

# iOS SDK API接口文档

本章介绍了iOS SDK API的相关接口，如有更多疑问，[请联系我们]

## 1.启动个推SDK

```
+ (void)startSdkWithAppId:(NSString *)appid appKey:(NSString *)appKey appSecret:(NSString *)appSecret delegate:(id<GeTuiSdkDelegate>)delegate;
```

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

返回:

- 无

说明:

- 启动SDK，该方法需要在主线程中调用
- 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（已弃用）

```
+ (void)stopSdk;
```

参数:

- 无

返回:

- 无

说明:

- 停止SDK，并且释放资源，该方法需要在主线程中调用。

头文件:

- GeTuiSdk.h

示例:

```
[GeTuiSdk stopSdk];// 停止SDK
```

### 3.销毁个推SDK

```
+ (void)destroy;
```

参数:

- 无

返回:

- 无

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;
```

参数:

- deviceToken:APNS返回的client标识。

返回:

- 无

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deviceToken数据后，调用这个接口提交deviceToken，具体实现请参考Demo工程。

头文件:

- GeTuiSdk.h

示例:

```
/** 远程通知注册成功委托 */
- (void)application:(UIApplication *)application didRegisterForRemoteNotifications
WithDeviceToken:(NSData *)deviceToken {
    NSString *token = [[deviceToken description] stringByTrimmingCharactersInSet:[
    NSStringCharacterSet characterSetWithCharactersInString:@"<>"]];
    token = [token stringByReplacingOccurrencesOfString:@" " withString:@""];
    NSLog(@"\n>>>[DeviceToken Success]:%@", token);

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

## 5.根据payloadId取回推送消息（已弃用）

```
+ (NSData *)retrivePayloadById:(NSString *)payloadId;
```

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:

- NSData\* : payload 数据, 无法取到消息内容，返回nil

说明:

- 1.通过payloadId 取回收到的Payload数据，具体事项可参考Demo
- 2.该接口需要配合 GeTuiSdkDelegate 中

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

- 3. 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：

```
//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;
```

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：

- BOOL: 返回是否设置成功,成功返回YES,失败返回NO。 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。

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;
```

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。

返回：

- 无

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。

返回:

- 无

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

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

参数:

- body: 发送的数据
- error: 如果发送错误给出错误消息

返回:

- NSString\* : 如果发送成功返回messageid，发送失败返回nil。

说明:

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

头文件:

- GeTuiSdk.h

示例:

```
NSError *aError = nil;  
NSData *sendData = [@"sendMessage" dataUsingEncoding:NSUTF8StringEncoding];  
[GeTuiSdk sendMessage:sendData error:&aError]
```

## 10.是否允许SDK 后台运行

```
+ (void)runBackgroundEnable:(BOOL) isEnabled;
```

参数:

- isEnabled : 设置是否允许后台运行。

返回:

- 无

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;
```

参数:

- 无

返回:

- 无

说明: \* 该方法需要在主线程中调用 \* 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;
```

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

返回:

- 无

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

头文件:

- GeTuiSdk.h

示例:

```
[GeTuiSdk lbsLocationEnable:YES andUserVerify:YES];
```

## 13.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。

返回:

- 无

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

头文件:

- GeTuiSdk.h

示例:

```
[GeTuiSdk setPushModeForOff:NO];
```



## 14.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:

- BOOL:返回BOOL类型, 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

头文件:

- GeTuiSdk.h

示例:

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

## 15.获取SDK 系统版本号

```
+ (NSString *)version;
```

参数:

- 无

返回:

- NSString\* : 版本信息

说明:

- 返回系统版本号

头文件:

- GeTuiSdk.h

示例:

---

```
[GeTuiSdk version];
```

## 16. 获取clientid

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

参数:

- 无

返回:

- NSString\* : clientId 字符串

说明:

- sdk登入成功后返回clientId

头文件:

- GeTuiSdk.h

示例:

```
[GeTuiSdk clientId];
```

## 17. 获取status

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

参数:

- 无

返回:

- SdkStatus : sdk状态
- SdkStatusStarting 正在启动
- SdkStatusStarted 启动
- SdkStatusStoped 停止

说明:

- 返回sdk的运行状态

头文件:

- GeTuiSdk.h

示例:

```
[GeTuiSdk status];
```

## 18.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)

返回:

- 无

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。

## 19.设置角标

```
+ (void)setBadge:(NSInteger)value;
```

参数:

- value:设置的角标值, 取值范围:[0, 99999]。

返回:

- 无

说明: \*设置角标功能,同步服务器角标计数 \*APP角标显示需额外调用[[UIApplication sharedApplication] setApplicationIconBadgeNumber:badge];进行设置。

头文件:

- GeTuiSdk.h

示例:

```
[GeTuiSdk setBadge:5];

//如果需要角标显示需要调用系统方法设置
[[UIApplication sharedApplication] setApplicationIconBadgeNumber:5];
```

## 20.复位角标

```
+ (void)resetBadge;
```

参数:

- 无

返回:

- 无

说明: \* 复位角标功能,设置服务器角标计数为0。 \* APP角标复位需额外调用[[UIApplication sharedApplication] setApplicationIconBadgeNumber:0];进行设置。

头文件:

- GeTuiSdk.h

示例:

```
//重置角标
[GeTuiSdk resetBadge];

//如果需要角标清空需要调用系统方法设置
[[UIApplication sharedApplication] setApplicationIconBadgeNumber:0];
```

## 21.GeTuiSdkDelegate 回调

## 1.SDK登入成功返回clientId

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

参数:

- clientId:返回SDK登入成功后获取到的clientId, 唯一标示用户

返回:

- 无

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通知应用注册成功获取到ClientId。
- 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。

头文件:

- GeTuiSdk.h

示例:

```
- (void)GeTuiSdkDidRegisterClient:(NSString *)clientId {  
    NSLog(@"[GetuiSDK ClientId]:%@", clientId);  
}
```

## 2.SDK接收个推推送的透传消息 (已弃用)

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

参数:

- payloadId:接收到的透传数据消息id
- taskId:推送消息的任务id,唯一标示透传任务id
- aMsgId:推送消息的messageid,唯一标示当前消息的id
- offLine:是否是离线消息, YES.是离线消息
- appId:下发消息的应用ID

返回:

- 无

说明:

- 该方法已经弃用,请参考(3)接入,方法替换为:

```
- (void)GeTuiSdkDidReceivePayloadData:(NSData *)payloadData andTaskId:(NSString *)  
taskId andMsgId:(NSString *)msgId andOffLine:(BOOL)offLine fromGtAppId:(NSString *)  
appId;
```

头文件:

- GeTuiSdk.h

示例:

```
/** SDK收到透传消息回调 (该方式已经弃用) */  
- (void)GeTuiSdkDidReceivePayload:(NSString *)payloadId andTaskId:(NSString *)task  
Id andMessageId:(NSString *)aMsgId andOffLine:(BOOL)offLine fromApplication:(NSStr  
ing *)appId {  
    NSData *payload = [GeTuiSdk retrievePayloadById:payloadId];  
    NSString *payloadMsg = nil;  
    if (payload) {  
        payloadMsg = [[NSString alloc] initWithBytes:payload.bytes  
                                                    length:payload.length  
                                                    encoding:NSUTF8StringEncoding];  
    }  
    NSLog(@"Payload Msg:%@", payloadMsg);  
    // 汇报个推自定义事件  
    [GeTuiSdk sendFeedbackMessage:90001 taskId:taskId msgId:aMsgId];  
}
```

### 3.SDK接收个推推送的透传消息

