FAQ

Canton Network Quickstart FAQ | 2025

Version: 1.0.0-2025-02-27

Contents

- CN-QS Frequently Asked Questions
  - System Requirements & Setup
  - Common Issues & Troubleshooting
  - Development & Testing
  - Infrastructure & Environment
  - Best Practices & Common Pitfalls
  - Database & Query Access
- CN-QS Make Target Reference
  - UI Opening Commands
- LocalNet URLs

CN-QS Frequently Asked Questions

System Requirements & Setup

What are the minimum system requirements to run CN-QS LocalNet?

The CN-QS requires Docker Desktop with at least 25 GB of memory allocated to run `LocalNet` properly. If your machine has less memory, consider declining Observability when prompted during setup.

Which browsers are supported for running CN-QS?

Chromium browsers such as Chrome, Edge, and Brave, as well as Firefox are recommended. Safari has known issues with local URLs and should be avoided.

How should I test `Participant` and `User` interactions on `LocalNet` and `DevNet`?

For testing multiple users, use separate browsers or one browser in standard mode and another in incognito to avoid session/cookie interference.

How do I handle authentication for JFrog Artifactory?

You need to create a `~/.netrc` file with the following format:

```
machine digitalasset.jfrog.io
login <your-email>
password <your-api-key>
```

Set permissions with `chmod 600 ~/.netrc`

For more information see: Installation Guide

Why is Nix-shell unable to download my SSL certificate?

The Nix prerequisite may introduce hurdles to installation if your enterprise runs behind a corporate proxy. If `nix-shell` is not found, then verify that

`/nix/var/nix/profiles/default/etc/ssl/certs/ca-bundle.crt`

contains your corporate CA.

CN, PQS, Daml Shell and other CN-QS related services run on a user-supplied JVM. CN-QS assumes that you have access to JVM v17+ with access to the internet. If your organization operates behind a web proxy then JVM may not have automatic knowledge of the corporate certificate. In these instances, JVM must be instructed to trust the certificate.

If Nix-related errors occur, verify that the correct certificates exist by looking at the log file.

```
$ sudo HOME=/var/root
NIX_SSL_CERT_FILE=/nix/var/nix/profiles/default/etc/ssl/certs/ca-bundle.crt
/nix/store/dfqs9x0l0r4dn7zjp1hymmv9wvpp9x2k-nix-2.26.2/bin/nix-channel --update
nixpkgs
```

If the log returns an error message such as:

```
error: unable to download
'https://nixos.org/channels/nixpkgs-unstable': SSL peer certificate
or SSH remote key was not OK (60)
```

Then the required corporate CA does not exist. Request your corporate CA from your organization's tech administrator and merge the certificate into the Nix `certs ca-bundle.crt`.

If you need additional support, the Nix reference manual offers guidance regarding the order at which cert files are detected and used on the host, as well as environment variables to override default file locations.

Graham Christensen's Determinate Systems blog offers a solution for Nix corporate TLS certificates problems on MacOS. The NixOS team forked this solution as an experimental installer that is stable on most operating systems.

Common Issues & Troubleshooting

How can I check if my CN-QS deployment is running correctly?

Use `make status` to see all running containers and their health status.

What should I do if containers show as "unhealthy" after startup?

The most common cause is insufficient memory allocation to Docker. Try:

1. Increase Docker memory allocation to at least 25 GB
2. Run `make stop` followed by `make clean-all`
3. Run `make setup` and turn off `observability`
4. Restart with `make start`

How can I monitor system metrics?

You can use Grafana at http://localhost:3030/ to monitor system metrics if `observability` is enabled.

For more information see: Observability and Troubleshooting Overview

What should I do if I need to completely reset my environment?

Execute the following commands in order:

1. `make stop`
2. `make clean-all`
3. `make setup` (to reconfigure environment options)
4. `make start`

Development & Testing

How do I access the Daml Shell for debugging?

Run `make shell` from the quickstart directory. This provides access to useful commands like:

- `active` - shows summary of contracts
- `active quickstart:Main:Asset` - shows Asset contract details
- `contract [contract-id]` - shows full contract details

How can I monitor application logs and traces?

