



fluct 広告配信 SDK 導入仕様書 for iOS

改版履歴

Ver	日付	変更内容
1.0.0	2011/09/20	新規リリース
1.1.0	2012/01/04	デモグラ機能追加、ブラウザ起動コンバージョン通知対応
1.1.1	2012/01/27	armv6 対応
1.1.2	2012/03/13	メモリリーク対応
1.1.3	2012/03/29	コンバージョン計測用 UDID 処理削除
1.2.0	2012/05/11	コンバージョン計測用 UUID 対応
1.3.0	2012/07/20	広告表示高速化対応
1.4.0	2012/09/28	armv7s 対応 デフォルト View サイズ変更(300x50)
1.4.1	2013/03/01	広告表示効率改善
1.4.2	2013/03/05	AdNetwork 向け対応
1.4.3	2013/05/17	他 SDK との衝突回避対応
1.4.4	2013/05/31	Ver1.4.3 の armv7s 漏れ対応
1.5.0	2013/09/11	広告リフレッシュ時のアニメーション対応
2.0.0	2013/11/26	インタースティシャル広告対応 クラス名変更(ZucksAdnet→FluctSDK)
3.0.0	2014/06/26	arm64 対応 IDFA 対応
3.0.2	2014/10/22	Xcode6 ビルド時のインタースティシャル広告向き問題対応
4.0.0	2015/03/20	動画広告対応 バナー広告コールバック追加
4.0.1	2015/04/17	Import 対象ファイル変更
4.1.0	2015/05/19	動画広告の改善
4.1.1	2015/05/28	動画広告の仕様変更対応
4.2.0	2015/07/01	コンバージョン計測機能削除
4.2.1	2015/07/22	バナー広告の View 再追加時に、広告が表示されない不具合対応 動画広告の特定条件下における不具合対応
4.3.0	2015/11/02	ATS(App Transport Security)対応
4.3.1	2016/01/13	CocoaPods 対応 フレームワーク化
4.3.2	2016/01/20	Swift プロジェクト時の CocoaPods 利用でうまく Import 出来ないバグの修正
4.3.3	2016/04/26	Bitcode に対応 modulecache の warning が出してしまうバグの修正
4.3.4	2016/05/17	IP v6 Only Network に正式対応
4.3.5	2016/06/27	サポートバージョンを iOS6.0 以上に変更
4.3.6	2016/08/04	Swift のサンプルプロジェクト/ドキュメントを追加
4.3.7	2016/09/12	ライフサイクルメソッドの不具合修正 FluctBannerView のインタフェース一部変更

1. 動作開発環境

- 対象 OS
iOS6～

- パッケージ内容

FluctSDK/		SDK ディレクトリ
	FluctSDK.framework	FluctSDK 本体
	GSMMovieResources.bundle	リソースファイル
fluct 広告配信 SDK 導入仕様書 (iOS).pdf		
LICENSE		SDK のライセンス
README.md		SDK 利用の README
SampleApp/		広告表示サンプルプロジェクトディレクトリ
	Swift	Swift のサンプルプロジェクト
	Objective-C	Objective-C のサンプルプロジェクト

- 依存フレームワーク

- libxml2.tbd(Xcode6 以前では libxml2.dylib)
- SystemConfiguration.framework
- AdSupport.framework
- AVFoundation.framework
- MediaPlayer.framework
- CFNetwork.framework
- CoreMedia.framework
- CoreTelephony.framework
- Social.framework
- StoreKit.framework

