OpenSea API

OpenSea Stream API

Other

Stream API Overview

Overview

The OpenSea Stream API is a websocket-based service that enables developers to receive events as they occur, ensuring that your service has the most up-to-date details without the need to continuously poll for updates. With the Stream API, your service can subscribe to receive a range of different types of events, either globally or for specific collections that you're interested in.

The Stream API currently supports the following set of events:

- Item listed
  - an item listed for sale on the OpenSea marketplace
- Item sold
  - sale of an item on the OpenSea marketplace
- Item transferred
  - transfer of an item between wallets
- Item metadata update
  - update detected on the metadata provided in
- tokenURI
- for an item
- Item cancelled
  - cancellation of an order on the OpenSea marketplace
- Item received offer
  - offer received on an item in the OpenSea marketplace
- Item received bid
  - bid received on an item in the OpenSea marketplace
- Collection offer
  - offer on a collection
- Trait offer
  - offer on all items with a specific trait within a collection
- Order Invalidate
  - the order is now invalid and can not be filled
- Order Revalidate
  - the order which was previously invalidate can now once again be filled

What can you build with Stream API?

The real-time, push nature of the Stream API enables a variety of use cases that rely on dynamic, up-to-date information. Here are some examples of services that can be built using the Stream API:

- Push Notifications -
- Developers can create a push notifications service in their product that provides their users with timely updates as events happen on NFTs and in the OpenSea marketplace.
- Activity Feed
-
    - Developers can build an activity feed into their product that provides their users with a timeline of events as they happen on NFTs and in the OpenSea marketplace.
- Real-Time Data Monitoring
-
    - Developers can create real-time dashboards that enable people to visualize and track key trends, metrics, etc as part of an analytics service.
- Ownership Changes -
- Developers can automatically reflect any changes in ownership of an NFT in a person's connected wallet.
- Metadata Updates -
- Developers can ensure that NFT metadata is always up-to-date, immediately reflecting any changes that take place.

Getting Started

We've created a JavaScript SDK to help developers manage connections and individual subscriptions to the Stream API and its various events. To get started, we recommend that you install ourStream SDK and following the setup instructions in itsreadme .

Client Setup

Our SDK includes a client that manages authentication, connections and event subscriptions on the Stream API. To create a new client in your project, you can create it as follows:

JSX import { OpenSeaStreamClient } from '@opensea/stream-js';

const client = new OpenSeaStreamClient({ token: 'openseaApiKey' }); Authentication with the Stream API uses the same API key as our other APIs. If you don't have an API key, pleaserequest one .

The client also supports the following optional configuration parameters:

Parameter Description Values network The network for which events will be returned by the Stream API. Please see below for blockchain coverage. Network.MAINNET (Default), Network.TESTNET onError A callback to handle errors from the client. Specified callback handler in your projects. Defaults to console.error . logLevel Specifies the scope of logging on the client. LogLevel.DEBUG , LogLevel.INFO (Default), LogLevel.WARN , LogLevel.ERROR The Stream API is available for both Mainnet and Testnet networks:

Network Endpoint Supported Blockchains Mainnet wss://stream.openseabeta.com/socket SeeSupported Chains - Mainnets Testnet wss://testnets-stream.openseabeta.com/socket SeeSupported Chains - Testnets

Connecting & Subscribing to Events

The client will automatically connect to the socket as soon as you setup your first subscription to one of the event types listed below. If you'd prefer to manually connect to the socket, we provide a function to explicitly establish the connection:

JSX client.connect(); The following event subscriptions are available on the Stream API:

Event Client Subscription Event Schema Item listed onItemListed Item Listed Payload Schema Item sold onItemSold Item Sold Payload Schema Item transferred onItemTransferred Item Transferred Payload Schema Item metadata update onItemMetadataUpdated Item Metadata Update Payload Schema Item cancelled onItemCancelled Item Cancelled Update Schema Item received offer onItemReceivedOffer Item Received Offer Schema Item received bid onItemReceivedBid Item Received Bid Schema Collection offer onCollectionOffer Collection Offer Schema Trait offer onTraitOffer Trait Offer Schema Order Invalidate onOrderInvalidate Order Invalidate Schema Order Revalidate onOrderRevalidate Order Revalidate Schema Each subscription function includes a collectionSlug parameter which can be used to subscribe to an event from a specific collection. Here's how to subscribe to receive all new listings of items from a specific collection:

JSX client.onItemListed('collection-slug', (event) => { // handle event }); If you'd like to subscribe to events across all collections, you can use * wildcard for the collectionSlug parameter. Here's how to subscriber to all new offers across all collections:

JSX client.onItemReceivedOffer('*', (event) => { // handle event });

Unsubscribing from Events

Each subscription method returns a callback function that is used to unsubscribe from a set of events when invoked.

JSX const unsubscribe = client.onItemMetadataUpdated('collection-slug', noop);

unsubscribe();

Disconnecting

To disconnect your client from the socket completely, simply call its disconnect() method.

JSX client.disconnect();

FAQs

Do I need an API key for Mainnet and Testnet to use the Stream API?

If you want to use the API in Mainnet, an API key is required. If you don't have an API keysign-up through the developer portal.

Do the events received from my subscriptions count towards any rate limits on my API key?

No, events are not counted towards any API rate limits.

What is the typical streaming rate that I should expect from the Stream API?

The streaming rate depends on a range of factors - from the amount of collections that you're monitoring to the type and amount of events that you're subscribed to. For instance, if you're subscribed to receive bid events across all collections on OpenSea, you'll be receiving messages at a significantly higher rate than you will for a subscription to order cancellations on a small collection.

Should I expect that some events can be received out of order?

You should be prepared to handle receipt of events out of order as we don't guarantee delivery of events in the order that they occur. Payloads include the event_timestamp field, which represents the time at which the event occurred and is the most definitive resource in determining order. We also include a sent_at field that refers to the time at which we sent the message out through the websocket.

Is it possible for some events to be missing?

The Stream API is a best-effort delivery messaging system and messages that are not received due to connection errors will not be re-sent. So it's possible that there can be missing messages if the socket connection is unstable.

What blockchains does the Stream API support?

All supported chains work with the Stream API except for Solana. SeeSupported Chains - Mainnets andSupported Chains - Testnets .