@git.zone/tsdoc

Documentation for @git.zone/tsdoc

- readme.md for @git.zone/tsdoc
- changelog.md for @git.zone/tsdoc

readme.md for @git.zone/tsdoc

AI-Powered Documentation & Commit Intelligence for TypeScript Projects □□

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

```
# Global installation (recommended for CLI usage)
pnpm add -g @git.zone/tsdoc

# Or use with npx (no install needed)
npx @git.zone/tsdoc

# Or install locally as a project dependency
pnpm add @git.zone/tsdoc
```

Usage

`@git.zone/tsdoc` is a TypeScript documentation powerhouse that combines traditional TypeDoc API docs with AI-powered documentation workflows. It uses OpenAI models via `@push.rocks/smartai` and autonomous agents via `@push.rocks/smartagent` to generate READMEs, project descriptions, keywords, and semantic commit messages — all by intelligently exploring your project's codebase with scoped filesystem tools.

CLI Commands

Command	Description
`tsdoc`	□□ Auto-detects project type and runs TypeDoc
`tsdoc aidoc`	□□ Generates AI-powered README + description/keywords
`tsdoc readme`	□□ Generates AI-powered README only
`tsdoc description`	□□ Generates AI-powered description and keywords only
`tsdoc commit`	□□ Generates a semantic commit message from uncommitted changes
`tsdoc typedoc`	□□ Generates traditional TypeDoc API documentation

□□ AI-Powered Documentation (`aidoc`)

The `aidoc` command is the all-in-one workflow that combines README generation and description/keyword generation:

```
# In your project root
tsdoc aidoc
```

This will:

1. Spin up an AI agent with read-only filesystem access scoped to your project
2. The agent autonomously explores your project structure, reads source files, and understands the API
3. Generate a comprehensive `readme.md` with install instructions, usage examples, and architecture overview
4. Update `package.json` and `.smartconfig.json` with an AI-generated description and keywords

You can also run these steps individually:

```
# Generate only the README
tsdoc readme

# Generate only the description and keywords
tsdoc description
```

□□ Smart Commit Messages (`commit`)

The `commit` command analyzes your uncommitted changes and produces a structured commit object following Conventional Commits:

```
tsdoc commit
```

Output is a JSON object:

```
{
  "recommendedNextVersionLevel": "feat",
  "recommendedNextVersionScope": "core",
  "recommendedNextVersionMessage": "add smart diff processing for large changesets",
  "recommendedNextVersionDetails": [
    "implemented intelligent diff sampling with head/tail extraction",
    "added file prioritization by importance score"
  ],
  "recommendedNextVersion": "1.13.0",
  "changelog": "# Changelog\n\n## 2026-03-24 - 1.13.0 - core\n..."
}
```

Under the hood, the commit flow:

- **Excludes noise**: Lock files, build artifacts (`dist/`, `dist_*/`), IDE directories, caches, and source maps are filtered out before processing
- **Prioritizes what matters**: Source files rank higher than test files, which rank higher than config, docs, and build artifacts
- **Handles large diffs gracefully**: The `DiffProcessor` categorizes files by size — small files (< 300 lines) are included in full, medium files (< 800 lines) get head/tail sampling, and large files are metadata-only
- **Respects token budgets**: Dynamically calculates available tokens based on the model's context limit minus overhead
- **Auto-generates changelogs**: If no `changelog.md` exists, one is created from the full git history

□□ TypeDoc Generation (`typedoc`)

For traditional API documentation:

```
# Generate to default ./public directory
tsdoc typedoc

# Generate to a specific subdirectory
```

```
tsdoc typedoc --publicSubdir docs
```

TypeDoc generation auto-detects your source directories (`ts/` and `ts_web/`) and creates a temporary tsconfig for compilation.

□□ Monorepo Support

When generating READMEs, tsdoc automatically detects monorepo submodules via `@git.zone/tspublish` conventions. Each submodule directory containing a `tspublish.json` gets its own generated README with the legal section appended.

