TTkGameSDK-Lite API Document

(iOS)

Name	Date	version	remark
Yuanjun	2020-06-02	1.0.0	created

1. Description and definition

1.1 API call

Every API is called by `[TTkGameManager defaultManager]` instance

1.2 Callback

`error` is a common field for every callback class. if `error` equals to `nil`, it means the api call is success.

1.3 Callback classes:

1.3.1 Login Callback

`class:` `TTGCUserCompletionHandler`

`fields:`

`userInfo` //the user information

`error`

1.3.2 Receive Notification Callback

class: TTGCNotificationCompletionHandler

fields:

result //the notification message

error

1.3.3 Logout Callback

class: TTGCLogoutCompleteHandler

fields:

success // BOOL type，indicate operation result

error

1.3.4 Get Friends Callback

class: TTGCFriendsCompletionHandler

fields:

list //fields list, contains from 0 -> N friend object

error

1.3.5 Share Callback

class: TTGCShareCompleteHandler

fields:

success // BOOL type，indicate operation result

error

1.3.6 Payment Callback

class: TTGCOrderProgressHandler

fields:

TTGCOderStatus //Payment order status

1.3.7 Payment Result Callback

class: TTGCOrderCompletionHandler

fields:

orderInfo //the payment order object

error

1.3.8 AppVersion Info Callback

class: TTGCVersionCompletionHandler

fields:

hasNewVersion //has new version

versionInfo //the version detail information. class model <TTGCAppVersion>

error

1.3.9 Common Callback

class: TTGCCompletionHandler

fields:

result //operation result

error

1.4 Errors

TTGCPlatformErrorType_Unknow	unkown error
TTGCPlatformErrorType_AuthorizeFailed	authorized failed
TTGCPlatformErrorType_ForUserInfoFailed	request user data failed
TTGCPlatformErrorType_LoginFailed	login failed
TTGCPlatformErrorType_FriendsListFailed	get firends failed
TTGCPlatformErrorType_MessageSendFailed	send message failed
TTGCPlatformErrorType_PayFailed	payment failed

TTGCPlatformErrorType_ShareFailed	share failed
TTGCPlatformErrorType_NotInstall	app is not installed
TTGCPlatformErrorType_NotNetWork	network exception
TTGCPlatformErrorType_SourceError	third error
TTGCPlatformErrorType_NoPermission	no permissions
TTGCPlatformErrorType_UnauthorizedRequest	unauthorized request, need login.

2. Setup Initialize (must)

2.1 Launch

Methods in AppDelegate are implement like below:

2.1.1 Setup Configuration:

The launch animation for SDK-1.0.5 has been removed and the specific launch animation needs to be implemented by the game developers.

```
- (BOOL)application:(UIApplication *)application
didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
    // set SDK Api Enviroment
    [[TTkGameManager  defaultManager] setServerDomainTest];
   // set Pay Enviroment
    [[TTkGameManager  defaultManager] setPayEnvironmentTest];
     // open log
    [[TTkGameManager  defaultManager] openLogInfo];

    // launch SDK
    NSString *appId = @"xxxxx";
    NSString *appSecret = @"xxxxxx";
   NSString *gameid = @"xxxxxx";
    [[TTkGameManager  defaultManager] setupWithAppId:appId Secret:appSecret
GameId:gameid Application:application Options:launchOptions];
    return YES;
```

```
}
```

2.1.2 External Callback:

```
- (BOOL)application:(UIApplication *)app openURL:(NSURL *)url
options:(NSDictionary<UIApplicationOpenURLOptionsKey,id> *)options {
    return [[TTkGameManager  defaultManager] application:app openURL:url
options:options];
}
```

2.1.3 App Events:

```
- (void)applicationDidEnterBackground:(UIApplication *)application {
    [[TTkGameManager  defaultManager] applicationDidEnterBackground:application];
}
- (void)applicationWillEnterForeground:(UIApplication *)application {
    [[TTkGameManager  defaultManager] applicationWillEnterForeground:application];
}
- (void)applicationDidBecomeActive:(UIApplication *)application {
    [[TTkGameManager  defaultManager] applicationDidBecomeActive:application];
}
- (void)applicationWillTerminate:(UIApplication *)application {
    [[TTkGameManager  defaultManager] applicationWillTerminate:application];
}
```

