# OpenSea API

- Get Item Offers get
- Get Listings get
- Get Order get
- Get Trait Offers get

## OpenSea Stream API

- Stream API Overview
- Stream API Event Example Payloads
- Using Stream API without SDK

## Other

- Supported Chains

# OpenAPI Definition

The OpenAPI Definition for V2 OpenSea API Endpoints

The API is now OpenAPI compliant. This allows users to download the OpenAPI spec to import it into a Postman Collections or generate a client library in many popular languages/frameworks including C#, Groovy, Java, Python, Rust, Swift, etc.

OpenAPI Definition { "openapi": "3.0.3", "info": { "title": "OpenSea API", "version": "2.0.0", "description": "The API for OpenSea", "contact": { "name": "OpenSea", "url": "https://www.opensea.io", "email": "[email protected] " } }, "paths": { "/api/v2/accounts/{address}": { "get": { "operationId": "get_account", "description": "Get an OpenSea Account Profile including details such as bio, social media usernames, and profile image.", "summary": "Get Account", "parameters": [ { "in": "path", "name": "address", "schema": { "type": "string" }, "description": "The unique public blockchain identifier for the contract or wallet.", "required": true } ], "tags": [ "NFT Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/DetailedAccountDataModel" } } }, "description": "" }, "400": { "description": "Returned if the Account does not exist or can not be parsed." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/chain/{chain}/account/{address}/nfts": { "get": { "operationId": "list_nfts_by_account", "description": "Get NFTs owned by a given account address.", "summary": "Get NFTs (by account)", "parameters": [ { "in": "path", "name": "address", "schema": { "type": "string" }, "description": "The unique public blockchain identifier for the contract or wallet.", "required": true }, { "in": "path", "name": "chain", "schema": { "type": "string", "enum": [ "arbitrum", "arbitrum_goerli", "arbitrum_nova", "avalanche", "avalanche_fuji", "baobab", "base", "base_goerli", "bsc", "bsctestnet", "ethereum", "goerli", "klaytn", "matic", "mumbai", "optimism", "optimism_goerli", "sepolia", "solana", "soldev", "zora", "zora_testnet" ] }, "description": "The blockchain on which to filter the results.", "required": true }, { "in": "query", "name": "collection", "schema": { "type": "string" }, "description": "Unique string to identify a collection on OpenSea. This can be found by visiting the collection on the OpenSea website and noting the last path parameter." }, { "in": "query", "name": "limit", "schema": { "type": "integer" }, "description": "The number of NFTs to return. Must be between 1 and 50. Default: 50" }, { "in": "query", "name": "next", "schema": { "type": "string" }, "description": "The cursor for the next page of results. This is returned from a previous request." } ], "tags": [ "NFT Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/ListNftsResponse" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/chain/{chain}/contract/{address}": { "get": { "operationId": "get_contract", "description": "Get a smart contract for a given chain and address.", "summary": "Get Contract", "parameters": [ { "in": "path", "name": "address", "schema": { "type": "string" }, "description": "The unique public blockchain identifier for the contract or wallet.", "required": true }, { "in": "path", "name": "chain", "schema": { "type": "string", "enum": [ "arbitrum", "arbitrum_goerli", "arbitrum_nova", "avalanche", "avalanche_fuji", "baobab", "base", "base_goerli", "bsc", "bsctestnet", "ethereum", "goerli", "klaytn", "matic", "mumbai", "optimism", "optimism_goerli", "sepolia", "solana", "soldev", "zora", "zora_testnet" ] }, "description": "The blockchain on which to filter the results.", "required": true } ], "tags": [ "NFT Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/ListNftsResponse" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/chain/{chain}/contract/{address}/nfts": { "get": { "operationId": "list_nfts_by_contract", "description": "Get multiple NFTs for a smart contract.", "summary": "Get NFTs (by contract)", "parameters": [ { "in": "path", "name": "address", "schema": { "type": "string" }, "description": "The unique public blockchain identifier for the contract or wallet.", "required": true }, { "in": "path", "name": "chain", "schema": { "type": "string", "enum": [ "arbitrum", "arbitrum_goerli", "arbitrum_nova", "avalanche", "avalanche_fuji", "baobab", "base", "base_goerli", "bsc", "bsctestnet", "ethereum", "goerli", "klaytn", "matic", "mumbai", "optimism", "optimism_goerli", "sepolia", "solana", "soldev", "zora", "zora_testnet" ] }, "description": "The blockchain on which to filter the results.", "required": true }, { "in": "query", "name": "limit", "schema": { "type": "integer" }, "description": "The number of NFTs to return. Must be between 1 and 50. Default: 50" }, { "in": "query", "name": "next", "schema": { "type": "string" }, "description": "The cursor for the next page of results. This is returned from a previous request." } ], "tags": [ "NFT

Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/ListNftsResponse" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/chain/{chain}/contract/{address}/nfts/{identifier}": { "get": { "operationId": "get_nft", "description": "Get metadata, traits, ownership information, and rarity for a single NFT.", "summary": "Get an NFT", "parameters": [ { "in": "path", "name": "address", "schema": { "type": "string" }, "description": "The unique public blockchain identifier for the contract or wallet.", "required": true }, { "in": "path", "name": "chain", "schema": { "type": "string", "enum": [ "arbitrum", "arbitrum_goerli", "arbitrum_nova", "avalanche", "avalanche_fuji", "baobab", "base", "base_goerli", "bsc", "bsctestnet", "ethereum", "goerli", "klaytn", "matic", "mumbai", "optimism", "optimism_goerli", "sepolia", "solana", "soldev", "zora", "zora_testnet" ] }, "description": "The blockchain on which to filter the results.", "required": true }, { "in": "path", "name": "identifier", "schema": { "type": "string" }, "description": "The NFT token id.", "required": true } ], "tags": [ "NFT Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/GetNftResponse" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/chain/{chain}/contract/{address}/nfts/{identifier}/refresh": { "post": { "operationId": "refresh_nft", "description": "Refresh metadata for a single NFT.", "summary": "Refresh NFT Metadata", "parameters": [ { "in": "path", "name": "address", "schema": { "type": "string" }, "description": "The unique public blockchain identifier for the contract or wallet.", "required": true }, { "in": "path", "name": "chain", "schema": { "type": "string", "enum": [ "arbitrum", "arbitrum_goerli", "arbitrum_nova", "avalanche", "avalanche_fuji", "baobab", "base", "base_goerli", "bsc", "bsctestnet", "ethereum", "goerli", "klaytn", "matic", "mumbai", "optimism", "optimism_goerli", "sepolia", "solana", "soldev", "zora", "zora_testnet" ] }, "description": "The blockchain on which to filter the results.", "required": true }, { "in": "path", "name": "identifier", "schema": { "type": "string" }, "description": "The NFT token id.", "required": true } ], "tags": [ "NFT Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "description": "Metadata has been succesfully queued for refresh." }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/collection/{collection_slug}/nfts": { "get": { "operationId": "list_nfts_by_collection", "description": "Get multiple NFTs for a collection.", "summary": "Get NFTs (by collection)", "parameters": [ { "in": "path", "name": "collection_slug", "schema": { "type": "string" }, "description": "Unique string to identify a collection on OpenSea. This can be found by visiting the collection on the OpenSea website and noting the last path parameter.", "required": true }, { "in": "query", "name": "limit", "schema": { "type": "integer" }, "description": "The number of NFTs to return. Must be between 1 and 50. Default: 50" }, { "in": "query", "name": "next", "schema": { "type": "string" }, "description": "The cursor for the next page of results. This is returned from a previous request." } ], "tags": [ "NFT Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/ListNftsResponse" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/collections": { "get": { "operationId": "list_collections", "description": "Get a list of OpenSea collections.", "summary": "Get Collections", "parameters": [ { "in": "query", "name": "chain_identifier", "schema": { "type": "string" }, "description": "The blockchain on which to filter the results" }, { "in": "query", "name": "include_hidden", "schema": { "type": "boolean" }, "description": "If true, will return hidden collections. Default: false" }, { "in": "query", "name": "limit", "schema": { "type": "integer" }, "description": "The number of collections to return. Must be between 1 and 100. Default: 100" }, { "in": "query", "name": "next", "schema": { "type": "string" }, "description": "The cursor for the next page of results. This is returned from a previous request." } ], "tags": [ "NFT Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/ListCollectionsResponse" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/collections/{collection_slug}": { "get": { "operationId": "get_collection", "description": "Get a single collection including details such as fees, traits, and links.", "summary": "Get a Collection", "parameters": [ { "in": "path", "name": "collection_slug", "schema": { "type": "string" }, "description": "Unique string to identify a collection on OpenSea. This can be found by visiting the collection on the OpenSea website and noting the last path parameter.", "required": true } ], "tags": [ "NFT Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/DetailedCollectionModel" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/collections/{collection_slug}/stats": { "get": { "operationId": "get_collection_stats", "description": "Get stats for a single collection.", "summary": "Get Collection Stats", "parameters": [ { "in": "path", "name": "collection_slug", "schema": { "type": "string" }, "description": "Unique string to identify a collection on OpenSea. This can be found by visiting the collection on the OpenSea website and noting the last path parameter.", "required": true } ], "tags": [ "Analytics Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/GetCollectionStatsResponse" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/events/accounts/{address}": { "get": { "operationId": "list_events_by_account", "description": "Get a list of events for an account. The list will be paginated and include up to 100 events per page.", "summary": "Get Events (by account)", "parameters": [ { "in": "path", "name": "address", "schema": { "type": "string" }, "description": "The unique public blockchain identifier for the contract or wallet.", "required": true }, { "in": "query", "name": "after", "schema": { "type": "number" }, "description": "Filter to only include events that occurred at or after the given timestamp. The Unix epoch timstamp must be in seconds" }, { "in": "query", "name": "before", "schema": { "type": "number" }, "description": "Filter to only include events that occurred before the given timestamp. The Unix epoch timstamp must be in seconds." }, { "in": "query", "name": "chain", "schema": { "type": "string", "enum": [ "arbitrum", "arbitrum_goerli", "arbitrum_nova", "avalanche", "avalanche_fuji", "baobab", "base", "base_goerli", "bsc", "bsctestnet", "ethereum", "goerli", "klaytn", "matic", "mumbai", "optimism", "optimism_goerli", "sepolia", "solana", "soldev", "zora", "zora_testnet" ] }, "description": "The blockchain on which to filter the results." }, { "in": "query", "name": "event_type", "schema": { "type": "array", "items": { "type": "string", "enum": [ "all", "cancel", "order",

