TapeTrack Command Line Utilities - Markdown Report Output

Overview

TapeTrack provides command-line utilities that can be used for reporting and task automation. The TapeTrack documentation identifies these utilities as a separate component of the TapeTrack system and provides a Command Line Utility reference. TapeTrack Documentation

TapeTrack utility output can be formatted as Markdown by setting the `TMSSREPORTFORMAT` environment variable to `MARKDOWN`.

This can be useful when the output is intended to be:

* Saved as a Markdown file * Viewed with a Markdown renderer such as Glow * Included in documentation * Compared with another generated report * Processed as part of an automated reporting workflow

Setting the Report Format

Set the following environment variable before running the TapeTrack utility:

```
set TMSSREPORTFORMAT=MARKDOWN
```

The variable instructs the TapeTrack utility to produce Markdown-formatted report output.

Windows Command Prompt

For a temporary setting within the current command session:

```
set TMSSREPORTFORMAT=MARKDOWN
```

The setting applies to commands run from that command session.

Batch File

When using a Windows batch file, the following form can be used:

```
@echo off
set "TMSSREPORTFORMAT=MARKDOWN"

TMSS10Inventory
```

Using the quoted `set` syntax prevents accidental trailing spaces from becoming part of the environment variable value.

Writing Utility Output to a Markdown File

Standard output can be redirected to a Markdown file.

For example:

```
TMSS10Inventory > out.md
```

The result is a Markdown report saved as:

```
out.md
```

The original command-line output is therefore retained as a file that can be viewed or processed independently.

Capturing Standard Error

Standard error can be redirected separately from the report output.

For example:

```
TMSS10Inventory > out.md 2> err.txt
```

This creates:

```
out.md
err.txt
```

The Markdown report is written to `out.md`, while messages written to standard error are written to `err.txt`.

This separation is useful when validating automated report generation because the report itself can remain clean Markdown while diagnostic or error output is retained separately.

Complete Example

A simple batch file can therefore be:

```
@echo off
set "TMSSREPORTFORMAT=MARKDOWN"

TMSS10Inventory > out.md 2> err.txt
```

The processing flow is:

```
TMSS10Inventory
       |
       +---- standard output ----> out.md
       |
       +---- standard error -----> err.txt
```

Generating Multiple Reports

When two reports are generated at different times, they can be saved separately.

For example:

```
@echo off
set "TMSSREPORTFORMAT=MARKDOWN"

TMSS10Inventory > out.md 2> err.txt
```

A subsequent execution can save the second report as:

```
@echo off
set "TMSSREPORTFORMAT=MARKDOWN"

TMSS10Inventory > out2.md 2> err2.txt
```

The directory may then contain:

```
out.md
out2.md
err.txt
err2.txt
```

Viewing the Markdown Report

Once the report has been generated, it can be displayed using Glow.

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

A specific Glow style can be selected:

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

For a wide TapeTrack inventory table:

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

The second report can be viewed using:

```
glow -s dark -w 120 .\out2.md
```

Glow provides terminal-based Markdown rendering and colour styling. It does not alter the generated Markdown file.

Example TapeTrack Inventory Report

A Markdown inventory report generated from `TMSS10Inventory` can contain a table such as:

```
| Seq. | Barcode | Location | Repository | Expiry | Next Move | Last Move |
GDup | CDup | Container |
| ---: | --- | --- | --- | --- | --- | --- | ---: | ---: | --- |
| 1 | US02.LTO.0001 | Red 11 [1.12] | Offsite Vault [12] | Permanent |
Permanent | 2026-09-08 13:14:04 | 6 | 1 | |
```

Glow renders the Markdown table in a terminal-friendly format.

Comparing Two Generated Reports

When two Markdown reports have been generated, PowerShell can be used to compare their contents.

A basic line comparison is:

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

For reports where records remain in the same order, changed lines can be highlighted while displaying the first report:

```
$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 example displays the contents of `out.md` and colours lines red when the corresponding line in `out2.md` is different.

This comparison method is intended for reports where the line order is consistent. For inventory reports where records can move position, a comparison based on a unique field such as Barcode is more robust.

## Example Workflow

A practical workflow for generating, rendering and comparing TapeTrack Markdown reports is:

```
1. Set TMSSREPORTFORMAT=MARKDOWN
             |
             v
2. Run TMSS10Inventory
             |
             +---- stdout ----> out.md
             |
             +---- stderr ----> err.txt
             |
             v
3. Render with Glow
             |
             v
4. Generate a second report
             |
             v
5. Render out2.md with Glow
             |
             v
6. Compare out.md and out2.md
```

Example commands:

```
set "TMSSREPORTFORMAT=MARKDOWN"

TMSS10Inventory > out.md 2> err.txt

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

After generating a second report:

```
glow -s dark -w 120 .\out2.md
```
```

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

Validation

After setting up Markdown report generation, verify the following:

* The TMSSREPORTFORMAT environment variable is set to MARKDOWN. * The TapeTrack utility completes normally. * The standard output file contains Markdown. * The report can be opened with Glow. * The Markdown table is rendered correctly. * Standard error is captured separately when required. * A second report can be generated without overwriting the first report. * Differences between reports can be identified using a comparison tool.

Troubleshooting

Markdown file is empty

Check that the TapeTrack utility produced output and that the command is being executed from the expected environment.

Check the error output:

```
Get-Content .\err.txt
```

Glow is not recognised

If Glow has just been installed using WinGet, close and reopen PowerShell so that the updated PATH is loaded.

Verify:

```
glow --version
```

The Markdown table is difficult to read

Specify a wider rendering width:

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

Differences appear on many lines

The simple PowerShell comparison compares corresponding line positions. If records have been inserted, deleted or reordered, many lines may appear different.

For TapeTrack inventory reports, compare records using the Barcode field when a record-level comparison is required.

References

* TapeTrack Documentation * Glow Markdown Renderer * Glow CLI

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

Permanent link:
https://docs.tapetrack.com/technote/markdown_reports?rev=1789694470

Last update: **2026/09/18 01:21**