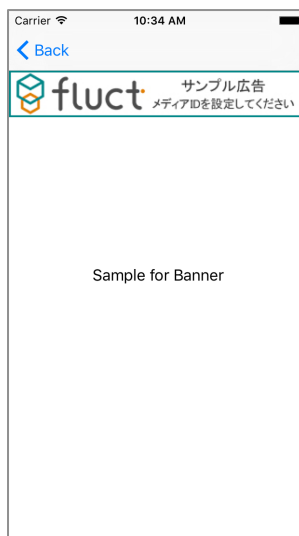
2. メディア ID について

「メディア ID」は弊社システムでアプリを識別する ID となっております。
貴社アプリのメディア ID は弊社担当営業にお問い合わせください。

本ドキュメントではテスト用メディア ID「0000000108」を使用しております。

3. バナー広告について

Web で一般的なバナー型の広告を表示します。
標準サイズは 320 x 50 (pt)です。



4. インタースティシャル広告について

アプリケーションの画面全体を覆うように表示する
広告です。



SDK のプロジェクトへの導入方法

■ CocoaPods を利用して追加する場合

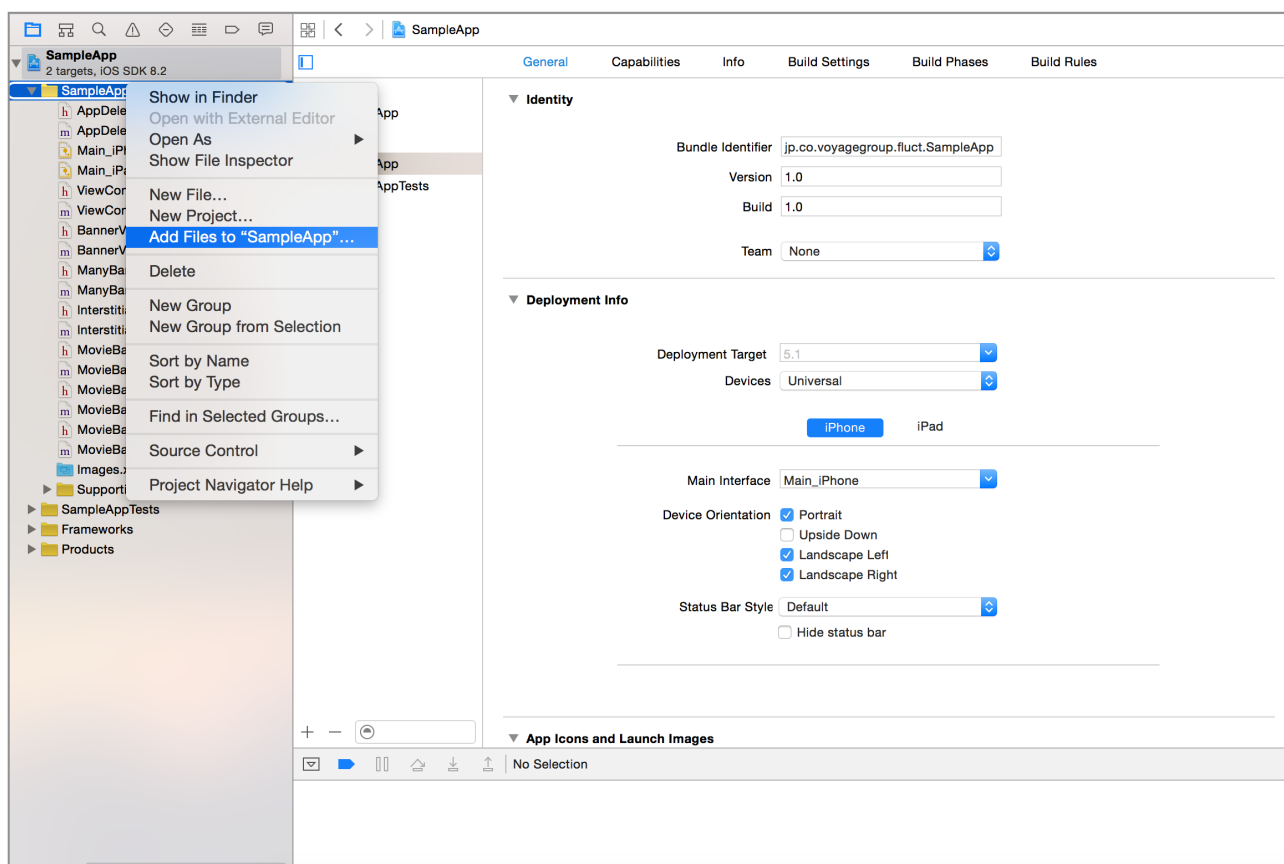
1. Podfile のアプリケーションのターゲットに対して pod 'FluctSDK' と追加してください。
2. コマンドラインで pod install を実行してください。
3. CocoaPods によって作られた.xcworkspace ファイルを開いてください。

