

Flutter SDK

flutter_rapidoreach

A plugin for Flutter that supports rendering surveys using RapidoReach SDKs. You can install rapidoreach flutter plugin from [here](#)

Note: RapidoReach iOS SDK utilizes Apple's Advertising ID (IDFA) to identify and retarget users with RapidoReach surveys. As of iOS 14 you should initialize RapidoReach Flutter plugin in iOS only if the relevant IDFA permission was granted by the user

Initializing the plugin

The RapidoReach plugin must be initialized with a RapidoReach API Key. You can retrieve an API key from RapidoReach Dashboard when you sign up and create a new app.

Usage

Initialize RapidoReach

First, you need to initialize the RapidoReach instance with `init` call.

```
// Import RapidoReach package
import 'package:rapidoreach/RapidoReach.dart';

RapidoReach.instance.init(apiKey: 'YOUR_API_TOKEN', userId: 'YOUR_USER_ID')
```

Reward Center

Next, implement the logic to display the reward center. Call the `show` method when you are ready to send the user into the reward center where they can complete surveys in exchange for your virtual currency. We automatically convert the amount of currency a user gets based on the

conversion rate specified in your app.

```
RapidoReach.instance.show(),
```

Reward Callback

To ensure safety and privacy, we recommend using a server side callback to notify you of all awards. In the developer dashboard for your App add the server callback that we should call to notify you when a user has completed an offer. Note the user ID pass into the initialize call will be returned to you in the server side callback. More information about setting up the callback can be found in the developer dashboard.

The quantity value will automatically be converted to your virtual currency based on the exchange rate you specified in your app. Currency is always rounded in favor of the app user to improve happiness and engagement.

Client Side Award Callback

If you do not have a server to handle server side callbacks we additionally provide you with the ability to listen to client side reward notification.

```
RapidoReach.instance.setOnRewardListener( onRapidoReachReward );
```

Implement the callback:

```
void onRapidoReachReward(int quantity) {  
    print(' TR: $quantity');  
}
```

Reward Center Events

You can optionally listen for the `setRewardCenterOpened` and `setRewardCenterClosed` events that are fired when your Reward Center modal is opened and closed.

Add event listeners for `onRewardCenterOpened` and `onRewardCenterClosed`:

```
RapidoReach.instance  
    . setRewardCenterClosed( onRewardCenterClosed );  
RapidoReach.instance  
    . setRewardCenterOpened( onRewardCenterOpened );
```

Implement event callbacks:

```
void onRewardCenterOpened() {  
    print(' onRewardCenterOpened called! ');  
}  
  
void onRewardCenterClosed() {  
    print(' onRewardCenterClosed called! ');  
}
```

Survey Available Callback

If you'd like to be proactively alerted to when a survey is available for a user you can add this event listener.

First, import Native Module Event Emitter:

```
RapidoReach.instance  
    .setSurveyAvaialbleListener( onRapidoReachSurveyAvailable);
```

Implement the callback:

```
void onRapidoReachSurveyAvailable(int survey) {  
    print(' TR: $survey' );  
}
```

Following the rewarded and/or the Offerwall approach

An example is provided on Github that demonstrates how a publisher can implement the rewarded and/or the Offerwall approach. Upon survey completion, the publisher can reward the user.

Limitations / Minimum Requirements

This is just an initial version of the plugin. There are still some limitations:

- You cannot pass custom attributes during initialization
- No tests implemented yet
- Minimum iOS is 9.0 and minimum Android version is 16

For other RapidoReach products, see RapidoReach docs.

Getting Started

If you would like to review an example in code please review the Github project.

Revision #1

Created Tue, Mar 16, 2021 8:35 AM by [Sudarshan Kondgekar](#)

Updated Tue, Mar 16, 2021 8:35 AM by [Sudarshan Kondgekar](#)