ToTokGameSDK Installation

(iOS)

Name	Date	version	remark
Yuanjun	2020.3.18	1.0.0	created
Yuanjun	2020.3.27	1.0.1	improvements
Yuanjun	2020.4.08	1.0.2	registration app details (1.1 & 1.2 & 1.3)
Yuanjun	2020.4.15	1.0.4	firebase created(1.4) related supplement(4.1.1 & 4.1.3)
Yuanjun	2020.4.20	1.0.5	1. Add the manual firebase integration step and don't care if using cocopods.(3.2-3 & 4.1.4) 2. Domain Changed & Sign in with Apple(4.1.1) 3. Sign in with Apple configure of the Key.(6)
Yuanjun	2020.4.27	1.0.6	improvements
Yuanjun	2020.5.13	1.0.7	improvements
Yuanjun	2020.5.27	1.0.8	SDK rename 1. Upgrade from old version : (1) Replace this line in Podfile old: `pod 'ToTokGameSDK'` new: `pod 'TTkGameSDK'` (2) `$ pod install` 2. Use the new version directly Reference 3.1

Yuanjun	2020.5.29	1.1.0	improvements
Yuanjun	2020.6.24	1.1.1	1. twitter created (1.5) 2. related supplement (4.1.3)
Yuanjun	2020.7.06	1.1.2	1. AppsFlyer created(1.6) related supplement(4.1.3) 2. Tapjoy created(1.7) related supplement(4.1.3)
Yuanjun	2020.7.10	1.1.3	1. fix AppsFlyer 2. add note for pod command (3)
Yuanjun	2020.7.14	1.1.4	improvements
Yuanjun	2020.7.15	1.1.5	improvements
Yuanjun	2020.7.20	1.1.6	add Note (4.1)
Yuanjun	2020.7.22	1.1.7	improvements

1. Register and create app

1.1 Apple Developer

- Log in to the Apple Developer Website and enter your developer account.

https://developer.apple.com/

- Add your app identifier here.
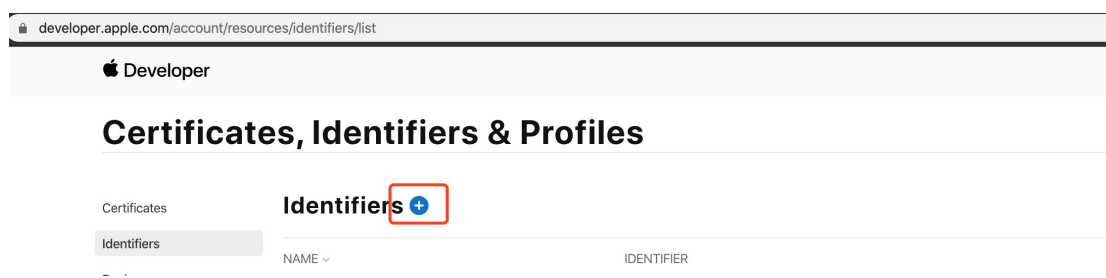
- Select Create App IDs

 Developer

Certificates, Identifiers & Profiles

‹ All Identifiers

Register a New Identifier

- **App IDs**
  Register an App ID to enable your app to access available services and identify your app in a provisioning profile. You can enable app services when you create an App ID or modify these settings later.

- **Services IDs**
  For each website that uses Sign in with Apple, register a services identifier (Services ID), configure your domain and return URL, and create an associated private key.

- **Pass Type IDs**
  Register a pass type identifier (Pass Type ID) for each kind of pass you create (i.e. gift cards). Registering your Pass Type IDs lets you generate Apple-issued certificates which are used to digitally sign and send updates to your passes, and allow your passes to be recognized by Wallet.

- **Website Push IDs**
  Register a Website Push Identifier (Website Push ID). Registering your Website Push IDs lets you generate Apple-issued certificates which are used to digitally sign and send push notifications from your website to macOS.

- **iCloud Containers**
  Registering your iCloud Container lets you use the iCloud Storage APIs to enable your apps to store data and documents in iCloud, keeping your apps up to date automatically.

- Create the App's Bundle ID and record it, both 1.2 and 1.3 will be used.

Certificates, Identifiers & Profiles

‹ All Identifiers

Register an App ID

Back Continue

Platform
iOS, macOS, tvOS, watchOS

App ID Prefix
■■ (Team ID)

Description

You cannot use special characters such as @, &, *, ', "

Bundle ID Explicit ○ Wildcard

We recommend using a reverse-domain name style string (i.e., com.domainname.appname). It cannot contain an asterisk (*).

- Finally, don't forget to set up Capabilities.(4.1.1 for details)

1.2 ToTokGame

Register a new app in ToTokGame platform
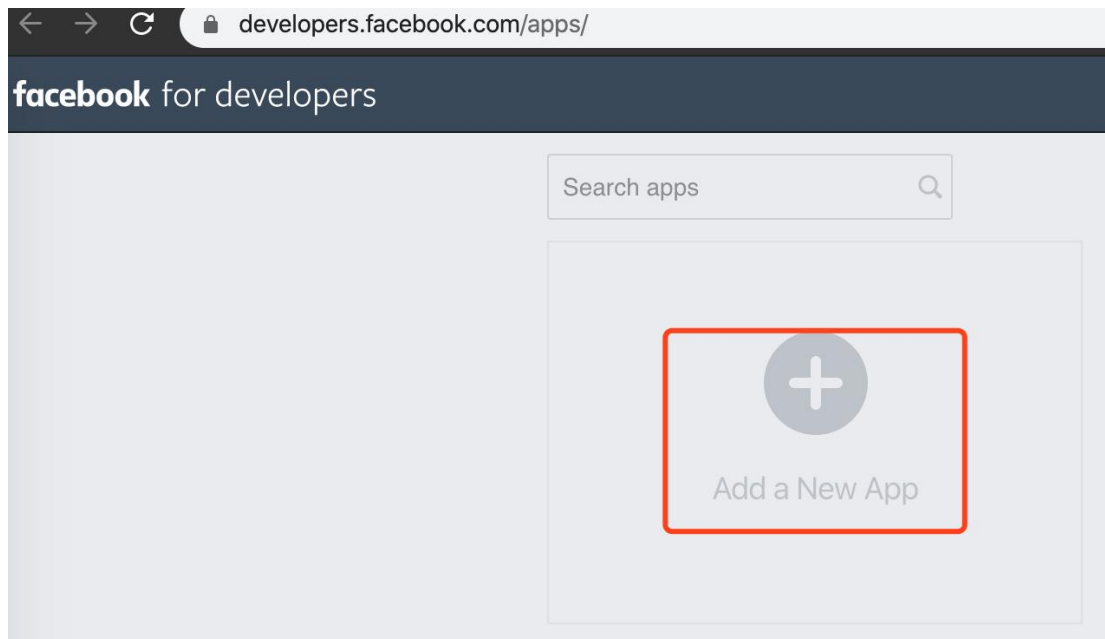
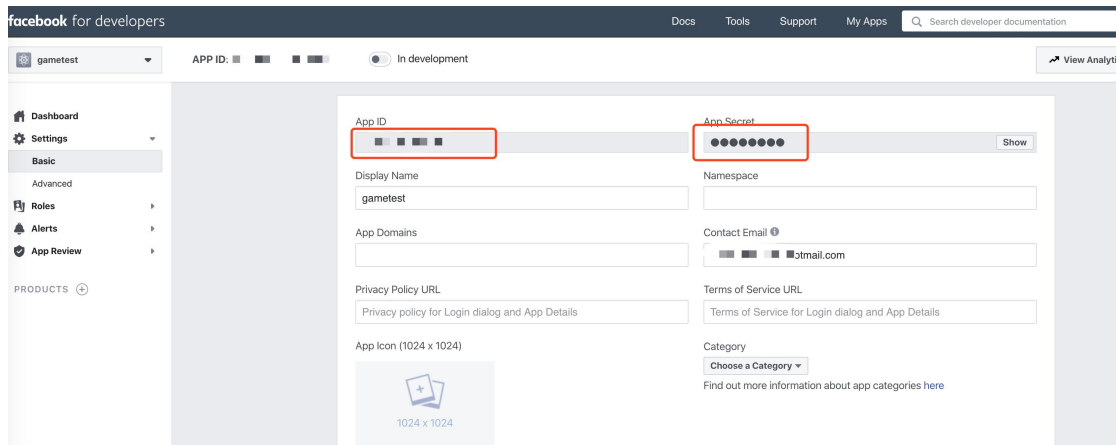Get application information such as appId, appsecret, gameid.

