2015.10.22 v3.2.2

fluct 広告配信 SDK 導入仕様書
for Android

改版履歴

Ver	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

■ 対象 OS

Android 2.3 以上の AndroidSDK がインストールされ、正常に動作する必要があります。

■ 前提条件

Java SE Development Kit のバージョン 5 以上がインストールされている必要があります。

■ パッケージ内容

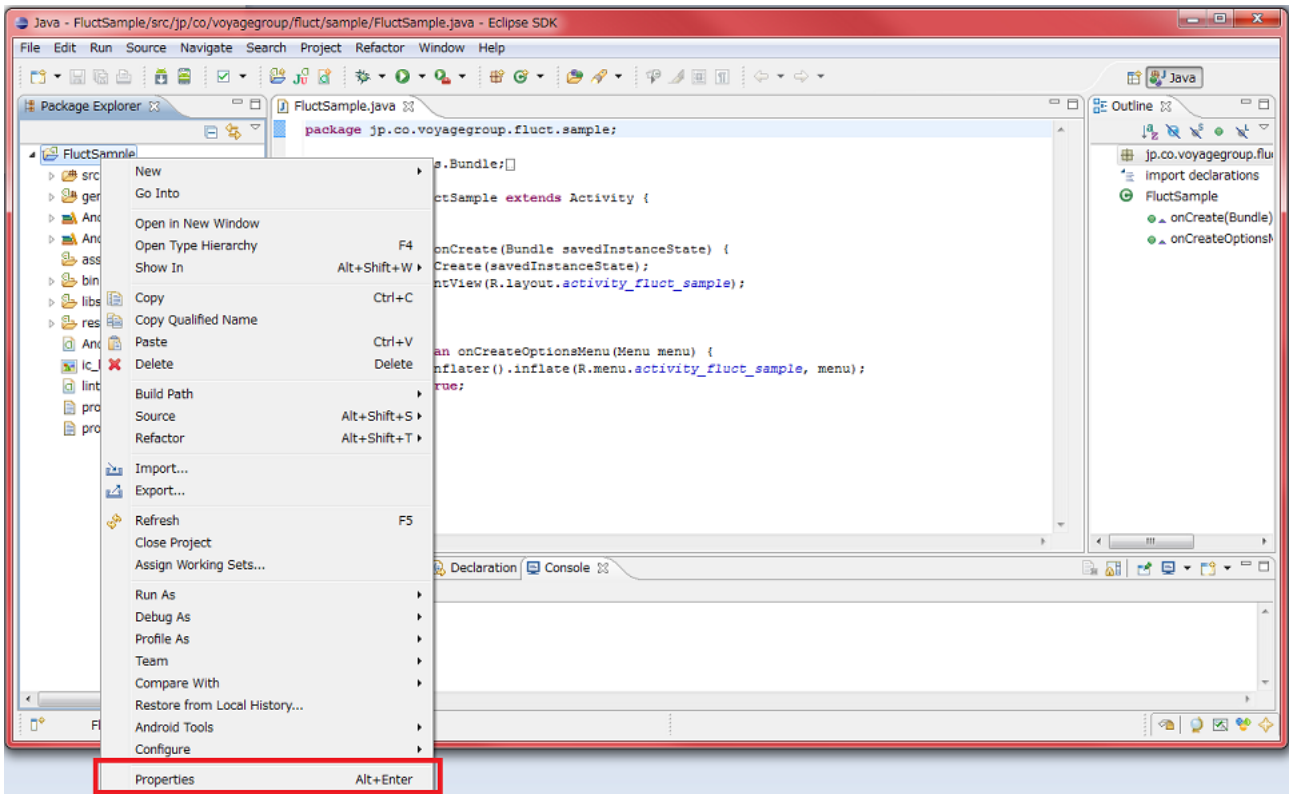
libs/		SDK ディレクトリ
	FluctSDK.jar	提供 jar ファイル
SampleApplication/		サンプルアプリディレクトリ
	FluctSample	広告表示サンプルアプリ

