nendSDK for iOS

ver 2.3.3

設定ガイド

2014年3月6日

更新履歴

バージョン　公開日付	更新内容
iOS_ver 1.2.0　2011/8/16	-
iOS_ver 1.2.1　2011/8/24	シミュレータ向け不具合の修正
iOS_ver 1.2.2　2012/3/27	デバイス ID（UDID）取得の停止 サイズ変更（320×48）から(320×50)の View を生成するように変更
iOS_ver 1.2.3　2012/4/11	広告取得時に不必要なメモリをリリースしない不具合の修正
iOS_ver 1.3.0　2012/6/25	広告受信成功通知の追加 広告受信エラー通知の追加 定期ロード中断の追加 定期ロード再開の追加
iOS_ver 1.3.1　2012/8/31	WebView タイプの場合ローテーションが管理されない不具合修正
iOS_ver 1.3.2　2012/9/20	iOS SDK 6 & iPhone 5 (armv7s) 対応
iOS_ver 2.0.0　2013/4/2	ターゲティング広告配信及びオプトアウト機能の実装 広告識別子 Advertising　Identifier (IDFA) 利用開始 ログ出力設定プロパティ追加 その他不具合修正
iOS_ver 2.0.1　2013/4/9	特定のライブラリ使用時に重複エラーが起きる問題に対応
iOS_ver 2.0.2　2013/4/11	特定のライブラリ使用時に重複エラーが起きる問題に追加対応 AdSupport.framework の Link 設定に関する注意事項を追記
iOS_ver 2.1.0　2013/5/29	Click イベントの通知を追加 (メディエーション対応として) 受信エラー通知メソッド内で、NADView を release した後にクラッシュする問題修正 release に関するサンプルコードを修正
iOS_ver 2.2.0　2013/7/22	広告サイズ追加対応 NSError プロパティの追加、広告サイズごとのテスト ID の追加 ◆広告サイズについて　を追加 ◆よくある質問　を WEB へ移動
iOS_ver 2.2.1　2013/9/26	iOS7 対応 arm64 アーキテクチャに対応
iOS_ver 2.3.0　2013/11/5	アイコン型広告対応
iOS_ver 2.3.1　2013/12/2	アイコン型広告サイズ変更対応 アイコン型広告余白部分を非表示にする設定を追加
iOS_ver 2.3.2　2014/1/20	不具合修正
iOS_ver 2.3.3　2014/3/6	Interface Builder での実装に対応 delegate 通知 nadViewDidFinishLoad メソッドを任意に変更 NadView, NadIconLoader の delegate 処理見直し

目次

◆nendSDK iOS について

1．nendSDK 導入のおおまかな流れ

① nend 管理画面でアプリ登録と広告枠登録を行います。

※ 本マニュアルは広告枠登録後、「apiKey」「spotID」を発行し SDK を入手している前提で説明を行います。

※ 広告枠を登録していない場合には、別紙「管理画面マニュアル」をご参照の上、ご申請ください。

※ 広告枠を申請後、広告枠の管理＞広告枠＞SDK＞「SDK をダウンロード」 で SDK を入手できます。

②本マニュアルに従って nendSDK をアプリに組み込みます。

2．ファイル構成

NendSDK_iOS.zip

NendAd/	SDK フォルダ
libNendAd.a	ライブラリ
NADView.h	バナー広告用ヘッダファイル
NADIconView.h	アイコン広告ビュー用ヘッダファイル
NADIconLoader.h	アイコンローダー用ヘッダファイル
NADIconArrayView.h	アイコン広告ビュー用ヘッダファイル(InterfaceBuilder 用)
NADView_readme.txt	ライセンス文等
Samples/	サンプルソース
nendSDK2.3.0_manual.pdf	本マニュアル

3．対応環境

デバイスは以下の環境にて動作確認を行っています。

デバイス	モデル
iPhone	iPhone3GS、iPhone4、iPhone4S、iPhone5、iPhone5S、iPhone5C
iPad	iPad、iPad2、iPad（第 3 世代以降）、iPad mini
iPod Touch	iPodTouch（第 3 世代以降）

OS バージョンは、iOS 4.3 以上 iOS7.0.4 (2014/1/20 時点現在最新) が動作保障対象となります。

それ以外の端末では正常に動作しない場合があります。

開発環境

Xcode 5.0 以上が必要です。

４．インフォメーションボタンについて

アプリに配信される広告バナーに**「インフォメーションボタン」**が表示されます。
この機能が追加された nendSDK を利用することにより、弊社システムが提供する一部ターゲティング広告

に対して、よりユーザ（アプリ利用者）自身が容易にオプトアウトの設定をすることが可能になります。

平成 24 年より、総務省においてスマートフォンのプライバシーに関する議論がなされ、
「スマートフォンプライバシーイニシアティブ」として取りまとめがされております。

参考ＵＲＬ
http://www.soumu.go.jp/menu_news/s-news/01kiban08_02000087.html

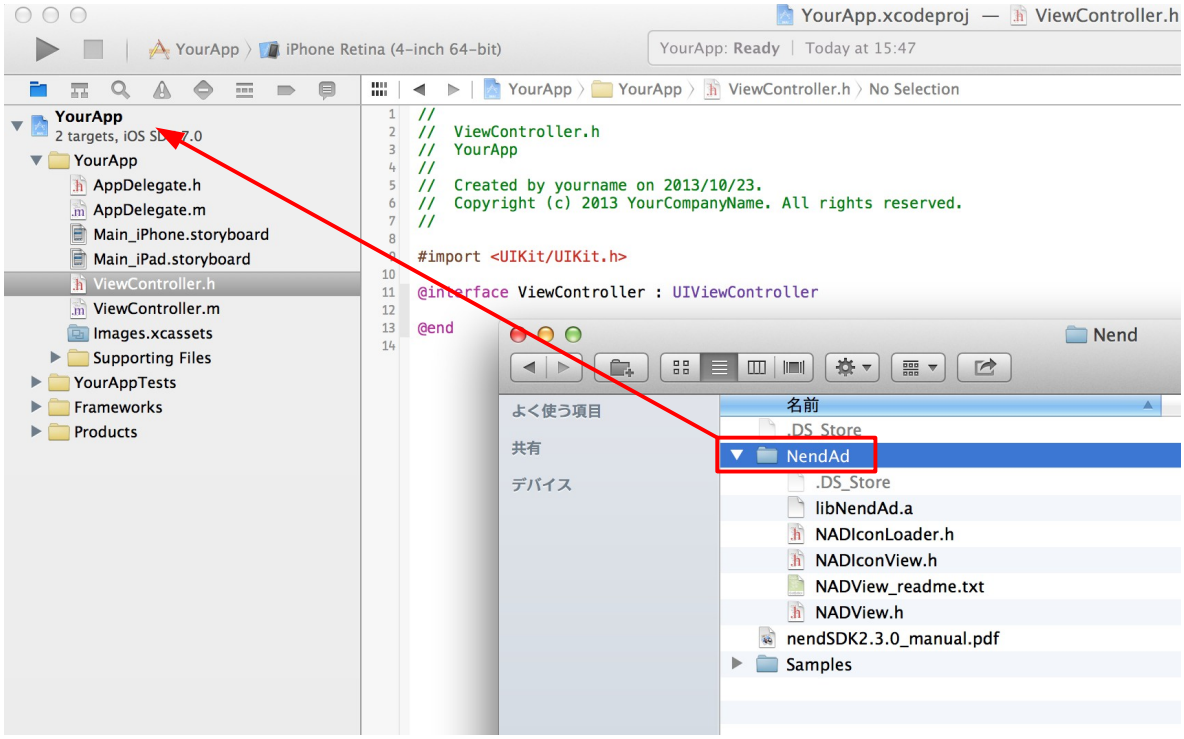
アプリ提供者、情報収集モジュール提供者、広告配信事業者の自主的かつ積極的な取り組みを期待されておりますので、メディアパートナー様におかれましてもご確認頂きますようお願いいたします。また、弊社においても広告配信システム提供事業者として、より一層透明性の高い取り組みを実施していく所存でございます。
尚、当ＳＤＫにおける取得情報は、以下のようになっており、個人情報に該当するものは取得しておりません。

・匿名ＩＤ （ android ではＵＩＩＤ / iOS では UIID または広告識別子）、端末情報（ＯＳ情報、端末機種種類、ＵＲＬスキーム情報、Package 名等）
※ＵＲＬスキーム情報、Package 名に関しては、ＳＤＫ内でのみ利用しており、外部送信および保存等を行うことは一切ございません。

