Introduction

Resources

NFT API

Transfers API (Tx History)

Transaction Receipts API

Token API

Subgraphs

Webhooks

Websockets

Trace API

Debug API

ACCOUNT ABSTRACTION

Embedded Accounts

Gas Manager Admin API

Alchemy Transact

Alchemy SDK

  - getTokenMetadata - SDK
-
  - getTransactionReceipt - SDK
-
  - send - SDK
-
  - getTransactionReceipts - SDK
-
  - getTransaction - SDK
-
  - isContractAddress - SDK
-
  - getNftMetadata - SDK
-
  - getNftSales - SDK
-
  - cancelPrivateTransaction - SDK
-
  - sendPrivateTransaction - SDK
-
  - traceTransaction - SDK
-
  - simulateExecutionBundle - SDK
-
  - simulateExecution - SDK
-
  - getMaxPriorityFeePerGas - SDK
-
  - simulateAssetChangesBundle - SDK
-
  - estimateGas - SDK
-
  - simulateAssetChanges - SDK
-
  - traceBlock - SDK
-
  - waitForTransaction - SDK
-
  - traceCall - SDK
-
  - sendTransaction - SDK
-
  - updateWebhook - SDK
-
  - refreshNftMetadata -SDK
-
  - createWebhook - SDK
-
  - getNftFilters - SDK
-
  - getAddresses - SDK
-
  - summarizeNftAttributes - SDK
-
  - deleteWebhook - SDK
-
  - searchContractMetadata - SDK
-
  - getAllWebhooks - SDK
-
  - getNftsForOwnerIterator - SDK
-
  - getNftsForContractIterator -SDK
-
  - getContractMetadata - SDK
-
  - getTransfersForOwner - SDK
-

Ethereum

-
  - net_version - Ethereum post
-
  - web3_clientVersion - Ethereum post
-
  - web3_sha3 - Ethereum post
-
  - eth_getTransactionByBlockNumberAndIndex - Ethereum post
-
  - eth_call - Ethereum post
-
  - eth_getBlockByHash - Ethereum post
-
  - eth_createAccessList - Ethereum post
-
  - eth_newFilter - Ethereum post
-
  - eth_getFilterChanges - Ethereum post
-
  - eth_getFilterLogs - Ethereum post
-
  - eth_newBlockFilter - Ethereum post
-
  - eth_newPendingTransactionFilter - Ethereum post
-
  - eth_uninstallFilter - Ethereum post
-
  - eth_subscribe
-
  - eth_unsubscribe

Polygon PoS

- Polygon PoS API Quickstart
-
  - Polygon SDK Examples
- Polygon PoS API FAQ
- Polygon PoS API Endpoints
-
  - bor_getAuthor - Polygon PoS post
-
  - bor_getCurrentProposer - Polygon PoS post
-
  - bor_getCurrentValidators - Polygon PoS post
-
  - bor_getRootHash - Polygon PoS post
-
  - eth_accounts - Polygon PoS post
-
  - eth_call - Polygon PoS post
-
  - eth_chainId - Polygon PoS post
-
  - eth_estimateGas - Polygon PoS post
-
  - eth_gasPrice - Polygon PoS post
-
  - eth_getBalance - Polygon PoS post
-
  - eth_getBlockByHash - Polygon PoS post
-
  - eth_getBlockByNumber - Polygon PoS post
-
  - eth_getBlockTransactionCountByHash - Polygon PoS post
-
  - eth_getBlockTransactionCountByNumber - Polygon PoS post
-
  - eth_getCode - Polygon PoS post
-

