

# 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: <code>pod 'ToTokGameSDK'</code> new: <code>pod 'TtkGameSDK'</code> (2) <code>\$ pod install</code> 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)

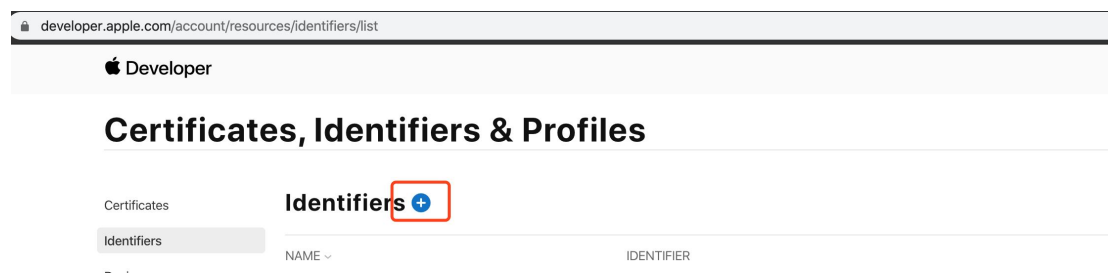
# 1. Register and create app

## 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.apple.com/account/resources/identifiers/add/bundleid

Apple 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

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

App ID Prefix  
[Team ID] (Team ID)

**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

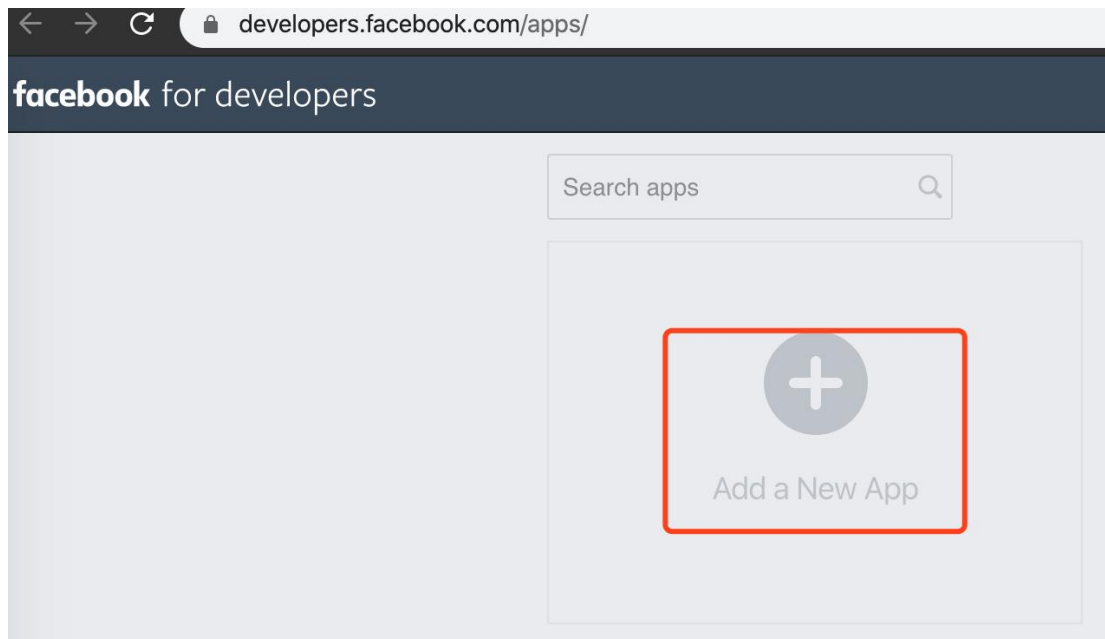
Get application information such as appld, appsecret, gameid.

## 1.3 Facebook

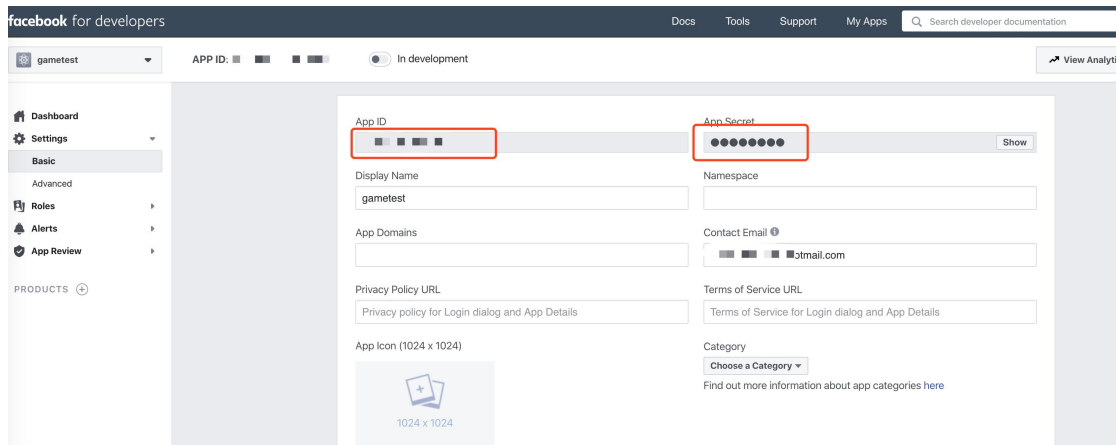
If you use Facebook related functions, you need to register with the Facebook development platform to create an application.

<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.



