-

Queue.Pop

```
package main

import (
    "fmt"
    "time"
    "github.com/gogf/gf/v2/os/gtimer"
    "github.com/gogf/gf/v2/container/gqueue"
)

func main() {
    q := gqueue.New()

    // 1
    gtimer.SetInterval(time.Second, func() {
        v := gtime.Now().String()
        q.Push(v)
        fmt.Println("Push:", v)
    })

    // 3
    gtimer.SetTimeout(3*time.Second, func() {
        q.Close()
    })

    //
        for {
        if v := q.Pop(); v != nil {
            fmt.Println(" Pop:", v)
        } else {
            break
        }
    }

        // 32
        // Output:
        // Push: 2021-09-07 14:03:00
        //  Pop: 2021-09-07 14:03:00
        // Push: 2021-09-07 14:03:01
        //  Pop: 2021-09-07 14:03:01
}
```

Queue.C

```
package main

import (
        "context"
        "fmt"
        "time"

        _ "github.com/gogf/gf/contrib/drivers/mysql/v2"
        "github.com/gogf/gf/v2/container/gqueue"
        "github.com/gogf/gf/v2/os/gctx"
        "github.com/gogf/gf/v2/os/gtime"
        "github.com/gogf/gf/v2/os/gtimer"
)

func main() {
        queue := gqueue.New()
        gtimer.AddTimes(gctx.GetInitCtx(), time.Second, 3, func(ctx
context.Context) {
                queue.Push(gtime.Now().String())
        })
        for {
                select {
                case queueItem := <-queue.C:
                        fmt.Println(queueItem)

                case <-time.After(3 * time.Second):
                        fmt.Println("timeout, exit loop")
                        return
                }
        }
}
```

/

```
package main

import (
    "fmt"
    "time"
    "github.com/gogf/gf/v2/os/gtimer"
    "github.com/gogf/gf/v2/container/gqueue"
)

func main() {
        q := gqueue.New()

        for i := 0; i < 10; i++ {
                q.Push(i)
        }

        fmt.Println(q.Pop())
        fmt.Println(q.Pop())
        fmt.Println(q.Pop())

        // Output:
        // 0
        // 1
        // 2
}
```

```
package main

import (
    "fmt"
    "time"
    "github.com/gogf/gf/v2/os/gtimer"
    "github.com/gogf/gf/v2/container/gqueue"
)

func main() {
        q := gqueue.New()

        q.Push(1)
        q.Push(2)

        fmt.Println(q.Len())
        // sizelen
        fmt.Println(q.Size())

        // May Output:
        // 2
        // 2
}
```

```
package main

import (
    "fmt"
    "time"
    "github.com/gogf/gf/v2/os/gtimer"
    "github.com/gogf/gf/v2/container/gqueue"
)

func main() {
        q := gqueue.New()

        for i := 0; i < 10; i++ {
                q.Push(i)
        }

        fmt.Println(q.Pop())
        q.Close()
        fmt.Println(q.Pop())
        fmt.Println(q.Len())

        // Output:
        // 0
        // <nil>
        // 0
}
```

gqueueglist

gqueueglist

glistlist