

Introduction

- [Alchemy API Reference Overview](#)
- - [Chain APIs Overview](#)
 - [Enhanced APIs Overview](#)
- [Alchemy Quickstart Guide](#)

Resources

- [FAQ](#)
- - [Feature Support By Chain](#)
 - [Throughput](#)
 - [Batch Requests](#)
 - [Gas Limits](#)
 - [Error Reference](#)
- [Compute Units](#)
- - [Pricing Plans](#)
 - [Compute Unit Costs](#)

NFT API

- [NFT API Quickstart](#)
- [NFT API Endpoints Overview](#)
- [NFT API FAQ](#)
- [Ownership & Token Gating](#)
- - [getNFTsForOwner get](#)
 - [getOwnersForNFT get](#)
 - [getOwnersForContract get](#)
 - [isHolderOfContract get](#)
 - [getContractsForOwner get](#)
 - [getCollectionsForOwner get](#)
- [NFT Metadata Access](#)
- - [getNFTMetadata get](#)
 - [getNFTMetadataBatch post](#)
 - [getContractMetadata get](#)
 - [getCollectionMetadata get](#)
 - [invalidateContract get](#)
 - [getContractMetadataBatch post](#)
 - [getNFTsForContract get](#)
 - [getNFTsForCollection get](#)
 - [searchContractMetadata get](#)
 - [refreshNftMetadata post](#)

- [Spam Detection](#)
- - [getSpamContracts.get](#)
 - [isSpamContract.get](#)
 - [isAirdropNFT.get](#)
 - [reportSpam.get](#)
- [Rarity Data](#)
- - [summarizeNFTAttributes.get](#)
 - [computeRarity.get](#)
- [Sales & Marketplace Data](#)
- - [getFloorPrice.get](#)
 - [getNFTSales.get](#)
- [NFT API V2 to V3 Migration Guide](#)
- - [NFT API V2 vs. V3 Endpoint Differences](#)
- [NFT API V2 Methods \(Older Version\)](#)
- - [getNFTs.get](#)
 - [getNFTMetadata.get](#)
 - [getNFTMetadataBatch.post](#)
 - [getContractMetadata.get](#)
 - [getContractMetadataBatch.post](#)
 - [getNFTsForCollection.get](#)
 - [getOwnersForToken.get](#)
 - [getOwnersForCollection.get](#)
 - [getSpamContracts.get](#)
 - [isSpamContract.get](#)
 - [isAirdrop.get](#)
 - [invalidateContract.get](#)
 - [getFloorPrice.get](#)
 - [computeRarity.get](#)
 - [searchContractMetadata.get](#)
 - [summarizeNFTAttributes.get](#)
 - [isHolderOfCollection.get](#)
 - [getNFTSales.get](#)
 - [getContractsForOwner.get](#)
 - [reportSpam.get](#)

Transfers API (Tx History)

- [Transfers API Quickstart](#)
- [Transfers API Endpoints](#)

-
- [alchemy_getAssetTransfers.post](#)

Transaction Receipts API

- [Transaction Receipts Endpoints](#)
-
- [alchemy_getTransactionReceipts.post](#)

Token API

- [Token API Quickstart](#)
- [Token API Endpoints](#)
-
- [alchemy_getTokenBalances.post](#)
-
- [alchemy_getTokenMetadata.post](#)
-
- [alchemy_getTokenAllowance.post](#)

Subgraphs

- [Subgraphs Quickstart](#)
- [Supported Subgraph Chains](#)
- [Developing a Subgraph](#)
-
- [Graph CLI](#)
-
- [Creating a Subgraph](#)
-
- [Project Structure](#)
-
- [Data Sources](#)
-
- [Writing Mappings](#)
- [Moving your Subgraph to Production](#)
-
- [Deploying a Subgraph](#)
-
- [Subgraph Versioning](#)
-
- [Querying a Subgraph](#)
-
- [Deleting a Subgraph](#)
-
- [Direct Database Access](#)
- [Community subgraphs](#)

Webhooks

- [Notify API Quickstart](#)
-
- [Notify Tutorials and Applications](#)
-
- [Notify API FAQ](#)
- [Custom Webhooks Quickstart](#)
-
- [Custom Webhooks FAQ](#)
-
- [Custom Webhooks GraphQL Examples](#)
-
- [Custom Webhook Filters](#)
-
- [Custom Webhook Variables](#)
- [Custom Webhook API Methods](#)
-

- - [Read Variable Elements get](#)
- - [Create a Variable post](#)
- - [Delete a Variable delete](#)
- - [Update a Variable patch](#)
- [Notify API Methods](#)
- - [Get all webhooks get](#)
- - [Get all addresses for an Address Activity webhook get](#)
- - [Create webhook post](#)
- - [Add and remove webhook addresses patch](#)
- - [Replace webhook addresses put](#)
- - [Update webhook status put](#)
- - [Update webhook NFT filters patch](#)
- - [Update NFT metadata webhook filters patch](#)
- - [Get all webhook NFT filters get](#)
- - [Delete webhook delete](#)
- [Webhook Types](#)
- - [Custom Webhook](#)
- - [Address Activity Webhook](#)
- - [Mined Transaction Webhook](#)
- - [Dropped Transaction Webhook](#)
- - [NFT Activity Webhook](#)
- - [NFT Metadata Updates Webhook](#)

