GoReleaser Ship Go releases

CONCEPTS

GoReleaser releases the latest git tag. Use semver tags (v1.2.3) and a clean working tree, or add --snapshot to skip the checks.

Git tag v1.2.3
↓ build
Binaries per OS/arch
↓ archive
Archives + checksums
↓ sign
Signatures optional
↓ publish
GitHub release + changelog
What a release does

Everything lands in dist/. Use --clean on every run so old artifacts do not mix with new ones.

SETUP

goreleaser init
Write an example .goreleaser.yaml

goreleaser check
Validate the config file

goreleaser healthcheck
Check that needed tools are installed

goreleaser --help
List commands and global flags

goreleaser *<command>* **--help**
Flags of one command

goreleaser --version
Show the installed version

goreleaser --config *<file>* *<command>*
Use a custom config path

BUILD

goreleaser build --snapshot --clean
Build all targets without a tag

goreleaser build --single-target --clean
Build only for this machine

GOOS=*<os>* **GOARCH=***<arch>* **goreleaser build --single-target**
Build one chosen target

BUILD (cont.)

goreleaser build --id *<build>*
Build one build ID

goreleaser build --output *<path>*
Copy the binary to a path

RELEASE

goreleaser release --clean
Publish the release for the current tag

goreleaser release --snapshot --clean
Full local run, nothing published

goreleaser release --skip=publish
Build and package but do not publish

goreleaser release --release-notes *<file>*
Use your own release notes

git tag -a v*<x.y.z>* **-m "***<msg>***"**
Create the release tag

git push origin v*<x.y.z>*
Push the tag

GITHUB_TOKEN=*<token>* **goreleaser release**
Pass the token for one run

CONFIG KEYS

version: 2
First line of the config file

builds:
Binaries to compile

archives:
How binaries are packaged

checksum:
Checksum file settings

changelog:
Release notes from commits

release:
GitHub/GitLab/Gitea release options

BUILDS

main: ./cmd/*<name>*
Path of the main package

BUILDS (cont.)

goos: [linux, darwin, windows]
Target operating systems

goarch: [amd64, arm64]
Target architectures

env: [CGO_ENABLED=0]
Build environment variables

ldflags: [-s -w -X main.version={{.Version}}]
Linker flags with version stamp

ARCHIVES AND CHECKSUMS

formats: [tar.gz]
Archive formats (tar.gz is default)

name_template: "{{ .ProjectName }}_{{ .Version }}_{{ .Os }}_{{ .Arch }}"
Archive file name

format_overrides: [{goos: windows, formats: [zip]}]
Zip for Windows only

CHANGELOG AND RELEASE

changelog: filters: exclude: ["^docs:"]
Hide commits matching a regex

release: prerelease: auto
Mark rc/beta tags as pre-releases

TEMPLATES AND SIGNING

{{ .Version }}
Version without the leading v

RECIPES

Minimal .goreleaser.yaml

```
version: 2
builds:
  - env: [CGO_ENABLED=0]
    goos: [linux, darwin, windows]
    goarch: [amd64, arm64]
    ldflags: [-s -w -X
main.version={{.Version}}]
archives:
  - formats: [tar.gz]
    format_overrides:
      - {goos: windows, formats:
[zip]}
```