changelog.md for @push.rocks/taskbuffer

2026-03-23 - 8.0.2 - fix(servicemanager)

cancel startup timeout once service initialization completes

- Replaces the startup timeout race delay with a cancellable Timeout instance
- Prevents the global startup timeout from lingering after startup finishes or fails

2026-03-23 - 8.0.1 - fix(servicemanager)

cancel shutdown timeouts after services stop

- Replace the shutdown race delay with a cancellable Timeout in ServiceManager.
- Prevent timeout handlers from lingering after a service stops successfully during shutdown.

2026-03-20 - 8.0.0 - BREAKING CHANGE(service)

expand service lifecycle management with instance-aware hooks, startup timeouts, labels, readiness waits, and auto-restart support

- Change service stop and health check callbacks to receive the started instance and expose it via service.instance
- Add per-service and global startup timeout handling plus waitForState, waitForRunning, and waitForStopped readiness helpers
- Support service labels, label-based manager queries, and auto-restart lifecycle events with configurable backoff

2026-02-15 - 6.1.2 - fix(deps)

bump @push.rocks/smarttime to ^4.2.3

- Updated @push.rocks/smarttime from ^4.1.1 to ^4.2.3
- Non-breaking dependency version bump; increment patch version

2026-02-15 - 6.1.1 - fix(tests)

improve buffered task tests: add chain, concurrency and queue behavior tests

- Replace tools.delayFor with @push.rocks/smartdelay for more deterministic timing in tests
- Add tests for afterTask chaining, bufferMax concurrency, queued-run limits, and re-trigger behavior
- Rename tasks to descriptive names and fix afterTask chaining order to avoid circular references
- Change test runner invocation to export default tap.start() instead of calling tap.start() directly

2026-02-15 - 6.1.0 - feat(taskbuffer)

add sliding-window rate limiting and result-sharing to TaskConstraintGroup and integrate with TaskManager

- Added IRateLimitConfig and TResultSharingMode types and exported them from the public index
- TaskConstraintGroup: added rateLimit and resultSharingMode options, internal completion timestamp tracking, and last-result storage

- TaskConstraintGroup: new helpers - pruneCompletionTimestamps, getRateLimitDelay, getNextAvailableDelay, recordResult, getLastResult, hasResultSharing
- TaskConstraintGroup: rate-limit logic enforces maxPerWindow (counts running + completions) and composes with cooldown/maxConcurrent
- TaskManager: records successful task results to constraint groups and resolves queued entries immediately when a shared result exists
- TaskManager: queue drain now considers unified next-available delay (cooldown + rate limit) when scheduling retries
- Documentation updated: README and hints with usage examples for sliding-window rate limiting and result sharing
- Comprehensive tests added for rate limiting, concurrency interaction, and result-sharing behavior

2026-02-15 - 6.0.1 - fix(taskbuffer)

no changes to commit

- Git diff shows no changes
- package.json current version is 6.0.0; no version bump required

2026-02-15 - 6.0.0 - BREAKING CHANGE(constraints)

make TaskConstraintGroup constraint matcher input-aware and add shouldExecute pre-execution hook

- Rename ITaskConstraintGroupOptions.constraintKeyForTask -> constraintKeyForExecution(task, input?) and update TaskConstraintGroup.getConstraintKey signature
- Add optional shouldExecute(task, input?) hook; TaskManager checks shouldExecute before immediate runs, after acquiring slots, and when draining the constraint queue (queued tasks are skipped when shouldExecute returns false)
- Export ITaskExecution type and store constraintKeys on queued entries (IConstrainedTaskEntry.constraintKeys)
- Documentation and tests updated to demonstrate input-aware constraint keys and shouldExecute pruning

2026-02-15 - 5.0.1 - fix(tests)

add and tighten constraint-related tests covering return values, error propagation, concurrency, cooldown timing, and constraint removal

- Tightened cooldown timing assertion from >=100ms to >=250ms to reflect 300ms cooldown with 50ms tolerance.
- Added tests for queued task return values, error propagation when catchErrors is false, and error swallowing behavior when catchErrors is true.
- Added concurrency and cooldown interaction tests to ensure maxConcurrent is respected and batch timing is correct.
- Added test verifying removing a constraint group unblocks queued tasks and drain behavior completes correctly.

2026-02-15 - 5.0.0 - BREAKING CHANGE(taskbuffer)

Introduce constraint-based concurrency with TaskConstraintGroup and TaskManager integration; remove legacy TaskRunner and several Task APIs (breaking); add typed Task.data and update exports and tests.