Websockets

- [Subscription API Quickstart](#)
- [Best Practices for Using WebSockets in Web3](#)
- [Subscription API Endpoints](#)
- - [alchemy_minedTransactions](#)
- - [alchemy_pendingTransactions](#)
- - [newPendingTransactions](#)
- - [newHeads](#)
- - [logs](#)

Trace API

- [Trace API Quickstart](#)
- [Trace API Endpoints](#)
- - [trace_block post](#)
- - [trace_call post](#)
-

- [trace_get post](#)
-
- [trace_rawTransaction post](#)
-
- [trace_replayBlockTransactions post](#)
-
- [trace_replayTransaction post](#)
-
- [trace_transaction post](#)
-
- [trace_filter post](#)
- [Trace API Resources](#)
-
- [What are EVM Traces?](#)
-
- [Trace API vs. Debug API](#)
-
- [What is trace_transaction?](#)
-
- [What is trace_block?](#)
-
- [What is trace_filter?](#)
-
- [trace_call vs debug_traceCall](#)

Debug API

- [Debug API Quickstart](#)
- [Debug API Endpoints](#)
-
- [debug_traceCall post](#)
-
- [debug_traceTransaction post](#)
-
- [debug_traceBlockByNumber post](#)
-
- [debug_traceBlockByHash post](#)

ACCOUNT ABSTRACTION

- [Bundler API Quickstart](#)
- [Bundler API Endpoints](#)
-
- [eth_getUserOperationReceipt post](#)
-
- [eth_supportedEntryPoints post](#)
-
- [eth_getUserOperationByHash post](#)
-
- [eth_sendUserOperation post](#)
-
- [rundler_maxPriorityFeePerGas post](#)
-
- [eth_estimateUserOperationGas post](#)
- [Bundler API Fee Logic](#)
- [Factory Addresses](#)
- [Gas Manager Coverage API Quickstart](#)
- [Gas Manager Coverage API Endpoints](#)
-
- [alchemy_requestPaymasterAndData post](#)
-
- [alchemy_requestGasAndPaymasterAndData post](#)
- [Gas Manager Coverage API Fee Logic](#)
- [Gas Manager Deployment Addresses](#)
- [UserOperation Simulation Endpoints](#)
-
- [alchemy_simulateUserOperationAssetChanges post](#)

- [AA-SDK](#)
- [Account Abstraction FAQ](#)

Embedded Accounts

- [Accounts API Endpoints](#)
-
- [Create Account post](#)
-
- [Send Auth Email post](#)
-
- [Authenticate User post](#)
-
- [Get User post](#)
-
- [Sign Message post](#)
-
- [Register New Authenticator post](#)

Gas Manager Admin API

- [Gas Manager Admin API Quickstart](#)
- [Gas Manager Admin API Endpoints](#)
-
- [Create Policy post](#)
-
- [Get Policy get](#)
-
- [Delete Policy delete](#)
-
- [Replace Policy put](#)
-
- [Get All Policies get](#)
-
- [Update Policy Status put](#)
-
- [Get Policy Stats get](#)
-
- [Get Sponsorships get](#)

Alchemy Transact

- [Transact Quickstart](#)
- [Reinforced Transactions](#)
- [Transaction Simulation](#)
-
- [Asset Changes](#)
-
- [Execution Simulation](#)
-
- [Bundle Simulation](#)
-
- [Transaction Simulation Examples](#)
-
- [Transaction Simulation FAQs](#)
- [Transaction Simulation Endpoints](#)
-
- [alchemy_simulateAssetChanges post](#)
-
- [alchemy_simulateAssetChangesBundle post](#)
-
- [alchemy_simulateExecution post](#)
-
- [alchemy_simulateExecutionBundle post](#)
- [Gas Optimized Transactions](#)
-
- [alchemy_getGasOptimizedTransactionStatus post](#)

-
- [alchemy_sendGasOptimizedTransaction post](#)
- [Private Transactions](#)
-
- [eth_cancelPrivateTransaction post](#)
-
- [eth_sendPrivateTransaction post](#)

Alchemy SDK

- [Alchemy SDK Quickstart](#)
-
- [How to use Alchemy SDK with Typescript](#)
-
- [Examples Using the Alchemy SDK](#)
-
- [How to Manage a Multichain Project Using Alchemy SDK](#)
- [Alchemy SDK Surface Overview](#)
-
- [Alchemy SDK vs. Raw API Methods](#)
- [SDK Core Methods](#)
-
- [call - SDK](#)
-
- [send - SDK](#)
-
- [estimateGas - SDK](#)
-
- [findContractDeployer - SDK](#)
-
- [getBalance - SDK](#)
-
- [getBlock - SDK](#)
-
- [getBlockNumber - SDK](#)
-
- [getBlockWithTransactions - SDK](#)
-
- [getCode - SDK](#)
-
- [getFeeData - SDK](#)
-
- [getGasPrice - SDK](#)
-
- [getLogs - SDK](#)
-
- [getStorageAt - SDK](#)
-
- [getTokenBalances - SDK](#)
-
- [getTokenMetadata - SDK](#)
-
- [getTokensForOwner - SDK](#)
-
- [getTransactionCount - SDK](#)
-
- [getTransactionReceipt - SDK](#)
-
- [getTransactionReceipts - SDK](#)
-
- [isContractAddress - SDK](#)
-
- [getAssetTransfers - SDK](#)
- [SDK NFT Methods](#)
-
- [getNftsForOwner - SDK](#)
-
- [getNftMetadata - SDK](#)

