

5.8 Environment Verification

[Go To Top](#)

[Return to Development Environment](#)

The team should provide a repeatable way to verify the development environment before coding begins. The `checkPrerequisites.sh` script should check required tools, versions, commands, paths, environment variables, and basic runtime assumptions.

Environment verification should fail clearly when a required tool is missing, misconfigured, or unsupported. The script should explain what failed, why it matters, and what the developer should check next. It should return a non-zero exit code when the environment does not satisfy the [Phase 0 Baseline](#).

The Verification process should distinguish required tools from optional tools. Missing required tools should block Baseline execution. Missing optional tools should produce warnings rather than failures unless the developer requested a workflow that depends on them.

Developers should run environment Verification before generating [Types](#), building containers, running [Nodes](#), or collecting acceptance [Evidence](#). Reviewers may also use environment Verification to confirm that the [Repository](#) instructions match actual executable behaviour.

The practical goal is simple: a developer should be able to clone the Repository, run the prerequisite check, fix reported issues, initialise the environment, generate required [Artifacts](#), build the containers, run the Node network, observe status, stop the network, and inspect the resulting Evidence without relying on undocumented local knowledge.

© 2026 Dido Solutions, Inc. and Jackrabbit Consulting, Inc.

From:
<https://wiki.didosolutions.com/> - **Dido Solutions, Inc Wiki**

Permanent link:
<https://wiki.didosolutions.com/fxdemo/05-part/05-development-environment/05-8-environment-verification/start>

Last update: **2026/08/08 19:30**

