

하이퍼레저 패브릭 구동 실습

2022-02-14

빅픽처랩(주)

안 휘

실습 목차

- **실행 환경 구성 (1시간)**
 - WSL 2 설치
 - Visual Studio Code 설치
 - docker, docker-compose 설치
- **하이퍼레저 패브릭 설치 (1시간)**
 - 하이퍼레저 패브릭 설치
 - test-network 실행
- **하이퍼레저 패브릭 실행 (1시간)**
 - 채널 생성
 - 체인코드 패키징 및 배포
- **하이퍼레저 패브릭 Client Server 실행 (1시간)**
 - node.js 설치
 - Client Server 실행

실행 환경 구성

- WSL 2 설치
- Visual Studio Code 설치
- docker, docker-compose 설치

Windows 버전 확인

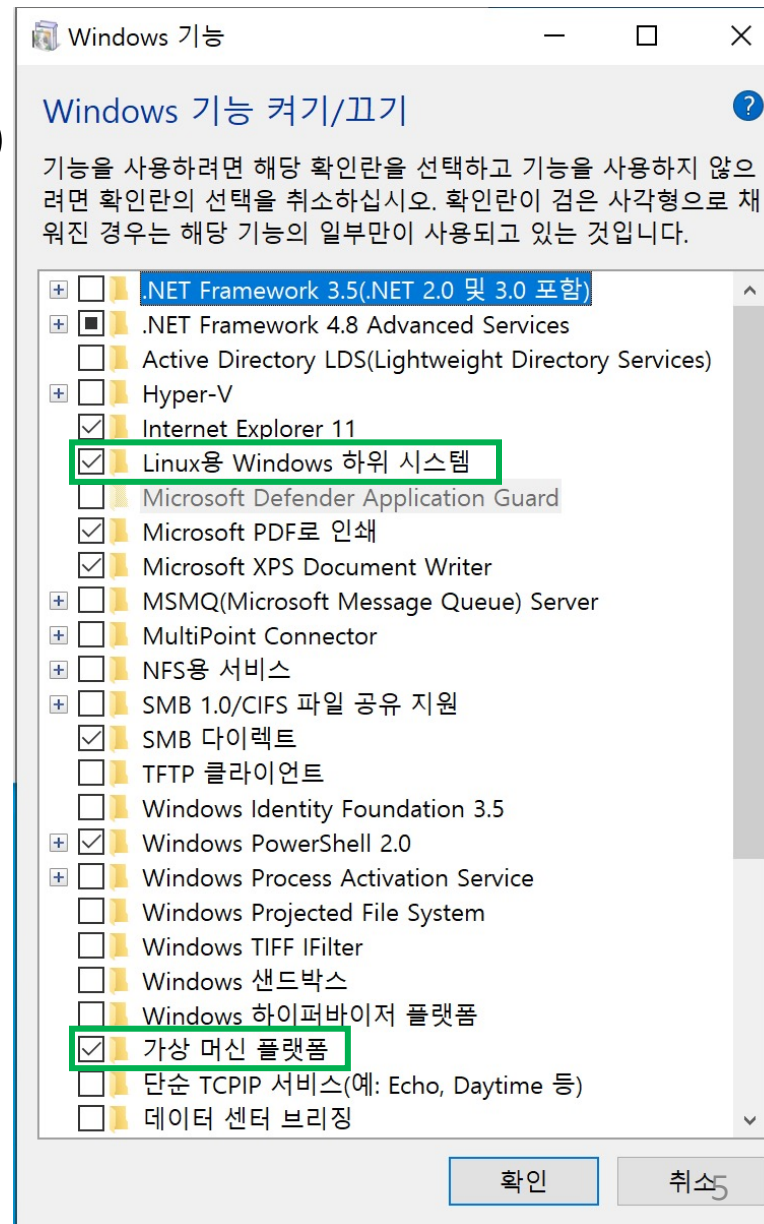
- 설정 -> 시스템 -> 정보



WSL 2 설치 준비

- Windows 10 버전 1903 (빌드 18362)

- Windows 검색
- “제어판” 검색
- “프로그램” 선택
- “프로그램 및 기능” 섹션의
“Windows 기능 켜기/끄기” 선택
- “Linux용 Windows 하위 시스템”,
“가상 머신 플랫폼” 선택
- “확인”
- 시스템 다시 시작



WSL 2 설치 (최신 버전)

- Windows 10 버전 2004 (빌드 19041) 이상 또는 Windows 11
 - Windows 검색
 - “PowerShell” 검색
 - “관리자로 실행” 선택
 - 다음 커맨드 입력
 - `wsl --install`
- 완료 후 재시작

관리자: Windows PowerShell

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

새로운 크로스 플랫폼 PowerShell 사용 <https://aka.ms/pscore6>

PS C:\Windows\system32> `wsl --install`

설치 중: 가상 머신 플랫폼

가상 머신 플랫폼이(가) 설치되었습니다.

설치 중: Linux용 Windows 하위 시스템

Linux용 Windows 하위 시스템이(가) 설치되었습니다.

다운로드 중: WSL 커널

설치 중: WSL 커널

WSL 커널이(가) 설치되었습니다.

다운로드 중: Ubuntu

요청한 작업이 잘 실행되었습니다. 시스템을 다시 시작하면 변경 사항이 적용됩니다.