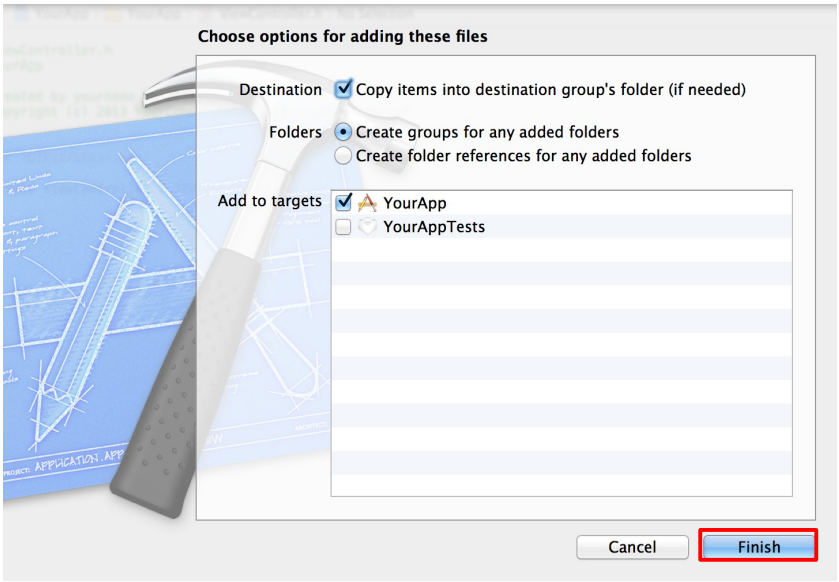
◆SDK の組み込み

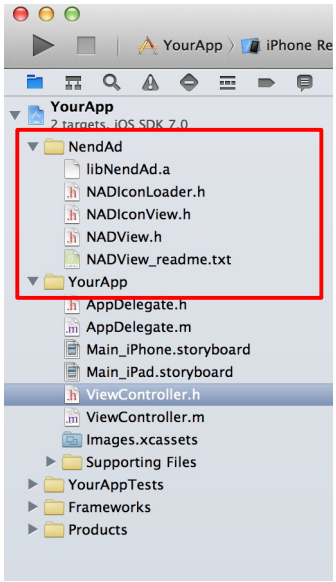
1．プロジェクトへの nendSDK 追加

Xcode 上で対象プロジェクトに「NendAd」フォルダごとドラッグ＆ドロップします。



下記、左図ダイアログの表示で必要に応じて任意の設定(※)を選び「Finish」をクリックします。





追加が完了すると Xcode のファイルリストに
右図のように表示されます。

※ 既にプロジェクトフォルダ配下に NendAd フォルダを移動させたものに対して参照設定を追加する場合には、
ファイルをコピーする必要がないため、“Copy items into destination group's folder (if needed) ” にチェックを入れる必要はありません。

2．必須フレームワークの追加

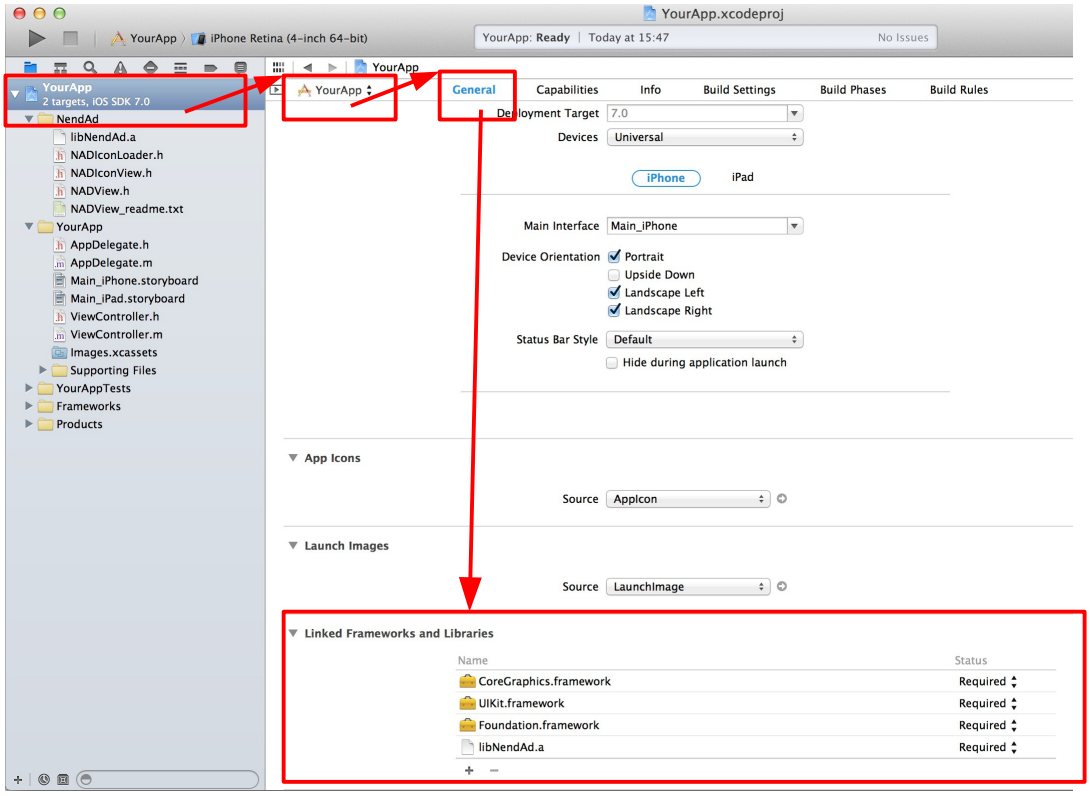
nendSDK の利用には、

・**AdSupport.framework**

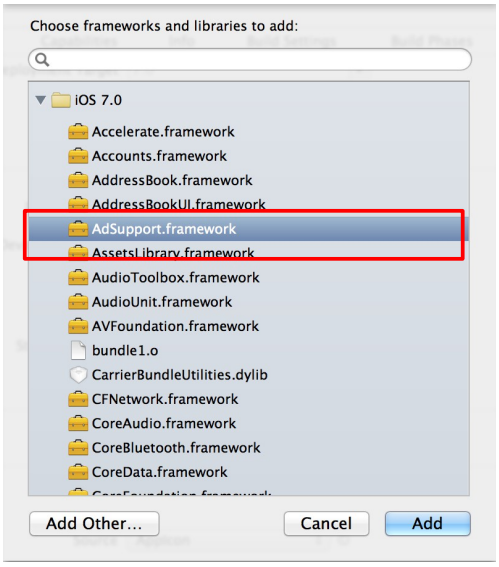
・**Security.framework**

の追加が必要です。

左側のプロジェクトナビゲーターから、プロジェクトをクリックして
TARGETS＞General＞Link Binary With Libraries 項目を開きます。

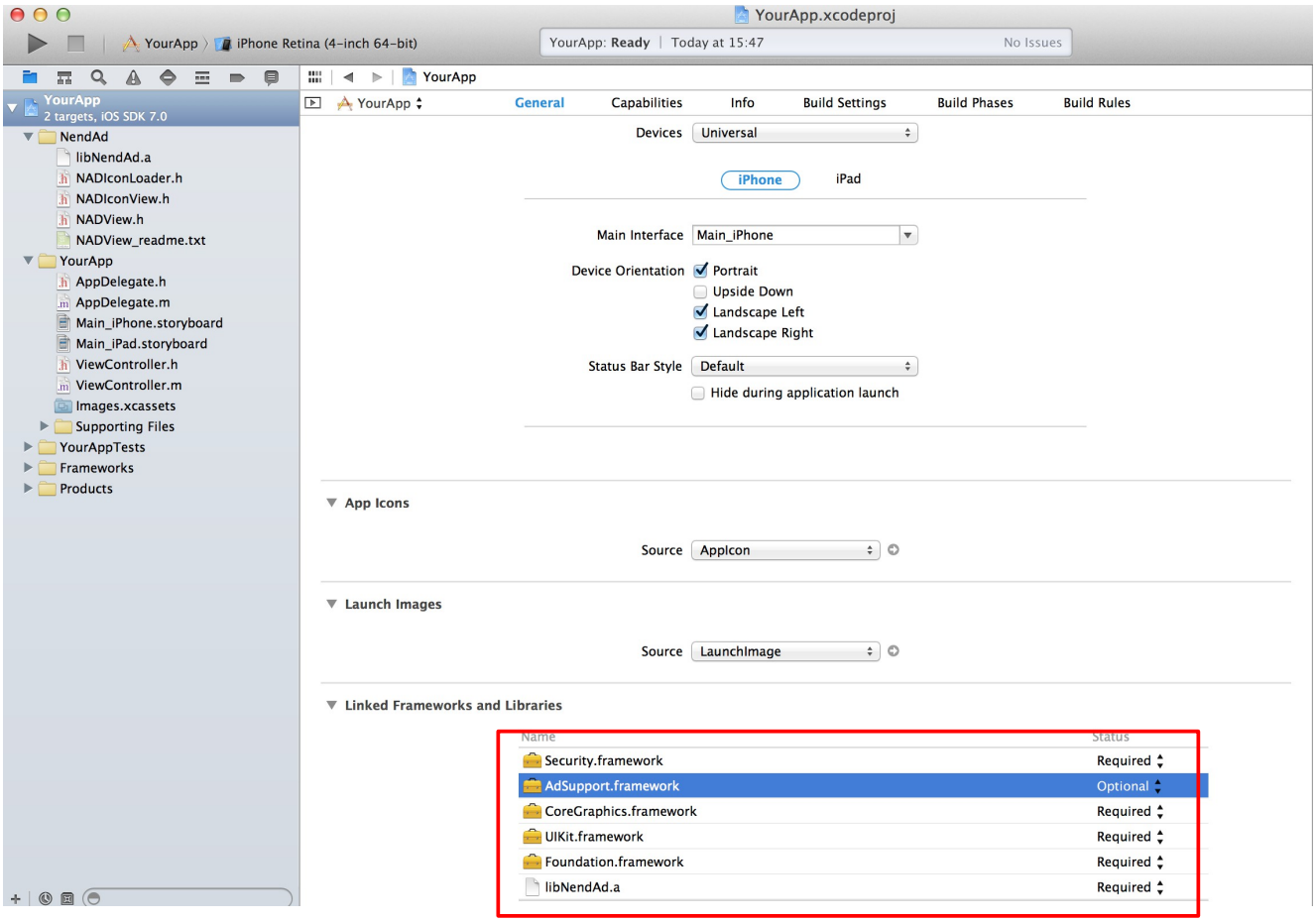


「＋」をクリックし
追加できるライブラリの一覧を開いて
AdSupprot.framework を選択して追加してください。
その後、**Security.framework** も同様に追加します。



