

@push.rocks/smartexit

A library for managing graceful shutdowns of Node.js processes by handling cleanup operations, including terminating child processes.

- [readme.md for @push.rocks/smartexit](#)
- [changelog.md for @push.rocks/smartexit](#)

readme.md for @push.rocks/smartexit

A library for managing graceful shutdowns of Node.js processes by handling cleanup operations, including terminating child processes.

Issue Reporting and Security

For reporting bugs, issues, or security vulnerabilities, please visit community.foss.global/. This is the central community hub for all issue reporting. Developers who sign and comply with our contribution agreement and go through identification can also get a code.foss.global/ account to submit Pull Requests directly.

Install

To install `@push.rocks/smartexit`, use pnpm:

```
pnpm install @push.rocks/smartexit
```

Usage

`@push.rocks/smartexit` helps you gracefully shut down Node.js applications by running cleanup operations before the process exits. It handles signal interception (`SIGINT`, `SIGTERM`), uncaught exception management, and child process termination.

Basic Setup

```
import { SmartExit } from '@push.rocks/smartexit';
```

```
// Create an instance - this automatically hooks into process signals
const smartExit = new SmartExit();
```

Registering Cleanup Functions

Define custom async cleanup functions that execute before the process exits:

```
smartExit.addCleanupFunction(async () => {
  console.log('Closing database connections...');
  await database.close();
});

smartExit.addCleanupFunction(async () => {
  console.log('Flushing logs...');
  await logger.flush();
});
```

Managing Child Processes

Track spawned child processes so they get properly terminated on exit:

```
import { spawn } from 'child_process';

const childProcess = spawn('node', ['worker.js']);
smartExit.addProcess(childProcess);

// Remove from tracking if the process ends naturally
childProcess.on('exit', () => {
  smartExit.removeProcess(childProcess);
});
```

Kill Process Trees by PID

Terminate an entire process tree using the static `killTreeByPid` method:

```
import { SmartExit, type TProcessSignal } from '@push.rocks/smartexit';
```

```
// Kill with default SIGKILL
await SmartExit.killTreeByPid(12345);

// Kill with a specific signal
await SmartExit.killTreeByPid(12345, 'SIGTERM');
```

Triggering Cleanup Manually

While `SmartExit` automatically hooks into `SIGINT`, `SIGTERM`, and uncaught exceptions, you can manually trigger cleanup:

```
await smartExit.killAll();
process.exit(0);
```

Integrating with Express

Gracefully close an Express server on shutdown:

```
import express from 'express';
import { SmartExit } from '@push.rocks/smartexit';

const app = express();
const smartExit = new SmartExit();

const server = app.listen(3000, () => {
  console.log('Server running on port 3000');
});

smartExit.addCleanupFunction(async () => {
  console.log('Closing Express server...');
  await new Promise<void>((resolve) => server.close(() => resolve()));
});
```

Available Process Signals

The `TProcessSignal` type provides all standard POSIX signals:

```
import type { TProcessSignal } from '@push.rocks/smartexit';

// Examples: 'SIGINT', 'SIGTERM', 'SIGKILL', 'SIGHUP', 'SIGUSR1', 'SIGUSR2', etc.
```

How It Works

When you create a `SmartExit` instance, it automatically:

1. **Hooks `SIGINT`** (Ctrl+C): Runs cleanup and exits with code 0
2. **Hooks process `exit`**: Runs cleanup when `process.exit(0)` is called
3. **Catches uncaught exceptions**: Logs the error, runs cleanup, and exits with code 1

On `killall()`, it:

- Sends `SIGINT` to all tracked child processes
- Waits 10 seconds, then sends `SIGKILL` if processes are still alive
- Executes all registered cleanup functions

License and Legal Information

This repository contains open-source code licensed under the MIT License. A copy of the license can be found in the [LICENSE](#) file.

Please note: The MIT License does not grant permission to use the trade names, trademarks, service marks, or product names of the project, except as required for reasonable and customary use in describing the origin of the work and reproducing the content of the NOTICE file.

Trademarks

This project is owned and maintained by Task Venture Capital GmbH. The names and logos associated with Task Venture Capital GmbH and any related products or services are trademarks of Task Venture Capital GmbH or third parties, and are not included within the scope of the MIT license granted herein.