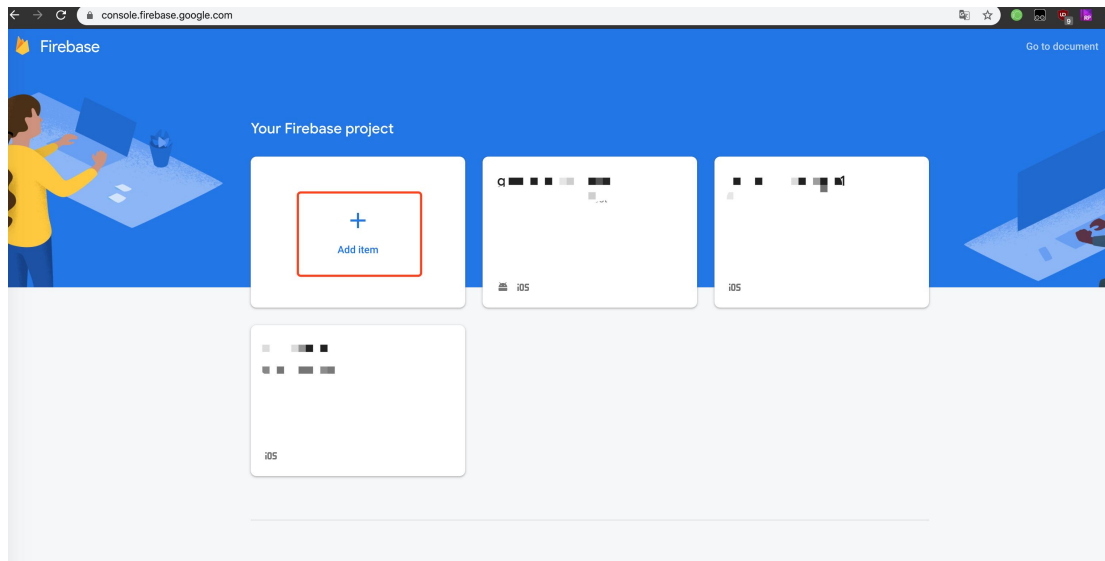
## 1.4 Firebase

### 1.4.1 Firebase project

- Create a Firebase project

<https://console.firebase.google.com/>

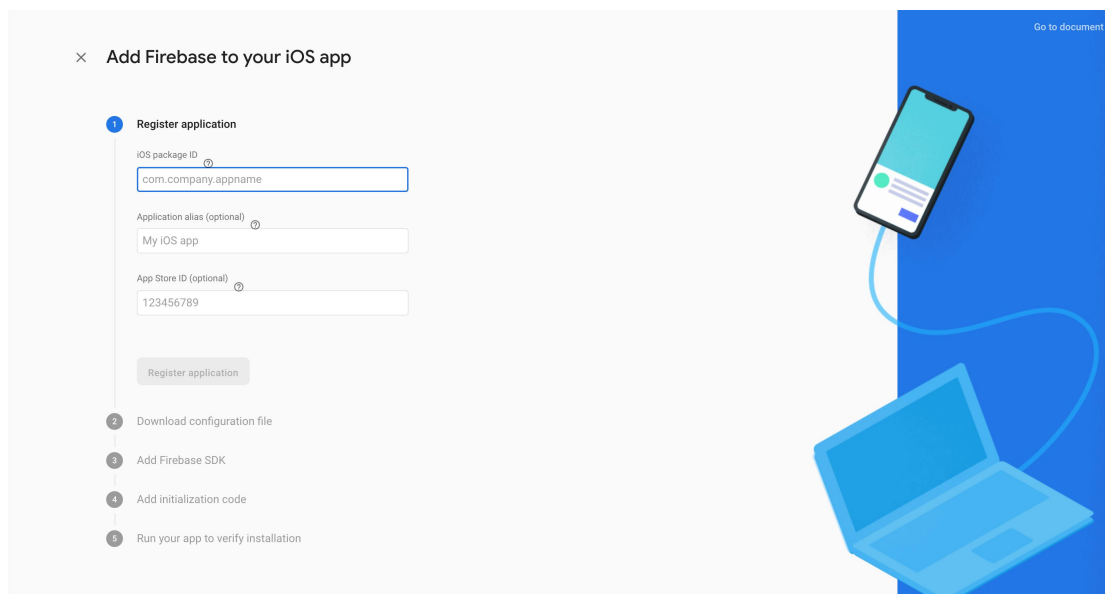
Before you can add Firebase to your iOS app, you need to create a Firebase project to connect to your iOS app.



- Register your app with Firebase

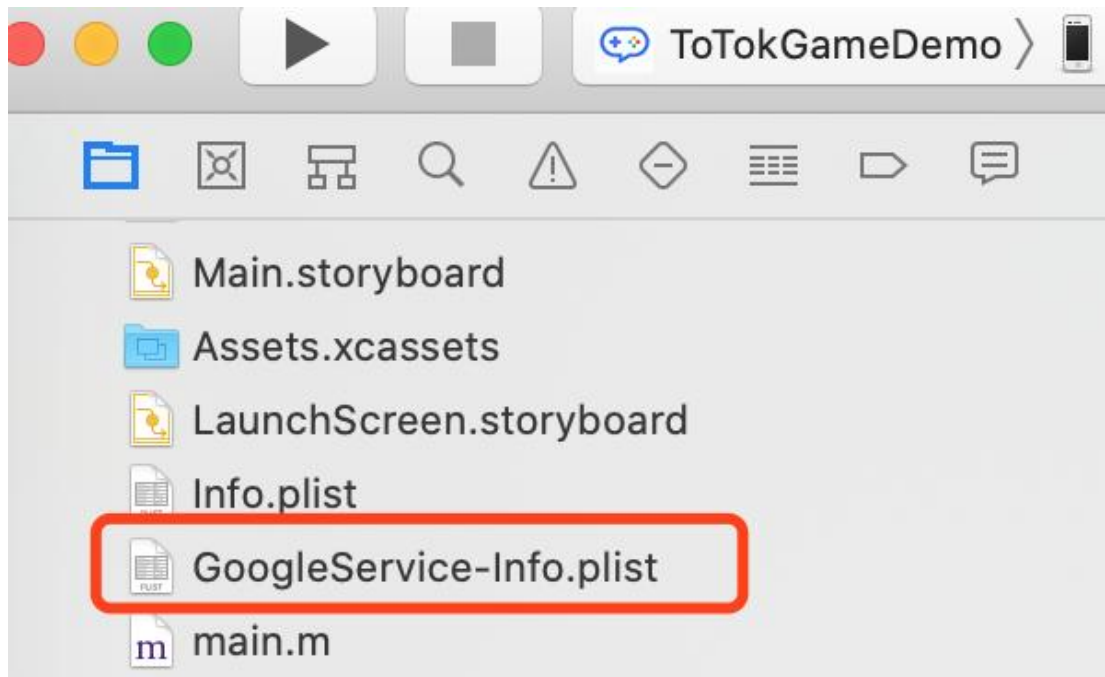
After you have a Firebase project, you can add your iOS app to it.

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



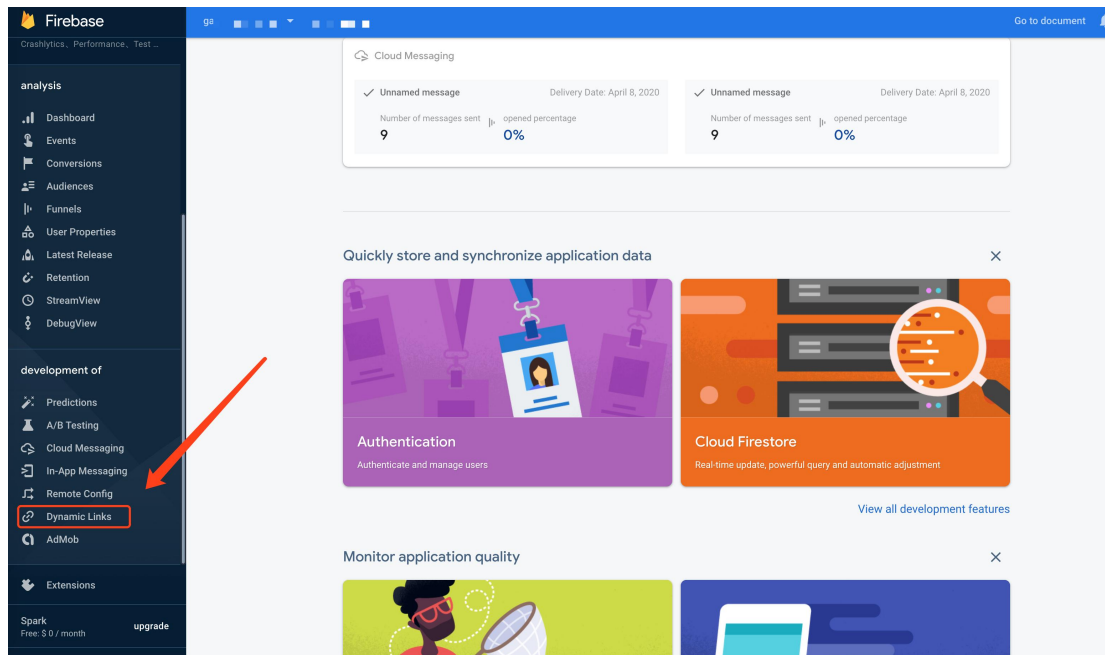
- Add a Firebase configuration file

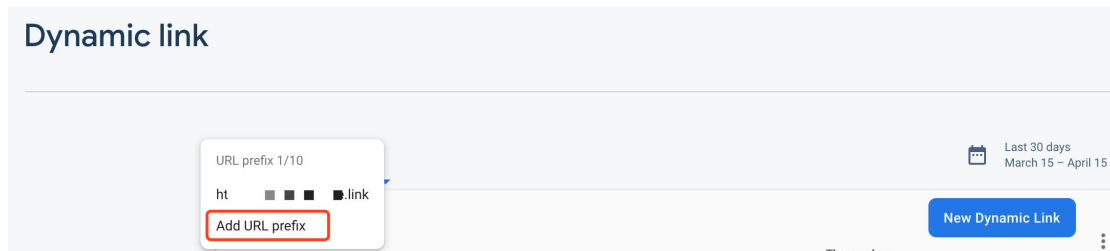
Click **Download GoogleService-Info.plist** to obtain your Firebase iOS config file. Move your config file into the root of your Xcode project. If prompted, select to add the config file to all targets.



