消息推送方式

本章介绍了Python3 API的推送方式实例，推送模板结合推送方式即可完成整套推送方案。

1. 对单个用户推送消息

1.1 接口说明

接口名称	支持推送类型	说明
pushMessageToSingle	透传（payload）、点击通知启动应用、点击通知打开网页等	对单个用户（clientid）推送

接口名称正常推送不需要传requestId，如果发生异常重试时将requestId传入，具体用法详见示例代码。

```
def pushMessageToSingle(self, message, target, requestId=None)
```

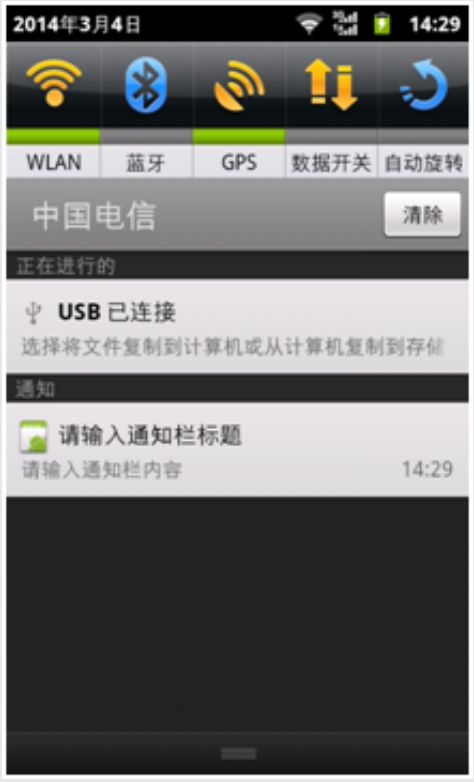
1.2 pushMessageToSingle代码实例

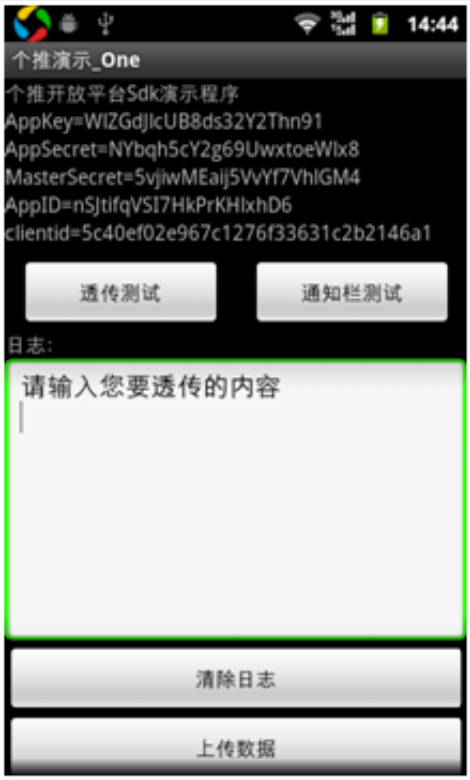
```
# -*- coding: utf-8 -*-
from igt_push import *
from igetui.template import *
from igetui.template.igt_base_template import *
from igetui.template.igt_transmission_template import *
from igetui.template.igt_link_template import *
from igetui.template.igt_notification_template import *
from igetui.template.igt_notypopload_template import *
from igetui.igt_message import *
from igetui.igt_target import *
from igetui.template import *

#采用"Python3 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 = ""
```

```
#别名推送方式
#ALIAS = "";
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

def pushMessageToSingle():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    #push = IGeTui("",APPKEY,MASTERSECRET)#此方式可通过获取服务端
地址列表判断最快域名后进行消息推送，每10分钟检查一次最快域名
    #消息模版：
    #TransmissionTemplate:透传功能模板，定义透传内容，应用启动形式
    template = TransmissionTemplateDemo()
    # 定义"SingleMessage"消息体，设置是否离线，离线有效时间，模板设置
    message = IGtSingleMessage()
    message.isOffline = True
    message.offlineExpireTime = 1000 * 3600 * 12
    message.data = template
    message.pushNetWorkType = 1#设置是否根据WIFI推送消息，2为
4G/3G/2G,1为wifi推送，0为不限制推送
    target = Target()
    target.appId = APPID
    target.clientId = CID
    #target.alias = ALIAS

    try:
        ret = push.pushMessageToSingle(message, target)
        print (ret)
    except RequestException as e:
        # 发生异常重新发送
        requstId = e.getRequestId()
        ret = push.pushMessageToSingle(message, target, requstId)
        print (ret)
#透传模板动作内容
def TransmissionTemplateDemo():
    template = TransmissionTemplate()
    template.transmissionType = 1
    template.appId = APPID
    template.appKey = APPKEY
    template.transmissionContent = '请输入您要透传内容'
#    iOS setAPNInfo
#    apnpayload = APNPayload()
#    apnpayload.badge = 4
#    apnpayload.sound = "sound"
#    apnpayload.addCustomMsg("payload", "payload")
# #    apnpayload.contentAvailable = 1
# #    apnpayload.category = "ACTIONABLE"
#
#    alertMsg = DictionaryAlertMsg()
```

```
#    alertMsg.body = 'body'
#    alertMsg.actionLocKey = 'actionLockey'
#    alertMsg.locKey = 'lockey'
#    alertMsg.locArgs=['locArgs']
#    alertMsg.launchImage = 'launchImage'
#    # iOS8.2以上版本支持
# #    alertMsg.title = 'Title'
# #    alertMsg.titleLocArgs = ['TitleLocArg']
# #    alertMsg.titleLocKey = 'TitleLocKey'
#    apnpayload.alertMsg=alertMsg
#    template.setApnInfo(apnpayload)

   #设置通知定时展示时间，结束时间与开始时间相差需大于6分钟（误差6分钟），消息推送后，客户端将在指定时间差内展示消息
   #begin = "2015-03-04 17:40:22";
   #end = "2015-03-04 17:47:24";
   #template.setDuration(begin, end)
   return template

pushMessageToSingle()
```