- Add TaskConstraintGroup class with per-key maxConcurrent, cooldownMs, and helper methods (canRun, acquireSlot, releaseSlot, getCooldownRemaining, getRunningCount, reset).
- Task generic signature extended to Task<T, TSteps, TData> and a new typed data property (data) with default {}.
- TaskManager now supports addConstraintGroup/removeConstraintGroup, triggerTaskConstrained, queues blocked tasks, drains queue with cooldown timers, and routes triggerTask/triggerTaskByName through the constraint system.
- Removed TaskRunner, plus Task APIs: blockingTasks, execDelay, finished promise and associated behavior have been removed (breaking changes).
- Exports and interfaces updated: TaskConstraintGroup and ITaskConstraintGroupOptions added; TaskRunner removed from public API.
- Updated README and added comprehensive tests for constraint behavior; adjusted other tests to remove TaskRunner usage and reflect new APIs.

2026-02-15 - 4.2.1 - fix(deps)

bump @push.rocks/smartlog and @types/node; update dependency list version and license link in docs

- package.json: update @push.rocks/smartlog from ^3.1.10 to ^3.1.11
- package.json: update @types/node from ^25.1.0 to ^25.2.3
- readme.hints.md: update 'Dependencies (as of v4.1.1)' to 'Dependencies (as of v4.2.0)' and reflect bumped dependency versions
- readme.md: change license link text to 'LICENSE'

2026-01-29 - 4.2.0 - feat(ts_web)

support TC39 'accessor' decorators for web components; bump dependencies and devDependencies; rename browser tests to .chromium.ts; move LICENSE to license.md and update readme

- Convert web component class fields to use the TC39 'accessor' keyword in ts_web/taskbuffer-dashboard.ts to be compatible with @design.estate/dees-element v2.1.6
- Bump @design.estate/dees-element to ^2.1.6 and update devDependencies (@git.zone/tsbuild, @git.zone/tsbundle, @git.zone/tsrun, @git.zone/tstest, @types/node) to newer versions
- Replace test/test.10.webcomponent.browser.ts with test/test.10.webcomponent.chromium.ts and update testing guidance in readme.hints.md to prefer .chromium.ts
- Move LICENSE file content to license.md and update readme.md to reference the new license file
- Small test cleanups: remove obsolete tslint:disable comments

2026-01-26 - 4.1.1 - fix(ts_web)

fix web dashboard typings and update generated commit info

- Updated generated commit info file ts_web/00_commitinfo_data.ts to version 4.1.0
- Large changes applied to web/TS build files (net +529 additions, -399 deletions) — likely fixes and typing/refactor improvements in ts_web/dashboard
- package.json remains at 4.1.0; recommend a patch bump to 4.1.1 for these fixes

2026-01-26 - 4.1.0 - feat(task)

add task labels and push-based task events

- Introduce Task labels: Task accepts labels in constructor and exposes setLabel/getLabel/removeLabel/hasLabel; labels are included (shallow copy) in getMetadata().
- Add push-based events: Task.eventSubject (rxjs Subject<ITaskEvent>) emits 'started','step','completed','failed' with timestamp; 'step' includes stepName and 'failed' includes error string.
- Task now emits events during lifecycle: emits 'started' at run start, 'step' on notifyStep, and 'completed' or 'failed' when finished or errored. getMetadata() now includes labels.
- TaskManager aggregates task events into taskSubject, subscribes on addTask and unsubscribes on removeTask/stop; includes helper methods getTasksByLabel and getTasksMetadataByLabel.
- Public API updated: exported ITaskEvent and TTaskEventType in ts/index.ts and interfaces updated (labels in metadata, new event types).
- Tests and docs: added test/test.12.labels-and-events.ts and updated readme.hints.md to document labels and push-based events.

2026-01-25 - 4.0.0 - BREAKING CHANGE(taskbuffer)

Change default Task error handling: trigger() now rejects when taskFunction throws; add catchErrors option (default false) to preserve previous swallow behavior; track errors (lastError, errorCount) and expose them in metadata; improve error propagation and logging across runners, chains, parallels and debounced tasks; add tests and documentation for new behavior.