Programmatic API

You can use tsdoc programmatically in your own tools:

```
import { AiDoc } from '@git.zone/tsdoc';

const aidoc = new AiDoc();

// Initialize — prompts for OpenAI token on first run, then persists it
await aidoc.start();

// Generate a comprehensive README for a project
const readmeContent = await aidoc.buildReadme('/path/to/project');

// Generate description and keywords, updating package.json and .smartconfig.json
await aidoc.buildDescription('/path/to/project');

// Generate a structured commit message object from uncommitted changes
const commitObj = await aidoc.buildNextCommitObject('/path/to/project');
console.log(commitObj.recommendedNextVersionLevel); // 'fix' | 'feat' | 'BREAKING CHANGE'
console.log(commitObj.recommendedNextVersionMessage);
console.log(commitObj.changelog);

// Get gathered project files (package.json, source files, tests, config)
const context = await aidoc.getProjectContext('/path/to/project');

// Get token count for a project's context
const tokenCount = await aidoc.getProjectContextTokenCount('/path/to/project');
```

```
// Estimate tokens in arbitrary text
const tokens = aidoc.countTokens('some text here');

await aidoc.stop();
```

You can also pass the OpenAI token directly via the constructor:

```
const aidoc = new AiDoc({ OPENAI_TOKEN: 'sk-...' });
await aidoc.start();
```

Configuration

OpenAI Token

An OpenAI API key is required for all AI features. It can be provided in three ways (checked in order):

1. **Environment variable**: Set `OPENAI_TOKEN` in your environment or `.env` file
2. **Constructor argument**: Pass `{ OPENAI_TOKEN: 'sk-...' }` to `new AiDoc()`
3. **Interactive prompt**: On first run, tsdoc will prompt for the token and persist it

The token is persisted at `~/.smartconfig/kv/@git.zone/tsdoc.json` for subsequent runs.

.smartconfig.json

tsdoc uses `.smartconfig.json` for project metadata. The `tsdoc` key holds legal information that gets appended to generated READMEs:

```
{
  "tsdoc": {
    "legal": "\n## License and Legal Information\n\n..."
  },
  "gitzone": {
    "module": {
      "githost": "gitlab.com",
      "gitscope": "gitzone",
      "gitrepo": "tsdoc",
```

```
      "npmPackagename": "@git.zone/tsdoc",
      "description": "...",
      "keywords": ["..."]
    }
  }
}
```

The `description` command writes updated description/keywords to both `gitzone.module` in `.smartconfig.json` and to `package.json`.

Architecture

Core Components

```
@git.zone/tsdoc
├── AiDoc                # Main orchestrator — manages AI model, delegates to task classes
├── TypeDoc              # Traditional TypeDoc API documentation generation
├── ProjectContext       # Gathers project files (package.json, source, tests, config)
├── DiffProcessor        # Intelligent git diff processing with prioritization & sampling
├── Readme               # AI agent-driven README generation with filesystem tools
├── Commit               # AI agent-driven commit message generation with diff analysis
├── Description          # AI agent-driven description and keyword generation
└── CLI                  # Command-line interface built on @push.rocks/smartcli
```

□□ AI Agent Architecture

Each documentation task (readme, commit, description) runs an autonomous AI agent via `@push.rocks/smartagent`'s `runAgent()`:

1. **System prompt** defines the agent's role, constraints, and output format
2. **Filesystem tools** give the agent scoped, read-only access to the project directory
3. **Autonomous exploration** — the agent decides which files to read, in what order
4. **Structured output** — README markdown, commit JSON, or description JSON

The agents use `@push.rocks/smartai`'s `getModel()` to create a language model instance backed by OpenAI.

⚡ Diff Processing Pipeline

The `DiffProcessor` handles large git diffs without blowing up token budgets:

File Category	Threshold	Treatment
Small	< 300 lines changed	Included in full
Medium	< 800 lines changed	Head (75 lines) + tail (75 lines) sampling
Large	≥ 800 lines changed	Metadata only (filepath + stats)

Files are scored by importance:

- **100** — Source files (`src/`, `lib/`, `app/`, `components/`, `pages/`, `api/`)
- **80** — Test files (`test/`, `*.test.ts`, `*.spec.ts`)
- **70** — Interface/type files, entry points (`index.ts`, `mod.ts`)
- **60** — Configuration files (`.json`, `.yaml`, `.config.ts`)
- **40** — Documentation (`.md`, `.txt`)
- **10** — Build artifacts (`dist/`, `build/`, `.next/`)

Token budget is calculated dynamically: `context_limit - safety_margin - overhead - prompt_size`.

