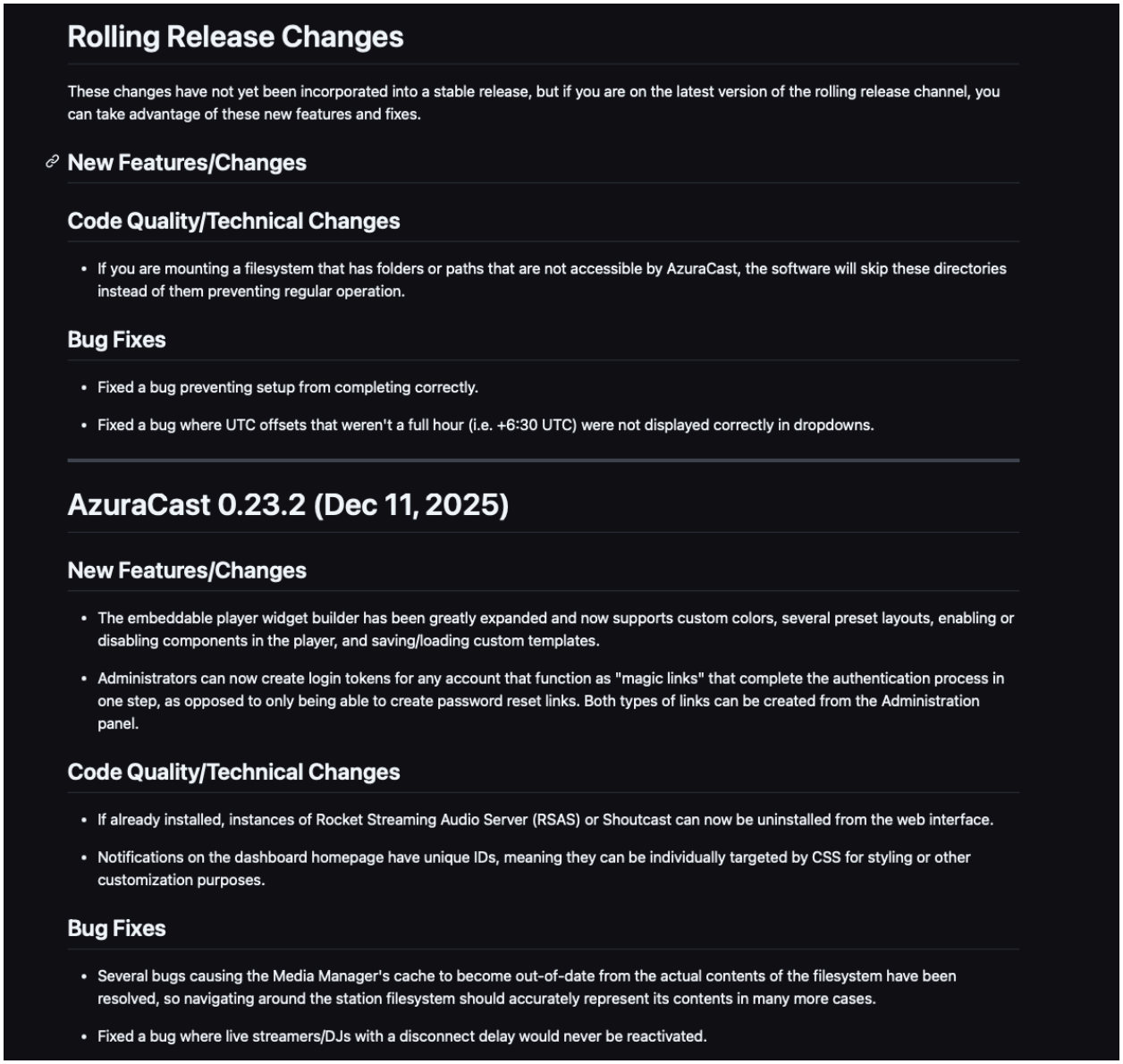
**<u>Professional Writing Genre #2,</u> (due by classtime Wed, Jan. 28):**

**Genre Name:** Change Logs & Revision Notes

**General Analysis: What are the characteristics of the genre? What are its purposes? Who is the audience? Who are the authors? What is the writing style?**

A change log or revision note is a common type of chronological, structured list of updates made to a project. The primary authors are developers, typically the person who made the changes. Change logs are intended to make it easier for users or developers to see the notable changes made between different versions of a project. They are often short, primarily use bullet points, and concise allowing readers to scan through it quickly. Change logs are primarily for developers or contributors but secondary audiences can also be anyone.