1.3 Facebook

If you use Facebook related functions, you need to register with the Facebook development platform to create an application.

https://developers.facebook.com/apps/

- First create your app here.

- Remember your App ID and App Secret, which may be used during 1.2 and 4.1.

1.4 Firebase

1.4.1 Firebase project

- Create a Firebase project
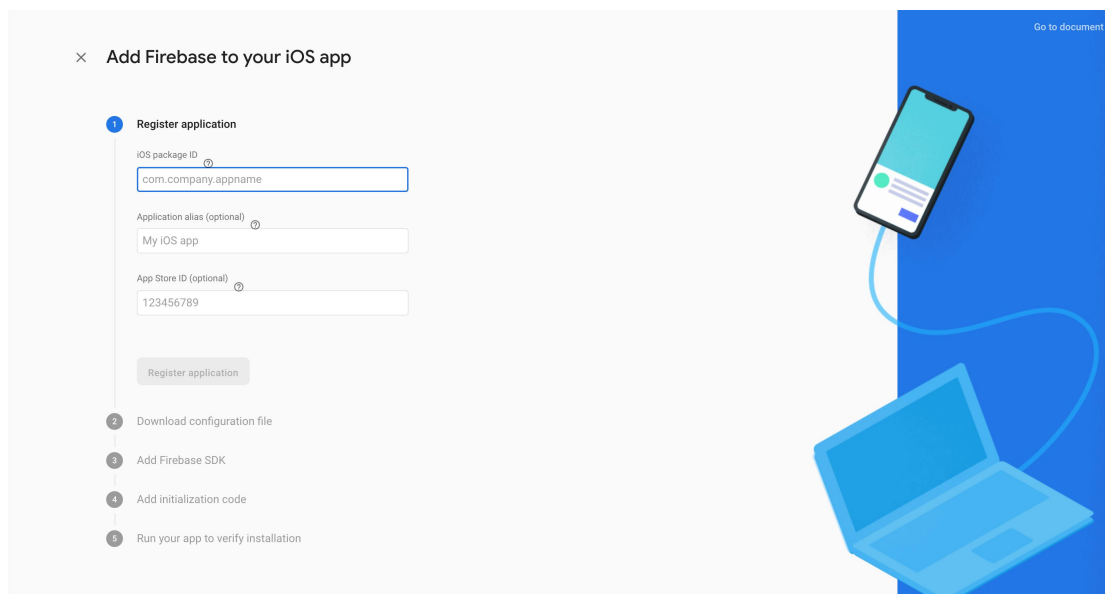
https://console.firebase.google.com/

Before you can add Firebase to your iOS app, you need to create a Firebase project to connect to your iOS app.
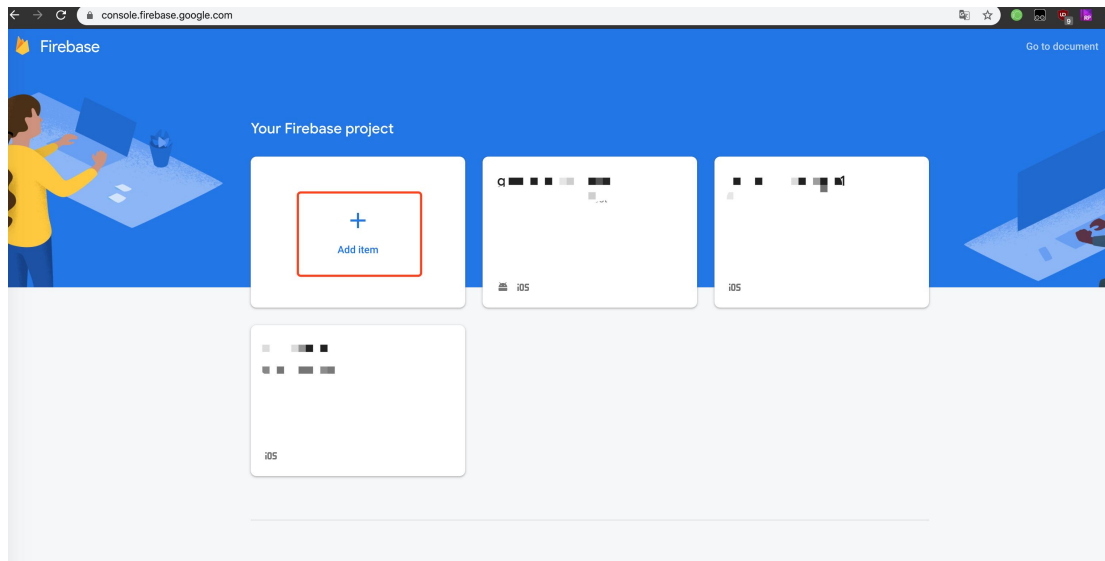
- Register your app with Firebase

After you have a Firebase project, you can add your iOS app to it.

In the center of the project overview page, click the **iOS** icon to launch the setup workflow.

- Add a Firebase configuration file

Click **Download GoogleService-Info.plist** to obtain your Firebase iOS config file.

Move your config file into the root of your Xcode project. If prompted, select to add the config file to all targets.
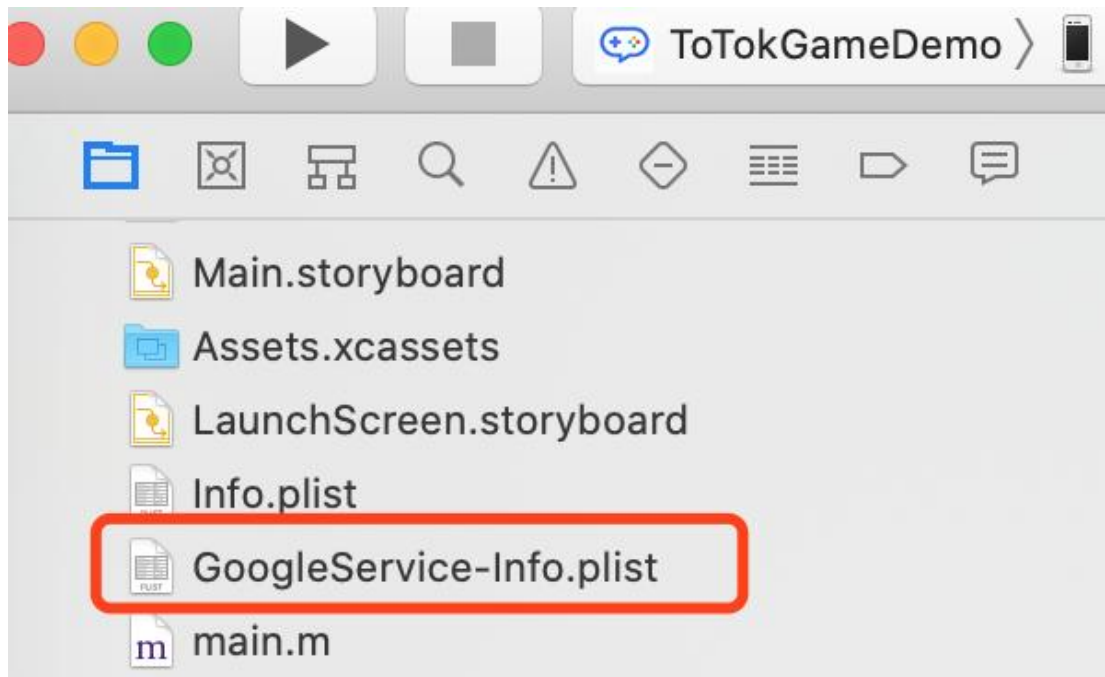
If you have multiple bundle IDs in your project, you must associate each bundle ID with a registered app in the Firebase console so that each app can have its own GoogleService-Info.plist file.

