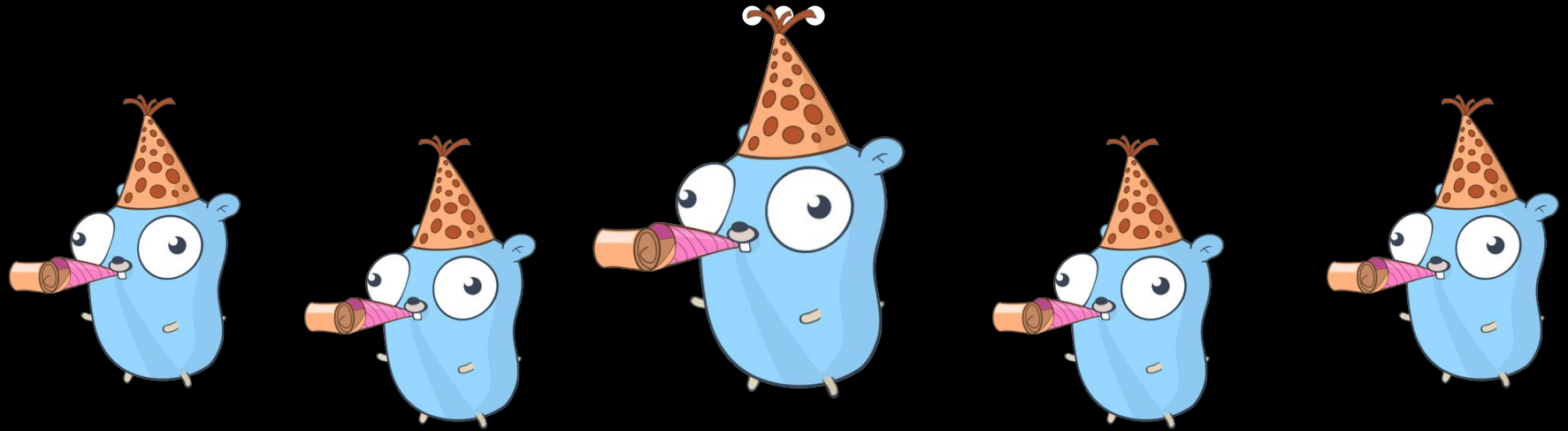
kevsersrca kev_src

GO ILE CLI UYGULAMA GELİŞTİRME



whoami

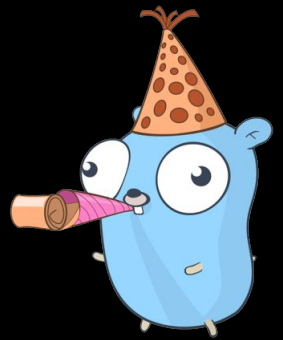
Software Engineer @Binalyze
formerly hepsiburada, wope (seo.do), digivity, netinernet(ilkbyte.com)

Social
Twitter: kev_src
Discord: kevser#5887

Github
kevsersrca

Blog
https://architecturecoding.notion.site/

Mail
kevser.sirca@gmail.com



```
ls -lah
```

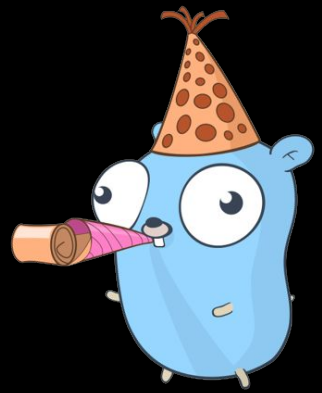
```
curl --proto '=https' --tlsv1.2 -sSf
https://sh.rustup.rs
```

```
git push
```

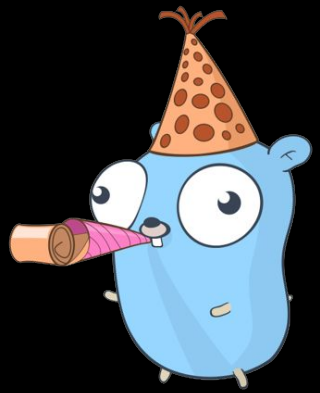
```
docker ps
```

```
kubectl proxy
```

```
aws s3 mb s3://bucket-name
```

