消息推送方式

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
pushMessageToSingleByReqId	与pushMessageToSingle支持类型相同	异常重试发送使用

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

```
static void pushByRequestIdDemo() {
                //准备数据
        Message msg = {0};
        msg.isOffline = true;//是否离线下发
        msg.offlineExpireTime = 1000*3600*2;//离线下发有效期 毫秒
        msg.pushNetWorkType = 0;//0不限 1wifi
        SingleMessage singleMsg = {0};
    singleMsg.msg = msg;

        //目标用户
    Target target = {0};
    target.appId = appId;
    target.clientId = cid;
        //target.alias = "test";
        IPushResult result = {0};

        NotyPopLoadTemplate tmpl= {0};
    NotyPopLoadTemplateDemo(& tmpl);
        char *reqId ="random request id";   //reqId每次推送应唯一
        try {
                result = pushMessageToSingleByReqId(appKey, &singleMsg, &tmpl, NotyPopLoad, &target, reqId);
        } catch (const exception &e) {
                result = pushMessageToSingleByReqId(appKey, &singleMsg, &tmpl, NotyPopLoad, &target, reqId);
        }
}
```

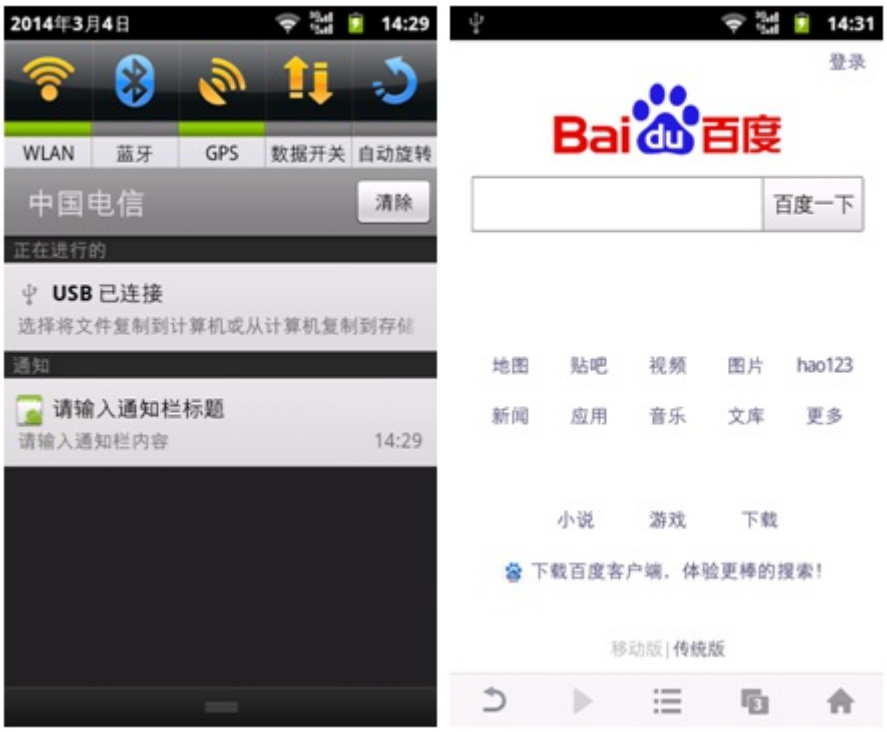
1.2 pushMessageToSingle代码实例

```
#include "IGtPush.h"
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <string>
#ifdef WIN32
#include <windows.h>
#endif

using namespace std;
static void printResult(IPushResult &result);
static void TransmissionTemplateDemo(TransmissionTemplate* templ)

// http的域名
//static char *host ="http://sdk.open.api.igexin.com/apiex.htm";

//https的域名
static char *host ="https://api.getui.com/apiex.htm";

//定义常量, appId、appKey、masterSecret 采用本文档 "第二步 获取访问凭证 "中获得的应用配置
static char *appId = "";
static char *appKey = "";
static char *masterSecret = "";
```

```
static char *cid = "";                          //要推送用户的cid
//static char *alias = "";          //要推送用户的别名
static char *dt="";                                     //iOS设备唯一标识号

void tosingletest();

void tosingletest(){

        //准备数据
    Message msg = {0};
    msg.isOffline = true;//是否离线下发
    msg.offlineExpireTime = 1000*3600*2;//离线下发有效期 毫秒
    msg.pushNetWorkType = 0;//0不限 1wifi 2:4G/3G/2G
    SingleMessage singleMsg = {0};
    singleMsg.msg = msg;

    //目标用户
    Target target = {0};
    target.appId = appId;
    target.clientId = cid;
    //别名推送方式
   // target.alias = alias;
    IPushResult result = {0};


    TransmissionTemplate tmpl= {0};
  TransmissionTemplateDemo(& tmpl);
    result = pushMessageToSingle(appKey, &singleMsg, &tmpl, Transmission, &target);

    printResult(result);
}

/*      透传消息模板    */
void TransmissionTemplateDemo(TransmissionTemplate* templ)
{
    templ->t.appId = appId;
    templ->t.appKey = appKey;
     //应用启动类型，1：强制应用启动 2：等待应用启动
    templ->transmissionType = 1;
    //透传内容
    templ->transmissionContent = "您输入的透传内容";

        //templ->t.duration_start="2015-07-10 18:00:00";
        //templ->t.duration_end="2015-07-10 19:00:00";
}

static void printResult(IPushResult &result) {
    cout << "print result:-------------" << endl;
    for (int i = 0; i < result.size; i++) {
        cout << result.entry[i].key << ": " << result.entry[i].value << endl;
    }
    cout << "print end:----------------" << endl;
}

int main(){
        // 注意：接口传入字符必须为UTF-8编码，因ASCII字符UTF－8编码后与原先一样，所以可以不编码，但中文等非ASCII字符必须编码
        // 如果返回的类似错误"post http data failed, code=6"，错误码可百度CURL返回的错误码是什么意思，如
http://www.cnblogs.com/wainiwann/p/3492939.html
        ////程序运行前只需初始化一遍，若已经初始化过即可不用，"编码"两个字为固定写法，不需要做转换
        Result r = pushInit(host, appKey, masterSecret, "编码");
        if(r!=SUCCESS){
                printf("pushInit for app failed: ret=%d\n", r);
                return -1;
        }
    tosingletest();

        return 0;
}
```