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





## 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 ?

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 for blue market)

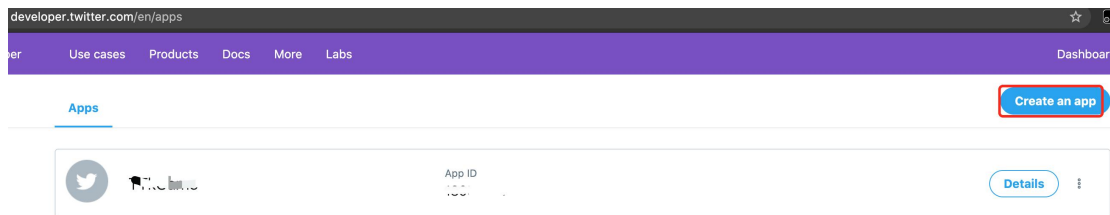
2. In the Capabilities tab of your app's Xcode project, enable Associated Domains and add the Associated Domains list.(See 4.1.1 for details)

## 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.

### Callback URLs (required) ⓘ

OAuth 1.0a applications should specify their `oauth_callback` URL on the request token step, which must match the URLs provided here. To restrict your application from using callbacks, leave these blank.

twitterkit-M\_504C`L`2`B`AIC3M0Nc://

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/TTGCAalytics/
  - 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'

## 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>

v5.13.1  
54219ba  
Compare

## Facebook SDK v5.13.1

facebook-sdk-bot released this on 17 Dec 2019

Consult Changelog v5.13.1

Assets 14

FacebookSDK_Dynamic.framework.zip	17.9 MB
FacebookSDK_Static.zip	40 MB
FBSDKCoreKit-Swift.zip	10.7 MB
FBSDKCoreKit.zip	26.1 MB
FBSDKCoreKit_Basics.zip	1.38 MB
FBSDKLoginKit-Swift.zip	1.42 MB
FBSDKLoginKit.zip	2.69 MB
FBSDKPlacesKit.zip	528 KB
FBSDKShareKit-Swift.zip	4.09 MB
FBSDKShareKit.zip	10.2 MB
FBSDKTVOSKit.zip	602 KB
SwiftDynamic.zip	8.12 MB
Source code (zip)	
Source code (tar.gz)	

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.

