消息推送方式

1. 对单个用户推送消息

1.1 描述

向单个clientid或别名用户推送消息。

注：个推使用clientid来标识每个独立的用户，每一台终端上每一个app拥有一个独立的clientid。

1.2 应用场景

- 场景1：某用户发生了一笔交易，银行及时下发一条推送消息给该用户。
- 场景2：用户定制了某本书的预订更新，当本书有更新时，需要向该用户及时下发一条更新提醒信息。
  这些需要向指定某个用户推送消息的场景，即需要使用对单个用户推送消息的接口。

1.3 对应接口（**pushMessageToSingle**）

函数说明：

接口定义	说明
IPushResult pushMessageToSingle(SingleMessage message, Target target)	正常发送使用
IPushResult pushMessageToSingle(SingleMessage message, Target target, String requestId)	异常重试发送使用

参数说明：

参数名	类型	必需	默认值	参数描述
message	SingleMessage	是	无	消息体
target	Target	是	无	推送目标
requestId	String	否	无	用于重试发送使用

推送返回值：

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

代码实例：

```java
import com.gexin.rp.sdk.base.IPushResult;
import com.gexin.rp.sdk.base.impl.SingleMessage;
import com.gexin.rp.sdk.base.impl.Target;
import com.gexin.rp.sdk.exceptions.RequestException;
import com.gexin.rp.sdk.http.IGtPush;
import com.gexin.rp.sdk.template.LinkTemplate;

public class PushtoSingle {
    //采用"Java SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置，用户可以自行替换
    private static String appId = "";
    private static String appKey = "";
    private static String masterSecret = "";

    static String CID = "";
  //别名推送方式
   // static String Alias = "";
    static String host = "http://sdk.open.api.igexin.com/apiex.htm";

    public static void main(String[] args) throws Exception {
        IGtPush push = new IGtPush(host, appKey, masterSecret);
        LinkTemplate template = linkTemplateDemo();
        SingleMessage message = new SingleMessage();
        message.setOffline(true);
        // 离线有效时间，单位为毫秒，可选
        message.setOfflineExpireTime(24 * 3600 * 1000);
        message.setData(template);
        // 可选，1为wifi，0为不限制网络环境。根据手机处于的网络情况，决定是否下发
        message.setPushNetWorkType(0);
        Target target = new Target();
        target.setAppId(appId);
        target.setClientId(CID);
        //target.setAlias(Alias);
        IPushResult ret = null;
        try {
            ret = push.pushMessageToSingle(message, target);
        } catch (RequestException e) {
            e.printStackTrace();
            ret = push.pushMessageToSingle(message, target, e.getRequestId());
        }
        if (ret != null) {
            System.out.println(ret.getResponse().toString());
        } else {
            System.out.println("服务器响应异常");
        }
    }
    public static LinkTemplate linkTemplateDemo() {
        LinkTemplate template = new LinkTemplate();
        // 设置APPID与APPKEY
        template.setAppId(appId);
        template.setAppkey(appKey);

        Style0 style = new Style0();
        // 设置通知栏标题与内容
```

```
        style.setTitle("请输入通知栏标题");
        style.setText("请输入通知栏内容");
        // 配置通知栏图标
        style.setLogo("icon.png");
        // 配置通知栏网络图标
        style.setLogoUrl("");
        // 设置通知是否响铃，震动，或者可清除
        style.setRing(true);
        style.setVibrate(true);
        style.setClearable(true);
        template.setStyle(style);

        // 设置打开的网址地址
        template.setUrl("http://www.baidu.com");
        return template;
    }
}
```

2. 对指定列表用户推送消息

2.1 描述

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，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。

注：个推使用clientid来标识每个独立的用户，每一台终端上每一个app拥有一个独立的clientid。

2.2 应用场景

- 场景1，对于抽奖活动的应用，需要对已知的某些用户推送中奖消息，就可以通过clientid列表方式推送消息。
- 场景2，向新客用户发放抵用券，提升新客的转化率，就可以事先提取新客列表，将消息指定发送给这部分指定CID用户。

2.3 对应接口

2.3.1 getContentId-获取taskId接口

函数说明**(getContentId)**：

接口定义	
String getContentId(message)	
String getContentId(message,taskGroupName)	

