changelog.md for @apiclient.xyz/docker

2026-03-28 - 5.1.2 - fix(deps)

upgrade core tooling dependencies and adapt Docker client internals for compatibility

- replace removed smartfile filesystem APIs with node:fs and SmartFileFactory usage
- update imagestore archive handling for smartarchive v5 and smartbucket v4 overwrite behavior
- improve Docker resource creation and stream handling with stricter null checks, cleanup, and timeout safeguards
- adjust tests and runtime behavior for Deno and newer dependency constraints

2026-03-16 - 5.1.1 - fix(paths)

use the system temp directory for nogit storage and add release metadata

- Changes nogitDir to resolve under the OS temporary directory instead of a local .nogit folder
- Adds @git.zone/cli release and module metadata to npmextra.json for npm publishing configuration

2025-11-25 - 5.1.0 - feat(host)

Add DockerHost version & image-prune APIs, extend network creation options, return exec inspect info, and improve image import/store and streaming

- Add DockerHost.getVersion() to retrieve Docker daemon version and build info
- Add DockerHost.pruneImages() with dangling and filters support (calls /images/prune)
- Extend INetworkCreationDescriptor and DockerNetwork._create() to accept Driver, IPAM, EnableIPv6, Attachable and Labels

- Enhance DockerContainer.exec() to return an inspect() helper and introduce IExecInspectInfo to expose exec state and exit code
- Improve DockerImage._createFromTarStream() parsing of docker-load output and error messages when loaded image cannot be determined
- Implement DockerImageStore.storeImage() to persist, repackage and upload images (local processing and s3 support)
- Various streaming/request improvements for compatibility with Node streams and better handling of streaming endpoints
- Update tests to cover new features (network creation, exec inspect, etc.)

2025-11-24 - 5.0.2 - fix(DockerContainer)

Fix getContainerById to return undefined for non-existent containers

- Prevented creation of an invalid DockerContainer from Docker API error responses when a container does not exist.
- Changed DockerContainer._fromId to use the list+find pattern and return Promise<DockerContainer | undefined>.
- Updated DockerHost.getContainerById to return Promise<DockerContainer | undefined> for type safety and consistent behavior.
- Added tests to verify undefined is returned for non-existent container IDs and that valid IDs return DockerContainer instances.
- Bumped package version to 5.0.1 and updated changelog and readme hints to document the fix.

2025-11-24 - 5.0.0 - BREAKING CHANGE(DockerHost)

Rename array-returning get* methods to list* on DockerHost and related resource classes; update docs, tests and changelog

- Renamed public DockerHost methods: getContainers → listContainers, getNetworks → listNetworks, getServices → listServices, getImages → listImages, getSecrets → listSecrets.
- Renamed DockerNetwork.getContainersOnNetwork → DockerNetwork.listContainersOnNetwork and updated usages (e.g. getContainersOnNetworkForService).

- Updated internal/static method docs/comments to recommend dockerHost.list*() usage and adjusted implementations accordingly.
- Updated README, readme.hints.md, tests (test.nonci.node+deno.ts) and changelog to reflect the new list* method names.
- Bumped package version to 4.0.0.
- Migration note: replace calls to get*() with list*() for methods that return multiple items (arrays). Single-item getters such as getContainerById or getNetworkByName remain unchanged.

2025-11-24 - 5.0.1 - fix(DockerContainer)

Fix getContainerById() to return undefined instead of invalid container object when container doesn't exist

Bug Fixed:

- `getContainerById()` was creating a DockerContainer object from error responses when a container didn't exist
- The error object `{ message: "No such container: ..." }` was being passed to the constructor
- Calling `.logs()` on this invalid container returned "[object Object]" instead of logs

Solution:

- Changed `DockerContainer._fromId()` to use the list+filter pattern (consistent with all other resource getters)
- Now returns `undefined` when container is not found (matches DockerImage, DockerNetwork, DockerService, DockerSecret behavior)
- Updated return type to `Promise<DockerContainer | undefined>` for type safety
- Added tests to verify undefined is returned for non-existent containers

Migration: No breaking changes - users should already be checking for undefined/null based on TypeScript types and documentation.

2025-11-24 - 4.0.0 - BREAKING CHANGE: Rename list methods for

consistency

Breaking Changes:

- Renamed all "get*" methods that return arrays to "list*" methods for better clarity:
  - `getContainers()` → `listContainers()`
  - `getNetworks()` → `listNetworks()`
  - `getServices()` → `listServices()`
  - `getImages()` → `listImages()`
  - `getSecrets()` → `listSecrets()`
  - `getContainersOnNetwork()` → `listContainersOnNetwork()` (on DockerNetwork class)

Migration Guide: Update all method calls from `get*()` to `list*()` where the method returns an array of resources. Single-item getters like `getContainerById()`, `getNetworkByName()`, etc. remain unchanged.