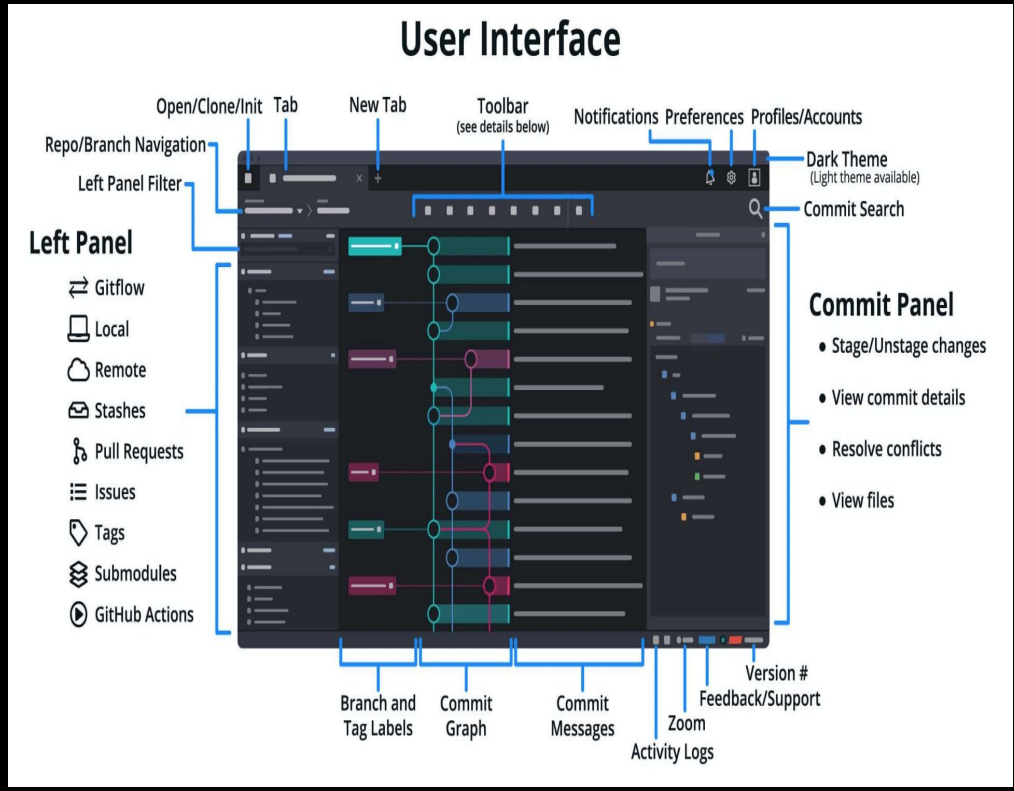



GUI Users
"You guys always act like you're better than me"
CLI Users



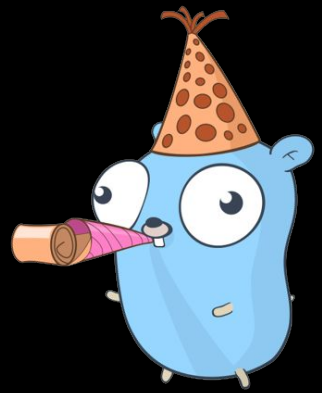
CLI vs GUI

```
kevsersirca:~/ $ git help
usage: git [--version] [--help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--super-prefix=<path>] [--config-env=<name>=<envvar>]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone             Clone a repository into a new directory
   init              Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add               Add file contents to the index
   mv                Move or rename a file, a directory, or a symlink
   restore           Restore working tree files
   rm                Remove files from the working tree and from the index
   sparse-checkout   Initialize and modify the sparse-checkout

examine the history and state (see also: git help revisions)
   bisect            Use binary search to find the commit that introduced a bug
   diff              Show changes between commits, commit and working tree, etc
   grep              Print lines matching a pattern
   log               Show commit logs
   show              Show various types of objects
   status            Show the working tree status

grow, mark and tweak your common history
   branch            List, create, or delete branches
   commit            Record changes to the repository
   merge             Join two or more development histories together
   rebase            Reapply commits on top of another base tip
   reset             Reset current HEAD to the specified state
   switch            Switch branches
   tag               Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
   fetch             Download objects and refs from another repository
   pull              Fetch from and integrate with another repository or a local branch
   push              Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.
```