1. FluctSDK.jar ファイル追加手順

(1) FluctSDK.jar を libs フォルダ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) FlcutSDK.jar を利用するように Eclipse を設定する。
Android SDK r17 以降はアプリフォルダ内の libs フォルダを作成することで、本手順は不要となります。

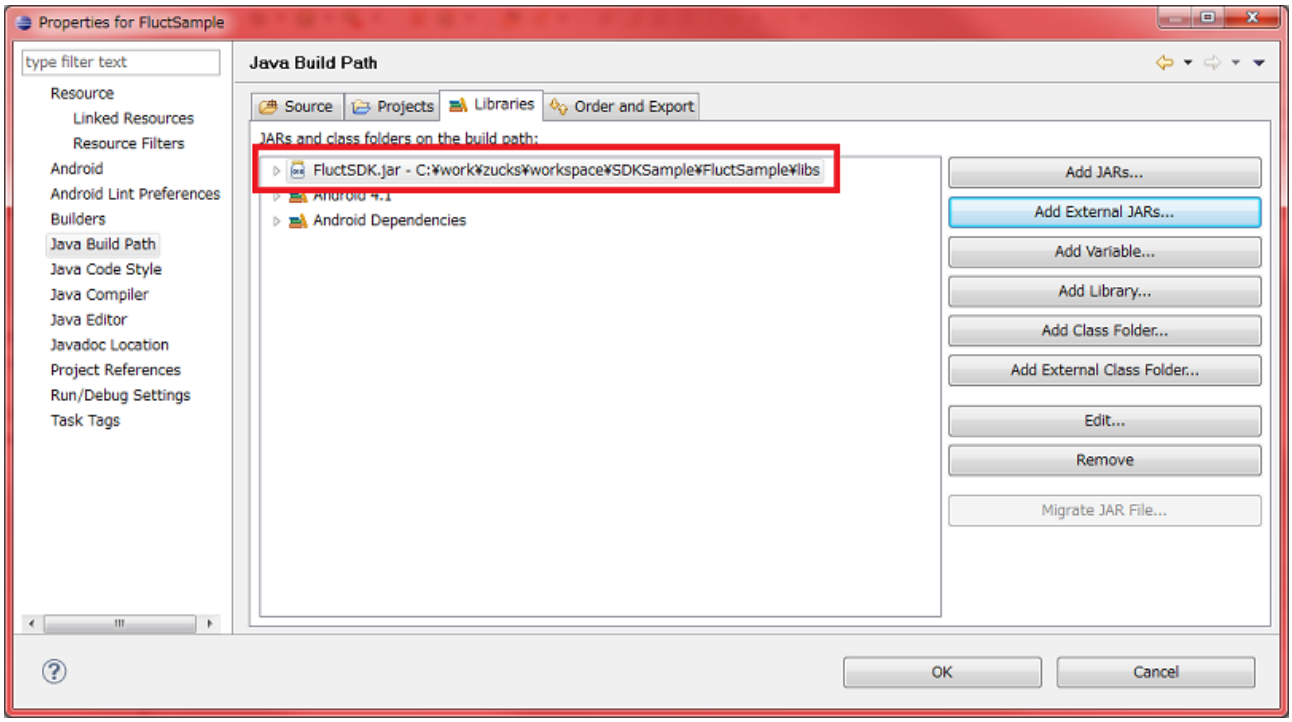
プロジェクトで右クリックをし、ポップアップするメニューより、「プロパティ」を選択します。