1.3 返回值

字段	返回码
result	请查询PushResult返回值
客户端展示	



2. 对指定用户列表推送消息

2.1 接口说明

如果仅对单个用户推送务必使用单推接口，否则会严重影响推送性能，如果对少量甚至几个用户推送同样的消息，建议使用单推实现，性能会更高

接口名称	支持推送类型	说明
pushMessageToList	透传（payload）、点击通知启动应用、点击通知打开网页等	（通过clientid列表）群推，可查看clientid列表中每个用户的在线状态

2.2 pushMessageToList代码实例

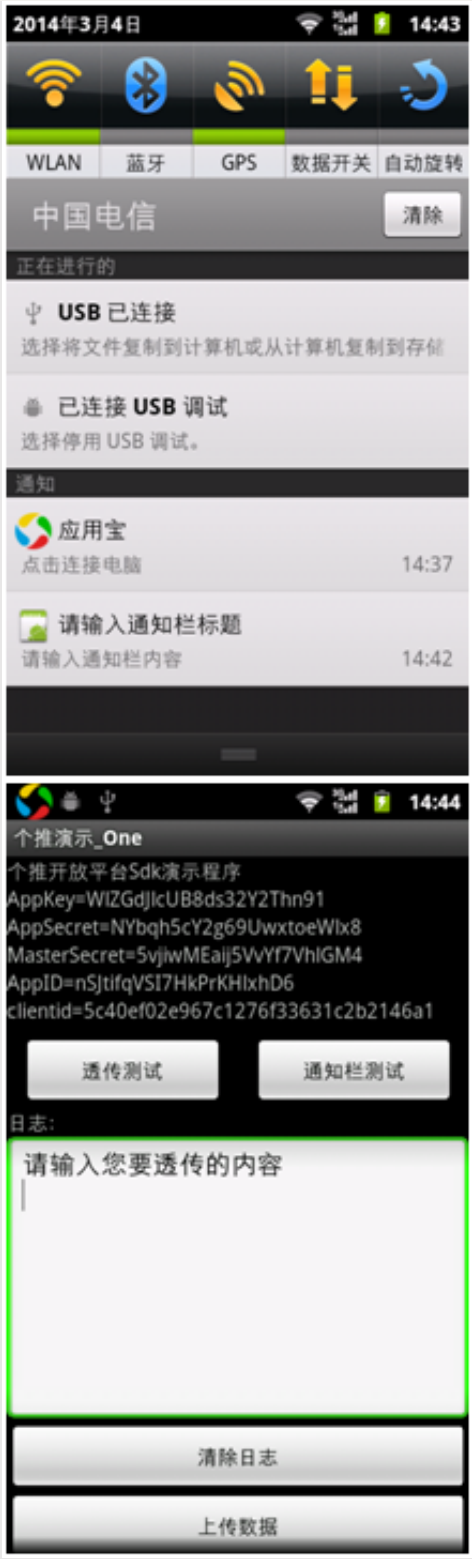
```
# -*- coding: utf-8 -*-
from igt_push import *
from igetui.template import *
from igetui.template.igt_base_template import *
from igetui.template.igt_transmission_template import *
from igetui.template.igt_link_template import *
from igetui.template.igt_notification_template import *
from igetui.template.igt_notypopload_template import *
```

```
from igetui.igt_message import *
from igetui.igt_target import *
from igetui.template 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'] = 'false'

#采用"Python3 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 = ""
CID1 = ""
CID2 = ""
#ALIAS1= ""
#ALIAS2= ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

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

    # 消息模版：
    # NotificationTemplate：通知透传功能模板
    template = TransmissionTemplateDemo()

    message = IGtListMessage()
    message.data = template
    message.isOffline = True
    message.offlineExpireTime = 1000 * 3600 * 12
    message.pushNetWorkType = 0

    target1 = Target()
    target1.appId = APPID
    target1.clientId = CID1
#   target1.alias = Alias1
    target2 = Target()
    target2.appId = APPID
    target2.clientId = CID2
#   target2.alias = Alias2
    arr = []

    arr.append(target1)
    arr.append(target2)
    contentId = push.getContentId(message, 'ToList_任务别名_可为空')
    ret = push.pushMessageToList(contentId, arr)
    print (ret)

# 通知透传模板动作内容
```

```
def TransmissionTemplateDemo():
    template = TransmissionTemplate()
    template.transmissionType = 1
    template.appId = APPID
    template.appKey = APPKEY
    template.transmissionContent = '请输入您要透传内容'
#   iOS setAPNInfo
#   apnpayload = APNPayload()
#   apnpayload.badge = 4
#   apnpayload.sound = "sound"
#   apnpayload.addCustomMsg("payload", "payload")
#   apnpayload.contentAvailable = 1
#   apnpayload.category = "ACTIONABLE"
#
#   alertMsg = DictionaryAlertMsg()
#   alertMsg.body = 'body'
#   alertMsg.actionLocKey = 'actionLockey'
#   alertMsg.locKey = 'lockey'
#   alertMsg.locArgs=['locArgs']
#   alertMsg.launchImage = 'launchImage'
#   iOS8.2以上版本支持
#   alertMsg.title = 'Title'
#   alertMsg.titleLocArgs = ['TitleLocArg']
#   alertMsg.titleLocKey = 'TitleLocKey'
#   apnpayload.alertMsg=alertMsg
#   template.setApnInfo(apnpayload)

    # 设置通知定时展示时间，结束时间与开始时间相差需大于6分钟，消息推送后，客户端将在指定时间差内展示消息（误差6分钟）
    #begin = "2015-03-04 17:40:22";
    #end = "2015-03-04 17:47:24";
    #template.setDuration(begin, end)
    return template
    print (ret)

pushMessageToList()
```

