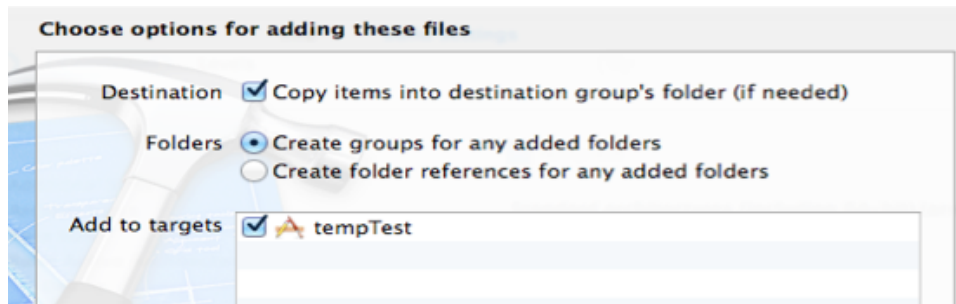


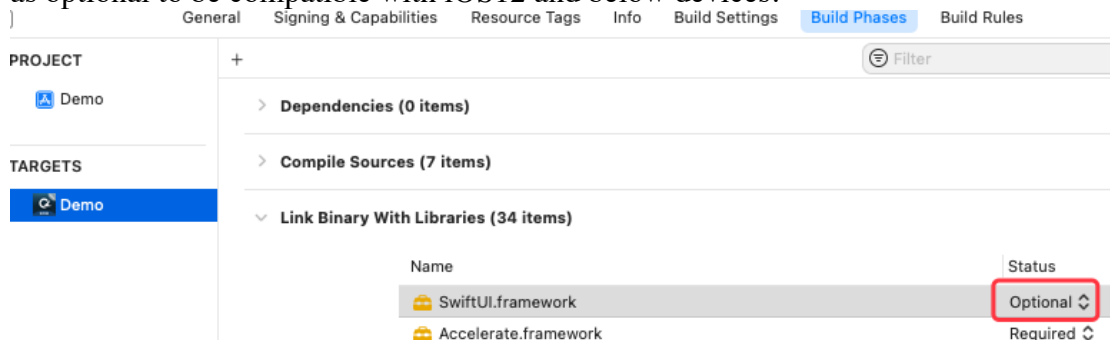
iOS SDK Document for Firebase data

Import SDK files

Drag into the relevant framework under the Analytics folder under the SDK folder and select the correct target.



FirebaseSDK9.1.0 and above need to add the system dependency library SwiftUI.framework in target->Build Phases->Link Binary With Libraries and set it as optional to be compatible with iOS12 and below devices.



For the xcode project exported using Unity version 2019.3 and later, it contains the UnityFramework dynamic library. When importing the SDK file, you need to pay attention to:

The .framework files in the Analytics folder are all static libraries, and TargetMembership needs to be associated with UnityFramework;

The .bundle resource file TargetMembership needs to be linked to Unity-iPhone.

Xcode configuration

1. Get the configuration file of the iOS app
 - 1.1 Sign in to Firebase and open your project.
<https://firebase.google.com/docs/ios/setup>



1.2 click

And select project settings

1.3 In your application card, select the package ID of the application for which you need to obtain a profile from the list.

1.4 click  to download **GoogleService-Info.plist**.

1.5 Put the downloaded plist file in the root directory of the xcode project

Note: The xcode project without this plist file will crash abnormally

The event has been delivered inside the SDK (the access party does not need to process it)

Firebase SDK has delivered installation, startup and other events. The default events recorded by this SDK are:

1.Login success: kFIREEventSignUp

If you need to deliver other events, the game needs to call the following custom event delivery interface at the appropriate time to deliver the event.

Interface call

```
import REDeLoginKit.h
```

```
#import <JYouLoginKit/REDeLoginKit.h>
```

Init

**** It is recommended to call this method to initialize firebase when initializing the SDK. ****

```
/** start init */  
+ (void)configFireBase;
```

Code example:

```
[REDeLoginKit configFireBase];
```

Delivery custom event interface

```
/** Statistics custom events. paramDict: Statistical parameters */  
+ (void)logEventWithName:(NSString *)eventName  
    andWithParam:(NSDictionary *)paramDict;
```

Delivery virtual currency income event interface

```
/** Statistics of virtual currency income */  
+ (void)logGetVirtualCurrencyWithCurrencyName:(NSString *)currency-  
Name  
                                andWithValue:(NSString *)value;
```

Post to join group event interface

```
/** Statistics join group */  
+ (void)logJoinGroupWithGroupId:(NSString *)groupId;
```

Post role upgrade event interface

```
/** Statistics role upgrade */  
+ (void)logLevelUpWithLevel:(NSString *)level  
                                andWithCharacter:(NSString *)character;
```

Delivery event interface of virtual currency expenditure

```
/** Statistics on virtual currency expenditures */  
+ (void)logSpendVirtualCurrencyWithItemName:(NSString *)itemName  
                                andWithVirtualCurrencyName:(NSString *)currency-  
Name  
                                andWithValue:(NSString *)value;
```

Delivery start learning event interface

```
/** Statistics start learning */  
+ (void)logTutorialBegin;
```

Post learning end event interface

```
/** End of statistical learning */  
+ (void)logTutorialComplete;
```