Project: Mobile SDK (Collection)
Prepared by: Mark Hayden, VP Product
Prepared On: July 31, 2022

Summary

Mobile interactions make up greater than 40% of all user interactions across our customers, even more globally. Historically Lytics has relied solely on server-to-server integrations with existing analytics or event collection tools that offer robust SDKs to handle data collection from these devices. Though this is a viable solution on a case-by-case basis, customers require a rich set of options for securely and effectively collecting first-party user data and accessing the resulting profiles to power in-app personalization, all in real time. The following document outlines the expectations of the desired mobile solution(s), proposed delivery phases, and requirements.

Background

Lytics's JavaScript SDK enables web-based data collection and personalization. All collection and personalization APIs required to power the mobile solution are based on this existing implementation and are tested, scalable, and documented.

Our Commitment

As previously stated, mobile represents a critical piece of marketing architecture. As such, we seek a partner who can lead the development efforts related to mobile SDKs and provide essential guidance on best practices and industry standards. Ensuring we deliver a reliable and compliant SDK that solves the near-term need is vital, as is the documentation, ease of installation, usability, ongoing maintenance, and flexibility to extensibility as we look to expand functionality.

We are committed to providing direct access to executive-level Product leadership and Sr. level Engineering leadership throughout all project phases. Ensuring quick and thorough answers to technical questions, as well as a means of removing any roadblocks encountered efficiently, is our goal.

Proposed Build Phases

1. **Data Collection** (included in this document)
    a. iOS SDK
    b. Android SDK
2. **Personalization** (not in scope for initial development)

Requirements

The following section outlines the minimum requirements for fulfilling the proposed phase 1 above. During the final scoping process, requirements will be reviewed, refined, and translated into an official scope of work. These requirements are categorized as follows:

- Functionality
    - SDK Configuration
    - User Identification
    - Event Handling
    - Device Storage
- Distribution
    - Dependency Management
    - Security & Compliance
- Delivery & Quality Assurance
    - Source Code
    - Technical Documentation
    - Demo Application
- Maintenance
- Timeline

Functionality

SDK Configuration

Several configuration options must be available during installation to offer customers flexibility based on their particular needs. At a minimum app-level configuration will include the following options.

API Key

A key generated from the customer's Lytics account to be included as the Authorization header in most calls to send and receive data from the Lytics APIs.

Collection URLs

Single Event Collection Endpoint: The destination endpoint(s) for processing individual events.

Bulk Event Collection Endpoint: The destination endpoint(s) for processing batches of events. This will be a single URL by default, but customers may opt to send a copy of events to multiple destinations. The need for both single event and bulk collection endpoints may be removed during the final scoping.

Default Stream

Value of the default collection stream. This particular configuration will act as a global that can then be overridden as part of the configuration of individual events. In most cases, this stream definition will ultimately impact the destination URL for the collection-related API calls.

Session Duration

Each time the app is launched, a new session will begin, and a session event will be processed. This session should be maintained for as long as the app is in the foreground. Any events during that foreground session will be attributed to that session even if the events are queued and processed outside of that session.

Upon navigating away or sending the app to the background, the SDK will maintain a timer to expire the current session. If the user navigates back to the app at any point before the session expires, the session timer will be reset, and the session will continue.

A default length for the session must be put into place. However, this session configuration value will allow for a custom override enabling customers to determine how long they want a session to last. During the final scoping process and based on input from the development partner, a maximum session length may also be instated.

Consent

Ensuring the SDK complies with all global data privacy laws or regulations such as GDPR is critical. As such several consent-related configuration parameters will be required to allow customers to define their requirements around consent. At a minimum, the SDK should supply an API for confirming or denying consent and all subsequent calls should have behavior altered based on that response. Furthermore, the customer's consent state must be stored for a to-be-determined period, likely defined in the configuration. Specific details will be defined in partnership with the development partner. As an example, please refer to mParticles data privacy control documentation.

Collect Advertising ID (IDFA)

The collection and inclusion of the end users' IDFA must be configurable options. IDFA is not a requirement but should the customer prefer to collect it, the SDK must facilitate the proper notification, collection, and storage of consent to ensure IDFA is collected and attributed to each event emitted.

Debug Mode (Logging)

An optional parameter to enable debug level logging must be included. This ensures customers have the tools to test, validate, debug and relay critical information to the Lytics team to support conflict resolution.

