

-gfsnotify

gfsnotify/

```
import "github.com/gogf/gf/v2/os/gfsnotify"
```

<https://pkg.go.dev/github.com/gogf/gf/v2/os/gfsnotify>

gfsnotifyAddRemove

New*gfsnotify.Event

Content Menu

-
-
- [fs.inotify.](#)
[max_user_instancesfs.](#)
[inotify.max_user_watches](#)

```
package main

import (
    "github.com/gogf/gf/v2/os/gctx"
    "github.com/gogf/gf/v2/os/gfsnotify"
    "github.com/gogf/gf/v2/os/glog"
)

func main() {
    // /home/john/temp
    path := "/home/john/temp"
    ctx := gctx.New()
    _, err := gfsnotify.Add(path, func(event *gfsnotify.Event) {
        if event.IsCreate() {
            glog.Debug(ctx, " : ", event.Path)
        }
        if event.IsWrite() {
            glog.Debug(ctx, " : ", event.Path)
        }
        if event.IsRemove() {
            glog.Debug(ctx, " : ", event.Path)
        }
        if event.IsRename() {
            glog.Debug(ctx, " : ", event.Path)
        }
        if event.IsChmod() {
            glog.Debug(ctx, " : ", event.Path)
        }
        glog.Debug(ctx, event)
    }, true)

    // path
    // gfsnotify.Remove(path)

    if err != nil {
        glog.Fatal(ctx, err)
    } else {
        select {}
    }
}
```

/home/johngfsnotify.Add()

/home/john//gfsnotify

Remove/

/RemoveCallbackcallbackIdCallbackID

1

```
package main

import (
    "github.com/gogf/gf/v2/os/gctx"
    "github.com/gogf/gf/v2/os/gfsnotify"
    "github.com/gogf/gf/v2/os/glog"
    "github.com/gogf/gf/v2/os/gtimer"
    "time"
)

func main() {
    ctx := gctx.New()
    c1, err := gfsnotify.Add("/home/john/temp/log", func(event *gfsnotify.
Event) {
        glog.Debug(ctx, "callback1")
    })
    if err != nil {
        panic(err)
    }
    c2, err := gfsnotify.Add("/home/john/temp/log", func(event *gfsnotify.
Event) {
        glog.Debug(ctx, "callback2")
    })
    if err != nil {
        panic(err)
    }
    // 5c1c2
    gtimer.SetTimeout(5*time.Second, func() {
        gfsnotify.RemoveCallback(c1.Id)
        glog.Debug(ctx, "remove callback c1")
    })
    // 10c2
    gtimer.SetTimeout(10*time.Second, func() {
        gfsnotify.RemoveCallback(c2.Id)
        glog.Debug(ctx, "remove callback c2")
    })

    select {}
}
```

2

```

package main

import (
    "github.com/gogf/gf/v2/os/gctx"
    "github.com/gogf/gf/v2/os/gfsnotify"
    "github.com/gogf/gf/v2/os/glog"
    "github.com/gogf/gf/v2/os/gtimer"
    "time"
)

func main() {
    ctx := gctx.New()
    callback, err := gfsnotify.Add("/home/john/temp", func(event
*gfsnotify.Event) {
        glog.Debug(ctx, "callback")
    })
    if err != nil {
        panic(err)
    }

    //

    // 20
    gtimer.SetTimeout(20*time.Second, func() {
        gfsnotify.RemoveCallback(callback.Id)
        glog.Debug(ctx, "remove callback")
    })

    select {}
}

```

fs.inotify.max_user_instancesfs.inotify.
max_user_watches

*nixgfsnotifyinotify/

- fs.inotify.max_user_instancesinotifygfsnotify.NewWatcherWatcherinotify128
- fs.inotify.max_user_watchesinotifyinotify8192()