fluct 広告配信 SDK for Android

導入仕様書

改版履歴

バージョン	日付	変更内容
1.0.0	2011/09/30	新規リリース
1.0.1	2011/10/17	不具合修正
1.0.2	2011/12/02	複数広告表示対応
1.1.0	2012/01/04	デモグラ機能追加
1.1.1	2012/02/16	不具合修正
1.1.2	2012/03/15	アップデート時の不具合修正
1.1.3	2012/04/02	F-05D/ ISW11F にて初回起動時にリセットが発生する不具合に対応
1.1.4	2012/07/03	不具合修正
1.1.5	2012/07/06	Unity や jsWaffle にて広告をタップした際にリセットが発生する不具合に対応
1.3.0	2012/07/18	広告表示高速化対応
1.3.1	2012/09/20	動的 API 追加対応 AdNetwork 向け対応
1.3.2	2012/09/27	不具合修正
2.0.0	2013/02/12	広告表示高速化対応
2.0.1	2013/02/13	AdNetwork 向け対応
2.0.2	2013/05/21	不具合修正
2.0.3	2013/09/12	広告更新の際に自動でスクロールしてしまう不具合に対応
2.1.0	2013/10/17	アニメーション対応追加
3.0.0	2013/10/25	インタースティシャル広告対応追加
3.0.1	2013/12/10	アプリ起動時に圏外の際に広告表示を行うと View サイズが拡大する 不具合に対応。
3.1.0	2014/06/01	メディア ID 英数字対応
3.2.0	2014/08/12	Google 広告 ID ("advertising ID") 対応。デモグラ機能の削除
3.2.1	2014/12/11	内部リソースの最適化
3.2.2	2015/10/22	内部コードの Android6 対応
4.0.0	2016/10/17	主開発環境を Android Studio に変更 ライブラリファイル形式を AAR に変更 ライブラリ提供方式を リモート maven リポジトリ に変更 FluctSDK クラスパス を変更 FluctView コールバックを追加 FluctInterstitial スペルミスの修正 コールバックを変更

目次

概要

このドキュメントは fluct 広告配信 SDK for Android (以下 FluctSDK) の 導入手順、使用ガイド、API 仕様、および 旧バージョンからの移行ガイド を記述します。

対象 OS

Android OS 2.3 (Gingerbread) 以降

開発環境

最新の Android Studio とその Android Studio が動作する環境となります。
JAR ファイル での提供がご希望の場合は、弊社担当営業までお問い合わせください。

使用追加モジュール

Google Play Services
https://developers.google.com/android/
com.google.android.gms:play-services-base を使用。

FluctSDK 追加手順

以下に FluctSDK を使用するための追加手順を説明します。

1. app/build.gradle に FluctSDK の maven リポジトリ url と dependencies に FluctSDK と Google Play Services (base) を追加します。

```
repositories {
    maven {
        // FluctSDK maven リポジトリ
        url 'https://raw.github.com/voyagegroup/FluctSDK-Android/master/m2/repository/'
    }
}

dependencies {
    …
    // FluctSDK
    compile 'jp.fluct:FluctSDK:+'
    // Google play services - base
    compile 'com.google.android.gms:play-services-base:+'
    …
}
```

※ この記述だと最新のバージョンを使用するようになっていますが、バージョン番号指定をしてもらうと、問題発生時にサポートしやすくなります。

2. app/src/main/AndroidManifest.xml に 必要な uses-permission と meta-data を追加します。
インタースティシャル広告を使用する場合は、 activity として FluctInterstitialActivity を追加します。

```
<!— FluctSDK で使用 -->
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>

<!-- Google Play Services -->
<meta-data android:name="com.google.android.gms.version"
         android:value="@integer/google_play_services_version"/>

<!— インタースティシャル を使用する場合、以下の Activity 定義を記述 -->
<activity
       android:name="jp.fluct.fluctsdk.FluctInterstitialActivity"
       android:configChanges="orientation|keyboardHidden|screenSize"
       android:theme="@android:style/Theme.Translucent.NoTitleBar"
       android:launchMode="singleTask">
</activity>
```

