

# 其他接口

## 1. 别名接口

### 1.1 别名说明

个推使用clientid来标识每个独立的用户，但clientid不等于开发者应用上的用户名，如果希望将消息发给应用上指定用户名的用户，则需要将用户名指定一个用户别名。

为一个或者一批clientid用户定义一个用户别名，通过这个用户别名对一个或一批用户进行推送。目前一个别名最多允许绑定10个clientid。

别名规则说明：

- 1. 有效的别名组成：字母（区分大小写）、数字、下划线、汉字。
- 2. 任务备注名长度限制为 40 字节。（ UTF-8 ）
- 3. 一个别名最多允许绑定10个clientid。

### 1.2 对应接口

#### 1.2.1 bindAlias-单个clientid绑定别名

一个clientid只能绑定一个别名，若已绑定过别名的clientid再次绑定新别名，则认为与前一个别名自动解绑，绑定新别名。

函数说明：

```
bindAlias(APPID,ALIAS,CID)
```

参数说明：

参数名	类型	必需	默认值	参数描述
APPID	String	是	无	用户所属应用id
ALIAS	String	是	无	用户别名
CID	String	是	无	用户id(clientid)

代码实例

```

# -*- coding: utf-8 -*-

from igt_push import *
from igetui.template import *
from igetui.igt_message import *
from igetui.igt_target import *
import os

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'
ALIAS = '请输入别名'

#单个clientid绑定别名功能
def bindAlias():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    ret = push.bindAlias(APPID,ALIAS,CID)
    print ret

bindAlias()

```

## 1.2.2 bindAliasBatch-多个clientid绑定别名

允许将多个clientid和一个别名绑定，如用户使用多终端，则可将多终端对应的clientid绑定为一个别名，目前一个别名最多支持绑定10个clientid。

函数说明：

```
bindAliasBatch(APPID,targets)
```

参数说明：

参数名	类型	必需	默认值	参数描述
APPID	String	是	无	用户所属应用id
targets	List	是	无	用户列表

代码实例

```

# -*- coding: utf-8 -*-

from igt_push import *
from igetui.template import *
from igetui.igt_message import *
from igetui.igt_target import *
import os

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID = "请输入用户cid"
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'
ALIAS = '请输入别名'

def bindAliasBatch():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)

    target = Target()
    target.clientId = CID
    target.alias = ALIAS

    target1 = Target()
    target1.alias = ALIAS

    targets=[target]
    targets.append(target1)

    ret = push.bindAliasBatch(APPID,targets)
    print ret

bindAliasBatch()
#注：只要有一个cid绑定成功，返回结果就为true

```

### 1.2.3 queryClientId-根据别名获取clientId信息

函数说明：

```
queryClientId(APPID,ALIAS)
```

参数说明：

参数名	类型	必需	默认值	参数描述
APPID	String	是	无	用户所属应用id
ALIAS	String	是	无	用户别名

代码实例

```
# -*- coding: utf-8 -*-

from igt_push import *
from igetui.template import *
from igetui.igt_message import *
from igetui.igt_target import *
import os

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID = "请输入用户cid"
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'
ALIAS ='请输入别名'

def queryClientId():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    ret = push.queryClientId(APPID,ALIAS)
    print ret

queryClientId()
```

1.2.4 queryAlias-通过clientid获取别名信息

函数说明：

```
queryAlias(APPID,CID)
```

参数说明：

参数名	类型	必需	默认值	参数描述
APPID	String	是	无	用户所属应用id
CID	String	是	无	用户id(clientid)

代码实例

```
# -*- coding: utf-8 -*-

from igt_push import *
from igetui.template import *
from igetui.igt_message import *
from igetui.igt_target import *
import os

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID = "请输入用户cid"
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'
ALIAS ='请输入别名'

def queryAlias():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    dictz = push.queryAlias(APPID,CID)
    for key in dictz:
        print key+": "+dictz[key].decode("utf-8")

queryAlias()
```