■ 手動で追加する場合

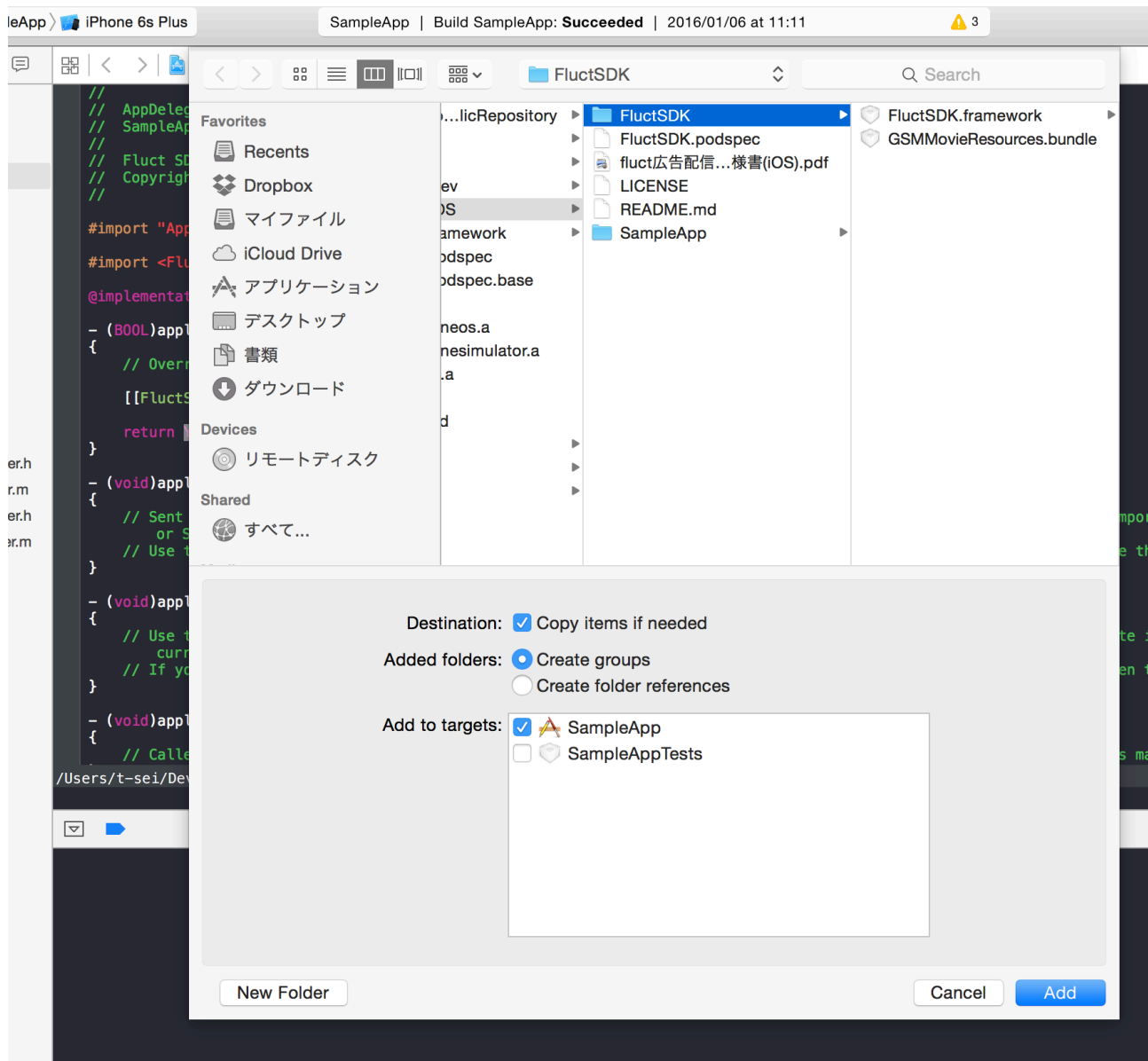
1. SDK ファイルの追加

FluctSDK-iOS リポジトリ(<https://github.com/voyagegroup/FluctSDK-iOS>)内の FluctSDK フォルダ配下のファイルを全て対象のプロジェクトに追加してください。

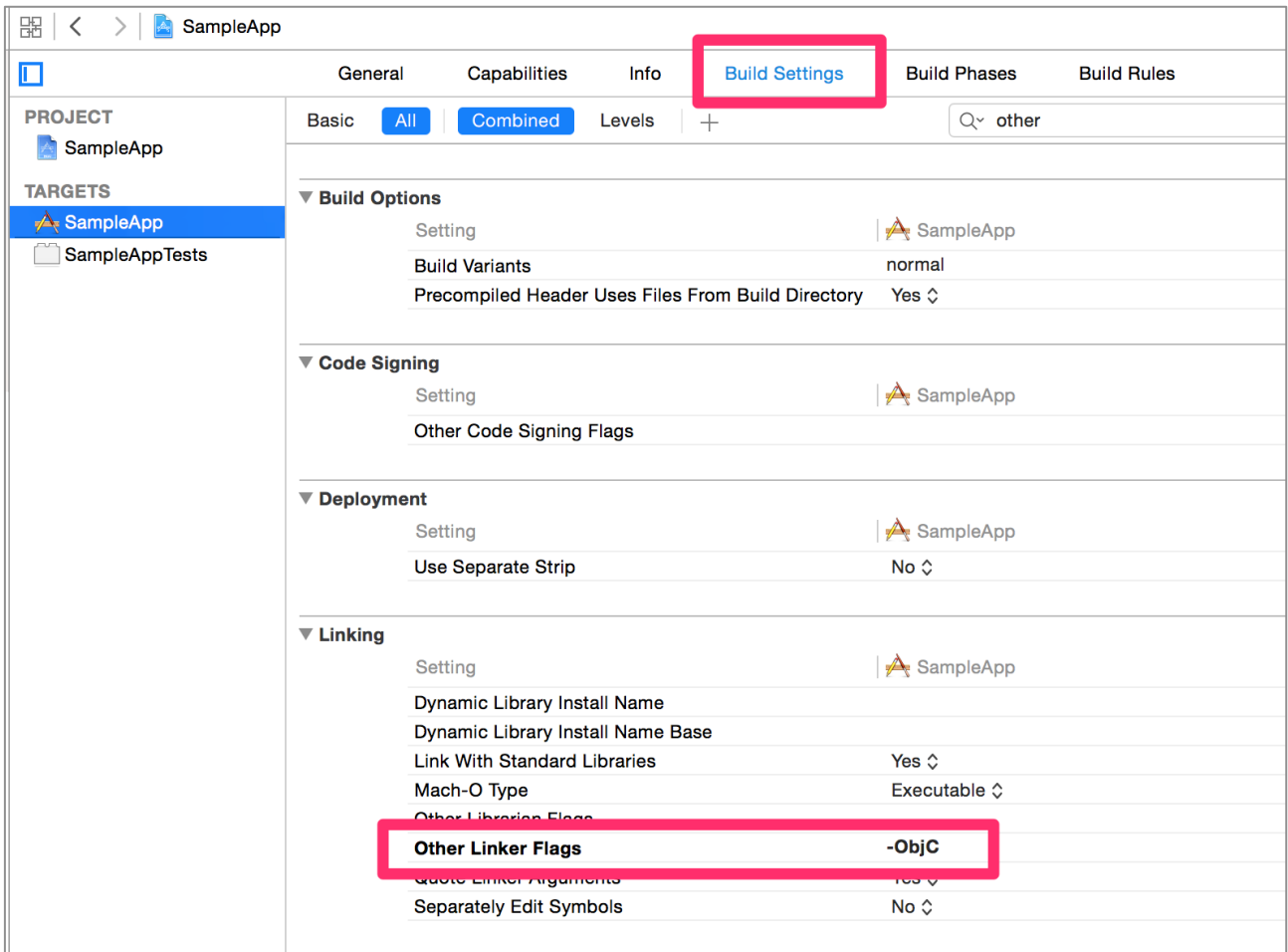
- (1) SDK ファイル一式を任意のディレクトリに配置してください。導入するプロジェクトで参照します。
- (2) SDK ファイル一式を利用するように Xcode を設定してください。
プロジェクト内の任意のグループを右クリックし、ポップアップメニューより、「Add Files to ...」を選択し、ファイル選択ダイアログを表示させます。



- (3) ファイル選択ダイアログにて、SDK ファイル一式が入った FluctSDK フォルダを選択し、Add ボタンをクリックします。



- (4) プロジェクトを選択し、広告を導入する TARGETS を選択。
Build Settings の OtherLinkerFlags に[-ObjC]を設定します。



The screenshot shows the Xcode interface for the 'SampleApp' project. The 'Build Settings' tab is selected, and the 'SampleApp' target is chosen from the left sidebar. The 'Other Linker Flags' setting under the 'Linking' section is highlighted with a red box, showing the value '-ObjC'.