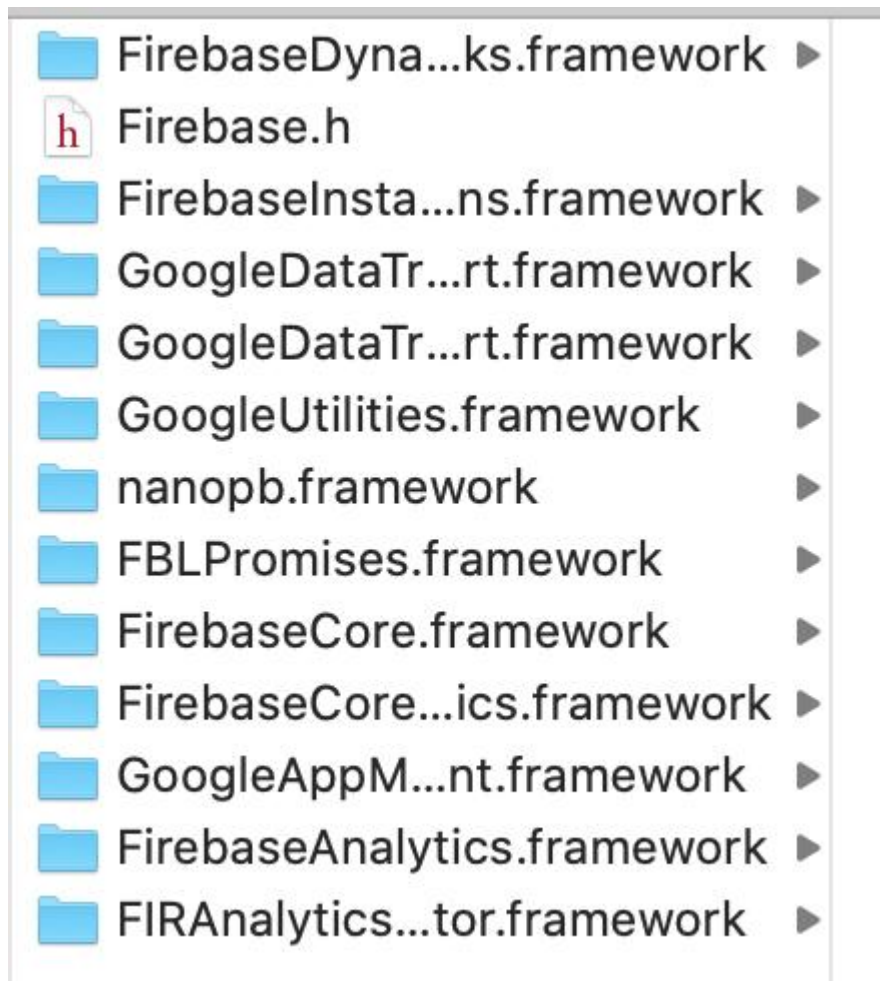
<https://firebase.google.com/docs/ios/setup#available-pods>

```
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>

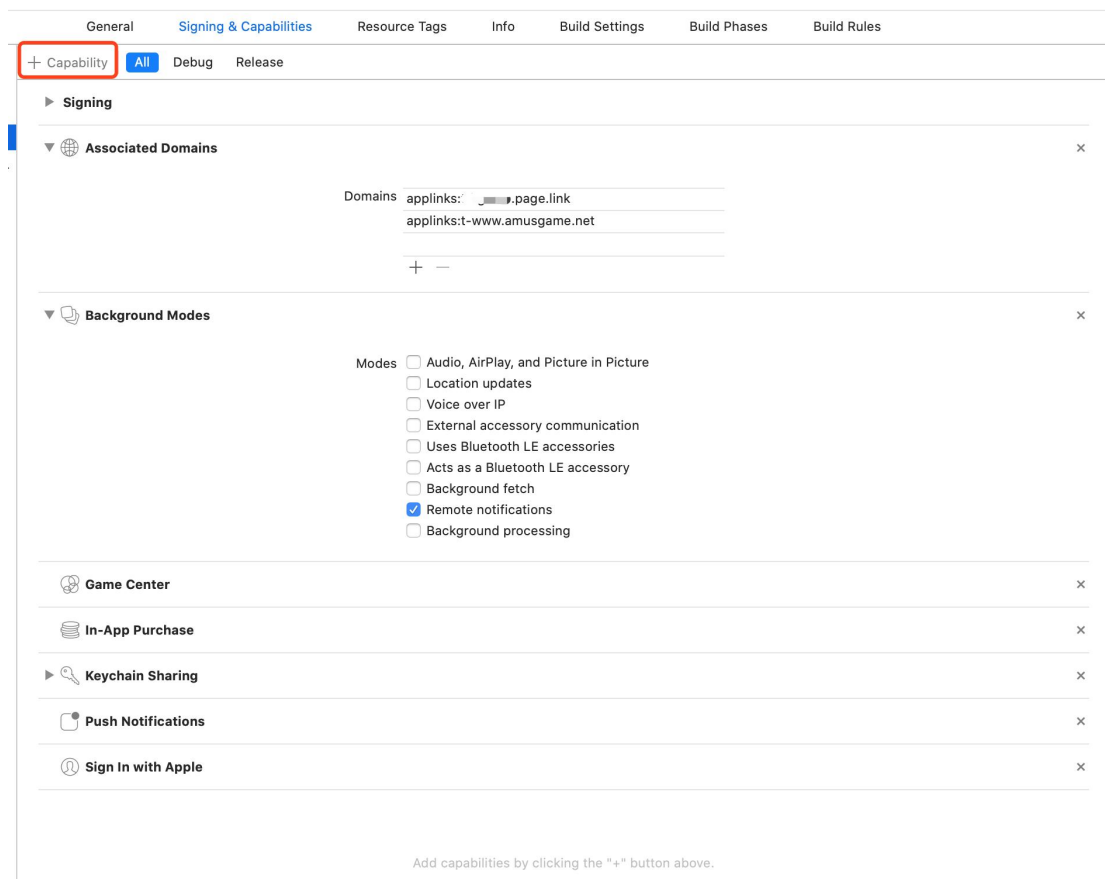


## **4. Project Configuration**

### **4.1 Build Configuration**

**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



#### 4.1.3 Set plist.info

Relevant function configuration:

### Facebook related:

key: CFBundleURLSchemes value: fbxxxxxxx (fb+FacebookAppID)  
key: CFBundleURLSchemes value: xxx.xxx.xxx (bundle id)  
key: FacebookAppID value: xxxxxxxx (FacebookAppID)  
key: AppName value: xxxxxxxx (AppName)

### Twitter related:

key: CFBundleURLSchemes value: twitterkit-xxxxxxx (twitterkit+Twitter key)  
key: TwitterConsumerKey value: xxxxxxxx (Twitter key)  
key: TwitterConsumerSecret value: xxxxxxxx (Twitter secret)

### 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>xxxxxxx</string>  
<key>AppID</key>  
<string>xxxxxxx</string>  
<key>TapjoyKey</key>  
<string>xxxxxxx</string>  
<key>TwitterConsumerKey</key>  
<string>xxxxxxx</string>  
<key>TwitterConsumerSecret</key>  
<string>xxxxxxx</string>  
<key>CFBundleURLTypes</key>  
<array>  
<dict>  
<key>CFBundleURLSchemes</key>  
<array>
```

```
<string>fbxxxxxxxx</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-xxxxxxxx</string>
</array>
</dict>
</array>
<key>FacebookAppID</key>
<string>xxxxxxxx</string>
<key>FacebookDisplayName</key>
<string>AppName</string>
<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>
    <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>

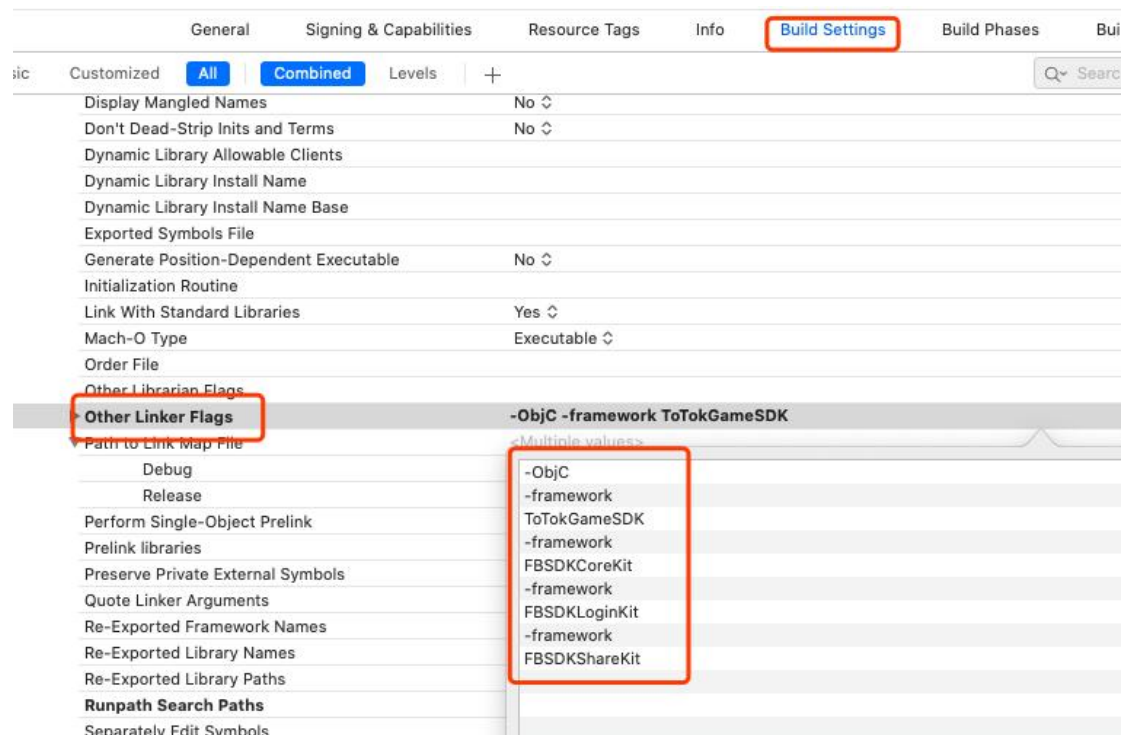
</dict>

```

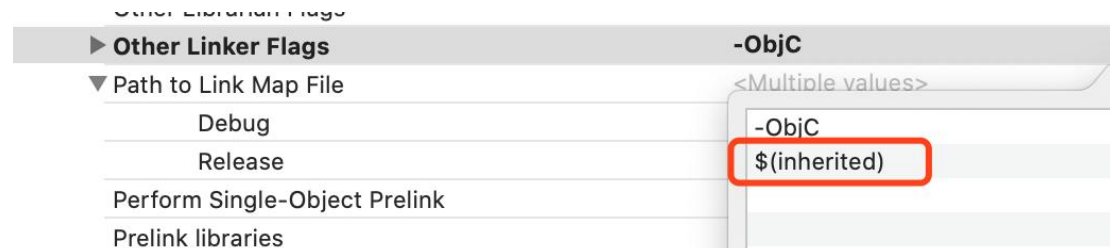
If you choose pod installation, the above configuration is enough. Otherwise, please proceed to the following configuration steps.