1.2.5 unBindAlias-单个clientid和别名解绑

函数说明：

```
unBindAlias(APPID,ALIAS,CID)
```

参数说明：

参数名	类型	必需	默认值	参数描述
APPID	String	是	无	用户所属应用id
ALIAS	String	是	无	用户别名
CID	String	是	无	用户id(clientid)

代码实例

```

# -*- coding: utf-8 -*-

from igt_push import *
from igetui.template import *
from igetui.igt_message import *
from igetui.igt_target import *
import os

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID = "请输入用户id"
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'
ALIAS ='请输入别名'

def aliasUnBind():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    ret = push.unBindAlias(APPID,ALIAS,CID)
    print ret

aliasUnBind()

```

## 1.2.6 unBindAliasAll-绑定别名的所有clientid解绑

函数说明：

```
unBindAliasAll(APPID,ALIAS);
```

参数说明

参数名	类型	必需	默认值	参数描述
APPID	String	是	无	用户所属应用id
ALIAS	String	是	无	用户别名

代码实例：

```
# -*- coding: utf-8 -*-

from igt_push import *
from igetui.template import *
from igetui.igt_message import *
from igetui.igt_target import *
import os

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID = "请输入用户id"
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'
ALIAS = '请输入别名'

def unBindAliasAll():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    ret = push.unBindAliasAll(APPID,ALIAS)
    print ret

unBindAliasAll()
```

## 2. 标签

### 2.1 描述

tag即为用户标签，个推提供了服务端和客户端接口，允许app针对每个clientid设置标签。用户的喜好、习惯、性别、年龄段等信息，这些信息均可以做为用户分组的标签。

通过标签（tag）方式实现用户分组，将消息发给指定标签用户，更进一步筛选了用户，实现精细化运营。

### 2.2 应用场景

场景1：一个用户经常看电影，给该用户打一个“movie”标签，当有最新电影更新了，可给tag为movie的这一群用户推送消息。

场景2：音频播放器应用。对不同音乐类型喜好的人群推送不同类别的新音乐通知。

### 2.3 对应接口

#### 2.3.1 setClientTag-对指定用户设置tag属性

##### 函数说明

```
setClientTag(APPID,CID,tagList)
```

## 参数说明

参数名	类型	必需	默认值	参数描述
APPID	String	是	无	用户所属应用id
CID	String	是	无	目标用户id(clientid)
tagList	List	是	无	用户tag列表

注：此接口有频次控制，tag的长度、个数、总长度也有限制，申请修改请联系邮箱：kegf@getui.com。

## 代码实例

```
# -*- coding: utf-8 -*-
from array import array
__author__ = 'wei'

from igt_push import *

#采用"Python SDK 快速入门", "第二步 获取访问凭证"中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID = "请输入用户id"
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

# 根据clientid设置标签功能
def setTag():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    tagList = ['标签1', '标签2', '.....']
    print push.setClientTag(APPID, CID, tagList);

setTag()
```

## 2.3.2 getUserTags-获取指定用户的tag属性

### 函数说明

```
getUserTags(appId, cid)
```

### 参数说明



参数名	类型	必需	默认值	参数描述
appld	String	是	无	用户所属应用id
cid	String	是	无	目标用户id(clientid)

代码实例

```
# -*- coding: utf-8 -*-
from array import array
__author__ = 'wei'

from igt_push import *

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID = "请输入用户id"
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

# 根据clientid查询标签
def getUserTagsTest():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    dictz = push.getUserTags(APPID, CID)
    for key in dictz:
        print key + ":" + dictz[key].decode("utf-8")

getUserTagsTest()
```

### 3. 停止任务接口

#### 3.1 接口说明

接口名称	参数	类型	说明
stop	taskId(格式OSL-yyMM_XXXXXX)	String	根据taskId对正处于推送状态，或者未接收的消息停止下发

