Lab. Nodejs Web Application Container

1. https://www.fastify.io/docs/latest/Getting-Started/ 접속

2. 사전 Test

```
$ mkdir demo
$ cd demo
$ apt install npm
$ npm init
$ npm i fastify --save

$ vim app.js
  // Require the framework and instantiate it
  const fastify = require('fastify')({
    logger: true
  })

  // Declare a route
  fastify.get('/', function (request, reply) {
    reply.send({ hello: 'world' })
  })

  // Run the server!
  fastify.listen(3000, '0.0.0.0', function (err, address) {
    if (err) {
      fastify.log.error(err)
      process.exit(1)
    }
    fastify.log.info(`server listening on ${address}`)
  })

$ node app.js
-Web Browser에서 확인
  --http://{IP}:3000
```

3. Docker Image 생성하기

1)app.js 수정

```
  // Require the framework and instantiate it
  const fastify = require('fastify')({
    logger: true
  })

  // Declare a route
  fastify.get('/', function (request, reply) {
    reply.send({ hello: 'world' })
  })

  // Run the server!
  fastify.listen(3000, '0.0.0.0', function (err, address) {   <----여기 수정
    if (err) {
      fastify.log.error(err)
      process.exit(1)
    }
    fastify.log.info(`server listening on ${address}`)
  })
```

2)Dockerfile 생성하기

```
    # 1. nodejs 설치
    FROM       ubuntu:20.04
    RUN        apt-get update
    RUN        DEBIAN_FRONTEND=noninteractive apt-get -y install nodejs npm

    # 2. Source File 복사
    COPY       .  /usr/src/app

    # 3. Nodejs Packages 설치
    WORKDIR   /usr/src/app
    RUN        npm install

    # 4. Web Server 실행
    EXPOSE    3000
    CMD        node app.js
```

3).dockerignore 파일 생성

```
node_modules/*
```

4)Docker Image 생성

```
$ docker build -t myweb .
$ docker images

$ docker run --rm -d -p 3000:3000 myweb
```

-Web Browser 확인

5)Source Code 수정

-app.js

```
// Require the framework and instantiate it
const fastify = require('fastify')({
  logger: true
})

// Declare a route
fastify.get('/', function (request, reply) {
  reply.send({ hello: 'docker world' })                <---여기 수정
})

// Run the server!
fastify.listen(3000, '0.0.0.0', function (err, address) {
  if (err) {
    fastify.log.error(err)
    process.exit(1)
  }
  fastify.log.info(`server listening on ${address}`)
})
```

6)재 Build

```
$ docker build -t myweb .
$ docker images

$ docker run --rm -d -p 3000:3000 myweb
```

-Web Browser 확인

4. Dockerfile 최적화

1)Dockerfile 수정

```
    # 1. nodejs 설치
```

```
FROM  node:12     <---이미 Node가 설치되어 있는 이미지

# 2. Source File 복사
COPY        .  /usr/src/app

# 3. Nodejs Packages 설치
WORKDIR   /usr/src/app
RUN         npm install

# 4. Web Server 실행
EXPOSE    3000
CMD         node app.js
```

2)한번 build 하면 cache에 남아있기 때문에 소스코드가 변경되지 않으면 Cache 그냥 사용한다. 그래서 빨리 끝난다.
3)지금 느려지는 부분은 바로 Nodejs Packages 설치의 npm install 부분이다.
4)이것을 최적화할 수 있는데, 먼저 패키지 설치하고 소스코드 복사하면 그만큼 속도가 더 빨라진다.

```
# 1. nodejs 설치
FROM        node:12

# 2. 패키지 우선 복사
COPY        ./package* /usr/src/app/
WORKDIR   /usr/src/app
RUN         npm install
COPY        .  /usr/src/app

# 3. 소스코드 복사
COPY        .  /usr/src/app

# 4. Web Server 실행
EXPOSE    3000
CMD         node app.js
```

5)npm install까지 cache되어서 빌드가 빨라짐

6)용량이 너무 크면 alpine 버전을 사용할 것
-현재 하나의 이미지 용량이 거의 924MB

```
# 1. nodejs 설치
FROM        node:12-alpine
```

-거의 1/10으로 줄어듦.