AppsFlyer iOS SDK Integration Guide

Version 3.3.1

Table of Contents

1. Introduction

AppsFlyer's SDK provides app installation and event tracking functionality. Our goal is to provide you with an SDK that is highly robust (8+ billion SDK installations to date), secure, lightweight and very simple for the developer to embed. You can track installs, updates and sessions (by following the mandatory steps below), and you can also track additional in-app events beyond app installs (including in-app purchases, game levels, etc.) to evaluate ROI and user engagement levels.

The mandatory steps are explained in Section 3 and Section 4 below, followed by additional optional features.

The iOS SDK is compatible with all iOS devices (iPhone, iPod, iPad) with iOS version 6 and above (including 9).

2. What's New in this Version?

2.1 Support iOS 9
2.2 Support TLS 1.2 for iOS 9
2.3 In-App Purchase Receipt Validation (section 6.7) - Updated this API to support the transaction validation as well
2.4 The SDK is available both as a framework and as a static library

3. Embed the SDK into Your Application (Mandatory)

AppsFlyer SDK is available both as a framework and as a static library.

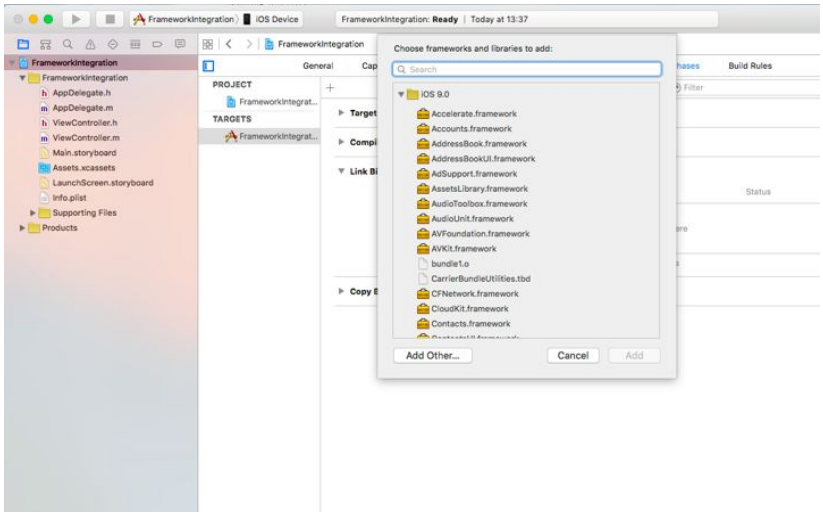
AppsFlyer SDK Framework

The simplest way to integrate the framework is using cocoapods:

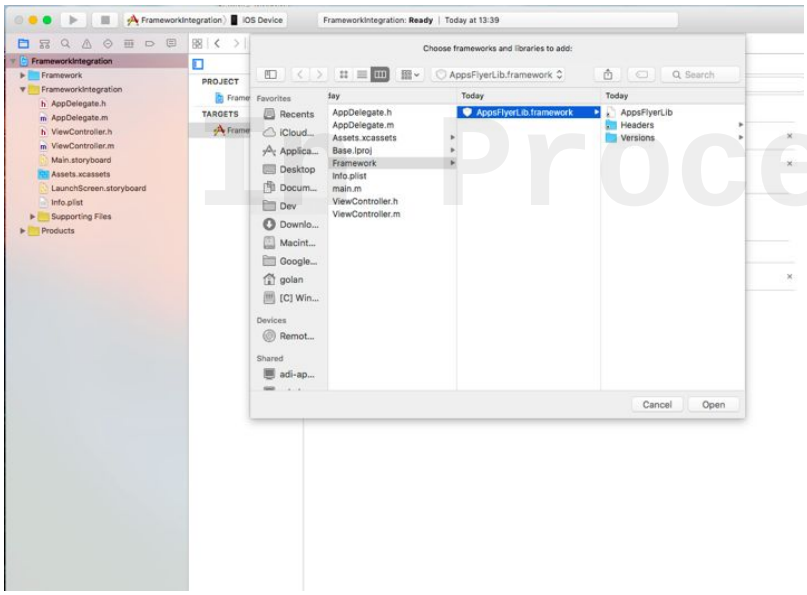
- Add the following line to your pod file: *pod 'AppsFlyerFramework'*

