

Spring Cloud로 개발하는 マイクロ서비스 アプリケイ션



Microservices

+



Spring
Cloud

```
class Book {
    def self, title, price, author;
    self.title = title
    self.price = price
    self.author = author
}

public static void main(String[] args)
{
    var fs = require('fs');
    fs.readFile('/JONE.txt' /* 1 */,
        function (err, data) {
            console.log(data); // 3
        });
}

<@interface NextInnovationDelegate : NSObject <UIApplicationDelegate> >

<@implementation NextInnovationDelegate >
<@end>
```



목차

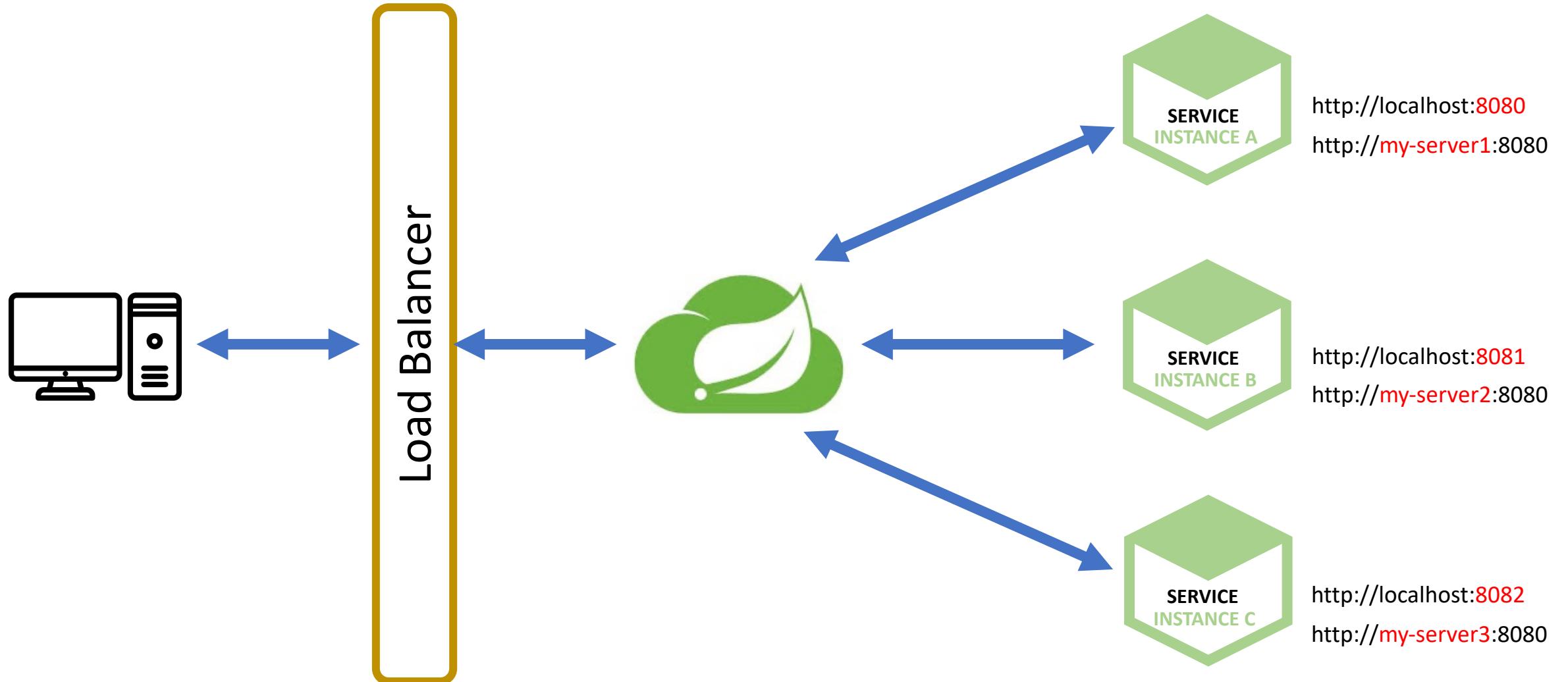
Part II

- Section 0: Microservice와 Spring Cloud 소개
- **Section 1: Service Discovery**
- Section 2: API Gateway Service
- Section 3: E-commerce 애플리케이션
- Section 4: Users Microservice - ①
- Section 5: Catalogs, Orders Microservice
- Section 6: Users Microservice - ②
- Section 7: Configuration Service
- Section 8: Spring Cloud Bus