参数说明：

参数名	类型	必需	默认值	参数描述
Message	ListMessage	是	无	消息体
taskGroupName	string	否	无	任务组名

注：此接口有频次控制，申请修改请联系邮箱：kegf@getui.com 。

2.3.2 pushMessageToList-对指定用户列表推送消息

函数说明**(pushMessageToList)**：

接口定义	
pushMessageToList(contentID ,targetList)	

参数说明：

参数名	类型	必需	默认值	参数描述
contentID	string	是	无	就是taskId，任务ID(格式OSL-yyMM_XXXXXX)
targetList	List	是	无	推送目标列表

返回值

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

注：此接口有频次控制，申请修改请联系邮箱：kegf@getui.com 。

代码实例

```java
import java.util.ArrayList;
import java.util.List;
import com.gexin.rp.sdk.base.IPushResult;
import com.gexin.rp.sdk.base.impl.ListMessage;
import com.gexin.rp.sdk.base.impl.Target;
import com.gexin.rp.sdk.http.IGtPush;
import com.gexin.rp.sdk.template.NotificationTemplate;
public class PushList {
    //采用"Java SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置，用户可以自行替换
    private static String appId = "";
    private static String appKey = "";
    private static String masterSecret = "";

    static String CID1 = "";
    static String CID2 = "";
    //别名推送方式
    // static String Alias1 = "";
    // static String Alias2 = "";
    static String host = "http://sdk.open.api.igexin.com/apiex.htm";
    public static void main(String[] args) throws Exception {
        // 配置返回每个用户返回用户状态，可选
        System.setProperty("gexin_pushList_needDetails", "true");
    // 配置返回每个别名及其对应cid的用户状态，可选
    // System.setProperty("gexin_pushList_needAliasDetails", "true");
        IGtPush push = new IGtPush(host, appKey, masterSecret);
        // 通知透传模板
        NotificationTemplate template = notificationTemplateDemo();
        ListMessage message = new ListMessage();
        message.setData(template);
        // 设置消息离线，并设置离线时间
        message.setOffline(true);
        // 离线有效时间，单位为毫秒，可选
        message.setOfflineExpireTime(24 * 1000 * 3600);
        // 配置推送目标
        List targets = new ArrayList();
        Target target1 = new Target();
        Target target2 = new Target();
        target1.setAppId(appId);
        target1.setClientId(CID1);
//      target1.setAlias(Alias1);
        target2.setAppId(appId);
        target2.setClientId(CID2);
//      target2.setAlias(Alias2);
        targets.add(target1);
        targets.add(target2);
        // taskId用于在推送时去查找对应的message
        String taskId = push.getContentId(message);
        IPushResult ret = push.pushMessageToList(taskId, targets);
        System.out.println(ret.getResponse().toString());
    }
    public static NotificationTemplate notificationTemplateDemo() {
        NotificationTemplate template = new NotificationTemplate();

        // 设置APPID与APPKEY
```

```
        template.setAppId(appId);
        template.setAppkey(appKey);

        Style0 style = new Style0();
        // 设置通知栏标题与内容
        style.setTitle("请输入通知栏标题");
        style.setText("请输入通知栏内容");
        // 配置通知栏图标
        style.setLogo("icon.png");
        // 配置通知栏网络图标
        style.setLogoUrl("");
        // 设置通知是否响铃，震动，或者可清除
        style.setRing(true);
        style.setVibrate(true);
        style.setClearable(true);
        template.setStyle(style);

        // 透传消息设置，1为强制启动应用，客户端接收到消息后就会立即启动应用；2为等待应用启动
        template.setTransmissionType(2);
        template.setTransmissionContent("请输入您要透传的内容");
        return template;
    }
}
```

2.3.3 CancelContentId-取消taskid接口

taskid被标识为无效

函数说明：

接口定义
boolean CancelContentId(String taskid)

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(格式OSL-yyMM_XXXXXX)

代码实例:

```
import com.gexin.rp.sdk.http.IGtPush;
public class CancelContentId {
    //采用"Java SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置，用户可以自行替换
    static String appId = "";
    static String appkey = "";
    static String master = "";
    static String host = "http://sdk.open.api.igexin.com/apiex.htm";
    //taskid就是ContentId
    static String taskid = "";

    public static void main(String[] args) {
        IGtPush push = new IGtPush(host, appkey, master);
        boolean result = push.cancelContentId(taskid);
        System.out.println(result);
    }
}
```

