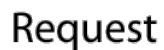


/



## Response

- Registry Manager
- Status Code Redirect
- Error Handler
- Cache Middleware
- Session Middleware
- Routes Middleware
- Pylons App

```
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>

## :

- 
- 
- 
- 
- 
- 
- 1
- 2
- 3
- 4
- 5

```
// 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"
)

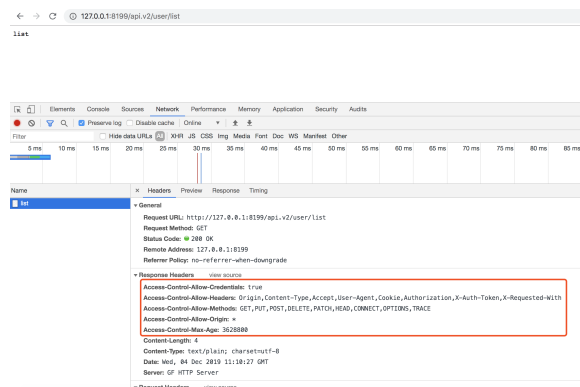
func MiddlewareCORS(r *ghttp.Request) {
    r.Response.CORSDefault()
    r.Middleware.Next()
}

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



The screenshot shows a web browser at the address `127.0.0.1:8199/api.v2/user/list` displaying the text `list`. Below the browser window, the network tab is open, showing a GET request to `http://127.0.0.1:8199/api.v2/user/list`. The response headers are expanded, showing:

```

Access-Control-Allow-Credentials: true
Access-Control-Allow-Headers: Origin, Content-Type, Accept, User-Agent, Cookie, Authorization, X-Auth-Token, X-Requested-With
Access-Control-Allow-Methods: GET, PUT, POST, DELETE, PATCH, HEAD, CONNECT, OPTIONS, TRACE
Access-Control-Allow-Origin: *
Access-Control-Max-Age: 3600000
Content-Length: 4
Content-Type: text/plain; charset=utf-8
Date: Wed, 24 Dec 2019 11:18:27 GMT
Server: GF HTTP Server

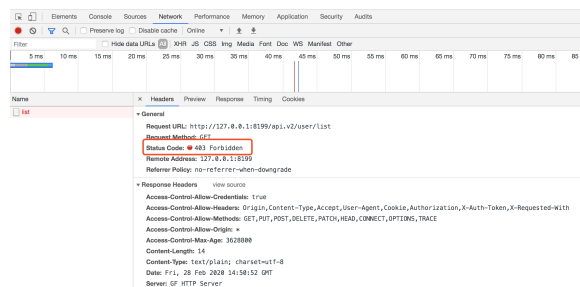
```

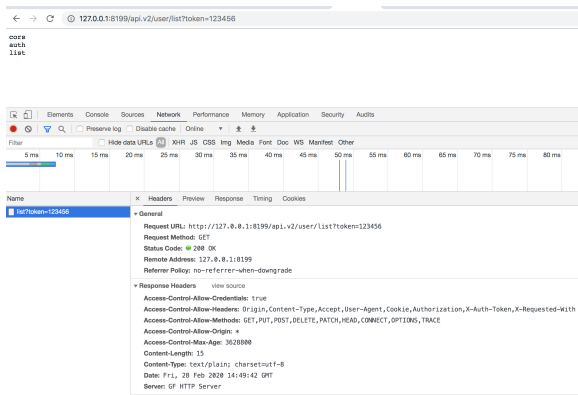
2

token123456403 Forbidden

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

← → ↻ 127.0.0.1:8199/api.v2/user/list





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
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>

127.0.0.1:8199/admin/login

login

Network tab showing the login request. The request is a GET to http://127.0.0.1:8199/admin/login with status 200 OK.

Name	Headers	Preview	Response	Timing
login	<p><b>General</b></p> <p>Request URL: http://127.0.0.1:8199/admin/login Request Method: GET Status Code: 200 OK Remote Address: 127.0.0.1:8199 Referrer Policy: no-referrer-when-downgrade</p> <p><b>Response Headers</b></p> <p>Content-Length: 6 Content-Type: text/plain; charset=utf-8 Date: Wed, 04 Dec 2019 11:28:58 GMT Server: GF HTTP Server</p> <p><b>Request Headers</b></p>			

127.0.0.1:8199/admin/dashboard

Forbidden

Network tab showing the dashboard request. The request is a GET to http://127.0.0.1:8199/admin/dashboard with status 403 Forbidden.

