使用多通道

```
$ cd ~/hyfa/fabric-samples/chaincode-docker-devmode

$ sudo docker-compose -f docker-compose-simple.yaml down

$ cd ../first-network
```

创建一个应用通道的配置交易

务必替换$CHANNEL_NAME或设置CHANNEL_NAME为可在整个说明中使用的环境变量

```
$ export CHANNEL_NAME=mychannel2
```

指定使用 `configtx.yaml` 配置文件中的 `TwoOrgsChannel` 模板, 来生成新建通道的配置交易文件,

```
$ sudo ../bin/configtxgen -profile TwoOrgsChannel -
outputCreateChannelTx ./channel-artifacts/channel.tx -channelID
$CHANNEL_NAME
```

输出如下

```
11:13:24.984 CST [common/tools/configtxgen] main -> INFO 001 Loading
configuration
11:13:24.992 CST [common/tools/configtxgen] doOutputChannelCreateTx
-> INFO 002 Generating new channel configtx
11:13:24.993 CST [msp] getMspConfig -> INFO 003 Loading NodeOUs
11:13:24.994 CST [msp] getMspConfig -> INFO 004 Loading NodeOUs
11:13:25.016 CST [common/tools/configtxgen] doOutputChannelCreateTx
-> INFO 005 Writing new channel tx
```

生成锚节点配置更新文件

锚节点配置更新文件用来对组织的锚节点进行配置

同样基于 `configtx.yaml` 配置文件生成新建通道文件，每个组织都需要分别生成且注意指定对应的组织名称

```
$ sudo ../bin/configtxgen -profile TwoOrgsChannel -
outputAnchorPeersUpdate ./channel-artifacts/Org1MSPanchors.tx -
channelID $CHANNEL_NAME -asOrg Org1MSP

$ sudo ../bin/configtxgen -profile TwoOrgsChannel -
outputAnchorPeersUpdate ./channel-artifacts/Org2MSPanchors.tx -
channelID $CHANNEL_NAME -asOrg Org2MSP
```

启动网络

`$ sudo docker-compose -f docker-compose-cli.yaml up -d`

进入Docker容器

执行如下命令进入到CLI容器中

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

创建通道

检查环境变量是否正确设置

```
echo $CHANNEL_NAME
```

设置环境变量

```
export CHANNEL_NAME=mychannel2
```

创建通道

```
peer channel create -o orderer.example.com:7050 -c $CHANNEL_NAME -f ./channel-artifacts/channel.tx --tls --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem
```

加入通道

应用通道所包含组织的成员节点可以加入通道中

```
peer channel join -b mychannel2.block -o orderer:7050
```

更新锚点

使用Org1的管理员身份更新锚节点配置

```
peer channel update -o orderer.example.com:7050 -c $CHANNEL_NAME -f ./channel-artifacts/Org1MSPanchors.tx --tls --cafile /opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem
```

使用Org2的管理员身份更新锚节点配置

```
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=peer0.org2.example.com:7051
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 peer channel update -o
orderer.example.com:7050 -c $CHANNEL_NAME -f ./channel-artifacts/Org2MSPanchors.tx --tls --cafile
/opt/gopath/src/github.com/hyperledger/fabric/peer/crypto/ordererOrganizations/example.com/orderers/orderer.example.com/msp/tlscacerts/tlsca.example.com-cert.pem
```

列出所加入的通道

```
peer channel list
```

list命令会列出指定的Peer节点已经加入的所有应用通道的列表.