Viewing Deposits and Withdrawals by Address

In this tutorial, you'll learn how to use the Optimism SDK(opens in a new tab) to view all of the Standard Bridge deposits and withdrawals triggered by a given address.

Check out the tutorial on Bridging ERC-20 Tokens With the Optimism SDK to learn how to create deposits and withdrawals.

Dependencies

- node(opens in a new tab)
- pnpm(opens in a new tab)

Create a Demo Project

You're going to use the Optimism SDK for this tutorial. Since the Optimism SDK is a Node.js(opens in a new tab) library, you'll need to create a Node.js project to use it.

Make a Project Folder

mkdir

op-sample-project cd

op-sample-project

Initialize the Project

pnpm

init

Install the Optimism SDK

pnpm

add

@eth-optimism/sdk

Install ethers.js

pnpm

add

ethers@^5

Add RPC URLs to Your Environment

You'll be using thegetDepositsByAddress andgetWithdrawalsByAddress functions from the Optimism SDK during this tutorial. These functions use event queries to retrieve the deposits and withdrawals made by a given address. Since these functions use large event queries, you'll need to use an RPC provider like Alchemy(opens in a new tab) that supports indexed event queries. Grab an L1 and L2 RPC URL for Sepolia and OP Sepolia, respectively.

export L1_RPC_URL = ...

Sepolia RPC URL

export L2_RPC_URL = ...

OP Sepolia RPC URL

The Optimism SDK may be updated in the future to use a different method of retrieving deposits and

withdrawals under the hood that does not require indexed event queries. This tutorial will be updated to reflect those changes if and when they occur.

Start the Node REPL

You're going to use the Node REPL to interact with the Optimism SDK. To start the Node REPL run the following command in your terminal:

node This will bring up a Node REPL prompt that allows you to run javascript code.

Import Dependencies

You need to import some dependencies into your Node REPL session.

Import the Optimism SDK

const

optimism

=

require ("@eth-optimism/sdk")

Import ethers.js

const

ethers

=

require ("ethers")

Set Session Variables

You'll need a few variables throughout this tutorial. Let's set those up now.

Import RPC URLs

const

l1RpcUrl

=

process . env . L1_RPC_URL const

l2RpcUrl

=

process . env . L2_RPC_URL

Set the address to query

Here you'll be querying over an address that has already made some deposits and withdrawals. You can replace this address with your own address if you'd like.

const

address

=

'0x5A07785F07D8ba8a9e5323181fBDab51FE9a36c3'

Create the RPC providers

const

l1Provider

=

new

ethers . providers .StaticJsonRpcProvider (l1RpcUrl) const

l2Provider

=

new

ethers . providers .StaticJsonRpcProvider (l2RpcUrl)

Create the CrossDomainMessenger

The Optimism SDK exports aCrossChainMessenger class that makes it easy to interact with the Standard Bridge contracts.

Create an instance of theCrossChainMessenger class:

const

messenger

=

new

optimism .CrossChainMessenger ({ l1ChainId :

11155111 ,

// 11155111 for Sepolia, 1 for Ethereum l2ChainId :

11155420 ,

// 11155420 for OP Sepolia, 10 for OP Mainnet l1SignerOrProvider : l1Provider , l2SignerOrProvider : l2Provider , })

Query for Deposits

You'll first query for all of the deposits made by the target address. TheCrossChainMessenger has a convenient function calledgetDepositsByAddress that makes this easy.

Grab all deposits

const

deposits

=

await

messenger .getDepositsByAddress (address)

Display the deposits

for (const

deposit

of deposits) { console .log ('----------') console .log ('From: ' ,

deposit .from) console .log ('To: ' ,

deposit .to) console .log ('L1 Token:' ,

deposit .l1Token) console .log ('L2 Token:' ,

deposit .l2Token) console .log ('Amount: ' ,

deposit . amount .toString ()) }

Query for Withdrawals

You'll next query for all of the withdrawals made by the target address. TheCrossChainMessenger has a convenient function calledgetWithdrawalsByAddress that makes this easy.

Grab all withdrawals

const

withdrawals

=

await

messenger .getWithdrawalsByAddress (address)

Display the withdrawals

for (const

withdrawal

of withdrawals) { console .log ('---') console .log ('From: ' ,

withdrawal .from) console .log ('To: ' ,

withdrawal .to) console .log ('L1 Token:' ,

withdrawal .l1Token) console .log ('L2 Token:' ,

withdrawal .l2Token) console .log ('Amount: ' ,

withdrawal . amount .toString ()) }

Bridging Your Standard ERC-20 Token to OP MainnetTracing Deposits and Withdrawals