Link Binary With Libraries に下記のリンクが正常に追加されていることを確認します。

- **AdSupport.framework**
- **Security.framework**
- **libNendAd.a**

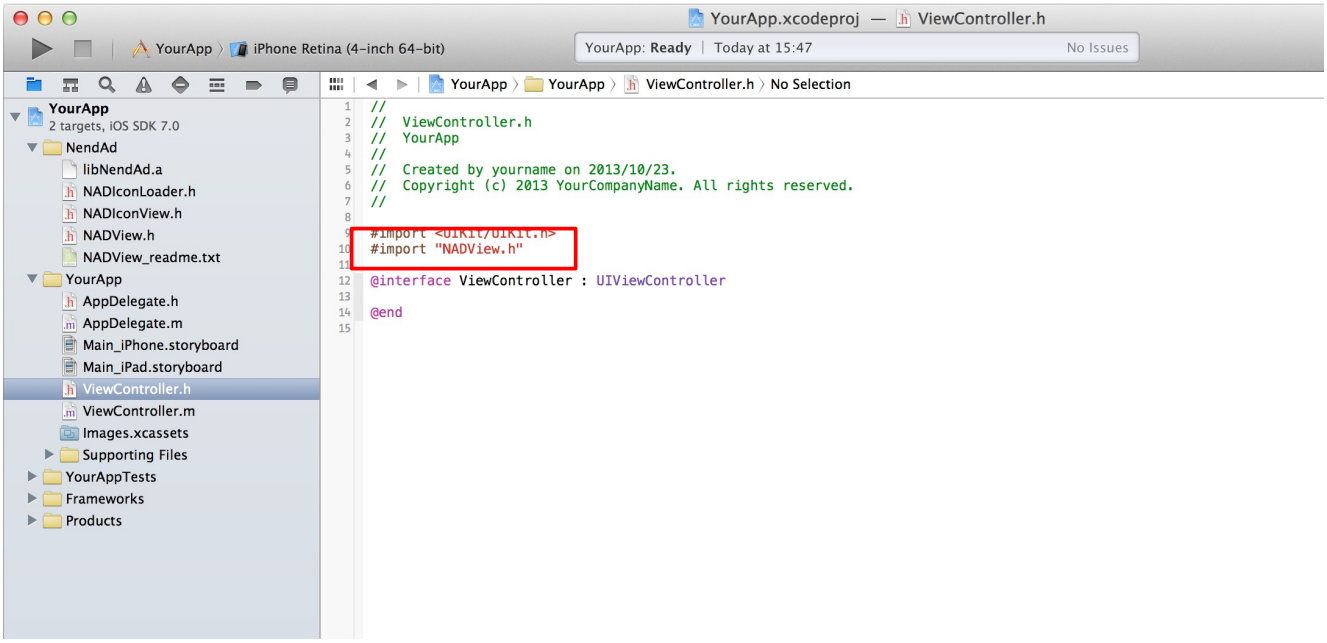


このとき、AdSupport.framework を Required → Optional に変更します。

※iOS6 未満を動作対象に含めている （Deployment Target の指定が iOS 6 未満）アプリケーションである場合は、必ず Optional に変更してください。

3．ビルド

任意のヘッダファイルに「**#import “NADView.h”**」を記述し、一旦ビルド (Clear&Build) します。エラーが発生せずにビルドが成功すれば、正常に SDK が追加できています。



※この時点でエラーが発生する場合は、Link Binary With Libraries の内容を再度確認してください。

4．広告ビューの設置

4－1．バナー型広告

4－1－1．広告サイズについて

SDK ver2.2.0 から以下の広告サイズが使用出来るようになりました。

320 × 50	iPhone, iPod Touch, iPad	従来のバナーサイズ
320 × 100	iPhone, iPod Touch, iPad	
300 × 100	iPhone, iPod Touch, iPad	
300 × 250	iPhone, iPod Touch, iPad	
728 × 90	iPad のみ	タブレット専用バナーサイズ

※広告サイズの指定は、nend 管理画面での広告枠作成時に行います。
　1 つの広告枠に対して、1 つの広告サイズが指定出来ます。

(1) 広告ビューサイズの指定
広告ビューを作成する際に「該当広告枠で指定したサイズ」を指定するようにしてください。
※アプリ側からのサイズ指定と異なる場合は、管理画面でのサイズ指定が優先されます。

(2) 端末のディスプレイサイズよりも大きい広告サイズを指定した場合
端末ディスプレイサイズよりも大きい広告サイズが指定されている場合は、
広告を表示しません。アプリ側へは受信エラーが通知されます。

上図のようにディスプレイに収まりきらない広告サイズの場合は表示されません。

４－１－２．バナー型広告の実装手順

ここでは最もシンプルな構造を例にして NADView の実装方法について説明します。
それ以外の詳しい実装についてはサンプルソースをご参照ください。

○ヘッダファイル

(1) delegate 準拠

広告ビューを保持するクラスのヘッダファイルで NADView.h をインポートして NADViewDelegate に準拠します。

例) YourAppViewController.h

```
#import <UIKit/UIKit.h>
#import "NADView.h"

@interface YourAppViewController : UIViewController <NADViewDelegate> {}

@property (nonatomic, retain) NADView * nadView;

@end
```

注意：複数画面で構成されるアプリケーションで、各 ViewController ごとにインスタンス生成するような実装方法の場合には、画面ごとに必ず後述の**（8）定期ロードの管理**を行ってください。

○実装ファイル

例) YourAppViewController.m（一部）　- NADView 生成からロードまで

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

    // (2) NADView の作成
    self.nadView = [[NADView alloc] initWithFrame:CGRectMake(0, 0, 320, 50)];

   // (3) ログ出力の指定
   [self.nadView setIsOutputLog:NO];
    // (4) set apiKey, spotId.
    [self.nadView setNendID:@"a6eca9dd074372c898dd1df549301f277c53f2b9"
                     spotID:@"3172"];
    [self.nadView setDelegate:self]; //(5)
    [self.nadView load]; //(6)

    [self.view addSubview:self.nadView]; // 最初から表示する場合
```

(2)NADView 生成

任意の位置、320x50 サイズの CGRect を引数にインスタンスを生成します。

```
self.nadView = [[NADView alloc] initWithFrame: CGRectMake(0,0, 320, 50)];
```

※上記は、addSubview する先の View の 0,0 の位置を左上に指定して広告を配置する例です。

(3) isOutputLog の設定

エラーログや警告ログを NSLog で出力するかどうかを指定します。

```
[self.nadView setIsOutputLog:NO];
```

初期値は YES です。出力したくない場合は NO をセットしてください。

(4) apiKey,spotID の設定

nend 管理画面で発行した、該当アプリの広告枠の apiKey、spotID をセットします。

```
[self.nadView setNendID:@"a6eca9dd074372c898dd1df549301f277c53f2b9"
                 spotID:@"3172"];
```

※この時点で広告設定情報へのアクセスを開始します。
広告枠ステータスが「承認中」の場合、広告のロードをしても受信エラーになります。nend 管理画面で該当アプリの広告枠ステータスが「アクティブ」であることを確認してください。

広告枠の申請は、承認済みになるとステータスが「アクティブ」に変わり、登録メールアドレス宛に広告枠承認のお知らせが届きます。メール到着の数時間後には広告配信ができる状態になります。

単に実装方法の確認を行う場合は一時的に**表示テスト用 ID**（→◆検証）を利用するなどしてください。

(5) デリゲートオブジェクトの設定

NADView が広告を受信開始した場合に指定されたデリゲートの nadViewDidFinishLoad メソッドを呼んで通知を行います。指定するデリゲートは「nadViewDidFinishLoad」メソッドを実装する「NADViewDelegate」プロトコルに準拠させたクラスを指定します。

```
[self.nadView setDelegate:self];
```

(6) 広告のロード

NADView の load メソッドで広告のロードを開始します。

```
[self.nadView load];
```

※リトライ間隔を標準値以外で指定したい場合には、
以下のメソッドを利用してロードを開始します。

```
// 例) 問い合わせエラー時には 60 分間隔で再問い合わせする
[self.nadView load:[NSDictionary dictionaryWithObjectsAndKeys: @"3600", @"retry", nil]];
```

(7) Delegate 通知

※ver2.0.0 より、delegate メソッドでの通知の際、一律メインスレッドで通知を行うよう変更されました。旧バージョンからの入れ替えの際、サブスレッドで通知を受ける前提での実装がある場合には十分にご注意ください。

【任意】ロード完了

広告のロードが完了すると(5)で指定したデリゲートの nadViewDidFinishLoad メソッドに通知されます。ロードが完了してから NADView を表示したい場合はここで行うことができます。

```
-(void)nadViewDidFinishLoad:(NADView *)adView {
    NSLog(@"delegate nadViewDidFinishLoad:");
}
```

【任意】広告バナークリック通知

表示されている広告バナーをクリックした際に通知されます。
ただし、このイベントをカウントしても、ネットワーク状況や環境により「実際に任意の広告表示ができた数(サーバ側でのクリック数としてのカウント)」とは、異なる場合がありますので、注意してください。

