**Hyperledger Fabric programming**

**SharedChain**

CLI Commands to test (https://github.com/hyperledger/fabric-samples/tree/release-1.3/first-network)  
  
peer chaincode install -n sharedchain -v 1 -p github.com/drmillerne/SharedChain-Hackathon/gocode  
  
peer chaincode instantiate -o orderer.example.com:7050 --tls true --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C mychannel -n sharedchain -l golang -v 1.0 -c '{"Args":[]}' -P 'AND ('\''Org1MSP.peer'\'','\''Org2MSP.peer'\'')'  
  
peer chaincode invoke -o orderer.example.com:7050 --tls true --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem -C mychannel -n sharedchain --peerAddresses peer0.org1.example.com:7051 --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt --peerAddresses peer0.org2.example.com:7051 --tlsRootCertFiles /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt -c '{"Args":["invoke","1234","Coffee","10","BlackSheep","6226f7cbe59e99a90b5cef6f94f966fd"]}'  
  
peer chaincode query -C mychannel -n mycc -c '{"Args":["query","1234"]}'  
\*/  
  
package main  
  
import (  
"fmt" //"strconv"  
  
"github.com/hyperledger/fabric/core/chaincode/shim"  
pb "github.com/hyperledger/fabric/protos/peer"  
)  
  
// SharedChainChaincode Chaincode implementation  
type SharedChainChaincode struct {  
}  
  
  
// Init callback representing the invocation of a chaincode  
func (t \*SharedChainChaincode) Init(stub shim.ChaincodeStubInterface) pb.Response {  
return shim.Success(nil)  
}  
  
func (t \*SharedChainChaincode) invoke(stub shim.ChaincodeStubInterface, args []string) pb.Response {  
fmt.Println("SharedChainChaincode Invoke Called")  
var err error  
  
if len(args) != 5 {  
return shim.Error("Incorrect number of arguments")  
}  
  
SharedChainJSONasString := `{"docType":"SharedChain", "AssetID": "` + args[0] + `", "AssetName": "` + args[1] + `", "Quantity": "` + args[2] + `", "Owner": "` + args[3] + `", "Hash": "` + args[4] +`"}`  
SharedChainJSONasBytes := []byte(SharedChainJSONasString)  
  
// === Save ArtistView to state ===  
err = stub.PutState("artistView", SharedChainJSONasBytes)  
if err != nil {  
return shim.Error(err.Error())  
}  
  
return shim.Success(nil)  
}  
  
func (t \*SharedChainChaincode) Invoke(stub shim.ChaincodeStubInterface) pb.Response {  
function, args := stub.GetFunctionAndParameters()  
if function == "invoke" {  
return t.invoke(stub, args)  
}  
  
return shim.Error("Invalid invoke function name. Expecting \"invoke\"")  
}  
  
func main() {  
err := shim.Start(new(SharedChainChaincode))  
if err != nil {  
fmt.Printf("Error starting SharedChainChaincode: %s", err)  
}  
}