Section	Setting	Value
Build Options	Setting	SampleApp
	Build Variants	normal
	Precompiled Header Uses Files From Build Directory	Yes
Code Signing	Setting	SampleApp
	Other Code Signing Flags	
Deployment	Setting	SampleApp
	Use Separate Strip	No
Linking	Setting	SampleApp
	Dynamic Library Install Name	
	Dynamic Library Install Name Base	
	Link With Standard Libraries	Yes
	Mach-O Type	Executable
	Other Linker Flags	-ObjC
	Quote Linker Arguments	Yes
	Separately Edit Symbols	No

広告の実装 (Swift)

1. バナー広告表示

バナー型の広告を表示します。

サンプルコード

```
import UIKit
import FluctSDK //(1.)

class BannerViewController: UIViewController {

    override func viewDidLoad() {
        super.viewDidLoad()

        let bannerView = FluctBannerView.init() //(2.)
        bannerView.setMediaID("0000000108") //(3.)
        self.view.addSubview(bannerView) //(4.)
    }
}
```

1. FluctSDK をインポートします。
2. FluctBannerView を生成します。
3. FluctBannerView.setMediaID() を使用しメディア ID を設定します。
4. self.view に追加します。

貴社アプリのメディア ID は弊社担当営業にお問い合わせください。

※ FluctBannerView の デフォルトの frame は {0.0, 0.0, 320.0, 50.0} です。必要に応じて変更してください。

frame サイズ(広告サイズ) 変更サンプルコード

```
let bannerView = FluctBannerView.init(frame: CGRectMake(90, 200, 320, 250))
self.view.addSubview(bannerView)
```


バナー広告の各状態のコールバックを受け取ることが可能です。(任意)

```
import UIKit
import FluctSDK

class BannerViewController: UIViewController, FluctBannerViewDelegate { //(1.)

    override func viewDidLoad() {
        super.viewDidLoad()

        let bannerView = FluctBannerView.init()
        bannerView.setMediaID("0000000108")
        bannerView.delegate = self //(2.)
        self.view.addSubview(bannerView)
    }

    func fluctBannerView(bannerView: FluctBannerView!, callbackValue: Int) { //(3.)
        print(callbackValue)
    }
}
```

1. コールバックを受け取るため FluctBannerViewDelegate を採用します。
2. delegate を設定します。
3. func fluctBannerView(bannerView: FluctBannerView!, callbackValue: Int) が呼ばれます。
引数 callbackValue は FluctBannerViewCallbackType を参照してください。

バナー広告 コールバック (FluctBannerViewCallbackType)

バナー広告の表示状態を通知します。通知される状態は下記の通りです。

- 0: FluctBannerLoad
広告ロードの際に通知されます。
- 1: FluctBannerTap
広告がタップされた際に通知されます。
- 2: FluctBannerOffline
広告表示の際に圏外状態の際に通知されます。
- 3: FluctBannerMediaIdError
設定されている FLUCT_MEDIA_ID が不正な際に通知されます。
- 4: FluctBannerNoConfig
広告情報が取得できなかった際に通知されます。
設定している FLUCT_MEDIA_ID を再度ご確認ください。
- 5: FluctBannerGetConfigError
広告設定情報が取得できなかった際に通知されます。
設定している FLUCT_MEDIA_ID を再度ご確認ください。
- 100: FluctBannerOtherError
上記以外のエラーの際に通知されます。

2. UITabBarController 及び UINavigationController の View にバナー広告を表示する場合

UITabBarController 及び UINavigationController の View に直接バナー広告を表示する際は、UIViewController で表示する処理に加えて、下記のように setRootViewController メソッドを呼んでください

サンプルコード

```
import UIKit
import FluctSDK

class TabBarController: UITabBarController {

    override func viewDidLoad() {
        super.viewDidLoad()

        let bannerView = FluctBannerView.init()
        bannerView.setMediaID("00000000108")
        bannerView.setRootViewController(self) // 追加分
        self.view.addSubview(bannerView)
    }
}
```

3. 複数バナー表示

1つの画面内に複数のバナー型広告を表示することが可能です。

必要な数の FluctBannerView を生成し self.view に追加します。

各々の広告の位置は FluctBannerView.init(frame: CGRectMake()) で適宜変更してください。

※詳細はサンプルプロジェクトを確認して下さい。

<https://github.com/voyagegroup/FluctSDK-iOS/tree/master/SampleApp/Swift>

4. 共通メディア ID の設定

全てのバナー広告、インタースティシャル広告をデフォルトで使用するメディア ID を設定します。

通常、アプリケーション起動時に一度設定します。

貴社アプリのメディア ID は弊社担当営業にお問い合わせください。

サンプルコード

```
import UIKit
import FluctSDK //(1.)

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {

    var window: UIWindow?

    func application(application: UIApplication,
        didFinishLaunchingWithOptions launchOptions: [NSObject: AnyObject]?) -> Bool {
        //(2.)メディアIDの設定
        FluctSDK.sharedInstance().setBannerConfiguration("00000000108")
        return true
    }
}
```