- Introduce catchErrors option on Task (default: false) — previously errors were swallowed by default
- Tasks now set lastError and increment errorCount when failures occur; clearError() added to reset error state
- getMetadata() now reports status 'failed' and includes lastError and errorCount
- Task.run flow updated to reset error state at start, log errors, and either swallow or rethrow based on catchErrors
- BufferRunner, TaskRunner, Taskchain, Taskparallel, TaskDebounced and TaskManager updated to handle errors, avoid hanging promises, and use logger instead of console
- Added comprehensive tests (test/test.11.errorhandling.ts) and readme hints documenting the new error-handling behavior (v3.6.0+)
- npmextra.json updated for @git.zone/cli and release registries

2025-12-04 - 3.5.0 - feat(core)

Add debounced tasks and step-based progress tracking; upgrade deps and improve dashboard and scheduling

- Add TaskDebounced class to coalesce rapid triggers into a single execution (debounce behavior).
- Introduce step tracking and progress reporting on Task via TaskStep, getProgress(), getStepsMetadata(), getMetadata(), resetSteps(), and completeAllSteps().
- Enhance buffered execution flow: BufferRunner and CycleCounter improvements to better coordinate buffered runs and cycle promises.
- Standardize concurrent runner naming (Taskparallel) and update related exports/usages (ts/index.ts, readme examples).
- Enhance TaskManager scheduling/metadata: getScheduledTasks now returns schedule and nextRun, addExecuteRemoveTask collects execution report metadata and cleans up after execution, distributed coordination hooks retained.
- Add/upgrade web dashboard UI, demos and refresh logic to surface task metadata, scheduled tasks and progress.
- Bump runtime and dev dependencies (multiple @push.rocks packages and @git.zone tooling).
- Update tests: reduce iteration threshold and tighten schedule interval in test/test.4.taskmanager.ts.
- Remove several .serena memory files (project overview, style guides and suggested commands) as cleanup.

2025-09-07 - 3.4.0 - feat(taskbuffer-dashboard)

Add TaskBuffer dashboard web component, demo and browser tests; add HTML entry and update dependencies

- Introduce a new web component taskbuffer-dashboard for real-time visualization of tasks and schedules (ts_web/taskbuffer-dashboard.ts).
- Add a demo wrapper and interactive UI for the dashboard (ts_web/elements/taskbuffer-dashboard.demo.ts).
- Provide web exports and typings for web usage (ts_web/index.ts) and include an HTML entry (html/index.html).
- Add browser-oriented tests to validate metadata structures for the web component (test/test.10.webcomponent.browser.ts).
- Bump package version to 3.3.0 in package.json as part of this change.
- Update/add dependencies and devDependencies (@design.estate/dees-element added; smartlog, @git.zone/tsbuild and @git.zone/tstest bumped).

2025-09-06 - 3.2.0 - feat(core)

Add step-based progress tracking, task metadata and enhanced TaskManager scheduling/metadata APIs

- Introduce TaskStep class for named, weighted steps with timing and status (pending|active|completed).
- Add step-tracking to Task: notifyStep, getProgress, getStepsMetadata, getMetadata, resetSteps and internal step lifecycle handling.
- Task now records runCount and lastRun; Task.run flow resets/cleans steps and aggregates progress.
- TaskManager enhancements: schedule/deschedule improvements, performDistributedConsultation, and new metadata-focused APIs: getTaskMetadata, getAllTasksMetadata, getScheduledTasks, getNextScheduledRuns, addExecuteRemoveTask (exec + collect report).
- Exports updated: TaskStep and related types exported from index, plus Task metadata interfaces.
- Comprehensive README updates documenting step-based progress tracking, metadata, TaskManager and examples.
- New/updated tests added for step behavior and metadata (test/test.9.steps.ts) and other TS additions.
- Minor build/script change: build script updated to use 'tsbuild tsfolders'.

2025-08-26 - 3.1.10 - fix(task)

Implement core Task execution flow, buffering and lifecycle; update README with generics and buffer docs

- Implement Task.runTask including preTask/afterTask chaining, touched-task cycle prevention and error handling.
- Add Task helpers: extractTask, isTask, isTaskTouched and emptyTaskFunction (resolved promise).
- Introduce task lifecycle coordination: finished promise, resolveFinished, and blockingTasks to await dependent tasks.
- Support taskSetup/setupValue, execDelay handling, and wait-for-blocking-tasks before execution.
- Wire up trigger() to choose buffered vs unbuffered execution (triggerBuffered / triggerUnBuffered) and integrate BufferRunner.
- Improve logging and safer promise handling (caught errors are logged).
- Update README with extended TypeScript generics examples and expanded buffer behavior and strategies documentation.

