

---

# NAVER Cafe SDK インテグレーション ガイド

---

---

# 著作権

---

Copyright © NAVER Corp. All Rights Reserved.

本書は、NAVER(株)の知的財産であるためいかなる場合にも NAVER(株)の許可なく本書の一部または全部をコピー・転送・配布・変更して使用することはできません。

本書は、情報提供の目的に限って提供されます。NAVER(株)は、本書に収められた情報については完全性と正確性を期しておりますが、記載内容の誤りや漏れによって発生する問題については責任を負いません。

したがって、本書の使用や使用した結果に伴う責任は全面的に利用者であり、NAVER(株)はこれについて明示的または黙示的にいかなる保証もいたしません。

関連 URL 情報を含め、本書で言及した特定のソフトウェア商品および製品は、当該所有者の属した現地および国内外の関連法に従い、当該法律を遵守しないことにより発生するあらゆる結果に対する責任は全面的に利用者自身にあります。

NAVER(株)は、本書の内容を予告なく変更する場合があります。

---

# はじめに

---

## 本書について

本書では、NAVER Cafe SDK をアプリケーションに組み込む方法について説明します。

## 読者

本書の読者は、アプリケーションへの NAVER Cafe SDK 組み込み作業を行う開発者です。

## お問い合わせ

本書の内容に関する問い合わせは下記までご連絡ください。

連絡先: [dl\\_gamesdkpartner@navercorp.com](mailto:dl_gamesdkpartner@navercorp.com)

## 改版履歴

日付	履歴
2015 年 12 月 23 日	初版発行

---

# 表記規則

---

## 参考表記

---

### 参考

読者が参考にすべき内容を記述します。

---

## ユーザー入力値

本書において UI 文言、ユーザーの固定・可変入力値は下記のように表記します。

- UI 文言: **メニュー** > **サブメニュー**
- ユーザー固定入力値: **localhost** を入力します。
- ユーザー可変入力値: <http://www.naver.com/>{企業名}

## ソースコードの表記

本書においてソースコードは、灰地に黒字で表記します。

```
COPYDATASTRUCT st;  
st.dwData = PURPLE_OUTBOUND_ENDING;  
st.cbData = sizeof(pp);  
st.lpData = &pp;  
::SendMessage(GetTargetHwnd(), WM_COPYDATA, (WPARAM) this->m_hWnd, (LPARAM) &st);
```

---

# 目次

---

<b>SDK 概要</b>	<b>9</b>
NAVER Cafe SDK とは？	9
NAVER Cafe SDK の機能と特徴	9
主要機能	9
使用環境	9
<b>Android 向け</b>	<b>11</b>
開発環境	11
ソフトウェア要件	11
ライブラリの構成	12
開発環境セットアップ	12
NAVER Cafe SDK の組み込み手順	13
AndroidManifest.xmlの設定	13
NAVER Cafe SDKの初期化	13
NAVER Cafe SDK画面を開く	13
NAVER Cafe SDK画面を閉じる	14
投稿作成	14
App Schemeの処理	14
ユーザーのゲームIDとのマッピング	14
リソース画像の変更	15
API リファレンス	15
Glink	15
init()	15
isShowGlink()	16
popBackStack()	17
setGameUserId()	17
startEvent()	18
startImageWrite()	18
startHome()	19
startMenu()	19
startNotice()	20

---

startProfile()	20
startVideoWrite()	21
startWrite()	21
stop()	22
<b>iOS 向け</b>	<b>23</b>
<b>開発環境</b>	<b>23</b>
ソフトウェア要件	23
ライブラリの構成	23
開発環境セットアップ	24
<b>NAVER Cafe SDK の組み込み手順</b>	<b>27</b>
NAVER Cafe SDKの初期化	27
NAVER Loginの設定	27
NAVER Cafe SDK画面を開く	27
投稿作成	28
<b>API リファレンス</b>	<b>28</b>
NCSDKManager	28
@property (nonatomic, weak) id parentViewController	29
(void)dismissViewController	29
(void)dismissTopViewController	29
(NCSDKManager *)getSharedInstance	30
(id)NAVER CafeRootViewController	30
(void)resetSharedInstance	31
(void)presentArticlePostViewControllerWithMenuId	31
(void)presentArticlePostViewControllerWithType	32
(void)presentMainViewController	33
(void)presentMainViewControllerWithArticleId	33
(void)presentMainViewControllerWithTabIndex	33
(void)presentViewController	34
(void)setGameUserId	34
(void)setNaverLoginClientId	35
NCSDKLoginManager	35
@property (nonatomic, weak) UIViewController *rootViewController	36
(NSSString *)accessToken	36
(NSSString *)accessTokenExpireTime	37
(BOOL)finishNaverLoginWithURL	37
(NCSDKLoginManager *)getSharedInstance	38
(void)isLoginWithFinish	38
(BOOL)isValidAccessTokenExpireTimeNow	38
(void)loginWithFinish	39
(void)logout	39
(void)refreshAccessToken	40

(void)refreshAccessTokenWithFinish	40
(void)requestDeleteToken	41
(void)setIsInAppOAuthEnable	41
(void)setIsNaverAppOAuthEnable	42
<b>Unity 向け</b>	<b>43</b>
<b>開発環境</b>	<b>43</b>
ソフトウェア要件	43
ライブラリの構成	43
開発環境セットアップ	43
<b>NAVER Cafe SDK の組み込み手順</b>	<b>47</b>
実行	47
ユーザーのゲームIDとNAVER IDとのマッピング	47
AFNetworkingの交替	48
<b>API リファレンス</b>	<b>48</b>
GLinkUnity	48
executeArticlePost()	48
executeArticlePostWithImage()	49
executeArticlePostWithVideo()	50
executeMain()	50

---

# 図表一覧

---

## 表一覧

表 1 Android向けNAVER Cafe SDKライブラリ	11
表 2 iOS向けNAVER Cafe SDKライブラリ	23

## 図一覧

図 1 NAVER Cafe SDK	9
--------------------	---



# SDK 概要

## NAVER Cafe SDK とは？

NAVER Cafe SDK はゲームエンジンに関係なくモバイルゲームに NAVER Cafe を簡単に組み込めるライブラリです。

NAVER Cafe SDK を組み込むことでゲームユーザーはゲームを続けたままゲームコミュニティ(NAVER Cafe)との交流が可能になります。



図 1 NAVER Cafe SDK

## NAVER Cafe SDK の機能と特徴

### 主要機能

NAVER Cafe SDK の主な機能は下記のとおりです。

- iOS、Android に対応
- Unity 4 以上と Cocos2d-x 2.1 以上のゲームエンジンに対応
- Cafe 機能を利用できる画面を提供
- OAuth 2.0 ベースの NAVER Login を用いたユーザー認証

### 使用環境

#### オペレーティングシステム

NAVER Cafe SDK は以下の環境で動作します。

- Android: Android 4.2(Jelly Bean) API Level 17 以上
- iOS: iOS 7.0 以上(ARMv7、ARMv7s、ARM64 に対応)

#### NAVER Login の登録

NAVER Cafe SDK において NAVER Login 機能を用いてユーザー認証を行うにはクライアント ID とクライアントシークレットが必要です。

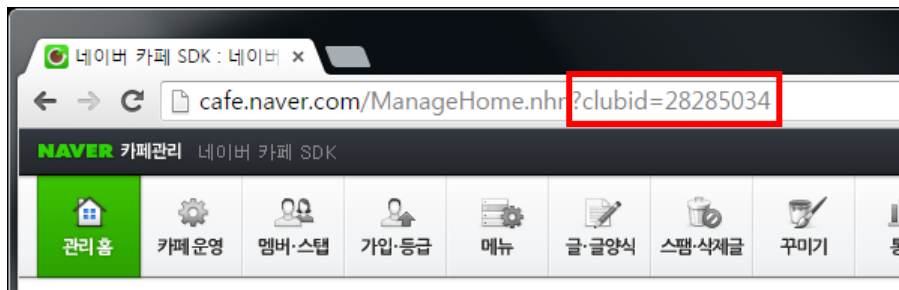
- クライアント ID: NAVER Login にゲームを登録すると発行されるクライアント ID。
- クライアントシークレット: NAVER Login にゲームを登録すると発行されるクライアントシークレット。

クライアント ID とクライアントシークレットは、NAVER Login にアプリケーションを登録すると発行されます。NAVER Login にアプリケーションを登録してクライアント ID とクライアントシークレットを確認する方法については、NAVER Login Developers のドキュメントを参考にするか、または担当([dl\\_gamesdkpartner@navercorp.com](mailto:dl_gamesdkpartner@navercorp.com))までお問い合わせください。

- NAVER Login Developers(韓国語): <https://nid.naver.com/devcenter/main.nhn>

### NAVER Cafe ID

NAVER Cafe SDK を組み込むには NAVER Cafe の Cafe ID が必要です。Cafe ID は、NAVER Cafe 管理ページ URL の **clubid** パラメータ値にあたります。