Section 1. Service Discovery

- Spring Cloud Netflix Eureka
- Eureka Service Discovery – 프로젝트 생성
- User Service – 프로젝트 생성
- User Service – 등록
- User Service – Load Balancer

Spring Cloud Netflix Eureka





EcommerceDiscoveryService

- IntelliJ

- Create New Project > Spring Initializr

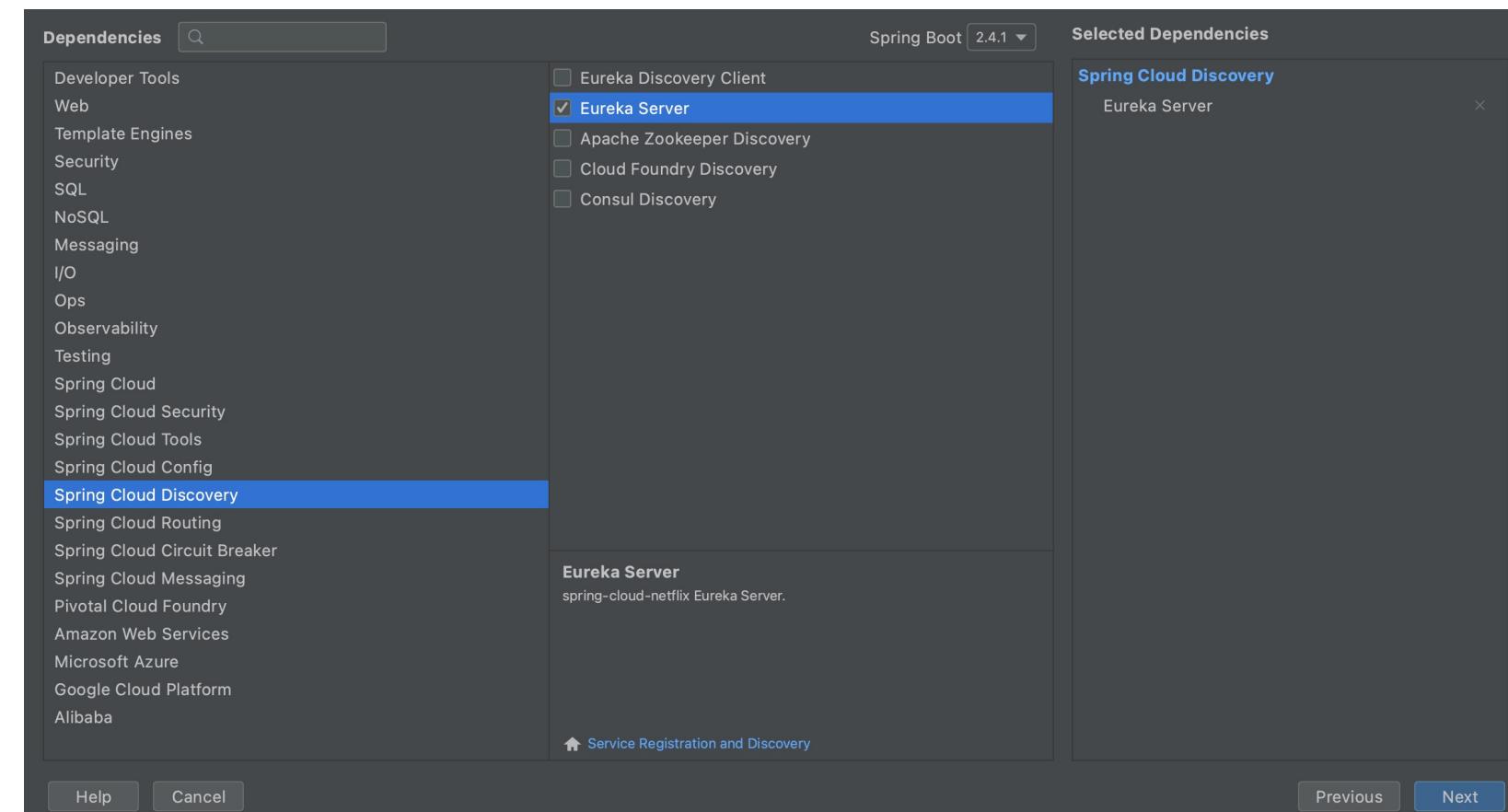
Spring Initializr Project Settings

Group:	com.example
Artifact:	ecoomerce
Type:	Maven Project (Generate a Maven based project archive.)
Language:	Java
Packaging:	Jar
Java Version:	11
Version:	0.0.1-SNAPSHOT
Name:	ecoomerce
Description:	Demo project for Spring Boot
Package:	com.example.ecoomerce