3. Android Studio の Build > Clean Project か Rebuild Project を実行すると、使用出来るようになります。

メディア ID

広告を表示する際には、弊社システムで 貴社アプリ広告枠 を識別するため発行される「メディア ID」が必要となります。
貴社アプリ広告枠 の「メディア ID」は弊社担当営業にお問い合わせください。

テスト用 メディア ID

0000000108 が バナー広告 および インタースティシャル広告 の「テスト用 メディア ID」となっています。
アプリの開発時およびテスト時には、必ず「テスト用 メディア ID」を使用してください。
公開前に実際の広告を表示する必要がある場合も、表示された広告をタップしないでください。

メディア ID の指定

以下の箇所で メディア ID を設定できます。

レイアウト XML にて定義

この方法は、バナー広告を レイアウト xml にて組み込む場合の実装方法となります。
バナー広告での推奨実装方法となります。
FluctView ノード属性値として FLUCT_MEDIA_ID を追加します。
(例)

```
<jp.fluct.fluctsdk.FluctView
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       FLUCT_MEDIA_ID="yourMediaId"/>
```

※ “yourMediaId”を貴社アプリ広告枠用のメディア ID に置き換えてください。

FluctView / FluctInterstitial コンストラクタ の引数に設定

FluctView, FluctInterstitial オブジェクト毎に使用する メディア ID を指定します。
インタースティシャル広告での推奨実装方法となります。

実装については、後の節か、サンプルコードを参照してください。

AndroidManifest.xml にて定義

アプリ内で使用される共通の 広告メディア ID となります。
この実装方法は **オプション** となります。

application ノード に meta-data ノードを追加します。

```
<meta-data android:name="FLUCT_MEDIA_ID" android:value="yourMediaId"/>
```

※ “yourMediaId”を貴社アプリ広告枠用のメディア ID に置き換えてください。

使用優先順としては コンストラクタ引数 > レイアウト定義 > meta-data 定義 となります。

バナー広告を実装

バナー広告表示は、レイアウト XML にて行うか、 java コードにて実装します。
広告表示は、固定の dp 値にて表示されます。

FluctView を レイアウト XML にて実装

レイアウト XML のバナー広告表示を行いたい任意の箇所に、 FluctView クラスノードを追加してください。
高さは自動的に調整されますので、 layout_height の値は wrap_content を指定してください。
固定値で指定する場合、最低でも 幅 320dp、 高さ 50dp の表示領域は確保するようにしてください。

FLUCT_MEDIA_ID 属性で メディア ID の指定を行います。
この方法がバナー広告表示実装での推奨方法となります。
（例）

```
<jp.fluct.fluctsdk.FluctView
       android:layout_width="match_parent"
       android:layout_height="wrap_content"
       FLUCT_MEDIA_ID="yourMediaId"/>
```

※ “yourMediaId”を貴社アプリ広告枠用のメディア ID に置き換えてください。

FLUCT_MEDIA_ID 属性を指定しない場合、AndroidManifest.xml に定義してある metadata の メディア ID を使用します。

FluctView を コード にて実装

Activity, Fragment などの適切な場所に、処理を記述します。

AndroidManifest.xml に定義してあるメディア ID を使用して FluctView を作成する場合は、以下のようになります。
高さは自動的に調整されますので、 height の値は ViewGroup.LayoutParams.WRAP_CONTENT を指定してください。
固定値で指定する場合、最低でも 幅 320dp、 高さ 50dp の表示領域は確保するようにしてください。
(例)

```
RelativeLayout layout = (RelativeLayout)inflater.inflate(R.layout.fragment_activity, container,
false);

// FluctView を作成
FluctView fluctView = new FluctView(getContext(), “yourMediaId”);

// レイアウトパラメータ を作成
RelativeLayout.LayoutParams params = new
RelativeLayout.LayoutParams(ViewGroup.LayoutParams.MATCH_PARENT,
ViewGroup.LayoutParams.WRAP_CONTENT);
params.addRule(RelativeLayout.ALIGN_PARENT_TOP);

// レイアウトに追加
layout.addView(fluctView, params);
```

