golang建立一个基本的分布式系统

> 黎跃春,孔壹学院创始人，微信，liyc1215

功能

- 能够发送/接收请求和响应
- 能够连接到集群
- 如果无法连接到群集（如果它是第一个节点），则可以作为主节点启动节点
- 每个节点有唯一的标识
- 能够在节点之间交换json数据包
- 接受命令行参数中的所有信息（将来在我们系统升级时将会很有用）

源码

```
package main

import (
    "fmt"
    "strconv"
    "time"
    "math/rand"
    "net"
    "flag"
    "strings"
    "encoding/json"
)

// 节点数据信息
type NodeInfo struct {

    // 节点ID，通过随机数生成
    NodeId int `json:"nodeId"`
    // 节点IP地址
    NodeIpAddr string `json:"nodeIpAddr"`
    // 节点端口
    Port string `json: "port"`
}

// 将节点数据信息格式化输出
//NodeInfo:{nodeId: 89423,nodeIpAddr: 127.0.0.1/8,port: 8001}
func (node *NodeInfo) String() string {
```

```
    return "NodeInfo:{ nodeId:" + strconv.Itoa(node.NodeId) + ",nodeIpAddr:"
 + node.NodeIpAddr + ",port:" + node.Port + "}"
}


/* 添加一个节点到集群的一个请求或者响应的标准格式 */
type AddToClusterMessage struct {
    // 源节点
    Source NodeInfo  `json:"source"`
    // 目的节点
    Dest NodeInfo  `json:"dest"`
    // 两个节点连接时发送的消息
    Message string  `json:"message"`
}

/* Request/Response 信息格式化输出 */
func (req AddToClusterMessage) String() string {
    return "AddToClusterMessage:{\n  source:" + req.Source.String() + ",\n
dest: " + req.Dest.String() + ",\n  message:" + req.Message + " }"
}

// cat vi go
// rm


func main()  {

    // 解析命令行参数
    makeMasterOnError := flag.Bool("makeMasterOnError", false, "如果IP地址没有
连接到集群中，我们将其作为Master节点.")
    clusterip := flag.String("clusterip", "127.0.0.1:8001", "任何的节点连接都连
接这个IP")
    myport := flag.String("myport", "8001", "ip address to run this node on.
 default is 8001.")
    flag.Parse() //解析

    fmt.Println(*makeMasterOnError)
    fmt.Println(*clusterip)
    fmt.Println(*myport)

    /* 为节点生成ID */
    rand.Seed(time.Now().UTC().UnixNano()) //种子
    myid := rand.Intn(99999999) // 随机

    //fmt.Println(myid)

    // 获取IP地址
```

```
    myIp,_ := net.InterfaceAddrs()
    fmt.Println(myIp[0])

    // 创建NodeInfo结构体对象
    me := NodeInfo{NodeId: myid, NodeIpAddr: myIp[0].String(), Port: *myport
}
    // 输出结构体数据信息
    fmt.Println(me.String())
    dest := NodeInfo{ NodeId: -1, NodeIpAddr: strings.Split(*clusterip, ":")
[0], Port: strings.Split(*clusterip, ":")[1]}

    /* 尝试连接到集群，在已连接的情况下并且向集群发送请求 */
    ableToConnect := connectToCluster(me, dest)

    /*
     * 监听其他节点将要加入到集群的请求
     */
    if ableToConnect || (!ableToConnect && *makeMasterOnError) {
        if *makeMasterOnError {fmt.Println("Will start this node as master."
)}
        listenOnPort(me)
    } else {
        fmt.Println("Quitting system. Set makeMasterOnError flag to make the
 node master.", myid)
    }


}



/*
 * 这是发送请求时格式化json包有用的工具
 * 这是非常重要的，如果不经过数据格式化，你最终发送的将是空白消息
 */
func getAddToClusterMessage(source NodeInfo, dest NodeInfo, message string)
(AddToClusterMessage){
    return AddToClusterMessage{
        Source: NodeInfo{
            NodeId: source.NodeId,
            NodeIpAddr: source.NodeIpAddr,
            Port: source.Port,
        },
        Dest: NodeInfo{
            NodeId: dest.NodeId,
            NodeIpAddr: dest.NodeIpAddr,
            Port: dest.Port,
        },
        Message: message,
```

```
    }
}


func connectToCluster(me NodeInfo, dest NodeInfo) (bool){
    /* 连接到socket的相关细节信息 */
    connOut, err := net.DialTimeout("tcp", dest.NodeIpAddr + ":" + dest.Port
, time.Duration(10) * time.Second)
    if err != nil {
        if _, ok := err.(net.Error); ok {
            fmt.Println("未连接到集群.", me.NodeId)
            return false
        }
    } else {
        fmt.Println("连接到集群. 发送消息到节点.")
        text := "Hi nody.. 请添加我到集群.."
        requestMessage := getAddToClusterMessage(me, dest, text)
        json.NewEncoder(connOut).Encode(&requestMessage)

        decoder := json.NewDecoder(connOut)
        var responseMessage AddToClusterMessage
        decoder.Decode(&responseMessage)
        fmt.Println("得到数据响应:\n" + responseMessage.String())

        return true
    }
    return false
}


func listenOnPort(me NodeInfo){
    /* 监听即将到来的消息 */
    ln, _ := net.Listen("tcp", fmt.Sprint(":" + me.Port))
    /* 接受连接 */
    for {
        connIn, err := ln.Accept()
        if err != nil {
            if _, ok := err.(net.Error); ok {
                fmt.Println("Error received while listening.", me.NodeId)
            }
        } else {
            var requestMessage AddToClusterMessage
            json.NewDecoder(connIn).Decode(&requestMessage)
            fmt.Println("Got request:\n" + requestMessage.String())

            text := "Sure buddy.. too easy.."
            responseMessage := getAddToClusterMessage(me, requestMessage.Sou
rce, text)
```

```
            json.NewEncoder(connIn).Encode(&responseMessage)
            connIn.Close()
        }
    }
}
```

