

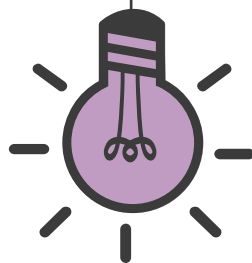
Spring Framework



S p r i n g F r a m e w o r k

01

스프링 프레임 워크



1. Spring의 등장 배경

- EJB를 사용하면 애플리케이션 작성을 쉽게 할 수 있다.
- 저 수준의 트랜잭션이나 상태관리, 멀티 쓰레딩, 리소스 풀링과 같은 복잡한 저 수준의 API 따위를 이해하지 못하더라도 아무 문제 없이 애플리케이션을 개발할 수 있다.

– Enterprise JavaBean 1.0 Specification, Chapter 2 Goals

- **EJB의 장점**

- 웹사이트가 점점 커지면서 EJB가 자바 진영에서 엔터프라이즈급의 서비스를 제공
- 세션빈에서 Transaction 관리가 용이하고 로그인, 분산처리, 보안등에 적합

- **EJB의 문제점**

- 코드 수정 후 반영하는 과정 자체가 거창해 기능은 좋지만 현실에서의 반영은 어렵다.
- 복잡한 스펙으로 인한 개발의 효율성이 떨어진다.
- 어플리케이션을 테스트하기 위해서는 반드시 EJB서버가 필요하다.
- EJB스펙에 정의된 인터페이스에 따라 코드를 작성하므로 기존에 작성된 POJO를 변경해야 한다.
- 컨테이너에 배포를 해야 테스트가 가능하기 때문에 개발속도가 저하된다.
- 배우기 어렵고, 설정해야 할 부분이 많다.

- **해결방안**

- Rod Johnson이 'Expert One-on-One J2EE Development without EJB'라는 저서에서 EJB를 사용하지 않고 엔터프라이즈 어플리케이션을 개발하는 방법을 소개하였다.

(스프링의 모태)

- AOP나 DI같은 새로운 프로그래밍 방법론으로 가능하다.
- POJO로 선언적인 프로그래밍 모델이 가능해 졌다.

- **POJO(Plain Old Java Object)**

- 특정 프레임워크나 기술에 의존적이지 않은 자바 객체
- 특정 기술에 종속적이지 않기 때문에 생산성, 이식성 향상
- Plain : component interface를 상속받지 않는 특징 (특정 framework에 종속되지 않음)
- Old : EJB 이전의 java class를 의미

- **경량 프레임 워크**

- EJB가 제공하는 서비스를 지원해 줄 수 있는 프레임워크 등장
- Hibernate, JDO, iBatis(MyBatis), Spring

- **POJO + Framework**

- EJB서버와 같은 거창한 컨테이너가 필요 없다.
- 오픈소스 프레임워크라 사용이 무료이다.
- 각종 기업용 어플리케이션 개발에 필요한 상당히 많은 라이브러리가 지원한다.
- 스프링프레임워크는 모든 플랫폼에서 사용이 가능하다.
- 스프링은 웹 분야 뿐만이 아니라 어플리케이션 등 모든 분야에 적용이 가능한 다양한 라이브러리를 가지고 있다.

2. Spring Framework

- 엔터프라이즈급 애플리케이션을 만들기 위한 모든 기능을 종합적으로 제공하는 경량화 된 솔루션이다.
- JEE(Java Enterprise Edition)가 제공하는 다수의 기능을 지원하고 있기 때문에, JEE를 대체하는 Framework로 자리잡고 있다.
- Spring Framework는 JEE가 제공하는 다양한 기능을 제공하는 것 뿐만 아니라, **DI(Dependency Injection)**나 **AOP (Aspect Oriented Programming)**와 같은 기능도 지원 한다.
- Spring Framework는 자바로 Enterprise Application을 만들 때 포괄적으로 사용하는 Programming 및 Configuration Model을 제공해 주는 Framework로 Application 수준의 인프라 스트럭처를 제공한다.
- 즉, 개발자가 복잡하고 실수하기 쉬운 Low Level에 신경 쓰지 않고 Business Logic개발에 전념 할 수 있도록 해준다.

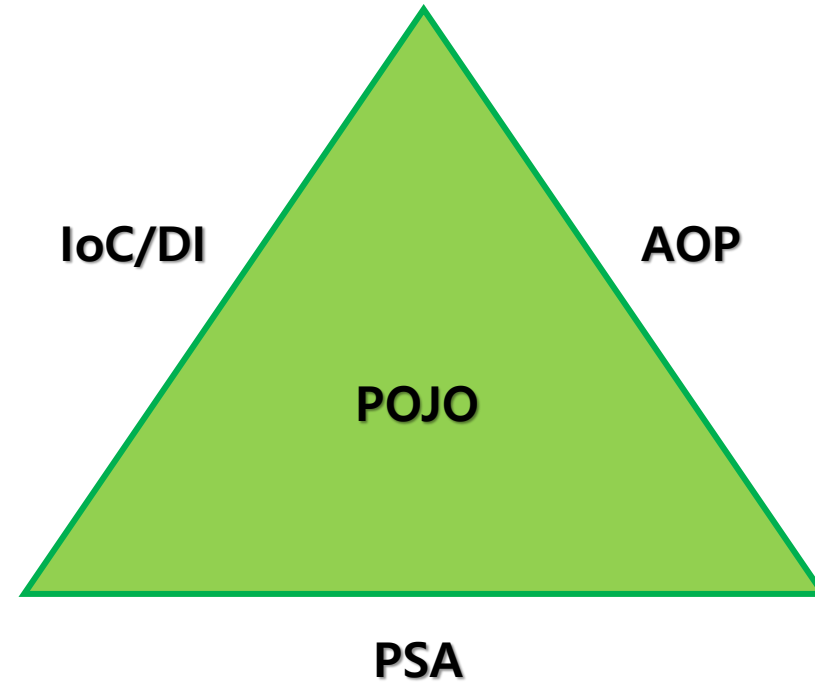
- Spring Framework

- **POJO**

- PSA

- IoC/DI

- AOP



- **POJO (Plain Old Java Object)**

- 테스트하기 용이하며, 객체지향 설계를 자유롭게 적용할 수 있다.

- 특정 환경이나 기술에 종속적이지 않은 객체지향 원리에 충실한 자바객체 이다.

- Spring Framework

- POJO

- **PSA**

- IoC/DI

- AOP

- **PSA (Portable Service Abstraction)**

- 환경과 세부기술의 변경과 관계없이 일관된 방식으로 기술에 접근할 수 있게 해주는 설계 원칙.
 - 트랜잭션 추상화, OXM 추상화, 데이터 액세스의 Exception 변환기능 등 기술적인 복잡함은 추상화를 통해 Low Level의 기술 구현 부분과 기술을 사용하는 인터페이스로 분리.
 - 예를 들어, 데이터베이스에 관계없이 동일하게 적용 할 수 있는 트랜잭션 처리방식.

- Spring Framework

- POJO

- PSA

- **IoC/DI**

- AOP

- **IoC/DI (Dependency Injection)**

- DI는 유연하게 확장 가능한 객체를 만들어 두고 객체 간의 의존관계는 외부에서 다이나믹하게 설정

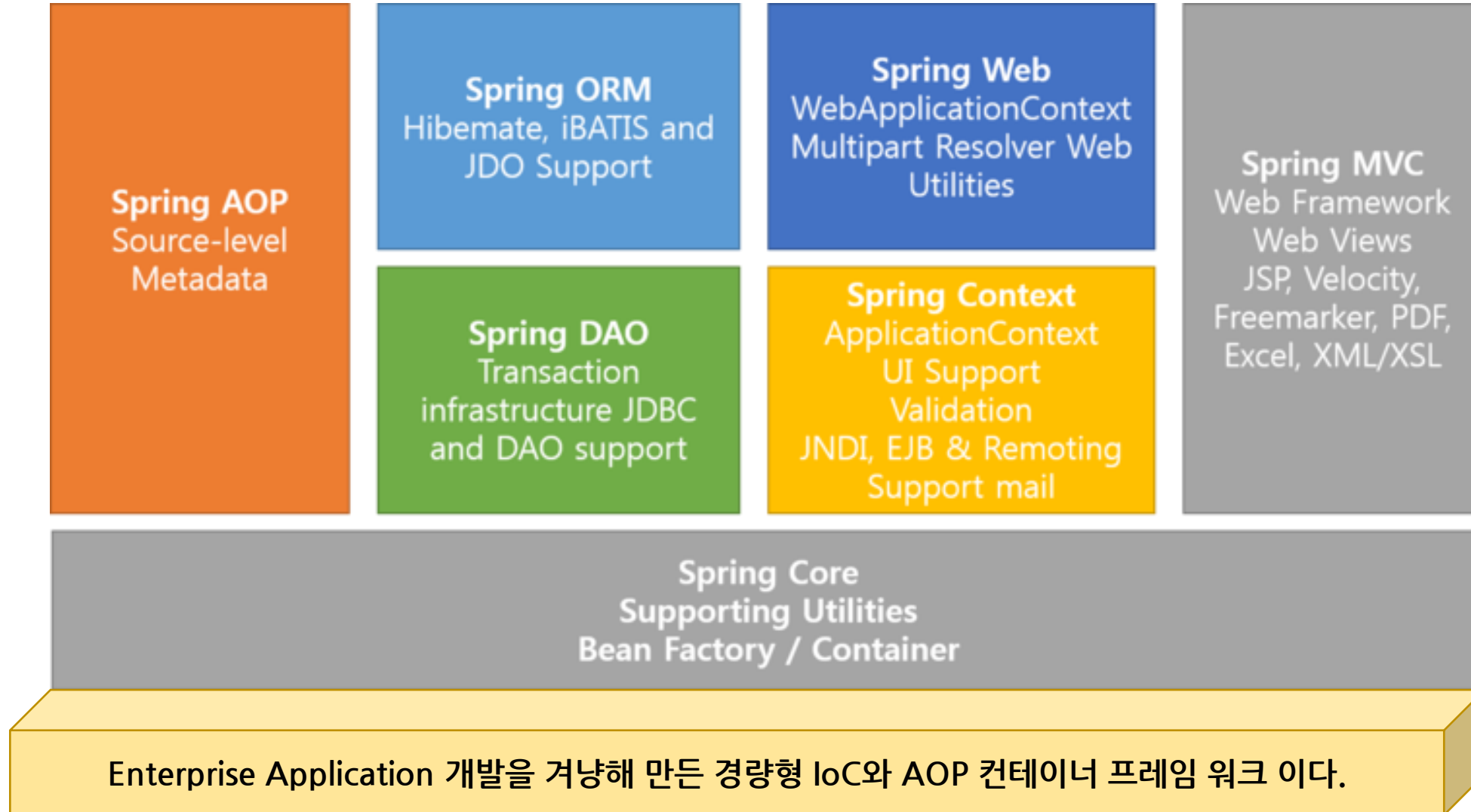
- Spring Framework

- POJO
- PSA
- IoC/DI
- **AOP**

- **AOP (Aspect Oriented Programming)**

- 관심사의 분리를 통해서 소프트웨어의 모듈성을 향상
- 공통 모듈을 여러 코드에 쉽게 적용가능

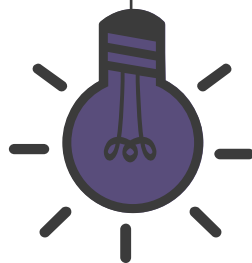
3. Spring Framework Module(1/2)



3. Spring Framework Module(2/2)

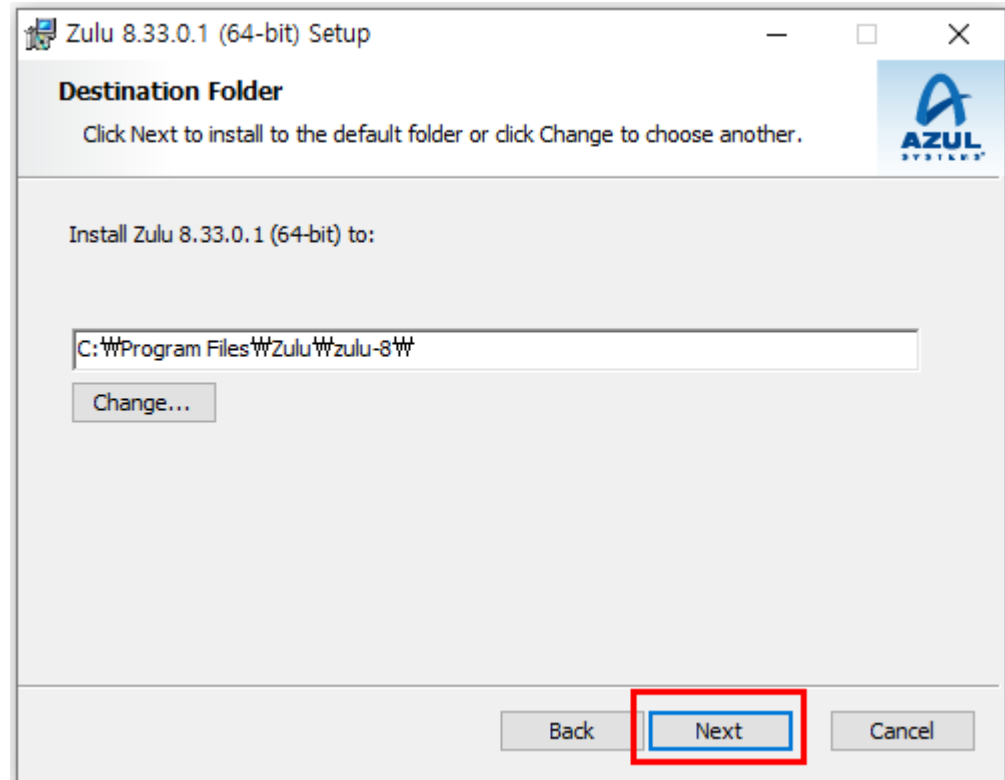
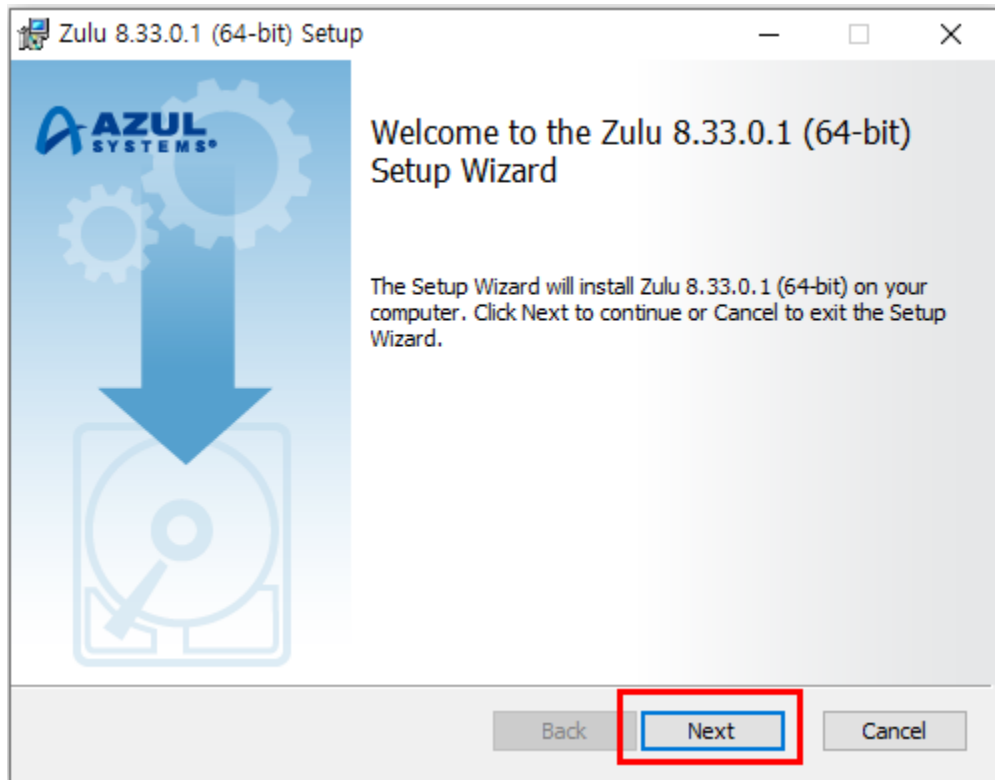
1. Spring Core	Spring Framework의 핵심기능을 제공한다. Core 컨테이너의 주요 컴포넌트는 Bean Factory이다
2. Spring Content	Spring을 컨테이너로 만든 것이 Spring Core의 Bean Factory라면 Spring을 Framework로 만든 것은 Context module이다. 이 module은 국제화된 메시지 Application 생명주기 이벤트, 유효성 검증 등을 지원함으로써 BeanFactory의 개념을 확장한다
3. Spring AOP	설정 관리기능을 통해 AOP기능을 Spring Framework와 직접 통합 시킨다.
4. Spring DAO	Spring JDBC DAO추상 레이어는 다른 데이터베이스 벤더들의 예외 핸들링과 오류 메시지를 관리하는 중요한 예외 계층을 제공한다.
5. Spring ORM	Spring Framework는 여러 ORM(Object Relational Mapping) Framework에 플러그인 되어, Object Relation 툴(JDO, Hibernate, iBatis)을 제공한다.
6. Spring Web	Web Context module은 Application Context module 상위에 구현되어 Web기반 Application Context를 제공한다
7. Spring Web MVC	Spring Framework는 자체적으로 MVC프레임워크를 제공하고 있으며, 스프링만 사용해도 MVC기반의 웹 어플리케이션을 어렵지 않게 개발하는 것이 가능하다

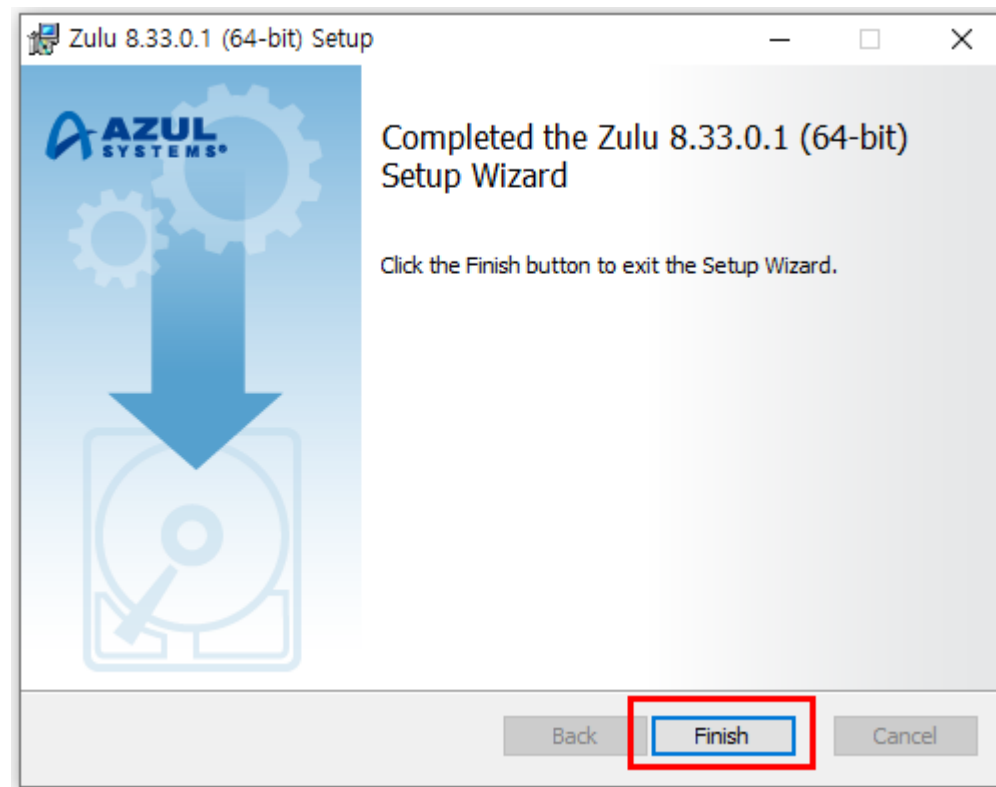
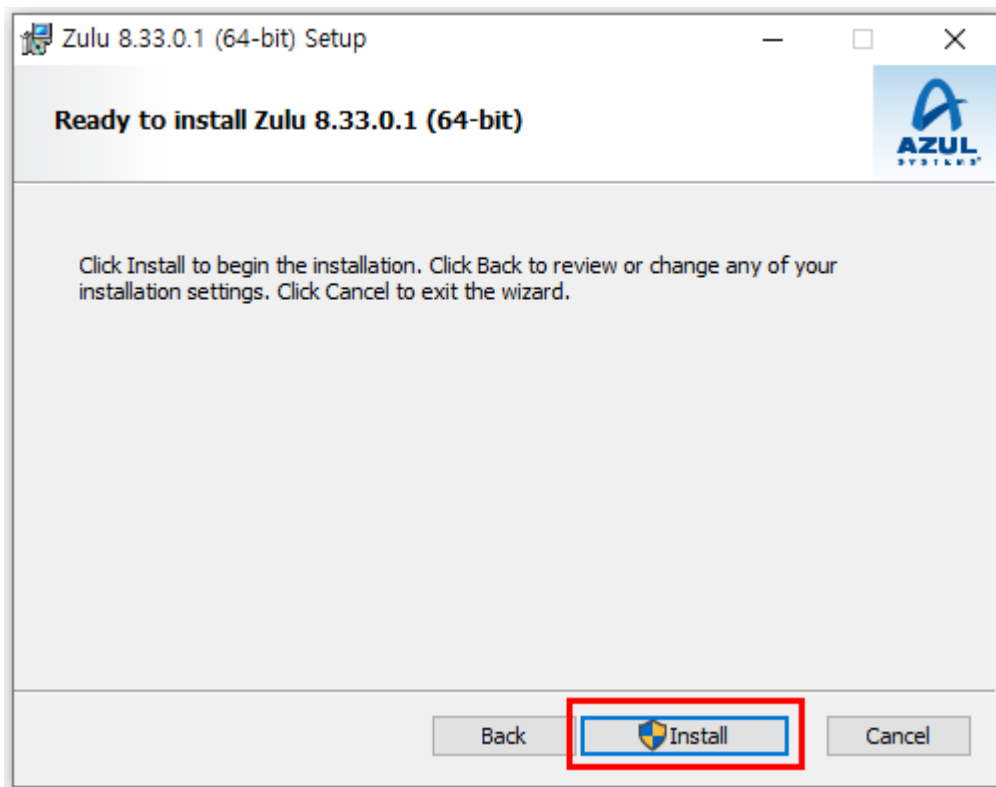
02 스프링 개발 환경

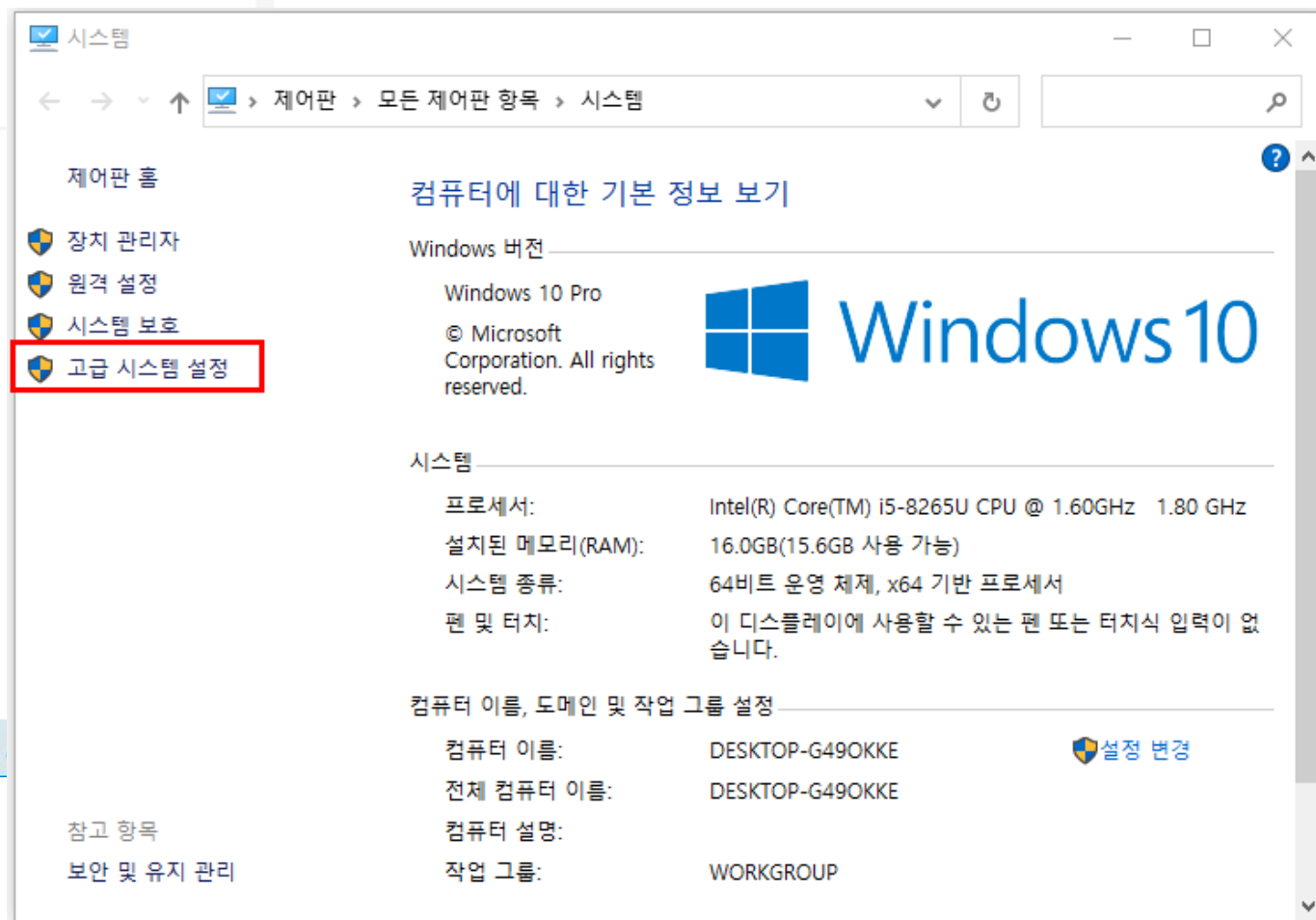
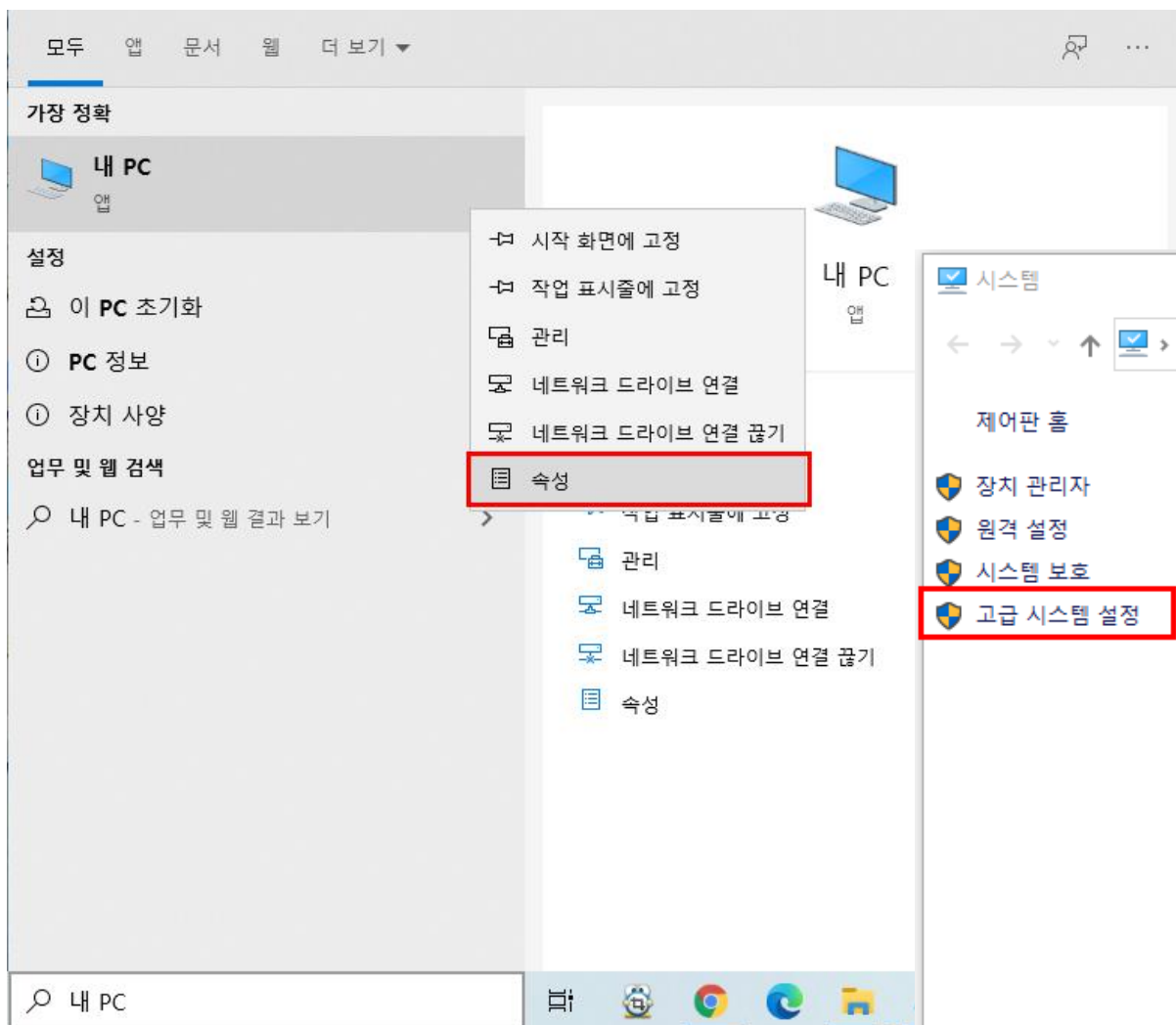


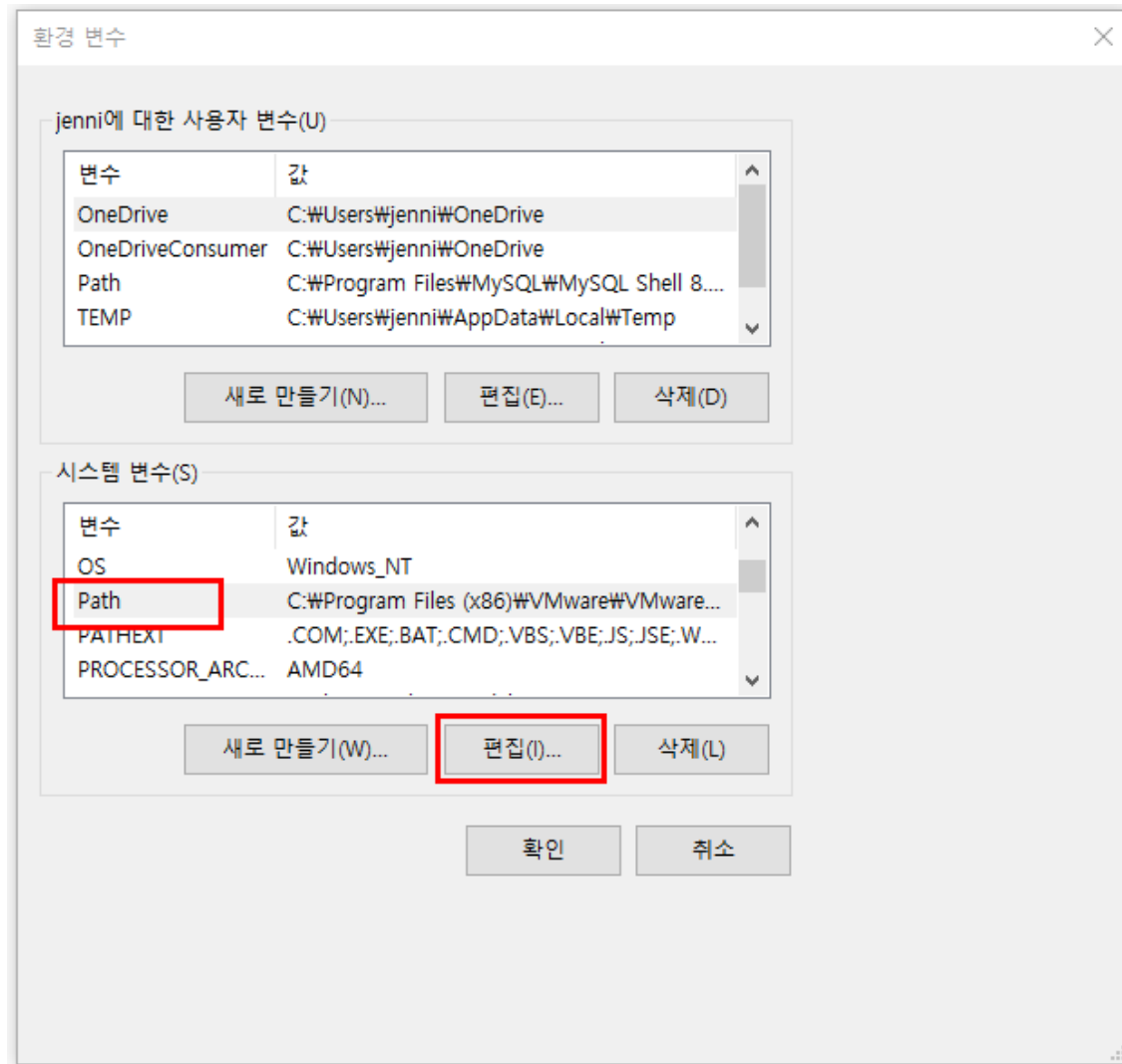
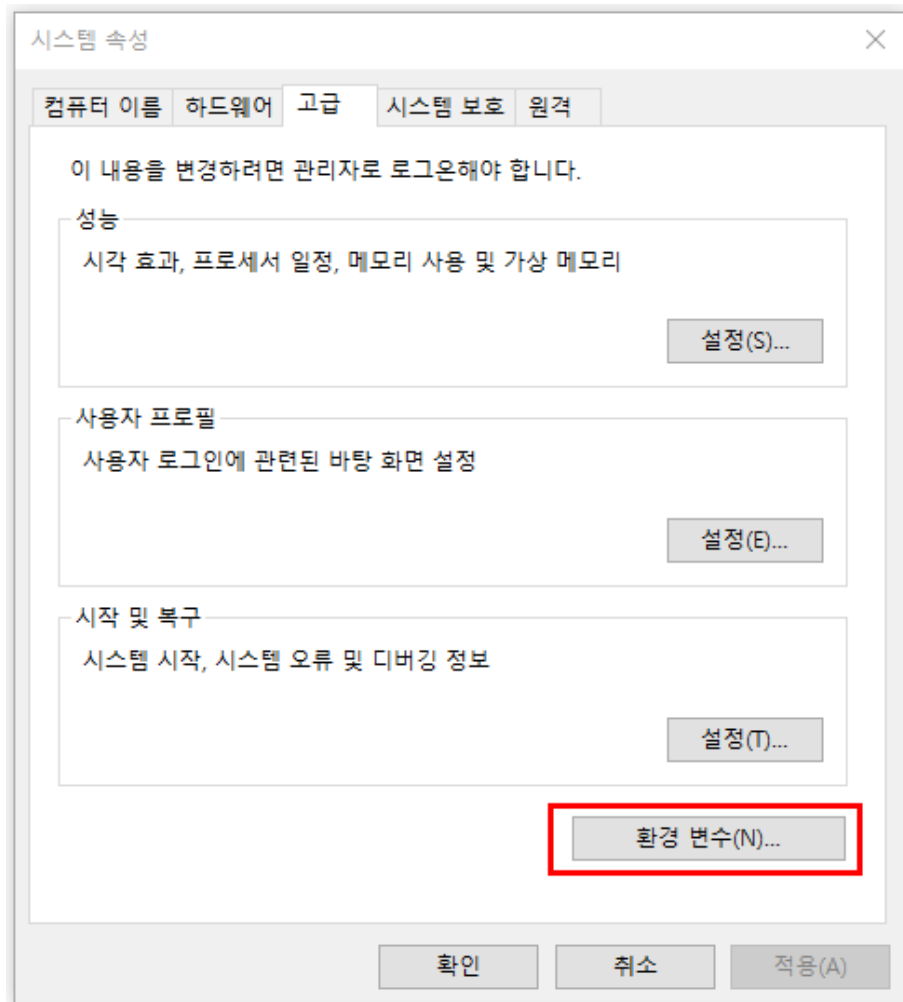
1. JDK 1.8설치하기

- 프로그램 개발을 위한 Java SE Development Kit을 설치한다 .
- 버전은 **zulu 8.33.0.1**으로 셋팅 할 것이며 아래 링크를 눌러 다운로드한다.
https://cdn.azul.com/zulu/bin/zulu8.33.0.1-jdk8.0.192-win_x64.msi
- 다운로드 된 파일을 더블 클릭해서 (**기본값으로 Next**) 설치한다.