PS C:\Windows\system32>

WSL 2 설치 (최신 버전)

- Windows 10 버전 2004 (빌드 19041) 이상 또는 Windows 11
 - 재시작 후에 자동으로 Ubuntu 실행되며 계정 설정 진행

 ahnhwi@HWIAHNCD12: ~

```
Installing, this may take a few minutes...  
Please create a default UNIX user account. The username does not need to match your Windows username.  
For more information visit: https://aka.ms/wslusers  
Enter new UNIX username: ahnhwi  
New password:  
Retype new password:  
passwd: password updated successfully  
Installation successful!  
To run the command prompt as administrator (run "cmd") use "cmd" as command.
```

WSL 2 설치 (과거 버전)

- Windows 10 WSL 2 설치

- Linux 커널 업데이트 패키지 다운로드

- https://wslstorestorage.blob.core.windows.net/wslblob/wsl_update_x64.msi

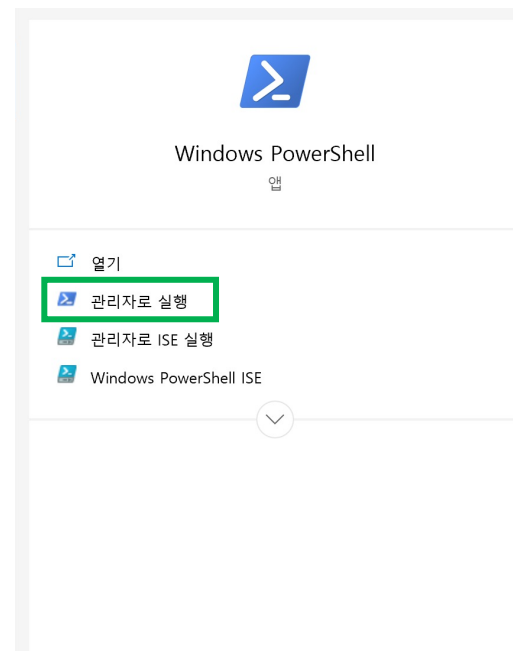
- 구글에서 “wsl 설치” 검색 -> 마이크로소프트 공식 문서 “4단계” 섹션 참조

- 설치

- WSL 2를 기본 버전으로 설정

- Windows 검색
- “PowerShell” 검색
- “관리자로 실행” 선택
- 다음 커맨드 입력

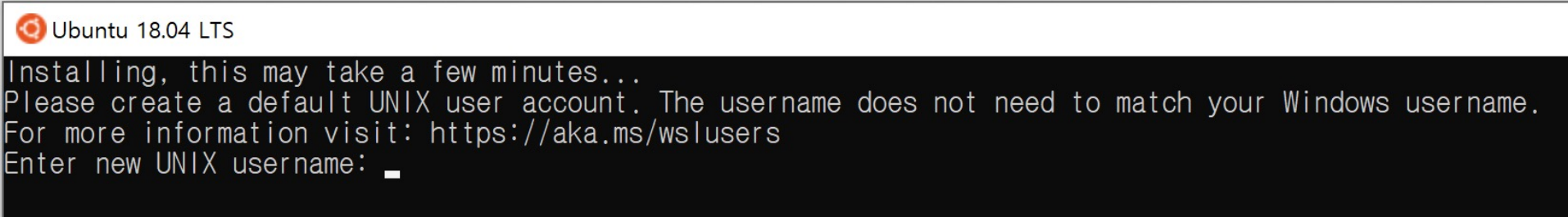
- `wsl --set-default-version 2`



WSL 2 설치 (과거 버전)

- Windows 10 WSL 2 설치

- Linux 배포판 설치
 - Windows 검색
 - “Microsoft Store” 검색
 - “Ubuntu 20.04” 검색
 - “Ubuntu 20.04” 설치
 - Microsoft 로그인 필요 없음
 - 다운로드 완료 후 실행
 - 사용자 계정 생성
 - 비밀번호 입력 때는 원래 아무 텍스트도 타이핑 되지 않으니 당황하지 말 것

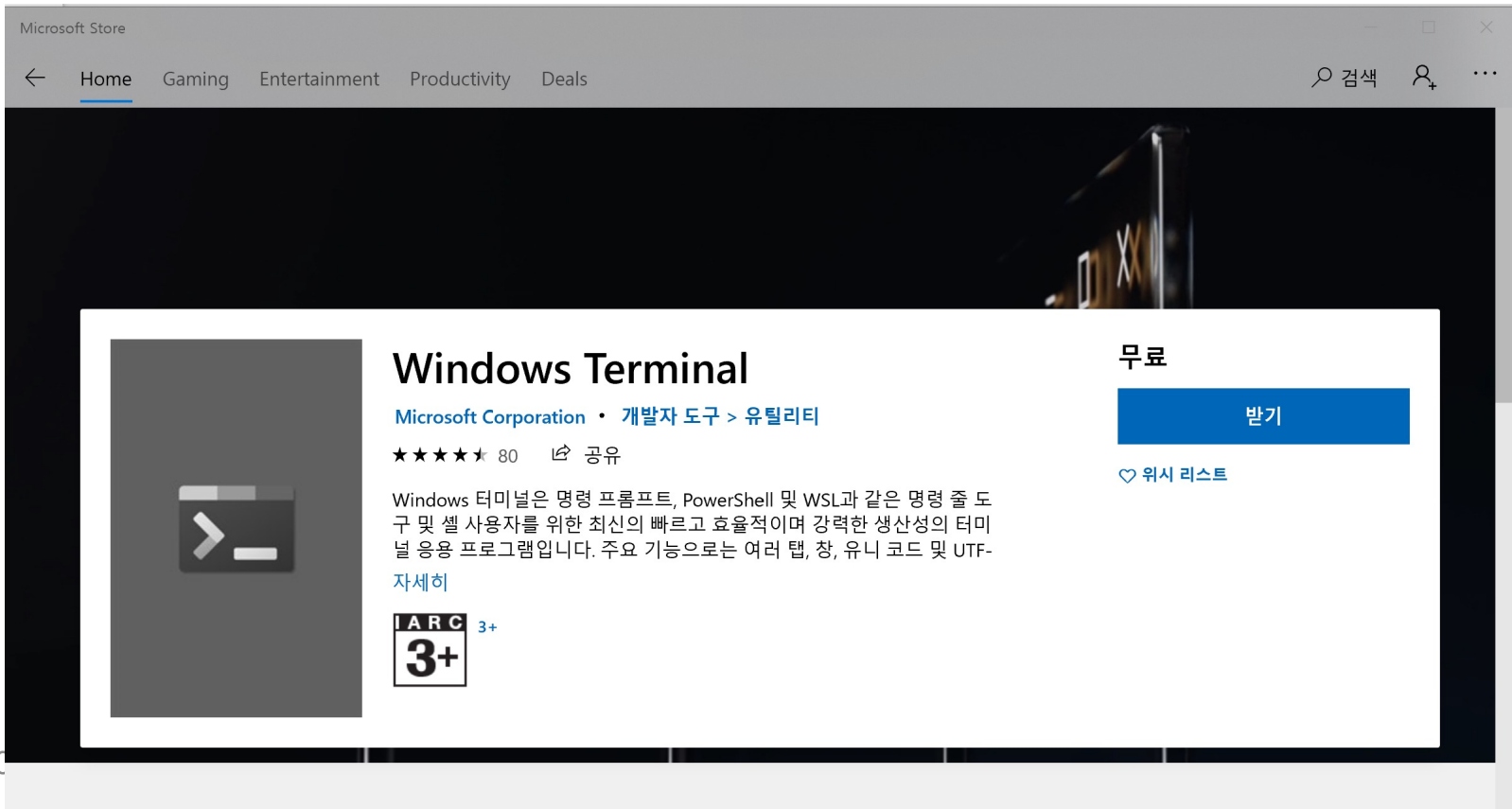


```
Ubuntu 18.04 LTS
Installing, this may take a few minutes...
Please create a default UNIX user account. The username does not need to match your Windows username.
For more information visit: https://aka.ms/wslusers
Enter new UNIX username: _
```

