Flurry Advertising

iOS SDK Instructions

SDK version 4.0.4
Updated: 09/17/2012

Welcome to Flurry Advertising!

This README contains:

---

1. Introduction

Flurry Advertising is a flexible ad serving solution that allows you to easily manage the complete monetization of your mobile applications. Through integration with Flurry Analytics and various mobile ad networks, publishers can easily generate the maximum value from ad campaigns. Both AppCircle and AppSpot (direct and custom network) ad campaigns are created and managed in the same interface.

With Flurry Ads you will be able to:

1. Define your inventory
2. Traffic your ad campaigns
3. Track & optimize your performance

The Flurry Ads SDK is modular and contains only the functionality related to serving advertisements. It is designed to be as easy as possible with a basic setup completed in under 5 minutes.

For more information on changes to Flurry's Ad Space set up, see the Getting Started Guide: http://support.flurry.com/index.php?title=Publisher/GettingStarted. Please note, it is **required** that you create your ad space on the server before retrieving ads.

These instructions assume that you have already integrated Flurry Analytics into your application. If you have not done so, please refer to **Analytics-README** to get started.

---

2. Basic Integration

Follow these steps to quickly integrate Flurry Ads into your app:

1. In the finder, drag FlurryAds/ into project's file folder.
2. Now add it to your project: Project > Add to project > FlurryAds - Choose 'Recursively create groups for any added folders'
3. In your source code, import FlurryAds and initialize the ad system. To intialize ads, add the following line to your app after the call to [Flurry startSession:].

```
+ (void) initialize: (UIViewController *) rvc;
```

The parameter is as follows:

- UIViewController rvc - The root view controller of your application window

The final integration will look like:

```
#import "FlurryAds.h"
- (BOOL)application:(UIApplication *)application
            didFinishLaunchingWithOptions:(NSDictionary *)launchOptions {
        [Flurry startSession:@"YOUR_API_KEY"];
        [FlurryAds initialize:rvc];
        //your code
 }
```

3. Requesting Ads

Call [FlurryAds showAdForSpace:view:size:timeout] after the session has started. Note: please see **Analytics-README** for details on [Flurry startSession:]

```
+ (BOOL)showAdForSpace:(NSString*)space view:(UIView *)viewContainer
size:(FlurryAdSize)size timeout:(int64_t)timeout;
```

If the ad request is filled then this will return true, and false if no ad was available. The parameters are as follows:

- **NSString space** - Name of the adSpace that you defined on the Flurry website
- **UIView viewContainer** - UIView that you want the ad to reside in
- **FlurryAdSize size** - The default size for your adSpace. This is overwritten by any value set on the server.
- **int64_t timeout** - Maximum time (in milliseconds) this method should block before it returns. To display ad completely asynchronously set the timeout to 0. This is useful for the display of banners where interaction with the app can continue unimpeded by the ad display.

Note that the same ad call is used for all ad formats and sizes including Banner, Takeover and Custom ad formats. In order to specify the format and size, you should log into the Flurry Website and select the Publisher tab to specify the configuration for the adSpace name you have chosen.

---

4. Additional Ad Controls (Optional)

Checking if an ad is available

Flurry provides a method to allow your app to check for the availability of an ad without displaying it. This is useful if you provide an option for a user to view an ad (e.g. - display a button to View a Video). Simply add the following call within your app with the name of the adSpace that will ultimately display the ad

```
+ (BOOL)isAdAvailableForSpace:(NSString*)space view:(UIView *)viewContainer
size:(FlurryAdSize)size timeout:(int64_t)timeout;
```

This call will check the Flurry server for the availability of an ad. If an ad is available then this will return true, and false if no ad was available. If this returns true you are guaranteed an ad will be available for the given adSpace when [FlurryAds showAdForSpace:view:timeout] is called. The parameters are as follows:

- **NSString space** - Name of the adSpace that you defined on the Flurry website
- **UIView viewContainer** - UIView that you want the ad to reside in
- **FlurryAdSize size** - The default size for your adSpace. This is overwritten by any value set on the server.
- **int64_t timeout** - Maximum time (in milliseconds) this method should block before it returns.