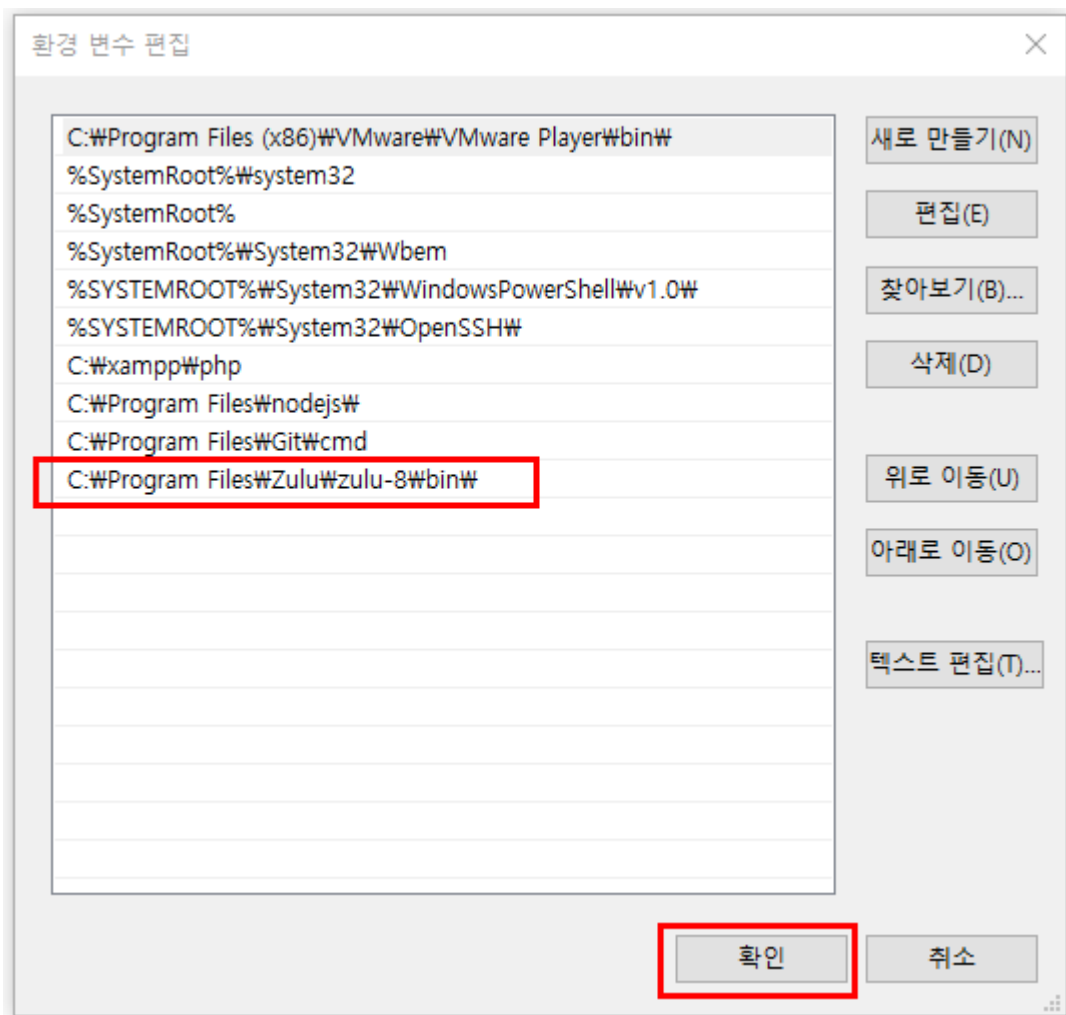




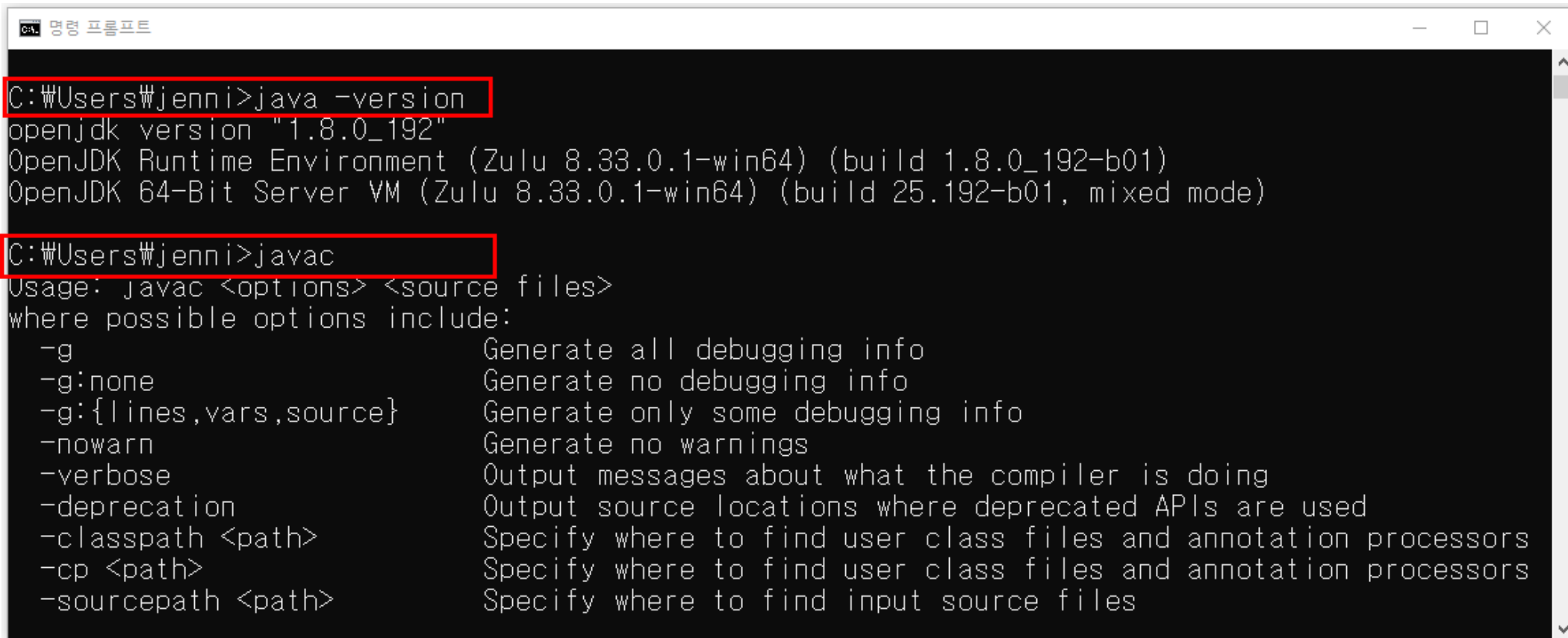




- C:\Program Files\Zulu\zulu-8\bin 을 확인한다.



➤  입력한 후 아래처럼 입력하여 버전을 확인 한다.



```
C:\Users\jenni>java -version
openjdk version "1.8.0_192"
OpenJDK Runtime Environment (Zulu 8.33.0.1-win64) (build 1.8.0_192-b01)
OpenJDK 64-Bit Server VM (Zulu 8.33.0.1-win64) (build 25.192-b01, mixed mode)

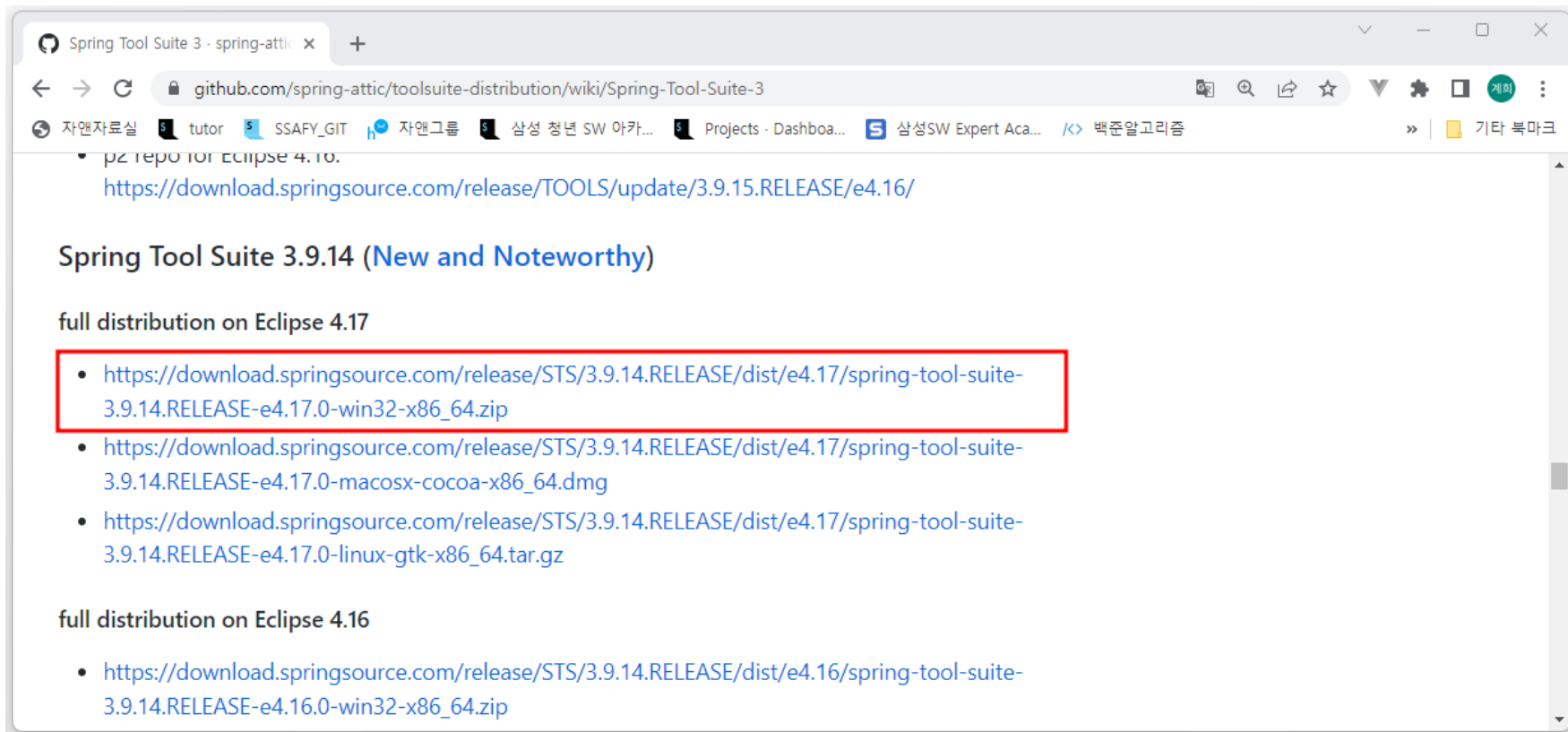
C:\Users\jenni>javac
Usage: javac <options> <source files>
where possible options include:
  -g                  Generate all debugging info
  -g:none             Generate no debugging info
  -g:{lines,vars,source}  Generate only some debugging info
  -nowarn             Generate no warnings
  -verbose            Output messages about what the compiler is doing
  -deprecation        Output source locations where deprecated APIs are used
  -classpath <path>   Specify where to find user class files and annotation processors
  -cp <path>          Specify where to find user class files and annotation processors
  -sourcepath <path>  Specify where to find input source files
```

2. STS 설치하기

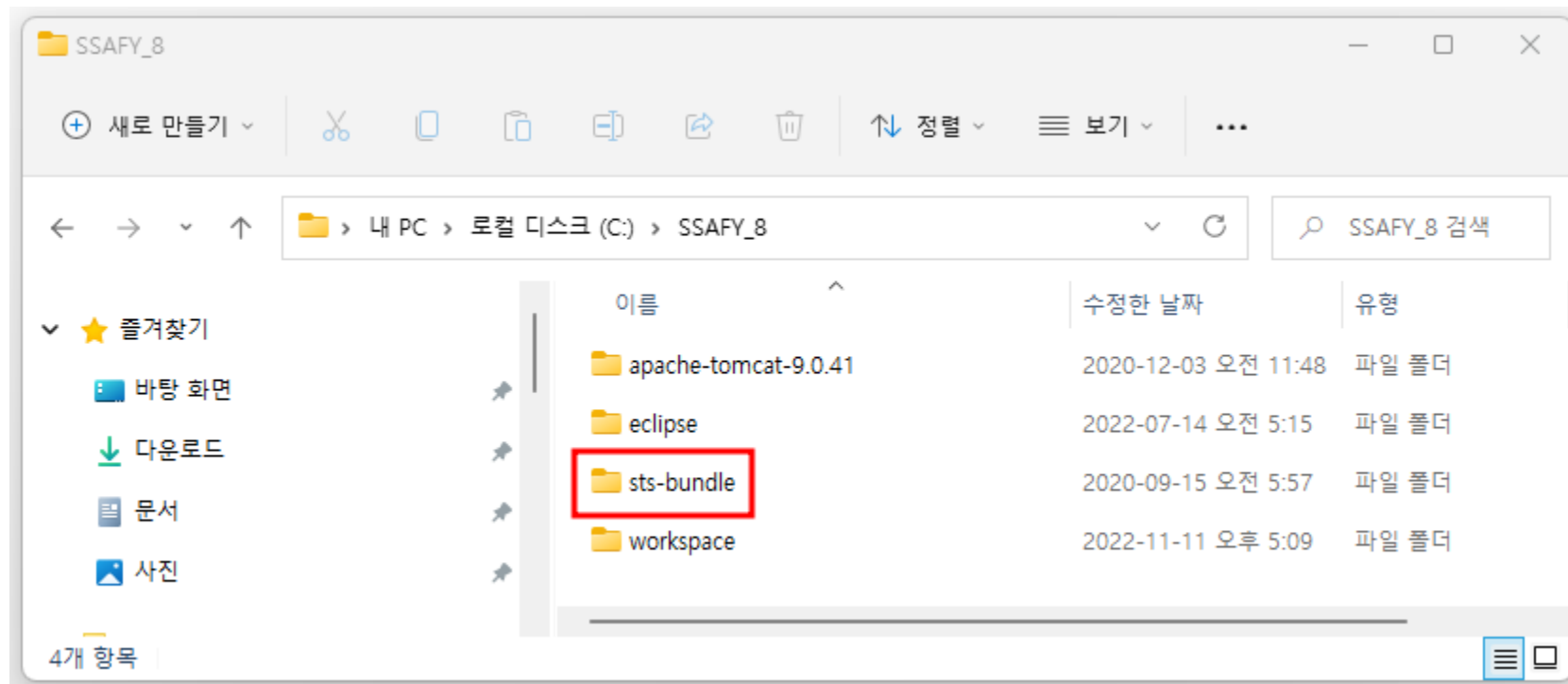
- 스프링 코드를 작성하기 위한 개발 툴이다.
- 버전은 **3.9.14**로 셋팅 할 것이며 아래 링크를 눌러 다운로드한다.

<https://github.com/spring-projects/toolsuite-distribution/wiki/Spring-Tool-Suite-3>

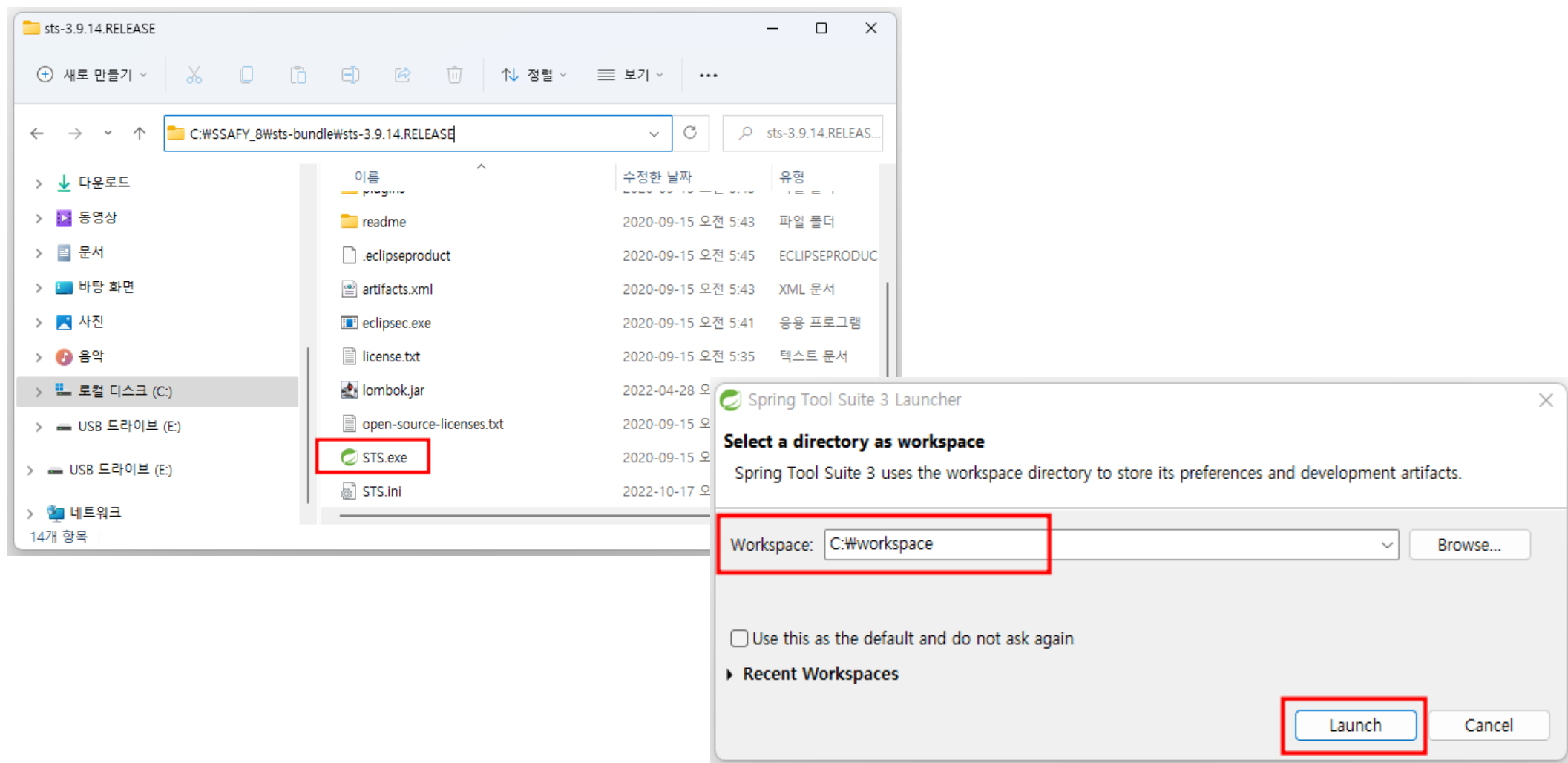
Version: 3.9.14.RELEASE(4.15.0)=> spring-tool-suite-3.9.14.RELEASE-e4.17.0-win32-x86_64.zip

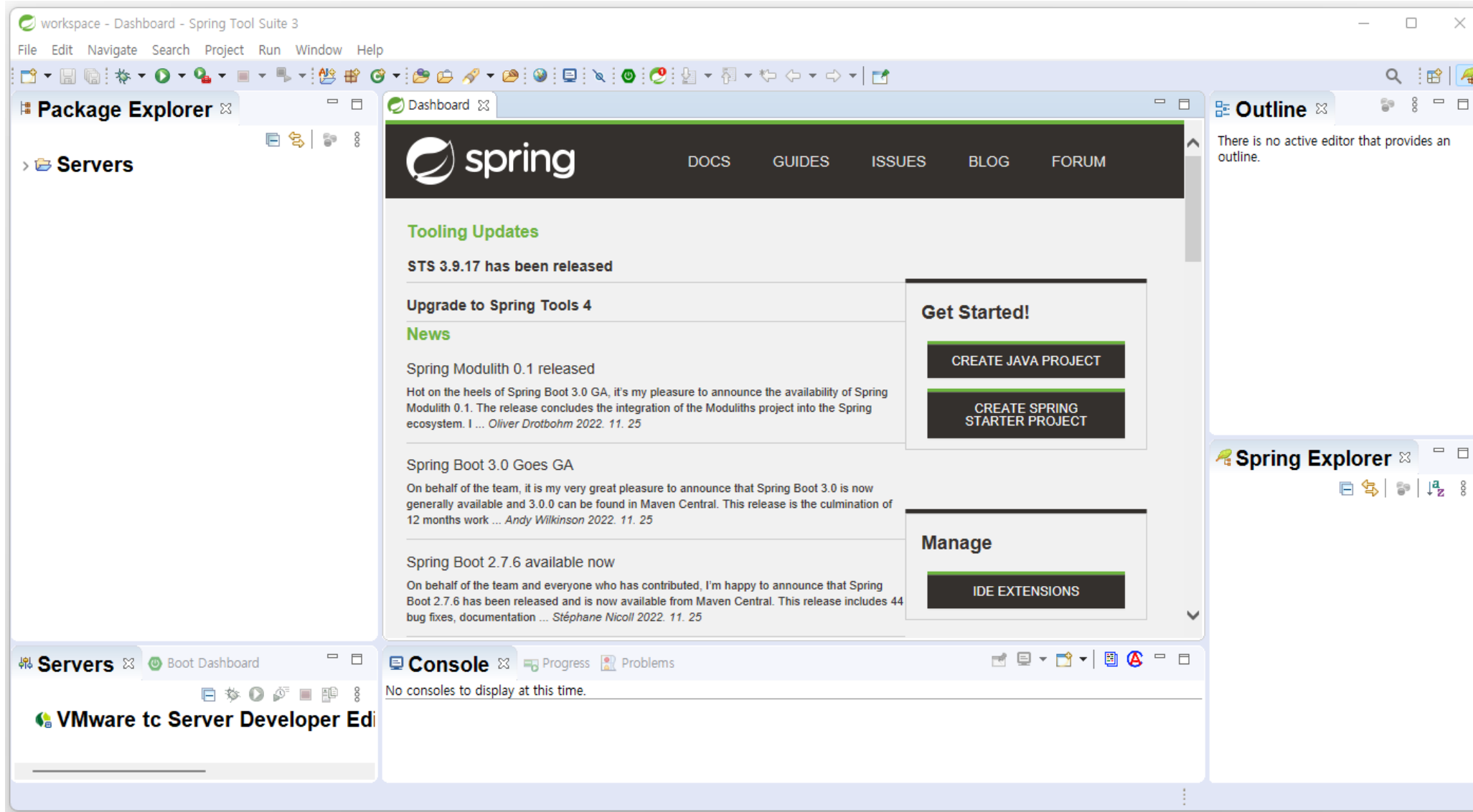


➤ 원하는 위치에 압축을 풀면 sts-bundle 폴더가 생긴다.

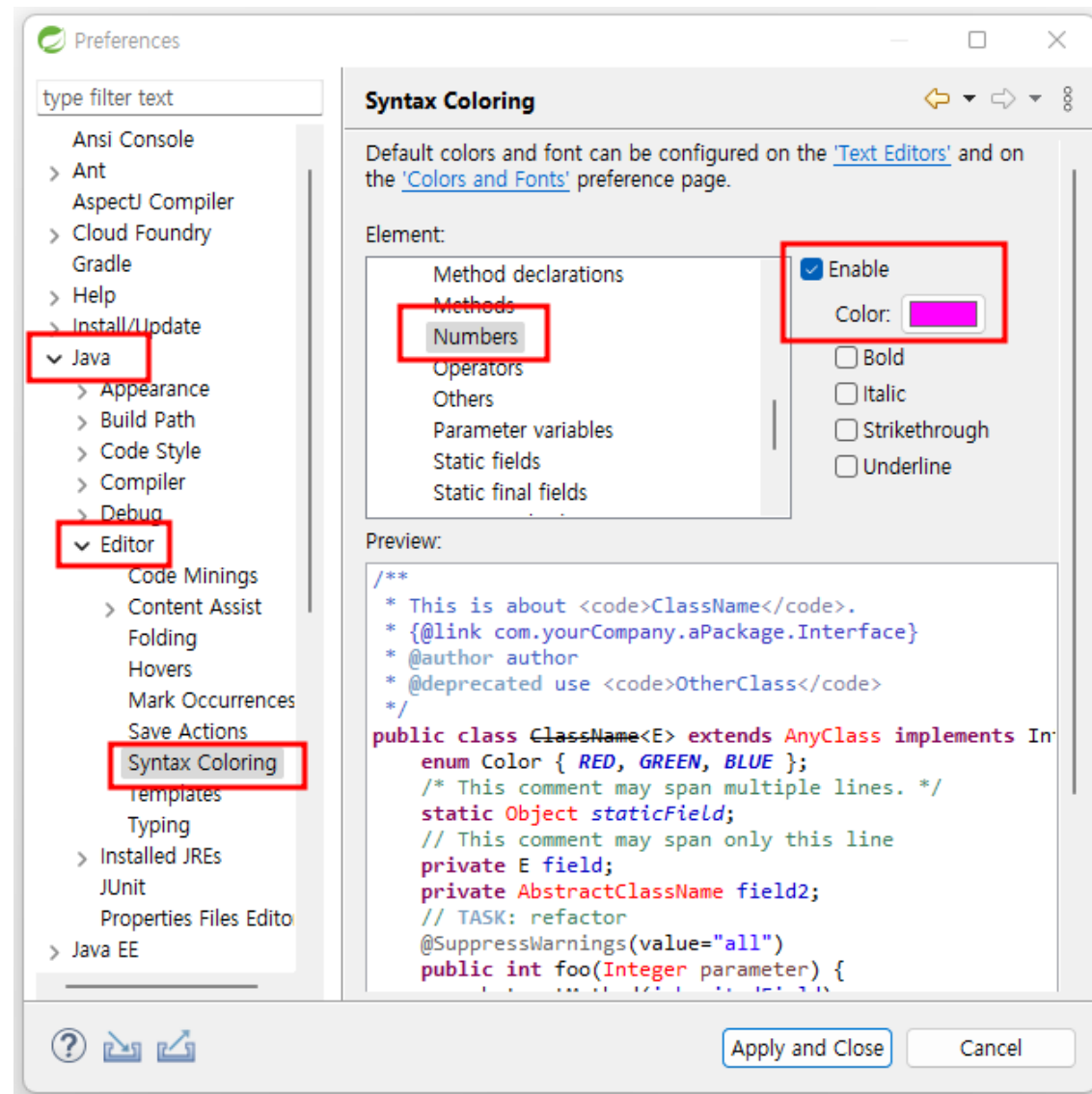
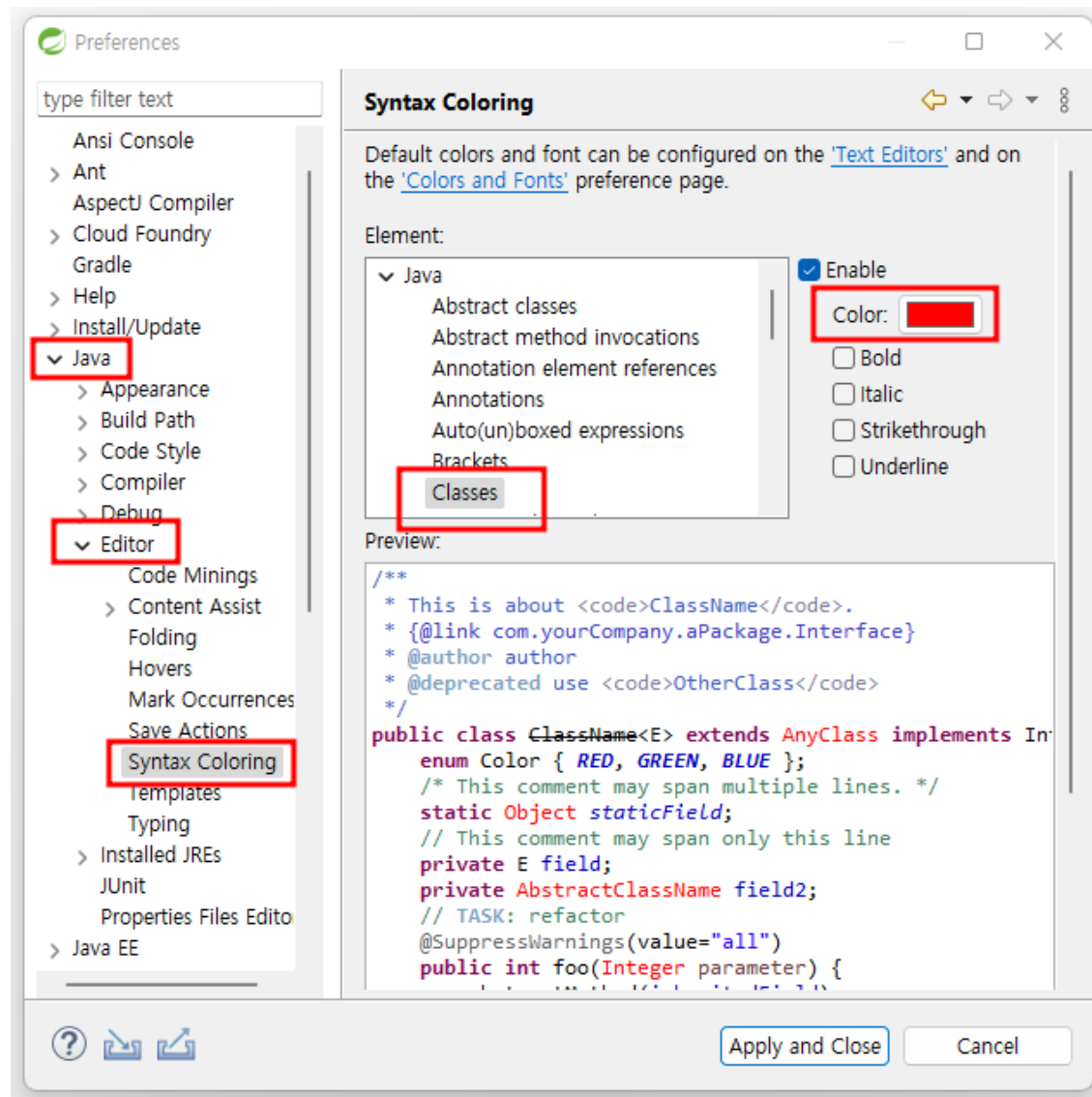


➤ sts-bundle 폴더 안에 **sts.exe** 가 실행파일이다. 바탕화면에 단축 아이콘을 만들어 사용한다.





