# 其他接口

## 1. 别名接口

该接口下的代码示例全为同一个类里

### 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, clientid)
```

**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 |
| clientid | String | 是 | 无 | 用户id |

代码实例

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;

namespace GetuiServerApiSDKDemo
{
   public class demo
   {
      //参数设置 <-----参数需要重新设置----->
      //http的域名
      private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";

      //https的域名
      //private static String HOST = "https://api.getui.com/apiex.htm";
      //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置
      private static String APPID = "";
      private static String APPKEY = "";
      private static String MASTERSECRET = "";
      private static String CLIENTID = "";               //您获取的clientid
      private static String ALIAS = "请输入别名";

      static void Main(string[] args)
      {

         bindAlias();
      }
      public static void bindAlias()
      {
         IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
         String ret = push.bindAlias(APPID, ALIAS, CLIENTID);
         System.Console.WriteLine(ret);
      }

  }
}
```

### 1.2.2 bindAlias-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：

```
bindAlias(appid, Lcids)
```

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 |
| Lcids | List | 是 | 无 | 别名绑定用户列表 |

代码实例

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;
using System.Collections.Generic;

namespace GetuiServerApiSDKDemo
{
    public class demo
    {
        //参数设置 <-----参数需要重新设置----->
        //http的域名
        private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";

        //https的域名
        //private static String HOST = "https://api.getui.com/apiex.htm";

        //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置
```

```
        private static String APPID = "";
        private static String APPKEY = "";
        private static String MASTERSECRET = "";
        private static String CLIENTID = "";                //您获取的clientid
        private static String CLIENTID1 = "";
        private static String ALIAS = "请输入别名";
        private static String ALIAS1 = "别名1";

        static void Main(string[] args)
        {

            bindAliasAll();
        }
        public static void bindAliasAll()
        {
            IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
            List<com.igetui.api.openservice.igetui.Target> Lcids = new
List<com.igetui.api.openservice.igetui.Target>();
            com.igetui.api.openservice.igetui.Target target = new
com.igetui.api.openservice.igetui.Target();
            target.clientId = CLIENTID;
            target.alias = ALIAS;

            com.igetui.api.openservice.igetui.Target target1 = new
com.igetui.api.openservice.igetui.Target();
            target1.clientId = "";
            target1.alias = ALIAS1;



            Lcids.Add(target);
            Lcids.Add(target1);

            String ret = push.bindAlias(APPID, Lcids);
            System.Console.WriteLine(ret);
        }
    }
}
//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

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;

namespace GetuiServerApiSDKDemo
{
   public class demo
   {
      //参数设置 <-----参数需要重新设置----->
      //http的域名
      private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";

      //https的域名
      //private static String HOST = "https://api.getui.com/apiex.htm";
      //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置
      private static String APPID = "";
      private static String APPKEY = "";
      private static String MASTERSECRET = "";
      private static String CLIENTID = "";              //您获取的clientID
      private static String ALIAS = "请输入别名";

      static void Main(string[] args)
      {
```

```
        queryClientId();
    }
    public static void queryClientId()
    {
        IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
        String ret = push.queryClientId(APPID, ALIAS);
        System.Console.WriteLine(ret);
    }

 }
}
```

### 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 |

代码实例

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;

namespace GetuiServerApiSDKDemo
{
  public class demo
  {
```

```
        //参数设置 <-----参数需要重新设置----->
        //http的域名
        private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";

        //https的域名
        //private static String HOST = "https://api.getui.com/apiex.htm";
        //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置
        private static String APPID = "";
        private static String APPKEY = "";
        private static String MASTERSECRET = "";
        private static String CLIENTID = "";                //您获取的clientID
        private static String ALIAS = "请输入别名";

        static void Main(string[] args)
        {

            queryAlias();
        }
        public static void queryAlias()
        {
            IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
            String ret = push.queryAlias(APPID, CLIENTID);
            System.Console.WriteLine(ret);
        }

    }
}
```

### 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 |
| cid | String | 是 | 无 | 用户id |
| Alias | String | 是 | 无 | 用户别名 |

代码实例

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;

namespace GetuiServerApiSDKDemo
{
   public class demo
   {
      //参数设置 <-----参数需要重新设置----->
      //http的域名
      private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";