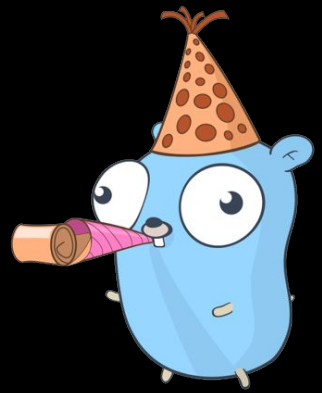
Why should we choose Go for CLI Application?

- Simple
- Modular
- Single-binary
- Cross-platform
- Performance
- Portability



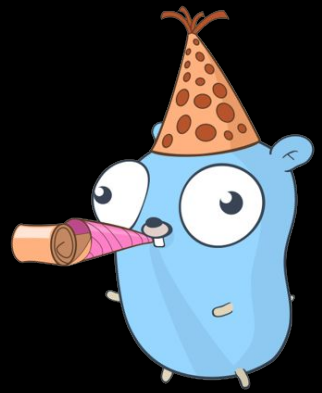
Why should we choose Go for CLI Application?

- **Error updating heroku cli on windows**
  #2045 opened on Dec 10, 2016 by carduque
- **Need help for Heroku installation.**
  #2043 opened on Dec 8, 2016 by divshriv
- **Heroku login fails on macOS**
  #2039 opened on Nov 27, 2016 by kalyandechiraju
- **heroku login fails on OS X**
  #2036 opened on Nov 17, 2016 by malafeev



Why should we choose Go for CLI Application?

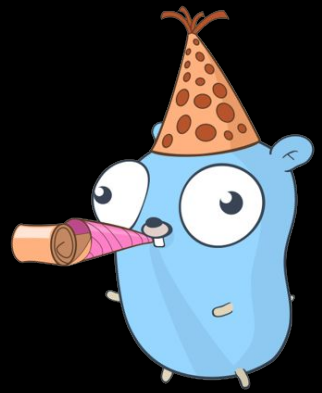
- **Pip install fails AWS CLI 1.7.34 wants simpleJSON in python 2.6** bug
  #1384 by dstuebe was closed on Jun 12, 2015
- **CloudWatch Logs not respecting proxy configuration** bug
  #1308 by philwill-nap was closed on Apr 28, 2015
- **Windows Error** bug
  #1226 by SimplyCorey was closed on Jun 19, 2015
- **Regression: Endpoint URL not honored in s3 commands** bug s3
  #1142 by kwinters was closed on Feb 23, 2015
- **Latest MSI install is broken on Windows 7 and 8.1 64-bit** bug
  #1053 by lloydcotten was closed on Dec 9, 2014
- **ImportError: cannot import name shlex_quote** bug
  #1051 by gergnz was closed on Dec 11, 2014



What are arguments?

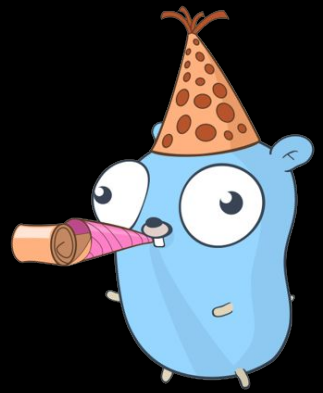
- https://dev.to/paulasantamaria/command-line-interfaces-structure-syntax-2533#1-introduction

```
NAME
    rm, unlink – remove directory entries

SYNOPSIS
    rm [-f | -i] [-dIRrvWx] file ...
```



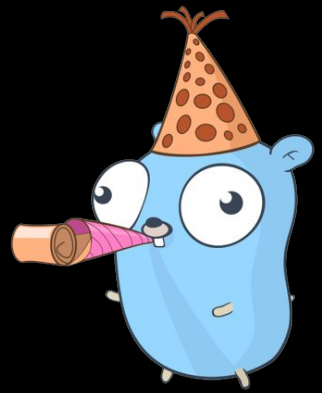
What are arguments?

```
package main

import (
    "fmt"
    "os"
)

func main() {
    // Program Name is always the first (implicit)
argument
    cmd := os.Args[0]

    fmt.Printf("Program Name: %s\n", cmd)
}
```



What are arguments?