2.3 返回值

字段	返回码
result	请查询PushResult返回值
客户端展示	



注：此接口有频次控制，申请修改请联系邮箱：kegf@getui.com 。

3. 对指定应用群推消息

3.1 接口说明

接口名称	支持推送类型	说明

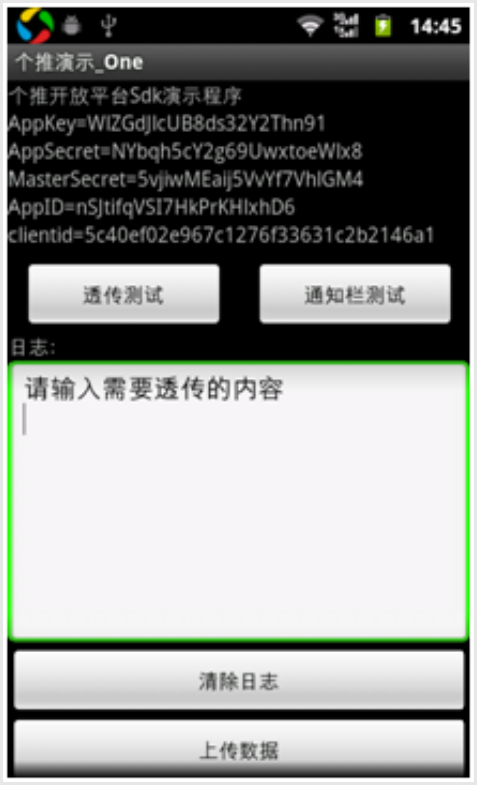
pushMessageToApp	透传（payload）、点击通知启动应用、点击通知打开网页等	（通过应用AppID）群推，给所有符合条件的客户端用户推送

3.2 pushMessageToApp代码实例

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

from igt_push import *
from igetui.template import *
from igetui.template.igt_base_template import *
from igetui.template.igt_transmission_template import *
from igetui.template.igt_link_template import *
from igetui.template.igt_notification_template import *
from igetui.template.igt_notypopload_template import *
from igetui.igt_message import *
from igetui.igt_target import *
from igetui.template import *

#采用"Python3 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'

def pushMessageToApp():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    #push = IGeTui("",APPKEY,MASTERSECRET)#此方式可通过获取服务端地址列表判断最快域名后进行消息推送，每10分钟检查一次最快域名
    #消息模版：
    #NotificationTemplate：通知透传功能模板

    template = TransmissionTemplateDemo()
    #定义"AppMessage"，设置是否离线，离线有效时间，推送模板，推送速度等
    message = IGtAppMessage()
    message.speed = 100 #设置消息推送速度，单位为条/秒，例如填写100，则为100条/秒。仅支持对指定应用群推接口。
    message.data = template
    message.pushNetWorkType = 1#设置是否根据WIFI推送消息，1为wifi推送，0为不限制推送
    message.isOffline = True
    message.offlineExpireTime = 1000 * 3600 * 12
```

```
    message.appIdList.extend([APPID])
    message.phoneTypeList.extend(["ANDROID", "IOS"])
    message.provinceList.extend(["浙江", "上海","北京"])
    message.tagList.extend(["开心"])
    ret = push.pushMessageToApp(message)
    print (ret)

#通知透传模板动作内容
def TransmissionTemplateDemo():
    template = TransmissionTemplate()
    template.transmissionType = 1
    template.appId = APPID
    template.appKey = APPKEY
    template.transmissionContent = '请输入您要透传内容'
#    iOS setAPNInfo
#    apnpayload = APNPayload()
#    apnpayload.badge = 4
#    apnpayload.sound = "sound"
#    apnpayload.addCustomMsg("payload", "payload")
#    apnpayload.contentAvailable = 1
#    apnpayload.category = "ACTIONABLE"
#
#    alertMsg = DictionaryAlertMsg()
#    alertMsg.body = 'body'
#    alertMsg.actionLocKey = 'actionLockey'
#    alertMsg.locKey = 'lockey'
#    alertMsg.locArgs=['locArgs']
#    alertMsg.launchImage = 'launchImage'
#    iOS8.2以上版本支持
#    alertMsg.title = 'Title'
#    alertMsg.titleLocArgs = ['TitleLocArg']
#    alertMsg.titleLocKey = 'TitleLocKey'
#    apnpayload.alertMsg=alertMsg
#    template.setApnInfo(apnpayload)

    # 设置通知定时展示时间，结束时间与开始时间相差需大于6分钟，消息推送后，客户端将在指定时间差内展示消息（误差6分钟）
    #begin = "2015-03-04 17:40:22";
    #end = "2015-03-04 17:47:24";
    #template.setDuration(begin, end)

    return template

pushMessageToApp()
```

3.3 返回值

字段	返回码
result	请查询PushResult返回值
客户端展示	

4. 推送结果返回值

正确返回	返回码	结果说明
	successed_online	用户在线，消息在线下发
	successed_offline	用户离线，消息存入离线系统
	Ok	发送成功
	details	返回用户状态的详细信息
	contentId	任务ID（当result值为ok时，有此字段）

错误返回	返回码	结果说明
	Error	请求信息填写有误
	action_error	未找到对应的action动作
	appkey_error	Appkey填写错误
	domain_error	填写的域名错误或者无法解析
	sign_error	Appkey与ClientId不匹配，鉴权失败

	AppidNoMatchAppKey	appid和鉴权的appkey不匹配
	PushMsgToListOrAppTimesOverLimit	群推次数超过最大值
	PushTotalNumOverLimit	推送个数总数超过最大值
	AppIdNoUsers	该AppId下的用户总数为0
	SendError	消息推送发送错误
	SynSendError	报文发送错误
	flow_exceeded	接口消息推送流量已超限
	TargetListIsNullOrSizeIs0	推送target列表为空
	PushTotalNumOverLimit	推送消息个数总数超限
	TokenMD5NoUsers	target列表没有有效的clientID
	NullMsgCommon	未找到contentId对应的任务
	TaskIdHasBeanCanceled	任务已经被取消
	AppidError	clientid绑定的appid与推送的appid不符
	successed_ignore	无效用户，消息丢弃
	TokenMD5Error	clientID填写有误
	SendError	消息发送错误
	AppidNoAppSecret	appid未找到对应的appSecret
	OtherError	未知错误，无法判定错误类型

注：此接口有频次控制，申请修改请联系邮箱：kegf@getui.com 。

5. 任务组名推送

5.1 描述

一个应用同时下发了n个推送任务，为了更好地跟踪这n个任务的推送效果，可以把他们组成一个任务组，在查询数据时，只需要输入该任务组名即可同时查到n个任务的数据结果。

5.2 应用场景

- 场景：做AB test，分别下发A组、B组推送任务，将A、B任务建成“任务组1”，查数据时，仅需要查找任务组1，即可以一起看到

A、B两组测试的结果，可以更直观地对比数据。

5.3 对应接口

命名同一个应用的不同taskid为同一个任务组名，任务组名由第三方填写。tolist（对指定用户列表推送消息）、toapp（对指定应用群推消息）接口支持该功能。

5.3.1 Tolist接口代码实例

