Components

Browser   Server

Connection Modes ❓

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

Kissmetrics is a behavioral email and analytics platform. It pulls cross-platform behavior reports so marketers can analyze key audience growth segments. It also provides an overview of custom populations, population change and growth, so marketers can analyze populations from customers who have completed actions or events.

Getting Started

To enable Kissmetrics in Segment:

1. From the Segment web app, click **Catalog**.

2. Search for "Kissmetrics" in the Catalog, select it, and choose which of your sources to connect the destination to.

3. In the destination settings, enter your Kissmetrics "API Key".

4. If you are using Kissmetrics using Segment's client-side analytics.js library, Segment asynchronously loads Kissmetrics JavaScript library onto the page. (This means you should remove Kissmetrics's snippet from your page.)

Your Kissmetrics source starts automatically collecting "Visited Site" events and other automatically tracked events.

Page

If you're not familiar with the Segment Specs, take a look to understand what the Page method does. An example call would look like:

```
analytics.page('Docs');
```

By default page calls with `name` and `category` properties will automatically be sent to Kissmetrics in your `page` calls. You can adjust this behavior in your Kissmetrics destination settings by toggling on and off the 'Track Categorized Pages' and 'Track Named Pages' settings.

Here's an example call on docs pages:

```
analytics.page('Docs', { url: 'http:example.com/docs', referrer: 'http://google.com' })
```

This `page` call is translated into an event labeled: **Viewed Docs Page** and will have the properties:

```
'Page - url': 'http:example.com/docs'
'Page - referrer': 'http://google.com'
```

Note: Kissmetrics requires an initial pageview to load the library. By default we include a call to `page` in your snippet. This call must be made at least once on any page where you expect Kissmetrics to be loaded.

Identify

If you're not familiar with the Segment Specs, take a look to understand what the Identify method does. An example call would look like:

```
analytics.identify('userId123');
```

When you call `identify`, we first call Kissmetrics' `identify` method to store the `userId`. Then we call `set` to store the `traits`.

Track

If you're not familiar with the Segment Specs, take a look to understand what the Track method does. An example call would look like:

```
analytics.track('Clicked Button');
```

When you call `track` or one of its helper functions (`trackLink` and `trackForm`,) we will call Kissmetrics' `record` with the exact same parameters.

The Kissmetrics javascript library automatically tracks a bunch of events (Visited Site, Ad Campaign Hit, Search Engine Hit, Form Submit, Pageview, etc.) These will all still work when you use Kissmetrics through Segment.

Note: If you send us an event with a property called "revenue" and we'll pass that on to Kissmetrics as `Billing amount`.

Group

If you're not familiar with the Segment Specs, take a look to understand what the Group method does. An example call would look like:

```
analytics.group('123');
```

When you call `group`, we first call Kissmetrics' `identify` method to store the `userId`. Then we call `set` to store the company `traits` on the user. We prefix these traits with 'Group -'. For example,

```
analytics.group('123', {
  name: 'Test Inc',
  employees: 250
})
```

will add the following traits to the user:

```
'Group - id': '123',
'Group - Name': 'Test Inc'
'Group - Employees': 250
```

Alias

If you're not familiar with the Segment Specs, take a look to understand what the Alias method does. An example call would look like:

```
analytics.alias();
```

Kissmetrics automatically aliases anonymous visitors the first time you call `identify` from the browser. That means there's no need to explicitly call `alias` if you're tracking anonymous visitors and identified users in client-side javascript.

Aliasing on iOS

We will automatically call `alias` for you the first time you `identify` users from our iOS SDK. That way it works exactly like web browser tracking - you don't have to manually `alias` new users.

You can read more about how Kissmetrics recommends using `alias` in their docs.

Best Practices

Merging Identities

A common use of `alias` for Kissmetrics is when an already identified user's unique `id` changes. In this case you need to merge the old identity with a new one.

For example, if you're identifying people by their email address and they change it. In that case you'll need to alias from the previous `id` (their old email address) to the new `userId` (their new email address). Here's an example in `node`:

```
analytics.alias({
  previousId: 'old_email@aol.com',
  userId: 'new_email@example.com'
});
```

Aliasing New Users Server-Side

In order to `identify` **new users** server side and connect that user profile to an existing anonymous visitor profile there's some work to be done.

