Flurry Analytics

iOS SDK Instructions

SDK version 4.2.1
Updated: 4/25/2013

Welcome to Flurry Analytics!

This file contains:

---

1. Introduction

The Flurry iOS Analytics Agent allows you to track the usage and behavior of your iOS application on users' phones for viewing in the Flurry Analytics system. It is designed to be as easy as possible with a basic setup complete in under 5 minutes.

Please note that this SDK will only work with Xcode 4.5 or above. If you need an SDK for an older Xcode version please email support.

This archive should contain these files:

- **Analytics-README.pdf** : This file containing instructions on how to use Flurry Analytics.
- **Flurry/Flurry.h** : The required header file header file containing methods for Flurry Analytics.
- **Flurry/libFlurry.a** : The required library containing Flurry's collection and reporting code.

Flurry Agent does not require CoreLocation framework and will not collect GPS location by default. Developers who use their own CLLocationManager can set GPS location information in the Flurry Agent (see Optional Features for more information).

We also recommend calling Flurry Analytics from the main thread. Flurry Analytics is not supported when called from other threads.

There are additional folders for use with Flurry Ads. These optional libraries provide alternate streams of revenue for your apps. If you would like to use Flurry Ads please refer to FlurryAds-iOS-README.pdf.

---

2. Integration

1. In the finder, drag Flurry/ into project's file folder. *(NOTE: If you are upgrading the Flurry iOS SDK, be sure to remove any existing Flurry library folders from your project's file folder before proceeding.)*

2. Now add it to your project:
File > Add Files to “Your Project” … > Flurry
- Destination: select Copy items into destination group’s folder (if needed)
- Folders: Choose 'Create groups for any added folders'
- Add to targets: select all targets that the lib will be used for

3. Add `SystemConfiguration.framework` to your app. This is required for Reachability to manage network operations efficiently.

4. In your Application Delegate:
    - Import Flurry and inside "applicationDidFinishLaunching:" add: [Flurry startSession:@"YOUR_API_KEY"];

```
#import "Flurry.h"
- (void)applicationDidFinishLaunching:(UIApplication *)application {
[Flurry startSession:@"YOUR_API_KEY"];
//your code
}
```

You're done! That's all you need to do to begin receiving basic metric data.

---

3. Optional / Advanced Features

You can use the following methods to report additional data.

Tracking User Behavior

```
[Flurry logEvent:@"EVENT_NAME"];
```

Use logEvent to count the number of times certain events happen during a session of your application. This can be useful for measuring how often users perform various actions, for example. Your application is currently limited to counting occurrences for 300 different event ids (maximum length 255 characters).

```
[Flurry logEvent:@"EVENT_NAME" withParameters:YOUR_NSDictionary];
```

Use this version of logEvent to count the number of times certain events happen during a session of your application and to pass dynamic parameters to be recorded with that event. Event parameters can be passed in as a NSDictionary object where the key and value objects must be NSString objects. For example, you could record that a user used your search box tool and also dynamically record which search terms the user entered. Your application is currently limited to counting occurrences for 100 different event ids (maximum length 255 characters). Maximum of 10 event parameters per event is supported.

An example NSDictionary to use with this method could be:

```
NSDictionary *dictionary =
[NSDictionary dictionaryWithObjectsAndKeys:@"your dynamic parameter value",
                                           @"your dynamic parameter name",
                                           nil];
```

```
[Flurry logEvent:@"EVENT_NAME" timed:YES];
```

Use this version of logEvent to start timed event.

```
[Flurry logEvent:@"EVENT_NAME" withParameters:YOUR_NSDictionary timed:YES];
```

Use this version of logEvent to start timed event with event parameters.

```
[Flurry endTimedEvent:@"EVENT_NAME" withParameters:YOUR_NSDictionary];
```

Use endTimedEvent to end timed event before app exits, otherwise timed events automatically end when app exits. When ending the timed event, a new event parameters NSDictionary object can be used to update event parameters. To keep event parameters the same, pass in nil for the event parameters NSDictionary object.

```
[Flurry logAllPageViews:navigationController];
```

To enable Flurry agent to automatically detect and log page view, pass in an instance of UINavigationController or UITabBarController to countPageViews. Flurry agent will create a delegate on your object to detect user interactions. Each detected user interaction will automatically be logged as a page view. Each instance needs to only be passed to Flurry agent once. Multiple UINavigationController or UITabBarController instances can be passed to Flurry agent.

```
[Flurry logPageView];
```

In the absence of UINavigationController and UITabBarController, you can manually detect user interactions. For each user interaction you want to manually log, you can use logPageView to log the page view.

