QVLx LABS / DESIGN & ENGINEERING

ET Writer

THE FILE WORKSHOP

Text. Bytes. Transformation.
One deliberate workspace.



FILE WORKFLOW

01	READ	TXT
02	INSPECT	HEX
03	EXPORT	BULK

WORKSPACE / REFERENCE ROUTINE

```
if (file.type === "text") {
  edit(file);
} else {
  inspectBytes(file);
}
```

01 EDIT 02 INSPECT 03 TRANSFORM

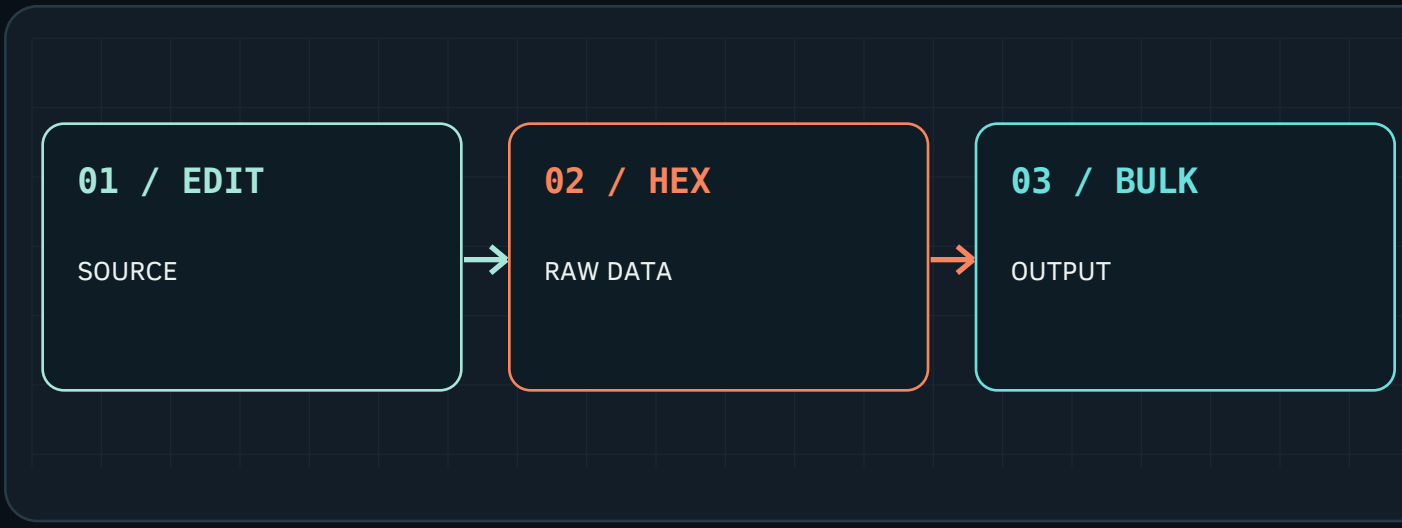
OPEN ETWRITER DOCUMENTATION

11 JUN 2025

01 / FIELD NOTES

Three ways to read a file.

The same content becomes a document, a byte stream, or a repeatable transformation.



SOURCE EDITOR

Structured text, syntax highlighting, and a plain-text fallback.

HEX WORKBENCH

Literal bytes, ASCII preview, and focused binary modification.

BULK PROCESSOR

A sequential instruction language for repeatable text changes across selected folders. Commands are executed in order, and generated outputs preserve the relative file tree.

OPERATING PRINCIPLE Make the change visible. Keep the original recoverable.

02 / FIELD NOTES

Edit the surface.

Text and source editing without losing sight of structure or encoding.

config.json UTF-8 / EDITOR

```
{
  "project": "ETWriter",
  "mode": "Editor",
  "encoding": "UTF-8",
  "ready": true
}
```

LANGUAGE AWARE

CodeMirror-backed syntax color, line numbers, active-line guidance, and recognized extension modes.

