iOS Android Server Connection Modes

Device-mode	Cloud-mode
⊘ Web	⊘ Web
✓ Mobile	✓ Mobile
⊘ Server	✓ Server

Getting Started

Segment's **Optimizely Full Stack (previously Optimizely X)** destination supports the following Optimizely products:

- Optimizely Full Stack (server)
- Optimizely Full Stack Android (Cloud-mode)

- Optimizely Full Stack iOS (Cloud-mode)

If you're interested in implementing Optimizely X Web or Optimizely Full Stack with the JavaScript SDK, see Segment's **Optimizely Web Destination**, or follow the links below:

- Optimizely X Web
- Optimizely Full Stack (JavaScript SDK)

Implementation Prerequisite

Optimizely works differently than other Segment destinations: It requires that customers implement some Optimizely functionality natively to make sure your experiment logic runs correctly.

Segment maps `track` events to Optimizely's `track` method, customers must implement all Optimizely decision-based methods, such as `activate` and `isFeatureEnabled` natively. Segment's API does not include methods that correspond to decision-based methods.

This requires that customers include a native Optimizely implementation before their Segment implementation on pages or in mobile apps where Optimizely experiments run.

Server Side

Getting Started

1. In your Segment source dashboard, enable the "Optimizely Full Stack" destination (*not the "Optimizely Web" destination*).
2. Include your Optimizely project's `datafile` URL in your Segment settings.
3. Create a native Optimizely instance in your server environment so you can access Optimizely decisioning methods like `activate`, `isFeatureEnabled`.
4. Finally, define any `events` and `attributes` in your Optimizely dashboard, and to associate `metrics` with the appropriate Optimizely Experiments. Segment maps `track` event names to Optimizely `eventName` - the `eventName` corresponds to an experiment `metric`. In addition, Segment maps `track` event `context.traits` to Optimizely `attributes`.

Note: If you are using Optimizely SDKs v1.x or v2.x: if a visitor has any `activate` or `isFeatureEnabled` calls, their `attributes` object for these calls must match the `attributes` object passed to any `track` calls for that user id so that it can be correctly attributed on the Optimizely results page.

If you are using Optimizely SDKs v3+, Easy Event Tracking is enabled by default for decision events. Set up does not require maintaining the attributes of a user as long as the user id stays the same between Optimizely `activate` and `isFeatureEnabled` calls and Segment `track` calls to have Optimizely `metrics` populated in the Optimizely results page. If you would like to segment your Optimizely results by user `attribute`, then make sure the `attributes` passed in for the `activate` and `isFeatureEnabled` calls match the `attributes` passed in for the `track` calls for that user id.

For more details on how events are attributed on the Optimizely results page, refer to their documentation How Optimzely Experimentation counts conversions.

Track

Upon invocation of a Segment `track` event, Segment maps the event to an Optimizely `track` event:

- If the Segment event name matches exactly the name of an active experiment `metric` set up in the Optimizely dashboard;

- If the experiment `metric` is associated with a running experiment;
- If the current user has been assigned a `userId` using Segment's `identify` method (for example, `analytics.identify('123')`);
- If the current user is activated in a running experiment with the associated `metric`.

Segment also handles the following mapping:

- Segment `track` event name to Optimizely `eventName`.
- Segment `track` event `properties` to Optimizely `eventTags`.
- Segment `track` event `context.traits` to Optimizely `attributes`.

`revenue` values should be passed as a Segment `property`. The value should be an integer and represent the value in cents, so, for example, $1 should be represented by `100`.

Note: Custom Event Tags in Optimizely, which include all Event Tags except `revenue` and `value`, are not displayed on the Optimizely results page, however they are available in a Data Export report. Event Tags can be strings, integers, floating point numbers, or boolean values. Optimizely rejects events with any other data types (for example, arrays).

Segment defaults to identifying users with their `anonymousId`. Enabling the "Use User ID" setting in your Segment settings means that only `track` events triggered by identified users are passed downstream to Optimizely. You may optionally fall back to `anonymousId` when `userId` is unavailable by setting `fallbackToAnonymousId` to `true`.

Notification Listeners

Segment's server-side integration with Optimizely Full Stack does not support notification listeners for Segment `track` events. Notification listeners are still available with any native call invoked from your Optimizely client instance.

Android Cloud-mode Implementation

Getting Started

1. In your Segment source dashboard, enable the "Optimizely Full Stack" destination (*not the "Optimizely Web" destination*).
2. Include the latest version of Optimizely Full Stack's native SDK in your your app-level build.gradle file and implement Optimizely as your would natively.
3. Finally, define any `events` and `attributes` in your Optimizely dashboard, and associate `metrics` with the appropriate Optimizely Experiments. Segment maps `track` event names to Optimizely `eventName` - the `eventName` corresponds to an experiment `metric`. In addition, Segment maps `identify` `traits` to Optimizely `attributes`.

