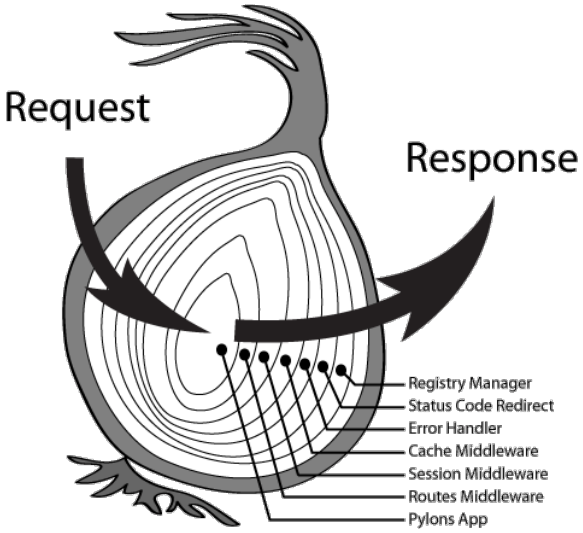
/

GFWebServerWebServer



HTTPHandlerFuncRequestMiddleware

```
func MiddlewareCORS(r *ghttp.Request) {
        r.Response.CORSDefault()
        r.Middleware.Next()
}

r.Middleware.Next()r.Middleware.Next()
```

```
func Middleware(r *ghttp.Request) {
        //
        r.Middleware.Next()
}
```

```
func Middleware(r *ghttp.Request) {
        r.Middleware.Next()
        //
}
```

https://godoc.org/github.com/gogf/gf/net/ghttp

```
// Server
func (s *Server) BindMiddleware(pattern string, handlers ...HandlerFunc)
func (s *Server) BindMiddlewareDefault(handlers ...HandlerFunc)

// BindMiddlewareDefault
func (s *Server) Use(handlers ...HandlerFunc)

// Domain
func (d *Domain) BindMiddleware(pattern string, handlers ...HandlerFunc)
func (d *Domain) BindMiddlewareDefault(handlers ...HandlerFunc)

// BindMiddlewareDefault
func (d *Domain) Use(handlers ...HandlerFunc)
```

Server/Domain""""

1. BindMiddleware
2. BindMiddlewareDefault/*
3. UseBindMiddlewareDefault

```
func (g *RouterGroup) Middleware(handlers ...HandlerFunc) *RouterGroup
```

Middleware

1.
2.
3.

> *gRPC*

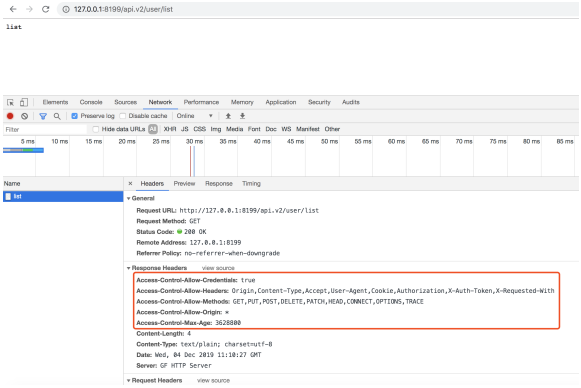
1

APIHeader

```
package main

import (
        "github.com/gogf/gf/frame/g"
        "github.com/gogf/gf/net/ghttp"
)

func MiddlewareCORS(r *ghttp.Request) {
        r.Response.CORSDefault()
        r.Middleware.Next()
}

func main() {
        s := g.Server()
        s.Group("/api.v2", func(group *ghttp.RouterGroup) {
                group.Middleware(MiddlewareCORS)
                group.ALL("/user/list", func(r *ghttp.Request) {
                        r.Response.Writeln("list")
                })
        })
        s.SetPort(8199)
        s.Run()
}
```

```
  SERVER  | DOMAIN  | ADDRESS | METHOD |       ROUTE       |
HANDLER      |     MIDDLEWARE
|---------|---------|---------|--------|-------------------
|--------------------|----------------------|
  default | default | :8199   | ALL    | /api.v2/user/list | main.main.func1.
1 | main.MiddlewareCORS
|---------|---------|---------|--------|-------------------
|--------------------|----------------------|
```