#### 4.1.4 Build configuration

Add Other Linker Flags (Pay attention to case)

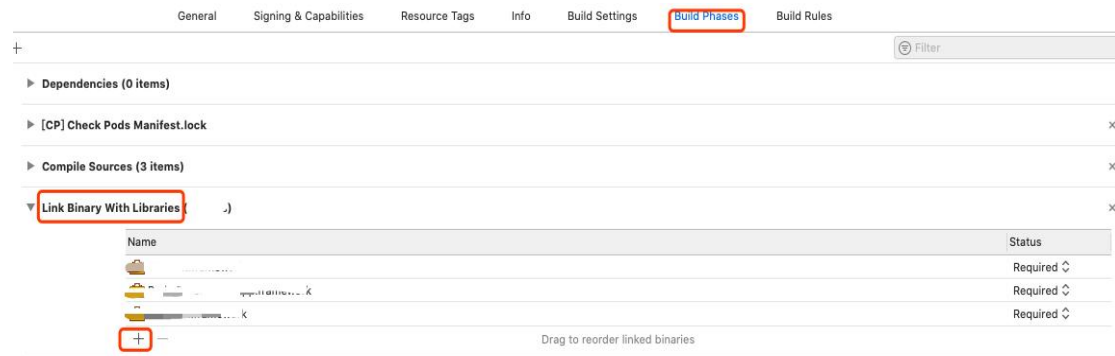


#### And firebase related



**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'



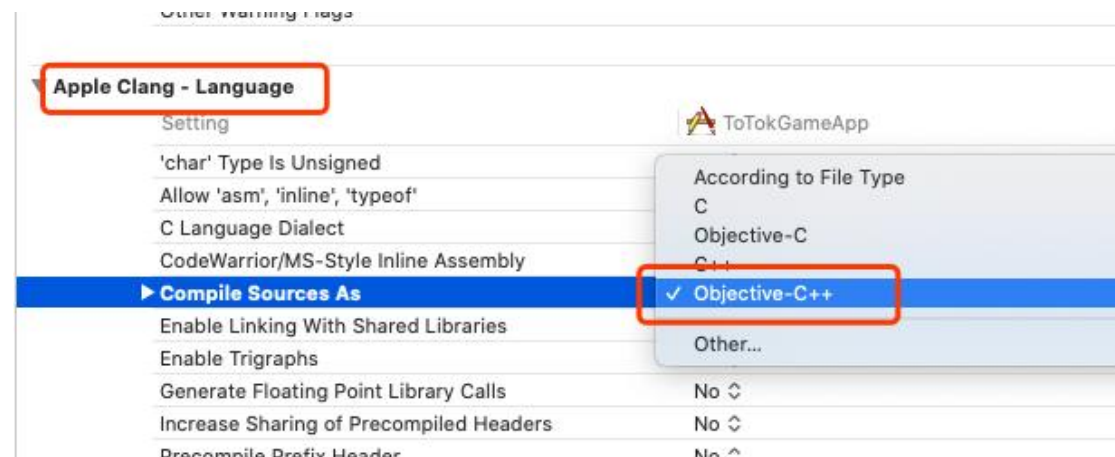
Add the system library required above here.

#### 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:

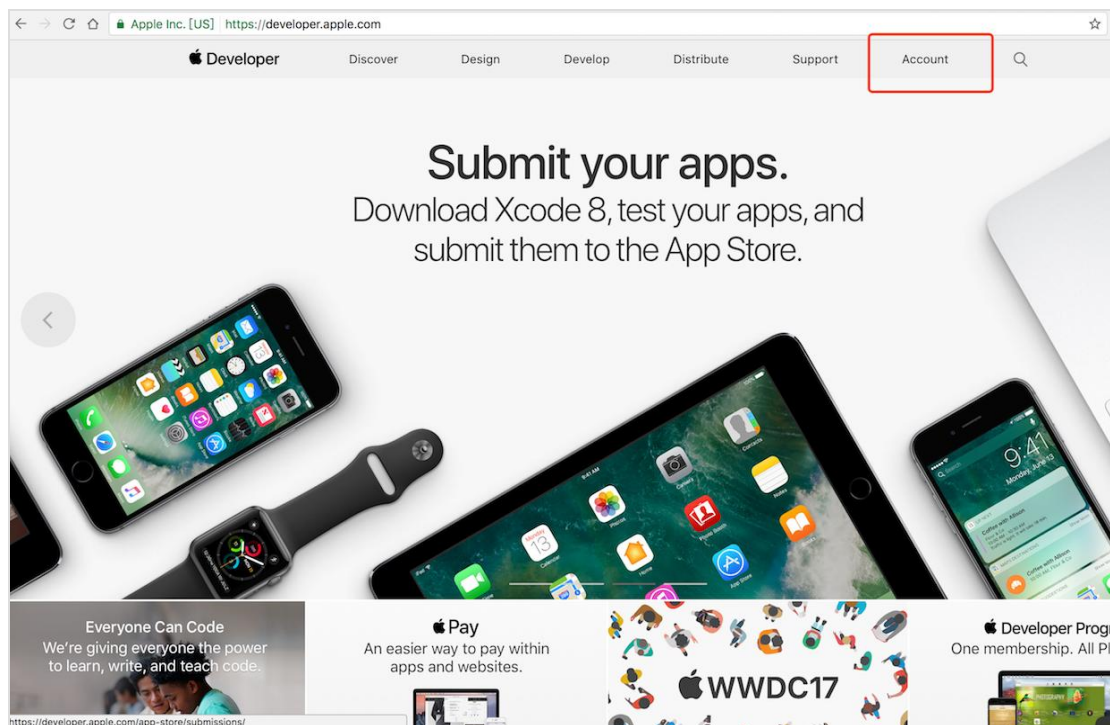
1. No need to re-generate the push certificate every year.
2. One auth key can be used for all your apps – this avoids the complication of maintaining different certificates.

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.

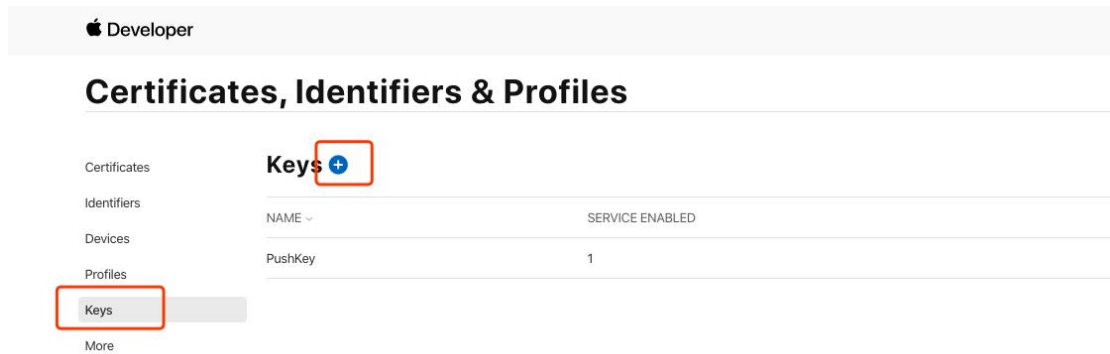
- Log in to the Apple Developer Website and enter your developer account.