Polygon zkEVM

  - eth_getTransactionCount - Polygon zkEVM post
-
  - eth_call - Polygon zkEVM post
-
  - eth_chainId - Polygon zkEVM post
-
  - eth_newBlockFilter - Polygon zkEVM post
-
  - eth_estimateGas - Polygon zkEVM post
-
  - eth_newFilter - Polygon zkEVM post
-
  - eth_gasPrice - Polygon zkEVM post
-
  - eth_sendRawTransaction - Polygon zkEVM post
-
  - eth_getBalance - Polygon zkEVM post
-
  - eth_uninstallFilter - Polygon zkEVM post
-
  - eth_getBlockByHash - Polygon zkEVM post
-
  - net_version - Polygon zkEVM post
-
  - eth_getBlockByNumber - Polygon zkEVM post
-
  - web3_clientVersion - Polygon zkEVM post
-
  - eth_getBlockTransactionCountByHash - Polygon zkEVM post
-
  - eth_getBlockTransactionCountByNumber - Polygon zkEVM post
-
  - zkevm_batchNumber - Polygon zkEVM post
-
  - eth_getCode - Polygon zkEVM post
-
  - eth_getFilterChanges - Polygon zkEVM post
-
  - eth_getFilterLogs - Polygon zkEVM post
-
  - zkevm_getBatchByNumber - Polygon zkEVM post
-
  - eth_getLogs - Polygon zkEVM post
-
  - zkevm_getBroadcastURI - Polygon zkEVM post
-
  - eth_getStorageAt - Polygon zkEVM post
-
  - zkevm_isBlockConsolidated - Polygon zkEVM post
-
  - eth_getTransactionByBlockHashAndIndex - Polygon zkEVM post
-
  - zkevm_isBlockVirtualized - Polygon zkEVM post
-
  - eth_getTransactionByBlockNumberAndIndex - Polygon zkEVM post
-
  - zkevm_verifiedBatchNumber - Polygon zkEVM post
-
  - eth_getTransactionByHash - Polygon zkEVM post
-
  - zkevm_virtualBatchNumber - Polygon zkEVM post
-
  - eth_getCompilers - Polygon zkEVM post
-
  - eth_getUncleByBlockHashAndIndex - Polygon zkEVM post
-
  - eth_getUncleByBlockNumberAndIndex - Polygon zkEVM post
-

  - eth_getUncleCountByBlockHash - Polygon zkEVM post
-
  - eth_getUncleCountByBlockNumber - Polygon zkEVM post
-
  - eth_protocolVersion - Polygon zkEVM post
-
  - eth_blockNumber - Polygon zkEVM post
-
  - eth_getTransactionReceipt - Polygon zkEVM post
-
  - zkevm_batchNumberByBlockNumber - Polygon zkEVM post
-
  - zkevm_consolidatedBlockNumber - Polygon zkEVM post
-
  - zkevm_estimateFee API - Polygon zkEVM post
-
  - zkevm_estimateGasPrice API - Polygon zkEVM post

Arbitrum

- Arbitrum API Quickstart
-
  - Arbitrum SDK Examples
- Arbitrum API FAQ
-
  - Arbitrum vs. Ethereum API Differences
- Arbitrum API Endpoints
-
  - eth_call - Arbitrum post
-
  - eth_estimateGas - Arbitrum post
-
  - eth_accounts - Arbitrum post
-
  - eth_blockNumber - Arbitrum post
-
  - eth_chainId - Arbitrum post
-
  - eth_gasPrice - Arbitrum post
-
  - eth_getBalance - Arbitrum post
-
  - eth_getBlockTransactionCountByHash - Arbitrum post
-
  - eth_getBlockTransactionCountByNumber - Arbitrum post
-
  - eth_getCode - Arbitrum post
-
  - eth_getFilterChanges - Arbitrum post
-
  - eth_getFilterLogs - Arbitrum post
-
  - eth_getLogs - Arbitrum post
-
  - eth_getStorageAt - Arbitrum post
-
  - eth_getTransactionByBlockHashAndIndex - Arbitrum post
-
  - eth_getTransactionCount - Arbitrum post
-
  - eth_getUncleByBlockNumberAndIndex - Arbitrum post
-
  - eth_getUncleCountByBlockHash - Arbitrum post
-
  - eth_getUncleCountByBlockNumber - Arbitrum post
-
  - eth_newBlockFilter - Arbitrum post
-