-
- [getNftMetadataBatch - SDK](#)
-
- [refreshNftMetadata - SDK](#)
-
- [getNftSales - SDK](#)
-
- [searchContractMetadata - SDK](#)
-
- [summarizeNftAttributes - SDK](#)
-
- [getNftsForOwnerIterator - SDK](#)
-
- [getNftsForContractIterator - SDK](#)
-
- [getContractMetadata - SDK](#)
-
- [getNftsForContract - SDK](#)
-
- [getTransfersForOwner - SDK](#)
-
- [getTransfersForContract - SDK](#)
-
- [getMintedNfts - SDK](#)
-
- [getOwnersForNft - SDK](#)
-
- [getOwnersForContract - SDK](#)
-
- [getSpamContracts - SDK](#)
-
- [isSpamContract - SDK](#)
-
- [refreshContract - SDK](#)
-
- [getContractsForOwner - SDK](#)
-
- [getFloorPrice - SDK](#)
-
- [computeRarity - SDK](#)
-
- [verifyNftOwnership - SDK](#)
- [SDK Transact Methods](#)
-
- [getTransaction - SDK](#)
-
- [sendTransaction - SDK](#)
-
- [sendPrivateTransaction - SDK](#)
-
- [cancelPrivateTransaction - SDK](#)
-
- [waitForTransaction - SDK](#)
-
- [estimateGas - SDK](#)
-
- [getMaxPriorityFeePerGas - SDK](#)
-
- [simulateAssetChanges - SDK](#)
-
- [simulateAssetChangesBundle - SDK](#)
-
- [simulateExecution - SDK](#)
-
- [simulateExecutionBundle - SDK](#)
- [SDK Debug Methods](#)
-
- [traceCall - SDK](#)

- - [traceTransaction - SDK](#)
- - [traceBlock - SDK](#)
- [SDK Notify Methods](#)
- - [getAllWebhooks - SDK](#)
- - [getAddresses - SDK](#)
- - [getNftFilters - SDK](#)
- - [createWebhook - SDK](#)
- - [updateWebhook - SDK](#)
- - [deleteWebhook - SDK](#)
- [SDK WebSockets Endpoints](#)
- [SDK Ethers Utils](#)
- - [arrayify](#)
- - [formatUnits](#)
- - [concat](#)
- - [hexConcat](#)
- - [dnsEncode](#)
- - [hexDataLength](#)
- - [formatEther](#)
- - [hexDataSlice](#)
- - [hexStripZeros](#)
- - [hashMessage](#)
- - [isHexString](#)
- - [isValidName](#)
- - [joinSignature](#)
- - [splitSignature](#)
- - [toUtf8Bytes](#)
- - [hexValue](#)
- - [toUtf8String](#)
- - [hexZeroPad](#)
- - [zeroPad](#)
- - [hexlify](#)
- - [id](#)
- - [isBytes](#)
- - [isBytesLike](#)
- - [Interface](#)
-

- [namehash](#)
-
- [parseEther](#)
-
- [parseUnits](#)
-
- [stripZeros](#)
- [Alchemy SDK V2 to V3 Migration Guide](#)
-
- [Alchemy SDK V2 vs. V3 Method Differences](#)
- [SDK V2 Methods](#)
-
- [call - SDK](#)
-
- [getAssetTransfers - SDK](#)
-
- [getMintedNfts - SDK](#)
-
- [verifyNftOwnership - SDK](#)
-
- [getOwnersForNft - SDK](#)
-
- [computeRarity - SDK](#)
-
- [getTransfersForContract - SDK](#)
-
- [getNftsForOwner - SDK](#)
-
- [refreshContract - SDK](#)
-
- [getOwnersForContract - SDK](#)
-
- [getFloorPrice - SDK](#)
-
- [isSpamContract - SDK](#)
-
- [findContractDeployer - SDK](#)
-
- [getSpamContracts - SDK](#)
-
- [getGasPrice - SDK](#)
-
- [getBalance - SDK](#)
-
- [getBlock - SDK](#)
-
- [getBlockWithTransactions - SDK](#)
-
- [estimateGas - SDK](#)
-
- [getBlockNumber - SDK](#)
-
- [getCode - SDK](#)
-
- [getFeeData - SDK](#)
-
- [getLogs - SDK](#)
-
- [getNftMetadataBatch - SDK](#)
-
- [getTokensForOwner - SDK](#)
-
- [getStorageAt - SDK](#)
-
- [getTokenBalances - SDK](#)
-
- [getTransactionCount - 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](#)

- [getNftsForContract - SDK](#)