Windows Terminal 설치 (선택)

- Ubuntu 기본 터미널보다 편리

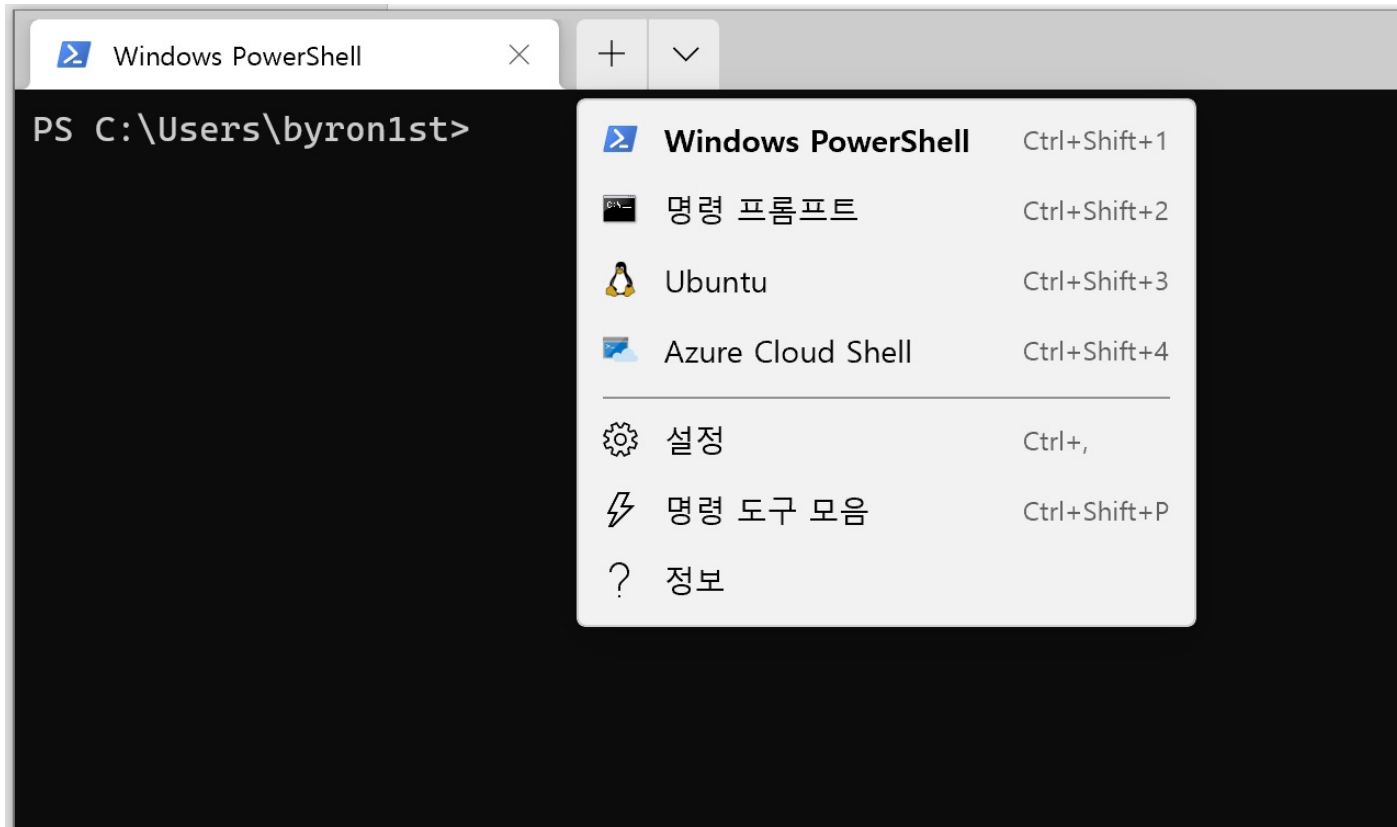
- Windows 검색
- “Microsoft Store” 검색
- “Windows Terminal” 검색 (Preview 버전 설치하지 않도록 주의, 로그인 안해도 됨)



Windows Terminal 설치 (선택)