```
def toListOfGroupName(host, appkey, mastersecret, message,
taskGroupName):
    push = IGeTui(host, appkey, mastersecret)
    contentId = push.getContentId(message, taskGroupName)
    target1 = Target()
    target1.appId = APPID
    target1.clientId = CID
    target1.alias = ""
    arr = []
    arr.append(target1)
    ret = push.pushMessageToList(contentId, arr)
    print (ret)
```

具体推送代码详见**pushMessageToList**代码实例

5.3.2 Toapp接口代码实例

```
def toListOfGroupName(host, appkey, mastersecret, message,
taskGroupName):
    push = IGeTui(host, appkey, mastersecret)
    push.pushMessageToApp(message, taskGroupName)
```

6. 定速推送

6.1 描述

定速推送旨在解决个推群推系统在全量推送时速度过快，导致部分客户服务器连接压力过大的问题。提供接口设置让用户按自身情况控制推送速度。

6.2 应用场景

全量推送时希望能控制推送速度不要太快，缓减服务器连接压力，可设置定速推送。如果未设置则按默认推送速度发送。

6.3 对应接口

在message中设置setSpeed为100，则全量送时个推控制下发速度在100条/秒左右。

只有toapp（对指定应用群推消息）支持定速推送。

```
# message.speed = 100
```

6.4 代码实例

```
def pushMessageToApp():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    template = LinkTemplateDemo()
    #个推信息体
    message = IGtAppMessage()
    message.data = template #设置推送消息类型
    message.speed = 100 #定速推送
```

以上示例代码仅部分展示，若要查询完整demo请看单个用户推送示例

7. 定时任务推送

7.1 定时对指定应用群推消息

描述

对单个指定应用的所有用户群发推送消息。该消息可以在用户设定的时间点进行推送。

注：此接口需要申请开通，申请邮箱：kegf@getui.com 。

推送接口

代码示例

```
# -*- coding: utf-8 -*-
#
from igt_push import *
from igetui.template import *
from igetui.template.igt_base_template import *
from igetui.template.igt_transmission_template import *
from igetui.template.igt_link_template import *
from igetui.template.igt_notification_template import *
from igetui.template.igt_notypopload_template import *
from igetui.igt_message import *
from igetui.igt_target import *
from igetui.template import *

#采用"Python3 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'
def pushMessageToApp():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)

    # 消息模版：
    # 1.TransmissionTemplate:透传功能模板
    # 2.LinkTemplate:通知打开链接功能模板
    # 3.NotificationTemplate：通知透传功能模板
    # 4.NotyPopLoadTemplate：通知弹框下载功能模板

    template = NotificationTemplateDemo()
    # template = LinkTemplateDemo()
    # template = TransmissionTemplateDemo()
    # template = NotyPopLoadTemplateDemo()

    message = IGtAppMessage()
    message.data = template
    message.isOffline = True
    message.offlineExpireTime = 1000 * 3600 * 12
    message.appIdList.extend([APPID])
    # 定时任务
    message.pushTime = "201809051950"

    conditions = AppConditions()
    phoneType = ['ANDROID']
    tags = ['5555', '']
    # conditions.addCondition('phoneType', phoneType, OptType.OR)
    # conditions.addCondition(AppConditions.TAG, tags, OptType.AND)
```

```
    message.setConditions(conditions)
    # message.phoneTypeList.extend(["ANDROID", "IOS"])
    # message.provinceList.extend(["浙江", "上海"])
    # message.tagList.extend(["开心"])
    # message.pushNetWorkType = 1
    # 控速推送(条/秒)
    # message.speed = 100
    # ret = push.getPushResultByGroupName(APPID,GroupName)
    ret = push.pushMessageToApp(message, 'toApp_任务别名_可为空')
    # print (message.getSpeed())
    print (ret)
# 通知链接模板动作内容
def LinkTemplateDemo():
    template = LinkTemplate()
    template.appId = APPID
    template.appKey = APPKEY
    template.title = "请填入通知标题"
    template.text = "请填入通知内容"
    template.logo = ""
    template.url = "http://www.baidu.com"
    template.transmissionType = 1
    template.transmissionContent = ''
    template.isRing = True
    template.isVibrate = True
    template.isClearable = True
    return template
?>
```

接口部分参数详细说明

- 设定推送的时间格式为yyyyMMddHHmm 例如:201710251900,任务将会在2017年10月25日17点00分推送。
- 对时间的设定有一定的要求:
    - 时间格式不正确 提交任务时 将直接返回失败。
    - 下发时间小于当前时间 提交任务时将直接返回失败。
    - 下发时间超过系统所允许的最大时间点 提交任务 将直接返回失败

7.2 定时任务查询接口

描述

该接口主要用来返回 已提交的定时任务的相关信息。

对应接口

```
def getScheduleTask(self, taskId, appId)
```

参数说明

参数名	类型	必需	默认值	参数描述
taskId	String	是	无	任务ID
appId	String	是	无	应用ID

代码示例

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

from igt_push import *
import os
APPKEY = ""
APPID = ""
MASTERSECRET = ""
TASKID = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'
def getScheduleTaskDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    res = push.getScheduleTask(TASKID,APPID)
    print (res)
```

返回的主要参数如下：

参数名称	参数含义
pushContent	推送类容（transmission的内容）
pushTime	推送时间
creatTime	任务创建时间
sendResult	任务状态

