

## Markdown for Both Kramdown and XeLaTeX PDF

To generate PDFs for my Jekyll blog using Markdown, I use the following Pandoc command:

```
command = [  
    'pandoc',  
    input_markdown_path,  
    '-o', output_pdf_path,  
    '-f', 'markdown',  
    '--pdf-engine', 'xelatex',  
    '--resource-path=.:assets',  
    '-V', f'CJKmainfont={CJK_FONT}',  
    '-V', f'CJKsansfont={CJK_FONT}',  
    '-V', f'CJKmonofont={CJK_FONT}',  
    '-V', f'geometry:{GEOMETRY}',  
    '-V', 'classoption=16pt',  
    '-V', 'CJKoptions=Scale=1.1',  
    '-V', 'linestretch=1.5'  
]
```

Supporting Both Kramdown and XeLaTeX

When writing Markdown that needs to work with kramdown (for Jekyll HTML output) and XeLaTeX (for PDF output via Pandoc), there are a few considerations:

1. Image Path Compatibility
  - Kramdown (HTML): Prefers paths starting with / for referencing assets.
  - XeLaTeX (PDF): Prefers relative paths without a leading /.

Solution: Use relative paths that work for both:

```

```

2. Handling kramdown Attributes
  - {:.responsive} is specific to kramdown for styling HTML output.
  - XeLaTeX does not support these attributes and will throw an error.

Solution: Remove kramdown-specific attributes in Markdown intended for PDF generation. For example:

```
{: .responsive }
```

```

```

If {:.responsive} is critical for your Jekyll HTML layout, consider adding it selectively for web output while omitting it in the PDF generation process.

Workflow for Dual Compatibility 1. Write Markdown content with minimal dependencies on kramdown-specific features. 2. For advanced styling in HTML, apply CSS classes directly in your Jekyll templates rather than inline in Markdown. 3. Use Pandoc options to control PDF formatting while maintaining Markdown portability.

By following these practices, Markdown content remains compatible across Jekyll HTML rendering and XeLaTeX PDF generation, ensuring a seamless workflow for multi-format publishing.