"redemption", "sale", "transfer" ] } }, "description": "The type of event to filter by. If not provided, only sales will be returned." }, { "in": "query", "name": "next", "schema": { "type": "string" }, "description": "The cursor for the next page of results. This is returned from a previous request." } ], "tags": [ "Analytics Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/ListEventsResponse" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/events/chain/{chain}/contract/{address}/nfts/{identifier}": { "get": { "operationId": "list_events_by_nft", "description": "Get a list of events for a single NFT. The list will be paginated and include up to 100 events per page.", "summary": "Get Events (by NFT)", "parameters": [ { "in": "path", "name": "address", "schema": { "type": "string" }, "description": "The unique public blockchain identifier for the contract or wallet.", "required": true }, { "in": "query", "name": "after", "schema": { "type": "number" }, "description": "Filter to only include events that occurred at or after the given timestamp. The Unix epoch timstamp must be in seconds" }, { "in": "query", "name": "before", "schema": { "type": "number" }, "description": "Filter to only include events that occurred before the given timestamp. The Unix epoch timstamp must be in seconds." }, { "in": "path", "name": "chain", "schema": { "type": "string", "enum": [ "arbitrum", "arbitrum_goerli", "arbitrum_nova", "avalanche", "avalanche_fuji", "baobab", "base", "base_goerli", "bsc", "bsctestnet", "ethereum", "goerli", "klaytn", "matic", "mumbai", "optimism", "optimism_goerli", "sepolia", "solana", "soldev", "zora", "zora_testnet" ] }, "description": "The blockchain on which to filter the results.", "required": true }, { "in": "query", "name": "event_type", "schema": { "type": "array", "items": { "type": "string", "enum": [ "all", "cancel", "order", "redemption", "sale", "transfer" ] } }, "description": "The type of event to filter by. If not provided, only sales will be returned." }, { "in": "path", "name": "identifier", "schema": { "type": "string" }, "description": "The NFT token id.", "required": true }, { "in": "query", "name": "next", "schema": { "type": "string" }, "description": "The cursor for the next page of results. This is returned from a previous request." } ], "tags": [ "Analytics Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/ListEventsResponse" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/events/collection/{collection_slug}": { "get": { "operationId": "list_events_by_collection", "description": "Get a list of events for a collection. The list will be paginated and include up to 100 events per page.", "summary": "Get Events (by collection)", "parameters": [ { "in": "query", "name": "after", "schema": { "type": "number" }, "description": "Filter to only include events that occurred at or after the given timestamp. The Unix epoch timstamp must be in seconds" }, { "in": "query", "name": "before", "schema": { "type": "number" }, "description": "Filter to only include events that occurred before the given timestamp. The Unix epoch timstamp must be in seconds." }, { "in": "path", "name": "collection_slug", "schema": { "type": "string" }, "description": "Unique string to identify a collection on OpenSea. This can be found by visiting the collection on the OpenSea website and noting the last path parameter.", "required": true }, { "in": "query", "name": "event_type", "schema": { "type": "array", "items": { "type": "string", "enum": [ "all", "cancel", "order", "redemption", "sale", "transfer" ] } }, "description": "The type of event to filter by. If not provided, only sales will be returned." }, { "in": "query", "name": "next", "schema": { "type": "string" }, "description": "The cursor for the next page of results. This is returned from a previous request." } ], "tags": [ "Analytics Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/ListEventsResponse" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/listings/collection/{collection_slug}/all": { "get": { "operationId": "get_all_listings_on_collection_v2", "description": "Get all active, valid listings for a single collection.", "summary": "Get All Listings (by collection)", "parameters": [ { "in": "path", "name": "collection_slug", "schema": { "type": "string" }, "description": "Unique string to identify a collection on OpenSea. This can be found by visiting the collection on the OpenSea website and noting the last path parameter.", "required": true }, { "in": "query", "name": "limit", "schema": { "type": "integer" }, "description": "The number of listings to return. Must be between 1 and 100. Default: 100" }, { "in": "query", "name": "next", "schema": { "type": "string" }, "description": "The cursor for the next page of results. This is returned from a previous request." } ], "tags": [ "Transaction Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/PaginatedListingList" } } }, "description": "" }, "400": { "description": "The collection requested does not exist.\nThe query parameters can not be parsed." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/listings/fulfillment_data": { "post": { "operationId": "generate_listing_fulfillment_data_v2", "description": "Retrieve all the information, including signatures, needed to fulfill a listing directly onchain.", "summary": "Fulfill a Listing", "tags": [ "Transaction Endpoints" ], "requestBody": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/GenerateListingFulfillmentInput" } }, "application/x-www-form-urlencoded": { "schema": { "ref": "#/components/schemas/GenerateListingFulfillmentInput" } }, "multipart/form-data": { "schema": { "ref": "#/components/schemas/GenerateListingFulfillmentInput" } } }, "required": true }, "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/FulfillmentOutput" } } }, "description": "" }, "400": { "description": "The request is invalid\nThe order_hash does not exist\nThe chain is not an EVM Chain\nThe protocol_address is not a supported Seaport contract\nFor other error reasons, see the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/offers": { "post": { "operationId": "post_criteria_offer_v2", "description": "Create a criteria offer to purchase any NFT in a collection or which matches the specified trait.", "summary": "Create Criteria Offer", "tags": [ "Transaction Endpoints" ], "requestBody": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/PostCriteriaOfferInput" } }, "application/x-www-form-urlencoded": { "schema": { "ref": "#/components/schemas/PostCriteriaOfferInput" } }, "multipart/form-data": { "schema": { "ref": "#/components/schemas/PostCriteriaOfferInput" } } }, "required": true }, "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/Offer" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/offers/build": { "post": { "operationId": "build_offer_v2", "description": "Build a portion of a criteria offer including the merkle tree needed to post an offer.", "summary": "Build an Offer", "tags": [ "Transaction Endpoints" ], "requestBody": { "content": { "application/json": { "schema": { "ref":

"#/components/schemas/BuildOfferInput" } }, "application/x-www-form-urlencoded": { "schema": { "ref": "#/components/schemas/BuildOfferInput" } }, "multipart/form-data": { "schema": { "ref": "#/components/schemas/BuildOfferInput" } } }, "required": true }, "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/BuildOffer" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/offers/collection/{collection_slug}": { "get": { "operationId": "get_collection_offers_v2", "description": "Get the active, valid collection offers for the specified collection.", "summary": "Get Collection Offers", "parameters": [ { "in": "path", "name": "collection_slug", "schema": { "type": "string" }, "description": "Unique string to identify a collection on OpenSea. This can be found by visiting the collection on the OpenSea website and noting the last path parameter.", "required": true } ], "tags": [ "Transaction Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/OfferList" } } }, "description": "" }, "404": { "description": "Returned when the collection does not exist." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/offers/collection/{collection_slug}/all": { "get": { "operationId": "get_all_offers_on_collection_v2", "description": "Get all active, valid offers for the specified collection. This includes individual and criteria offers.", "summary": "Get All Offers (by collection)", "parameters": [ { "in": "path", "name": "collection_slug", "schema": { "type": "string" }, "description": "Unique string to identify a collection on OpenSea. This can be found by visiting the collection on the OpenSea website and noting the last path parameter.", "required": true }, { "in": "query", "name": "limit", "schema": { "type": "integer" }, "description": "The number of offers to return. Must be between 1 and 100. Default: 100" }, { "in": "query", "name": "next", "schema": { "type": "string" }, "description": "The cursor for the next page of results. This is returned from a previous request." } ], "tags": [ "Transaction Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/PaginatedOfferList" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "404": { "description": "Returned when the collection does not exist." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/offers/collection/{collection_slug}/traits": { "get": { "operationId": "get_trait_offers_v2", "description": "Get the active, valid trait offers for the specified collection.", "summary": "Get Trait Offers", "parameters": [ { "in": "path", "name": "collection_slug", "schema": { "type": "string" }, "description": "Unique string to identify a collection on OpenSea. This can be found by visiting the collection on the OpenSea website and noting the last path parameter.", "required": true }, { "in": "query", "name": "float_value", "schema": { "type": "number", "format": "float" }, "description": "The value of the trait (e.g. 0.5). This is only used for decimal-based numeric traits to ensure it is parsed correctly." }, { "in": "query", "name": "int_value", "schema": { "type": "integer" }, "description": "The value of the trait (e.g. 10). This is only used for numeric traits to ensure it is parsed correctly." }, { "in": "query", "name": "type", "schema": { "type": "string" }, "description": "The name of the trait (e.g. 'Background')" }, { "in": "query", "name": "value", "schema": { "type": "string" }, "description": "The value of the trait (e.g. 'Red')" } ], "tags": [ "Transaction Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/OfferList" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "404": { "description": "Returned when the collection does not exist." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/offers/fulfillment_data": { "post": { "operationId": "generate_offer_fulfillment_data_v2", "description": "Retrieve all the information, including signatures, needed to fulfill an offer directly onchain.", "summary": "Fullfill an Offer", "tags": [ "Transaction Endpoints" ], "requestBody": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/GenerateOfferFulfillmentInput" } }, "application/x-www-form-urlencoded": { "schema": { "ref": "#/components/schemas/GenerateOfferFulfillmentInput" } }, "multipart/form-data": { "schema": { "ref": "#/components/schemas/GenerateOfferFulfillmentInput" } } }, "required": true }, "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/FulfillmentOutput" } } }, "description": "" }, "400": { "description": "The request is invalid\nThe order_hash does not exist\nThe chain is not an EVM Chain\nThe protocol_address is not a supported Seaport contract\nFor other error reasons, see the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/orders/{chain}/{protocol}/listings": { "get": { "operationId": "get_listings", "description": "Get the complete set of active, valid listings.", "summary": "Get Listings", "parameters": [ { "in": "query", "name": "asset_contract_address", "schema": { "type": "string" }, "description": "Filter results by the contract address for NFT(s). \n NOTE: If used, token_ids or token_id is required." }, { "in": "query", "name": "bundled", "schema": { "type": "boolean" }, "description": "Restricts results to only include orders that are bundles of NFTs. Default: false" }, { "in": "path", "name": "chain", "schema": { "type": "string", "enum": [ "arbitrum", "arbitrum_goerli", "arbitrum_nova", "avalanche", "avalanche_fuji", "baobab", "base", "base_goerli", "bsc", "bsctestnet", "ethereum", "goerli", "klaytn", "matic", "mumbai", "optimism", "optimism_goerli", "sepolia", "solana", "soldev", "zora", "zora_testnet" ] }, "description": "The blockchain on which to filter the results.", "required": true }, { "in": "query", "name": "cursor", "schema": { "type": "string" }, "description": "The cursor for the next page of results. This is returned from a previous request." }, { "in": "query", "name": "limit", "schema": { "type": "integer" }, "description": "The number of orders to return. Must be between 1 and 50. Default: 20" }, { "in": "query", "name": "listed_after", "schema": { "type": "string", "format": "date-time" }, "description": "Filter to only include orders that were listed after the given timestamp. This is a Unix epoch timestamp in seconds." }, { "in": "query", "name": "listed_before", "schema": { "type": "string", "format": "date-time" }, "description": "Filter to only include orders that were listed before the given timestamp. This is a Unix epoch timestamp in seconds." }, { "in": "query", "name": "maker", "schema": { "type": "string" }, "description": "Filter results by the order maker's wallet address." }, { "in": "query", "name": "order_by", "schema": { "type": "string", "enum": [ "created_date", "eth_price" ] }, "description": "The order in which to sort the results. Default: created_date \n NOTE: If eth_price is used, asset_contract_address and token_id are required." }, { "in": "query", "name": "order_direction", "schema": { "type": "string", "enum": [ "asc", "desc" ] }, "description": "The direction in which to sort the results. Default: desc" }, { "in": "query", "name": "payment_token_address", "schema": { "type": "string" }, "description": "Payment Token Address to filter results. This ensures all returned orders are listed in a single currency." }, { "in": "path", "name": "protocol", "schema": { "type": "string", "enum": [ "seaport" ] }, "description": "The token settlement protocol to use in the request.", "required": true }, { "in": "query", "name": "taker",

