API接口文档

1. 初始化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SDK

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

参数:

- 无

返回:

- 无

说明:

- 销毁SDK，并且释放资源。销毁后SDK前后台切换将不再启动SDK。

头文件:

- GeTuiSdk.h

示例:

```
[GeTuiSdk destroy];// 销毁SDK
```

3. 注册 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:[
NSCharacterSet characterSetWithCharactersInString:@"<>"]];
    token = [token stringByReplacingOccurrencesOfString:@" " withString:@""];
    NSLog(@"\n>>>[DeviceToken Success]:%@\n\n", token);

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

4. 注册 VoipToken

```
+ (BOOL)registerVoipToken:(NSString *)voipToken;
```

参数:

- voipToken：苹果 VOIP 服务返回的 Token 标识。

返回:

- 无

说明:

- 在iOS8，苹果引入了一个新的被用于 VoIP APP 类型的推送消息，这可以使用户收到一个 VOIP 推送时唤醒 APP。有了这种新的推送，开发者们不需要让 APP 持续保持后台运行。当需要保持这个 APP 在后台运行时，这个 VOIP 消息类型将会在后台唤醒 APP，并执行制定的代码（在弹出消息给用户之前）。个推 VOIP 服务通过该接口绑定 VoipToken 和 个推 ClinetID 的关系，从而进行 VOIP 推送服务。

头文件:

- GeTuiSdk.h

示例:

```
- (void)pushRegistry:(PKPushRegistry *)registry didUpdatePushCredentials:(PKPushCr
edentials *)credentials forType:(NSString *)type {
    // Register VoIP push token (a property of PKPushCredentials) with server
    NSString *voiptoken = [credentials.token.description stringByTrimmingCharacter
sInSet:[NSCharacterSet characterSetWithCharactersInString:@"<>"]];
    voiptoken = [voiptoken stringByReplacingOccurrencesOfString:@" " withString:@"
"];
    //向个推服务器注册 VoipToken
    [GeTuiSdk registerVoipToken:voiptoken];
}
```

5. 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、+-*.的组合（不支持空格）。

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];
}
```

6. 绑定用户别名

```
+ (void)bindAlias:(NSString *)alias andSequenceNum:(NSString *)aSn;
```

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。
- aSn： 绑定序列码,不为nil。

返回：

- 无

说明：

- 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；
- 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；
- 该接口在一天内最多调用50次；
- aSn 不为nil，绑定回调将带回该sn标示。

头文件：

- GeTuiSdk.h

示例：

```
[GeTuiSdk bindAlias:@"个推" andSequenceNum:@"seq-1"];
```

7. 解绑用户别名

```
+ (void)unbindAlias:(NSString *)alias andSequenceNum:(NSString *)aSn andIsSelf:(BO
OL) isSelf;
```

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。
- aSn： 绑定序列码,不为nil。
- isSelf ： 是否只对当前cid有效，如果是true，只对当前cid做解绑；如果是false，对所有绑定该别名的cid列表做解绑

返回:

- 无

说明:

- 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；
- sn 不为nil，解绑回调将待会该sn标示。

头文件:

- GeTuiSdk.h

示例:

```
[GeTuiSdk unbindAlias:@"个推" andSequenceNum:@"seq-2" andIsSelf:YES];
```

8. 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; 具体可参考 GeTuiSdkDelegate 回调中的说明。

头文件:

- GeTuiSdk.h

示例:

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

9. 是否允许SDK 后台运行

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

参数:

- isEnable：设置是否允许后台运行。
- isEnable取值：YES：允许，NO：不允许。

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];
```

10. 恢复SDK运行（已弃用）

```
+ (void)resume;
```

11. 是否启用SDK地理围栏功能

```
+ (void)lbsLocationEnable:(BOOL)isEnable andUserVerify:(BOOL)isVerify;
```

参数:

- isEnable：设置地理围栏功能是否运行，YES：允许，NO：不允许。默认为 NO。

- isVerify：设置是否SDK主动弹出用户定位请求，YES：允许，NO：不允许。默认为 NO。

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];
```

12. 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;具体可参考 GeTuiSdkDelegate 回调中的说明。

头文件：

- GeTuiSdk.h

示例:

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

13. 上行第三方自定义回执

```
+ (BOOL)sendFeedbackMessage:(NSInteger)actionId andTaskId:(NSString *)taskId andMsgId:(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 andTaskId:taskId andMsgId:aMsgId];
```

14. 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];
```