If you don't use cocoapods, follow the steps below:

- In Xcode, go to "Build Phases" >> "Link Binary with Libraries" >> Press on the "+" sign to add a new library



➢ Press on “Add Other” and add the “AppsFlyerLib.framework”



➢ Include the following header: *#import "AppsFlyerLib/AppsFlyerTracker.h"*

AppsFlyer SDK Static Library

The simplest way to integrate the static library is using cocoapods:

➢ Add the following line to your pod file: *pod 'AppsFlyer-SDK'*

If you don’t use cocoapods, follow the steps below:

➢ Add the header and lib files to your project.
AppsFlyer SDK components (**libAppsFlyerLib.a & AppsFlyerTracker.h**) are available here.

➢ Add the AppsFlyer header import: *#import "AppsFlyerTracker.h"*

➢ Add the **AdSupport.framework** to your project and set it as “Optional”. You can find the submission instructions here.
Note: AppsFlyer will collect IDFA only if you include **AdSupport.framework**

➢ Add the **iAd.framework** to your project.

4. SDK Initialization & Installation Event (Minimum Requirement for Tracking)

Note that this is the minimum requirement to start tracking your app installs.

You need to initialize the SDK on the first launch of the app. Please make sure that the SDK is initialized before sending the tracking event below.

4.1 In order to initialize the SDK, add the following to your **"didFinishLaunchingWithOptions"** function:

```
[AppsFlyerTracker sharedTracker].appsFlyerDevKey = @"[Dev_Key]";
[AppsFlyerTracker sharedTracker].appleAppID = @"REPLACE THIS WITH YOUR App_ID";
```

Replace **[Dev_Key]** with your own **Dev_Key** (accessible from your account, see "Settings" >> "Integrate the SDK into..." in the AppsFlyer dashboard).

4.2 Add the following code to your **AppDelegate.m** source file at **"applicationDidBecomeActive"** function:


```
#import "AppsFlyerTracker.h"
-(void)applicationDidBecomeActive:(UIApplication *)application
{
    // Track Installs, updates & sessions(app opens) (You must include this API to enable tracking)
    [[AppsFlyerTracker sharedTracker] trackAppLaunch];
}
```

5. In-App Events Tracking API (Optional)

This API enables AppsFlyer to track post install events. These events are defined by the advertiser and include an event name in addition to optional event values.

Tracking in-app events helps you measure and analyse how loyal users discover your app, and attribute them to specific campaigns/media sources. It is recommended to take the time and define the events you would like to measure to allow you to track ROI (Return on Investment) and LTV (Lifetime Value).

Syntax:

```
- (void) trackEvent:(NSString *)eventName withValues:(NSDictionary*)values
```

eventName is any string to define the event name. You can find a list of recommended constant event names in Appendix A.
values is a dictionary of event parameters that comprise a rich event. You can find a list of recommended parameters in Appendix A.

Counting revenue as part of a rich in-app event: Use the 'af_revenue" (*AFEventParamRevenue* constant) event parameter to count revenue as part of an in-app event. You can populate it with any

numeric value, positive or negative. Note that “af_price” (*AFEventParamPrice*) will not be counted as revenue and is a descriptive parameter which does not affect revenue and LTV measurements.