"schema": { "type": "string" }, "description": "Filter results by the order taker's wallet address." }, { "in": "query", "name": "token_ids", "schema": { "type": "integer" }, "description": "An array of token IDs to search for (e.g. ?token_ids=1&token_ids=209). This endpoint will return a list of orders with token_id matching any of the IDs in this array. \n NOTE: If used, asset_contract_address is required." } ], "tags": [ "Transaction Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/GetListingsResponse" } } }, "description": "" }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } }, "post": { "operationId": "post_listing", "description": "List a single NFT (ERC721 or ERC1155) for sale on the OpenSea marketplace.", "summary": "Create Listing", "parameters": [ { "in": "path", "name": "chain", "schema": { "type": "string", "enum": [ "arbitrum", "arbitrum_goerli", "arbitrum_nova", "avalanche", "avalanche_fuji", "baobab", "base", "base_goerli", "bsc", "bsctestnet", "ethereum", "goerli", "klaytn", "matic", "mumbai", "optimism", "optimism_goerli", "sepolia", "solana", "soldev", "zora", "zora_testnet" ] }, "description": "The blockchain on which to filter the results.", "required": true }, { "in": "path", "name": "protocol", "schema": { "type": "string", "enum": [ "seaport" ] }, "description": "The token settlement protocol to use in the request.", "required": true } ], "tags": [ "Transaction Endpoints" ], "requestBody": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/OrderInputWithProtocol" } }, "application/x-www-form-urlencoded": { "schema": { "ref": "#/components/schemas/OrderInputWithProtocol" } }, "multipart/form-data": { "schema": { "ref": "#/components/schemas/OrderInputWithProtocol" } } }, "required": true }, "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/CreateListingResponse" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/orders/{chain}/{protocol}/offers": { "get": { "operationId": "get_offers", "description": "Get the active, valid individual offers. This does not include criteria offers.", "summary": "Get Individual Offers", "parameters": [ { "in": "query", "name": "asset_contract_address", "schema": { "type": "string" }, "description": "Filter results by the contract address for NFT(s). \n NOTE: If used, token_ids or token_id is required." }, { "in": "query", "name": "bundled", "schema": { "type": "boolean" }, "description": "Restricts results to only include orders that are bundles of NFTs. Default: false" }, { "in": "path", "name": "chain", "schema": { "type": "string", "enum": [ "arbitrum", "arbitrum_goerli", "arbitrum_nova", "avalanche", "avalanche_fuji", "baobab", "base", "base_goerli", "bsc", "bsctestnet", "ethereum", "goerli", "klaytn", "matic", "mumbai", "optimism", "optimism_goerli", "sepolia", "solana", "soldev", "zora", "zora_testnet" ] }, "description": "The blockchain on which to filter the results.", "required": true }, { "in": "query", "name": "cursor", "schema": { "type": "string" }, "description": "The cursor for the next page of results. This is returned from a previous request." }, { "in": "query", "name": "limit", "schema": { "type": "integer" }, "description": "The number of orders to return. Must be between 1 and 50. Default: 20" }, { "in": "query", "name": "listed_after", "schema": { "type": "string", "format": "date-time" }, "description": "Filter to only include orders that were listed after the given timestamp. This is a Unix epoch timestamp in seconds." }, { "in": "query", "name": "listed_before", "schema": { "type": "string", "format": "date-time" }, "description": "Filter to only include orders that were listed before the given timestamp. This is a Unix epoch timestamp in seconds." }, { "in": "query", "name": "maker", "schema": { "type": "string" }, "description": "Filter results by the order maker's wallet address." }, { "in": "query", "name": "order_by", "schema": { "type": "string", "enum": [ "created_date", "eth_price" ] }, "description": "The order in which to sort the results. Default: created_date \n NOTE: If eth_price is used, asset_contract_address and token_id are required." }, { "in": "query", "name": "order_direction", "schema": { "type": "string", "enum": [ "asc", "desc" ] }, "description": "The direction in which to sort the results. Default: desc" }, { "in": "query", "name": "payment_token_address", "schema": { "type": "string" }, "description": "Payment Token Address to filter results. This ensures all returned orders are listed in a single currency." }, { "in": "path", "name": "protocol", "schema": { "type": "string", "enum": [ "seaport" ] }, "description": "The token settlement protocol to use in the request.", "required": true }, { "in": "query", "name": "taker", "schema": { "type": "string" }, "description": "Filter results by the order taker's wallet address." }, { "in": "query", "name": "token_ids", "schema": { "type": "integer" }, "description": "An array of token IDs to search for (e.g. ?token_ids=1&token_ids=209). This endpoint will return a list of orders with token_id matching any of the IDs in this array. \n NOTE: If used, asset_contract_address is required." } ], "tags": [ "Transaction Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/GetOfferResponse" } } }, "description": "" }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } }, "post": { "operationId": "post_offer", "description": "Create an offer to purchase a single NFT (ERC721 or ERC1155).", "summary": "Create Individual Offer", "parameters": [ { "in": "path", "name": "chain", "schema": { "type": "string", "enum": [ "arbitrum", "arbitrum_goerli", "arbitrum_nova", "avalanche", "avalanche_fuji", "baobab", "base", "base_goerli", "bsc", "bsctestnet", "ethereum", "goerli", "klaytn", "matic", "mumbai", "optimism", "optimism_goerli", "sepolia", "solana", "soldev", "zora", "zora_testnet" ] }, "description": "The blockchain on which to filter the results.", "required": true }, { "in": "path", "name": "protocol", "schema": { "type": "string", "enum": [ "seaport" ] }, "description": "The token settlement protocol to use in the request.", "required": true } ], "tags": [ "Transaction Endpoints" ], "requestBody": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/OrderInputWithProtocol" } }, "application/x-www-form-urlencoded": { "schema": { "ref": "#/components/schemas/OrderInputWithProtocol" } }, "multipart/form-data": { "schema": { "ref": "#/components/schemas/OrderInputWithProtocol" } } }, "required": true }, "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/CreateOfferResponse" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } },
"/api/v2/orders/chain/{chain}/protocol/{protocol_address}/{order_hash}": { "get": { "operationId": "get_order", "description": "Get a single order, offer or listing, by its order hash. Protocol and Chain are required to prevent hash collisions.", "summary": "Get Order", "parameters": [ { "in": "path", "name": "chain", "schema": { "type": "string", "enum": [ "arbitrum", "arbitrum_goerli", "arbitrum_nova", "avalanche", "avalanche_fuji", "baobab", "base", "base_goerli", "bsc", "bsctestnet", "ethereum", "goerli", "klaytn", "matic", "mumbai", "optimism", "optimism_goerli", "sepolia", "solana", "soldev", "zora", "zora_testnet" ] }, "description": "The blockchain on which to filter the results.", "required": true }, { "in": "path", "name": "order_hash", "schema": { "type": "string" }, "description": "The hash of the order to retrieve.", "required": true }, { "in": "path", "name": "protocol_address", "schema": { "type": "string", "enum": [ [ "0x00000000000000adc04c56bf30ac9d3c0aaf14dc" ] ] },