プロパティ選択画面

プロパティ画面にて「JAVA のビルド・パス」を選択し、タブの「ライブラリー」を選択します。

「外部 JAR の追加」を押下し、出てきたダイアログから、配置した FluctSDK.jar を選択します。



JAR のビルド・パス設定画面

※JAR ファイルを置くフォルダ名は**「libs」**としてください。環境によっては正しく JAR ファイルが読み込めない場合がございます。

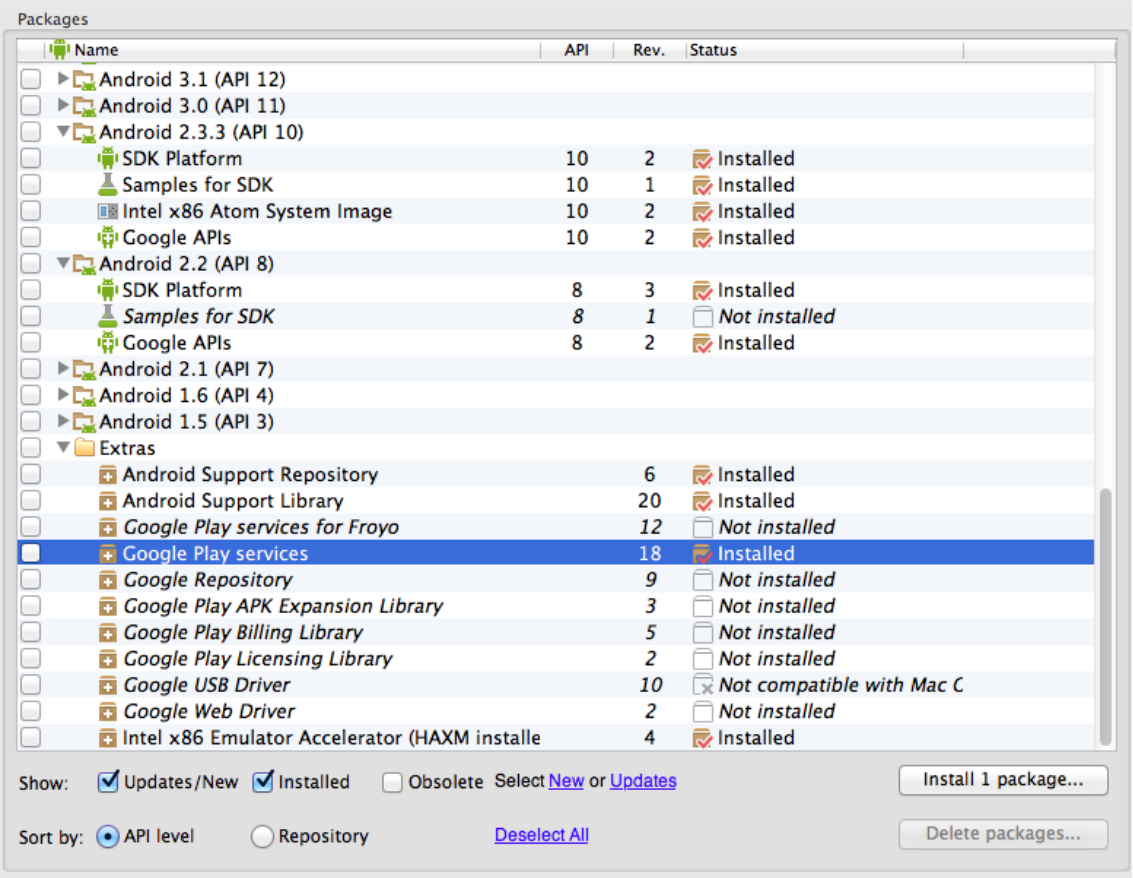
2. Google Play services を Android プロジェクトに組み込む

※インストール手順は Google によって予告なく変更される場合があります。最新情報は https://developers.google.com/android/guides/setup を参照ください。

2-1. Google Play services をセットアップする

FluctSDK を Android プロジェクトに組み込んで開発を進めるには、Google Play services をプロジェクトにセットアップする必要があります。

Eclipse あるいは Android Studio 画面内で SDK Manager を起動します。
Extras カテゴリ内の Google Play services にチェックを入れ、インストールを開始します。Google Play services は Android SDK 内の extras/google ディレクトリにセットアップされます。