**Find a Primary Source:** Copy and paste an example of the genre (or include a link to it). Respond to the analytical questions above. What are the characteristics of the genre? What are its purposes? Who is the audience? Who are the authors? What is the writing style? Speak generally about the genre, but also explain a bit about the primary source itself (i.e. the author, their specific purpose, where they work, etc.)



## Rolling Release Changes

These changes have not yet been incorporated into a stable release, but if you are on the latest version of the rolling release channel, you can take advantage of these new features and fixes.

### New Features/Changes

### Code Quality/Technical Changes

- If you are mounting a filesystem that has folders or paths that are not accessible by AzuraCast, the software will skip these directories instead of them preventing regular operation.

### Bug Fixes

- Fixed a bug preventing setup from completing correctly.
- Fixed a bug where UTC offsets that weren't a full hour (i.e. +6:30 UTC) were not displayed correctly in dropdowns.

---

## AzuraCast 0.23.2 (Dec 11, 2025)

### New Features/Changes

- The embeddable player widget builder has been greatly expanded and now supports custom colors, several preset layouts, enabling or disabling components in the player, and saving/loading custom templates.
- Administrators can now create login tokens for any account that function as "magic links" that complete the authentication process in one step, as opposed to only being able to create password reset links. Both types of links can be created from the Administration panel.

### Code Quality/Technical Changes

- If already installed, instances of Rocket Streaming Audio Server (RSAS) or Shoutcast can now be uninstalled from the web interface.
- Notifications on the dashboard homepage have unique IDs, meaning they can be individually targeted by CSS for styling or other customization purposes.

### Bug Fixes

- Several bugs causing the Media Manager's cache to become out-of-date from the actual contents of the filesystem have been resolved, so navigating around the station filesystem should accurately represent its contents in many more cases.
- Fixed a bug where live streamers/DJs with a disconnect delay would never be reactivated.

https://github.com/AzuraCast/AzuraCast/blob/main/CHANGELOG.md

The following is a snippet from the AzuraCast change log hosted on github. AzuraCast is a free and open source, self hosted web radio station software used by people who want to broadcast their music across the internet. This change log, as the name implies, outlines the changes made to the newest version of the application. It uses a structured list where each change is split into its various types like new features, bug fixes, security updates, deprecated(no longer supported features), or general changes to existing features. The change log uses no paragraph style writing primarily using bullet points instead. Each bullet point is completely independent from each other highlighting a unique change made in the project. The bullet points themselves are short, and concise leaving little room for ambiguity. The software itself is targeted towards people who have

the technical skills to set up AzuraCast. As such, they make use of technical terminology. However, authors don't expect readers to understand every change made to the project. Similar to documentation readers often look for information that they need rather than reading it in its entirety. Since this is an open source project these change logs have many authors from the community of contributors that manage it.

**Secondary Source** (for example, a person or industry expert talking ABOUT what the genre is & how people in the industry use it): Who is the person? What does the person say? Respond to the analytical questions above based on what the person says. **Summarize** their perspective here and include a link to the source. Include a link to the source.

https://keepachangelog.com/en/1.1.0/

Change logs aren't standardized across the industry however good change logs often follow core principles outlined in this article. The article is written and maintained by Oliver Lacan whose career is centered around technology education. They pioneered the shields metadata badge project with over 1 million GitHub repositories using them in their descriptions. They have had a profound impact on the way developers manage descriptions, documentation, and change logs. Similar to the core principles of workplace writing, the focal point is on the reader: “change logs are for humans, not machines.” This article outlines common strategies to reduce the burden of writing change by simply keeping an unreleased section of changes, serving two purposes. One so that people can see upcoming plans for a new project release and two when its time to release a new version you can simply move the written changes into the change log. The key is to keep dates consistent, acknowledging deprecations, and removing any unnecessary comments or information. The goal for change logs is not to be detail oriented but a quick high level overview of changes.

