# 百度号码归属地查询

# API 接口文档

| | |
|---|---|
| **版本** | 6．0 |
| **作者** | 百度移动安全部 |
| **发布日期** | -/-/- |

# 修订记录

| 版本 | 修订内容 | 修订人 | 修改日期 |
| --- | --- | --- | --- |
| 6．0 | 接口文档排版 | 傅正(f u z h e n g @b) a i | 2016．21 |
| | | | |

# 目录

## 1.1 API 背景

本接口提供号码归属地信息查询服务。查询方通过提供相应参数信息（具体参数信息及示例见下）以 POST 方式查询号码相关信息，系统对请求处理后返回结果（具体信息格式及示例见下）。

## 2.1 API 使用说明

### 2.1.1 服务请求域名

| 环境 | 域名 |
| --- | --- |
| 生产环境 | http://mobsec-dianhua.baidu.com |
| 沙盒环境 | http://sandbox.sjws.baidu.com:8080 |

**其中使用云端提供的沙盒环境，将使用到的相关参数如下所示：**

**appkey:** 50b13132bb394901f151bc12

**appsecret:** 50b13132bb394901f151bc12

### 2.1.2 请求格式及参数说明

| 格式 | 说明 |
| --- | --- |
| 方法 | POST |
| URL 格式 | http://域名/dianhua_api/location/query/1.0 |
| URL 必填参数名 | appkey，tk(设备标识)，nonce（当前时间，UTC 时间），auth_ver(签名版本，默认写 2), checkStr（加密后的请求体签名），s（url 参数签名） |

**注意：**顾名思义以上 URL 必填参数名为必填项，否则会被拦截。

**POST 参数：**

采用 JSON 对象，具体格式（样式）如下：

```
{
        "createTime" : "utc 时间，long 型"，  //毫秒
        "msg": "加密后的 base64 字符串"

}
```

### 2.1.3 返回格式及参数说明

#### 返回的 HTTP 头中状态码及含义

| 状态码 | 含义 |
| --- | --- |
| 200 | OK |
| 304 | 支持 |
| 400 | 请求的参数有误 |
| 401 | 未认证 |

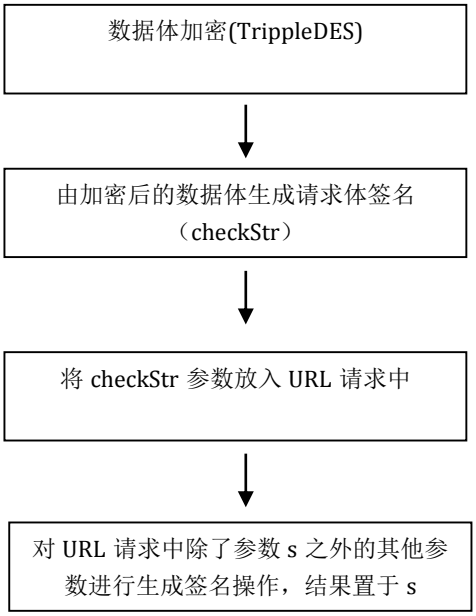
#### 返回 Body 体样式

返回的 Body 体样式如下所示


```
{
	"response" : {
		"datas" : {
			"13838383838" : {
				"operation" : "移动",
				"province" : "河南",
				"city" : "郑州"
			},
			"18610328820" : {
				"operation" : "联通",
				"province" : "北京",
				"city" : "北京"
			}
		}
	},
	"responseHeader" : {
		"status" : 200,
		"time" : 1441780364613,
		"version" : "1.1.0"
	}
}
```

# 3.1 数据传输安全

## 3.1.1 传输安全加密流程图




## 3.1.2 URL 参数签名（即 URL 中参数 s）

为了防止 URL 被恶意篡改和验证请求，需要对 URL 的参数做签名校验，流程是对请求参数排序，拼接成字符串，然后计算 MD5。（URL 参数签名是对除了 s 参数外的其他参数计算 MD5(appkey+参数值+auth_ver+其数值+.....)）

```
Map<String, String>signParams =new TreeMap<String,String>()
for (Map.Entry<String, String> entry : valueMap.entrySet()) {
       signParams.put(entry.getKey(), entry.getValue());
}
StringBuilder paramsStr = new StringBuilder();
Set<Map.Entry<String, String>>  paramsEntry = signParams.entrySet();
for (Map.Entry<String, String> entry : paramsEntry) {
        paramsStr.append(entry.getKey()).append(entry.getValue()
             == null ? "-" : entry.getValue());
}
paramsStr.append(app.getAppSecret());
String sign = DigestUtils.md5Hex(paramsStr.toString().getBytes("UTF-8"));
```