7.3 定时任务删除接口

描述

用来删除还未下发的任务

对应接口

```
def delScheduleTask(self, taskId, appId)
```

参数说明

参数名	类型	必需	默认值	参数描述
taskId	String	是	无	任务ID
appId	String	是	无	应用ID

代码示例

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

from igt_push import *
import os

APPKEY = ""
APPID = ""
MASTERSECRET = ""
TASKID = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'
def delScheduleTaskDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    res = push.delScheduleTask(TASKID,APPID)
    print (res)
```

需要补充说明的是:

- 距离下发还有一分钟的任务 将无法删除 也即 停止任务下发。

8. 应用群推条件交并补功能

8.1 描述

应用群推对于复杂的查询条件新增加的交并补功能，以对应查询语义中的与或非的关系

8.2 应用场景

- 场景：需要发送给城市在A,B,C里面，没有设置tagtest标签，手机型号为android的用户，用条件交并补功能可以实现，city(A|B|C) && !tag(tagtest) && phonetype(andriod)

8.3 对应接口

```
// AppConditions类的实例方法
AppConditions addCondition(self, key, values, optType = 0)
```

参数说明：

参数名	类型	必需	默认值	参数描述
key	str	是	无	查询条件键(phoneType 手机类型,region 省市,tag 用户标签)
values	list	是	无	查询条件值列表
optType	int	否	0	条件类型(OptType.or 或, OptType.and 与, OptType.not 非)

8.4 代码实例

```
# -*- coding: utf-8 -*-
from array import array
from igt_push import *
from igetui.template import *
from igetui.template.igt_base_template import *
from igetui.template.igt_transmission_template import *
from igetui.template.igt_link_template import *
from igetui.template.igt_notification_template import *
from igetui.template.igt_notypopload_template import *
from igetui.igt_message import *
from igetui.template import *
from igetui.utils.AppConditions import *

APPKEY = ""
```

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

def pushMessageToApp():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    template = NotificationTemplateDemo()

    message = IGtAppMessage()
    message.data = template
    message.isOffline = True
    message.offlineExpireTime = 1000 * 3600 * 12
    message.appIdList.extend([APPID])

    # conditions = AppConditions();
    # phoneType = ['ANDROID']
    # tags = ['dddd', 'ceshi2']
    # conditions.addCondition(AppConditions.PHONE_TYPE, phoneType,
OptType.OR)
    # conditions.addCondition(AppConditions.TAG, tags, OptType.AND)
    # message.setConditions(conditions)

    ret = push.pushMessageToApp(message, 'toApp_任务别名_可为空')
    # print (message.getSpeed())
    print (ret)

# 通知透传模板动作内容
def NotificationTemplateDemo():
    template = NotificationTemplate()
    template.appId = APPID
    template.appKey = APPKEY
    template.transmissionType = 1
    template.transmissionContent = "请填入透传内容"
    template.title = "请填入通知标题"
    template.text = "请填入通知内容"
    template.logo = "icon.png"
    template.logoURL = ""
    template.isRing = True
    template.isVibrate = True
    template.isClearable = True
    # iOS 推送需要的PushInfo字段 前三项必填，后四项可以填空字符串
    # template.setPushInfo(actionLocKey, badge, message, sound, payload,
locKey, locArgs, launchImage)
    # template.setPushInfo("open",4,"message","","","","","");
    # begin = "2015-03-04 17:40:22";
    # end = "2015-03-04 17:47:24";
    # template.setDuration(begin, end)
```

```
    return template
```

9. 批量单推功能

9.1 描述

用于一次创建提交多个单推任务。

9.2应用场景

当单推任务较多时，推荐使用该接口，可以减少与服务端的交互次数。

9.3 对应接口

函数说明：

```
def add(self, message, target)          # 追加单推消息
def submit(self)                        # 提交消息
def retry(self)                         # 重新提交
```

参数说明：

推送参数message和target与对单个用户推送消息使用的的参数相同

推送返回值：

批量单推每个单推消息的返回结果在submit返回值的info字段中，具体为加入时的序号和对应的返回结果。返回值详情请点击PushResult返回值

9.4 代码实例

```
# -*- coding: utf-8 -*-
from BatchImpl import *
from igetui.igt_target import *
from igetui.template.igt_notification_template import *
from igt_push import *

APPKEY = ""
```

```
APPID = ""
MASTERSECRET = ""
CID1 = ""
CID2 = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

def pushMessageToSingleBatch():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    batch = BatchImpl(APPKEY, push)

    # 消息模版：
    # 1.TransmissionTemplate:透传功能模板
    # 2.LinkTemplate:通知打开链接功能模板
    # 3.NotificationTemplate：通知透传功能模板
    # 4.NotyPopLoadTemplate：通知弹框下载功能模板

    templateNoti = NotificationTemplateDemo()
    templateLink = LinkTemplateDemo()
    # template = TransmissionTemplateDemo()
    # template = NotyPopLoadTemplateDemo()

    messageNoti = IGtSingleMessage()
    messageNoti.isOffline = True
    messageNoti.offlineExpireTime = 1000 * 3600 * 12
    messageNoti.data = templateNoti

    targetNoti = Target()
    targetNoti.appId = APPID
    targetNoti.clientId = CID1

    batch.add(messageNoti, targetNoti)

    messageLink = IGtSingleMessage()
    messageLink.isOffline = True
    messageLink.offlineExpireTime = 1000 * 3600 * 12
    messageLink.data = templateLink

    targetLink = Target()
    targetLink.appId = APPID
    targetLink.clientId = CID2

    batch.add(messageLink, targetLink)
        try:
            ret = batch.submit()
            print (ret)
        except Exception as e:
            ret = batch.retry()
```

```
        print (ret)


# 通知透传模板动作内容
def NotificationTemplateDemo():
    template = NotificationTemplate()
    template.appId = APPID
    template.appKey = APPKEY
    template.transmissionType = 1
    template.transmissionContent = "请填入透传内容"
    template.title = "请填入通知标题"
    template.text = "请填入通知内容"
    template.logo = "icon.png"
    template.logoURL = ""
    template.isRing = True
    template.isVibrate = True
    template.isClearable = True
    # begin = "2015-03-04 17:40:22";
    # end = "2015-03-04 17:47:24";
    # template.setDuration(begin, end)
    return template

# 通知链接模板动作内容
def LinkTemplateDemo():
    template = LinkTemplate()
    template.appId = APPID
    template.appKey = APPKEY
    template.title = "请填入通知标题"
    template.text = "请填入通知内容"
    template.logo = ""
    template.url = "http://www.baidu.com"
    template.transmissionType = 1
    template.transmissionContent = ''
    template.isRing = True
    template.isVibrate = True
    template.isClearable = True
    return template
```