Use of these trademarks must comply with Task Venture Capital GmbH's Trademark Guidelines or the guidelines of the respective third-party owners, and any usage must be approved in writing. Third-party trademarks used herein are the property of their respective owners and used only in a descriptive manner, e.g. for an implementation of an API or similar.

Company Information

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

For any legal inquiries or further information, please contact us via email at hello@task.vc.

By using this repository, you acknowledge that you have read this section, agree to comply with its terms, and understand that the licensing of the code does not imply endorsement by Task Venture Capital GmbH of any derivative works.

changelog.md for @push.rocks/smartexit

2026-03-04 - 2.0.3 - fix(smartexit)

use native OS methods to kill process trees; remove tree-kill dependency

- Replaced tree-kill usage with taskkill on Windows and process.kill(-pid) on POSIX.
- Removed tree-kill package from dependencies and removed its export from plugins.
- Handle ESRCH (no such process/group) as non-error when killing process groups.
- Ensure trackedPids is cleared when a child process is removed.

2026-03-03 - 1.1.1 - fix(shutdown)

kill full child process trees and add synchronous exit handler to force-kill remaining child processes

- Use tree-kill via SmartExit.killTreeByPid to terminate entire process trees (attempt SIGTERM, fallback to SIGKILL).
- Replace previous delayed/process.kill calls with awaitable tree-kill for more reliable termination.
- Add a synchronous process.on('exit') handler that force-kills any remaining child processes as a last-resort safety net.
- No public API changes; internal robustness/bugfix to shutdown behavior.

2025-12-15 - 1.1.0 - feat(smartexit)

Add silent logging option, structured shutdown logs, and killAll return stats

- Introduce ISmartExitOptions with a silent flag to disable console logging
- Add internal log() helper and use a [smartexit] prefix for shutdown/error messages

- killAll() now returns Promise<{ processesKilled, cleanupFunctionsRan }> and tallies processes and cleanup functions run
- Constructor accepts options (backwards compatible) to configure behavior
- Documentation (readme.hints.md) updated with usage and example output

2025-12-15 - 1.0.24 - fix(deps)

bump dependencies, update tests and docs, adjust tsconfig and npm release config

- Upgrade devDependencies: @git.zone/tsbuild -> ^4.0.2, @git.zone/tsrun -> ^2.0.1, @git.zone/tstest -> ^3.1.3, @types/node -> ^24.0.3 (removed @push.rocks/tapbundle)
- Bump runtime deps: @push.rocks/lik -> ^6.2.2, @push.rocks/smartpromise -> ^4.2.3
- Switch test imports to @git.zone/tstest/tapbundle and export default tap.start()
- Revise README: expanded documentation, new install instructions (pnpm), issue reporting guidance, examples (Express, killTreeByPid), and wording/formatting improvements
- Remove experimentalDecorators and useDefineForClassFields from tsconfig compilerOptions
- Add @git.zone/cli release registry configuration to npmextra.json (verdaccio + npm registry)

2024-05-29 - 1.0.23 - docs

Update package description.

- Updated project/package description text.

2024-04-18 - 1.0.22 - fix(core)

Core fixes and maintenance.

- Miscellaneous core updates and bug fixes.

2024-04-18 - 1.0.21 - core/build

Maintenance and config updates for 1.0.21.

- Miscellaneous core fixes.
- Updated tsconfig.
- Updated npmextra.json (githost) across multiple commits.

2023-09-11 - 1.0.20 - maintenance

Organization and core adjustments.

- Switched to new organization scheme.
- Miscellaneous core fixes.

2021-08-17 - 1.0.19 - 1.0.2 - maintenance

Multiple minor core fixes and release/tag-only commits (grouped).

- Series of minor core fixes and maintenance updates across versions 1.0.2 through 1.0.19.
- Many commits are release tags or generic core updates without additional user-facing changes.

2019-05-23 - 1.0.8 - fix(core)

Improve process termination behavior.

- Now killing groups of processes with negative PIDs to ensure child process groups are handled correctly.

2019-05-22 - 1.0.7 - fix(core)

Stronger process kill handling.

- Added SIGKILL fallback with a 10-second delay to ensure processes are terminated when needed.

2019-05-16 - 1.0.1 - init

Initial core commit.

- Initial core implementation and setup (recorded as `fix(core): initial`).