```
- (void)nadViewDidClickAd:(NADView *)adView {
      NSLog(@"delegate nadViewDidClickAd:");
}
```

【任意】広告受信通知

広告の受信に成功した場合通知されます。
広告を受信するたびに任意の処理を行いたい場合に利用します。

```
-(void)nadViewDidReceiveAd:(NADView *)adView {
      NSLog(@"delegate nadViewDidReceiveAd:");
}
```

【任意】広告受信エラー通知

広告の受信に失敗した場合に通知されます。
通信エラー、広告在庫がなくなった場合など、何らかの理由で広告を表示できない場合に通知します。エラー時に広告を非表示にするなどの処理が必要な場合に利用します。

```
-(void)nadViewDidFailToReceiveAd:(NADView *)adView {
      NSLog(@"delegate nadViewDidFailToLoad:");
}
```

ver2.2.0 より、受信エラー通知で一部のエラー内容を取得出来るようになりました。
NADView のプロパティとして、NSError オブジェクトが追加されました。
エラー内容によって処理を分ける必要がある場合は、次ページのように実装してください。

【任意】広告受信エラー通知

エラー内容によって処理を分けるサンプル

```
-(void)nadViewDidFailToReceiveAd:(NADView *)adView
{
    NSLog(@"delegate nadViewDidFailToLoad:");

    // エラーごとに分岐する
    NSError* error = adView.error;
    NSString* domain = error.domain;
    int errorCode = error.code;

    // isOutputLog = NO でも、domain を利用してアプリ側で任意出力が可能
    NSLog(@"log %d", adView.isOutputLog);
    NSLog(@"%@",[NSString stringWithFormat: @"code=%d, message=%@",
                                                    errorCode, domain]);

    switch (errorCode) {
        case NADVIEW_AD_SIZE_TOO_LARGE:
            // 広告サイズがディスプレイサイズよりも大きい
            break;
        case NADVIEW_INVALID_RESPONSE_TYPE:
            // 不明な広告ビュータイプ
            break;
        case NADVIEW_FAILED_AD_REQUEST:
            // 広告取得失敗
            break;
        case NADVIEW_FAILED_AD_DOWNLOAD:
            // 広告画像の取得失敗
            break;
        case NADVIEW_AD_SIZE_DIFFERENCES:
            // リクエストしたサイズと取得したサイズが異なる
            break;
        default:
            break;
    }
}
```

NADViewErrorCode (NADView.NSError.code) の内容

NADViewErrorCode	エラー内容
NADVIEW_FAILED_AD_REQUEST	広告取得失敗 (ネットワークエラー、サーバエラー、 在庫切れなど)
NADVIEW_AD_SIZE_TOO_LARGE	広告サイズがディスプレイサイズよりも大きい
NADVIEW_AD_SIZE_DIFFERENCES	リクエストしたサイズと取得したサイズが異なる※
NADVIEW_INVALID_RESPONSE_TYPE	不明な広告ビュータイプ※
NADVIEW_FAILED_AD_DOWNLOAD	広告画像の取得失敗

※基本的に発生することは稀です

(8) 定期ロードの管理

広告のロードを開始した後に画面内に表示しないケースがある場合には、
必ず以下の処理を行ってください。

これには、

・ 複数画面で構成された画面遷移が発生するアプリケーションにおいて
各画面で個別に広告のインスタンスを生成している場合

・ **複数広告の切り替え処理（広告スイッチング SDK の利用含む）を行う場合**

などが該当します。

定期ロードの中断

広告を画面内に表示しない、もしくは画面遷移等で View 自体が表示されない場合には
pause メッセージを送信、広告の定期ロードを中断します。

例）画面が隠れたら定期ロードを中断

```
- (void)viewWillDisappear:(BOOL)animated {
        [self.nadView pause];
}
```

定期ロードの再開

広告を再び画面内に表示、または画面遷移等で View 自体を表示する場合には
resume メッセージを送信、広告の定期ロードを再開します。

例）画面が表示されたら定期ロードを再開

```
- (void)viewWillAppear:(BOOL)animated {
        [self.nadView resume];
}
```

(9) リリース

dealloc 時には、**必ず delegate プロパティに nil をセットして**からリリースを行うようにしてください。

```
- (void) dealloc {
    [self.nadView setDelegate:nil]; // delegate に nil をセット
    self.nadView = nil;             // プロパティ経由で release、nil をセット