-
  - eth_newFilter - Arbitrum post
-
  - eth_newPendingTransactionFilter - Arbitrum post
-
  - eth_uninstallFilter - Arbitrum post
-
  - net_listening - Arbitrum post
-
  - net_version - Arbitrum post
-
  - web3_clientVersion - Arbitrum post
-
  - web3_sha3 - Arbitrum post
-
  - eth_sendRawTransaction - Arbitrum post
-
  - eth_createAccessList - Arbitrum post
-
  - eth_maxPriorityFeePerGas - Arbitrum post
-
  - eth_feeHistory - Arbitrum post
-
  - eth_getBlockByHash - Arbitrum post
-
  - eth_getBlockByNumber - Arbitrum post
-
  - eth_getTransactionByBlockNumberAndIndex - Arbitrum post
-
  - eth_getTransactionByHash - Arbitrum post
-
  - eth_getProof - Arbitrum post
-
  - eth_getTransactionReceipt - Arbitrum post
-
  - eth_getUncleByBlockHashAndIndex - Arbitrum post
-
  - eth_subscribe
-
  - eth_unsubscribe

Optimism

- Optimism API Quickstart
-
  - Optimism SDK Examples
- Optimism API FAQ
-
  - Optimism Error Codes
- Optimism API Endpoints
-
  - eth_call - Optimism post
-
  - eth_estimateGas - Optimism post
-
  - eth_accounts - Optimism post
-
  - eth_blockNumber - Optimism post
-
  - eth_chainId - Optimism post
-
  - eth_gasPrice - Optimism post
-
  - eth_getBalance - Optimism post
-
  - eth_getBlockTransactionCountByHash - Optimism post
-
  - eth_getBlockTransactionCountByNumber - Optimism post
-

  - eth_getCode - Optimism post
-
  - eth_getFilterChanges - Optimism post
-
  - eth_getFilterLogs - Optimism post
-
  - eth_getLogs - Optimism post
-
  - eth_getStorageAt - Optimism post
-
  - eth_getTransactionByBlockHashAndIndex - Optimism post
-
  - eth_getTransactionByBlockNumberAndIndex - Optimism post
-
  - eth_getTransactionByHash - Optimism post
-
  - eth_getTransactionCount - Optimism post
-
  - eth_getTransactionReceipt - Optimism post
-
  - eth_getUncleByBlockHashAndIndex - Optimism post
-
  - eth_getUncleByBlockNumberAndIndex - Optimism post
-
  - eth_getUncleCountByBlockHash - Optimism post
-
  - eth_getUncleCountByBlockNumber - Optimism post
-
  - eth_newBlockFilter - Optimism post
-
  - eth_newFilter - Optimism post
-
  - eth_newPendingTransactionFilter - Optimism post
-
  - eth_protocolVersion - Optimism post
-
  - eth_sendRawTransaction - Optimism post
-
  - eth_syncing - Optimism post
-
  - eth_uninstallFilter - Optimism post
-
  - net_listening - Optimism post
-
  - net_version - Optimism post
-
  - web3_clientVersion - Optimism post
-
  - web3_sha3 - Optimism post
-
  - eth_getBlockByHash - Optimism post
-
  - eth_getBlockByNumber - Optimism post
-
  - eth_getProof - Optimism post
-
  - eth_subscribe
-
  - eth_unsubscribe

Base

- Base API Quickstart
- Base API FAQ
- Base API Endpoints
-
  - eth_accounts - Base post
-

  - eth_blockNumber - Base post
-
  - eth_call - Base post
-
  - eth_chainId - Base post
-
  - eth_estimateGas - Base post
-
  - eth_feeHistory - Base post
-
  - eth_gasPrice - Base post
-
  - eth_getBalance - Base post
-
  - eth_getBlockByHash - Base post
-
  - eth_getBlockByNumber - Base post
-
  - eth_getBlockTransactionCountByHash - Base post
-
  - eth_getBlockTransactionCountByNumber - Base post
-
  - eth_getCode - Base post
-
  - eth_getFilterChanges - Base post
-
  - eth_getFilterLogs - Base post
-
  - eth_getLogs - Base post
-
  - eth_getProof - Base post
-
  - eth_getStorageAt - Base post
-
  - eth_getTransactionByBlockHashAndIndex - Base post
-
  - eth_getTransactionByBlockNumberAndIndex - Base post
-
  - eth_getTransactionByHash - Base post
-
  - eth_getTransactionCount - Base post
-
  - eth_getTransactionReceipt - Base post
-
  - eth_getUncleByBlockHashAndIndex - Base post
-
  - eth_getUncleByBlockNumberAndIndex - Base post
-
  - eth_getUncleCountByBlockHash - Base post
-
  - eth_getUncleCountByBlockNumber - Base post
-
  - eth_maxPriorityFeePerGas - Base post
-
  - eth_newBlockFilter - Base post
-
  - eth_newFilter - Base post
-
  - eth_newPendingTransactionFilter - Base post
-
  - eth_protocolVersion - Base post
-
  - eth_sendRawTransaction - Base post
-
  - eth_syncing - Base post
-
  - eth_uninstallFilter - Base post
-

