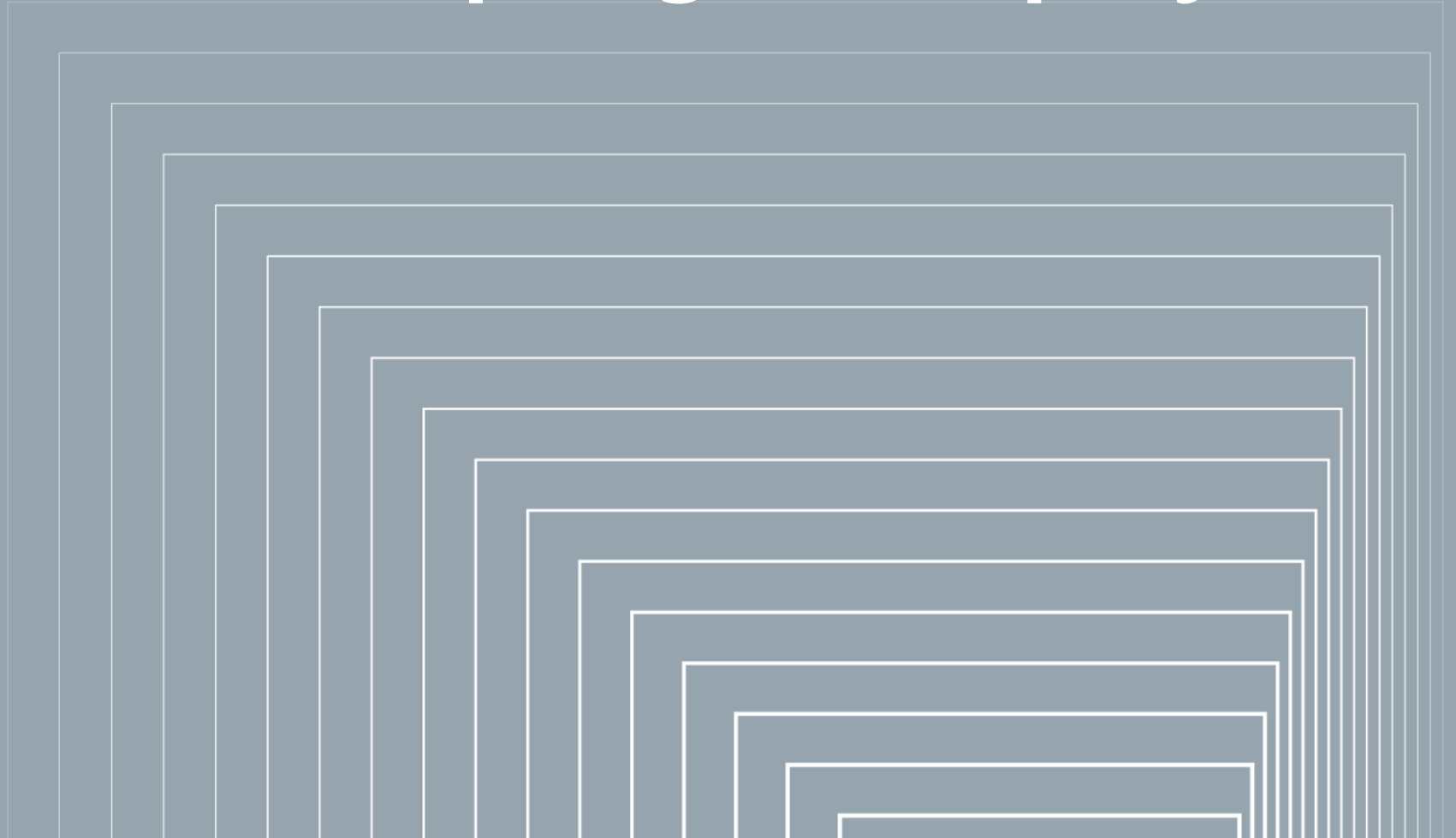


3. Initialize and Run Spring-Boot project



Spring Boot initializr

- <https://start.spring.io>

IntelliJ IDEA Ultimate

- community edition 에서는 지원하지 않음

Spring Tools 4 for Eclipse

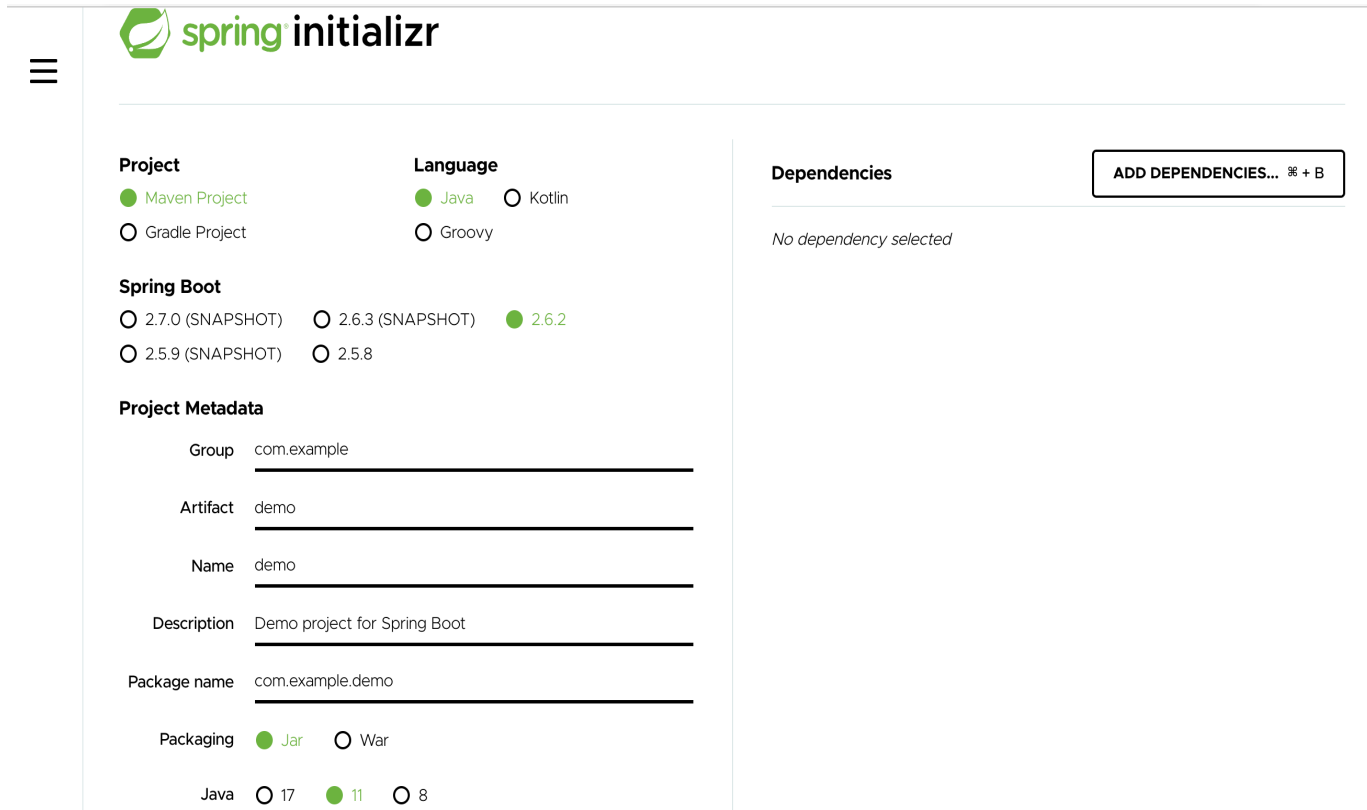
- eclipse 프로젝트 기반
- <https://spring.io/tools>

Spring Tools 4 for Visual Studio Code

- Spring Boot 확장팩 설치 후 사용가능
- [Spring Boot support in Visual Studio Code](#)

Spring Boot initializr

- 웹기반 Spring Boot 프로젝트 생성 도구
 - <https://start.spring.io>
- 선택 옵션
 - build tool
 - language,
 - spring-boot version
 - java version
 - 라이브러리 의존성(dependency)
- Spring Boot 프로젝트팀 통계 수집