1.4.2 Dynamic link

Dynamic link

URL prefix 1/10

ht ■ ■ ■ ■.link

Add URL prefix

Last 30 days
March 15 – April 15
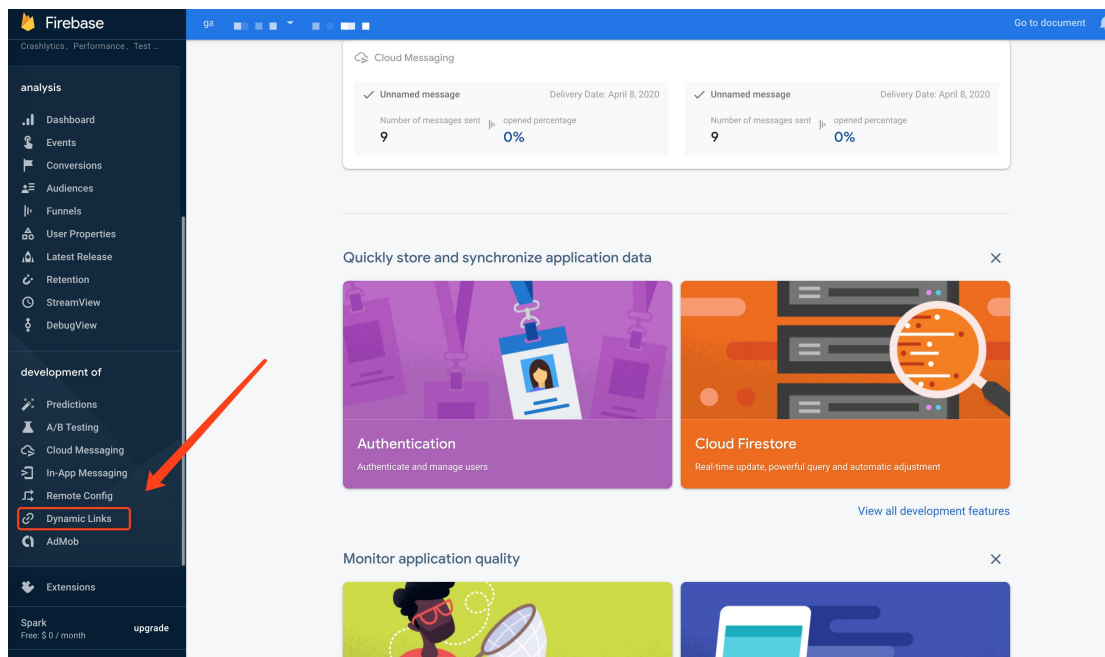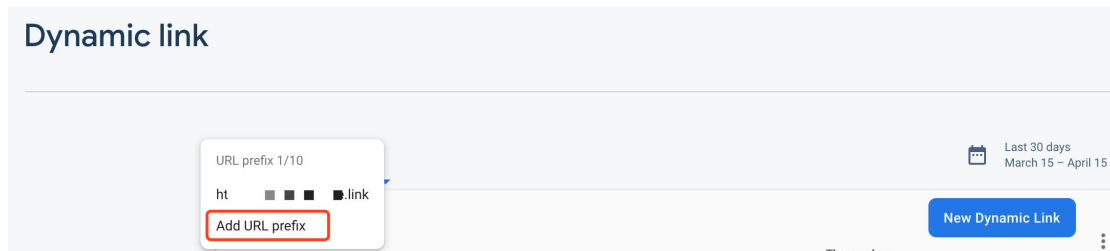
New Dynamic Link

Add URL prefix

1 create —— 2 Configuration —— 3 verification —— 4 carry out

Use a URL prefix consisting of a custom dynamic link domain name (such as yourdomain.com) and an optional path (such as yourdomain.com/mypath) to completely change your link format. The custom domain name looks more professional and provides more background information, which is conducive to more link clicks.

Domain ⑦

cancel   carry on

After creating the URL, the game developer needs to configure the domain in the development project.

1. In the Info tab of your app's Xcode project, create a new URL type to be used for Dynamic Links. Set the Identifier field to a unique value and the URL scheme field to be your bundle identifier, which is the default URL scheme used by Dynamic Links.(See 4.1.3)

2. In the Capabilities tab of your app's Xcode project, enable Associated Domains and add the Associated Domains list.(See 4.1.1 for details)

1.5 Twitter

If you use Twitter related functions, you need to register with the Twitter development platform to create an application.

https://developer.twitter.com/en/apps/

- First create your app here.
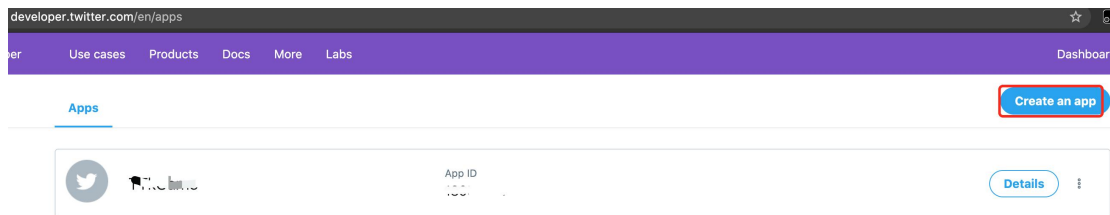
- Edit your app detail.
- Note the format of callback urls.

format: *twitterkit-API Key://*

1.6 AppsFlyer

If you use AppsFlyer related functions, you need to register with the AppsFlyer development platform to create an application.

https://hq1.appsflyer.com/apps/myapps

1.7 Tapjoy

If you use Tapjoy related functions, you need to register with the Tapjoy development platform to create an application.

https://ltv.tapjoy.com/

2. Installation and Runtime Environment

- IDE : Xcode 10.3 or higher
- iOS : iOS 9.0 or higher

3. Install SDK

SDK Directory Structure :

- TTkGameSDK

    -Core/TTkGameSDK.frame

    -SocialLibraries/Facebook/

    -SocialLibraries/TTGCAnalytics/

    -SocialLibraries/Twitter/

    -SocialLibraries/AppsFlyer/

    -SocialLibraries/TapjoySDK/

Information of third-party SDK:

1. Facebook: use cocoapods dependency

'FBSDKCoreKit'   '~> 6.5.1'

'FBSDKLoginKit'   '~> 6.5.1'

'FBSDKShareKit'   '~> 6.5.1'

2. Firebase : use cocoapods dependency

'Firebase/Analytics'

'Firebase/DynamicLinks'

3. Twitter: use cocoapods dependency

'TwitterKit5'

4. AppsFlyer: use cocoapods dependency

'AppsFlyerFramework'

5. Tapjoy: use cocoapods dependency

'TapjoySDK'

Note:

1. If you don't need any third-party SDK, you can use the command:

```
pod 'TTkGameSDK/Core'
```

2. If you only use the Firebase third-party SDK, you can use the command:

```
pod 'TTkGameSDK/Analytics'
```

3. If you only use the Facebook third-party SDK, you can use the command:

```
pod 'TTkGameSDK/Facebook'
```

4. If you only use the Twitter third-party SDK, you can use the command:

```
pod 'TTkGameSDK/Twitter'
```

5. If you only use the Appsflyer third-party SDK, you can use the command:

```
pod 'TTkGameSDK/AppsFlyer'
```

6. If youonly use the Tapjoy third-party SDK, you can use the command:

```
pod 'TTkGameSDK/Tapjoy'
```

3.1 Use Pod Install SDK（Recommend）

It is recommended to use the cocoapods install SDK. Cocoapods provides a simple dependency management system to avoid errors caused by manual import .

first you need to confirm that Cocoapods has been installed, if you have not installed Cocoapods, refer to the official website user guide:

https://cocoapods.org/.

