13.3 Python Utility Script Conventions

Go To Top

Return to Build and Run Scripts

Python utility scripts may support setup, validation, type-generation orchestration, log inspection, test harnesses, catalogue extraction, documentation generation, and Evidence preparation. Python utilities should support repeatability and review; they should not become undocumented logic hidden outside the architecture and handbook.

Each Python utility script should:

1. Include a module docstring that identifies the script purpose, expected inputs, outputs, and failure behavior.
2. Provide a clear command-line interface where developers run it directly.
3. Validate inputs before reading, writing, generating, or deleting files.
4. Use Repository-relative paths where practical.
5. Avoid hard-coded machine-specific paths.
6. Return meaningful process exit codes.
7. Write useful diagnostic output.
8. Keep generated Artifacts separate from handwritten source files.
9. Use functions with docstrings for non-trivial behavior.
10. Avoid silently swallowing exceptions.
11. Document any dependencies on Python packages or external tools.

Python scripts should use the project-local virtual environment defined in the Development Environment section. The Repository should declare dependencies through the approved dependency file or package Configuration.

© 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/13-build-and-run-scripts/13-3-python-utility-script-conventions/start

Last update: **2026/08/10 14:59**