Example 1: Level achieved in-app event

```
[[AppsFlyerTracker sharedTracker] trackEvent: AFEventLevelAchieved withValues:@{
    AFEventParamLevel: @9,
    AFEventParamScore : @100 }];
```

This will generate an event of type “af_level_achieved” with the following event values:
{af_level: 9 , af_score: 100}

Example 2: Purchase Event

```
[[AppsFlyerTracker sharedTracker] trackEvent:AFEventPurchase withValues: @{
    AFEventParamContentId:@"1234567",
    AFEventParamContentType : @"category_a",
    AFEventParamRevenue: @200,
    AFEventParamCurrency:@"USD"}];
```


This will generate an event of type “af_purchase” with the following event values:
{af_content_id: “1234567” , af_content_type: “category_a”, af_revenue: 200, af_currency: “USD”}

The purchase event above contains a $200 revenue which will appear as revenue in the dashboard.

For more information see Appendix A: AppsFlyer Rich In-App Events

6. Advanced Integration

The APIs below are optional and are part of the advanced integration with AppsFlyer SDK.

6.1 Set Currency Code (Optional)

You can set a global currency code using the API below, in addition to specific currency codes that can be used as part of each in-app event that is sent to AppsFlyer. The global currency code will be used when “af_currency” (*AFEventParamCurrency*) is not sent as part of an in-app event.

USD is the default value. Please find acceptable ISO currency codes **here**.

Use the following API in order to set the currency code:

```
[[AppsFlyerTracker sharedTracker].currencyCode = @"GBP"];
```

6.2 Get AppsFlyer Unique ID (Optional)

An AppsFlyer proprietary unique ID is created for every new install of an app. AppsFlyer's unique ID is the main ID used by AppsFlyer in the Reports and APIs.

Use the following API In order to get AppsFlyer's unique ID:

```
(NSString *) [AppsFlyerTracker sharedTracker] getAppsFlyerUID
```

6.3 Set Customer User ID (Optional)

Setting your own customer ID will enable you to cross-reference your own unique ID with AppsFlyer's unique ID and the other devices' IDs. This ID will be available in AppsFlyer CSV reports along with postbacks APIs for cross-referencing with your internal IDs.

To set your customer user ID:

```
[[AppsFlyerTracker sharedTracker].customerUserID = @"YOUR_CUSTOM_DEVICE_ID"];
```


Important Notes:

- It is recommended to set your customer user ID as soon as possible as it will only be associated to events reported after setting this attribute.
- You must set the Customer User ID using this API in order to use AppsFlyer's integrations with Analytics platforms such as Mixpanel and Swrve.

6.4 Get Conversion Data (Optional)

AppsFlyer allows you to access the user attribution data in real time directly at the SDK level. It enables you to customize the landing page a user sees on the very first app open after a fresh app install.

For more information regarding this advanced functionality, read here.

6.5 Set User Email (Optional)

AppsFlyer enables you to report one or more of the user's email addresses. The developer should collect the email addresses from the user and report it to AppsFlyer according to the developer's desired encryption method. The following encryption methods are available: Sha1, MD5 and plain.

Example:

```
[[AppsFlyerTracker sharedTracker] setUserEmails:@[@"email1@domain.com",
@"email2@domain.com"] withCryptType:EmailCryptTypeSHA1];
```

6.6 Reporting Deeplinks for Re-Targeting Attribution (Optional)

AppsFlyer enables you to report launches initiated through deeplinks and Universal Links and to analyse the performance of your re-targeting attribution campaigns.

To report such launches, add the following code from the app delegate:

```
// Reports app open from a Universal Link for iOS 9
- (BOOL) application:(UIApplication *)application continueUserActivity:(NSUserActivity
*)userActivity restorationHandler:(void (^)(NSArray *_Nullable))restorationHandler
{
        [[AppsFlyerTracker sharedTracker] continueUserActivity:userActivity
restorationHandler:restorationHandler];
        return YES;
}

// Reports app open from deeplink for iOS 8 or below
- (BOOL)application:(UIApplication *)application openURL:(NSURL *)url
sourceApplication:(NSString*)sourceApplication annotation:(id)annotation
{
        [[AppsFlyerTracker sharedTracker] handleOpenURL:url
        sourceApplication:sourceApplication];
        return YES;
}
```


6.7 In-App Purchase Receipt Validation (Optional)

Note that this function is supported for iOS7 and above.

