# @push.rocks/smartcontext

A module to enrich logs with context, featuring async log contexts and scope management.

- readme.md for @push.rocks/smartcontext
- changelog.md for @push.rocks/smartcontext

# readme.md for @push.rocks/smartcontext

A zero-dependency module for hierarchical async context management in Node.js, built on `AsyncLocalStorage`.

## Install

```
npm install @push.rocks/smartcontext
# or
pnpm install @push.rocks/smartcontext
```

## Usage

`@push.rocks/smartcontext` provides scoped, hierarchical key-value stores that follow async execution flow. Child scopes inherit parent data, can add or delete keys without affecting the parent, and automatically clean up when the scope exits.

This is useful for log enrichment, request-scoped metadata, transaction tracking, and any scenario where contextual data needs to flow through deeply nested async calls.

## Basic Setup

```
import { AsyncContext } from '@push.rocks/smartcontext';

const ctx = new AsyncContext();

// Add data to the root store
ctx.store.add('userId', 'u_123');
console.log(ctx.store.get('userId')); // 'u_123'
```

# Scoped Execution with `runScoped`

`runScoped` creates an isolated child store for the duration of the callback. Inside the callback, `ctx.store` transparently points to the child store. The child inherits all parent data but any additions or deletions are local to the child.

```
ctx.store.add('requestId', 'req_abc');

await ctx.runScoped(async () => {
  // Child store sees parent data
  console.log(ctx.store.get('requestId')); // 'req_abc'

  // Add child-only data
  ctx.store.add('spanId', 'span_001');
  console.log(ctx.store.get('spanId')); // 'span_001'
});

// After scope exits, child data is gone
console.log(ctx.store.get('spanId')); // undefined
// Parent data is untouched
console.log(ctx.store.get('requestId')); // 'req_abc'
```

# Deleting Keys in a Child Scope

Deleting a parent key inside a child scope only shadows it within that scope. The parent retains the original value.

```
ctx.store.add('token', 'secret_value');

await ctx.runScoped(async () => {
  ctx.store.delete('token');
  console.log(ctx.store.get('token')); // undefined (shadowed)
});

console.log(ctx.store.get('token')); // 'secret_value' (parent unaffected)
```

# Parallel Scopes

Multiple concurrent `runScoped` calls each get their own isolated child store, preventing data collisions across async tasks.

```
await Promise.all([
  ctx.runScoped(async () => {
    ctx.store.add('worker', 'A');
    // Only this scope sees worker=A
  }),
  ctx.runScoped(async () => {
    ctx.store.add('worker', 'B');
    // Only this scope sees worker=B
  }),
]);

console.log(ctx.store.get('worker')); // undefined
```

# Retrieving All Data

`store.getAll()` returns a merged view of the store hierarchy, with child values overriding parent values and deleted keys excluded.

```
ctx.store.add('env', 'production');
ctx.store.add('region', 'eu-west-1');

await ctx.runScoped(async () => {
  ctx.store.add('traceId', 'tr_xyz');
  console.log(ctx.store.getAll());
  // { env: 'production', region: 'eu-west-1', traceId: 'tr_xyz' }
});
```

# API Reference

## `AsyncContext`

| Method / Property | Description |
| --- | --- |
| `store` | The current `AsyncStore` (root or scoped child) |
| `runScoped(fn)` | Execute `fn` with an isolated child store |

## `AsyncStore`

| Method | Description |
| --- | --- |
| `add(key, value)` | Set a key-value pair |
| `get(key)` | Get a value (checks local store, then parent chain) |
| `delete(key)` | Remove a key (shadows parent key if inherited) |
| `getAll()` | Get all key-value pairs merged from the full parent chain |

# License and Legal Information

This repository is under the MIT License. Please note that the MIT License does not grant permission to use the trade names, trademarks, service marks, or product names of the project, except as necessary for reasonable use in describing the origin of the work.

# Trademarks

This project is owned and maintained by Task Venture Capital GmbH. The names and logos associated with Task Venture Capital GmbH are trademarks of Task Venture Capital GmbH and are not included within the scope of the MIT license granted herein. Usage must be approved in writing by Task Venture Capital GmbH.

# Company Information

Task Venture Capital GmbH Registered at District Court Bremen HRB 35230 HB, Germany

For any legal inquiries, please contact us at hello@task.vc. By using this repository, you acknowledge that you have read this section and agree to comply with its terms.