Requirements

- **Node.js** >= 18
- **TypeScript** project with a `ts/` source directory
- **OpenAI API key** for AI features

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 @git.zone/tsdoc

2026-03-24 - 2.0.2 - fix(smartconfig)

migrate project metadata and config handling to .smartconfig.json

- replace npmextra.json with .smartconfig.json across packaging, project context, README generation, and description prompts
- fix description updates to write metadata under gitzone.module and use smartconfig KeyValueStore for persisted settings
- refresh documentation and dependency versions to reflect the new config location and storage path

2026-03-24 - 2.0.1 - fix(aidocs, config)

migrate aidocs configuration handling from npmextra to smartconfig

- replace the @push.rocks/npmextra dependency with @push.rocks/smartconfig
- load project metadata from smartconfig.json instead of npmextra.json in aidocs workflows
- move user key-value store paths from ~/.npmextra to ~/.smartconfig and preserve existing OPENAI_TOKEN data through migration

2026-03-11 - 2.0.0 - BREAKING CHANGE(aidoc)

migrate agent orchestration to new runAgent API and filesystem tools; refactor model handling and update README and tests

- Replace DualAgentOrchestrator with plugins.smartagent.runAgent and scoped filesystem tools
- Introduce smartagentTools export and use filesystemTool for agents
- Replace smartAiInstance with model via plugins.smartai.getModel() and remove previous lifecycle methods (breaking API change)
- Normalize agent output property from result to text and standardize log messages (removed emojis)
- Update changelog/README/description generation flows to use new agent interface
- Bump several devDependencies and dependencies (tsbuild, tstest, @types/node, tspublish, push.rocks packages, typedoc, typescript)
- Change test entry to export default tap.start()
- Revise README content and structure

2026-01-04 - 1.12.0 - feat(commit)

add token budgeting and dynamic diff token calculation to avoid OpenAI context limit issues

- Introduce TOKEN_BUDGET constants and calculateMaxDiffTokens() in ts/aidocs_classes/commit.ts
- Use dynamic maxDiffTokens for DiffProcessor and validate/log warnings when estimated tokens approach limits
- Add token budgeting notes to readme.hints.md (guidance for splitting large commits and adjusting overhead)
- Bump dependencies/devDependencies: @git.zone/tstest ^3.1.4, @types/node ^25.0.3, @git.zone/tspublish ^1.11.0, @push.rocks/smartfs ^1.3.1

2025-12-16 - 1.11.4 - fix(aidocs_classes)

clarify recommendedNextVersionMessage field to require only the description body without the type(scope) prefix

- Updated inline documentation in ts/aidocs_classes/commit.ts to explicitly state that recommendedNextVersionMessage must be only the description body (example: 'bump dependency to ^1.2.6') and not include the type(scope) prefix.

- Removes ambiguity in the example text and improves guidance for commit message generation.

2025-12-15 - 1.11.0 - feat(commit)

Integrate DualAgentOrchestrator for commit message generation and improve diff/context handling

- Add @push.rocks/smartagent dependency and export it from plugins
- Use DualAgentOrchestrator to generate and guardian-validate commit messages
- Use DualAgentOrchestrator for changelog generation with guardian validation
- Switch commit flow to TaskContextFactory and DiffProcessor for token-efficient context
- Expose getOpenaiToken() and wire orchestrator with the project OpenAI token
- Enhance iterative context builder and context components to better manage token budgets and sampling
- Update npmextra.json with release config for @git.zone/cli and reference local smartagent package in package.json

2025-12-02 - 1.10.0 - feat(diff-processor)

Improve diff sampling and file prioritization: increase inclusion thresholds, expand sampled context, and boost priority for interface/type and entry-point files

- Raise small/medium file thresholds used by DiffProcessor (smallFileLines 50 -> 300, mediumFileLines 200 -> 800) so more source files are included fully or summarized rather than treated as large metadata-only files
- Increase sample window for medium files (sampleHeadLines/sampleTailLines 20 -> 75) to provide more context when summarizing diffs
- Boost importance scoring for interfaces/type files and entry points (adds +20 for interfaces/.types and +15 for index/mod entry files) to prioritize critical API surface in diff processing
- Keep other prioritization rules intact (source/test/config/docs/build heuristics), and align the aidoc commit DiffProcessor usage with the new defaults

2025-11-04 - 1.9.2 - fix(deps)

Update dependencies and devDependencies to newer versions (bump multiple packages)

- Bumped devDependencies: @git.zone/tsbuild 2.6.8 -> 2.7.1, @git.zone/tsrun 1.2.46 -> 1.6.2, @git.zone/tstest 2.3.6 -> 2.7.0
- Bumped runtime dependencies: @push.rocks/smartai 0.5.11 -> 0.8.0, @push.rocks/smartcli 4.0.11 -> 4.0.19, @push.rocks/smartgit 3.2.1 -> 3.3.1, @push.rocks/smartlog 3.1.9 -> 3.1.10, gpt-tokenizer 3.0.1 -> 3.2.0, typedoc 0.28.12 -> 0.28.14, typescript 5.9.2 -> 5.9.3
- No source code changes in this commit; dependency-only updates. Run the test suite and CI to verify compatibility.