15. 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];
```

16. 获取 SDK 状态

```
+ (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];
```

17. 设置角标

```
+ (void)setBadge:(NSUInteger)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];
```

18. 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];
```

19. 设置渠道

```
+ (void)setChannelId:(NSString *)aChannelId;
```

参数：

- aChannelId： 渠道信息

返回：

- 无

说明：

- 根据用户分发的渠道进行设置，个推后台可根据渠道信息进行统计分析，生成报表。

头文件：

- GeTuiSdk.h

示例：

```
//设置渠道信息
[GeTuiSdk setChannelId:@"GT-Channel"];
```

20. 处理远程推送消息

```
+ (void)handleRemoteNotification:(NSDictionary *)userInfo;
```

参数：

- userInfo： 接收到的APNs信息

返回：

- 无

说明：

- 接收APNs通知栏展示信息，统计有效用户点击数。

头文件：

- GeTuiSdk.h

示例：

```
- (void)application:(UIApplication *)application didReceiveRemoteNotification:(NSD
ictionary *)userInfo fetchCompletionHandler:(void (^)(UIBackgroundFetchResult))com
pletionHandler {
    // 将收到的APNs信息传给个推统计
    [GeTuiSdk handleRemoteNotification:userInfo];
    completionHandler(UIBackgroundFetchResultNewData);
}
```

21. VOIP推送消息回执

```
+ (void)handleVoipNotification:(NSDictionary *)payload;
```

参数：

- payload：VOIP 推送中携带的payload信息

返回：

- 无

说明：

- 接收 VOIP 消息后调用，统计 VOIP 推送到达数。

头文件：

- GeTuiSdk.h

示例：

```
- (void)pushRegistry:(PKPushRegistry *)registry didReceiveIncomingPushWithPayload:
(PKPushPayload *)payload forType:(NSString *)type {
    //3-1: TODO:接收 VOIP 推送中的 Payload 信息进行业务处理。
    <!--TODO: 具体 VOIP 业务处理-->
    NSLog(@"[Voip Payload]:%@,%@", payload, payload.dictionaryPayload);

    //3-2:调用个推 VOIP 回执统计接口
    [GeTuiSdk handleVoipNotification:payload.dictionaryPayload];
}
```

22.清空通知

```
+ (void)clearAllNotificationForNotificationBar;
```

参数：

- 无

返回：

- 无

说明：

- 清空下拉通知栏全部通知,并将角标置 0，即不显示角标。

头文件：

- GeTuiSdk.h

示例：

```
 //清空通知和角标
 [GeTuiSdk clearAllNotificationForNotificationBar];
```

23.APPLink 消息回执

```
+ (NSString*) handleApplinkFeedback:(NSURL* ) webUrl;
```

参数：

- webUrl: applink的 url 对象。

返回：

- 返回 weburl 中的 payload 信息。

说明：

- APPLink 用户点击回执。
- 返回下发中的透传信息，开发者可在获取到payload内容后处理具体业务逻辑。
- 该接口可统计APPLink用户点击次数。

头文件：

- GeTuiSdk.h

示例：

```
-(BOOL) application:(UIApplication *)application continueUserActivity:(NSUserActiv
ity *)userActivity restorationHandler:(void (^)(NSArray * _Nullable))restorationHa
ndler {

    //系统用 NSUserActivityTypeBrowsingWeb 表示对应的 universal HTTP links 触发
    if ([userActivity.activityType isEqualToString:NSUserActivityTypeBrowsingWeb])
 {
        NSURL* webUrl = userActivity.webpageURL;

        //处理个推APPLink回执统计
        //APPLink url 示例：https://link.applk.cn/getui?n=payload&p=mid， 其中 n=pay
load 字段存储下发的透传信息，可以根据透传内容进行业务操作。
        NSString* payload = [GeTuiSdk handleApplinkFeedback:webUrl];
        if (payload) {
            NSLog(@"个推APPLink中携带的透传payload信息: %@,URL : %@", payload, webUrl)
;
            //TODO:用户可根据具体 payload 进行业务处理
        }
    }
    return true;
}
```

24. GeTuiSdkDelegate 回调

24.1. 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);
}
```

24.2. 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];
}
```

24.3. 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。 注意: 需第三方服务器接入个推,SendMessage 到达第三方服务器后返回 1。