The CN-QS provides several observability options:

1. Direct container logs: `docker logs <container-name>`
2. Grafana dashboards: http://localhost:3030/
3. Consolidated logs view in Grafana

Infrastructure & Environment

What's the difference between LocalNet and DevNet deployment?

`LocalNet` runs everything locally including a Super Validator and Canton Coin wallet, making it more resource intensive but self-contained.

`DevNet` connects to actual decentralized Global Synchronizer infrastructure operated by Super Validators. `DevNet` requires less local resources but needs whitelisted VPN access and connectivity.

For more information see: Project Structure Guide

Do I need VPN access to use CN-QS?

VPN access is only required for `DevNet` connections. You need either:

- Access to the DAML-VPN
- Access to a SV Node that is whitelisted on the CN. Contact your sponsoring Super Validator agent for connection information.

For more information see: Explore the Demo

Best Practices & Common Pitfalls

How should I handle multiple user testing in the local environment?

Best practices include:

1. Use separate browsers for different users
2. Follow proper logout procedures between user switches
3. Be aware that even incognito mode in the same browser may have session interference
4. Consider using the make commands for testing specific operations (e.g., `make create-app-install-request`)

Database & Query Access

What's the recommended way to query ledger data?

The Participant Query Store (PQS) is recommended for querying ledger data.

CN-QS Make Target Reference

Target	Description
`build`	Build frontend, backend, Daml model and docker images
`build-frontend`	Build the frontend application
`build-backend`	Build the backend service
`build-daml`	Build the Daml model
`create-app-install-request`	Submit an App Install Request from the App User participant node
`restart-backend`	Build and restart the backend service
`restart-frontend`	Build and restart the frontend application

`start`	Start the application and observability services if enabled
`stop`	Stop the application and observability services
`stop-application`	Stop only the application, leaving observability services running
`restart`	Restart the entire application
`status`	Show status of Docker containers
`logs`	Show logs of Docker containers
`tail`	Tail logs of Docker containers
`setup`	Configure local development environment (enable DevNet/LocalNet, Observability)
`console`	Start the Canton console
`clean-console`	Stop and remove the Canton console container
`shell`	Start Daml Shell
`clean-shell`	Stop and remove the Daml Shell container
`clean`	Clean the build artifacts
`clean-docker`	Stop and remove application Docker containers and volumes
`clean-application`	Like clean-docker, but leave observability services running
`clean-all`	Stop and remove all build artifacts, Docker containers and volumes
`install-daml-sdk`	Install the Daml SDK
`generate-NOTICES`	Generate the NOTICES.txt file
`update-env-sdk-runtime-version`	Helper to update DAML_RUNTIME_VERSION in .env based on daml/daml.yaml sdk-version

UI Opening Commands

Target	Description
`open-app-ui`	Open the Application UI in the active browser
`open-observe`	Open the Grafana UI in the active browser
`open-sv-gateway`	Open the Super Validator gateway UI in the active browser
`open-sv-wallet`	Open the Super Validator wallet UI in the active browser
`open-sv-interface`	Open the Super Validator interface UI in the active browser
`open-sv-scan`	Open the Super Validator Scan UI in the active browser
`open-app-user-wallet`	Open the App User wallet UI in the active browser

LocalNet URLs

URL	Description
`http://localhost:3000`	Main application UI
`http://localhost:3030`	Grafana observability dashboard (if enabled)
`http://localhost:4000`	Super Validator gateway - lists available web UI options
`http://wallet.localhost:2000`	Canton Coin wallet interface
`http://sv.localhost:4000`	Super Validator Operations
`http://scan.localhost:4000`	Canton Coin Scan web UI - shows balances and validator rewards
`http://localhost:7575`	Ledger API service
`http://localhost:5003`	Validator API service

In `DevNet` mode, Super Validator and wallet services are hosted externally rather than locally. The exact URLs for those services are provided by your sponsoring Super Validator.