shfmt Shell script formatter

START HERE

`shfmt` **`script.sh`**
Print formatted script

`shfmt` **`-l -w .`**
Fix a whole tree

`shfmt` **`-d .`**
Show diffs, fail if any

`shfmt` **`-i 2 -ci`**
2 spaces, indent cases

`shfmt` **`-l .`**
List unformatted files

`cat script.sh | shfmt`
Format stdin

`shfmt` **`-w script.sh`**
Format in place

`shfmt` **`-f .`**
Find shell files

Script *.sh or shebang → `parse` → **Syntax tree** dialect rules → `print` → **Formatted code** flags / .editorconfig → `-w / -d / -l` → **Result** file, diff or list

CONCEPTS

Default output uses tabs. Pick your style once with flags or an .editorconfig, then let CI enforce it with -d.

Any parser or printer flag on the command line makes shfmt ignore EditorConfig formatting options. Use -i=0 as a harmless way to do that.

RUN

`shfmt` *<file>*
Print the formatted script

`shfmt` **`-w`** *<file>*
Write the result back to the file

`shfmt` **`-d`** *<path>*
Show a diff, exit non-zero if different
The CI-friendly check. Color is off with NO_COLOR or TERM=dumb.

`shfmt` **`-l`** *<path>*
List files whose formatting differs

`shfmt` **`-l -w .`**
Format everything under here
A directory path walks it and picks up shell scripts only.

`shfmt` **`-`**
Read from stdin
Without arguments on a terminal it prints usage instead of waiting.

`shfmt` **`-l=0`** *<path>*
List files separated by NUL

RUN (cont.)

`shfmt` **`--version`**
Show version

STYLE (PRINTER)

`shfmt` **`-i`** *<n>*
Indent with n spaces (0 = tabs)
Tabs are the default.

`shfmt` **`-ci`**
Indent case branches

`shfmt` **`-bn`**
Binary ops like && and | may start a line

`shfmt` **`-sr`**
Space after redirect operators

`shfmt` **`-bl`**
Opening brace, then and do on their own line

`shfmt` **`-kp`**
Keep column alignment padding
Deprecated; slated for removal.

`shfmt` **`-mn`**
Minify the script
Implies -s.

`shfmt` **`-i 2 -ci -bn`**
Roughly Google's shell style

DIALECT AND PARSING

`shfmt` **`-ln`** *<dialect>*
Set dialect: bash, posix, mksh, bats, zsh
Default is auto: from file extension, then shebang, then bash.

DIALECT AND PARSING (cont.)

`shfmt` **`-p`**
Shorthand for -ln=posix
A .sh extension alone implies posix unless the shebang says otherwise.

`shfmt` **`-s`**
Simplify code where safe

`shfmt` **`--filename`** *<name>*
Name for stdin input
Needed for EditorConfig and dialect detection when reading stdin.

`shfmt` **`--detect`** *<mode>*
How to find files: default, exec or all

`shfmt` **`--apply-ignore`**
Always honor EditorConfig ignore rules

`shfmt` **`-f`** *<path>*
Print shell files found, no formatting

`shfmt` **`--to-json`**
Print the syntax tree as JSON

CONFIG (.EDITORCONFIG)

.editorconfig

```
[*.sh]
indent_style = space
indent_size = 2
case_indent = true
binary_next_line = true
space_redirects = true
language_dialect = bash
```

CONFIG (.EDITORCONFIG) (cont.)

`indent_style / indent_size`
Same as -i

`case_indent, binary_next_line`
Same as -ci and -bn

`[[shell]] section header`
Match scripts by shebang, no extension
Outside the EditorConfig spec and may change.

`ignore = true`
Skip a path when walking directories

RECIPES

CI check

```
# fails the job when anything differs
shfmt -d -i 2 -ci .
```

Format only tracked scripts

```
git ls-files -z '*.sh' | xargs -0 shfmt -w
```

Syntax check instead of bash -n

```
shfmt -ln posix script.sh >/dev/null
# prints line:col and error if invalid
```

pre-commit hook

```
# .pre-commit-config.yaml
- repo: https://github.com/scop/pre-commit-shfmt
  rev: v3.14.1-1
  hooks:
    - id: shfmt
```