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);
}
```

24.4. SDK设置关闭推送模式回调

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

参数：

- 默认值：NO，服务器开启推送功能
- 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);
}
```

24.5. 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);
}
```

24.6. SDK绑定/解绑回调

```
- (void)GeTuiSdkDidAliasAction:(NSString *)action result:(BOOL)isSuccess sequenceN
um:(NSString *)aSn error:(NSError *)aError;
```

参数：

- action：回调动作类型 kGtResponseBindType 或 kGtResponseUnBindType
- isSuccess：成功返回 YES, 失败返回 NO
- aSn：返回请求的序列编号
- aError：成功返回nil, 错误返回相应error信息

错误信息：

- 30001：请求频次超限
- 30002：参数错误，别名不符合规则
- 30003：请求过滤
- 30004：操作别名失败
- 30005：clientid 未获取到调用绑定/解绑
- 30006：网络错误
- 30007：别名为空
- 30008：sequenceNum为空

返回：

- 无

示例：

```
- (void)GeTuiSdkDidAliasAction:(NSString *)action result:(BOOL)isSuccess sequenceN
um:(NSString *)aSn error:(NSError *)aError {
    if ([kGtResponseBindType isEqualToString:action]) {
        NSLog(@"绑定结果 ：%@ !, sn : %@", isSuccess ? @"成功" : @"失败", aSn);
        if (!isSuccess) {
            NSLog(@"绑定失败原因: %@", aError);
        }
    } else if ([kGtResponseUnBindType isEqualToString:action]) {
        NSLog(@"解绑结果 ：%@ !, sn : %@", isSuccess ? @"成功" : @"失败", aSn);
        if (!isSuccess) {
            NSLog(@"解绑失败原因: %@", aError);
        }
    }
}
```

24.7. 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);
}
```

25. iOS Extension SDK API 接口文档

25.1.通知个推SDK展示回执

```
+ (void)handelNotificationServiceRequest:(UNNotificationRequest *) request withCom
plete:(void (^)(void))completeBlock;
```

参数:

- request：推送送达的APNs信息。
- completeBlock：个推统计完成后回调用户操作，完成用户修改通知样式等要求。

返回:

- 无

说明:

- 接收到 APNs 通知但未展示前调用该方法，通知个推服务器APNs送达并统计，送达返回后执行用户设置的 block 操作，如修改APNs展示内容等。
- 该接口正常送达后将回调 block 执行， 如果网络异常请求超时，该接口的超时时间最多为5s。5s后将回调 block 执行。
- 最后 block 里需调用 self.contentHandler(self.bestAttemptContent); 来完成更新信息的正常展示。

头文件:

- GeTuiExtSdk.h

示例:

```
- (void)didReceiveNotificationRequest:(UNNotificationRequest *)request withContent
Handler:(void (^)(UNNotificationContent * _Nonnull))contentHandler {
    self.contentHandler = contentHandler;
    self.bestAttemptContent = [request.content mutableCopy];

    //通知个推服务器APNs信息送达
    [GeTuiExtSdk handelNotificationServiceRequest:request withComplete:^{

        // Modify the notification content here...
        self.bestAttemptContent.title = [NSString stringWithFormat:@"%@ [modified]
", self.bestAttemptContent.title];
        //返回新的通知内容,展示APNs通知。
        self.contentHandler(self.bestAttemptContent);
    }];
}
```

25.2.多媒体推送接口

```
+ (void)handelNotificationServiceRequest:(UNNotificationRequest *)request withAttachmentsComplete:(void (^)(NSArray* attachments, NSArray* errors))completeBlock;
```

参数:

- request：推送送达的APNs信息。
- completeBlock：个推统计和多媒体资源下载完成后回调用户操作，完成用户修改通知样式等要求。

返回:

- attachments：SDK 下载完成的资源数组，类型 `NSArray` 。当个资源对象为 `UNNotificationAttachment` 类型。
- errors：返回错误信息，类型NSArray。当个对象为 `NSError` 类型，描述资源下载错误信息。

说明:

1).接收到 APNs 通知后，SDK 判断是否有多媒体资源，如果多媒体资源存在则下载资源，下载完成后以 Block 方式回调返回 attachments 资源数组对象和 errors 错误数组信息。

2).多媒体大小限制：

资源	类型	大小	超时
图片	1	<=10MB	120s
音频	2	<=5MB	60s
视频	3	<=50MB	180s

3).可设置是否仅在 WIFI 下下载资源，参考服务端API。如果为True，数据网络下将不下载资源，展示通知不带附件。默认是:False,支持 WIFI 和 数据网络下下载。

