Go in der Praxis

Take GitHub to the command line

GitHub CLI brings GitHub to your terminal. Free and open source.

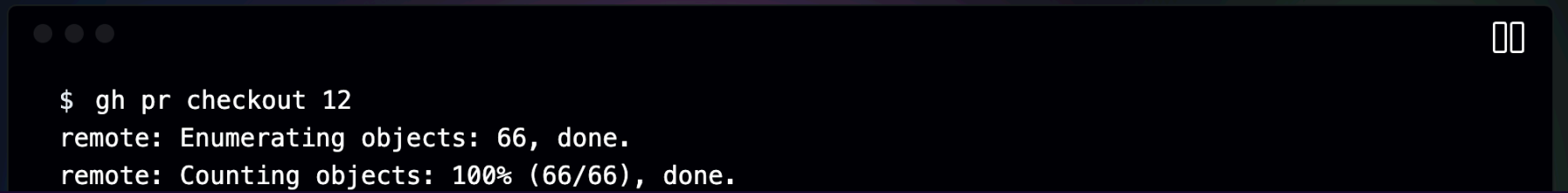
```
brew install gh
```

or **Download for Mac**

View installation instructions →

$ gh pr checkout

Check out pull requests locally.



gh Github CLI

149.298 Zeilen Code

490 Contributors

1.  CLI Tools
2.  Web App
3.  Container Workloads

5 Fakten zu Go

1. statisches Typsystem
2. Garbage Collection
3. keine Vererbung
4. Concurrency eingebaut
5. native Ausführung
   Linux, Win, z/OS, 386, amd64, ARM, wasm, ...

CLI Tool
Go Proverbs

Go Proverbs

18 Prinzipien zur Go Entwicklung
von Rob Pike 2015 vorgestellt
"Clear is better than clever."

CLI Proverbs ohne Argumente

```
> progo-cli
```

```
Clear is better than clever.
```

CLI Projekt aufsetzen

```
// 1. Go Modul erstellen (erzeugt go.mod)
> go mod init crossnative.com/progo-cli
```

```
// 2. Datei main.go erstellen
package main

import "fmt"

func main() {
    fmt.Println("Hello Go Proverbs!")
}
```

Ausführen

```
go build . // 1. Code kompilieren
./progo-cli      // 2. Binary ausführen
```

```
go run .   // Code kompilieren und ausführen
```


Entwicklung
GO
JetBrains
GoLand
Visual
Studio Code
Vim
Vim Go

Proverbs als Go Modul

pkg.go.dev/github.com/jboursiquot/go-proverbs@v0.0.2

proverbs package module

Version: v0.0.2 Latest

Jump to ... f

README

Documentation

- Overview
- Index
- Constants
- Variables
- Functions
- Types

Source Files

Index

Types

type ErrNthProverbNotFound

```
type ErrNthProverbNotFound struct {
    N int
}
```

ErrNthProverbNotFound is returned when the requested nth

func (*ErrNthProverbNotFound) Error

```
func (e *ErrNthProverbNotFound) Error() string
```

type Proverb

```
type Proverb struct {
    Saying string
    Link   string
}
```

Proverbs Modul nutzen

```
// 1. Proverbs Modul Dependency
> go get github.com/jboursiquot/go-proverbs
```

```
// 2. Go Modul Descriptor go.mod
module crossnative.com/progo-cli

go 1.23

require github.com/jboursiquot/go-proverbs v0.0.2
```

Zufälliges Proverb ausgeben

```go
package main

import (
    "fmt"
    "github.com/jboursiquot/go-proverbs"
)

func main() {
    fmt.Println(proverbs.Random())
}
```

```
// Ausgabe `go run .`
&{Make the zero value useful. http://youtube.com/322}
```

💡 Structs und Pointer

```go
package main

import (
    "fmt"
    "github.com/jboursiquot/go-proverbs"
)

func main() {
    // Pointer Variable auf Proverb Struct
    var p *proverbs.Proverb = proverbs.Random()

    fmt.Println(p)
}
```

```go
// Struct statt Klasse
type Proverb struct {
    Saying string
    Link   string
}
```

Zufälliges Proverb ausgeben

```go
package main

import (
    "fmt"
    "github.com/jboursiquot/go-proverbs"
)

func main() {
    var p *proverbs.Proverb = proverbs.Random()

    // Zugriff auf Property des Structs
    fmt.Println(p.Saying)
}
```

```
// Ausgabe `go run .`
Make the zero value useful.
```

CLI Proverbs mit Count

```
> pro -count=3
```

```
Clear is better than clever.
Documentation is for users.
Don't panic.
```