Name	Headers	Preview	Response	Timing
login				
dashboard	<p><b>General</b></p> <p>Request URL: http://127.0.0.1:8199/admin/dashboard Request Method: GET Status Code: 403 Forbidden Remote Address: 127.0.0.1:8199 Referrer Policy: no-referrer-when-downgrade</p> <p><b>Response Headers</b></p> <p>Content-Length: 9 Content-Type: text/plain; charset=utf-8 Date: Wed, 04 Dec 2019 11:29:26 GMT Server: GF HTTP Server</p> <p><b>Request Headers</b></p>			

127.0.0.1:8199/admin/dashboard?token=123456

dashboard

Network tab showing the dashboard request with a token. The request is a GET to http://127.0.0.1:8199/admin/dashboard?token=123456 with status 200 OK.

Name	Headers	Preview	Response	Timing
login				
dashboard				
dashboard?token=123456	<p><b>General</b></p> <p>Request URL: http://127.0.0.1:8199/admin/dashboard?token=123456 Request Method: GET Status Code: 200 OK Remote Address: 127.0.0.1:8199 Referrer Policy: no-referrer-when-downgrade</p> <p><b>Response Headers</b></p> <p>Content-Length: 10 Content-Type: text/plain; charset=utf-8 Date: Wed, 04 Dec 2019 11:29:39 GMT Server: GF HTTP Server</p> <p><b>Request Headers</b></p>			

```

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 xxxxxxxx")
        })
    })
    s.SetPort(8199)
    s.Run()
}

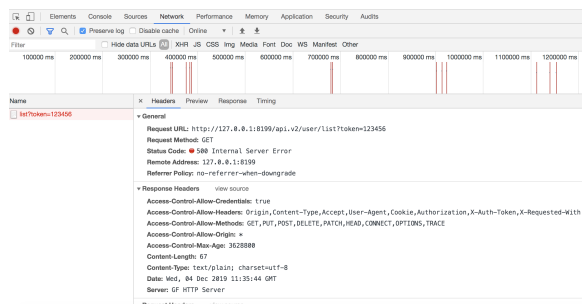
```

SERVER HANDLER	DOMAIN	ADDRESS	METHOD	ROUTE	
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>

127.0.0.1:8199/api.v2/user/list?token=123456  
 校验失败，服务器居然开小差了，请再试试吧！



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)
    }
}

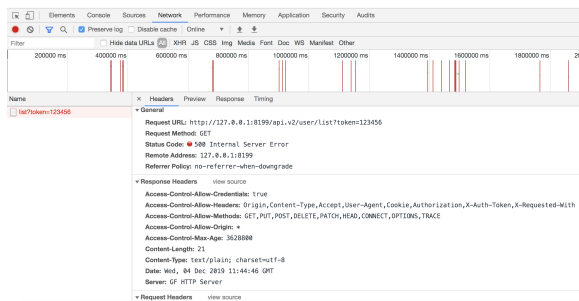
func MiddlewareCORS(r *ghttp.Request) {
    r.Response.CORSDefault()
    r.Middleware.Next()
}

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()
}
```

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

啊！我出错了！





```
33 func main() {
34     s := g.Server()
35     s.SetConfigWithMap(g.Map{
36         "AccessLogEnabled": false,
37         "ErrorLogEnabled": false,
38     })
39     s.Use(MiddlewareLog, MiddlewareCORS)
40     s.Group("/api.v2", func(group *ghttp.RouterGroup) {
41         group.Middleware(MiddlewareAuth)
42         group.ALL("/user/list", func(r *ghttp.Request) {
43             panic("啊！我出错了！")
44         })
45     })
46     s.SetPort(8199)
47     s.Run()
48 }
```

Run: go build github.com/gogitgl/example/other

ut:180s MaxHeaderBytes:1024 KeepAlive:true ServerAgent:GOF HTTP Server View:nil> Routes:map[] IndexFiles:[index.html index.ht  
sServerEnabled:false CookieMaxAge:876000ms CookiePath:/ CookieDomain: SessionMaxAge:30000ms SessionName:gofessionid SessionP  
ri:LogStdout:true ErrorStack:true ErrorLogEnabled:false ErrorLogPattern:error-(YmD).log AccessLogEnabled:false AccessLogPattern  
ry:184807680 NameOfUriType:# RouteOverwrite:false DumpRouterMap:true Graceful:true

2020-02-28 22:38:22.094 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 22:38:31.073 580 /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>