      //https的域名
      //private static String HOST = "https://api.getui.com/apiex.htm";

      //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置
      private static String APPID = "";
      private static String APPKEY = "";
      private static String MASTERSECRET = "";
      private static String CLIENTID = "";               //您获取的clientID
      private static String ALIAS = "请输入别名";

      static void Main(string[] args)
      {

         aliasUnBind();
      }
      public static void aliasUnBind()
      {
         IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
         String ret = push.unBindAlias(APPID, ALIAS, CLIENTID);
         System.Console.WriteLine(ret);
      }

  }
}
```

### 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：

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;

namespace GetuiServerApiSDKDemo
{
    public class demo
    {
        //参数设置 <-----参数需要重新设置----->
        //http的域名
        private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";

        //https的域名
        //private static String HOST = "https://api.getui.com/apiex.htm";

        //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置
        private static String APPID = "";
        private static String APPKEY = "";
```

```
        private static String MASTERSECRET = "";
        private static String CLIENTID = "";                //您获取的clientid
        private static String ALIAS = "请输入别名";

        static void Main(string[] args)
        {

          aliasUnBindAll();
        }
        public static void aliasUnBindAll()
        {
           IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
           String ret = push.unBindAliasAll(APPID, ALIAS);
           System.Console.WriteLine(ret);
        }

    }
}
```

## 2. 标签接口

### 2.1 标签说明

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 对应接口

#### 2.2.1 setClientTag-通过cid设置用户标签(可设置多个)

接口名称：**setClientTag(APPID, clientid, list)**

参数：

| 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 |
| clientid | String | 是 | 无 | 用户唯一识别id |
| list | | List | 是 | 无 |

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

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;
using System.Collections.Generic;

namespace GetuiServerApiSDKDemo
{
   public class demo
   {
      //参数设置 <-----参数需要重新设置----->
      //http的域名
      private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";

      //https的域名
      //private static String HOST = "https://api.getui.com/apiex.htm";

      //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置
      private static String APPID = "";
      private static String APPKEY = "";
      private static String MASTERSECRET = "";
      private static String CLIENTID = "";               //您获取的clientID

      static void Main(string[] args)
      {

         setTag();
      }
      public static void setTag() {
         IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);

         List<String>  list=new List<String>();
         list.Add("");
```

```
        String ret =push.setClientTag(APPID, CLIENTID, list);
        System.Console.WriteLine(ret);
    }
 }
}
```

### 2.2.2 getUserTags-通过cid获取用户标签

接口名称： **getUserTags(APPID, clientid)**

参数：

| 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 |
| clientid | String | 是 | 无 | 用户唯一识别id |

代码实例

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;

namespace GetuiServerApiSDKDemo
{
  public class demo
  {
    //参数设置 <-----参数需要重新设置----->
    //http的域名
    private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";

    //https的域名
    //private static String HOST = "https://api.getui.com/apiex.htm";

    //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置
```

```
        private static String APPID = "";
        private static String APPKEY = "";
        private static String MASTERSECRET = "";
        private static String CLIENTID = "";                //您获取的clientID

        static void Main(string[] args)
        {

            getUserTags();
        }
        public static void getUserTags() {
            IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
            String ret = push.getUserTags(APPID,CLIENTID);
            System.Console.WriteLine(ret);
        }
    }
}
```

## 3. 停止任务接口

### 3.1 描述

stop-对正处于推送状态，或者未接收的消息停止下发（list或app任务）

### 3.2 接口名称:

`stop(taskId)`

### 3.3 参数

| 参数 | 类型 | 说明 |
| --- | --- | --- |
| taskId | String | 任务ID(格式OSL-yyMM_XXXXXX) |

### 3.4 stop代码实例

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
```

```
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;

namespace GetuiServerApiSDKDemo
{
   public class demo
   {
      //参数设置 <-----参数需要重新设置----->
      //http的域名
      private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";

      //https的域名
      //private static String HOST = "https://api.getui.com/apiex.htm";