#### 3.2 代码实例

```

# -*- coding: utf-8 -*-
from array import array
__author__ = 'wei'

from igt_push import *
#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
MASTERSECRET = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

def stopTask():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    print push.stop("OSA-0226_50RYYPFmos9eQEHZrkAf27");
stopTask()

```

## 4. 查询用户当前状态

### 4.1 接口说明

接口名称	方法名	类型	说明
getClientIdStatus	APPID	String	设置推送的appid
	clientid	String	用户唯一标识符

### 4.2 代码实例

```

# -*- coding: utf-8 -*-

from igt_push import *
import os

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID = "请输入用户id"
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

def getUserStatus():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    print push.getClientIdStatus(APPID, CID)

getUserStatus()

```

## 5. Badge设置

### 5.1 描述

Badge即ios用户应用icon上显示的数字，该接口提供了三种设置badge的方式：

- 1.在原有badge上+N;
- 2.在原有badge上-N;
- 3.直接设置badge(数字，会覆盖原有的badge值)

### 5.2 接口函数说明

```
// 根据 DeviceToken 设置 Badge
def setBadgeForDeviceToken(self, badge, appid, deviceTokenList)
// 根据 clientid 设置 Badge
def setBadgeForCID(self, badge, appid, cidList)
```

### 5.3 参数说明

参数名	类型	必需	默认值	参数描述
badge	String	是	无	用户应用icon上显示的数字
appid	String	是	无	用户所属应用id
cidList	Array	是	无	目标用户clientid列表
deviceTokenList	List	是	无	iOS用户DeviceToken列表

### 5.4 代码实例-根据 deviceToken 设置 Badge

```

# -*- coding: utf-8 -*-
from igt_push import *
import os

APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID = ""
HOST = "http://sdk.open.api.igexin.com/apiex.htm"
Alias = '123'
Badge = '50'
TASKID = 'OSA-0903_bWHwhpFPEC7i5nZwHmc6d'
PN = '13550347892'
DEVICETOKEN1 = ""
DEVICETOKEN2 = ""

def setBadgeForDeviceTokenDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    deviceTokenList = [DEVICETOKEN1, DEVICETOKEN2]
    res = push.setBadgeForDeviceToken(Badge, APPID, deviceTokenList)
    print(res)
setBadgeForDeviceTokenDemo()

```

## 5.6 代码实例-根据 clientid 设置 Badge

```

# -*- coding: utf-8 -*-
from igt_push import *
import os

APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID1 = ""
CID2 = ""
HOST = "http://sdk.open.api.igexin.com/apiex.htm"
Alias = '123'
Badge = '50'
TASKID = 'OSA-0903_bWHwhpFPEC7i5nZwHmc6d'
PN = '13550347892'
DEVICETOKEN1 = ""
DEVICETOKEN2 = ""
GroupName = 'app推送'

def setBadgeForCIDDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    cidList = [CID1, CID2]
    res = push.setBadgeForCID(Badge, APPID, cidList)
    print(res)

setBadgeForCIDDemo()

```

## 6. 黑名单用户管理

---

### 6.1 描述

将指定cid列表中的用户加入/移除黑名单

### 6.2 接口函数说明

```

// 添加黑名单用户
def addCidListToBlk(self, appId, cidList)
// 移除黑名单用户
def restoreCidListFromBlk(self, appId, cidList)

```

### 6.3 参数说明

参数名	类型	必需	默认值	参数描述
appld	String	是	无	appld
cidList	list	是	无	该appld下的用户cid列表

## 6.4 代码实例-添加用户到黑名单

```
# -*- coding: utf-8 -*-
from igt_push import *

import os
APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID1 = ""
CID2 = ""
HOST = "http://sdk.open.api.igexin.com/apiex.htm"
def addCidListToBlkDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    cidList = [CID1, CID2]
    res = push.addCidListToBlk(APPID,cidList)
    print(res)
addCidListToBlkDemo()
```

