jc CLI output to JSON

CONCEPTS

jc only converts text to JSON; it does not filter. Pair it with jq. Most parsers return an array of objects, but some (like uptime) return one object, so check the schema with jc -h --<parser>.

Command output ps, dig, ls, ...
↓ text
jc --<parser> or magic
↓ JSON
jq / script filter, select, format

jc sits between text tools and JSON tools

RUNNING JC

jc *<command>* **[args]**
Magic syntax: run and parse in one step

<command> **| jc --***<parser>*
Standard syntax: parse piped text

cat *<file>* **| jc --***<parser>*
Parse a saved file or capture

jc /proc/*<path>*
Parse a /proc file

jc *<start>*:*<end>* **--***<parser>*
Parse only a slice of input lines

jc -s --*<parser>*
Slurp many one-line inputs into an array

jc dig *<host>* **| jq -r '.[].answer[].data'**
Magic syntax feeding jq

ps aux | jc --ps
Same, with the standard syntax

OUTPUT OPTIONS

jc -p *<command>*
Pretty-print JSON

jc -y *<command>*
YAML output

jc -r *<command>*
Raw output: strings, no int or date conversion

jc -M *<command>*
Add _jc_meta: parser, timestamp, exit code

jc -m *<command>*
Monochrome output

jc -q *<command>*
Suppress warnings

OUTPUT OPTIONS (cont.)

jc -d *<command>*
Debug a parse failure (-dd verbose)

COMMON PARSERS

jc ps aux
Processes (--ps)

jc ls -l *<dir>*
Directory listing (--ls)

jc df -h
Disk usage per filesystem (--df)

jc dig *<host>*
DNS answers and timings (--dig)

jc ss -tulpn
Sockets (--ss)

jc ip route
Routing table (--ip-route)

jc lsblk
Block devices as a tree (--lsblk)

jc systemctl list-units
Units and their state (--systemctl)

jc stat *<file>*
File metadata (--stat)

jc free
Memory use (--free)

jc uptime
Load averages and uptime (--uptime)

jc mount
Mounted filesystems (--mount)

jc env
Environment variables (--env)

jc netstat -an
Connections (--netstat)

jc ping -c 3 *<host>*
Round-trip stats (--ping)

FILE AND STRING PARSERS

jc --csv < *<file>***.csv**
CSV rows to objects

jc --yaml < *<file>***.yml**
YAML file to JSON

jc --toml < *<file>***.toml**
TOML file to JSON

jc --ini < *<file>***.ini**
INI file to JSON

FILE AND STRING PARSERS (cont.)

jc --xml < *<file>***.xml**
XML file to JSON

jc --kv < *<file>*
key=value text to JSON

jc --jwt *<<< <token>*
Decode a JWT without verifying

jc --asciitable < *<table>***.txt**
Any aligned text table to JSON

STREAMING AND HELP

<command> **| jc --***<parser>***-s**
Streaming parser: one JSON line per input line

ping *<host>* **| jc --ping-s -u | jq .**
Live ping as it arrives

jc -h
List parsers

jc -h --*<parser>*
Parser docs, schema and examples

jc -v
Show version

CUSTOM PARSERS

A plugin is a Python file named like a module (letters, digits, underscore) with a parse() function. Copy jc/parsers/foo.py as the template; a plugin with a built-in's name overrides it.

RECIPES

Biggest memory users

```
jc ps aux \
  | jq -r 'sort_by(-.rss)[:5][] |
"\(.rss) \(.command)"'
```

Disks above 80% full

```
jc df | jq -r '.[]
  | select((."use%"|tonumber) >
80) | .mounted_on'
```

Listening ports and processes

```
jc ss -tlnp \
  | jq -r '.[] | "\(.local_port)
\(.process_name)"'
```

Failed systemd units

```
jc systemctl --failed \
  | jq -r '.[].unit'
```