2025-11-04 - 1.9.1 - fix(iterative-context-builder)

Rely on DiffProcessor for git diff pre-processing; remove raw char truncation, raise diff token safety, and improve logging

- Removed raw character-based truncation of additionalContext — diffs are expected to be pre-processed by DiffProcessor instead of blind substring truncation.
- Now validates pre-processed diff token count only and treats DiffProcessor as the primary sampler (DiffProcessor typically uses a ~100k token budget).
- Increased MAX_DIFF_TOKENS safety net to 200,000 to cover edge cases and avoid false positives; updated logs to reflect pre-processed diffs.
- Improved error messaging to indicate a likely DiffProcessor misconfiguration when pre-processed diffs exceed the safety limit.
- Updated informational logs to state that a pre-processed git diff was added to context.

2025-11-04 - 1.9.0 - feat(context)

Add intelligent DiffProcessor to summarize and prioritize git diffs and integrate it into the commit context pipeline

- Add DiffProcessor (ts/context/diff-processor.ts) to intelligently process git diffs: include small files fully, summarize medium files (head/tail sampling), and mark very large files as metadata-only to stay within token budgets.
- Integrate DiffProcessor into commit workflow (ts/aidocs_classes/commit.ts): preprocess raw diffs, emit processed diff statistics, and pass a token-efficient diff section into the TaskContextFactory for commit context generation.
- Export DiffProcessor and its types through the context index and types (ts/context/index.ts, ts/context/types.ts) so other context components can reuse it.

- Add comprehensive tests for the DiffProcessor behavior and integration (test/test.diffprocessor.node.ts) covering small/medium/large diffs, added/deleted files, prioritization, token budgets, and formatting for context.
- Minor adjustments across context/task factories and builders to accept and propagate processed diff strings rather than raw diffs, reducing risk of token overflows during iterative context building.

2025-11-04 - 1.8.3 - fix(context)

Prevent enormous git diffs and OOM during context building by adding exclusion patterns, truncation, and diagnostic logging

- Add comprehensive git diff exclusion globs (locks, build artifacts, maps, bundles, IDE folders, logs, caches) when collecting uncommitted diffs to avoid noisy/huge diffs
- Pass glob patterns directly to smartgit.getUncommittedDiff for efficient server-side matching
- Emit diagnostic statistics for diffs (files changed, total characters, estimated tokens, number of exclusion patterns) and warn on unusually large diffs
- Introduce pre-tokenization safety checks in iterative context builder: truncate raw diff text if it exceeds MAX_DIFF_CHARS and throw a clear error if token count still exceeds MAX_DIFF_TOKENS
- Format and log token counts using locale-aware formatting for clarity
- Improve robustness of commit context generation to reduce risk of OOM / model-limit overruns

2025-11-03 - 1.8.0 - feat(context)

Wire OpenAI provider through task context factory and add git-diff support to iterative context builder

- Pass AiDoc.openaiInstance through TaskContextFactory into IterativeContextBuilder to reuse the same OpenAI provider and avoid reinitialization.
- IterativeContextBuilder now accepts an optional OpenAiProvider and an additionalContext string; when provided, git diffs (or other extra context) are prepended to the AI context and token counts are updated.
- createContextForCommit now forwards the git diff into the iterative builder so commit-specific context includes the diff.
- Updated aidocs_classes (commit, description, readme) to supply the existing openaiInstance when creating the TaskContextFactory.

2025-11-03 - 1.7.0 - feat(IterativeContextBuilder)

Add iterative AI-driven context builder and integrate into task factory; add tests and iterative configuration

- Introduce IterativeContextBuilder: iterative, token-aware context construction that asks the AI which files to load and evaluates context sufficiency.
- Switch TaskContextFactory to use IterativeContextBuilder for readme, description and commit tasks (replaces earlier EnhancedContext flow for these tasks).
- Add iterative configuration options (maxIterations, firstPassFileLimit, subsequentPassFileLimit, temperature, model) in types and ConfigManager and merge support for user config.
- Update CLI (tokens and aidoc flows) to use the iterative context factory and improve task handling and messaging.
- Add test coverage: test/test.iterativecontextbuilder.node.ts to validate initialization, iterative builds, token budget respect and multiple task types.
- Enhance ContextCache, LazyFileLoader, ContextAnalyzer and ContextTrimmer to support the iterative pipeline and smarter prioritization/prompts.

