< 하이퍼레져 페브릭-3 >

1. ChainCode 작성하기

해당경로에 sacc라는 디렉토리 생성(-p는 없으면 전부 다 생성하라는 옵션)
touch명령어를 사용해 sacc.go라는 빈파일 생성(만지다는 뜻 : modify를 최신으로)

```
guru@guru:~$ echo $GOPATH
/home/guru/gowork
guru@guru:~$
guru@guru:~$ mkdir -p $GOPATH/src/sacc
guru@guru:~$ cd $GOPATH/src/sacc
guru@guru:~/gowork/src/sacc$ ll
total 8
drwxrwxr-x 2 guru guru 4096 Nov 19 09:36 ./
drwxrwxr-x 4 guru guru 4096 Nov 19 09:36 ../
guru@guru:~/gowork/src/sacc$
guru@guru:~/gowork/src/sacc$
guru@guru:~/gowork/src/sacc$ touch sacc.go
guru@guru:~/gowork/src/sacc$
guru@guru:~/gowork/src/sacc$ ll
total 8
drwxrwxr-x 2 guru guru 4096 Nov 19 09:36 ./
drwxrwxr-x 4 guru guru 4096 Nov 19 09:36 ../
-rw-rw-r-- 1 guru guru    0 Nov 19 09:36 sacc.go
```

stub : client와 CC간 통신하는 에이전시(각기 다른 환경에 있는 두 놈을 연결해주는 역할을 해주는 변수)
각각은 다른 서버(윈도우, 리눅스) 상에 설치되어있으므로 shim.ChaincodeStubInterface 이용해 연결!
fmt : format 문자열로 변환해주는 라이브러리

파이썬 문법 중 _ : 공간은 필요한데 데이터는 사용하지 않을 때
ex) carAsBytes, _ := APIstub.GetState(args[0])
-> GetState로 받아온 값을 carAsBytes에 넣고, err은 사용하지 않겠다!(공간만 지정해준 것)

$ rmate –p 52698 sacc.go를 통해 파일 띄워서 golang으로 코드 작성(Init, Invoke 함수)
작성 후 네트워크에 배포를 위해 **$ go build** → **$ ls**로 확인해보면 sacc 실행파일 생성!!
$ go get –u github.com/hyperledger/fabric/core/chaincode/shim(안되면 다운 후 build)

```
guru@guru:~/gowork/src/sacc$ go build
guru@guru:~/gowork/src/sacc$
guru@guru:~/gowork/src/sacc$ ls
sacc  sacc.go
```

$ docker stop $(docker ps -qa) && docker rm $(docker ps –qa)로 현재 실행중인 모든 프로세스 삭제
$ cd fabric-sample/chaincode-docker-devmode 디렉토리 이동 후 docker-compose파일 실행!

```
guru@guru:~/fabric-samples/chaincode-docker-devmode$ ll
total 44
drwxrwxr-x  3 guru guru 4096 Nov 14 13:42 ./
drwxrwxr-x 14 guru guru 4096 Nov 15 13:20 ../
-rw-rw-r--  1 guru guru 2702 Nov 14 13:42 docker-compose-simple.yaml
-rw-rw-r--  1 guru guru   83 Nov 14 13:42 .gitignore
drwxrwxr-x  8 guru guru 4096 Nov 14 13:42 msp/
-rw-rw-r--  1 guru guru  274 Nov 14 13:42 myc.tx
-rw-rw-r--  1 guru guru 7902 Nov 14 13:42 orderer.block
-rw-rw-r--  1 guru guru 5270 Nov 14 13:42 README.rst
-rwxrwxr-x  1 guru guru 1019 Nov 14 13:42 script.sh*
```

$ docker-compose –f docker-compose-simple.yaml up –d
: 간단하게 실행할 수 있는 실행파일(개발자모드에서 임시로 network모드 띄움)

```
guru@guru:~/fabric-samples/chaincode-docker-devmode$ docker-compose -f docker-compose-simple.yaml up -d
Creating network "chaincodedockerdevmode_default" with the default driver
Creating orderer
Creating peer
Creating cli
Creating chaincode
```

: network가 default로 켜지고, orderer, peer, cli, chaincode 등이 활성화 됨!(실습준비 완료)

$ docker exec –it chaincode bash 해서 체인코드를 컨트롤 할 수 있는 새로운 창으로 입장
$ cd sacc로 이동 → **$ ll**해보면 go파일과 빌드가 완료된 초록(*)파일이 있음!!(없으면 **$ go build** 실시)

```
root@a9f8766e7797:/opt/gopath/src/chaincode/sacc# ll
total 17096
drwxrwxr-x 2 1000 1000     4096 Nov 19 05:34 ./
drwxrwxr-x 9 1000 1000     4096 Nov 14 07:45 ../
-rwxr-xr-x 1 root root 17493760 Nov 19 05:34 sacc*
-rw-rw-r-- 1 1000 1000     2851 Nov 14 04:42 sacc.go
```