# Android 向け

## 開発環境

### ソフトウェア要件

#### 開発ツール

- IDE: Android Studio、または Eclipse
- Android Support Library v7

#### ライブラリ

NAVER Cafe SDK を組み込むには、下記のライブラリをプロジェクトに追加してビルドを行います。

表 1 Android 向け NAVER Cafe SDK ライブラリ

ライブラリ	ダウンロード URL
NAVER Login	<ul style="list-style-type: none"><li>• NAVER Cafe SDK ライブラリファイルに含まれている(4.1.4 バージョン)。</li><li>• ダウンロード URL: <a href="https://static.nid.naver.com/images/web/devcenter/3rdparty_login_library_android_4.1.4.zip">https://static.nid.naver.com/images/web/devcenter/3rdparty_login_library_android_4.1.4.zip</a></li></ul>
NAVER Volleyer	<ul style="list-style-type: none"><li>• NAVER Cafe SDK ライブラリファイルに含まれている(2.0.1 バージョン)。</li><li>• ダウンロード URL: <a href="http://mvnrepository.com/artifact/com.navercorp.volleyextensions/volleyer">http://mvnrepository.com/artifact/com.navercorp.volleyextensions/volleyer</a></li></ul>
Volley	<ul style="list-style-type: none"><li>• NAVER Cafe SDK ライブラリファイルに含まれている(1.0.2 バージョン)。</li><li>• ダウンロード URL: <a href="http://mvnrepository.com/artifact/com.mcxiaoke.volley/library">http://mvnrepository.com/artifact/com.mcxiaoke.volley/library</a></li></ul>
Google Gson	<ul style="list-style-type: none"><li>• NAVER Cafe SDK ライブラリファイルに含まれている(2.3.1 バージョン)。</li><li>• ダウンロード URL: <a href="http://mvnrepository.com/artifact/com.google.code.gson/gson">http://mvnrepository.com/artifact/com.google.code.gson/gson</a></li></ul>
Glide	<ul style="list-style-type: none"><li>• NAVER Cafe SDK ライブラリファイルに含まれている(3.6.1 バージョン)。</li><li>• ダウンロード URL: <a href="http://mvnrepository.com/artifact/com.github.bumptech.glide/glide">http://mvnrepository.com/artifact/com.github.bumptech.glide/glide</a></li></ul>
Otto	<ul style="list-style-type: none"><li>• NAVER Cafe SDK ライブラリファイルに含まれている(1.3.8 バージョン)。</li><li>• ダウンロード URL: <a href="http://mvnrepository.com/artifact/com.squareup.otto">http://mvnrepository.com/artifact/com.squareup.otto</a></li></ul>

## ライブラリの構成

Android 向け NAVER Cafe SDK ライブラリは、下記のように構成されています。

- lib: NAVER Cafe SDK ライブラリと必須ライブラリ
  - cafeSdk-x.x.x.aar: Android Studio で使用する NAVER Cafe SDK ライブラリ
  - cafeSdk-x.x.x.zip: Eclipse で使用する NAVER Cafe SDK ライブラリ
  - library: NAVER Cafe SDK を使用する際に必要なライブラリフォルダ
- sample: NAVER Cafe SDK ライブラリを使ったサンプルプロジェクトフォルダ
  - NAVER Cafesdk-sample-android-studio: Android Studio で使える NAVER Cafe SDK のサンプルプロジェクトフォルダ
  - NAVER Cafesdk-sample-eclipse-master: Eclipse で使える NAVER Cafe SDK のサンプルプロジェクトフォルダ

## 開発環境セットアップ

NAVER Cafe SDK を組み込むには下記のように IDE ごとに開発環境をセットアップします。

### Android Studio

1. Android プロジェクトの **libs** フォルダに NAVER Cafe SDK ライブラリファイル(cafeSdk-x.x.x.aar)をコピーします。
2. Android プロジェクトの **libs** フォルダに NAVER Login ライブラリファイル(naveroauthlogin-4.x.x.jar)をコピーします。
3. メニューから、**File > Project Structure** をクリックし、**Project Structure** ダイアログボックスから **App > Dependencies** をクリックして、ライブラリを追加します。または、次のように Android プロジェクトの **build.gradle** ファイルに直接ライブラリを追加します。

```
compile 'com.android.support:support-v13:23.1.0'
compile 'com.navercorp.volleyextensions:volleyer:2.0.1'
compile 'com.google.code.gson:gson:2.3.1'
compile 'com.github.bumptech.glide:glide:3.6.1'
compile 'com.squareup:otto:1.3.8'
```

### Eclipse

1. NAVER Cafe SDK ライブラリファイル(cafeSdk-x.x.x.zip)を解凍します。
2. 解凍した NAVER Cafe SDK ライブラリフォルダ内の **libs** フォルダにビルドに必要な必須ライブラリファイルを追加します。
3. Android プロジェクトの **AndroidManifest.xml** ファイルに、下記のように NAVER Login で使うアクティビティと NAVER Cafe SDK で使うアクティビティを追加します。

```
<activity
    android:name="com.naver.glink.android.sdk.ui.VideoPlayActivity"
    android:screenOrientation="sensorLandscape"
    android:theme="@android:style/Theme.Translucent.NoTitleBar.Fullscreen" />
<activity
    android:name="com.nhn.android.naverlogin.ui.OAuthLoginActivity"
    android:screenOrientation="sensorLandscape"
    android:theme="@android:style/Theme.Translucent.NoTitleBar" />
<activity
    android:name="com.nhn.android.naverlogin.ui.OAuthLoginInAppBrowserActivity"
    android:label="OAuth2.0 In-app"
    android:screenOrientation="sensorLandscape" />
```

## NAVER Cafe SDK の組み込み手順

### AndroidManifest.xml の設定

Android プロジェクトの **AndroidManifest.xml** ファイルに下記の内容を追加します。

package プロパティには、NAVER Login にアプリケーションを登録した際の **Android Intent** の設定内容を入力します。  
NAVER Cafe SDK をコントロールするオブジェクトである Glink クラスのアクセス権も設定します。

```
<!--package は、NAVER Login Developers に登録した Android Intent と同じでなければなりません。-->
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.naver.glink.sample">
<!-- Glink に必要なアクセス権-->
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

### NAVER Cafe SDK の初期化

NAVER Cafe SDK を初期化する機能は、Glink.init()メソッドで実装します。

クライアント ID とクライアントシークレットは、NAVER Login にアプリケーションを登録した際に発行された値です。

```
/**
 * "NAVER Login Developers"の発行情報を利用して SDK の初期化処理を行います。
 * Glink の他のメソッドを呼び出す前に、必ず初期化処理を行ってください。
 * NAVER Login Developers: https://nid.naver.com/devcenter/main.nhn
 */
final int cafeId = 28266581;
final String consumerKey = "クライアント ID";
final String consumerSecret = "クライアントシークレット";
Glink.init(consumerKey, consumerSecret, cafeId);
```

### NAVER Cafe SDK 画面を開く

NAVER Cafe SDK の画面は 5 つのメニュー(タブ)で構成されています。したがって、NAVER Cafe SDK を立ち上げる際のスタート画面をどの画面にするかを定めておく必要があります。

例えば、トップ画面でスタートさせるには、下記のように Glink.startHome()メソッドを使います。

```
// トップ画面でスタートさせます。
Glink.startHome(activity);
```

トップ画面以外のお知らせ、イベント、掲示板、プロフィールメニューでスタートさせるには、下記のメソッドを使います。

- Glink.startNotice()メソッド: お知らせメニューでスタート
- Glink.startEvent()メソッド: イベントメニューでスタート
- Glink.startMenu()メソッド: 掲示板メニューでスタート
- Glink.startProfile()メソッド: プロフィールメニューでスタート

## NAVER Cafe SDK 画面を閉じる

NAVER Cafe SDK 画面を閉じる機能は、Glink.stop()メソッド、または Glink.popBackStack()メソッドで実装します。

- Glink.stop()メソッド: 即時に画面を閉じて NAVER Cafe SDK を終了します。
- Glink.popBackStack()メソッド: backStack に溜まっている画面をひとつずつ閉じます。すべての画面を閉じたら NAVER Cafe SDK を終了します。backStack に溜まる画面は投稿閲覧画面、投稿作成画面、検索画面です。

下記は、Glink.stop()メソッドで NAVER Cafe SDK 画面を閉じる例です。

```
// stop()メソッドで終了する。  
Glink.stop(activity);
```

下記は、Glink.popBackStack()メソッドで NAVER Cafe SDK 画面を閉じる例です。

```
// popBackStack で終了する。  
Glink.popBackStack(activity);
```

## 投稿作成

ゲームユーザーが Cafe で投稿を作成する画面は下記のように Glink.startWrite()メソッドで実装します。

```
// 基本タイトルと内容を入れて投稿画面を実行します。  
int menuId = 4; // 0 ならメニューを選択しません。  
String text = "基本タイトルと内容が入って投稿画面を実行します。";  
Glink.startWrite(MainActivity.this, "subject", text);
```

Glink.startImageWrite()メソッドや Glink.startVideoWrite()メソッドを使って画像や動画を添付したまま掲示板の投稿画面を実行することができます。

## App Scheme の処理

App Scheme で移動するバナー画像がある場合、下記のように onClickAppSchemeBannerListener を設定するとタッチイベントが発生した際に App Scheme を処理する機能を実装することができます。

```
// アプリスキームタッチリスナーの設定  
Glink.setOnClickAppSchemeBannerListener(new Glink.OnClickAppSchemeBannerListener() {  
    @Override public void onClickAppSchemeBanner(String appScheme) {  
        // Cafe 管理で設定した App Scheme の文字列を NAVER Cafe SDK から渡します。  
        // App Scheme を処理するコードを実装します。  
    }  
});
```

## ユーザーのゲーム ID とのマッピング

ユーザーのゲーム ID と NAVER ID をマッピングする機能は、下記のように Glink.setGameUserId()メソッドで実装します。

```
// ユーザのゲーム ID と NAVER ID をマッピングします。  
// プロフィール画面からマッピングされたゲーム ID を確認することができます。  
Glink.setGameUserId(this, "gameUserId", "ゲーム ID");
```

## リソース画像の変更

NAVER Cafe SDK ライブラリに含まれたリソース画像は、下記のように変更できます。

1. NAVER Cafe SDK ライブラリファイル(.aaa ファイル)を解凍します。
2. 解凍したフォルダの/**res/drawable-xhdpi** フォルダにある画像を希望の画像に変更します。
3. 解凍したフォルダを再度 NAVER Cafe SDK ライブラリファイル(.aaa ファイル)に圧縮します。
4. 再圧縮した NAVER Cafe SDK ライブラリを適用してプロジェクトをビルドします。

---

### 注意

画像を変更する際の画像サイズは、元画像のサイズと同じサイズでなければなりません。

---

## API リファレンス

### Glink

NAVER Cafe SDK のコントロールクラス。Glink クラスのメソッドを使って NAVER Cafe SDK を初期化したり、開始、中止する機能を実装します。Glink クラスのメソッドは、下記のとおりです。

- `init()`
- `isShowGlink()`
- `popBackStack()`
- `setGameUserId()`
- `startEvent()`
- `startImageWrite()`
- `startHome()`
- `startMenu()`
- `startNotice()`
- `startProfile()`
- `startVideoWrite()`
- `startWrite()`
- `stop()`

### init()

#### 説明

NAVER Cafe SDK の初期化を行います。

#### 構文