2.1.4 Push Register:

```
- (void)application:(UIApplication *)application
didRegisterForRemoteNotificationsWithDeviceToken:(NSData *)deviceToken {
    [[TTkGameManager  defaultManager] registerDeviceToken:deviceToken];
}
- (void)application:(UIApplication *)application
didReceiveRemoteNotification:(NSDictionary *)userInfo fetchCompletionHandler:(void
(^)(UIBackgroundFetchResult))completionHandler {
    [[TTkGameManager  defaultManager] application:application
didReceiveRemoteNotification:userInfo];
}
```

2.1.5 Universal Link:

```
- (BOOL)application:(UIApplication *)application continueUserActivity:(NSUserActivity
*)userActivity restorationHandler:(void (^)(NSArray<id<UIUserActivityRestoring>> *
_Nullable))restorationHandler {
    return [[TTkGameManager  defaultManager] application:application
continueUserActivity:userActivity restorationHandler:restorationHandler];
}
```

2.2 Check App Version

After SDK is launched, be sure to check the version to ensure that the current version is available. If you need to force upgrade, you can go to the appstore to download the latest version, otherwise the SDK function may not be available.

example

```
- (void)checkVersion {
    [[TTkGameManager  defaultManager] checkAppVersionCompletion:^(BOOL hasNewVersion,
TTGCAppVersionModel * _Nullable versionInfo, NSError * _Nullable error) {
        if (hasNewVersion) {
            BOOL forceUpdate = versionInfo.forceUpdate;
            if (forceUpdate) {
                // If you need to force update, you cannot continue to use it, and you need
to jump to the appstore to update.
                // Open appstore
                [[TTkGameManager  defaultManager]
openAppStoreWithStoreIdentifier:@"xxx"];
            } else {
                // If update is not required, you can optionally prompt for an upgrade.
            }
        }
    }];
}
```

2.2.1 Check Version

Method	checkAppVersionCompletion

Callback	`TTGCVersionCompletionHandler` `hasNewVersion` // has new version <BOOL> `versionInfo` // the version detail information. class model <TTGCAppVersion> `build` // app build code `version` // app new version `versionInfo` // the version information `forceUpdate` // need to force upgrade <BOOL> `currentVersionCode` // current version code

2.2.2 Open AppStore

`Method`	`openAppStoreWithStoreIdentifier`
Params	`identifier` //App Store item identifier (NSNumber) of the product

3. API: Configuration

3.1 Enbale Log Output

```
[[TTkGameManager  defaultManager] openLogInfo];
```

3.2 Disable Log Output(default)

```
[[TTkGameManager  defaultManager] closeLogInfo];
```

3.3 Set Test Environment

```
[[TTkGameManager  defaultManager] setServerDomainTest];
```

3.4 Set Production Environment(default)

```
[[TTkGameManager  defaultManager] setServerDomainProduction];
```

3.5 Set Sandbox Payment

```
[[TTkGameManager  defaultManager] setPayEnvironmentTest];
```

3.6 Set Production Payment(default)

```
[[TTkGameManager  defaultManager] setPayEnvironmentProduction];
```

4. API: Login&User Info

User must agree the User Agreement & Privacy before do Login action

4.1 Show Agreement View

```
- (IBAction)agreement:(id)sender {
    [[TTkGameManager  defaultManager] showAgreementView];
}
```

4.2 Login with ToTok

Method	loginWithGameCenterCompletion
Callback	TTGCUserCompletionHandler

example:

```
- (IBAction)totokLogin:(id)sender {
    TTGCHUD_NO_Stop(@"login...")
    __weak __typeof(self)weakSelf = self;
    [[TTkGameManager  defaultManager] loginWithTTkCompletion:^(id  _Nonnull userInfo,
NSError * _Nonnull error) {
```

```
        if (!error) {
            //login success
            TTGCHUD_SUCCESS(@"success");
            [weakSelf closeLoginView];
        } else {
            //error info
            TTGCHUD_HINT([error.userInfo objectForKey:@"errorMsg"]);
        }
    }];
}
```