Rationale: The `list*` naming convention more clearly indicates that these methods return multiple items (arrays), while `get*` methods are reserved for retrieving single items by ID or name. This follows standard API design patterns and improves code readability.

2025-11-24 - 3.0.2 - fix(readme)

Update README to document 3.0.0+ changes: architecture refactor, streaming improvements, health check and circular dependency fixes

- Documented major refactor to a Clean OOP / Facade pattern with DockerHost as the single entry point
- Added/clarified real-time container streaming APIs: streamLogs(), attach(), exec()
- Clarified support for flexible descriptors (accept both string references and class instances)
- Documented complete container lifecycle API (start, stop, remove, logs, inspect, stats)
- Documented new ping() health check method to verify Docker daemon availability
- Noted fix for circular dependency issues in Node.js by using type-only imports
- Mentioned improved TypeScript definitions and expanded examples, migration guides, and real-world use cases

2025-11-24 - 3.0.1 - fix(classes.base)

Use type-only import for DockerHost in classes.base to avoid runtime side-effects

- Changed the import in ts/classes.base.ts to a type-only import: import type { DockerHost } from './classes.host.js';
- Prevents a runtime import of classes.host when only the type is needed, reducing risk of circular dependencies and unintended side-effects during module initialization.
- No behavior changes to the public API — TypeScript-only change; intended to improve bundling and runtime stability.

2025-11-24 - 3.0.0 - BREAKING CHANGE(DockerHost)

Refactor public API to DockerHost facade; introduce DockerResource base; make resource static methods internal; support flexible descriptors and stream compatibility

- Refactored architecture: DockerHost is now the single public entry point (Facade) for all operations; direct static calls like DockerImage.createFromRegistry(...) are now internal and replaced by DockerHost.createImageFromRegistry(...) and similar factory methods.
- Introduced DockerResource abstract base class used by all resource classes (DockerContainer, DockerImage, DockerNetwork, DockerSecret, DockerService) with a required refresh() method and standardized dockerHost property.
- Static methods on resource classes were renamed / scoped as internal (prefixed with _): _list, _fromName/_fromId, _create, _createFromRegistry, _createFromTarStream, _build, etc. Consumers should call DockerHost methods instead.
- Creation descriptor interfaces (container, service, etc.) now accept either string identifiers or resource instances (e.g. image: string | DockerImage, networks: (string | DockerNetwork)[], secrets: (string | DockerSecret)[]). DockerHost resolves instances internally.
- DockerImageStore imageStore has been made private on DockerHost; new public methods DockerHost.storeImage(name, stream) and DockerHost.retrieveImage(name) provide access to the image store.
- Streaming compatibility: updated requestStreaming to convert web ReadableStreams (smartrequest v5+) to Node.js streams via smartstream.nodewebhelpers, preserving backward compatibility for existing streaming APIs (container logs, attach, exec, image import/export, events).
- Container enhancements: added full lifecycle and streaming/interactive APIs on DockerContainer: refresh(), inspect(), start(), stop(), remove(), logs(), stats(), streamLogs(), attach(), exec().
- Service creation updated: resolves image/network/secret descriptors (strings or instances); adds labels.version from image; improved resource handling and port/secret/network resolution.

- Network and Secret classes updated to extend DockerResource and to expose refresh(), remove() and lookup methods via DockerHost (createNetwork/listNetworks/getNetworkByName, createSecret/listSecrets/getSecretByName/getSecretById).
- Tests and docs updated: migration guide and examples added (readme.hints.md, README); test timeout reduced from 600s to 300s in package.json.
- BREAKING: Public API changes require consumers to migrate away from direct resource static calls and direct imageStore access to the new DockerHost-based factory methods and storeImage/retrieveImage APIs.

2025-11-18 - 2.1.0 - feat(DockerHost)

Add DockerHost.ping() to check Docker daemon availability and document health-check usage

- Add DockerHost.ping() method that issues a GET to /_ping and throws an error if the response status is not 200
- Update README: show ping() in Quick Start, add health check examples (isDockerHealthy, waitForDocker) and mention Health Checks in Key Concepts

2025-11-18 - 2.0.0 - BREAKING CHANGE(DockerHost)

Rename DockerHost constructor option 'dockerSockPath' to 'socketPath' and update internal socket path handling

- Breaking: constructor option renamed from 'dockerSockPath' to 'socketPath' — callers must update their code.
- Constructor now reads the provided 'socketPath' option first, then falls back to DOCKER_HOST, CI, and finally the default unix socket.
- README examples and documentation updated to use 'socketPath'.