"description": "The contract address of the protocol to use in the request.", "required": true } ], "tags": [ "Transaction Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/GetOrderResult" } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } }, "/api/v2/traits/{collection_slug}": { "get": { "operationId": "get_traits", "description": "Get the traits in a collection.", "summary": "Get Traits", "parameters": [ { "in": "path", "name": "collection_slug", "schema": { "type": "string" }, "description": "Unique string to identify a collection on OpenSea. This can be found by visiting the collection on the OpenSea website and noting the last path parameter.", "required": true } ], "tags": [ "NFT Endpoints" ], "security": [ { "OpenSeaAuth": [] } ], "responses": { "200": { "content": { "application/json": { "schema": { "ref": "#/components/schemas/GetTraitResponse" }, "examples": { "GetTraitResponse": { "value": "{\"categories\": {\"face\": \"string\", \"background\": \"string\", \"level\": \"number\"}, \"counts\": { \"face\": {\"glasses\": 4, \"sunglasses\": 2}, \"background\": {\"red\": 6}, \"level\": {\"min\": 1, \"max\": 99}}}" } } } }, "description": "" }, "400": { "description": "For error reasons, review the response data." }, "500": { "description": "Internal server error. Please open a support ticket so OpenSea can investigate." } } } } }, "components": { "schemas": { "BasicListingPrice": { "title": "BasicListingPrice", "type": "object", "properties": { "current": { "title": "Price", "allOf": [ { "ref": "#/components/schemas/PriceModel" } ] } }, "required": [ "current" ] }, "BuildOffer": { "title": "BuildOffer", "type": "object", "properties": { "partialParameters": { "title": "Partialparameters", "description": "Partial set of Seaport Order Parameters", "allOf": [ { "ref": "#/components/schemas/PartialParameters" } ] }, "encoded_token_ids": { "title": "Encoded Token Ids", "description": "Represents a list of token ids which can be used to fulfill the criteria offer. When decoded using the provided SDK function, developers can now see a list of all tokens that could be used to fulfill the offer.", "type": "string" } }, "required": [ "partialParameters" ] }, "BuildOfferInput": { "title": "BuildOfferInput", "type": "object", "properties": { "offerer": { "title": "Offerer", "description": "The address which supplies all the items in the offer.", "type": "string" }, "quantity": { "title": "Quantity", "description": "The number of offers to place.", "default": 1, "type": "integer" }, "criteria": { "title": "Criteria", "description": "Criteria for the collection or trait offer", "allOf": [ { "ref": "#/components/schemas/Criteria" } ] }, "protocol_address": { "title": "Protocol Address", "description": "Exchange contract address. Must be one of ['0x00000000000000adc04c56bf30ac9d3c0aaf14dc']", "type": "string" }, "offer_protection_enabled": { "title": "Offer Protection Enabled", "description": "Builds the offer on OpenSea's signed zone to provide offer protections from receiving an item which is disabled from trading.", "default": true, "type": "boolean" } }, "required": [ "offerer", "criteria", "protocol_address" ], "additionalProperties": false }, "CancelEventModel": { "title": "CancelEventModel", "type": "object", "properties": { "event_type": { "allOf": [ { "ref": "#/components/schemas/CancelEventModelEventTypeEnum" } ], "title": "Event Type", "default": "cancel" }, "order_hash": { "title": "Order Hash", "description": "Order hash for the order which was cancelled", "type": "string" }, "chain": { "description": "The chain on which the cancelled order originated", "allOf": [ { "ref": "#/components/schemas/ChainIdentifier" } ] } }, "required": [ "order_hash", "chain" ] }, "CancelEventModelEventTypeEnum": { "enum": [ "cancel" ], "type": "string" }, "CategoryType": { "title": "CategoryType", "description": "An enumeration.", "enum": [ "string", "number" ], "type": "string", "properties": {} }, "ChainIdentifier": { "title": "ChainIdentifier", "description": "OpenSea supported chains.", "enum": [ "ethereum", "matic", "klaytn", "bsc", "solana", "arbitrum", "arbitrum_nova", "avalanche", "optimism", "base", "zora", "sepolia", "rinkeby", "mumbai", "baobab", "bsctestnet", "goerli", "soldev", "arbitrum_goerli", "avalanche_fuji", "optimism_goerli", "base_goerli", "zora_testnet" ], "properties": {} }, "Collection": { "title": "Collection", "type": "object", "properties": { "slug": { "title": "Slug", "description": "Unique string to identify a collection on OpenSea. This can be found by visiting the collection on the OpenSea website and noting the last path parameter.", "type": "string" } }, "required": [ "slug" ] }, "CollectionContractModel": { "title": "CollectionContractModel", "description": "Define the Contract's Addresses and Chain", "type": "object", "properties": { "address": { "title": "Address", "description": "The unique public blockchain identifier, address, for the contract.", "type": "string" }, "chain": { "description": "The chain on which the contract exists", "allOf": [ { "ref": "#/components/schemas/ChainIdentifier" } ] } }, "required": [ "address", "chain" ] }, "CollectionFeeModel": { "title": "CollectionFeeModel", "type": "object", "properties": { "fee": { "title": "Fee", "description": "Percentage of the sale price that is paid to the recipient", "type": "number" }, "recipient": { "title": "Recipient", "description": "The unique public blockchain identifier, address, for the recipient", "type": "string" }, "required": { "title": "Required", "description": "If the fee is required for the collection", "default": false, "type": "boolean" } }, "required": [ "fee", "recipient" ] }, "CollectionModel": { "title": "CollectionModel", "type": "object", "properties": { "collection": { "title": "Collection", "description": "Collection slug. A unique string to identify a collection on OpenSea", "type": "string" }, "name": { "title": "Name", "description": "Name of the collection", "type": "string" }, "description": { "title": "Description", "description": "Description of the collection", "default": "", "type": "string" }, "owner": { "title": "Owner", "description": "The unique public blockchain identifier, address, for the owner wallet.", "type": "string" }, "safelist_status": { "ref": "#/components/schemas/SafelistRequestStatus" }, "category": { "title": "Category", "description": "Category of the collection (e.g. PFPs, Memberships, Art)", "type": "string" }, "is_disabled": { "title": "Is Disabled", "description": "If the collection is currently able to be bought or sold using OpenSea", "type": "boolean" }, "is_nsfw": { "title": "Is Nsfw", "description": "If the collection is currently classified as 'Not Safe for Work' by OpenSea", "type": "boolean" }, "trait_offers_enabled": { "title": "Trait Offers Enabled", "description": "If trait offers are currently being accepted for the collection", "type": "boolean" }, "opensea_url": { "title": "Opensea Url", "description": "OpenSea Link to collection", "type": "string" }, "project_url": { "title": "Project Url", "description": "External URL for the collection's website", "default": "", "type": "string" }, "wiki_url": { "title": "Wiki Url", "description": "External URL for the collection's wiki", "default": "", "type": "string" }, "discord_url": { "title": "Discord Url", "description": "External URL for the collection's Discord server", "default": "", "type": "string" }, "telegram_url": { "title": "Telegram Url", "description": "External URL for the collection's Telegram group", "default": "", "type": "string" }, "twitter_username": { "title": "Twitter Username", "description": "Username for the collection's Twitter account", "default": "", "type": "string" }, "instagram_username": { "title": "Instagram Username", "description": "Username for the collection's Instagram account", "default": "", "type": "string" }, "contracts": { "title": "Contract", "type": "array", "items": { "ref": "#/components/schemas/CollectionContractModel" } } }, "required": [ "collection", "name", "owner", "safelist_status", "category", "is_disabled", "is_nsfw", "trait_offers_enabled", "opensea_url", "contracts" ] }, "CollectionStatsInterval": { "title": "CollectionStatsInterval", "description": "The interval for which the stats are calculated", "enum": [ "one_day", "one_week", "one_month" ], "type": "string", "properties": {} }, "CollectionStatsIntervalModel": { "title":

"CollectionStatsIntervalModel", "type": "object", "properties": { "interval": { "ref": "#/components/schemas/CollectionStatsInterval" }, "volume": { "title": "Volume", "description": "The volume of sales for the collection during the specified interval", "type": "number" }, "volume_diff": { "title": "Volume Diff", "description": "The volume differential compared to the previous interval", "type": "number" }, "volume_change": { "title": "Volume Change", "description": "The percentage change in volume compared to the previous interval", "type": "number" }, "sales": { "title": "Sales", "description": "The number of sales for the collection during the specified interval", "type": "integer" }, "sales_diff": { "title": "Sales Diff", "description": "The percentage change in number of sales compared to the previous interval", "type": "number" }, "average_price": { "title": "Average Price", "description": "The average sale price of NFTs in the collection during the interval", "type": "number" } }, "required": [ "interval", "volume", "volume_diff", "volume_change", "sales", "sales_diff", "average_price" ] }, "CollectionStatsModel": { "title": "CollectionStatsModel", "type": "object", "properties": { "volume": { "title": "Volume", "description": "The all time volume of sales for the collection", "type": "number" }, "sales": { "title": "Sales", "description": "The all time number of sales for the collection", "type": "integer" }, "average_price": { "title": "Average Price", "description": "The all time average sale price of NFTs in the collection", "type": "number" }, "num_owners": { "title": "Num Owners", "description": "The current number of unique owners of NFTs in the collection", "type": "integer" }, "market_cap": { "title": "Market Cap", "description": "The current market cap of the collection", "type": "number" }, "floor_price": { "title": "Floor Price", "description": "The current lowest price of NFTs in the collection", "type": "number" }, "floor_price_symbol": { "title": "Floor Price Symbol", "description": "The symbol of the payment asset for the floor price", "type": "string" } }, "required": [ "volume", "sales", "average_price", "num_owners", "market_cap", "floor_price", "floor_price_symbol" ] }, "ConfigEnum": { "enum": [ "affiliate", "affiliate_partner", "affiliate_requested", "affiliate_blacklisted", "verified", "moderator", "staff", "employee" ], "type": "string", "description": " *affiliate* - *affiliate*\n affiliate_partner - affiliate_partner\n *affiliate_requested* - *affiliate_requested*\n affiliate_blacklisted - affiliate_blacklisted\n *verified* - *verified*\n moderator - moderator\n *staff* - *staff*\n employee - employee" }, "ConsiderationInput": { "title": "ConsiderationInput", "type": "object", "properties": { "asset_contract_address": { "title": "Asset Contract Address", "type": "string" }, "token_id": { "title": "Token Id", "description": "NFT Token ID which will be used to fulfill the offer.", "type": "string" } }, "required": [ "asset_contract_address", "token_id" ] }, "ConsiderationItem": { "title": "ConsiderationItem", "type": "object", "properties": { "itemType": { "ref": "#/components/schemas/ItemType" }, "token": { "title": "Token", "description": "The item's token contract (with the null address used for native tokens)", "type": "string" }, "identifierOrCriteria": { "title": "Identifierorcriteria", "description": "The ERC721 or ERC1155 token identifier or, in the case of a criteria-based item type, a merkle root composed of the valid set of token identifiers for the item. This value will be ignored for Ether and ERC20 item types, and can optionally be zero for criteria-based item types to allow for any identifier.", "type": "integer" }, "startAmount": { "title": "Startamount", "description": "The amount of the token in question that will be required should the order be fulfilled.", "type": "integer" }, "endAmount": { "title": "Endamount", "description": "When endAmount differs from startAmount, the realized amount is calculated linearly based on the time elapsed since the order became active.", "type": "integer" }, "recipient": { "title": "Recipient", "description": "The address which will receive the consideration item when the order is executed.", "type": "string" } }, "required": [ "itemType", "token", "identifierOrCriteria", "startAmount", "endAmount", "recipient" ] }, "Contract": { "title": "Contract", "type": "object", "properties": { "address": { "title": "Address", "type": "string" } }, "required": [ "address" ] }, "CreateListingResponse": { "type": "object", "properties": { "order": { "ref": "#/components/schemas/SignedSimpleOrderV2" } }, "required": [ "order" ] }, "CreateOfferResponse": { "type": "object", "properties": { "order": { "ref": "#/components/schemas/SignedSimpleOrderV2" } }, "required": [ "order" ] }, "CreatedAtEnum": { "enum": [ " " ], "type": "string" }, "Criteria": { "title": "Criteria", "type": "object", "properties": { "collection": { "title": "Collection", "description": "The collection in which the criteria offer is being made for.", "allOf": [ { "ref": "#/components/schemas/Collection" } ] }, "contract": { "title": "Contract", "description": "The unique public blockchain identifier, address, for the NFT contract", "allOf": [ { "ref": "#/components/schemas/Contract" } ] }, "trait": { "title": "Trait", "description": "The trait that the criteria offer is being made for.", "allOf": [ { "ref": "#/components/schemas/Trait" } ] }, "encoded_token_ids": { "title": "Encoded Token Ids", "description": "Represents a list of token ids which can be used to fulfill the criteria offer. When decoded using the provided SDK function, developers can now see a list of all tokens that could be used to fulfill the offer.", "type": "string" } }, "required": [ "collection", "contract" ] }, "DetailedAccountDataModel": { "title": "DetailedAccountDataModel", "type": "object", "properties": { "address": { "title": "Address", "description": "The unique public blockchain identifier for the wallet.", "default": "", "type": "string" }, "username": { "title": "Username", "description": "The OpenSea account's username.", "default": "", "type": "string" }, "profile_image_url": { "title": "Profile Image Url", "description": "The OpenSea account's image url.", "default": "", "type": "string" }, "banner_image_url": { "title": "Banner Image Url", "description": "The OpenSea account's banner url.", "default": "", "type": "string" }, "website": { "title": "Website", "description": "Personal website for the OpenSea user.", "default": "", "type": "string" }, "social_media_accounts": { "title": "Social Media Account", "default": [], "type": "array", "items": { "ref": "#/components/schemas/SocialMediaAccountModel" } }, "bio": { "title": "Bio", "description": "The OpenSea account's bio.", "default": "", "type": "string" }, "joined_date": { "title": "Joined Date", "description": "Date the account was first added to OpenSea.", "type": "string", "format": "date" } } }, "DetailedCollectionModel": { "title": "DetailedCollectionModel", "type": "object", "properties": { "collection": { "title": "Collection", "description": "Collection slug. A unique string to identify a collection on OpenSea", "type": "string" }, "name": { "title": "Name", "description": "Name of the collection", "type": "string" }, "description": { "title": "Description", "description": "Description of the collection", "default": "", "type": "string" }, "owner": { "title": "Owner", "description": "The unique public blockchain identifier, address, for the owner wallet.", "type": "string" }, "safelist_status": { "ref": "#/components/schemas/SafelistRequestStatus" }, "category": { "title": "Category", "description": "Category of the collection (e.g. PFPs, Memberships, Art)", "type": "string" }, "is_disabled": { "title": "Is Disabled", "description": "If the collection is currently able to be bought or sold using OpenSea", "type": "boolean" }, "is_nsfw": { "title": "Is Nsfw", "description": "If the collection is currently classified as 'Not Safe for Work' by OpenSea", "type": "boolean" }, "trait_offers_enabled": { "title": "Trait Offers Enabled", "description": "If trait offers are currently being accepted for the collection", "type": "boolean" }, "opensea_url": { "title": "Opensea Url", "description": "OpenSea Link to collection", "type": "string" }, "project_url": { "title": "Project Url", "description": "External URL for the collection's website", "default": "", "type": "string" }, "wiki_url": { "title": "Wiki Url", "description": "External URL for the collection's wiki", "default": "", "type": "string" }, "discord_url": { "title": "Discord Url",