FAIL-SAFE EDITING

Line wrapping for prose, Markdown, and configuration; plain textarea fallback if CodeMirror cannot load.

LANGUAGES	JS / TS / JSX HTML / CSS JSON / YAML / XML Markdown Python / Java C / C++ Go / Rust Shell TOML / INI

03 / FIELD NOTES

Read beneath the text.

The byte workbench makes the underlying data legible - and editable.

OFFSET	00	01	02	03	04	ASCII
00000000	48	65	6C	6C	6F	Hello
00000005	20	45	54	57	72	ETWr
0000000A	69	74	65	72	0A	iter.

BEFORE / CAT

43 61 74

C a t

AFTER / BAT

42 61 74

B a t

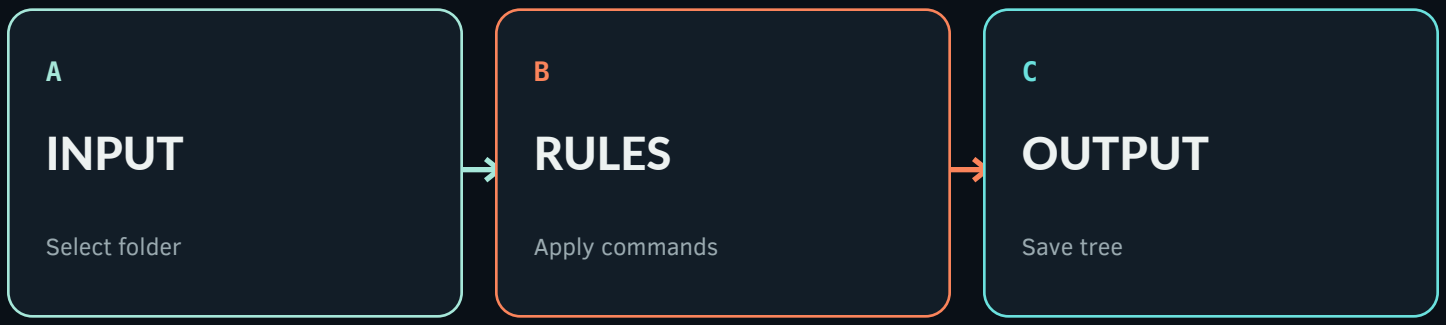
HANDLE WITH INTENT

A byte change can alter a file header, length field, or checksum. ETWriter does not automatically repair binary formats or recompute checksums.

04 / FIELD NOTES

Change an entire tree.

A sequential pipeline of explicit rules over supported text files.



SOURCE TREE

```
project/
  index.html
  styles.css
  scripts/
    app.js
```

PROCESSED TREE

```
output/
  index.html
  styles.css
  scripts/
    app.js
```

BULK ENGINE / INSTRUCTION STREAM

```
FIND_REPLACE|version=1|version=2
FIND_APPEND|TODO|// Follow-up required
LINE_BREAK|-1
```

RULE ORDER MATTERS: each action sees the previous result.

05 / FIELD NOTES

Nine instruments.

A compact command language for precise text transformation.

01

FIND_APPEND

Insert a line after a matching line.

02

FIND_BREAK

Insert a line break at a match.

03

FIND_INSERT

Insert text after a matching token.

04

FIND_REPLACE

Replace matching text literally.

05

FIND_DELETE

Remove matching text.

06

LINE_APPEND

Append at a selected line.

07

LINE_BREAK

Insert a break at a position.

08

LINE_REPLACE

Replace a numbered line.

09

LINE_DELETE

Remove a selected line.

PARSER NOTES

Commands use pipe separators. FIND matches are literal and case-sensitive, not regular expressions. Escapes include \n, \t, and \\. The parser does not escape a literal pipe.

06 / FIELD NOTES

Make small changes.

Practical transformations, each narrow enough to inspect and understand.