      //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置

      private static String APPKEY = "";
      private static String MASTERSECRET = "";

      static void Main(string[] args)
      {

         taskStop();
      }
   public static void taskStop()
   {
      IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
      Boolean result = push.stop("");
      System.Console.WriteLine("----------------------------------------------
---");
      System.Console.WriteLine(result);
   }
 }
}
```

## 4. 查询用户状态

### 4.1 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。

## 4.2 接口名称： getClientIdStatus(APPID,clientid)

## 4.3 接口参数

| 参数 | 类型 | 说明 |
| --- | --- | --- |
| APPID | String | 设置推送的appid |
| clientid | String | 用户唯一标识符（获取方式请参考<前期准备>第一章节） |

## 4.4 getClientIdStatus代码实例

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;

namespace GetuiServerApiSDKDemo
{
   public class demo
   {
      //参数设置 <-----参数需要重新设置----->
      //http的域名
      private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";

      //https的域名
      //private static String HOST = "https://api.getui.com/apiex.htm";

      //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置
      private static String APPID = "";
      private static String APPKEY = "";
      private static String MASTERSECRET = "";
      private static String CLIENTID = "";              //您获取的clientID

      static void Main(string[] args)
      {
         getUserStatus();
      }
```

```
        public static void getUserStatus()
        {
            IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
            String ret = push.getClientIdStatus(APPID, CLIENTID);
            System.Console.WriteLine("----------------------------------------
-----");
            Console.WriteLine("用户状态:" + ret);
        }
    }
}
```

## 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
setBadgeForDeviceToken(badge, appId，deviceTokenList)

// 根据 clientid 设置 Badge
setBadgeForCID(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 | List | 是 | 无 | 目标用户clientid列表 |
| deviceTokenList | List | 是 | 无 | iOS用户DeviceToken列表 |

### 5.4 返回值

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

### 5.5 代码实例-根据 deviceToken 设置 Badge

```
private static void setBadgeForDeviceTokenDemo()
    {
      IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
      List<string> deviceTokenList = new List<string>();
      deviceTokenList.Add("");
      string res = push.setBadgeForDeviceToken(Badge, APPID,
deviceTokenList);
      Console.WriteLine("----------------------------------------------");
      Console.WriteLine(res);
    }
```

### 5.6 代码实例-根据 clientid 设置 Badge

```
private static void setBadgeForCIDDemo()
    {
      IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
      List<string> cidList = new List<string>();
      cidList.Add(CLIENTID);
      //cidList.Add(CLIENTID1);
      string res = push.setBadgeForCID(Badge, APPID, cidList);
      Console.WriteLine("----------------------------------------------");
      Console.WriteLine(res);
    }
```

## 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
addCidListToBlk(String appId, List<String> cidList)

// 移除黑名单用户
restoreCidListFromBlk(String appId, List<String> cidList)
```

### 6.3 参数说明

| 参数名 | 类型 | 必需 | 默认值 | 参数描述 |
| --- | --- | --- | --- | --- |
| appId | String | 是 | 无 | appId |
| cidList | List | 是 | 无 | 该appId下的用户cid列表 |

### 6.4 返回值

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

### 6.5 代码实例-添加用户到黑名单

```
private static void addCidListToBlkDemo()
    {
      IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
      List<string> cidList = new List<string>();
      cidList.Add(CLIENTID);
      cidList.Add(CLIENTID1);
      cidList.Add(CLIENTID2);
      string res = push.addCidListToBlk(APPID, cidList);
      Console.WriteLine("----------------------------------------------");
      Console.WriteLine(res);
    }
```

### 6.6 代码实例-将用户从黑名单移除