※ “yourMediaId”を貴社アプリ広告枠用のメディア ID に置き換えてください。

AndroidManifest.xml で メディア ID を指定していない、または、個別でメディア ID を指定する場合は、以下のようになります。
(例)

```
// FluctView を作成
FluctView fluctView = new FluctView(getContext(),"yourMediaId");
```

※ “yourMediaId”を貴社アプリ広告枠用のメディア ID に置き換えてください。

FluctView コールバック設定

設定する FluctView に setCallback() にて設定できます。
以下のイベントがコールバックされます。

1. 表示 onDisplayDone()
2. タップ onTap()
3. エラー onError()

(例)

```
FluctView banner = (FluctView) findViewById(R.id.banner);
banner.setCallback(new FluctView.Callback() {
    @Override
    public void onDisplayDone(boolean displayed) {
        if (!displayed) {
            // TODO: 表示率に基づき表示されなかった
        } else {
            // TODO: 表示された
        }
    }

    @Override
    public void onTap() {
        // TODO: タップされた
    }

    @Override
    public void onError(FluctView.FluctViewError fluctViewError) {
        if (FluctView.ErrorType.InternalError == fluctViewError.getType()) {
            // TODO: 内部エラー
        } else if (FluctView.ErrorType.InvalidRequest == fluctViewError.getType()) {
            // TODO: 無効なリクエスト
        } else if (FluctView.ErrorType.NetworkError == fluctViewError.getType()) {
            // TODO: ネットワーク エラー
        }
    }
});
```

インタースティシャル広告を実装

インタースティシャル広告は、コードにてのみ実装をする事ができます。
広告表示は、固定の dp 値にて表示されます。
表示後、Back ボタン、広告のクローズボタン、画面の回転にて表示が閉じられます。

Activity, Fragment などの適切な箇所で、 処理を記述します。

FluctInterstitial を作成する時に、メディア ID を指定する場合は、以下のようになります。
この方法がインタースティシャル広告表示実装での推奨方法となります。
（例）

```
// FluctInterstitial を作成（メディア ID を指定）
mInterstitial = new FluctInterstitial(getActivity(),"yourMediaId");
```

※ “yourMediaId”を貴社アプリ広告枠用のメディア ID に置き換えてください。

FLUCT_MEDIA_ID 属性を指定しない場合、AndroidManifest.xml に定義してある metadata の メディア ID を使用します。

インタースティシャル広告を表示する場合、以下のようになります。
(例)

```
// インタースティシャル広告を表示
mInterstitial.showInterstitialAd();
```

インタースティシャル広告を閉じる場合、以下のようになります。
(例)

```
// インタースティシャル広告を閉じる
mInterstitial.dismissInterstitialAd();
```

FluctInterstitial を破棄状態/終了化する場合、以下のようになります。
(例)

```
// FluctInterstitial を終了化
mInterstitial.destroy();
mInterstitial = null;
```

FluctInterstitial コールバック設定

設定する FluctInterstitial に setCallback() にて設定できます。
以下のイベントがコールバックされます。

1. 表示 onDisplayDone()
2. タップ onTap()
3. 閉じられた onClose()
4. エラー onError()

(例)

```
mInterstitial = new FluctInterstitial(this, "yourMediaId");
mInterstitial.setCallback(new FluctInterstitial.Callback() {
    @Override
    public void onDisplayDone(boolean displayed) {
        if (!displayed) {
            // TODO: 表示率に基づき表示されなかった
        } else {
            // TODO: 表示された
        }
    }

    @Override
    public void onTap() {
        // TODO: タップされた
    }

    @Override
    public void onClose() {
        // TODO: 閉じられた
    }

    @Override
    public void onError(FluctInterstitial.FluctInterstitialError fluctInterstitialError) {
        if (FluctInterstitial.ErrorType.InternalError == fluctInterstitialError.getType()) {
            // TODO: 内部エラー
        } else if (FluctInterstitial.ErrorType.InvalidRequest == fluctInterstitialError.getType()) {
            // TODO: 無効なリクエスト
        } else if (FluctInterstitial.ErrorType.NetworkError == fluctInterstitialError.getType()) {
            // TODO: ネットワークエラー
        }
    }
});
```