EcommerceDiscoveryService

■ Dependencies

- Spring Boot > 2.4.1
- Spring Cloud Discovery > Eureka Server





EcommerceDiscoveryService

- Spring Cloud 2020.0.X
 - Spring Boot > 2.4.1

The screenshot shows the official Spring Cloud website. At the top, there's a navigation bar with links for "Why Spring", "Learn", "Projects", "Training", "Support", and "Community". On the far right of the nav bar is a gear icon. Below the nav bar, the main title "Spring Cloud" is displayed in large bold letters, with a green button next to it labeled "2020.0.0". To the right of the title are two icons: one for GitHub and another for Slack. On the left side of the page, there's a sidebar with links for "Spring Boot", "Spring Framework", "Spring Data" (which has a dropdown arrow), and "Spring Cloud" (which is highlighted with a black background). Under "Spring Cloud", there's a list of sub-links: "Spring Cloud Azure", "Spring Cloud Alibaba", "Spring Cloud for Amazon Web Services", "Spring Cloud Bus", "Spring Cloud CLI", and "Spring Cloud for Cloud Foundry". The main content area on the right contains a paragraph about what Spring Cloud does, mentioning tools for building distributed systems like configuration management, service discovery, circuit breakers, and intelligent routing.

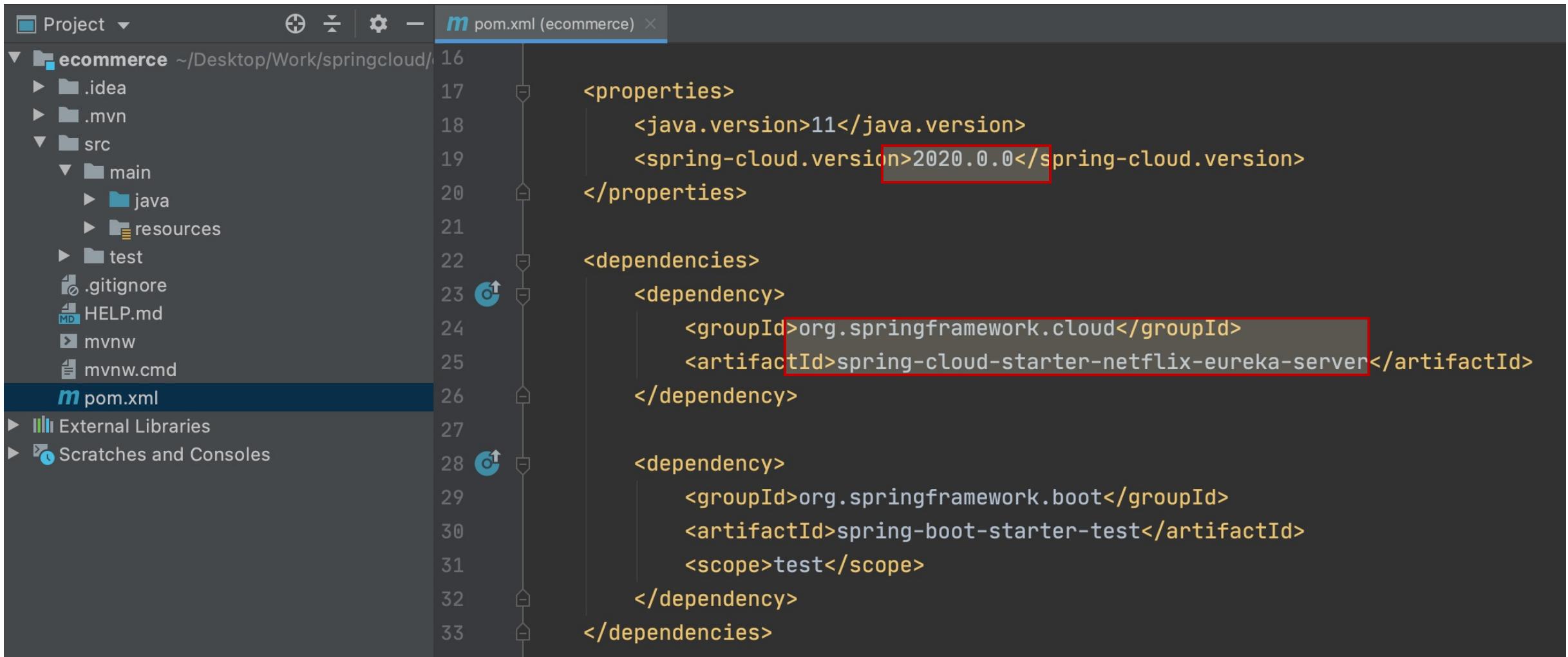
Spring Cloud 2020.0.0

OVERVIEW LEARN SAMPLES

Spring Cloud provides tools for developers to quickly build some of the common patterns in distributed systems (e.g. configuration management, service discovery, circuit breakers, intelligent routing, micro-proxy, control bus, one-time tokens, global locks, leadership election, distributed sessions, cluster state). Coordination of distributed systems leads to boiler plate patterns, and using Spring Cloud developers can quickly stand up services and applications that implement those patterns. They will work well in any distributed environment, including the developer's own laptop, bare metal data centres, and managed platforms such as Cloud Foundry.

EcommerceDiscoveryService

■ pom.xml



The screenshot shows a Java IDE interface with the following details:

- Project Bar:** Shows the project name "ecommerce" and its path: ~/Desktop/Work/springcloud/.