```
private static void restoreCidListFromBlkDemo()
    {
      IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
      List<string> cidList = new List<string>();
      cidList.Add(CLIENTID);
      cidList.Add(CLIENTID1);
      cidList.Add(CLIENTID2);
      string res = push.restoreCidListFromBlk(APPID, cidList);
      Console.WriteLine("----------------------------------------------");
      Console.WriteLine(res);
    }
```

## 7. 获取推送结果

## 7.1 描述

## 7.2 接口名称：

`getPushResult(taskId)`

## 7.3 参数说明

| 参数 | 参数说明 |
| --- | --- |
| url | 接口地址(http://sdk.open.api.igexin.com/apiex.htm) |
| Appkey | 用于鉴定身份是否合法 |
| masterSecret | 第三方客户端个推集成鉴权码，用于验证第三方合法性。在客户端集成SDK时需要提供 |
| taskId | 任务唯一识别号 (格式OSL-yyMM_XXXXXX) |

## 7.4 返回值

| 回执 | 参数说明 |
| --- | --- |
| | {“taskeId”:”********,”result”:”ok”,”msgTotal”:,”msgProcess”:0} |
| | tasked：任务IDmsgTotal:表示有效可下发总数result：OK执行成功<br>result：sign_error 表示校验失败 msgProcess：收到消息回执总数 |

## 7.5 获取推送结果实例

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;

namespace GetuiServerApiSDKDemo
{
    public class demo
```

```
  {
    //参数设置 <-----参数需要重新设置----->
    //http的域名
    private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";

    //https的域名
    //private static String HOST = "https://api.getui.com/apiex.htm";

    //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置

    private static String APPKEY = "";
    private static String MASTERSECRET = "";


    static void Main(string[] args)
    {

       getPushResult();
    }
     public static void getPushResult()
    {
       IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
       String ret = push.getPushResult("");
       System.Console.WriteLine(ret);

    }

 }
}
```

## 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：

`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 |
| appId | 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 | 新注册用户数 |
| activeCount | data | 活跃用户数 |
| onlineCount | data | 在线用户数 |

## 8.5 queryAppUserDataByDate代码示例

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;

namespace GetuiServerApiSDKDemo
{
    public class demo
    {
        //参数设置 <-----参数需要重新设置----->
        //http的域名
        private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";
```

```
        //https的域名
        //private static String HOST = "https://api.getui.com/apiex.htm";

        //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置
        private static String APPID = "";
        private static String APPKEY = "";
        private static String MASTERSECRET = "";


        static void Main(string[] args)
        {

            queryAppUserDataByDate();
        }
        public static void queryAppUserDataByDate()
        {
            IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
            String ret = push.queryAppUserDataByDate(APPID, "20150910");
            System.Console.WriteLine(ret);
        }

    }
}
```

## 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:

`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 queryAppPushDataByDate代码示例：

```
using System;
using System.Linq;
using System.Text;
using Google.ProtocolBuffers;
using com.gexin.rp.sdk.dto;
using com.igetui.api.openservice;
using com.igetui.api.openservice.igetui;
using com.igetui.api.openservice.igetui.template;
using com.igetui.api.openservice.payload;
using System.Net;

namespace GetuiServerApiSDKDemo
{
    public class demo
    {
        //参数设置 <-----参数需要重新设置----->
        //http的域名
        private static String HOST =
"http://sdk.open.api.igexin.com/apiex.htm";

        //https的域名
        //private static String HOST = "https://api.getui.com/apiex.htm";
        //采用"C# SDK 快速入门"， "第二步 获取访问凭证 "中获得的应用配置
        private static String APPID = "";
        private static String APPKEY = "";
        private static String MASTERSECRET = "";
```

```
        static void Main(string[] args)
        {
            queryAppPushDataByDate();
        }
        public static void queryAppPushDataByDate()
        {
            IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
            String ret = push.queryAppPushDataByDate(APPID, "20150910");
            System.Console.WriteLine(ret);
        }

    }
}
```

## 10. 获取任务组名推送结果

### 10.1 描述

根据任务组名查询推送结果，返回结果包括百日内联网用户数（活跃用户数）、实际下发数、到达数、展示数、点击数。

### 10.2 函数说明

```
getPushResultByGroupName(String appId, String groupName)
```

### 10.3 参数说明

| 参数名 | 类型 | 必需 | 默认值 | 参数描述 |
| --- | --- | --- | --- | --- |
| appId | String | 是 | 无 | appId |
| groupName | String | 是 | 无 | 推送任务组名 |

### 10.4 返回值

```
{
   "result": "",
   "groupName": "",
   "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
   }
}
```

### 10.5 代码实例

```
private static void getPushResultByGroupNameDemo()
      {
         IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
         string res = push.getPushResultByGroupName(APPID, GroupName);
         Console.WriteLine("----------------------------------------------");
         Console.WriteLine(res);
      }
```

## 11. 获取24小时在线用户数

### 11.1 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）

### 11.2 接口

### 11.3 函数说明