4.3 Login with Third-Party

4.3.1 Field Description

TTGCAppleIDLoginParaModel

Params	Note
user	<NSString> user of appleIDCredential
familyName	<NSString> fullName.familyName of appleIDCredential (optional)
givenName	<NSString> fullName.givenName of appleIDCredential (optional)
identityToken	<NSData> identityToken of appleIDCredential

TTGCGameCenterLoginParaModel

Params	Note
playerID	<NSString> localPlayer.playerID of [GKLocalPlayer localPlayer]
displayName	<NSString> localPlayer.playerID of [GKLocalPlayer localPlayer]
publicKeyUrl	<NSURL> publicKeyUrl of verification
accessToken	<NSData> accessToken of verification
signature	<NSData> signature of verification
salt	<NSData> salt of verification
timestamp	<uint64_t> timestamp of verification
photoData	<NSData> user photo data (optional)

TTGCFacebookLoginParaModel

Params	Note
accessToken	<NSString> result.token.tokenString of FacebookLoginResult
accessExpire	<NSDate> result.token.expirationDate of FacebookLoginResult

TTGCTwitterLoginParaModel

Params	Note
authToken	<NSString> The authorization token.
authTokenSecret	<NSString> The authorization token secret.
userId	<NSString> The user ID associated with the access token.

4.3.2 Method

Method	loginWithThirdPartParaModel:Completion:
Params	
model	<id> type : TTGCGameCenterLoginParaModel TTGCAppleIDLoginParaModel TTGCFacebookLoginParaModel TTGCTwitterLoginParaModel
completion	TTGCUserCompletionHandler

