Version Information

Current Version

CineFlux-AutoXML v1.0.0

Released: May 18, 2025

CineFlux-AutoXML has reached its first production-ready release. This version includes all core functionality and is stable for production use.

Key Features in v1.0.0

- Complete TypeScript type system implementation
- Advanced WebAssembly loading and caching optimizations
- Browser compatibility improvements with fallbacks
- Performance monitoring and optimization utilities
- Comprehensive documentation for deployment and usage

Versioning Scheme

CineFlux-AutoXML follows Semantic Versioning 2.0.0 for version numbering.

The version number format is: **MAJOR.MINOR.PATCH**

- **MAJOR** version increments indicate incompatible API changes or significant architectural changes
- **MINOR** version increments indicate new functionality added in a backward-compatible manner
- **PATCH** version increments indicate backward-compatible bug fixes

Pre-release Versions

Pre-release versions may include additional labels:

- **alpha**: Early development versions with potentially unstable features
- **beta**: Feature-complete versions undergoing testing
- **rc**: Release candidates ready for final testing

Example: `1.0.0-beta.1`

Version History

v1.0.0 (2025-05-18)

- First production-ready release
- Complete TypeScript type system implementation
- Advanced WebAssembly loading and caching optimizations
- Browser compatibility improvements with fallbacks
- Performance monitoring and optimization utilities

- Comprehensive documentation for deployment and usage

v0.1.0 (2025-05-17)

- Initial release with core functionality
- Audio and video analysis features
- Beat detection and synchronization
- Scene detection and content analysis
- XML export for professional editing software
- Real-time preview of edits

Version Bumping Guidelines

When to Bump MAJOR Version

Increment the MAJOR version when:

- Making incompatible API changes
- Changing the core architecture significantly
- Removing deprecated features
- Changing the user interface dramatically
- Making changes that require users to modify their projects

Example: Changing from v1.2.3 to v2.0.0

When to Bump MINOR Version

Increment the MINOR version when:

- Adding new features in a backward-compatible manner
- Deprecating functionality (but not removing it)
- Adding significant new capabilities
- Making substantial improvements to existing features
- Adding new export formats or analysis algorithms

Example: Changing from v1.2.3 to v1.3.0

When to Bump PATCH Version

Increment the PATCH version when:

- Fixing bugs without changing the API
- Making performance improvements
- Updating dependencies
- Fixing security vulnerabilities
- Making documentation changes
- Refactoring code without changing functionality

Example: Changing from v1.2.3 to v1.2.4

Version Update Process

1. Determine the appropriate version increment based on the changes
2. Update the version number in:
   - `package.json`
   - `VERSION.md` (this file)
   - Any other version-specific files
3. Update the `CHANGELOG.md` with details of the changes
4. Commit the changes with a message like “Bump version to x.y.z”
5. Tag the commit with the version: `git tag vx.y.z`
6. Push the changes and tags to the repository

Version Display

The current version is displayed in:

- The application’s About screen
- The console on application startup
- The exported XML files as metadata
- API responses (if applicable)

Compatibility Information

Backward Compatibility

- **v0.x.x**: Initial development phase, no backward compatibility guarantees
- **v1.x.x**: Stable API, backward compatibility within the same major version
- **v2.x.x+**: Future major versions with potential breaking changes

Forward Compatibility

Projects created with an older version of CineFlux-AutoXML should be openable in newer versions within the same major version number. However, projects saved in newer versions may not be compatible with older versions of the software.