`group.Middleware(MiddlewareCORS)/api.v2` http://127.0.0.1:8199/api.v2/user/list `Header`



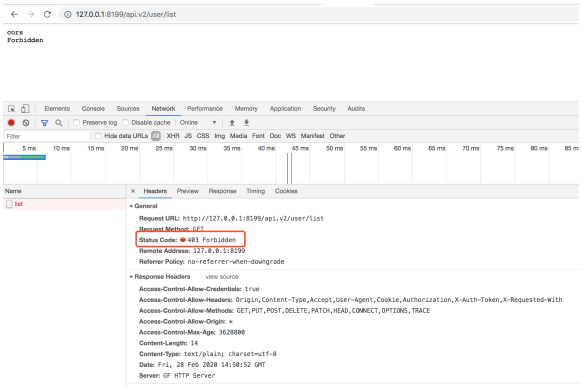
2

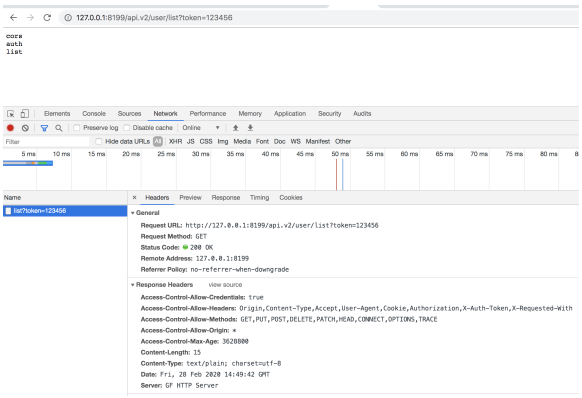
`token123456403 Forbidden`

```
package main

import (
	"net/http"

	"github.com/gogf/gf/frame/g"
	"github.com/gogf/gf/net/ghttp"
)

func MiddlewareAuth(r *ghttp.Request) {
	token := r.Get("token")
	if token == "123456" {
		r.Response.Writeln("auth")
		r.Middleware.Next()
	} else {
		r.Response.WriteStatus(http.StatusForbidden)
	}
}

func MiddlewareCORS(r *ghttp.Request) {
	r.Response.Writeln("cors")
	r.Response.CORSDefault()
	r.Middleware.Next()
}

func main() {
	s := g.Server()
	s.Group("/api.v2", func(group *ghttp.RouterGroup) {
		group.Middleware(MiddlewareCORS, MiddlewareAuth)
		group.ALL("/user/list", func(r *ghttp.Request) {
			r.Response.Writeln("list")
		})
	})
	s.SetPort(8199)
	s.Run()
}
```

```
  SERVER  | DOMAIN  | ADDRESS | METHOD |       ROUTE       |
HANDLER      |              MIDDLEWARE
|---------|---------|---------|--------|-------------------
|-------------------|-----------------------------------------|
  default | default | :8199   | ALL    | /api.v2/user/list | main.main.func1.
1 | main.MiddlewareCORS,main.MiddlewareAuth
|---------|---------|---------|--------|-------------------
|-------------------|-----------------------------------------|
```

MiddlewareCORSMiddlewareAuth http://127.0.0.1:8199/api.v2/user/list http://127.0.0.1:8199/api.v2/user/list?token=123456





