lXe
G1CZXI1WkoxdU9kdyJ9.rWsN36aPszamm7gMG5dH5eNqNLoEXDAMZGwwqlMSDuw2mUpWV8Cgay15z0NdQBEUhfe
WMKsyCjSXMAs-sVetptdTKOw0HwuWaTDtsrIUBMLcFEM-93_AK_Hm_ad-
0HFw4N7B5gTFX82scszHzaMDCdpgdnandfT7nLsGgb1tWMGw5Z3bBtVb31-
ktHIzSg4z9JJAvn7JHgZ5Od6hqDW8oR1TvemqTIq0-
NVYtiXGrQMbLps_4cykyTr1xGKlFsDHxXSB0uhfz1eYgqIPsr-
e9TKJLSJB__DrOQwEPiIGwiAyGXGxDtsJOnCSUqQrslO_oDWWM3s0gVdP9ZSQ40nqcA' \
  --data-raw $'{\n"address": "上海上海市嘉定区江桥镇金沙江西路1555弄西郊商务21号楼8
层",\n"mobile": "17701854571"\n}' \
  --compressed

  {"result":1,"message":"匹配成功","statusCode":"SYS000","status":true}
```

1.6 参考代码

签名代码参考：

```
package com.zto.intelligence;
import org.apache.commons.codec.digest.DigestUtils;
import java.util.Base64;
public class SignTest {

    public static void main(String[] args) throws Exception{
        String key = "xxxxxx"; // 找我私下获取
        long time = System.currentTimeMillis();
        String body = "{\n" +
                "\"address\": \"上海上海市嘉定区江桥镇金沙江西路1555弄西郊商务21号楼8层\",\n" +
                "\"mobile\": \"17701854571\"\n" +
                "}";
        byte[] md5 = DigestUtils.md5((time + body + key).getBytes("UTF8"));
        String sign = new String(Base64.getEncoder().encode(md5));
        System.out.println("x-timestamp: " + time);
        System.out.println("x-sign: " + sign);
    }

}
```

目前底层的用户信息由大数据定期清洗获得，最新快递单的地址数据可能存在延迟写入。

目前匹配率完全依赖于传入的地址是否标准，其中我们的地址库只能补全到镇的级别，请尽可能传入类似电商渠道的标准地址。

2. 场景二：用户消费等级查询（是否高净值人群）

> 查询用户所属人群等级，适合判断是否定向人群的场景。

说明：该接口返回数据包含3个层级（1 高净值人群，2 中端人群，3 普通人群），数据结合地址和房产数据计算获取。

1.1 接口信息

接口地址： https://leoapi.zt-express.com/getUserLevel

请求类型：POST

接口功能：查询用户所属人群等级

签名公式：在文档第3节

1.2 请求头信息

HTTP请求Header头参数：

```
x-zop-ns：leo-api
Content-Type：application/json
x-timestamp：当前时间戳（必须传递当前调用方操作时的13位时间戳，为了重放攻击，时间戳有效时间是一小时）
x-sign：签名（计算方法下面说明）
```

1.3 请求参数

参数名称	参数类型	必填	备注
`mobile`	String	是	11位手机号
`address`	String	是	详细地址， 如：上海上海市嘉定区XXX路XX小区4栋101室

```
{
  "mobile": "17701854571",
  "address": "上海上海市嘉定区江桥镇金沙江西路1555弄西郊商务21号楼8层"
}
```

> 注意： mobile 手机号参数必须为11位的标准手机号，不用加 86 等额外信息 address 详细地址信息，目前必须包含省市区信息（后续升级地址标准化接口能自动补全）

1.4 响应结果

```
{
  "result": 1, // 结果信息 （具体取值如下列表）
  "message": "匹配成功", // 提示信息
  "statusCode": "SYS000", // 状态码code，用来排查问题
  "status": true // 请求是否出错
}
```

通过返回的result参数能判断数据匹配情况，具体参考如下：

- 0 无用户信息
- 1 高净值人群
- 2 中端消费人群
- 3 普通人群

签名错误的提示：

```
{
  "message": "签名错误",
  "result": null,
  "status": false,
  "statusCode": "S211"
}
```

参数错误的提示：

```
{
  "message": "参数错误，请检查参数是否正确",
  "result": null,
  "status": false,
  "statusCode": "S400"
}
```

3. 签名计算公式

```
// timestamp为请求头中的x-timestamp值
// body为json格式的参数
// sign = base64(md5(timestamp + body + 签名key))

byte[] md5 = DigestUtils.md5((time + body + key).getBytes("UTF8")); //依赖commons-codec-1.10的包
String sign = new String(Base64.getEncoder().encode(md5));
```

> 注意:签名key需要开发人员线下索取，明文写在文档中存在安全风险，谢谢!