- File Explorer:** Displays the project structure with folders ".idea", ".mvn", "src" (containing "main" and "test"), and files ".gitignore", "HELP.md", "mvnw", and "mvnw.cmd".
- Code Editor:** The file "pom.xml" is open. The code content is as follows:

```
<properties>
    <java.version>11</java.version>
    <spring-cloud.version>2020.0.0</spring-cloud.version>
</properties>

<dependencies>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>
```

Two specific lines in the XML code are highlighted with red boxes:

- The line containing the Spring Cloud version: <spring-cloud.version>2020.0.0</spring-cloud.version>
- The line containing the artifact ID for the Netflix Eureka Server dependency: <artifactId>spring-cloud-starter-netflix-eureka-server</artifactId>



EcommerceDiscoveryService

- EcommerceApplication.java

```
7  @SpringBootApplication
8  @EnableEurekaServer
9  public class DiscoveryserviceApplication {
10
11     public static void main(String[] args) { SpringApplication.run(DiscoveryserviceApplication.class, args); }
12
13 }
```



EcommerceDiscoveryService

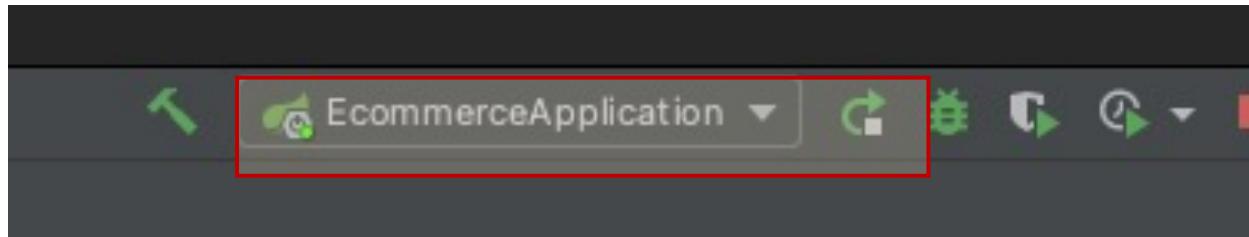
- application.yml (or application.properties)

```
1  - server:  
2    - port: 8761  
3  
4  - spring:  
5    - application:  
6      name: discoveryservice  
7  
8  - eureka:  
9    - client:  
10      register-with-eureka: false  
11      fetch-registry: false
```



EcommerceDiscoveryService

■ 실행



```
o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8761 (http) with context path ''
.s.c.n.e.s.EurekaAutoServiceRegistration : Updating port to 8761
c.e.ecommerce.EcommerceApplication       : Started EcommerceApplication in 3.252 seconds (JVM running for 3.799)
o.a.c.c.C.[Tomcat].[localhost].[/]        : Initializing Spring DispatcherServlet 'dispatcherServlet'
o.s.web.servlet.DispatcherServlet         : Initializing Servlet 'dispatcherServlet'
o.s.web.servlet.DispatcherServlet         : Completed initialization in 1 ms
```

EcommerceDiscoveryService

■ 실행화면

- <http://localhost:8761>

The screenshot shows the Spring Eureka dashboard at localhost:8761. The top navigation bar includes links for HOME and LAST 1000 SINCE STARTUP, along with various browser icons. The main content area displays the System Status and DS Replicas sections.