3

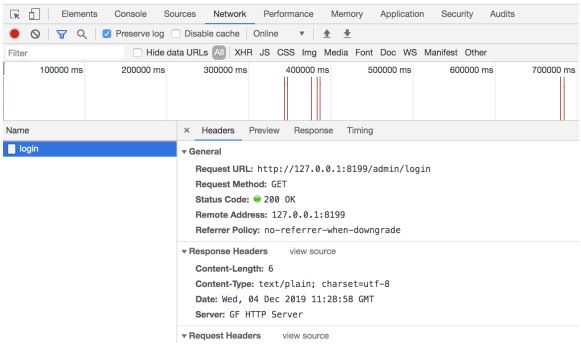
```
package main

import (
        "net/http"

        "github.com/gogf/gf/frame/g"
        "github.com/gogf/gf/net/ghttp"
)

func MiddlewareAuth(r *ghttp.Request) {
        token := r.Get("token")
        if token == "123456" {
                r.Middleware.Next()
        } else {
                r.Response.WriteStatus(http.StatusForbidden)
        }
}

func main() {
        s := g.Server()
        s.Group("/admin", func(group *ghttp.RouterGroup) {
                group.ALL("/login", func(r *ghttp.Request) {
                        r.Response.Writeln("login")
                })
                group.Group("/", func(group *ghttp.RouterGroup) {
                        group.Middleware(MiddlewareAuth)
                        group.ALL("/dashboard", func(r *ghttp.Request) {
                                r.Response.Writeln("dashboard")
                        })
                })
        })
        s.SetPort(8199)
        s.Run()
}
```

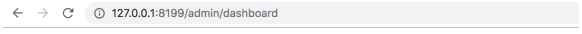
```
  SERVER  | ADDRESS | DOMAIN  | METHOD | P |      ROUTE       |
HANDLER       |     MIDDLEWARE
|---------|---------|---------|--------|---|------------------
|---------------------|---------------------|
  default |  :8199  | default |  ALL   | 2 | /admin/dashboard | main.main.
func1.2.1 | main.MiddlewareAuth
|---------|---------|---------|--------|---|------------------
|---------------------|---------------------|
  default |  :8199  | default |  ALL   | 2 | /admin/login     | main.main.
func1.1   |
|---------|---------|---------|--------|---|------------------
|---------------------|---------------------|
```

/admin/dashboardmain.MiddlewareAuth/admin/login URL

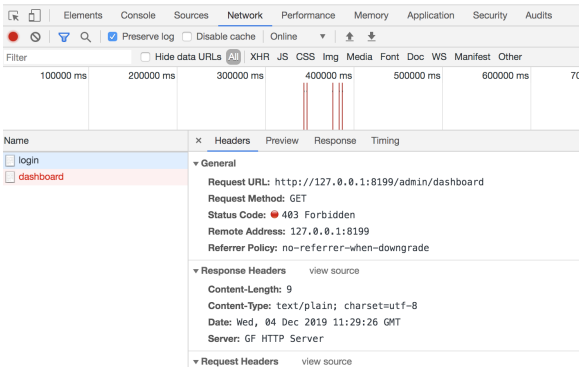
1. http://127.0.0.1:8199/admin/login
2. http://127.0.0.1:8199/admin/dashboard
3. http://127.0.0.1:8199/admin/dashboard?token=123456

```
login
```

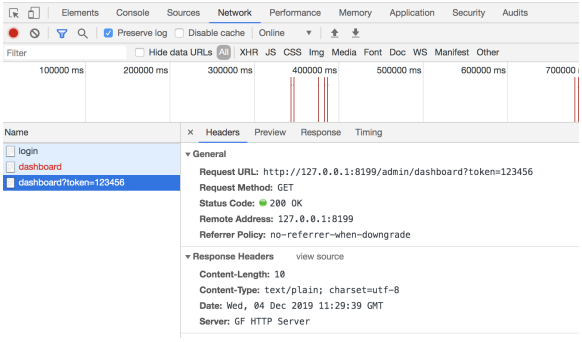





```
Forbidden
```




```
dashboard
```