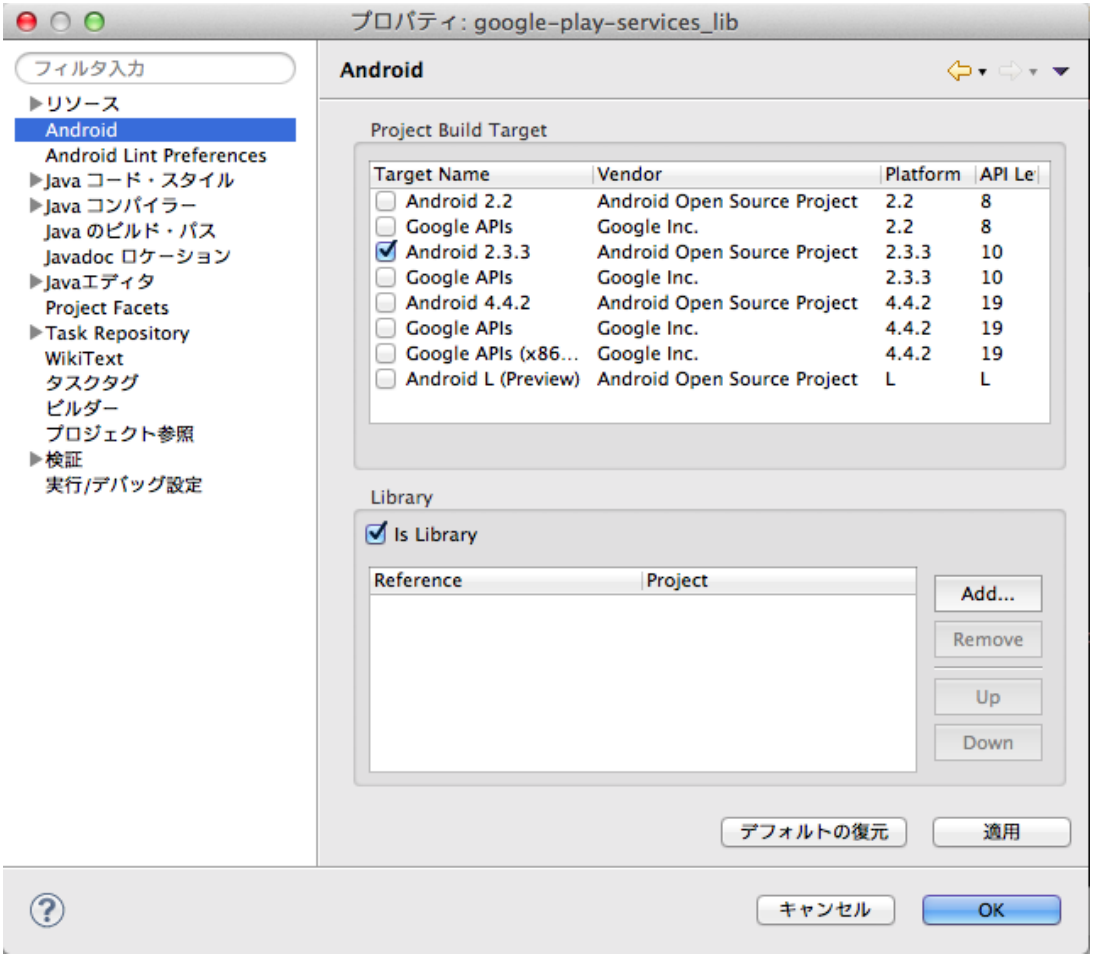


2-2. Eclipse の場合

Google Play services を以下の手順でライブラリプロジェクトとしてワークスペースにインポートします。

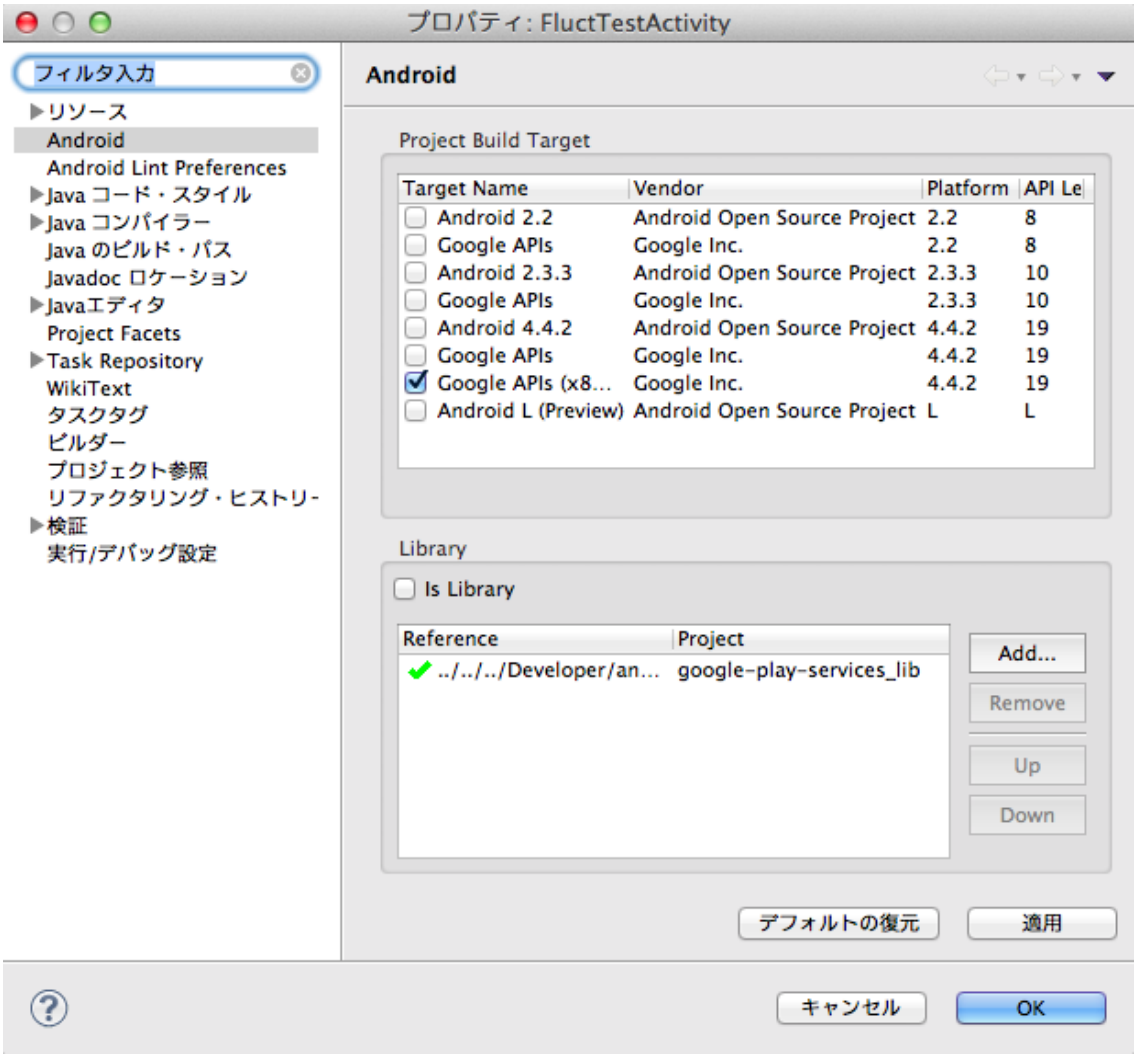
1. 「ファイル」→インポートをクリック
2. 「Existing Android code into Workspace」を選択して次へ

3. Root Directory 欄に先ほどセットアップした Google Play services のフォルダを選択します。
（Android SDK のディレクトリ/extras/google/google-play-services/libproject/google-play-services_lib）
正しくプロジェクトを選択できたらダイアログを終了します。
4. インポートした google-play-services_lib プロジェクトのプロパティを開き、Android プロパティグループ画面で is Library にチェックが入っていることを確認します。



Google Play services のインポートが完了したら、以下の手順で開発中の Android プロジェクトと関連付けを行い、Android プロジェクトから Google Play services を利用可能にします。

1. google-play-services_lib プロジェクトと Android プロジェクトが同じワークスペースにあることを確認します。
2. Android プロジェクトのプロパティを開き、Android プロジェクトグループ画面で追加ボタンをクリックし、google-play-services_lib プロジェクトを選択します。