"description": "External URL for the collection's Discord server", "default": "", "type": "string" }, "telegram_url": { "title": "Telegram Url", "description": "External URL for the collection's Telegram group", "default": "", "type": "string" }, "twitter_username": { "title": "Twitter Username", "description": "Username for the collection's Twitter account", "default": "", "type": "string" }, "instagram_username": { "title": "Instagram Username", "description": "Username for the collection's Instagram account", "default": "", "type": "string" }, "contracts": { "title": "Contract", "type": "array", "items": { "ref": "#/components/schemas/CollectionContractModel" } }, "editors": { "title": "Editors", "description": "List of editor addresses for the collection", "type": "array", "items": { "type": "string" } }, "fees": { "title": "Fees", "description": "List of fees for the collection including creator earnings and OpenSea fees", "type": "array", "items": { "ref": "#/components/schemas/CollectionFeeModel" } } }, "required": [ "collection", "name", "owner", "safelist_status", "category", "is_disabled", "is_nsfw", "trait_offers_enabled", "opensea_url", "contracts", "editors", "fees" ] }, "DetailedNftModel": { "title": "DetailedNftModel", "type": "object", "properties": { "identifier": { "title": "Identifier", "description": "The NFT's unique identifier within the smart contract (also referred to as token_id)", "type": "string" }, "collection": { "title": "Collection", "description": "Collection slug. A unique string to identify a collection on OpenSea", "type": "string" }, "contract": { "title": "Contract", "description": "The unique public blockchain identifier for the contract", "type": "string" }, "token_standard": { "title": "Token Standard", "description": "ERC standard of the token (erc721, erc1155)", "type": "string" }, "name": { "title": "Name", "description": "Name of the NFT", "type": "string" }, "description": { "title": "Description", "description": "Description of the NFT", "type": "string" }, "image_url": { "title": "Image Url", "description": "Link to the NFT's original image. This may be an HTTP url, SVG data, or other directly embedded data.", "default": "", "type": "string" }, "metadata_url": { "title": "Metadata Url", "description": "Link to the offchain metadata store", "default": "", "type": "string" }, "created_at": { "allOf": [ { "ref": "#/components/schemas/CreatedAtEnum" } ], "title": "Created At", "description": "Deprecated Field", "default": " " }, "updated_at": { "title": "Updated At", "description": "Last time that the NFT's metadata was updated by OpenSea", "type": "string" }, "is_disabled": { "title": "Is Disabled", "description": "If the item is currently able to be bought or sold using OpenSea", "type": "boolean" }, "is_nsfw": { "title": "Is Nsfw", "description": "If the item is currently classified as 'Not Safe for Work' by OpenSea", "type": "boolean" }, "animation_url": { "title": "Animation Url", "description": "Link to the NFT's original animation.", "default": "", "type": "string" }, "is_suspicious": { "title": "Is Suspicious", "description": "If the item has been reported for suspicious activity by OpenSea", "type": "boolean" }, "creator": { "title": "Creator", "description": "The unique public blockchain identifier, wallet address, for the creator", "type": "string" }, "traits": { "title": "Traits", "description": "List of Trait objects. The field will be null if the NFT has more than 50 traits", "type": "array", "items": { "ref": "#/components/schemas/TraitModel" } }, "owners": { "title": "Owners", "description": "List of Owners. The field will be null if the NFT has more than 50 owners", "type": "array", "items": { "ref": "#/components/schemas/OwnerModel" } }, "rarity": { "title": "Rarity", "description": "Rarity data for the NFT", "allOf": [ { "ref": "#/components/schemas/RarityDataModel" } ] } }, "required": [ "identifier", "collection", "contract", "token_standard", "name", "description", "updated_at", "is_disabled", "is_nsfw", "is_suspicious", "creator", "traits", "owners", "rarity" ] }, "DisplayTypeField": { "title": "DisplayTypeField", "description": "A field indicating how to display. None is used for string traits.", "enum": [ "number", "boost_percentage", "boost_number", "author", "date", "None" ], "type": "string", "properties": {} }, "EventPaymentModel": { "title": "EventPaymentModel", "type": "object", "properties": { "quantity": { "title": "Quantity", "description": "Amount of tokens in the order", "type": "integer" }, "token_address": { "title": "Token Address", "description": "The contract address for the ERC20 token", "type": "string" }, "decimals": { "title": "Decimals", "description": "Returns the number of decimals the token uses - e.g. 8, means to divide the token amount by 100000000 to get its user representation.", "type": "integer" }, "symbol": { "title": "Symbol", "description": "Returns the symbol of the token, e.g. ETH, WETH, USDC, etc", "type": "string" } }, "required": [ "quantity", "token_address", "decimals", "symbol" ] }, "FulfillerInput": { "title": "FulfillerInput", "type": "object", "properties": { "address": { "title": "Address", "type": "string" } }, "required": [ "address" ] }, "FulfillmentData": { "title": "FulfillmentData", "type": "object", "properties": { "transaction": { "title": "Transaction", "description": "The name of the fulfillment method and associated call data.", "allOf": [ { "ref": "#/components/schemas/Transaction" } ] }, "orders": { "title": "Order", "description": "Array of Seaport Orders.", "type": "array", "items": { "ref": "#/components/schemas/SerializedOrder" } } }, "required": [ "transaction", "orders" ] }, "FulfillmentInput": { "title": "FulfillmentInput", "type": "object", "properties": { "hash": { "title": "Hash", "description": "Hash of the order to fulfill.", "type": "string" }, "chain": { "title": "Chain", "type": "string" }, "protocol_address": { "title": "Protocol Address", "description": "Exchange contract address. Must be one of ['0x00000000000000adc04c56bf30ac9d3c0aaf14dc']", "type": "string" } }, "required": [ "hash", "chain" ] }, "FulfillmentOutput": { "title": "FulfillmentOutput", "type": "object", "properties": { "protocol": { "title": "Protocol", "description": "Exchange contract address. Must be one of ['0x00000000000000adc04c56bf30ac9d3c0aaf14dc']", "type": "string" }, "fulfillment_data": { "title": "Fulfillment Data", "description": "All the information, including signatures, needed to fulfill an order directly onchain.", "allOf": [ { "ref": "#/components/schemas/FulfillmentData" } ] } }, "required": [ "protocol", "fulfillment_data" ] }, "GenerateListingFulfillmentInput": { "title": "GenerateListingFulfillmentInput", "type": "object", "properties": { "listing": { "title": "Listing", "description": "Listing to get fullfillment data for.", "allOf": [ { "ref": "#/components/schemas/FulfillmentInput" } ] }, "fulfiller": { "title": "Fulfiller", "description": "Fulfiller address.", "allOf": [ { "ref": "#/components/schemas/FulfillerInput" } ] } }, "required": [ "listing", "fulfiller" ] }, "GenerateOfferFulfillmentInput": { "title": "GenerateOfferFulfillmentInput", "type": "object", "properties": { "offer": { "title": "Offer", "description": "Offer to get fullfillment data for.", "allOf": [ { "ref": "#/components/schemas/FulfillmentInput" } ] }, "fulfiller": { "title": "Fulfiller", "description": "Fulfiller address.", "allOf": [ { "ref": "#/components/schemas/FulfillerInput" } ] }, "consideration": { "title": "Consideration", "description": "If the offer you are fulfilling is a criteria offer, the NFT you are using to fulfill the offer with. The fulfiller account must own this NFT or the request will not succeed.", "allOf": [ { "ref": "#/components/schemas/ConsiderationInput" } ] } }, "required": [ "offer", "fulfiller" ], "additionalProperties": false }, "GetCollectionStatsResponse": { "title": "GetCollectionStatsResponse", "type": "object", "properties": { "total": { "title": "Total", "description": "The aggregate stats over the collection's lifetime", "allOf": [ { "ref": "#/components/schemas/CollectionStatsModel" } ] }, "intervals": { "title": "Interval Stats", "description": "The stats for each interval", "type": "array", "items": { "ref": "#/components/schemas/CollectionStatsIntervalModel" } } }, "required": [ "total", "intervals" ] }, "GetListingsResponse": { "type": "object", "properties": { "next": { "type": "string", "description": "The cursor for the next page of results." }, "previous": { "type": "string", "description": "The cursor for the previous page of results." }, "orders": { "type": "array", "items": { "ref": "#/components/schemas/OrderV2" } } }, "required": [ "next", "orders", "previous" ] },