10. 短信补量推送接口

10.1 说明

针对推送，在消息有效时间内，对未收到推送消息的用户进行短信补发，既提升消息触达又节省成本

10.2 接入流程

10.2.1 获取应用参数

进入个推开发者中心后，个推消息推送模块,注册app,获取 appId、appKey 和 masterSecret 参数（妥善保管）。

10.2.2 模板报备

进行短信补量推送前，必须在个推申请开通短信补量并报备短信模板。短信模板审核通过后，才能进行短信补量推送。短信模板报备审核的日期为1-2个工作日。

10.2.3 模板示例

模板ID	模板内容	相关参数解释
10000	示例【***】尊敬的*name*，你的余额还剩{money}，请尽快充值。	变量格式:${name}，。

10.2.4 pn绑定

进行短信补量推送前，需要调用接口绑定cid与pn。推送时获取pn，进行消息下发。其中文档中所说的pn 都是指md5之后的手机号。

10.3 短信补量补发逻辑

个推推送消息下发时，对于在线用户，直接通过sdk通道下发，对于离线用户消息将存储在离线空间，等到用户设定的短信补发时间，将开始对已绑定pn的用户进行短信补发。 对于在补发前，经过个推通道成功下发消息的用户，将不再进行短信补发。

10.4 cid与pn绑定接口

10.4.1 接口描述

绑定cid和pn的关系，用户短信补量根据cid查询pn下发短信。

10.4.2 功能代码示例

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

from igt_push import *
import os

APPKEY = ""
APPID = ""
MASTERSECRET = ""
TASKID = ""
PN = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'
def bindCidPnDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    cid = hashlib.md5(PN.encode('utf8')).hexdigest()
    params = {}
    params[CID] = cid
    res = push.bindCidPn(APPID,params)
    print (res)
```

10.4.3 具体参数

参数	类型	是否必填	最大长度	描述
appId	String	是	32	个推分配给开发者的ID
cidAndPn	list	是	50	key是cid value是pnmd5【32位小写】

10.4.4 返回值说明

参数	类型	是否必填	描述	示例值	其他
result	String	是	响应码	0	详细code码含义。参加文档最后【附录一】
batchRet	String	否	返回每个	[{"code":"0","cid":"xxxx"}]	

	cid的绑定 结果

10.4.5 注意事项

1）目前接口最多支持一次绑定50个cid与pn的对应关系。绑定后会返回每个cid的绑定结果。

2）同个应用下pn和cid是一对一的关系，否则原关系会被替换

10.5 cid与pn解绑接口

10.5.1 接口描述

用户通过接口传递cid列表, 可以批量解绑与之相对应的pn的关系。

10.5.2 功能代码示例

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

from igt_push import *
import os

APPKEY = ""
APPID = ""
MASTERSECRET = ""
TASKID = ""
PN = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'
def unbindCidPnDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    cids = [CID]
    res = push.unbindCidPn(APPID,cids)
    print (res)
```

10.5.3 具体参数

参数	类型	是否必填	最大长度	描述

appId	String	是	32	开发者ID
cids	list	是	50	客户端身份ID列表

10.5.4 返回值说明

参数	类型	是否必填	描述	示例值	其他
result	String	是	响应码	0	详细code码含义。参加文档最后【附录一】
batchRet	String	否	返回每个cid的绑定结果	[{"code":"0","cid":"xxxx"}]	

10.5.5 注意事项

1）目前接口最多支持一次解绑50个cid

10.6 pn查询接口

10.6.1 接口描述

用户通过接口传递cid列表, 可以查询与之相对的pn,接口会批量返回绑定关系。

10.6.2 用法举例

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

from igt_push import *
import os

APPKEY = ""
APPID = ""
```

```
MASTERSECRET = ""
TASKID = ""
PN = ""
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'
def queryCidPnDemo():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    cidList = [CID]
    res = push.queryCidPn(APPID,cidList)
    print (res)
```

10.6.3 具体参数

参数	类型	是否必填	最大长度	描述
appId	String	是	32	开发者ID
cidList	list	是	50	客户端身份ID列表

10.6.4 返回值说明

参数	类型	是否必填	描述	示例值	其他
result	String	是	响应码	0	详细code码含义。参加文档最后【附录一】
batchRet	String	否	返回每个cid的绑定结果	[{"code":"0","cid":"xxxx"}]	

10.6.5 注意事项

1）目前接口最多支持一次解绑50个cid

10.7.1 接口列表

接口定义	说明

pushMessageToApp	对应用的所有用户群发推送消息
pushMessageToSingle	向单个clientid或别名用户推送消息
pushMessageToList	上传clientid或别名列表，对列表中所有clientid或别名用户进行消息推送
注：详细的推送逻辑参考个推官网。这里只是补充短信推送的逻辑 参考地址：http://docs.getui.com/getui/server/php/push//	

10.7.2 支持短信模板的个推消息模板

消息模板	中文含义	是否支持短信模板	其他
Notification	通知模板	是	消息模板具体含义参考：http://docs.getui.com/getui/server/php/template/
LinkTemplate	网页模板	是	
NotyPopLoadTemplate	下载模板	是	
TransmissionTemplate	透传模板	是	

10.7.3 短信推送模板参数

成员名或方法名	类型	是否必填	描述	示例值	其他
smsTemplateId	String	是	推送的短信模板ID（短信模板说明）	smsMessage.smsTemplateId = "xxx"	具体使用，参考下面的代码中LinkTemplate中使用短信模板的示例。其他模板如此相同
smsContent	map	否	推送的短信模板中占位符的内容。	smsContent = {} smsContent["name"] = "zhangsan" smsContent["money"] = "0000" smsMessage.smsContent = smsContent	
offlineSendtime	long	是	推送后多久进行短信补发（单位：ms）	smsMessage.offlineSendtime = 1000	
isApplink	boolean	否	推送的短信模板中是否选用APPLink进行推送。	smsMessage.isApplink = True	
url	String	否	推送的短信模板中的APPLink链接地址。	smsMessage.url = "www.baidu.com"	
payload	String	否	推送的短信模板中的APPLink自定义字段。	smsMessage.payload = "自定义"	

10.7.4 pushMessageToApp 完整示例

