Pre requisites

Git
Curl
Docker
Jq

Install Fabric and Fabric Samples

mkdir fabric
cd fabric

Download fabric samples

curl -sSLO https://raw.githubusercontent.com/hyperledger/fabric/main/scripts/install-fabric.sh && chmod +x install-fabric.sh

Pull the docker containers

./install-fabric.sh

Navigate to test network directory

cd fabric-samples/test-network

Remove any containers or artifacts

./network.sh down

Up the network

./network.sh up

Verify the containers

docker ps -a

Create a channel

./network.sh createChannel -c mychannel

Up and create channel in single step (already done)

./network.sh up createChannel

Deploy chaincode on peers and channel

./network.sh deployCC -ccn basic -ccp ../asset-transfer-basic/chaincode-typescript -ccl typescript

Interacting with the network

Set the path for peer binary and config for core.yaml

export PATH=${PWD}/../bin:$PATH
export FABRIC_CFG_PATH=$PWD/../config/

Set the environment variables to operate Peer as Org1

export CORE_PEER_TLS_ENABLED=true
export CORE_PEER_LOCALMSPID="Org1MSP"
export CORE_PEER_TLS_ROOTCERT_FILE=${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt
export CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org1.example.com/users/Admin@org1.example.com/msp
export CORE_PEER_ADDRESS=localhost:7051

Command to initialize the ledger with assets

peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride orderer.example.com --tls --cafile "${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n basic --peerAddresses localhost:7051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.com/tls/ca.crt" --peerAddresses localhost:9051 --tlsRootCertFiles "${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.com/tls/ca.crt" -c '{"function":"InitLedger","Args":[]}'

Query the ledger

peer chaincode query -C mychannel -n basic -c '{"Args":["GetAllAssets"]}'

Transfer the asset

```
peer chaincode invoke -o localhost:7050 --ordererTLSHostnameOverride
orderer.example.com --tls --cafile
"${PWD}/organizations/ordererOrganizations/example.com/orderers/orderer.example.com/m
sp/tlscacerts/tlsca.example.com-cert.pem" -C mychannel -n basic --peerAddresses
localhost:7051 --tlsRootCertFiles
"${PWD}/organizations/peerOrganizations/org1.example.com/peers/peer0.org1.example.co
m/tls/ca.crt" --peerAddresses localhost:9051 --tlsRootCertFiles
"${PWD}/organizations/peerOrganizations/org2.example.com/peers/peer0.org2.example.co
m/tls/ca.crt" -c '{"function":"TransferAsset","Args":["asset6","Christopher"]}'
```

Lets query the ledger from Org2 peer

Set the environment variables to operate Peer as Org2

```
export CORE_PEER_TLS_ENABLED=true
export CORE_PEER_LOCALMSPID="Org2MSP"
export
CORE_PEER_TLS_ROOTCERT_FILE=${PWD}/organizations/peerOrganizations/org2.exa
mple.com/peers/peer0.org2.example.com/tls/ca.crt
export
CORE_PEER_MSPCONFIGPATH=${PWD}/organizations/peerOrganizations/org2.example.
com/users/Admin@org2.example.com/msp
export CORE_PEER_ADDRESS=localhost:9051
```

Query the ledger

```
peer chaincode query -C mychannel -n basic -c '{"Args":["ReadAsset","asset6"]}'
```

Bring the network down

```
./network.sh down
```

Running a Application using a Fabric Gateway

Up and create the channel

./network.sh up createChannel -c mychannel -ca

Deploy the smart contract

./network.sh deployCC -ccn basic -ccp ../asset-transfer-basic/chaincode-typescript/ -ccl typescript

Open new terminal

Sample Application

cd asset-transfer-basic/application-gateway-typescript

Install the node modules

npm install

Run the application

npm start

Bring the network down

./network.sh down