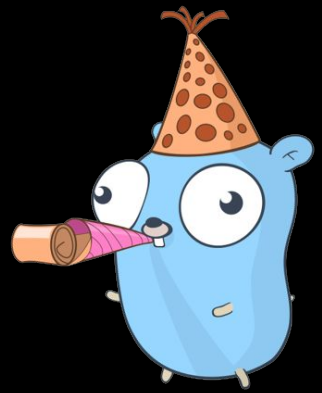
```
package main

import (
    "fmt"
    "os"
)

func main() {
    argCount := len(os.Args[1:])
    fmt.Printf("Total Arguments %d\n", argCount)
}
```



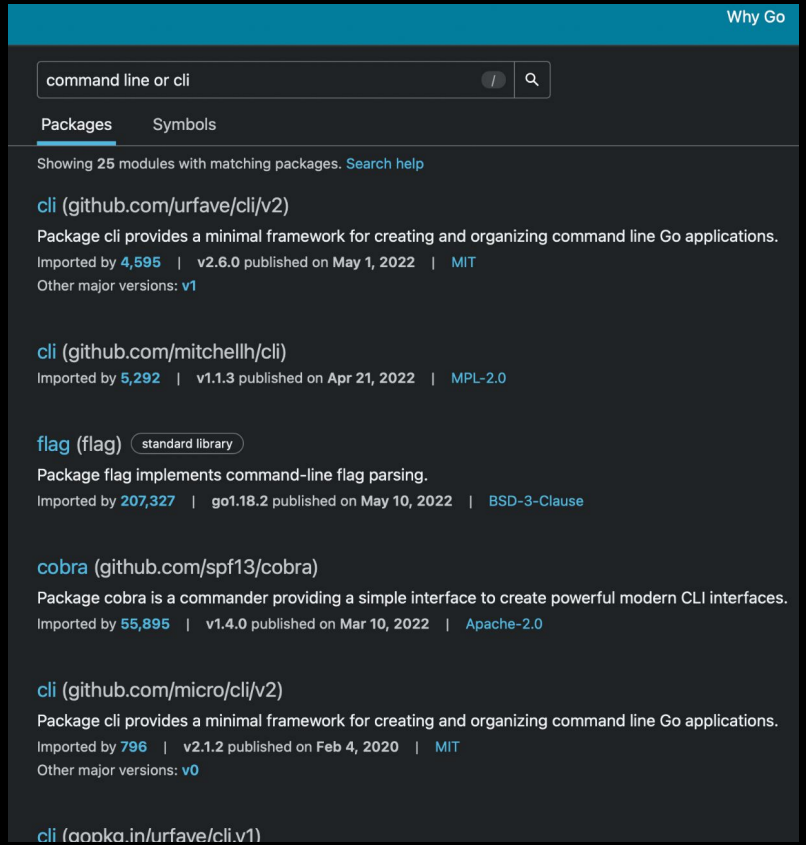
What are arguments?

```
func main() {
    for i, a := range os.Args[1:] {
        fmt.Printf("Argument %d is %s\n", i+1, a)
    }

}
```

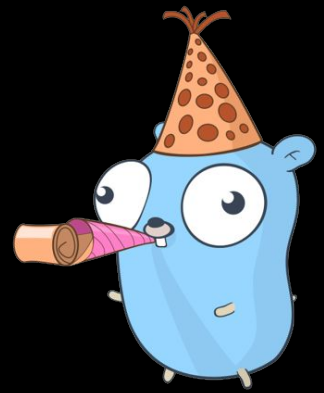
```
$ ./example -local u=admin --help
Argument 1 is -local
Argument 2 is u=admin
Argument 3 is --help
```



What are the alternatives?