```
getLast24HoursOnlineUserStatistics(appId)
```

### 11.4 参数说明

| 参数名 | 类型 | 必需 | 默认值 | 参数描述 |
| --- | --- | --- | --- | --- |

| | | | | |
|---|---|---|---|---|
| appId | String | 是 | 无 | appId |

### 11.5返回值

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

| 字段 | 取值 | 说明 |
|---|---|---|
| appId | String | 用户appId |
| onlineStatics | Dict | 24小时用户在线数统计 |

### 11.6 代码实例

```
private static void getLast24HoursOnlineUserStatisticsDemo()
    {
      IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
      string appid = APPID;
      string res = push.getLast24HoursOnlineUserStatistics(appid);
      Console.WriteLine("----------------------------------------------");
      Console.WriteLine(res);
    }
```

## 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

```
queryUserCount(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 |

### 12.4 返回值

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

### 12.5 代码实例

```
public static void queryUserCountDemo()
      {
         IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
         AppConditions conditions = new AppConditions();
         List<string> phonelist = new List<string>();
         phonelist.Add("ANDROID");
         //phonelist.Add("IOS");
         List<string> provinceList = new List<string>();
         provinceList.Add("浙江省");
         provinceList.Add("上海市");
         List<string> tagList = new List<string>();
         tagList.Add("lala");
         tagList.Add("66");
         conditions.addCondition(AppConditions.PHONE_TYPE, phonelist);
         conditions.addCondition(AppConditions.REGION,
provinceList,AppConditions.OptType.not);
         conditions.addCondition(AppConditions.TAG, tagList
,AppConditions.OptType.or);

         string res = push.queryUserCount(APPID, conditions);
         Console.WriteLine("-----------------------------------------------");
         Console.WriteLine(res);
      }
```

## 13. 大数据综合分析用户得到的标签:即用户画像

### 13.1 描述

通过接口查询当前bi统计的用户画像标签,该接口需要申请后才可正常使用，且主要是让APP查看自己能使用那些标签进行推送

申请用户画像标签联系邮箱：kegf@getui.com

### 13.2 函数说明

```
getPersonaTags(appId)
```

### 13.3 参数说明

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

### 13.4 返回值

具体返回值

| 字段 | 取值 | 说明 |
| --- | --- | --- |
| result | String | 获取tags的结果,success：成功 |
| tags | List | 应用的用户画像标签 |

### 13.5 代码实例

```
private static void getPersonaTagsDemo()
    {
        IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
        string res = push.getPersonaTags(APPID);
        Console.WriteLine("----------------------------------------------");
        Console.WriteLine(res);
    }
```

## 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

```
getPushResultByTaskidList(List<String> 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

```
private static void getPushResultByTaskidListDemo()
      {
         IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET);
         List<string> taskIdList = new List<string>();
         taskIdList.Add(TASKID);
         taskIdList.Add(TASKID1);
         taskIdList.Add(TASKID2);
         string res = push.getPushResultByTaskidList(taskIdList);
         Console.WriteLine("----------------------------------------------");
         Console.WriteLine(res);
      }
```

# 14. 日志集成

## 14.1 描述

考虑到log4net版本兼容性，所以提供ILog接口，由用户自主实现

## 14.2 代码实例

```
public class LogUtil : ILog
  {
    void ILog.debug(params object[] values)
    {
      Console.WriteLine(values);
    }
    void ILog.error(Exception e, params object[] values)
    {
      Console.WriteLine(values);
    }
    void ILog.info(params object[] values)
    {
      Console.WriteLine(values);
    }
    void ILog.slow(params object[] values)
    {
      Console.WriteLine(values);
    }
  }
private static void getPushResultByTaskidListDemo()
    {
      GetuiServerApiSDK.utils.ILog log = new LogUtil();
      IGtPush push = new IGtPush(HOST, APPKEY, MASTERSECRET,log);
      List<string> taskIdList = new List<string>();
      taskIdList.Add(TASKID);
      taskIdList.Add(TASKID1);
      taskIdList.Add(TASKID2);
      string res = push.getPushResultByTaskidList(taskIdList);
      Console.WriteLine("----------------------------------------------");
      Console.WriteLine(res);
    }
```