gron Greppable JSON

CONCEPTS

gron turns every leaf of a JSON document into one line such as json.user.tags[0] = "a"; and the root is always called json. grep keeps whole paths, so you see where a value lives. Do not rely on grep and sed pipelines in scripts; use jq there.

JSON file, URL or stdin
↓ gron
Assignments json.a.b[0] = 1;
↓ grep / sed / diff
Fewer lines still valid statements
↓ gron -u
JSON again
Round trip: flatten, filter, rebuild

READING INPUT

gron *<file>*
Flatten a JSON file

gron *<url>*
Fetch a URL and flatten the body

curl -s *<url>* **| gron**
Flatten JSON from stdin

gron -
Read from stdin explicitly

gron -s *<file>*
Treat each line as its own JSON object

gron -k *<url>*
Skip TLS certificate checks for a URL

OUTPUT CONTROL

gron -m *<file>*
Monochrome, no ANSI colours

gron -c *<file>* **| less -R**
Force colour through a pager

gron --no-sort *<file>*
Keep document order, faster

gron --version
Print the version

SEARCHING

gron *<file>* **| grep** *<text>*
Show paths whose line matches

gron *<file>* **| grep -i** *<text>*
Case-insensitive match

gron *<file>* **| grep -v** *<text>*
Drop lines that match

gron *<file>* **| grep '^json\.**<key>**'**
Anchor to a top-level key

gron *<file>* **| grep -F '[0]'**
Match a literal array index

gron *<file>* **| grep -E '**<a>**|**<b>**'**
Match either of two patterns

REBUILDING JSON

gron -u *<file>*
Turn assignments back into JSON

gron *<file>* **| grep** *<text>* **| gron -u**
Keep a slice of a document as valid JSON

gron --ungron
Long form of -u

gron *<file>* **| gron -v**
Print the values of the statements

COMPARING AND EDITING

diff <(gron *<a>***) <(gron** *<b>***)**
See which paths differ between two files

diff -u <(gron *<a>***) <(gron** *<b>***)**
Unified diff of two documents

gron *<file>* **| sed 's/**<old>**/**<new>**/' | gron -u**
Substitute a value, then rebuild

gron *<file>* **| grep -v** *<path>* **| gron -u**
Delete a key