※ “yourMediaId”を貴社アプリ広告枠用のメディア ID に置き換えてください。

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 バージョン番号	OS バージョン 4.1 の場合、「4」になります。
2	アプリ Package Id	本 SDK を組み込んだ アプリ の パッケージ ID になります。
3	使用 FluctSDK バージョン	組み込んだ FluctSDK のバージョン文字列となります。
4	Google 広告追跡制限フラグ	アプリ Google 設定 の 広告 > 広告のパーソナライズをオプトアウトする になります。
5	Google 広告 ID (Google Advertising ID)	アプリ Google 設定 の 広告 > 広告 ID になります。

※ Google 広告 ID が使用される条件は、 Android OS が 2.3 以上、 Google Play バージョン 4.0 以上、 Google Play サービス が有効、 広告追跡制限をしていない場合となります。

※ 取得情報の取り扱いについては、 弊社の プライバシーポリシー (http://corp.fluct.jp/privacy/) を参照してください。

クラス リファレンス

FluctSDK で提供しているクラスのリファレンスとなります。

FluctView

jp.fluct.fluctsdk.FluctView

バナー広告を処理するためのクラスとなります。
レイアウト XML もしくは java ソースコード にて実装を行います。

Public constructors

FluctView

```
FluctView(Context context)
```

メディア ID は AndroidManifest.xml で meta-data 定義された メディア ID を使用します。

Parameters	
context	Context: 現在の Activity コンテキスト

FluctView (メディア ID 指定)

```
FluctView(Context context, String mediaId)
```

Parameters	
context	Context: 現在の Activity コンテキスト
mediaId	String: 弊社から発行された メディア ID

メディア ID に null が指定された場合、AndroidManifest.xml で meta-data 定義された メディア ID を使用します。

Public methods

---

destroy

```
void destroy()
```

FluctView オブジェクト で使用されているリソースを解放し、破棄できる状態にします。

---

getSDKVersion

```
String getSDKVersion()
```

Returns	
String	SDK バージョン文字列

FluctSDK のバージョン文字列を取得します。

---

prepareConfig

```
void FluctView.prepareConfig(Context context)
```

Parameters	
context	Context: 現在の Activity コンテキスト

広告表示をするための設定情報取得のみを行います。
AndroidManifest.xml の meta-data 指定されている メディア ID にて取得処理を行います。
表示までの処理を短縮させる事ができます。

---

prepareConfig (メディア ID 指定)

```
void FluctView.prepareConfig(Context context, String mediaId)
```

Parameters	
context	Context: 現在の Activity コンテキスト
mediaId	String: 弊社から発行された メディア ID

広告表示をするための設定情報の取得のみを行います。
指定された メディア ID にて取得処理を行います。
表示までの処理を短縮させる事ができます。

---

setCallback

```
void setCallback(FluctView.Callback callback);
```

Parameters	
callback	FluctView.Callback: FluctView.Callback オブジェクトを設定、もしくは、 interface FluctView.Callback を implements したクラスを設定します。

該当の状況が発生した場合、コールバックが呼び出されます。

Nested classes

interface FluctView.Callback

コールバッククラスです。
コールバックは、メインスレッド上にて呼び出されます。

Public methods

onDisplayDone

```
void onDisplayDone(boolean displayed)
```

Parameters	
displayed	boolean: 広告表示状態を表します。 true : 広告が表示されました。 false : 表示率に基づき表示されませんでした。

広告表示処理が完了すると呼び出されます。

onTap

```
void onTap()
```

広告がタップされた時に呼び出されます。

onError

```
void onError(FluctView.FluctViewError error)
```

Parameters	
error	FluctView.FluctViewError: エラー情報が設定されます。

エラーが発生すると呼び出されます。

FluctView.ErrorType

エラー種別の列挙型クラスです。

Values	
InternalError	内部的な問題が発生しました。
InvalidRequest	メディア ID が有効な値ではありません。
NetworkError	ネットワーク接続に問題が発生しました。