  - net_listening - Base post
-
  - web3_sha3 - Base post

✻ Solana

- Solana API Quickstart
- Solana API FAQ
- Solana API Endpoints
-
  - getAccountInfo post
-
  - simulateTransaction post
-
  - getBalance post
-
  - getBlock post
-
  - getBlockCommitment post
-
  - getBlockProduction post
-
  - getBlocks post
-
  - getBlocksWithLimit post
-
  - getBlockTime post
-
  - getClusterNodes post
-
  - getEpochInfo post
-
  - getEpochSchedule post
-
  - getFeeForMessage post
-
  - getFirstAvailableBlock post
-
  - getGenesisHash post
-
  - getHealth post
-
  - getHighestSnapshotSlot post
-
  - getIdentity post
-
  - getInflationGovernor post
-
  - getInflationRate post
-
  - getInflationReward post
-
  - getLargestAccounts post
-
  - getMaxRetransmitSlot post
-
  - getMaxShredInsertSlot post
-
  - getMinimumBalanceForRentExemption post
-
  - getMultipleAccounts post
-
  - getProgramAccounts post
-
  - getRecentPerformanceSamples post
-
  - getSignaturesForAddress post
-

Astar

  - eth_getTransactionByBlockHashAndIndex - Astar post
-
  - eth_getTransactionByBlockNumberAndIndex - Astar post
-
  - eth_getTransactionByHash - Astar post
-
  - eth_getTransactionCount - Astar post
-
  - eth_getUncleByBlockNumberAndIndex - Astar post
-
  - eth_sendRawTransaction - Astar post
-
  - net_version - Astar post
-
  - web3_clientVersion - Astar post
-
  - web3_sha3 - Astar post
-
  - eth_getLogs - Astar post
-
  - eth_getFilterChanges - Astar post
-
  - eth_getFilterLogs - Astar post
-
  - eth_newFilter - Astar post
-
  - eth_newPendingTransactionFilter - Astar post
-
  - eth_uninstallFilter - Astar post
-
  - eth_newBlockFilter - Astar post
-
  - eth_estimateGas - Astar post
-
  - eth_subscribe
-
  - eth_unsubscribe

STARKNET

- Starknet API Quickstart
- Starknet API FAQ
- Starknet API Endpoints
-
  - starknet_addDeclareTransaction post
-
  - starknet_getClassAt post
-
  - starknet_addDeployAccountTransaction post
-
  - starknet_getClassHashAt post
-
  - starknet_addInvokeTransaction post
-
  - starknet_getEvents post
-
  - starknet_blockHashAndNumber post
-
  - starknet_getNonce post
-
  - starknet_blockNumber post
-
  - starknet_getStateUpdate post
-
  - starknet_call post
-
  - starknet_getStorageAt post
-

  - starknet_chainId post
-
  - starknet_getTransactionByBlockIdAndIndex post
-
  - starknet_estimateFee post
-
  - starknet_getTransactionByHash post
-
  - starknet_getBlockTransactionCount post
-
  - starknet_getTransactionReceipt post
-
  - starknet_getBlockWithTxHashes post
-
  - starknet_pendingTransactions post
-
  - starknet_getBlockWithTxs post
-
  - starknet_syncing post
-
  - starknet_getClass post
-
  - starknet_estimateMessageFee post

Alchemy SDK Surface Overview

An in-depth look at the APIs available in the Alchemy SDK and how it differs from Ethers.js.