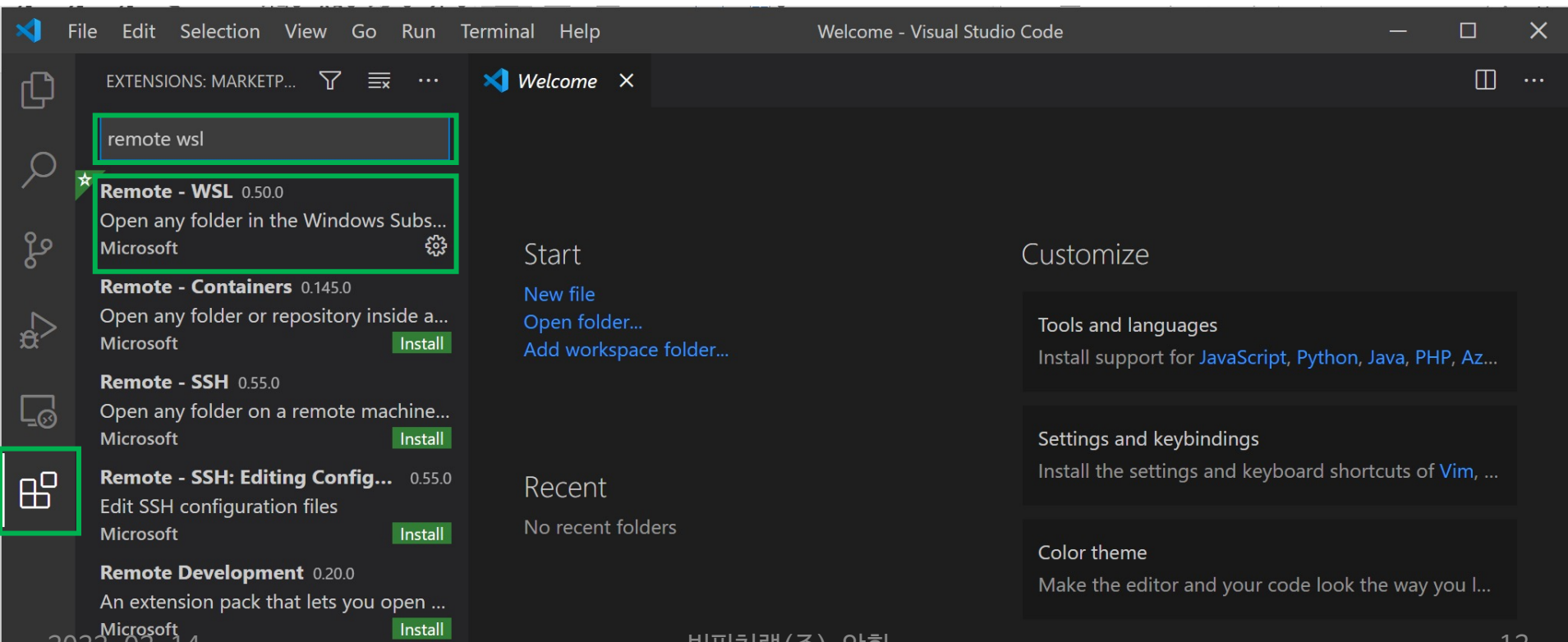
- Ubuntu 기본 터미널보다 편리

- Windows Terminal 실행
- 좌측 상단 첫번째 탭 옆에 아래 방향 화살표 클릭 후, Ubuntu 선택



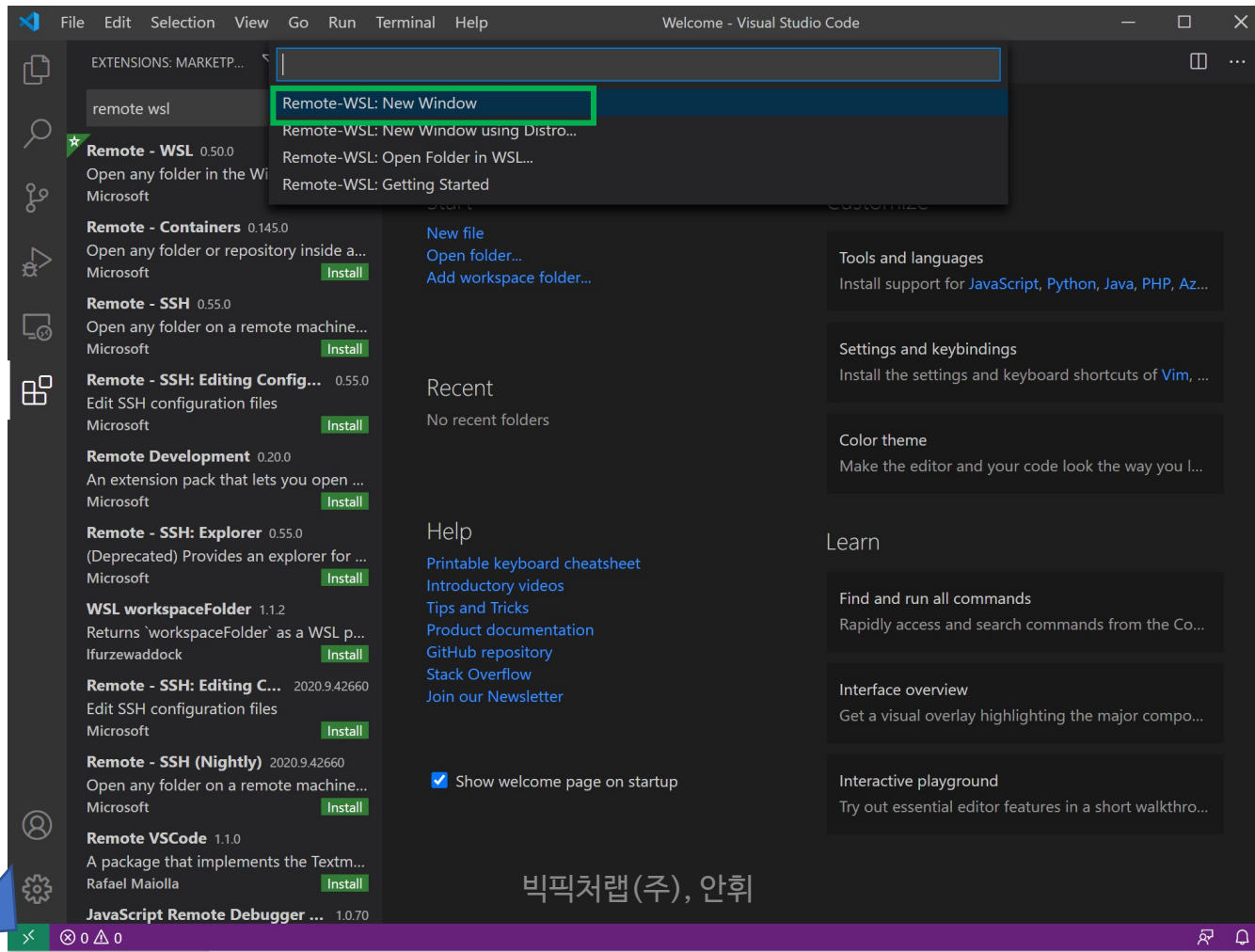
Visual Studio Code 설치

- <https://code.visualstudio.com/>
- 다운로드, 설치 및 실행
- (Windows 10) WSL 접속 확장 프로그램 설치