System Status

Environment	N/A
Data center	N/A

DS Replicas

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
No instances available			

User Service

- Eureka Discovery Service에 등록

The screenshot shows the Spring Initializr web application interface. On the left, there's a sidebar with various project categories: Developer Tools, Web, Template Engines, Security, SQL, NoSQL, Messaging, I/O, Ops, Observability, Testing, Spring Cloud, Spring Cloud Security, Spring Cloud Tools, Spring Cloud Config, Spring Cloud Discovery (which is highlighted with a blue bar), Spring Cloud Routing, and Spring Cloud Circuit Breaker. In the center, under the 'Dependencies' tab, there's a search bar and a dropdown for 'Spring Boot 2.4.1'. Below the search bar, there's a list of discovery service options: Eureka Discovery Client (selected with a checked checkbox), Eureka Server, Apache Zookeeper Discovery, Cloud Foundry Discovery, and Consul Discovery. On the right, a 'Selected Dependencies' panel is shown, which includes sections for Developer Tools (Spring Boot DevTools, Lombok), Web (Spring Web), and Spring Cloud Discovery (Eureka Discovery Client). A red box highlights the 'Selected Dependencies' panel.

Dependencies

Spring Boot 2.4.1

Eureka Discovery Client

Eureka Server

Apache Zookeeper Discovery

Cloud Foundry Discovery

Consul Discovery

Selected Dependencies

Developer Tools

Spring Boot DevTools

Lombok

Web

Spring Web

Spring Cloud Discovery

Eureka Discovery Client



User Service

■ pom.xml

```
<properties>
    <java.version>11</java.version>
    <spring-cloud.version>2020.0.0</spring-cloud.version>
</properties>

<dependencies>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-netflix-eureka-client</artifactId>
    </dependency>
</dependencies>
```

```
<dependencyManagement>
    <dependencies>
        <dependency>
            <groupId>org.springframework.cloud</groupId>
            <artifactId>spring-cloud-dependencies</artifactId>
            <version>${spring-cloud.version}</version>
            <type>pom</type>
            <scope>import</scope>
        </dependency>
    </dependencies>
</dependencyManagement>
```



User Service

- application.yml

```
7 @SpringBootApplication
8     @EnableDiscoveryClient
9 public class UserServiceApplication {
10
11     public static void main(String[] args) { SpringApplication.run(UserServiceApplication.class, args); }
12
13 }
```