Alchemy SDK Quickstart

If you'd like to start writing code immediately with the SDK, go here!The Alchemy SDK is the most comprehensive, stable, and powerful Javascript SDK available today to interact with the blockchain.

It supports the exact same syntax and functionality of the Ethers.js AlchemyProvider and WebSocketProvider, making it a 1:1 mapping for anyone using the Ethers.js Provider. However, it adds a significant amount of improved functionality on top of Ethers, such as easy access to Alchemy's Enhanced and NFT APIs, robust WebSockets, and quality-of life improvements such as automated retries.

The SDK currently supports the following chains:

- Ethereum
- : Mainnet, Goerli
- Polygon
- : Mainnet, Mumbai
- Optimism
- : Mainnet, Goerli, Kovan
- Arbitrum
- : Mainnet, Goerli, Rinkeby
- Astar
- : Mainnet

API Surface

The Alchemy SDK currently supports five different namespaces, including:

- core
- : All commonly-used Ethers.js Provider methods and Alchemy Enhanced API methods
- nft
- : All Alchemy NFT API methods
- ws
- : All WebSockets methods
- transact
- : All Alchemy Transaction API methods
- notify

- : CRUD endpoints for modifying Alchemy Notify Webhooks
- debug
- : Methods to inspect and replay transactions and blocks

If you are already using Ethers.js, you should be simply able to replace the Ethers.js Provider object with alchemy.core and it should just work.

ENS Resolution

The Alchemy SDK now supports ENS names (e.g. vitalik.eth) for every parameter where you can pass in an Externally Owned Address, or user address (e.g. 0xd8dA6BF26964aF9D7eEd9e03E53415D37aA96045) ENS Name Resolution import {Alchemy ,AlchemySubscription }from 'alchemy-sdk' ;// Using default settings - pass in a settings object to specify your API key and network const alchemy = new Alchemy ();// Access standard Ethers.js JSON-RPC node request alchemy .core .getBlockNumber ().then (console .log);// Access Alchemy Enhanced API requests alchemy .core .getTokenBalances ('0x3f5CE5FBFe3E9af3971dD833D26bA9b5C936f0bE') .then (console .log);// Access the Alchemy NFT API alchemy .nft .getNftsForOwner ('vitalik.eth').then (console .log);// Access WebSockets and Alchemy-specific WS methods alchemy .ws .on ({method :AlchemySubscription .PENDING_TRANSACTIONS },res => console .log (res)); The Alchemy SDK also supports a number of Ethers.js objects that streamline the development process:

- Utils
- : Equivalent to
- ethers.utils
- , this provides a number of common Ethers.js utility methods for developers.*Interface
-
    - : Found in
-
    - Utils.Interface
-
    - , this class abstracts the encoding and decoding required to interact with contracts on the Ethereum network.
- Contract
- : An abstraction for smart contract code deployed to the blockchain.
- ContractFactory
- : Allows developers to build a
- Contract
- object.
- Wallet
- : An implementation of
- Signer
- that can sign transactions and messages using a private key as a standard Externally Owned Account.

Alchemy Settings

An AlchemySettings object can be passed on instantiation to the Alchemy object, with the following optional parameters:

- apiKey
- : API key that can be found in the Alchemy dashboard. Defaults to
- demo
- : a rate-limited public key.
- network
- : Name of the network. Defaults to
- Network.ETH_MAINNET
- maxRetries
- : The maximum number of retries to attempt if a request fails. Defaults to 5.
- url
- : Optional URL endpoint to use for all requests. Setting this field will override the URL generated by the
- network
- and
- apiKey
- fields.
- authToken
- : Alchemy auth token required to use the Notify API. This token can be found in the Alchemy Dashboard on theWebhooks tab
- .
- batchRequests
- : Optional setting that automatically batches and sends json-rpc requests for higher throughput and reduced network

IO. Defaults to false.