2025-11-03 - 1.6.1 - fix(context)

Improve context building, caching and test robustness

- EnhancedContext: refactored smart context building to use the analyzer and TaskContextFactory by default; taskType now defaults to 'description' and task-specific modes are applied.
- ConfigManager: simplified analyzer configuration (removed enabled flag) and fixed getAnalyzerConfig fallback shape.
- ContextCache: more robust mtime handling and persistence; tests updated to use real file mtimes so cache validation works reliably.
- LazyFileLoader: adjusted token estimation tolerance and improved metadata caching behavior.
- ContextAnalyzer & trimming pipeline: improved prioritization and trimming integration to better enforce token budgets.
- Tests: relaxed strict timing/boolean checks and made assertions more tolerant (toEqual vs toBe) to reduce false negatives.

2025-11-02 - 1.6.0 - feat(context)

Introduce smart context system: analyzer, lazy loader, cache and README/docs improvements

- Add ContextAnalyzer for dependency-based file scoring and prioritization (PageRank-like centrality, relevance, efficiency, recency)
- Add LazyFileLoader to scan metadata and load files in parallel with lightweight token estimates
- Add ContextCache for persistent file content/token caching with TTL and max-size eviction
- Enhance ContextTrimmer with tier-based trimming and configurable light/aggressive levels
- Integrate new components into EnhancedContext and TaskContextFactory to build task-aware, token-optimized contexts
- Extend ConfigManager and types to support cache, analyzer, prioritization weights and tier configs (npmextra.json driven)
- Add comprehensive unit tests for ContextAnalyzer, ContextCache and LazyFileLoader
- Update README with Smart Context Building docs, examples, configuration options and CI workflow snippet

2025-09-07 - 1.5.2 - fix(package)

Bump dependencies, refine test script and imports, and overhaul README and docs

- Bumped multiple dependencies and devDependencies (including @git.zone/tspublish, @git.zone/tsbuild, @git.zone/tstest, @push.rocks/npmextra, @push.rocks/qenv, @push.rocks/smartfile, @push.rocks/smartlog, @push.rocks/smartshell, gpt-tokenizer, typedoc, etc.).
- Updated test script to run tstest with verbose, logfile and increased timeout; adjusted testCli script invocation.
- Fixed test import in test/test.aidoc.nonci.ts to use @git.zone/tstest tapbundle.
- Large README rewrite: reorganized and expanded content, added quick start, CLI commands, examples, configuration, troubleshooting and usage sections.
- Minor clarification added to commit prompt in ts/aidocs_classes/commit.ts (text cleanup and guidance).

2025-08-16 - 1.5.1 - fix(aidoc)

Bump dependencies, add pnpm workspace config, and add AiDoc.stop()

- Bumped multiple dependencies and devDependencies in package.json (notable upgrades: @git.zone/tsbuild, @git.zone/tspublish, @push.rocks/npmextra, @push.rocks/qenv, @push.rocks/smartai, @push.rocks/smartfile, @push.rocks/smartgit, @push.rocks/smartlog, @push.rocks/smartpath, @push.rocks/smartshell, typedoc, typescript).
- Added pnpm-workspace.yaml with onlyBuiltDependencies (esbuild, mongodb-memory-server, puppeteer, sharp).
- Added AiDoc.stop() to properly stop the OpenAI provider (resource/client shutdown).
- Updated packageManager field in package.json to a newer pnpm version/hash.

2025-05-14 - 1.5.0 - feat(docs)

Update project metadata and documentation to reflect comprehensive AI-enhanced features and improved installation and usage instructions

- Revised descriptions in package.json and npmextra.json to emphasize comprehensive documentation capabilities
- Expanded README with detailed installation options and extended usage examples for both CLI and API-like integrations
- Added new dependency (gpt-tokenizer) to support token counting for AI context building
- Adjusted keywords to better reflect project functionalities such as commit message automation and context trimming

2025-05-13 - 1.4.5 - fix(dependencies)

Upgrade various dependency versions and update package manager configuration

- Bump @git.zone/tsbuild from ^2.1.80 to ^2.3.2
- Upgrade @push.rocks/tapbundle from ^5.0.23 to ^6.0.3
- Update @types/node from ^22.8.1 to ^22.15.17
- Bump @push.rocks/smartai from ^0.4.2 to ^0.5.4
- Upgrade @push.rocks/smartlog from ^3.0.7 to ^3.0.9
- Update typedoc from ^0.27.9 to ^0.28.4
- Bump typescript from ^5.5.2 to ^5.8.3
- Add packageManager field with pnpm@10.10.0 specification

2025-02-25 - 1.4.4 - fix(dependencies)

Update dependencies to latest versions

- Updated '@push.rocks/smartai' from '^0.0.17' to '^0.4.2'
- Updated 'typedoc' from '^0.26.1' to '^0.27.9'

2025-01-14 - 1.4.3 - fix(aidocs_classes)

Improve readme generation instructions to ensure correct markdown formatting.