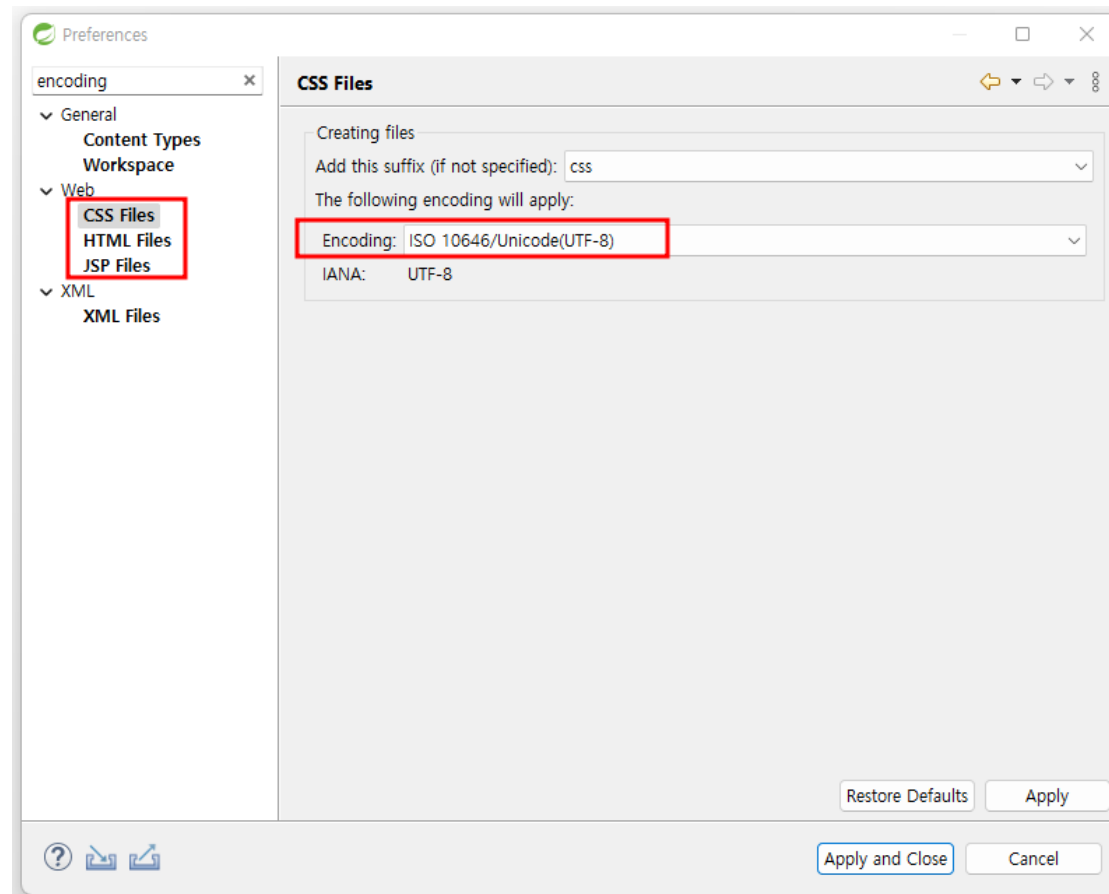
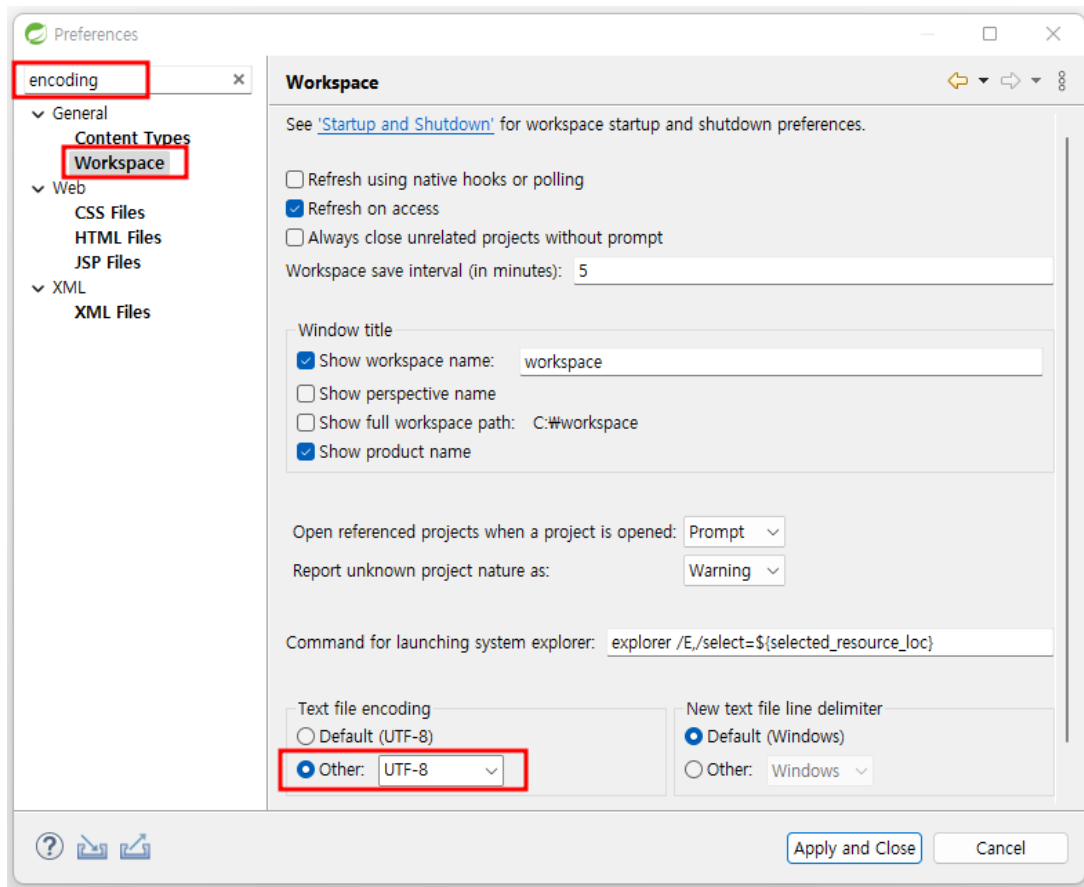
➤ sts 환경설정 [자바 Class와 Number 색상변경] : Window ➔ Preferences ➔ General



➤ sts 환경설정 [UTF-8 설정하기]

: **Window ➔ Preferences ➔ General ➔** 검색창에 **encoding**을 입력한다.

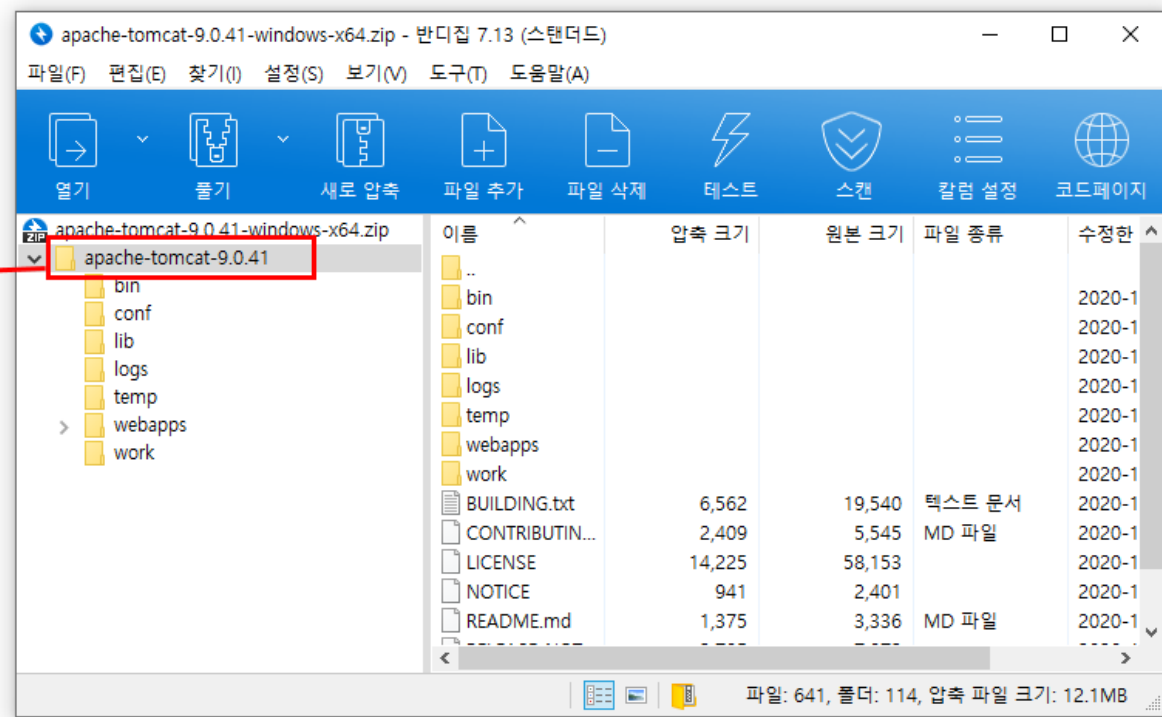
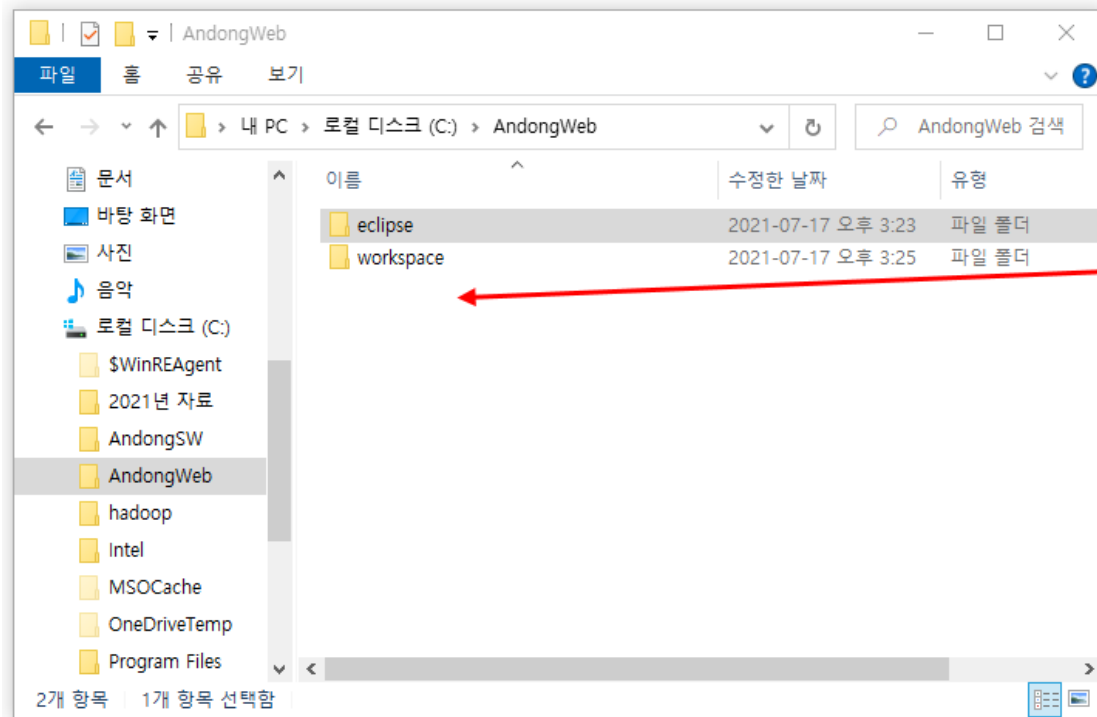
(Workspace, CSS, HTML, JSP 를 모두 ISO 10646/Unicode(UTF-8)로 바꾼다.)



3. Tomcat 셋팅 하기

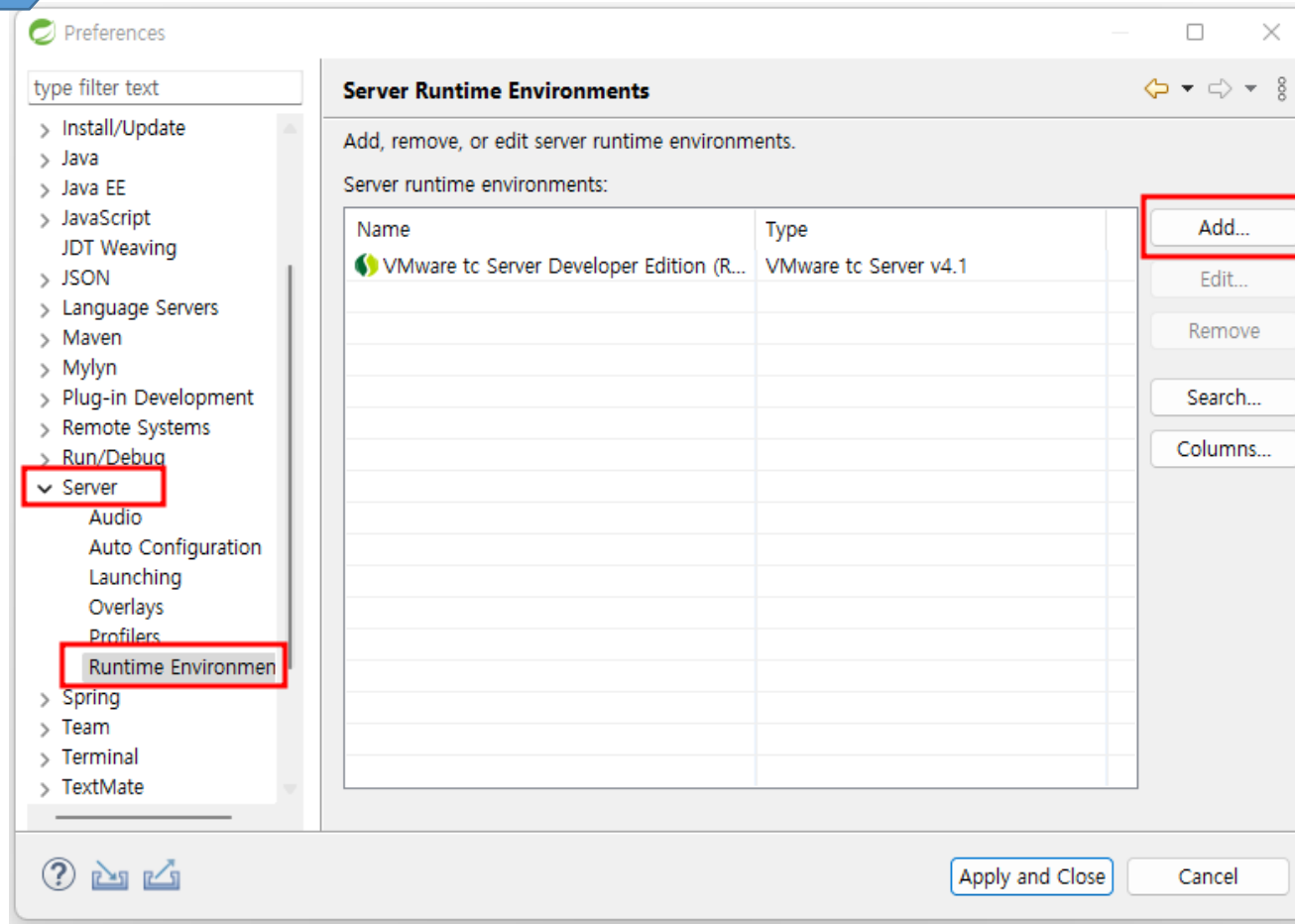
- 웹 서버이며 Apache Tomcat을 사용한다.
- 버전은 **9.0.41**으로 셋팅 할 것이며 아래 링크를 눌러 다운로드한다.

<https://archive.apache.org/dist/tomcat/tomcat-9/v9.0.41/bin/apache-tomcat-9.0.41-windows-x64.zip>

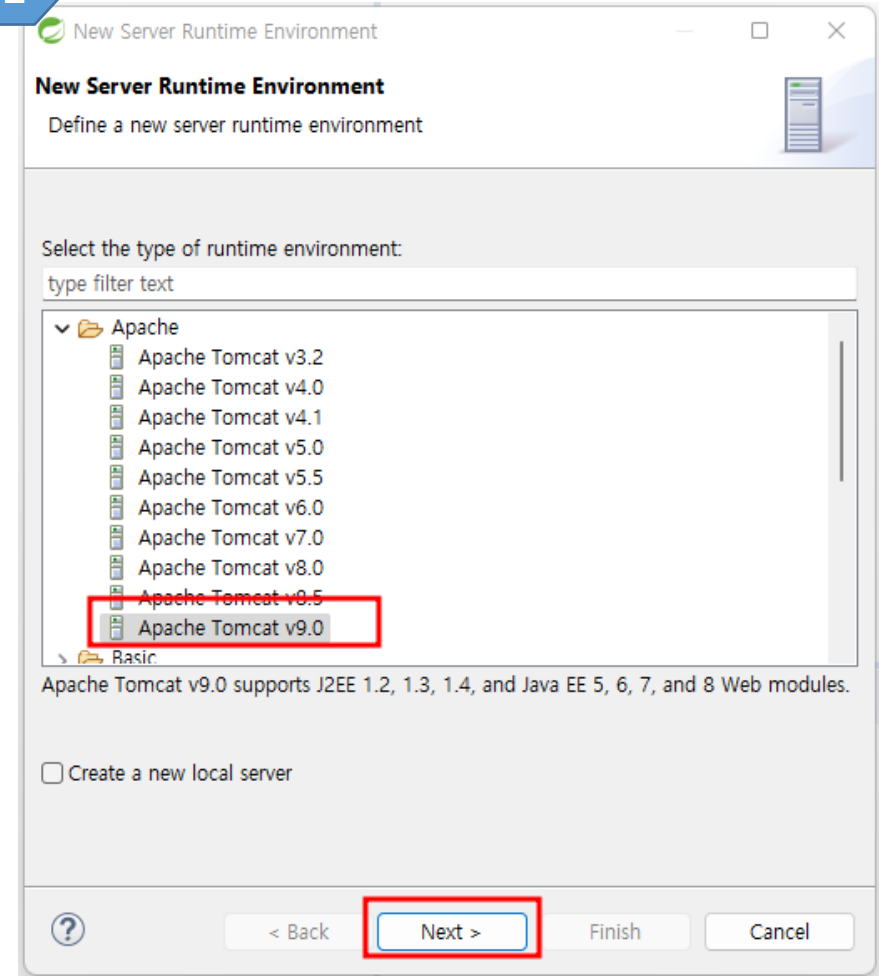


➤ Window ➔ Preference ➔ Server ➔ Runtime Environment

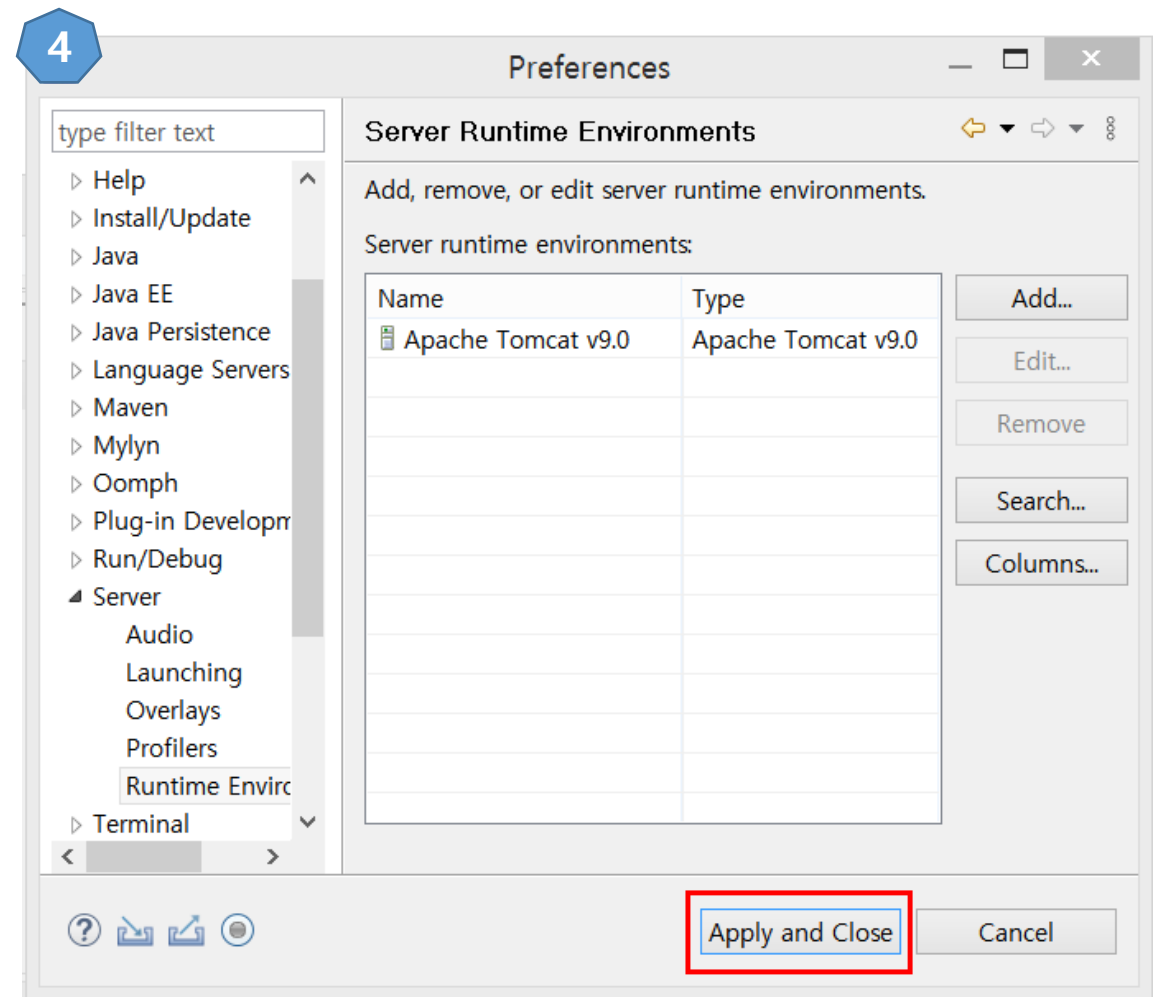
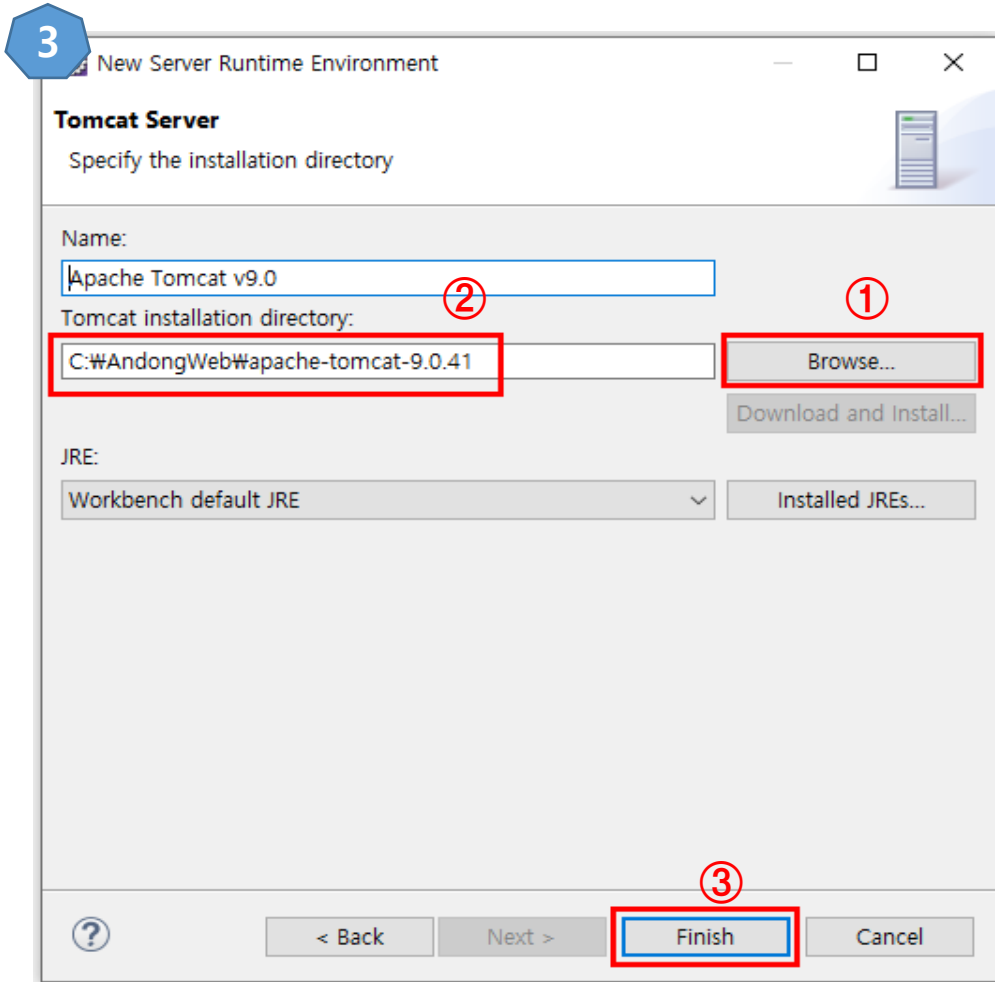
1



2



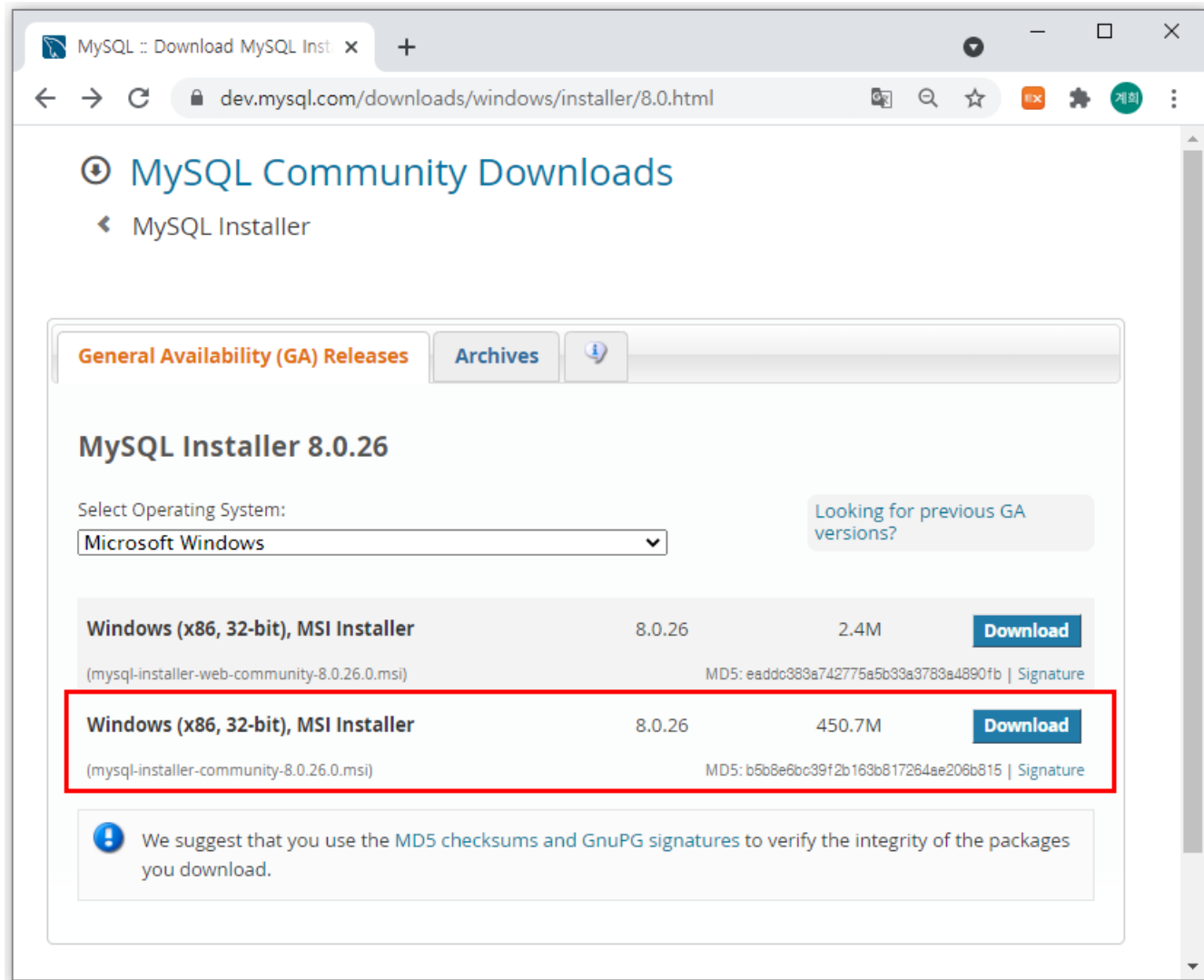
➤ Browse ➔ C:\Wapache-tomcat-9.0.41 (선택)



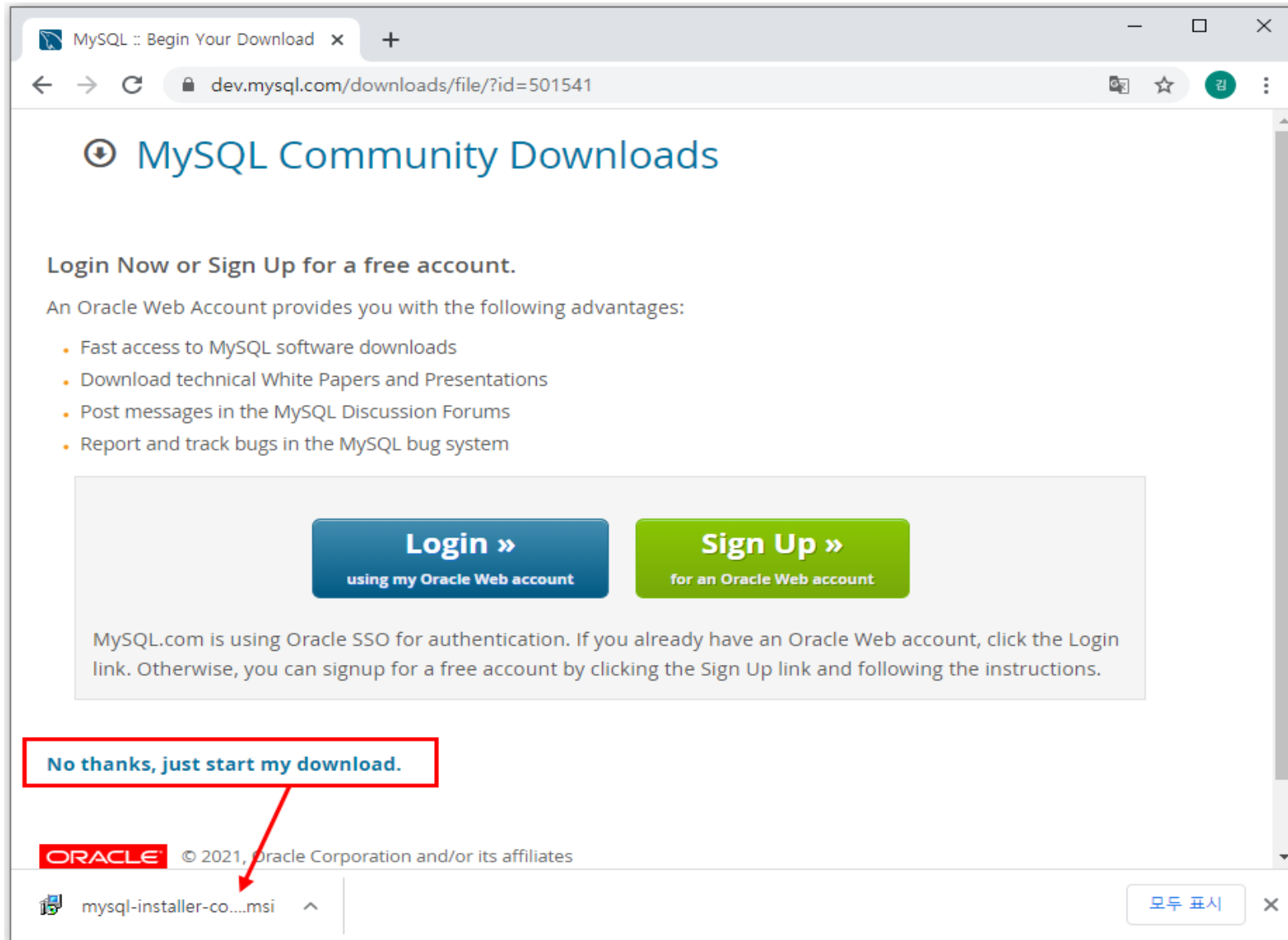
4. MySQL 설치하기

- 데이터베이스를 이용하기 위해 Workbench를 설치한다.
- 버전은 **MySQL installer 8.0.26**으로 셋팅 할 것이며 아래 링크를 눌러 다운로드한다.