"GetNftResponse": { "title": "GetNftResponse", "type": "object", "properties": { "nft": { "ref": "#/components/schemas/DetailedNftModel" } }, "required": [ "nft" ] }, "GetOfferResponse": { "type": "object", "properties": { "next": { "type": "string", "description": "The cursor for the next page of results." }, "previous": { "type": "string", "description": "The cursor for the previous page of results." }, "orders": { "type": "array", "items": { "ref": "#/components/schemas/OrderV2" } } }, "required": [ "next", "orders", "previous" ] }, "GetOrderResult": { "title": "GetOrderResult", "type": "object", "properties": { "order": { "title": "Order", "anyOf": [ { "ref": "#/components/schemas/Listing" }, { "ref": "#/components/schemas/Offer" } ] } }, "required": [ "order" ] }, "GetTraitResponse": { "title": "GetTraitResponse", "type": "object", "properties": { "categories": { "description": "List of trait categories, e.g. Background, in the collection and their type, e.g. string", "default": {}, "type": "object", "additionalProperties": { "ref": "#/components/schemas/CategoryType" } }, "counts": { "title": "Counts", "description": "If the category type is STRING, the dict will contain each trait value and its count. Otherwise, the dict will contain the min and max value seen in the collection", "default": [], "type": "object", "additionalProperties": { "type": "object", "additionalProperties": { "type": "integer" } } } } }, "ItemType": { "title": "ItemType", "description": "0 - Native - Ether (or other native token for the given chain)\n1 - ERC20\n2 - ERC721\n3 - ERC1155\n4 - ERC721 with criteria\n5 - ERC1155 with criteria", "enum": [ 0, 1, 2, 3, 4, 5 ], "type": "integer", "properties": {} }, "ListCollectionsResponse": { "title": "ListCollectionsResponse", "type": "object", "properties": { "collections": { "title": "Collection", "type": "array", "items": { "ref": "#/components/schemas/CollectionModel" } }, "next": { "title": "Next", "description": "Cursor for the next page of results", "type": "string" } }, "required": [ "collections", "next" ] }, "ListEventsResponse": { "title": "ListEventsResponse", "type": "object", "properties": { "asset_events": { "title": "Events", "type": "array", "items": { "anyOf": [ { "ref": "#/components/schemas/CancelEventModel" }, { "ref": "#/components/schemas/OrderEventModel" }, { "ref": "#/components/schemas/SaleEventModel" }, { "ref": "#/components/schemas/TransferEventModel" }, { "ref": "#/components/schemas/RedemptionEventModel" } ] } }, "next": { "title": "Next", "description": "Cursor for the next page of results", "type": "string" } }, "required": [ "asset_events", "next" ] }, "ListNftsResponse": { "title": "ListNftsResponse", "type": "object", "properties": { "nfts": { "title": "NFT", "type": "array", "items": { "ref": "#/components/schemas/NftModel" } }, "next": { "title": "Next", "description": "Cursor for the next page of results", "type": "string" } }, "required": [ "nfts", "next" ] }, "Listing": { "title": "Listing", "type": "object", "properties": { "order_hash": { "title": "Order Hash", "description": "Order hash", "type": "string" }, "chain": { "description": "Chain the listing is on.", "allOf": [ { "ref": "#/components/schemas/ChainIdentifier" } ] }, "type": { "ref": "#/components/schemas/core__types__OrderType" }, "price": { "ref": "#/components/schemas/BasicListingPrice" }, "protocol_data": { "title": "Protocol Data", "description": "The onchain order data.", "allOf": [ { "ref": "#/components/schemas/SerializedOrder" } ] }, "protocol_address": { "title": "Protocol Address", "description": "Exchange contract address", "type": "string" } }, "required": [ "order_hash", "chain", "type", "price", "protocol_data", "protocol_address" ] }, "NftModel": { "title": "NftModel", "type": "object", "properties": { "identifier": { "title": "Identifier", "description": "The NFT's unique identifier within the smart contract (also referred to as token_id)", "type": "string" }, "collection": { "title": "Collection", "description": "Collection slug. A unique string to identify a collection on OpenSea", "type": "string" }, "contract": { "title": "Contract", "description": "The unique public blockchain identifier for the contract", "type": "string" }, "token_standard": { "title": "Token Standard", "description": "ERC standard of the token (erc721, erc1155)", "type": "string" }, "name": { "title": "Name", "description": "Name of the NFT", "type": "string" }, "description": { "title": "Description", "description": "Description of the NFT", "type": "string" }, "image_url": { "title": "Image Url", "description": "Link to the image associated with the NFT", "default": "", "type": "string" }, "metadata_url": { "title": "Metadata Url", "description": "Link to the offchain metadata store", "default": "", "type": "string" }, "created_at": { "allOf": [ { "ref": "#/components/schemas/CreatedAtEnum" } ], "title": "Created At", "description": "Deprecated Field", "default": " " }, "updated_at": { "title": "Updated At", "description": "Last time that the NFT's metadata was updated by OpenSea", "type": "string" }, "is_disabled": { "title": "Is Disabled", "description": "If the item is currently able to be bought or sold using OpenSea", "type": "boolean" }, "is_nsfw": { "title": "Is Nsfw", "description": "If the item is currently classified as 'Not Safe for Work' by OpenSea", "type": "boolean" } }, "required": [ "identifier", "collection", "contract", "token_standard", "name", "description", "updated_at", "is_disabled", "is_nsfw" ] }, "Offer": { "title": "Offer", "type": "object", "properties": { "order_hash": { "title": "Order Hash", "description": "Order hash", "type": "string" }, "chain": { "ref": "#/components/schemas/ChainIdentifier" }, "criteria": { "title": "Criteria", "description": "Criteria for collection or trait offers", "allOf": [ { "ref": "#/components/schemas/Criteria" } ] }, "protocol_data": { "title": "Protocol Data", "description": "The onchain order data.", "allOf": [ { "ref": "#/components/schemas/SerializedOrder" } ] }, "protocol_address": { "title": "Protocol Address", "description": "Exchange contract address", "type": "string" } }, "required": [ "order_hash", "chain", "protocol_data", "protocol_address" ] }, "OfferItem": { "title": "OfferItem", "type": "object", "properties": { "itemType": { "ref": "#/components/schemas/ItemType" }, "token": { "title": "Token", "description": "The item's token contract (with the null address used for native tokens)", "type": "string" }, "identifierOrCriteria": { "title": "Identifierorcriteria", "description": "The ERC721 or ERC1155 token identifier or, in the case of a criteria-based item type, a merkle root composed of the valid set of token identifiers for the item. This value will be ignored for Ether and ERC20 item types, and can optionally be zero for criteria-based item types to allow for any identifier.", "type": "integer" }, "startAmount": { "title": "Startamount", "description": "The amount of the token in question that will be required should the order be fulfilled.", "type": "integer" }, "endAmount": { "title": "Endamount", "description": "When endAmount differs from startAmount, the realized amount is calculated linearly based on the time elapsed since the order became active.", "type": "integer" } }, "required": [ "itemType", "token", "identifierOrCriteria", "startAmount", "endAmount" ] }, "OfferList": { "title": "Offer List", "type": "object", "properties": { "offers": { "title": "Offers", "type": "array", "items": { "ref": "#/components/schemas/Offer" } } }, "required": [ "offers" ] }, "OrderEventModel": { "title": "OrderEventModel", "type": "object", "properties": { "event_type": { "allOf": [ { "ref": "#/components/schemas/OrderEventModelEventTypeEnum" } ], "title": "Event Type", "default": "order" }, "order_hash": { "title": "Order Hash", "description": "Order hash for the newly created order", "type": "string" }, "order_type": { "ref": "#/components/schemas/OrderType" }, "chain": { "description": "The chain on which the order was created", "allOf": [ { "ref": "#/components/schemas/ChainIdentifier" } ] }, "protocol_address": { "title": "Protocol Address", "description": "Exchange contract address for the order", "type": "string" }, "start_date": { "title": "Start Date", "description": "The Posix timestamp at which the order was created", "type": "integer" }, "expiration_date": { "title": "Expiration Date", "description": "The Posix timestamp at which the order will close. When no expiration date is set, this value will be 0.",

"type": "integer" }, "asset": { "title": "Asset", "description": "The asset being listed or bid on. Empty object for collection or trait offers.", "anyOf": [ { "ref": "#/components/schemas/NftModel" }, { "type": "object" } ] }, "quantity": { "title": "Quantity", "description": "Number of assets in the order", "type": "integer" }, "maker": { "title": "Maker", "description": "Maker of the order", "type": "string" }, "taker": { "title": "Taker", "description": "Taker of the order. This will only be set for private listings.", "type": "string" }, "payment": { "title": "Payment", "allOf": [ { "ref": "#/components/schemas/EventPaymentModel" } ] }, "criteria": { "title": "Criteria", "description": "For collection and trait offers, this object will contain the criteria needed to fulfill the offer.", "anyOf": [ { "ref": "#/components/schemas/Criteria" }, { "type": "object" } ] } }, "required": [ "order_hash", "order_type", "chain", "protocol_address", "start_date", "expiration_date", "asset", "quantity", "maker", "taker", "payment", "criteria" ] }, "OrderEventModelEventTypeEnum": { "enum": [ "order" ], "type": "string" }, "OrderInput": { "title": "OrderInput", "type": "object", "properties": { "parameters": { "title": "Order Components", "allOf": [ { "ref": "#/components/schemas/OrderInputComponents" } ] }, "signature": { "title": "Signature", "description": "Signature of the signed type data represented by the parameters field.", "type": "string" } }, "required": [ "parameters", "signature" ] }, "OrderInputComponents": { "title": "OrderInputComponents", "type": "object", "properties": { "offerer": { "title": "Offerer", "description": "The address which supplies all the items in the offer.", "type": "string" }, "offer": { "title": "Offer", "description": "Array of items that may be transferred from the offerer's account.", "type": "array", "items": { "ref": "#/components/schemas/OfferItem" } }, "consideration": { "title": "Consideration", "description": "Array of items which must be received by a recipient to fulfill the order. One of the consideration items must be the OpenSea marketplace fee.", "type": "array", "items": { "ref": "#/components/schemas/ConsiderationItem" } }, "startTime": { "title": "Starttime", "description": "The block timestamp at which the order becomes active", "type": "integer" }, "endTime": { "title": "Endtime", "description": "The block timestamp at which the order expires.", "type": "integer" }, "orderType": { "ref": "#/components/schemas/OrderType" }, "zone": { "title": "Zone", "description": "Optional secondary account attached the order which can cancel orders. Additionally, when the OrderType is Restricted, the zone or the offerer are the only entities which can execute the order.\nFor open orders, use the zero address.\nFor restricted orders, use the signed zone address ", "type": "string" }, "zoneHash": { "title": "Zonehash", "description": "A value that will be supplied to the zone when fulfilling restricted orders that the zone can utilize when making a determination on whether to authorize the order. Most often this value will be the zero hash 0x0000000000000000000000000000000000000000000000000000000000000000", "type": "string" }, "salt": { "title": "Salt", "description": "an arbitrary source of entropy for the order", "type": "string" }, "conduitKey": { "title": "Conduitkey", "description": "Indicates what conduit, if any, should be utilized as a source for token approvals when performing transfers. By default (i.e. when conduitKey is set to the zero hash), the offerer will grant transfer approvals to Seaport directly. \nTo utilize OpenSea's conduit, use 0x0000007b02230091a7ed01230072f7006a004d60a8d4e71d599b8104250f0000", "type": "string" }, "totalOriginalConsiderationItems": { "title": "Totaloriginalconsiderationitems", "description": "Size of the consideration array.", "type": "integer" }, "counter": { "title": "Counter", "description": "Must match the current counter for the given offerer. If you are unsure of the current counter, it can be read from the contract on etherscan.", "type": "string" } }, "required": [ "offerer", "offer", "consideration", "startTime", "endTime", "orderType", "zone", "zoneHash", "salt", "conduitKey", "counter" ] }, "OrderInputWithProtocol": { "title": "OrderInputWithProtocol", "type": "object", "properties": { "parameters": { "title": "Order Components", "allOf": [ { "ref": "#/components/schemas/OrderInputComponents" } ] }, "signature": { "title": "Signature", "description": "Signature of the signed type data represented by the parameters field.", "type": "string" }, "protocol_address": { "title": "Protocol Address", "description": "Exchange contract address. Must be one of ['0x00000000000000adc04c56bf30ac9d3c0aaf14dc']", "type": "string" } }, "required": [ "parameters", "signature", "protocol_address" ] }, "OrderType": { "title": "OrderType", "description": "An enumeration.", "enum": [ "listing", "auction", "item_offer", "collection_offer", "trait_offer" ], "properties": {} }, "OrderTypeEnum": { "enum": [ "basic", "dutch", "english", "criteria" ], "type": "string", "description": " *basic* - *basic*\n*n* dutch - dutch\n *english* - *english*\n*n* criteria - criteria" }, "OrderV2": { "type": "object", "description": "Models OrderV2 objects to serialize to a 'similar' schema to what we have with OrderV1s", "properties": { "created_date": { "type": "string", "format": "date-time", "description": "Date the order was created" }, "closing_date": { "type": "string", "format": "date-time", "description": "Date the order was closed" }, "listing_time": { "type": "integer", "readOnly": true, "description": "Timestamp representation of created_date" }, "expiration_time": { "type": "integer", "readOnly": true, "description": "Timestamp representation of closing_date" }, "order_hash": { "type": "string", "description": "An identifier for the order" }, "protocol_data": { "allOf": [ { "ref": "#/components/schemas/SerializedOrder" } ], "readOnly": true }, "protocol_address": { "type": "string", "nullable": true, "readOnly": true, "description": "Exchange Contract Address. Typically the address of the Seaport contract." }, "current_price": { "type": "string", "description": "Current price of the order" }, "maker": { "allOf": [ { "ref": "#/components/schemas/SimpleAccount" } ], "readOnly": true, "description": "The unique blockchain identifier, address, of the wallet which is the order maker." }, "taker": { "allOf": [ { "ref": "#/components/schemas/SimpleAccount" } ], "readOnly": true, "description": "The unique blockchain identifier, address, of the wallet which is the order taker." }, "maker_fees": { "type": "array", "items": { "ref": "#/components/schemas/SimpleFee" }, "readOnly": true }, "taker_fees": { "type": "array", "items": { "ref": "#/components/schemas/SimpleFee" }, "readOnly": true }, "side": { "type": "string", "description": "The side of the order, either bid (offer) or ask(listing)." }, "order_type": { "allOf": [ { "ref": "#/components/schemas/OrderTypeEnum" } ], "readOnly": true }, "cancelled": { "type": "boolean", "description": "If true, the order maker has canceled the order which means it can no longer be filled." }, "finalized": { "type": "boolean", "description": "If true, the order has already been filled." }, "marked_invalid": { "type": "boolean", "readOnly": true, "description": "If true, the order is currently invalid and can not be filled." }, "remaining_quantity": { "type": "integer", "description": "The remaining quantity of the order that has not been filled. This is useful for erc1155 orders." }, "relay_id": { "type": "string", "readOnly": true, "description": "Deprecated Field" }, "criteria_proof": { "type": "array", "items": { "type": "string" }, "nullable": true, "readOnly": true, "description": "A merkle root composed of the valid set of token identifiers for the order" }, "maker_asset_bundle": { "type": "object", "additionalProperties": {}, "description": "Deprecated Field.", "readOnly": true }, "taker_asset_bundle": { "type": "object", "additionalProperties": {}, "description": "Deprecated Field.", "readOnly": true } }, "required": [ "cancelled", "closing_date", "created_date", "criteria_proof", "current_price", "expiration_time", "finalized", "listing_time", "maker", "maker_asset_bundle", "maker_fees", "marked_invalid", "order_hash", "order_type", "protocol_address", "protocol_data", "relay_id", "remaining_quantity", "side", "taker", "taker_asset_bundle", "taker_fees" ] },

