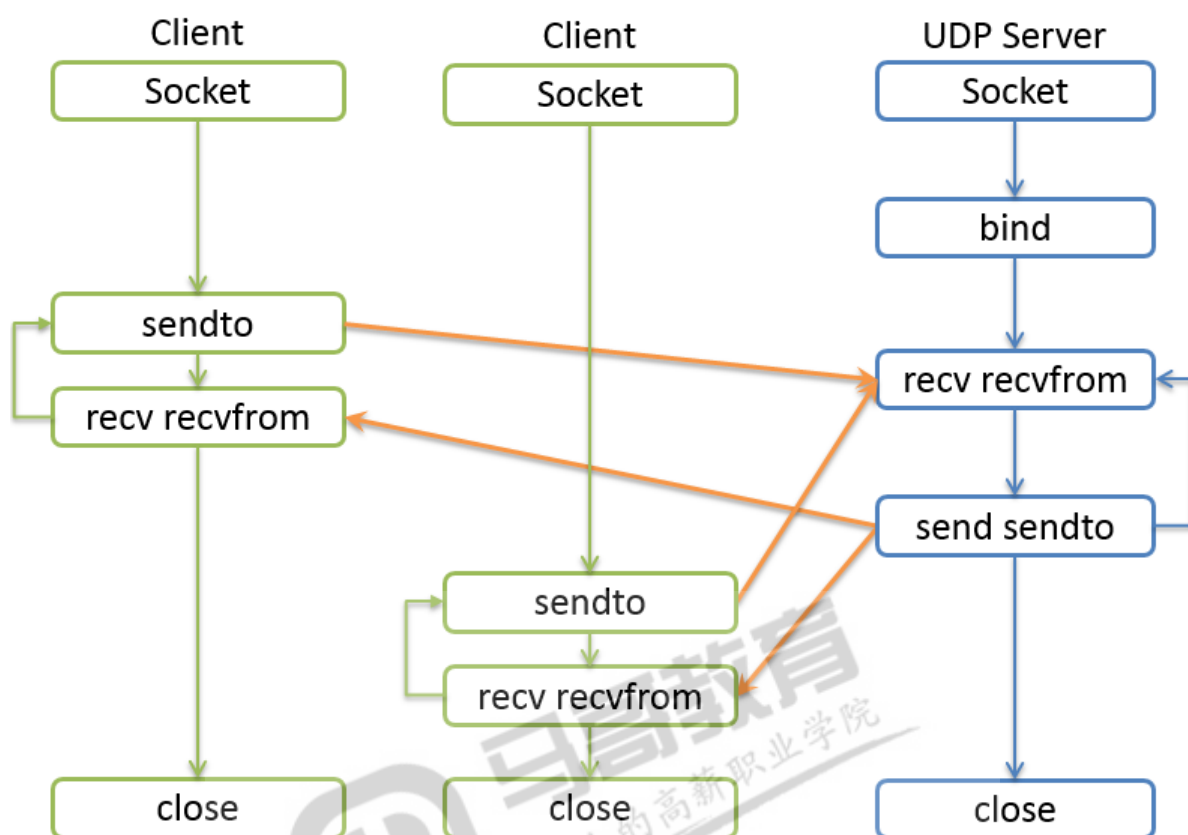


UDP编程

服务器端编程



- 创建socket对象。socket.SOCK_DGRAM
- 绑定IP和Port, bind()方法
- 传输数据
 - 接收数据, socket.recvfrom(bufsize[, flags]), 获得一个二元组(string, address)
 - 发送数据, socket.sendto(string, address) 发给某地址某信息
- 释放资源

```
1 package main
2
3 import (
4     "fmt"
5     "log"
6     "net"
7 )
8
9 func catchErr(err error) {
10     if err != nil {
11         log.Fatalln(err)
12     }
13 }
14
15 func main() {
16     laddr, err := net.ResolveUDPAddr("udp", "127.0.0.1:9999")
```

```

17     catchErr(err)
18     fmt.Println(laddr)
19     // 1 socket 2 bind
20     server, err := net.ListenUDP("udp", laddr)
21     catchErr(err)
22     defer server.Close()
23     // 不需要accept
24
25     // 3 收发
26     buffer := make([]byte, 1024)
27     n, raddr, err := server.ReadFromUDP(buffer)
28     fmt.Println(err)
29     msg := fmt.Sprintf("Client=%v, data=%s", raddr, string(buffer[:n]))
30     fmt.Println(msg)
31     server.WriteToUDP([]byte(msg), raddr)
32 }

