



Documentation

Getting Started

What is Segment?
[How Segment Works](#)
Getting Started Guide
A Basic Segment Installation
Planning a Full Installation
A Full Segment Installation
Sending Data to Destinations
Testing and Debugging
What's Next
Use Cases

Guides

Connections

Unify

Engage

Privacy

Protocols

Segment App

API

Partners

Glossary

Config API

Help

- Follow Segment's security guidance.
- Request [access to the Segment Developer Center](#).
- Create an App.
- Build and test your Component(s).
- Publish documentation.
- Submit your App for review.
- Launch into *Public Beta*!

Build

Begin by selecting the *Subscription* card in your Developer Center UI after creating an App and selecting *I want to build my own HTTP Endpoint*. Next, you will see a field to input your API endpoint and an interface to test events against this.

Continue reading below to understand what is expected when accepting and responding to Segment data.

Accepting Segment Data

To receive data from Segment, you must provide a server with a static endpoint that can accept HTTPS requests.

The endpoint must:

- *Accept POST requests.* Segment sends customer data to the endpoint you designate in POST requests.
- *Accept JSON data.* Segment sends data in JSON.
- *Use HTTPS.* Segment transmits potentially sensitive data on behalf of customers, and HTTPS is the first step in making sure their data stays safe.

Authorization

Segment sends your user's API key with requests, and you can use it to authenticate requests. This is the API key *you* give to your users; it is not a Segment API key.

Segment sends the key in the `Authorization` header using the `Basic` authentication type. It is Base64 encoded with your user's API key as the username, and an empty password.

For example, if your user's API key was `segment`, Segment would Base64 encode the string `'segment:'` and prepend the string `'Basic '`. The colon is always present, even when the password is absent.

This would result in a final string of `'Basic c2VnbWVudDo='`. This is what is contained in the `Authorization` header. Like any `Authorization` header, you must decode the string when you receive it.

See the [headers](#) section for more details.

Custom Settings

All subscriptions have an API key setting by default, which Segment will send in the `Authorization` Header. To add more custom settings, go to the `Settings Builder` page under `App Info`.

The screenshot shows the Segment Developer Center interface for the 'Settings Builder'. The left sidebar contains navigation links: 'Basic Info', 'Launch Info', 'Settings Builder' (highlighted), 'Apps', 'Settings', 'OAuth', and 'Delete'. The main content area is titled 'Settings Builder' and includes a subtitle: 'Define the settings that will be displayed to a Segment user when they configure your app.' Below this, there are three settings cards: 1. 'API Key *' with an input field containing 'example input' and a label 'hello'. 2. 'publishable API key' with an input field containing 'example input'. 3. 'Custom Setting' with an input field containing 'example input' and a description 'Go to the settings page of our site and'. To the right of these cards is a form for creating a new setting, with fields for 'Setting Label' (containing 'Custom Setting'), 'Setting Key' (containing 'customSetting' and a 'Copy' button), 'Description' (containing 'Go to the settings page of our site and'), and 'Validation' (with checkboxes for 'Required' and 'Sensitive', and a note about sensitive data). At the bottom right, there is a 'Show Setting For' section with a checked checkbox for 'Subscription'.

Any custom settings you add will be sent in the custom header `x-Segment-Settings` (See the [headers](#) section for more details.)

Headers

Segment sends you the following HTTP headers with all requests:

HEADER	DESCRIPTION	EXAMPLE
Accept	Segment accepts any content type, but ignores responses unless this header is set to application/json .	Accept: */*
Authorization	Segment sends your user's API token in this header, with the Basic authentication type.	Authorization: Basic c2VnbWVudDo=
Cache-Control	Each request Segment sends is a new event. Segment does not expect your application to cache.	Cache-Control: no-cache
Connection	Segment uses HTTP/1.1's keep-alive functionality whenever possible, however this is optional.	Connection: Keep-Alive
Content-Length	Segment always sends you the length of the request in bytes.	Content-Length: 348
Content-Type	Segment indicates the type of data it sent you (this will always be JSON), along with Segment's vendor type.	Content-Type: application/json
User-Agent	Segment sends you this field every time.	User-Agent: Segment.io/1.0
X-Segment-Settings	Except for the API key (which is sent in the Authorization header), Segment will send the base 64 encoding of the rest of your custom settings encoded in this header.	X-Segment-Settings: eyJjdXN0b21TZXR0aw5nT251IjoiY3VzdG9tIHNldHRpbmcgdmFsdwUifQ==