"OwnerModel": { "title": "OwnerModel", "type": "object", "properties": { "address": { "title": "Address", "description": "The unique public blockchain identifier for the owner wallet", "type": "string" }, "quantity": { "title": "Quantity", "description": "The number of tokens owned ", "type": "integer" } }, "required": [ "address", "quantity" ] }, "PaginatedListingList": { "title": "Paginated Listing List", "type": "object", "properties": { "listings": { "title": "Listings", "description": "OpenSea Listings", "type": "array", "items": { "ref": "#/components/schemas/Listing" } }, "next": { "title": "Next", "description": "Cursor for the next page of results", "type": "string" } }, "required": [ "listings", "next" ] }, "PaginatedOfferList": { "title": "Paginated Offer List", "type": "object", "properties": { "offers": { "title": "Offers", "type": "array", "items": { "ref": "#/components/schemas/Offer" } }, "next": { "title": "Next", "description": "Cursor for the next page of results", "type": "string" } }, "required": [ "offers", "next" ] }, "PartialParameters": { "title": "PartialParameters", "type": "object", "properties": { "consideration": { "title": "Consideration", "description": "One of the consideration items used when creating criteria offers.", "type": "array", "items": { "ref": "#/components/schemas/SerializedConsiderationItem" } }, "zone": { "title": "Zone", "description": "Optional secondary account attached the order which can cancel orders. Additionally, when the OrderType is Restricted, the zone or the offerer are the only entities which can execute the order.\nFor open orders, use the zero address.\nFor restricted orders, use the signed zone address 0x000000e7ec00e7b300774b00001314b8610022b8", "type": "string" }, "zoneHash": { "title": "Zonehash", "description": "A value that will be supplied to the zone when fulfilling restricted orders that the zone can utilize when making a determination on whether to authorize the order. Most often this value will be the zero hash 0x0000000000000000000000000000000000000000000000000000000000000000", "type": "string" } }, "required": [ "consideration", "zone", "zoneHash" ] }, "PostCriteriaOfferInput": { "title": "PostCriteriaOfferInput", "type": "object", "properties": { "protocol_data": { "title": "Signed Seaport Order", "description": "The signed order which will be submitted to Seaport", "allOf": [ { "ref": "#/components/schemas/OrderInput" } ] }, "criteria": { "title": "Criteria", "description": "Criteria for the collection or trait offer", "allOf": [ { "ref": "#/components/schemas/Criteria" } ] }, "protocol_address": { "title": "Protocol Address", "description": "Exchange contract address. Must be one of ['0x00000000000000adc04c56bf30ac9d3c0aaf14dc']", "type": "string" } }, "required": [ "protocol_data", "criteria", "protocol_address" ], "additionalProperties": false }, "PriceModel": { "title": "PriceModel", "type": "object", "properties": { "currency": { "title": "Currency", "type": "string" }, "decimals": { "title": "Decimals", "type": "integer" }, "value": { "title": "Value", "type": "string" } }, "required": [ "currency", "decimals", "value" ] }, "RankingFeatures": { "title": "RankingFeatures", "type": "object", "properties": { "unique_attribute_count": { "title": "Unique Attribute Count", "description": "Deprecated Field.", "default": 0, "type": "integer" } } }, "RarityDataModel": { "title": "RarityDataModel", "type": "object", "properties": { "strategy_id": { "description": "Deprecated Field", "allOf": [ { "ref": "#/components/schemas/RarityStrategyId" } ] }, "strategy_version": { "title": "Strategy Version", "description": "Deprecated Field", "type": "string" }, "rank": { "title": "Rank", "description": "Rarity Rank of the NFT in the collection", "type": "integer" }, "score": { "title": "Score", "description": "Deprecated Field", "type": "number" }, "calculated_at": { "title": "Calculated At", "description": "Deprecated Field", "default": "", "type": "string" }, "max_rank": { "title": "Max Rank", "description": "Deprecated Field", "type": "integer" }, "total_supply": { "title": "Total Supply", "description": "Deprecated Field", "default": 0, "type": "integer" }, "ranking_features": { "title": "Ranking Features", "description": "Deprecated Field", "allOf": [ { "ref": "#/components/schemas/RankingFeatures" } ] } }, "required": [ "rank" ] }, "RarityStrategyId": { "title": "RarityStrategyId", "description": "Rarity algorithm used. Currently, always 'openrarity'", "enum": [ "openrarity" ], "properties": {} }, "RedemptionEventModel": { "title": "RedemptionEventModel", "type": "object", "properties": { "event_type": { "allOf": [ { "ref": "#/components/schemas/RedemptionEventModelEventTypeEnum" } ], "title": "Event Type", "default": "redemption" }, "chain": { "description": "The chain on which the rededemption occurred", "allOf": [ { "ref": "#/components/schemas/ChainIdentifier" } ] }, "from_address": { "title": "From Address", "description": "Address of the sender", "type": "string" }, "to_address": { "title": "To Address", "description": "Address of the recipient", "type": "string" }, "asset": { "title": "Asset", "description": "The asset being redeemed.", "anyOf": [ { "ref": "#/components/schemas/NftModel" }, { "type": "object" } ] }, "quantity": { "title": "Quantity", "description": "Number of assets redeemed", "type": "integer" }, "transaction": { "title": "Transaction", "description": "Transaction hash for the redemption", "type": "string" } }, "required": [ "chain", "from_address", "to_address", "asset", "quantity", "transaction" ] }, "RedemptionEventModelEventTypeEnum": { "enum": [ "redemption" ], "type": "string" }, "SafelistRequestStatus": { "title": "SafelistRequestStatus", "description": "Status of the collection verification requests.", "enum": [ "not_requested", "requested", "approved", "verified", "disabled_top_trending" ], "properties": {} }, "SaleEventModel": { "title": "SaleEventModel", "type": "object", "properties": { "event_type": { "allOf": [ { "ref": "#/components/schemas/SaleEventModelEventTypeEnum" } ], "title": "Event Type", "default": "sale" }, "order_hash": { "title": "Order Hash", "description": "Order hash for the order which was fulfilled", "type": "string" }, "chain": { "description": "The chain on which the order was fulfilled", "allOf": [ { "ref": "#/components/schemas/ChainIdentifier" } ] }, "protocol_address": { "title": "Protocol Address", "description": "Exchange contract address which fulfilled the order", "type": "string" }, "closing_date": { "title": "Closing Date", "description": "The Posix timestamp at which the transaction which filled the order occurred", "type": "integer" }, "quantity": { "title": "Quantity", "description": "Number of assets transferred", "type": "integer" }, "maker": { "title": "Maker", "description": "Maker of the order", "type": "string" }, "taker": { "title": "Taker", "description": "Taker of the order", "type": "string" }, "payment": { "title": "Payment", "allOf": [ { "ref": "#/components/schemas/EventPaymentModel" } ] }, "transaction": { "title": "Transaction", "description": "Transaction hash for the order fulfillment", "type": "string" } }, "required": [ "order_hash", "chain", "protocol_address", "closing_date", "quantity", "maker", "taker", "payment", "transaction" ] }, "SaleEventModelEventTypeEnum": { "enum": [ "sale" ], "type": "string" }, "SerializedConsiderationItem": { "title": "SerializedConsiderationItem", "type": "object", "properties": { "itemType": { "ref": "#/components/schemas/ItemType" }, "token": { "title": "Token", "description": "The item's token contract (with the null address used for native tokens)", "type": "string" }, "identifierOrCriteria": { "title": "Identifierorcriteria", "description": "The ERC721 or ERC1155 token identifier or, in the case of a criteria-based item type, a merkle root composed of the valid set of token identifiers for the item. This value will be ignored for Ether and ERC20 item types, and can optionally be zero for criteria-based item types to allow for any identifier.", "type": "string" }, "startAmount": { "title": "Startamount", "description": "The amount of the token in question that will be required should the order be fulfilled.", "type": "string" }, "endAmount": { "title": "Endamount", "description": "When endAmount differs from startAmount, the realized amount is calculated linearly based on the time elapsed since the order became active.", "type": "string" }, "recipient": { "title": "Recipient", "description": "The address which will receive the