```
# -*- coding: utf-8 -*-
#
from igt_push import *
from igetui.template import *
from igetui.template.igt_base_template import *
from igetui.template.igt_transmission_template import *
from igetui.template.igt_link_template import *
from igetui.template.igt_notification_template import *
from igetui.template.igt_notypopload_template import *
from igetui.igt_message import *
from igetui.igt_target import *
from igetui.template import *

#采用"Python3 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'

def pushMessageToApp():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    #push = IGeTui("",APPKEY,MASTERSECRET)#此方式可通过获取服务端地址列表判断最快域名后进行消息推送，每10分钟检查一次最快域名
    # 消息模版：
        # 1.TransmissionTemplate:透传功能模板
        # 2.LinkTemplate:通知打开链接功能模板
        # 3.NotificationTemplate：通知透传功能模板
        # 4.NotyPopLoadTemplate：通知弹框下载功能模板

        # template = NotificationTemplateDemo()
    template = LinkTemplateDemo()
        # template = TransmissionTemplateDemo()
        # template = NotyPopLoadTemplateDemo()
        # template = StartActivityTemplateDemo()
    #定义"AppMessage"，设置是否离线，离线有效时间，推送模板，推送速度等
    message = IGtAppMessage()
    message.setSpeed(100)#设置消息推送速度，单位为条/秒，例如填写100，则为100条/秒。仅支持对指定应用群推接口。
    message.data = template
    message.pushNetWorkType = 1#设置是否根据WIFI推送消息，1为wifi推送，0为不限制推送
    message.isOffline = True
    message.offlineExpireTime = 1000 * 3600 * 12
    message.appIdList.extend([APPID])
```

```
    message.phoneTypeList.extend(["ANDROID", "IOS"])
    message.provinceList.extend(["浙江", "上海","北京"])
    message.tagList.extend(["开心"])
    ret = push.pushMessageToApp(message)
    print (ret)

def LinkTemplateDemo():
    template = LinkTemplate()
    template.appId = APPID
    template.appKey = APPKEY
    template.title = "请填入通知标题"
    template.text = "请填入通知内容"
    template.logo = ""
    template.url = "http://www.baidu.com"
    template.transmissionType = 1
    template.transmissionContent = ''
    template.isRing = True
    template.isVibrate = True
    template.isClearable = True
    #短信消息类
    smsMessage = SmsMessage()
    #注意当使用AppLink时，smsContent不能传值url
    smsContent = {}
    smsContent["name"] = "zhangsan"
    smsContent["money"] = "0000"
    smsMessage.smsContent = smsContent
    #短信模板ID 需要在个推报备开通 才可使用
    smsMessage.smsTemplateId = ""
    #多久后进行离线补发的时间
    smsMessage.offlineSendtime = 1000
    #推送的短信模板中是否选用APPLink进行推送。
    smsMessage.isApplink = True
    #推送的短信模板中的APPLink链接地址。
    smsMessage.url = "www.bai"
    #推送的短信模板中的APPLink自定义字段。
    smsMessage.payload = "自定义"
    template.setSmsInfo(smsMessage)
    return template

pushMessageToApp()
```

10.7.5 pushMessageToSingle 完整示例

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

```
from igt_push import *
from igetui.template import *
from igetui.template.igt_base_template import *
from igetui.template.igt_transmission_template import *
from igetui.template.igt_link_template import *
from igetui.template.igt_notification_template import *
from igetui.template.igt_notypopload_template import *
from igetui.igt_message import *
from igetui.igt_target import *
from igetui.template import *

#采用"Python3 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 = ""
#别名推送方式
#ALIAS = "";
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'

def pushMessageToSingle():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)
    #push = IGeTui("",APPKEY,MASTERSECRET)#此方式可通过获取服务端地址列表判断最快域名后进行消息推送，每10分钟检查一次最快域名
    # 消息模版：
    # 1.TransmissionTemplate:透传功能模板
    # 2.LinkTemplate:通知打开链接功能模板
    # 3.NotificationTemplate：通知透传功能模板
    # 4.NotyPopLoadTemplate：通知弹框下载功能模板

    # template = NotificationTemplateDemo()
    template = LinkTemplateDemo()
    # template = TransmissionTemplateDemo()
    # template = NotyPopLoadTemplateDemo()
    # template = StartActivityTemplateDemo()
    # 定义"SingleMessage"消息体，设置是否离线，离线有效时间，模板设置
    message = IGtSingleMessage()
    message.isOffline = True
    message.offlineExpireTime = 1000 * 3600 * 12
    message.data = template
    message.pushNetWorkType = 1#设置是否根据WIFI推送消息，2为4G/3G/2G,1为wifi推送，0为不限制推送
    target = Target()
    target.appId = APPID
    target.clientId = CID
    #target.alias = ALIAS
```

```
    try:
        ret = push.pushMessageToSingle(message, target)
        print ret
    except RequestException as e:
        # 发生异常重新发送
        requstId = e.getRequestId()
        ret = push.pushMessageToSingle(message, target, requstId)
        print (ret)
def LinkTemplateDemo():
    template = LinkTemplate()
    template.appId = APPID
    template.appKey = APPKEY
    template.title = "请填入通知标题"
    template.text = "请填入通知内容"
    template.logo = ""
    template.url = "http://www.baidu.com"
    template.transmissionType = 1
    template.transmissionContent = ''
    template.isRing = True
    template.isVibrate = True
    template.isClearable = True