2025-11-17 - 1.3.6 - fix(streaming)

Convert smartrequest v5 web ReadableStreams to Node.js streams and update deps for streaming compatibility

- Upgrade @push.rocks/smartrequest to ^5.0.1 and bump @git.zone dev tooling (@git.zone/tsbuild, tsrun, tstest).
- requestStreaming now uses response.stream() (web ReadableStream) and converts it to a Node.js Readable via plugins.smartstream.nodewebhelpers.convertWebReadableToNodeReadable for backward compatibility.
- Updated consumers of streaming responses (DockerHost.getEventObservable, DockerImage.createFromTarStream, DockerImage.exportToTarStream) to work with the converted Node.js stream and preserve event/backpressure semantics (.on, .pause, .resume).
- Added readme.hints.md documenting the smartrequest v5 migration, conversion approach, modified files, and test/build status (type errors resolved and Node.js tests passing).
- Removed project metadata file (.serena/project.yml) from the repository.

2025-08-19 - 1.3.5 - fix(core)

Stabilize CI/workflows and runtime: update CI images/metadata, improve streaming requests and image handling, and fix tests & package metadata

- Update CI workflows and images: switch workflow IMAGE to code.foss.global/host.today/ht-docker-node:npmci, fix NPMCI_COMPUTED_REPOURL placeholders, and replace @shipzone/npmci with @ship.zone/npmci in workflows
- Update npmextra.json gitzone metadata (githost -> code.foss.global, gitscope -> apiclient.xyz, npmPackagename -> @apiclient.xyz/docker) and npmdocker.baseImage -> host.today/ht-docker-node:npmci
- Adjust package.json repository/bugs/homepage to code.foss.global, add pnpm overrides entry and normalize package metadata
- Improve DockerHost streaming and request handling: reduce requestStreaming timeout to 30s, enable autoDrain for streaming requests, improve response parsing for streaming vs JSON endpoints to avoid hangs
- Enhance DockerImage and DockerImageStore stream handling and tar processing: more robust import/export parsing, safer stream-to-file writes, repackaging steps, and error handling
- Unskip and update tests: re-enable DockerImageStore integration test, change stored image name to 'hello2', add formatting fixes and ensure cleanup stops the test DockerHost
- Miscellaneous code and docs cleanup: numerous formatting fixes and trailing-comma normalization across README and TS sources, update commitinfo and logger newline fixes, and add local tool ignores (.claude/.serena) to .gitignore

2025-08-19 - 1.3.4 - fix(test)

Increase test timeout, enable DockerImageStore test, update test image name, bump smartrequest patch, and add local claude settings

- Increase tstest timeout from 120s to 600s in package.json to accommodate longer-running integration tests.
- Unskip the DockerImageStore integration test and change stored image name from 'hello' to 'hello2' in test/test.nonci.node.ts.
- Bump dependency @push.rocks/smartrequest from ^4.3.0 to ^4.3.1.
- Add .claude/settings.local.json to allow local agent permissions for running tests and related tooling.

2025-08-19 - 1.3.3 - fix(classes.host)

Adjust requestStreaming timeout and autoDrain; stabilize tests

- Reduced requestStreaming timeout from 10 minutes to 30 seconds to avoid long-running hanging requests.
- Enabled autoDrain for streaming requests to ensure response streams are properly drained and reduce resource issues.
- Marked the DockerImageStore S3 integration test as skipped to avoid CI dependence on external S3 and added a cleanup test to stop the test DockerHost.
- Added local tool settings file (.claude/settings.local.json) with local permissions (development-only).

2025-08-18 - 1.3.2 - fix(package.json)

Fix test script timeout typo, update dependency versions, and add typings & project configs

- Fix test script: correct 'tineout' -> 'timeout' for npm test command and set timeout to 120s
- Add 'typings': 'dist_ts/index.d.ts' to package.json

- Bump dependencies to newer compatible versions (notable packages: @push.rocks/lik, @push.rocks/smartarchive, @push.rocks/smartbucket, @push.rocks/smartfile, @push.rocks/smartlog, @push.rocks/smartpromise, @push.rocks/smartstream, rxjs)
- Add project/config files: .serena/project.yml and .claude/settings.local.json (editor/CI metadata)
- Include generated cache/metadata files (typescript document symbols cache) — not source changes but tooling/cache artifacts

2025-08-18 - 1.3.1 - fix(test)

Update test setup and devDependencies; adjust test import and add package metadata

- Update test script to run with additional flags: --verbose, --logfile and --tineout 120
- Bump devDependencies: @git.zone/tsbuild -> ^2.6.7, @git.zone/tsrun -> ^1.3.3, @git.zone/tstest -> ^2.3.5, @push.rocks/qenv -> ^6.1.3
- Change test import from @push.rocks/tapbundle to @git.zone/tstest/tapbundle
- Add typings field (dist_ts/index.d.ts)
- Add packageManager field for pnpm@10.14.0 with integrity hash

2024-12-23 - 1.3.0 - feat(core)

Initial release of Docker client with TypeScript support

- Provides easy communication with Docker's remote API from Node.js
- Includes implementations for managing Docker services, networks, secrets, containers, and images

2024-12-23 - 1.2.8 - fix(core)

Improved the image creation process from tar stream in DockerImage class.

