INFINIT'GAME Platform

InfinitSDK 1.0.3

"Get the most of your games"

Document History

Document Version	Purpose of the release	Changes
1.0	Initial release	-
1.0.1	Update the imports+calls	- Added proper import tutorial - Added a tech description of all the different calls
1.0.2	Add call for loadProxyObject + send score	- Added "List of available API calls / Main Branch / loadProxyObject" description - Updated the full example process to include the positive callback triggering and loadProxyObject - Added description for callAPIForSendScore
1.0.3	Revamp of the APIs + new UX flow	- Removed APIConfig.h - Changed the API Configuration method (See APIs/Setup APIs/2.) - Stronger API - Changed the .h file packages. Some files have gone, some new are in

InfinitSDK

Description

InfinitSDK is designed to allow access to 3rd-parties to the benefits of INFINIT'GAME platform.

This platform allows your app to be listed on INFINIT'GAME application for users to download your games and get an access to the extra-features that you will have decided.

How to use the SDK

The SDK allows to do 2 things:

1. Check the user subscription and grant access if the result was positive
2. Launch the user subscription if not

When the user has been granted access, you can then "**unlock**" some features, some levels or even the full game via a simple API call (see APIs part of the documentation).

What do I need?

You need to have a bunch of things before starting:

1. Your application
2. The scenario of what will happen, where you set the different screens, and what to unlock
3. Information provided by INFINIT'GAME team
    a. **API Key** ; *Allowing you access to the SDK*
    b. **Partner ID** ; *Your INFINIT'GAME number*
    c. Game ID ; *For each game you will be doing, you will need a different Game ID*
    d. Game app ID ; *The AppStore ID of your app, typically of the pattern com.infinitgame.**

APIs

Description of the package

The package contains the following files:

- ~~**APIConfig.h** ; *This is the file that you will need to reconfigure with the parameters given by INFINIT'GAME team*~~ *(removed in 1.0.3)*

- **Constants.h** ; *A bunch of constants useful for the APIs to work*

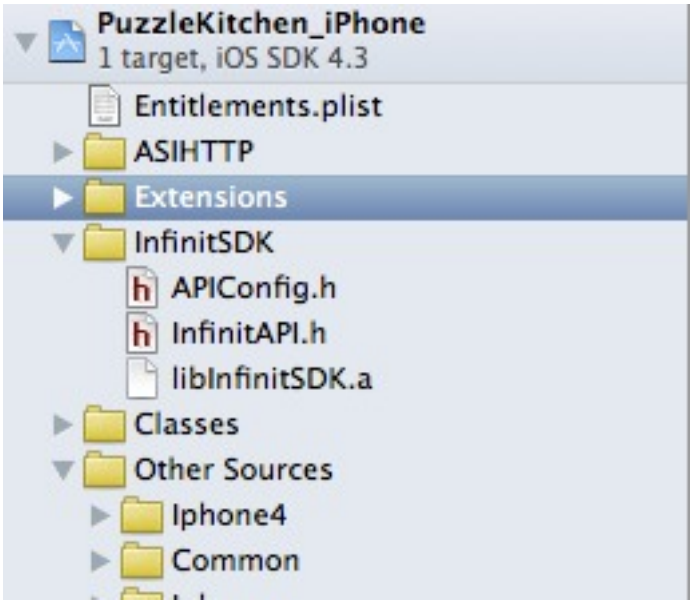
- **InfinitAPI.h** ; *The main SDK h file that tells xCode how the objects have been designed*

- **libInfinitSDK.a** ; *The a library by itself. It is to be dropped into xCode project (see Setup InfinidSDK)*

- **InfinitSDK - Documation.doc** ; *The current document*

Setup InfinitSDK in your application

1. Extract the SDK Package and drag/drop the ~~**APIConfig.h**~~, **InfinitAPI.h** and **libInfinitSDK.a** into a new "Group" in xCode named "**InfinitSDK**" by convention.



2. ~~Open the file "**APIConfig.h**" and edit the following line with your own information provided by INFINIT'GAME team.~~

NB: appName and appVersion are free.

```
static NSString *APIKey = @"8df639b301a1e10c36cc2f03bbdf8863";
static NSInteger PartnerID = 1;
static NSString *appName = @"PuzzleKitchen";
static NSString *appVersion = @"1.0";
static NSString *gameID = @"5";
```

1.0.3 UPDATE:

The 1.0.3 APIs change the way to configure the API with given parameters.