1. In the root directory of the Xcode project, create a new Podfile.

$ pod init

Add the following to this file :

```
source 'https://github.com/CocoaPods/Specs.git'
pod 'TTkGameSDK'
```

2. In the Podfile directory, execute the following instructions :

$ pod install

3. After executing pod install, open the project directory and find the .xcworkspace file to run.

4. Future SDK upgrades can use the update command.

```
$ pod update
```

3.2 Install SDK Manually

1. Download TTkGameSDK related releases on github
https://github.com/ToTokGames/ToTokGameSDK-iOS/releases

2. Download related FacebookSDK version 5.3.1 on github
https://github.com/facebook/facebook-ios-sdk/releases/tag/v5.13.1

3. The firebase related SDK can be downloaded in the following two ways:
(1) The new demo project installs with cocopods, then compiles with the simulator and the real machine, merges the compiled framework, and drags it into the main project.
https://firebase.google.com/docs/ios/setup#available-pods
```
pod 'Firebase/Analytics'
```

```
pod 'Firebase/DynamicLinks'
```

(2) Use the firebase source code to compile the following corresponding frameworks, and compile the simulator and the real machine, and finally merge and drag them into the project.

https://github.com/firebase/firebase-ios-sdk
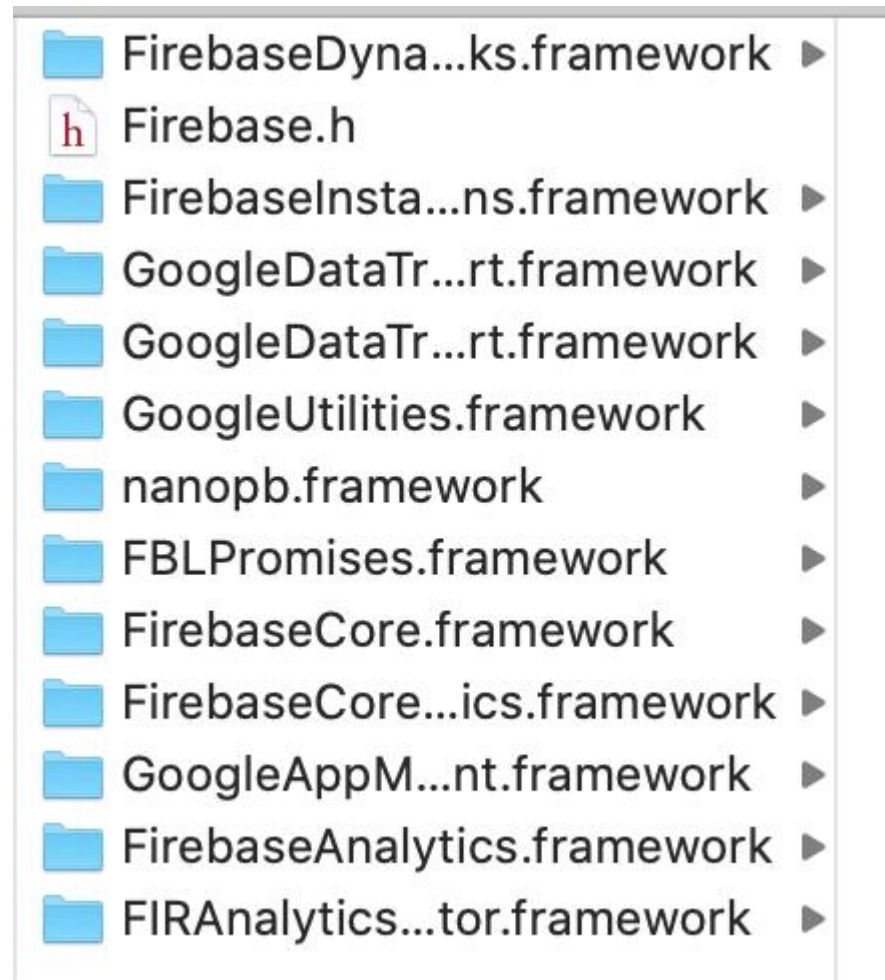
4. Project Configuration

4.1 Build Configuration

Note: The build item value needs to be a number.

4.1.1 Apple service configuration, push, keychain, in-app purchase, and Universal Link are the services provided by Apple and need to be configured here .

as shown below
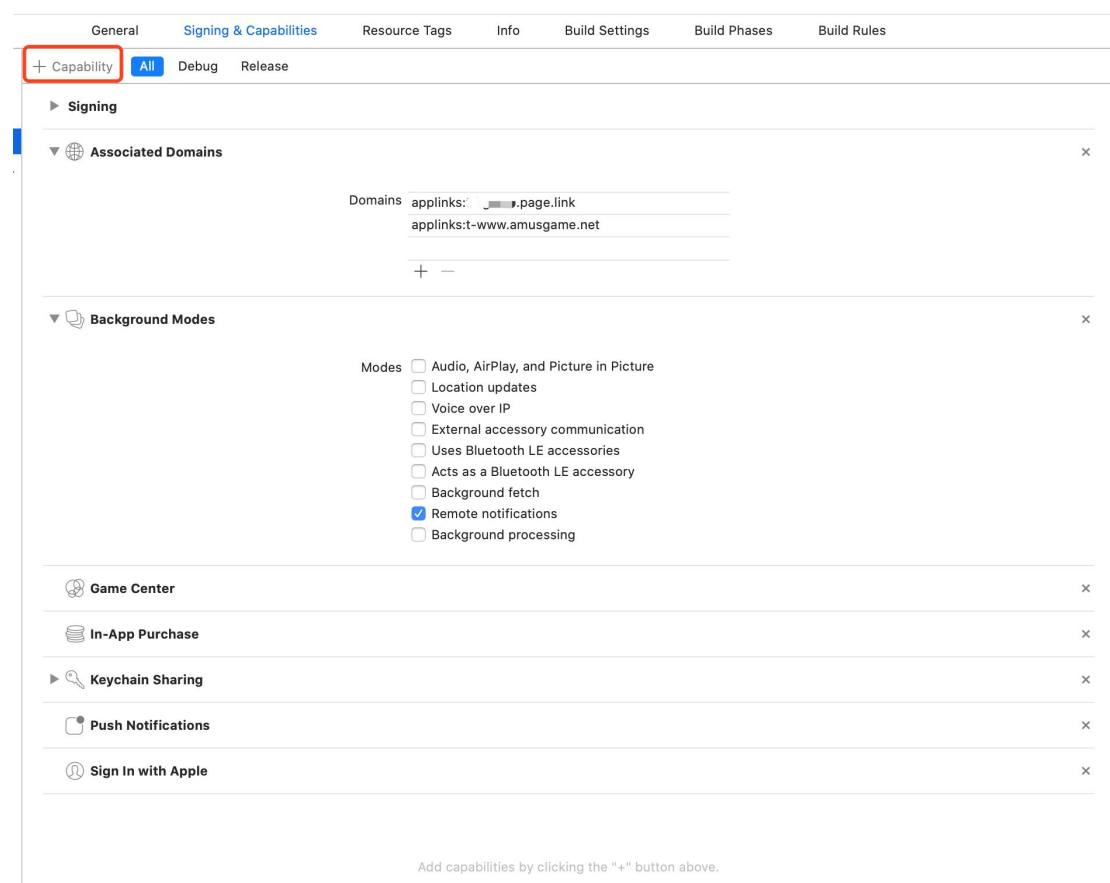
Domains(Test):

applinks:t-www.amusgame.net

applinks:*your_dynamic_links_domain* (From step 1.4.2)

Domains(Production):

applinks:www.falcongames.net

applinks:*your_dynamic_links_domain* (From step 1.4.2)

4.1.2 Set bitcode to NO (not supported by Facebook)

as shown in the below

Basic   Customized   All   Combined   Levels   +   Q~ bitcode

▼ Build Options

Setting	ToTokGameApp
Enable Bitcode	No ⌄

4.1.3 Set plist.info

Relevant function configuration:

Facebook related:

key: CFBundleURLSchemes   value: fbxxxxxxxx (fb+FacebookAppID)

key: FacebookAppID   value: xxxxxxx (FacebookAppID)

key: AppName   value: xxxxxxx (AppName)

Twitter related:

key: CFBundleURLSchemes   value: twitterkit-xxxxxxxxx (twitterkit-+Twitter key)

key: TwitterConsumerKey   value: xxxxxxx (Twitter key)

key: TwitterConsumerSecret   value: xxxxxxxx (Twitter secret)

Firebase related:

key: CFBundleURLSchemes   value: xxx.xxx.xxx (bundle id)

AppsFlyer related:

key: AppID   value: xxxxxxxx (AppID from AppStore)

key: AppsFlyerDevKey   value: xxxxxxxx (AppsFlyer key)

Tapjoy related:

key: TapjoyKey   value: xxxxxxxx (Tapjoy key)

```
<key>NSPhotoLibraryUsageDescription</key>

<string>Photo albums are required to share pictures, please allow access</string>

<key>AppsFlyerDevKey</key>

<string>xxxxxxxx</string>

<key>AppID</key>

<string>xxxxxxxx</string>

<key>TapjoyKey</key>
```

```xml
<string>xxxxxxxx</string>
<key>TwitterConsumerKey</key>
<string>xxxxxxxxx</string>
<key>TwitterConsumerSecret</key>
<string>xxxxxxxxx</string>
<key>CFBundleURLTypes</key>
<array>
<dict>
<key>CFBundleURLSchemes</key>
<array>
<string>fbxxxxxxxxx</string>
</array>
</dict>
<dict>
<key>CFBundleTypeRole</key>
<string>Editor</string>
<key>CFBundleURLName</key>
<string>Bundle</string>
<key>CFBundleURLSchemes</key>
<array>
<string>xxx.xxx.xxx</string>
</array>
</dict>
<dict>
<key>CFBundleTypeRole</key>
<string>Editor</string>
<key>CFBundleURLSchemes</key>
<array>
<string>twitterkit-xxxxxxxxx</string>
</array>
</dict>
</array>
<key>FacebookAppID</key>
<string>xxxxxxxxx</string>
<key>FacebookDisplayName</key>
<string>AppName</string>
```

```xml
<key>LSApplicationQueriesSchemes</key>
<array>
<string>whatsapp</string>
<string>fbapi</string>
<string>fbapi20130214</string>
<string>fbapi20130410</string>
<string>fbapi20130702</string>
<string>fbapi20131010</string>
<string>fbapi20131219</string>
<string>fbapi20140410</string>
<string>fbapi20140116</string>
<string>fbapi20150313</string>
<string>fbapi20150629</string>
<string>fbapi20160328</string>
<string>fbauth</string>
<string>fbauth2</string>
<string>fb-messenger-share-api</string>
<string>fbshareextension</string>
<string>fb-messenger-api20140430</string>
<string>fb-messenger-platform-20150128</string>
<string>fb-messenger-platform-20150218</string>
<string>fb-messenger-platform-20150305</string>
</array>
<key>NSAppTransportSecurity</key>
<dict>
<key>NSAllowsArbitraryLoads</key>
<true/>
<key>NSExceptionDomains</key>
<dict>
<key>akamaihd.net</key>
<dict>
<key>NSExceptionRequiresForwardSecrecy</key>
<false/>
<key>NSIncludesSubdomains</key>
<true/>
</dict>
```

```xml
<key>facebook.com</key>

<dict>

<key>NSExceptionRequiresForwardSecrecy</key>

<false/>

<key>NSIncludesSubdomains</key>

<true/>

</dict>

<key>fbcdn.net</key>

<dict>

<key>NSExceptionRequiresForwardSecrecy</key>

<false/>

<key>NSIncludesSubdomains</key>

<true/>

</dict>

</dict>

</dict>
```

If you choose pod installation, the above configuration is enough. Otherwise, please proceed to the following configuration steps.

4.1.4 Build configuration

Add Other Linker Flags （Pay attention to case）
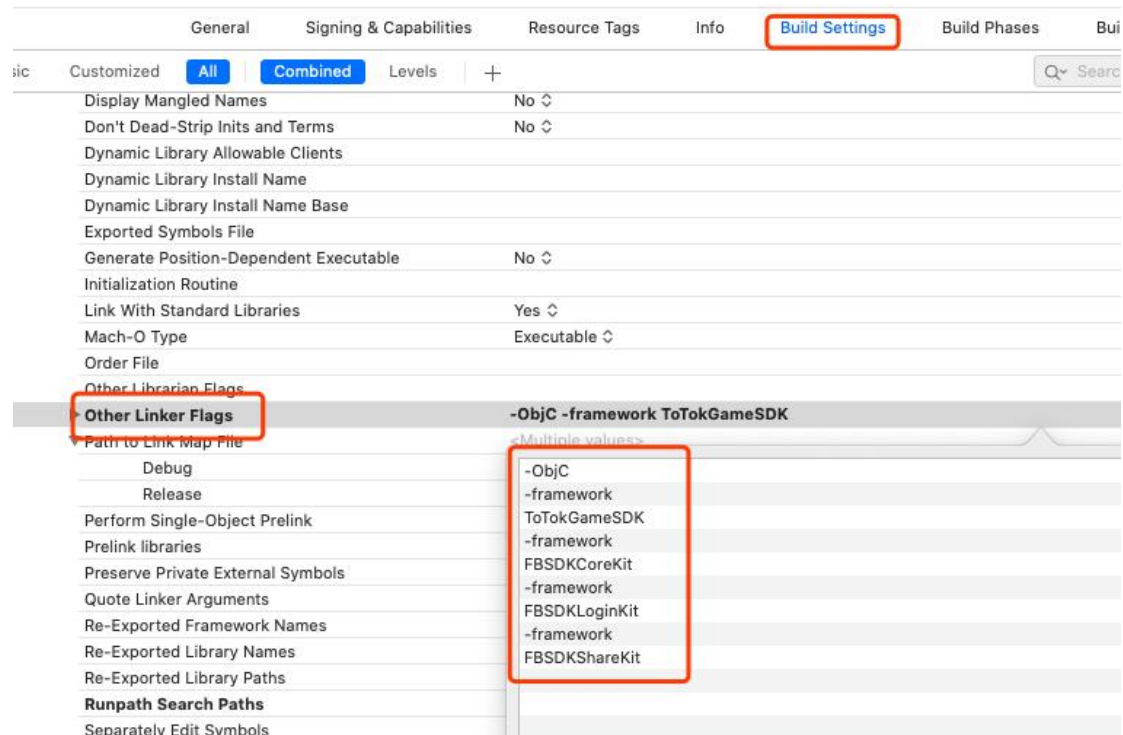
And firebase related
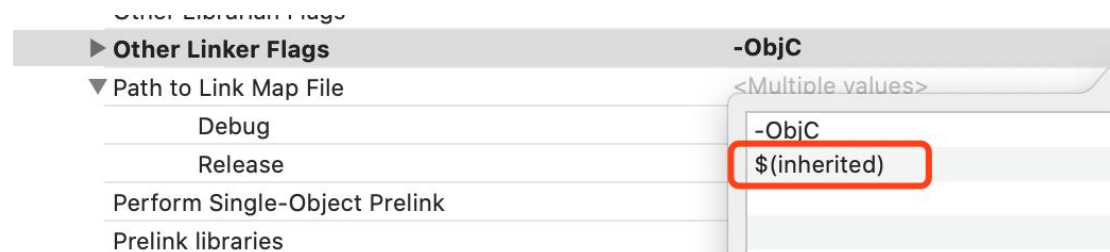
4.1.5 The system libraries used by related SDKs need to be configured as follows.