- requestTimeout
- : Optional setting that sets the timeout for requests in milliseconds for the NFT and Notify namespaces. Defaults to no timeout.

Alchemy Core

The core package contains all commonly-usedEthers.js Provider methods. If you are already using Ethers.js, you should be simply able to replace the Ethers.js Provider object with alchemy.core and it should just work.

It also includes the majority of Alchemy Enhanced APIs, including:

- getTokenMetadata()
- : Get the metadata for a token contract address.
- getTokenBalances()
- : Gets the token balances for an owner given a list of contracts.
- getAssetTransfers()
- : Get transactions for specific addresses.
- getTransactionReceipts()
- : Gets all transaction receipts for a given block.

You will also find the following utility methods:

- findContractDeployer()
- : Find the contract deployer and block number for a given contract address.

Accessing the full Ethers.js Provider

To keep the package clean, we don't support certain uncommonly-used Ethers.js Provider methods as top-level methods the Alchemy core namespace - for example, provider.formatter . If you'd like to access these methods, simply use the alchemy.config.getProvider() function to configure the Ethers.jsAlchemyProvider and return it.

SDK.js import {Alchemy }from 'alchemy-sdk' ;const alchemy = new Alchemy ();async function runAlchemy () {const ethersProvider = await alchemy .config .getProvider ();console .log (ethersProvider .formatter); }runAlchemy ();

Alchemy WebSockets

In addition to the built-in Ethers.js listeners, the Alchemy SDK includes support foAlchemy's Subscription API . This allows you to subscribe to events and receive updates as they occur.

The alchemy.ws instance can be used can be used like the standard Ethers.jsWebSocketProvider to add listeners for Alchemy events:

WebSockets.js import {Alchemy }from 'alchemy-sdk' ;const alchemy = new Alchemy ();// Listen to all new pending transactions. alchemy .ws .on ({method :'alchemy_pendingTransactions' },res => console .log (res));// Listen to only the next transaction on the USDC contract. alchemy .ws .once ({method :'alchemy_pendingTransactions' ,toAddress :'vitalik.eth' },res => console .log (res));// Remove all listeners. alchemy .ws .removeAllListeners (); The SDK brings multiple improvements to ensure correct WebSocket behavior in cases of temporary network failure or dropped connections. As with any network connection, you should not assume that a WebSocket will remain open forever without interruption, but correctly handling dropped connections and reconnection by hand can be challenging to get right. alchemy-sdk automatically handles these failures with no configuration necessary.

The main benefits are:

- Resilient event delivery:
- Unlike standard Web3.js or Ethers.js, you will not permanently miss events which arrive while the backing WebSocket is temporarily down. Instead, you will receive these events as soon as the connection is reopened. Note that if the connection is down for more than 120 blocks (approximately 20 minutes), you may still miss some events that were not part of the most recent 120 blocks.
- Lowered rate of failure:
- Compared to standard Web3.js or Ethers.js, there are fewer failures when sending requests over the WebSocket while the connection is down. Alchemy Web3 will attempt to send the requests once the connection is reopened. Note that it is still possible, with a lower likelihood, for outgoing requests to be lost, so you should still have error handling as with any network request.

Alchemy Transact

The transact namespace contains methods used for sending transactions. The unique methods to the transact namespace are:

- sendPrivateTransaction()
- : Send a private transaction through Flashbots.
- cancelPrivateTransaction()
- : Cancel a private transaction sent with Flashbots.

The transact namespace also aliases over several commonly used methods from the core namespace for convenience:

- getTransaction()
- : Returns the transaction for the given transaction hash.
- sendTransaction()
- : Sends a standard transaction to the network to be mined.
- waitForTransaction()
- : Waits for a transaction to be mined and returns the transaction receipt.