Ethereum

- [Ethereum API Quickstart](#)
- [Ethereum API FAQ](#)
- - [Ethereum Developer Guide to the Merge](#)
- - [How to decode an eth_call response](#)
- - [How do I distinguish between a contract address and a wallet address?](#)
- [Ethereum API Endpoints](#)
- - [eth_blockNumber - Ethereum post](#)
- - [eth_getBalance - Ethereum post](#)
- - [eth_getLogs - Ethereum post](#)
- - [eth_chainId - Ethereum post](#)
- - [eth_getBlockByNumber - Ethereum post](#)
- - [eth_accounts - Ethereum post](#)
- - [eth_feeHistory - Ethereum post](#)
- - [eth_estimateGas - Ethereum post](#)
- - [eth_gasPrice - Ethereum post](#)
- - [eth_getBlockTransactionCountByHash - Ethereum post](#)
- - [eth_getBlockReceipts - Ethereum post](#)
- - [eth_getBlockTransactionCountByNumber - Ethereum post](#)
- - [eth_getCode - Ethereum post](#)
- - [eth_getProof - Ethereum post](#)
- - [eth_getStorageAt - Ethereum post](#)
- - [eth_getTransactionByBlockHashAndIndex - Ethereum post](#)
- - [eth_getTransactionByHash - Ethereum post](#)
- - [eth_getTransactionCount - Ethereum post](#)
- - [eth_getTransactionReceipt - Ethereum post](#)
- - [eth_getUncleByBlockHashAndIndex - Ethereum post](#)
- - [eth_getUncleByBlockNumberAndIndex - Ethereum post](#)
- - [eth_getUncleCountByBlockHash - Ethereum post](#)
- - [eth_getUncleCountByBlockNumber - Ethereum post](#)
- - [eth_maxPriorityFeePerGas - Ethereum post](#)
- - [eth_protocolVersion - Ethereum post](#)
- - [eth_sendRawTransaction - Ethereum post](#)
- - [net_listening - Ethereum post](#)

- [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](#)
-

- [eth_getFilterChanges](#) - Polygon PoS post
-
- [eth_getFilterLogs](#) - Polygon PoS post
-
- [eth_getLogs](#) - Polygon PoS post
-
- [eth_getRootHash](#) - Polygon PoS post
-
- [eth_getSignersAtHash](#) - Polygon PoS post
-
- [eth_getStorageAt](#) - Polygon PoS post
-
- [eth_getTransactionByBlockHashAndIndex](#) - Polygon PoS post
-
- [eth_getTransactionByBlockNumberAndIndex](#) - Polygon PoS post
-
- [eth_getTransactionByHash](#) - Polygon PoS post
-
- [eth_getTransactionCount](#) - Polygon PoS post
-
- [eth_getTransactionReceipt](#) - Polygon PoS post
-
- [eth_getTransactionReceiptsByBlock](#) - Polygon PoS post
-
- [eth_sendRawTransaction](#) - Polygon PoS post
-
- [eth_uninstallFilter](#) - Polygon PoS post
-
- [net_listening](#) - Polygon PoS post
-
- [eth_getUncleCountByBlockHash](#) - Polygon PoS post
-
- [eth_getUncleCountByBlockNumber](#) - Polygon PoS post
-
- [eth_newBlockFilter](#) - Polygon PoS post
-
- [eth_newFilter](#) - Polygon PoS post
-
- [eth_newPendingTransactionFilter](#) - Polygon PoS post
-
- [web3_clientVersion](#) - Polygon PoS post
-
- [eth_createAccessList](#) - Polygon PoS post
-
- [eth_blockNumber](#) - Polygon PoS post
-
- [bor_getSignersAtHash](#) - Polygon PoS post
-
- [net_version](#) - Polygon PoS post
-
- [eth_getProof](#) - Polygon PoS post
-
- [eth_getUncleByBlockNumberAndIndex](#) - Polygon PoS post
-
- [eth_subscribe](#) - Polygon PoS
-
- [eth_unsubscribe](#) - Polygon PoS

Polygon zkEVM

- [Polygon zkEVM API Quickstart](#)
- [Polygon zkEVM API FAQ](#)
-
- [What is the difference between Polygon zkEVM and Ethereum?](#)
-
- [What is the difference between Polygon zkEVM and Polygon PoS?](#)
- [Polygon zkEVM Endpoints](#)
-

- [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](#)
-

- [getSignatureStatuses post](#)
-
- [getSlot post](#)
-
- [getSlotLeader post](#)
-
- [getSlotLeaders post](#)
-
- [getSupply post](#)
-
- [getTokenAccountBalance post](#)
-
- [getTokenAccountsByOwner post](#)
-
- [getTokenSupply post](#)
-
- [getTransaction post](#)
-
- [getVersion post](#)
-
- [getVoteAccounts post](#)
-
- [isBlockhashValid post](#)
-
- [minimumLedgerSlot post](#)
-
- [sendTransaction post](#)
-
- [requestAirdrop post](#)
-
- [getBlockHeight post](#)
-
- [getRecentBlockhash post](#)

Astar

- [Astar API Quickstart](#)
- [Astar API FAQ](#)
- [Astar API Endpoints](#)
-
- [eth_accounts - Astar post](#)
-
- [eth_getTransactionReceipt - Astar post](#)
-
- [eth_maxPriorityFeePerGas - Astar post](#)
-
- [eth_blockNumber - Astar post](#)
-
- [eth_call - Astar post](#)
-
- [eth_chainId - Astar post](#)
-
- [eth_gasPrice - Astar post](#)
-
- [eth_getBalance - Astar post](#)
-
- [eth_getBlockByHash - Astar post](#)
-
- [eth_getBlockByNumber - Astar post](#)
-
- [eth_getBlockTransactionCountByHash - Astar post](#)
-
- [eth_getBlockTransactionCountByNumber - Astar post](#)
-
- [eth_getCode - Astar post](#)
-
- [eth_getStorageAt - Astar post](#)
-

- [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](#)

Notify API Quickstart

The Notify API manages transaction notifications using webhooks.