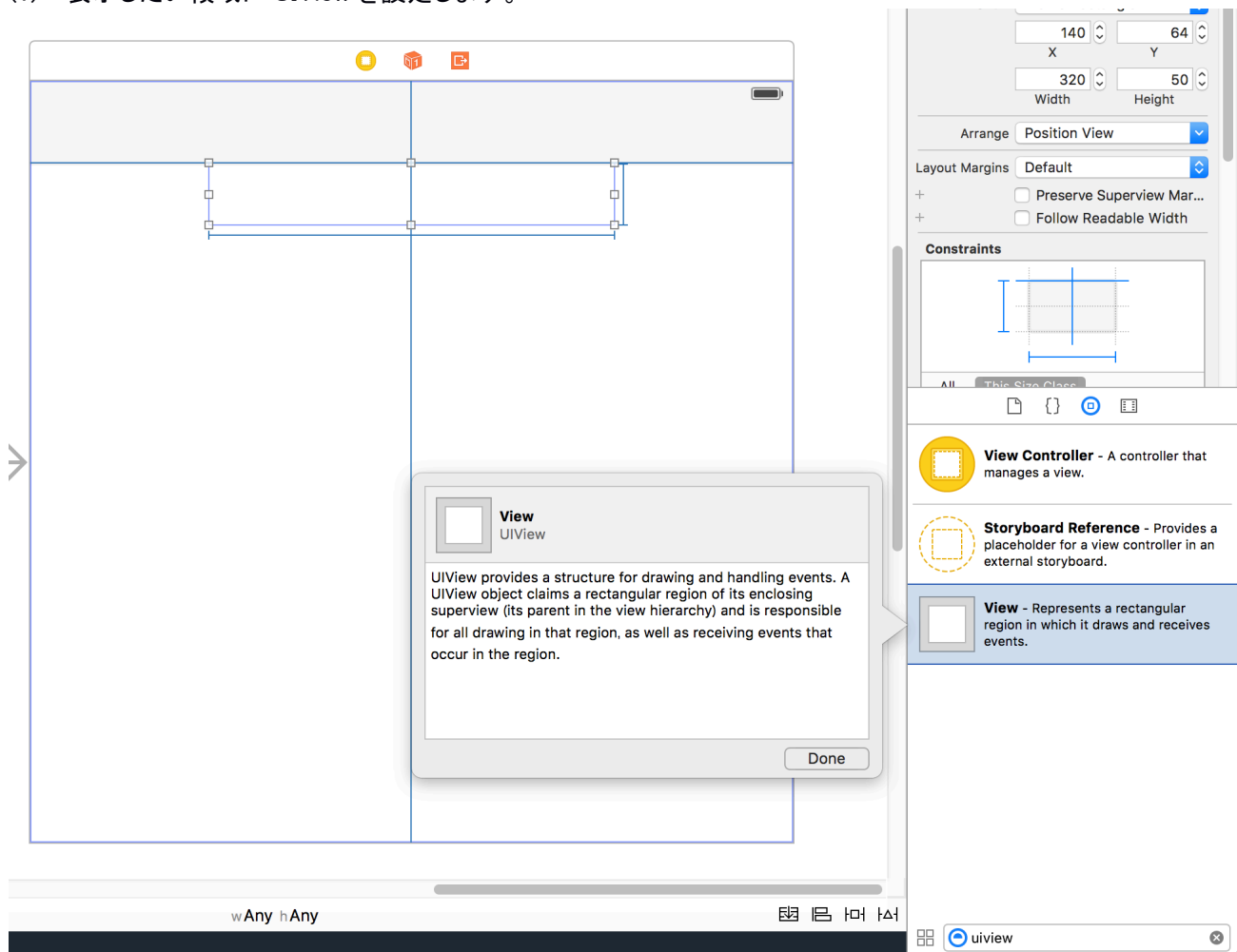
1. FluctSDK をインポートします。

2. FluctSDK.sharedInstance().setBannerConfiguration() を使用しメディア ID を設定します。

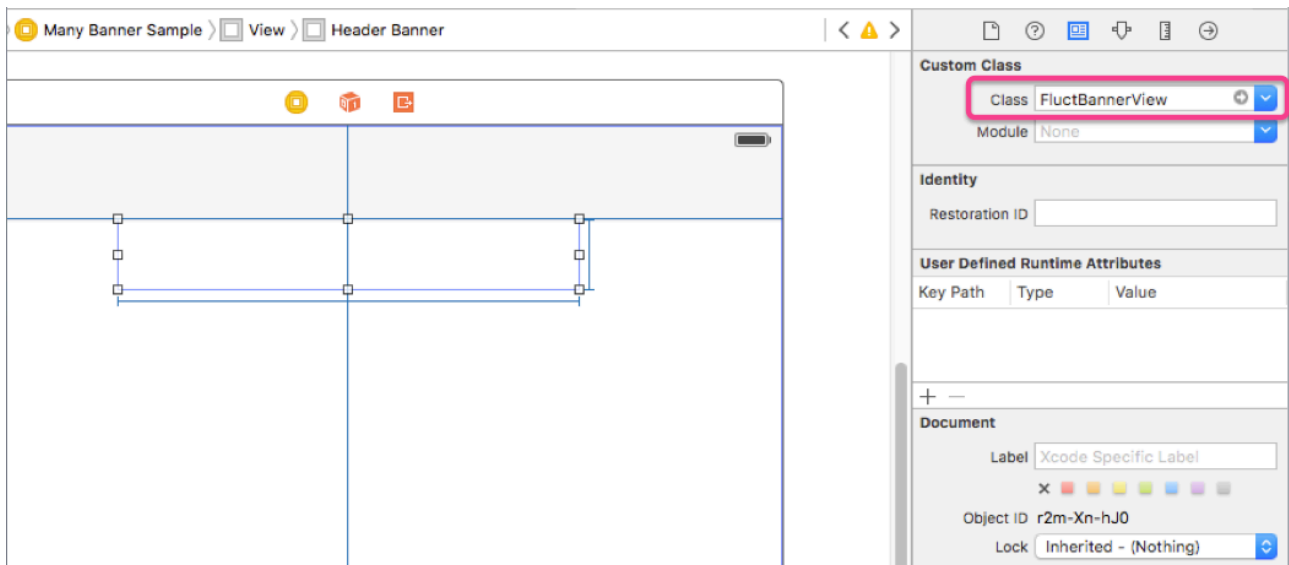
5. StoryBoard で UIView を設定して広告表示

Storyboard から UIView で広告表示領域を確保して広告を表示することが可能です。

(1) 表示したい領域に UIView を設定します。



(2) 設定した UIView の Custom Class に FluctBannerView を設定します。



(3) 設定した UIView に広告表示する実装を行います。

サンプルコード

```
import UIKit
import FluctSDK // (1.)

class ManyBannerViewController: UIViewController, FluctBannerViewDelegate {

    @IBOutlet weak var headerBanner: FluctBannerView! // (2.)

    override func viewDidLoad() {
        super.viewDidLoad()
        headerBanner?.setMediaID("0000000108") //(3.)
        self.view.addSubview(headerBanner) //(4.)
    }
}
```

1. FluctSDK をインポートします。
2. StoryBoard から対象のファイルに設定した UIView を紐付けます。
3. FluctBannerView.setMediaID() を使用しメディア ID を設定します。
4. self.view に追加します。

6. インターstitial広告表示

インターstitial広告を表示します。

サンプルコード

```
import UIKit
import FluctSDK //(1.)

class InterstitialViewController: UIViewController {
    var interstitialView: FluctInterstitialView? = nil //(2.)

    override func viewDidLoad() {
        super.viewDidLoad()
    }

    @IBAction func showInterstitial(sender: AnyObject) {
        interstitialView = FluctInterstitialView.init() //(3.)
        interstitialView?.mediaID = "0000000108" //(4.)
        // interstitialView = FluctInterstitialView(mediaID: "0000000108") //(5.)
        interstitialView?.showInterstitialAd() //(6.)
    }
}
```

1. FluctSDK をインポートします。
2. FluctInterstitialView を保持するプロパティを追加します。
3. FluctInterstitialView を生成します。
4. メディア ID を設定します。
5. 3.4.を一行で書くことも可能です
6. FluctInterstitialView.showInterstitialAd() を呼び出しインターstitial広告を表示します。

※ 弊社設定で表示率を設定することができます。表示率に関しては弊社担当営業にお問い合わせください。

インタースティシャル広告の各状態のコールバックを受け取ることが可能です。

```
import UIKit
