



个推 IOS SDK 1.3.2 接口 API

目录

个推 IOS SDK 1.3.2 接口 API.....	1
◆ SDK 方法说明:	2
1. 启动个推 SDK.....	2
2. 停止个推 SDK (已经弃用)	2
3. 销毁个推 SDK.....	3
4. 提交 APNS 注册后返回的 devicetoken.....	3
5. 根据 payloadId 取回推送消息.....	3
6. 设置用户标签.....	4
7. 绑定用户别名.....	5
8. 解绑用户别名.....	5
9. SDK 发送上行消息结果.....	6
10. 是否允许 SDK 后台运行.....	6
11. 恢复 SDK 运行.....	7
12. SDK 地理围栏功能, 设置地理围栏是否运行.....	7
13. 获取 SDK 系统版本号.....	7
14. 获取 clientid.....	8
15. 获取 status	8
16. 设置处理显示的 UIAlertView 是否随屏幕旋转.....	8
17. 设置关闭推送模式.....	9
18. 上行第三方自定义回执 actionid.....	9
◆ SDK Delegate 回调接口:	10
1. SDK 登入成功返回 clientId	10
2. SDK 通知收到个推推送消息.....	10
3. SDK 通知发送上行消息结果.....	11
4. SDK 遇到错误消息返回 error.....	11
5. SDK 运行状态通知.....	11
6. SDK 设置关闭推送模式回调.....	12



◆ SDK 方法说明:

1. 启动个推 SDK

```
+ (void)startSdkWithAppId:(NSString *)appid appKey:(NSString *)appKey  
appSecret:(NSString *)appSecret delegate:(id<GeTuiSdkDelegate>)delegate; // 启动  
SDK，该方法需要在主线程中调用
```

参数:

appid:个推登记应用的 appid

appKey:个推登记应用的 appKey

appSecret:个推登记应用的 appSecret

delegate:推送消息回调接口

说明:

appid、appKey 和 appSecret 必须正确，否则会导致推送消息无法接收。

头文件:

GeTuiSdk.h

示例:

```
[GeTuiSdk startSdkWithAppId:kGtAppId appKey:kGtAppKey  
appSecret:kGtAppSecret delegate:self];  
// 通过 appId、 appKey 、 appSecret 启动 SDK，注：该方法需要在主线程中  
调用
```

~~2. 停止个推 SDK（已经弃用）~~

```
+ (void)stopSdk; // 停止 SDK，并且释放资源，该方法已经弃用
```

说明:

停止 SDK，并且释放资源。

头文件:

GeTuiSdk.h

示例:

```
[GeTuiSdk stopSdk]; // 停止 SDK
```



3. 销毁个推 SDK

```
+ (void)destroy;    // 销毁 SDK，并且释放资源
```

说明：

销毁 SDK，并且释放资源

头文件：

GeTuiSdk.h

示例：

```
[GeTuiSdk destroy];    // 销毁 SDK
```

4. 提交 APNS 注册后返回的 devicetoken

```
+ (void)registerDeviceToken:(NSString *)deviceToken; // 注册 DeviceToken
```

参数：

deviceToken:APNS 返回的 client 标识。

说明：

个推服务需要使用 APNS 来辅助推送消息，如果不提交 deviceToken，会导致推送功能不正常。请在收到后将 deviceToken 字符取出后调用这个接口提交 deviceToken。具体实现请参考 Demo 工程。如果注册 APNS 失败导致无法获取 deviceToken，仍然需要上传空的 deviceToken，个推服务器在应用离线的时候将不会发送 APNS。

头文件：

GeTuiSdk.h

示例：

```
[GeTuiSdk registerDeviceToken:token];  
// [3]:向个推服务器注册 deviceToken
```

5. 根据 payloadId 取回推送消息

```
+ (NSData *)retrivePayloadById:(NSString *)payloadId; // 根据 payloadId 取回 Payload
```

参数：

payloadId:接收的推送消息的 id，如果 id 不正确，将无法取到消息内容。



返回值:

无法取到消息内容，返回 nil。

说明:

1.该接口需要配合 GeTuiSdkDelegate 中

-(void)GeTuiSdkDidReceivePayload:(NSString *)payloadId andTaskId:(NSString*)taskId andMessageId:(NSString*)aMsgId fromApplication:(NSString *)appId; 回调接口使用，通过 payloadId 取回收到的 Payload 数据，具体事项可参考 Demo。

2. SDK 仅保持 5 天内的消息，请及时获取。已获取的消息无法再次获取。

头文件:

GeTuiSdk.h

示例:

// [4]: 收到个推消息

```
NSData *payload = [GeTuiSdk retrievePayloadById:payloadId];
```

```
NSString *payloadMsg = nil;
```

```
if (payload) {
```

```
    payloadMsg = [[NSString alloc] initWithBytes:payload.bytes  
                                                  length:payload.length  
                                                  encoding:NSUTF8StringEncoding];
```

```
}
```

