

# docker network naming



- basic\_network/docker-compose.yaml

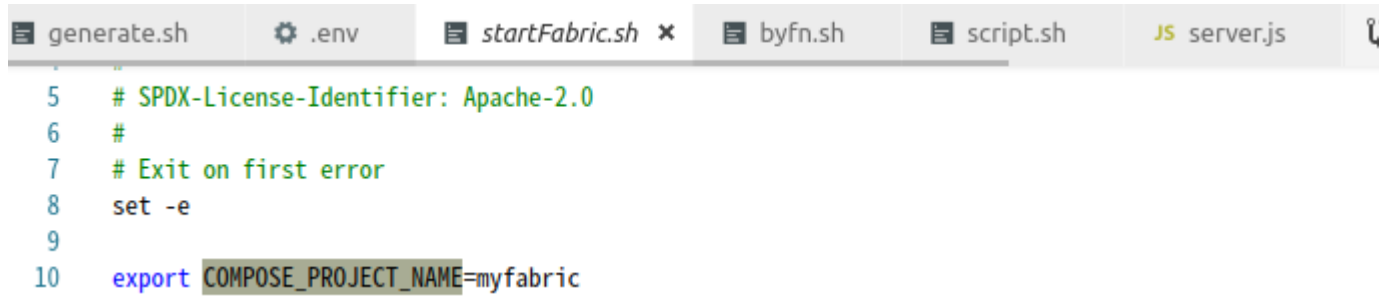
```
49 peer0.org1.example.com:
50   container_name: peer0.org1.example.com
51   image: hyperledger/fabric-peer
52   environment:
53     - CORE_VM_ENDPOINT=unix:///host/var/run/docker.sock
54     - CORE_PEER_ID=peer0.org1.example.com
55     - FABRIC_LOGGING_SPEC=info
56     - CORE_CHAINCODE_LOGGING_LEVEL=info
57     - CORE_PEER_LOCALMSPID=Org1MSP
58     - CORE_PEER_MSPCONFIGPATH=/etc/hyperledger/msp/peer/
59     - CORE_PEER_ADDRESS=peer0.org1.example.com:7051
60     # the following setting starts chaincode containers on the same
61     # bridge network as the peers
62     # https://docs.docker.com/compose/networking/
63     - CORE_VM_DOCKER_HOSTCONFIG_NETWORKMODE=${COMPOSE_PROJECT_NAME}_basic
64     - CORE_LEDGER_STATE_STATEDATABASE=CouchDB
65     - CORE_LEDGER_STATE_COUCHDBCONFIG_COUCHDBADDRESS=couchdb:5984
66     # The CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME and CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD
67     # provide the credentials for ledger to connect to CouchDB. The username and password must
68     # match the username and password set for the associated CouchDB.
69     - CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME=
70     - CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD=
71   working_dir: /opt/gopath/src/github.com/hyperledger/fabric
72   command: peer node start
73   # command: peer node start --peer-chaincodedev=true
74   ports:
75     - 7051:7051
76     - 7053:7053
77   volumes:
78     - /var/run:/host/var/run/
79     - ./crypto-config/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/msp:/etc/hyperledger
80     - ./crypto-config/peerOrganizations/org1.example.com/users:/etc/hyperledger/msp/users
81     - ./config:/etc/hyperledger/configtx
82   depends_on:
83     - orderer.example.com
84     - couchdb
85   networks:
86     - basic
```

# env설정

- basic\_network/.env

```
1 COMPOSE_PROJECT_NAME=net
2
```

- network 복사 수정시에 COMPOSE\_PROJECT\_NAME 설정

A screenshot of a code editor with several tabs: generate.sh, .env, startFabric.sh, byfn.sh, script.sh, and JS server.js. The .env tab is active, showing the following content:

```
5 # SPDX-License-Identifier: Apache-2.0
6 #
7 # Exit on first error
8 set -e
9
10 export COMPOSE_PROJECT_NAME=myfabric
```

# 외부참조 network사용



```
generate.sh  .env  docker-compose.yml x  byfn.sh  script.sh  JS server.js  ?
1  #
2  # Copyright IBM Corp All Rights Reserved
3  #
4  # SPDX-License-Identifier: Apache-2.0
5  #
6  version: '2'
7
8  networks:
9    basic:
10      external:
11        name: net_basic
12
13  services:
14    cliMagnetoCorp:
15      container_name: cliMagnetoCorp
16      image: hyperledger/fabric-tools
```