- Enhanced `DockerImage.createFromTarStream` method to handle streamed response and parse imported image details.
- Fixed the dependency version for `@push.rocks/smartarchive` in package.json.

2024-10-13 - 1.2.7 - fix(core)

Prepare patch release with minor fixes and improvements

2024-10-13 - 1.2.6 - fix(core)

Minor refactoring and code quality improvements.

2024-10-13 - 1.2.5 - fix(dependencies)

Update dependencies for stability improvements

- Updated @push.rocks/smartstream to version ^3.0.46
- Updated @push.rocks/tapbundle to version ^5.3.0
- Updated @types/node to version 22.7.5

2024-10-13 - 1.2.4 - fix(core)

Refactored DockerImageStore constructor to remove DockerHost dependency

- Adjusted DockerImageStore constructor to remove dependency on DockerHost
- Updated ts/classes.host.ts to align with DockerImageStore's new constructor signature

2024-08-21 - 1.2.3 - fix(dependencies)

Update dependencies to the latest versions and fix image export test

- Updated several dependencies to their latest versions in package.json.
- Enabled the previously skipped 'should export images' test.

2024-06-10 - 1.2.1-1.2.2 - Core/General

General updates and fixes.

- Fix core update

2024-06-10 - 1.2.0 - Core

Core updates and bug fixes.

- Fix core update

2024-06-08 - 1.2.0 - General/Core

Major release with core enhancements.

- Processing images with extraction, retagging, repackaging, and long-term storage

2024-06-06 - 1.1.4 - General/Imagestore

Significant feature addition.

- Add feature to process images with extraction, retagging, repackaging, and long-term storage

2024-05-08 - 1.0.112 - Images

Add new functionality for image handling.

- Can now import and export images
- Start work on local 100% JS OCI image registry

2024-06-05 - 1.1.0-1.1.3 - Core

Regular updates and fixes.

- Fix core update

2024-02-02 - 1.0.105-1.0.110 - Core

Routine core updates and fixes.

- Fix core update

2022-10-17 - 1.0.103-1.0.104 - Core

Routine core updates.

- Fix core update

2020-10-01 - 1.0.99-1.0.102 - Core

Routine core updates.

- Fix core update

2019-09-22 - 1.0.73-1.0.78 - Core

Routine updates and core fixes.

- Fix core update

2019-09-13 - 1.0.60-1.0.72 - Core

Routine updates and core fixes.

- Fix core update

2019-08-16 - 1.0.43-1.0.59 - Core

Routine updates and core fixes.

- Fix core update

2019-08-15 - 1.0.37-1.0.42 - Core

Routine updates and core fixes.

- Fix core update

2019-08-14 - 1.0.31-1.0.36 - Core

Routine updates and core fixes.

- Fix core update

2019-01-10 - 1.0.27-1.0.30 - Core

Routine updates and core fixes.

- Fix core update

2018-07-16 - 1.0.23-1.0.24 - Core

Routine updates and core fixes.

- Fix core shift to new style

2017-07-16 - 1.0.20-1.0.22 - General

Routine updates and fixes.

- Update node_modules within npmdocker

2017-04-02 - 1.0.18-1.0.19 - General

Routine updates and fixes.

- Work with npmdocker and npmts 7.x.x
- CI updates

2016-07-31 - 1.0.17 - General

Enhancements and fixes.

- Now waiting for response to be stored before ending streaming request
- Cosmetic fix

2016-07-29 - 1.0.14-1.0.16 - General

Multiple updates and features added.

- Fix request for change observable and add npmdocker
- Add request typings

2016-07-28 - 1.0.13 - Core

Fixes and preparations.

- Fixed request for newer docker
- Prepare for npmdocker

2016-06-16 - 1.0.0-1.0.2 - General

Initial sequence of releases, significant feature additions and CI setups.

- Implement container start and stop
- Implement list containers and related functions
- Add tests with in docker environment

2016-04-12 - unknown - Initial Commit

Initial project setup.

- Initial commit

Revision #4
Created 2026-03-28 12:50:19 UTC by foss.global Team
Updated 2026-03-29 16:48:10 UTC by foss.global Team