```
public static void init(String clientId, String clientSecret, int cafeId);
```

---

## パラメータ

パラメータ	タイプ	必須	説明
clientId	String	Y	クライアント ID。NAVER Login にアプリケーションを登録した際に発行される値です。
clientSecret	String	Y d	クライアントシークレット。NAVER Login にアプリケーションを登録した際に発行される値です。
cafeId	int	Y	Cafe ID。NAVER Cafe 管理ページ URL の <b>clubid</b> パラメータ値にあたります。

## 返却値

なし

## サンプルコード

```
// [1]NAVER Login 情報と Cafe ID を用いて NAVER Cafe SDK の初期化処理を行います。
Glink.init("abcd", "aaaa", 33);
```

## isShowGlink()

## 説明

NAVER Cafe SDK 画面の開閉有無を確認します。

## 構文

```
public static boolean isShowGlink(Activity activity);
```

## パラメータ

パラメータ	タイプ	必須	説明
activity	Activity	Y	メソッドを実行するアクティビティの Context オブジェクト

## 返却値

- true: NAVER Cafe SDK 画面が開いている。
- false: NAVER Cafe SDK 画面が開いていない。

## サンプルコード

```
// NAVER Cafe SDK 画面の開閉有無を確認します。
// 返却値が true なら開いている状態、false なら開いていない状態です。
Glink.isShowGlink(this);
```



## popBackStack()

### 説明

backStack に溜まっている画面を 1 つずつ閉じます。すべての画面を閉じると NAVER Cafe SDK を終了します。  
backStack に溜まる画面は投稿閲覧画面、投稿作成画面、検索画面です。

### 構文

```
public static void popBackStack(Activity activity);
```

### パラメータ

パラメータ	タイプ	必須	説明
activity	Activity	Y	メソッドを実行するアクティビティの Context オブジェクト

### 返却値

なし

### サンプルコード

```
// NAVER Cafe SDK 画面を 1 つずつ閉じます。  
Glink.popBackStack (activity);
```

## setGameUserId()

### 説明

ユーザーのゲーム ID と Cafe ID をマッピングします。

### 構文

```
public static void setGameUserId(Activity activity, String gameUserId, String fieldName);
```

### パラメータ

パラメータ	タイプ	必須	説明
activity	Activity	Y	メソッドを実行するアクティビティの Context オブジェクト
gameUserId	String	N	ユーザーのゲーム ID
fieldName	String	N	プロフィール画面に表示する ID(基本値: ユーザーのゲーム ID)

### 返却値

なし

### サンプルコード

```
// ユーザーのゲーム ID と NAVER ID をマッピングします。  
// プロフィール画面からマッピングされたゲーム ID を確認することができます。
```

```
Glink.setGameUserId(this, "gameUserId", "ゲーム ID");
```

## startEvent()

### 説明

イベントメニューが選択された状態で NAVER Cafe SDK を立ち上げます。

### 構文

```
public static void startEvent(Activity activity);
```

### パラメータ

パラメータ	タイプ	必須	説明
activity	Activity	Y	メソッドを実行するアクティビティの Context オブジェクト

### 返却値

なし

### サンプルコード

```
// イベントメニューが選択された状態で NAVER Cafe SDK を立ち上げます。
Glink.startEvent(activity);
```

## startImageWrite()

### 説明

画像を添付して掲示板の投稿作成画面を開きます。

### 構文

```
public static void startImageWrite(Activity activity, int menuId, String subject, String text, String imagery);
```

### パラメータ

パラメータ	タイプ	必須	説明
activity	Activity	Y	メソッドを実行するアクティビティの Context オブジェクト
menuId	int	N	掲示板の ID(基本値: -1)。掲示板の ID は、NAVER Cafe 掲示板 URL の <b>menuid</b> パラメータ値にあたります。
subject	String	N	投稿のタイトル
text	String	N	投稿の内容
imageUri	String	N	画像ファイルのパス(URI 形式)

**返却値**

なし

**サンプルコード**

```
// 基本タイトルと内容が入力されていて、画像が添付されている状態で投稿作成画面を立ち上げます。画像パスは、URI 形式で入力してください。
int menuId = 4; // 0 ならメニューを選択しません。
String text = "基本タイトルと内容が入力されていて、画像が添付されている状態で投稿作成画面を立ち上げます。¥n 画像パスは、URI 形式で入力してください。";
String path = "your image uri";
Glink.startImageWrite(MainActivity.this, menuId, "subject", text, path);
```

**startHome()****説明**

トップ画面で NAVER Cafe SDK を立ち上げます。

**構文**

```
public static void startHome(Activity activity);
```

**パラメータ**

パラメータ	タイプ	必須	説明
activity	Activity	Y	メソッドを実行するアクティビティの Context オブジェクト

**返却値**

なし

**サンプルコード**

```
// トップ画面で NAVER Cafe SDK を立ち上げます。
Glink.startHome(activity);
```

**startMenu()****説明**

掲示板が選択された状態で NAVER Cafe SDK 画面を開きます。

**構文**

```
public static void startMenu(Activity activity);
```

**パラメータ**

パラメータ	タイプ	必須	説明
activity	Activity	Y	メソッドを実行するアクティビティの Context オブジェクト

**返却値**

なし

**サンプルコード**

```
// 掲示板が選択された状態で NAVER Cafe SDK 画面を開きます。  
Glink.startMenu(activity);
```

**startNotice()****説明**

お知らせが選択された状態で NAVER Cafe SDK 画面を開きます。

**構文**

```
public static void startNotice(Activity activity);
```

**パラメータ**

パラメータ	タイプ	必須	説明
activity	Activity	Y	メソッドを実行するアクティビティの Context オブジェクト

**返却値**

なし

**サンプルコード**

```
// お知らせが選択された状態で NAVER Cafe SDK 画面を開きます。  
Glink.startNotice(activity);
```

**startProfile()****説明**

プロフィールを選択した状態で NAVER Cafe SDK 画面を開きます。

**構文**

```
public static void startProfile(Activity activity);
```

**パラメータ**

パラメータ	タイプ	必須	説明
activity	Activity	Y	メソッドを実行するアクティビティの Context オブジェクト

**返却値**

なし

## サンプルコード

```
// プロフィールが選択された状態で NAVER Cafe SDK 画面を開きます。
Glink.startProfile(activity);
```

## startVideoWrite()

## 説明

動画を添付して掲示板の投稿作成画面を開きます。

## 構文

```
public static void startVideoWrite(Activity activity, int menuId, String subject, String text, String videoUri);
```

## パラメータ

パラメータ	タイプ	必須	説明
activity	Activity	Y	メソッドを実行するアクティビティの Context オブジェクト
menuId	int	N	掲示板の ID(基本値: -1)。掲示板の ID は、NAVER Cafe 掲示板 URL の <b>menuid</b> パラメータ値にあたります。
subject	String	N	投稿のタイトル
text	String	N	投稿の内容
videoUri	String	N	動画ファイルのパス(URI 形式)

## 返却値

なし

## サンプルコード

```
// 基本タイトルと内容が入力されていて、動画が添付されている状態で投稿画面を立ち上げます。動画ファイルのパスは、URI 形式で入力してください。
int menuId = 4; // 0 ならメニューを選択しません。
String text = "基本タイトルと内容が入力されていて、動画が添付されている状態で投稿作成画面を起動します。¥n 動画ファイルのパスは、URI 形式で入力してください。";
String path = "your video uri";
Glink.startVideoWrite(MainActivity.this, menuId, "subject", text, path);
```

## startWrite()

## 説明

掲示板の投稿作成画面を開きます。

## 構文

```
public static void startWrite(Activity activity, int menuId, String subject, String text);
```

**パラメータ**

パラメータ	タイプ	必須	説明
activity	Activity	Y	メソッドを実行するアクティビティの Context オブジェクト
menuId	int	N	掲示板の ID(基本値: -1)。掲示板の ID は、NAVER Cafe 掲示板 URL の <b>menuid</b> パラメータ値にあたります。
subject	String	N	投稿のタイトル
text	String	N	投稿の内容

**返却値**

なし

**サンプルコード**

```
// 基本タイトル、内容を入れて投稿作成画面を立ち上げます。
int menuId = 4; // 0 なら、メニューを選択しません。
String text = "基本タイトル、内容を入れて投稿作成画面を立ち上げます。";
Glink.startWrite(MainActivity.this, "subject", text);
```

**stop()****説明**

NAVER Cafe SDK 画面を即時に閉じ、NAVER Cafe SDK を終了します。

**構文**

```
public static void stop(final Activity activity);
```

**パラメータ**

パラメータ	タイプ	必須	説明
activity	Activity	Y	メソッドを実行するアクティビティの Context オブジェクト

**返却値**

なし

**サンプルコード**