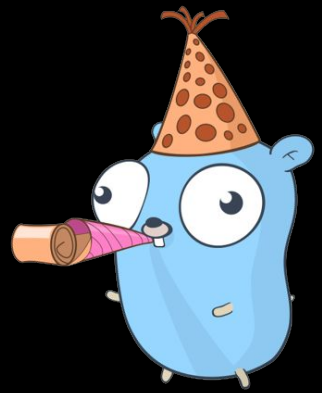


https://github.com/avelino/awesome-go#standard-cli



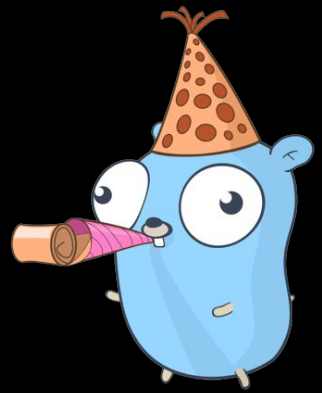
Flag Package - Flags

```
package main

import (
    "flag"
    "fmt"
)

func main() {
    var port int
    flag.IntVar(&port, "p", 8000, "specify port to use.  defaults to
8000.")
    flag.Parse()

    fmt.Printf("port = %d", port)
}
```



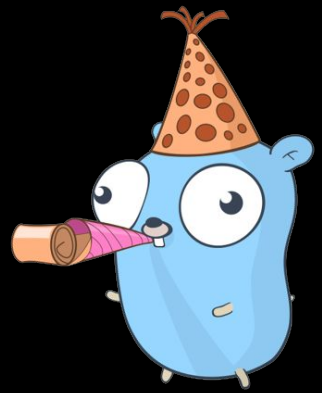
Flag Package - Flagset

```
func main() {
    args := os.Args

    f1 := flag.NewFlagSet("f1", flag.ContinueOnError)
    silent := f1.Bool("silent", false, "")

    f2 := flag.NewFlagSet("f2", flag.ContinueOnError)
    loud := f2.Bool("loud", false, "")

    switch args[1] {
    case "apply":
        if err := f1.Parse(args[2:]); err == nil {
            fmt.Println("apply", *silent)
        }
    case "reset":
        if err := f2.Parse(args[2:]); err == nil {
            fmt.Println("reset", *loud)
        }
    }
}
```



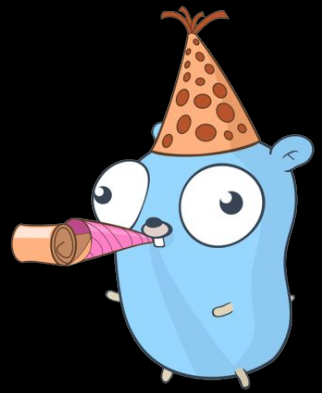
Flag

```
package main

import (
    "flag"
    "fmt"
    "os"
)

func main() {
    flag.Usage = func() {
        fmt.Printf("Usage of %s:\n", os.Args[0])
        fmt.Printf("example file1 file2 ...\n")
        flag.PrintDefaults()
    }
    flag.Parse()
}
```



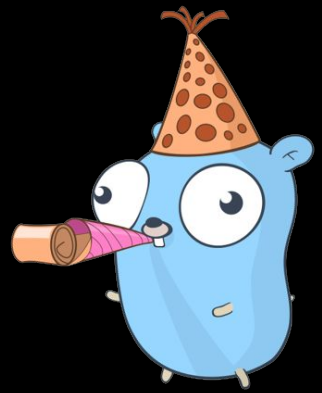
Guide

https://clig.dev



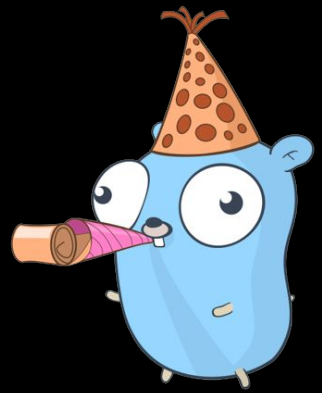
urfave/cli/v2 Package

```
app := &cli.App{}

app.Copyright = "2022 Ankara Gophers"
   app.Description = "Basic CRUD"
   app.Authors = []*cli.Author{
       {
           Name:  "Kevser Sirca",
           Email: "kevser.sirca@gmail.com",
       },
   }

app.EnableBashCompletion = true
app.Usage = "<program> <command> <options>"
```



