008 投票系统的chaincode 开发

> 孔壹学院：国内区块链职业教育领先品牌
> 官方网址：http://www.kongyixueyuan.com/

投票的主要功能

- 获取当前的投票列表
- 输入姓名就可以给相应的用户票数+1

投票的链码

给用户投票

- 查询用户是否存在
- 如果存在则该用户票数加1
- 如果不存在则新增一个用户

```
func (t *VoteChaincode) voteUser(stub shim.ChaincodeStubInterface , args []s
tring) peer.Response{
    // 查询当前用户的票数，如果用户不存在则新添一条数据，如果存在则给票数加1
    fmt.Println("start voteUser")
    vote := Vote{}
    username := args[0]
    voteAsBytes, err := stub.GetState(username)

    if err != nil {
        shim.Error("voteUser 获取用户信息失败！")
    }

    if voteAsBytes != nil {
        err = json.Unmarshal(voteAsBytes, &vote)
        if err != nil {
            shim.Error(err.Error())
        }
        vote.Votenum += 1
    }else {
        vote = Vote{ Username: args[0], Votenum: 1}
    }
```

```
    //将 Vote 对象 转为 JSON 对象
    voteJsonAsBytes, err := json.Marshal(vote)
    if err != nil {
        shim.Error(err.Error())
    }

    err = stub.PutState(username,voteJsonAsBytes)
    if err != nil {
        shim.Error("voteUser 写入账本失败！")
    }

    fmt.Println("end voteUser")
    return shim.Success(nil)
}
```

获取所有的用户信息

- GetStateByRange 获取所有用户的信息

```
func (t *VoteChaincode) getUserVote(stub shim.ChaincodeStubInterface, args [
]string) peer.Response{

    fmt.Println("start getUserVote")
    // 获取所有用户的票数
    resultIterator, err := stub.GetStateByRange("","")
    if err != nil {
        return shim.Error("获取用户票数失败！")
    }
    defer resultIterator.Close()

    var buffer bytes.Buffer
    buffer.WriteString("[")

    isWritten := false

    for resultIterator.HasNext() {
        queryResult , err := resultIterator.Next()
        if err != nil {
            return shim.Error(err.Error())
        }

        if isWritten == true {
            buffer.WriteString(",")
        }
```

```
        buffer.WriteString(string(queryResult.Value))
        isWritten = true
    }

    buffer.WriteString("]")

    fmt.Printf("查询结果：\n%s\n",buffer.String())
    fmt.Println("end getUserVote")
    return shim.Success(buffer.Bytes())
}
```

本地测试

测试代码 vote_test.go

```
package main

import (
    "fmt"
    "testing"
    "github.com/hyperledger/fabric/core/chaincode/shim"
)

func checkInvoke(t *testing.T, stub *shim.MockStub, args [][]byte) {
    res := stub.MockInvoke("1",args)
    if res.Status != shim.OK {
        fmt.Println("Invoke", "failed", string(res.Message))
        t.FailNow()
    }
}


func TestExample02_Invoke(t *testing.T) {
    scc := new(VoteChaincode)
    stub := shim.NewMockStub("ex02", scc)

    checkInvoke(t,stub,[][]byte{[]byte("getUserVote"), []byte("")})

    checkInvoke(t,stub,[][]byte{[]byte("voteUser"), []byte("kongyixueyuan")}
)

    checkInvoke(t,stub,[][]byte{[]byte("voteUser"), []byte("xiaoming")})

    checkInvoke(t,stub,[][]byte{[]byte("voteUser"), []byte("xiaowang")})
    checkInvoke(t,stub,[][]byte{[]byte("voteUser"), []byte("xiaowang")})
```

```
    checkInvoke(t,stub,[][]byte{[]byte("voteUser"), []byte("xiaowang")})

    checkInvoke(t,stub,[][]byte{[]byte("getUserVote"), []byte("")})

}
```

进入 chaincode 代码目录

```
cd chaincode008

go test -v vote_test.go vote.go
```