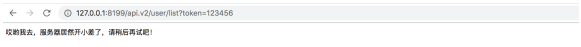


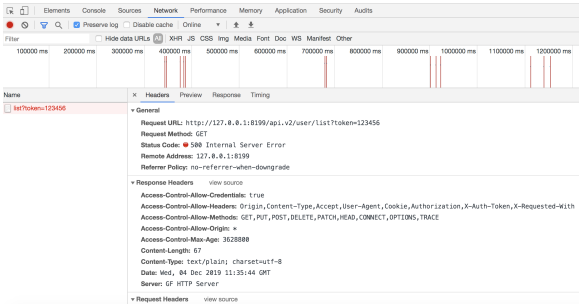
```
package main

import (
	"net/http"

	"github.com/gogf/gf/frame/g"
	"github.com/gogf/gf/net/ghttp"
)

func MiddlewareAuth(r *ghttp.Request) {
	token := r.Get("token")
	if token == "123456" {
		r.Middleware.Next()
	} else {
		r.Response.WriteStatus(http.StatusForbidden)
	}
}

func MiddlewareCORS(r *ghttp.Request) {
	r.Response.CORSDefault()
	r.Middleware.Next()
}

func MiddlewareErrorHandler(r *ghttp.Request) {
	r.Middleware.Next()
	if r.Response.Status >= http.StatusInternalServerError {
		r.Response.ClearBuffer()
		r.Response.Writeln("")
	}
}

func main() {
	s := g.Server()
	s.Use(MiddlewareCORS)
	s.Group("/api.v2", func(group *ghttp.RouterGroup) {
		group.Middleware(MiddlewareAuth, MiddlewareErrorHandler)
		group.ALL("/user/list", func(r *ghttp.Request) {
			panic("db error: sql is xxxxxxx")
		})
	})
	s.SetPort(8199)
	s.Run()
}
```

```
  SERVER  | DOMAIN  | ADDRESS | METHOD |       ROUTE       |
HANDLER       |                  MIDDLEWARE
|---------|---------|---------|--------|-------------------
|---------------------|-------------------------------------------------|
  default | default | :8199   | ALL    | /*                | main.
MiddlewareCORS | GLOBAL MIDDLEWARE
|---------|---------|---------|--------|-------------------
|---------------------|-------------------------------------------------|
  default | default | :8199   | ALL    | /api.v2/user/list | main.main.func1.
1   | main.MiddlewareAuth,main.MiddlewareErrorHandler
|---------|---------|---------|--------|-------------------
|---------------------|-------------------------------------------------|
```