Alchemy NFT API

The SDK currently supports the followingNFT API endpoints under the alchemy.nft namespace:

- getNftMetadata()
- : Get the NFT metadata for an NFT contract address and tokenId.
- getNftMetadataBatch()
- : Get the NFT metadata for multiple NFT contract addresses/token id pairs.
- getContractMetadata()
- : Get the metadata associated with an NFT contract
- getContractsForOwner()
- : Get all NFT contracts that the provided owner address owns.
- getNftsForOwner()
- : Get NFTs for an owner address.
- getNftsForOwnerIterator()
- : Get NFTs for an owner address as an async iterator (handles paging automatically).
- getNftsForContract()
- : Get all NFTs for a contract address.
- getNftsForContractIterator()
- : Get all NFTs for a contract address as an async iterator (handles paging automatically).
- getOwnersForNft()
- : Get all the owners for a given NFT contract address and a particular token ID.
- getOwnersForContract()
- : Get all the owners for a given NFT contract address.
- verifyNftOwnership()
- : Check whether the provided owner address owns the provided NFT contract addresses.
- isSpamContract()
- : Check whether the given NFT contract address is a spam contract as defined by Alchemy (see theNFT API FAQ
-)
- getSpamContracts()
- : Returns a list of all spam contracts marked by Alchemy.
- refreshNftMetadata()
- : Refresh the cached NFT metadata for a contract address and a single tokenId.
- refreshContract()
- : Enqueues the specified contract address to have all token ids' metadata refreshed.
- getFloorPrice()
- : Return the floor prices of a NFT contract by marketplace.
- computeRarity()
- : Get the rarity of each attribute of an NFT.
- getNftSales()
- : Returns NFT sales that have happened through on-chain marketplaces.
- summarizeNftAttributes()
- : Get the summary of attribute prevalence for all NFTs in a contract.
- searchContractMetadata()
- : Search for a keyword across metadata of all ERC-721 and ERC-1155 smart contracts.

Pagination

The Alchemy NFT endpoints return 100 results per page. To get the next page, you can pass in the pageKey returned by the previous call. To simplify paginating through all results, the SDK provides the getNftsIterator() and getNftsForCollectionIterator() functions that automatically paginate through all NFTs and yields them via an AsyncIterable .

Here's an example of how to paginate through all the NFTs in Vitalik's ENS address:

JavaScript import {Alchemy }from 'alchemy-sdk' ;const alchemy = new Alchemy ();async function main () {const ownerAddress = 'vitalik.eth' ;for await (const nft of alchemy .nft .getNftsForOwnerIterator (ownerAddress)) {console .log ('ownedNft:' ,nft); } }main ();

SDK vs NFT API Differences

The NFT API in the SDK standardizes response types to reduce developer friction, but note this results in some differences compared to the Alchemy REST endpoints:

- Methods referencing
- Collection
- have been renamed to use the name
- Contract
- for greater accuracy: e.g.
- getNftsForContract
- .
- Some methods have different naming that the REST API counterparts in order to provide a consistent API interface (
- e.g.
- getNftsForOwner()
- is
- alchemy_getNfts
- ,
- getOwnersForNft()
- is
- alchemy_getOwnersForToken
-).
- SDK standardizes to
- omitMetadata
- parameter (vs.
- withMetadata
-).
- Standardization to
- pageKey
- parameter for pagination (vs.
- nextToken
- /
- startToken
-)
- Empty
- TokenUri
- fields are omitted.
- Token ID is always normalized to an integer string on
- BaseNft
- and
- Nft
- .
- Some fields omitted in the REST response are included in the SDK response in order to return an
- Nft
- object.
- Some fields in the SDK's
- Nft
- object are named differently than the REST response.

Alchemy Notify

TheAlchemy Notify API helps developers set up webhooks in their apps. The namespace provides methods to programmatically create, read, update, and delete your webhooks along with typings for the different webhooks. To learn more about Webhooks, please refer to theAlchemy documentation .

Methods on the NotifyNamespace can be accessed via alchemy.notify . To use the methods, you must include your team's auth token in the authToken field of AlchemySettings when instantiating the SDK. The auth token can be found on the Alchemy Dashboard in the Notify Tab.

Methods include:

- getAllWebhooks()
- : Get all webhooks on your team.
- getAddresses()
- : Get all addresses tracked for the provided Address Activity Webhook.
- getNftFilters()
- : Get all NFT filters tracked for the provided NFT Activity Webhook.
- createWebhook()
- : Create a new webhook.
- updateWebhook()
- : Update an existing webhook's active status or tracked addresses and NFT filters.
- deleteWebhook()
- : Delete the provided webhook.