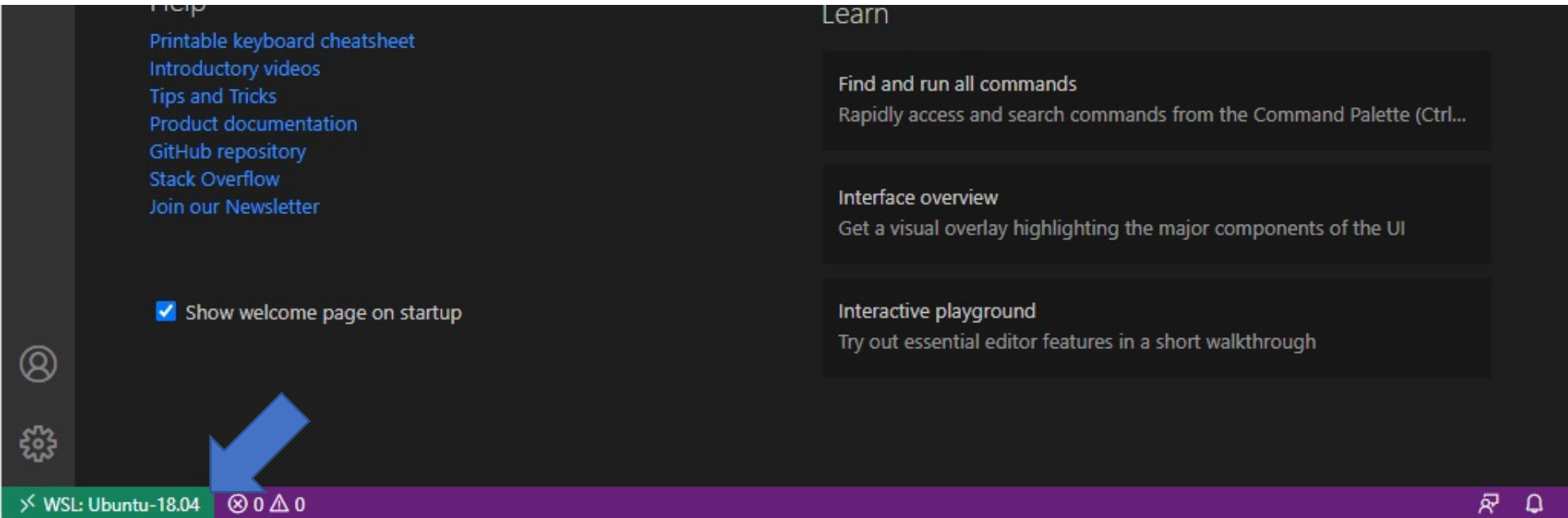
Visual Studio Code 설치

- (Windows 10) WSL 접속 확장 프로그램 설치



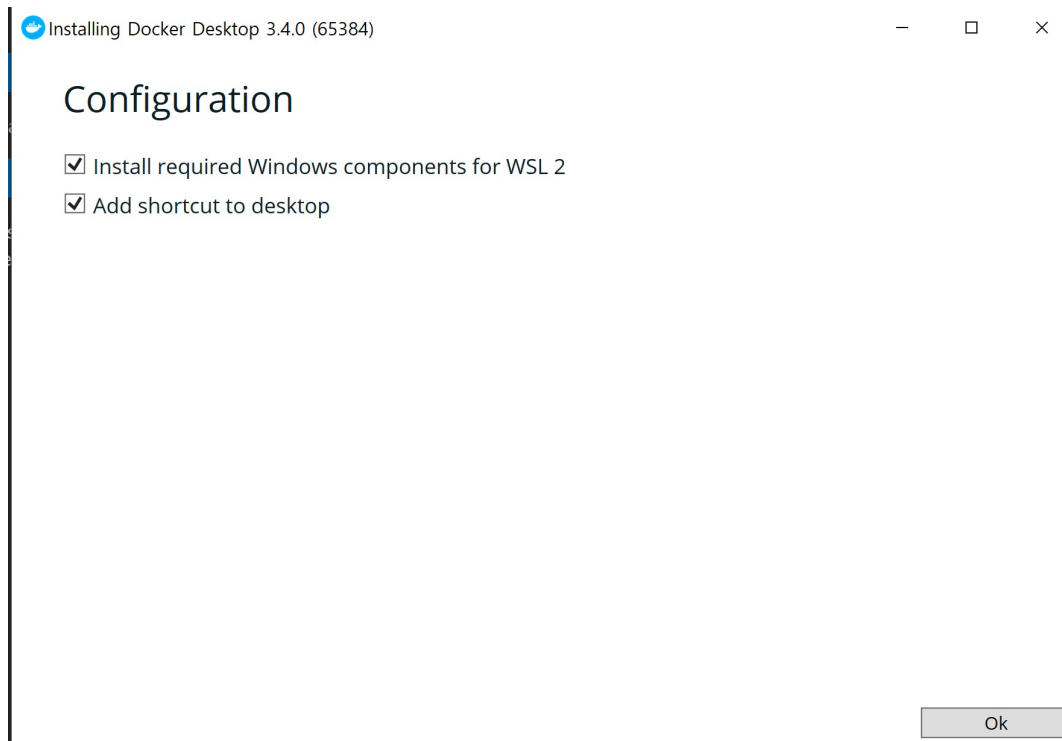
Visual Studio Code 설치

- (Windows 10) WSL 접속 확장 프로그램 설치



docker, docker-compose 설치

- <https://docs.docker.com/get-docker/>
 - Windows, macOS에 맞춰서 다운로드 후 설치
 - Windows의 경우 설치 중 나오는 WSL Integration 옵션 체크 (이미 선택되어 있음)
 - 설치 완료 후 Close And Logout 을 클릭하여 로그아웃 했다가 다시 접속



docker, docker-compose 설치

```
byron1st@HWIAHN7D0D:~$ docker ps
Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Get http://%2Fvar%2Frun%2Fdocker.sock/v1.24/containers/json: dial unix /var/run/docker.sock: connect: permission denied
byron1st@HWIAHN7D0D:~$ sudo usermod -aG docker $USER