Removing an ad

Flurry manages the lifecycle of the ads it displays, however, you can exercise finer control over display by choosing when to add and remove the ads from your app (e.g. - in viewWillAppear and viewDidDissapear). To remove an ad just call

```
+ (void)removeAdFromSpace:(NSString*)space;
```

The parameter is as follows:

- **NSString space** - Name of the adSpace that you defined on the Flurry website

---

5. Enabling Ad Network Mediation (Optional)

Once you have your Ad Spaces set up, you will have the option of selecting 3rd party ad networks to serve ads into your Ad Spaces using the Flurry website (in addition to AppCircle, AppSpot and your own ads). You can change which ad networks serve ads at any time on the Flurry website, but in order to enable them you need to add the ad network SDKs into your application and configure them. The list of currently supported Ad Networks contains:

- iAd
- Admob
- Millennial
- InMobi
- Mobclix
- Greystripe
- Jumptap

To implement an Ad Network you must perform the following steps:

1. Include the Ad Network iOS SDK with your app and add it to the project. Follow the instructions from the Ad network on how to complete this step.
2. Implement the appropriate delegate methods in FlurryAdDelegate

Here is the example of implementing the Admob SDK into FlurryAds:

```
@interface MyDelegateClass : NSObject <FlurryAdDelegate> {

 // definitions
}

// MyDelegateClass.m

@implementation MyDelegateClass
      - (NSString *) appSpotAdMobPublisherID
      {
            // Your AdMob Publisher ID, found from here:
                http://www.admob.com/my sites/ (click manage settings)
            return @"<your id>";
        }

        - (BOOL)appSpotTestMode
        {
            // If testing, return yes
            return YES;
        }

@end
```

---

6. Request Configuration (Optional)

There are a number of configuration parameters that you can use to modify the behavior of your ad spaces.

Option 1. Enable Test Ads
Add a call to receive test ads from the flurry server to ensure proper implementation. Test ads do not generate revenue and therefore MUST be disabled before submitting to the AppStore:
```
[FlurryAds enableTestAds:(BOOL)enable];
```

Option 2. Set Location
Add a call to set the location (lat,long) that you want associated with the ad request, to be used with geographical targeting:
```
[Flurry setLatitude:(double)latitude longitude:(double)longitude
horizontalAccuracy:(float)horizontalAccuracy verticalAccuracy:(float)
verticalAccuracy];
```

Option 3. User Cookies
Add a call to identify any user specific information you want associated with the ad request:
```
[FlurryAds setUserCookies:(NSDictionary *) userCookies];
```

To remove any user cookies call:
```
[FlurryAds clearUserCookies];
```

Option 4. Keyword Targeting
Add a call to specify keywords to be used when targeting ads:
```
[FlurryAds setKeywordsForTargeting:(NSDictionary *) keywords];
```

To clear the set keywords call:
```
[FlurryAds clearKeywords];
```

---

7. Implementing Ad Delegate (Optional)

To be notified of certain events during the full lifecycle of the Ad, you can implement the FlurryAdDelegate and then call the [FlurryAds setDelegate:] method to attach your delegate to the FlurryAds. When you implement the FlurryAdDelegate you will implement the following callback methods:

- (BOOL) spaceShouldDisplay:(NSString*)adSpace forType:(FlurryAdType)adType
    - This method will be called if there is an ad available to display for that adSpace. The FlurryAdType is the following enumeration: `WEB_BANNER, WEB_TAKEOVER, VIDEO_TAKEOVER`. You can choose whether or not to show the available ad by returning either true or false. If spaceShouldDisplay returns false, then the ad will not be displayed and showAdForSpace: will return false. If spaceShouldDisplay returns true, then the ad will be displayed and showAdForSpace: will return true.
- (void) spaceWillDismiss:(NSString *)adSpace;
    - This method will be called when the user dismisses the current Ad for the provided Ad Space name.
- (void) spaceWillLeaveApplication:(NSString *)adSpace
    - This method will be called when the user is leaving the application after following events associated with the current Ad in the provided Ad Space name.

Example usage:
```
@interface MyDelegateClass : NSObject <FlurryAdDelegate> {

 // definitions
}

// MyDelegateClass.m

@implementation MyDelegateClass

      -(void) init {
            [super init];
            [FlurryAds setAdDelegate:self];        // Set the delegate
      }

      // Other code
      - (BOOL) spaceShouldDisplay:(NSString*)adSpace
            forType:(FlurryAdType)adType
      {
            // Decide if the Ad should be displayed
            return true;
      }

      - (void)spaceWillDismiss:(NSString *)adSpace
      {
            // Handle the user dismissing the ad
      }

      - (void)spaceWillLeaveApplication:(NSString *)adSpace;
      {
            // Handle the user leaving the application
```

```
        }

@end
```