Changelog

All notable changes to MD2PDF will be documented in this file.

The format is based on Keep a Changelog,
and this project adheres to Semantic Versioning.

[1.1.0] - 2025-09-06

Added

- Dynamic CSS file loading for themes and styles - no more hardcoded lists
- Programmatic header file assignment via `--header` option accepts both files and directories
- Support for custom header markdown files and header directories

Changed

- Styles and themes are now dynamically discovered from CSS files
- CLI no longer uses hardcoded choice lists for styles and themes
- Improved error messages showing available styles/themes when invalid option is provided

Fixed

- Fixed inconsistency between `list-styles` command output and accepted theme values
- “agile” theme and other dynamically added themes now work correctly
- CLI help text now accurately reflects dynamic discovery behavior

[1.0.0] - 2025-09-04

Added

- Initial public release of MD2PDF
- Professional PDF conversion from Markdown with 7 style templates:
- Technical documentation style

- Academic paper style
- Story/narrative style
- Modern minimalist style
- Whitepaper style
- Business consultancy style
- Futuristic design style
- 10 color themes for each style:
- Default (light)
- Dark mode
- Forest (green tones)
- Oceanic (blue tones)
- Solarized
- Violet
- Warm (orange/red)
- Cool (blue/gray)
- Elegant (grayscale)
- Sepia (vintage)
- Modern CLI interface with Click and Rich libraries
- Batch processing capabilities
- Word document export support
- Smart font management with Google Fonts CDN fallbacks
- Emoji support with optional Twemoji download
- Syntax highlighting for code blocks
- Table of contents generation
- Header/footer customization
- Responsive layout optimizations

Technical Features

- Python 3.8+ support
- Modern packaging with pyproject.toml
- Comprehensive test suite
- Type hints throughout
- MIT License with full attribution
- Optimized package size (<20MB core)
- Cross-platform compatibility (Windows, macOS, Linux)

Dependencies

- WeasyPrint for PDF generation
- python-docx for Word document support
- Markdown parser with extensions
- Pygments for syntax highlighting
- BeautifulSoup4 for HTML processing
- Click for CLI
- Rich for beautiful terminal output
- Pydantic for configuration validation

Documentation

- Complete API documentation
- Usage examples and tutorials
- Contributing guidelines
- Installation instructions

[Unreleased]

Planned

- Custom CSS style creation wizard
- PDF merge and split utilities
- Markdown linting integration
- Extended font pack as optional package
- Performance optimizations for large documents
- Plugin system for custom processors
- Web UI interface
- Cloud conversion API