[sudo] password for byron1st:
byron1st@HWIAHN7D0D:~$
```

- Get permission denied 에러가 발생할 경우
 - `sudo usermod -aG docker $USER`
 - 비밀번호 입력
 - Ubuntu 꺾다가 다시 실행

하이퍼레저 패브릭 설치

- 하이퍼레저 패브릭 설치
- test-network 실행

하이퍼레저 패브릭 설치

- Docker 를 실행
- Ubuntu 를 실행
- 다음 커맨드 입력
 - `cd ~/`
 - `mkdir Workspace && cd ./Workspace`
 - `curl -sSL https://bit.ly/2ysb0FE | bash -s`
- 명령어를 실행한 곳으로부터 “./fabric-samples” 폴더에 복사됨
 - 버전은 최신 버전인 2.4.2 (Fabric CA v1.5.2)이 설치됨
- 설치가 완료된 후 아래 명령어로 VSCode 에서 열기
 - `code fabric-samples/`

test-network 실행

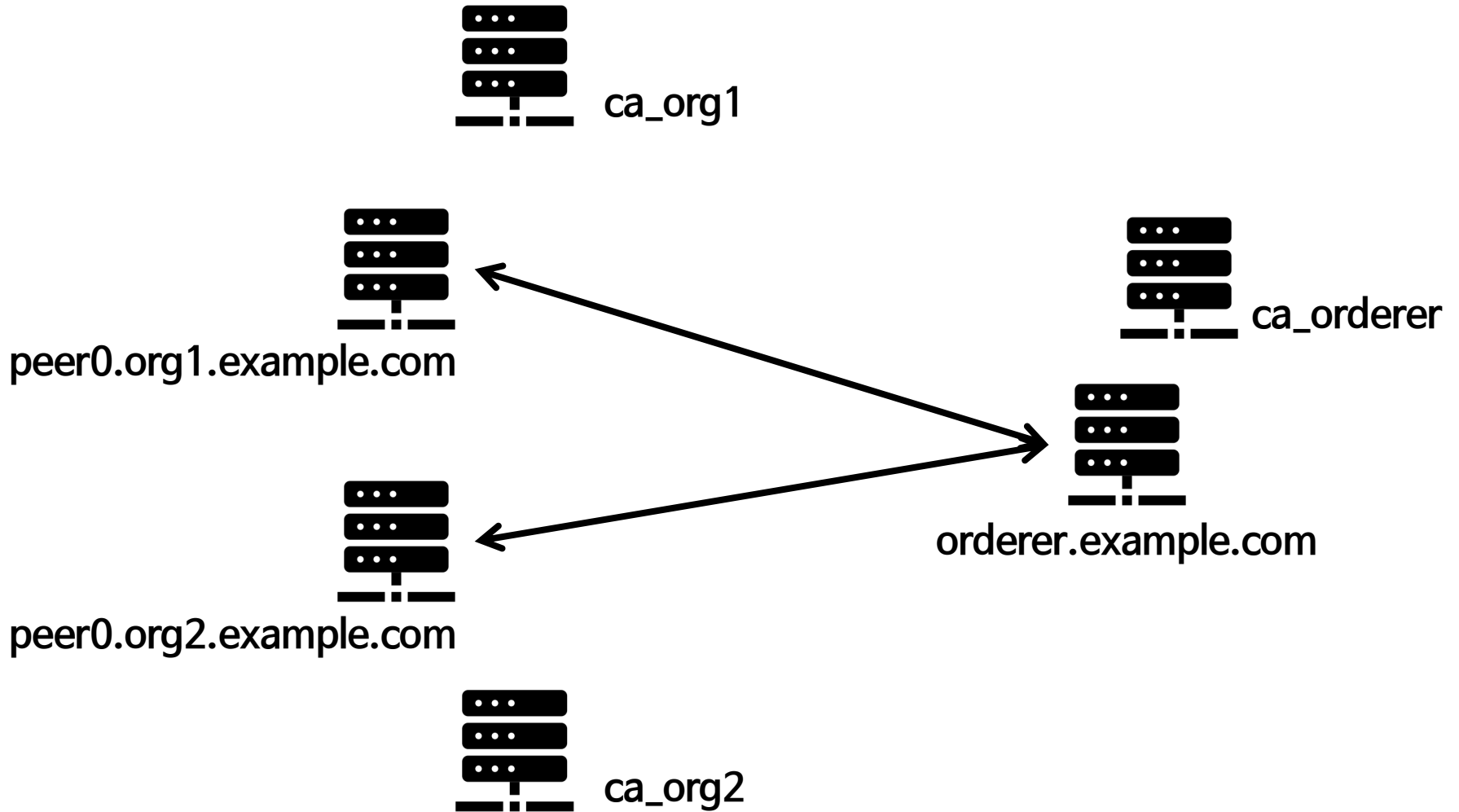
- Ubuntu 를 실행
- 다음 커맨드 입력
 - `cd ~/fabric-samples/test-network`
- 다음 커맨드 입력
 - `export COMPOSE_PROJECT_NAME=fabric`
 - `./network.sh up -ca`