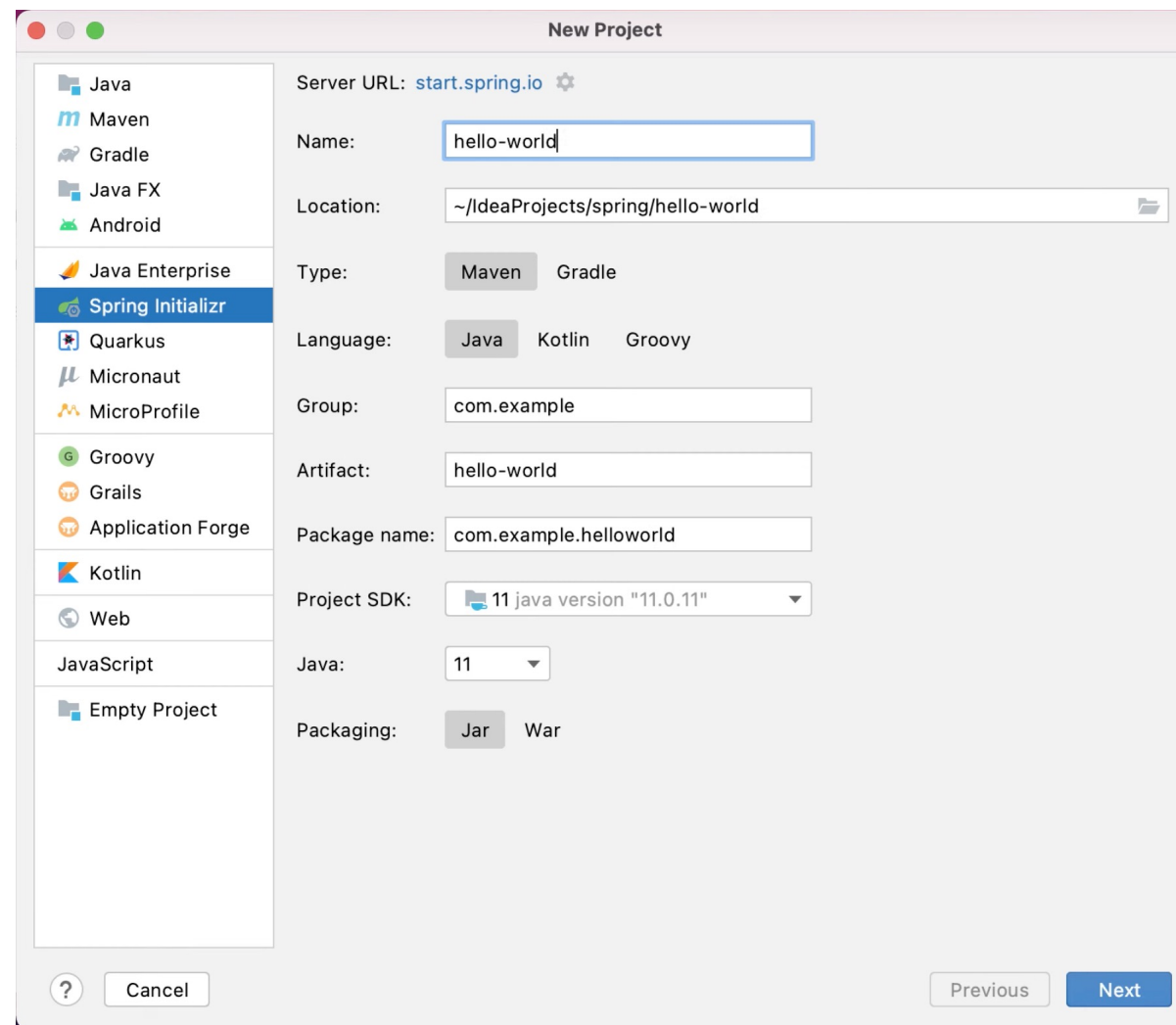


The screenshot shows the Spring Boot Initializr web application interface. The header includes the Spring logo and the text "spring initializr". A hamburger menu icon is on the left. The main content area is divided into several sections:

- Project**: Includes radio buttons for "Maven Project" (selected) and "Gradle Project".
- Language**: Includes radio buttons for "Java" (selected), "Kotlin", and "Groovy".
- Spring Boot**: Includes radio buttons for versions "2.7.0 (SNAPSHOT)", "2.6.3 (SNAPSHOT)", "2.6.2" (selected), "2.5.9 (SNAPSHOT)", and "2.5.8".
- Project Metadata**: Includes input fields for "Group" (com.example), "Artifact" (demo), "Name" (demo), "Description" (Demo project for Spring Boot), and "Package name" (com.example.demo).
- Packaging**: Includes radio buttons for "Jar" (selected) and "War".
- Java**: Includes radio buttons for versions "17", "11" (selected), and "8".
- Dependencies**: Includes a button "ADD DEPENDENCIES... ⌘ + B" and the text "No dependency selected".

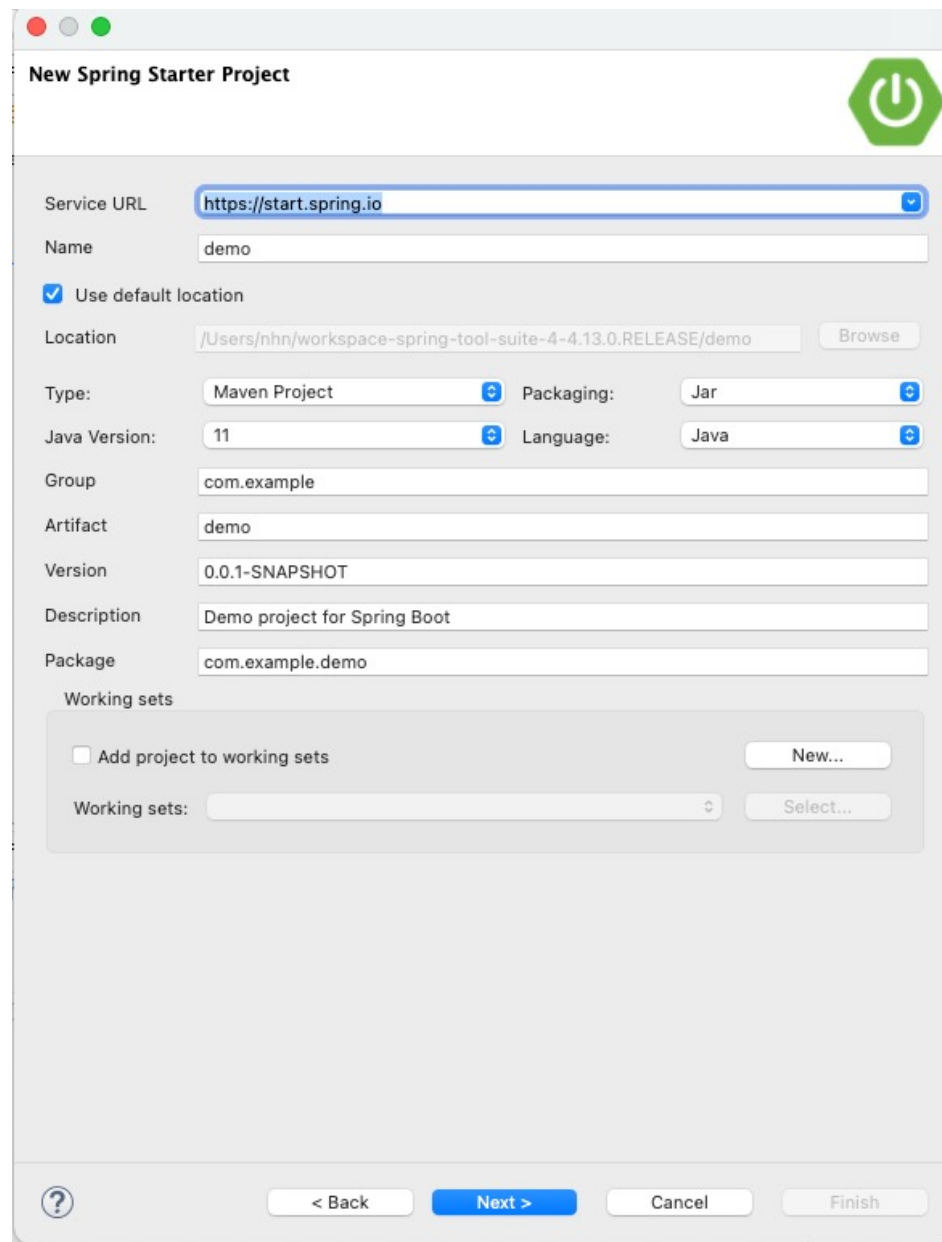
IntelliJ IDEA Ultimate

- 통합개발도구(IDE)
 - [Spring Boot | IntelliJ IDEA \(jetbrains.com\)](https://spring.io/projects/spring-boot#support)
 - 배포판에 포함된 Spring and SpringBoot 플러그인 사용
 - **Spring Boot initializr** 사용
- actuator endpoint 도구 제공
- Bean 조회 도구 제공



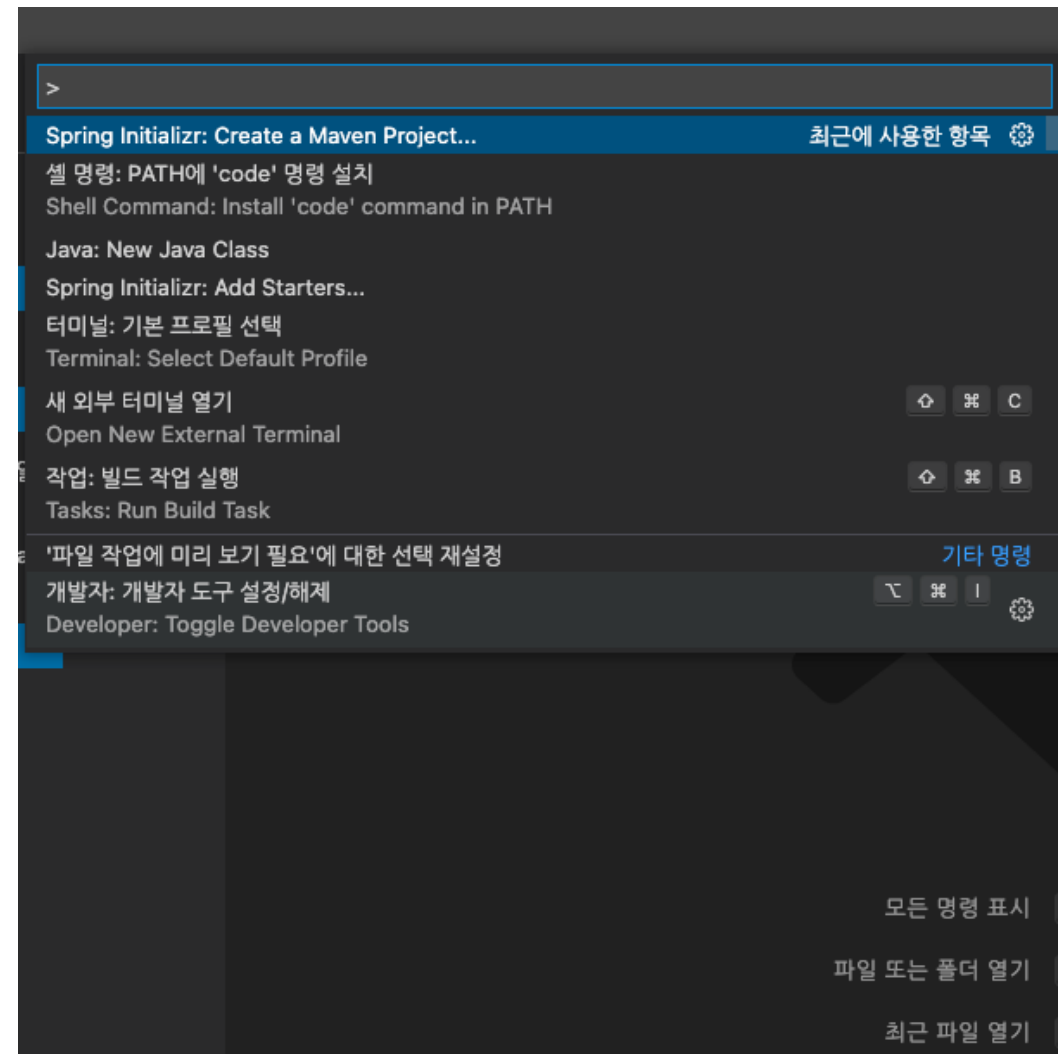
Spring Tools For Eclipse

- 통합개발도구(IDE)
 - Eclipse IDE 기반
 - spring 프로젝트에서 제공 [Spring | Tools](#)
 - Spring Tool Suite의 새버전