Sandbox Mode

Ensuring there is an option to enable a sandbox mode is required. This mode will add a "sandbox" flag to all outbound events when active. This flag then enables those events to be processed in an alternative way or skipped entirely upon delivery to the Lytics collection APIs.

Event Queue

Values that determine how the event queue will be handled. By default, we must use a setting that balances the real-time nature of data collection with optimizing battery life. Should a customer need to skew in either direction, this setting allows for overriding the default value.

User Identification

Capturing and maintaining an end user's identity is critical for Lytics and our customers. The following requirements ensure all data collected has every available identifier to maximize the effectiveness of the overall identity resolution strategy. All identifiers such as IDFA, user id, email address, etc. will be stored on the device and included in all outbound events per the final development spec.

Identifiers

Unique identifiers are a particular type of user field. Lytics uses these fields to match and merge user data across different data sources. The most common unique identifiers for web-based data are _uid and email address. Unique identifiers should represent valuable identification fields of an individual. Regarding the mobile SDK, any number of key/value pairs may be stored and leveraged as identified. Ideally, an "identity" API call would be made, and the associated data would be stored on the device to be included in future event calls.

UUID Generation

One of the most critical parts of any analytics platform is the ability to consistently and accurately identify users. To do this, the SDK must assign and persist some form of identification on the device so that you can analyze user actions effectively. This is especially important for funnel conversion analysis and retention analysis.

To comply with iOS 7+, the SDK must be capable of generating a UUID and storing it on the device. This UUID will then be included as part of all outbound event payloads and any other included identifiers. In addition to generation, there must also be an API for flushing the identifier and joining a new one to be leveraged at the customer's discretion.

Traits

Traits are pieces of information you know about a user that is included in an identify call but may not always represent identifiers. These could be demographics like age or gender, account-specific like a plan, or whether a user has seen a particular A/B test variation. This is flexible and ultimately up to the end user. In practice, these traits would be included as optional properties of the identity call.

Consent

In addition to the details found above in the SDK Configuration section related to consent. A consent API must exist, allowing customers to set the individual users' current state regarding consent. This state must then inform all subsequent data collection efforts. *See consent under SDK Configuration above.*

Event Handling

The goal of the Lytics mobile collect SDK(s) is to make collecting mobile app user data as easy as possible. It should be assumed that all actions tracked or triggered by the mobile SDK result in the delivery of an "event" or JSON object containing a variety of key/value pairs to the configured collection endpoint. This section outlines the minimum requirements for capturing, configuring and emitting said events. It should be assumed that common events such as app installation or opens should be emitted automatically. In contrast, custom events or identification events will be transmitted based on a "track" call to the SDK. In our public documentation, we can find the additional context of how event definitions and example use cases are specific to the web-based JavaScript SDK.

Configuration

All transmitted events will contain, at a minimum, the following components:

- **Type:** A string value that allows additional context when processing the transmitted events. For example, an event type may be "identify" or "pageview." Specific options will be defined during technical scope but be limited to less than ten types.
- **Stream:** A string value is pulled from the payload to inform the final collect endpoint to be used. In general, the stream is used as part of the final path.
- **Identity:** A set of key/value pairs is used to identify the user responsible for the tracked action. By default, for an anonymous user, this will be the value of the generated UUID but may also include things like an email address or IDFA. Identity values should be injected into all event payloads before transmission.
- **Screen**: A standard set of the common and readily available screen or device dimensions will be included with a subset of events based on type. For instance, this information may be during an app open event while it may not be included with a standard track event.
- **Properties:** A set of event-level key/value pairs can be added to gain additional context. These values are not stored on the device and must be included as part of the event call method within the SDK.

Types

It should be assumed that several "helper" methods will be made available to streamline the transmission of common event types. These types will likely include but are not limited to:

- App Install (Auto)

- App Open (Auto)
- App Updates (Auto)
- Screens/Pageview (Auto)
- Universal Link & Android App Link
- Consent
- Custom

API Call Queueing

The goal should always be to balance data delivery to the Lytics collection endpoints while considering battery life and device performance. The SDK must manage an "event queue" to ensure all transmissions occur in the background to not interfere with app performance. Likewise, rather than delivering each event to the endpoint in real time, a reasonable batch size will be leveraged to minimize the number of transmissions.