    // [super dealloc];  // MRC(非 ARC 時には必要)
}
```

【重要】

delegate プロパティに nil をセットしない場合、メモリの解放が適切に行われず、稀に予期しないクラッシュを引き起こす場合があります。 必ず delegate に nil をセットしてから release を行うようにしてください。

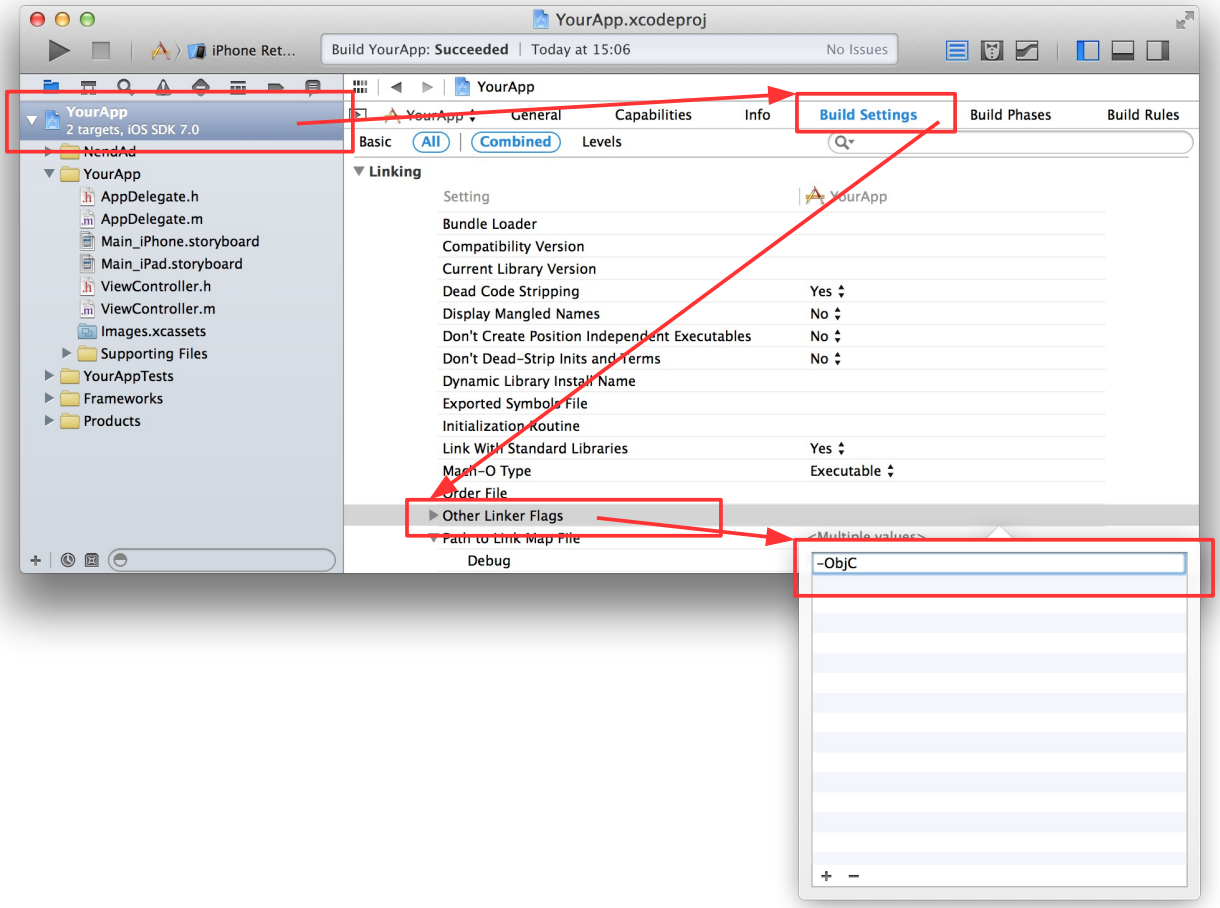
また、delegate に nil がセットされた際、自動的に広告受信ローテーションを中断(pause)します。これにより release 前の pause メッセージ送信が不要になります。

○Interface Builder を使用した実装手順
Interface Builder を使用してバナー広告の実装が可能です。

（※Deployment Target の設定が 5.1 未満ではご利用できません。）

(1)リンカフラグの追加

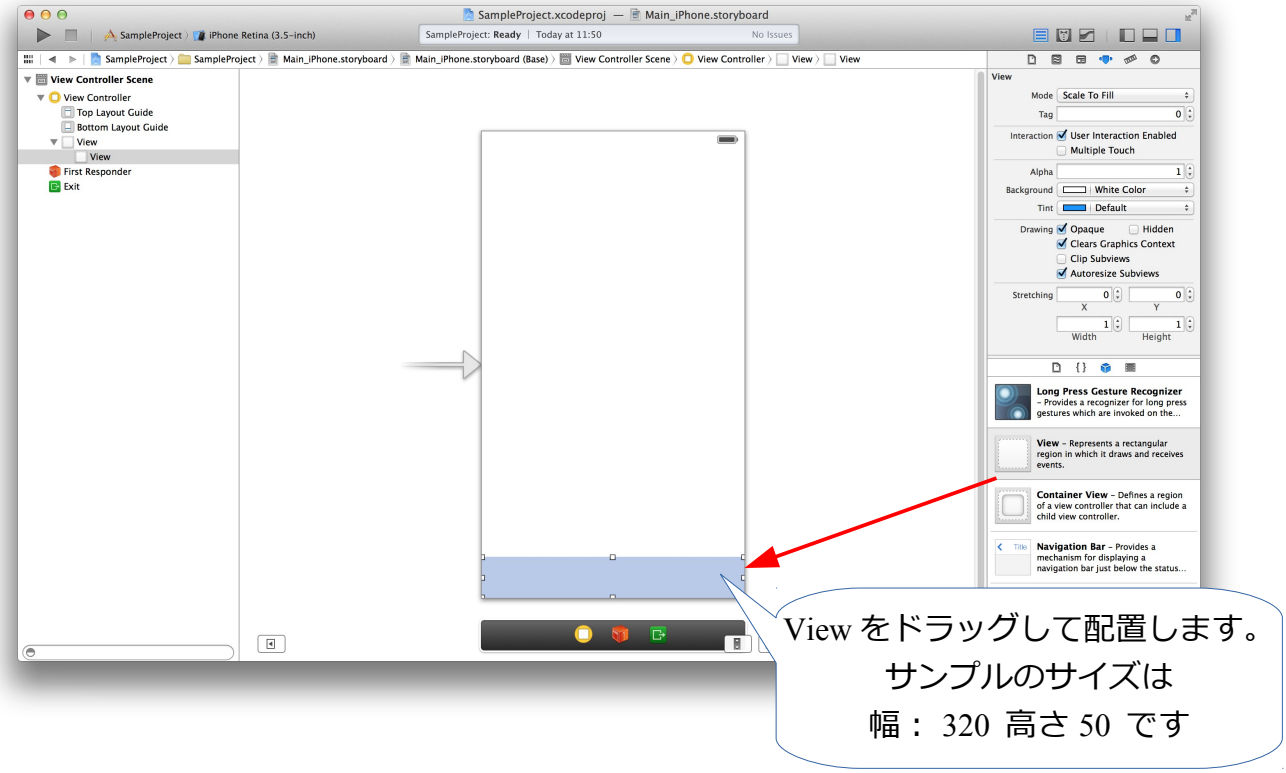
左側のプロジェクトナビゲーターから、プロジェクトをクリックして
TARGETS＞Build Setting＞Other Linker Flags 項目を開き、”-ObjC”を追加します。



(2)View の追加

広告を表示する View 上に View を追加します。

また、配置する際に View のサイズを広告サイズに合わせます。



(3)クラスの変更、プロパティの設定

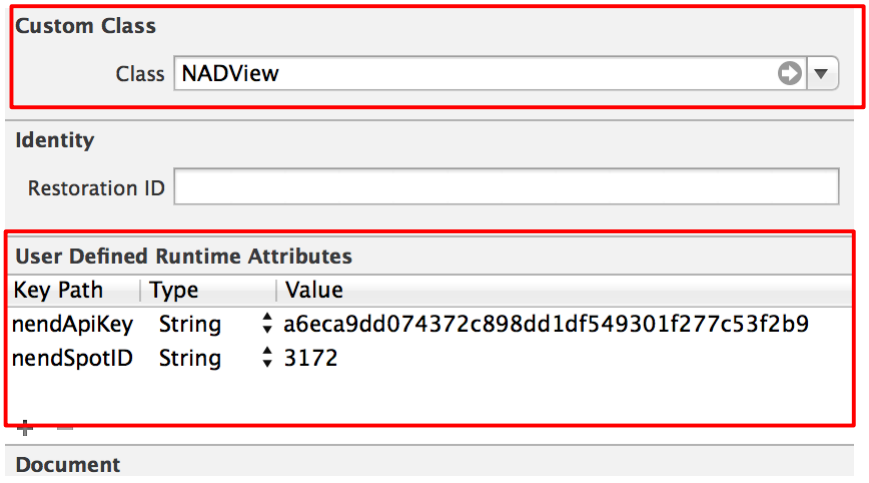
Utility area から Identity Inspector からクラスの変更とプロパティの変更を行います。

(1)**CustomClass** に **NADView** を指定

(2)**User Defined Runtime Attributes** に以下の値を追加する

・**nendApiKey** に**管理画面より発行された apiKey** を設定

・**nendSpotID** に**管理画面より発行された spotID** を設定



（4）User Defined Runtime Attributes で設定可能な項目

User Defined Runtime Attributes で設定可能な項目および、初期値は以下となります。

Key Path	Type	Value	必須 or 任意	初期値	説明
nendApiKey	String	発行された apiKey	必須	-	管理画面より発行された apiKey
nendSpotID	String	発行された SpotID	必須	-	管理画面より発行された SpotID
isOutputLog	Boolean	YES or NO	任意	NO	ログ出力可否

以上で設定完了となります。

4－1－3．NADViewの内容

○メソッド

- (void)setNendID:(NSString *)apiKey spotID:(NSString *)spotID;
広告枠のapiKeyとspotIDをセットします。

- (void)load;
ロードを開始します。

- (void)load:(NSDictionary *)parameter;
ロードを開始します。
接続エラーや広告設定受信エラーなどの場合にリトライする間隔を、NSDictionaryで任意指定出来ます。ただしkeyは「retry」、valueは30 - 3600の間で指定してください。範囲外指定された場合は標準で60秒が適用されます。標準で問題ない場合はparameterのないload; を利用してください。

- (void) pause;
広告の定期ロード中断を要求します

- (void) resume;
広告の定期ロード再開を要求します

○プロパティ

@property (nonatomic, assign) id <NADViewDelegate> delegate;
delegateオブジェクトの指定（任意）

@property (nonatomic) BOOL isOutputLog;
エラーログや警告ログを、NSLogとして出力するかどうかの指定（任意）

@property (nonatomic, assign) NSError error;
参照すると受信エラー時にその内容を動的に知ることが出来ます。

【廃止予定】
@property (nonatomic, assign) UIViewController *rootViewController;
モーダルビュー表示元のビューコントローラの指定(任意)

○**Delegate**

- (void) nadViewDidFinishLoad:(NADView *)adView;
広告ロードが初めて成功した際に通知されます。（任意）

- (void) nadViewDidReceiveAd:(NADView *)adView;
広告受信が成功した際に通知されます。（任意）

- (void) nadViewDidFailToReceiveAd:(NADView *)adView;
広告受信に失敗した際に通知されます。（任意）

- (void) nadViewDidClickAd:(NADView *)adView;
広告バナークリック時に通知されます。（任意）

4－2．アイコン型広告

4－2－1．アイコン型広告のサイズについて

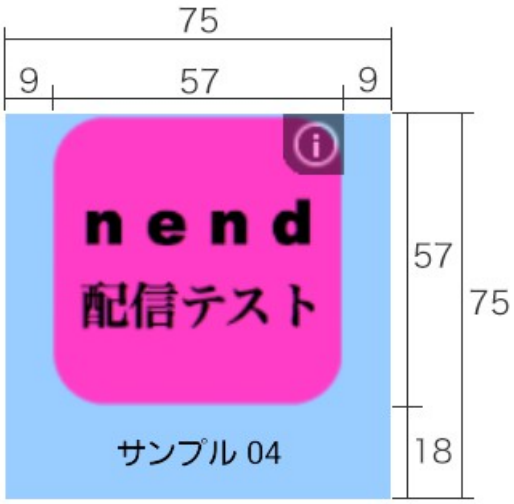
(1) アイコン型広告ビューのサイズ指定

アイコン型広告ビューは、サイズ設定を行わない場合、幅 75px、高さ 75px となります。

(2) アイコン型広告ビュー内の配置

アイコン型広告ビューは、アイコン画像とテキストを表示します。
テキストは、表示，非表示の選択、色の変更が可能です。
サンプルプログラムと同様の手順で実装した際の配置は以下になります。

・デフォルト（75×75）



・縦 100、横 50 と指定した場合

縦横で異なるサイズが指定された場合、

小さい方のサイズで縦横を揃えます。

(3) アイコン型広告サイズと各種設定について

アイコン型広告は、余白の有無・タイトル文字列の有無・サイズ指定の有無によって
下記の通りに画面上に表示されます。
レイアウトを決める上での参考にしてください。

例) アイコンサイズを５０×５０と指定した場合

<table>
<tr><td colspan="2" rowspan="2"></td><td colspan="2">タイトル文字列</td></tr>
<tr><td>有</td><td>無</td></tr>
<tr><td rowspan="2">余白</td><td>有</td><td> 

余白を含めた全体が５０×５０になります
アイコン画像のサイズはＳＤＫ側で最適化されます</td><td> 

余白を含めた全体が５０×５０になります
アイコン画像のサイズはＳＤＫ側で最適化されます</td></tr>
<tr><td>無</td><td> 

アイコン画像のサイズが５０×５０になります
タイトル文字列分の余白が下部に追加されます</td><td> 

アイコン画像のサイズが５０×５０になります</td></tr>
</table>

４－２－２．アイコン型広告の実装手順

ここでは最もシンプルな構造を例にして NADIconView の実装方法について説明します。
それ以外の詳しい実装についてはサンプルソースをご参照ください。

○ヘッダファイル

(1) delegate 準拠
広告ビューを保持するクラスのヘッダファイルで NADIconLoader.h をインポートして、NADIconLoaderDelegate に準拠します。

例) YourAppViewController.h