When implementing Optimizely Full Stack using cloud-mode, Segment will map `track` events to Optimizely `track` events on our servers and deliver the data to your Optimizely project as usual.

Note: If you are using Optimizely SDKs v1.x or v2.x: if a visitor has any `activate` or `isFeatureEnabled` calls, the `attributes` object for these calls must match the `attributes` object passed to any `track` calls for that user id so that it can be correctly attributed on the Optimizely results page.

If you are using Optimizely SDKs v3+, Easy Event Tracking is enabled by default for decision events. Set up does not require maintaining the attributes of a user as long as the user id stays the same between Optimizely `activate` and `isFeatureEnabled` calls and Segment `track` calls to have Optimizely `metrics` populated in the Optimizely results page. If you would like to segment your Optimizely results by user `attribute`, then make sure the `attributes` passed in for the `activate` and `isFeatureEnabled` calls match the `attributes` passed in for the `track` calls for that user id.

For more details on how events are attributed on the Optimizely results page, refer to their documentation How Optimzely Experimentation counts conversions.

Track

Upon invocation of a Segment `track` event, Segment maps the event to an Optimizely `track` event:

- If the Segment event name matches exactly the name of an active experiment `metric` set up in the Optimizely dashboard;
- If the experiment `metric` is associated with a running experiment;
- If the current user is activated in a running experiment with the associated `metric`.

Segment also handles the following mapping:

- Segment `track` event name to Optimizely `eventName`.
- Segment `track` event `properties` to Optimizely `eventTags`.

`revenue` values should be passed as a Segment `property`. The value should be an integer and represent the value in cents, so, for example, $1 should be represented by `100`.

Note: Custom Event Tags in Optimizely, which include all Event Tags except `revenue` and `value`, are not displayed on the Optimizely results page, however they are available in a Data Export report. Event Tags can be strings, integers, floating point numbers, or boolean values. Optimizely rejects events with any other data types (for example, arrays).

Segment defaults to identifying users with their `anonymousId`. Enabling "Use User ID" setting in your Segment dashboard means that only `track` events triggered by identified users are passed downstream to Optimizely. You may optionally fall back to `anonymousId` when `userId` is unavailable by setting `fallbackToAnonymousId` to `true`.

Identify

Invoking a Segment `identify` event sets Segment `traits` as Optimizely `attributes`. The `attributes` are sent downstream to Optimizely upon invocation of the next Segment `track` event.

Reset

Invoking this method invalidates the listener for the `Experiment Viewed` event.

Notification Listeners

If you want to use Optimizely's notification listeners, you must use the Optimizely native code to invoke them (in addition to using the Segment SDKs). Notification listeners require an instantiated Optimizely client to be accessed, and so are not available for Segment `track` events when you connect to Optimizely using cloud-mode.

iOS Cloud-mode Implementation

Getting Started

1. In your Segment source dashboard, enable the "Optimizely Full Stack" destination (*not the "Optimizely Web" destination*).
2. Include the latest version of Optimizely Full Stack's native SDK in your app and implement it as you would natively.
3. Finally, define any `events` and `attributes` in your Optimizely dashboard, and to associate `metrics` with the

appropriate Optimizely Experiments. Segment maps `track` event names to Optimizely `eventName` - the `eventName` corresponds to an experiment `metric`. In addition, Segment maps `identify` `traits` to Optimizely `attributes`.

When implementing Optimizely using cloud-mode, Segment will map `track` events to Optimizely `track` events on our servers and deliver the data to your Optimizely project as usual.

Note: If you are using Optimizely SDKs v1.x or v2.x: if a visitor has any `activate` or `isFeatureEnabled` calls, their `attributes` object for these calls must match the `attributes` object passed to any `track` calls for that user id so that it can be correctly attributed on the Optimizely results page.

If you are using Optimizely SDKs v3+, Easy Event Tracking is enabled by default for decision events. Set up does not require maintaining the attributes of a user as long as the user id stays the same between Optimizely `activate` and `isFeatureEnabled` calls and Segment `track` calls to have Optimizely `metrics` populated in the Optimizely results page. If you would like to segment your Optimizely results by user `attribute`, then make sure the `attributes` passed in for the `activate` and `isFeatureEnabled` calls match the `attributes` passed in for the `track` calls for that user id.

For more details on how events are attributed on the Optimizely results page, refer to their documentation How Optimzely Experimentation counts conversions.

Track

Upon invocation of a Segment `track` event, Segment maps the event to an Optimizely `track` event:

- If the Segment event name matches exactly the name of an active experiment `metric` set up in the Optimizely dashboard;
- If the experiment `metric` is associated with a running experiment;
- If the current user is activated in a running experiment with the associated `metric`.