## 6.6 代码实例-将用户从黑名单移除

```
# -*- coding: utf-8 -*-
from igt_push import *
import os
APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID1 = ""
CID2 = ""
HOST = "http://sdk.open.api.igexin.com/apiex.htm"
def restoreCidListFromBlkDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    cidList = [CID1, CID2]
    res = push.restoreCidListFromBlk(APPID,cidList)
    print(res)
restoreCidListFromBlkDemo()
```

# 7. 获取推送结果实例

## 7.1 接口说明

接口名称	参数	参数说明
getPushResult	taskId	任务唯一识别号(格式OSL-yyMM_XXXXXX)

## 7.2 返回值

回执	参数说明
	{taskId=OSA-0820_uQ7gevLuGS7Odz8FS2ZSB9, result=ok, msgTotal=59, msgProcess=0}
	taskId：表示任务ID
	msgTotal:表示有效可下发总数
	result：OK执行成功
	result：sign_error 表示校验失败
	msgProcess：收到消息回执总数

## 7.3 代码实例

```

# -*- coding: utf-8 -*-

from igt_push import *
import os

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
CID = "请输入用户id"
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

# 根据taskId返回推送结果
def getPushResultTest():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    #返回根据任务taskId返回数据
    print push.getPushResult("OSA-0304_LUNiKF8n6i8PXgvqARRUp8")
# 返回单日推送数据信息
# print push.queryAppPushDataByDate(APPID, "20150525")
# 返回单日用户结果信息
# print push.queryAppUserDataByDate(APPID, "20150525")

getPushResultTest()

```

## 8. 获取单日用户数据

### 8.1 描述

调用此接口可以获取某个应用单日的用户数据（用户数据包括：新增用户数，累计注册用户总数，在线峰值，日联网用户数）（目前只支持查询1天前的数据）

### 8.2 接口名称

```
def queryAppUserDataByDate(appId, date)
```

### 8.3 接口参数

字段	说明
appId	应用ID
date	查询的日期（格式：yyyyMMdd）



## 8.4 返回值

字段	上级	含义
result	-	成功：ok
data	-	查询数据对象
appld	data	请求的AppId
date	data	查询的日期（格式：yyyyMMdd）
newRegistCount	data	新注册用户数
registTotalCount	data	累计注册用户数
activeCount	data	活跃用户数
onlineCount	data	在线用户数

## 8.5 代码实例

```
# -*- coding: utf-8 -*-

from igt_push import *
import os

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

def queryAppUserDataByDateDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    res = push.queryAppUserDataByDate(APPID,"20150605")
    print(res)

queryAppUserDataByDateDemo()
```

# 9. 获取单日推送数据

## 9.1 描述

调用此接口可以获取某个应用单日的推送数据（推送数据包括：发送总数，在线发送数，接收数，展示数，点击数）（目前只支持查询1天前的数据）

## 9.2 接口名称：

```
def queryAppPushDataByDate(appId, date)
```

## 9.3 接口参数

字段	说明
appId	应用ID
date	查询的日期（格式：yyyyMMdd）

## 9.4 返回值

字段	上级	含义
result	-	成功：ok
data	-	查询数据对象
appId	data	请求的AppId
date	data	查询的日期（格式：yyyyMMdd）
sendCount	data	发送总数
sendOnlineCount	data	在线发送数
receiveCount	data	接收数
showCount	data	展示数
clickCount	data	点击数

## 9.5 代码实例

```
# -*- coding: utf-8 -*-

from igt_push import *
import os

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

def queryAppPushDataByDateDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    res = push.queryAppPushDataByDate(APPID, "20150605")
    print(res)

queryAppPushDataByDateDemo()
```

## 10. 获取24小时在线用户数

### 10.1 描述

通过接口查询当前时间一天内的在线数（十分钟一个点，一小时六个点）

### 10.2 接口

### 10.3 函数说明

```
def getLast24HoursOnlineUserStatistics(self, appId)
```

### 10.4 参数说明

参数名	类型	必需	默认值	参数描述
appId	String	是	无	appId

### 10.5返回值

字段	取值	说明
appId	String	用户appId
onlineStatics	Dictionary	24小时用户在线数统计

## 10.6 代码实例

```
# -*- coding: utf-8 -*-

from igt_push import *
import os

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

def getLast24HoursOnlineUserStatisticsDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    res = push.getLast24HoursOnlineUserStatistics(APPID)
    print(res)

getLast24HoursOnlineUserStatisticsDemo()
```

