# yq YAML processor

## CONCEPTS

This sheet is for mikefarah/yq v4 (the Go one). The Python yq wrapper around jq uses different flags and syntax.

**Input** YAML, JSON, CSV...
↓ `-p parse`
**Expression** .a[] | select(...)
↓ `each match`
**Output** YAML by default
↓ `-o encode`
**File or stdout** -i writes back

Expressions run once per document; eval-all sees every document of every file at once

## READING

`yq '.<key>' <file>`
Print one value

`yq '.<a>.<b>[0]' <file>`
Walk keys and array index

`yq '.[]' <file>`
Every child of a list or map

`yq '.<key>[]' <file>`
Every item of a list under a key

`yq '.["<odd key>"]' <file>`
Key with dots or spaces

`yq '.<key> // "<default>"' <file>`
Fallback when null or false

`yq '.. | select(has("<key>"))' <file>`
Find nodes anywhere in the tree

`yq <file>`
Pretty-print a whole file

## UPDATING

`yq -i '.<key> = "<value>"' <file>`
Set a value in place

`yq '.<key> = "<value>"' <file>`
Preview the change on stdout

`yq -i '.<a>.<b>.<c> = 1' <file>`
Creates missing parents

## UPDATING (cont.)

`yq -i '.<n> |= . + 1' <file>`
Update using the old value

`yq -i '.<list> += ["<x>"]' <file>`
Append to a list

`yq -i 'del(.<key>)' <file>`
Delete a key

## SELECTING AND LISTS

`yq '.[] | select(.<k> == "<v>")' <file>`
Items where a field matches

`yq '(.[] | select(.<k> == "<v>") | .<f>) = "<new>"' <file>`
Update only matching items

## FILES AND MERGING

`yq ea '<expr>' <file>...`
Evaluate over all documents together

`yq ea '. as $i ireduce({}; . * $i)' <f1> <f2>`
Deep merge files, later wins

`yq -n '.a = 1'`
Create a document from nothing

## FORMATS

`yq -o=json <file>`
Output as JSON

`yq -p=json -o=yaml <file>`
JSON in, YAML out

## VARIABLES, STRINGS AND ANCHORS

`NAME=<v> yq -i '.a = strenv(NAME)' <file>`
Inject an env var as string

## SETUP AND HELP

`yq --version`
Show installed version

`yq --help`
Flags and commands