Alchemy Notify is used to subscribe to event notifications that occur on your application. You create a webhook to receive notifications on different types of on-chain activity. Here are a few quick-links:

- [Webhook Types](#)
- [How to Set up Webhooks](#)

All CSS

```
/ dont_have_api_sec_start / .api_key_instruct_ban{ background: #F5FCFF; border: 1px solid rgba(207, 217, 240, 0.2);
border-radius: 12px; -webkit-border-radius: 12px; display: flex; flex-wrap: wrap; padding: 33px; } .api_key_instruct_ban_lft
h3{ font-size: 24px; line-height: 1.3; letter-spacing: -0.03em; font-weight: 700; font-family: 'Inter', sans-serif; margin-bottom:
7px; margin-top: 0px; } .api_key_instruct_ban_lft h3:last-child{ margin-bottom: 0; } .api_key_instruct_ban_lft p{ font-size:
14px; line-height: 1.3; color: #000000; font-family: 'Inter', sans-serif; font-weight: 400; } .gt_startd_vbtn{ display: inline-block;
color: #FFFFFF !important; background: linear-gradient(126.33deg, #36BEFF 5.38%, #733FF1 108.32%); border-radius:
6px; font-size: 16px; line-height: 1.3; font-weight: 600; font-family: 'Inter', sans-serif; padding: 9px 16px; }
.gt_startd_vbtn:hover{ background: #000; color: #fff; } .api_key_instruct_ban_lft{ flex-basis: 60%; max-width: 60%; padding-
right: 15px; } .api_key_instruct_ban_rt{ flex-basis: 40%; max-width: 40%; padding-left: 15px; align-self: center; } /
dont_have_api_sec_end /
```

```
/* ===== responsive css ===== */
```

```
@media(min-width:1025px) {}
```

```
@media(max-width:1199px) {
```

```
  .api_main {
    max-width: 100%;
```

```
  }
```

```
  .api_main_cont ul li a {
    width: 100%;
```

```
  }
```

```
  .left_icon .evm_part {
    width: 73%;
    margin: 22px auto auto auto;
```

```
  }
```

```
  .api_main_cont ul li a {
    padding: 10px 18px;
    border-radius: 12px;
```

```
  }
```

```
  .api_main_cont ul li a:hover::before {
    border-radius: 12px;
```

```
  }
```

```
  .learn_box,
```

```
  .lear_outer {
    height: 100%;
```

```

}
.navbar-nav>li>a {
  font-size: 15px;
  padding: 8px 10px;
}
.top_header_links ul li a{
  font-size: 15px;
}
.footer_links_box ul li a{
  font-size: 13px;
}
/* dont_have_api_sec_start */
.api_key_instruct_ban{
  display: block;
  text-align: center;
  background: linear-gradient(180deg, #EBF9FF 0%, #EEF3FE 100%);
}
.api_key_instruct_ban_lft,.api_key_instruct_ban_rtt{
  max-width: 100%;
  padding: 0;
  flex-basis: 100%;
}
.api_key_instruct_ban_lft{
  margin-bottom: 30px;
}
/* dont_have_api_sec_end */
}

@media (max-width: 768px) {
  /* dont_have_api_sec_start */

  .api_key_instruct_ban_lft h3{
    font-size: 30px;
  }
  .api_key_instruct_ban_lft p{
    font-size: 16px;
  }
  .wrapper_body_cmn_out{
    max-width: 100%;
  }
  .api_key_instruct_ban_lft h3 {
    font-size: 40px;
    margin-bottom: 16px;
  }

  /* dont_have_api_sec_end */
}

@media (max-width: 350px){
  /* dont_have_api_sec_start */
  .api_key_instruct_ban_lft h3{
    font-size: 36px;
  }
  .api_key_instruct_ban{
    padding: 30px;
  }
  /* dont_have_api_sec_end */
}

```

whole_Section_wrapperdont_have_api_sec_start### Don't have an API key?

Sign up or upgrade your plan for access. [Get started for free](#) dont_have_api_sec_end

Notify webhook types

Below are the Alchemy webhook types and their supported networks.

Webhook Type Description Network [Custom Webhooks](#) Custom Webhooks allows you to track any smart contract or marketplace activity, monitor any contract creation, or any other on-chain interaction. This gives you infinite data access with precise filter controls to get the blockchain data you need. All [Mined Transaction](#) The Mined Transaction webhook notifies your app when a transaction sent through your app (using your API key) gets mined. This is useful for you to further notify the users of your app about the status of the transaction. All [Dropped Transactions](#) The Dropped Transaction webhook notifies your app when a transaction sent through your app (using your API key) gets dropped. This is useful for you to further notify the users of your app about the status of the transaction. All [Address Activity](#) Alchemy's Address Activity webhook tracks all ETH, ERC20, ERC721 and ERC1155 transfers. This provides your app with real-time state changes when an address sends/receives tokens or ETH. You can specify the addresses for which you want to track this activity. A maximum of 50,000 addresses can be added to a single webhook. All [NFT Activity](#) The NFT Activity webhook allows you to

track ERC721 and ERC1155 token contracts. This provides your app with real-time state changes when an NFT is transferred between addresses. Ethereum, Polygon, Optimism, and Arbitrum [NFT Meta Updates](#) The NFT Metadata Updates webhook allows you to track metadata updates for ERC721 and ERC1155 token contracts for Ethereum and Polygon NFTs. This notifies your app when the metadata for an NFT is updated. Ethereum Mainnet & Goerli; Polygon Mainnet & Mumbai