3. 对指定应用群推消息

3.1 描述

对单个或多个指定应用的所有用户群发推送消息。

注：个推使用AppID来标识每个独立的应用。

3.2 应用场景

- 场景1，某app周年庆，群发消息给该app的所有用户，提醒用户参加周年庆活动。

3.3 接口(pushMessageToApp-对指定应用群推消息)

函数说明

接口定义	
IPushResult pushMessageToApp(message)	
IPushResult pushMessageToApp(message ， taskGroupName)	

参数说明

参数名	类型	必需	默认值	参数描述
Message	AppMessage	是	无	消息体
taskGroupName	String	否	无	任务别名

返回值

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

注：此接口有频次控制，申请修改请联系邮箱：kegf@getui.com 。

代码实例

```java
package com.getui.java.pushmessage;

import java.util.ArrayList;
import java.util.List;
import com.gexin.rp.sdk.base.IPushResult;
import com.gexin.rp.sdk.base.impl.AppMessage;
import com.gexin.rp.sdk.base.uitls.AppConditions;
import com.gexin.rp.sdk.http.IGtPush;
import com.gexin.rp.sdk.template.LinkTemplate;

public class PushtoAPP {
    //采用"Java SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置，用户可以自行替换
    private static String appId = "";
    private static String appKey = "";
    private static String masterSecret = "";
    static String host = "http://sdk.open.api.igexin.com/apiex.htm";

    public static void main(String[] args) throws Exception {

        IGtPush push = new IGtPush(host, appKey, masterSecret);

        LinkTemplate template = linkTemplateDemo();
        AppMessage message = new AppMessage();
        message.setData(template);

        message.setOffline(true);
        //离线有效时间，单位为毫秒，可选
        message.setOfflineExpireTime(24 * 1000 * 3600);
        //推送给App的目标用户需要满足的条件
        AppConditions cdt = new AppConditions();
        List<String> appIdList = new ArrayList<String>();
        appIdList.add(appId);
        message.setAppIdList(appIdList);
        //手机类型
        List<String> phoneTypeList = new ArrayList<String>();
        //省份
        List<String> provinceList = new ArrayList<String>();
        //自定义tag
        List<String> tagList = new ArrayList<String>();

        cdt.addCondition(AppConditions.PHONE_TYPE, phoneTypeList);
        cdt.addCondition(AppConditions.REGION, provinceList);
        cdt.addCondition(AppConditions.TAG,tagList);
        message.setConditions(cdt);

        IPushResult ret = push.pushMessageToApp(message,"任务别名_toApp");
        System.out.println(ret.getResponse().toString());
    }

    public static LinkTemplate linkTemplateDemo() throws Exception {
        LinkTemplate template = new LinkTemplate();
        template.setAppId(appId);

        template.setAppkey(appKey);
```

```
        Style0 style = new Style0();
        // 设置通知栏标题与内容
        style.setTitle("请输入通知栏标题");
        style.setText("请输入通知栏内容");
        // 配置通知栏图标
        style.setLogo("icon.png");
        // 配置通知栏网络图标
        style.setLogoUrl("");
        // 设置通知是否响铃，震动，或者可清除
        style.setRing(true);
        style.setVibrate(true);
        style.setClearable(true);
        template.setStyle(style);

        template.setUrl("http://www.baidu.com");

        return template;
    }

}
```

按城市接口推送

城市级别的推送基于个推原先的按省份推送的功能之上，使用同一个方法，这样可以减少开发者使用过程中的代码变更。省略具体的推送代码，下面展示相关变动的代码。

原来的省份推送是这样使用的：

```
AppConditions cdt = new AppConditions();
List<String> provinceList = new ArrayList<String>();
provinceList.add("浙江");
provinceList.add("上海");
provinceList.add("北京");
cdt.addCondition(AppConditions.REGION, provinceList);
```

现在按城市推送的使用方法如下：

```
AppConditions cdt = new AppConditions();
List<String> provinceList = new ArrayList<String>();
provinceList.add("浙江");
provinceList.add("上海");
provinceList.add("北京");
provinceList.add("33010000");//杭州市
provinceList.add("51010000");//成都市
cdt.addCondition(AppConditions.REGION, provinceList);
```