Flag für Count

```go
import (
    "flag"
    "fmt"
    "github.com/jboursiquot/go-proverbs"
)

func main() {
    // 1. Flags definieren
    var count int
    flag.IntVar(&count, "count", 1, "proverb count")

    // 2. Flags parsen
    flag.Parse()

    // 3. Ausgabe in Schleife
    for i := 0; i < count; i++ {
        var p *proverbs.Proverb = proverbs.Random()
        fmt.Println(p.Saying)
    }
}
```

```
// Compile mit `go build .`, Ausgabe mit `progo-cli -count=2`
Make the zero value useful.
Reflection is never clear.
```

CLI Proverbs mit ungültigem Count

```
> progo-cli -count=
```

```
flag needs an argument: -count
Usage of ./progo-cli:
  -count int
        proverb count (default 1)
```

Flag für Count Refactored

```go
import (
    "flag"
    "fmt"
    "github.com/jboursiquot/go-proverbs"
)

func main() {
    // 1. Flags definieren
    var count int
    flag.IntVar(&count, "count", 1, "proverb count")

    // 2. Flags parsen
    flag.Parse()

    // 3. Ausgabe in Schleife
    for range count {
        p := proverbs.Random()
        fmt.Println(p.Saying)
    }
}
```

```
// Compile mit `go build .`, Ausgabe mit `pro -count=2`
Make the zero value useful.
Reflection is never clear.
```

Cobra

About

Cobra is a CLI framework for Go. It contains a library for creating powerful modern CLI applications and a tool to rapidly generate Cobra based applications and command files.

It was created by Go team member, spf13 for hugo and has been adopted by the most popular Go projects.

Cobra provides:

- Easy subcommand-based CLIs: `app server`, `app fetch`, etc.
- Fully POSIX-compliant flags (including short & long versions)
- Nested subcommands
- Global, local and cascading flags
- Easy generation of applications & commands with `cobra init appname` & `cobra add cmdname`
- Intelligent suggestions (`app srver`... did you mean `app server`?)
- Automatic help generation for commands and flags
- Automatic help flag recognition of `-h`, `--help`, etc.
- Automatically generated bash autocomplete for your application
- Automatically generated man pages for your application
- Command aliases so you can change things without breaking them
- The flexibility to define your own help, usage, etc.
- Optional tight integration with viper for 12-factor apps

↑

🐍 Cobra Setup

```
# Aufbau Cobra CLIs
progo-cli {command} {subcommand} {args..} {flags..}
```

```
# Cobra CLI installieren
go install github.com/spf13/cobra-cli@latest
```

Cobra Setup

```
# Cobra Projekt aufsetzen
cobra-cli init
```

```
# Projektstruktur
├── cmd
│   ├── root.go
│   ├── cmd1.go
│   ├── print.go
│   └── {...}.go
└── main.go
```

Cobra Print Cmd ohne Fehlerhandling

```go
var printCmd = &cobra.Command{
    Use:   "print",
    Short: "Prints some proverbs",
    Long:  `Nifty tool that prints the Proverbs in your terminal.`,
    Args:  cobra.MatchAll(cobra.MaximumNArgs(2), cobra.OnlyValidArgs),
    Run: func(cmd *cobra.Command, args []string) {
      // 1. count Flag abfragen
      count, _ := cmd.Flags().GetInt("count")

      // 2. Ausgabe in Schleife
      for range count {
        p := proverbs.Random()
        fmt.Println(p.Saying)
      }
    },
  }

func init() {
  rootCmd.AddCommand(printCmd)

  // Flags
  printCmd.Flags().IntP("count", "c", 1, "Count of proverbs to print.")
}
```

Cobra Print Cmd mit Fehlerhandling

```go
var printCmd = &cobra.Command{
    Use:   "print",
    Short: "Prints some proverbs",
    Long:  `Nifty tool that prints the Proverbs in your terminal.`,
    Args:  cobra.MatchAll(cobra.MaximumNArgs(2), cobra.OnlyValidArgs),
    RunE: func(cmd *cobra.Command, args []string) error {
      // 1. count Flag abfragen
      count, err := cmd.Flags().GetInt("count")
      if err != nil {
        // 1a. Return mit Fehler
        return err
      }

      // 2. Ausgabe in Schleife
      for range count {
        p := proverbs.Random()
        fmt.Println(p.Saying)
      }

      // 3. Return ohne Fehler
      return nil
    },
  }
```