```

使用协程改造一下。

```

1  package main
2
3  import (
4      "fmt"
5      "log"
6      "net"
7      "runtime"
8      "time"
9  )
10
11 func catchErr(err error) {
12     if err != nil {
13         log.Fatalf(err)
14     }
15 }
16
17 func main() {
18     laddr, err := net.ResolveUDPAddr("udp", "127.0.0.1:9999")
19     catchErr(err)
20     fmt.Println(laddr)
21     // 1 socket 2 bind
22     server, err := net.ListenUDP("udp", laddr)
23     catchErr(err)
24     defer server.Close()
25     // 不需要accept
26
27     // 3 收发
28     buffer := make([]byte, 1024)
29     exit := make(chan struct{})
30
31     go func() {
32         for {
33             server.SetReadDeadline(time.Now().Add(time.Second))
34             n, raddr, err := server.ReadFromUDP(buffer)
35             fmt.Printf("%T, %[1]v\n", err)
36             if err != nil {

```

```

37         if _, ok := err.(*net.OpError); !ok {
38             exit <- struct{}{}
39             return
40         }
41         continue
42     }
43     msg := fmt.Sprintf("Client=%v, data=%s", raddr,
string(buffer[:n]))
44     fmt.Println(msg)
45     server.WriteToUDP([]byte(msg), raddr)
46 }
47 }()
48
49 t := time.NewTicker(3 * time.Second)
50 for {
51     select {
52     case <-exit:
53         goto EXIT
54     case <-t.C:
55         fmt.Println(runtime.NumGoroutine(), "!!!")
56     }
57 }
58
59 EXIT:
60     fmt.Println("~~~~~")
61 }

```

客户端编程

- 创建socket对象。socket.SOCK_DGRAM
- 传输数据
 - 接收数据，socket.recvfrom(bufsize[, flags])，获得一个二元组(string, address)
 - 发送数据，socket.sendto(string, address) 发给某地址某信息
- 释放资源

```

1 package main
2
3 import (
4     "fmt"
5     "log"
6     "net"
7     "runtime"
8     "time"
9 )
10
11 func catchErr(err error) {
12     if err != nil {
13         log.Fatalln(err)
14     }
15 }
16
17 func main() {

```

```

18     raddr, err := net.ResolveUDPAddr("udp", "127.0.0.1:9999")
19     catchErr(err)
20
21     // 1 socket
22     conn, err := net.DialUDP("udp", nil, raddr)
23     catchErr(err)
24     defer conn.Close()
25     // 不需要accept
26
27     // 2 收发
28     exit := make(chan struct{})
29
30     go func() {
31         buffer := make([]byte, 1024)
32         for {
33             conn.SetReadDeadline(time.Now().Add(time.Second))
34             n, _, err := conn.ReadFromUDP(buffer)
35             if err != nil {
36                 if _, ok := err.(*net.OpError); !ok {
37                     exit <- struct{}{}
38                     return
39                 }
40                 continue
41             }
42             fmt.Println(buffer[:n])
43         }
44     }()
45
46     go func() {
47         var input string
48         for {
49             fmt.Scanln(&input)
50             if input == "exit" {
51                 exit <- struct{}{}
52                 return
53             }
54             i, err2 := conn.Write([]byte(input)) // 要用write, 因为DialUDP已经
知道了对端
55             fmt.Println(i, err2)
56         }
57     }()
58 }()
59
60 t := time.NewTicker(3 * time.Second)
61 for {
62     select {
63     case <-exit:
64         goto EXIT
65     case <-t.C:
66         fmt.Println(runtime.NumGoroutine(), "!!!")
67     }
68 }
69
70 EXIT:
71     fmt.Println("~~~~~")

```

