

# How to run a local dev node with Stylus support

ALPHA RELEASE, PUBLIC PREVIEW DOCS Stylus is currently tagged as analpha release. The code has not been audited, and shouldnot be used in production scenarios . This documentation is currently in [public preview](#) .

To provide feedback, click theRequest an update button at the top of this document, [join the Arbitrum Discord](#) , or reach out to our team directly by completing [this form](#) . This how-to guides you through the process of setting up a local Arbitrum Nitro dev node, with Stylus support, to help you deploy and test smart contracts written in Solidity or any of the [languages supported by Stylus](#) in a fully controlled environment. By following these steps, you'll deploy and run a full development environment on your local machine that includes a Nitro dev node (L2), a dev-mode geth parent chain (L1), and multiple instances with different roles.

## Step 1. Install prerequisites

You'll need [docker](#) and [docker compose](#) to run your node. Follow the instructions in their site to install them.

## Step 2. Clone the [nitro-testnode](#)

repo

You'll need thestylus branch.

```
git clone -b stylus --recurse-submodules https://github.com/OffchainLabs/nitro-testnode.git &&
```

```
cd nitro-testnode
```

## Step 3. Run your node

```
./test-node.bash --init
```

## Step 4. Successive runs

To relaunch the node after the first installation, run the following command.

```
./test-node.bash Clear local data Note that running with the --init flag will clear all chain data and redeploy!
```

## Rollup contract addresses and chain configuration

You can obtain the rollup chain configuration by running the following command. The chain configuration also includes the addresses of the core contracts.

```
docker
```

```
exec nitro-testnode-sequencer-1 cat /config/l2_chain_info.json You can find other available configuration files by running:
```

```
docker
```

```
exec nitro-testnode-sequencer-1 ls /config
```

## Token bridge

An L1-L2 token bridge can be deployed by using the parameter--tokenbridge . The list of contracts can be found by running:

```
docker compose run --entrypoint sh tokenbridge -c "cat l1l2_network.json"
```

## Running an L3 chain

An L3 chain can be deployed on top of the L2 chain, by using the parameter--l3node . Its chain configuration can be found by running:

```
docker
```

```
exec nitro-testnode-sequencer-1 cat /config/l3_chain_info.json
```

## Additional arguments

You can find a list of additional arguments to use with `test-node.bash` by using `--help`.

```
./test-node.bash --help
```

## Helper scripts

The repository includes a set of helper scripts for basic actions like funding accounts or bridging funds. You can see a list of the available scripts by running:

`./test-node.bash script --help` If you want to see information of a particular script, you can add the name of the script to the help command.

`./test-node.bash script send-l1 --help` Here's an example of how to run the script that funds an address on L2. Replace `0x11223344556677889900` with the address you want to fund.

```
./test-node.bash script send-l2 --to address_0x11223344556677889900 --ethamount 5
```

## Blockscout

Nitro comes with a local [Blockscout](#) block explorer. To access it, add the param `--blockscout` when running your node.

```
./test-node.bash --blockscout
```

 The block explorer will be available at `http://localhost:4000`

## Default endpoints and addresses

Node RPC endpoints are available at:

Node Chain id RPC endpoint L1 geth devnet 1337 `http://localhost:8545` L2 nitro devnet 412346 `http://localhost:8547` and `ws://localhost:8548` L3 nitro (if enabled) 333333 `http://localhost:3347` Some important addresses:

Role	Public address	Private key
Sequencer	<code>0xe2148eE53c0755215Df69b2616E552154EdC584f0xcb5790da63720727af975f42c79f69918580209889225fa7128c92402a6d3a65</code>	<code>Validator</code>
	<code>0x6A568afe0f82d34759347bb36F14A6bB171d2CBe</code>	
	<code>0x182fecf15bdf909556a0f617a63e05ab22f1493d25a9f1e27c228266c772a890</code>	L2 rollup owner
	<code>0x5E1497dD1f08C87b2d8FE23e9AAB6c1De833D927</code>	
	<code>0xdc04c5399f82306ec4b4d654a342f40e2e0620fe39950d967e1e574b32d4dd36</code>	L3 rollup owner (if enabled)
	<code>0x863c904166E801527125D8672442D736194A3362</code>	
	<code>0xecdf21cb41c65afb51f91df408b7656e2c8739a5877f2814add0afd780cc210e</code>	L3 sequencer (if enabled)
	<code>0x3E6134aAD4C4d422FF2A4391Dc315c4DDf98D1a5</code>	
	<code>0x90f899754eb42949567d3576224bf533a20857bf0a60318507b75fcb3edc6f5f</code>	Dev account (prefunded with ETH in all networks)
	<code>0x3f1Eae7D46d88F08fc2F8ed27FCb2AB183EB2d0E</code>	
	<code>0xb6b15c8cb491557369f3c7d2c287b053eb229daa9c22138887752191c9520659</code>	You can fund other addresses by using the scripts <code>send-l1</code> and <code>send-l2</code> as explained <a href="#">here</a> .

Private keys publicly known Do not use any of these addresses in a production environment [Edit this page](#) Last updated on Mar 7, 2024 [Previous Quickstart: Write a smart contract in Rust using Stylus](#) [Next Stylus testnet information](#)