2025-08-26 - 3.1.9 - fix(tests)

Update CI workflows, fix tests and refresh README/package metadata

- CI: switch Docker image to code.foss.global/host.today/ht-docker-node:npmci and adjust NPMCI_COMPUTED_REPOURL; replace npmci installer package name from @shipzone/npmci to @ship.zone/npmci in Gitea workflows
- Tests: update test imports to use @git.zone/tstest/tapbundle and apply small formatting fixes to test files
- Package metadata: update bugs URL and homepage to code.foss.global, add a pnpm.overrides placeholder in package.json
- .gitignore: add AI/tooling directories (.claude, .serena) and reorganize custom section
- Code style/TS fixes: minor formatting changes across ts sources (trailing commas, line breaks, consistent object/argument commas) and small API surface formatting fixes
- Documentation: whitespace/formatting cleanups in README and add changelog entry for 3.1.8

2025-08-26 - 3.1.8 - fix(tests)

Update test runner and imports, refresh README and package metadata, add project tooling/config files

- Replaced test imports from '@push.rocks/tapbundle' to '@git.zone/tstest/tapbundle' across test files
- Updated test script in package.json to run tstest with --verbose --logfile --timeout 120
- Bumped devDependency @git.zone/tstest to ^2.3.5 and adjusted package.json fields (typings, packageManager)
- Expanded and rewrote README with detailed examples, API reference, and usage guidance
- Refactored TaskManager tests (removed duplicate both-file and added consolidated test/test.4.taskmanager.ts)
- Added development/project tooling and metadata files (.claude settings, .serena project/memories) to aid local development and CI

2024-05-29 - 3.1.7 - maintenance/config

Updated package metadata and build configuration.

- Updated package description.
- Multiple TypeScript configuration updates (tsconfig).
- Updated npmextra.json githost entries (changes across 2024-03-30, 2024-04-01, 2024-04-14).

2023-08-04 - 3.0.15 - feat(Task)

Tasks can now be blocked by other tasks.

- Introduced task blocking support in the Task implementation.
- Release contains related maintenance and patch fixes.

2023-01-07 to 2023-10-20 - 3.0.4..3.1.6 - maintenance

Series of patch releases focused on core fixes and stability.

- Numerous core fixes and small adjustments across many patch versions.
- General maintenance: bug fixes, internal updates and stability improvements.

2022-03-25 - 2.1.17 - BREAKING(core)

Switched module format to ESM (breaking).

- BREAKING CHANGE: project now uses ESM module format.
- Release includes the version bump and migration to ESM.

2019-11-28 - 2.0.16 - feat(taskrunner)

Introduce a working task runner.

- Added/activated a working taskrunner implementation.
- Improvements to task execution and orchestration.

2019-09-05 to 2022-11-14 - 2.0.3..2.1.16 - maintenance

Ongoing maintenance and incremental fixes between 2.0.x and 2.1.x series.

- Multiple fixes labeled as core maintenance updates.
- CI, packaging and small doc/test fixes rolled out across these releases.

2018-08-04 - 2.0.0 - major

Major release and scope change with CI/test updates.

- Released 2.0.0 with updated docs.
- BREAKING CHANGE: package scope switched to @pushrocks (scope migration).
- CI and testing updates (moved to new tstest), package.json adjustments.

2017-07-12 - 1.0.21 - enhancements

Feature additions around task utilities and manager.

- Introduced TaskOnce.
- Implemented TaskManager (added TaskManager class and improvements across 1.0.10–1.0.16).
- Implemented execDelay for tasks.
- Documentation and test improvements.

2016-08-03 - 1.0.6 - types

Type and promise improvements.

- Now returns correct Promise types.
- Dependency and typings updates.

2016-08-01 - 1.0.0 - stable

First stable 1.0.0 release.

- Exported public interfaces.
- Base API stabilized for 1.x line.

2016-05-15 to 2016-05-06 - 0.1.0..0.0.5 - initial features

Initial implementation of core task primitives and utilities.

- Added Taskparallel class to execute tasks in parallel.
- Introduced basic Task class and working taskchain.
- Added logging and initial task buffering behavior.
- Improvements to README, typings and packaging.
- Early CI and build setup (Travis/GitLab CI).

Revision #2
Created 2026-03-28 13:10:47 UTC by foss.global Team
Updated 2026-03-29 16:53:17 UTC by foss.global Team