import FluctSDK

class InterstitialViewController: UIViewController, FluctInterstitialViewDelegate { //(1.)

    var interstitialView: FluctInterstitialView? = nil

    override func viewDidLoad() {
        super.viewDidLoad()
    }

    @IBAction func showInterstitial(sender: AnyObject) {
        interstitialView = FluctInterstitialView(mediaID: "00000000108")
        interstitialView?.delegate = self //(2.)
        interstitialView?.showInterstitialAd()
    }

    func fluctInterstitialView(interstitialView: FluctInterstitialView!, callbackValue: Int) { //(3.)
        print(callbackValue)
    }
}
```

1. コールバックを受け取るため FluctInterstitialViewDelegate を採用します。
2. delegate を設定します。
3. func fluctInterstitialView:callbackValue が呼ばれます。
引数 callbackValue は FluctInterstitialViewCallbackType を参照してください。

インタースティシャル広告 コールバック(FluctInterstitialViewCallbackType)
インタースティシャル広告の表示状態を通知します。通知される状態は下記の通りです。

- 0: FluctInterstitialShow
広告表示がされた際に通知されます。
- 1: FluctInterstitialTap
広告がタップされた際に通知されます。
なお、広告がタップされた際はインタースティシャル広告は閉じられます
- 2: FluctInterstitialClose
広告が閉じられた際に通知されます。
- 3: FluctInterstitialCancel
広告表示が表示率に基づき表示されなかった際に通知されます。
表示率に関しては、弊社営業まで問い合わせください。
- 4: FluctInterstitialOffline
広告表示の際に圏外状態の際に通知されます。
- 5: FluctInterstitialMediaIDError
設定されている FLUCT_MEDIA_ID が不正な際に通知されます。
- 6: FluctInterstitialNoConfig
FLUCT_MEDIA_ID に設定されてない際に通知されます。
- 7: FluctInterstitialSizeError
広告サイズが表示する端末サイズを超えた際に通知されます。
- 8: FluctInterstitialGetConfigError
広告情報が取得できなかった際に通知されます。
設定している FLUCT_MEDIA_ID を再度ご確認ください。
- 100: FluctInterstitialOtherError
上記以外のエラーの際に通知されます。
広告表示中に、同一インスタンスから広告表示要求が呼び出された際も通知されます。

通常インタースティシャル広告はユーザー操作により非表示になりますが、明示的に非表示にすることも可能です。

```
override func willRotateToInterfaceOrientation(toInterfaceOrientation: UIInterfaceOrientation,
duration: NSTimeInterval) {
    interstitialView?.dismissInterstitialAd()
}
```