### 3.1.3 数据加密

#### 一、加密代码（可参考以下代码）

```
String data="{"location": "可缺省，例如：北京","iccid": "","cell" : { "mcc" :
"","mnc" : "","lac" : "","id": ""},"phones": ["123","345"]}";
String DEFAULT_CHARASET = "utf-8";
Long time = System.currentTimeMillis();
JsonObject content = new JsonObject();
content.addProperty("createTime",time);
content.addProperty("msg", encryptWithBase64 (appsecret 的前 24 位, time, data));
public static byte[] longToBytes(long v) {
   ByteBuffer byteBuffer = ByteBuffer.allocate(8);
   byteBuffer.order(ByteOrder.BIG_ENDIAN);
   byteBuffer.putLong(v);
   return byteBuffer.array();
    }
public static String encryptWithBase64(String key, long iv, String message) {
try {
   byte[] cipherText = encrypt(
            key.getBytes(DEFAULT_CHARASET),
            longToBytes(iv),
            message.getBytes(DEFAULT_CHARASET));
   return Base64.encodeBase64String(cipherText);
} catch (UnsupportedEncodingException e) {
          throw new SecurityException(e);
        }
}
public static byte[] encrypt(byte[] key, byte[] iv, byte[] message) {
        byte[] result = null;
try {
       Cipher cipher = Cipher.getInstance("DESede/CBC/PKCS5Padding");
       SecretKeyFactory skf = SecretKeyFactory.getInstance("DESede");
       SecretKey secretKey = skf.generateSecret(new DESedeKeySpec(key));
       IvParameterSpec ivParameterSpec = new IvParameterSpec(iv);
       cipher.init(Cipher.ENCRYPT_MODE, secretKey, ivParameterSpec);
       result = cipher.doFinal(message);
 } catch (Exception e) {
```

```
            throw new SecurityException(e);
        }
 return result;
}
```

## 二、加密样例

### （1） 加密前数据：

```
{
      "phones" : [ "123" ] // 电话号码，多号码查询是采用英文逗号分隔
}
```

### （2）加密后数据：

```
{
      "createTime"  : 123456789,          //请求体创建时间（毫秒）
      "msg": "加密后的 base64 字符串"
}
```

### 3.1.4 加密后的请求体签名（即 URL 中 checkStr 参数）

```
String dx_key = "AYUEJ1AWWHKTYENYUG1RP1B52SY5R1OJ"
String body = content.toString();       //JsonObject content = new JsonObject();
int len = body.length();
len = (len < 50 ? len : 50);
String str = body.substring(0, len) + dx_key;
MessageDigest messageDigest = MessageDigest.getInstance("MD5");
messageDigest.reset();
messageDigest.update(str.getBytes("UTF-8"));
String checkStr =String.valueOf(Hex.encodeHexString(messageDigest.digest()));
```

**注意**：签名得到的结果作为 URL 请求参数 checkStr，这个参数在提交 URL 时是必填项并且会用来校验请求体的的完整性。

### 3.1.5 POST 请求示例

**请求：**

```
curl  - XPOST 'http://api.dianhua.dianxin.net/dianhua_api/location/query/1.
0?appkey=7010470d62f39cbbfc0b77c0&auth_ver=2&  tk=dj2EoWq6eITWudAkkL9NHA%3D%3D&nonce=
1463389860123&s=6e847577295f31779ffcb594a30566e0&checkStr=ee4f44af7a9387c6da27a7cb7
7fedd25&location=true'
  - d '{"createTime":1463389860123,"msg":"2QctkhpsmOKiBQcomEFs1Db0Owl/4heMfoO
9r6B5yhQnzmQjM93fi6KpowoETMP"}'
```

**应答：**


```
{
    "response" : {
        "datas" : {
            "13838383838" : {
                "operation"  : "移动",
                "province"  : "河南",
                "city"  : "郑州"
            },
            "18610328820" : {
                "operation"  : "联通",
                "province"  : "北京",
                "city"  : "北京"
            }
        }
    },
    "responseHeader" : {
        "status"  : 200,
        "time " : 1466411072443,
        "version"  : "1.1.0"
    }
}
```


**注意**：上述两个样例中 URL 中参数信息是由程序加密生成，此处取出只是为了更清晰地呈现示例。