接口文档

**## 全局返回码（errcode）说明**返回码 | 说明

-2 | 服务器出错  
-1 | 服务器忙  
0 | 成功  
1 | 页面不存在，请求缺少必要参数  
2 | 用户名或密码验证失败  
-40001|APIMsgCryptCheck\_MissQueryArgument\_Error  
-40002|APIMsgCryptCheck\_MissBodyArgument\_Error  
-40003|APIMsgCryptCheck\_NonceLength\_Error  
-40004|APIMsgCryptCheck\_MessageSignature\_Error  
-40005|APIMsgCryptCheck\_NoValidatedAPIKey\_Error  
-40006|APIMsgCryptCheck\_ArgsType\_Error  
-40007|APIMsgCryptCheck\_RequestArgs\_Error  
-40008|APIMsgCryptCheck\_MsgDecryptOrSecretKey\_Error  
-40021|APIApplyCheck\_NoPermittedMethod

**API\_KEY：dMSHsWKy**

**SECRET\_KEY: nSemlZ2e7EVZAkROr5x44pMUkEbqKAlMIlJ2GdW0olrA2D4mwvRgLuZssy97AUByVYU**

**url : xiaosiapi.mindaxiaosi.com/api/wx\_card/wx\_signed**

|  |  |  |
| --- | --- | --- |
| 加密消息参数 | 是否必须 | 说明 |
| t\_uid | 是 | 工号 |
| passwd | 是 | 密码 |

返回值：

json

{

"errcode":0,

"errmsg":"ok",

"data":[

{

"name": "高等数学20班"

"students": [

{

"name": "张三",

"card\_number": "2016113012"

},

{

"name": "李四",

"card\_number": "2016113025"

},

....

]

},

]

}

**接口密钥 API\_KEY 及 SECRET\_KEY 说明**

API 请求数据 及 加密方式

|  |  |  |
| --- | --- | --- |
| URL携带参数 | 是否必须 | 说明 |
| nonce | 是 | 随机字符串(20-32位) |
| msg\_signature | 是 | 请求数据签名`（body + query） |

|  |  |  |
| --- | --- | --- |
| POST参数 | 是否必须 | 说明 |
| encrypt\_msg | 是 | 加密消息体 |
| api\_key | 是 | API\_KEY |

**示例：**

> API-KEY: AAAxxx

>SECRET\_KEY: JSZqvCElb16pqZc1l4tmdOrOiK27ppH6tO1oABhpWxbJoIH4PDMRKXjc8Omwea5UtQo

> NONCE: YOvmnLpUw1p3faRoQUg0qW689

> 请求参数: {"uid": "2014131201", "passwd": "208687"}

将所需数据名及所需数据列入Json中

例如：教务个人详细信息查询

```json

{

"uid": "2014131201",

"passwd": "208687"

}

```

**1. 生成长度为20-32位随机字符串**

**2. 加密消息体encrypt\_msg算法**

将请求所需json数据 结合 api key, secret key 和 nonce 进行签名 生成 encrypt\_msg

采用AES对称加密算法（AES/CBC/ZeroPadding 128位模式）为调用接口所需要的参数进行加密 生成 encrypt\_msg

AES密钥 | 值

--- | ----

KEY | `api\_key + nonce` 的MD5值的30, 28, 26, 24, ..., 0 位，共16位

IV | `nonce + api\_secret\_key` 的MD5值的30, 28, 26, 24, ..., 0 位，共16位

假设传送的参数如下：

```json

{

"uid": "2014131201",

"passwd": "208687"

}

```

> api\_key + nonce = AAAxxxYOvmnLpUw1p3faRoQUg0qW689

>nonce+api\_secret\_key= YOvmnLpUw1p3faRoQUg0qW689JSZqvCElb16pqZc1l4tmdOrOiK27ppH6tO1oABhpWxbJoIH4PDMRKXjc8Omwea5UtQo