Spring Tools 4 for Visual Studio Code

- 통합개발도구(IDE)
 - Visual Studio Code IDE 기반
 - spring 프로젝트에서 제공 [Spring | Tools](#)
 - 확장팩 설치 후 사용
 - [Spring Boot Extension Pack - Visual Studio Marketplace](#)



Executable Jar/War

- 실행가능한 jar, war 생성

Build Tool

- maven , gradle 로 직접 실행

Unix/Linux Services

- init.d Service
- systemd Service

Docker/Kubernetes

- Docker Image 생성 지원

Executable Jar / War

- maven 또는 gradle 로 실행가능한 jar 또는 war 를 빌드한다.
- spring boot의 maven plugin 이나 gradle plugin 을 사용한다면 자동으로 생성할 수 있다.

```
$ mvn package. //gradle bootjar

$ ls target
student-0.0.1-SNAPSHOT.jar

$ java -jar target/student-0.0.1-SNAPSHOT.jar
```


Build Tool 사용

- maven 또는 gradle 로 직접 실행한다.
- 로컬, 개발환경에서 사용할 수 있다.

```
$ mvn spring-boot:run
```

```
$ gradle bootRun
```

Linux Services (CentOS, Ubuntu)

- Linux Service 에서 실행하려면 완전 실행가능한 jar 를 빌드한다.
- maven, gradle에서 아래와 같이 spring-boot plugin 설정을 수정한다.

```
<plugin>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-maven-plugin</artifactId>
  <configuration>
    <executable>true</executable>
  </configuration>
</plugin>
```

```
bootJar {
    launchScript()
}
```

Linux Services (CentOS, Ubuntu)

- init.d Service 설정 및 실행

```
$ sudo ln -s /var/app/student.jar /etc/init.d/student
```

```
$ service student start
```

Linux Services (CentOS, Ubuntu)

- systemd Service 설정 및 실행
- /etc/systemd/system/student.service 파일을 생성한다.

```
[Unit]
Description=student
After=syslog.target

[Service]
User=irteam
ExecStart=/var/app/student.jar
SuccessExitStatus=143

[Install]
WantedBy=multi-user.target
```

```
$ systemctl enable student.service
```

Docker 실행

- Dockerfile 을 직접 만들거나 빌드툴로 Docker 이미지를 생성한다.

```
$ mvn spring-boot:build-image -Dspring-boot.build-image.imageName=student
```

```
$ gradle bootBuildImage --imageName=student
```

- Docker 로 컨테이너 실행

```
$ docker run -p8080:8080 student:latest
```

Kubernetes 실행

- tag 설정 및 registry 설정.

```
$ docker tag student:0.0.1-SNAPSHOT registry.op.internal.dooray.io/nhn-edu/student:latest  
$ docker push registry.op.internal.dooray.io/nhn-edu/student:latest
```

- kubernetes 배포용 YAML 작성.

```
$ kubectl create deployment student --image=registry.op.internal.dooray.io/nhn-edu/student:latest --dry-run=client -o yaml > student.yaml
```

Kubernetes 실행

- Kubernetes POD 배포

```
$ kubectl apply -f student.yaml  
deployment.apps/student created
```

- 8080 포트 노출.

```
$ kubectl expose deployment student --type=NodePort --name=student-service --port  
8080
```

목표

- 이전 실습에서 구현한 Account 서비스를 이용하여 Docker 이미지를 만들고 실행하여 결과를 확인한다.

예상시간

- 10분

```
$ curl -XGET http://localhost:8080/accounts
```

```
[{"id":1,"number":"123","balance":10000}, {"id":2,"name":"124","score":20000}]
```