```
// stop() メソッドで終了します。
Glink.stop(activity);
```

# iOS 向け

## 開発環境

### ソフトウェア要件

#### 開発ツール

- IDE: Xcode 6.0 以上

#### 注意

ライブラリに ARC(automatic reference counting)が適用されています。

#### ライブラリ

NAVER Cafe SDK を組み込むには下記のライブラリをプロジェクトに追加してビルドします。

表 2 iOS 向け NAVER Cafe SDK ライブラリ

ライブラリ	ダウンロード URL
NAVER Login	<ul style="list-style-type: none"><li>• NAVER Cafe SDK ライブラリファイルに含まれています(4.0.6 バージョン)。</li><li>• ダウンロード URL: <a href="https://static.nid.naver.com/images/web/devcenter/3rdparty_login_library_ios_4.0.6.zip">https://static.nid.naver.com/images/web/devcenter/3rdparty_login_library_ios_4.0.6.zip</a></li></ul>
AFNetworking 1.0 以上	<ul style="list-style-type: none"><li>• NAVER Cafe SDK ライブラリファイルに含まれています(2.6.1 バージョン)。</li><li>• ダウンロード URL: <a href="https://github.com/AFNetworking/AFNetworking">https://github.com/AFNetworking/AFNetworking</a></li></ul>

### ライブラリの構成

iOS 向け NAVER Cafe SDK ライブラリは、下記のように構成されています。

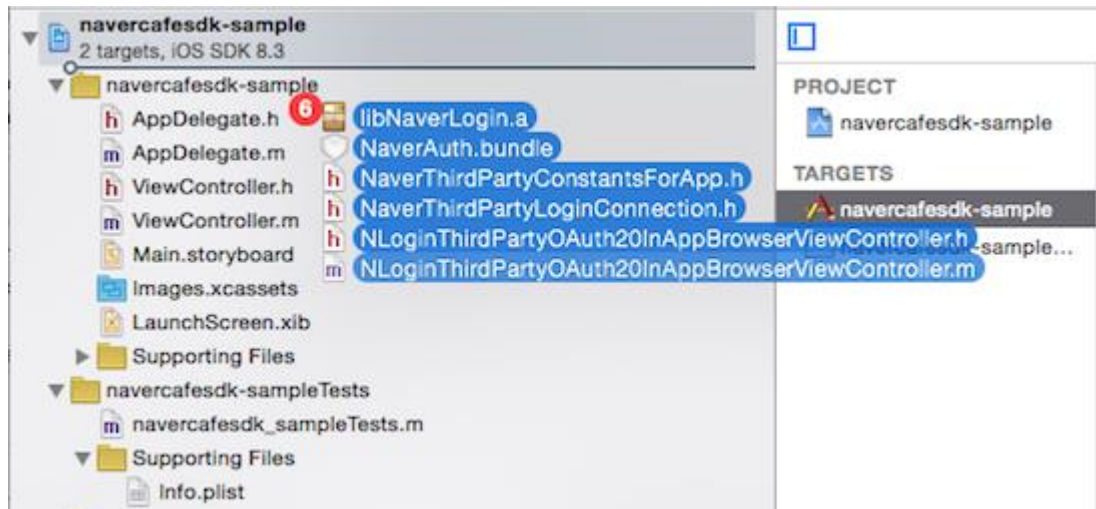
- lib: NAVER Cafe SDK ライブラリフォルダ
  - NAVER CafeSDK.bundle
  - NAVER CafeSDK.framework
- sample: NAVER Cafe SDK ライブラリを使用するサンプルプロジェクトと必須ライブラリフォルダ
  - external-lib: NAVER Login ライブラリと AFNetworking ライブラリフォルダ
  - NAVER Cafesdk-sample-ios: NAVER Cafe SDK サンプルプロジェクトフォルダ

## 開発環境セットアップ

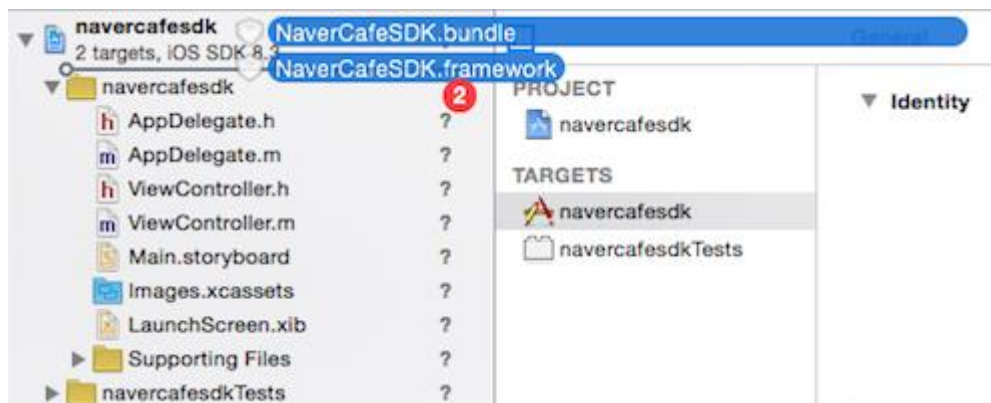
### Xcode の設定

NAVER Cafe SDK を組み込むには Xcode にて下記のように開発環境をセットアップします。

1. NAVER Login ライブラリを解凍します。
2. Xcode で NAVER Login ライブラリをプロジェクトに追加します。




3. Xcode で **AFNetworking** ライブラリをプロジェクトに追加します。
4. NAVER Cafe SDK ライブラリを解凍します。
5. Xcode で **NAVER CafeSDK.framework** と **NAVER CafeSDK.bundle** をプロジェクトに追加します。



6. NAVER Login でアプリケーション登録時に入力した URL Scheme をプロジェクトに登録します。



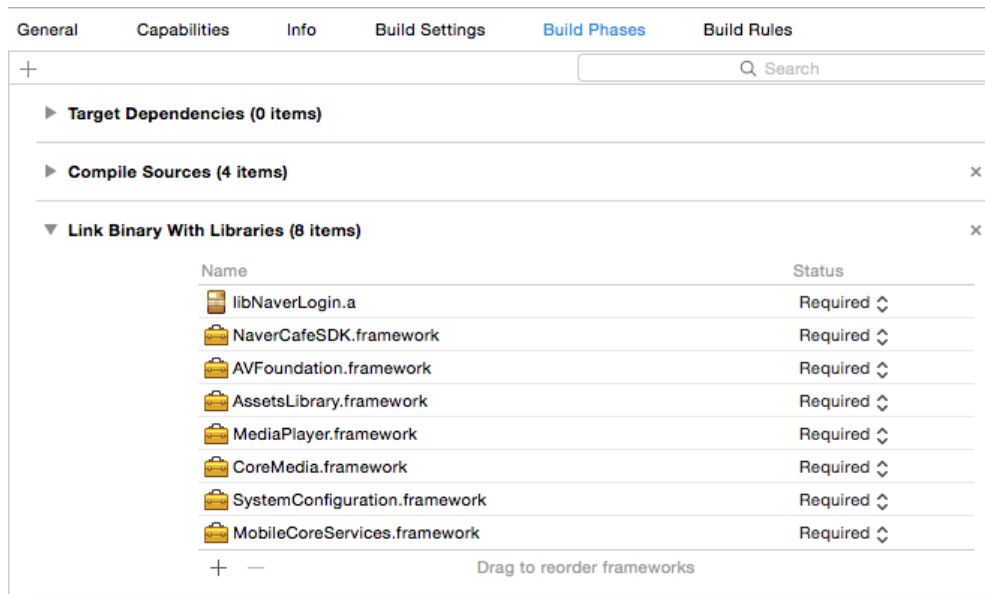
General	Capabilities	Info	Build Settings	Build Phases	Build Rules
▶ Custom iOS Target Properties					
▶ Document Types (0)					
▶ Exported UTIs (0)					
▶ Imported UTIs (0)					
▼ URL Types (1)					
<div> <div> <div>Untitled</div> <div>  </div> </div> <div> <div>Identifier</div> <input type="text" value="None"/> </div> <div> <div>URL Schemes</div> <input type="text" value="gLinkSample"/> </div> </div> <div> <div>Icon</div> <input type="text" value="None"/> </div> <div> <div>Role</div> <input type="text" value="Editor"/> </div>					

7. スタティックライブラリを使えるように **Build Settings** の **Other Linker Flags** に **-ObjC -all\_load** オプションを設定します。

General	Capabilities	Info	Build Settings	Build Phases	Build Rules
<div> <div>Basic</div> <div>All</div> <div>Combined</div> <div>Levels</div> <div>+</div> </div> <div> <input type="text" value="other linker flags"/> </div>					
▼ Linking					
Setting					
<div> <div>Link With Standard Libraries</div> <div>Yes</div> </div>					
<div> <div>Other Linker Flags</div> <div>-ObjC -all_load</div> </div>					

8. **Build Phases** の **Link Binary With Libraries** に下記のライブラリを追加します。

- MobileCoreServices.framework
- SystemConfiguration.framework
- MediaPlayer.framework
- AVFoundation.framework
- CoreMedia.framework



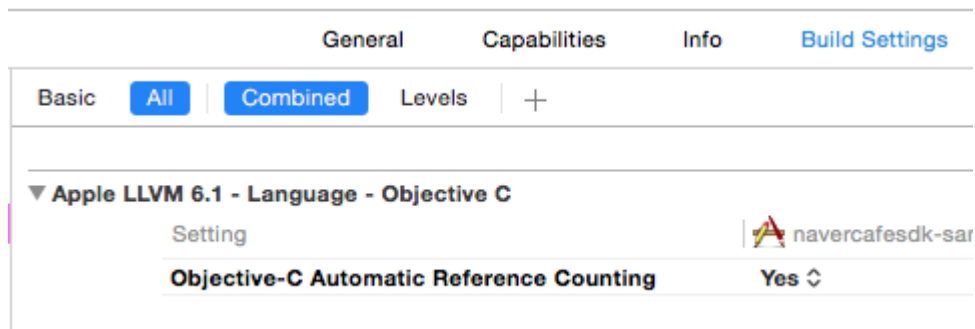
Cocos2d-x エンジンを使用する場合は、下記のライブラリを追加します。

- MobileCoreServices.framework
- SystemConfiguration.framework
- MediaPlayer.framework
- AVFoundation.framework
- CoreMedia.framework
- GameController.framework
- AssetsLibrary.framework
- Security.framework

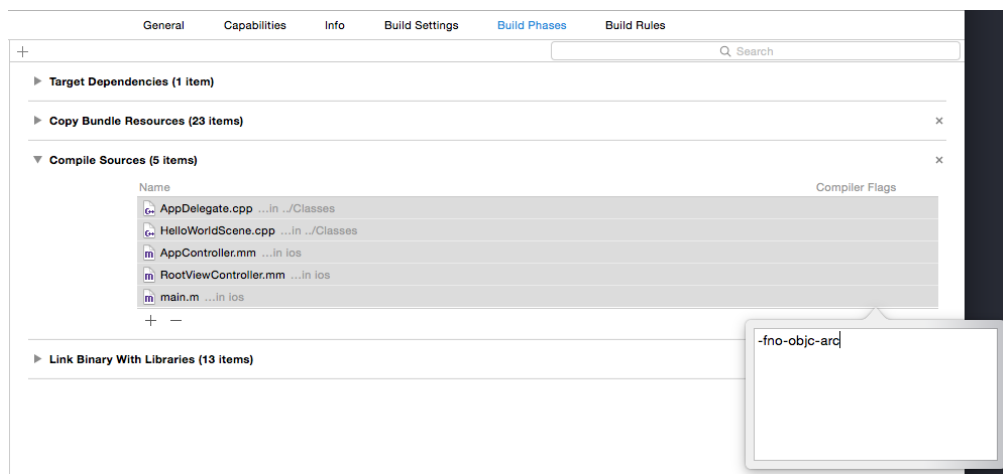
### MRC 環境を ARC 環境に切り替える

Cocos2d-x エンジンを使用するプロジェクトが MRC(manual reference counting)環境であれば、下記の方法で ARC(automatic reference counting)環境に切り替えます。

1. **Build Settings** の **Objective-C Automatic Reference Counting** を **YES** に設定します。



2. **Build Phases** の **Compile Sources** にてコンパイルするファイルの **Compile Flags** を **-fno-objc-arc** に設定します。



## NAVER Cafe SDK の組み込み手順

### NAVER Cafe SDK の初期化

下記のように Naver Cafe SDK の初期化処理を行います。クライアント ID とクライアントシークレットは、NAVER Login にアプリケーションを登録した際に発行された値です。