V2 Webhook

Field definitions

Field Description Value
webhookId Unique ID of the webhook destination. wh_octjglnywaupz6th id ID of the event.
whEvt_ogrc5v64myey69ux createdAt The timestamp when webhook was created. 2021-12-07T03:52:45.899Z type
Webhook event type. TYPE_STRING event Object-mined transaction. OBJECT

Example

```
V2 {"webhookId": "wh_octjglnywaupz6th", "id": "whEvt_ogrc5v64myey69ux", "createdAt": "2021-12-07T03:52:45.899Z", "type": TYPE_STRING, "event": OBJECT }
```

V1 Webhook Event Object

Field definitions

Field Description Value
app Alchemy app name sending the transaction webhook. Demo network Network for the webhook event. MAINNET webhookType The type of webhook. MINED_TRANSACTION timestamp Timestamp when the webhook was created. 2020-07-29T00:29:18.414Z event name Webhook event type. OBJECT For Notify full dependencies and more code examples, [go to the GitHub repo] (<https://github.com/alchemyplatform/webhook-examples>).

Example Response

```
v1 {"app": "Demo", "network": "MAINNET", "webhookType": "MINED_TRANSACTION", "timestamp": "2020-07-29T00:29:18.414Z", "event name": OBJECT }
```

How to set-up webhooks

Setting up a webhook is as simple as adding a new URL to your application.

NOTE:

If you need to add over 10 addresses to the address activity webhook, we recommend adding them through an API call.

Set-up webhooks in your dashboard

1. Navigate to the Notify tab in your [Alchemy Dashboard](#)
2. .
3. Determine which [type of webhook](#)
4. you want to activate.
5. Click the Create Webhook
6. button.
7. Select the app to add the webhook notifications.
8. Add in your unique webhook URL. This is the link to receive requests. The webhook payload might not always be compatible for 3rd party integrations.
9. Test your webhook by clicking the Test Webhook
10. button.
11. Click Create Webhook
12. and your webhook appears in the list.
13. Check your endpoint to see responses.

Set-up webhooks programmatically

Use the create-webhook endpoint: <https://docs.alchemy.com/reference/create-webhook> .

Webhook IP addresses

As an added security measure, you can ensure your webhook notification originates from Alchemy by using one of the following IP addresses:

- 54.236.136.17
- 34.237.24.169

Create webhook listeners

Webhook listeners receive requests and process event data.

The listener responds to the Alchemy server with a 200 status code once you've successfully received the webhook event. Your webhook listener can be a simple server or Slack integration to receive the webhook listener data.

After setting up the webhooks in your Alchemy dashboard (or programmatically) use the starter code in JavaScript, Python, Go, and Rust below. Here's the [GitHub repository](#) for the entire code.