# 11. 通过标签获取用户总数

## 11.1 描述

调用此接口可以获取该标签下的用户总数

## 11.2 接口名称：

```
def getUserCountByTags(self, appId, tagList)
```

## 11.3 接口参数：

字段	类型	说明
appId	String	应用ID
tagList	list	标签列表

## 11.4 getUserCountByTags代码示例：

```
# -*- coding: utf-8 -*-

from igt_push import *
import os

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

def getUserCountByTagsDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    tagList = ["18-20", "5555"]
    res = push.getUserCountByTags(APPID, tagList)
    print(res)

getUserCountByTagsDemo()
```

## 12. 查询符合条件的用户数

### 12.1 描述

通过接口查询符合当前查询条件的用户数

### 12.2 函数说明

```
def queryUserCount(self, appId, conditions)
```

### 12.3 参数说明

参数名	类型	必需	默认值	参数描述
appId	String	是	无	appId
conditions	AppConditions	是	无	conditions

### 12.5 代码实例

```
def queryUserCountDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    cdt = AppConditions()
    phoneTypeList = ['ANDROID']
    provinceList = ['浙江省']
    tagList = ['发生大幅度三']
    #新增机型
    cdt.addCondition(AppConditions.PHONE_TYPE, phoneTypeList)
    #新增地区
    cdt.addCondition(AppConditions.REGION,provinceList)
    #新增tag
    cdt.addCondition(AppConditions.TAG,tagList)

    result = push.queryUserCount(APPID,cdt)
    print(result)
```

# 13. 大数据综合分析用户得到的标签:即用户画像

## 13.1 接口描述

接口名称	参数	类型	参数说明
getPersonaTags	APPID	String	应用的唯一id

## 13.2 代码实例

```

# -*- coding: utf-8 -*-

from igt_push import *
import os

#toList接口每个用户返回用户状态开关,true: 打开 false: 关闭
os.environ['gexin_pushList_needDetails'] = 'true'

#采用"Python SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置
APPKEY = ""
APPID = ""
MASTERSECRET = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

def getPersonaTagsDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    res = push.getPersonaTags(APPID)
    print(json.dumps(res).decode("unicode_escape").encode("utf-8"))
    for result in res["tags"]:
        print(result["desc"])

getPersonaTagsDemo()

```

## 14. 批量获取推送结果

### 14.1 描述

调用此接口批量查询推送数据，可查询消息有效可下发总数，消息回执总数，用户点击数结果。

### 14.2 函数说明

```
def getPushResultByTaskidList(self, taskIdList)
```

### 14.3 参数说明

参数名	类型	必需	默认值	参数描述
taskIdList	list	是	无	任务Id列表

### 14.4 返回值

```

{
  "result": "",
  "resultList":[
    {
      "taskId": "",
      "GT": { //个推下发报表
        "sent": "", //成功下发数
        "displayed": "", //展示数
        "clicked": "", //点击
        "feedback": "", //到达
        "result": "" //成功(ok)或错误信息
      },
      "APN": { //ios apns下发
        "sent": "", //下发
        "displayed": "", //apns展示
        "clicked": "", //点击
        "result": "" //成功(ok)或错误信息
      }
    }
  ]
}

```

## 14.5 代码实例

```

# -*- coding: utf-8 -*-

from igt_push import *
import os

APPKEY = ""
APPID = ""
MASTERSECRET = ""
TASKID = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

def getPushResultByTaskidListDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    taskIdList = [TASKID]
    res = push.getPushResultByTaskidList(taskIdList)
    print(res)

getPushResultByTaskidListDemo()

```