只需要在原有的列表里加入城市编号即可。

注1：编号对应表如region_code.data文件。

注2：列表里的城市仅支持编号表示，建议省份也用编号表示。为了兼容老用户，省份列表里的汉字仍能继续使用。

4. Wi-Fi推送

4.1 描述

仅在wifi条件下才展示通知内容，充分帮用户节省流量。

4.2 应用场景

- 主要用于富媒体、视频、应用下载等推送，仅在wifi环境下展现推送消息，用较精美的富文本内容展示通知，充分帮用户节省流量。

4.3 对应接口

在message中设置setPushNetWorkType为1，推送时只有通过wifi登录在线的用户才收到消息，手机网络登录用户的消息进离线，等该用户wifi登录后才获取该条离线消息。

Tosingle（对单个用户推送消息）、tolist（对指定用户列表推送消息）、toapp（对指定应用群推消息）三个接口都支持wifi推送。

```
//1: wifi , 0: 不限，默认不限
message.setPushNetWorkType(1);
```

4.4 代码实例

```
package Push_Message_Demo;

import com.gexin.rp.sdk.base.impl.SingleMessage;
import com.gexin.rp.sdk.http.IGtPush;

public class PushToSingle {
   //采用"Java SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置，用户可以自行替换
   static String appId = "";
   static String appkey = "";
   static String mastersecret = "";
   //请输入你的cid
   static String CID = "";
   static String host = "http://sdk.open.api.igexin.com/apiex.htm";

   public static void main(String[] args) throws Exception {
      IGtPush push = new IGtPush(host, appkey, mastersecret);

      SingleMessage message = new SingleMessage();
      //判断客户端是否wifi环境下推送。1为仅在wifi环境下推送，0为不限制网络环境。
      message.setPushNetWorkType(1);

   }
}
```

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

5. 定速推送

5.1 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。

5.2 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。

5.3 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.setSpeed(100);
```

5.4 代码实例

```
package Push_Message_Demo;

import com.gexin.rp.sdk.base.impl.AppMessage;
import com.gexin.rp.sdk.http.IGtPush;

public class PushToAPP {
  //采用"Java SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置，用户可以自行替换
   static String appId = "";
   static String appkey = "";
   static String mastersecret = "";
   //请输入你的cid
   static String CID = "";
   static String host = "http://sdk.open.api.igexin.com/apiex.htm";

   public static void main(String[] args) throws Exception {
      IGtPush push = new IGtPush(host, appkey, mastersecret);

      AppMessage message = new AppMessage();
          message.setSpeed(100);//全量推送个推控制下发速度在100条/秒，只有toApp支持定速推送。
   }
}
```

以上示例代码仅部分展示，若要查询完整demo请看指定应用群推消息推送示例

6. 定时任务推送

6.1 定时对指定应用群推消息

描述

对单个或多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

```
public class PushtoAPP {
    //采用"Java SDK 快速入门", "第二步 获取访问凭证 "中获得的应用配置，用户可以自行替换
    private static String appId = "";
    private static String appKey = "";
    private static String masterSecret = "";
    static String host = "http://sdk.open.api.igexin.com/apiex.htm";
    public static void main(String[] args) throws Exception {
        IGtPush push = new IGtPush(host, appKey, masterSecret);
        LinkTemplate template = linkTemplateDemo();
        AppMessage message = new AppMessage();
        message.setData(template);
        message.setOffline(true);
        //离线有效时间，单位为毫秒，可选
        message.setOfflineExpireTime(24 * 1000 * 3600);
        //设置推送时间
        message.setPushTime("201710261050");
        //推送给App的目标用户需要满足的条件
        AppConditions cdt = new AppConditions();
        List<String> appIdList = new ArrayList<String>();
        appIdList.add(appId);
        message.setAppIdList(appIdList);
        //手机类型
        List<String> phoneTypeList = new ArrayList<String>();
        //省份
        List<String> provinceList = new ArrayList<String>();
        //自定义tag
        List<String> tagList = new ArrayList<String>();
        cdt.addCondition(AppConditions.PHONE_TYPE, phoneTypeList);
        cdt.addCondition(AppConditions.REGION, provinceList);
        cdt.addCondition(AppConditions.TAG, tagList);
        message.setConditions(cdt);

