

parser

- /
- /
- /
-

Content Menu

-
-
-
-
-
-
-

Command

```
package main

import (
    "context"
    "fmt"

    "github.com/gogf/gf/v2/frame/g"
    "github.com/gogf/gf/v2/os/gcmd"
    "github.com/gogf/gf/v2/os/gctx"
)

type cMain struct {
    g.Meta `name:"main"`
}

type cMainHttpInput struct {
    g.Meta `name:"http" brief:"start http server"`
}
type cMainHttpOutput struct{}

type cMainGrpcInput struct {
    g.Meta `name:"grpc" brief:"start grpc server"`
}
type cMainGrpcOutput struct{}

func (c *cMain) Http(ctx context.Context, in cMainHttpInput) (out
*cMainHttpOutput, err error) {
    fmt.Println("start http server")
    return
}

func (c *cMain) Grpc(ctx context.Context, in cMainGrpcInput) (out
*cMainGrpcOutput, err error) {
    fmt.Println("start grpc server")
    return
}

func main() {
    cmd, err := gcmd.NewFromObject(cMain{})
    if err != nil {
        panic(err)
    }
    cmd.Run(gctx.New())
}
```

Input//Outputnil

```
$ main
USAGE
    main COMMAND [OPTION]

COMMAND
    http      start http server
    grpc     start grpc server

DESCRIPTION
    this is the command entry for starting your process
```

http

```
$ main http
start http server
```

grpc

```
$ main grpc
start grpc server
```

/

httphttp

```

package main

import (
    "context"

    "github.com/gogf/gf/v2/frame/g"
    "github.com/gogf/gf/v2/net/ghttp"
    "github.com/gogf/gf/v2/os/gcmd"
    "github.com/gogf/gf/v2/os/gctx"
)

type cMain struct {
    g.Meta `name:"main" brief:"start http server"`
}

type cMainHttpInput struct {
    g.Meta `name:"http" brief:"start http server"`
    Name   string `v:"required" name:"NAME" arg:"true" brief:"server
name"`
    Port   int     `v:"required" short:"p" name:"port" brief:"port of
http server"`
}
type cMainHttpOutput struct{}


func (c *cMain) Http(ctx context.Context, in cMainHttpInput) (out
*cMainHttpOutput, err error) {
    s := g.Server(in.Name)
    s.BindHandler("/", func(r *ghttp.Request) {
        r.Response.Write("Hello world")
    })
    s.SetPort(in.Port)
    s.Run()
    return
}

func main() {
    cmd, err := gcmd.NewFromObject(cMain{})
    if err != nil {
        panic(err)
    }
    cmd.Run(gctx.New())
}

```

```

http

• NAME
• port p/port

v:"required"GoFrame

```

```
$ main http
arguments validation failed for command "http": The Name field is required
1. arguments validation failed for command "http"
   1). github.com/gogf/gf/v2/os/gcmd.newCommandFromMethod.func1
      /Users/john/Workspace/Go/GOPATH/src/github.com/gogf/gf/v2/os/gcmd
/gcmd_command_object.go:290
   2). github.com/gogf/gf/v2/os/gcmd.(*Command).doRun
      /Users/john/Workspace/Go/GOPATH/src/github.com/gogf/gf/v2/os/gcmd
/gcmd_command_run.go:120
   3). github.com/gogf/gf/v2/os/gcmd.(*Command).RunWithValueError
      /Users/john/Workspace/Go/GOPATH/src/github.com/gogf/gf/v2/os/gcmd
/gcmd_command_run.go:77
   4). github.com/gogf/gf/v2/os/gcmd.(*Command).RunWithValue
      /Users/john/Workspace/Go/GOPATH/src/github.com/gogf/gf/v2/os/gcmd
/gcmd_command_run.go:32
   5). github.com/gogf/gf/v2/os/gcmd.(*Command).Run
      /Users/john/Workspace/Go/GOPATH/src/github.com/gogf/gf/v2/os/gcmd
/gcmd_command_run.go:26
   6). main.main
      /Users/john/Workspace/Go/GOPATH/src/github.com/gogf/gf/.test/test.go:38
2. The Name field is required
```

Name/Port



GoFrameRunWithError

```
$ main http my-http-server -p 8199
2022-01-19 22:52:45.808 [DEBU] openapi specification is disabled

      SERVER      | DOMAIN     | ADDRESS    | METHOD | ROUTE
      |           |           |           |         |
MIDDLEWARE
-----|-----|-----|-----|-----|
-----|-----|-----|-----|-----|
my-http-server | default | :8199 | ALL     | /       | main.(*cMain).Http.
func1          |           |       |         |         |
-----|-----|-----|-----|-----|
-----|-----|-----|-----|-----|
my-http-server | default | :8199 | ALL     | /*     | github.com/gogf/gf
/v2/net/http/internalMiddlewareServerTracing | GLOBAL MIDDLEWARE
-----|-----|-----|-----|-----|
-----|-----|-----|-----|-----|
-----|-----|-----|-----|-----|
2022-01-19 22:52:45.810 66292: http server started listening on [:8199]
```

GoFrame <https://github.com/gogf/gf/tree/master/cmd/gf>

```

gofmt -d -l cmd.go
Project : gogf
File : cmd.go
109 type cmdInput struct {
110     g.Meta{name:"build", config:"gofit.build"}
111     File{string{name:"FILE", arg:"true", brief:"Building file path"}}
112     Arg{string{name:"arg", brief:"Building argument string", value:""}, valueSeparator:",", valueListSeparator:";"}
113     Verbose{string{short:"v", name:"verbose", brief:"Output binary version"}}
114     Arch{string{short:"a", name:"arch", brief:"Output binary architecture, multiple arch separated with ',' or '-' for cross-compiling, default is 'amd64'"}, value:"amd64", valueListSeparator:","}
115     Dirname{string{short:"o", name:"dirname", brief:"Output binary path, used when building static binary file"}}
116     Path{string{short:"p", name:"path", brief:"Output binary directory path, default is '.', like 'd:/bin'"}, value:"."}
117     Extra{string{short:"x", name:"extra", brief:"Extra context when building static binary file"}, value:""}, valueListSeparator:";"}
118     Cgo{bool{short:"c", name:"cgo", brief:"Enable or disable cgo feature, it's disabled by default", optional:true}}
119     Pack{string{name:"pack", brief:"Pack source built embedded variable into binary"}}
120 }
121 }
```

Command

name	-		namename
short	-		
usage	-		
brief	-		
arg	-		
orphan	-		bool
description	dc		
additional	ad		
examples	eg		
root	-		Meta
strict	-	//	Meta
config	-		Meta

HTTP/GRPC



MetaconfigGoFrame<https://github.com/gogf/gf/tree/master/cmd/gf>