# 실시간 peer 추가



- first\_network사용
- crypto-config.yaml 템플릿 수 증가(peer수)

```
77 | Users:
78 |   Count: 1
79 | # -----
80 | # Org2: See "Org1" for full specification
81 | # -----
82 | - Name: Org2
83 |   Domain: org2.example.com
84 |   EnableNodeOUs: true
85 |   Template:
86 |     Count: 3
87 | Users:
88 |   Count: 1
89
```

# peer2.org2 crypto-config추가



- ../bin/cryptogen extend --config=./crypto-config.yaml

```
server.key
└─ peer2.org2.example.com
    └─ msp
        ├── admincerts
        │   └─ Admin@org2.example.com-cert.pem
        ├── cacerts
        │   └─ ca.org2.example.com-cert.pem
        ├── config.yaml
        ├── keystore
        │   └─ 0a5e70f20651d73b28abaa902a547f58eacf267daf81261d30162693cb4aa04f_sk
        ├── signcerts
        │   └─ peer2.org2.example.com-cert.pem
        └─ tlscacerts
            └─ tlsca.org2.example.com-cert.pem
    └─ tls
        ├── ca.crt
        ├── server.crt
        └─ server.key
```

# peer, couchdb 컨테이너 추가



- docker-compose-new-peer.yaml

```
1 container_name: couchdb4
2 image: hyperledger/fabric-couchdb
3 # Populate the COUCHDB_USER and COUCHDB_PASSWORD to set an admin user and password
4 # for CouchDB. This will prevent CouchDB from operating in an "Admin Party" mode.
5 environment:
6   - COUCHDB_USER=
7   - COUCHDB_PASSWORD=
8 # Comment/Uncomment the port mapping if you want to hide/expose the CouchDB service,
9 # for example map it to utilize Fauxton User Interface in dev environments.
10 ports:
11   - "9984:5984"
12 networks:
13   - byfn
14 peer2.org2.example.com:
15   container_name: peer2.org2.example.com
16   extends:
17     file: base/peer-base.yaml
18     service: peer-base
19   environment:
20     - CORE_LEDGER_STATE_STATEDATABASE=CouchDB
21     - CORE_LEDGER_STATE_COUCHDBCONFIG_COUCHDBADDRESS=couchdb4:5984
22     # The CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME and CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD
23     # provide the credentials for ledger to connect to CouchDB. The username and password must
24     # match the username and password set for the associated CouchDB.
25     - CORE_LEDGER_STATE_COUCHDBCONFIG_USERNAME=
26     - CORE_LEDGER_STATE_COUCHDBCONFIG_PASSWORD=
27     - CORE_PEER_ID=peer2.org2.example.com
28     - CORE_PEER_ADDRESS=peer2.org2.example.com:7051
29     - CORE_PEER_GOSSIP_EXTERNALENDPOINT=peer2.org2.example.com:7051
30     - CORE_PEER_GOSSIP_BOOTSTRAP=peer1.org2.example.com:7051
31     - CORE_PEER_LOCALMSPID=Org2MSP
32   volumes:
33     - /var/run:/host/var/run/
34     - ./crypto-config/peerOrganizations/org2.example.com/peers/peer2.org2.example.com/msp/etc/
35       hyperledger/fabric/msp
36     - ./crypto-config/peerOrganizations/org2.example.com/peers/peer2.org2.example.com/tls/etc/
37       hyperledger/fabric/tls
38   ports:
39     - 11051:7051
40     - 11053:7053
41   depends_on:
42     - couchdb4
43   networks:
44     - byfn
```

# 컨테이너 실행



```
$ docker-compose -f docker-compose-new-peer.yaml up -d
```

```
bstudent@bstudent-VirtualBox:~/fabric-samples/first-network$ docker-compose -f docker-compose-new-peer.yaml up -d
Creating couchdb4
Creating peer2.org2.example.com
```

# 채널에 join



```
$ docker exec -it cli bash
```

피어2 명령 수행을 위한 환경변수 지정

```
export CHANNEL_NAME=mychannel
```

```
CORE_PEER_LOCALMSPID="Org2MSP"
```

```
CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com  
/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt
```

```
CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/users/Admin@org2.example.com/msp
```