        IPushResult ret = push.pushMessageToApp(message, "任务别名_toApp");
        System.out.println(ret.getResponse().toString());
    }
    public static LinkTemplate linkTemplateDemo() throws Exception {
        LinkTemplate template = new LinkTemplate();
        template.setAppId(appId);
        template.setAppkey(appKey);

        Style0 style = new Style0();
        // 设置通知栏标题与内容
        style.setTitle("请输入通知栏标题");
        style.setText("请输入通知栏内容");
        // 配置通知栏图标
        style.setLogo("icon.png");
        // 配置通知栏网络图标
        style.setLogoUrl("");
        // 设置通知是否响铃，震动，或者可清除
        style.setRing(true);
        style.setVibrate(true);
        style.setClearable(true);
        template.setStyle(style);
```

```
        template.setUrl("http://www.baidu.com");
        return template;
    }
}
```

接口部分参数详细说明

- 使用推送的sdk包版本必须大于等于4.0.1.15。
- 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:
    1. 时间格式不正确 提交任务时 将直接返回失败。
    2. 下发时间小于当前时间 提交任务时将直接返回失败。
    3. 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
- 其他参数设定参考个推官网 : http://docs.getui.com/server/java/push/#3

6.2 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。

函数说明

接口定义
IPushResult getScheduleTask(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

```
public class FindScheduleTask {
    private static String appId = "";
    private static String appKey = "";
    private static String masterSecret = "";
    static String host = "http://sdk.open.api.igexin.com/apiex.htm";

    public static void main(String[] args) {
        String taskid="XXXXXXX";
        IGtPush push = new IGtPush(host, appKey, masterSecret);
        IPushResult ret=push.getScheduleTask(taskid,appId);
        System.out.println(ret.getResponse().toString());
    }
}
```

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

6.3 定时任务删除接口

描述

用来删除还未下发的任务

函数说明

接口定义
IPushResult delScheduleTask(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

```
public class DeleteScheduleTask {
    private static String appId = "";
    private static String appKey = "";
    private static String masterSecret = "";
    static String host = "http://sdk.open.api.igexin.com/apiex.htm";

    public static void main(String[] args) {
        String taskid="xxxxxxx";
        IGtPush push = new IGtPush(host, appKey, masterSecret);
        IPushResult result=push.delScheduleTask(taskid,appId);
    }
}
```

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。

7. 任务组名推送

7.1 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。

7.2 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。

7.3 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。

7.3.1 toList接口代码实例

```
void toListOfGroupName(String host, String appkey, String mastersecret,ListMessage message,
String taskGroupName/*任务组名*/) {
    IGtPush push = new IGtPush(host, appkey, mastersecret);
    push.getContentId(message, taskGroupName);
}
```

7.3.2 toApp接口代码实例

```
void toAppOfGroupName(String host, String appkey, String mastersecret,AppMessage message, String
taskGroupName/*任务组名*/) {
    IGtPush push = new IGtPush(host, appkey, mastersecret);
    push.pushMessageToApp(message, taskGroupName);
}
```

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(String key, List<String> values, int optType)
```