CLI Release bauen

GoReleaser

Search

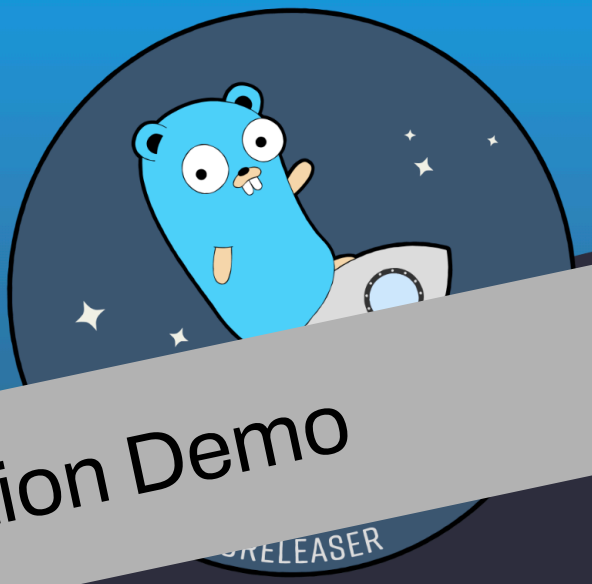
goreleaser/goreleaser
v2.4.4 13.8k 944

Home Getting Started Customization More resources Get Pro Blog

Release Go projects as fast and easily as possible!

GoReleaser is an open-source tool that simplifies the process of releasing your Go projects to multiple platforms, ensuring a consistent and reliable deployment experience.

Get started Get Pro



Goreleaser + Github Action Demo

👷 ...line your releases

...takes the hassle out of releasing your Go projects, allowing you to focus on building great software.

Cross-platform

Build and release your Go projects for and from multiple platforms, including Windows, macOS, and Linux.

CLI Tools

Standardlib Flags
Cobra CLI Lib wenn's komplexer wird
einfache Releases mit Goreleaser
Github, Kubernetes sind Go CLIs

Web App Cats

Cat API mit JSON

```
# Query cat API
curl -s http://localhost:8080/api/cats | jq
[
  {
    "name": "Ginger"
  }
]
```

Cat Web mit HTML

Cats App Home

Abyssinian

The Abyssinian is easy to care for, and a joy to have in your home. They're affectionate cats and love both people and other animals.

Aegean

Native to the Greek islands known as the Cyclades in the Aegean Sea, these are natural cats, meaning they developed without humans getting involved in their breeding. As a breed, Aegean Cats are rare, although they are numerous on their home islands. They are generally friendly toward people and can be excellent cats for families with children.

Set up Cats App

```
# 1. Verzeichnis erstellen
mkdir cats
cd cats
```

```
# 2. Go Modul aufsetzen
go mod init crossnative.com/cats
```

```
# 3. Go Datei erstellen
touch main.go
```

Cat API simple

```go
func catAPIHandler(w http.ResponseWriter, r *http.Request) {
  fmt.Fprintf(w, "Meow!")
  w.WriteHeader(http.StatusOK)
}

func main() {
  http.HandleFunc("GET /api/cats", catAPIHandler)
  http.ListenAndServe(":8080", nil)
}
```

Cat API mit JSON

```go
// 1. Struct mit JSON Tag definieren
type Cat struct {
  Name string `json:"name"`
}

func catAPIHandler(w http.ResponseWriter, r *http.Request) {
  // 2. Slice erstellen
  cats := make([]Cat, 1)

  // 3. Struct erstellen und einfügen
  cats[0] = Cat{Name: "Ginger"}

  // 4. JSON rendern
  json.NewEncoder(w).Encode(cats)
}

func main() {
  http.HandleFunc("GET /api/cats", catAPIHandler)
  http.ListenAndServe(":8080", nil)
}
```

Test Cat API `cat_api_test.go`

```go
func TestCatAPIHandler(t *testing.T) {
  // 1. Test Request erstellen
  req, _ := http.NewRequest("GET", "/api/cats", nil)

  // 2. HTTP Recorder erstellen (ist http.ResponseWriter)
  rec := httptest.NewRecorder()

  // 3. Handler aufrufen
  catAPIHandler(rec, req)

  // 4. Response prüfen
  if rec.Code != http.StatusOK {
    t.Errorf("got status %v expected %v", rec.Code, http.StatusOK)
  }
}
```

Test ausführen

```
1 go test -v ./...
2 === RUN   TestCatAPIHandler
3 --- PASS: TestCatAPIHandler (0.00s)
4 PASS
5 coverage: 50.0% of statements
6 ok      crossnative.com/cats      0.127s  coverage: 50.0% of statements
```

Web App Cats

Web App Cats

```go
func indexHandler(w http.ResponseWriter, r *http.Request) {
  tpl := template.Must(template.ParseFiles("index.html"))
  tpl.Execute(w, nil)
}

func main() {
  // 1. Router erzeugen
  router := http.NewServeMux()

  // 2. Handler registrieren
  router.HandleFunc("/", indexHandler)

  // 3. Server mit Router starten
  http.ListenAndServe(":8080", router)
}
```