3. 適用ボタンで変更を適用し、OK ボタンをクリックしてプロパティダイアログを閉じます。
4. ダイアログが閉じられると、Eclipse が自動的にプロジェクトをビルドし、Google Play services ライブラリが利用可能になります。自動でビルドされない場合は手動でプロジェクトをビルドします。

Google Play services をセットアップし、Android プロジェクトに関連付けを行った後、Android プロジェクトの manifest ファイルを開き、次のタグを<application>エレメントの子に追加します。

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

Eclipse の場合、インポートは以上です。これで Google Play services の機能が Android プロジェクト内で使えるようになります。

2-3. Android Studio の場合

Google Play services を以下の手順でセットアップします。

1. Android プロジェクトディレクトリ内の build.gradle ファイルを開きます。
2. dependencies グループに、最新バージョンの play-services を新しいビルドルールとして追加します。

```
apply plugin: 'com.android.application'
...
dependencies {
    compile 'com.google.android.gms:play-services:+'
}
```

3. 変更を保存し、ツールバーの Sync Project with Gradle Files  をクリックします

Android Studio の場合のセットアップは以上です。これで Google Play services の機能が Android プロジェクト内で使えるようになります。

3. 広告表示手順

貴社アプリ内の AndroidManifest.xml/Layout ファイルへの追加手順を記載致します。
広告を表示する際には、弊社システムでアプリを識別するための「メディア ID」が必要となります。
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。

(1) AndroidManifest.xml (meta-data)の設定

<meta-data>タグを利用して広告を表示するためのメディア ID を設定します。
<meta-data>タグは<application>タグの直後、または</application>タグの直前に記述してください。

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

“yourFluctMediaId”を貴社アプリ用のメディア ID に置き換えてください。

(2) AndroidManifest.xml (user-permisson)の設定

<user-permisson>タグは</manigest>タグの直前に記述してください。
なお、貴社アプリにて下記2つのアクセス権限が既に記述されている場合は記述不要となります。

```
        ...
    </application>
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
</manifest>
```

(3) レイアウトファイルへの記述

広告表示を行いたい任意のレイアウトファイルに、
「jp.co.voyagegroup.android.fluct.jar.FluctView」タグを記述してください。

```
    <jp.co.voyagegroup.android.fluct.jar.FluctView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
    />
```

サンプルアプリケーションについて

SampleApplication フォルダに、SDK 組み込み済みのサンプルアプリ FluctSample が置いてあります。
あわせてご参考して頂きますようお願い致します。

サンプルアプリ実行画面

何かご不明な点などございましたら、弊社担当までお問い合わせください。

さまざまな広告表示方法について

1. 1つの View に複数広告表示する場合の手順

複数広告を表示するための追加手順を記述致します。

複数広告を表示したい任意のレイアウトファイルに、
「jp.co.voyagegroup.android.fluct.jar.FluctView」タグを記述してください。
タグ内の「FLUCT_MEDIA_ID」は AndroidManifest.xml と異なるメディア ID を記述します。

```
<!-- 本ドキュメントの 2.広告表示手順内の処理 -->
<jp.co.voyagegroup.android.fluct.jar.FluctView
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      />

<!-- 貴社アプリ処理 -->

<!—複数広告表示手順処理 -->
<jp.co.voyagegroup.android.fluct.jar.FluctView
      android:layout_width="fill_parent"
      android:layout_height="wrap_content"
      FLUCT_MEDIA_ID="anotherFluctMediaId"
      />
```

“anotherFluctMediaId”を貴社アプリ用のメディア ID に置き換えてください。

2. 動的に広告を表示する方法

レイアウトファイルを使用せずに、動的に広告を表示する場合は、FluctView の下記 API を使用して表示処理を追加してください。

インスタンス生成 API(メディア ID 指定あり)

FluctView FluctView (Context context, String fluctMediaId)

FluctView のインスタンスを生成します。

Parameters

context : Context を設定してください。

fluctMediaId : 貴社アプリ用のメディア ID を設定してください。

インスタンス生成 API(メディア ID 指定なし)