JavaScript Python Go Rust import express from "express";import {getRequiredEnvVar ,setDefaultEnvVar }from
"./envHelpers" ;import {addAlchemyContextToRequest ,validateAlchemySignature ,AlchemyWebhookEvent , }from
"./webhooksUtil" ;async function main ():Promise < void

```
(const app = express ();setDefaultEnvVar ("PORT" , "8080" );setDefaultEnvVar ("HOST" , "127.0.0.1" );
setDefaultEnvVar ("SIGNING_KEY" , "whsec_test" );const port = + getRequiredEnvVar ("PORT" );const host =
getRequiredEnvVar ("HOST" );const signingKey = getRequiredEnvVar ("SIGNING_KEY" );// Middleware needed
to validate the alchemy signature app .use (express .json ({verify :addAlchemyContextToRequest , } ));app .use
(validateAlchemySignature (signingKey ));// Register handler for Alchemy Notify webhook events // TODO:
update to your own webhook path app .post ("/webhook-path" , (req ,res )=> {const webhookEvent = req .body
as AlchemyWebhookEvent ;// Do stuff with with webhook event here! console .log (Processing webhook event id: {
webhookEvent .id } );// Be sure to respond with 200 when you successfully process the event res .send ("Alchemy
Notify is the best!" );});// Listen to Alchemy Notify webhook events app .listen (port ,host , ()=> {console .log
(Example Alchemy Notify app listening at { host }:{ port } );}); }main ();import hmac import hashlib from django .core
.exceptions import PermissionDenied from webhook_server .settings import SIGNING_KEY import json from
types import SimpleNamespace def is_valid_signature_for_string_body (body :str ,signature :str ,signing_key :str
)-> bool :digest = hmac .new (bytes (signing_key ,"utf-8" ),msg = bytes (body ,"utf-8" ),digestmod = hashlib
.sha256 , ).hexdigest ()return signature == digest class AlchemyWebhookEvent :def init (self ,webhookId ,id
,createdAt ,type ,event ):self .webhook_id = webhookId self .id = id self .created_at = createdAt self .type = type
self .event = event class AlchemyRequestHandlerMiddleware :def init (self ,get_response ):self .get_response =
get_response def call (self ,request ):str_body = str (request .body ,request .encoding or "utf-8" )signature =
request .headers ["x-alchemy-signature" ]if not is_valid_signature_for_string_body (str_body ,signature
,SIGNING_KEY ):raise PermissionDenied ("Signature validation failed, unauthorized!")webhook_event = json
.loads (str_body )request .alchemy_webhook_event = AlchemyWebhookEvent ( webhook_event )response = self
.get_response (request )return response package notify import ("crypto/hmac" "crypto/sha256" "encoding/hex"
"encoding/json" "io/ioutil" "log" "net/http" "time" )type AlchemyWebhookEvent struct {WebhookId string Id string
CreatedAt time . Time Type string Event map [string ]interface {} }func jsonToAlchemyWebhookEvent (body
[]byte ) ( AlchemyWebhookEvent ,error ) {event := new (AlchemyWebhookEvent )if err := json . Unmarshal (body
,& event );err != nil {return nil ,err }return event ,nil }// Middleware helpers for handling an Alchemy Notify
webhook request type AlchemyRequestHandler func (http . ResponseWriter , http . Request ,
AlchemyWebhookEvent )type AlchemyRequestHandlerMiddleware struct {handler AlchemyRequestHandler
signingKey string }func NewAlchemyRequestHandlerMiddleware (handler AlchemyRequestHandler ,signingKey
string ) AlchemyRequestHandlerMiddleware {return & AlchemyRequestHandlerMiddleware {handler ,signingKey
} }func isValidSignatureForStringBody (body []byte ,signature string ,signingKey []byte , )bool {h := hmac . New
(sha256 . New ,signingKey )h . Write ([]byte (body ))digest := hex . EncodeToString (h . Sum (nil ))return digest
== signature }func (arh * AlchemyRequestHandlerMiddleware )ServeHTTP (w http . ResponseWriter ,r * http .
Request ) {signature := r . Header . Get ("x-alchemy-signature" )body ,err := ioutil . ReadAll (r . Body )if err != nil
{http . Error (w ,err . Error (),http . StatusInternalServerError)log . Panic (err )return }r . Body . Close
()isValidSignature := isValidSignatureForStringBody (body ,signature , []byte (arh . signingKey ))if !
isValidSignature {errMsg := "Signature validation failed, unauthorized!" http . Error (w ,errMsg ,http .
StatusForbidden)log . Panic (errMsg )return }event ,err := jsonToAlchemyWebhookEvent (body )if err != nil {http .
Error (w ,err . Error (),http . StatusBadRequest)log . Panic (err )return }arh . handler (w ,r ,event ) } use chrono ::
{DateTime ,FixedOffset };use hex ;use hmac ::{Hmac ,Mac };use sha2 ::Sha256 ;use std ::{future ::{ready ,Ready
},rc ::Rc , };use serde ::{de ,Deserialize ,Deserializer };use actix_web ::{dev ::{self ,Service ,ServiceRequest
,ServiceResponse ,Transform },error ::{ErrorBadRequest ,error ::ErrorUnauthorized ,web ::BytesMut ,Error
,HttpMessage , };use futures_util ::{future ::LocalBoxFuture ,stream ::StreamExt };#[derive(Deserialize)]
```

[serde(rename_all = "camelCase")]

```
pub struct AlchemyWebhookEvent {pub webhook_id :String ,pub id :String ,#[serde(deserialize_with =
"deserialize_date_time")] pub created_at :DateTime < FixedOffset

,#[serde(rename = "type")] pub webhook_type :String ,pub event :serde_json ::Value , }fn deserialize_date_time
< 'a ,D

(deserializer :D )-> Result < DateTime < FixedOffset

,D ::Error

where D :Deserializer < 'a

, {let date_time_string :String = Deserialize ::deserialize (deserializer )?;let date_time = DateTime ::< FixedOffset
::parse_from_rfc3339 (&date_time_string ) .map_err (|e |de ::Error ::custom (e .to_string ()))?;Ok (date_time ) }fn
is_valid_signature_for_string_body (body : &[u8 ],signature : &str ,signing_key : &str , )-> Result < bool ,Box <
dyn std ::error ::Error

{let signing_key_bytes :Vec < u8

= signing_key .bytes ().collect ();let mut mac = Hmac ::< Sha256

::new_from_slice (&signing_key_bytes )?;mac .update (&body );let hex_decode_signature = hex ::decode
(signature )?;let verification = mac .verify_slice (&hex_decode_signature ).is_ok ();Ok (verification ) }pub struct
AlchemyRequestHandlerMiddleware < S

{signing_key :String ,service :Rc < S

, }impl < S ,B

Service < ServiceRequest

for AlchemyRequestHandlerMiddleware < S

where S :Service < ServiceRequest ,Response = ServiceResponse < B
```