1.3 返回值

字段	返回码
result	请查询返回值

客户端展示	

2. 对指定列表用户推送消息

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

```
#include "IGtPush.h"
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <string>
#ifdef WIN32
#include <windows.h>
#endif

using namespace std;
static void printResult(IPushResult &result);
static void TransmissionTemplateDemo(TransmissionTemplate* templ)

// http的域名
//static char *host ="http://sdk.open.api.igexin.com/apiex.htm";

//https的域名
static char *host ="https://api.getui.com/apiex.htm";


//定义常量, appId、appKey、masterSecret 采用本文档 "第二步 获取访问凭证 "中获得的应用配置
static char *appId = "";
static char *appKey = "";
static char *masterSecret = "";
static char *cid1 = "";                          //要推送用户的cid1
```

```
static char *cid2 = "";                         //要推送用户的cid2
//static char *alias1 = "";          //要推送用户的别名alias1
//static char *alias2 = "";          //要推送用户的别名alias2


void tolisttest();

void tolisttest(){

        //准备数据
    Message msg = {0};
    msg.isOffline = true;//是否离线下发
    msg.offlineExpireTime = 1000*3600*2;//离线下发有效期 毫秒
    msg.pushNetWorkType = 0;//0不限 1wifi 2:4G/3G/2G
    ListMessage listMsg = {0};
    listMsg.msg = msg;


    TransmissionTemplate tmpl= {0};
    TransmissionTemplateDemo(& tmpl);
    Result ret;
    char contentId[50] = "";
    ret = getContentId(appKey, &listMsg, &tmpl, Transmission, contentId, sizeof(contentId));


    if (ret == SUCCESS) {
        Target* targetList = new Target[2];
        memset((void*)targetList, 0, sizeof(Target));
        targetList[0].appId = appId;
        targetList[0].clientId = cid1;
       // targetList[0].alias = alias;
        targetList[1].appId = appId;
        targetList[1].clientId = cid2;
       // targetList[1].alias = "bbb";
        PushDetail details[1] = {0};

        //cout << "============pushMessageToList============" << endl;
        IPushResult result = {0};
        result = pushMessageToList(appKey, contentId, targetList,1, details);
        cout<<details->cid<<","<<details->ret<<endl;
        printResult(result);

        ret = cancelContentId(appKey, contentId);
        cout << "cancelContentId ret=" << ret << endl;
    }
        }

}

/*      透传消息模板    */
void TransmissionTemplateDemo(TransmissionTemplate* templ)
{
    templ->t.appId = appId;
    templ->t.appKey = appKey;
     //应用启动类型，1：强制应用启动 2：等待应用启动
    templ->transmissionType = 1;
    //透传内容
    templ->transmissionContent = "您输入的透传内容";

        //templ->t.duration_start="2015-07-10 18:00:00";
        //templ->t.duration_end="2015-07-10 19:00:00";
}

static void printResult(IPushResult &result) {
    cout << "print result:-------------" << endl;
    for (int i = 0; i < result.size; i++) {
        cout << result.entry[i].key << ": " << result.entry[i].value << endl;
    }
    cout << "print end:----------------" << endl;
}

int main(){
        // 注意：接口传入字符必须为UTF-8编码，因ASCII字符UTF－8编码后与原先一样，所以可以不编码，但中文等非ASCII字符必须编码
        // 如果返回的类似错误"post http data failed, code=6"，错误码可百度CURL返回的错误码是什么意思，如
http://www.cnblogs.com/wainiwann/p/3492939.html
        //程序运行前只需初始化一遍，若已经初始化过即可不用，"编码"两个字为固定写法，不需要做转换
        Result r = pushInit(host, appKey, masterSecret, "编码");
```

```
        if(r!=SUCCESS){
                printf("pushInit for app failed: ret=%d\n", r);
                return -1;
        }
    tolisttest();

        return 0;
}
```