Tracking Application Errors

```
[Flurry logError:@"ERROR_NAME" message:@"ERROR_MESSAGE" exception:e];
```

Use this to log exceptions and/or errors that occur in your app. Flurry will report the first 10 errors that occur in each session.

For the following features, please call these APIs before calling `startSession`:

Tracking Demographics

```
[Flurry setUserID:@"USER_ID"];
```

Use this to log the user's assigned ID or username in your system after identifying the user.

```
[Flurry setAge:21];
```

Use this to log the user's age after identifying the user. Valid inputs are 0 or greater.

```
[Flurry setGender:@"m"];
```

Use this to log the user's gender after identifying the user. Valid inputs are `m` (male) or `f` (female)

Tracking Location

```
CLLocationManager *locationManager = [[CLLocationManager alloc] init];
[locationManager startUpdatingLocation];

CLLocation *location = locationManager.location;
[Flurry setLatitude:location.coordinate.latitude
          longitude:location.coordinate.longitude
          horizontalAccuracy:location.horizontalAccuracy
          verticalAccuracy:location.verticalAccuracy];
```

This allows you to set the current GPS location of the user. Flurry will keep only the last location information. If your app does not use location services in a meaningful way, using CLLocationManager can result in Apple rejecting the app submission.

Controlling Data Reporting

```
[Flurry setSessionReportsOnCloseEnabled:(BOOL)sendSessionReportsOnClose];
```

This option is on by default. When enabled, Flurry will attempt to send session data when the app is exited as well as it normally does when the app is started. This will improve the speed at which your application analytics are updated but can prolong the app termination process due to network latency.

```
[Flurry setSessionReportsOnPauseEnabled:(BOOL)sendSessionReportsOnPause];
```

This option is on by default. When enabled, Flurry will attempt to send session data when the app is paused as well as it normally does when the app is started. This will improve the speed at which your application analytics are updated but can prolong the app pause process due to network latency.

```
[Flurry setSecureTransportEnabled:(BOOL)secureTransport];
```

This option is off by default. When enabled, Flurry will send session data over SSL when the app is paused as well as it normally does when the app is started. This has the potential to prolong the app pause process due to added network latency from secure handshaking and encryption.

Crash Reporting [Beta]

```
[Flurry setCrashReportingEnabled:(BOOL)crashReportingEnabled];
```

This option is off by default. When enabled, Flurry will collect crash reports and send it in the session data. The errors that are logged using the Flurry library will include stack traces that are captured at the point when the error is logged. Note that when this feature is enabled Flurry installs an uncaught exception handler. We strongly recommend that developers do not install any uncaught exception handlers in their app as this will disable Flurry's ability to capture crash reports when an uncaught exception is thrown.

Crash reporting is only supported on armv7 architectures (armv7s inclusive). If an application is built using both armv6 and armv7 then crash reporting will work while running on armv7 devices but will be disabled while running on armv6 devices.

5. FAQ

How much does the Flurry Analytics SDK add to my app size?

The Flurry SDK will typically add 150 KB to the final app size.

When does the Flurry Agent send data?

By default, the Flurry Agent will send the stored metrics data to Flurry servers when the app starts, pauses, resumes, and terminates. To override default Agent behavior, you can turn off sending data on termination by adding the following call before you call `startSession`:

```
[Flurry setSessionReportsOnCloseEnabled:NO];
```

You can turn off sending data on pause by adding the following call before you call `startSession`:

```
[Flurry setSessionReportsOnPauseEnabled:NO];
```

How much data does the Agent send each session?

All data sent by the Flurry Agent is sent in a compact binary format. The total amount of data can vary but in most cases it is around 2Kb per session.

What data does the Agent send?

The data sent by the Flurry Agent includes time stamps, logged events, logged errors, and various device specific information. This is the same information that can be seen in the custom event logs on in the Event Analytics section. We do not collect personally identifiable information.

Does the Agent support iOS OS 3.x?

Yes, this version is a fat binary that includes slices for armv6, armv7, armv7s and i386. Support is provided for iOS 3.1 to iOS 6.

What version of XCode is required?

The Flurry SDK will support Xcode 4.0 and above. Please email support if you need to use older versions of the Flurry SDK.

Does this version collect the iOS UDID?

This version of the Flurry iOS SDK does not collect the iOS UDID.

Please let us know if you have any questions. If you need any help, just email iphonesupport@flurry.com!

Cheers,
The Flurry Team
http://www.flurry.com
iphonesupport@flurry.com