Alchemy Debug

Methods on the DebugNamespace can be accessed via alchemy.debug . These methods are used for inspecting and debugging transactions.

Methods include:

- traceCall()
- : Run an eth_call with the context of the provided block execution using the final state of the parent block as the base.
- traceTransaction()
- : Run the transaction in the exact same manner as it was executed on the network. It will replay any transaction that may have been executed prior to this one before it and will then attempt to execute the transaction that corresponds to the given hash.
- traceBlock()
- : Replay a block that has already been mined.

Documentation

The SDK is documented via tsdoc comments in the source code. The generated types and documentation are included when using an IDE. To browse the documentation separately, you can view the generated API interfaces in etc/alchemy-sdk.api.md . You can view generated Markdown files for each endpoint in the docs-md directory, or as a webpage by opening docs/index.html in your browser.

Usage Examples

Below are a few usage examples.

More Examples Available

For more examples using the Alchemy SDKvisit here .

Getting the NFTs owned by an address

```
Alchemy-SDK import {Alchemy ,NftExcludeFilters }from 'alchemy-sdk' ;const alchemy = new Alchemy ();// Get how many
NFTs an address owns. alchemy .nft .getNftsForOwner ('vitalik.eth' ).then (nfts => {console .log (nfts .totalCount ); });// Get
all the image urls for all the NFTs an address owns. async function main () {for await (const nft of alchemy .nft
.getNftsForOwnerIterator ('vitalik.eth' )) {console .log (nft .media ); } }main ();// Filter out spam NFTs. alchemy .nft
.getNftsForOwner ('vitalik.eth' , {excludeFilters : [NftExcludeFilters .SPAM ] }) .then (console .log );
```

Getting all the owners of the BAYC NFT

```
Alchemy-SDK import {Alchemy }from 'alchemy-sdk' ;const alchemy = new Alchemy ();// Bored Ape Yacht Club contract
address. const baycAddress = '0xBC4CA0EdA7647A8aB7C2061c2E118A18a936f13D' ;async function main () {for await
(const nft of alchemy .nft .getNftsForContractIterator (baycAddress , {// Omit the NFT metadata for smaller payloads.
omitMetadata :true })) {await alchemy .nft .getOwnersForNft (nft .contract .address ,nft .tokenId ) .then (response => console
.log ('owners:' ,response .owners ,'tokenId:' ,nft .tokenId ) ); } }main ();
```

Get all outbound transfers for a provided address

```
Alchemy-SDK import {Alchemy }from 'alchemy-sdk' ;const alchemy = new Alchemy ();alchemy .core .getTokenBalances
('vitalik.eth' ).then (console .log );
```

Utils Object

Alchemy provides a consolidated Utils object that supports a number of Debug

Methods on the most commonly used utils from Ethers. These should DebugNamespace can be able to be accessed via alchemy.debug . These methods are used as a drop-in replacement for ethers.utils . inspecting and debugging transactions.

- dnsEncode
- hashMessage
- id
- isValidName
- namehash
- arrayify
- concat
- hexConcat
- hexDataSlice
- hexDataLength
- hexlify
- hexStripZeros
- hexValue
- hexZeroPad
- isBytes
- isBytesLike
- isHexString
- joinSignature
- zeroPad
- splitSignature
- stripZeros
- formatEther

Questions and Feedback

If you have any questions, issues, or feedback, please file an issue onGitHub , or drop us a message on ourDiscord channel for the SDK.

Feature Requests

We'd love your thoughts on what would improve your web3 dev process the most! If you have 5 minutes, tell us what you want at ourFeature Request feedback form and we'd love to build it for you:

https://alchemyapi.typeform.com/sdk-feedback If you have any questions, issues, or feedback, please file an issue onGitHub , or drop us a message on ourDiscord channel for the SDK.

Updated 4 months ago

How to Manage a Multichain Project Using Alchemy SDKAlchemy SDK vs. Raw API Methods Did this page help you?Yes No