运行程序

```
/Users/liyuechun/go
liyuechun:go yuechunli$ go install main
liyuechun:go yuechunli$ main
My details: NodeInfo:{ nodeId:53163002, nodeIpAddr:127.0.0.1/8, port:8001 }
不能连接到集群. 53163002
Quitting system. Set makeMasterOnError flag to make the node master. 5316300
2
liyuechun:go yuechunli$
```

获取相关帮助信息

```
$ ./bin/main -h
```

```
liyuechun:go yuechunli$ ./bin/main -h
Usage of ./bin/main:
  -clusterip string
        ip address of any node to connnect (default "127.0.0.1:8001")
  -makeMasterOnError
        make this node master if unable to connect to the cluster ip provide
d.
  -myport string
        ip address to run this node on. default is 8001. (default "8001")
liyuechun:go yuechunli$
```

启动Node1主节点

```
$ ./bin/main --makeMasterOnError
```

```
liyuechun:go yuechunli$ ./bin/main --makeMasterOnError
My details: NodeInfo:{ nodeId:82381143, nodeIpAddr:127.0.0.1/8, port:8001 }
未连接到集群. 82381143
```

```
Will start this node as master.
```

添加节点Node2到集群

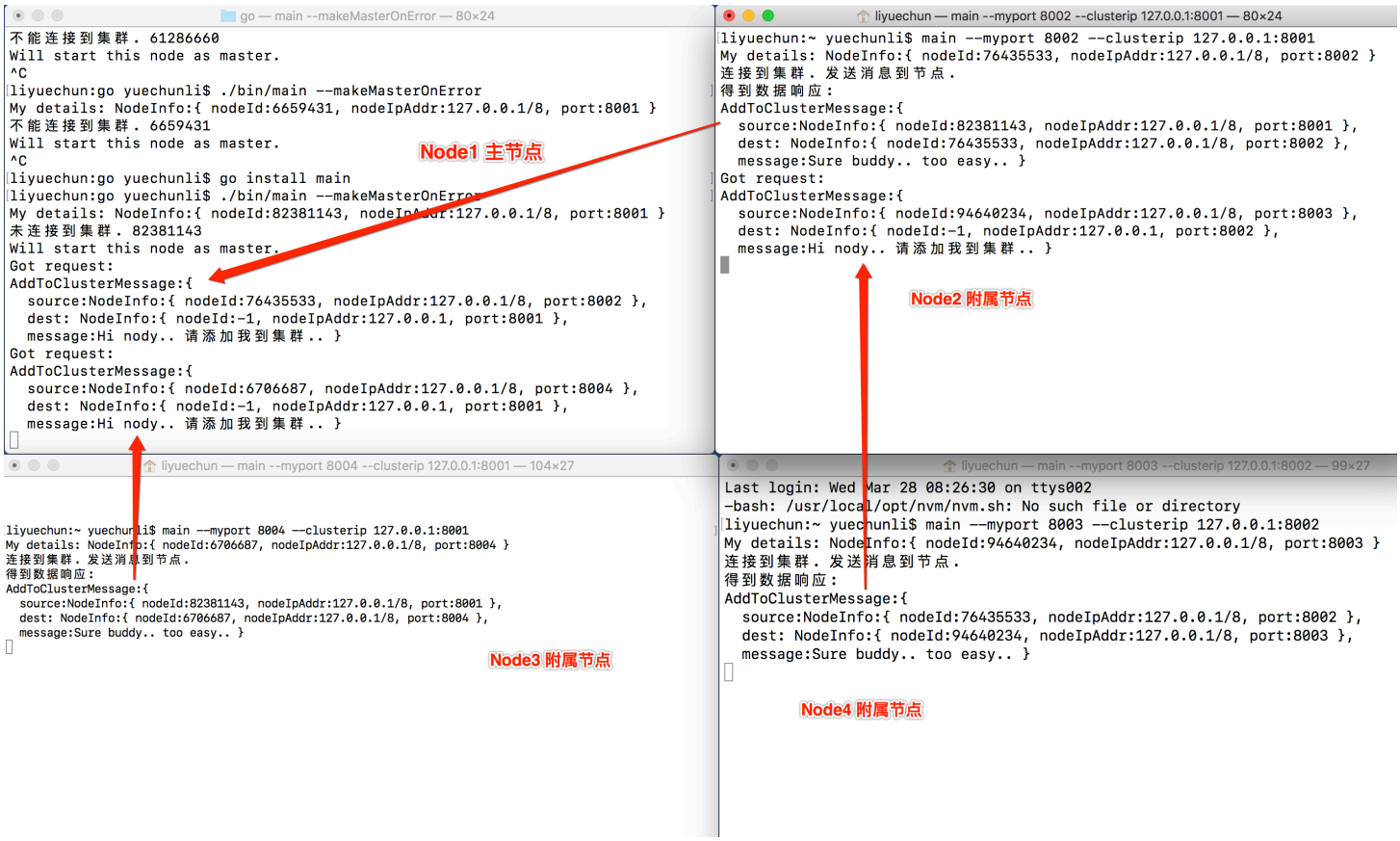
```
$ ./bin/main --myport 8002 --clusterip 127.0.0.1:8001
```

添加节点Node3到集群

```
main --myport 8004 --clusterip 127.0.0.1:8001
```

添加节点Node4到集群

```
$ main --myport 8003 --clusterip 127.0.0.1:8002
```



视频获取方式


区块链部落
专注于区块链技术
识别图中二维码关注我们