<https://dev.mysql.com/downloads/windows/installer/8.0.html>



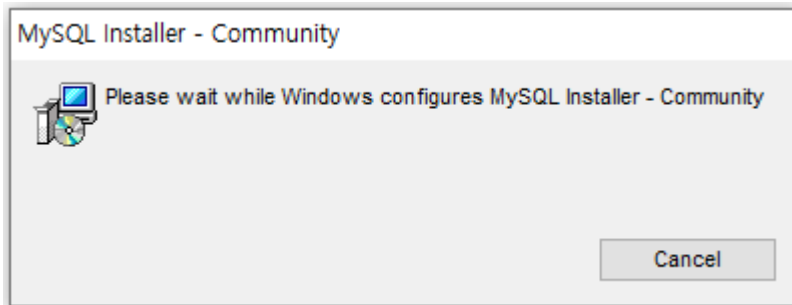
➤ No, thanks, just start my download를 선택 → mysql-installer-community-8.0.26.0.msi



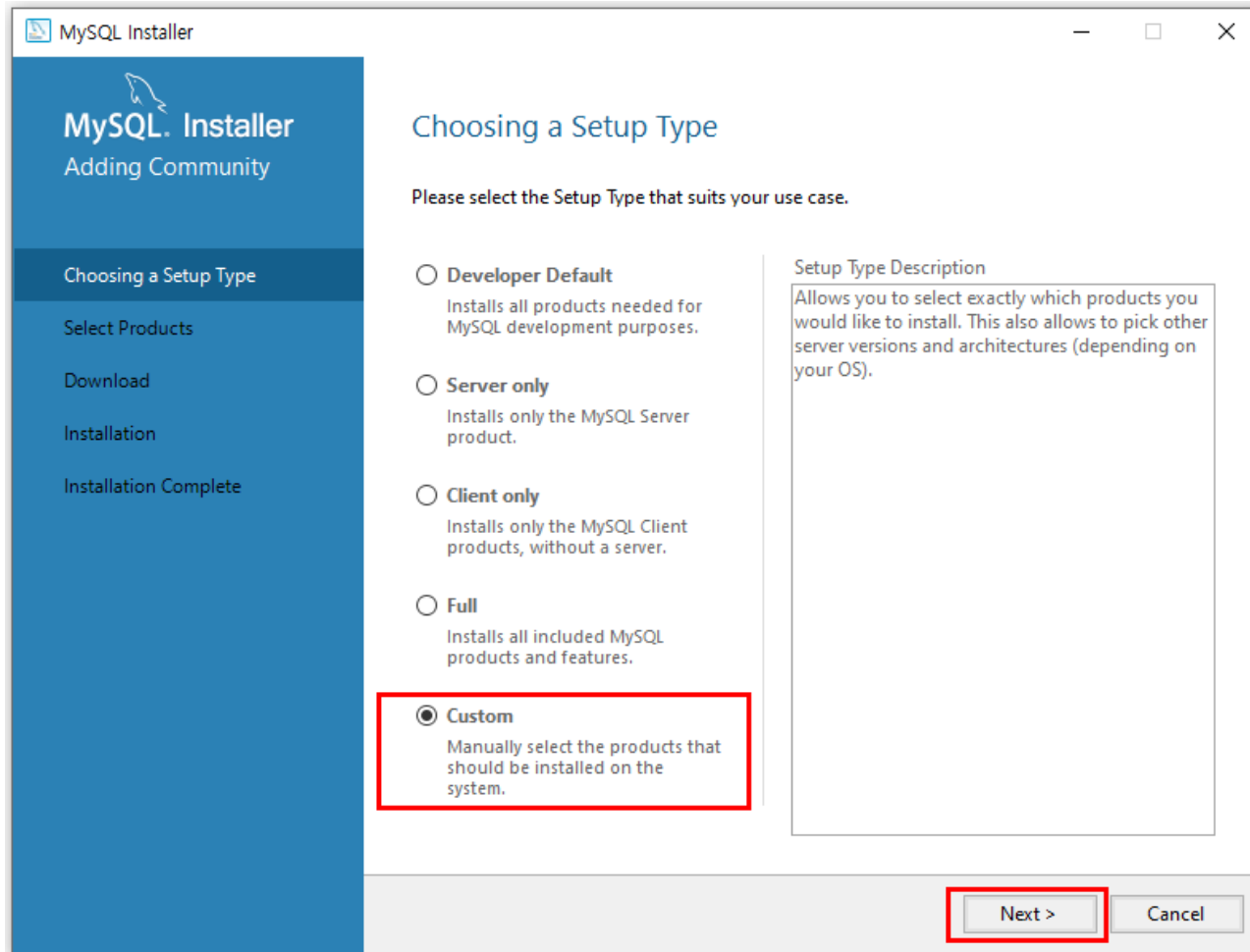
➤ 다운받은 MySQL 파일을 실행한다.

 mysql-installer-community-8.0.26.0.msi	2021-08-16 오후 2:06	Windows Installer...	461,472KB
--	--------------------	----------------------	-----------

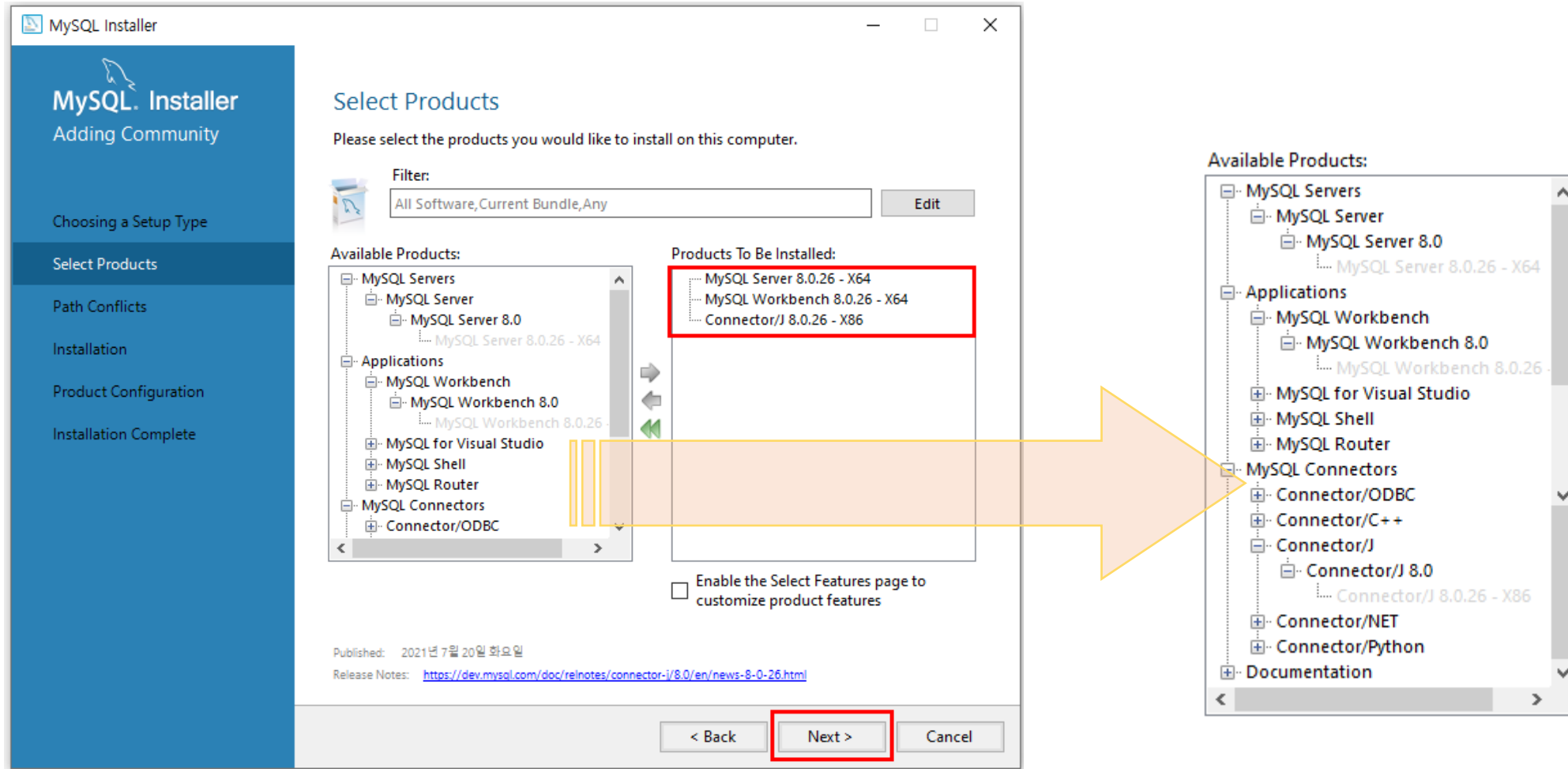
➤ Install 하기 위한 준비 화면이 나온다.



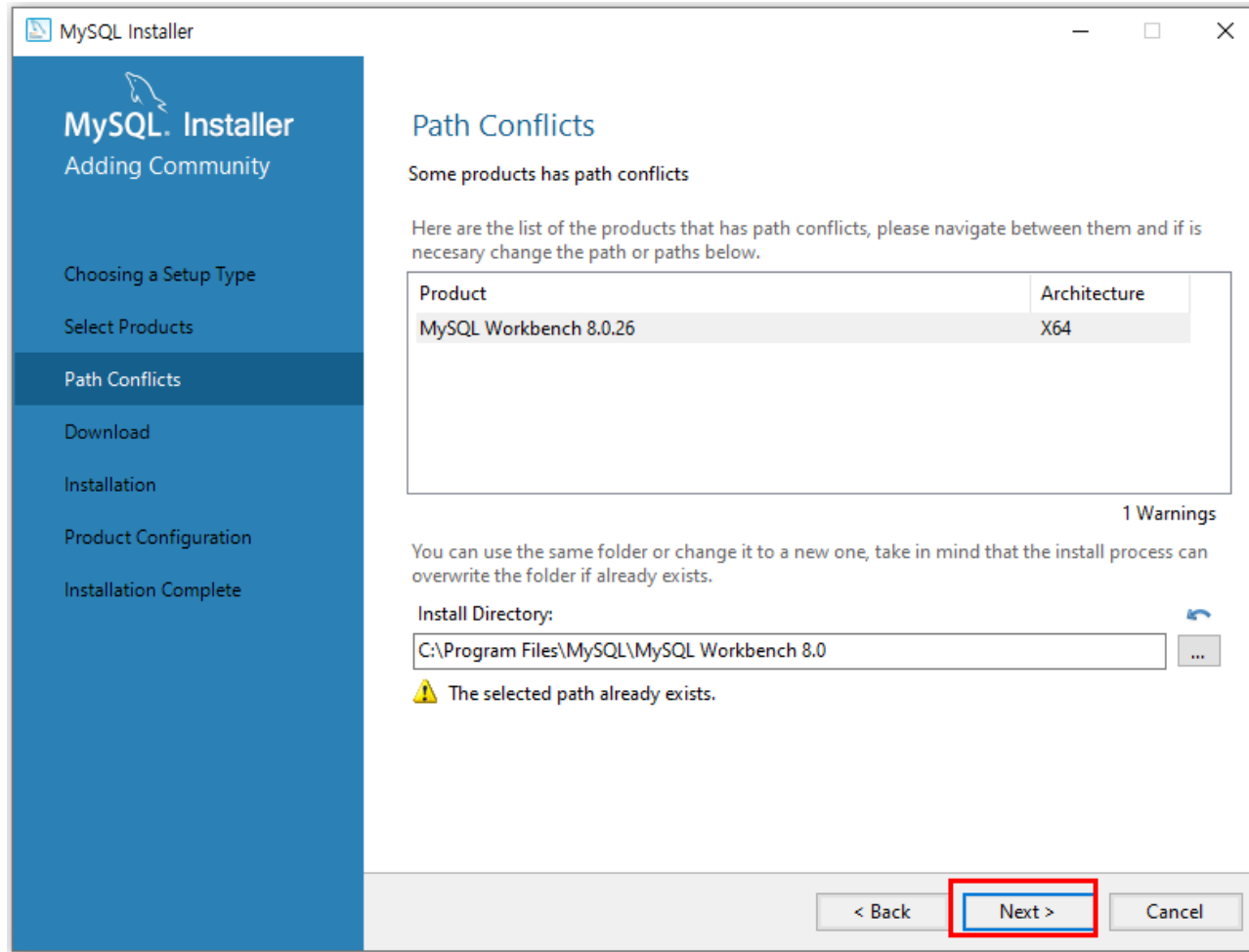
➤ Custom을 선택하고 Next를 누른다.



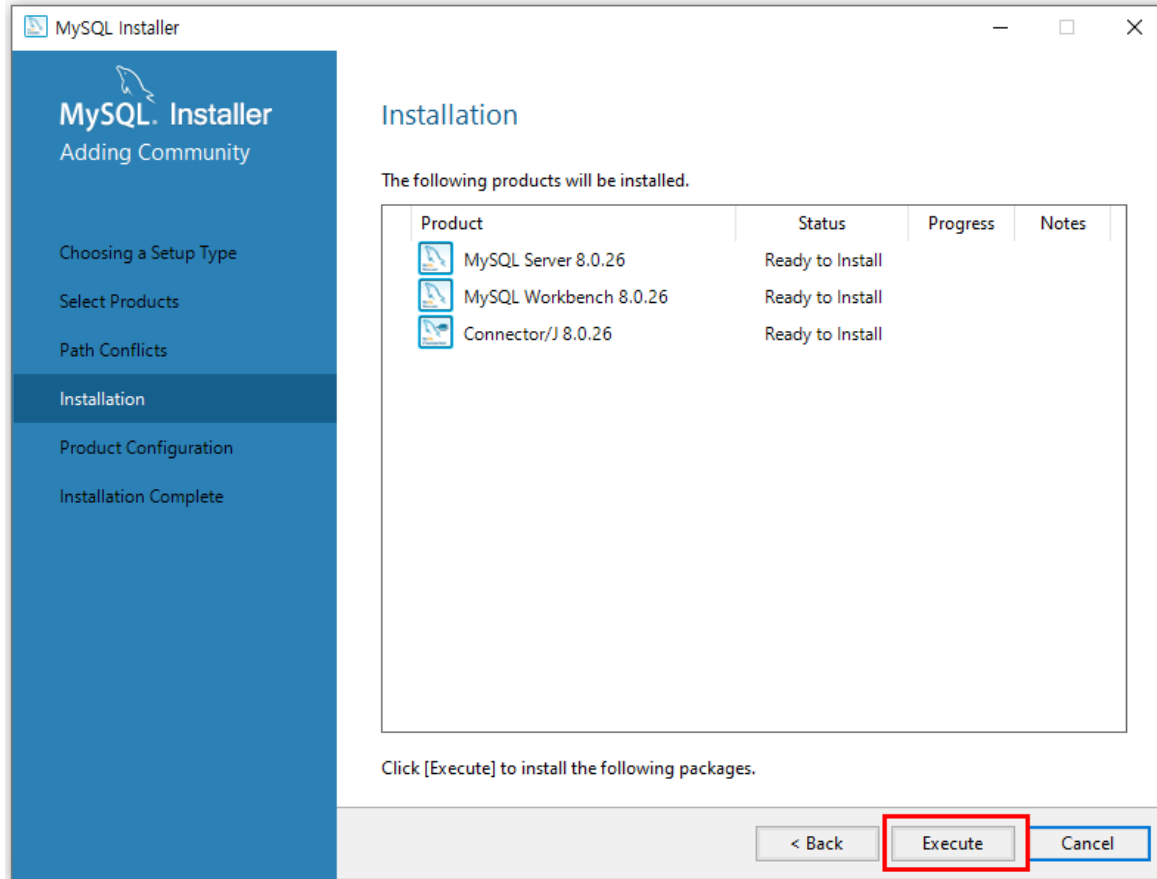
➤ 빨간 박스의 설치 파일들을 선택하고 Next를 클릭한다.



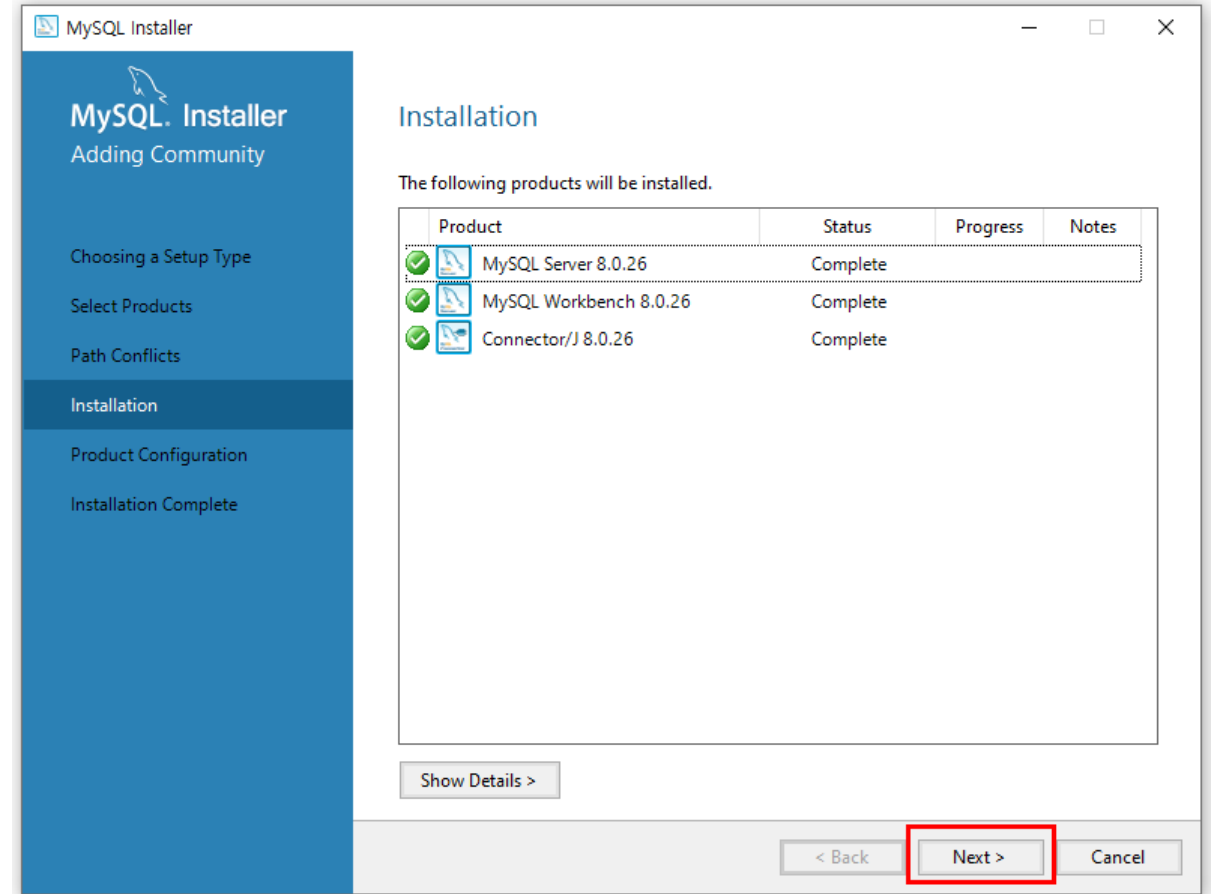
➤ 설치할 경로를 선택하고 Next를 클릭한다.



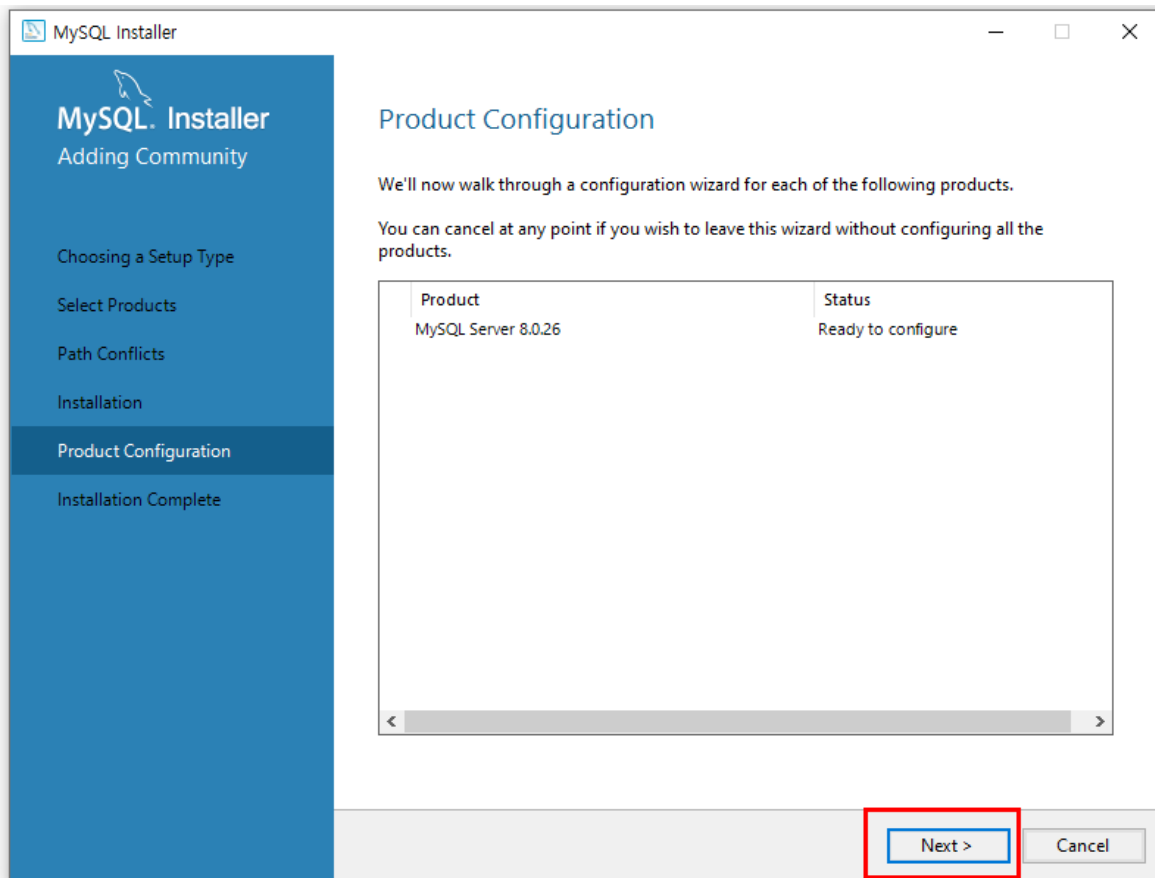
➤ MySQL과 Workbench, Connector을 설치(Execute)한다.



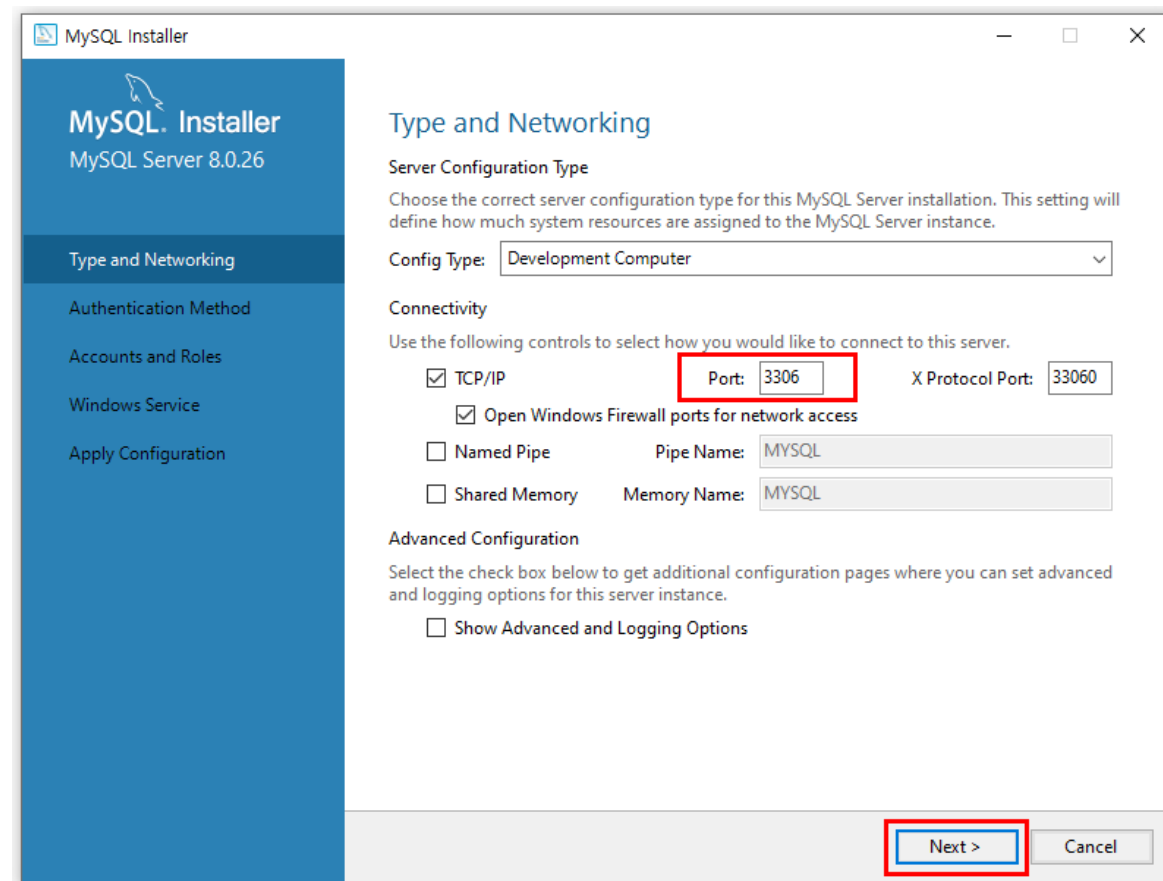
➤ 설치가 끝나고 나면 Next를 클릭한다.



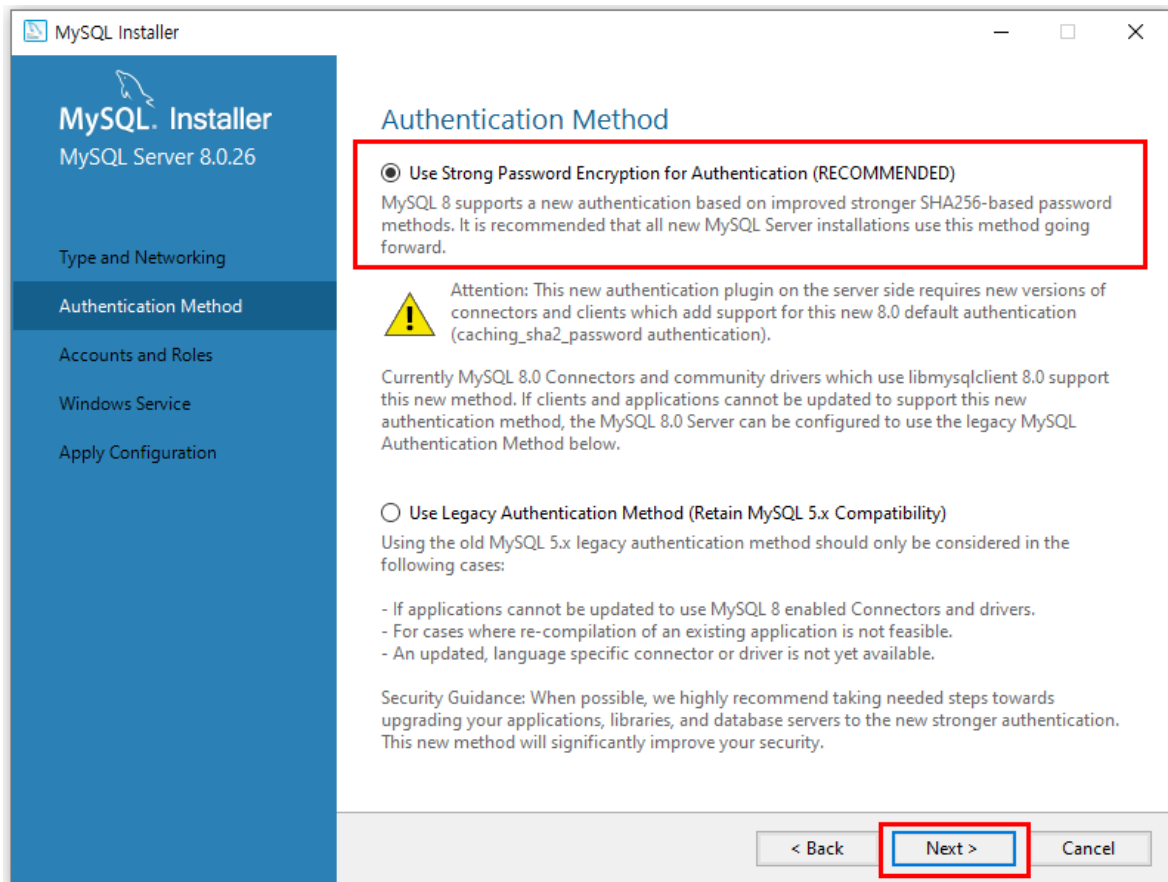
➤ Next를 클릭한다.



➤ 포트번호 3306을 확인하고 Next를 클릭한다.



➤ 추천하는 암호방식의 비밀번호를 선택하고 Next를 클릭한다.



➤ 2개의 계정을 생성 한다.

- 관리자 계정: root / 1234
- 사용자 계정: edu / 1234

The screenshot shows the MySQL Installer 'Accounts and Roles' step. The interface includes a left sidebar with navigation options: 'Type and Networking', 'Authentication Method', 'Accounts and Roles' (selected), 'Windows Service', and 'Apply Configuration'. The main area is titled 'Accounts and Roles' and contains a 'Root Account Password' section with two password input fields (both masked with dots) and a 'Password strength: Weak' indicator. Below this is a 'MySQL User Accounts' table with columns 'MySQL User Name', 'Host', and 'User Role'. To the right of the table are buttons for 'Add User', 'Edit User', and 'Delete'. At the bottom are '< Back', 'Next >', and 'Cancel' buttons. A red box highlights the 'Add User' button, with a red arrow pointing to a 'MySQL User Account' dialog box. The dialog box has a title bar 'MySQL User Account' and a close button. It contains the text 'Please specify the user name, password, and database role.' and fields for 'User Name' (containing 'edu'), 'Host' (set to '<All Hosts (%)>'), and 'Role' (set to 'DB Admin'). Below these is an 'Authentication' section with a radio button selected for 'MySQL'. At the bottom of the dialog is a 'MySQL user credentials' section with 'Password' and 'Confirm Password' fields (both masked with dots) and a 'Password strength: Weak' indicator. The 'OK' button is highlighted with a red box. Red annotations include: a blue hexagon with the number '1' pointing to the password fields; a blue hexagon with the number '2' pointing to the 'Add User' button; a blue hexagon with the number '3' pointing to the 'MySQL User Account' dialog; a blue hexagon with the number '4' pointing to the 'Next >' button; and red text labels '비밀번호 (1234)' (Password (1234)) pointing to the password fields, '계정생성 (edu)' (Account creation (edu)) pointing to the 'User Name' field, and '비밀번호 (1234)' (Password (1234)) pointing to the password fields in the dialog.

1

MySQL Installer
MySQL Server 8.0.26

Type and Networking
Authentication Method
Accounts and Roles
Windows Service
Apply Configuration

Accounts and Roles

Root Account Password
Enter the password for the root account. Please remember to store this password in a secure place.

MySQL Root Password:

Repeat Password:

Password strength: Weak

비밀번호 (1234)

MySQL User Accounts
Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

MySQL User Name	Host	User Role
-----------------	------	-----------

2

Add User
Edit User
Delete

3

MySQL User Account

Please specify the user name, password, and database role.

User Name:

Host: <All Hosts (%)>

Role: DB Admin

Authentication: ☒ MySQL

MySQL user credentials

Password:

Confirm Password:

Password strength: Weak

비밀번호 (1234)

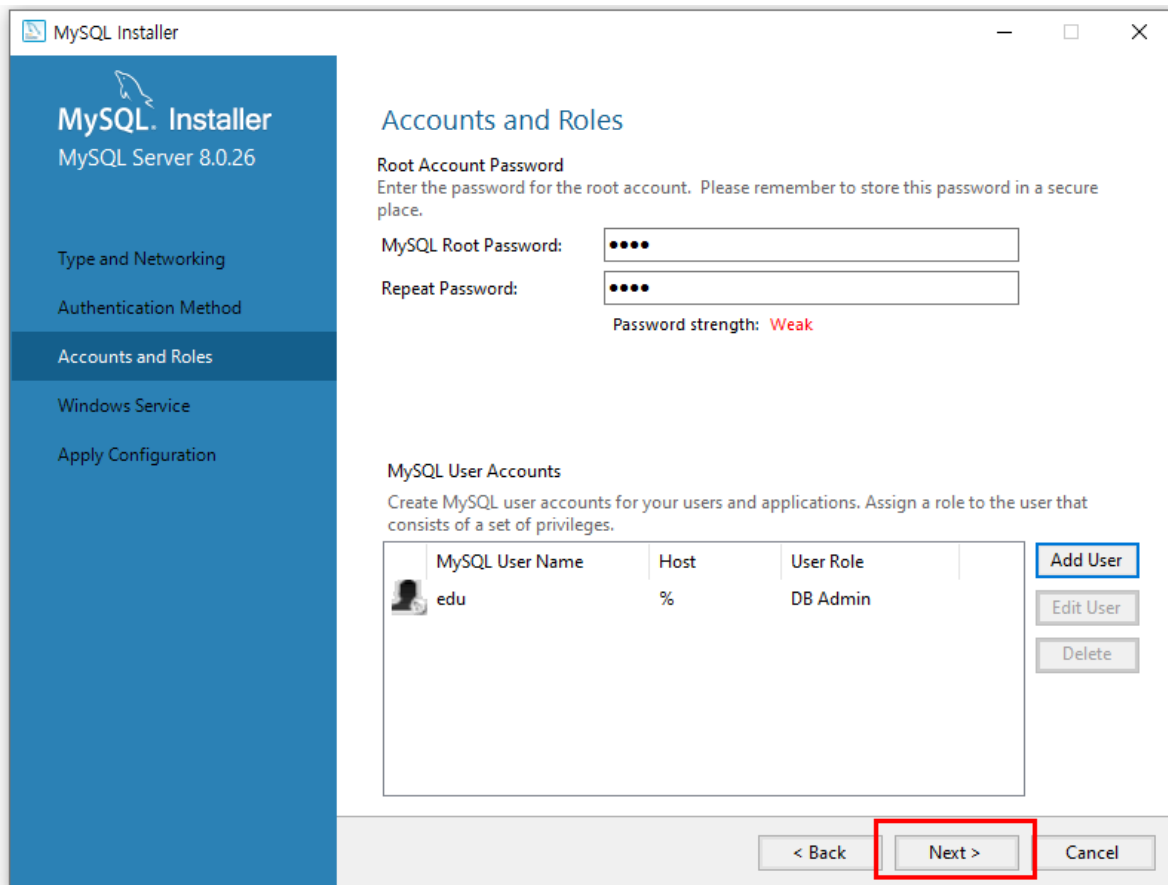
계정생성 (edu)

4

< Back Next > Cancel

OK Cancel

➤ Next를 클릭한다.



The screenshot shows the 'Accounts and Roles' step of the MySQL Installer. The left sidebar has 'Accounts and Roles' selected. The main area is titled 'Accounts and Roles' and contains a 'Root Account Password' section with two password input fields and a 'Password strength: Weak' indicator. Below this is a 'MySQL User Accounts' section with a table of existing users and buttons to 'Add User', 'Edit User', and 'Delete'. The 'Next >' button at the bottom is highlighted with a red box.

MySQL Installer

MySQL Server 8.0.26

Type and Networking

Authentication Method

Accounts and Roles

Windows Service

Apply Configuration

Accounts and Roles

Root Account Password

Enter the password for the root account. Please remember to store this password in a secure place.