JSON

http://127.0.0.1:8199/api.v2/user/list?token=123456

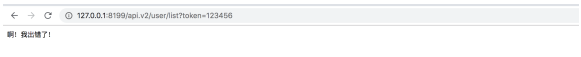


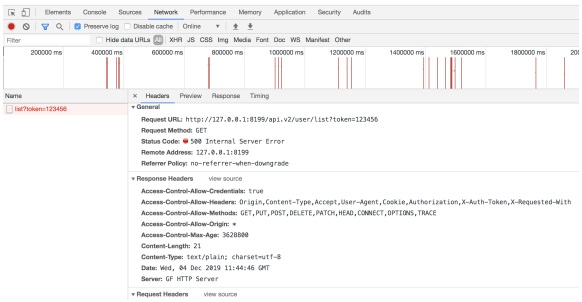


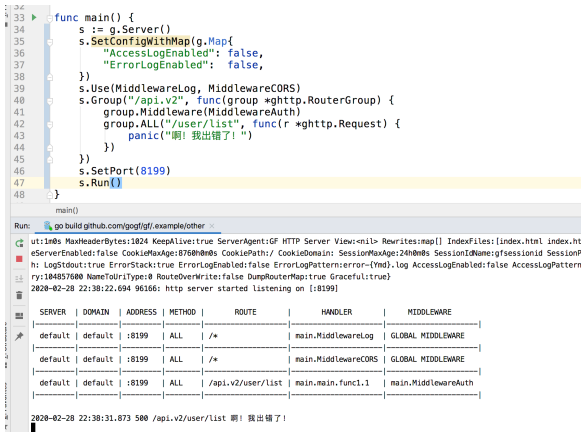
5

””404

```
package main

import (
	"net/http"

	"github.com/gogf/gf/frame/g"
	"github.com/gogf/gf/net/ghttp"
)

func MiddlewareAuth(r *ghttp.Request) {
	token := r.Get("token")
	if token == "123456" {
		r.Middleware.Next()
	} else {
		r.Response.WriteStatus(http.StatusForbidden)
	}
}

func MiddlewareCORS(r *ghttp.Request) {
	r.Response.CORSDefault()
	r.Middleware.Next()
}

func MiddlewareLog(r *ghttp.Request) {
	r.Middleware.Next()
	errStr := ""
	if err := r.GetError(); err != nil {
		errStr = err.Error()
	}
	g.Log().Println(r.Response.Status, r.URL.Path, errStr)
}

func main() {
	s := g.Server()
	s.SetConfigWithMap(g.Map{
		"AccessLogEnabled": false,
		"ErrorLogEnabled":  false,
	})
	s.Use(MiddlewareLog, MiddlewareCORS)
	s.Group("/api.v2", func(group *ghttp.RouterGroup) {
		group.Middleware(MiddlewareAuth)
		group.ALL("/user/list", func(r *ghttp.Request) {
			panic("")
		})
	})
	s.SetPort(8199)
	s.Run()
}
```







```go
func main() {
    s := g.Server()
    s.SetConfigWithMap(g.Map{
        "AccessLogEnabled": false,
        "ErrorLogEnabled":  false,
    })
    s.Use(MiddlewareLog, MiddlewareCORS)
    s.Group("/api.v2", func(group *ghttp.RouterGroup) {
        group.Middleware(MiddlewareAuth)
        group.ALL("/user/list", func(r *ghttp.Request) {
            panic("啊！我出错了！")
        })
    })
    s.SetPort(8199)
    s.Run()
}
```

```
main()
Run:  go build github.com/gogf/gf/.example/other
ut:1m0s MaxHeaderBytes:1024 KeepAlive:true ServerAgent:GF HTTP Server View:<nil> Rewrites:map[] IndexFiles:[index.html index.ht
eServerEnabled:false CookieMaxAge:8760h0m0s CookiePath:/ CookieDomain: SessionMaxAge:24h0m0s SessionIdName:gfsessionid SessionP
h: LogStdout:true ErrorStack:true ErrorLogEnabled:false ErrorLogPattern:error-{Ymd}.log AccessLogEnabled:false AccessLogPattern
ry:104857600 NameToUriType:0 RouteOverWrite:false DumpRouterMap:true Graceful:true}
2020-02-28 22:38:22.694 96166: http server started listening on [:8199]

  SERVER  | DOMAIN  | ADDRESS | METHOD |       ROUTE        |       HANDLER       |     MIDDLEWARE
|---------|---------|---------|--------|--------------------|---------------------|---------------------|
  default | default | :8199   | ALL    | /*                 | main.MiddlewareLog  | GLOBAL MIDDLEWARE
|---------|---------|---------|--------|--------------------|---------------------|---------------------|
  default | default | :8199   | ALL    | /*                 | main.MiddlewareCORS | GLOBAL MIDDLEWARE
|---------|---------|---------|--------|--------------------|---------------------|---------------------|
  default | default | :8199   | ALL    | /api.v2/user/list  | main.main.func1.1   | main.MiddlewareAuth
|---------|---------|---------|--------|--------------------|---------------------|---------------------|

2020-02-28 22:38:31.873 500 /api.v2/user/list 啊！我出错了！
```

`/api.v2`

http://127.0.0.1:8199/api.v2/user/list http://127.0.0.1:8199/api.v2/user/list?token=123456