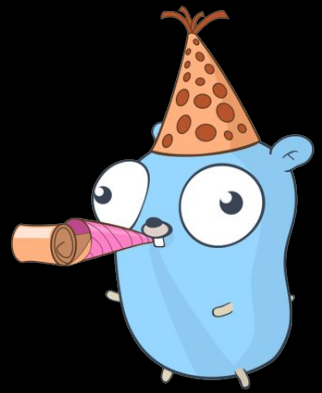
urfave/cli/v2 Package

```
err := app.Run(os.Args)
if err != nil {
    log.Fatal(err)
}
```



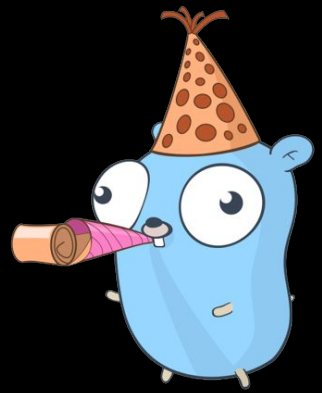
urfave/cli/v2 Package - Flags

```
app = &cli.App{
        Flags: []cli.Flag{
            &cli.StringFlag{
                Name:        "version",
                Aliases:     []string{"v"},
                Value:       "1.2.4",
                Usage:       "version of app",
                Destination: &version,
            },
        },
        Action: func(c *cli.Context) error {
            name := "someone"
            if version < "1.2.4" {
                fmt.Println("Please update", name)
            }
            return nil
        },
    }
```



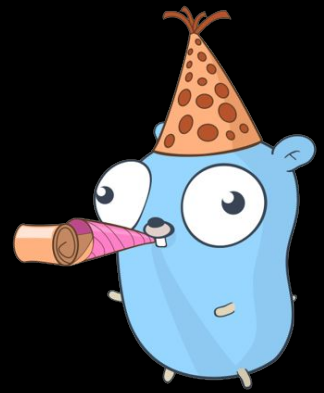
urfave/cli/v2 Package - Commands

```
addCmd := &cli.Command{
        Name:    "add",
        Aliases: []string{"a"},
        Usage:   "add a task to the list",
        Action: func(c *cli.Context) error {
            fmt.Println("added task: ",
c.Args().First())
            return nil
        },
    }
```



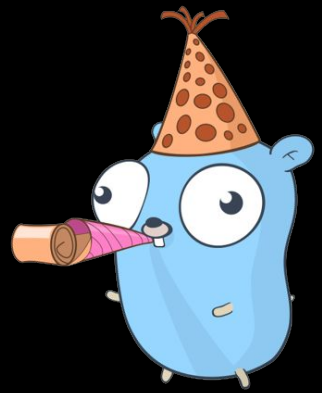
urfave/cli/v2 Package - Sub Commands

```
templateCmd := &cli.Command{
        Name:    "template",
        Aliases: []string{"t"},
        Usage:   "options for task templates",
        Subcommands: []*cli.Command{
            {
                Name:  "add",
                Usage: "add a new template",
                Action: func(c *cli.Context) error {
                    fmt.Println("new task template: ",
c.Args().First())
                    return nil
                },
            },
        },
    }
```



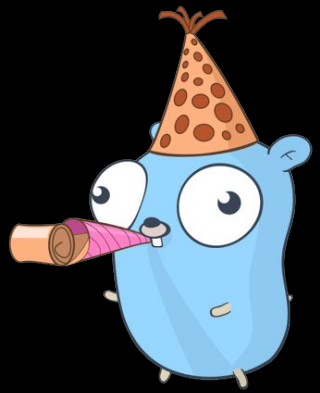
urfave/cli/v2 Package - Sub Commands

https://github.com/yakuter/gossl



Cobra package

https://umarcor.github.io/cobra



DEMO