MySQL Root Password:

Repeat Password:

Password strength: **Weak**

MySQL User Accounts

Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

MySQL User Name	Host	User Role
 edu	%	DB Admin

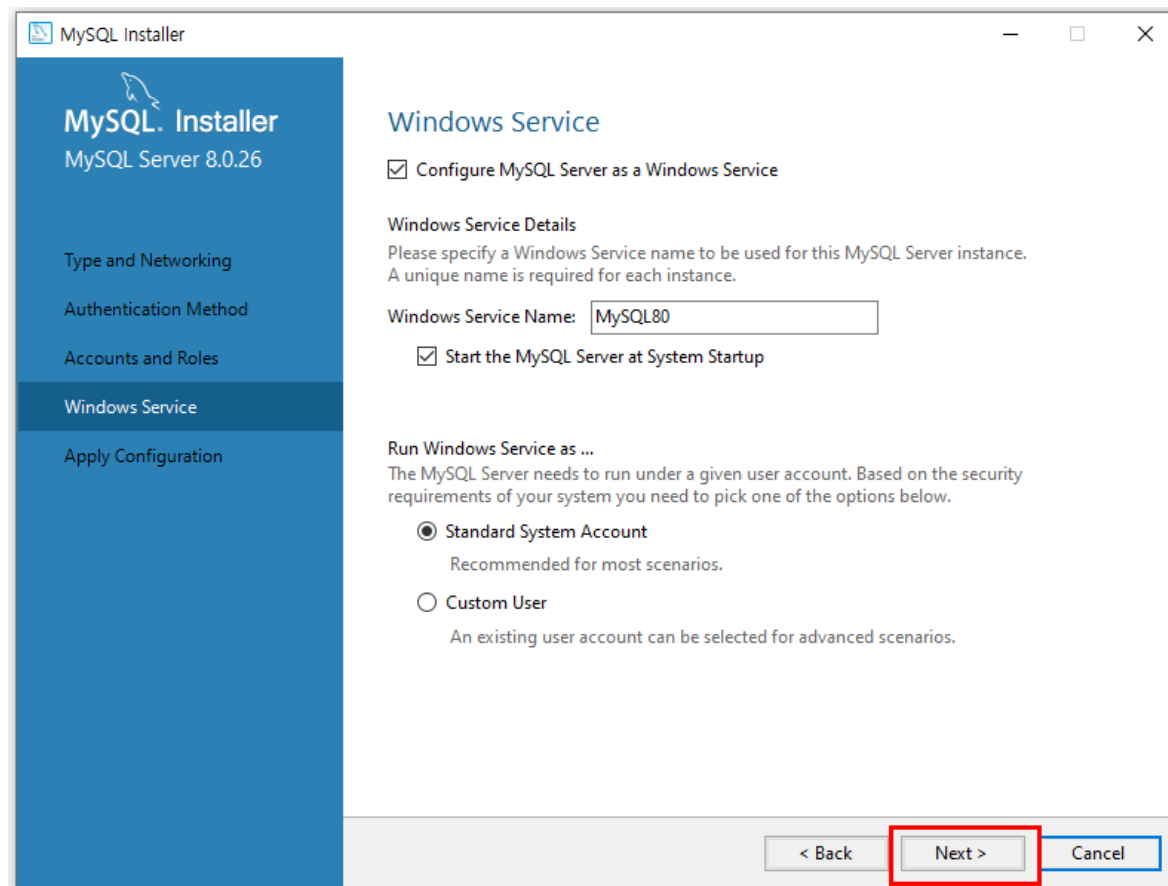
[Add User](#)

[Edit User](#)

[Delete](#)

< Back **Next >** Cancel

➤ Next를 클릭한다.



The screenshot shows the 'Windows Service' step of the MySQL Installer. The left sidebar has 'Windows Service' selected. The main area is titled 'Windows Service' and contains a checkbox to 'Configure MySQL Server as a Windows Service'. Below this is a 'Windows Service Details' section with a text input for 'Windows Service Name' (MySQL80) and a checkbox to 'Start the MySQL Server at System Startup'. The 'Next >' button at the bottom is highlighted with a red box.

MySQL Installer

MySQL Server 8.0.26

Type and Networking

Authentication Method

Accounts and Roles

Windows Service

Apply Configuration

Windows Service

☒ Configure MySQL Server as a Windows Service

Windows Service Details

Please specify a Windows Service name to be used for this MySQL Server instance. A unique name is required for each instance.

Windows Service Name:

☒ Start the MySQL Server at System Startup

Run Windows Service as ...

The MySQL Server needs to run under a given user account. Based on the security requirements of your system you need to pick one of the options below.

☒ Standard System Account

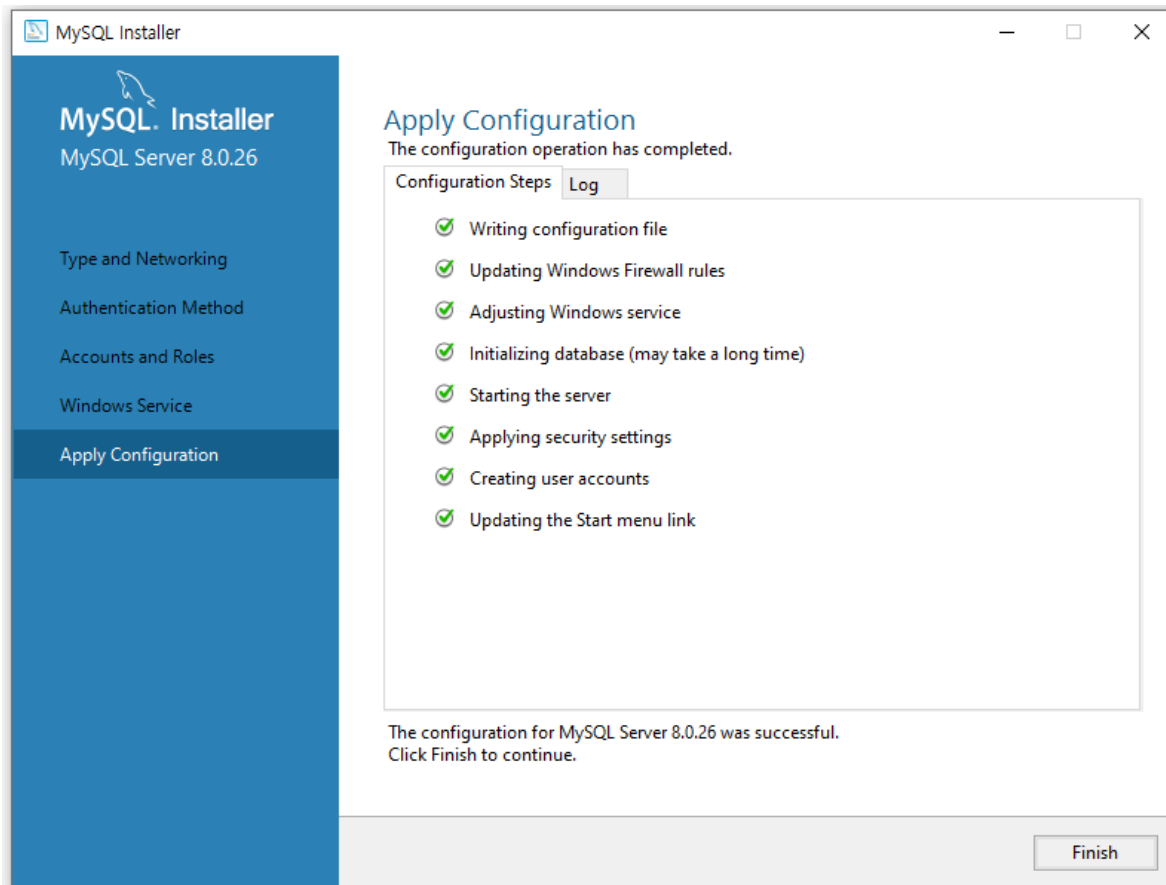
Recommended for most scenarios.

☐ Custom User

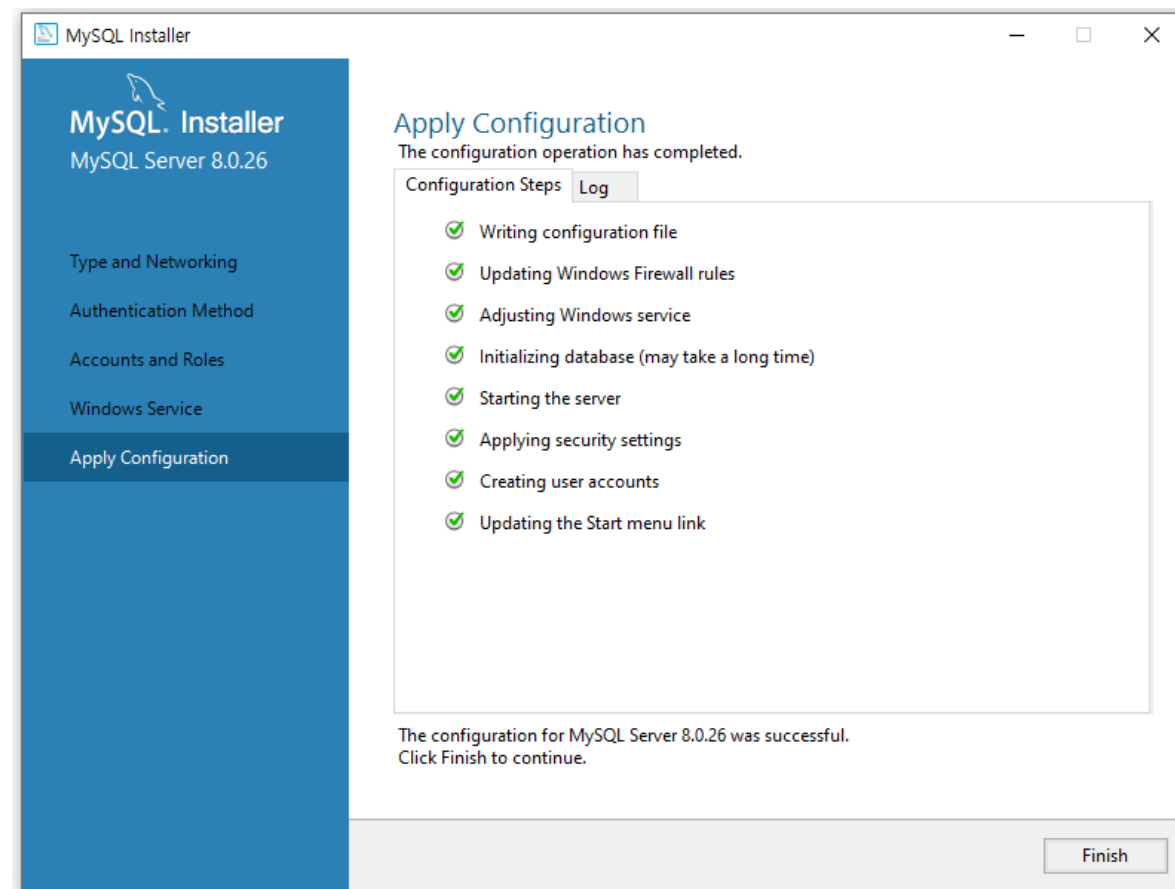
An existing user account can be selected for advanced scenarios.

< Back **Next >** Cancel

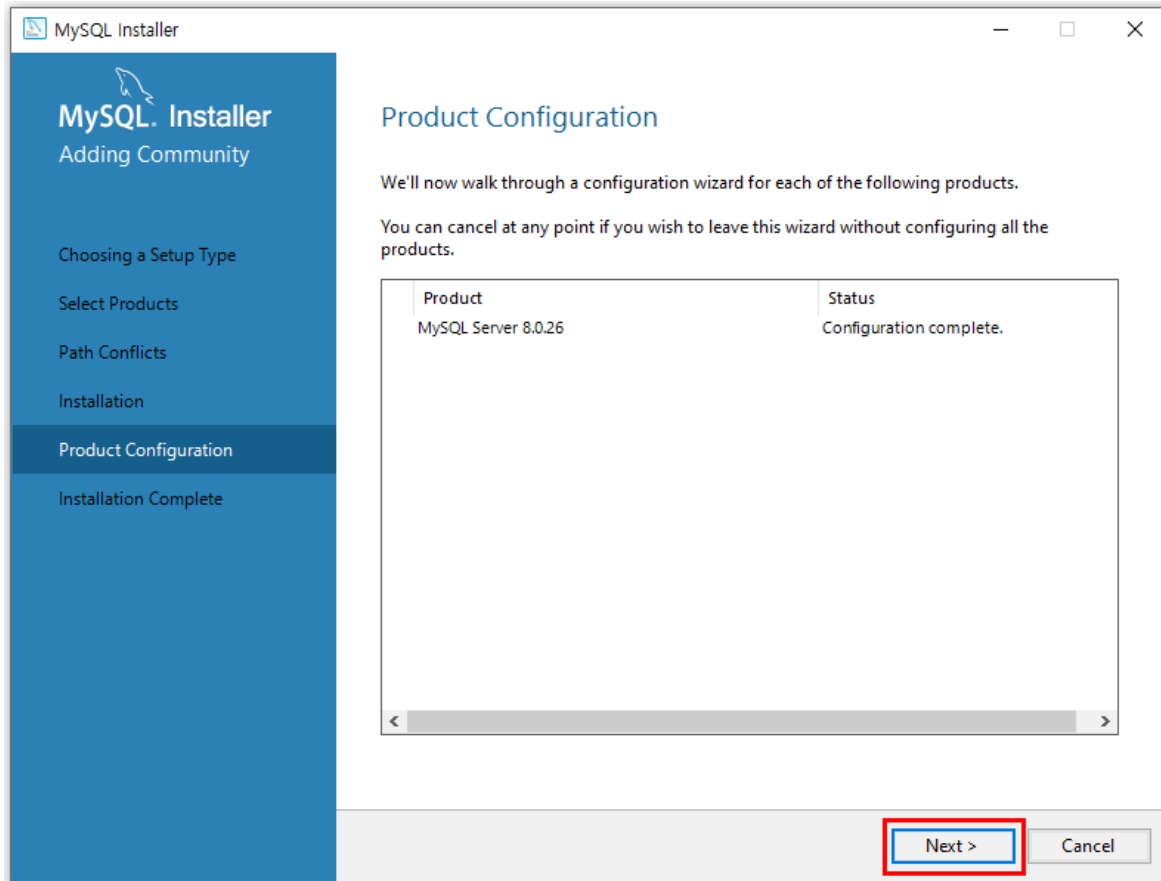
➤ Execute를 클릭한다.



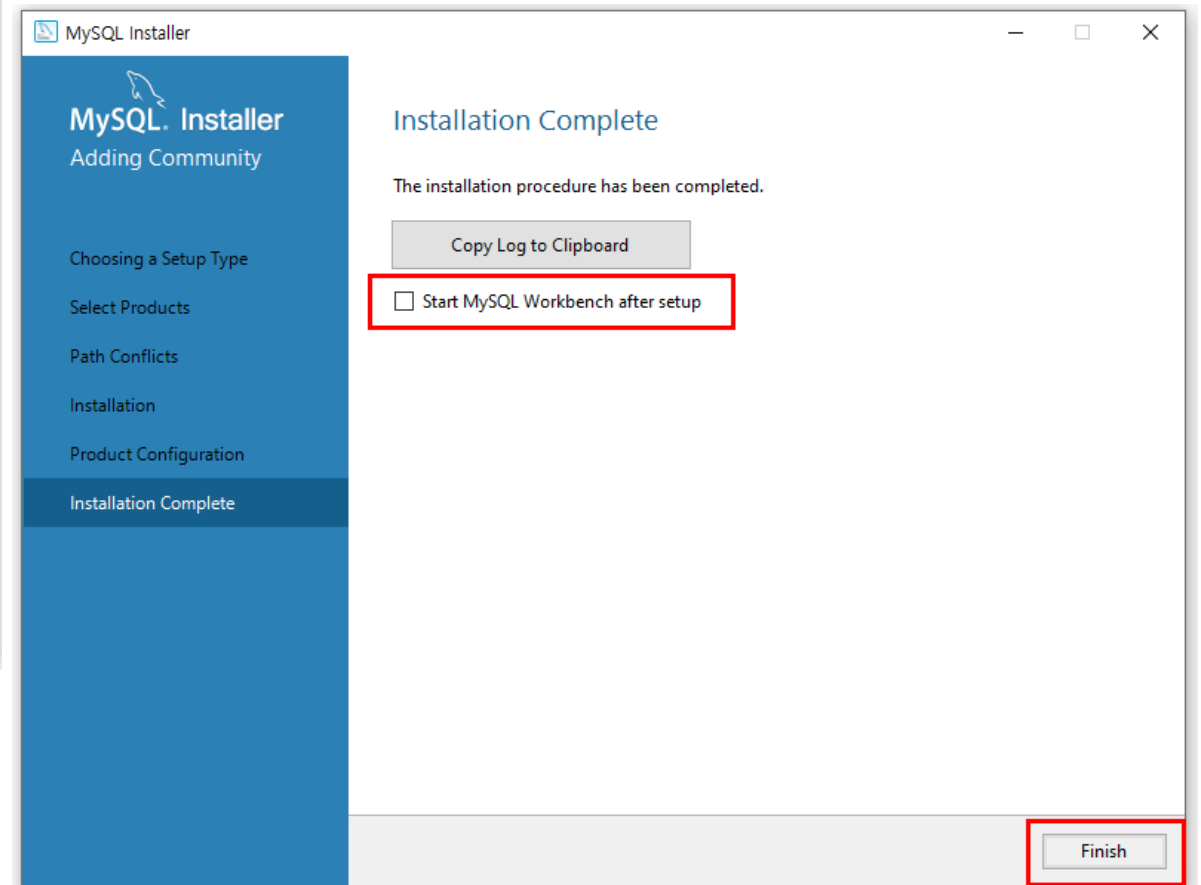
➤ Finish를 클릭한다.



➤ Next를 클릭한다.

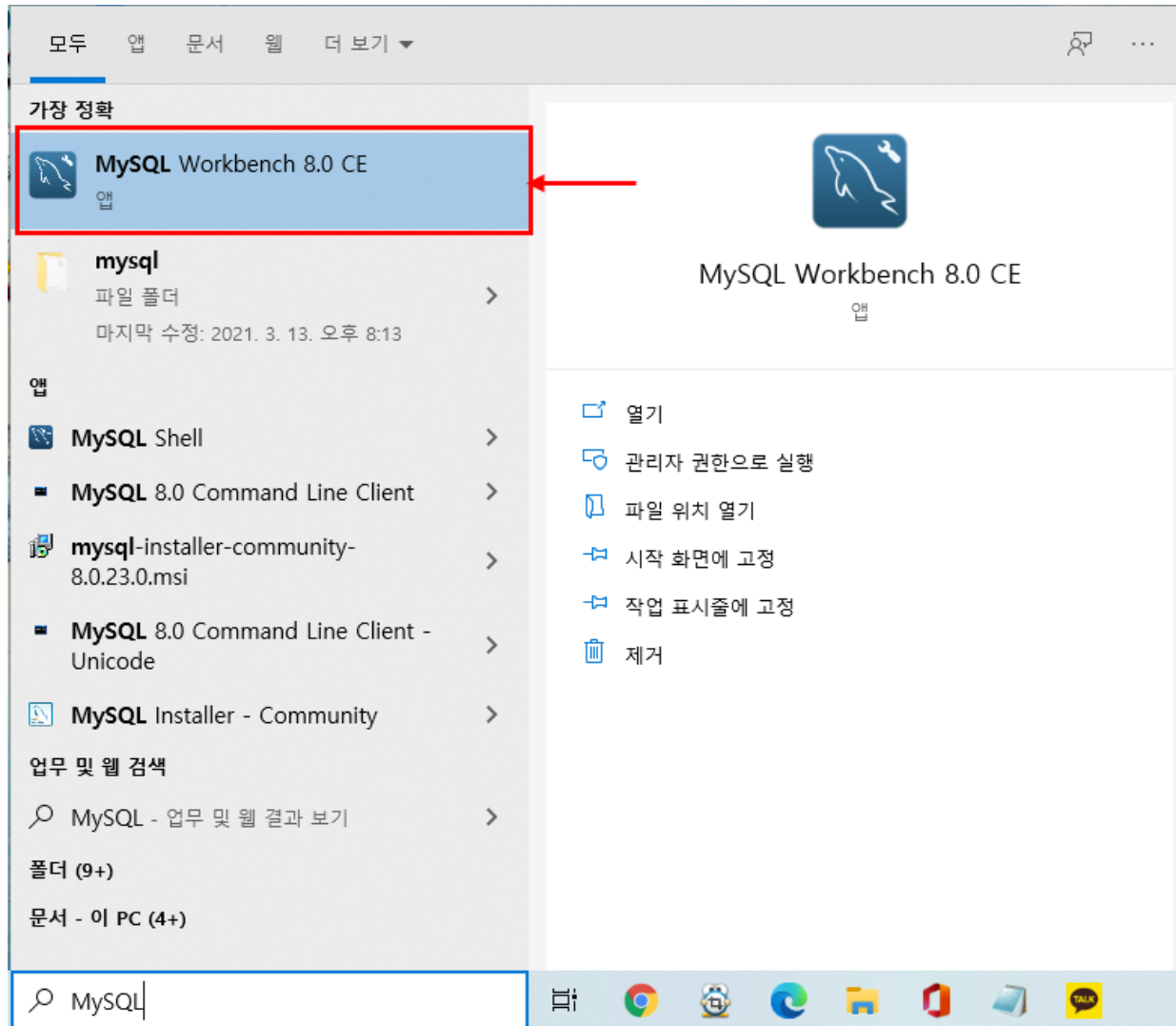


➤ 설치 및 설정이 완료되었다.
➤ Workbench를 실행하지 않고 종료(Finish)를 클릭한다.

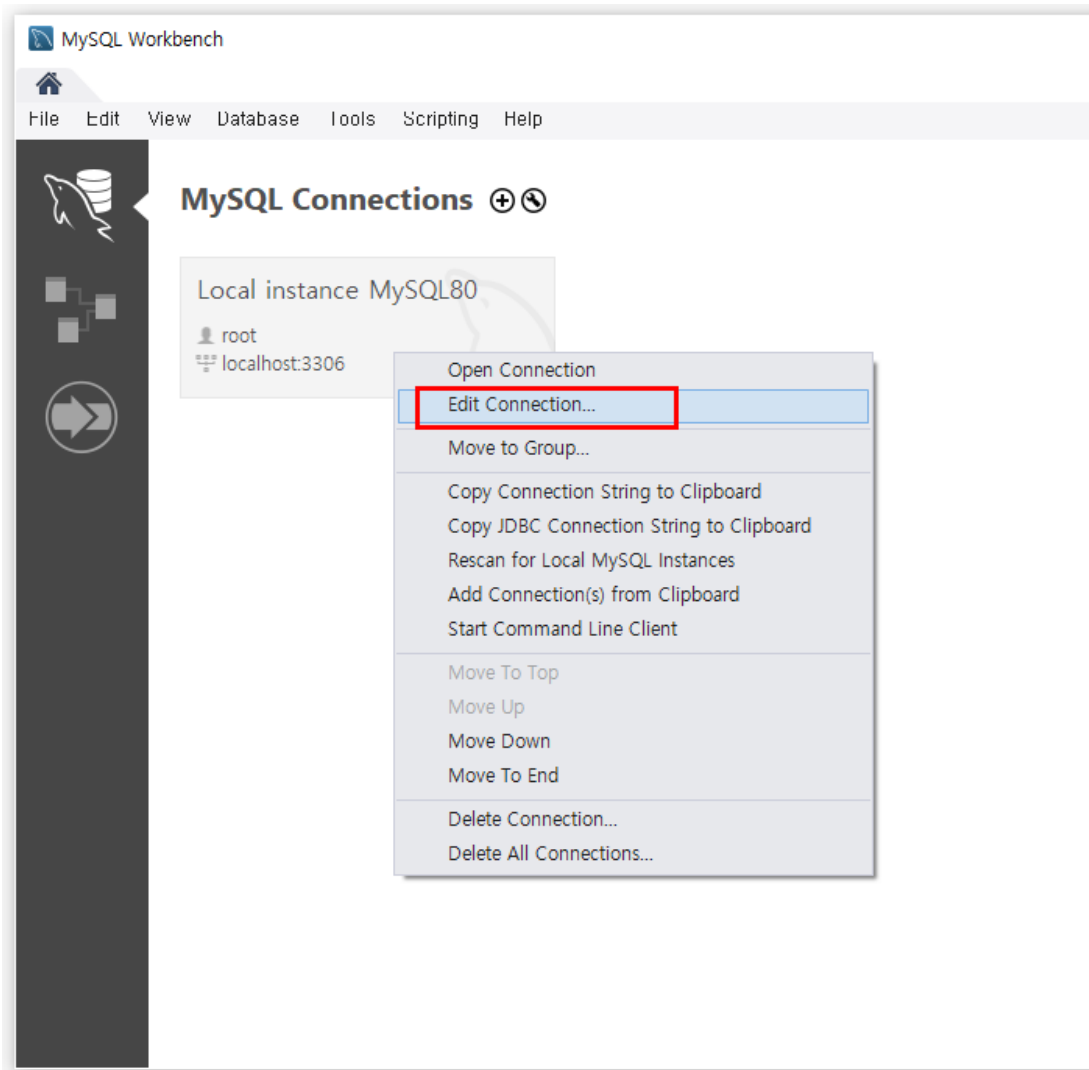


5. Workbench 8.0 실행하기

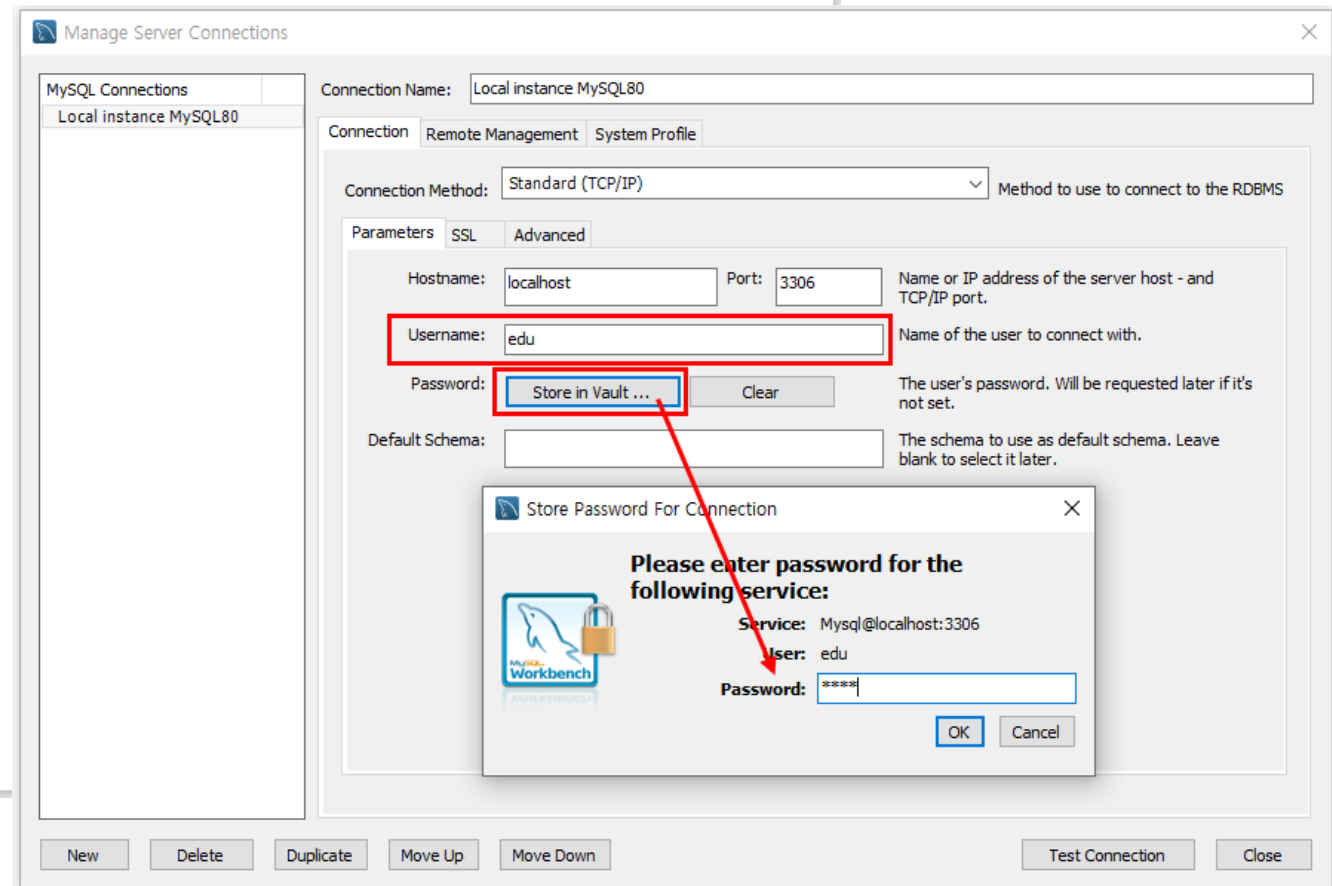
➤ 콘솔 창 → MySQL(입력) → MySQL Workbench 8.0 CE (실행)



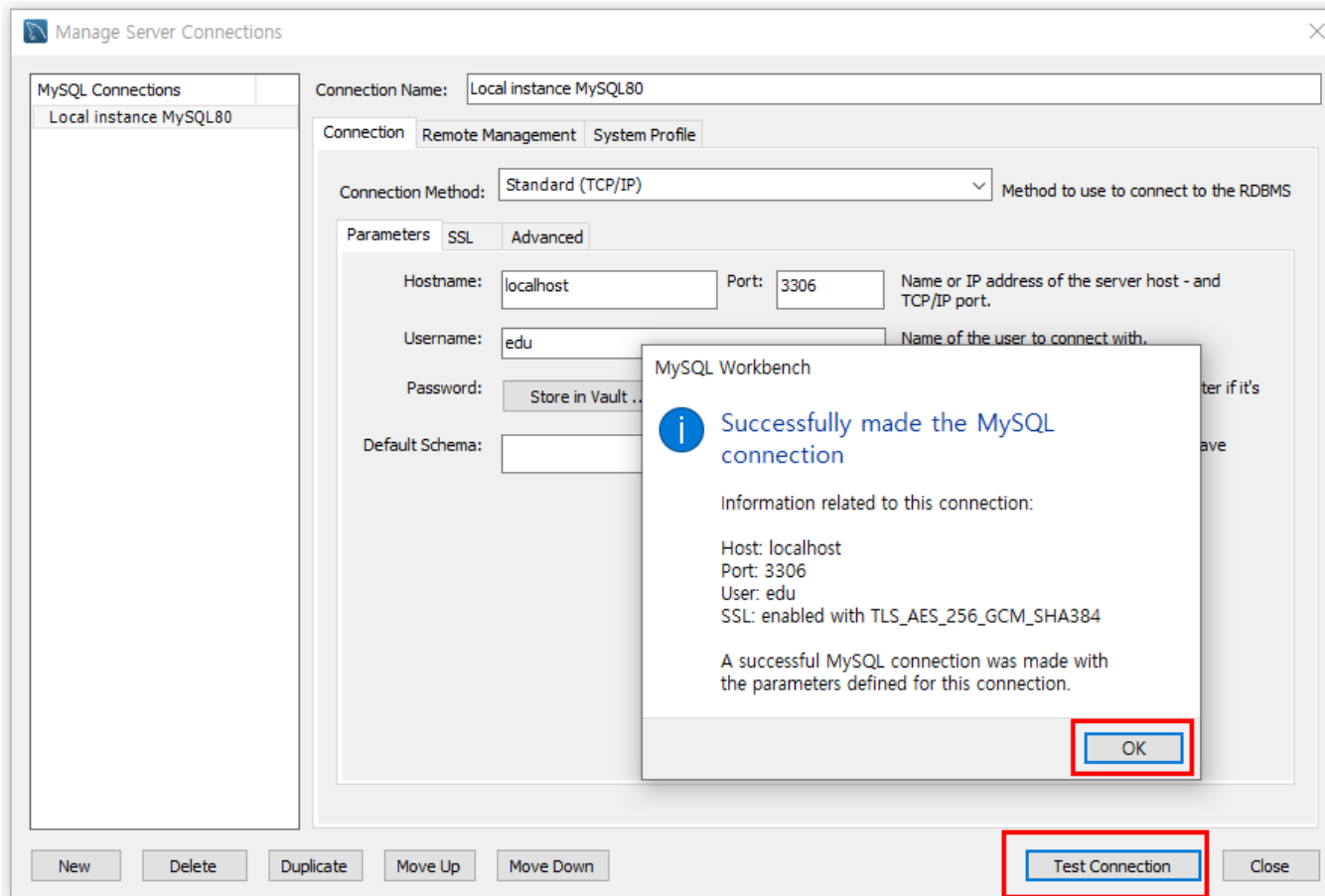
➤ root계정을 edu 계정으로 바꾸려면 Edit Connection을 선택한다.