```
//ViewController
#import <NAVER CafeSDK/NCSDKManager.h>

- (void)viewDidLoad {
// Naver Login 登録情報と Cafe ID で Naver Cafe SDK の初期化処理を行います。
    [[NCSDKManager sharedInstance] setNaverLoginConsumerKey:@"クライアント ID"
                                naverLoginConsumerSecret:@"クライアントシークレット"
                                cafeId:00000000];
}
```

### NAVER Login の設定

下記のようにアプリデリゲートを設定し、NAVER Login 処理が完了したら Naver Cafe SDK にログイン情報を設定します。

```
//AppDelegate
#import <NAVER CafeSDK/NCSDKLoginManager.h>

- (BOOL)application:(UIApplication *)application openURL:(NSURL *)url
sourceApplication:(NSString *)sourceApplication annotation:(id)annotation {
//ログイン情報を Naver Login オブジェクトに設定します。
    return [[NCSDKLoginManager sharedInstance] finishNaverLoginWithURL:url];
}
```

### NAVER Cafe SDK 画面を開く

NAVER Cafe SDK 画面を立ち上げるには Naver Cafe SDK ビューコントローラを設定し、presentMainViewController メソッドを実行します。

```
#import <NAVER CafeSDK/NCSDKManager.h>

//NAVER Cafe トップを実行します。
[[NCSDKManager sharedInstance] setParentViewController:self];
[[NCSDKManager sharedInstance] presentMainViewController];
```

presentMainViewController メソッドの代わりに presentMainViewControllerWithTabIndex メソッドを使えば NAVER Cafe SDK 画面の特定のメニューを選択して NAVER Cafe SDK 画面を立ち上げることができます。なお、presentMainViewControllerWithArticleId メソッドを使用すると、特定掲示板の投稿を開いた状態で NAVER Cafe SDK 画面を立ち上げることができます。

## 投稿作成

Cafe 掲示板の投稿作成画面を開くには、下記のように presentArticlePostViewControllerWithMenuId メソッドを使用します。

```
//投稿作成画面を実行します。
[[NCSDKManager sharedInstance] setParentViewController:self];
[[NCSDKManager sharedInstance] presentArticlePostViewControllerWithMenuId:0 subject:@"投稿のタイトル" content:@"投稿の内容" filePath:@"document"];
```

## API リファレンス

### NCSDKManager

NAVER Cafe SDK のコントロールクラス。NCSDKManager クラスのメソッドを使って NAVER Cafe SDK を初期化したり、開始、中止する機能を実装します。NAVER Cafe SDK を実行する最上位のビューコントローラである parentViewController クラスを持っています。

NCSDKManager クラスのメソッドは、下記のとおりです。

- dismissViewController
- dismissTopViewController
- sharedInstance
- NAVER CafeRootViewController
- resetSharedInstance
- presentArticlePostViewControllerWithMenuId
- presentArticlePostViewControllerWithType
- presentMainViewController
- presentMainViewControllerWithArticleId
- presentMainViewControllerWithTabIndex
- presentViewController
- setGameUserId
- setNaverLoginClientId

## @property(nonatomic, weak) id parentViewController

### 説明

NAVER Cafe SDK を実行するビューコントローラ。NAVER Cafe SDK を実行する親クラスです。

### 構文

```
@property (nonatomic, weak) id parentViewController;
```

### パラメータ

なし

### 返却値

UIViewController オブジェクト

### サンプルコード

```
[[NCSDKManager sharedInstance] setParentViewController:self]
```

## (void)dismissViewController

### 説明

NAVER Cafe SDK を実行するビューコントローラ上で実行された別のビューコントローラを消去します。

### 構文

```
- (void)dismissViewController:(id)viewController;
```

### パラメータ

パラメータ	タイプ	必須	説明
viewController	id	Y	NAVER Cafe SDK ビューコントローラ上で実行されたビューコントローラ

### 返却値

なし

### サンプルコード

```
[[NCSDKManager sharedInstance] dismissViewController:self];
```

## (void)dismissTopViewController

### 説明

NAVER Cafe SDK を実行するビューコントローラ上で実行された別のビューコントローラのうち、最上位にあるビューコントローラを消去します。

#### 構文

```
- (void)dismissTopViewController;
```

#### パラメータ

なし

#### 返却値

なし

#### サンプルコード

```
[NCSDKManager sharedInstance] dismissTopViewController];
```

### (NCSDKManager \*)getSharedInstance

#### 説明

NAVER Cafe SDK インスタンス(シングルトン・インスタンス)を獲得します。

#### 構文

```
+ (NCSDKManager *)getSharedInstance;
```

#### パラメータ

なし

#### 返却値

NCSDKManager オブジェクト

#### サンプルコード

```
[NCSDKManager sharedInstance]
```

### (id)NAVER CafeRootViewController

#### 説明

NAVER Cafe SDK の最上位ビューコントローラオブジェクトを獲得します。

#### 構文

```
- (id)NAVER CafeRootViewController;
```

#### パラメータ

なし

#### 返却値

UIViewController オブジェクト

### サンプルコード

```
[[NCSDKManager sharedInstance] NAVER CafeRootViewController]
```

## (void)resetSharedInstance

### 説明

NCSDKManager オブジェクトを削除します。

### 構文

```
+ (void)resetSharedInstance;
```

### パラメータ

なし

### 返却値

なし

### サンプルコード

```
[NCSDKManager resetSharedInstance]
```

## (void)presentArticlePostViewControllerWithMenuId

### 説明

掲示板の投稿作成画面を開きます。

### 構文

```
- (void)presentArticlePostViewControllerWithMenuId: (NSInteger)menuId
                                     subject:(NSString *)subject
                                     content:(NSString *)content;
```

### パラメータ

パラメータ	タイプ	必須	説明
menuId	NSInteger	Y	掲示板の ID(基本値: 0)。掲示板の ID は、NAVER Cafe 掲示板 URL の <b>menuid</b> パラメータ値にあたります。
subject	NSString	N	投稿のタイトル
content	NSString	N	投稿の内容

### 返却値

なし

### サンプルコード

```
[[NCSDKManager sharedInstance] presentArticlePostViewControllerWithMenuId:1
                                subject:@"タイトル"
                                content:@"内容"];
```

## (void)presentArticlePostViewControllerWithType

### 説明

ファイルを添付して掲示板の投稿作成画面を開きます。

### 構文

```
- (void)presentArticlePostViewControllerWithType: (GLArticlePostType) type
                                menuId: (NSInteger) menuId
                                subject: (NSString *) subject
                                content: (NSString *) content
                                filePath: (NSString *) filePath;
```

### パラメータ

パラメータ	タイプ	必須	説明
type	GLArticlePostType	Y	添付ファイルのタイプ <ul style="list-style-type: none"> <li>1: 画像ファイル</li> <li>2: 動画ファイル</li> </ul>
menuId	NSInteger	Y	掲示板の ID(基本値: 0)。掲示板の ID は、NAVER Cafe 掲示板 URL の <b>menuid</b> パラメータ値にあたります。
subject	NSString	N	投稿のタイトル
content	NSString	N	投稿の内容
filePath	NSString	Y	添付ファイルのパス

### 返却値

なし

### サンプルコード

```
[[NCSDKManager sharedInstance] presentArticlePostViewControllerWithType:kGLArticlePostTypeVideo
                                menuId:1
                                subject:@"タイトル"
                                content:@"内容"
                                filePath:@"private/var/mobile/Applications/0D1657F9-EACF-4D64-BC8A-4E01EB4FF247/tmp/trim.2CC623C7-78C3-4597-BA75-9BA12BFEF333.MOV"];
```



## (void)presentMainViewController

### 説明

NAVER Cafe SDK 画面を開きます。

### 構文

```
- (void)presentMainViewController;
```

### パラメータ

なし

### 返却値

なし

### サンプルコード

```
[[NCSDKManager sharedInstance] presentMainViewController];
```

## (void)presentMainViewControllerWithArticleId

### 説明

掲示板の投稿記事が開いている状態で NAVER Cafe SDK 画面を開きます。

### 構文

```
- (void)presentMainViewControllerWithArticleId:(NSUInteger)articleId;
```

### パラメータ

パラメータ	タイプ	必須	説明
articleId	NSUInteger	Y	投稿 ID(基本値: 0)

### 返却値

なし

### サンプルコード

```
[[NCSDKManager sharedInstance] presentMainViewControllerWithArticleId:10];
```

## (void)presentMainViewControllerWithTabIndex

### 説明

指定したメニューが選択された状態で NAVER Cafe SDK 画面を開きます。

### 構文