**Tertiary Source** (for example, a reference guide that explains the genre, including an encyclopedia, Credo, a credible industry database, Claude, Chat GPT and/or Perplexity. You can also use these platforms to find primary and secondary sources that are credible, authentic, and authoritative.) What did you discover? What is the platform and are they credible? What were the prompt/s you used to find this information? And answer the analytical questions above: what did you learn about the characteristics of the genre? What are its purposes? Who is the audience? Who are the authors? What is the writing style? Include a link to what it generated & your prompt/s.

**https://en.wikipedia.org/wiki/Changelog**

Wikipedia is a widely known tertiary source that summarizes information from primary and secondary sources. It's often useful for gaining an overview of a subject or sourcing more credible sources. Due to wikipedia being an interpreted summary of sources and

its open wiki the quality and accuracy of a wikipedia article can vary. The author could be anyone regardless of their background. However, they do have standards and often contain dozens of primary and secondary sources. Through wikipedia is how I discovered a secondary source which I had also seen referenced in many developer forums.

The wikipedia article itself covers the basics of what a change log is, as “primarily a change made to software”. They acknowledge the broad readership for change logs where they can be written in a more technical manner which is harder for the average reader to understand or simplify. They cover common naming conventions for changing log files, and typical formats that are commonly used.

**NOW IT’S YOUR TURN**

**How would you use this genre? Who is your audience? What would you say? What is your purpose?**

Since I will need to work on a team it's important that any changes I make towards a project are communicated to the rest of my team. Since hardware projects often have many dependencies, small changes could impact the development of another person's work. My audience will primarily be for my development team, and if the change logs are made public they would additionally be for the general user. The change log would use primarily bullet points and be short and concise.

# CHANGELOG

## [v1.1.0] - 2026-02-08

### Added

- Introduced configurable softening parameters epsilon (ε) in simulation constants.
- Added runtime parameter validation for ε > 0.
- Added runtime parameter for Barnes-Hut approximation theta (θ).
- Added tangential friction to colliding/touching objects.
- Added OpenGL based anim library to visualize simulation resolving I/O bottleneck.

### Changed

- Modified gravitational force equation from classical newtonian form to softened form.
- Migrated from direct-sum algorithm to Barnes-Hut approximation algorithm improving performance for high N-body simulations
- Migrated from SLOP/ impulse based collision handling to spring like collision improving stability.

### Fixed

- Resolved segmentation faults that occur when objects leave simulation bounds.
- Resolved energy pump bug that gave colliding objects tangential energy.
- Resolved explosive squeeze effect that occurs when an object is crushed from both ends.
- Resolved memory leak issues from local heap arrays.
- Resolved forced serialization from I/O operations when the program is running over multiple threads.

### Deprecated

- Removed SLOP/Impulse collision handling.
- Removed I/O functionality.

### Performance Notes

- Minor increase in arithmetic from additional softening terms.
- Improved long-term energy stability for dense initial conditions.
- Increased overhead for Barnes Hut Tree building
- Significant improvement in anim performance.

### Notes for Developers

- Recommended default for epsilon is 0.01. Larger values will result in over softening.

- Recommended default for theta is 0.05. Larger values are less accurate. 0 will result in a direct sum (slowest but most accurate)
- Recommend that epsilon scale with particle mass.
- Since visualization is now handled within the library itself it must be compiled alongside the program. Dependencies differ between Mac, Linux, and Windows computers please check Make file for more information.

## Backwards compatibility

- Any program that made use of earlier versions of the library will still compile without changes. However, unless specified, it will use default parameters for epsilon and theta. May result in different outcomes.

## Known Issues

- The new openGL anim library has frame pacing issues with n-body simulation fewer than a 100.