002 链码的结构

> 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/

Chaincode接口

```
type Chaincode interface {
    Init(stub shim.ChaincodeStubInterface) peer.Response
    Invoke(stub shim.ChaincodeStubInterface) peer.Response
}
```

链码的基本结构

```
package main

// 引入相应的依赖包
import (
    "fmt"
    "github.com/hyperledger/fabric/core/chaincode/shim"
    "github.com/hyperledger/fabric/protos/peer"
)

type SimpleChaincode struct {

}

// 链码实例化（instantiate）或 升级（upgrade）时调用 Init 方法
func (t *SimpleChaincode) Init(stub shim.ChaincodeStubInterface) peer.Respon
se{

    return shim.Success(nil)
}

// 链码收到调用（invoke） 或 查询 （query）时调用 Invoke 方法
func (t *SimpleChaincode) Invoke(stub shim.ChaincodeStubInterface) peer.Resp
onse {
    return shim.Success(nil)
}

// 主函数 , 调用 shim.Start 方法
func main() {
    err := shim.Start(new(SimpleChaincode))
```

```
    if( err!= nil){
        fmt.Printf("Error starting Simple Chaincode is %s \n",err)
    }
}
```

链码通过 `shim.ChaincodeStubInterface` 提供的方法来读取和修改账本状态。

helloworld

- 初始化的时候，给 str 赋值 helloworld
- 查询 str 当前的值
- 重新设置 str 的值
- 再次查询 str 的值

```
package main

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

type Helloworld struct {

}

func (t *Helloworld) Init(stub shim.ChaincodeStubInterface) peer.Response {

    fmt.Println("helloworld init")
    args := stub.GetStringArgs()
    err := stub.PutState(args[0],[]byte(args[1]))

    if err != nil {
        shim.Error(fmt.Sprintf("helloworld init err %s",err))
    }
    return shim.Success(nil)
}

func (t *Helloworld) Invoke(stub shim.ChaincodeStubInterface) peer.Response
{

    fn,args := stub.GetFunctionAndParameters()
```

```
    var result string
    var err error
    if fn == "set" {
        result,err = set(stub,args)
    }else{
        result,err = get(stub,args)
    }
    if err != nil {
        shim.Error(err.Error())
    }
    return shim.Success([]byte(result))
}

func set (stub shim.ChaincodeStubInterface, args []string) (string,error){

    fmt.Println("helloworld set")

    err := stub.PutState(args[0],[]byte(args[1]))
    if err != nil {
        return "", fmt.Errorf("Failed to set asset: %s", args[0])
    }
    return args[0],nil
}

func get(stub shim.ChaincodeStubInterface, args []string) (string,error){

    value, err := stub.GetState(args[0])

    if err != nil {
        return "", fmt.Errorf("Failed to get asset: %s", args[0])
    }
    if value == nil {
        return "", fmt.Errorf("Asset not found: %s", args[0])
    }

    return string(value),nil
}

func main() {
    err := shim.Start(new(Helloworld))

    if err != nil {
        fmt.Printf("shim start err %s",err)
    }
}
```

helloworld_test.go

```
package main

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

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

func checkQuery(t *testing.T, stub *shim.MockStub, name string) {
    res := stub.MockInvoke("1", [][]byte{[]byte("get"), []byte(name)})
    if res.Status != shim.OK {
        fmt.Println("Query", name, "failed", string(res.Message))
        t.FailNow()
    }
    if res.Payload == nil {
        fmt.Println("Query", name, "failed to get value")
        t.FailNow()
    }

    fmt.Println("Query value", name, "was ", string(res.Payload))

}

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

func Test_Helloworld(t *testing.T) {

    hello := new(Helloworld)
    stub := shim.NewMockStub("hello", hello)

    checkInit(t, stub, [][]byte{[]byte("str"), []byte("helloworld")})
```

```
    checkQuery(t, stub, "str")

    checkInvoke(t, stub, [][]byte{[]byte("set"), []byte("str"), []byte("hell
oworld-1111")})
    checkQuery(t, stub, "str")
}
```

通过 `go test -v helloworld_test.go helloworld.go` 测试相应的方法，运行结果：

```
=== RUN   Test_Helloworld
helloworld init
Query value str was  helloworld
helloworld set
Query value str was  helloworld-1111
--- PASS: Test_Helloworld (0.00s)
PASS
ok      command-line-arguments  0.028s
```