- Added guidance to avoid using backticks at the beginning and end of readme generation to prevent markdown issues.
- Clarified that the output is directly written to readme.md and backticks should only be used for code blocks.

2024-10-28 - 1.4.2 - fix(cli)

Ensure async completion for aidoc readme and description generation

- Added await statements for asynchronous methods buildReadme and buildDescription in the aidoc command.

2024-10-28 - 1.4.1 - fix(readme)

Correct async call to getModuleSubDirs in readme generation.

- Fixed an issue with asynchronous handling in readme generation for submodules.
- Ensured that getModuleSubDirs function is called with await to handle promises properly.

2024-10-28 - 1.4.0 - feat(aidocs)

Added support for building readmes for sub-modules in aidocs

- Updated the `Readme` class to handle monorepo projects by generating readmes for sub-modules.
- Integrated `tspublish` to identify sub-modules for readme generation.

2024-06-24 - 1.3.12 - fix(aidocs)

Fix changelog generation by handling leading newlines

- Fixed handling of leading newlines in the changelog to ensure proper formatting.

2024-06-23 - 1.3.11 - fix(core)

Fixed new changelog formatting issue to retain consistent spacing.

- Adjusted the new changelog generation to ensure consistent spacing for improved readability.

2024-06-23 - 1.3.10 - fix(aidocs_classes)

Fix changelog format to remove extra newline

- Updated `ts/aidocs_classes/commit.ts` to fix the changelog format.

2024-06-23 - 1.3.9 - fix(aidoc)

Fix changelog generation by properly stripping markdown code fences

- Corrected the changelog generation code to ensure markdown code fences are properly stripped.

2024-06-23 - 1.3.8 - fix(changelog)

Fix changelog generation by properly stripping markdown code fences

- Corrected the changelog generation code to ensure markdown code fences are properly stripped.

2024-06-23 - 1.3.7 - fix(aidoc)

Update to include package-lock.json in uncommitted changes check

- Modified the getUncommittedDiff method call in commit.ts to include package-lock.json along with pnpm-lock.yaml

2024-06-23 - 1.3.6 - fix(commit)

Fixed issue with retrieving uncommitted diffs in git repository

- Revised logic to correctly handle uncommitted changes by using an array for `getUncommittedDiff` method
- Ensured proper handling and representation of uncommitted changes in the output

2024-06-23 - 1.3.5 - fix(aidocs_classes)

Refactor and enhance changelog formatting

- Updated the `commit.ts` file to improve the changelog formatting and ensure consistency.
- Enhanced the changelog instructions to include summarizing messages for omitted commits.
- Removed unnecessary console logging in `projectcontext.ts`.

```
## 2024-06-23 - 1.3.3 - fix(aidocs_classes)
Fix changelog formatting issue in commit class

## 2024-06-23 - 1.3.2 - fix(aidocs_classes)
Fix minor bugs and update dependencies in aidocs_classes

## 2024-06-23 - 1.3.1 - fix(aidocs_classes)
Fix typo in INextCommitObject interface and update date format in changelog generation.

## 2024-06-23 - 1.3.0 - fix(aidocs_classes)
Fix typo in INextCommitObject interface

## 2024-06-23 - 1.2.4 - feat(core)
Added smarttime dependency and improved changelog generation

## 2024-06-23 - 1.2.3 - fix(logging)
Refactor logger initialization to use commitinfo data

## 2024-06-23 - 1.2.2 - fix(aidocs)
Fix bug in AiDoc class causing undefined token handling

## 2024-06-23 - 1.2.0 - fix(core)
Fixed usage of plugins in project context and readme generation

## 2024-06-23 - 1.1.42 - feat(aidocs_classes)
Enhance changelog generation by supporting complete generation in the absence of previous changelog files

## 2024-06-23 - 1.1.41 - fix(aidocs_classes)
Improve commit message generation by handling empty diffs and updating changelog instructions
```