jq JSON processor

CONCEPTS

Everything is a filter: it takes one input and emits zero or more outputs. The pipe | feeds each output of the left side into the right side. Quote the program in single quotes.

COMMAND-LINE FLAGS

jq -r '<*filter*>**'**
Print strings without quotes

jq -c '<*filter*>**'**
Compact, one value per line

jq -n '<*filter*>**'**
Use null as input, read nothing

jq -s '<*filter*>**'** <*file*>
Slurp all inputs into one array

jq -R '<*filter*>**'** <*file*>
Read each line as a string

jq --arg <*name*> <*value*> **'$**<*name*>**'**
Pass a string variable

SELECTING

.
The whole input

.<*key*>
Object field

.<*key*>**.**<*sub*>
Nested field

.[<*n*>**]**
Array element, negative counts from end

.[]
Every element or value

FILTERING AND LOGIC

select(<*cond*>**)**
Keep inputs where cond is true

map(<*f*>**)**
Apply f to each array element

== != < > <= >=
Compare values

and / or / not
Combine conditions

<*a*> **//** <*default*>
Use default if a is null or false

FILTERING AND LOGIC (cont.)

if <*cond*> **then** <*a*> **else** <*b*> **end**
Conditional

test("<*regex*>**")**
Regex match, returns boolean

CONSTRUCTING

{<*a*>**,** <*b*>**: .**<*c*>**}**
Build an object, shorthand for {a: .a}

[<*f*>**]**
Collect all outputs into an array

"<*text*> **\(.**<*key*>**)"**
String interpolation

.<*key*> **=** <*v*>
Set a field

.<*key*> **|=** <*f*>
Update a field with a filter

del(.<*key*>**)**
Remove a field or element

to_entries / from_entries / with_entries(<*f*>**)**
Object to key-value pairs and back

ARRAYS AND AGGREGATION

length
Size of array, object, string

keys
Sorted object keys or array indexes

sort / sort_by(.<*key*>**)**
Sort array

group_by(.<*key*>**)**
Group into sub-arrays

unique / unique_by(.<*key*>**)**
Remove duplicates

add
Sum numbers or concatenate

STRINGS AND FORMATS

tostring / tonumber
Convert type

split("<*sep*>**") / join("**<*sep*>**")**
Split and join strings

@csv / @tsv
Array to CSV or TSV row (use with -r)