Using OP Mainnet System Contracts

System contracts on Ethereum (L1) and OP Mainnet (L2) are an important part of the OP Mainnet ecosystem. You may want to interact with these system contracts for any number of reasons, including:

- Sending messages between L1 and L2
- Sending tokens between L1 and L2
- Querying information about the currentL1 data fee
- And lots more!

In this tutorial, you'll learn how to work with OP Mainnet contracts directly from other contracts and how to work with them from the client side.

Before You Begin

You'll need to find the address of the particular contract that you want to interact with before you can actually interact with it.

- Check out theNetworks and Connection Details page
- for links to public RPC endpoints and contract addresses.
- Find the addresses for all networks on theContract Addresses
- page.
- Use the JSON file including contract addresses for all Superchain networks in theSuperchain Registry(opens in a new tab)
- .

Using System Contracts in Solidity

All you need to interact with the OP Mainnet system contracts from another contract is an address and an interface. You can followthe instructions above to find the address of the contract you want to interact with. Now you simply need to import the appropriate contracts.

Installing via NPM

You can use the@eth-optimism/contracts-bedrock (opens in a new tab) npm package to import system contracts and their interfaces. Install the package as follows:

npm

install

@eth-optimism/contracts-bedrock

Importing Contracts

Simply import the desired contract or interface from the@eth-optimism/contracts-bedrock package:

import { SomeOptimismContract } from

"@eth-optimism/contracts-bedrock/path/to/SomeOptimismContract.sol" ; Please note thatpath/to/SomeOptimismContract is the path to the contractwithin this folder(opens in a new tab) . For example, if you wanted to import theL1CrossDomainMessenger (opens in a new tab) contract, you would use the following import:

import { L1CrossDomainMessenger } from

"@eth-optimism/contracts/L1/messaging/L1CrossDomainMessenger.sol" ;

Getting L2 Contract Addresses

System contracts on L2 are "pre-deployed" to special addresses that are the same on most OP Stack chains. You can find these addresses on theContract Addresses page. These addresses are also provided as constants in the Predeploys (opens in a new tab) contract for use in Solidity.

Using System Contracts in JavaScript

You can also interact with the OP Mainnet system contracts from the client side.

Installing via NPM

You can use the@eth-optimism/contracts-ts (opens in a new tab) npm package to import system contracts and their interfaces. Install the package as follows:

npm

install

@eth-optimism/contracts-ts

Getting Contract Artifacts and Interfaces

You can use the@eth-optimism/contracts-ts package to easily access the address or ABI of any OP Mainnet contract.

Here's an example of how you can grab the ABI and address of theL2OutputOracleProxy contract on OP Mainnet (chain ID 10):

import { l2OutputOracleProxyABI , l2OutputOracleAddresses , } from

'@eth-optimism/contracts-ts'

// Note that the address is keyed by chain ID! console .log (l2OutputOracleAddresses[10] , abi)

Interacting with the Contract

You can then feed this address and ABI into your favorite web3 library to interact with the contract.

@eth-optimism/contracts-ts also exportsReact hooks(opens in a new tab) andwagmi actions(opens in a new tab) for interacting with OP Mainnet contracts. Check out the fullREADME(opens in a new tab) for more information.

Solidity Compatibility Cost Optimization