个推 SDK 文档

getui.com

October 2016

目录

消息推送方式 3

1. 对单个用户推送消息 . . . 3
   - 1.1 描述 . . . 3
   - 1.2 应用场景 . . . 3
   - 1.3 对应接口（pushMessageToSingle） . . . 3
2. 对指定列表用户推送消息 . . . 5
   - 2.1 描述 . . . 5
   - 2.2 应用场景 . . . 5
   - 2.3 对应接口 . . . 5
3. 对指定应用群推消息 . . . 9
   - 3.1 描述 . . . 9
   - 3.2 应用场景 . . . 9
   - 3.3 接口（pushMessageToApp-对指定应用群推消息） . . . 9
4. wifi 推送 . . . 11
   - 4.1 描述 . . . 11
   - 4.2 应用场景 . . . 12
   - 4.3 对应接口 . . . 12
   - 4.4 代码实例 . . . 12
5. 定速推送 . . . 13
   - 5.1 描述 . . . 13
   - 5.2 应用场景 . . . 13
   - 5.3 对应接口 . . . 13
   - 5.4 代码实例 . . . 13
6. 任务组名推送 . . . 14
   - 6.1 描述 . . . 14
   - 6.2 应用场景 . . . 14
   - 6.3 对应接口 . . . 14

7. 应用群推条件交并补功能 14
7.1 描述 14
7.2 应用场景 14
7.3 对应接口 15
8. 批量单推功能 17
8.1 描述 17
8.2 应用场景 17
8.3 对应接口 17

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：

```
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;
```

1. 对单个用户推送消息 消息推送方式

```
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);
        // 设置通知栏标题与内容
        template.setTitle("请输入通知栏标题");
```

```
        template.setText("请输入通知栏内容");
        // 配置通知栏图标
        template.setLogo("icon.png");
        // 配置通知栏网络图标，填写图标URL地址
        template.setLogoUrl("");
        // 设置通知是否响铃，震动，或者可清除
        template.setIsRing(true);
        template.setIsVibrate(true);
        template.setIsClearable(true);
        // 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

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

代码实例

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

```
        // 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

代码实例

```
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);
        template.setTitle("请输入通知栏标题");
        template.setText("请输入通知栏内容");
        template.setLogo("icon.png");
        template.setLogoUrl("");
        template.setIsRing(true);
        template.setIsVibrate(true);
```

```
        template.setIsClearable(true);
        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. wifi 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. 任务组名推送

6.1 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。

6.2 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。

6.3 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。

6.3.1 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);
}
```

6.3.2 Toapp 接口代码实例

```
void toListOfGroupName(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);
}
```

7. 应用群推条件交并补功能

7.1 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

7.2 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)

7.3 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)

7.4 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);
        template.setTitle("ddd");
        template.setText("ddd");
        template.setLogo("icon.png");
        template.setLogoUrl("");
        template.setIsRing(true);
        template.setIsVibrate(true);
        template.setIsClearable(true);
        template.setUrl("http://www.getui.com");
        return template;
    }

}
```

8. 批量单推功能

8.1 描述

用于一次创建提交多个单推任务。

8.2 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。

8.3 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：

```
import com.gexin.rp.sdk.base.IBatch;
import com.gexin.rp.sdk.base.IIGtPush;
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.http.IGtPush;
import com.gexin.rp.sdk.template.TransmissionTemplate;

import java.io.IOException;
import java.util.Map;

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 = "";
//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();

    SingleMessage message = new SingleMessage();

    TransmissionTemplate template = new TransmissionTemplate();
    template.setAppId(appId);
    template.setAppkey(appKey);
    template.setTransmissionContent("测试用的透传文本");
    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);
    long addStart = System.currentTimeMillis();
    for (int i = 1; i < 10000000; i++) {
        try {
            batch.add(message, target);

            if (i % 500 == 0) {
                long addEnd = System.currentTimeMillis();
                System.out.println("添加500条消息共耗时" + (addEnd - addStart) + " ms");

                IPushResult whatServerGot = batch.submit();
                String result = (String) whatServerGot.getResponse().get("result");
                Map<String, Object> info = (Map<String, Object>)
                    whatServerGot.getResponse().get("info");
                System.out.println("result:" + result);
                System.out.println("info:" + info);
            }
        } catch (Exception e) {
            e.printStackTrace();
```

```
            }
        }
        batch.submit();
    }
}
```