Glow - Terminal Markdown Renderer

Overview

Glow is a terminal-based Markdown reader and renderer. It can be used to display Markdown reports directly in a command prompt or PowerShell session with formatted headings, tables, links, code blocks and terminal colour styling.

Glow is useful when working with TapeTrack command-line utilities that produce Markdown output, as it allows the generated report to be reviewed directly from the command line without opening a separate Markdown editor.

Glow can also browse Markdown files in a directory and provides a terminal user interface (TUI) for reading documents.

Installation

Windows

Glow can be installed on Windows using Windows Package Manager (WinGet).

Open PowerShell and run:

```
winget install charmbracelet.glow
```

The official Glow documentation lists WinGet as a supported Windows installation method. Glow documentation

After installation, close and reopen the PowerShell window. This is required if the installer has modified the PATH environment variable.

Verify the installation:

```
glow --version
```

Example

A successful installation will report the installed Glow version.

```
Glow v3.0.0
```

The exact version may change as newer releases become available.

Basic Use

Change to the directory containing the Markdown report:

```
cd "C:GazillaByte\Markdown"
```

Display a Markdown report:

```
glow .\out.md
```

A second report can be displayed in the same way:

```
glow .\out2.md
```

Glow does not modify the Markdown file. It reads the file and renders it in the terminal.

Rendering Styles

Glow supports terminal rendering styles. The standard styles include `dark` and `light`.

To explicitly use the dark style:

```
glow -s dark .\out.md
```

To use the light style:

```
glow -s light .\out.md
```

When no style is specified, Glow attempts to determine whether the terminal is using a light or dark background and selects an appropriate style automatically. Glow documentation

The style affects the appearance of the rendered Markdown, including terminal colours and formatting.

Word Wrapping

Reports containing wide tables may benefit from specifying the terminal width.

For example:

```
glow -s dark -w 120 .\out.md
```

The `-w` option specifies the maximum width used for wrapping rendered output.

Interactive Mode

Running Glow without a file argument starts its terminal user interface:

```
glow
```

Glow searches the current directory and subdirectories for Markdown files and allows them to be browsed from the terminal.

Paging

Glow can use a pager when displaying Markdown:

```
glow -p .\out.md
```

This is useful for reports that are longer than the visible terminal window.

Reading Markdown from Standard Input

Glow can also receive Markdown through standard input.

For example:

```
Get-Content .\out.md | glow -
```

The `-` tells Glow to read the Markdown from standard input.

This makes it possible to place Glow after another command in a command-line processing workflow.

Configuration

If the same Glow options are used regularly, Glow provides a configuration facility:

```
glow config
```

The configuration can be used to set options such as:

* Rendering style * Word-wrap width * Pager usage * Mouse support in TUI mode * Line numbers in TUI mode

See the Glow documentation for the current configuration options.

Testing Glow with a TapeTrack Report

The following procedure can be used to verify that Glow is correctly installed and can render a TapeTrack Markdown report.

First confirm the report exists:

```
dir .\*.md
```

Then render it:

```
glow -s dark .\out.md
```

A Markdown table produced by a TapeTrack utility should be rendered as a formatted terminal table rather than displayed as raw Markdown syntax.

Glow and Report Comparison

Glow is a Markdown renderer, not a file comparison utility.

For example, if the following files exist:

```
out.md
out2.md
```

Glow can render each report:

```
glow -s dark .\out.md
glow -s dark .\out2.md
```

A separate comparison tool should be used to identify differences.

For a simple PowerShell comparison:

```
Compare-Object (Get-Content .\out.md) (Get-Content .\out2.md)
```

Highlighting Changed Lines

PowerShell can be used to display `out.md` while colouring lines that differ from the corresponding line in `out2.md`.

The following example colours changed lines red:

```
$a = Get-Content .\out.md
$b = Get-Content .\out2.md
```

```
$max = [Math]::Max($a.Count, $b.Count)

for ($i = 0; $i -lt $max; $i++) {
$lineA = if ($i -lt $a.Count) { $a[$i] } else { "" }
$lineB = if ($i -lt $b.Count) { $b[$i] } else { "" }

```
if ($lineA -ne $lineB) {
    Write-Host $lineA -ForegroundColor Red
}
else {
    Write-Host $lineA
}
```


}
```

This is a PowerShell comparison and highlighting technique. The colour is being applied by PowerShell, not by Glow.

For reports where rows remain in the same order, this provides a simple way to identify changed report lines.

For production comparison of inventory reports, comparing records by Barcode rather than simply comparing line positions is preferable if rows may be added, removed or reordered.

Example Environment

An example working directory may contain:

```
Test Markdown\
    out.md
    out2.md
    err.txt
    runMarkdown.bat
    Report-2026-09-18-10-29-08.pdf
```

The Markdown reports can then be viewed with:

```
glow -s dark .\out.md
glow -s dark .\out2.md
```

References

* Glow * Glow CLI * Glow Styles

markdown, glow

From:
https://rtfm.tapetrack.com/ - **TapeTrack Documentation**

Permanent link:
https://rtfm.tapetrack.com/technote/glow?rev=1789694279

Last update: **2026/09/18 01:17**