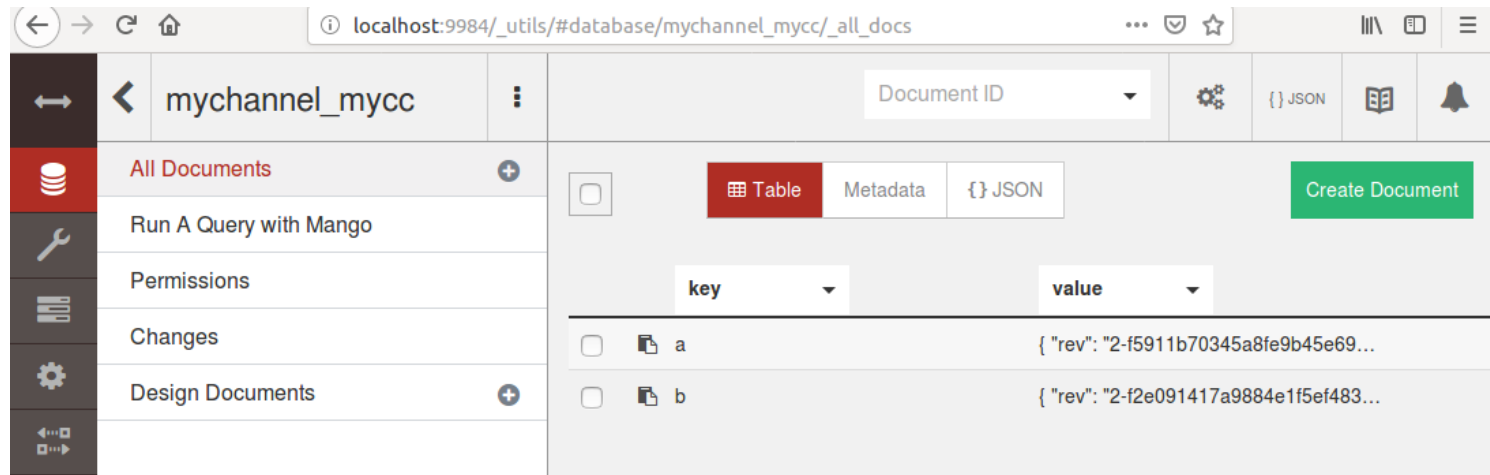
```
CORE_PEER_ADDRESS=peer2.org2.example.com:7051
```



\$ peer channel join -b mychannel.block

```
root@bb04fa0754cf:/opt/gopath/src/github.com/hyperledger/fabric/peer# export CHANNEL_NAME=mychannel
root@bb04fa0754cf:/opt/gopath/src/github.com/hyperledger/fabric/peer# CORE_PEER_LOCALMSPID="Org2MSP"
root@bb04fa0754cf:/opt/gopath/src/github.com/hyperledger/fabric/peer# CORE_PEER_TLS_ROOTCERT_FILE=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt
root@bb04fa0754cf:/opt/gopath/src/github.com/hyperledger/fabric/peer# CORE_PEER_MSPCONFIGPATH=/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/peerOrganizations/org2.example.com/users/Admin@org2.example.com/msp
root@bb04fa0754cf:/opt/gopath/src/github.com/hyperledger/fabric/peer# CORE_PEER_ADDRESS=peer2.org2.example.com:7051
root@bb04fa0754cf:/opt/gopath/src/github.com/hyperledger/fabric/peer# peer channel join -b mychannel.block
2019-05-04 07:07:00.280 UTC [channelCmd] InitCmdFactory -> INFO 001 Endorser and orderer connections initialized
2019-05-04 07:07:00.626 UTC [channelCmd] executeJoin -> INFO 002 Successfully submitted proposal to join channel
root@bb04fa0754cf:/opt/gopath/src/github.com/hyperledger/fabric/peer#
```

[http://localhost:9984/\\_utils/#database/mychannel\\_mycc/\\_all\\_docs](http://localhost:9984/_utils/#database/mychannel_mycc/_all_docs)

The screenshot shows the Hyperledger Explorer web interface. The browser address bar displays 'localhost:9984/\_utils/#database/mychannel\_mycc/\_all\_docs'. The interface has a sidebar on the left with navigation links: 'All Documents', 'Run A Query with Mango', 'Permissions', 'Changes', and 'Design Documents'. The main content area shows a table view of documents. At the top, there are tabs for 'Table', 'Metadata', and '{} JSON', and a 'Create Document' button. Below the tabs, there are dropdowns for 'key' and 'value'. The table lists two documents, 'a' and 'b', each with a 'rev' field containing a long alphanumeric string.

key	value
a	{ "rev": "2-f5911b70345a8fe9b45e69..." }
b	{ "rev": "2-f2e091417a9884e1f5ef483..." }