test-network 실행

```
2022/02/12 19:55:40 [INFO] Stored root CA certificate at /home/ahnhwi/Workspace/fabric-samples/test-network/organizations/ordererOrganizations/example.com/users/Admin@example.com/msp/cacerts/localhost-9054-ca-orderer.pem
2022/02/12 19:55:40 [INFO] Stored Issuer public key at /home/ahnhwi/Workspace/fabric-samples/test-network/organizations/ordererOrganizations/example.com/users/Admin@example.com/msp/IssuerPublicKey
2022/02/12 19:55:40 [INFO] Stored Issuer revocation public key at /home/ahnhwi/Workspace/fabric-samples/test-network/organizations/ordererOrganizations/example.com/users/Admin@example.com/msp/IssuerRevocationPublicKey
Generating CCP files for Org1 and Org2
WARN[0000] Found orphan containers ([ca_orderer ca_org2 ca_org1]) for this project. If you removed or renamed this service in your compose file, you can run this command with the --remove-orphans flag to clean it up.
[+] Running 7/7
  # Volume "compose_orderer.example.com"      Created                                0.0s
  # Volume "compose_peer0.org1.example.com"    Created                                0.0s
  # Volume "compose_peer0.org2.example.com"    Created                                0.0s
  # Container peer0.org2.example.com           Started                                5.7s
  # Container orderer.example.com              Started                                5.6s
  # Container peer0.org1.example.com           Started                                5.6s
  # Container cli                              Started                                7.8s
```

CONTAINER ID	IMAGE	NAMES	COMMAND	CREATED	STATUS	PORTS
e533146c7ffe	hyperledger/fabric-tools:latest	cli	"/bin/bash"	7 seconds ago	Up Less than a second	
e63deaf83020	hyperledger/fabric-peer:latest	peer0.org2.example.com	"peer node start"	8 seconds ago	Up 3 seconds	0.0.0.0:9051->9051/tcp, 7051/tcp
27ed1dd6833f	hyperledger/fabric-peer:latest	peer0.org1.example.com	"peer node start"	8 seconds ago	Up 3 seconds	0.0.0.0:7051->7051/tcp, 0.0.0.0:9444->9444/tcp
76cdc04d6800	hyperledger/fabric-orderer:latest	orderer.example.com	"orderer"	8 seconds ago	Up 3 seconds	0.0.0.0:7050->7050/tcp, 0.0.0.0:7053->7053/tcp, 0.0.0.0:9443->9443/tcp
908fef29ff69	hyperledger/fabric-ca:latest	ca_orderer	"sh -c 'fabric-ca-se..."	17 seconds ago	Up 13 seconds	0.0.0.0:9054->9054/tcp, 7054/tcp
35670c83f8a7	hyperledger/fabric-ca:latest	ca_org2	"sh -c 'fabric-ca-se..."	17 seconds ago	Up 12 seconds	0.0.0.0:8054->8054/tcp, 7054/tcp
e85f0bfeb71	hyperledger/fabric-ca:latest	ca_org1	"sh -c 'fabric-ca-se..."	17 seconds ago	Up 14 seconds	0.0.0.0:7054->7054/tcp, 0.0.0.0:17054->17054/tcp

