How to run a local Nitro dev node

Overview

This page provides step-by-step instructions for setting up and running a local Nitro node in--dev mode. This mode is ideal for developers who want to quickly test contracts using a single node, as it offers a simpler and faster setup compared to more complex environments.

While some teams usenitro-testnode for testing cross-layer messaging, which involves launching both Geth as L1 and Nitro as L2, this setup can be more complex and time-consuming. If your primary goal is to test contracts on a local node without needing cross-layer interactions, Nitro's--dev mode offers a lightweight and efficient alternative.

However, if you need more advanced functionality—such as cross-layer messaging, working with both L1 and L2 chains, or testing interactions between different layers—nitro-testnode is the preferred option. The testnode setup allows you to simulate a full L1-L2 environment, which is critical for those scenarios. See here for instructions.

Note that Nitro--dev mode is ideal for Stylus contract testing, as it is much lighter and faster to set up than the full nitro-testnode environment.

Prerequisites

Before beginning, ensure the following is installed and running on your machine:

- Docker: Required to run the Nitro dev node in a container. Install Docker by followingthe official installation guide
- for your operating system.
- cast: A command-line tool from Foundry for interacting with Ethereum smart contracts. You can install it via Foundry by followingthe installation instructions
- .
- jq: A lightweight JSON parsing tool used to extract contract addresses from the script output. Install jq by followingthe official installation guide
- for your operating system.

Clone thenitro-devnode

repository

Use the following command to clone the repository:

git clone https://github.com/OffchainLabs/nitro-devnode.git cd nitro-devnode

Run the dev node script:

Run the script to start the Nitro dev node, deploy the StylusCache Manager contract, and register it as a WASM cache manager using the default development account:

./run-dev-node.sh The script will:

- Start the Nitro dev node in the background using Docker.
- Deploy the StylusCache Manager
- contract on the local Nitro network.
- Register theCache Manager
- contract as a WASM cache manager.

Development account (used by default)

In--dev mode, the script uses a pre-funded development account by default. This account is pre-funded with ETH in all networks and is used to deploy contracts, interact with the chain, and assume chain ownership.

- Address: 0x3f1Eae7D46d88F08fc2F8ed27FCb2AB183EB2d0E
- Private key: 0xb6b15c8cb491557369f3c7d2c287b053eb229daa9c22138887752191c9520659

You don't need to set up a private key manually unless you prefer using your own key.

Chain ownership in--dev

mode

In Nitro--dev mode, the default chain owner is set to0x0000000000000000000000000000000000000000 . However, you can use theArbDebug precompile to set the chain owner. This precompile includes thebecomeChainOwner() function, which can be called to assume ownership of the chain.

Chain ownership is important because it allows the owner to perform certain critical functions within the Arbitrum environment, such as:

- Adding or removing other chain owners
- Setting the L1 and L2 base fees directly
- Adjusting the gas pricing inertia and backlog tolerance
- Modifying the computational speed limit and transaction gas limits
- Managing network and infrastructure fee accounts

The script automatically sets the chain owner to the pre-funded dev account before registering theCache Manager contract. Here’s how thebecomeChainOwner() function is called within the script:

cast send 0x00000000000000000000000000000000000000FF "becomeChainOwner()" --private-key 0xb6b15c8cb491557369f3c7d2c287b053eb229daa9c22138887752191c9520659 --rpc-url http://127.0.0.1:8547 This step ensures that the dev account has ownership of the chain, which is necessary to register theCache Manager as a WASM cache manager.

At the end of the process, you'll have the Nitrodev mode running with the necessary components deployed. This environment is ready for testing and interacting with your contracts, including those written in Stylus, using the deployedCache Manager to support enhanced functionality for Stylus-based smart contracts. Edit this page Last updatedonJan 27, 2025 Previous Run a local full chain simulationNext L1 Ethereum RPC providers