2.3 返回值

字段	返回码
result	请查询返回值
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

```
#include "IGtPush.h"
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <string>
#ifdef WIN32
#include <windows.h>
#endif
```

```
using namespace std;
static void printResult(IPushResult &result);
static void TransmissionTemplateDemo(TransmissionTemplate* templ)

// http的域名
//static char *host ="http://sdk.open.api.igexin.com/apiex.htm";

//https的域名
static char *host ="https://api.getui.com/apiex.htm";


//定义常量, appId、appKey、masterSecret 采用本文档 "第二步 获取访问凭证 "中获得的应用配置
static char *appId = "";
static char *appKey = "";
static char *masterSecret = "";

void toapptest();

void toapptest(){


     //准备数据
    Message msg = {0};
    msg.isOffline = 0;//是否离线下发
    msg.offlineExpireTime = 1000*3600*2;//离线下发有效期 毫秒
    msg.pushNetWorkType = 0;//0不限 1wifi 2:4G/3G/2G

    AppMessage appMsg = {0};
    appMsg.msg = msg;
    appMsg.speed = 10000;//定速
    char* appList[] = {appId};
    appMsg.appIdList = appList;
    appMsg.appIdListSize = 1;
    //通知接收者的手机操作系统类型
        char* phoneTypeList[] = {"ANDRIOD","IOS"};
        appMsg.phoneTypeList = phoneTypeList;

        char* provinceList[] = {"浙江","上海","北京"};
        appMsg.provinceList = provinceList;

        char* tagList[] = {"happy"};
        appMsg.tagList = tagList;

    IPushResult result = {0};
    //透传模板
    TransmissionTemplate tmpl= {0};
    TransmissionTemplateDemo(&tmpl);
    result = pushMessageToApp(appKey, &appMsg, &tmpl, Transmission);
    //打印结果
    printResult(result);
}

/*      透传消息模板    */
void TransmissionTemplateDemo(TransmissionTemplate* templ)
{
    templ->t.appId = appId;
    templ->t.appKey = appKey;
     //应用启动类型，1：强制应用启动 2：等待应用启动
    templ->transmissionType = 1;
    //透传内容
    templ->transmissionContent = "您输入的透传内容";

        //templ->t.duration_start="2015-07-10 18:00:00";
        //templ->t.duration_end="2015-07-10 19:00:00";
}

static void printResult(IPushResult &result) {
    cout << "print result:-------------" << endl;
    for (int i = 0; i < result.size; i++) {
        cout << result.entry[i].key << ": " << result.entry[i].value << endl;
    }
    cout << "print end:----------------" << endl;
}

int main(){
        // 注意：接口传入字符必须为UTF-8编码，因ASCII字符UTF－8编码后与原先一样，所以可以不编码，但中文等非ASCII字符必须编码
```

```
        // 如果返回的类似错误"post http data failed, code=6"，错误码可百度CURL返回的错误码是什么意思，如
http://www.cnblogs.com/wainiwann/p/3492939.html
        Result r = pushInit(host, appKey, masterSecret, "编码");//"编码"两个字为固定写法，不需要做转换
        if(r!=SUCCESS){
                printf("pushInit for app failed: ret=%d\n", r);
                return -1;
        }
    toapptest();

        return 0;
}
```

3.3 返回值

字段	返回码
result	请查询返回值
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 。

4. 任务组名推送

4.1 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。

4.2 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。

4.3 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。

4.3.1 Tolist接口代码实例

```
void toListGroupName(char *host, char *appkey, char *mastersecret, ListMessage *message, void *templ,
        TemplateType templateType, char *contentId, char *taskGroupname, Target * targetList, PushDetail *details) {
        Result r = pushInit(host, appKey, masterSecret, "编码");
    if(r!=SUCCESS){
        printf("pushInit for app failed: ret=%d\n", r);
        return;
    }
        getContentIdByGroup(appKey, message, templ, Transmission, contentId, sizeof(contentId), taskGroupname);
        pushMessageToList(appkey, contentId, targetList, sizeof(targetList), details);
}
```

4.3.2 ToApp接口代码实例

```
void toAppGroupName(char *host, char *appkey, char *mastersecret, AppMessage *message, void *templ,
        TemplateType templateType, char *taskGroupname) {
        Result r = pushInit(host, appKey, masterSecret, "编码");
    if(r!=SUCCESS){
        printf("pushInit for app failed: ret=%d\n", r);
        return;
    }
        pushMessageToAppByGroupName(appKey, message, templ, templateType, taskGroupname);
}
```

5. 获取用户状态

5.1 描述

调用此接口可获取用户状态，如在线不在线，cid和appid是否对应，appkey是否正确等。

5.2函数说明：

```
void getClientIdStatus(appId, cid);
```

5.3 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

5.4 返回值

IpushResult具体返回值详情查询，请点击IpushResult接口返回值

6. PushResult返回值查询

6.1 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