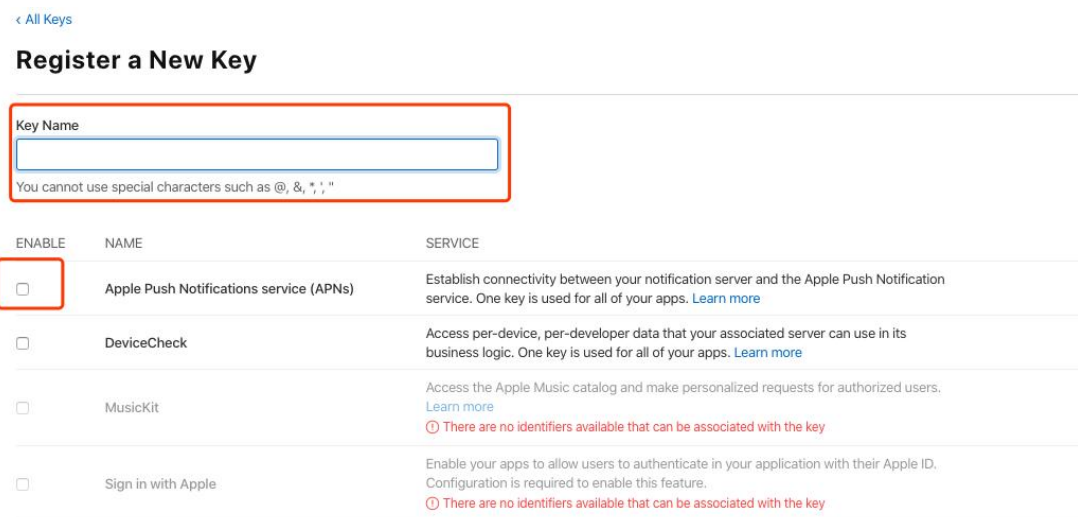
- In your developer account, go to Certificates, Identifiers & Profiles, and

under Keys, select All.

- Click the Add button (+) in the upper-right corner.



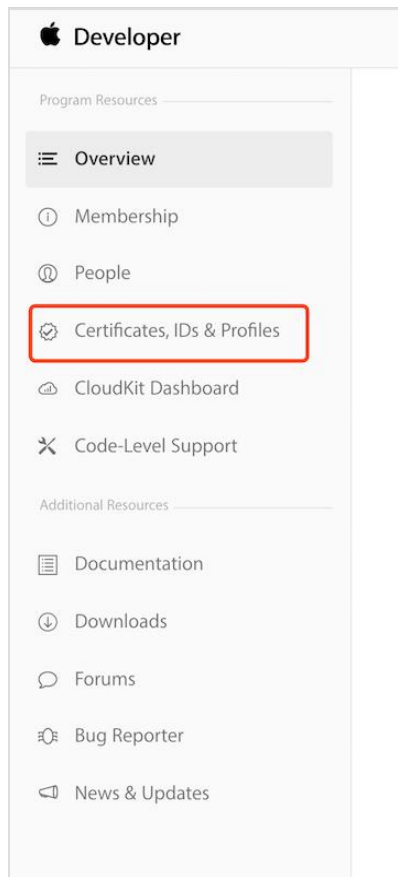
- Enter a description for the APNs Auth Key
- Under Key Services, select the APNs checkbox, and click Continue.



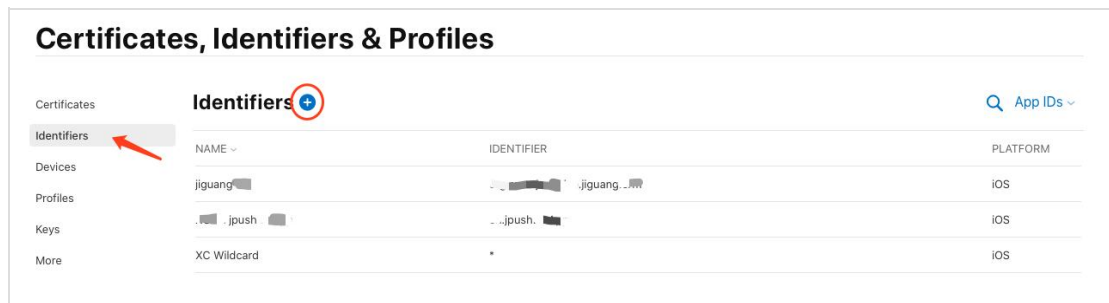
- Click Confirm and then Download. Save your key in a secure place. This is a one-time download, and the key cannot be retrieved later.

## 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.

## Certificates, Identifiers & Profiles

< All Identifiers

### Register an App ID

Back Continue

Platform  
☒ iOS, tvOS, watchOS ☐ macOS

Description  
  
You cannot use special characters such as @, &, '., '., \*

App ID Prefix  
 com.push.demo (Team ID)

Bundle ID  
☒ Explicit ☐ Wildcard  
  
We recommend using a reverse-domain name style string (i.e., com.domainname.appname). It cannot contain an asterisk (\*).

### Capabilities

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

< All Identifiers

### Register an App ID

Back Continue

**Capabilities**

ENABLED	NAME
<input type="checkbox"/>	Access WiFi Information
<input type="checkbox"/>	App Groups
<input type="checkbox"/>	Apple Pay Payment Processing
<input type="checkbox"/>	Associated Domains
<input type="checkbox"/>	AutoFill Credential Provider
<input type="checkbox"/>	Network Extensions
<input type="checkbox"/>	NFC Tag Reading
<input type="checkbox"/>	Personal VPN
<input checked="" type="checkbox"/>	Push Notifications
<input type="checkbox"/>	Sign In with Apple
<input type="checkbox"/>	SiriKit
<input type="checkbox"/>	System Extension (Supported only on: macOS)

Configure

- 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.

## Certificates, Identifiers & Profiles

Certificates
Identifiers
Devices
Profiles
Keys
More

### Certificates

NAME	TYPE	PLATFORM	CREATED BY	EXPIRATION
Development	iOS Development	iOS	Apple Developer	2020/08/21
Distribution	iOS Distribution	iOS	Apple Developer	2020/08/21
Development	Development	All	Apple Developer	2020/09/22
Distribution	Distribution	All	Apple Developer	2020/09/25
Apple Push	Apple Push Services	iOS	Apple Developer	2020/10/24

- 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.

### Create a New Certificate

[Continue](#)

#### Services

- ☐
**iOS Apple Push Notification service SSL (Sandbox)**
use for develop

Establish connectivity between your notification server and the Apple Push Notification service sandbox environment to deliver remote notifications to your app. A separate certificate is required for each app you develop.
- ☒
**Apple Push Notification service SSL (Sandbox & Production)**
use for develop and production

Establish connectivity between your notification server, the Apple Push Notification service sandbox, and production environments to deliver remote notifications to your app. When utilizing HTTP/2, the same certificate can be used to deliver app notifications, update ClockKit complication data, and alert background VoIP apps of incoming activity. A separate certificate is required for each app you distribute.
- ☐
**macOS Apple Push Notification service SSL (Production)**

Establish connectivity between your notification server and the Apple Push Notification service production environment. A separate certificate is required for each app you distribute.
- ☐
**Pass Type ID Certificate**

Sign and send updates to passes in Wallet.
- ☐
**Website Push ID Certificate**

Sign and send updates for Websites.
- ☐
**WatchKit Services Certificate**

Establish connectivity between your notification server, the Apple Push Notification service sandbox, and production environment to update ClockKit complication data. When utilizing HTTP/2, the same certificate can be used to deliver app notifications, update ClockKit complication data, and alert background VoIP apps of incoming activity. A separate certificate is required for each app you distribute.
- ☐
**VoIP Services Certificate**

Establish connectivity between your notification server, the Apple Push Notification service sandbox, and production environment to alert background VoIP apps of incoming activity. A separate certificate is required for each app you distribute.
- ☐
**Apple Pay Payment Processing Certificate**

Decrypt app transaction data sent by Apple to a merchant/developer.
- ☐
**Apple Pay Merchant Identity Certificate**

A client TLS certificate that is used to authenticate you to Apple Pay Payment Processing Servers. You need to accept the agreement 'Apple Pay Platform Web Merchant Terms and Conditions'. [Review Agreement](#)

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

## Certificates, Identifiers & Profiles

[← All Certificates](#)

### Create a New Certificate

[Back](#)

[Continue](#)

#### Select an App ID for your Apple Push Notification service SSL Certificate (Sandbox & Production)

All App IDs that you want to enable for remote notifications require their own Apple Push Notification service SSL certificate. The App ID-specific SSL certificate allows your server to connect to the Apple Push Notification service. Note that only explicit App IDs with a specific Bundle Identifier can be used to create an Apple Push Notification service SSL certificate.