$ CORE_PEER_ADDRESS=peer:7052 CORE_CHAINCODE_ID_NAME=mycc:0 ./sacc
: 체인코드 주소를 7052포트에 할당시키고, 체인코드 아이디는 mycc 버전은 0, 실행은 sacc를 실행하라!

```
root@a9f8766e7797:/opt/gopath/src/chaincode/sacc# CORE_PEER_ADDRESS=peer:7052 CORE_CHAINCODE_ID_NAME=mycc:0 ./sacc
2018-11-19 05:35:52.498 UTC [shim] setupChaincodeLogging -> INFO 001 Chaincode log level not provided; defaulting to: INFO
2018-11-19 05:35:52.503 UTC [shim] setupChaincodeLogging -> INFO 002 Chaincode (build level: ) starting up ...
```

터미널 창 하나 더 키고 네트워크 새로 접속해서 docker로 들어가기(docker container 컨트롤을 위해서)
즉, 좌측은 체인코드 / 우측은 cli로 코딩(우측 창에서 실제 container들 컨트롤)
$ docker exec –it cli bash로 모드 변경!

```
guru@guru:~$ docker exec -it cli bash
root@02b414d1dc11:/opt/gopath/src/chaincodedev#
```

앞으로 install → instantiate를 거쳐 invoke(set,get) 실행
$ peer chaincode install –p chaincodedev/chaincode/sacc –n mycc –v 0 : 체인코드 설치(맨 밑에 OK 뜸)

```
2018-11-19 05:40:15.165 UTC [golang-platform] func1 -> DEBU 048 Discarding provided package github.com/hyperledger/fabric/protos/p
eer
2018-11-19 05:40:15.162 UTC [golang-platform] func1 -> DEBU 047 Discarding provided package github.com/hyperledger/fabric/core/cha
incode/shim
2018-11-19 05:40:15.165 UTC [golang-platform] func1 -> DEBU 048 Discarding provided package github.com/hyperledger/fabric/protos/peer
2018-11-19 05:40:15.167 UTC [golang-platform] GetDeploymentPayload -> DEBU 049 done
2018-11-19 05:40:15.168 UTC [container] WriteFileToPackage -> DEBU 04a Writing file to tarball: src/chaincodedev/chaincode/sacc/sacc.go
2018-11-19 05:40:15.179 UTC [msp/identity] Sign -> DEBU 04b Sign: plaintext: 0AC3070A5B08031A0B08BF99C9DF0510...7719FF040000FFFF996692F90
0120000
2018-11-19 05:40:15.181 UTC [msp/identity] Sign -> DEBU 04c Sign: digest: 4939016F89A554CD2A6BFB71D74110AD31056D7DF4B383C0FAC530EA946A76B
7
2018-11-19 05:40:15.202 UTC [chaincodeCmd] install -> INFO 04d Installed remotely response:<status:200 payload:"OK" >
```

$ peer chaincode list —installed : 위에 설치한 놈이 잘 설치 되었는지 확인

```
Name: mycc, Version: 0, Path: chaincodedev/chainco
de/sacc, Id: 9a6c57f6b71aeebe5998f6bf0356283edcfb4
0cb8082c64f4c6830f00868eacf
```

$ peer chaincode instantiate –n mycc –v 0 –C myc –c '{"Args":["user1","100"]}'
: instance화 작업(init함수에 해당하는 인자 전달)

```
2018-11-19 05:43:18.620 UTC [msp/identity] Sign -> DEBU 0a4 Sign: plaintext: 0AC9070A6108031A0C08
F69AC9DF0510...30300A000A04657363630A0476736363
2018-11-19 05:43:18.623 UTC [msp/identity] Sign -> DEBU 0a5 Sign: digest: 1EBF323A26EBA5193DD0CA4
6FE5AC183A979F23862A9DC300B6EF0575F8C790D
2018-11-19 05:43:18.655 UTC [msp/identity] Sign -> DEBU 0a6 Sign: plaintext: 0AC9070A6108031A0C08
F69AC9DF0510...95A9C44931F7DCE628AC27CE6B2277C5
2018-11-19 05:43:18.658 UTC [msp/identity] Sign -> DEBU 0a7 Sign: digest: BE2389A0D6D7499F78E4C41
CD290FE9C4ECD138E819A8B1D0A160EADB5CA65A0
```

$ peer chaincode list —instantiated –C myc
: 맨 밑에 'Name:mycc Version:0~~' 이런 식으로 잘 생성됐다고 뜸!

```
Get instantiated chaincodes on channel myc:
Name: mycc, Version: 0, Path: chaincodedev/chaincode/sacc, Escc: escc, Vscc: vscc
```

install과 instantiate를 모두 완료하였기 때문에 query문을 통해 test(초기값 : user1, 100)
$ peer chaincode invoke –n mycc –c '{"Args":["set","user1","200"]}'
체인코드 중 invoke를 실행 set함수 호출 user1의 값을 200으로 입력(변경)

```
2018-11-19 05:48:36.915 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 0a7
Chaincode invoke successful. result: status:200 payload:"200"
```