```
- (void)GeTuiSdkDidReceivePayloadData:(NSData *)payloadData andTaskId:(NSString *)  
taskId andMsgId:(NSString *)msgId andOffLine:(BOOL)offLine fromGtAppId:(NSString *)  
appId;
```

参数:

- payloadData:接收到的透传数据
- taskId:推送消息的任务id,唯一标示透传任务id
- msgId:推送消息的messageid,唯一标示当前消息的id
- offLine:是否是离线消息, YES.是离线消息
- appId:下发消息的应用ID

返回:

- 无

说明:

- 接收服务器下发的Payload数据, 如果下发时ClientId离线, 则 offLine为True。
- 如需自定义回执, 须在接收到消息后调用sendFeedbackMessage方法提交回执, 以便数据跟踪。

头文件:

- GeTuiSdk.h

示例:

```
- (void)GeTuiSdkDidReceivePayloadData:(NSData *)payloadData andTaskId:(NSString *)
taskId andMsgId:(NSString *)msgId andOffLine:(BOOL)offLine fromGtAppId:(NSString *)
)appId {
    NSString *payloadMsg = nil;
    if (payloadData) {
        payloadMsg = [[NSString alloc] initWithBytes:payloadData.bytes
                                                    length:payloadData.length
                                                    encoding:NSUTF8StringEncoding];
    }
    NSLog(@"Payload Msg:%@", payloadMsg);
    // 汇报个推自定义事件
    [GeTuiSdk sendFeedbackMessage:90001 taskId:taskId msgId:msgId];
}
```

#### 4.SDK通知发送上行消息结果, 收到sendMessage消息回调

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

参数:

- messageId:返回的消息ID, 与sendMessage返回的messageid对应。
- result:处理结果, 成功返回1, 失败返回0

返回:

- 无

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1 说明消息发送成功,0为失败

示例:

---

```

/** SDK收到sendMessage消息回调 */
- (void)GeTuiSdkDidSendMessage:(NSString *)messageId result:(int)result {
    NSString *record = [NSString stringWithFormat:@"Received sendmessage:%@ result :%d", messageId, result];
    NSLog(@"GeTuiSdkDidSendMessage : %@", record);
}

```

## 5.SDK设置关闭推送模式回调

```

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

```

参数:

- isModeOff:是否为关闭模式, 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

返回:

- 无

说明:

- 调用SDK setPushModeForOff 方法回调设置结果, 如果设置为关闭模式服务器不会下发APNS和透传消息。

示例:

```

- (void)GeTuiSdkDidSetPushMode:(BOOL)isModeOff error:(NSError *)error {
    if (error) {
        NSString* errorInfo = [NSString stringWithFormat:@">>>[SetModeOff error]:%@", [error localizedDescription]];
        NSLog(@"GeTuiSdkDidSetPushMode with error : %@", errorInfo);
        return;
    }

    NSString* settingInfo = [NSString stringWithFormat:@">>>[GexinSdkSetModeOff]:%@", isModeOff ? @"开启" : @"关闭"];
    NSLog(@"%@", settingInfo);
}

```

## 6.SDK运行状态通知

```

- (void)GeTuiSdkDidNotifySdkState:(SdkStatus)aStatus;

```



参数:

- aStatus:SDK运行的状态。 SdkStatusStarting正在启动 SdkStatusStarted启动 SdkStatusStoped停止

返回:

- 无

说明:

- 返回SDK运行状态, SDK初始化到ClientId登入成功为正在启动状态, ClientId登入成功为启动状态, SDK调用Destory销毁SDK为停止状态。

示例:

```
/** SDK运行状态通知 */
- (void)GeTuiSdkDidNotifySdkState:(SdkStatus)aStatus {
    NSLog(@"[GetuiSdk Status]:%u", aStatus);
}
```

## 7.SDK运行遇到错误消息返回Error

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

参数:

- error: SDK内部发生错误, 通知第三方, 返回错误信息

返回:

- 无

说明:

- sdk 初始化异常或者运行时出现错误, 由该接口返回错误信息。

示例:

```
/** SDK遇到错误回调 */
- (void)GeTuiSdkDidOccurError:(NSError *)error {
    NSString* sdkErrorInfo = [NSString stringWithFormat:@">>>[GexinSdk error]:%@",
    [error localizedDescription]];
    NSLog(@"sdkErrorInfo : %@", sdkErrorInfo);
}
```