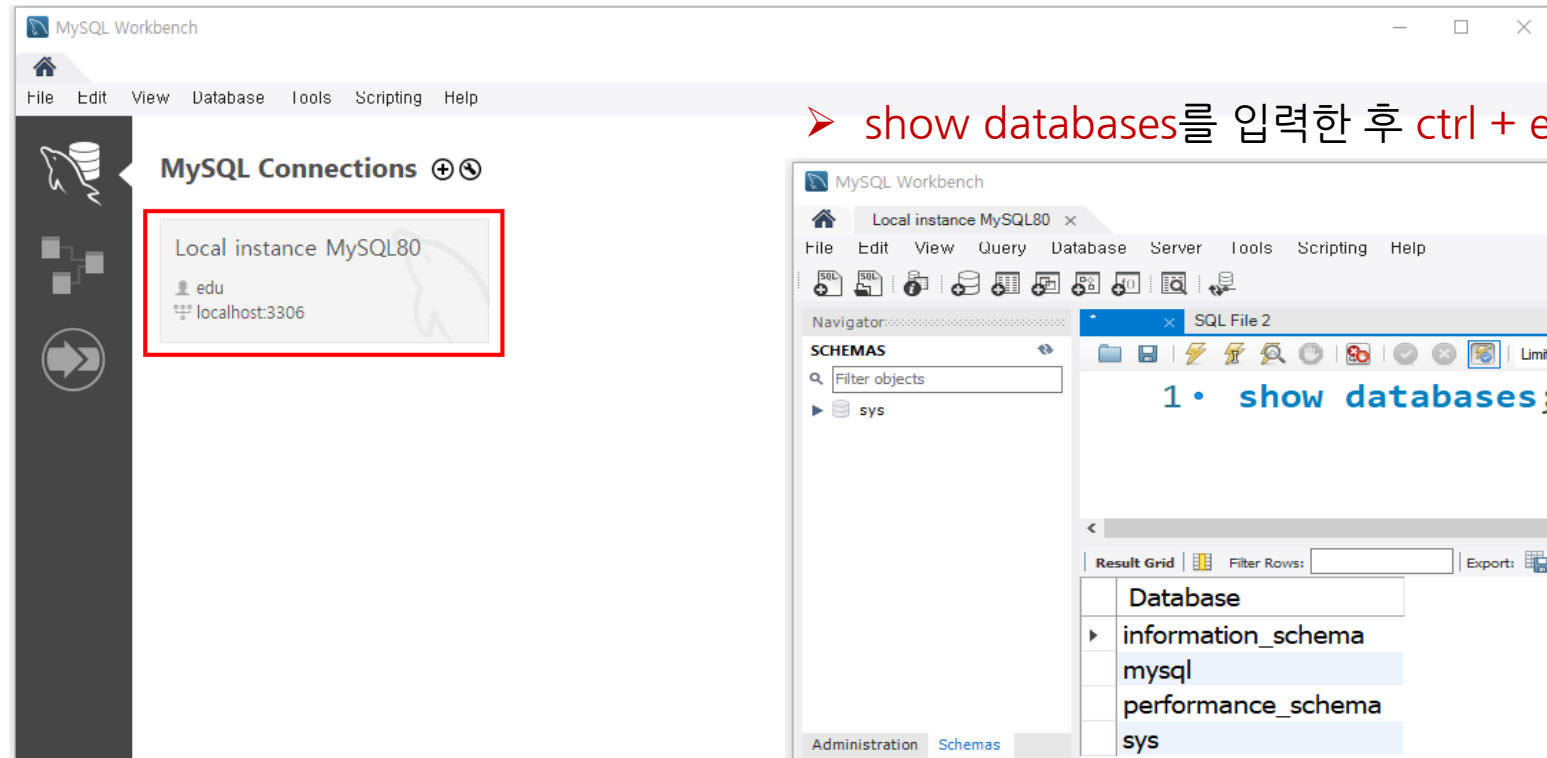
➤ 계정을 edu로 바꾸고 비밀번호 1234로 바꾼 후 OK를 클릭한다.



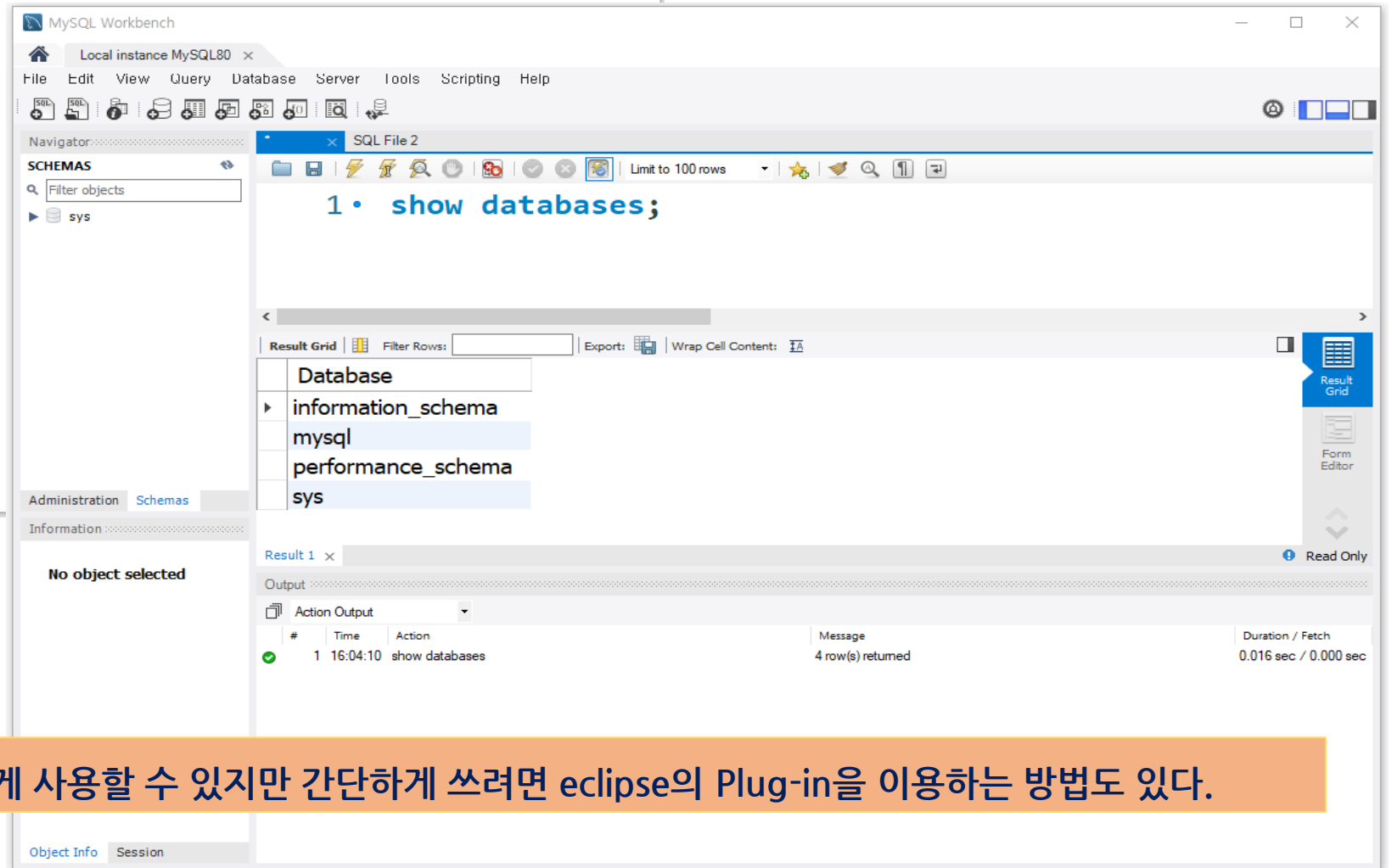
- Test Connection을 눌러서 Successfully를 확인하면 OK를 클릭한다.



➤ 계정이 edu로 바뀐 것을 확인할 수 있다.



➤ show databases를 입력한 후 ctrl + enter로 데이터베이스 목록을 확인 해 본다.



➤ Workbench는 MySQL을 편리하게 사용할 수 있지만 간단하게 쓰려면 eclipse의 Plug-in을 이용하는 방법도 있다.

❖ 연습용 테이블 설치

- MySQL 수업을 위해 제공된 “mysql_creobjects.sql” 을 설치 한다.

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows a tree view of databases. The 'testdb' database is selected, and its tables are listed: countries, departments, employees, job_history, jobs, locations, regions, and salgrades. A red box highlights this list, with a red circle and the number '3' next to it. In the center, the 'Query' editor shows a list of SQL commands. A red box highlights the first two commands: 'create database testdb;' and 'use testdb;', with a red circle and the number '2' next to it. At the bottom, the 'Output' pane shows the results of the SQL execution. A red box highlights the first two rows of the output, with a red circle and the number '3' next to it. The output shows the creation of indexes for the 'departments' and 'job_history' tables.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

happyhouse

mydb

ssafydb

sys

testdb

Tables

countries

departments

employees

job_history

jobs

locations

regions

salgrades

Views

Stored Procedures

Functions

world

Administration Schemas

Information

No object selected

Object Info Session

Query

mysql_creobjects

Limit to 100 rows

```
1 • create database testdb;
2
3 • use testdb;
4
5 -- ALTER TABLE employees DROP FOREIGN KEY emp_dept_fk ;
6
7 -- ALTER TABLE departments DROP FOREIGN KEY dept_loc_fk ;
8
9 • drop table IF EXISTS job_history CASCADE;
10
11 • drop table IF EXISTS employees CASCADE;
12
13 • drop table IF EXISTS jobs CASCADE;
14
15 • drop table IF EXISTS departments CASCADE;
```

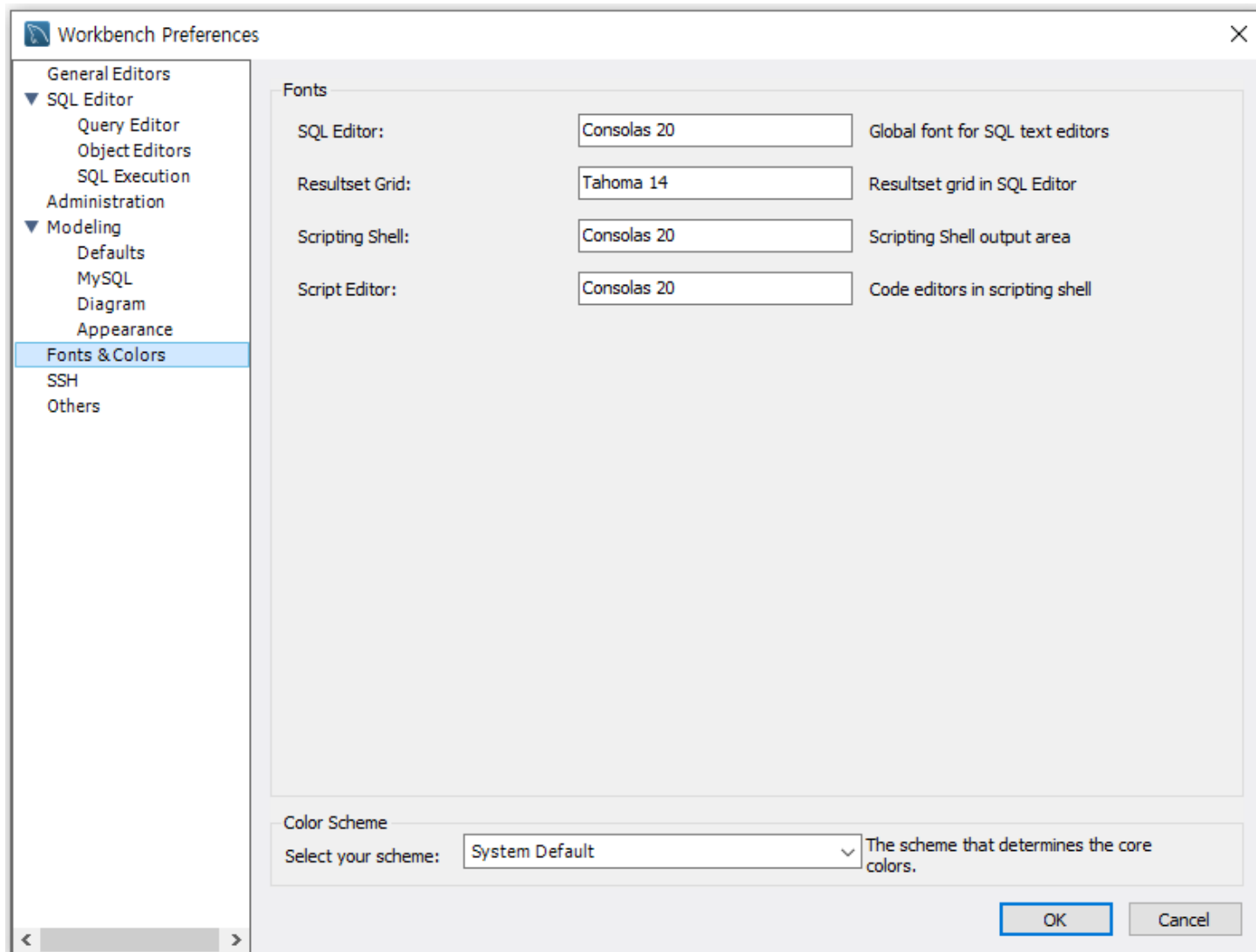
Output

Action Output

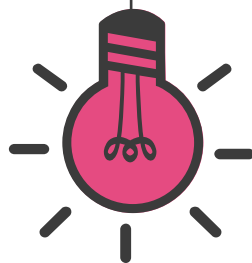
#	Time	Action	Message	Duration / Fetch
✓ 258	04:11:28	CREATE INDEX dept_location_ix ON departments (location_id)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
✓ 259	04:11:28	CREATE INDEX jhist_job_ix ON job_history (job_id)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.015 sec
✓ 260	04:11:28	CREATE INDEX jhist_employee_ix ON job_history (employee_id)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec
✓ 261	04:11:29	CREATE INDEX jhist_department_ix ON job_history (department_id)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.015 sec
✓ 262	04:11:29	CREATE INDEX loc_city_ix ON locations (city)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.032 sec
✓ 263	04:11:29	CREATE INDEX loc_state_province_ix ON locations (state_province)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.031 sec
✓ 264	04:11:29	CREATE INDEX loc_country_ix ON locations (country_id)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.015 sec
✓ 265	04:11:29	COMMIT	0 row(s) affected	0.000 sec

❖ 기타 환경 설정

- MySQL Workbench 글꼴 설정: edit → preferences

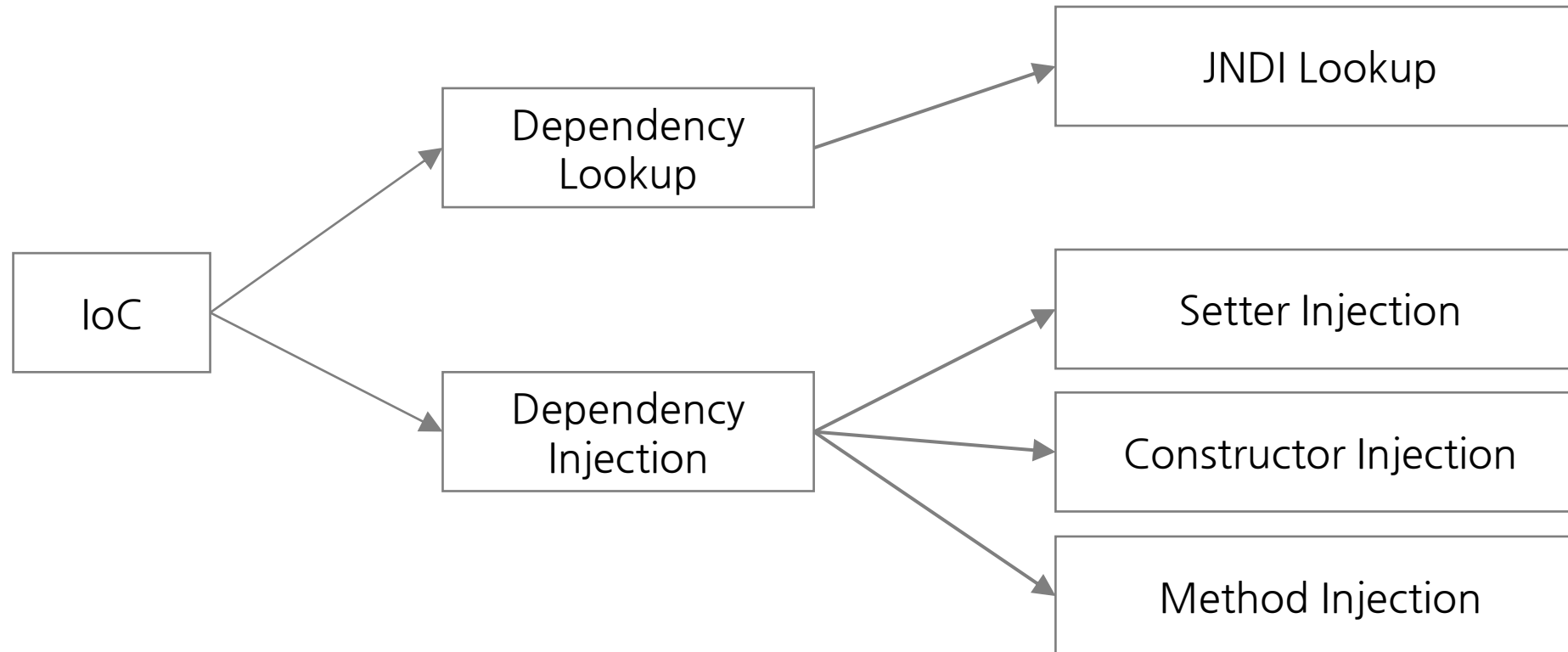


03 DI 와 AOP



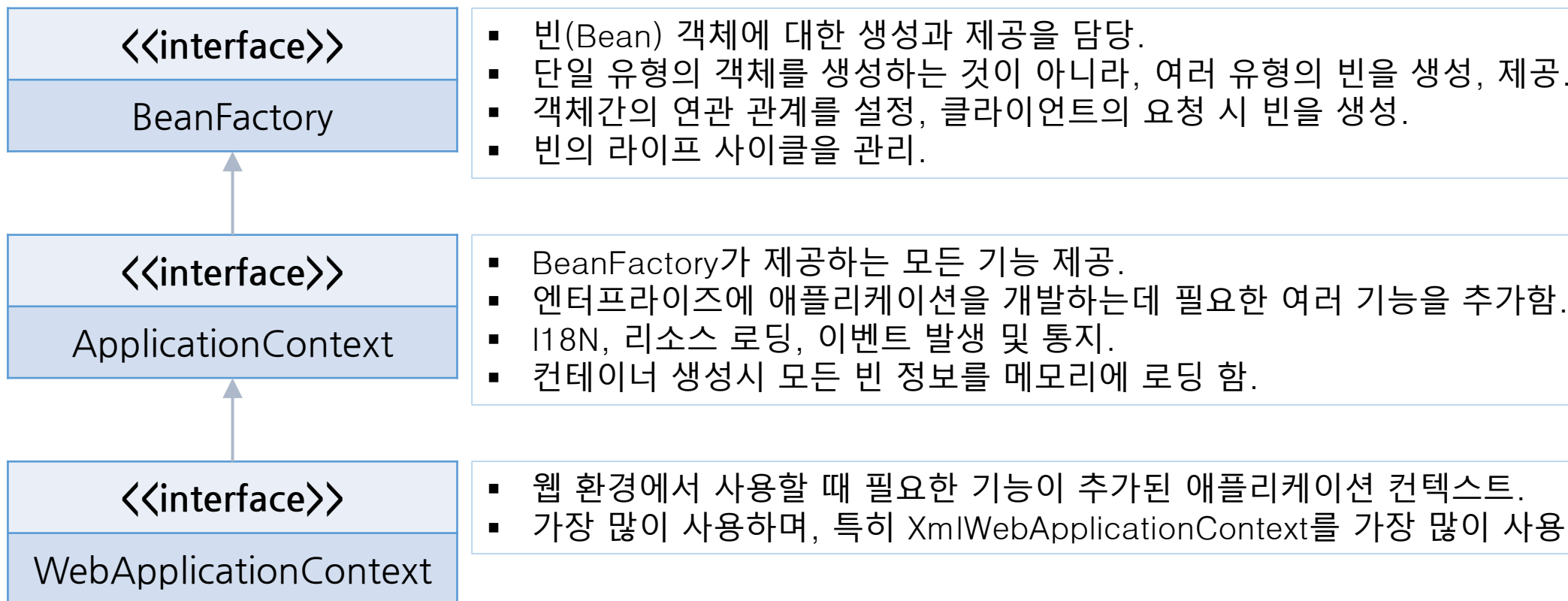
1. IoC (Inversion of Control, 제어의 역행)

- 객체지향 언어에서 Object간의 연결 관계를 런타임에 결정
- 객체 간의 관계가 느슨하게 연결됨(loose coupling)
- IoC의 구현 방법 중 하나가 DI(Dependency Injection)

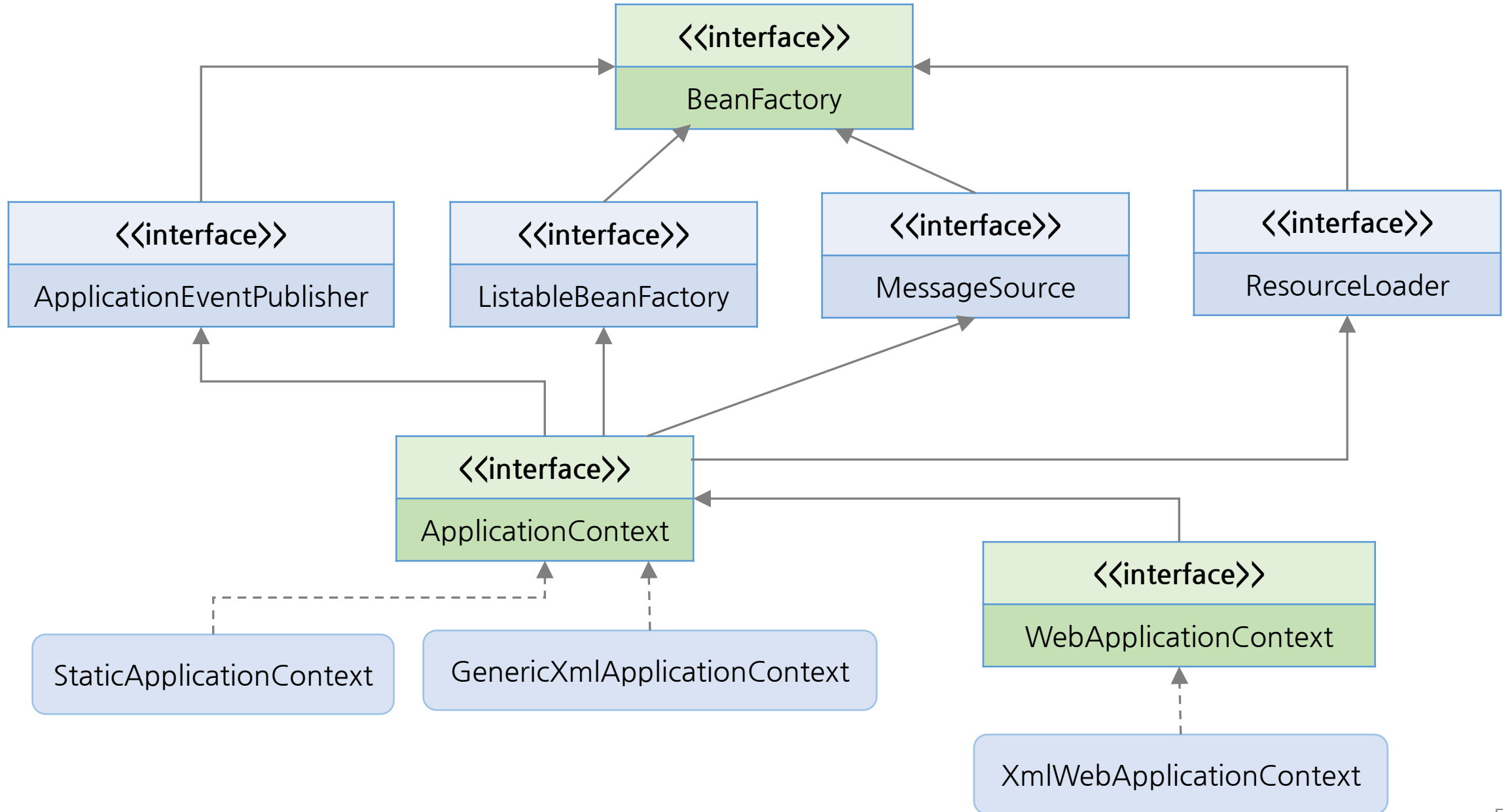


❖ Spring DI Container

- Spring DI Container가 관리하는 객체를 빈(Bein)이라 한다.
- 빈 들의 생명주기(Life-Cycle)를 관리한다는 뜻으로 빈 팩토리(BeinFactory)라 한다.
- BeinFactory에 여러 가지 컨테이너 기능을 추가한 ApplicationContext가 있다.



❖ BeanFactory & ApplicationContext



❖ IoC 개념

(1) 객체 제어 방식

- 기존 : 필요한 위치에서 개발자가 필요한 객체 생성 로직 구현 한다.
- IoC : 객체 생성을 Container에게 위임하여 처리 한다.

(2) IoC 사용에 따른 장점

- 객체 간의 결합도를 떨어뜨릴 수 있다 (loose coupling)

(3) 객체간 결합도가 높은 경우

- 해당 클래스가 유지보수 될 때 그 클래스와 결합된 다른 클래스도 같이 유지보수 되어야 할 가능성이 높다

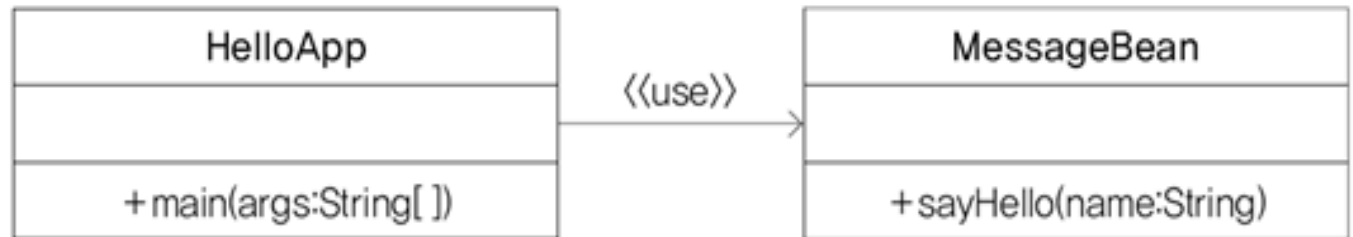
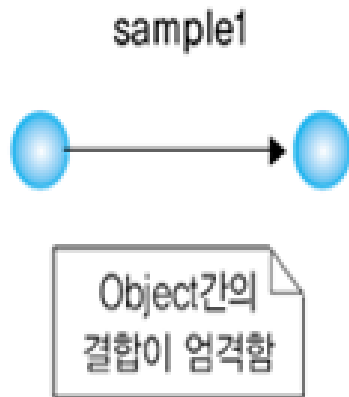
2. DI (Dependency Injection)

(1) 의존성 주입(DI) 개념

- 객체 간의 결합을 느슨하게 하는 스프링의 핵심 기술이다.
- 객체사이의 의존관계를 자기자신이 아닌 외부에 의해서 설정된다는 개념이다.
- 제어의 역행(inversion of Control, IoC) 이라는 의미로 사용한다.
- DI컨테이너는 어떤 클래스가 필요로 하는 인스턴스를 자동으로 생성,취득하여 연결시켜주는 역할을 한다.
- 느슨한 결합(loose coupling)의 주요강점
 - 객체는 인터페이스에 의한 의존관계만을 알고 있으며, 이 의존관계는 구현 클래스에 대한 차이를 모르는 채 서로 다른 구현으로 대체가 가능하다.

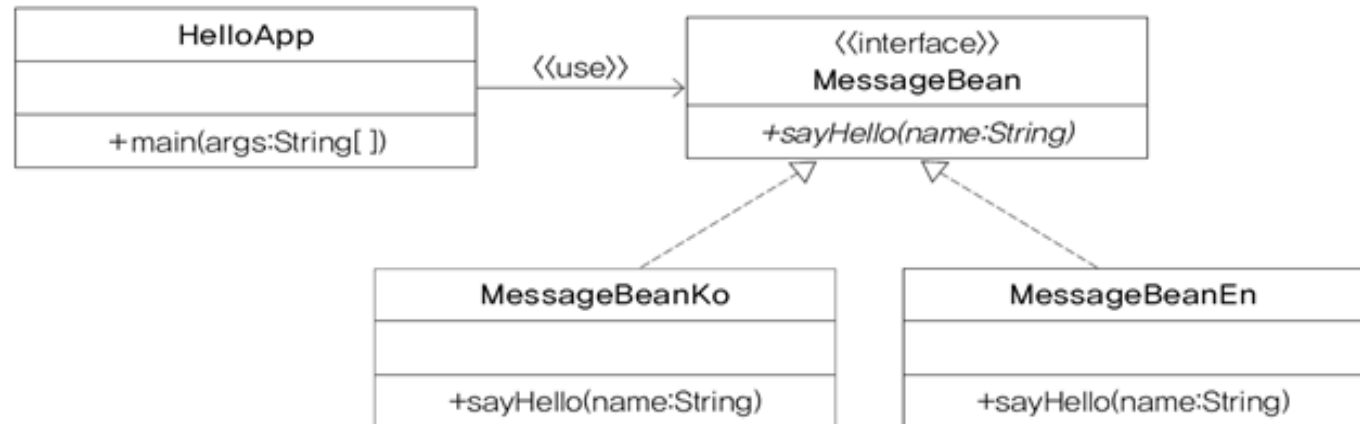
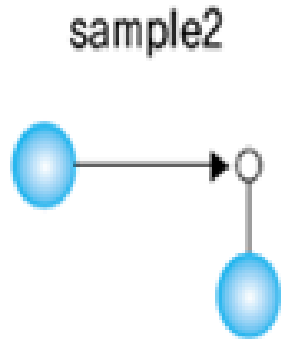
❖ 객체간 결합도가 강한 프로그램

- HelloApp 에서 MessageBean 을 직접 객체 생성하여 사용하고 있다.
- MessageBean 클래스를 다른 클래스로 변경할 경우 HelloApp 의 소스를 같이 수정해 주어야 한다.



❖ 인터페이스를 사용하여 객체간 결합도를 낮춘 프로그램

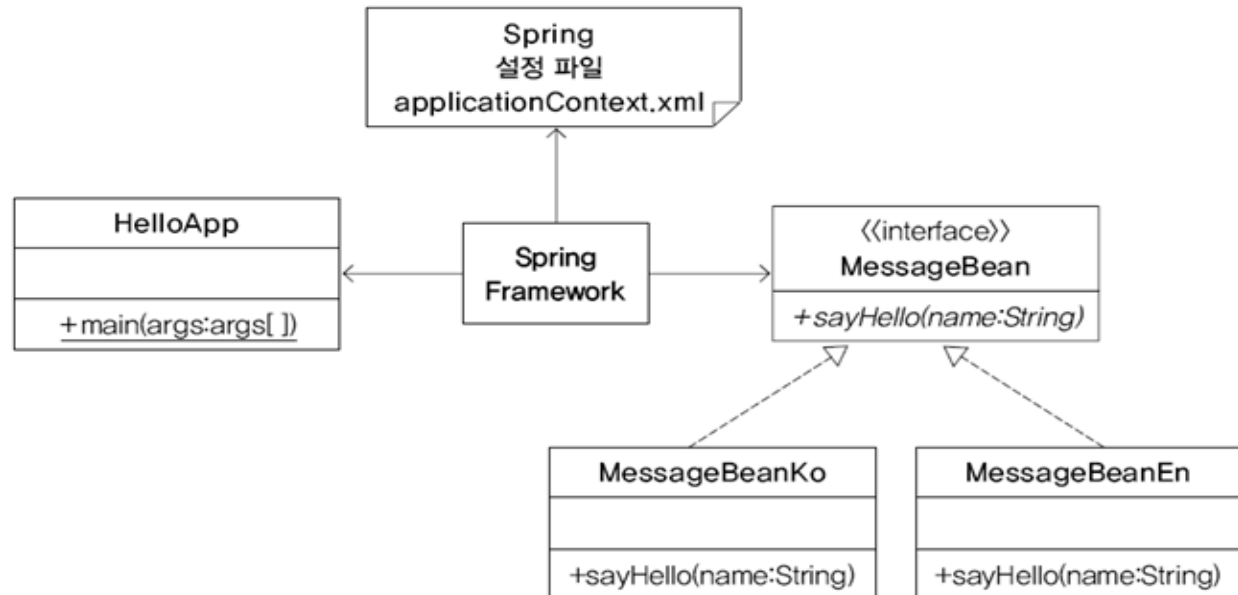
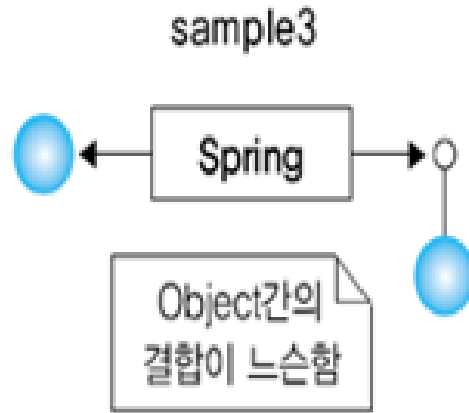
- HelloApp 는 MessageBean 이라는 인터페이스를 통해서 객체를 사용한다.
- 일반적으로 팩토리 메서드를 활용하여 사용할 객체(MessageBeanKo 또는 MessageBeanEn)를 생성한다. MessageBean 이라는 MessageBeanKo의 객체가 생성되든 MessageBeanEn의 객체가 생성되든 HelloApp 는 수정될 사항이 없다.



❖ 스프링을 사용한 객체간 결합도를 낮춘 프로그램

- 프로그램에서 필요한 객체를 스프링컨테이너가 미리 생성하여 이 객체를 필요로 하는 프로그램에 생성자 또는 Setter 메서드를 통해서 전달(주입)한다.
- 어떠한 객체를 생성하여 전달할지는 디스크립터 파일(**XML**로 작성)을 사용한다

선언적인 프로그래밍



(2) 스프링 설정파일

❖ BeanFactory인터페이스

- Object getBean() : 인수에 지정된 이름의 Bean인스턴스를 생성해서 반환한다.
- boolean containsBean(String name) : 인수에 지정된 이름의 Bean이 정의 되어 있는지 여부를 반환한다.
- String []getAliases(String name) : Bean이름에 Alias 가 정의 되어 있는 경우, 그 Alias를 반환한다.