File Server aus Binary

```go
//go:embed assets
var assets embed.FS

func main() {
  router := http.NewServeMux()
  router.HandleFunc("/", indexHandler)

  // Serve Files
  router.Handle("/assets/", http.FileServer(http.FS(assets)))

  http.ListenAndServe(":8080", router)
}
```

Index Handler Template mit Daten

```go
type Cat struct {
  Name string
}

func indexHandler(w ResponseWriter, r *Request) {
  // 1. Slice erstellen
  cat := make([]Cat, 1)

  // 2. Struct erstellen und einfügen
  cat[0] = Cat{Name: "Ginger"}

  // 3. Template rendern
  tpl := template.Must(template.ParseFiles("index.html"))
  tpl.Execute(w, cat)
}
```

```html
<body>
  <h1>Cats App</h1>
  {{ range . }}
    <h2>{{ .Name }}</h2>
  {{ end }}
</body>
```

TheCatAPI

PRICING DOCUMENTATION MORE APIS SHOW

The Cat API
Cats as a service.

cause everyday is a Caturday.

PI all about cat.
- Images. Breeds. Facts.

ET YOUR API KEY | READ OUR GUIDES


```
const headers = new Headers({
  "Content-Type": "application/json",
```

Index Handler mit Cats API

Query Cats API

```
GET https://api.thecatapi.com/v1/breeds?limit=5
```

```
[
   {
      "id": "abys",
      "name": "Abyssinian",
      "image": {
        "url": "https://cdn2.thecatapi.com/0XYvRd7oD.jpg"
      }
   },
   {
      "id": "aege",
      "name": "Aegean",
      "image": {
        "url": "https://cdn2.thecatapi.com/ozEvzdVM-.jpg"
      }
   },
   ...
]
```

Map JSON auf Struct

```
type Cat struct {
  ID    string `json:"id"`
  Name  string `json:"name"`
  Image struct {
    URL string `json:"url"`
  } `json:"image"`
}
```

Index Handler mit Cats API

```go
func indexHandler(w http.ResponseWriter, r *http.Request) {
  // 1. Cat API Aufruf (Fehler ignorieren)
  resp, _ := http.Get("https://api.thecatapi.com/v1/breeds?limit=5")

  // 2. Slice für Cat Structs
  cat := make([]Cat, 5)

  // 3. Response Body parsen
  defer resp.Body.Close()
  body, _ := ioutil.ReadAll(resp.Body)
  json.Unmarshal(body, &cat)

  // 4. Template rendern
  tpl.Execute(w, cat)
}
```

Index Handler mit Fehlerhandling 💣

```go
func indexHandler(w http.ResponseWriter, r *http.Request) {
  // 1. Cat API Aufruf
  resp, err := http.Get("https://api.thecatapi.com/v1/breeds?limit=5")
  //    Fehler behandeln
  if err != nil {
    http.Error(w, "Cats API error", http.StatusInternalServerError)
    return
  }

  // 2. Slice für Cat Structs
  cat := make([]Cat, 5)

  // 3. Response Body parsen
  defer resp.Body.Close()
  body, _ := ioutil.ReadAll(resp.Body)
  err := json.Unmarshal(body, &cat)
  // TODO: Fehler behandeln

  // 4. Template rendern
  err := tpl.Execute(w, cat)
  // TODO: Fehler behandeln
}
```

Web App Cats

Standardlib JSON und Router
HTTP Tests
Querschnittsfeatures durch Middleware

Container Workloads

Container Workloads

Kubernetes Services

Serverless Containers

GCP Cloud Run, AWS ECS, Azure Container Apps

Dockerfile für Cat API

```
1  # 1. App Builder
2  FROM golang AS builder
3  WORKDIR /app
4  ADD . /app
5  RUN CGO_EN                                    build/cats .
6
                  oless/static
                 rom=builder /app/build/cats /usr/bin/
10 EXPOSE 8080
11 ENTRYPOINT ["/usr/bin/cats"]
```

🚢 Google Distroless Image **8,5 MB**

Go Cat Container

Zahlen, Daten, Fakten

schlankes Container Image **8,5 MB**

Container Cold Start in **50-60 ms**

Hauptspeicher **25 MB**

Antwortzeit **15 ms**

CLI Tools

Projekt Setup
Flags
Cobra
Goreleaser

Web App

Web Server
JSON API
HTML Template
Unit Test
Fehlerhandling

Container Workloads

Docker Container
Resourcen Verbrauch

3 Gründe für Go

1. Einfach
2. Mächtig
3. Langweilig

Go liebt

Microservices

{ }

Serverless Functions

Kommandozeilen-Tools

DevOps und Cloud


GO!
GO!

Jan Stamer

jan.stamer@crossnative.com

Go in der Praxis