```
#import <UIKit/UIKit.h>
#import "NADIconLoader.h"

@interface YourAppViewController : UIViewController <NADIconLoaderDelegate> {
    NADIconLoader* iconLoader;
   NADIconView* iconView;
}

@end
```

注意：複数画面で構成されるアプリケーションで、各 ViewController ごとにインスタンス生成するような実装方法の場合には、画面ごとに必ず後述の**（10）定期ロードの管理**を行ってください。

○実装ファイル

例) YourAppViewController.m（一部）- NADIconLoader, NADIconView 生成からロードまで

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

    // NADIconView クラスの生成
    iconView = [[NADIconView alloc] initWithFrame:CGRectMake(0, 0, 75, 75)];
    // NADIconView の配置
    [self.view addSubview:iconView];

    // NADIconLoader クラスの生成
    iconLoader = [[NADIconLoader alloc] init];

    // ログ出力の指定
    [iconLoader setIsOutputLog:YES];

    // NADIconLoader へ NADIconView を追加
    [iconLoader addIconView:iconView];

    // API キーと SPOTID を設定
    [iconLoader setNendID:@"[管理画面より発行された apiKey]"
                   spotID:@"[管理画面より発行された spotID]"];

    // デリゲートオブジェクトの設定
    [iconLoader setDelegate:self];

    // 広告のロード
    [iconLoader load];
}
```

(2) NADIconView 生成

任意の位置、75x75 サイズの CGRect を引数にインスタンスを生成します。

```
iconView = [[NADIconView alloc] initWithFrame:CGRectMake(0, 0, 75, 75)];
```

※上記は、addSubview する先の View の 0,0 の位置を左上に指定して広告を配置する例です。

(3)NADIconLoader 生成

アイコン広告の情報を制御するクラスを生成します。

```
iconLoader = [[NADIconLoader alloc] init];
```

(4) isOutputLog の設定

エラーログや警告ログを NSLog で出力するかどうかを指定します。

```
[iconLoader setIsOutputLog:YES];
```

※初期値は YES です。出力したくない場合は NO をセットしてください。

(5) NADIconLoader に NADIconVIew の登録

NADIconLoader へ NADIconView を登録します。

```
[iconLoader addIconView:iconView];
```

※NADIconView は最大 6 つまで登録可能です。

(6) apiKey,spotID の設定

nend 管理画面で発行した、該当アプリの広告枠の apiKey、spotID をセットします。

```
[iconLoader setNendID:@"[管理画面より発行された apiKey]"
                spotID:@"[管理画面より発行された spotID]"];
```

※この時点で広告設定情報へのアクセスを開始します。
広告枠ステータスが「承認中」の場合、広告のロードをしても受信エラーになります。
nend 管理画面で該当アプリの広告枠ステータスが「アクティブ」であることを
確認してください。

広告枠の申請は、承認済みになるとステータスが「アクティブ」に変わり、登録メールアドレス宛に広告枠承認のお知らせが届きます。メール到着の数時間後には広告配信ができる状態になります。

単に実装方法の確認を行う場合は一時的に**表示テスト用 ID**（→◆検証）を利用するなどしてください。

(7) デリゲートオブジェクトの設定

NADIconLoader が広告を受信開始した場合に指定されたデリゲートの
nadIconLoaderDidFinishLoad メソッドを呼んで通知を行います。
指定するデリゲートは「nadIconLoaderDidFinishLoad」メソッドを実装する
NADIconViewDelegate」プロトコルに準拠させたクラスを指定します。

```
[iconLoader setDelegate:self];
```

(8) 広告のロード

NADView の load メソッドで広告のロードを開始します。

```
[iconLoader load];
```

(9) Delegate 通知

【任意】ロード完了

広告のロードが完了すると(7)で指定したデリゲートの nadIconLoaderDidFinishLoad メソッドに通知されます。ロードが完了してから NADIconView を表示したい場合はここで行うことができます。

```
-(void)nadIconLoaderDidFinishLoad:(NADIconLoader *)iconLoader{
    NSLog(@"delegate nadIconLoaderDidFinishLoad:");
}
```

【任意】広告バナークリック通知

表示されているアイコン広告をクリックした際に通知されます。
引数にクリックされた NADIconView が設定されます。
ただし、このイベントをカウントしても、ネットワーク状況や環境により「実際に任意の広告表示ができた数(サーバ側でのクリック数としてのカウント)」とは、異なる場合がありますので、注意してください。

```
-(void)nadIconLoaderDidClickAd:(NADIconLoader *)iconLoader
        nadIconView:(NADIconView *)nadIconView{
    NSLog(@"delegate nadIconLoaderDidClickAd:");
}
```

【任意】広告受信通知

広告の受信に成功した場合通知されます。引数に広告を受信した NADIconView が設定されます。広告を受信するたびに任意の処理を行いたい場合に利用します。

```
-(void)nadIconLoaderDidReceiveAd:(NADIconLoader *)iconLoader
        nadIconView:(NADIconView *)nadIconView{
    NSLog(@"delegate nadIconLoaderDidReceiveAd:");
}
```

【任意】広告受信エラー通知

広告の受信に失敗した場合に通知されます。
通信エラー、広告在庫がなくなった場合など、何らかの理由で広告を表示できない場合に通知します。エラー時に広告を非表示にするなどの処理が必要な場合に利用します。

```
-(void)nadIconLoaderDidFailToReceiveAd:(NADIconLoader *)iconLoader
        nadIconView:(NADIconView *)nadIconView{
    NSLog(@"delegate nadIconLoaderDidFailToReceiveAd:");
}
```

エラー内容によって処理を分ける必要がある場合は、次ページのように実装してください。

【任意】広告受信エラー通知

エラー内容によって処理を分けるサンプル

```
-(void)nadViewDidFailToReceiveAd:(NADView *)adView
{
    NSLog(@"delegate nadViewDidFailToLoad:");

    // エラーごとに分岐する
    NSError* error = adView.error;
    NSString* domain = error.domain;
    int errorCode = error.code;

    // isOutputLog = NO でも、domain を利用してアプリ側で任意出力が可能
    NSLog(@"log %d", adView.isOutputLog);
    NSLog(@"%@",[NSString stringWithFormat: @"code=%d, message=%@",
                                                errorCode, domain]);

    switch (errorCode) {
        case NADVIEW_AD_SIZE_TOO_LARGE:
            // 広告サイズがディスプレイサイズよりも大きい
            break;
        case NADVIEW_INVALID_RESPONSE_TYPE:
            // 不明な広告ビュータイプ
            break;
        case NADVIEW_FAILED_AD_REQUEST:
            // 広告取得失敗
            break;
        case NADVIEW_FAILED_AD_DOWNLOAD:
            // 広告画像の取得失敗
            break;
        case NADVIEW_AD_SIZE_DIFFERENCES:
            // リクエストしたサイズと取得したサイズが異なる
            break;
        default:
            break;
    }
}
```

NADViewErrorCode (NADView.NSError.code) の内容

NADViewErrorCode	エラー内容
NADVIEW_FAILED_AD_REQUEST	広告取得失敗 (ネットワークエラー、サーバエラー、在庫切れなど)
NADVIEW_AD_SIZE_TOO_LARGE	広告サイズがディスプレイサイズよりも大きい
NADVIEW_AD_SIZE_DIFFERENCES	リクエストしたサイズと取得したサイズが異なる※
NADVIEW_INVALID_RESPONSE_TYPE	不明な広告ビュータイプ※
NADVIEW_FAILED_AD_DOWNLOAD	広告画像の取得失敗

※基本的に発生することは稀です

(10) 定期ロードの管理

広告のロードを開始した後に画面内に表示しないケースがある場合には、
必ず以下の処理を行ってください。

これには、

・ 複数画面で構成された画面遷移が発生するアプリケーションにおいて
各画面で個別に広告のインスタンスを生成している場合

・ **複数広告の切り替え処理（広告スイッチング SDK の利用含む）を行う場合**

などが該当します。

定期ロードの中断
広告を画面内に表示しない、もしくは画面遷移等で View 自体が表示されない場合には
pause メッセージを送信、広告の定期ロードを中断します。

例）画面が隠れたら定期ロードを中断

```
- (void)viewWillDisappear:(BOOL)animated {
        [iconLoader pause];
}
```

定期ロードの再開
広告を再び画面内に表示、または画面遷移等で View 自体を表示する場合には
resume メッセージを送信、広告の定期ロードを再開します。

例）画面が表示されたら定期ロードを再開

```
- (void)viewWillAppear:(BOOL)animated {
        [iconLoader resume];
}
```

(11) リリース

dealloc 時には、**必ず delegate プロパティに nil をセットして**からリリースを行うようにしてください。

```
- (void) dealloc {
    [iconLoader setDelegate:nil]; // delegate に nil をセット
    iconLoader = nil;             // プロパティ経由で release、nil をセット

    // [super dealloc];  // MRC(非 ARC 時には必要)
}
```

【重要】

delegate プロパティに nil をセットしない場合、メモリの解放が適切に行われず、稀に予期しないクラッシュを引き起こす場合があります。必ず delegate に nil をセットしてから release を行うようにしてください。

