2PDF: Markdown to PDF Converter

Converts Markdown and HTML files to PDFs with token replacement, styling, and content processing features.

What it does

- Converts Markdown and HTML to PDF using Chrome/Puppeteer
- Replaces tokens like `{{DATE}}` and `{{DEVELOPER_NAME}}`
- Applies Inter font styling and consistent layout
- Handles PDF-only content sections
- Supports page breaks and special formatting
- Works on macOS, Linux, and Windows

Quick start

```
# Install dependencies
npm install

# Convert files
node bin/2pdf.js README.md          # converts README.md to README.pdf
node bin/2pdf.js document.md        # converts any markdown file
node bin/2pdf.js webpage.html       # converts HTML files too
node bin/2pdf.js file.md —o out.pdf  # custom output name
```

File types supported

Markdown files (.md)

- Always uses 2pdf's Inter font styling and layout
- Supports all markdown features (headers, lists, code blocks, etc.)
- Token replacement and special content features

HTML files (.html, .htm)

- **With existing CSS**: Preserves your styles (internal, external, inline)
- **Without CSS**: Falls back to 2pdf's default styling
- Token replacement works the same way
- Color theming injects CSS custom properties

Core features

Token replacement

Insert dynamic content using `{{TOKEN_NAME}}` placeholders:

```
# {{PROJECT_NAME}} Documentation
**Version:** {{PROJECT_VERSION}}
**Generated:** {{DATE}} by {{DEVELOPER_NAME}}
```

Automatic tokens (always available)
- `{{DATE}}` - Current date (2024-01-15)
- `{{DATE_LONG}}` - Current date (January 15, 2024)
- `{{TIME}}` - Current time (14:30)
- `{{YEAR}}`, `{{MONTH}}`, `{{DAY}}` - Date components
- `{{TIMEZONE}}` - Current timezone (PST, EST, UTC, etc.)
- `{{USERNAME}}`, `{{HOSTNAME}}` - System info
- `{{PROJECT_NAME}}` - Auto-generated from directory name

Custom tokens from config file

Create `2pdf.config` with your own tokens:

```
DEVELOPER_NAME=John Doe
PROJECT_VERSION=2.1.0
COMPANY_NAME=Acme Corp
```

Special content features

1. PDF-only content

Content that only appears in the PDF:

```
<!-- PDF ONLY This text only appears in the PDF version -->
```

2. Page breaks

Force a new page:

```
Content here...
<!-- PAGE-BREAK -->
Content on next page...
```

3. Live site shields

Style a paragraph as a badge:

```
<!-- live-site-shield -->
Live Site: https://example.com
```

🌐 Live Site   **https://example.com**

4. Table column widths

Control table column widths with HTML comments:

Example 1

Name	Description
Item	Long description text that needs more space

```
<!-- col-widths: 30% 70% -->
| Name | Description |
|------|-------------|
| Item | Long description text that needs more space |
```

Example 2

```
<!-- col-widths: 100px 200px 50% -->
| ID | Name | Notes |
|----|------|-------|
| 1  | Test | Some notes |
```

ID	Name	Notes
1	Test	Some notes

Document styling

Set theme colors, text colors, and font sizes directly in your document:

```
<!-- theme-color: #ff6b35 -->
<!-- body-color: #2c3e50 -->
<!-- link-color: #3498db -->
<!-- link-underline: off -->
<!-- font-size: 1.2em -->
<!-- header-size: 1.3em -->
<!-- body-size: 0.9em -->
<!-- page-numbers: on -->

# Your Document Title
Content with custom styling...
```

Color options:

- **theme-color**: Headers, blockquotes, inline code (`#ff6b35` , `#1434cb` , `#27ae60`)
- **body-color**: All body text, paragraphs, lists (`#2c3e50` , `#333333`)
- **link-color**: Hyperlinks (`#3498db` , `#e74c3c`)

Text decoration:

- **link-underline**: `on` or `off` - controls link underlines

Font sizing:

- **font-size**: Root font size (`1.2em` , `14px` , `16px`)
- **header-size**: All headers scale from this (`1.3em` , `1.1em`)
- **body-size**: Paragraphs, lists, tables (`0.9em` , `1.1em`)

Page features:

- **page-numbers**: `on` , `off` , `X` , or `X of Y` - controls footer page numbering
- **disclosure**: Brief text for center footer (e.g., `Internal Use Only` , `CONFIDENTIAL`)

Output features:

- **sequential-output**: `on` or `off` - enables versioned output filenames
- **version-number**: `XX.YY` format - auto-increments with each PDF generation