,Error

Error

```
+ 'static ,S ::Future : 'static ,B : 'static , {type Response = ServiceResponse < B
```

```
;type Error = Error ;type Future = LocalBoxFuture < 'static ,Result < Self ::Response ,Self ::Error
```

```
;dev ::forward_ready! (service );fn call (&self ,mut req :ServiceRequest )-> Self ::Future {let svc = self
.service .clone ();let signing_key = self .signing_key .clone ();Box ::pin (async move {let mut body =
BytesMut ::new ();let mut stream = req .take_payload ();while let Some (chunk )= stream .next ().await
{body .extend_from_slice (&chunk ?); }let signature = req .headers ().get ("x-alchemy-signature" )
.ok_or (ErrorBadRequest ("Signature validation failed, missing x-alchemy-signature header!" , ))?
.to_str () .map_err (| | {ErrorBadRequest ("Signature validation failed, x-alchemy-signature header is
not a string!" , ) }?);let is_valid_signature = is_valid_signature_for_string_body (&body ,signature ,
&signing_key )?;if ! is_valid_signature {return Err (ErrorUnauthorized ("Signature validation failed,
signature and body do not match!" , )); }let webhook :AlchemyWebhookEvent = serde_json
::from_slice (&body ).map_err (| | {ErrorBadRequest ("Bad format, body could not be mapped to
AlchemyWebhookEvent" ) }?);req .extensions_mut ().insert ::< Rc < AlchemyWebhookEvent
```

```
(Rc ::new (webhook ));let res = svc .call (req ).await ?;Ok (res ) } }pub struct
AlchemyRequestHandlerMiddlewareFactory {signing_key :String , }impl
AlchemyRequestHandlerMiddlewareFactory {pub fn new (signing_key :String )-> Self
{AlchemyRequestHandlerMiddlewareFactory {signing_key } } }impl < S ,B
```

```
Transform < S ,ServiceRequest
```

```
for AlchemyRequestHandlerMiddlewareFactory where B : 'static ,S :Service < ServiceRequest ,Response =
ServiceResponse < B
```

,Error

Error

```
+ 'static , {type Response = ServiceResponse < B

;type Error = Error ;type Transform = AlchemyRequestHandlerMiddleware < S

;type InitError = ();type Future = Ready < Result < Self ::Transform ,Self ::InitError

;fn new_transform (&self ,service :S )-> Self ::Future {ready (Ok (AlchemyRequestHandlerMiddleware
{signing_key :self .signing_key .clone (),service :Rc ::new (service ), ))) } }
```

Test webhooks with Ngrok

1. Sign-up for a [free Ngrok account](#)
2. .
3. Install Ngrok using [the Ngrok guide](#)
4. . On macOS run
5. brew install ngrok
6. .
7. Connect your Ngrok account by running
8. ngrok authtoken YOUR_AUTH_TOKEN
9. .
10. Start your local forwarding tunnel:
11. ngrok http 80
12. .

Once you have a URL to test your webhook (in this case <https://461a-199-116-73-171.ngrok.io> pictured above), follow the steps below:

1. Navigate to your [Notify dashboard](#)
2. .
3. Click Create Webhook
4. on the webhook you want to test.
5. Paste your unique URL
6. and hit the Test Webhook
7. button.
8. You'll see the webhooks here:
9. <http://localhost:4040/inspect/http>
10. .

Webhook signature & security

To make your webhooks secure, you can verify that they originated from Alchemy by generating a HMAC SHA-256 hash code using your unique webhook signing key.

Find your signing key

Navigate to the [Notify page](#) in your dashboard. Next, click on the three dots of the webhook you want to get the signature for and copy the Signing Key.

Validate the signature received

Every outbound request contains a hashed authentication signature in the header. It's computed by concatenating your signing key and request body. Then generates a hash using the HMAC SHA256 hash algorithm.

To verify the signature came from Alchemy, you generate the HMAC SHA256 hash and compare it with the signature received.

Example request header

Request header POST /yourWebhookServer/push HTTP/1.1 Content-Type: application/json; X-Alchemy-Signature: your-hashed-signature

Example signature validation

JavaScript Python Go Rust import * as crypto from "crypto";function isValidSignatureForStringBody (body :string ,// must be raw string body, not json transformed version of the body signature :string ,// your "X-Alchemy-Signature" from header signingKey :string ,// taken from dashboard for specific webhook):boolean {const hmac = crypto .createHmac ("sha256" ,signingKey);// Create a HMAC SHA256 hash using the signing key hmac .update (body ,"utf8");// Update the token hash with the request body using utf8 const digest = hmac .digest ("hex");return signature === digest ; } import hmac import hashlib def isValidSignatureForStringBody (body :str ,signature :str ,signing_key :str)-> bool :digest = hmac .new (bytes (signing_key ,"utf-8"),msg = bytes (body ,"utf-8"),digestmod = hashlib .sha256 ,).hexdigest ()return signature == digest func isValidSignatureForStringBody (body []byte ,// must be raw string body, not json transformed version of the body signature string ,// your "X-Alchemy-Signature" from header signingKey []byte ,// taken from dashboard for specific webhook)bool {h := hmac . New (sha256 . New ,signingKey)h . Write ([]byte (body))digest := hex . EncodeToString (h . Sum (nil))return digest == signature } fn is_valid_signature_for_string_body (body : &[u8],// must be raw string body, not json transformed version of the body signature : &str ,// your "X-Alchemy-Signature" from header signing_key : &str ,// taken from dashboard for specific webhook)-> Result < bool ,Box < dyn std ::error ::Error

```
{let signing_key_bytes :Vec < u8
= signing_key .bytes ().collect ();let mut mac = Hmac ::< Sha256
::new_from_slice (&signing_key_bytes )?;mac .update (&body );let hex_decode_signature = hex ::decode
(signature )?;let verification = mac .verify_slice (&hex_decode_signature ).is_ok ();Ok (verification ) }
```

Webhook retry logic

Alchemy Notify V2 has built-in retry-logic for webhooks. Requests are retried for non-200 response codes in failures to reach your server.

Requests are retried up to 6 times before failing. Below are the request retry intervals.

- 15 seconds
- 1 minute
- 10 minutes
- 1 hour
- 1 day

Updated 2 months ago

[Community subgraphs Notify Tutorials and Applications](#) Did this page help you?Yes No