AppsFlyer's SDK can provide in-app purchase server verification. To set receipt validation tracking you need to call the **validateAndTrackInAppPurchase** method inside the SKStoreKit's **completeTransaction**: callback. This call will automatically generate an "af_purchase" in-app event.

```
- (void) validateAndTrackInAppPurchase:(NSString *) productIdentifier
        price:(NSString *) price
        currency:(NSString *) currency
        transactionId:(NSString *) tranactionId
        additionalParameters:(NSDictionary *) params
        success:(void (^)(NSDictionary *response)) successBlock
        failure:(void (^)(NSError *error, id reponse)) failedBlock;
```

This call has two callback blocks, one for 'success' and one for 'failure' (for any reason, including validation fail). On success, a dictionary will be returned with the receipt validation data provided by Apple's servers.
Important: For testing purposes, we recommend to set the **useReceiptValidationSandbox** flag to YES, as this will redirect the requests to Apple sandbox servers.

```
[AppsFlyerTracker sharedTracker].useReceiptValidationSandbox = YES;
```

Example:

```
[[AppsFlyerTracker sharedTracker]
validateAndTrackInAppPurchase:product.productIdentifier
price:product.price.stringValue
currency:@"USD"
transactionId:trans.transactionIdentifier
additionalParameters:@{@"test": @"val" , @"test1" : @"val 1"}
success:^(NSDictionary *result){
        NSLog(@"Purcahse succeeded And verified!!! response: %@",
result[@"receipt"]);
} failure:^(NSError *error, id response) {
        NSLog(@"response = %@", response);
}];
```


6.8 Set HTTPS (Optional)

AppsFlyer's SDK communicates with its servers over HTTPS. In case you choose to disable this **(not recommended)**, you can set the isHTTPS property to NO. The default is YES.

```
[AppsFlyerTracker sharedTracker].isHTTPS = YES;
```

6.9 End User Opt-out (Optional)

AppsFlyer provides you a method to opt-out specific users from AppsFlyer analytics. This method complies with the latest privacy requirements and complies with Facebook data and privacy policies. Default is NO, meaning tracking is enabled by default. Please use this API during the SDK initialization in Section 4 to explicitly opt-out:

```
[AppsFlyerTracker sharedTracker].deviceTrackingDisabled = YES;
```

6.10 Explicit opt-out from ID for Advertisers – IDFA/IFA (Optional)

AppsFlyer SDK will collect IDFA only if AdSupport.framework library is included in your project. No need to explicitly opt-in or opt-out. However, in case you would like to explicitly opt-out IDFA please

use the following API during the SDK initialization in Section 4:

```
[AppsFlyerTracker sharedTracker].disableAppleAdSupportTracking = YES;
```

7. Testing the SDK Integration

To learn how to test the SDK integration before and after submitting to the App Store - read here

Appendix A: AppsFlyer Rich In-App Events

AppsFlyer's rich in-app events provide advertisers with the ability to track any post-install event and attribute it to a media source and campaign.

An in-app event is comprised of an event name and event parameters (see below lists of recommended event and parameter names)

Syntax:

```
- (void) trackEvent:(NSString *)eventName withValues:(NSDictionary*)values
```

eventName is any string to define the event name.
values is a dictionary of event parameters that comprise a rich event.