01

REPLACE A VALUE

Replace literal, case-sensitive occurrences.

```
FIND_REPLACE|old|new
```

02

REMOVE A TOKEN

Remove matching text without deleting the line.

```
FIND_DELETE|debug
```

03

INSERT AFTER A MATCH

Insert a literal suffix after a match.

```
FIND_INSERT|name|_backup
```

04

ADD A FOLLOW-UP LINE

Create a new line following a matching line.

```
FIND_APPEND|TODO|// review
```

05

REPLACE BY LINE

Replace a line by position; confirm indexing for each command.

```
LINE_REPLACE|2|updated
```

07 / FIELD NOTES

Control the file boundary.

Browser features determine how a document is opened, modified, and exported.

DIRECT FILE ACCESS

Where browser support and permission allow: open, save, and choose folders using native file APIs.

FALLBACK PATH

Ordinary file pickers and downloadable exports; bulk results can be packaged as a ZIP.

PRE-FLIGHT

01

COPY

Keep a recoverable baseline.

02

FILTER

Only supported text extensions.

03

REVIEW

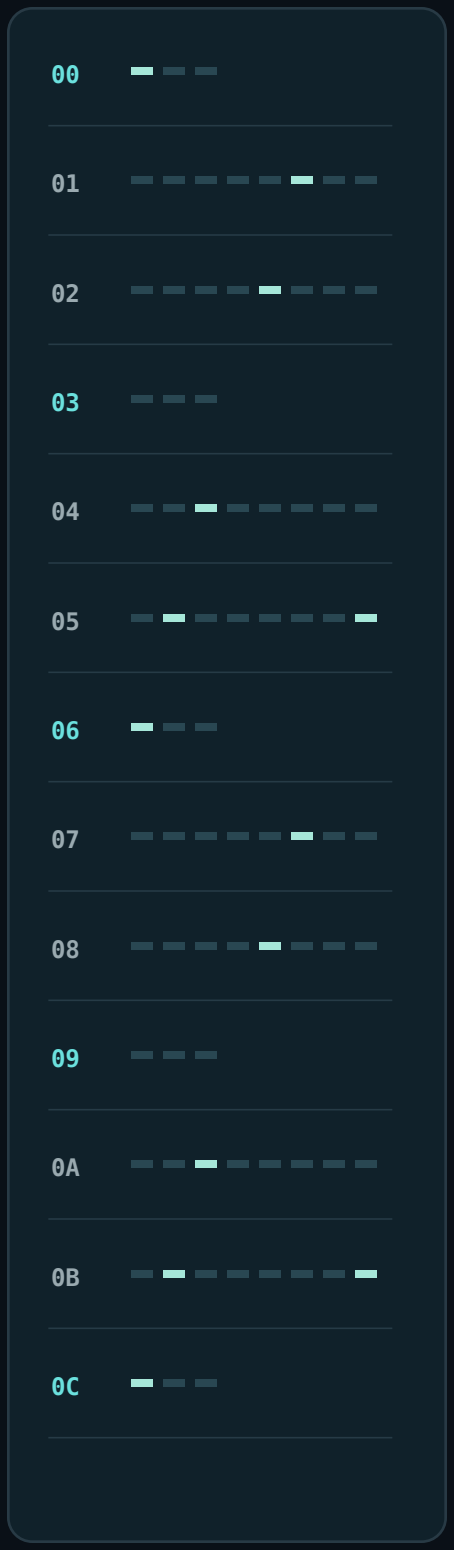
Check output and line endings.

BOUNDARIES THAT MATTER

Some workflows may normalize CRLF to LF. Unsupported or binary files are skipped during bulk processing. Native write access requires browser permission. Inspect generated files before replacing source files.

THE FUTURE IS IN THE DETAIL.

Open the words. Inspect the bytes. Apply the change. ETWriter makes the underlying structure accessible without turning simple work into ceremony.



01 **EDITOR**
UTF-8 source

02 **HEX**
Raw byte view

03 **BULK**
Repeatable rules

EXPLORE ETWRITER

LAUNCH APP | DOCUMENTATION | CONTACT QVLx