Footer Layout

When page numbers are enabled, the footer has three sections:

Left: Document title (from H1) | Date (DD MMM YYYY format)
Center: Disclosure text (uppercase, optional)
Right: Page numbers (X or X of Y format)

```
<!-- page-numbers: X -->
<!-- disclosure: Internal Use Only -->
```

```
# My Document Title
```

Results in footer: `My Document Title | 25 Dec 2024` | `INTERNAL USE ONLY` | `1`

Sequential Output

Automatically version your PDF outputs with incremental version numbers:

```
<!-- sequential-output: on -->
<!-- version-number: 00.00 -->

# My Document
```

How it works:

1. **First run**: Outputs `my-document-v00.00.pdf`, updates version to `00.01`
2. **Second run**: Outputs `my-document-v00.01.pdf`, updates version to `00.02`
3. **Continues**: `00.02` → `00.03` → ... → `00.99` → `01.00` → `01.01`

Version format: `XX.YY` (major.minor, 00-99 each)
Auto-increment: Source file is automatically updated with new version
Filename: `document-vXX.YY.pdf` format

Advanced options

Color theming

Customize header and accent colors with the `--color` flag:

```
# Predefined colors (from config file)
node bin/2pdf.js doc.md --color blue
node bin/2pdf.js doc.md --color corporate

# Custom hex colors
node bin/2pdf.js doc.md --color 1434cb
node bin/2pdf.js doc.md --color "#ff6b35"
```

Add predefined colors to your `2pdf.config`:

```
COLOR_BLUE=#1434cb
COLOR_RED=#e74c3c
COLOR_GREEN=#27ae60
COLOR_CORPORATE=#2c3e50
COLOR_ORANGE=#ff6b35
```

Single-page mode

Generate PDFs as one continuous page instead of multiple US Letter pages:

```
node bin/2pdf.js file.md --single-page
node bin/2pdf.js file.md --single-page --color blue
```

This is perfect for:

- Web page captures
- Long documents that look better as continuous scrolls
- Documents with content that shouldn't be split across pages

Other options

```
node bin/2pdf.js file.md -s custom.css   # custom CSS file
node bin/2pdf.js file.md --debug          # keep temp files
node bin/2pdf.js file.md --verbose        # detailed output
```

Quick reference

Special comments for enhanced formatting:

```
<!-- PDF ONLY Content only in PDF -->
<!-- PAGE-BREAK -->
<!-- live-site-shield -->
<!-- col-widths: 30% 70% -->
<!-- theme-color: #ff6b35 -->
<!-- body-color: #2c3e50 -->
<!-- link-color: #3498db -->
<!-- link-underline: off -->
<!-- font-size: 1.2em -->
<!-- header-size: 1.3em -->
<!-- body-size: 0.9em -->
<!-- page-numbers: X of Y -->
<!-- disclosure: Internal Use Only -->
<!-- sequential-output: on -->
<!-- version-number: 00.00 -->
```

Configuration

Config file format

Create `2pdf.config` with key=value pairs:

```
# Project info
PROJECT_VERSION=2.1.0
DEVELOPER_NAME=Jane Smith
COMPANY_NAME=Acme Corp

# Color themes
COLOR_BRAND=#1434cb
COLOR_ACCENT=#ff6b35

# Custom content
FOOTER_TEXT=Generated on {{DATE}} by {{DEVELOPER_NAME}}
```

Config file locations

2pdf searches for `2pdf.config` in this order:

1. Same directory as the input file
2. Current working directory
3. 2pdf installation directory
4. `config/` subdirectory in any of the above

Token processing details

- **Multiple passes**: Tokens can reference other tokens
- **Case sensitive**: `{{NAME}}` ≠ `{{name}}`
- **No whitespace**: Use `{{TOKEN}}`, not `{{ TOKEN }}`
- **Nested replacement**: `GREETING=Hello {{USERNAME}}` works
- **Missing tokens**: Left unchanged for debugging

Example workflow

```
# 1. Create config file
echo "PROJECT_VERSION=2.1.0" > 2pdf.config
echo "DEVELOPER_NAME=Jane Smith" >> 2pdf.config

# 2. Use in markdown
echo "# Project v{{PROJECT_VERSION}} by {{DEVELOPER_NAME}}" > doc.md

# 3. Convert
node bin/2pdf.js doc.md

# Result: "Project v2.1.0 by Jane Smith" in the PDF
```

License

© 2025 Feral Creative • **MIT License**