    #短信消息类
    smsMessage = SmsMessage()
    #注意当使用AppLink时，smsContent不能传值url
    smsContent = {}
    smsContent["name"] = "zhangsan"
    smsContent["money"] = "0000"
    smsMessage.smsContent = smsContent
    #短信模板ID 需要在个推报备开通 才可使用
    smsMessage.smsTemplateId = ""
    #多久后进行离线补发的时间
    smsMessage.offlineSendtime = 1000
    #推送的短信模板中是否选用APPLink进行推送。
    smsMessage.isApplink = True
    #推送的短信模板中的APPLink链接地址。
    smsMessage.url = "www.bai"
    #推送的短信模板中的APPLink自定义字段。
    smsMessage.payload = "自定义"
    template.setSmsInfo(smsMessage)
    return template

pushMessageToSingle()
```

10.7.6 pushMessageToList 完整示例

```
# -*- coding: utf-8 -*-
__author__ = 'wei'
from igt_push import *
from igetui.template import *
from igetui.template.igt_base_template import *
from igetui.template.igt_transmission_template import *
from igetui.template.igt_link_template import *
from igetui.template.igt_notification_template import *
from igetui.template.igt_notypopload_template import *
from igetui.igt_message import *
from igetui.igt_target import *
from igetui.template import *

#采用"Python3 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 = ""
#别名推送方式
#ALIAS = "";
HOST = 'http://sdk.open.api.igexin.com/apiex.htm'
def pushMessageToList():
    push = IGeTui(HOST, APPKEY, MASTERSECRET)

    # 消息模版：
    # 1.TransmissionTemplate:透传功能模板
    # 2.LinkTemplate:通知打开链接功能模板
    # 3.NotificationTemplate：通知透传功能模板
    # 4.NotyPopLoadTemplate：通知弹框下载功能模板

    # template = NotificationTemplateDemo()
    template = LinkTemplateDemo()
    # template = TransmissionTemplateDemo()
    # template = NotyPopLoadTemplateDemo()

    message = IGtListMessage()
    message.data = template
    message.isOffline = True
    message.offlineExpireTime = 1000 * 3600 * 12
    message.pushNetWorkType = 0

    target1 = Target()
    target1.appId = APPID
    # target1.clientId = ""
    target1.alias = "123"
    target2 = Target()
```

```
    target2.appId = APPID
    # target2.clientId = ""
    target2.alias = "456"
    arr = []

    arr.append(target1)
    arr.append(target2)
    contentId = push.getContentId(message, 'ToList_任务别名_可为空')
    ret = push.pushMessageToList(contentId, arr)
    print (ret)

def LinkTemplateDemo():
    template = LinkTemplate()
    template.appId = APPID
    template.appKey = APPKEY
    template.title = "请填入通知标题"
    template.text = "请填入通知内容"
    template.logo = ""
    template.url = "http://www.baidu.com"
    template.transmissionType = 1
    template.transmissionContent = ''
    template.isRing = True
    template.isVibrate = True
    template.isClearable = True

    #短信消息类
    smsMessage = SmsMessage()
    #注意当使用AppLink时，smsContent不能传值url
    smsContent = {}
    smsContent["name"] = "zhangc"
    smsContent["money"] = "0000"
    smsMessage.smsContent = smsContent
    #短信模板ID 需要在个推报备开通 才可使用
    smsMessage.smsTemplateId = ""
    #多久后进行离线补发的时间
    smsMessage.offlineSendtime = 1000
    #推送的短信模板中是否选用APPLink进行推送。
    smsMessage.isApplink = True
    #推送的短信模板中的APPLink链接地址。
    smsMessage.url = "http://www.baidu.com"
    #推送的短信模板中的APPLink自定义字段。
    smsMessage.payload = "自定义"
    template.setSmsInfo(smsMessage)
    return template

pushMessageToList()
```

10.7.7 返回值说明

参数	类型	是否必填	描述	示例值
result	String	是	响应码	result=ok
contentId	String	否	任务ID	contentId=xxxxxxx
smsResult	String	否	短信提交结果	smsResult=accept
smsResultMsg	String	否	短信提交失败结果	smsResultMsg=模板不存在

10.7.8 推送返回的完整结果示例

结果	示例	说明
短信补量提交成功	{result=ok, contentId=xxxxx, smsResult=accept}	消息推送成功，短信提交成功。
短信补量提交失败	{result=ok, smsResultMsg=模板不存在, contentId=xxxxxxx, smsResult=error}	消息推送成功，短信提交失败。

10.7.9 注意事项

1）短信参数填错，将不会进行短信补量，但是通过个推的sdk推送的消息会正常下发

2）短信补量的开始时间必须在个推通道的离线时间内，且不大于服务端限制值（目前设置是3天）。

3）短信补发的整个内容的总长度不能超过70字。

10.8 pn与cid操作的相关状态码

状态码	含义
0	成功
1	cid不存在
2	cid与appid不匹配
3	不能覆写低级别的PN绑定
4	应用下该PN已经绑定更活跃的cid
5	cid为空

6	pn为空
7	该cid没有绑定PN值
8	其他原因导致的失败

11. 推送结果返回值

具体返回值请查询下表

正确返回	返回码	结果说明
	successed_online	用户在线，消息在线下发
	successed_offline	用户离线，消息存入离线系统
	Ok	发送成功
	details	返回用户状态的详细信息
	contentId	任务ID（当result值为ok时，有此字段）

错误返回	返回码	结果说明
	Error	请求信息填写有误
	action_error	未找到对应的action动作
	appkey_error	Appkey填写错误
	domain_error	填写的域名错误或者无法解析
	sign_error	Appkey与ClientId不匹配，鉴权失败
	AppidNoMatchAppKey	appid和鉴权的appkey不匹配
	PushMsgToListOrAppTimesOverLimit	群推次数超过最大值
	PushTotalNumOverLimit	推送个数总数超过最大值
	AppIdNoUsers	该AppId下的用户总数为0
	SendError	消息推送发送错误
	SynSendError	报文发送错误
	flow_exceeded	接口消息推送流量已超限
	TargetListIsNullOrSizeIs0	推送target列表为空
	PushTotalNumOverLimit	推送消息个数总数超限
	TokenMD5NoUsers	target列表没有有效的clientID

	NullMsgCommon	未找到contentId对应的任务
	TaskIdHasBeanCanceled	任务已经被取消
	AppidError	clientid绑定的appid与推送的appid不符
	successed_ignore	无效用户，消息丢弃
	TokenMD5Error	clientID填写有误
	SendError	消息发送错误
	AppidNoAppSecret	appid未找到对应的appSecret
	OtherError	未知错误，无法判定错误类型