Segment also handles the following mapping:

- Segment `track` event name to Optimizely `eventName`.
- Segment `track` event `properties` to Optimizely `eventTags`.

`revenue` values should be passed as a Segment `property`. The value should be an integer and represent the value in cents, so, for example, $1 should be represented by `100`.

Note: Custom Event Tags in Optimizely, which include all Event Tags except `revenue` and `value`, are not displayed on the Optimizely results page, however they are available in a Data Export report. Event Tags can be strings, integers, floating point numbers, or boolean values. Optimizely rejects events with any other data types (for example, arrays).

Segment defaults to identifying users with their `anonymousId`. Enabling "Use User ID" setting in your Segment dashboard means that only `track` events triggered by identified users are passed downstream to Optimizely. You may optionally fall back to `anonymousId` when `userId` is unavailable by setting `fallbackToAnonymousId` to `true`.

Identify

Invoking a Segment `identify` event sets Segment `traits` as Optimizely `attributes`. The `attributes` are sent downstream to Optimizely upon invocation of the next Segment `track` event.

Notification Listeners

Notification listeners are not available for Segment `track` events when implementing Optimizely using Segment using cloud-mode. Notification listeners are still available with any native call invoked from your Optimizely client instance.

Engage

Follow these instructions on how to set up Engage and Optimizely:

- Using Segment Personas and Optimizely Full Stack for Omnichannel Experiments

GDPR Support

Segment supports deleting/suppressing users in Optimizely using the Segment app. In order to do this however, you will need to create a Personal Access Token in Optimizely and provide it as the value of the Access Token setting.

Settings

Segment lets you change these destination settings from the Segment app without having to touch any code.

SETTING	DESCRIPTION
Account ID	`string`. In order to use Optimizely X via *server side*, you must enter your **Account ID** from your Optimizely account. You can find this ID by visiting https://app.optimizely.com/v2/accountsettings/account/plan
Cache Exp	`number`, defaults to 300. To optimize the server side integration, we will cache the fetched *Datafile* that you have provided for this amount of time (in seconds) in Redis. Since the datafile should not change unless you modified the conditions or variation rules of your experiments, it is advised to have a minimum floor of 300 seconds (5 minutes).
Datafile URL	`string`. In order to use Optimizely X *server side*, you must enter the entire URL for your datafile. It should look something like `https://cdn.optimizely.com/json/9218021209.json`
Fall Back To Anonymous Id	`boolean`, defaults to `FALSE`. Optionally fall back to `anonymousId` when `userId` is not present.

Sends the experiment and variation information as properties on a track call.	`boolean`, defaults to `TRUE`.
	
	Send experiment data to other tools (as a track call). An "Experiment Viewed" track event will be triggered each time you access an Optimizely live variable. If you're regularly accessing live variables and want to reduce the number of track events triggered, pass the "false" flag, for example our Android library would be:
	`Boolean myVariable = optimizelyClient.getVariableBoolean("myVariable",`
	`userId,`
	`false);`
	And for our iOS library:
	`bool myVariable = [optimizely variableBoolean:@"myVariable"`
	`userId:userId]`
	`activateExperiment:False];`

Specifies the Experiment Viewed as a non-interaction event for Google Analytics	`boolean`, defaults to `TRUE`. Send `Experiment Viewed` as a non-interaction event
Only Track Known Users	`boolean`, defaults to `FALSE`. Optimizely does not alias known and unknown users. By default, Segment will only send the `anonymousId` to Optimizely. Enable this to only send the `userId` to Optimizely. *Important*: This setting only applies if you are bundling this integration for mobile sources.
Use Optimizely 3	`boolean`, defaults to `FALSE`. Enable this setting to instantiate an Optimizely client using v3.2.2 of Optimizely's SDK. The default Optimizely client version Segment instantiates is v2.1.3. NOTE: Do not enable this setting if the SDK version you are using to activate users into your Optimizely experiments is less than 3.0. You can find more information on how migrating from Optimizely 2.x to 3.x will affect your experiment tracking in Optimizely's documentation: https://docs.developers.optimizely.com/full-stack/docs/changelog#section-may-2019

SETTING	DESCRIPTION
Use Optimizely User ID	`boolean`, defaults to `FALSE`. Enable this if you want the server side integration to use Optimizely User ID instead of User ID or anonymous ID in your calls. You can include Optimizely User ID in your calls via `integrations.['Optimizely X'].optimizelyUserId` or `integrations.['Optimizely Full Stack'].optimizelyUserId`.
Use User ID	`boolean`, defaults to `FALSE`. Enable this if you want the server side integration to use User ID instead of anonymous ID in your identify calls

This page was last modified: 13 Aug 2024