example

```
//appleid
- (void)authorizationController:(ASAuthorizationController *)controller
didCompleteWithAuthorization:(ASAuthorization *)authorization  API_AVAILABLE(ios(13.0))
{

    __weak __typeof(self)weakSelf = self;
    if ([authorization.credential isKindOfClass:[ASAuthorizationAppleIDCredential
class]]) {
```

```
        ASAuthorizationAppleIDCredential *appleIDCredential =
(ASAuthorizationAppleIDCredential *)authorization.credential;


        TTGCAppleIDLoginParaModel *model = [[TTGCAppleIDLoginParaModel alloc] init];
        model.user = appleIDCredential.user;
        if (appleIDCredential.fullName) {
            model.familyName = appleIDCredential.fullName.familyName;
            model.givenName = appleIDCredential.fullName.givenName;
        }
        model.identityToken = appleIDCredential.identityToken;
        [[TTkGameManager defaultManager] loginWithThirdPartParaModel:model
Completion:^(id  _Nullable userInfo, NSError * _Nullable error) {
            if (!error) {
                //login success
                TTGCHUD_SUCCESS(@"success");
                [weakSelf closeLoginView];
            } else {
                //error info
                TTGCHUD_HINT([error.userInfo objectForKey:@"errorMsg"]);
            }
        }];
    }
}


//gamecenter
- (void)authenticatedPlayer:(GKLocalPlayer *)localPlayer {
    __weak __typeof(self)weakSelf = self;
    __block NSData *photoData = nil;
    [localPlayer loadPhotoForSize:GKPhotoSizeNormal withCompletionHandler:^(UIImage *
_Nullable photo, NSError * _Nullable error) {
        if (photo != nil) {
            photoData = UIImageJPEGRepresentation(photo, 0.3);
        }
```

```
        [localPlayer generateIdentityVerificationSignatureWithCompletionHandler:^(NSURL
* _Nullable publicKeyUrl, NSData * _Nullable signature, NSData * _Nullable salt, uint64_t
timestamp, NSError * _Nullable error) {
            if (error) {
                NSLog(@"%@",error.description);
                return;
            }

            TTGCGameCenterLoginParaModel *model = [[TTGCGameCenterLoginParaModel alloc]
init];
            model.playerID = localPlayer.playerID;
            model.displayName = localPlayer.displayName;
            model.publicKeyUrl = publicKeyUrl;
            model.signature = signature;
            model.salt = salt;
            model.timestamp = timestamp;
            model.photoData = photoData;

            [[TTkGameManager defaultManager] loginWithThirdPartParaModel:model
Completion:^(id  _Nullable userInfo, NSError * _Nullable error) {
                if (!error) {
                    //login success
                    TTGCHUD_SUCCESS(@"success");
                    [weakSelf closeLoginView];
                } else {
                    //error info
                    TTGCHUD_HINT([error.userInfo objectForKey:@"errorMsg"]);
                }
            }];
        }];
    }];
}

//facebook
- (IBAction)facebookLogin:(id)sender {
    TTGCHUD_NO_Stop(@"login...")
```

```
    __weak __typeof(self)weakSelf = self;
    FBSDKLoginManager *manager = [[FBSDKLoginManager alloc] init];
        if ([FBSDKAccessToken currentAccessToken]) {
             [manager logOut];
        }
        [manager logInWithPermissions:@[@"email",@"public_profile"]
fromViewController:self handler:^(FBSDKLoginManagerLoginResult * _Nullable result,
NSError * _Nullable error) {

            if (error) {

            } else {
                TTGCFacebookLoginParaModel *model = [[TTGCFacebookLoginParaModel alloc]
init];
                model.accessToken = result.token.tokenString;
                model.accessExpire = result.token.expirationDate;
                [[TTkGameManager defaultManager] loginWithThirdPartParaModel:model
Completion:^(id  _Nullable userInfo, NSError * _Nullable error) {
                    if (!error) {
                        //login success
                        TTGCHUD_SUCCESS(@"success");
                        [weakSelf closeLoginView];
                    } else {
                        //error info
                        TTGCHUD_HINT([error.userInfo objectForKey:@"errorMsg"]);
                    }
                }];
            }
        }];
}

//twitter
- (IBAction)twitterLogin:(id)sender {
    TTGCHUD_NO_Stop(@"login...")
    __weak __typeof(self)weakSelf = self;
```

```
    [[Twitter sharedInstance] logInWithViewController:self completion:^(TWTRSession
*session, NSError *error) {
      if (session) {
          TTGCTwitterLoginParaModel *model = [[TTGCTwitterLoginParaModel alloc] init];
          model.authToken = session.authToken;
          model.authTokenSecret = session.authTokenSecret;
          model.userID = session.userID;

          [[TTkGameManager defaultManager] loginWithThirdPartParaModel:model
Completion:^(id  _Nullable userInfo, NSError * _Nullable error) {
              if (!error) {
                  //login success
                  TTGCHUD_SUCCESS(@"success");
                  [weakSelf closeLoginView];
              } else {
                  //error info
                  TTGCHUD_HINT([error.userInfo objectForKey:@"errorMsg"]);
              }
          }];
      } else {
          NSString *str = [NSString stringWithFormat:@"%@",error.description];
          TTGCHUD_HINT(str);
      }
    }];
}
```

4.4 Login with Guest

Method	guestLoginCompletion
Callback	TTGCUserCompletionHandler

example

```
- (void)guestLogin {
```

```
    TTGCHUD_NO_Stop(@"login...")
    __weak __typeof(self)weakSelf = self;
    [[TTkGameManager  defaultManager] guestLoginCompletion:^(id  _Nonnull userInfo,
NSError * _Nonnull error) {
        if (!error) {
            //login success
            TTGCHUD_HINT(@"success")
            [weakSelf closeLoginView];
        } else {
            //error info
            TTGCHUD_Stop
        }
    }];
}
```