1. 任意のタイミングで FluctInterstitialView.dismissInterstitialAd() を呼び出しインタースティシャル広告を非表示にします。上記サンプルでは画面の回転時に非表示にしています。

広告の実装 (Objective-C)

1. バナー広告表示

バナー型の広告を表示します。

サンプルコード

```
#import "ViewController.h"

#import FluctSDK; // (1.)

@implementation ViewController

- (void)viewDidLoad
{
    [super viewDidLoad];
    FluctBannerView *bannerView = [[FluctBannerView alloc] init]; // (2.)
    [bannerView setMediaID:@"0000000108"]; // (3.)
    [self.view addSubview:bannerView]; // (4.)
}
```

1. FluctSDK をインポートします。
※ Objective-C の場合、Build Setting → Enable Modules (C and Objective-C) → Yes に設定
2. FluctBannerView を生成します。
3. [FluctBannerView setMediaID:] を使用しメディア ID を設定します。
4. self.view に追加します。

※ FluctBannerView の デフォルトの frame は {0.0, 0.0, 320.0, 50.0} です。必要に応じて変更してください。

frame サイズ(広告サイズ) 変更サンプルコード

```
FluctBannerView *bannerView =
    [[FluctBannerView alloc] initWithFrame:CGRectMake(90, 200, 300, 250)];
[self.view addSubview:bannerView];
[bannerView addSubview:bannerView];
```

バナー広告の各状態のコールバックを受け取ることが可能です。(任意)

```
#import "ViewController.h"

#import FluctSDK;

@interface ViewController () <FluctBannerViewDelegate> // (1.)

@end

@implementation ViewController

- (void)viewDidLoad
{
    [super viewDidLoad];
    FluctBannerView *bannerView = [[FluctBannerView alloc] init];
    [bannerView setMediaID:@"0000000108"];
    bannerView.delegate = self; // (2.)
    [self.view addSubview:bannerView];
}

- (void)fluctBannerView:(FluctBannerView *)bannerView callbackValue:(NSInteger)callbackValue {
    // (3.)
    NSLog(@"%d", (int)callbackValue);
}
```

1. コールバックを受け取るため FluctBannerViewDelegate を採用します。
2. delegate を設定します。
3. - (void)fluctBannerView:(FluctBannerView *)bannerView callbackValue:(NSInteger)callbackValue が呼ばれます。引数 callbackValue は FluctBannerViewCallbackType を参照してください。

バナー広告 コールバック (FluctBannerViewCallbackType)

バナー広告の表示状態を通知します。通知される状態は下記の通りです。

- 0: FluctBannerLoad
広告ロードの際に通知されます。
- 1: FluctBannerTap
広告がタップされた際に通知されます。
- 2: FluctBannerOffline
広告表示の際に圏外状態の際に通知されます。
- 3: FluctBannerMediaIdError
設定されている FLUCT_MEDIA_ID が不正な際に通知されます。
- 4: FluctBannerNoConfig
広告情報が取得できなかった際に通知されます。
設定している FLUCT_MEDIA_ID を再度ご確認ください。
- 5: FluctBannerGetConfigError
広告設定情報が取得できなかった際に通知されます。
設定している FLUCT_MEDIA_ID を再度ご確認ください。
- 100: FluctBannerOtherError
上記以外のエラーの際に通知されます。

2. UITabBarController 及び UINavigationController の View にバナー広告を表示する場合

UITabBarController 及び UINavigationController の View に直接バナー広告を表示する際は、UIViewController で表示する処理に加えて、下記のように setRootViewController メソッドを呼んでください

サンプルコード

```
#import "TabBarController.h"

#import FluctSDK;

@implementation TabBarController

- (void)viewDidLoad
{
    [super viewDidLoad];
    FluctBannerView *bannerView = [[FluctBannerView alloc] init];
    [bannerView setMediaID:@"0000000108"];
    [bannerView setRootViewController:self]; // 追加分
    [self.view addSubview:bannerView];
}
```

3. 複数バナー表示

1つの画面内に複数のバナー型広告を表示することが可能です。

必要な数の `FluctBannerView` を生成し `self.view` に追加します。各々の広告の位置は `-[FluctBannerView setFrame:]` で適宜変更してください。

※詳細はサンプルプロジェクトを確認して下さい。

<https://github.com/voyagegroup/FluctSDK-iOS/tree/master/SampleApp/Objective-C>

4. 共通メディア ID の設定

全てのバナー広告、インタースティシャル広告をデフォルトで使用されるメディア ID を設定します。通常、アプリケーション起動時に一度設定します。

貴社アプリのメディア ID は弊社担当営業にお問い合わせください。

サンプルコード

```
#import "AppDelegate.h"

#import FluctSDK; // (1.)

@implementation AppDelegate

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:(NSDictionary *)launchOptions
{
    // Override point for customization after application launch.

    // (2.) メディア ID の設定
    [[FluctSDK sharedInstance] setBannerConfiguration:@"0000000108"];
    return YES;
}
```

1. `FluctSDK` をインポートします。

2.- `[[FluctSDK sharedInstance] setBannerConfiguration:]` を使用しメディア ID を設定します。

5. インターstitial広告表示

インターstitial広告を表示します。

サンプルコード

```
#import "ViewController.h"

#import FluctSDK; // (1.)

@interface ViewController ()
@property (nonatomic, strong, readonly) FluctInterstitialView *interstitialView; // (2.)
@end

@implementation ViewController

- (void)viewDidLoad
{
    [super viewDidLoad];
}

- (IBAction)showInterstitial:(id)sender {
    self.interstitialView = [[FluctInterstitialView alloc] init]; // (3.)
    [self.interstitialView setMediaId: @"00000000108"]; // (4.)
    [self.interstitialView showInterstitialAd]; // (5.)
}
```

