

UDP-

gudpgudp.Conn

```
import "github.com/gogf/gf/v2/net/gudp"
```

<https://pkg.go.dev/github.com/gogf/gf/v2/net/gudp>

```
type Conn
func NewConn(raddr string, laddr ...string) (*Conn, error)
func NewConnByNetConn(udp *net.UDPConn) *Conn
func (c *Conn) Close() error
func (c *Conn) LocalAddr() net.Addr
func (c *Conn) Recv(length int, retry ...Retry) ([]byte, error)
func (c *Conn) RecvPkg(retry ...Retry) (result []byte, err error)
func (c *Conn) RecvPkgWithTimeout(timeout time.Duration, retry ...
Retry) ([]byte, error)
func (c *Conn) RecvWithTimeout(length int, timeout time.Duration,
retry ...Retry) ([]byte, error)
func (c *Conn) RemoteAddr() net.Addr
func (c *Conn) Send(data []byte, retry ...Retry) error
func (c *Conn) SendPkg(data []byte, retry ...Retry) error
func (c *Conn) SendPkgWithTimeout(data []byte, timeout time.Duration,
retry ...Retry) error
func (c *Conn) SendRecv(data []byte, receive int, retry ...Retry) ([]
byte, error)
func (c *Conn) SendRecvPkg(data []byte, retry ...Retry) ([]byte, error)
func (c *Conn) SendRecvPkgWithTimeout(data []byte, timeout time.
Duration, retry ...Retry) ([]byte, error)
func (c *Conn) SendRecvWithTimeout(data []byte, receive int, timeout
time.Duration, retry ...Retry) ([]byte, error)
func (c *Conn) SendWithTimeout(data []byte, timeout time.Duration,
retry ...Retry) error
func (c *Conn) SetDeadline(t time.Time) error
func (c *Conn) SetRecvBufferWait(d time.Duration)
func (c *Conn) SetRecvDeadline(t time.Time) error
func (c *Conn) SetSendDeadline(t time.Time) error
```

gudp.Connngtcp.Conn

gudp.ConnngtcpUDPgudp.ConnConn

Content Menu

•
•

```

package main

import (
    "fmt"
    "github.com/gogf/gf/v2/frame/g"
    "github.com/gogf/gf/v2/net/gudp"
    "github.com/gogf/gf/v2/os/gtime"
    "time"
)

func main() {
    // Server
    go gudp.NewServer("127.0.0.1:8999", func(conn *gudp.Conn) {
        defer conn.Close()
        for {
            data, err := conn.Recv(-1)
            if len(data) > 0 {
                if err := conn.Send(append([]byte("> "),
data...)); err != nil {
                    g.Log().Error(err)
                }
            }
            if err != nil {
                g.Log().Error(err)
            }
        }
    }).Run()

    time.Sleep(time.Second)

    // Client
    for {
        if conn, err := gudp.NewConn("127.0.0.1:8999"); err == nil
{
            if b, err := conn.SendRecv([]byte(gtime.
Datetime()), -1); err == nil {
                fmt.Println(string(b), conn.LocalAddr(),
conn.RemoteAddr())
            } else {
                g.Log().Error(err)
            }
            conn.Close()
        } else {
            g.Log().Error(err)
        }
        time.Sleep(time.Second)
    }
}

```

gtcp.Conn

```

> 2018-07-21 23:13:31 127.0.0.1:33271 127.0.0.1:8999
> 2018-07-21 23:13:32 127.0.0.1:45826 127.0.0.1:8999
> 2018-07-21 23:13:33 127.0.0.1:58027 127.0.0.1:8999
> 2018-07-21 23:13:34 127.0.0.1:33056 127.0.0.1:8999
> 2018-07-21 23:13:35 127.0.0.1:39260 127.0.0.1:8999
> 2018-07-21 23:13:36 127.0.0.1:33967 127.0.0.1:8999
> 2018-07-21 23:13:37 127.0.0.1:52359 127.0.0.1:8999
...

```