Delve Go debugger

CONCEPTS

> dlv debug and dlv test build with optimizations off; for dlv exec build with -gcflags="all=-N -l" yourself.

Launch debug, test, exec
↓ `break + continue`
Stopped goroutine, frame
↓ `inspect`
Inspect print, locals, stack
↓ `next / step`
Resume continue, restart

START A SESSION

`dlv debug [<pkg>]`
Build and debug a main package

`dlv debug -- <args>`
Debug with program arguments

`dlv test [<pkg>]`
Build and debug a test binary

`dlv test -- -test.run <TestName>`
Debug a single test

`dlv exec <binary> -- <args>`
Debug a prebuilt binary

`dlv attach <pid>`
Attach to a running process

REMOTE AND HEADLESS

`dlv debug --headless --listen=:2345`
Run only the debug server

`dlv connect <host>:<port>`
Attach the terminal client

BREAKPOINTS

`break <func>`
Break at a function (b)

`break <file>:<line>`
Break at a line

`break <loc> if <cond>`
Break when a condition holds

BREAKPOINTS (cont.)

`breakpoints`
List breakpoints (bp)

`clear <id>`
Delete one breakpoint

`condition <id> <expr>`
Add a condition later (cond)

RUN CONTROL

`continue`
Run to the next breakpoint (c)

`next`
Step over the next line (n)

`step`
Step into calls (s)

`stepout`
Run until the function returns (so)

`restart`
Restart with the same arguments (r)

`exit`
Quit the debugger (q)

INSPECT DATA

`print <expr>`
Evaluate an expression (p)

`locals`
Show local variables

`args`
Show function arguments

GOROUTINES

`goroutines`
List goroutines (grs)

STACK AND SOURCE

`stack`
Show the call stack (bt)

`up / down`
Move one frame up / down

`list`
Show source around current line (l)