User Service

- application.yml

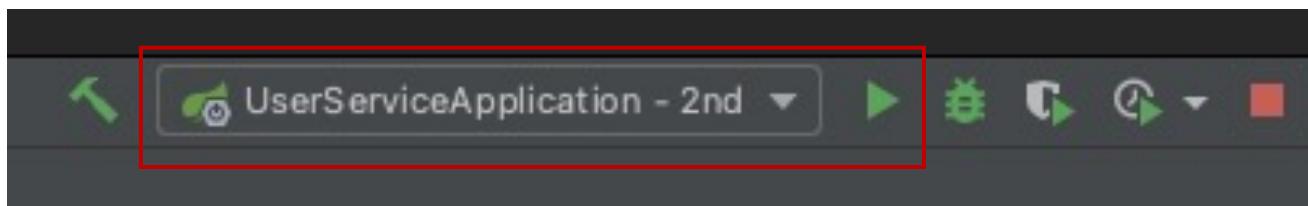
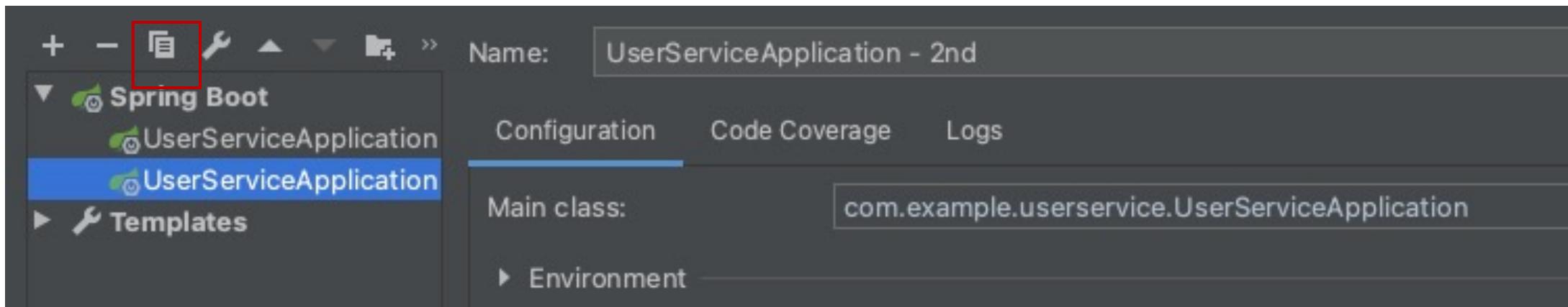
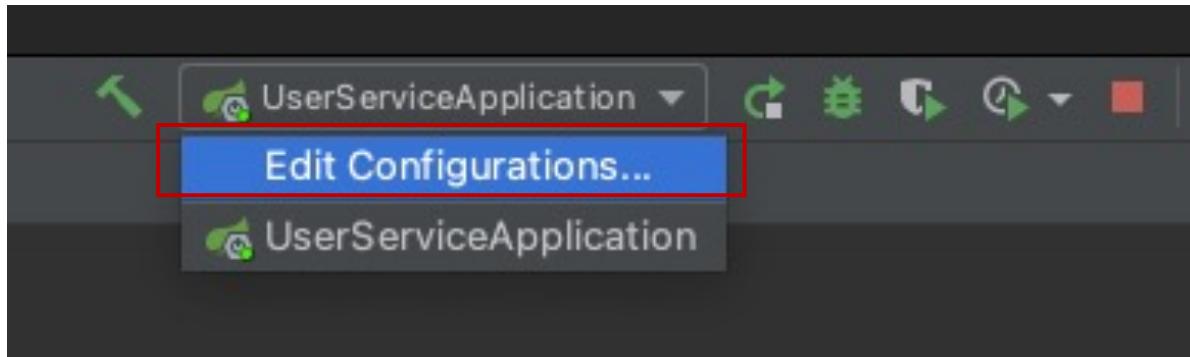
```
1  server:  
2    port: 9001  
3  
4  spring:  
5    application:  
6      name: user-service  
7  
8  eureka:  
9    client:  
10      register-with-eureka: true  
11      fetch-registry: true  
12      service-url:  
13        defaultZone: http://localhost:8761/eureka
```

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
USER-SERVICE	n/a (1)	(1)	UP (1) - 10.90.1.91:user-service:9001

User Service

- Scaling – 같은 서비스 추가 실행 #1





User Service

- Scaling – 같은 서비스 추가 실행 #1

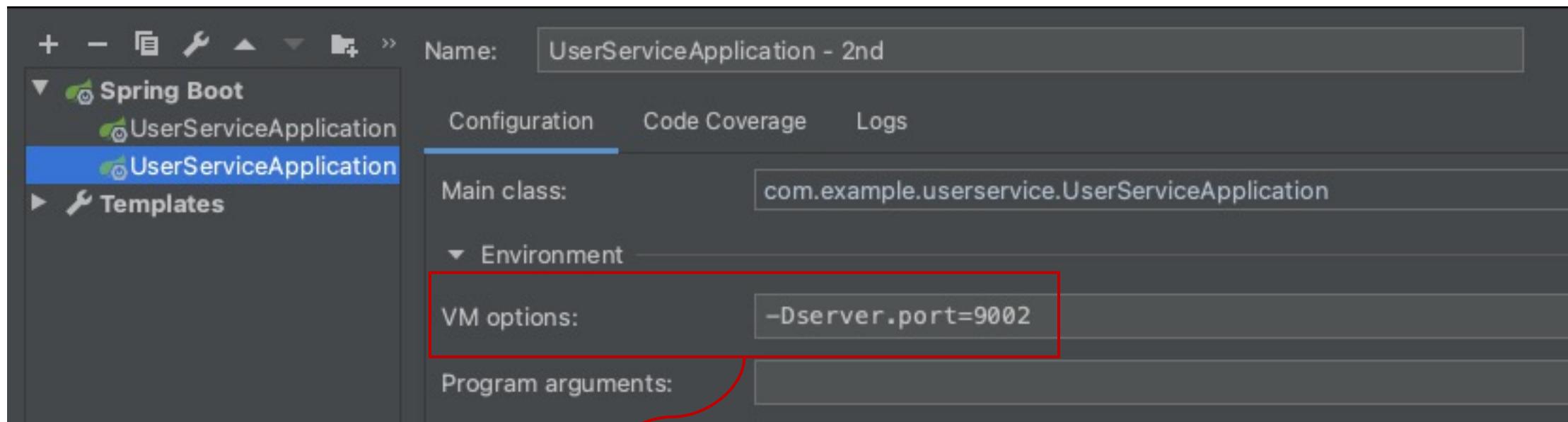
- **Port 충돌**

```
Error starting ApplicationContext. To display the conditions report re-run your application with 'debug' enabled.  
2021-01-15 10:54:05.701 ERROR 68628 --- [ restartedMain] o.s.b.d.LoggingFailureAnalysisReporter :  
*****  
APPLICATION FAILED TO START  
*****  
  
Description:  
  
Web server failed to start. Port 9001 was already in use.
```