1. FluctSDK をインポートします。
※ Objective-C の場合、Build Setting → Enable Modules (C and Objective-C) → Yes に設定
2. FluctInterstitialView を保持するプロパティを追加します。
3. FluctInterstitialView を生成します。
4. メディア ID を設定します。
5. -[FluctInterstitialView showInterstitialAd] を呼び出しインターstitial広告を表示します。弊社設定で表示率を設定することができます。表示率に関しては弊社担当営業にお問い合わせください。

インタースティシャル広告の各状態のコールバックを受け取ることが可能です。

```
#import "ViewController.h"

#import FluctSDK;

@interface ViewController () <FluctInterstitialViewDelegate> // (1.)
@property (nonatomic, strong, readonly) FluctInterstitialView *interstitialView;
@end

@implementation ViewController

- (void)viewDidLoad
{
    [super viewDidLoad];
}

- (IBAction)showInterstitial:(id)sender {
    self.interstitialView = [[FluctInterstitialView alloc] init];
    [self.interstitialView setMediald: @"0000000108"];
    self.interstitialView.delegate = self; // (2.)
    [self.interstitialView showInterstitialAd];
}

// (3.)
- (void)fluctInterstitialView:(FluctInterstitialView *)interstitialView callbackValue:(NSInteger)callbackValue
{
    NSLog(@"%ld", (long)callbackValue);
}
```

1. コールバックを受け取るため `FluctInterstitialViewDelegate` を採用します。
2. `delegate` を設定します。
3. `-[FluctInterstitialViewDelegate fluctInterstitialView:callbackValue:]` が呼ばれます。
引数 `callbackValue` は `FluctInterstitialViewCallbackType` を参照してください。

インタースティシャル広告 コールバック(FluctInterstitialViewCallbackType)
インタースティシャル広告の表示状態を通知します。通知される状態は下記の通りです。

- 0: FluctInterstitialShow
広告表示がされた際に通知されます。
- 1: FluctInterstitialTap
広告がタップされた際に通知されます。
なお、広告がタップされた際はインタースティシャル広告は閉じられます
- 2: FluctInterstitialClose
広告が閉じられた際に通知されます。
- 3: FluctInterstitialCancel
広告表示が表示率に基づき表示されなかった際に通知されます。
表示率に関しては、弊社営業まで問い合わせください。
- 4: FluctInterstitialOffline
広告表示の際に圏外状態の際に通知されます。
- 5: FluctInterstitialMediaIDError
設定されている FLUCT_MEDIA_ID が不正な際に通知されます。
- 6: FluctInterstitialNoConfig
FLUCT_MEDIA_ID に設定されてない際に通知されます。
- 7: FluctInterstitialSizeError
広告サイズが表示する端末サイズを超えた際に通知されます。
- 8: FluctInterstitialGetConfigError
広告情報が取得できなかった際に通知されます。
設定している FLUCT_MEDIA_ID を再度ご確認ください。
- 100: FluctInterstitialOtherError
上記以外のエラーの際に通知されます。
広告表示中に、同一インスタンスから広告表示要求が呼び出された際も通知されます。

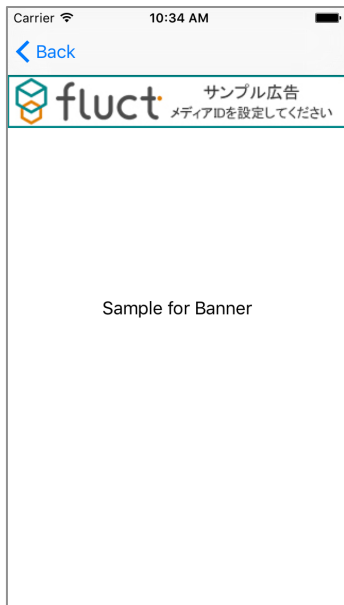
通常インタースティシャル広告はユーザー操作により非表示になりますが、明示的に非表示にすることも可能です。

```
- (void)willRotateToInterfaceOrientation:(UIInterfaceOrientation)toInterfaceOrientation
duration:(NSTimeInterval)duration
{
    [self.interstitialView dismissInterstitialAd]; // (1.)
}
```

1. 任意のタイミングで `-[FluctInterstitialView dismissInterstitialAd]` を呼び出しインタースティシャル広告を非表示にします。上記サンプルでは画面の回転時に非表示にしています。

サンプルアプリケーションについて

SampleApp フォルダ内のプロジェクトは、SDK が組み込まれた広告表示のサンプルアプリとなっております。導入の際にあわせてご参考して頂きますようお願い致します。



バナー広告表示画面



インタースティシャル広告表示画面

サンプルアプリ実行画面

取得情報について

FluctSDK では下記情報を取得し、アプリ利用者に適切な広告を表示する目的で使用しています。

	取得情報	備考
1	OS のメジャーバージョン	OS バージョン 6.1.2 の場合、「6」
2	Bundle Identifier	アプリを固有の識別子 例：jp.fluct.SampleApp-Swift
3	IDFA (Identification For Advertisers)	ユーザーが IDFA の使用を制限している場合は、取得及び広告サーバへの送信は行いません
4	キャリア情報	mobileCountryCode-mobileNetworkCode 例：440-20
5	プラットフォーム情報	例：iPhone5,2