consideration item when the order is executed.", "type": "string" } }, "required": [ "itemType", "token", "identifierOrCriteria", "startAmount", "endAmount", "recipient" ] }, "SerializedOfferItem": { "title": "SerializedOfferItem", "type": "object", "properties": { "itemType": { "ref": "#/components/schemas/ItemType" }, "token": { "title": "Token", "description": "The item's token contract (with the null address used for native tokens)", "type": "string" }, "identifierOrCriteria": { "title": "Identifierorcriteria", "description": "The ERC721 or ERC1155 token identifier or, in the case of a criteria-based item type, a merkle root composed of the valid set of token identifiers for the item. This value will be ignored for Ether and ERC20 item types, and can optionally be zero for criteria-based item types to allow for any identifier.", "type": "string" }, "startAmount": { "title": "Startamount", "description": "The amount of the token in question that will be required should the order be fulfilled.", "type": "string" }, "endAmount": { "title": "Endamount", "description": "When endAmount differs from startAmount, the realized amount is calculated linearly based on the time elapsed since the order became active.", "type": "string" } }, "required": [ "itemType", "token", "identifierOrCriteria", "startAmount", "endAmount" ] }, "SerializedOrder": { "title": "SerializedOrder", "type": "object", "properties": { "parameters": { "title": "Order", "allOf": [ { "ref": "#/components/schemas/SerializedOrderComponents" } ] }, "signature": { "title": "Signature", "description": "The order maker's signature used to validate the order.", "type": "string" } }, "required": [ "parameters" ] }, "SerializedOrderComponents": { "title": "SerializedOrderComponents", "type": "object", "properties": { "offerer": { "title": "Offerer", "description": "The address which supplies all the items in the offer.", "type": "string" }, "offer": { "title": "Offer", "type": "array", "items": { "ref": "#/components/schemas/SerializedOfferItem" } }, "consideration": { "title": "Consideration", "type": "array", "items": { "ref": "#/components/schemas/SerializedConsiderationItem" } }, "startTime": { "title": "Starttime", "description": "The block timestamp at which the order becomes active", "type": "string" }, "endTime": { "title": "Endtime", "description": "The block timestamp at which the order expires", "type": "string" }, "orderType": { "ref": "#/components/schemas/core__blockchain__evm__abi__models__seaport__OrderType" }, "zone": { "title": "Zone", "description": "Optional secondary account attached the order which can cancel orders. Additionally, when the OrderType is Restricted, the zone or the offerer are the only entities which can execute the order.\nFor open orders, use the zero address.\nFor restricted orders, use the signed zone address ", "type": "string" }, "zoneHash": { "title": "Zonehash", "description": "A value that will be supplied to the zone when fulfilling restricted orders that the zone can utilize when making a determination on whether to authorize the order. Most often this value will be the zero hash 0x0000000000000000000000000000000000000000000000000000000000000000", "type": "string" }, "salt": { "title": "Salt", "description": "an arbitrary source of entropy for the order", "type": "string" }, "conduitKey": { "title": "Conduitkey", "description": "Indicates what conduit, if any, should be utilized as a source for token approvals when performing transfers. By default (i.e. when conduitKey is set to the zero hash), the offerer will grant transfer approvals to Seaport directly. \nTo utilize OpenSea's conduit, use 0x0000007b02230091a7ed01230072f7006a004d60a8d4e71d599b8104250f0000", "type": "string" }, "totalOriginalConsiderationItems": { "title": "Totaloriginalconsiderationitems", "description": "Size of the consideration array.", "type": "integer" }, "counter": { "title": "Counter", "anyOf": [ { "type": "integer" }, { "type": "string" } ] } }, "required": [ "offerer", "offer", "consideration", "startTime", "endTime", "orderType", "zone", "zoneHash", "salt", "conduitKey", "counter" ] }, "SignedSimpleOrderV2": { "type": "object", "description": "OpenSea Order Object", "properties": { "created_date": { "type": "string", "format": "date-time", "description": "Date the order was created" }, "closing_date": { "type": "string", "format": "date-time", "description": "Date the order was closed" }, "listing_time": { "type": "integer", "readOnly": true, "description": "Timestamp representation of created_date" }, "expiration_time": { "type": "integer", "readOnly": true, "description": "Timestamp representation of closing_date" }, "order_hash": { "type": "string", "description": "An identifier for the order" }, "protocol_data": { "allOf": [ { "ref": "#/components/schemas/SerializedOrder" } ], "readOnly": true }, "protocol_address": { "type": "string", "nullable": true, "readOnly": true, "description": "Exchange Contract Address. Typically the address of the Seaport contract." }, "current_price": { "type": "string", "description": "Current price of the order" }, "maker": { "allOf": [ { "ref": "#/components/schemas/SimpleAccount" } ], "readOnly": true, "description": "The unique blockchain identifier, address, of the wallet which is the order maker." }, "taker": { "allOf": [ { "ref": "#/components/schemas/SimpleAccount" } ], "readOnly": true, "description": "The unique blockchain identifier, address, of the wallet which is the order taker." }, "maker_fees": { "type": "array", "items": { "ref": "#/components/schemas/SimpleFee" }, "readOnly": true }, "taker_fees": { "type": "array", "items": { "ref": "#/components/schemas/SimpleFee" }, "readOnly": true }, "side": { "type": "string", "description": "The side of the order, either bid (offer) or ask(listing)." }, "order_type": { "allOf": [ { "ref": "#/components/schemas/OrderTypeEnum" } ], "readOnly": true }, "cancelled": { "type": "boolean", "description": "If true, the order maker has canceled the order which means it can no longer be filled." }, "finalized": { "type": "boolean", "description": "If true, the order has already been filled." }, "marked_invalid": { "type": "boolean", "readOnly": true, "description": "If true, the order is currently invalid and can not be filled." }, "remaining_quantity": { "type": "integer", "description": "The remaining quantity of the order that has not been filled. This is useful for erc1155 orders." }, "relay_id": { "type": "string", "readOnly": true, "description": "Deprecated Field" }, "criteria_proof": { "type": "array", "items": { "type": "string" }, "nullable": true, "readOnly": true, "description": "A merkle root composed of the valid set of token identifiers for the order" } }, "required": [ "cancelled", "closing_date", "created_date", "criteria_proof", "current_price", "expiration_time", "finalized", "listing_time", "maker", "maker_fees", "marked_invalid", "order_hash", "order_type", "protocol_address", "protocol_data", "relay_id", "remaining_quantity", "side", "taker", "taker_fees" ] }, "SimpleAccount": { "type": "object", "properties": { "user": { "type": "integer", "readOnly": true, "nullable": true }, "profile_img_url": { "type": "string", "readOnly": true, "description": "A placeholder image. For the actual profile image, call the Get Account endpoint." }, "address": { "type": "string", "readOnly": true, "description": "The unique blockchain identifier, address, of the account." }, "config": { "allOf": [ { "ref": "#/components/schemas/ConfigEnum" } ], "readOnly": true } }, "required": [ "address", "config", "profile_img_url", "user" ] }, "SimpleFee": { "type": "object", "properties": { "account": { "allOf": [ { "ref": "#/components/schemas/SimpleAccount" } ], "readOnly": true }, "basis_points": { "type": "string" } }, "required": [ "account", "basis_points" ] }, "SocialMediaAccountModel": { "title": "SocialMediaAccountModel", "type": "object", "properties": { "platform": { "title": "Platform", "description": "The social media platform, e.g. twitter or instagram", "type": "string" }, "username": { "title": "Username", "description": "The username for the social media platform", "type": "string" } }, "required": [ "platform", "username" ] }, "Trait": { "title": "Trait", "type": "object", "properties": { "type": { "title": "Type", "type": "string" }, "value": { "title": "Value", "type": "string" } }, "required": [ "type",

"value" ] }, "TraitModel": { "title": "TraitModel", "type": "object", "properties": { "trait_type": { "title": "Trait Type", "description": "The name of the trait category (e.g. 'Background')", "maxLength": 150, "type": "string" }, "display_type": { "ref": "#/components/schemas/DisplayTypeField" }, "max_value": { "title": "Max Value", "description": "Ceiling for possible numeric trait values", "type": "string" }, "trait_count": { "title": "Trait Count", "description": "Deprecated Field. Use Get Collection API instead.", "type": "integer" }, "order": { "title": "Order", "description": "Deprecated Field", "type": "integer" }, "value": { "title": "Value", "description": "The value of the trait (e.g. 'Red')", "anyOf": [ { "type": "number" }, { "type": "integer" }, { "type": "string", "format": "date" }, { "type": "string" } ] } }, "required": [ "pk", "trait_type", "max_value", "value" ] }, "Transaction": { "title": "Transaction", "type": "object", "properties": { "function": { "title": "Function", "description": "Seaport protocol contract method to use to fulfill the order.", "type": "string" }, "chain": { "title": "Chain", "description": "Numeric Chain Identifier.", "type": "integer" }, "to": { "title": "To", "description": "Protocol contract address to use fto fulfill the order.", "type": "string" }, "value": { "title": "Value", "description": "Wei value of the transaction.", "type": "integer" }, "input_data": { "title": "Input Data", "description": "Decoded Call Data.", "type": "object" } }, "required": [ "function", "chain", "to", "value", "input_data" ] }, "TransferEventModel": { "title": "TransferEventModel", "type": "object", "properties": { "event_type": { "allOf": [ { "ref": "#/components/schemas/TransferEventModelEventTypeEnum" } ], "title": "Event Type", "default": "transfer" }, "chain": { "description": "The chain on which the transfer occurred", "allOf": [ { "ref": "#/components/schemas/ChainIdentifier" } ] }, "transaction": { "title": "Transaction", "description": "Transaction hash for the transfer", "type": "string" }, "from_address": { "title": "From Address", "description": "Address of the sender", "type": "string" }, "to_address": { "title": "To Address", "description": "Address of the recipient", "type": "string" }, "quantity": { "title": "Quantity", "description": "Number of assets transferred", "type": "integer" } }, "required": [ "chain", "transaction", "from_address", "to_address", "quantity" ] }, "TransferEventModelEventTypeEnum": { "enum": [ "transfer" ], "type": "string" }, "core__blockchain__evm__abi__models__seaport__OrderType": { "title": "OrderType", "description": "The type of order, which determines how it can be executed.\n0 = Full Open - No partial fills, anyone can execute\n1 = Partial Open - Partial fills supported, anyone can execute\n2 = Full Restricted - No partial fills, only offerer or zone can check if it can be executed\n3 = Partial Restricted - Partial fills supported, only offerer or zone can check if it can be executed\n4 = Contract - Contract order type, for contract offerers that can dynamically generate orders. Introduced in Seaport v1.4 and currently unsupported", "enum": [ 0, 1, 2, 3, 4 ], "type": "integer", "properties": {} }, "core__types__OrderType": { "title": "OrderType", "description": "basic - Quantities are fixed. Used for fixed price listings and offers.\ndutch - The quantity represents the starting price.\nenglish - The quantity represents the minimum price.\ncriteria - The items that are accepted by this offer will be found in the criteria fields.", "enum": [ "basic", "dutch", "english", "criteria" ], "properties": {} } }, "securitySchemes": { "OpenSeaAuth": { "type": "apiKey", "in": "header", "name": "x-api-key" } } }, "servers": [ { "url": "https://api.opensea.io/", "description": "Mainnet" }, { "url": "https://testnets-api.opensea.io/", "description": "Testnet" } ] }