'sqlite3', 'UIKit', 'StoreKit', 'GameKit', 'WebKit', 'UserNotifications', 'Photos', 'ImageIO', 'Foundation', 'CoreFoundation', 'QuartzCore', 'CoreGraphics', 'AVFoundation', 'CoreTelephony', 'Security', 'CoreMotion', 'ExternalAccessory', 'SystemConfiguration', 'Accounts', 'Social', 'AudioToolbox', 'Accelerate'
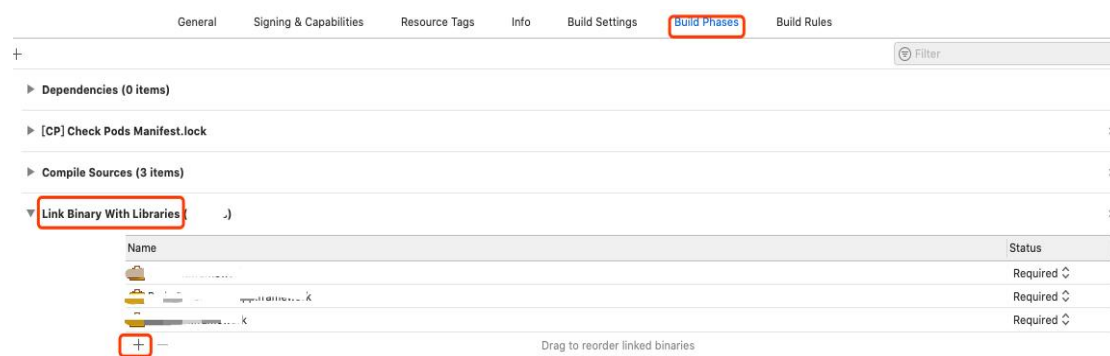
Add the system library required above here.

4.1.6 Compiling FBSDK requires setting compiling C++

Also set in build phases

Set Apple Clang – Language

as shown below
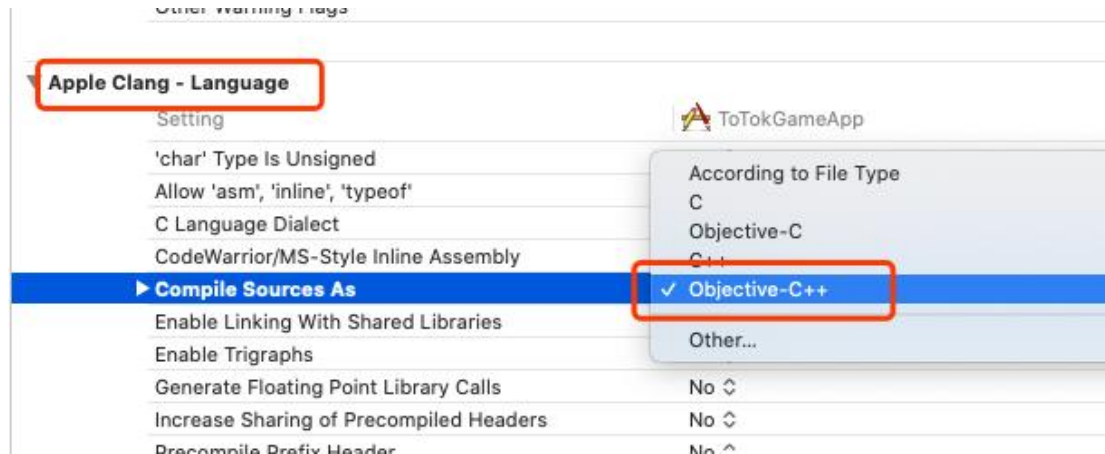
Compile configuration is over here.

5. Push configuration

If you'd like to send push notifications to your iOS users, you will need to provide either an APNs Push Certificate, or an APNs Auth Key.

5.1 APNs Auth Key（Recommend）

We recommend that you create and provide an APNs Auth Key for the following reasons:

1. No need to re-generate the push certificate every year.

2. One auth key can be used for all your apps – this avoids the complication of maintaining different certificates.

When sending push notifications using an APNs Auth Key, we require the following information about your app:

- Auth Key file

- Team ID
- Your app's bundle ID

To create an APNs auth key, follow the steps below.

- Log in to the Apple Developer Website and enter your developer account.
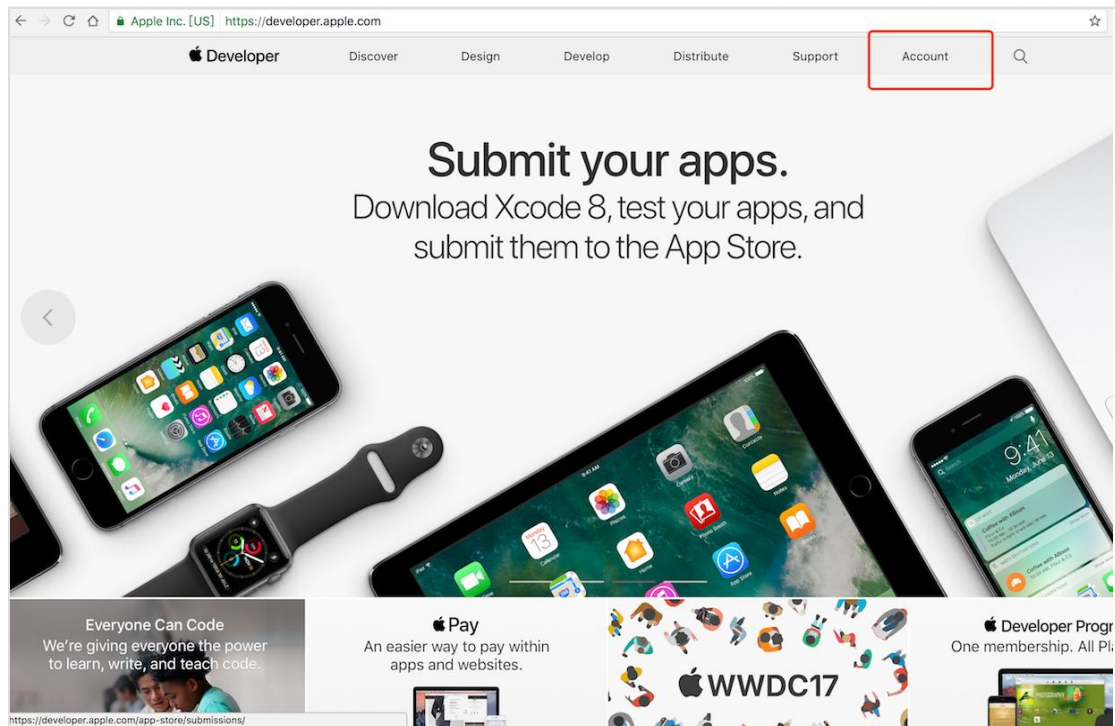
- In your developer account, go to Certificates, Identifiers & Profiles, and under Keys, select All.
- Click the Add button (+) in the upper-right corner.
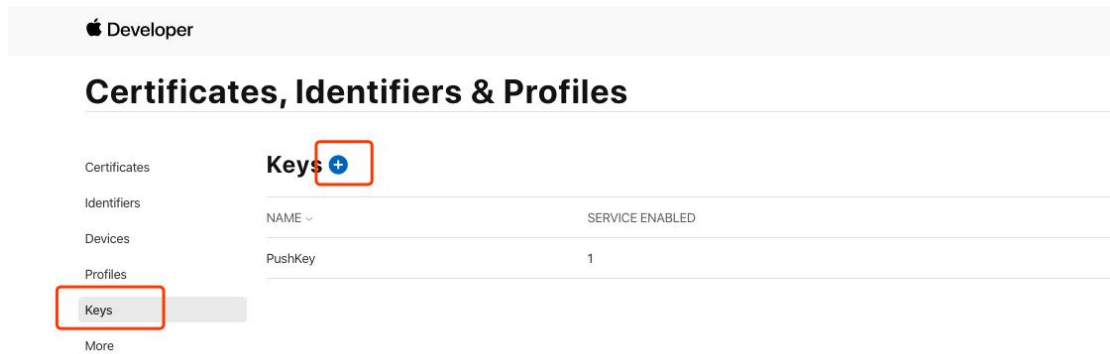
- Enter a description for the APNs Auth Key
- Under Key Services, select the APNs checkbox, and click Continue.