4.5 Get User Info

Method	userInfoCompletion	
Callback	TTGCUserCompletionHandler	
Params	TTGCUserModel	
	userType	user type (TTGCLoginType by 4.6)
	userId	user's indentifier
	nickname	nickname
	photoUrl	head image url
	pushOff	push switch 0:push switch on 1:push switch off

example

```
- (void)getUserInfo {
    TTGCHUD_NO_Stop(@"")
    [[TTkGameManager  defaultManager] userInfoCompletion:^(id  _Nullable userInfo,
NSError * _Nullable error) {
        if (!error) {
            TTGCHUD_HINT(@"success")
            TTGCUserModel *model = userInfo;
```

```
            [self freshUserInfo:model];
        } else {
            //error info
            TTGCHUD_HINT([error.userInfo objectForKey:@"errorMsg"]);
        }
    }];
}
```

4.6 Get Login Type

Method	loginType
Return	TTGCLoginType

Definition:

Enum	TTGCLoginType
Description	Enum type for Login
Enum Values	
TTGCLoginType_unloggedIn	not login
TTGCLoginType_ToTok	ToTok user
TTGCLoginType_GameCenter	GameCenter player
TTGCLoginType_Facebook	Facebook user
TTGCLoginType_Guest	ToTokGame guest
TTGCLoginType_Twitter	Twitter user

example:

```
- (void)showLoginView {
    if ([[TTkGameManager  defaultManager] loginType] == TTGCLoginType_unloggedIn) {
        //unlogged in and show login UI
        [self backgroundUI];
        [self loginView];
    } else {
        dispatch_after(dispatch_time(DISPATCH_TIME_NOW, (int64_t)(2.5 * NSEC_PER_SEC)),
dispatch_get_main_queue(), ^{
```

```
            [self getUserInfo];
        });
    }
}
```

4.7 Notification of offline

Method	accountKickedNotification
Callback	TTGCAccountKickedHandler

example:

```
    [[TTkGameManager  defaultManager] accountKickedNotification:^(NSString * _Nullable information) {
        //Your account has been logged in on another device, please login again.
        TTGCHUD_HINT(information);
        [self showLoginView];
    }];
```

5. API：Logout

Method	logout
Callback	TTGCLogoutCompleteHandler

example:

```
- (IBAction)logout:(id)sender {
    TTGCHUD_NO_Stop(@"logout...")
    [[TTkGameManager  defaultManager] logout:^(BOOL success, NSError * _Nullable error) {
        if (success) {
            TTGCHUD_HINT(@"success")
            [self showLoginView];
        } else {
            //error info
```

```
            NSLog(@"%@",error.userInfo);

            TTGCHUD_HINT([error.userInfo objectForKey:@"errorMsg"]);

        }

    }];

}
```

6. API: Game Payment

6.1 Product Purchase

Order Status Definition:

TTGCOrderStatus_ProductQuerying	query product
TTGCOrderStatus_Generating	order init
TTGCOrderStatus_ProductPurchasing	order in trading progress
TTGCOrderStatus_ProductPurchased	order purchased
TTGCOrderStatus_ReceiptChecking	order verifying

Method	buyProductWithSKU: Progress: Completion:
Params	
sku	String,product id
Progress	TTGCOrderProgressHandler
Completion	TTGCOrderCompletionHandler class TTGCOrderModel orderId orderRef //transaction_id channel //1 env //0=sandbox, 1=production sku //product id

	amount //price createTime payTime cancelTime status //0=order created, 1=order payed, 2=order reset

remark:

the Game Server should send the product to user's account after the order be payed, then check the order status by server api, Order status check is necessary for anti-spam

Server API: <ToTokGame Server API Document (to Game Developer)>

example:

```
- (IBAction)pay:(id)sender {
    TTGCHUD_NO_Stop(@"pay...")
    [[TTkGameManager  defaultManager] buyProductWithSKU:@"LangChaoJD001"
Progress:^(TTGCOderStatus orderStatus) {
        NSLog(@"order status: %ld",(long)orderStatus);
    } Completion:^(id  _Nullable orderInfo, NSError * _Nullable error) {
        if (orderInfo) {
            TTGCHUD_HINT(@"success")
        }
        if (error) {
            NSLog(@"%@",error.userInfo);
            TTGCHUD_HINT([error.userInfo objectForKey:@"errorMsg"]);
        }
    }];
}
```