Example 1:
"af_add_to_cart" of a single item with the value of 9.99 USD, content_id "234234" and of type "category_a":

```
[[AppsFlyerTracker sharedTracker] trackEvent:AFEventAddToCart withValues:@{
    AFEventParamPrice: @9.99,
    AFEventParamContentType : @"category_a",
    AFEventParamContentId: @"234234",
    AFEventParamCurrency : @"USD",
    AFEventParamQuantity : @1
}];
```


The following in-app event types are defined and recommended:

Event String Constant	Event String Name
AFEventLevelAchieved	af_level_achieved
AFEventAddPaymentInfo	af_add_payment_info
AFEventAddToCart	af_add_to_cart
AFEventAddToWishlist	af_add_to_wishlist
AFEventCompleteRegistration	af_complete_registration
AFEventTutorialCompletion	af_tutorial_completion
AFEventInitiatedCheckout	af_initiated_checkout
AFEventPurchase	af_purchase
AFEventRate	af_rate
AFEventSearch	af_search
AFEventSpentCredits	af_spent_credits
AFEventAchievementUnlocked	af_achievement_unlocked

AFEventContentView	af_content_view
AFEventListView	af_list_view
AFEventTravelBooking	af_travel_booking
AFEventShare	af_share
AFEventInvite	af_invite
AFEventLogin	af_login
AFEventReEngage	af_re_engage
AFEventUpdate	af_update
AFEventOpenedFromPushNotification	af_opened_from_push_notification

The following section describes the recommended structure of each event type together with the parameters mappings to Facebook, Twitter and Criteo:

af_level_achieved

Description: Used to track game level events.
Recommended attributes: af_level, af_score
Facebook Mapped Event: fb_mobile_level_achieved
Twitter Mapped Event: LEVEL_ACHIEVED
Criteo Mapped Event: None
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_level	fb_level	level	-
af_score	-	-	-

af_add_payment_info

Description: Used to track payment info configuration status.
Recommended attributes: af_success
Facebook Mapped Event: fb_mobile_add_payment_info
Twitter Mapped Event: ADDED_PAYMENT_INFO
Criteo Mapped Event: None
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_success	fb_success	user_payment_info	-

af_add_to_cart

Description: Used to track add to cart events of specific items.
Recommended attributes: af_price, af_content_type, af_content_id, af_currency, af_quantity
Facebook Mapped Event: fb_mobile_add_to_cart
Twitter Mapped Event: ADD_TO_CART
Criteo Mapped Event: viewBasket
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_price	_valueToSum	price_micro*	price**
af_content_type	fb_content_type	content_type	-
af_content_id	fb_content_id	content_id	item_id
af_currency	fb_currency	price_currency	-
af_quantity	-	number_items	quantity

af_add_to_wishlist

Description: Used to track add to wishlist events of specific items.
Recommended attributes: af_price, af_content_type, af_content_id, af_currency, af_quantity
Facebook Mapped Event: fb_mobile_add_to_wishlist
Twitter Mapped Event: ADD_TO_WISHLIST
Criteo Mapped Event: None
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_price	_valueToSum	price_micro*	-
af_content_type	fb_content_type	content_type	-
af_content_id	fb_content_id	content_id	-
af_currency	fb_currency	price_currency	-
af_quantity	-	number_items	-

af_complete_registration

Description: Used to track user registration methods.
Recommended attributes: af_registration_method
Facebook Mapped Event: fb_mobile_complete_registration
Twitter Mapped Event: SIGN_UP
Criteo Mapped Event: None
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_registration_method	fb_registration_method	registration_method	-

af_tutorial_completion

Description: Used to track tutorial completions.
Recommended attributes: af_success, af_content_id
Facebook Mapped Event: fb_mobile_tutorial_completion
Twitter Mapped Event: TUTORIAL_COMPLETE
Criteo Mapped Event: None
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_success	fb_success	-	-
af_content_id	fb_content_id	content_id	-

af_initiated_checkout

Description: Used to track checkout events.
Recommended attributes: af_price, af_content_type, af_content_id, af_quantity, af_payment_info_available, af_currency
Facebook Mapped Event: fb_mobile_initiated_checkout
Twitter Mapped Event: CHECKOUT_INITIATED
Criteo Mapped Event: None
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_price	_valueToSum	price_micro*	-
af_content_type	fb_content_type	content_type	-
af_content_id	fb_content_id	content_id	-