❖ bean요소의 설정

속성	설명
id	식별자(고유 값)
name	id에 대한 별칭, 복수정의 가능
class	Bean클래스 이름, 완전한 형태의 클래스이름을 기술
parent	Bean정의를 상속하는 경우 지정한 새로운 Bean의 id

❖ bean요소의 설정

속성	설명
abstract	Bean클래스가 추상 클래스인지 여부
singleton	Bean이 싱글 톤을 관리하는지 여부
lazy-init	Bean의 로딩을 지연시킬지 여부
autowire	자동 와이어 여부
dependency-check	의존관계확인 방법
depends-on	이 Bean이 의존할 Bean이름, 먼저 초기화 되는 것이 보장된다
init-method	Bean초기화시 실행시킬 메서드
destroy-method	Bean컨테이너 종료 시 실행시킬 메서드

❖ bean요소의 설정

속성	설명
scope	객체생성을 어떻게 할지 여부
	<p><bean... scope="singleton"></p> <ul style="list-style-type: none"> - 객체를 한번만 생성하고, 그 후부터는 이미 생성된 객체를 재활용(default) - 객체 생성 시점 ApplicationContext context = new ClassPathXmlApplicationContext("xml파일"); ➔ 객체를 공유 <p><bean... scope="prototype"></p> <ul style="list-style-type: none"> - 객체를 매번 new instance로 생성해서 반환함 - 객체 생성 시점 : Object obj = (Object) context.getBean() 시 매번 생성. <p>other scopes : (web관련 scope)</p> <ul style="list-style-type: none"> - scope="request" (HTTP Request별로 새로운 인스턴스 생성) - scope="session" (HTTP Session별로 새로운 인스턴스 생성) - scope="globalSession"
factory-method	속성값으로 static method를 지정해서 해당 method를 이용하여 빈을 생성하도록 설정

(3) 의존 관계를 관리하기 위한 방법

1. Construction Injection

생성자를 통해서 의존관계를 연결시키는 것을 말한다

2. Setter Injection

클래스 사이의 의존관계를 연결시키기 위해서 setter메소드를 이용하는 방법을 말한다

❖ ApplicationContext.xml

(1) constructor-arg요소속성

속성	설명
index	constructor의 몇 번째의 인수에 값을 전달할 것인지 지정
type	constructor의 어느 데이터형의 인수에 값을 전달할 것인지 지정
ref	자식요소<ref bean="빈 이름"/>대신에 사용할 수 있다
value	자식요소 <value>값</value> 대신에 사용할 수 있다

(2) property요소의 속성

속성	설명
ref	자식요소<ref bean="빈 이름"/>대신에 사용할 수 있다.
value	자식요소 <value>값</value> 대신에 사용할 수 있다.

3. AOP(Aspect Oriented Programming)

(1) 관점지향 프로그램(AOP)의 개념

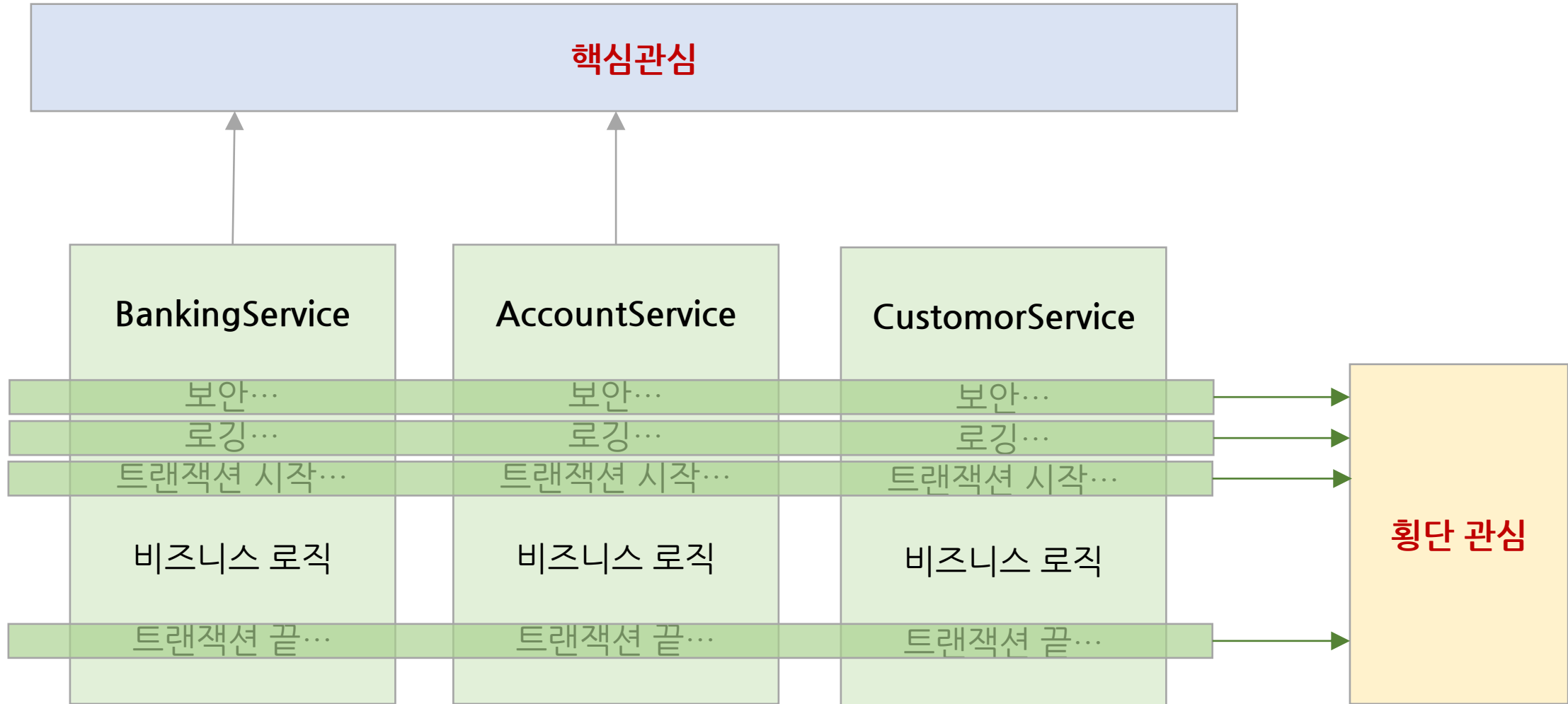
- AOP는 객체지향프로그램의 뒤를 이은 또 하나의 프로그래밍언어구조이다
- 관점지향의 중요한 개념은 '횡단 관점의 분리(Separation of Cross-Cutting Concern)' 이다

AOP

문제를 바라보는 관점을 기준으로
프로그래밍 하는 기법

- 핵심 관심 사항(core concern)과 공통 관심 사항(cross-cutting concern)
- 기존 OOP에서는 공통관심사항을 여러 모듈에서 적용하는데 있어 중복된 코드를 양상 하는 한계가 존재한다. 이를 해결하기 위해 AOP가 등장하였다.
- Aspect Oriented Programming은 문제를 해결하기 위한 핵심 관심 사항과 전체에 적용되는 공통 관심 사항을 기준으로 프로그래밍함으로써 공통 모듈을 손쉽게 적용할 수 있게 하였다.

❖ 관심의 산재



❖ 관심의 산재

AOP기법 적용 전
(Before)



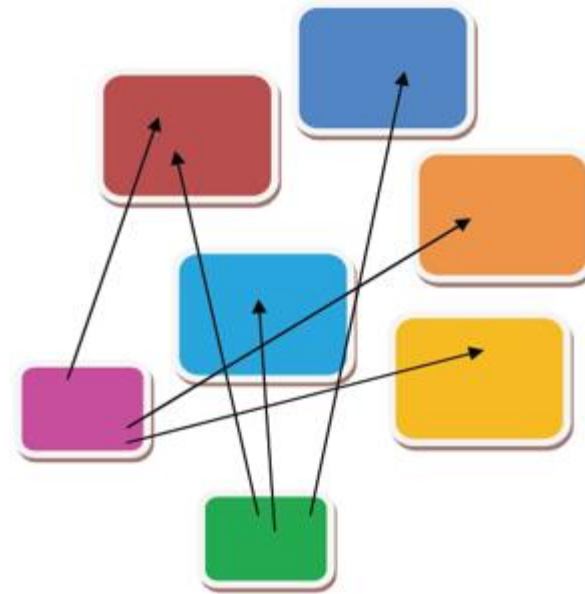
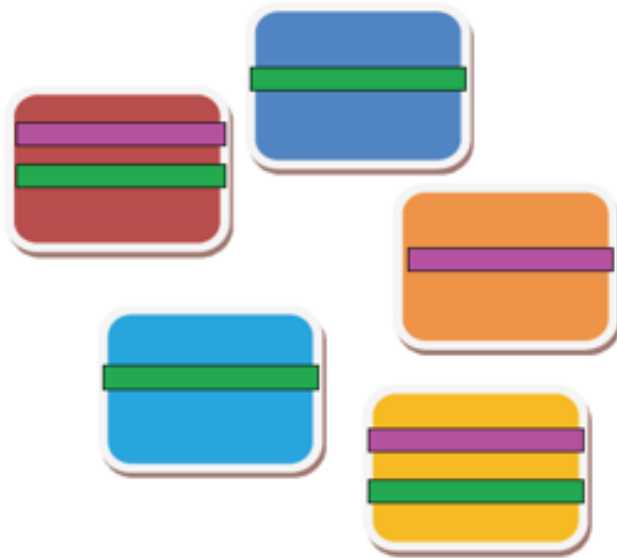
→
AOP 적용

AOP기법 적용 후
(After)

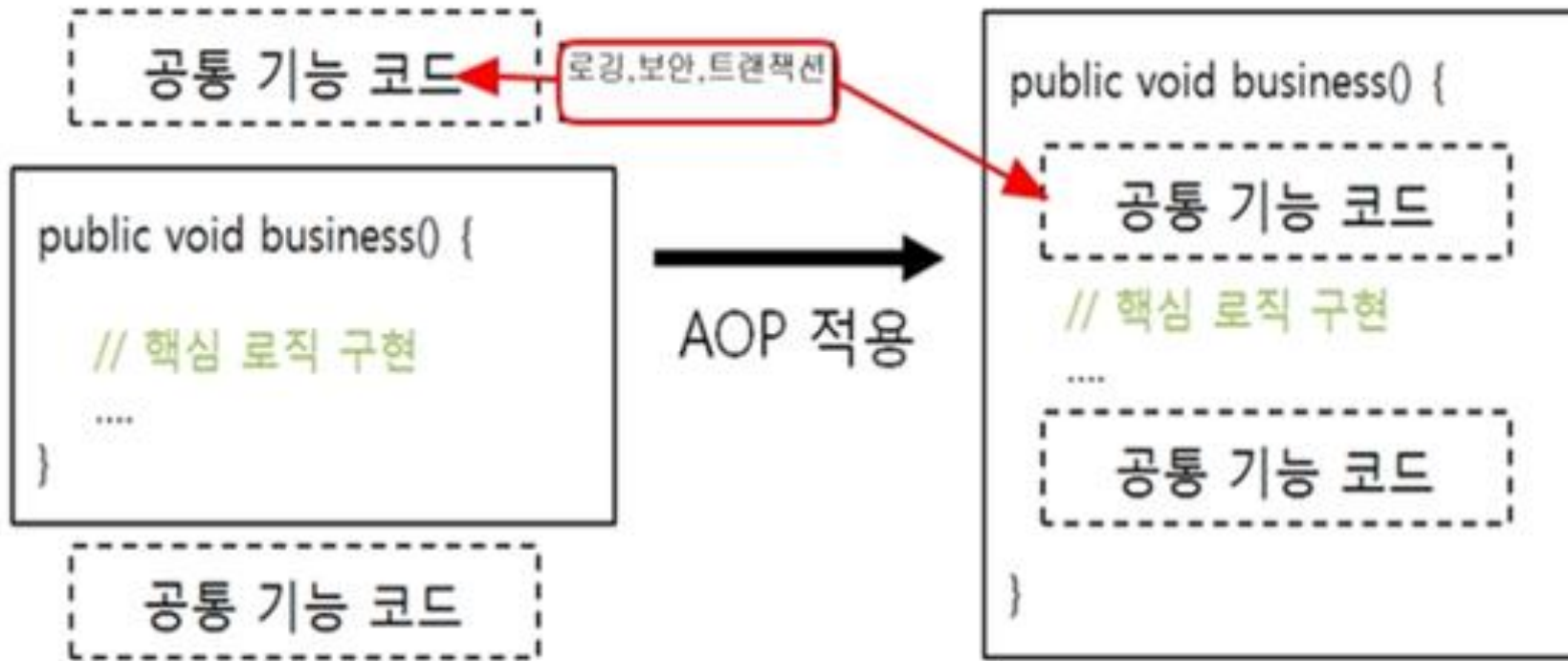


❖ 관심의 산재

AOP를 적용한 OOP 프로그래밍



❖ 관심의 산재



공통관심 사항을 구현한 코드를 AOP기법으로 적용하면 그림과 같이 핵심 로직 안에 삽입된다.

(2) AOP 적용 예 (1 / 2)

- 간단한 메소드의 성능 검사

- 개발 도중 특히 DB에 다양한 데이터를 넣고 빼는 등의 배치 작업에 대하여 시간을 측정해 보고 쿼리를 개선하는 작업은 매우 의미가 있다.
- 매번 해당 메소드의 처음과 끝에 `System.currentTimeMillis();`를 사용하거나, 스프링이 제공하는 `StopWatch`코드를 사용하기는 매우 번거롭다.
- 이 경우 해당 작업을 하는 코드를 밖에서 설정하고 해당 부분을 사용하는 것이 편리하다.

- 트랜잭션 처리

- 트랜잭션의 경우 비즈니스 로직의 전후에 설정된다.
- 하지만, 매번 사용하는 트랜잭션(`try{ ~~~ }catch{ }`)의 코드는 번거롭고 소스를 더욱 복잡하게 만든다.

(2) AOP 적용 예 (2 / 2)

- 예외반환

- 스프링에는 `DataAccessException`이라는 매우 잘 정의되어 있는 예외 계층 구조가 있다.
- 예전 하이버네이트 예외들은 몇 개 없었고 그나마도 `UncheckedException`이 아니었다.
- 이렇게 구조가 별로 안 좋은 예외들이 발생했을 때, 그걸 잡아서 잘 정의되어 있는 예외 계층 구조로 변환해서 다시 던지는 Aspect는 제 3의 프레임워크를 사용할 때, 본인의 프레임워크나 애플리케이션에서 별도의 예외 계층 구조로 변환하고 싶을 때 유용하다.

- 아키텍처 검증

- 기타

- 하이버네이트와 JDBC를 같이 사용할 경우, DB 동기화 문제 해결
- 멀티 쓰레드 Safety 관련하여 작업해야 하는 경우, 메소드들에 일괄적으로 락을 설정하는 Aspect
- 데드 락 등으로 인한 `PessimisticLockingFailureException`등의 예외를 만났을 때 재시도하는 Aspect
- 로깅, 인증, 권한 등

(3) AOP 용어 정리 (1 / 3)

- **Target**

- 핵심기능을 담고 있는 모듈로 타겟은 부가기능을 부여할 대상이 된다.

- **Advice**

- 어느 시점(예: method 수행 전/후, 예외발생 이후)에 어떤 공통 관심 기능(Aspect)을 적용할 지 정의한 것이다.
- 타겟에 제공할 부가기능을 담고 있는 모듈이다.

- **JoinPoint**

- Aspect가 적용 될 수 있는 지점 (예 : method, field)
- 즉 타겟 객체가 구현한 인터페이스의 모든 method는 JoinPoint가 된다.

(3) AOP 용어 정리 (2 / 3)

- **Pointcut**

- 공통 관심 사항이 적용될 JoinPoint이다.
- Advice를 적용할 타겟의 method를 선별하는 정규표현식이다.
- Pointcut 표현식은 execution으로 시작하고, method의 Signature를 비교하는 방법을 주로 이용한다.

- **Aspect**

- 여러 객체에서 공통으로 적용되는 공통 관심 사항 (transaction, logging, security..)
- AOP의 기본 모듈이다.
- **Aspect = Advice + Pointcut**
- Aspect는 Singleton 형태의 객체로 존재한다.

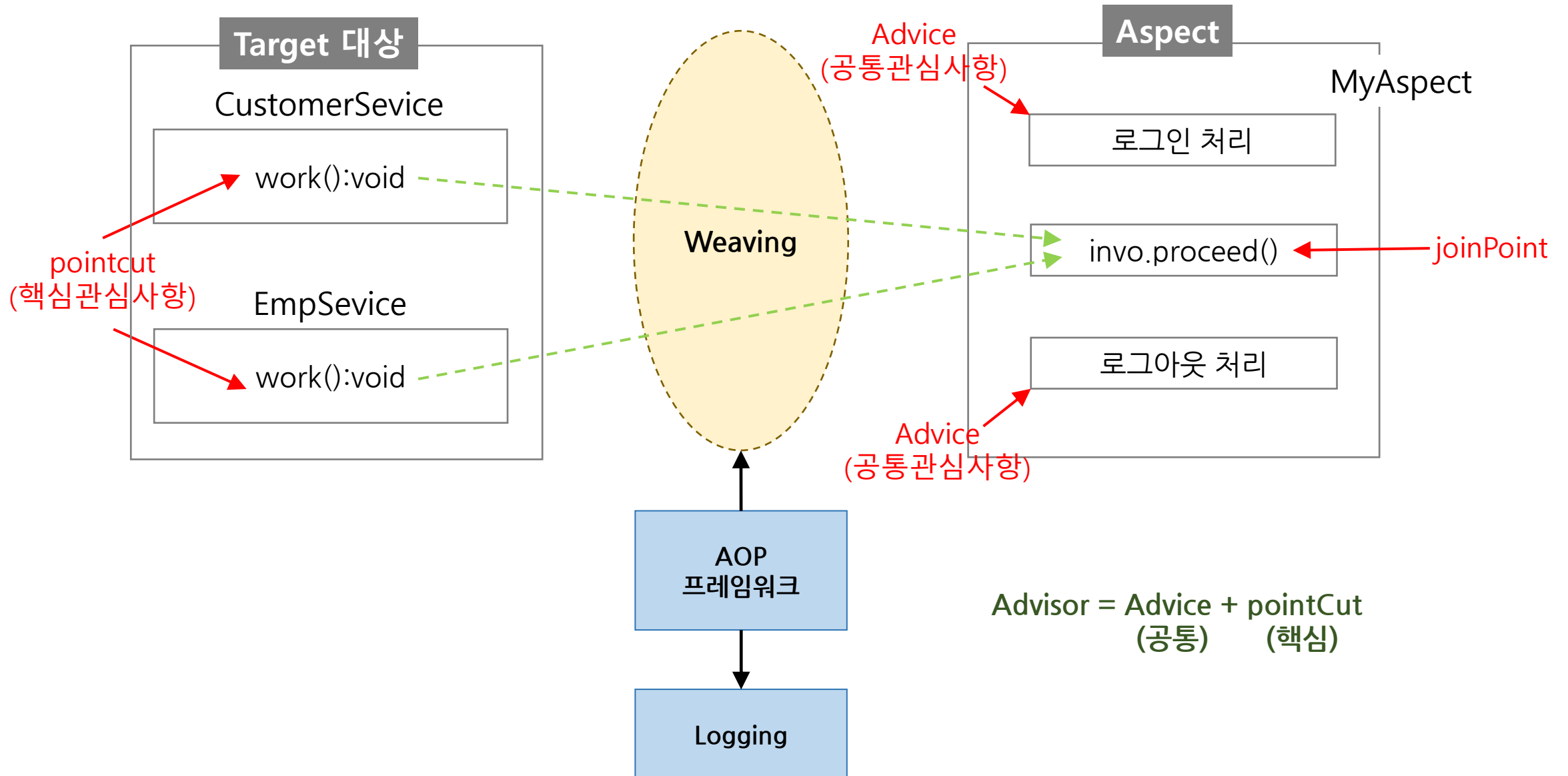
(3) AOP 용어 정리 (3 / 3)

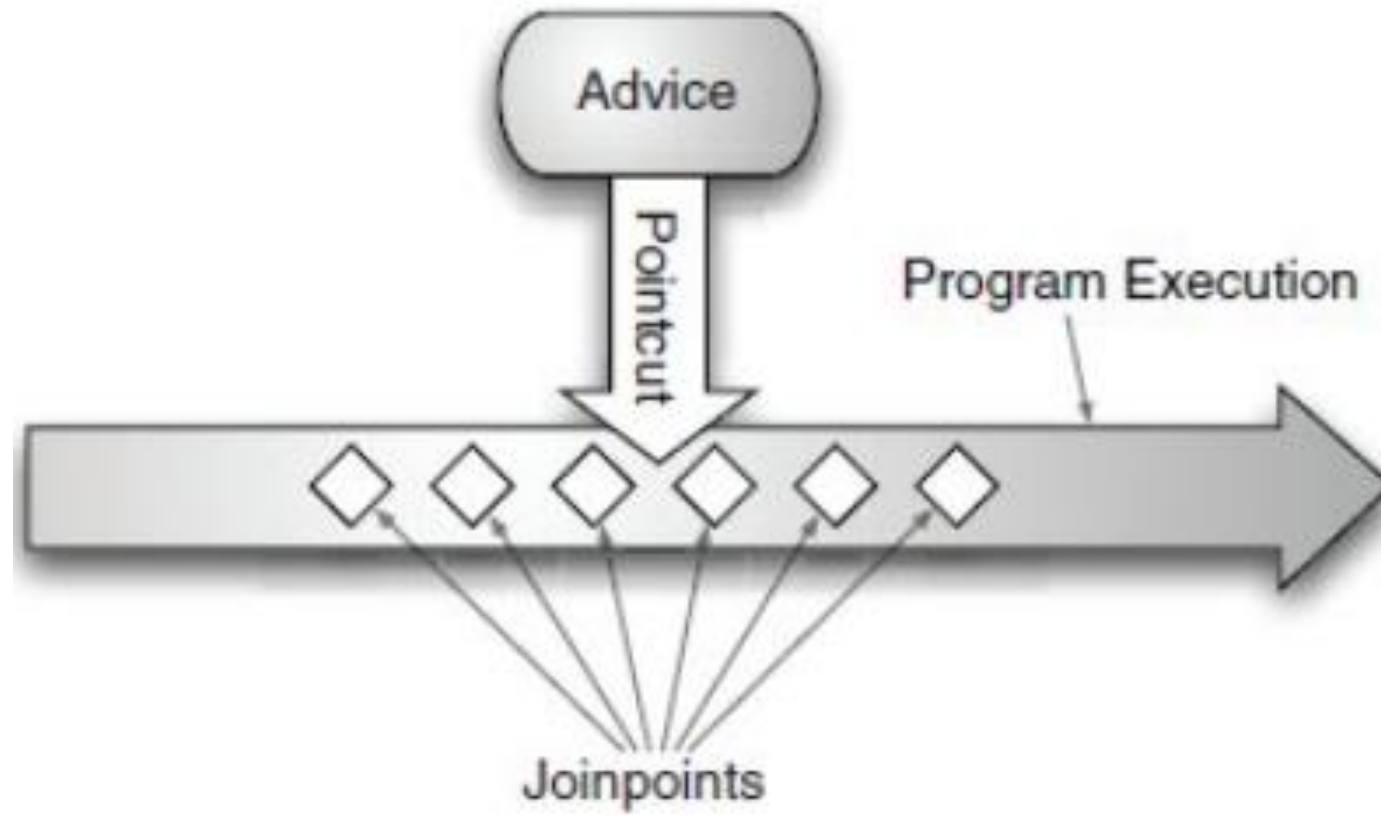
- **Advisor**

- **Advisor = Advice + Pointcut**
- Advisor는 Spring AOP에서만 사용되는 특별한 용어이다.

- **Weaving**

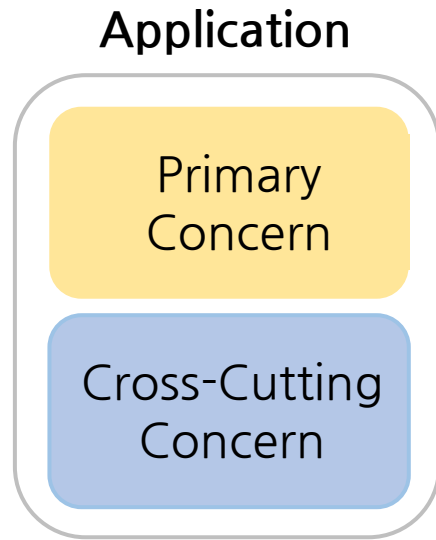
- 어떤 Advice를 어떤 Pointcut(핵심사항)에 적용시킬 것인지에 대한 설정(Advisor)이다.
- 즉 Pointcut에 의해서 결정된 타겟의 Joinpoint에 부가기능(Advice)을 삽입하는 과정을 뜻한다.
- Weaving은 AOP의 핵심기능(Target)의 코드에 영향을 주지 않으면서 필요한 부가기능(Advice)을 추가할 수 있도록 해주는 핵심적인 처리과정이다.



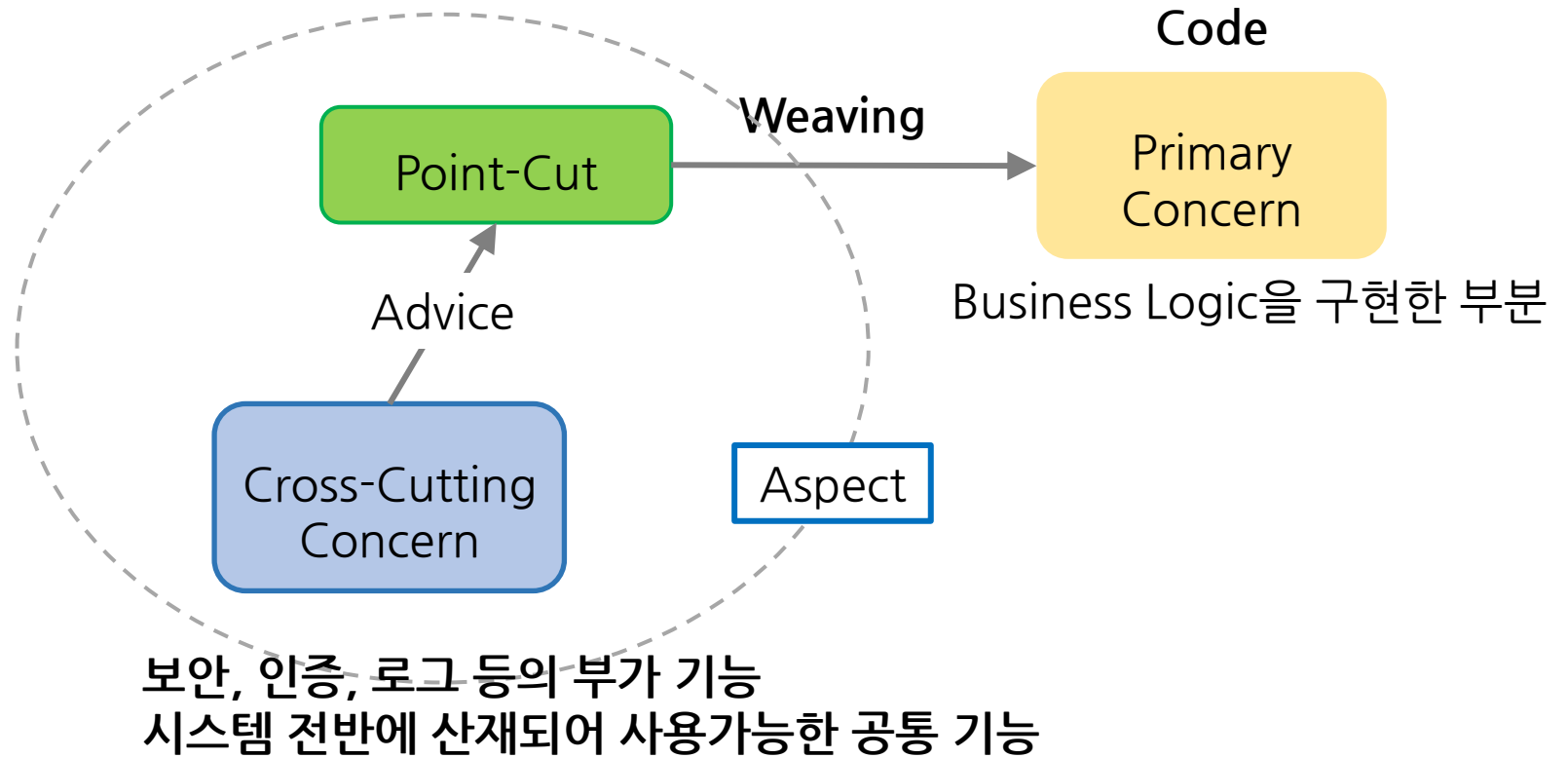


(4) AOP 비교

기존



AOP



(5) AOP 표현식

- AspectJ표현식에서 사용되는 와일드 카드

표현식	설명
*	*는 마침표를 제외한 문자가 0개 이상 나열된 것을 의미한다(.을 포함하지 않는 모든 문자열)
..	..은 마침표를 포함한 문자가 0개 이상 나열된 것을 의미한다(.을 포함한 모든 문자열)
+	+는 주어진 타입의 임의의 서브 클래스나 서브 인터페이스를 의미한다 (하위클래스 또는 하위 인터페이스)

(5) AOP 표현식

- AspectJ에서 포인트 컷 지정 시 사용되는 논리연산자

표현식	설명
!	표시된 조인 포인트를 제외한 모든 조인 포인트를 지정할 때 사용된다.
&&	두 조건을 모두 만족시키는 조인포인트를 지정할 때 사용된다.
	두 조건 중 하나만 만족하는 조인 포인트를 지정할 때 사용한다.

(5) AOP 표현식

- 매핑의 기본

표현식	설명
get*(..)	get으로 시작하는 모든 메서드를 지정한다. ex) <code>getUser()</code> , <code>getUserList()</code> , <code>get()</code> <code>getUser(UserVo vo)</code> , <code>getUserList(UserVo vo, int num)</code>
get(..)	리턴 타입이 *이면 어떤 타입이 오더라도 상관없다. ex) <code>void getUser()</code> , <code>int getUser()</code> , <code>UserVo getUser()</code>
execution(public * set*(..)) : public 이면서 set으로 시작하는 메서드 execution(* sample1.*.*(..)) : sample1 패키지에 있는 모든 클래스의 모든 메서드 execution(* sample1..*.*(..)) : sample1 패키지 하위의 모든 클래스의 모든 메서드 execution(public * set*(..)) execution(* sample1.*.*(..))	

(5) AOP 표현식

- 클래스 지정

표현식	설명
com.elitebiz.*	<ul style="list-style-type: none">*를 이용하면 특정패키지의 모든 클래스를 지정할 수 있다com.elite.biz 패키지의 모든 클래스를 지정한다
com.elite..*	<ul style="list-style-type: none">com.elite패키지 및 그 서브 패키지의 모든 클래스를 지정한다.
UserService+	<ul style="list-style-type: none">+는 서브타입을 표현할 때 사용한다UserService클래스 혹은 UserService인터페이스로 부터 파생된 모든 서브클래스를 지정한다.
!UserService	<ul style="list-style-type: none">UserService클래스 혹은 UserService인터페이스로 부터 파생되지 않은 모든 서브클래스를 지정한다.(!은 논리부정 조건을 의미한다.)
int Integer	<ul style="list-style-type: none"> 연산자는 or 연산을 표현함으로 int타입으로 표현되거나 Integer 타입으로 표현되는 요소를 지칭한다

(5) AOP 표현식

• 매개변수 지정

표현식	설명
* addUser(..)	• 매개변수의 개수와 타입이 뭐가 오든 상관없이 처리한다.
* addUser(*)	• 반드시 1개의 임의의 매개변수가 선언되어야 한다.
* addUser(*, ..)	• 최소 1개의 매개변수가 선언되어야 하며, 이후 다른 매개변수가 있어도 되고 없어도 된다 ex) addUser(UserVo vo) , addUser(String id, String pwd, int no)

• 생성자 지정

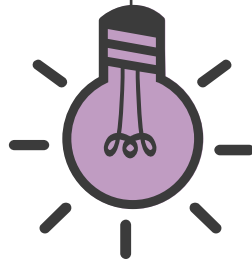
표현식	설명
new(..)	• new는 메서드의 이름을 지정하는게 아니라 생성자를 지정한다 • 매개변수 상관없이 생성자를 지정한다.
UserServ.new(..)	• UserServ 클래스의 new메서드가 아니라 UserServ 클래스의 생성자를 지정한다.

(6) AOP 구현 과정

- 1) Advice클래스를 작성한다
- 2) 설정파일에 Pointcut을 설정한다
- 3) 설정파일에 Advice와 Pointcut을 묶어 놓는 Advisor를 설정한다
- 4) 설정파일에 ProxyFactoryBean클래스를 이용하여 대상 객체에 Advisor를 적용한다
- 5) getBean()메소드로 빈 객체를 가져와 사용한다

04

Annotation



- @Autowired, @Resource

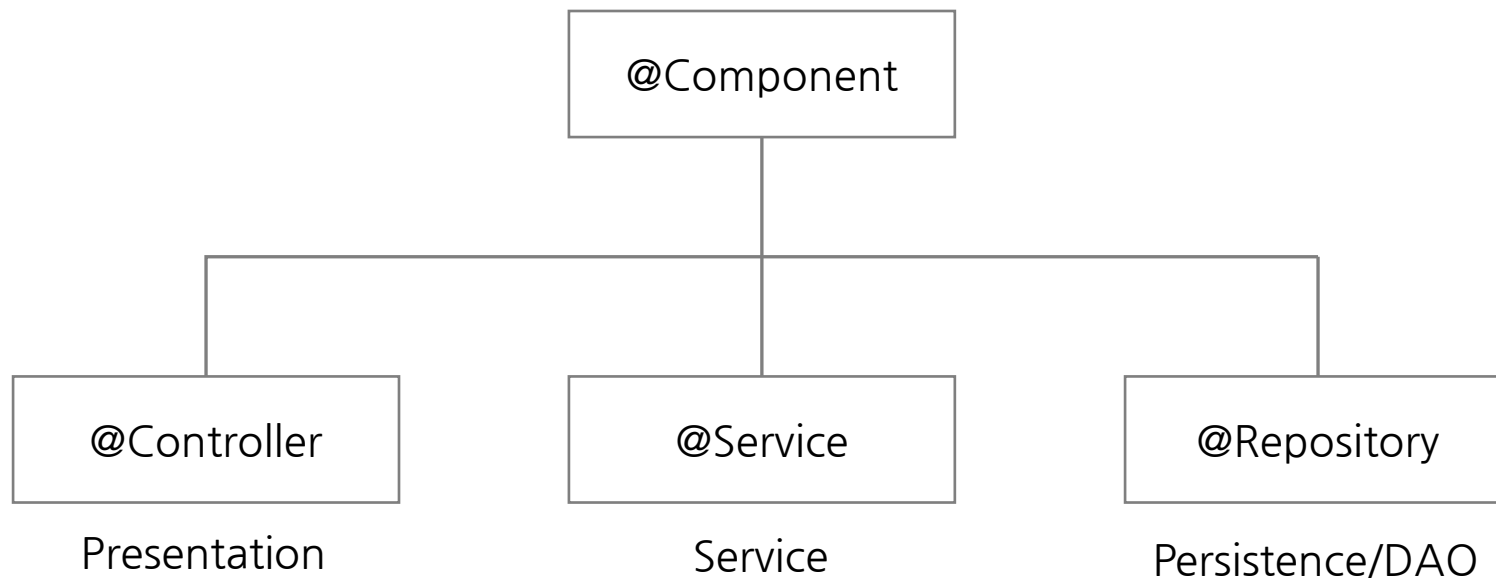
➤ xml설정파일에 <context:annotation-config></context:annotation-config> 태그를 추가한다.

