watchexec Run on file change

CONCEPTS

Put -- before your command so its own flags are not read by watchexec.

File event save, create, delete
↓ debounce
Filters paths, ignores, -e, -f
↓ pass
Command start or restart
↓ exit
Wait for next change

By default it watches the current directory recursively and honours .gitignore and .ignore, so build output folders are usually skipped already.

The default shell is $SHELL (sh if unset); with -n the command is split on spaces, so no pipes, globs or redirects.

RUNNING

`watchexec <cmd> [args]`
Run a command whenever files change

`watchexec -- <cmd> --flag`
Stop option parsing before the command

`watchexec -p <cmd>`
Postpone: skip the initial run

`watchexec -r <cmd>`
Restart the command on change

`watchexec -c <cmd>`
Clear the screen before each run

`watchexec -n <cmd>`
No shell: run the program directly

`watchexec --shell <shell> <cmd>`
Choose the shell, e.g. bash or zsh

`watchexec -E <KEY=VALUE> <cmd>`
Set an env var for the command only

`watchexec --workdir <dir> <cmd>`
Run the command in another directory

RUNNING (cont.)

`watchexec -I <cmd>`
Interactive: r restart, s stop, p pause, q quit

`watchexec --exit-on-error <cmd>`
Quit when the command fails

`watchexec --timeout <time> <cmd>`
Stop a run that takes too long

`watchexec --delay-run <time> <cmd>`
Sleep before each run

WHAT TO WATCH

`-w <path>`
Watch a file or directory (repeatable)

`-W <dir>`
Watch a directory without recursion

`-e <ext,ext>`
Only react to these extensions

`-f <glob>`
Only react to paths matching a pattern

`-i <glob>`
Ignore paths matching a pattern

`--no-vcs-ignore`
Watch files that .gitignore excludes

`--fs-events <list>`
Filter by event: create, remove, rename, modify, access

`--no-meta`
Skip metadata-only events

BUSY AND SIGNALS

`-o <mode>`
If busy: do-nothing, queue, restart or signal

`-r`
Same as -o restart

`-s <signal>`
Send a signal to the running command on change

`--stop-signal <signal>`
Signal used to stop it (default SIGTERM)

BUSY AND SIGNALS (cont.)

`--stop-timeout <time>`
Grace time before a force kill (default 10s)

`--wrap-process <mode>`
auto, group, session or none

DEBOUNCE AND POLLING

`-d <time>`
Wait for quiet before acting (default 50ms)

`--poll [interval]`
Poll instead of OS events (network drives)

CHANGED PATHS

`--emit-events-to <mode>`
none, environment, stdio, file, json-stdio, json-file

`--print-events`
Print the events that triggered a run

SETUP AND DEBUG

`watchexec --completions <shell>`
Print shell completions

`watchexec --manual`
Show the full manual

`watchexec -v / -vvvv`
More diagnostic logging

`watchexec -V`
Show the version

RECIPES

Run tests on Rust changes

```
watchexec -c -e rs,toml -- cargo test
```

Dev server with restart

```
watchexec -r -e py,html -- python manage.py runserver --noreload
```

Rebuild assets, skip output

```
watchexec -w src -i "dist/**" -e js,css -- npm run build
```

Debounced build, then notify

```
watchexec -d 500 -N -p -e md -- make site
```