```
- (void)presentMainViewControllerWithTabIndex:(NSUInteger)tabIndex;
```

---

**パラメータ**

パラメータ	タイプ	必須	説明
tabIndex	NSUInteger	Y	選択するメニューのインデックス値(基本値: 0) <ul style="list-style-type: none"> <li>0: トップ</li> <li>1: お知らせ</li> <li>2: イベント</li> <li>3: 統合掲示板</li> <li>4: プロフィール</li> </ul>

**返却値**

なし

**サンプルコード**

```
[[NCSDKManager sharedInstance] presentMainViewControllerWithTabIndex:1];
```

**(void)presentViewController****説明**

NAVER Cafe SDK の最上位ビューコンローラから他のビューコントローラを実行します。

**構文**

```
- (void)presentViewController:(id)viewController;
```

**パラメータ**

パラメータ	タイプ	必須	説明
viewController	id	Y	ビューコントローラオブジェクトの ID

**返却値**

なし

**サンプルコード**

```
[[NCSDKManager sharedInstance] presentViewController:self];
```

**(void)setGameUserId****説明**

ユーザーのゲーム ID と NAVER ID を連携してプロフィール画面に ID を表示します。

**構文**

```
- (void)setGameUserId:(NSString *)gameUserId fieldName:(NSString *)fieldName;
```

**パラメータ**

パラメータ	タイプ	必須	説明
gameUserId	NSString	Y	ユーザーのゲーム ID
fieldName	NSString	N	プロフィール画面に表示する ID(基本値: ゲーム ID)

**返却値**

なし

**サンプルコード**

```
[[NCSDKManager sharedInstance] setGameUserId:@"abc3251235" fieldName:@"ゲーム ID"];
```

**(void)setNaverLoginClientId****説明**

NAVER Login オブジェクトを設定します。

**構文**

```
- (void)setNaverLoginClientId:(NSString *)naverLoginClientId
    naverLoginClientSecret:(NSString *)naverLoginClientSecret
    cafeId:(NSInteger)cafeId;
```

**パラメータ**

なし

**返却値**

なし

**サンプルコード**

```
[[NCSDKManager sharedInstance] setNaverLoginConsumerKey:@"Consumer ID"
    naverLoginConsumerSecret:@"Secret ID"
    cafeId:00000000];
```

**NCSDKLoginManager**

NAVER Cafe SDK で NAVER Login 機能をコントロールするクラス。

NCSDKLoginManager クラスのメソッドは下記のとおりです。

- accessToken
- accessTokenExpireTime
- finishNaverLoginWithURL
- sharedInstance

- isLoginWithFinish
- isValidAccessTokenExpireTimeNow
- loginWithFinish
- logout
- refreshAccessToken
- refreshAccessTokenWithFinish
- requestDeleteToken
- setIsInAppOAuthEnable
- setIsNaverAppOAuthEnable

## @property (nonatomic, weak) UIViewController \*rootViewController

### 説明

ログインのウェブビューを実行するビューコントローラ。

### 構文

```
@property (nonatomic, weak) UIViewController *rootViewController;
```

### パラメータ

なし

### 返却値

UIViewController オブジェクト

### サンプルコード

```
[[NCSDKLoginManager sharedInstance] setRootViewController:self];
```

## (NSString \*)accessToken

### 説明

NAVER Login 認証後に NAVER サーバから受け取ったアクセストークン(access token)が返されます。

### 構文

```
- (NSString *)accessToken;
```

### パラメータ

なし

### 返却値

アクセストークン

### サンプルコード

```
[[NCSDKLoginManager sharedInstance] accessToken];
```

## (NSString \*)accessTokenExpireTime

### 説明

アクセストークン(access token)の満了期限を確認します。

### 構文

```
- (NSString *)accessTokenExpireTime;
```

### パラメータ

なし

### 返却値

アクセストークン満了期限

### サンプルコード

```
[[NCSDKLoginManager sharedInstance] accessTokenExpireTime];
```

## (BOOL)finishNaverLoginWithURL

### 説明

NAVER ID でのログイン後にアプリデリゲートを実行します。

### 構文

```
- (BOOL)finishNaverLoginWithURL:(NSURL *)url;
```

### パラメータ

パラメータ	タイプ	必須	説明
url	NSURL	Y	NAVER ID でのログイン後にアプリデリゲートでコールバックされる URL Scheme

### 返却値

- true: ログインに成功
- false: ログインに失敗

### サンプルコード

```
- (BOOL)application:(UIApplication *)application openURL:(NSURL *)url  
sourceApplication:(NSString *)sourceApplication annotation:(id)annotation {  
    return [[NCSDKLoginManager sharedInstance] finishNaverLoginWithURL:url];  
}
```

## (NCSDKLoginManager \*)getSharedInstance

### 説明

NAVER Login インスタンス(シングルトン・インスタンス)を獲得します。

### 構文

```
+ (NCSDKLoginManager *)getSharedInstance;
```

### パラメータ

なし

### 返却値

NCSDKLoginManager オブジェクト

### サンプルコード

```
[NCSDKLoginManager getSharedInstance]
```

## (void)isLoginWithFinish

### 説明

NAVER Login 認証後に NAVER サーバから受け取ったアクセストークン(access token)の有効有無を確認します。

### 構文

```
- (void)isLoginWithFinish:(void (^)(BOOL successAccessToken))finish;
```

### パラメータ

なし

### 返却値

- true: 有効なアクセストークンです。
- false: アクセストークンの期限が満了になり、無効になったアクセストークンです。

### サンプルコード

```
[[NCSDKLoginManager getSharedInstance] isLoginWithFinish:^(BOOL successAccessToken) {  
    if (successAccessToken) {  
          
    }  
}];
```

## (BOOL)isValidAccessTokenExpireTimeNow

### 説明

アクセストークン(access token)があるか、有効期限は残っているかを確認します。ただし、サーバーで有効期限が満了した場合には確認できません。

---

**構文**

```
- (BOOL)isValidAccessTokenExpireTimeNow;
```

**パラメータ**

なし

**返却値**

- true: 有効なアクセストークンです。
- false: アクセストークンの期限が満了になり、無効になったアクセストークンです。

**サンプルコード**

```
[[NCSDKLoginManager sharedInstance] isValidAccessTokenExpireTimeNow];
```

**(void)loginWithFinish****説明**

NAVER Login 認証を開始し、アクセストークン(access token)が返されます。

**構文**

```
- (void)loginWithFinish:(void (^)(BOOL successAccessToken))finish;
```

**パラメータ**

なし

**返却値**

- true: 有効なアクセストークンです。
- false: アクセストークンの期限が満了になり、無効になったアクセストークンです。

**サンプルコード**

```
[[NCSDKLoginManager sharedInstance] loginWithFinish:^(BOOL successAccessToken) {  
    if (successAccessToken) {  
        }  
    }  
}];
```

**(void)logout****説明**

クライアントに保存されているアクセストークン(access token)と更新トークン(refresh token)を削除してログアウトします。

**構文**

```
- (void)logout;//delete local accesstoken
```

**パラメータ**

なし

**返却値**

なし

**サンプルコード**

```
[[NCSDKLoginManager sharedInstance] logout];
```

**(void)refreshAccessToken****説明**

NAVER Login 認証後に NAVER サーバーから受け取った更新トークン(refresh token)でアクセストークン(access token)を更新します。

**構文**

```
- (void)refreshAccessToken;
```

**パラメータ**

なし

**返却値**

なし

**サンプルコード**

```
[[NCSDKLoginManager sharedInstance]refreshAccessToken];
```

**(void)refreshAccessTokenWithFinish****説明**

NAVER Login 認証後に NAVER サーバーから更新トークン(refresh token)が返されます。

**構文**

```
- (void)refreshAccessTokenWithFinish:(void (^)(BOOL successAccessToken))finish;
```

**パラメータ**

なし

**返却値**

- true: アクセストークンの更新に成功
- false: アクセストークンの更新に失敗



**サンプルコード**

```
[[NCSDKLoginManager sharedInstance] refreshAccessTokenWithFinish:^(BOOL successAccessToken)
{
    if (successAccessToken) {
    }
}
];
```

**(void)requestDeleteToken****説明**

クライアントとサーバーに保存されているアクセストークン(access token)と更新トークン(referesh token)を削除します。

**構文**

```
- (void)requestDeleteToken;//delete server authorization
```

**パラメータ**

なし

**返却値**

なし

**サンプルコード**

```
[[NCSDKLoginManager sharedInstance] requestDeleteToken];
```

**(void)setIsInAppOAuthEnable****説明**

アプリ内ブラウザを実行してログイン手続きを行います。

**構文**

```
- (void)setIsInAppOAuthEnable:(BOOL)enable;
```

**パラメータ**

パラメータ	タイプ	必須	説明
enable	BOOL	Y	アプリ内ブラウザの実行有無

**返却値**

なし

**サンプルコード**

```
[[NCSDKLoginManager sharedInstance] setIsInAppOAuthEnable:YES];
```

## (void)setIsNaverAppOauthEnable

### 説明

NAVER アプリを実行してログイン手続きを行います。

### 構文

```
- (void)setIsNaverAppOauthEnable:(BOOL)enable;
```

### パラメータ

パラメータ	タイプ	必須	説明
enable	BOOL	Y	NAVER アプリの実行有無

### 返却値

なし

### サンプルコード