af_currency	fb_currency	price_currency	-
af_quantity	fb_num_items	number_items	-
af_payment_info_available	fb_payment_info_available	user_payment_info	

af_purchase

Description: Used to track purchase events (and associate revenue to them).
Recommended attributes: af_revenue, af_content_type, af_content_id, af_quantity, af_currency
Facebook Mapped Event: fb_mobile_purchase
Twitter Mapped Event: PURCHASE
Criteo Mapped Event: trackTransaction
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_revenue*	_valueToSum	price_micro**	price***
af_content_type	fb_content_type	content_type	-
af_content_id	fb_content_id	content_id	item_id
af_currency	fb_currency	price_currency	-
af_quantity	fb_num_items	number_items	quantity
af_validated	-	-	-
af_receipt_id	-	-	-

* af_revenue will be counted as revenue in AppsFlyer's platform

af_rate

Description: Used to track app/item rating events.
Recommended attributes: af_rating_value, af_content_type, af_content_id, af_max_rating_value
Facebook Mapped Event: fb_mobile_rate
Twitter Mapped Event: RATED
Criteo Mapped Event: None
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_rating_value	_valueToSum	price_micro	-

af_content_type	fb_content_type	content_type	-
af_content_id	fb_content_id	content_id	-
af_max_rating_value	fb_max_rating_value	max_rated_value	-

af_search

Description: Used to track search events.
Recommended attributes: af_content_type, af_search_string, af_success
Facebook Mapped Event: fb_mobile_search
Twitter Mapped Event: SEARCH
Criteo Mapped Event: viewSearch
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_content_type	fb_content_type	content_type	-
af_search_string	fb_search_string	search_string	-
af_date_a	-	-	din
af_date_b	-	-	dout
af_destination_a	-	-	-
af_destination_b	-	-	-
af_success	fb_success	-	-

af_spent_credits

Description: Used to track credit spend events.
Recommended attributes: af_price, af_content_type, af_content_id
Facebook Mapped Event: fb_mobile_spent_credits
Twitter Mapped Event: SPENT_CREDITS
Criteo Mapped Event: trackTransaction
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_price	_valueToSum	price_micro*	price**
af_content_type	fb_content_type	content_type	-

af_content_id	fb_content_id	content_id	item_id

af_achievement_unlocked

Description: Used to track achievement unlocking events.
Recommended attributes: af_description
Facebook Mapped Event: fb_mobile_achievement_unlocked
Twitter Mapped Event: ACHIEVEMENT_UNLOCKED
Criteo Mapped Event: None
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_description	fb_description	description	-

af_content_view

Description: Used to track content view events.
Recommended attributes: af_price, af_content_type, af_content_id, af_currency
Facebook Mapped Event: fb_mobile_content_view
Twitter Mapped Event: CONTENT_VIEW
Criteo Mapped Event: viewProduct/viewListing
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_price	_valueToSum	price_micro*	price
af_content_type	fb_content_type	content_type	-
af_content_id	fb_content_id	content_id	item_id
af_currency	fb_currency	price_currency	-

af_list_view

Description: Used to track listings view events.
Recommended attributes: af_content_type, af_content_list
Facebook Mapped Event:
Twitter Mapped Event:
Criteo Mapped Event: viewListing
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_content_type	-	-	-
af_content_list	-	-	product

af_travel_booking

Description: Used to track travel booking events (and associate revenue to them).
Recommended attributes: af_revenue, af_destination_a, af_destination_b, af_class, af_description, af_customer_user_id, af_content_type, af_content_id, af_date_a, af_date_b
Facebook Mapped Event: fb_mobile_purchase
Twitter Mapped Event: PURCHASE
Criteo Mapped Event: trackTransaction
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_revenue*	_valueToSum	price_micro**	price***
af_customer_user_id	-	-	cid
af_content_type	fb_content_type	content_type	-
af_content_id	fb_content_id	content_id	-
af_class	-	-	-
af_date_a	-	-	din
af_date_b	-	-	dout
af_destination_a	-	-	-
af_destination_b	-	-	-
af_success	fb_success	-	-