Remember: Kissmetrics aliases automatically the first time you call `identify` in client-side JavaScript, so in most cases you don't have to call `alias` at all.

We don't recommend handling `alias` server side, but if you must, here's how to make it happen. There are two options: aliasing **in conjunction with client-side tracking** or aliasing when **tracking exclusively server side**.

In Conjunction with Client-Side Tracking

If you're already tracking anonymous users on the client side you'll need to pass the Kissmetrics identity from the browser to your servers in order to `alias` it to the new `userId`.

First, use `analytics.ready` to grab the Kissmetrics identity:

```
analytics.ready(function(){
  var anonIdentity = KM.i();
});
```

Next, pass the `anonIdentity` to your server and `alias`, `identify`, `track` your new user.

Here's a `node` example where the new `userId` is `12345`:

```
analytics.alias({ previousId: anonIdentity, userId: '12345' });
analytics.identify('12345');
analytics.track('Connected Facebook');
```

Tracking Exclusively Server-Side

If you're only tracking anonymous users with one of our server-side sources that makes things easier. All you have to do is `alias` the anonymous `id` to the new `userId`.

Here's a Python example of the `alias`, `identify`, `track` sequence where the server-side anonymous `id` was `92fh49fqh9849hf` and the new `userId` is `12345`:

```
analytics.alias('92fh49fqh9849hf', '12345')
analytics.identify('12345')
analytics.track('Registered')
```

Appendix

User Properties

You can set Kissmetrics user properties in 2 ways with Segment:

1. Make an `identify` call and include a `traits` object.
2. Make a `track` call and include a `properties` object.

Nested Objects or Arrays

Keep in mind that if you send arrays, we will stringify them. Also if you pass a nested object as `traits` or `properties` inside the `identify` or `track` call, we will flatten them.

For example:

```
analytics.track('Signed Up', {
  foo: {
    bar: {
      prop: [1, 2, 3]
    }
  }
});
```

The properties would be sent as `foo.bar.prop: '1,2,3'`.

Note that this is without the prefix setting enabled. If you had enabled that feature, it would be
`Signed Up - foo.bar.prop: '1,2,3'`.

Receive experiment data from A/B Testing tools

You can track A/B testing event data like `Experiment Viewed` and send it to Kissmetrics using Segment.

In order to enable this feature,

1. Find your A/B testing tool in your Segment dashboard

2. Select "Send experiment data to other tools (as an identify call) in Overview

3. Save and Close

4. You should see these events in the debugger

E-Commerce

If you are using our ecommerce api, we will forward that data along to Kissmetrics following the Kissmetrics Ecommerce Essentials Guide.

Engage

You can send computed traits and audiences generated using Engage to this destination as a **user property**. To learn more about Engage, schedule a demo.

For user-property destinations, an identify call is sent to the destination for each user being added and removed. The property name is the snake_cased version of the audience name, with a true/false value to indicate membership. For example, when a user first completes an order in the last 30 days, Engage sends an Identify call with the property `order_completed_last_30days: true`. When the user no longer satisfies this condition (for example, it's been more than 30 days since their last order), Engage sets that value to `false`.

When you first create an audience, Engage sends an Identify call for every user in that audience. Later audience syncs only send updates for users whose membership has changed since the last sync.

> ℹ️
> **Real-time to batch destination sync frequency**
>
> Real-time audience syncs to Kissmetrics may take six or more hours for the initial sync to complete. Upon completion, a sync frequency of two to three hours is expected.

Settings

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

SETTING	DESCRIPTION
API Key (required)	`string` . You can find your API Key on the Kissmetrics Settings page.

SETTING	DESCRIPTION
Prefix Properties	`boolean` , defaults to `TRUE` . Prefix the properties with the page or event name.
Track Categorized Pages	`boolean` , defaults to `TRUE` . Send an event for every page with a category.
Track Named Pages	`boolean` , defaults to `TRUE` . Send an event for every page with a name.

This page was last modified: 27 Oct 2023

Need support?

Questions? Problems? Need more info? Contact Segment Support for assistance!

Visit our Support page

Help improve these docs!

Edit this page

Request docs change

Was this page helpful?

👍 Yes

👎 No

Get started with Segment

Segment is the easiest way to integrate your websites & mobile apps data to over 300 analytics and growth tools.

Your work e-mail

Request Demo

or

Create free account