```
[[NCSDKLoginManager sharedInstance] setIsNaverAppOauthEnable:YES];
```

# Unity 向け

## 開発環境

### ソフトウェア要件

#### 開発ツール

- ゲームエンジン: Unity 4 以上

### ライブラリの構成

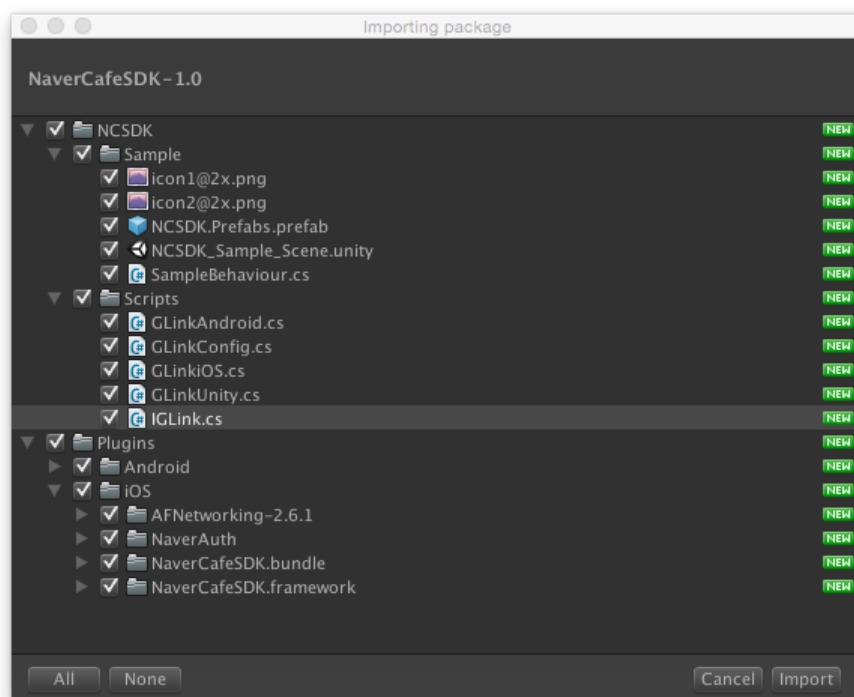
- NAVER CafeSDK-1.0.unitypackage: Unity 向け NAVER Cafe SDK ライブラリ

### 開発環境セットアップ

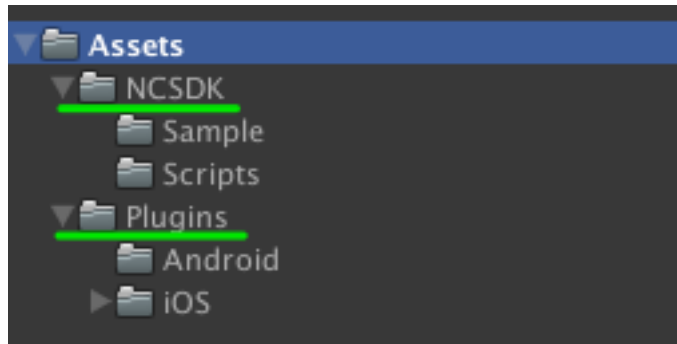
NAVER Cafe SDK を組み込むには下記のように開発環境をセットアップします。

#### Unity の設定

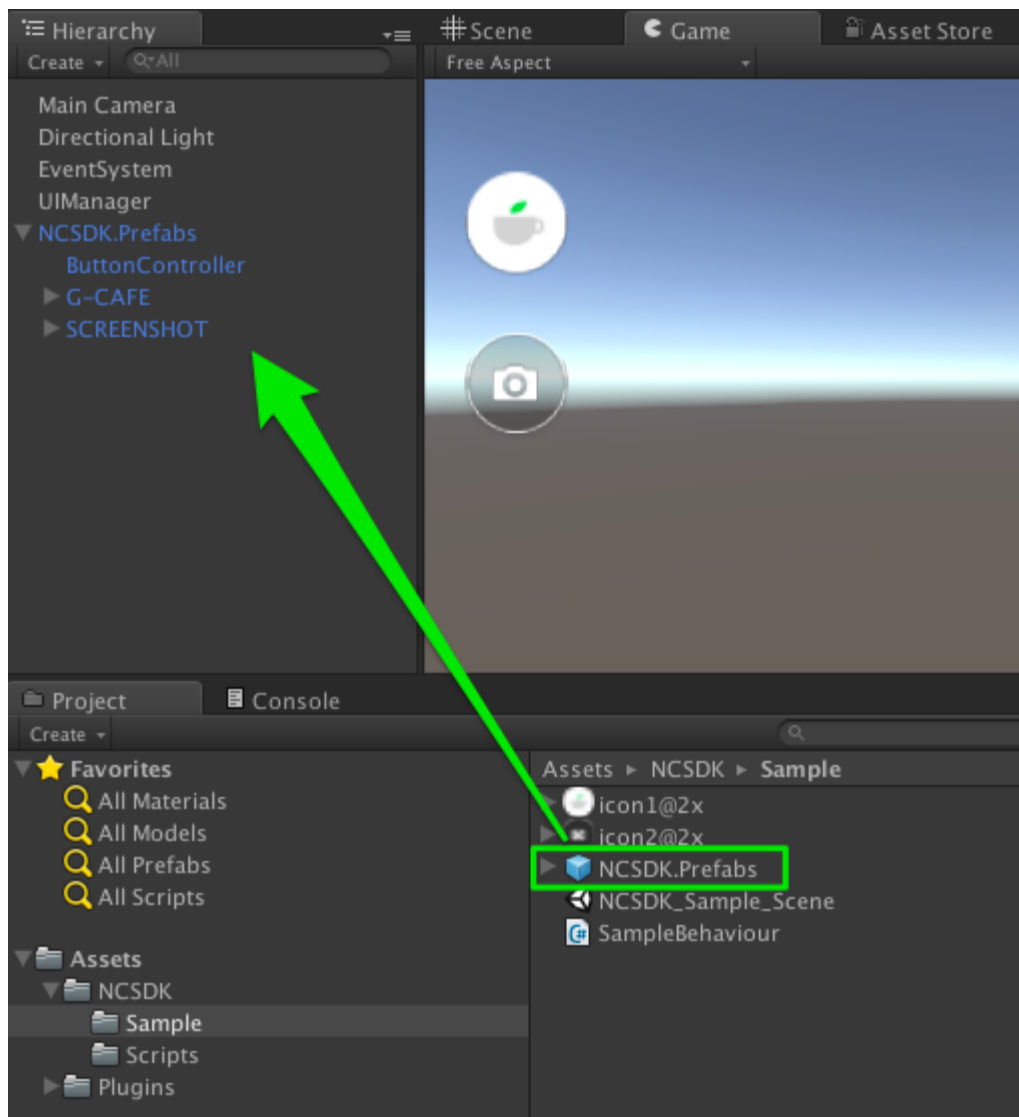
- Unity メニューから、**Asset > Import package** をクリックし **Custom Package** ダイアログボックスからダウンロードした **NAVER CafeSDK-1.0.unityPackage** ファイルを選択します。



- Custom Package** ダイアログボックスから **Import** をクリックすると **NCSDK** フォルダと **Plugins** フォルダが作成されます。



3. **NCSDK** フォルダの **Sample** フォルダにある **NCSDK.Prefabs** を希望のシーンヘドラッグします。NAVER Cafe SDK で基本提供するボタンが作成されます。



4. **NCSDK** フォルダの **Scripts** フォルダにある **GLinkConfig** ファイルに Cafe ID とクライアント ID、クライアントシークレットを入力します。

```
//NCSDK/Scripts/GLinkConfig
static class GLinkConfig
{
    public const string NaverLoginConsumerKey =
```

```
        "Consumer ID";

        public const string NaverLoginConsumerSecret =
            "Secret ID";

        public const int CafeId =
            00000000;
    }
```

## Android の設定

Unity 5 エンジンを使用する場合は、下記のように Naver Cafe SDK ライブラリを追加します。

- Naver Cafe SDK ライブラリファイル(cafeSdk-x.x.x.aar ファイル)と必須ライブラリファイルを **Assets/Plugins/Android** フォルダに追加します。

Unity 4 エンジンを使用する場合は、Eclipse で下記の追加設定が必要です。

1. Naver Cafe SDK ライブラリファイルのうち、cafeSdk-x.x.x.zip ファイルを解凍します。
2. 解凍した Naver Cafe SDK ライブラリフォルダ内の **libs** フォルダにビルドに必要な必須ライブラリファイルを追加します。
3. Eclipse から解凍したフォルダをプロジェクトで読み込みます。
4. プロジェクト設定ダイアログボックスから **Android** をクリックして **Is Library** を選択します。**AndroidManifest.xml** ファイルにて、下記のように Naver Cafe SDK で使用するアクティビティを追加します。