User Service

- Scaling – 같은 서비스 추가 실행 #1
 - VM Options → -Dserver.port=[다른포트]



```
: Tomcat started on port(s): 9002 (http) with context path ''
: Updating port to 9002
: DiscoveryClient_USER-SERVICE/10.90.1.91:user-service:9002 - registration
: Started UserServiceApplication in 2.881 seconds (JVM running for 3.455)
```



User Service

- Scaling – 같은 서비스 추가 실행 #1

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
USER-SERVICE	n/a (2)	(2)	UP (2) - 10.90.1.91:user-service:9001 , 10.90.1.91:user-service:9002

- Scaling - 같은 서비스 추가 실행 #2

```
$ mvn spring-boot:run -Dspring-boot.run.jvmArguments=''-Dserver.port=9003'
```

- Scaling - 같은 서비스 추가 실행 #3

```
$ mvn clean compile package
```

```
$ java -jar -Dserver.port=9004 ./target/user-service-0.0.1-SNAPSHOT.jar
```



User Service

- Scaling – 같은 서비스 추가 실행

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
USER-SERVICE	n/a (4)	(4)	UP (4) - 10.90.1.91:user-service:9003 , 10.90.1.91:user-service:9004 , 10.90.1.91:user-service:9001 , 10.90.1.91:user-service:9002



UP (4) - 10.90.1.91:user-service:9003 , 10.90.1.91:user-service:9004 , 10.90.1.91:user-service:9001 , 10.90.1.91:user-service:9002



User Service

- Scaling – Random 포트 사용으로 같은 서비스 추가 실행
 - application.yml

```
1  server:  
2    port: 0  
3  
4  spring:  
5    application:  
6      name: user-service  
7
```

```
Spring Cloud LoadBalancer is currently working with the default  
Tomcat started on port(s): 55002 (http) with context path ''  
Updating port to 55002
```

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

```
Spring Cloud LoadBalancer is currently working with the default  
Tomcat started on port(s): 55019 (http) with context path ''  
Updating port to 55019
```



User Service

- Scaling – Random 포트 사용으로 같은 서비스 추가 실행
 - 같은 서비스 명으로 인해 Eureka에 하나의 앱만 등록

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
USER-SERVICE	n/a (1)	(1)	UP (1) - 10.90.1.91:user-service:0



User Service

- Scaling – Random 포트 사용으로 같은 서비스 추가 실행
 - application.yml에 instance 정보 등록
 - *eureka.instance.instanceId*

```
1  server:  
2    port: 0  
3  
4  spring:  
5    application:  
6      name: user-service  
7  
8  eureka:  
9  
10   instance:  
11     instanceId: ${spring.cloud.client.hostname}:${spring.application.instance_id:${random.value}}  
12  
13   client:  
14     register-with-eureka: true  
15     fetch-registry: true
```



User Service

- Scaling – Random 포트 사용으로 같은 서비스 추가 실행
 - application.yml에 instance 정보 등록
 - **eureka.instance.instanceId**

Instances currently registered with Eureka

Application	AMIs	Availability Zones	Status
USER-SERVICE	n/a (2)	(2)	UP (2) - 10.90.1.91:c1ee7e1823a74cd3d69e63dcb7153bec , 10.90.1.91:4cf342c93cde5ba18a1da6155fe292f4

10.90.1.91:55377/actuator/info