6.2 Query Order

Method	queryOrderWithOrderId: Completion:
Params	

orderId	the order's id
Completion	TTGCOrderCompletionHandler TTGCOrderModel by 5.1

example:

```
- (IBAction)query:(id)sender {
    [[TTkGameManager  defaultManager] queryOrderWithOrderId:@"xxx"
Completion:^(id  _Nullable orderInfo, NSError * _Nullable error) {
        if (orderInfo) {
            //get order info
        } else {
            //error message
        }
    }];
}
```

7. API: SNS & Share

7.1 Field Description

TTGCFriendModel

Params	Note
ttkUid	totok user id
name	friend's name
photoUrl	friends' avatars

TTGCPlayingFriendModel

Params	Note
ttkUid	totok user id
userId	user id of friend
name	friend's name
photoUrl	friends' avatars

TTGCSocialTTkNotice

Params	Note
ttkUid	totok user id
title	shared title
contentString	shared text content
imageUrl	shared web picture address

TTGCSocialTTkInvite

Params	Note
ttkUid	totok user id
contenString	shared text content
imageUrl	shared web picture address
buttonTitle	shared click button name

7.2 Get ToTok Friends

Method	getFriendsFromServerCompletion
Callback	TTGCTTkFriendsCompletionHandler

example:

```
- (void)getFriends {

    [[TTkGameManager defaultManager] getFriendsFromServerCompletion:^(NSArray * _Nullable
list, NSError * _Nullable error) {

        // list of ttk friends

    }];

}
```

7.3 Get Playing Game Friends

Method	getPlayingFriendsFromServerCompletion
Callback	TTGCTTkFriendsCompletionHandler

example:

```
- (void)getPlayingFriends {

    [[TTkGameManager defaultManager] getPlayingFriendsFromServerCompletion:^(NSArray *
_Nullable list, NSError * _Nullable error) {

        // list of playing friends

    }];

}
```

7.4 Share

7.4.1 Share Invite Message

Method	sendToMessageTottk:(id)message Completion:
Callback	TTGCShareCompleteHandler

example:

```
- (void)inviteFriend {

    TTGCSocialTTkInvite *model = [[TTGCSocialTTkInvite alloc] init];

    model.ttkUid = @"xxxxxxxx";

    model.contentString = @"come and join us";

    model.buttonTitle = @"Accept";

    model.imageUrl =
@"http://t8.baidu.com/it/u=1484500186,1503043093&fm=79&app=86&f=JPEG?w=1280&h=853";

    [[TTkGameManager defaultManager] sendToMessageTottk:model Completion:^(BOOL success,
NSError * _Nullable error) {


    }];

}
```

7.4.2 Share Notice Message

Method	sendToMessageTottk:(id)message Completion:
Callback	TTGCShareCompleteHandler

example:

```
- (void)sendNoticeToFriend {

    TTGCSocialTTkNotice *model = [[TTGCSocialTTkNotice alloc] init];

    model.ttkUid = @"xxxxxxxxx";

    model.contentString = @"come and join us";

    model.title = @"play game";

    model.imageUrl =
@"http://t8.baidu.com/it/u=1484500186,1503043093&fm=79&app=86&f=JPEG?w=1280&h=853";

    [[TTkGameManager defaultManager] sendToMessageTottk:model Completion:^(BOOL success,
NSError * _Nullable error) {


    }];
}
```

8. API: Push

8.1 Register

in AppDelegate:

```
- (void)application:(UIApplication *)application
didRegisterForRemoteNotificationsWithDeviceToken:(NSData *)deviceToken {

    [[TTkGameManager  defaultManager] registerDeviceToken:deviceToken];

}
```

8.2 Receive Push Message

Method	getRemoteNotification
Callback	TTGCNotificationCompletionHandler

example:

```
- (void)recieveMessage {

    [[TTkGameManager  defaultManager] getRemoteNotification:^(id  _Nullable result,
NSError * _Nullable error) {

        //Receive a push message and can customize the event based on the message.

        //Do something ...

    }];

}
```