FluctView FluctView(Context context)

FluctView のインスタンスを生成します。AndroidManifest に設定されているメディア ID を使用します。

Parameters

context : Context を設定してください。

広告表示/非表示制御 API

void setVisibility(int visibility)

FluctView の可視状態を変更します。

Parameters

visibility : View.VISIBLE, View.INVISIBLE, View.GONE を設定してください。

詳細は下記 Developer サイトを参照してください。

http://developer.android.com/intl/ja/reference/android/view/View.html#setVisibility(int)

インスタンス破棄 API

void destroy()

FluctView のインスタンスを破棄します。

広告設定事前読み込み API(メディア ID 指定あり)

広告表示までに Splash 画面等が存在する場合は、初回広告表示が早くなります。

void prepareConfig(Context context, String fluctMediaId)

Parameters

context : Context を設定してください。

fluctMediaId : 貴社アプリ用のメディア ID を設定してください。

広告設定事前読み込み API(メディア ID 指定なし)

広告表示までに Splash 画面等が存在する場合は、初回広告表示が早くなります。

void prepareConfig(Context context)

Parameters

context : Context を設定してください。

コーディングイメージ(インスタンス生成/広告表示)

```
FluctView mFluctView;

// FluctViewの生成.
mFluctView = new FluctView(getApplicationcontext(), “0000000108”);
//広告をVISIBLE状態に設定.
mFluctView.setVisibility(View.VISIBLE);
```

コーディングイメージ(広告非表示)

```
mFluctView.setVisibility(View.INVISIBLE);
```

コーディングイメージ(インスタンス破棄)

```
mFluctView.destroy();
```

コーディングイメージ(広告設定事前読み込み)

```
FluctView.prepareConfig(getApplicationContext());
```

3. インタースティシャル広告を表示する方法

(1) AndroidManifest.xml の設定

FluctInterstitialActivity を記述してください
なお、android:theme に関しては貴社アプリの実装に沿った形で記述してください。

```
<activity
    android:name="jp.co.voyagegroup.android.fluct.jar.FluctInterstitialActivity"
    android:configChanges="orientation|keyboardHidden|screenSize"
    android:theme="@android:style/Theme.Translucent.NoTitleBar"
    android:launchMode="singleTask"
    />
```

(2)インタースティシャル API

インスタンス生成 API(メディア ID 指定あり)

FluctInterstitial FluctInterstitial (Context context, String fluctMediaId)

FluctInterstitial のインスタンスを生成します。

Parameters

context : Context を設定してください。

fluctMediaId : 貴社アプリ用のメディア ID を設定してください。

インスタンス生成 API(メディア ID 指定なし)

FluctInterstitial FluctInterstitial (Context context)

FluctInterstitial のインスタンスを生成します。

AndroidManifest に設定されているメディア ID を使用します。

Parameters

context : Context を設定してください。

インタースティシャル広告表示 API

void showIntersitialAd ()

インタースティシャル広告を表示します。

表示状態は、FluctInterstitialCallback で通知されます。

FluctInterstitialCallback の詳細は、【FluctInterstitialCallback】を参照してください。

FluctInterstitialCallback 登録 API

void setFluctInterstitialCallback (FluctInterstitialCallback callback)

インタースティシャル広告の表示状態を通知するための Callback を設定します。
本 API が設定されていない場合は、表示状態は通知されません。

Parameters

callback : FluctInterstitialCallback を設定してください。

FluctInterstitialCallback

void onReceiveAdInfo(int status)

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: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、広告がタップされた際は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
広告表示の際に圏外状態の際に通知されます。

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。
表示中に、同一 FluctInterstitial インスタンスから広告表示要求が呼び出された際にも通知されます。

インタースティシャルの実装に関しては付属の Sample アプリもご参照ください。

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	OS バージョン 4.1 の場合、「4」
2	Google 広告 ID (Google Advertising ID)	ユーザーが広告 ID 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わない

※Google 広告 ID が使用される条件は、Android2.3 以上かつ GooglePlay のバージョン 4.0 以上となります。