FluctView.FluctViewError

エラー情報クラスです。
種別と詳細情報が設定されます。

public methods

getType

```
FluctView.FluctViewError getType()
```

エラー種別を取得します。

Returns	
FluctView.ErrorType	エラー種別

---

getDescription

```
String getDescription()
```

エラー詳細情報文字列を取得します。

Returns	
String	エラー詳細情報文字列

FluctInterstitial

jp.fluct.fluctsdk.FluctInterstitial

インタースティシャル広告を処理するためのクラスとなります。
java ソースコード にて実装を行います。

Public constructors

FluctInterstitial

```
FluctInterstitial(Context context)
```

Parameters	
context	Context: 現在の Activity コンテキスト

FluctInterstitial (メディア ID 指定)

```
FluctInterstitial(Context context, String mediaId)
```

Parameters	
context	Context: 現在の Activity コンテキスト
mediaId	String: 弊社から発行された メディア ID

Public methods

destroy

```
void destroy()
```

FluctInterstitial オブジェクト内で使用されているリソースを解放し、破棄できる状態にします。

getSDKVersion

```
String getSDKVersion()
```

FluctSDK のバージョン文字列を取得します。

Returns	
String	SDK バージョン文字列

showInterstitialAd

```
void showInterstitialAd()
```

インタースティシャル広告を表示します。

showInterstitialAd (枠色指定)

```
void showInterstitialAd(int frameColor)
```

Parameters	
frameColor	int: 枠色を Color の int 値にて指定します。

表示枠の色を指定して、インタースティシャル広告を表示します。
色のアルファ値は無効となります。

dismissInterstitialAd

```
void dismissInterstitialAd()
```

インタースティシャル広告表示を閉じます。

setCallback

```
void setCallback(FluctInterstitial.Callback callback)
```

Parameters	
callback	FluctInterstitial.Callback: FluctInterstitial.Callback オブジェクトを設定、もしくは、 interface FluctInterstitial.Callback を implements したクラスを設定します。

該当の状況が発生した場合、コールバックが呼び出されます。

Nested classes

interface FluctInterstitial.Callback

FluctInterstitial のコールバッククラスです。
コールバックは、メインスレッド上にて呼び出されます。

Public methods

onDisplayDone

```
void onDisplayDone(boolean displayed)
```

広告表示処理が完了すると呼び出されます。

Parameters	
displayed	boolean: 表示状態を表します。 true : 広告が表示されました。 false : 表示率に基づき表示されませんでした。

onTap

```
void onTap()
```

広告がユーザ操作によりタップされると呼び出されます。

onClose

```
void onClose()
```

広告がユーザ操作により閉じると呼び出されます。

onError

```
void onError(FluctInterstitial.FluctInterstitialError error)
```

Parameters	
error	エラー情報が設定された FluctInterstitialError オブジェクト

エラーが発生すると呼び出されます。

FluctInterstitial.ErrorType

エラー種別の列挙型クラスです。

Values	
InternalError	内部的な問題が発生しました。
InvalidRequest	メディア ID が有効な値ではありません。
NetworkError	ネットワーク接続に問題が発生しました。

FluctInterstitial.FluctInterstitialError

エラー情報クラスです。
種別と詳細情報が設定されます。

public methods

getType

```
FluctInterstitial.ErrorType getType()
```

Returns	
FluctInterstitial.ErrorType	エラー種別

getDescription

```
String getDescription()
```

Returns	
String	エラーの詳細情報

Deprecated メソッド

今後のバージョンではなくなる予定のメソッドとなります。
同等のメソッドに切り替えをお願いします。

FluctInterstitial

Public methods

---

showIntersitialAd

```
void showInterstitialAd()
```

インタースティシャル広告を表示します。

---

showIntersitialAd

```
void showIntersitialAd(int frameColor)
```

Parameters	
frameColor	int: 枠の色を Color の int 値にて指定します。

インタースティシャル広告を表示します。

---

dissmissIntersitialAd

```
void dismissIntersitialAd()
```

表示中のインタースティシャル広告を閉じます。

---

setFluctInterstitialCallback

```
void setFluctInterstitialCallback(FluctInterstitialCallback callback)
```

Parameters	
callback	FluctInterstitialCallback: FluctInterstitialCallback オブジェクトを設定