Responding to Segment Data

Request Body

Segment's [Spec](#) standardizes the data that you can expect from Segment. You can choose to implement four types types of calls:

- Who is this? `.identify(userId, traits)`
- What are they doing? `.track(userId, event, properties)`
- Where are they doing it? `.page(userId, pageName, properties)`
- What group are they part of? `.group(userId, groupId, groupTraits)`

For example, you might implement the `.identify(userId, traits)` call to create contacts in an email marketing application. You can expect the following customer information as a JSON object in the call body:

```
{
  "anonymousId": "1234",
  "context": {
    "ip": "8.8.8.8",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/40.0.2214.115 Safari/537.36"
  },
  "messageId": "022bb90c-bbac-11e4-8dfc-aa07a5b093db",
  "receivedAt": "2015-02-23T22:28:55.387Z",
  "sentAt": "2015-02-23T22:28:55.111Z",
  "traits": {
    "name": "John Doe",
    "email": "john.doe@email.com",
    "plan": "premium",
    "logins": 5
  },
  "type": "identify",
  "userId": "5678",
  "version": "1.1"
}
```



The casing on these fields will vary by customer, so be ready to accept any casing.

Status Code

Segment uses standard HTTP status code conventions to help diagnose problems quickly and give better insight into how the destination is working.

Upon receiving data, your endpoint should reply with one of the following status codes:

CODE	REASON
200	You've accepted and successfully processed the message.
202	You've accepted the message, but have not yet processed it.
400	The message is malformed, or otherwise contains an error that is the client's fault.
401	The client's API key is malformed, has expired, or is otherwise no longer valid.
403	The client's API key is valid, but has been rejected due to inadequate permissions.
500	If you encounter an internal error when processing the message, reply with this code. (Hopefully you won't have to send too many of these.)
501	If Segment sends you an API call type (indicated by the type property included on all messages) you don't support, reply with this code. Read more about the API call types Segment supports in the Segment Spec documentation .
503	Send Segment this code when your endpoint is temporarily down for maintenance or otherwise not accepting messages. This helps Segment avoid dropping users' messages during your downtime.

Response Body

You can normally send back an empty body, but when sending back a 4xx- or 5xx-class error, you can optionally send Segment a diagnostic message that explains the error. This message is displayed to the user in the Segment debugger, and is be used in Segment's Event Delivery summaries.

Be sure to send JSON (and set your Content-Type header to `application/json`), and send your message in the `message` property.

Here's an example of a 401 response that helps a user track down why their calls aren't appearing in your tool's UI:

```
{
  "message": "API token expired"
}
```

Or, if your tool requires an email address in order to accept calls, use this example 400 reply:

```
{
  "message": "Missing email address"
}
```

Test

When testing your integration, proceed through two separate flows:

- 1. Test that your endpoint successfully ingests data in the way you would expect.

- 2. Mimic a user implementing your integration within their Segment workspace.

Your Endpoint

Test your code directly from the Developer Center UI. Use the **Send Test Event** button and review the test event to make sure your function works as expected.

The screenshot shows the Segment Developer Center interface. On the left is a sidebar with 'Apps' and 'Docs' tabs. The main area is titled 'Build' and contains a code editor with the following JavaScript code:

```
1 const endpoint = "https://en5ra9jwrh9u6.x.pipedream.net/"
2
3 async function track(event, settings) {
4   const res = await fetch(endpoint, {
5     body: JSON.stringify(event),
6     headers: new Headers({
7       "Authentication": `Bearer ${settings.apiKey}`,
8     }),
9     method: "post",
10  })
11
12  return await res.json() // or res.json() for JSON APIs
13 }
14
15 // identify demonstrates how to filter event data, e.g. for removing PII
16 // and how to enrich data using fetch
17 async function identify(event, settings) {
18   const blacklist = ['ssn', 'first_name', 'last_name', 'name', 'email'];
19   blacklist.forEach(i => delete event[i]);
20
21   const resp = await fetch('https://reqres.in/api/users/2');
22   const user = await resp.json();
23
24   event.traits = event.traits || {};
25   event.traits.avatar = user.data.avatar;
26
27   const res = await fetch(endpoint, {
28     body: JSON.stringify(event),
29     method: "post",
30   })
31
32   return await res.json()
33 }
34
35 // group demonstrates how to handle an invalid event
36 async function group(event, settings) {
```

On the right side of the code editor, there is a panel with the following elements:

- A 'Send Test Event' button and a 'View Output' button.
- A green box with the text 'Successfully executed Function'.
- Buttons for 'Callback Return' and 'Raw Log Output'.
- A JSON output box showing:

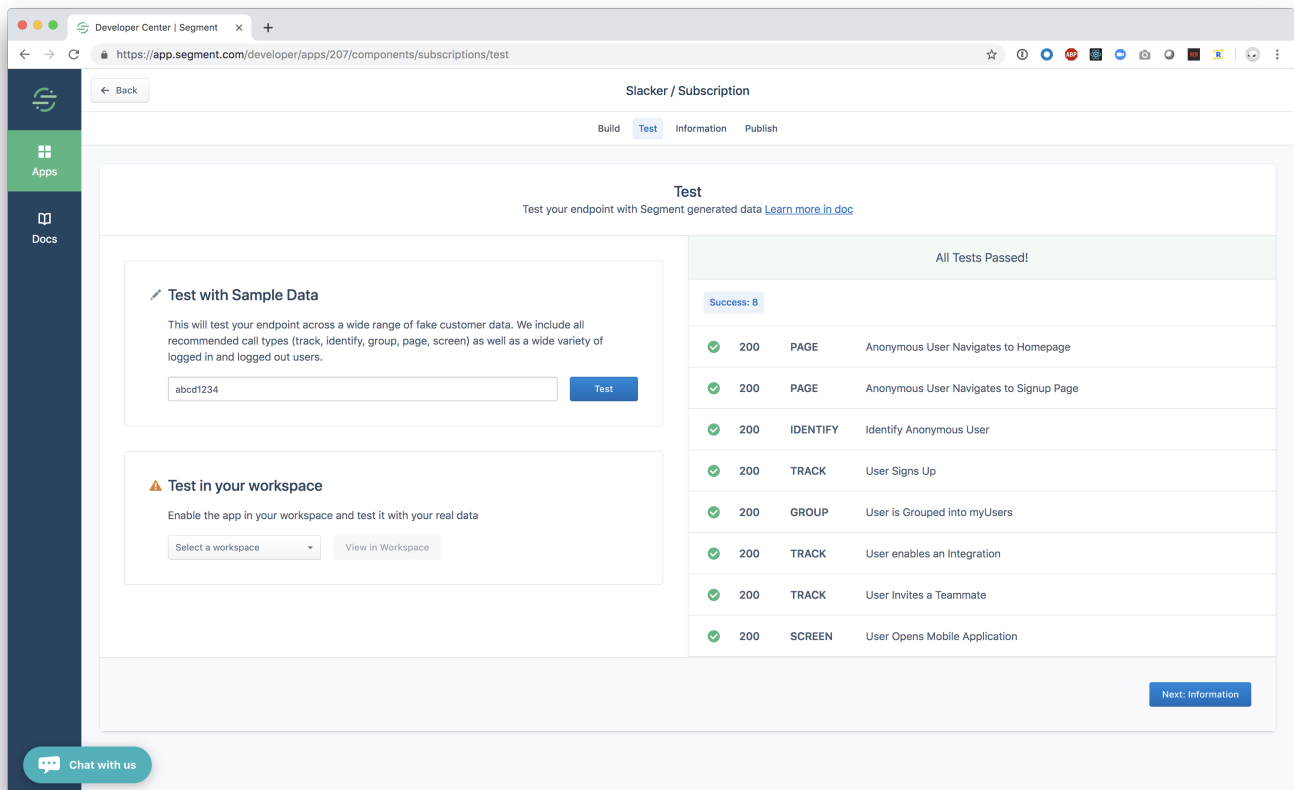
```
{
  "success": true
}
```
- A 'Send Test Event' button at the bottom.
- At the very bottom of the panel, there are 'Save' and 'Save & Next: Test' buttons.

In the debugger panel, check the two outputs. The **Callback Return** and the **Log Output**.

- 1. **Callback Return** - What data your function returned or error it threw.

- 2. **Log Output** - The raw log. Any messages to `console.log()` from your function appear here.

When your code is working with one event you can test it with a suite of more Segment events. Click **Save** and **Next: Test**, fill in an **API Key** and click **Test**. You will see the results of additional types of Segment data.



The User Flow

The ultimate goal is for Partners like yourself to create and publish high quality Destinations in [the Segment Catalog](#). Your Segment account doubles as a sandbox account to test your destination while you are still in a private “building” state.