* af_revenue will be counted as revenue in AppsFlyer's platform

af_share

Description: Used to track sharing events.
Recommended attributes: af_description
Facebook Mapped Event: None
Twitter Mapped Event: SHARE

Criteo Mapped Event: None
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_description	-	description	-

af_invite

Description: Used to track invite (social) events.
Recommended attributes: None
Facebook Mapped Event: None
Twitter Mapped Event: INVITE
Criteo Mapped Event: None
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_description	-	description	-

af_login

Description: Used to track user login events.
Recommended attributes: None
Facebook Mapped Event: None
Twitter Mapped Event: LOGIN
Criteo Mapped Event: None

af_reengage

Description: Used to track user re-engagement events.
Recommended attributes: None
Facebook Mapped Event: None
Twitter Mapped Event: RE_ENGAGE
Criteo Mapped Event: None
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_description	-	description	-

af_opened_from_push_notification

Description: Used to track app opens from push notification events.
Recommended attributes: None
Facebook Mapped Event: None
Twitter Mapped Event: None
Criteo Mapped Event: None

af_update

Description: Used to track update events.
Recommended attributes: af_content_id
Facebook Mapped Event: None
Twitter Mapped Event: UPDATE
Criteo Mapped Event: None
Attributes Mapping:

AppsFlyer	Facebook	Twitter	Criteo
af_content_id	-	content_id	-

Optional Parameters:

In addition to the recommended parameters that could be passed with each event, the parameters below are defined and can be sent as part of the event dictionary value:

Parameter String Constant	**Parameter String Name**	**Recommended Value Type**
AFEventParamRevenue	af_revenue*	Float
AFEventParamPrice	af_price	Float
AFEventParamLevel	af_level	Int
AFEventParamSuccess	af_success	Boolean
AFEventParamContentType	af_content_type	String
AFEventParamContentList	af_content_list	Array of strings
AFEventParamContentId	af_content_id	String
AFEventParamCurrency	af_currency	String
AFEventParamRegistrationMethod	af_registration_method	String
AFEventParamQuantity	af_quantity	Int
AFEventParamPaymentInfoAvailable	af_payment_info_available	Boolean
AFEventParamRatingValue	af_rating_value	Float
AFEventParamMaxRatingValue	af_max_rating_value	Float
AFEventParamSearchString	af_search_string	String

AFEventParamDescription	af_description	String
AFEventParamScore	af_score	Int
AFEventParamDestinationA	af_destination_a	String
AFEventParamDestinationB	af_destination_b	String
AFEventParamClass	af_class	String
AFEventParamDateA	af_date_a	String
AFEventParamDateB	af_date_b	String
AFEventParamEventStart	af_event_start	Unixtime
AFEventParamEventEnd	af_event_end	Unixtime
AFEventParamLat	af_lat	Int
AFEventParamLong	af_long	Int
AFEventParamCustomerUserId	af_customer_user_id	String
AFEventParamValidated	af_validated	String
AFEventParamReceiptId	af_receipt_id	String
AFEventParam1	af_param_1	String
AFEventParam2	af_param_2	String
AFEventParam3	af_param_3	String
AFEventParam4	af_param_4	String
AFEventParam5	af_param_5	String
AFEventParam6	af_param_6	String
AFEventParam7	af_param_7	String
AFEventParam8	af_param_8	String
AFEventParam9	af_param_9	String
AFEventParam10	af_param_10	String

* af_revenue is the only parameter that is used for revenue calculations. Use it for events that actually represent revenue generation in your business logic. You can use af_price as a monetary parameter that will not be counted as revenue (such as in an “Add to Cart” event).