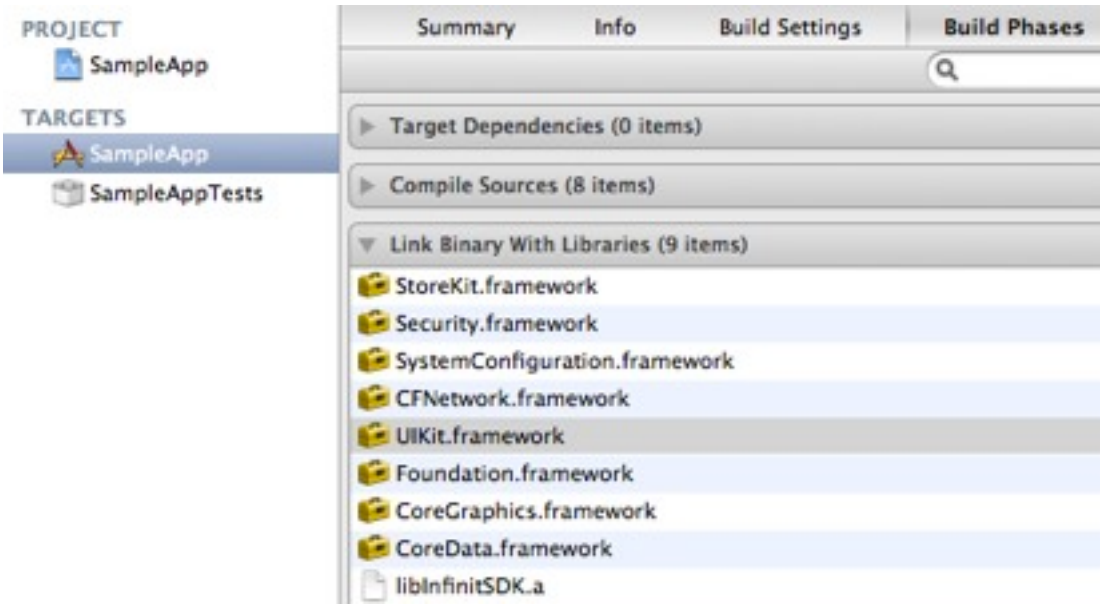
In the AppDelegate, call the following functions:

```
api = [InfinitAPI sharedInstance];
[api setAPIKey:@"nil"];
[api setAppName:@"App"];
[api setAppVersion:@"1.0"];
[api setPartnerID:1];
[api setGameID:@"1"];
```

3. Add the JSON library (added into the package) + the following .h files (this should change in the future)

4. Configure your xCode Project to the given Identifier

5. Make sure to include at the following Frameworks



6. Include the image resources contained into the zip file: *"InfinitSDK-1.0-To Include-Image Source.zip"*

You should be ready!

List of available API calls

Main Branch

API calls that will play a major role into the process

- **callAPIForQuietLogin** ; *This is the setup call (modal) that resumes the user session.*

- to be called at the beginning of the app. Typically AppDelegate:didFinishLaunchingWithOptions

- **callPopupForCommercialMessage** ; *This call will open the panel to show contests to come and redirect the user to the Subscription Screen if he's not yet a valid customer (customer having a running subscription)*

- to be called whenever necessary, accordingly to defined scenario with INFINIT'GAME team

- **callPopupForSubscriptionScreen** ; *This call will open the panel to show the user's Account and engage the customer to purchase a subscription if he's not yet a valid customer*

    - *to be called whenever necessary, accordingly to defined scenario with INFINIT'GAME team*

- **callAPIForBuyProcess** ; *This call rules the purchasing process. It will check the status and enter into the appropriate mode*

    - *to be called whenever necessary. You typically don't need it as it is in the callPopupForSubscriptionScreen call.*

- **loadProxyObject:***__class__* **selector:@selector**(*__callback_method__*) ; *This method loads the callback that will be triggered after the purchasing process, if the purchasing is successful.*

    - *to be called before the "Account" panel or before the user is supposed to purchase the package. After the purchase, the given selector from the given object (typically "self") will be triggered to allow refreshing the UI with the unlocked material.*

- callAPIForSendScore:(NSInteger)score at:(NSString*)date ; *This method will send the given score (NSInteger) at the given date (NSString*). It is recommended to set date to nil to let the system manage the date.*

    - *to be called when you need to send a score for the current user.*

    - *make sure that you configured the appropriate "GameID" and "PartnerID" in APIConfig.h*

Utils

API calls that will provide vualuable information and helpeers on the current instance

- **isValidCustomer** ; *This function returns true if the current customer is a valid customer. (Currently having a running subscription)*

- **sharedInstance** ; *This function returns a pointer to the APIs. It is static hence can be called anytime using [InfinitAPI sharedInstance] and will always return (InfinitAPI*) pointer.*

    - *Requires #import "InfinitAPI.h"*

User Information

API calls to provide user's personal information

- **getUserName** ; *Returns the current user's pseudo*

- **getUserStartSubscription** ; *Returns the current user's subscription start date (tyoe: setDateFormat:@"yyyy-MM-dd HH:mm:ss")*

- **getUserEndSubscription** ; *Returns the current user's subscription end date (tyoe: setDateFormat:@"yyyy-MM-dd HH:mm:ss")*

- **getUserEmail** ; *Returns current user's email*

- **getUserID** ; *Returns current user's unique ID into INFINIT'GAME platform*

- **getUserAvatar** ; *Returns current user's avatar (type: PNG, encoding: Base64)*

- **getUserAge** ; *Returns current user's age*

- **getUserGender** ; *Returns current user's Gender*

Example of a full process

1. App Launch
2. System enters into AppDelegate:didFinishLaunchingWithOption
    a. init api with [InfinitAPI sharedInstance]
    b. do some internal init (normal app init)
    c. launch the modal API call [api callAPIForQuietLogin:self selector:@selector(apiDidEnd)]

i. will auto-redirect to apiDidEnd function after the session is resumed

3. App launches and process normally
4. **Case**: User wants to show the Commercial Panel
    a. call [api callPopupForCommercialMessage]
5. **Case**: User wants to show the Subscription Panel
    a. Load the callback via the method [api loadProxyObject:self selector:@selector (purchaseIsSuccessful)];
    b. call [api callPopupForSubscriptionScreen]
    c. user clicks on "Go! Purchase"
        i. System launches the purchase process with In-App panels
        ii. Confirmation screen
            1. The purchase process triggers the callback self:purchaseIsSuccessful
6. **Case**: **User wants to play a function that has been "locked" to INFINIT'GAME subscribers**
    a. call [api isValidCustomer]
        i. If returns true, grant access
        ii. Otherwise, show the Commercial Panel

Sample application

Please have a look at the sample application for more info.

Have fun and monetize using INFINIT'GAME platform' SDK.