```
<activity
    android:name="com.nhn.android.naverlogin.ui.OAuthLoginActivity"
    android:screenOrientation="sensorLandscape"
    android:theme="@android:style/Theme.Translucent.NoTitleBar"/>
<activity
    android:name="com.nhn.android.naverlogin.ui.OAuthLoginInAppBrowserActivity"
    android:screenOrientation="sensorLandscape"
    android:label="OAuth2.0 In-app"/>
<activity
    android:name=".ui.VideoPlayActivity"
    android:screenOrientation="sensorLandscape"
    android:theme="@android:style/Theme.Translucent.NoTitleBar.Fullscreen"/>
```

5. プロジェクトをビルドします。
6. Eclipse でビルドしたファイルを Unity プロジェクトの **Assets/Plugins/Android** フォルダに追加します。
7. Unity プロジェクトをビルドします。

## iOS の設定

iOS アプリケーションを開発する際は、下記のように Xcode を設定します。

1. Naver Login でアプリケーション登録時に入力した URL Scheme を Xcode プロジェクトに登録します。

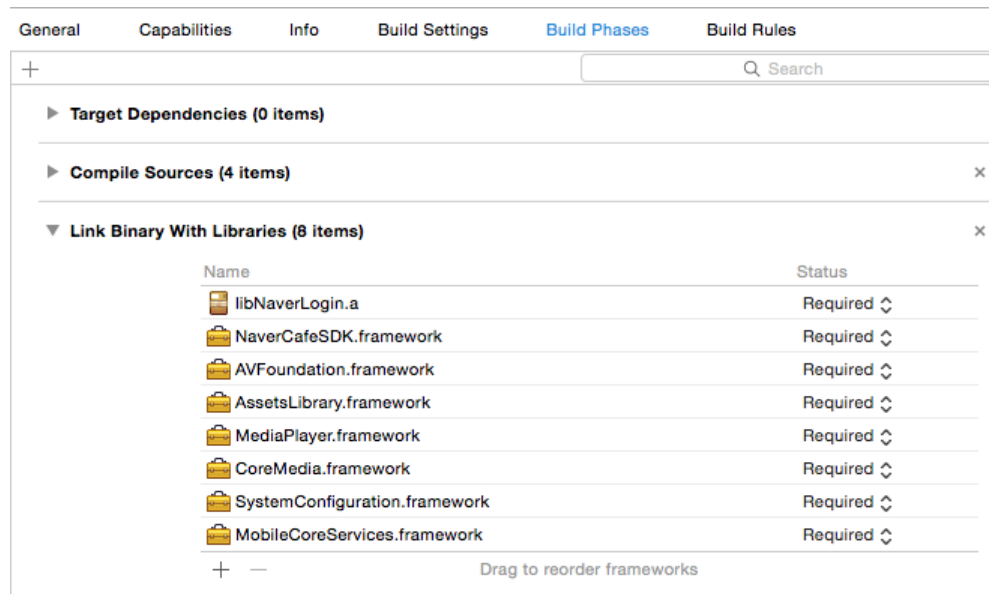
General	Capabilities	Info	Build Settings	Build Phases	Build Rules
▶ Custom iOS Target Properties					
▶ Document Types (0)					
▶ Exported UTIs (0)					
▶ Imported UTIs (0)					
▼ URL Types (1)					
<div> <div> <div>Untitled</div> <div>  </div> </div> <div> <div>Identifier</div> <div>None</div> </div> <div> <div>Icon</div> <div>None</div> </div> <div> <div>URL Schemes</div> <div>gLinkSample</div> </div> <div> <div>Role</div> <div>Editor</div> </div> </div>					

2. スタティックライブラリを使えるように **Build Settings** の **Other Linker Flags** に **-ObjC -all\_load** オプションを設定します。

General	Capabilities	Info	Build Settings	Build Phases	Build Rules						
<div> <div>Basic</div> <div>All</div> <div>Combined</div> <div>Levels</div> <div>+</div> <div> <input type="text" value="other linker flags"/> </div> </div>											
▼ Linking											
<table> <thead> <tr> <th>Setting</th> <th>navercafesdk-sample</th> </tr> </thead> <tbody> <tr> <td>Link With Standard Libraries</td> <td>Yes ⇅</td> </tr> <tr> <td>Other Linker Flags</td> <td>-ObjC -all_load</td> </tr> </tbody> </table>						Setting	navercafesdk-sample	Link With Standard Libraries	Yes ⇅	Other Linker Flags	-ObjC -all_load
Setting	navercafesdk-sample										
Link With Standard Libraries	Yes ⇅										
Other Linker Flags	-ObjC -all_load										

3. **Build Phases** の **Link Binary With Libraries** に下記のライブラリを追加します。

- MobileCoreServices.framework
- SystemConfiguration.framework
- MediaPlayer.framework
- AVFoundation.framework
- CoreMedia.framework
- Security.framework
- AssetsLibrary.framework



4. 下記のようにアプリデリゲートを設定し、NAVER Login が完了したらアプリデリゲートを呼び出します。

```
//AppDelegate
# import <NAVER CafeSDK/NCSDKLoginManager.h>

- (BOOL)application:(UIApplication *)application openURL:(NSURL *)url
sourceApplication:(NSString *)sourceApplication annotation:(id)annotation {
    return [[NCSDKLoginManager sharedInstance] finishNaverLoginWithURL:url];
}
```

## NAVER Cafe SDK の組み込み手順

### 実行

NAVER Cafe SDK 画面を立ち上げるには、下記のように `GlinkUnity.executeMain()` メソッドを呼び出します。

```
// Cafe トップ
GlinkUnity.executeMain ();
```

投稿作成画面は、下記のように `GlinkUnity.executeArticlePostWithImgae()` メソッドで実装します。

```
//スクリーンショット
GlinkUnity.executeArticlePostWithImgae(menuId, "投稿のタイトル", "投稿の内容", image path);
```

`GlinkUnity.executeArticlePostWithImage()` メソッドや `GlinkUnity.executeArticlePostWithVideo()` メソッドを使用すると画像や動画を添付した状態で掲示板の投稿作成画面を開くことができます。

### ユーザーのゲーム ID と NAVER ID とのマッピング

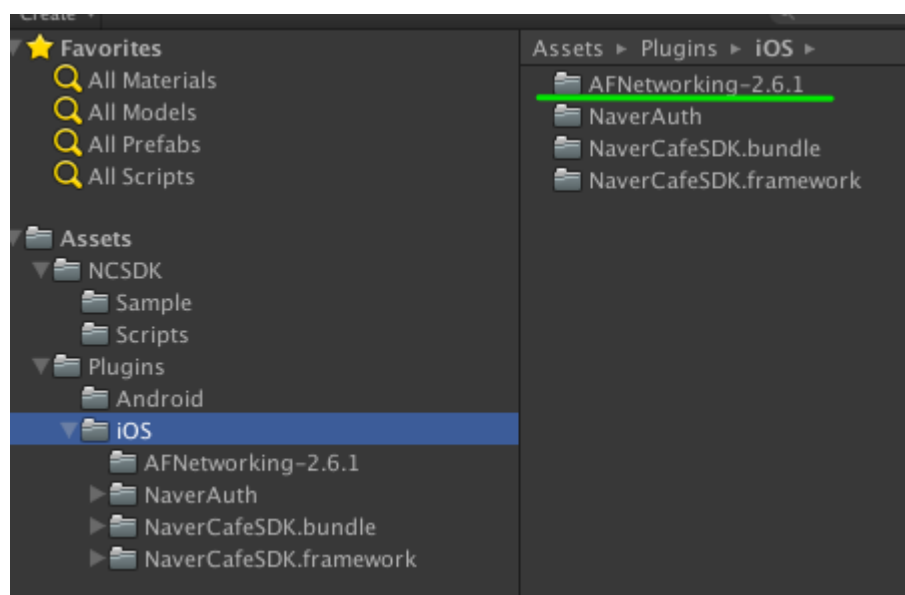
NAVER Cafe SDK は、ユーザーのゲーム ID と NAVER ID をマッピングする機能を提供します。PC の Cafe 管理者ページへアクセスすると、マッピングされたゲーム ID と NAVER ID のリストを確認することができます。

下記は、iOS アプリケーションでゲーム ID と NAVER ID をマッピングする機能を実装した例です。

```
/*
NAVER IDとゲームIDをマッピングさせるためにユーザーのゲームIDを連携します。
fieldName は、単純にプロフィールエリアに表示する文字列です(基本値: ゲームID)。
*/
- (void)setGameUserId:(NSString *)gameUserId fieldName:(NSString *)fieldName;
```

## AFNetworking の交替

**NAVER CafeSDK.unitypackage** には、ネットワーク通信のためのライブラリである AFNetworking 2.6.1 が基本的に含まれています。プロジェクトの状況によって AFNetworking ライブラリを削除したり、別のバージョンでビルドしても構いません。ただし、NAVER Cafe SDK は AFNetworking 1.x 以上に対応します。



## API リファレンス

### GLinkUnity

NAVER Cafe SDK のコントロールクラス。GLinkUnity クラスのメソッドを使って NAVER Cafe SDK を開始したり、投稿作成画面を開く機能を実装します。

GLinkUnity クラスのメソッドは、下記のとおりです。

- executeArticlePost()
- executeArticlePostWithImage()
- executeArticlePostWithVideo()
- executeMain()

### executeArticlePost()

#### 説明

掲示板の投稿作成画面を立ち上げます。

---



**構文**

```
public static void executeArticlePost(int menuId, string subject, string content) {
    sharedInstance().executeArticlePost (menuId, subject, content);
}
```

**パラメータ**

パラメータ	タイプ	必須	説明
menuId	int	Y	掲示板の ID(基本値: -1). 掲示板の ID は、NAVER Cafe 掲示板の URL の <b>menuid</b> パラメータ値にあたります。
subject	string	Y	投稿のタイトル
content	string	Y	投稿の内容

**返却値**

なし

**サンプルコード**

```
GlinkUnity.executeArticlePost(28290504, "投稿のタイトル", "投稿の内容" );
```

**executeArticlePostWithImage()****説明**

画像を添付した状態で掲示板の投稿作成画面を立ち上げます。

**構文**

```
public static void executeArticlePostWithImage(int menuId, string subject, string
content, string filePath) {
    sharedInstance().executeArticlePostWithImage (menuId, subject, content, filePath);
}
```

**パラメータ**

パラメータ	タイプ	必須	説明
menuId	int	Y	掲示板の ID(基本値: -1). 掲示板の ID は、NAVER Cafe 掲示板 URL の <b>menuid</b> パラメータ値にあたります。
subject	string	Y	投稿のタイトル
content	string	Y	投稿の内容
filePath	string	Y	画像ファイルのパス(URI 形式)

**返却値**

なし

### サンプルコード

```
{
GlinkUnity.executeArticlePostWithImgae(28290504, "投稿のタイトル", "投稿の内容", "/NAVER
Cafesdk/glink.png" );
}
```

## executeArticlePostWithVideo()

### 説明

動画を添付した状態で掲示板の投稿作成画面を立ち上げます。

### 構文

```
public static void executeArticlePostWithVideo(int menuId, string subject, string
content, string filePath) {
    sharedInstance().executeArticlePostWithVideo (menuId, subject, content, filePath);
}
```

### パラメータ

パラメータ	タイプ	必須	説明
menuId	int	Y	掲示板の ID(基本値: -1). 掲示板の ID は、NAVER Cafe 掲示板の URL の <b>menuid</b> パラメータ値にあたります。
subject	string	Y	投稿のタイトル
content	string	Y	投稿の内容
filePath	string	Y	動画ファイルのパス(URI 形式)

### 返却値

なし

### サンプルコード

```
{
GlinkUnity.executeArticlePostWithImgae(28290504, "投稿のタイトル", "投稿の内容", "/NAVER
Cafesdk/glink.avi" );
}
```

## executeMain()

### 説明

NAVER Cafe SDK 画面を立ち上げます。

### 構文

```
public static void executeMain() {
    sharedInstance().executeMain ();
}
```

### パラメータ

なし

### 返却値

なし

### サンプルコード

```
{  
  GlinkUnity.executeMain ();  
}
```