Justfile Cheat Sheet

by linux_china via cheatography.com/141366/cs/30282/

Simple justfile

```
#!/usr/bin/env just --justfile
# hello is recipe's name
hello:
   echo " Hello World! "
```

Attention: don't use keywords as recipe name, such as "import", "export", "alias" etc.

default Recipe - first or [default]

```
default: lint build test
# default recipe to display help
inform ation
default:
    @just --list
# default recipe with [default]
attribute
[default]
hello:
     echo hello
# if no default recipe, first
recipe will be default
```

Aliases

```
alias t := test
alias c := check
```

Settings

```
set shell := ["zsh", "-cu"]
#set shell := ["bu n", " exe c"]
set dotenv -re quired
set dotenv -load := true
serv:
     echo " $DA TAB ASE _AD -
DRESS from .env"
set positi ona l-a rgu ments :=
true
foo:
   echo $0
   echo $1
```

Strings - escape with Double-quoted

```
string-with-tab := "\t"
string -wi th- newline := " \n"
escapes := '\t\n \r\"\\'
shell- exp and ed-path :=
x'~/${ FOO }/$ {BAR}'
message := f'hello {{name}}'
# this string will evaluate to
`foo\n bar\n`
x := '''
   foo
   bar
'''
```

just command line

```
# run recipe
$ just hello param1
# list recipes in alphabetical order
$ just --list
$ just --summary
# Show full information about recipe
just --show test
# select recipes to run interactively
$ just --choose
# shell completion
just --completions zsh
```

GitHub Actions

```
- uses: extractions/setup-
just@v1
    with:
        jus t-v ersion: 1.45.0
```

IDE integration

VS Code: https://marketplace.visualstud-io.com/items?itemName=skellock.just

JetBrains: https://plugins.jetbrains.com/plu-gin/18658-just

PROJECT_DIR env variable available for JetBrains IDEs

Just module/import

```
# load bar/justfile,
bar/.justfile, bar.just
mod bar
# include the contents of
another justfile
import 'foo/b ar.j ust'
```

just --unstable bar::hello

Recipe with parameters

```
filter PATTERN:
     echo "{{PATTERN}}"
# param with default value
email addres s=' mas ter @ex -
amp le.c om':
     echo "{{address}}"
# param with validation pattern
[arg('n', patter n=' \d+')]
double n:
   echo $(({{n}} * 2))
# param with expression
test triple =(a rch() + " -un -
kno wn- unk now n"):
   ./test "{{triple}}"
# variadic param: '+' aceept one
or more values
[doc(' Backup files')]
backup +FILES:
   scp {{FILES}}me@exa mpl -
e.com
# variadic param with *: zero or
more values
commit MESSAGE *FLAGS:
```


Sponsored by **Readable.com**
Measure your website readability!
https://readable.com

Recipe with parameters (cont)

```
> git commit {{FLAGS}} -m "{{MESSAGE}}"
```

If possible, please put param in quotation mark and friendly for syntax highlight.

Recipe with env variable for command

```
# recipe param as env variable with $ sign
hello $name:
    echo $name
```

Recipe Dependencies - Before, After & Around

```
# execution sequence: a -> b -> c -> d
b: a && c d
# execute recipe 'a' around b:
   echo 'B start!'
   just a
   echo 'B end!'
# depend with params by expression
default: (build " mai n")
build target:
    @echo 'Building {{target}}...'
```

Command annotate: quiet(@), suppress(-), invert(!)

```
hello:
   @ echo " command will not be echoed "
   - echo " ignore none-zero exit status and contin ue"
@hello2:
   echo " command will not be echoed "
# Invert command exit status by ! - shell feature
hello3:
   # if command succee ds(exit status is 0), exit just
   ! git branch | grep '* master'
```

Recipe with other Languages

```
bash-test:
        #!/ usr /bi n/env bash
       set -euxo pipefail
        hel lo='Yo'
       echo " $hello from bash!"
[scrip t("b ash ")]
bash-t est2:
       set -euxo pipefail
        hel lo='Yo'
[scrip t(' bun', 'run')]
[exten sio n(".t s")]
bun-hello:
     let name: string = " Jac - kie "
      con sol e.log(`Hello ${name }!)
[scrip t("d uck db", " -f")]
query:
      select version()
       echo " $hello from bash!"
```

Private Recipes - name starts with _

```
test: _test-helper
   ./b in/test
# ommited from 'just --list'
_test- helper:
   ./b in/ sup er- sec ret -te - st- hel per -stuff
```

Recipes as shell alias

```
for recipe in `just -f ~/.justfile --summary`; do
    alias $recip e="just -f ~/.jus tfile -d. $recip e"
done
```

Recipe with Python venv

```
venv:
   [ -d .venv ] || uv venv
run: venv
   ./.v en v/b in/ python3 main.py
```

Variables

```
version := "0.2.7"
tardir := " awe som esa uce -" + version
tarball := tardir + ".ta r.g z"
path := " a" / " b" # join path
var1 := '' && 'goodbye' # ''
var2 := 'hello' && 'goodbye' # 'goodbye'
var3 := '' || 'goodbye' # 'goodbye'
var4 := 'hello' || 'goodbye' # 'hello'
test:
     echo "{{version}}"
# set/ov erride variables from just command line
$ just --set version 1.1.0
```

- Substitutions: {{NAME}}
- Logical Operators: || &&
- Joining Path: /

Environment variable for commands

```
export RUST_BACKTRACE := "1"
test:
       # will print a stack trace if it crashes
        cargo test
```

backtick - capture output from evaluation

```
JAVA_HOME := `jbang jdk home 11`
# backtick code block
stuff := ```
      foo ="he llo "
     echo $foo " wor ld"
 ```
```



Sponsored by **Readable.com**
Measure your website readability!
https://readable.com
```

backtick - capture output from evaluation (cont)

> done BRANCH=`git rev-parse --abbrev-ref HEAD`:
  git checkout master
sloc:
    @echo "`wc -l *.c` lines of code"
 # backtick works anywhere: string/variable/params

Just functions

```
hello name:
     echo "{{os()}}"
     echo "{{uppercase(name)}}"
# function categories
* System Inform ation
* Enviro nment Variables
* Justfile and Justfile Directory
* String Manipu lation
* Path Manipu lation
# String contact: (key + " :" + value)
```

Conditional expressions: if, loop and while

```
# regular expression match
fo := if " hi" =~ 'h.+' { " - mat ch" } else { " mis mat ch" }
test:
     if true; then echo 'True!'; fi
     for file in `ls .`; do echo $file; done
      while `serve r-i s-d ead`; do ping -c 1 server; done
foo bar:
   echo '{{ if bar == " bar " { " hel lo" } else { " bye " } }}'
```

Attention

```
# Each command line is executed by a new shell.
# If a command line failed, just will exit, \
# and subsequent command lines will not be executed.
change -wo rki ng-dir:
     cd bar && pwd
     # multi-line construct - escape newline with slash
     if true; then \
               echo 'True!'; \
     fi
# justfile is case insens itive: Justfile, JUSTFILE etc
# justfile could be hidden: '.just file'
# Call recipe from sub dir: `~/app 1/t arg et>$ just build`
```


Sponsored by **Readable.com**
Measure your website readability!
https://readable.com