promtool Prometheus CLI toolkit

CONCEPTS

promtool ships in the Prometheus release tarball and works offline on files; only query, push and debug talk to a server.

Write rules rules.yml
↓ `check rules`
Unit test tests.yml
↓ `test rules`
Check config prometheus.yml
↓ `reload`
Live server query to verify

Typical rule workflow

Exit code 0 means ok, 1 an error, and 3 a lint finding when --lint-fatal is set, so every check fits a CI step.

CHECK

`promtool check config <file>`
Validate config and referenced rule files

`promtool check rules <rules.yml>`
Validate rule files

`promtool check config --syntax-only <file>`
Skip referenced files

`promtool check rules --lint=all <rules.yml>`
Enable all lint checks

`promtool check config --lint-fatal <file>`
Exit 3 on lint findings

`promtool check web-config <file>`
Validate the web config (TLS/ auth)

`promtool check service-discovery <file> <job>`
Show targets after relabeling

`promtool check metrics < <file>`
Lint metric names and types from stdin

`promtool check healthy --url=<url>`
Is the server healthy?

CHECK (cont.)

`promtool check ready --url=<url>`
Is the server ready?

TEST RULES

`promtool test rules <tests.yml>`
Run rule unit tests

`promtool test rules <a.yml> <b.yml>`
Run several test files

`promtool test rules --run=<group> <tests.yml>`
Run only matching test groups

`promtool test rules --debug <tests.yml>`
Print evaluation detail

Test file keys

Key	Purpose
`rule_files`	Rule files under test
`evaluation_interval`	Rule eval step (default 1m)
`tests[].interval`	Sample step for input_series
`input_series`	series + values pairs
`alert_rule_test`	eval_time, alertname, exp_alerts
`promql_expr_test`	expr, eval_time, exp_samples

Series value notation

Value	Means
`1+2x3`	1 3 5 7 (start, step, count)
`5x3`	5 5 5 5
`_`	missing sample
`stale`	stale marker
`_x3`	three gaps

In a test file, rule_files paths are relative to the test file, not to your shell directory.

QUERY A SERVER

`promtool query instant <url> '<expr>'`
Run an instant query

`promtool query range <url> '<expr>' --start=<t> --end=<t> --step=1m`
Run a range query

QUERY A SERVER (cont.)

`promtool query instant --time=<t> <url> '<expr>'`
Evaluate at a point in time

`promtool query series <url> --match='<selector>'`
List matching series

`promtool query labels <url> <label>`
List values of a label

`promtool query labels <url> __name__`
List all metric names

`promtool query instant -o json <url> '<expr>'`
Output JSON instead of PromQL text

TSDB

`promtool tsdb list <dir>`
List blocks in a data dir

`promtool tsdb list -r <dir>`
Show human-readable times and sizes

`promtool tsdb analyze <dir>`
Cardinality and churn report

`promtool tsdb analyze --limit=50 <dir> <block-id>`
Top 50 for one block

`promtool tsdb dump <dir> --match='<selector>'`
Dump samples as text

PUSH, DEBUG, PROMQL

`promtool --version`
Print the version

`promtool --help`
List all commands

RECIPES

Validate before deploy

```
promtool check config
prometheus.yml
promtool check rules rules/*.yml
promtool test rules tests/*.yml
```

Reload after a successful check

```
promtool check config
prometheus.yml \
  && curl -X POST
localhost:9090/-/reload
```