参数说明：

参数名	类型	必需	默认值	参数描述
key	String	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

```
package os_push;

import java.util.ArrayList;
import java.util.List;

import com.gexin.rp.sdk.base.IPushResult;
import com.gexin.rp.sdk.base.impl.AppMessage;
import com.gexin.rp.sdk.base.uitls.AppConditions;
import com.gexin.rp.sdk.base.uitls.AppConditions.OptType;
import com.gexin.rp.sdk.http.IGtPush;
import com.gexin.rp.sdk.template.LinkTemplate;
import com.gexin.rp.sdk.template.TransmissionTemplate;

public class pushtoAPP {
        static String appId = "";
        static String appkey = "";
        static String master = "";
        static String host = "https://api.getui.com/apiex.htm";

        public static void main(String[] args) throws Exception {

                IGtPush push = new IGtPush(host, appkey, master);
                LinkTemplate template = linkTemplateDemo();
                AppMessage message = new AppMessage();
                message.setData(template);

                message.setOffline(true);
                message.setOfflineExpireTime(24 * 1000 * 3600);

                List<String> appIdList = new ArrayList<String>();
                List<String> phoneTypeList = new ArrayList<String>();
                List<String> provinceList = new ArrayList<String>();
                List<String> tagList = new ArrayList<String>();

                appIdList.add(appId);

                phoneTypeList.add("ANDROID");
                phoneTypeList.add("IOS");

                provinceList.add("湖南省");

                tagList.add("tag1");
                tagList.add("tag2");

                message.setAppIdList(appIdList);

                // 设置省市平台tag的新方式
                AppConditions cdt = new AppConditions();
                cdt.addCondition(AppConditions.PHONE_TYPE, phoneTypeList, OptType.or);
                cdt.addCondition(AppConditions.REGION, provinceList, OptType.or);
                cdt.addCondition(AppConditions.TAG, tagList, OptType.or);
                message.setConditions(cdt);
```

```
                IPushResult ret = push.pushMessageToApp(message, "taskName_toApp");
                System.out.println(ret.getResponse().toString());
        }

        public static LinkTemplate linkTemplateDemo() throws Exception {
                LinkTemplate template = new LinkTemplate();
                template.setAppId(appId);
                template.setAppkey(appkey);

                Style0 style = new Style0();
        // 设置通知栏标题与内容
        style.setTitle("请输入通知栏标题");
        style.setText("请输入通知栏内容");
        // 配置通知栏图标
        style.setLogo("icon.png");
        // 配置通知栏网络图标
        style.setLogoUrl("");
        // 设置通知是否响铃，震动，或者可清除
        style.setRing(true);
        style.setVibrate(true);
        style.setClearable(true);
        template.setStyle(style);

                template.setUrl("http://www.getui.com");
                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：

接口定义	说明
String add(SingleMessage message, Target target)	追加单推消息
IPushResult submit()	提交消息
IPushResult retry()	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IPushResult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IpushResult 返回值

代码实例：

```
package com.igexin.sdk.http.test;
```

```
import java.io.IOException;
```

```
import com.gexin.rp.sdk.base.IBatch;
import com.gexin.rp.sdk.base.IIGtPush;
import com.gexin.rp.sdk.base.impl.SingleMessage;
import com.gexin.rp.sdk.base.impl.Target;
import com.gexin.rp.sdk.http.IGtPush;
import com.gexin.rp.sdk.template.LinkTemplate;
import com.gexin.rp.sdk.template.TransmissionTemplate;

public class MyBatchPushDemo {
        //采用"Java SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置，用户可以自行替换
        private static String appId = "";
        private static String appKey = "";
        private static String masterSecret = "";

        static String CID_A = "";
        static String CID_B = "";
        //别名推送方式
        // static String Alias = "";
        static String host = "http://sdk.open.api.igexin.com/apiex.htm";

        public static void main(String[] args) throws IOException {

                IIGtPush push = new IGtPush(host, appKey, masterSecret);
                IBatch batch = push.getBatch();

                try {
                        //构建客户a的透传消息a
                        constructClientTransMsg(CID_A,"msgA",batch);
                        //构建客户B的点击通知打开网页消息b
                        constructClientLinkMsg(CID_B,"msgB",batch);
                } catch (Exception e) {
                        e.printStackTrace();
                }
                batch.submit();
        }

        private static void constructClientTransMsg(String cid, String msg ,IBatch batch) throws
Exception {

                SingleMessage message = new SingleMessage();
                TransmissionTemplate template = new TransmissionTemplate();
                template.setAppId(appId);
                template.setAppkey(appKey);
                template.setTransmissionContent(msg);
                template.setTransmissionType(1); // 这个Type为int型，填写1则自动启动app

                message.setData(template);
                message.setOffline(true);
                message.setOfflineExpireTime(1 * 1000);

                // 设置推送目标，填入appid和clientId
                Target target = new Target();
                target.setAppId(appId);
```

```
                target.setClientId(cid);
                batch.add(message, target);
        }

        private static void constructClientLinkMsg(String cid, String msg ,IBatch batch) throws
Exception {

                SingleMessage message = new SingleMessage();
                LinkTemplate template = new LinkTemplate();
                template.setAppId(appId);
                template.setAppkey(appKey);
                template.setTitle("title");
                template.setText("msg");
                template.setLogo("push.png");
                template.setLogoUrl("logoUrl");
                template.setUrl("url");

                message.setData(template);
                message.setOffline(true);
                message.setOfflineExpireTime(1 * 1000);

                // 设置推送目标，填入appid和clientId
                Target target = new Target();
                target.setAppId(appId);
                target.setClientId(cid);
                batch.add(message, target);
        }
}
```