The queue will transmit data based on three triggers. The first is the total queue size. Once the desired number of events are queued, they will be transmitted. The second is the queue timer, which aims to flush the queue at a standard interval that can be overridden at the app configuration level. Finally, the queue will automatically flush when an app is moved to the background.

It should be assumed that a maximum queue size will also be enforced to limit memory and disk usage. Likewise, all queued events must be stored on the device so as never to risk data loss before they can be transmitted.

Resetting Stored Values

To facilitate a variety of use cases and to test and debug, a method or methods should exist to thoroughly flush all stored values such as event queues, identifiers, etc.

Distribution

Dependency Management

Ensuring the initial installation and updates related to the Lytics mobile SDKs are simple and leveraging platform-based best practices is essential. The developed SDKs must support delivery via the most common dependency management systems.

It should be assumed that in the case of iOS, the SDK must have the ability to be distributed via Cocoapods and Swift Package Manager (SPM). Additional services such as Carthage may also be supported based on our development partner's final scope and input.

In the case of Android, the SDK must, at minimum, support installation via Gradle using Maven.

Security & Compliance

Ensuring the mobile SDKs can be leveraged by large enterprise customers across several highly regulated verticals such as Fintech is essential. As such, the SDK must strictly adhere to all security and privacy requirements outlined by Apple (iOS) and Google (Android).

General Data Protection Regulation (GDPR)

Lytics provides service to many enterprise-level customers outside of the US who must comply with strict user privacy regulations such as GDPR. In addition to the requirements outlined in this document, it should be assumed that the SDKs must allow for compliance with GDPR at a minimum. For additional information and an example of SDK implications related to GDPR, refer to mParticles mobile documentation.

Approval

It, too, should be assumed that the selected development partner will aid in any security or compliance-related conversations for a minimum of five deployments of each mobile SDK and assist in ensuring the associated application attains approval by Apple/Google.

Ownership

Lytics will fully own all materials of any kind that are created as a result of or in support of this SDK project.

Delivery & Quality Assurance

Source Code

Ensuring the developed assets are high quality, maintainable, and scalable is imperative. The following development-related requirements should be met to ensure that minimal technical debt is acquired during this process.

In addition, all source control will be done within the Lytics-managed Github account. A repository(s) will be created and access granted to all required parties.

Code Quality

Source code should be of high quality and include adequate comments and readme-type documentation to support future development efforts. In addition, all source code should be written with the current recommended generation language for each platform, such as Swift in the case of iOS.

Unit Testing

Adequate unit and/or integration tests must be in place. It should be assumed that, at minimum, 80% of the source code should be covered by valid and high-quality tests.

OS Version Support

Android: 8+ (Recommended we try to support back to 4.1 as long as it does not substantially increase lift or put the target delivery date at risk.)

iOS: 14+ (Recommended we try to support back to 13 as long as it does not substantially increase lift or put the target delivery date at risk.)

Target Languages

Android: Kotlin
iOS: Swift

Version Control

These SDKs aim to become the foundation for all mobile collection and personalization for our customer base. Ensuring adequate version management in support of distribution, documentation, and debugging. The specific implementation is at the discretion of our development partner but should align with standards outlined as part of SemVer.

Training & Documentation

Training

As part of the SDK delivery, training documentation and a formal, recorded training session or sessions will be held with technical team members on the Lytics side. Specifics related to this training will be outlined at the project scope/kickoff.

Documentation

Complete technical documentation will be prepared in support of the Lytics team. This document should outline the source code structure, installation, configuration, debugging, test management, deployment, and distribution management processes.

Demo Application

Delivery will contain a simple sample application to demonstrate how the SDK is installed, identity is defined, and ultimately, events are transmitted to Lytics.

Maintenance

Upon delivery of the final, thoroughly tested, and approved SDK, the development partner will provide ongoing support for a minimum of three months. This period may be extended based on a separate agreement based on need.

Project Plan

Milestones

- Develop Requirements Document [complete]
- Document Sample of all required API Calls [in progress]
- Finalize Scope of Work & Development Milestones
- Begin Development
- Development Milestones [Broken Up into Logically Related Batches]
- Code Complete
- Full QA
- SDK Delivery
- Dependency Configuration
- Project Complete

Timeline

The first phase of this project, iOS and Android Data Collection, will be completed on October 24, 2022.