test-network 실행



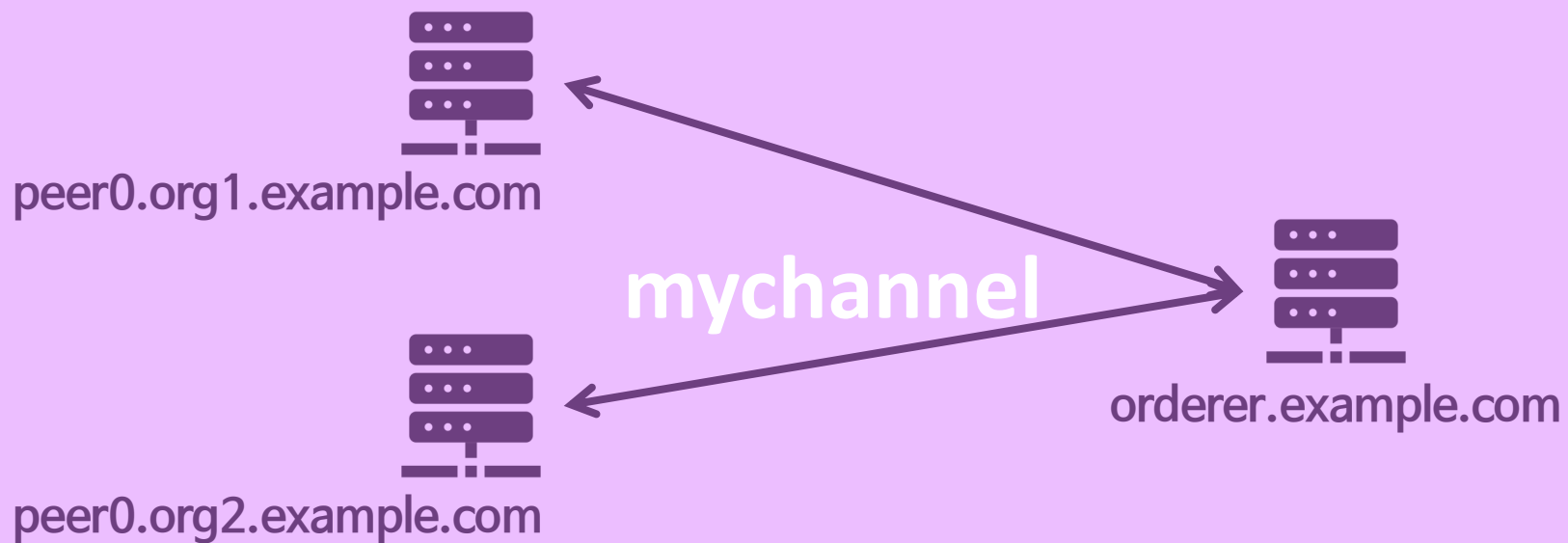
하이퍼레저 패브릭 실행

- 채널 생성
- 체인코드 패키징 및 배포

채널 생성

- `./test-network` 폴더에서 다음 커맨드 실행
 - `./network.sh createChannel`
 - 채널 생성, 피어 Join
- 채널 생성 직접 확인
 - `docker exec -it peer0.org1.example.com /bin/sh`
 - `cd /var/hyperledger/production/ledgersData/chains/chains`
 - `cd /var/hyperledger/production/ledgersData/stateLevelDb`
- 피어 노드 로그 확인
 - `docker logs peer0.org1.example.com`

채널 생성



체인코드 패키징 및 배포

- Prerequisite
- asset-transfer-basic 체인코드 배포 (JavaScript 버전)
 - `./network.sh deployCC -ccn basic -ccp ../asset-transfer-basic/chaincode-javascript -ccl javascript`

```
byron1st@DESKTOP-D240NBD:~/fabric-samples/test-network$ docker ps
```

CONTAINER ID	IMAGE	STATUS	PORTS
320cffffafe346	dev-peer0.org2.example.com-basic_1.0-3e6efb14d2337	Up 5 minutes	
1dad109710d8	dev-peer0.org1.example.com-basic_1.0-3e6efb14d2337	Up 5 minutes	
a74ab9620dc1	hyperledger/fabric-orderer:latest	Up 11 minutes	0.0.0.0:7050->7050/tcp
7fab09212cb8	hyperledger/fabric-peer:latest	Up 11 minutes	7051/tcp, 0.0.0.0:9051->9051/tcp
c281d2758196	hyperledger/fabric-peer:latest	Up 11 minutes	0.0.0.0:7051->7051/tcp

체인코드 패키징 및 배포

- **asset-transfer-basic** 체인코드 배포 (JavaScript 버전)
 - 체인코드 **package** - 체인코드 별로 수행
 - .tar.gz 형태로 메타 데이터와 함께 압축
 - `mkdir basic_cc && tar -xvf basic.tar.gz -C ./basic_cc`
 - 체인코드 **install** - Peer 별로 수행
 - 바이너리 형태로 Peer에 저장
 - 체인코드 **approve** - 기관 별로 수행
 - version: 코드 기준
 - sequence: 설정 등을 변경해서 재배포 가능
 - 체인코드 **commit** - 채널 별로 수행

체인코드 패키징 및 배포

- 체인코드 함수 호출

- 로컬에 있는 peer 바이너리를 이용하여 체인코드 함수를 호출 해볼 것임.
- 다음 커맨드들을 순서대로 호출 (test-network 폴더에서 진행)
- https://hyperledger-fabric.readthedocs.io/en/latest/test_network.html#interacting-with-the-network
- `export PATH=${PWD}/../bin:$PATH`
- `export FABRIC_CFG_PATH=$PWD/../config/`
- `export CORE_PEER_TLS_ENABLED=true`
- `export CORE_PEER_LOCALMSPID="Org1MSP"`
- `export CORE_PEER_TLS_ROOTCERT_FILE=${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`

체인코드 패키징 및 배포

- 체인코드 함수 호출

- `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":[]}'`

- docker logs 로그 확인

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

하이퍼레저 패브릭 Client Server 실행

- node.js 설치
- Client Server 실행

node.js 설치

- JavaScript

- 프론트엔드, 백엔드에서 사용이 가능한 프로그래밍 언어
- JavaScript 코드를 실행하기 위해서는 JavaScript 인터프리터 필요 -> JavaScript 엔진
 - Chrome (Google): V8
 - Firefox (Mozilla): SpiderMonkey
 - Safari (Apple): JavaScript Core
 - Edge (Microsoft): ~~Chakra Core~~ -> V8
- JavaScript 엔진을 웹 브라우저 밖에 설치하여 웹 브라우저 밖에서도 JavaScript 코드를 실행할 수 있도록 개발
 - V8: [node.js](https://nodejs.org/) (서버), Electron (데스크톱 어플리케이션)
 - JavaScript Core: React Native (iOS, Android 앱)

node.js 설치

- node.js 설치

- Linux, macOS: Terminal 실행
- Windows: Ubuntu 18.04 실행
- 다음 커맨드 입력
 - `curl -L https://raw.githubusercontent.com/tj/n/master/bin/n -o n`
- 다음 커맨드 입력
 - `sudo bash n install 14`
- 비밀번호 입력
- 14.19.0 버전이 설치됨
- 다음 커맨드 결과 버전 넘버가 잘 출력되는지 확인
 - `node --version`

Client Server 실행

- 다음 폴더로 이동

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

- 다음 커맨드 실행

- `sudo apt-get update`
 - `sudo apt-get install build-essential`
 - `npm install`

- 서버를 실행

- `npm run build`
 - `npm start`