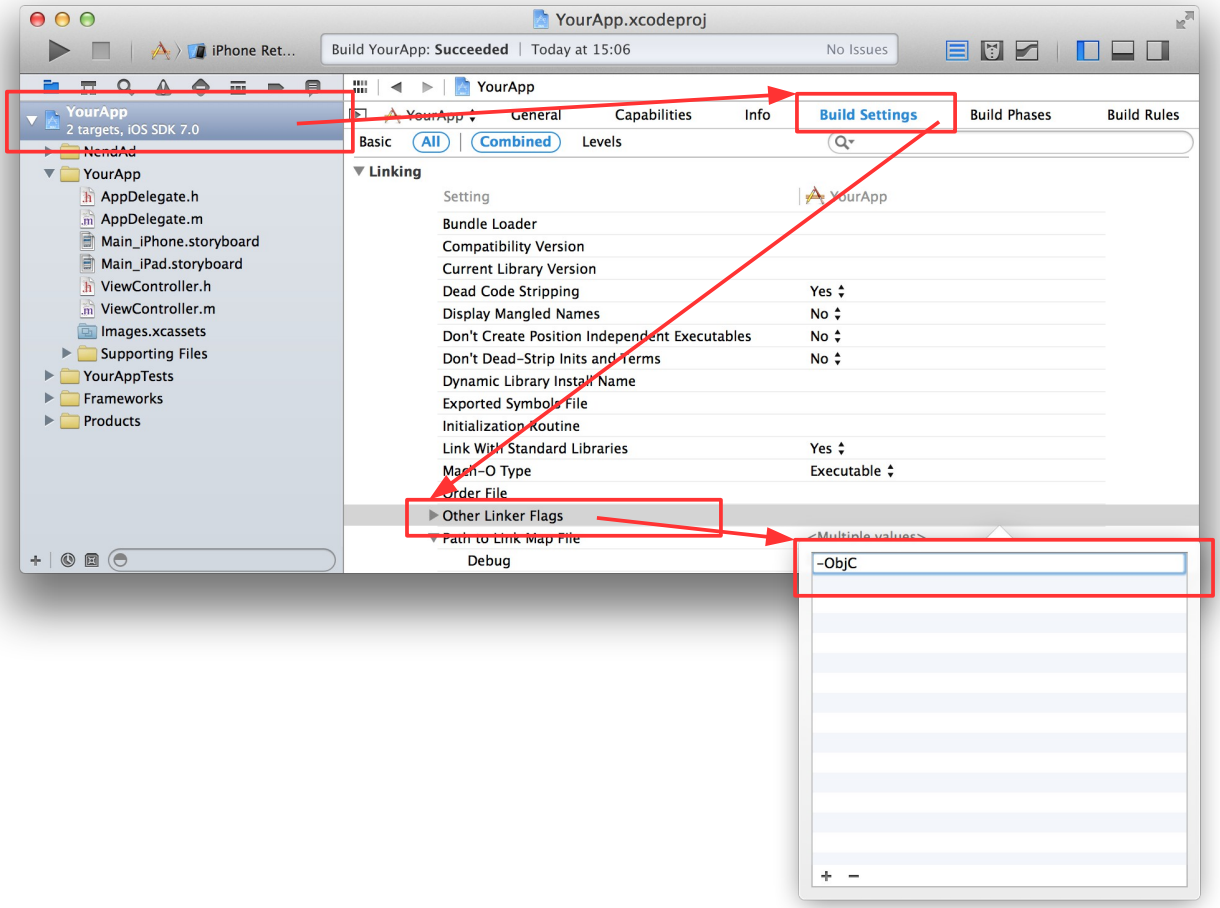
また、delegate に nil がセットされた際、自動的に広告受信ローテーションを中断(pause)します。これにより release 前の pause メッセージ送信が不要になります。

○Interface Builder を使用した実装手順
Interface Builder を使用してバナー広告の実装が可能です。

（※Deployment Target の設定が 5.1 未満ではご利用できません。）

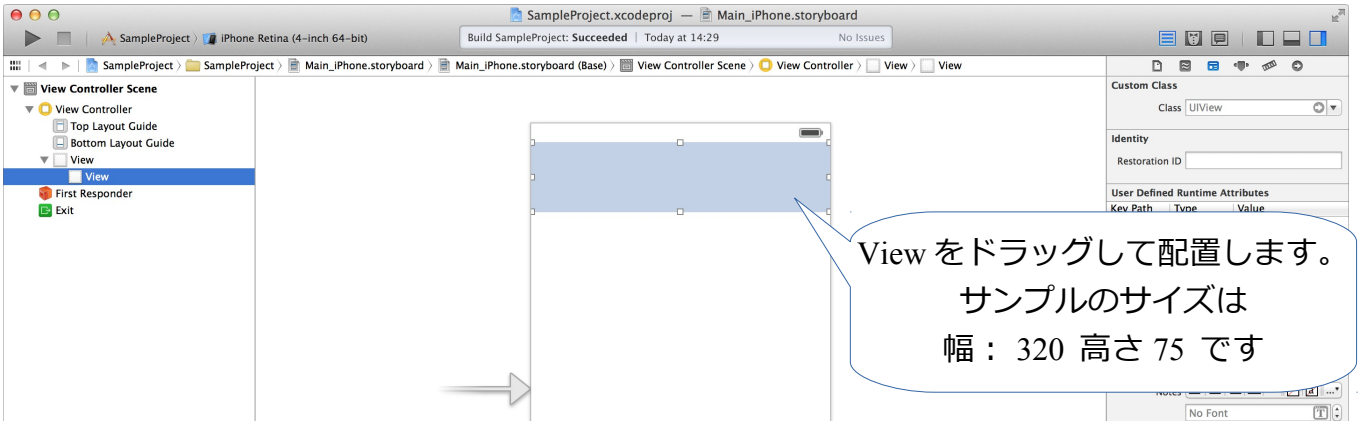
(1)リンカフラグの追加

左側のプロジェクトナビゲーターから、プロジェクトをクリックして
TARGETS＞Build Setting＞Other Linker Flags 項目を開き、”-ObjC”を追加します。



(2)View の追加

広告を表示する View 上に View を追加します。



(3)クラスの変更、プロパティの設定

Utility area - Identity Inspector からクラスの変更とプロパティの変更を行います。

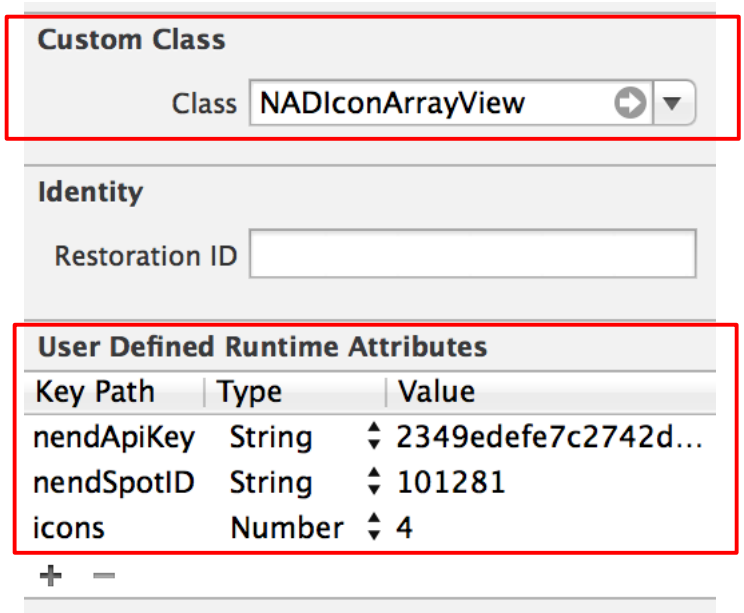
(1)**CustomClass** に **NADIconArrayView** を指定