- Click Confirm and then Download. Save your key in a secure place. This is a one-time download, and the key cannot be retrieved later.

5.2 Certificate configuration

- Go to the "Certificates, IDs & Profiles" page from the left entry of the developer account page.

- Create an App ID, fill in the name of the App ID and the Bundle ID (skip this step if the ID already exists).

Note: You need to specify the specific bundle ID here. Do not use wildcard characters.

- Enable Push Notification for the app. If the App ID has already been created, you can also enable the Push Notification function through settings.

- After filling in the above attributes, click "Continue" to confirm the correctness of the AppId attribute, and click "Register" to register the AppId successfully.
- If you haven't created a Push certificate before or want to re-create a new one, please create a new one below the certificate list.

- When creating a new certificate, you need to choose the type of APNs certificate. As shown in the figure APNs certificate has two types of development (development) and production (production).

Note: The development certificate is used for development and debugging; the production certificate can be used for both development and debugging and product release. Here we choose the production certificate as an example.

- Click "Continue", then select the certificate and the AppID to be bound.

- Clicking "Continue" again will let you upload the CSR file. (CSR file will be created in the next step)

- Open KeychainAccess that comes with the mac system to create a Certificate Signing Request.

Operate as shown below:
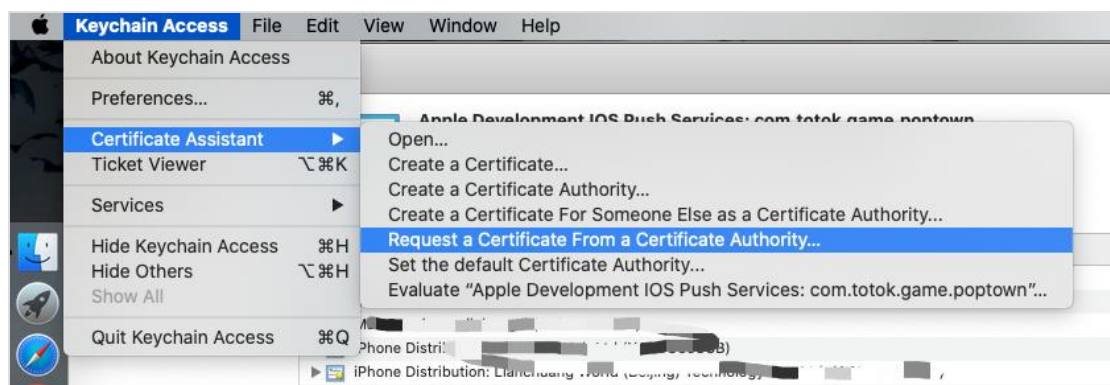
- Fill in "User Email" and "Common Name" and select "Save to Disk". The certificate file suffix is .certSigningRequest.
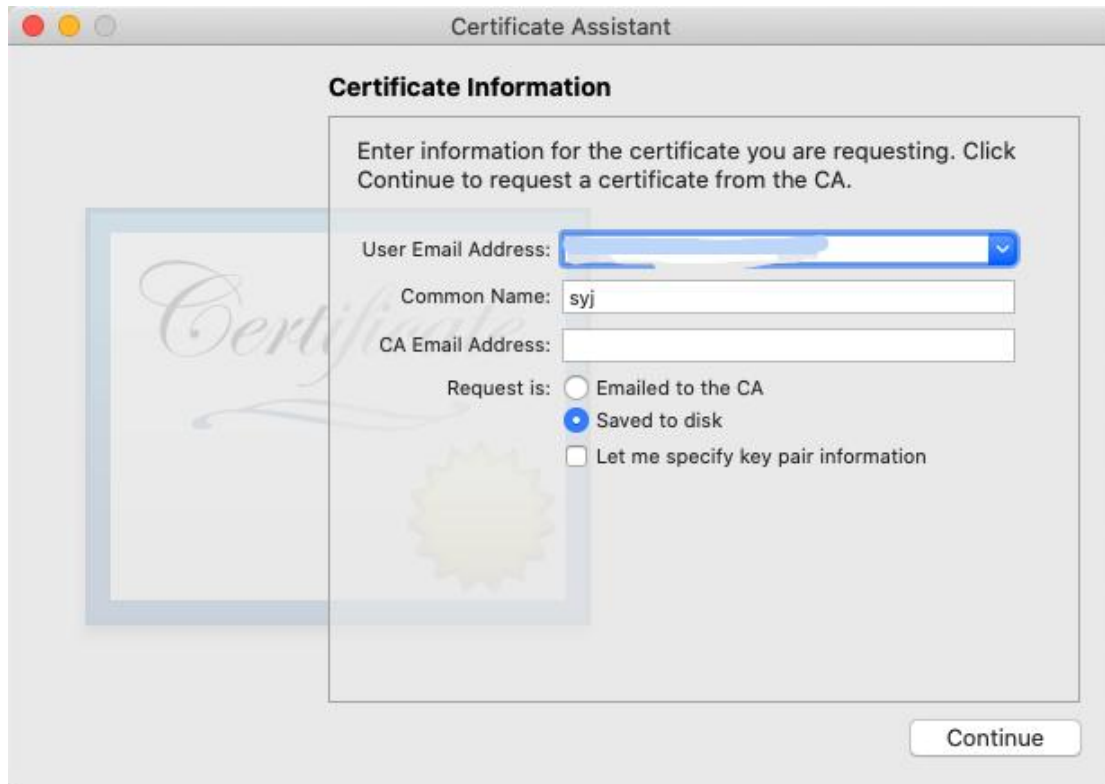
- Go back to the CSR upload page in your browser and upload the file with the suffix .certSigningRequest just generated.

