Cabal Haskell build tool

START HERE

`cabal` **`init --interactive`**
New package

`cabal` **`install`** *`<exe>`*
Install a program

`cabal` **`build`**
Compile everything

`cabal` **`freeze`**
Pin solved versions

`cabal` **`run`** *`<exe>`* **`--`** *`<args>`*
Run an executable

`cabal` **`test`**
Run test suites

`cabal` **`repl`**
GHCi with the project

`cabal` **`update`**
Refresh package index

Project cabal.project
Package <name>.cabal
Component library, exe, test, bench
Module *.hs

CONCEPTS

Commands work on a project: a cabal.project (or just one .cabal file) and the packages it lists. Builds land in dist-newstyle.

Targets look like pkg:component, e.g. mypkg:exe:app or mypkg:test:spec. Use all, a module name or a file path as well.

cabal build only reads the package index you last fetched: run cabal update first on a new machine.

PROJECTS

`cabal` **`init --interactive`**
Create a package with prompts

`cabal` **`init --non-interactive`**
Create a package with defaults

`cabal` **`check`**
Lint the .cabal file for Hackage

BUILD

`cabal` **`build`**
Build all default targets

`cabal` **`build`** *`<target>`*
Build one package or component

BUILD (cont.)

`cabal` **`build all`**
Build every package in the project

`cabal` **`build --enable-tests --enable-benchmarks`**
Include tests and benchmarks

`cabal` **`build -O2`**
Optimise (default is -O1)

`cabal` **`build --ghc-options=-Wall`**
Pass flags to GHC

`cabal` **`build -w ghc-`***`<version>`*
Use a specific compiler

`cabal` **`build --only-dependencies`**
Build just the dependencies

`cabal` **`clean`**
Delete dist-newstyle

RUN & REPL

`cabal` **`run`**
Run the only executable

`cabal` **`run`** *`<exe>`* **`--`** *`<args>`*
Run an executable with arguments
Everything after -- goes to your program, not to cabal.

`cabal` **`repl`**
Open GHCi on the default target

RUN & REPL (cont.)

`cabal` **`repl`** *`<target>`*
Open GHCi on one component

`cabal` **`repl --enable-tests`**
Load test modules too

`cabal` **`repl -b`** *`<pkg>`*
Start a REPL with an extra package

`cabal` **`exec --`** *`<cmd>`*
Run a command in the project environment

`cabal` **`list-bin`** *`<exe>`*
Print the path of a built binary

TEST & DOCS

`cabal` **`test`**
Build and run all test suites

`cabal` **`test`** *`<suite>`*
Run one test suite

`cabal` **`test --test-show-details=direct`**
Stream test output live

`cabal` **`test --test-options="`***`<args>`***`"`**
Pass arguments to the test program

`cabal` **`bench`**
Run benchmarks

`cabal` **`haddock`**
Build API documentation

DEPENDENCIES

`cabal` **`update`**
Download the latest package index

`cabal` **`list`** *`<text>`*
Search package names

`cabal` **`info`** *`<pkg>`*
Show details of a package

`cabal` **`outdated`**
List dependencies with newer versions

`cabal` **`freeze`**
Write cabal.project.freeze

`cabal` **`build --allow-newer`**
Ignore upper version bounds
Handy to test a bump, but fix the bounds in the .cabal file afterwards.

INSTALL

`cabal` **`install`** *`<exe>`*
Install an executable from Hackage

`cabal` **`install --installdir=`***`<dir>`*
Choose where binaries go

`cabal` **`install --overwrite-policy=always`**
Replace an installed binary

`cabal` **`install exe:`***`<name>`*
Install the executable of this project

PUBLISH

`cabal` **`sdist`**
Make a source tarball

`cabal` **`upload`** *`<tarball>`*
Upload a candidate to Hackage

PACKAGE FILE

.cabal stanzas

Stanza	Use
`library`	Reusable code; list exposed-modules
`executable <name>`	Program; needs main-is
`test-suite <name>`	Tests; type: exitcode-stdio-1.0
`benchmark <name>`	Benchmarks; same form as tests
`common <name>`	Shared fields, pulled in with import:

Common fields

Field	Meaning
`cabal-version`	Spec version, e.g. 3.0
`build-depends`	Packages with version bounds
`hs-source-dirs`	Where the .hs files live
`default-language`	GHC2021 or Haskell2010
`ghc-options`	Flags such as -Wall
`other-modules`	Internal, not exported modules

PACKAGE FILE (cont.)

.cabal skeleton

```
cabal-version: 3.0
name:          demo
version:       0.1.0.0
build-type:    Simple

common shared
    default-language: GHC2021
    ghc-options:      -Wall
    build-depends:    base <5

library
    import:           shared
    hs-source-dirs:   src
    exposed-modules:  Demo

executable demo
    import:           shared
    hs-source-dirs:   app
    main-is:          Main.hs
    build-depends:    demo
```

CABAL.PROJECT

`packages: .`
List the packages in the project

`packages: ./*/*.cabal`
Glob over several packages

`tests: True`
Always enable test suites

`constraints:` *`<pkg>`* **`==`***`<ver>`*
Pin a version for the project

`with-compiler: ghc-`*`<version>`*
Fix the compiler