App ID:

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

## Certificates, Identifiers & Profiles

[← All Certificates](#)

### Create a New Certificate

[Back](#)

[Continue](#)

#### Upload a Certificate Signing Request

To manually generate a Certificate, you need a Certificate Signing Request (CSR) file from your Mac.

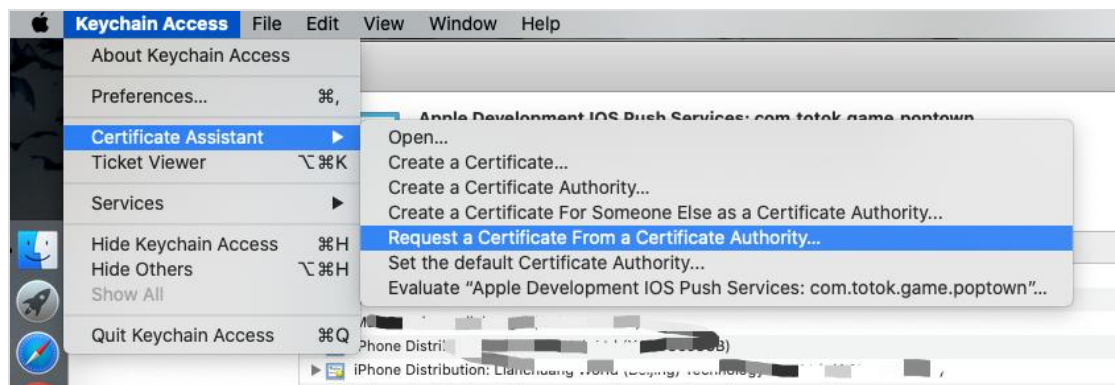
[Learn more >](#)

[Choose File](#)

CertificateSigningRequest.certSigningRequest

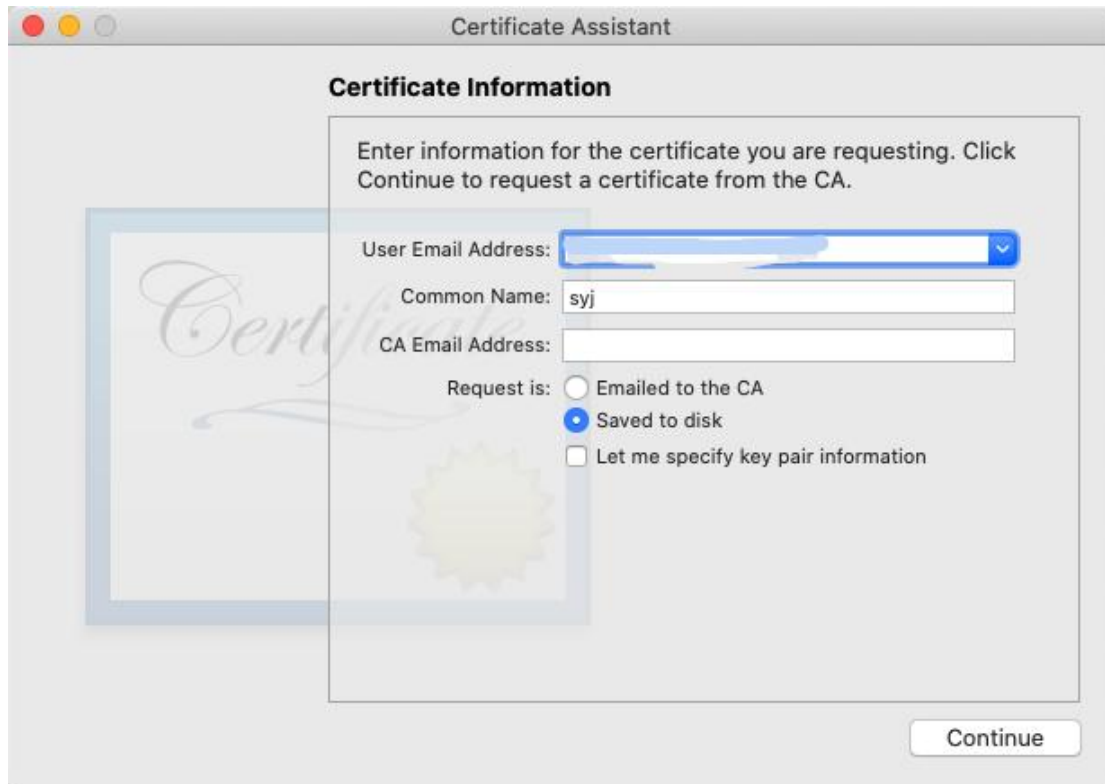
- Open KeychainAccess that comes with the mac system to create a Certificate Signing Request.

Operate as shown below:

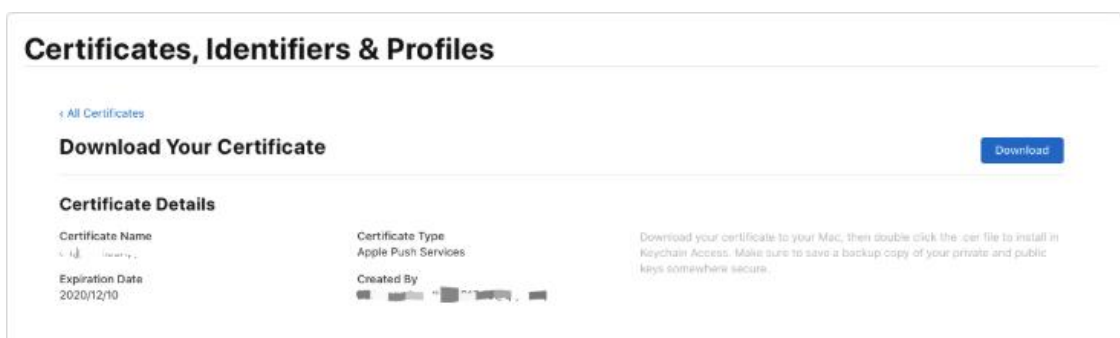


- Fill in "User Email" and "Common Name" and select "Save to Disk". The certificate file suffix is .certSigningRequest.