- After the certificate is generated successfully, click the "Download" button to download the certificate. It is a file with the extension .cer.
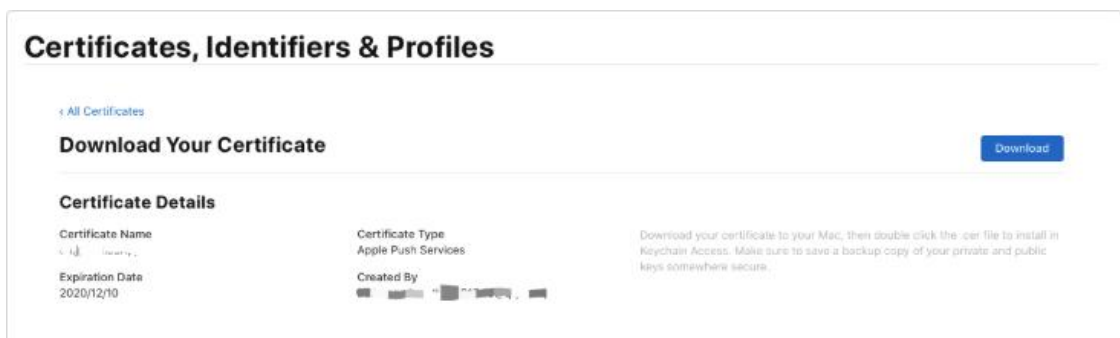
- After double-clicking the certificate, it will open in "KeychainAccess", select "Login" in the "Keychain" list on the left, and "My certificate" in the "Type" list, find the certificate you just downloaded, and export it as a .p12 file. As shown below:
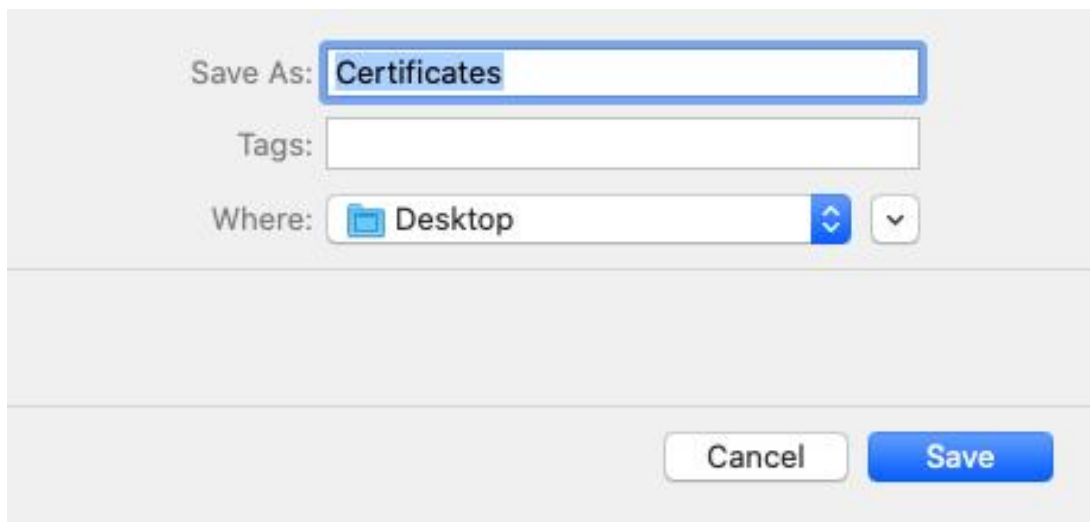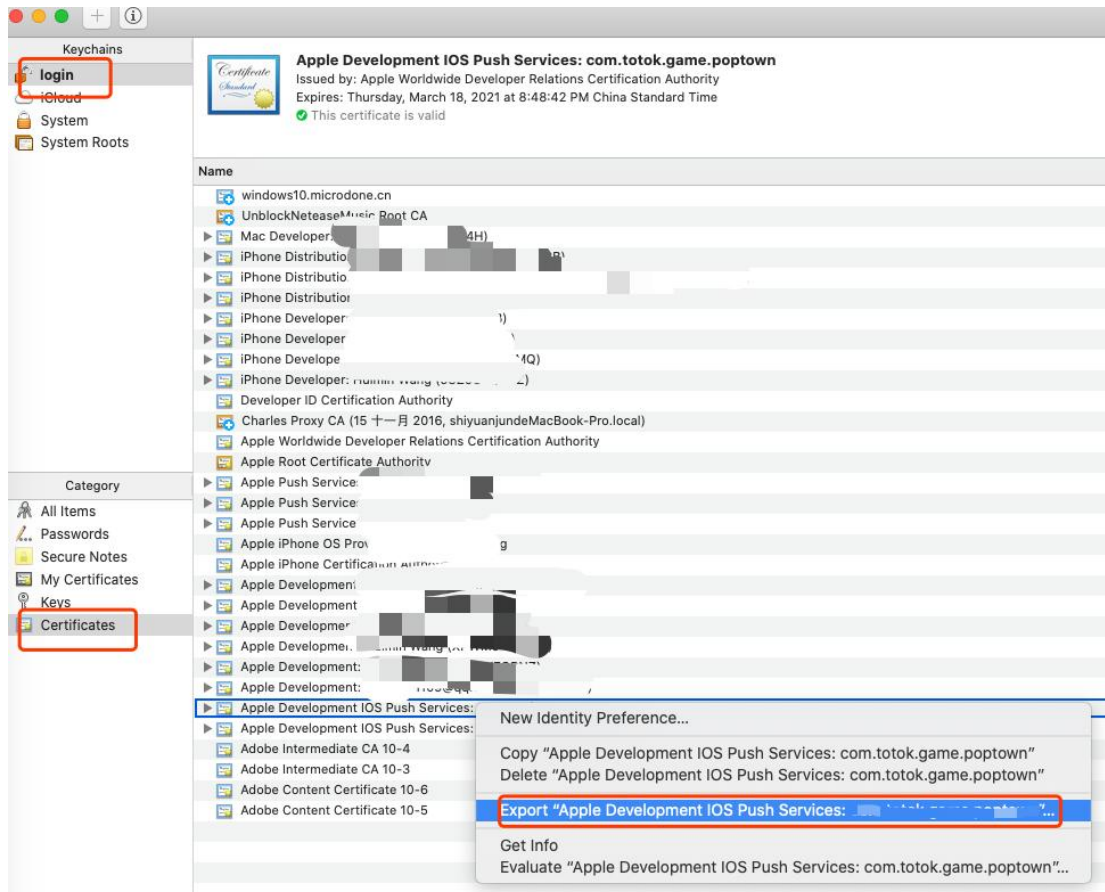
6. Sign in with Apple configuration

Edit your App ID Configuration

Edit your App ID Configuration

☐	⊘	Maps ⓘ			
☐	⇄	Multipath ⓘ			
☐	⊡	Network Extensions ⓘ			
☐	N))	NFC Tag Reading ⓘ			
☐	VPN	Personal VPN ⓘ			
☑	⊡	Push Notifications ⓘ	Edit	Certificates (1)	
☑	◉	Sign In with Apple ⓘ	Edit	Enable as a primary App ID	
☐	⊕	SiriKit ⓘ			
☐	⊞	System Extension			

Register a New Key for Sign in with Apple

 Developer

Certificates, Identifiers & Profiles

Certificates

Keys ⊕

Identifiers

NAME ⌄	SERVICE ENABLED
Devices	
PushKey	1
Profiles	
Keys	
SigninKey	1
More	

Register a New Key

Continue

Key Name

You cannot use special characters such as @, &, *, ', "

ENABLE	NAME	SERVICE	
☐	Apple Push Notifications service (APNs)	Establish connectivity between your notification server and the Apple Push Notification service. One key is used for all of your apps. Learn more	
☐	DeviceCheck	Access per-device, per-developer data that your associated server can use in its business logic. One key is used for all of your apps. Learn more	
☐	MusicKit	Access the Apple Music catalog and make personalized requests for authorized users. Learn more ⓘ There are no identifiers available that can be associated with the key	Configure
☑	Sign in with Apple	Enable your apps to allow users to authenticate in your application with their Apple ID. Configuration is required to enable this feature.	Configure

Configure the App ID

Configure Key

Create a key for each of your primary App IDs in order to implement Sign in with Apple. This key will also be used for any App IDs grouped with the primary. The user will see your primary app's icon at sign in and in their Apple ID account settings.

Primary App ID: 1 App ID

Select...   ⌄

• Common Problem

1. pod related

If you have a pod installed, running pod install directly may cause the following problems

Need to update the pod repo

```
$ pod install --repo-update
```

2. Project related

If you use xcode11 to create a new project, the iOS13 + real machine or simulator has a black screen

Reason: For the multi-window function of the iPad, the life cycle has changed. At present, the SDK does not support the multi-window function, so developer who use the SDK can change the project to the previous structure.

Specific steps are as follows:

1. First delete the Application Scene Manifest column from info.plist

2. In the project directory, delete the Scenedelegate.h and Scenedelegate.m files

3. Then enter AppDelegate.m to comment or delete the content shown below

```objc
#pragma mark - UISceneSession lifecycle


- (UISceneConfiguration *)application:(UIApplication *)application
    configurationForConnectingSceneSession:(UISceneSession
    *)connectingSceneSession options:(UISceneConnectionOptions
    *)options {
    // Called when a new scene session is being created.
    // Use this method to select a configuration to create the new
        scene with.
    return [[UISceneConfiguration alloc] initWithName:@"Default
        Configuration" sessionRole:connectingSceneSession.role];
}


- (void)application:(UIApplication *)application
    didDiscardSceneSessions:(NSSet<UISceneSession *> *)sceneSessions {
    // Called when the user discards a scene session.
    // If any sessions were discarded while the application was not
        running, this will be called shortly after
        application:didFinishLaunchingWithOptions.
    // Use this method to release any resources that were specific to
        the discarded scenes, as they will not return.
}
```

4. The last step is to add the window property in AppDelegate.h

```objc
@property (strong, nonatomic) UIWindow * window;
```