$ peer chaincode query –n mycc –c '{"Args":["query","user1"]}'
: 실행해보면 user1의 값이 200으로 바뀜!!(query는 blockchain이 기본 제공하는 함수)

```
E7715C33769F1E254264A6F0A8B6B5CD7278FB8402AEE85572B9AD6323EA
200
```

$ peer chaincode invoke –n mycc –c '{"Args":["get","user1"]}'
: user1의 값 200이 나옴

```
2018-11-19 05:50:53.936 UTC [chaincodeCmd] chaincodeInvokeOrQuery -> INFO 0a7
Chaincode invoke successful. result: status:200 payload:"200"
```

2. ChainCode 버전 업그레이드 : transfer와 delete 기능 추가구현!

PutState(key, val)에서 val값을 +,- / DelState(key)로 만들기

앞에서 작성한 sacc를 가지고 sacc2파일 추가 작성(version 업그레이드)

```
guru@guru:~/fabric-samples/chaincode$ cp -rf sacc sacc2
guru@guru:~/fabric-samples/chaincode$ cd sacc2
```

```
guru@guru:~/fabric-samples/chaincode/sacc2$ rm sacc
guru@guru:~/fabric-samples/chaincode/sacc2$ mv sacc.go sacc2.go
```

: mv라는 명령어는 이동의 개념도 있지만, rename의 기능도 함(sacc2.go로 이름 바꾸기)

```
guru@guru:~/fabric-samples/chaincode/sacc2$ ll
total 12
drwxrwxr-x  2 guru guru 4096 Nov 19 14:54 ./
drwxrwxr-x 10 guru guru 4096 Nov 19 14:54 ../
-rw-rw-r--  1 guru guru 2851 Nov 19 14:54 sacc2.go
```

앞에서 sacc.go를 sacc2.go로 이름만 바꿔서 옮겨놓고 **$ rmate –p 52698 sacc2.go**로 파일 띄워서 코드 수정!!(두 가지 함수를 추가 코딩)

```
if fn == "set" {
    result, err = set(stub, args)
} else if fn == "transfer" {
    result, err = transfer(stub, args)
} else if fn == "delete" {
    result, err = delete(stub, args)
} else { // assume 'get' even if fn is nil
    result, err = get(stub, args)
}
```

- transfer(송금)

: 입력받은 값을 +,- 해줘야 하는데 문자로 입력받기 때문에 숫자로 변환해주는 과정을 거쳐야 함.

: go lang은 strconv라는 라이브러리(맨 위에 import)를 제공해 줌($ Aval, _ = strconv.Atoi(문자))

알파벳 → integer로 바뀐다(↔Itoa)

- delete(삭제)

: delState()를 이용하는데 하나의 값만 출력함 → 그 값을 삭제!

최종적으로 코드를 다 작성하면 **$ go build → $ ll**로 확인해보면 실행파일 생성됨!!

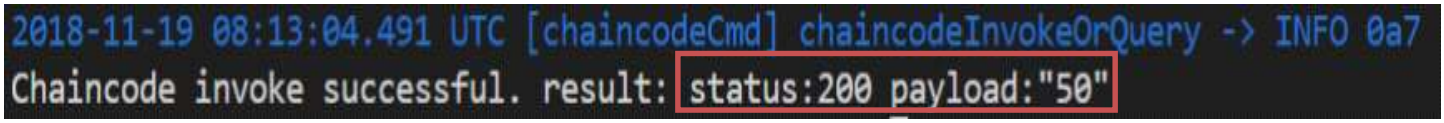
```
guru@guru:~/fabric-samples/chaincode/sacc2$ go build
guru@guru:~/fabric-samples/chaincode/sacc2$ ll
total 17116
drwxrwxr-x  2 guru guru     4096 Nov 19 16:26 ./
drwxrwxr-x 10 guru guru     4096 Nov 19 14:54 ../
-rwxrwxr-x  1 guru guru 17507776 Nov 19 16:26 sacc2*
-rw-rw-r--  1 guru guru     5516 Nov 19 16:16 sacc2.go
```

sacc는 처음 만드는거니까 instantiate의 과정을 거치지만, sacc2는 버전 업그레이드의 개념이므로 upgrade를 실시(현재로써는 chaincode를 삭제하는 방법이 없음. 그러기 위해서는 채널을 지워야 함!)

$ peer chaincode upgrade –n mycc –v 1 –c '{"Args":["a","10"]}' -C myc

instantiate가 아닌 upgrade

$ peer chaincode invoke –n mycc -C myc –c '{"Args":["transfer","user1","a","50"]}'
: transfer를 통해 user1이 a에게 50을 송금!



$ peer chaincode invoke –n mycc -C myc –c '{"Args":["delete","c"]}'
: delete를 통해 c를 삭제! → 이후 get이나 query 명령어를 통해 확인해보면 "not found c"