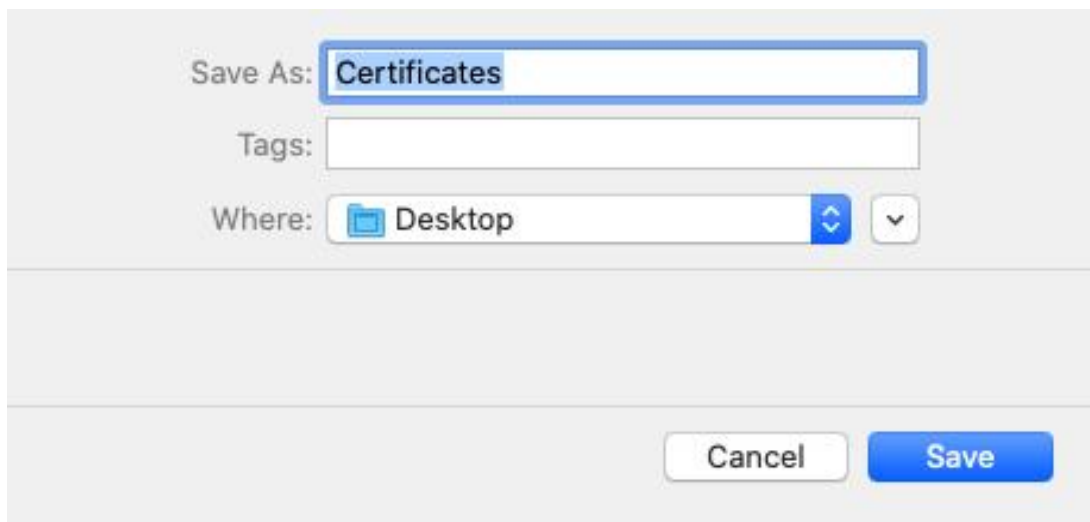
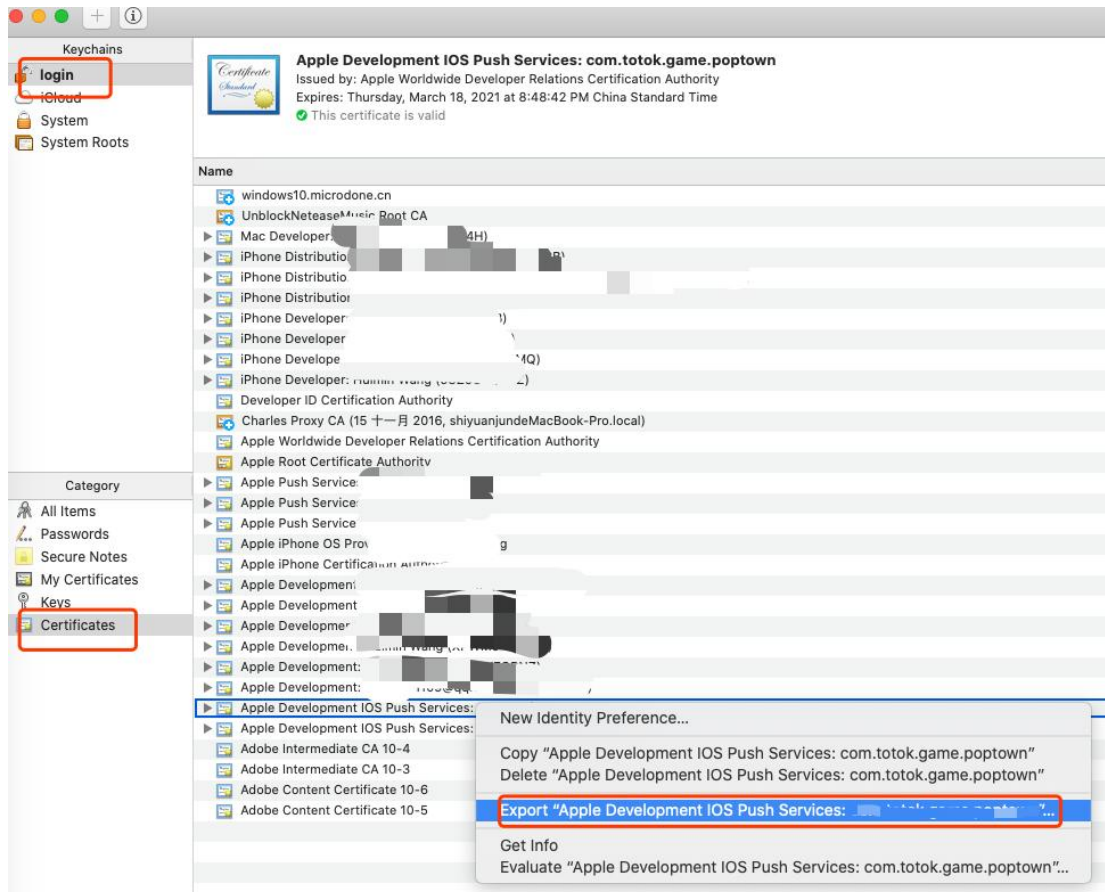




- Go back to the CSR upload page in your browser and upload the file with the suffix .certSigningRequest just generated.
- After the certificate is generated successfully, click the "Download" button to download the certificate. It is a file with the extension .cer.



- 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

[< All Identifiers](#)

## Edit your App ID Configuration

<input type="checkbox"/>	 Maps ⓘ	
<input type="checkbox"/>	 Multipath ⓘ	
<input type="checkbox"/>	 Network Extensions ⓘ	
<input type="checkbox"/>	 NFC Tag Reading ⓘ	
<input type="checkbox"/>	 Personal VPN ⓘ	
<input checked="" type="checkbox"/>	 Push Notifications ⓘ	<a href="#">Edit</a> Certificates (1)
<input checked="" type="checkbox"/>	 Sign In with Apple ⓘ	<a href="#">Edit</a> Enable as a primary App ID
<input type="checkbox"/>	 SiriKit ⓘ	
<input type="checkbox"/>	 System Extension ⓘ	

## Register a New Key for Sign in with Apple

Apple Developer

## Certificates, Identifiers & Profiles

Certificates	<b>Keys +</b>						
Identifiers							
Devices							
Profiles							
Keys							
More							
	<table><thead><tr><th>NAME</th><th>SERVICE ENABLED</th></tr></thead><tbody><tr><td>PushKey</td><td>1</td></tr><tr><td>SigninKey</td><td>1</td></tr></tbody></table>	NAME	SERVICE ENABLED	PushKey	1	SigninKey	1
NAME	SERVICE ENABLED						
PushKey	1						
SigninKey	1						

[< All Keys](#)

## Register a New Key

[Continue](#)

Key Name

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

ENABLE	NAME	SERVICE	
<input type="checkbox"/>	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. <a href="#">Learn more</a>	
<input type="checkbox"/>	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. <a href="#">Learn more</a>	
<input type="checkbox"/>	MusicKit	Access the Apple Music catalog and make personalized requests for authorized users. <a href="#">Learn more</a> ⓘ There are no identifiers available that can be associated with the key	<a href="#">Configure</a>
<input checked="" type="checkbox"/>	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.	<a href="#">Configure</a>

## Configure the App ID

[← View Key](#)

## 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

## • Common Problem

### 1. pod related

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

```
→ ToTokGameApp git:(master) ✗ pod install
Analyzing dependencies
[!] Unable to find a specification for 'ToTokGameSDK'

You have either:
* out-of-date source repos which you can update with 'pod repo update' or with 'pod install --repo-update'.
* mistyped the name or version.
* not added the source repo that hosts the Podspec to your Podfile.
```

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

Key	Type	Value
▼ Information Property List	Dictionary	(15 items)
Localization native development re...	String	\$(DEVELOPMENT_LANGUAGE)
Executable file	String	\$(EXECUTABLE_NAME)
Bundle identifier	String	\$(PRODUCT_BUNDLE_IDENTIFIER)
InfoDictionary version	String	6.0
Bundle name	String	\$(PRODUCT_NAME)
Bundle OS Type code	String	\$(PRODUCT_BUNDLE_PACKAGE_TYPE)
Bundle versions string, short	String	1.0
Bundle version	String	1
Application requires iPhone enviro...	Boolean	YES
▶ Application Scene Manifest	Dictionary	(2 items)
Launch screen interface file base...	String	LaunchScreen
Main storyboard file base name	String	Main
▶ Required device capabilities	Array	(1 item)
▶ Supported interface orientations	Array	(3 items)
▶ Supported interface orientations (i...	Array	(4 items)

## 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

```
#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

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