4). 资源URL格式注意：不支持中文及转译地址，请使用英文合法地址。

5). 错误信息：

- 10001：多媒体地址错误,无法下载。
- 10002：不支持 Http URL 下载，修改为支持 ATS 使用 Https 地址或者将 NotificationService 配置为支持 Http 请求。
- 10003：多媒体资源超过大小限制。

头文件:

- GeTuiExtSdk.h

示例：

```
- (void)didReceiveNotificationRequest:(UNNotificationRequest *)request withContent
Handler:(void (^)(UNNotificationContent * _Nonnull))contentHandler {

    self.contentHandler = contentHandler;
    self.bestAttemptContent = [request.content mutableCopy];

    NSLog(@"----将APNs信息交由个推处理----");

    [GeTuiExtSdk handelNotificationServiceRequest:request withAttachmentsComplete:
^(NSArray *attachments, NSArray* errors) {

        //TODO:用户可以在这里处理通知样式的修改，eg:修改标题
        self.bestAttemptContent.title = [NSString stringWithFormat:@"%@ [modified]
", self.bestAttemptContent.title]; //修改展示title信息

        self.bestAttemptContent.attachments = attachments; //设置通知中的多媒体附件

        NSLog(@"个推处理APNs消息遇到错误：%@",errors); //如果APNs处理有错误，可以在这里查看
相关错误详情

        self.contentHandler(self.bestAttemptContent); //展示推送的回调处理需要放到个推回
执完成的回调中
    }];
}
```

25.3.销毁接口

```
+ (void)destory;
```

参数：

- 无

返回：

- 无

说明：

- sdk销毁，释放资源

头文件：

- GeTuiExtSdk.h

示例：

```
- (void)serviceExtensionTimeWillExpire {
    //超时时销毁个推SDK释放资源
    [GeTuiExtSdk destory];
    self.contentHandler(self.bestAttemptContent);
}
```

26. 接口删除部分

26.1. 停止个推SDK（已删除）

```
+ (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
```

26.2. 根据payloadId取回推送消息（已删除）

```
+ (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：

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

```
-(void)GeTuiSdkDidReceivePayload:(NSString *)payloadId andTaskId:(NSString *)taskI
d andMessageId:(NSString *)aMsgId andOffLine:(BOOL)offLine fromApplication:(NSStri
ng *)appId;
```

- 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 retrivePayloadById:payloadId];
NSString *payloadMsg = nil;
if (payload) {
    payloadMsg = [[NSString alloc] initWithBytes:payload.bytes
                                          length:payload.length
                                        encoding:NSUTF8StringEncoding];
}
```

26.3. 设置处理显示的AlertView是否随屏幕旋转（已删除）

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

参数：

- orientations：支持的屏幕方向列表，具体值请参照 UIInterfaceOrientation 中设置

```
typedef NS_ENUM(NSInteger, UIInterfaceOrientation) {
    UIInterfaceOrientationUnknown            = UIDeviceOrientationUnknown,
      //方向未知
    UIInterfaceOrientationPortrait           = UIDeviceOrientationPortrait,
      //屏幕直立
    UIInterfaceOrientationPortraitUpsideDown = UIDeviceOrientationPortraitUpsideDo
wn,   //屏幕直立,上下颠倒
    UIInterfaceOrientationLandscapeLeft      = UIDeviceOrientationLandscapeRight,
      //屏幕向右横置
    UIInterfaceOrientationLandscapeRight     = UIDeviceOrientationLandscapeLeft
      //屏幕向左横置
}
```

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:UIInterfaceOrie
ntationPortrait]]];
```

注意：如果不关心UI方向，不要调用这个方法。

26.4. GeTuiSdkDelegate 回调中

1. SDK接收个推推送的透传消息（已删除）

```
- (void)GeTuiSdkDidReceivePayload:(NSString *)payloadId andTaskId:(NSString *)task
Id andMessageId:(NSString *)aMsgId andOffLine:(BOOL)offLine fromApplication:(NSStr
ing *)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 *)taskId andMessageId:(NSString *)aMsgId andOffLine:(BOOL)offLine fromApplication:(NSString *)appId {
    NSData *payload = [GeTuiSdk retrivePayloadById: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];
}
```