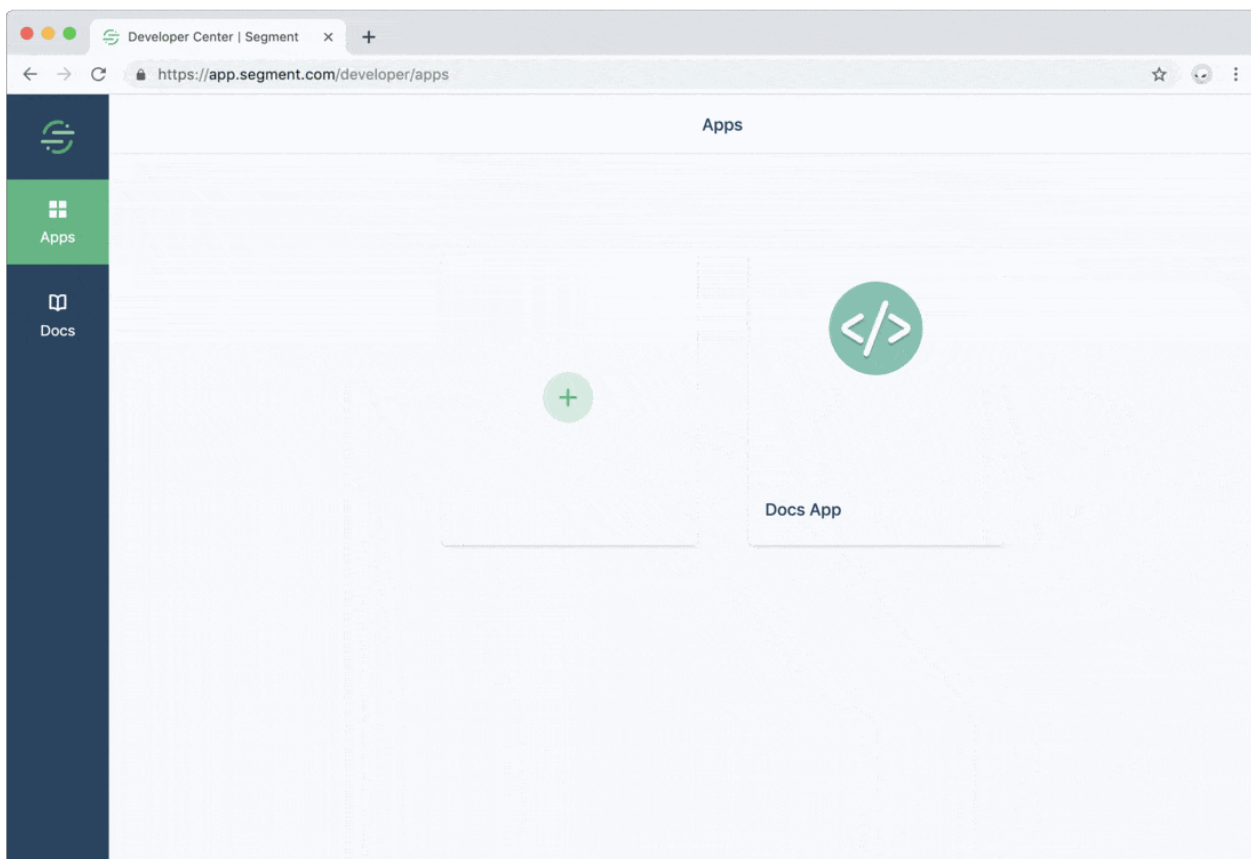
To test your Destination in the Catalog, click the “Test” tab in the Developer Center Component builder. In the “Test in your workspace” section, select your personal workspace and click “view”. This redirects to you a URL like <https://app.segment.com/WORKSPACE-SLUG/destinations/catalog/APP-SLUG>, which is your catalog entry.

From here, click “Configure App”, select a Source, and click “Confirm Source”. You can now configure your destination by setting the “API Key”, then clicking the toggle to enable the destination.

Next you can click the “Event Tester” tab to send data to your destination. Here you can see what requests Segment sends to your destination and introspect the response you are returning. Learn more about the event tester in the [Event Tester](#) docs.

Now you can use the JavaScript SDK in a browser to generate real analytics events.

Finally you should verify the data in your service.



Next Steps

Complete the remaining steps as outlined in the [Developer Center Overview](#)

This page was last modified: 12 Aug 2024

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

© 2025 Segment.io, Inc.

[Privacy](#)

[Terms](#)

[Website Data Collection Preferences](#)