得出的key和iv分别为7c5498820b94db25, 8a6a9c797c00101a

加密消息题为{"passwd": "208687", "uid": "2014131201"}

最后生成的结果encrypt\_msg将为b6836bead2c1e1d614aa4975f522826a2ad92f49c711ec8be6e76a35858f39bd191a09ed4ecd56c4b16f33d502ab664f

**\* 请注意,上方示例中json格式为分隔符后有空格，部分语言生成的json分隔符后没空格，均可正常访问API(无空格版{"passwd":"208687","uid":"2014131201"}加密结果为70556b90f8e66b9d6cfafa96f18c6b1c566feccd4735fff5efc0be7c34b91e71970cb464dce4a8a49dbe1d3b2446172e)**

**3. 请求数据签名msg\_signature算法**

nonce, encrypt\_msg, api\_key参数进行排序后求sha1的大写值：

**1. 将nonce, encrypt\_msg, api\_key参数进行字典序排序**

排序的好的List为：['AAAxxx', 'YOvmnLpUw1p3faRoQUg0qW689', 'b6836bead2c1e1d614aa4975f522826a2ad92f49c711ec8be6e76a35858f39bd191a09ed4ecd56c4b16f33d502ab664f']

**2. 将三个参数字符串拼接成一个字符串**

排序结果：AAAxxxYOvmnLpUw1p3faRoQUg0qW689b6836bead2c1e1d614aa4975f522826a2ad92f49c711ec8be6e76a35858f39bd191a09ed4ecd56c4b16f33d502ab664f

**3. 进行sha1加密后转换大写**

msg\_signature = 7A570136B811E5515B945AC291FC7A93147FDA58

**4. 调用API方法**

URL Params携带参数

nonce 为第一步生成的随机字符串

msg\_signature 为第三步生成的sha1签名结果

POST Body携带参数

encrypt\_msg 为第二步生成的消息加密体

api\_key 为访问接口的apikey

**示例：以post的形式请求：<http://api.mindaxiaosi.com/api/urp/usr\_dinfo>**

**python**

**import requests**

**requests.post(**

**url="http://api.mindaxiaosi.com/api/urp/usr\_dinfo",**

**parame={**

**'nonce': 'YOvmnLpUw1p3faRoQUg0qW689',**

**'msg\_signature': '7A570136B811E5515B945AC291FC7A93147FDA58',**

**},**

**data={**

**'encrypt\_msg': 'b6836bead2c1e1d614aa4975f522826a2ad92f49c711ec8be6e76a35858f39bd191a09ed4ecd56c4b16f33d502ab664f',**

**'api-key': 'AAAxxx'**

**}**

**)**

**返回值使用 AES对称加密算法（AES/CBC/ZeroPadding 128位模式）解密**

**加密 解密示例程序：**

**def get\_sha1(\*args):**

**try:**

**sort\_list = map(lambda x: str(x), args)**

**sort\_list.sort()**

**sha = sha1()**

**sha.update("".join(sort\_list))**

**return sha.hexdigest().upper()**

**except Exception, e:**

**return e**

**def api\_encrypt(api\_key, api\_secret\_key, nonce, raw):**

**key = md5(api\_key + nonce).hexdigest()[::-2]**

**iv = md5(nonce + api\_secret\_key).hexdigest()[::-2]**

**raw += (16 - len(raw) % 16) \* '\0'**

**generator = AES.new(key, AES.MODE\_CBC, IV=iv)**

**return binascii.b2a\_hex(generator.encrypt(raw))**

**def api\_decrypt(api\_key, api\_secret\_key, nonce, enc):**

**key = md5(api\_key + nonce).hexdigest()[::-2]**

**iv = md5(nonce + api\_secret\_key).hexdigest()[::-2]**

**enc = binascii.unhexlify(enc)**

**cipher = AES.new(key, AES.MODE\_CBC, iv)**

**return json.loads(cipher.decrypt(enc).decode('utf-8').rstrip("\0"))**