(2)**User Defined Runtime Attributes** に以下の値を追加する

・**nendApiKey** に**管理画面より発行された apiKey** を設定

・**nendSpotID** に**管理画面より発行された spotID** を設定

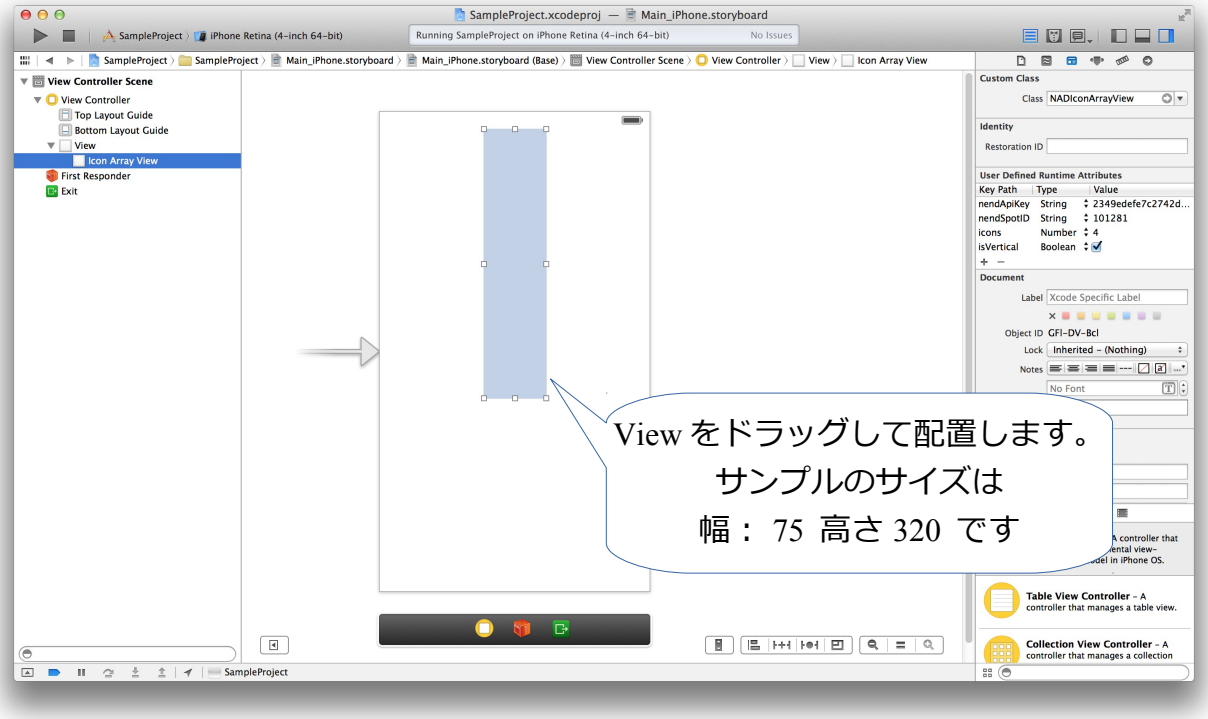
・**icons** に**表示するアイコン個数**を設定



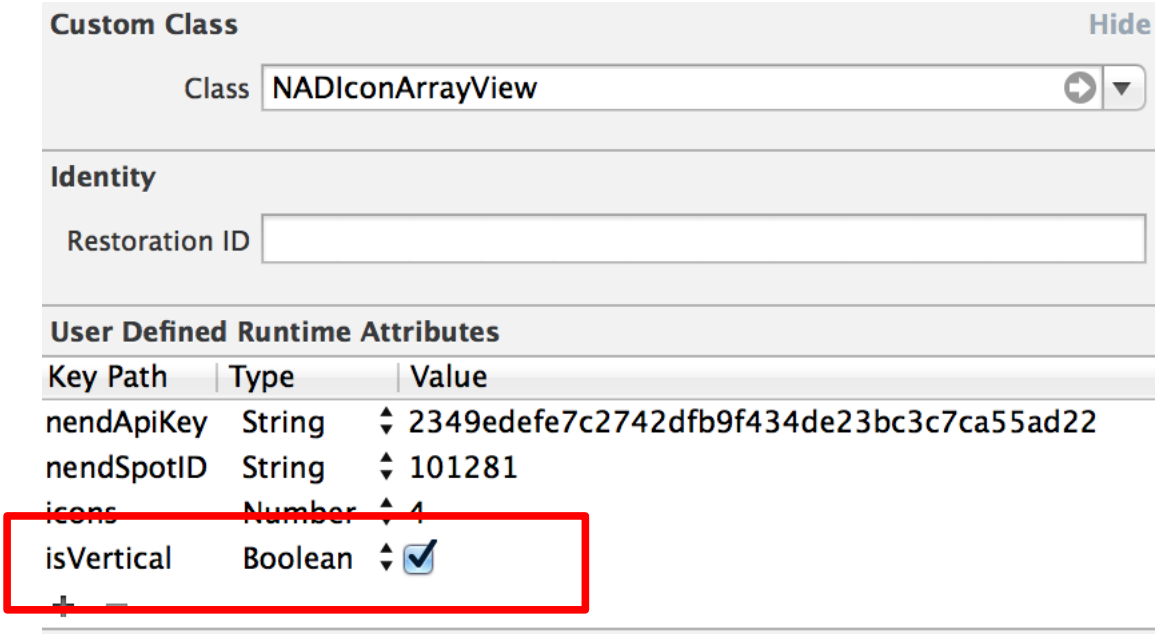
ビルドを行い、追加したViewにアイコンが表示できれば組込み完了です。

(4)アイコンを縦に並べる

縦に並べるための View を追加します。



User Defined Runtime Attributes に **isVertical** を追加し、チェックを入れます。



ビルドを行い、追加した View にアイコンが表示できれば組込み完了です。

（5）User Defined Runtime Attributes で設定可能な項目

User Defined Runtime Attributes で設定可能な項目および、初期値は以下となります。

Key Path	Type	Value	必須 or 任意	初期値	説明
nendApiKey	String	発行された apiKey	必須	-	管理画面より発行された apiKey
nendSpotID	String	発行された SpotID	必須	-	管理画面より発行された SpotID
isOutputLog	Boolean	YES or NO	任意	NO	ログ出力可否
textHidden	Boolean	YES or NO	任意	NO	タイトル表示可否
iconSpaceEnabled	Boolean	YES or NO	任意	YES	余白有無
isVertical	Boolean	YES or NO	任意	NO	表示方向(YES=縦，NO=横)
icons	Number	1 ～ 6	任意	1	アイコン表示数

以上で設定完了となります。

4 – 2 – 3．NADIconLoaderの内容

○メソッド

- (void)addIconView:(NADIconView *)iconView;
ローダーにアイコン広告のビューを登録します。

- (void)removeIconView:(NADIconView *)iconView;
ローダーからアイコン広告のビューを登録解除します。

- (void)setNendID:(NSString *)apiKey spotID:(NSString *)spotID;
広告枠の apiKey と spotID をセットします。

- (void)load;
ロードを開始します。

- (void) pause;
広告の定期ロード中断を要求します

- (void) resume;
広告の定期ロード再開を要求します

○プロパティ

@property (nonatomic, assign) id <NADIconLoaderDelegate> delegate;
delegate オブジェクトの指定（任意）

@property (nonatomic) BOOL isOutputLog;
エラーログや警告ログを、NSLog として出力するかどうかの指定（任意）

@property (nonatomic, assign) NSError error;
参照すると受信エラー時にその内容を動的に知ることが出来ます。

○Delegate

- (void) nadIconLoaderDidFinishLoad:(NADIconLoader *)iconLoader;
広告ロードが初めて成功した際に通知されます。（任意）

-(void)nadIconLoaderDidReceiveAd:(NADIconLoader *)iconLoader
nadIconView:(NADIconView*)nadIconView;
広告受信が成功した際に通知されます。（任意）

- (void)nadIconLoaderDidFailToReceiveAd:(NADIconLoader *)iconLoader
nadIconView:(NADIconView*)nadIconView;
広告受信に失敗した際に通知されます。（任意）

- (void)nadIconLoaderDidClickAd:(NADIconLoader *)iconLoader
nadIconView:(NADIconView*)nadIconView;
広告バナークリック時に通知されます。（任意）

4－2－4．NADIconView の内容

○メソッド
- (void)setTextColor:(UIColor *)setColor;
テキストの色を変更します。

○プロパティ
@property (nonatomic) BOOL textHidden;
テキストを表示するかどうかの指定

@property (nonatomic) BOOL iconSpaceEnabled;
余白部分を表示するかどうかの指定

4－2－5．NADIconArrayView の内容

○メソッド
- (void) pause;
広告の定期ロード中断を要求します

- (void) resume;
広告の定期ロード再開を要求します

- (void)setTextColor:(UIColor *)setColor;
テキストの色を変更します。

○プロパティ

@property (nonatomic) BOOL isOutputLog;
エラーログや警告ログを、NSLog として出力するかどうかの指定（任意）

@property (nonatomic) BOOL textHidden;
テキストを表示するかどうかの指定

@property (nonatomic) BOOL iconSpaceEnabled;
余白部分を表示するかどうかの指定

@property (nonatomic, assign) NSString* nendApiKey;
広告枠の apiKey の指定

@property (nonatomic, assign) NSString* nendSpotID;
広告枠の spotID の指定

@property (nonatomic) BOOL isVertical;
アイコンを縦に並べて表示するかどうかの指定

@property (nonatomic, copy) NSNumber* icons;
表示するアイコンの個数の指定

@property (nonatomic, readonly) NADIconLoader* iconLoader;
NADIconLoader の参照

◆検証

iOS アプリ向け表示テスト用の apiKey と spotID を設定していただくことで
対象アプリケーション用の広告枠のステータスが「承認中」(非アクティブ)である場合でも
広告表示の確認ができます。

iOS アプリ向け表示テスト用 ID

サイズ	apiKey	spotID
320 x 50	a6eca9dd074372c898dd1df549301f277c53f2b9	3172
320 x100	eb5ca11fa8e46315c2df1b8e283149049e8d235e	70996
300 x100	25eb32adddc4f7311c3ec7b28eac3b72bbca5656	70998
300 x250	88d88a288fdea5c01d17ea8e494168e834860fd6	70356
728 x 90	2e0b9e0b3f40d952e6000f1a8c4d455fffc4ca3a	70999
アイコン	2349edefe7c2742dfb9f434de23bc3c7ca55ad22	101281

※確認後は必ず各アプリケーション用広告枠の apiKey と spotID に書き換えてください。

※テスト用 ID のままアプリケーションをストアへ申請、配布された場合、広告効果は正しく集計されませんのでご注意ください。また、テスト用 ID のまま申請してしまった場合、nend 側での保障等はできかねますのでご了承ください。

広告受信エラーの最も簡単な再現方法はオフラインにすることです。現状、nendSDK 内でオンライン状況のチェックは行っておりませんので、オフライン時に何らかの処理を行う場合は、アプリケーション側で実装する必要があります。

◆よくある質問

詳しくはメディアパートナー様向けヘルプをご覧ください
https://www.nend.net/m/help/index/20

お問合せ先

nend - お問合せフォーム

https://www.nend.net/inquiries/form/

※お問い合わせ時には、メディア登録名、apikey/spotID、SDKバージョン番号、必要に応じてご利用中の開発環境や端末機種の詳細などを情報としてお寄せ頂けますとより回答がスムースになります。