iOS SDK Integration via CocoaPods

Integration Overview

Integrations are done in a few simple steps:

1. Integrate Sardine's JavaScript SDK in your frontend.
2. Set the Ingo provided `clientID` into the JS SDK. This ID is unique to the SDK and varies from your `participant_id`.
3. Call IngoPayments RiskSession API to obtain a `risk_session_token`.
4. Inject the `risk_session_token` as the `sessionKey` value inside the JS SDK.
5. Inject the `customer_ID` value that will be used in the subsequent RiskScore request as the `userIDHash` inside the JS SDK.
6. Call IngoPayments RiskScore API to obtain a risk level assessment of the transaction prior to committing.

—

VARIABLE	DESCRIPTION
flow	Client defined value (ex. payment, deposit, account_funding)
clientId	Ingo Payments provided client identifier

Using CocoaPods

1. Contact your assigned Ingo Payments Integration Manager to provide you with your github access token.
2. Then update your Podfile by adding our GitHub repository url as source before your target and add "SardineSDK" pod. It will look like this:

```
source 'https://github.com/CocoaPods/Specs.git'
source 'git@github.com:sardine-sdk/SardineSDK-PodSpec.git'

target 'testing-sardine-pod (iOS)' do
pod 'SardineSDK'
end
```

Initialize SDK

Initialize SDK in AppDelegate.swift with options.

```
import MobileIntelligence

func application(_ application: UIApplication, didFinishLaunchingWithOptions
launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
    // Override point for customization after application launch.


    let options = OptionsBuilder()
        .setClientId(with: "616c9cea-4801-4503-bfd7-01b37167ee4f")
        .setSessionKey(with: "d3863f93-dab5-4633-a53f-7fa8aa06b8ac")
        .setUserIdHash(with: "ga4tatga3t3qt5")
        .setEnvironment(with: Options.ENV_SANDBOX)
        .setSourcePlatform("Native")
        .build()

    MobileIntelligence(withOptions: options)

    return true
}
```

1. You can update the userIDHash, flow and/or session key by: *(Please pass userIDHash as "" to unset the userIDHash)

> *If you update the options, the SDK will invoke submitData (step 2) to change the existing values. However, Step 2 must be performed again to add any new configuration related data.*

```
var options = UpdateOptions()
options.sessionKey = "sessionKey"
options.userIdHash = "userID"
options.flow = "flow" // login, onboarding, payment, etc

MobileIntelligence.updateOptions(options: options)
```

2. Submit data and log event

```
// SDK function to log the event for Device Intelligence
    MobileIntelligence.submitData { (response) in
        // Handling the response
        if response.status ?? false {
            self.showAlert(msg: response.message ?? "Event logged successfully!")
        } else {
            // showing error on failed to log event
            self.showAlert(msg: response.message ?? "Failed to log an event")
        }
    }
```