インタースティシャル広告の状態を通知するための Callback を設定します。

Nested classes

---

interface FluctInterstitialCallback

Public methods

---

onReceiveAdInfo

```
void onReceiveAdInfo(int status)
```

Parameters

0:STATUS_DISPLAY_DONE	広告表示がされた際に通知されます
1:STATUS_AD_TAP	広告がタップされた際に通知されます。 なお、広告がタップされると、インタースティシャル広告は閉じられます
2:STATUS_AD_CLOSE	広告が閉じられた際に通知されます。
3:STATUS_AD_RATE_CANCEL	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:STATUS_NETWORK_ERROR	広告表示時にネットワーク接続がない際に通知されます。
5:STATUS_MEDIA_ID_ERROR	メディア ID が設定されていない際に通知されます。
6:STATUS_NONE_DATA	広告がメディア ID に設定されていない際に通知されます。
7:STATUS_SIZE_ERROR	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:STATUS_AD_INFO_ERROR	広告情報が取得できなかった際に通知されます。 設定しているメディア ID を再度ご確認ください。
100:STATUS_ANOTHER_ERROR	上記以外のエラーの際に通知されます。 また、広告表示中に、同一 FluctInterstitial オブジェクトから広告表示要求された際にも通知されます。

該当の状況が発生した場合、コールバックが呼び出されます。

旧バージョンからの移行について

v4.0.0 からライブラリファイル形式の変更になり、また、クラスパスも変更になったため、差し替えて更新する事ができなくなりました。
そのため、古い SDK を削除して、 新しい SDK を追加する手順になります。

Android Studio で 旧 FluctSDK を使用中

旧バージョン 削除

追加方法によって削除する方法が違います。

● FluctSDK の JAR ファイルを置いて使用していた場合
1. app/libs 等に現在使用している FluctSDK の JAR ファイルを削除します。

● サブモジュール追加にて JAR ファイルを追加して使用していた場合
1. File > Project Structure を選択します。
2. 左ペインの Modules にある FluctSDK のサブモジュールを選択します。
3. 上にある「−」を押下でサブモジュールを削除します。

build.gradle 修正

1. FluctSDK を直接参照している記述がある場合、その記述を削除します。

2. dependencies の Google play services 依存記述を変更します。
compile 'com.google.android.gms:play-services:+'
を
compile 'com.google.android.gms:play-services-base:+'
に変更します。

新バージョン 追加

前にある「FluctSDK 追加手順」に沿って追加します。

クラスパス 修正

java ソース、レイアウト XML、AndroidManifest.xml で指定してあるクラスパスを変更します。

1. FluctView
jp.co.voyagegroup.android.fluct.jar.FluctView
を
jp.fluct.fluctsdk.FluctView
に変更します。

2. FluctInterstitial
jp.co.voyagegroup.android.fluct.jar.FluctInterstitial
を
jp.fluct.fluctsdk.FluctInterstitial
に変更します。

3. FluctInterstitialActivity
jp.co.voyagegroup.android.fluct.jar.FluctInterstitialActivity
を
jp.fluct.fluctsdk.FluctInterstitialActivity
に変更します。

開発環境が Eclipse でそのまま使用したい

基本的に Android Studio への移行をお勧めしますが、

どうしても Eclipse でのビルドを行いたい場合、JAR ファイルを個別にて提供しますので、弊社担当営業までご連絡ください。

JAR ファイルを差し替え

現在使用している SDK の JAR ファイル を 新しい SDK の JAR ファイル に差し替えます。

ライブラリ参照 修正

プロジェクトプロパティ > Java Build Path > Libraries で、旧 SDK を削除し、新 SDK を追加します。

クラスパス 修正

java ソース、レイアウト XML、AndroidManifest.xml で指定してあるクラスパスを変更します。

1. FluctView
   jp.co.voyagegroup.android.fluct.jar.FluctView
   を
   jp.fluct.fluctsdk.FluctView
   に変更します。

2. FluctInterstitial
   jp.co.voyagegroup.android.fluct.jar.FluctInterstitial
   を
   jp.fluct.fluctsdk.FluctInterstitial
   に変更します。

3. FluctInterstitialActivity
   jp.co.voyagegroup.android.fluct.jar.FluctInterstitialActivity
   を
   jp.fluct.fluctsdk.FluctInterstitialActivity
   に変更します。