6. 设置用户标签

```
+(BOOL)setTags:(NSArray *)tags; // 给用户打标签，后台可以根据标签进行推送
```

参数:

tags:NSString 的对象数组，不能为 nil。只能包含中文字符、英文字母、0-9、空格、+-.的组合。

返回值:

如果 tags 参数不合法，返回 NO。

说明:

给用户打标签，可以在个推后端根据标签内容做差异化推送。具体请询问技术支持。会覆盖上一次的调用结果。

头文件:

GeTuiSdk.h

示例:

```
NSString *tagName = @"个推,推送,iOS";
```



```
NSArray *tagNames = [tagName componentsSeparatedByString:@" "];
if (![GeTuiSdk setTags:tagNames]) {
    UIAlertView *alertView = [[UIAlertView alloc] initWithTitle:@"Failed"
message:@"设置失败" delegate:nil cancelButtonTitle:@"OK" otherButtonTitles:nil];
    [alertView show];
}
```

7. 绑定用户别名

+ (void)bindAlias:(NSString *)alias; // 绑定别名功能:后台可以根据别名进行推送

参数:

alias:别名名称: 长度 40 字节, 支持中、英文(区分大小写)、数字以及下划线。

说明:

- 1、同一个别名最多绑定 10 个 ClientID (适用于允许多设备同时登陆的应用), 当已绑定 10 个 ClientID 时, 再次调用此接口会自动解绑最早绑定的记录;
- 2、当 ClientID 已绑定了别名 A, 若调用此接口绑定别名 B, 则与别名 A 的绑定关系会自动解除;
- 3、此接口与 unBindAlias 一天内最多调用 100 次, 两次调用的间隔需大于 5s;

头文件:

GeTuiSdk.h

示例:

```
[GeTuiSdk bindAlias:@"个推"];
```

8. 解绑用户别名

+ (void)unbindAlias:(NSString *)alias;

参数:

alias:别名名称: 长度 40 字节, 支持中、英文(区分大小写)、数字以及下划线。

说明:

- 1、此接口与 bindAlias 一天内最多调用 100 次, 两次调用的间隔需大于 5s;
- 2、只能解绑当前手机 ClientID 与别名的关系, 不能解绑其他手机上 ClientID 与别名的关系;



头文件:
GeTuiSdk.h

示例:
[GeTuiSdk unbindAlias:@"个推"];

9. SDK 发送上行消息结果

+ (NSString *)sendMessage:(NSData *)body error:(NSError **)error;

参数:

body: 发送数据, error: 如果发送错误给出错误消息。

返回: 如果发送成功返回 messageid, 发送失败返回 nil。

该方法回调 - (void)GeTuiSdkDidSendMessage:(NSString *)messageId
result:(int)result;

头文件:
GeTuiSdk.h

示例:
[GeTuiSdk sendMessage:[content dataUsingEncoding:NSUTF8StringEncoding]
error:&err];

10. 是否允许 SDK 后台运行

+ (void)runBackgroundEnable:(BOOL)isEnabled;

参数:

isEnabled

说明:

SDK 支持当 APP 进入后台后, 个推是否运行。

头文件:
GeTuiSdk.h

示例:
[GeTuiSdk runBackgroundEnable:YES];



11. 恢复 SDK 运行

+ (void)resume; // 该方法需要在主线程中调用

说明:

IOS7 以后支持 Background Fetch 方式，后台定期更新数据。
该接口需要在 Fetch 起来后被调用，保证 SDK 数据获取。

头文件:

GeTuiSdk.h

示例:

[GeTuiSdk resume];

12. SDK 地理围栏功能，设置地理围栏是否运行

+ (void)lbsLocationEnable:(BOOL)isEnabled andUserVerify:(BOOL)isVerify;

说明:

本功能为使用个推 2.0 智能标签和个推 3.0 应景推送的必选功能，建议勾选

参数:

isEnabled: 设置地理围栏功能是否运行

isVerify: 设置是否 SDK 主动弹出用户定位请求

头文件:

GeTuiSdk.h

示例:

[GeTuiSdk lbsLocationEnable:YES andUserVerify:YES];

13. 获取 SDK 系统版本号

+ (NSString *)version;

说明:

返回系统版本号

头文件:

GeTuiSdk.h



示例:

```
[GeTuiSdk version];
```

14. 获取 **clientId**

```
+ (NSString *)clientId;
```

说明:

sdk 登入成功后返回 **clientId**

头文件:

GeTuiSdk.h

示例:

```
[GeTuiSdk clientId];
```

15. 获取 **status**

```
+ (SdkStatus)status;
```

说明:

返回 sdk 的运行状态

头文件:

GeTuiSdk.h

示例:

```
[GeTuiSdk status]
```

16. 设置处理显示的 **AlertView** 是否随屏幕旋转

```
+ (void)setAllowedRotateUiOrientations:(NSArray *)orientations;
```

参数:

orientations:支持的屏幕方向列表，具体值请参照 `UIInterfaceOrientation`（From iOS SDK）

说明:

设置为与您的应用中 `shouldAutorotateToInterfaceOrientation` 中相同的参数，这样不会因为推送消息弹框导致您的 UI 意外旋转到不希望的模式。



头文件:
GeTuiSdk.h

示例:
`[GeTuiSdk setAllowedRotateUiOrientations:@[[NSNumber numberWithInt:UIInterfaceOrientationPortrait]]];`

注意:
如果不关心 UI 方向，不要调用这个方法。

17. 设置关闭推送模式

+ (void)setPushModeForOff:(BOOL)isValue;

参数:

isValue:设置关闭推送模式功能，YES.关闭推送 NO.开启推送。

该方法会回调 - (void)GeTuiSdkDidSetPushMode:(BOOL)isModeOff
error:(NSError *)error;

说明:

客户端可以关闭服务器的推送功能，当该功能为“YES”后，服务器将不给该客户端发推送消息，默认该功能为“NO”。

头文件:
GeTuiSdk.h

示例:
`[GeTuiSdk setPushModeForOff:NO];`

18. 上行第三方自定义回执 actionid

+ (BOOL)sendFeedbackMessage:(NSInteger)actionId taskId:(NSString *)taskId msgId:(NSString *)msgId;

参数:

actionId: 用户自定义的 actionid, int 类型, 取值 90001-90999。

taskId: 下发任务的任务 ID。

msgId: 下发任务的消息 ID。

返回值: BOOL, YES 表示该命令已经提交, NO 表示该命令未提交成功。注: 该结果不代表服务器收到该条命令

该方法需要在回调方法



“GeTuiSdkDidReceivePayload:andTaskId:andMessageId:andOffLine:fromApplication:” 使用。

说明:

上行用户在一个推开发者网站中的自定义事件，统计 App 中接受到消息数。

头文件:

GeTuiSdk.h

示例:

```
// 汇报个推自定义事件
[GeTuiSdk sendFeedbackMessage:90001 taskId:taskId msgId:aMsgId];
```

◆ SDK Delegate 回调接口:

1. SDK 登入成功返回 clientId

```
- (void)GeTuiSdkDidRegisterClient:(NSString *)clientId;
```

参数:

clientId:标识用户的 clientId

说明:

启动 GeTuiSdk 后，SDK 会自动向个推服务器注册 SDK，当成功注册时，SDK 通知应用注册成功。

注意:

注册成功仅表示推送通道建立，如果 appid/appkey/appSecret 等验证不通过，依然无法接收到推送消息，请确保验证信息正确。

2. SDK 通知收到个推推送消息

```
- (void)GeTuiSdkDidReceivePayload:(NSString *)payloadId andTaskId:(NSString *)taskId andMessageId:(NSString *)aMsgId andOffLine:(BOOL)offLine fromApplication:(NSString *)appId;
```

参数:

payloadId:代表推送消息的唯一 id



taskId:推送消息的任务 id

aMsgId: 推送消息的 messageid

offLine:是否是离线消息, YES.是离线消息

appId:应用的 appId

说明:

SDK 会将推送消息在本地数据库中保留 5 天, 请及时取出 (See `retrivePayloadById:`), 取出后消息将被删除。

3. SDK 通知发送上行消息结果

```
- (void)GeTuiSdkDidSendMessage:(NSString *)messageId result:(int)result;
```

参数:

messageId:sendMessage:error:返回的 id

result:成功返回 0

说明:

当调用 `sendMessage:error:`接口时, 消息推送到个推服务器, 服务器通过该接口通知 sdk 到达结果, result 为 0 说明消息发送成功。

4. SDK 遇到错误消息返回 error

```
- (void)GeTuiSdkDidOccurError:(NSError *)error;
```

参数:

error: SDK 内部发生错误, 通知第三方, 返回错误。

5. SDK 运行状态通知

```
- (void)GeTuiSdkDidNotifySdkState:(SdkStatus)aStatus;
```

参数:

aStatus: 返回 SDK 运行状态

说明:

状态定义如下

```
typedef enum {  
    SdkStatusStarting, // 正在启动  
    SdkStatusStarted,  // 启动
```



```
    SdkStatusStoped    // 停止  
} SdkStatus;
```

6. SDK 设置关闭推送模式回调

```
- (void)GeTuiSdkDidSetPushMode:(BOOL)isModeOff error:(NSError *)error;
```

参数:

isModeOff: 关闭模式, YES.服务器关闭推送功能 NO.服务器开启推送功能

error: 错误回调, 返回设置时的错误信息

UnAuthenticated: SDK 未认证

AppKeyOrAppSecretInvalid: appKey 或 appSecret 无效

AppIdInvalid: appId 无效

CidMissing: cid 丢失

PushModeError: 模式错误

FrequencyOverrun: 频率超限

ConnectionLost: 连接丢失/离线

TimeoutError: 请求超时 (15s)

CallFrequentlyError: 调用太频繁(1s 只能调用一次)

说明:

当调用 setPushModeForOff:接口时, 设置推送到个推服务器, 服务器通过该接口通知 sdk 到达结果, error 为 nil 说明消息发送成功。