# changelog.md for @push.rocks/smartcontext

## 2026-02-15 - 2.3.0 - feat(asynccontext)

replace simple-async-context with Node's AsyncLocalStorage and update implementation, tests, docs, and devDependencies

- Switched internal implementation to node:async_hooks AsyncLocalStorage (ts/plugins.ts and classes.asynccontext.ts) and adapted AsyncContext to use getStore()
- Removed dependency on simple-async-context and updated package.json (devDependencies bumped for @git.zone/* and @types/node; removed @push.rocks/tapbundle and simple-async-context)
- Updated tests: removed test/test.both.ts and added test/test.node+bun.ts using @git.zone/tstest/tapbundle
- Rewrote README to reflect new examples and usage (variable names/usage and API docs)
- Updated build script in package.json (removed --web) and adjusted npmextra.json with registry/release entries and namespaced keys

## 2025-01-25 - 2.2.1 - fix(core)

Remove unused logcontext classes and update exports

- Removed classes and plugins from logcontext that were not in use.
- Adjusted exports to not include deprecated logger-related modules.

## 2025-01-25 - 2.2.0 - feat(tests)

Added a new test script to demonstrate and validate AsyncContext functionality

- Introduced jstrial.js in the test directory to ensure the correct working of AsyncContext.
- The new tests handle various scenarios, including the independence of child stores and data deletion.

# 2025-01-23 - 2.1.8 - fix(core)

Refactor and clean up class imports and exports

- Simplified class file structure by unifying imports and exports.
- Removed redundant `logcontext.*` prefixes from filenames.
- Ensured consistent path references in index.ts.

# 2025-01-23 - 2.1.7 - fix(core)

Enhanced debugging and improved dependency tracking

- Updated @types/node to version ^22.10.10.
- Updated simple-async-context to version ^0.0.23.
- Enhanced the logDebug method to check for the existence of process and environment variables.
- Fixed dependency versions in package.json.

# 2025-01-19 - 2.1.6 - fix(core)

Updated dependencies and improved AsyncStore debugging and cleanup

- Upgraded 'simple-async-context' dependency to version ^0.0.16 for consistency and improvements.
- Added detailed debugging information in AsyncStore when DEBUG environment variable is set.
- Enhanced cleanup process for deleted keys in AsyncStore.
- Removed redundant dependencies from package.json and logcontext.plugins.ts.

# 2025-01-19 - 2.1.5 - fix(dependencies)

Update dependencies for improved compatibility

- Updated @types/node to version ^22.10.7
- Updated @types/shortid to version 2.2.0
- Updated simple-async-context to version ^0.0.15

# 2025-01-19 - 2.1.4 - fix(documentation)

Remove unnecessary conclusion section from the README for better clarity.

- Removed the 'Conclusion' header which was redundant in the README.

# 2025-01-19 - 2.1.3 - fix(readme)

Update README.md for better clarity and examples.

- Enhanced documentation with clearer examples of AsyncContext usage.
- Reorganized sections for improved understanding of `runScoped` and its benefits.
- Added detailed test script example in README.md.
- Clarified data isolation and deletion mechanisms within scoped functions.

# 2025-01-19 - 2.1.2 - fix(core)

Improve scope handling in async contexts.

- Modified the handling of scoped stores within `AsyncContext.runScoped` to simplify the API usage.
- Refactored tests to verify the context's behavior and isolation between parent and child stores.

# 2025-01-18 - 2.1.1 - fix(build)

Fix tsbuild script to include missing flag

- Updated the build script in package.json to include --allowimplicitany flag.

# 2025-01-18 - 2.1.0 - feat(ci)

Add GitHub Actions workflows for CI/CD

- Introduce GitHub Actions workflows to handle CI/CD processes for different events and branch types.
- Remove GitLab CI configuration in favor of GitHub Actions.
- Ensure security audits and tests are run as part of the CI pipeline.

# 2024-05-29 to 2024-03-30 - 2.0.0 - Configuration Updates

Improvements and updates to configuration files.

- Updated project description
- Modified TypeScript configuration
- Updated npmextra.json for githost details

# 2023-07-11 to 2023-07-10 - 2.0.0 - Organizational Change

Transitioned to a new organizational scheme for improved project structure.

- Implemented new organizational scheme

# 2023-01-12 - 1.0.29 - Breaking Change

Important breaking changes introduced, switched project to ECMAScript Module (ESM) format.

- Switched to ECMAScript Module (ESM)
- Core functionality updated

# 2021-09-17 to 2020-07-20 - 1.0.28 to 1.0.23 - Core Fixes

Multiple core improvements and bug fixes across several versions.

- Core functionality updates across versions

# 2018-03-09 to 2018-03-05 - 1.0.21 to 1.0.20 - Documentation Updates

Enhancements and updates to project documentation.

- License information updated in README
- General README updates

# 2018-03-05 - 1.0.19 - Standards Update

Updated project to comply with the latest coding standards.

- Applied latest standard updates

# 2018-03-03 - 1.0.18 - Initial Working Version

Initial release with all dependencies configured.

- First working project version with dependencies

# 2017-10-16 - 1.0.2 - Documentation

Initial project documentation added.

- Added initial README file

# 2017-10-16 - 1.0.1 - Initial Release

The initial release of the project.

- Project setup and initial commit