파일	지원	설명
@Autowired	스프링	<p>1)의존관계를 자동으로 설정해주는 어노테이션</p> <ul style="list-style-type: none"> •Autowired 어노테이션이 붙은 인스턴스 변수는 해당 변수 타입과 일치하는 컨텍스트 내의 빈을 찾아 인스턴스 변수에 주입해 준다. • 의존성 주입을 위해선 생성자나 setter가 필요하지만 없어도 사용 가능하다. <p>2)타입을 이용하여 의존객체를 자동으로 설정한다(byType--> byName)</p> <p>3)생성자, 필드, 메서드 세 곳에 적용 가능하다</p>
@Resource	자바	<ul style="list-style-type: none"> • 이름으로 연결한다. (byName--> byType)
@Qualifier	스프링	<ul style="list-style-type: none"> • 빈 객체 중 특정빈을 사용하도록 선언 • @Autowired는 메서드 앞에서 사용가능하다, • 다만, 인젝션 가능한 형이 2개 이상이면 오류가 발생한다.이때 @Qualifier를 사용한다
@Inject	자바	<ul style="list-style-type: none"> • 타입으로 연결한다.(byType)

- **@Repository, @Component, @Controller, @Service**

- xml설정파일에 <context:component-scan base-package="패키지명" /> 태그를 추가한다.
- 스프링은 지정한 패키지에서 이 어노테이션이 적용된 클래스를 검색하여 자동으로 빈을 생성한다.
- base-package는 어느 범위에 있는 객체를 생성해 줄 것인지를 선언한다
- 선언된 패키지 내에 있는 객체들을 다 생성해준다.

파일	설명
@Controller	Presentation Layer에서 Controller를 명시하기 위해서 사용
@Service	Business Layer에서 Service를 명시하기 위해서 사용
@Repository	Persistence Layer에서 DAO를 명시하기 위해서 사용
@Component	그 외에 자동으로 스캔해서 등록하고 싶은 것들을 위해 사용, 해당 클래스를 자동으로 bean으로 등록



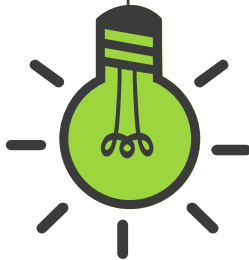
- **@Required, @Autowired, @Configuration, @Resource**
 - xml설정파일에 <context:annotation-config/> 태그를 추가한다.
 - 어플리케이션 컨텍스트안에 이미 등록된 빈들의 어노테이션을 활성화를 위해 사용한다.
 - <context:component-scan> 를 선언했다면 <context:annotation-config>를 선언할 필요가 없다.

- 빈 객체 간의 의존관계를 설정 하기 위해 <property>를 통해서 설정
- autowire=" "를 이용

파일	설명
byName	속성의 이름과 빈의 이름이 동일한 빈을 찾아서 해당 빈의 속성에 빈객체를 설정한다.
byType	속성의 타입과 빈의 타입이 동일한 빈을 찾아서 해당 빈의 속성에 빈객체를 설정한다.
constructor	생성자의 파라미터 타입과 동일한 타입의 빈을 찾아서 생성자의 파라미터에 설정한다.
autodetect	먼저 constructor 방식을 적용하고 실패한 경우에는 byType 식을 적용한다.
no (default)	연결 안함

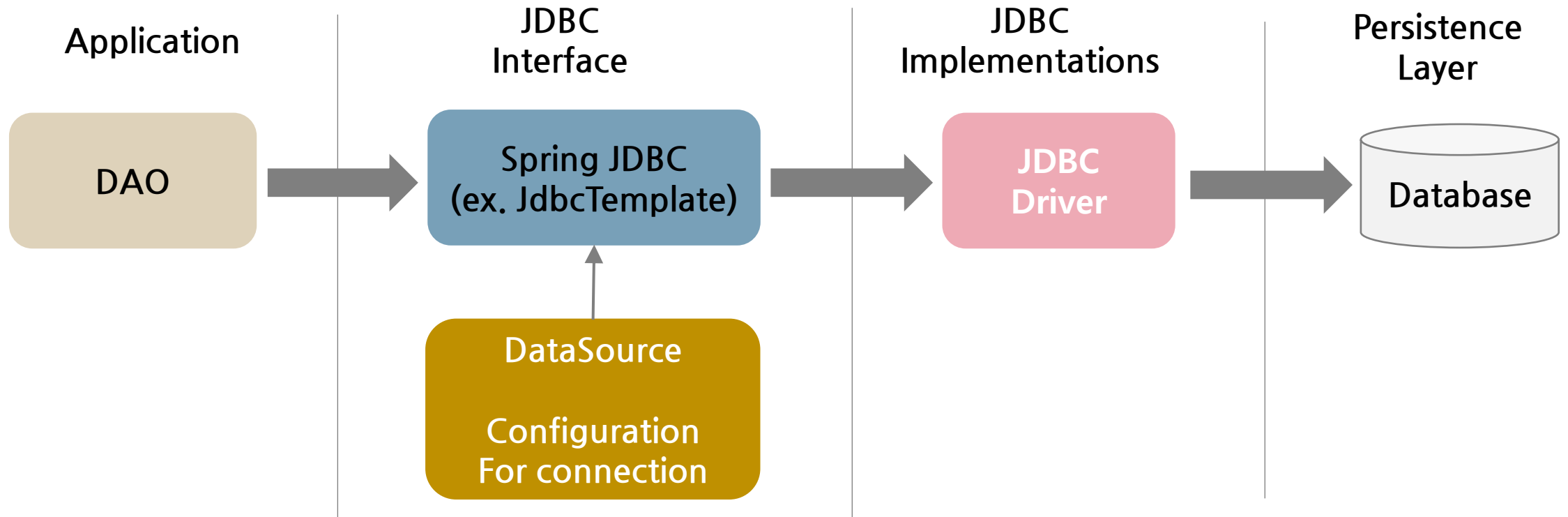
05

JDBC와 MyBatis

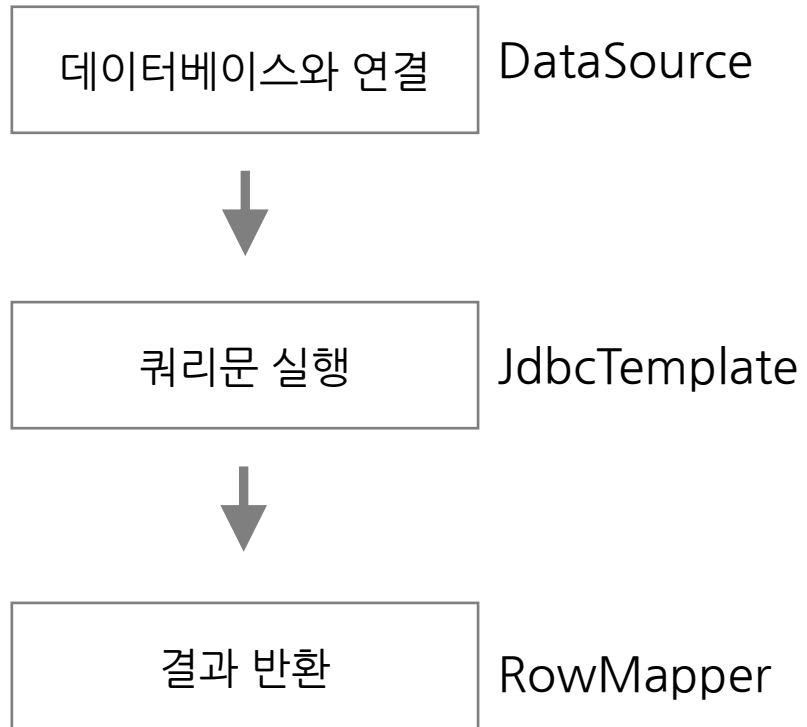


1. JDBC(Java Database Connectivity)

- JDBC는 DB에 접근할 수 있도록 Java에서 제공하는 API이다.
 - 모든 Java의 Data Access 기술의 근간
 - 즉, 모든 Persistence Framework는 내부적으로 JDBC API를 이용한다.
- JDBC는 데이터베이스에서 자료를 쿼리하거나 업데이트하는 방법을 제공한다.



❖ Spring과 JDBC 연결



❖ JDBC Template API사용

파일	설명
RowMapper	<ul style="list-style-type: none">SELECT문을 수행하는 결과값을 단순히 정수 값이나 문자열로 받지 않고 VO(Value Object)객체에 담아야 하는 경우가 있다.이 경우 VO 객체에 DB에서 SELECT에 데이터가 어떻게 매핑 되어야 하는지에 대한 정의가 필요하다이 매핑과 관련된 설정을 위해 제공되는 것이 RowMapper인터페이스 이다.
queryForInt()	<ul style="list-style-type: none">SELECT문 실행 결과로 리턴 되는 하나의 정수 값을 리턴 받기 위한 JdbcTemplate클래스 메서드 <p>ex) <code>int count = getJdbcTemplate().queryForInt("SELECT COUNT(*) FROM USERS");</code></p>
queryForObject()	<ul style="list-style-type: none">SQL수행 결과를 Object로 리턴 하는 경우에 이용된다 <p>ex) <code>String name = getJdbcTemplate().queryForObject("SELECT NAME FROM USER WHERE ID=?, new Object[]{"test"}, String.class);</code></p>

❖ JDBC Template API사용

파일	설명
query()	<ul style="list-style-type: none">SELECT문의 실행결과가 여러 목록으로 리턴 되는 경우에 사용되는 메서드이다 ex) <code>Object []args = {"%" + searchKeyword + "%"};</code> <code>ArrayList list = getJdbcTemplate().query("SELECT * FROM USERS WHERE USER_NAME LIKE ?", args, new UserRowMapper());</code>
update()	<ul style="list-style-type: none">INSERT, DELETE, UPDATE SQL문을 수행하기 위해 사용되는 메서드 이다. ex) <code>getJdbcTemplate().update("INSERT INTO USER VALUES(?,?,?)", new Object[]{ name, age, phone});</code>

2. MyBatis 개요

- MyBatis는 Java Object와 SQL문 사이의 자동 Mapping기능을 지원하는 ORM Framework이다

<https://blog.mybatis.org>

- MyBatis의 기본설정 및 사용 기술문서 참조

[MyBatis-3-User-Guide_ko.pdf](#)

❖ 특징

- ① 개발자가 지정한 SQL, 저장 프로시저 그리고 몇 가지 고급 매핑을 지원하는 SQL Mapper이다.
- ② JDBC로 처리하는 상당 부분의 코드와 파라미터 설정 및 결과 매핑을 대신해준다.
 - 기존에 JDBC를 사용할 때는 DB와 관련된 여러 복잡한 설정(Connection)들을 다루어야 했지만 SQL Mapper는 자바 객체를 실제 SQL문에 연결함으로써, 빠른 개발과 편리한 테스트 환경을 제공한다.
- ③ 데이터베이스 record에 원시 타입과 Map 인터페이스 그리고 자바 POJO를 설정해서 매핑하기 위해 xml과 Annotation을 사용할 수 있다.

❖ 특징

- ④ MyBatis는 원래 Apache Foundation의 iBatis였으나 생산성, 개발 프로세스, 커뮤니티 등의 이유로 Google Code로 이전되면서 이름이 바뀌었다.
- ⑤ MyBatis는 Hibernate나 JPA(Java Persistence API)처럼 새로운 DB 프로그래밍 패러다임을 익혀야 하는 부담이 없이, 개발자가 익숙한 SQL을 그대로 이용하면서 JDBC 코드 작성의 불편함을 제거해 주고, 도메인 객체나 VO 객체를 중심으로 개발이 가능하다.

❖ 장점

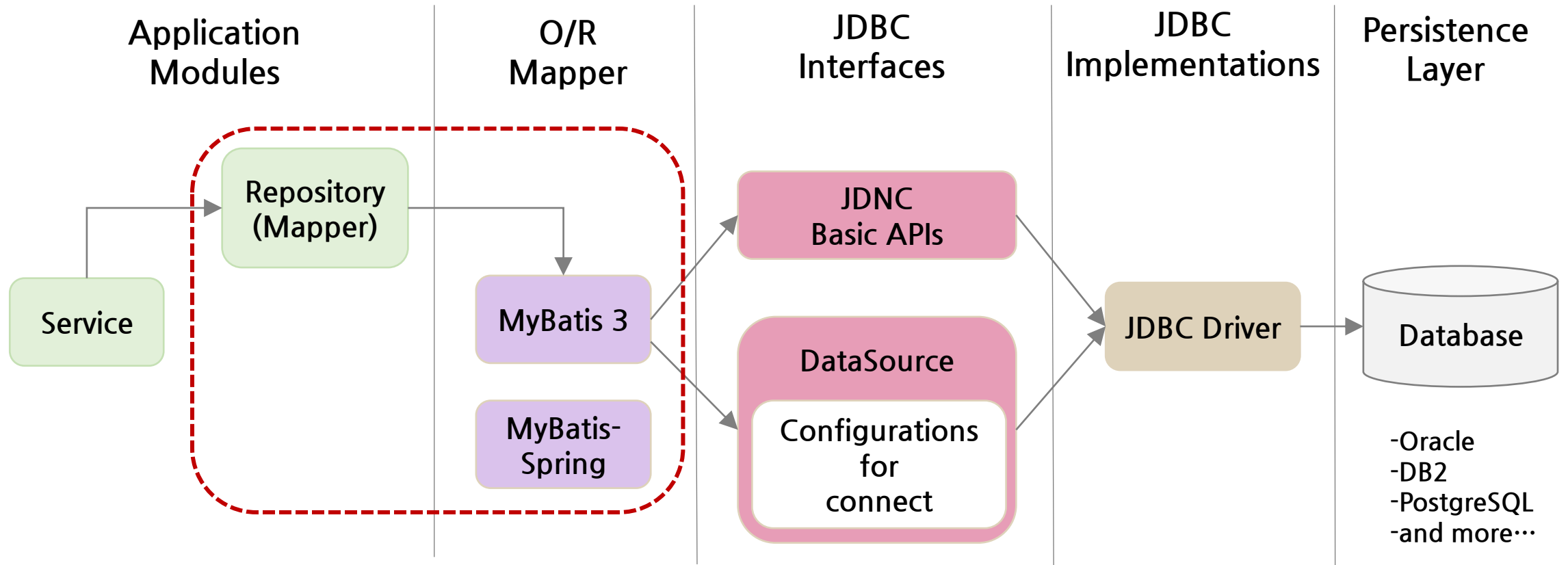
- SQL에 대한 모든 컨트롤을 하고자 할 때 매우 적합하다.
- SQL쿼리들이 매우 잘 최적화되어 있을 때에 유용하다.

❖ 단점

- 애플리케이션과 데이터베이스 간의 설계에 대한 모든 조작을 하고자 할 때는 적합하지 않다.
- 애플리케이션과 데이터베이스 간에 서로 잘 구조화되도록 많은 설정이 바뀌어야 하기 때문이다.

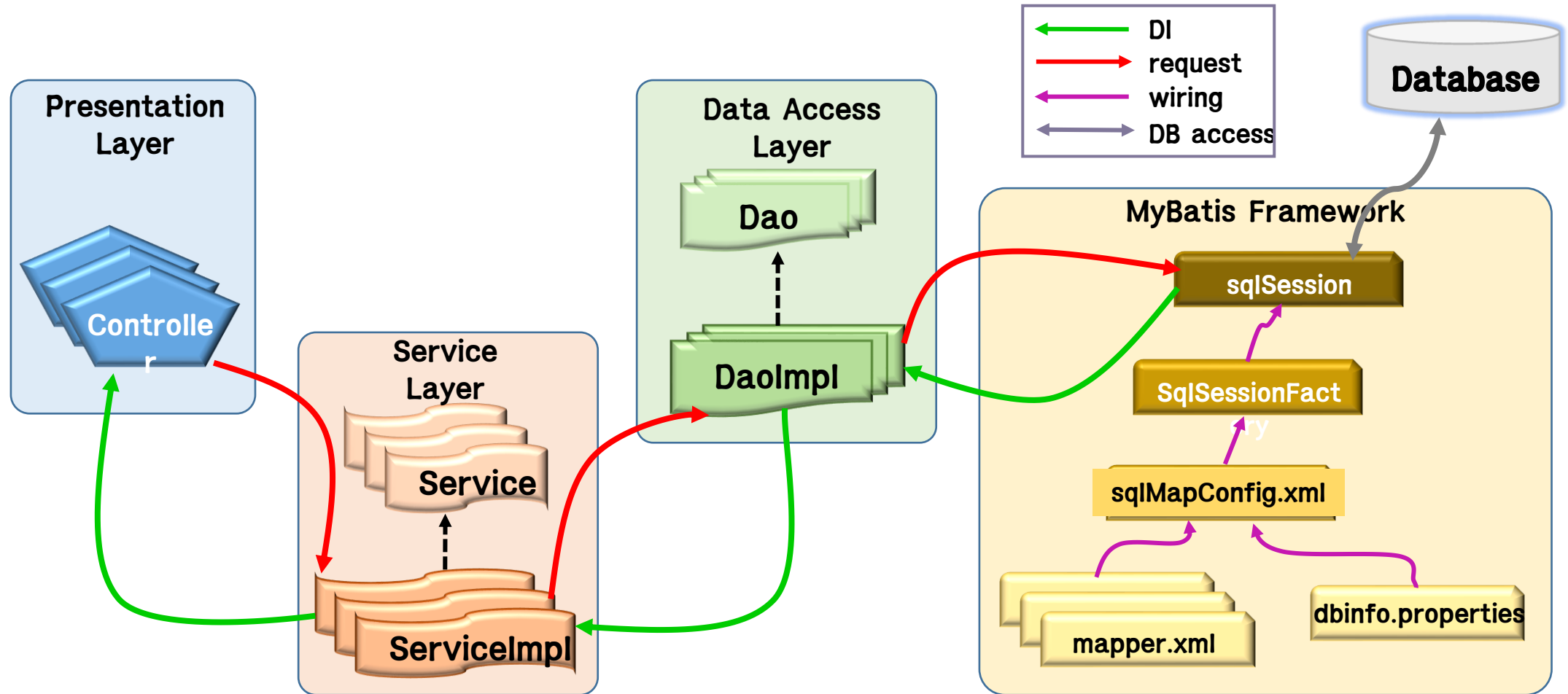
❖ MyBatis와 MyBatis-Spring 주요 Component

- MyBatis와 MyBatis-Spring을 사용한 DB Access Architecture



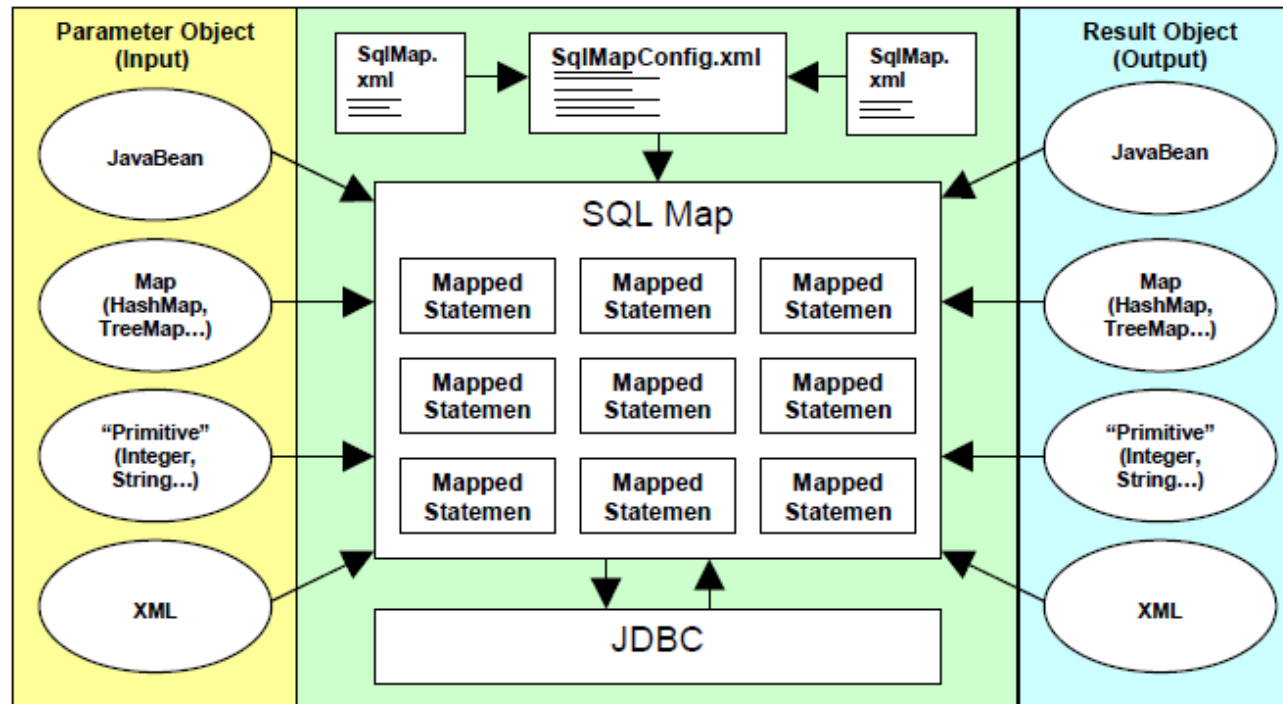
❖ MyBatis와 MyBatis-Spring 주요 Component

- MyBatis를 사용하는 Data Access Layer



3. SQL Mapper 와 ORM(Object-Relational Mapping)

- Persistence Framework는 SQL Mapper와 ORM으로 나눌 수 있다.
 - ORM은 데이터베이스 객체를 자바 객체로 매핑함으로써 객체 간의 관계를 바탕으로 SQL을 자동으로 생성해주지만 SQL Mapper는 SQL을 명시해줘야 한다.
 - ORM은 관계형 데이터베이스의 '관계'를 Object에 반영하자는 것이 목적이라면, SQL Mapper는 단순히 필드를 매핑 시키는 것이 목적이라는 점에서 지향점의 차이가 있다.



SQL Mapper

- SQL <—매핑—> Object 필드
- SQL Mapper는 SQL 문장으로 직접 데이터베이스 데이터를 다룬다.
 - 즉, SQL Mapper는 SQL을 명시해줘야 한다.
 - 예) MyBatis, JdbcTemplate 등

ORM, 객체-관계 매핑

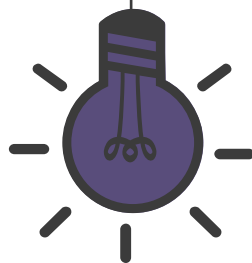
- 데이터베이스 데이터 <—매핑—> Object
 - 필드객체를 통해 간접적으로 데이터베이스 데이터를 다룬다.
- 객체와 관계형 데이터베이스의 데이터를 자동으로 매핑(연결)해주는 것을 말한다.
 - ORM을 이용하면 SQL Query가 아닌 직관적인 코드(메서드)로 데이터를 조작할 수 있다.
 - 객체 간의 관계를 바탕으로 SQL을 자동으로 생성한다.
- Persistence API라고도 할 수 있다.
 - 예) JPA, Hibernate 등

❖ MyBatis-Spring 주요 Component의 역할

파일	설명
MyBatis 설정파일 (sqlMapConfig.xml)	데이터베이스의 접속 주소 정보나 객체의 alias Mapping 파일의 경로 등의 고정된 환경 정보를 설정.
SqlSessionFactoryBean	MyBatis 설정 파일을 바탕으로 SqlSessionFactory를 생성. Spring Bean으로 등록 해야 함
SqlSessionTemplate	핵심적인 역할을 하는 클래스로서 SQL 실행이나 Transaction 관리를 실행 SqlSession interface를 구현하며, Thread-safe하다. Spring Bean으로 등록해야 함
mapping 파일 (member.xml)	SQL 문과 ORMapping을 설정
Spring Bean 설정파일 (beans.xml)	SqlSessionFactoryBean을 Bean에 등록할 때 DataSource 정보와 MyBatis Config 파일 정보, Mapping 파일의 정보를 함께 설정함. SqlSessionTemplate을 Bean으로 등록.

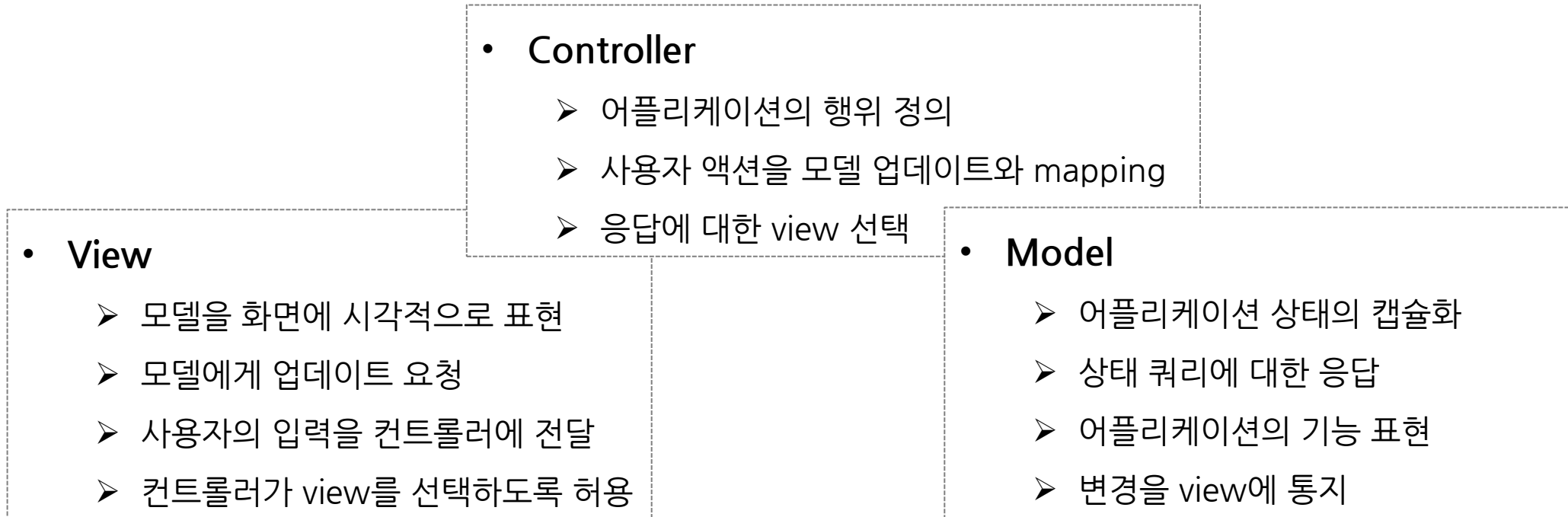
06

Spring MVC



1. MVC(Model-View-Controller) Pattern 개념

- 스프링 프레임워크에서 지원하는 Spring MVC는 모델-뷰-컨트롤러 (MVC) 구현을 포함한다.
- 도메인 모델코드와 웹 폼을 깔끔하게 분리할 수 있도록 하고 스프링 프레임워크의 다른 모든 기능과 통합할 수 있게 한다.
- DI 와 선언적인 방식으로 MVC 기반의 웹 프로그램 개발을 효율적으로 할 수 있도록 지원한다.



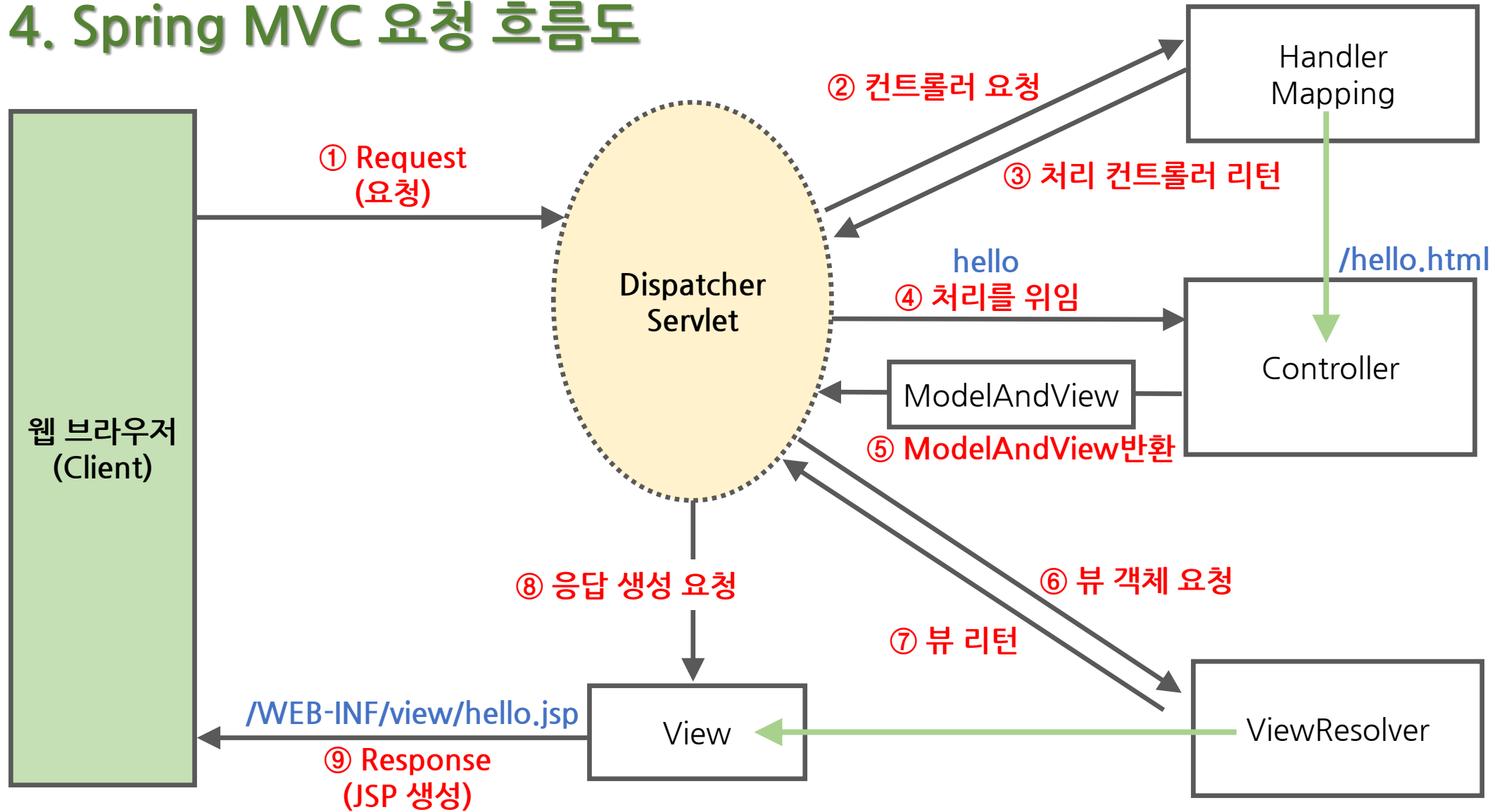
2. Spring MVC의 특징

- 어플리케이션의 확장을 위해 Model, View, Controller 세가지 영역으로 분리한다.
- 컴포넌트의 변경이 다른 영역 컴포넌트에 영향을 미치지 않는다. (유지보수 용이)
- 컴포넌트 간의 결합성이 낮아 프로그램 수정이 용이하다.(확장성이 뛰어남)
- **장점**
 - 화면과 비즈니스 로직을 분리해서 작업 가능하다.
 - 영역별 개발로 인하여 확장성이 뛰어나다.
 - 표준화된 코드를 사용하므로 공동작업이 용이하고 유지보수성이 좋다.
- **단점**
 - 개발과정이 복잡해 초기 개발속도가 늦다.
 - 초보자가 이해하고 개발하기에 다소 어렵다.

3. Spring MVC의 구성요소

파일	설명
DispatcherServlet (Front Controller)	<ul style="list-style-type: none">• 모든 클라이언트의 요청을 전달받는다.• Controller에게 클라이언트의 요청을 전달한다.• Controller가 리턴 한 결과값을 View에게 전달하여 알맞은 응답을 생성한다
HandlerMapping	<ul style="list-style-type: none">• 클라이언트의 요청 URL을 어떤 Controller가 처리할지를 결정한다.• URL과 요청 정보를 기준으로 어떤 핸들러 객체를 사용할지 결정하는 객체이다.• DispatcherServlet은 하나 이상의 핸들러 매핑을 가질 수 있다.
Controller	<ul style="list-style-type: none">• 클라이언트의 요청을 처리한 뒤, Model을 호출하고 그 결과를 DispatcherServlet에 알려준다.
ModelAndView	<ul style="list-style-type: none">• Controller가 처리한 데이터 및 화면에 대한 정보를 보유한 객체이다.
ViewResolver	<ul style="list-style-type: none">• Controller가 리턴 한 뷰 이름을 기반으로 Controller의 처리 결과를 보여줄 View를 결정한다.
View	<ul style="list-style-type: none">• Controller의 처리결과를 보여줄 응답화면을 생성한다.

4. Spring MVC 요청 흐름도

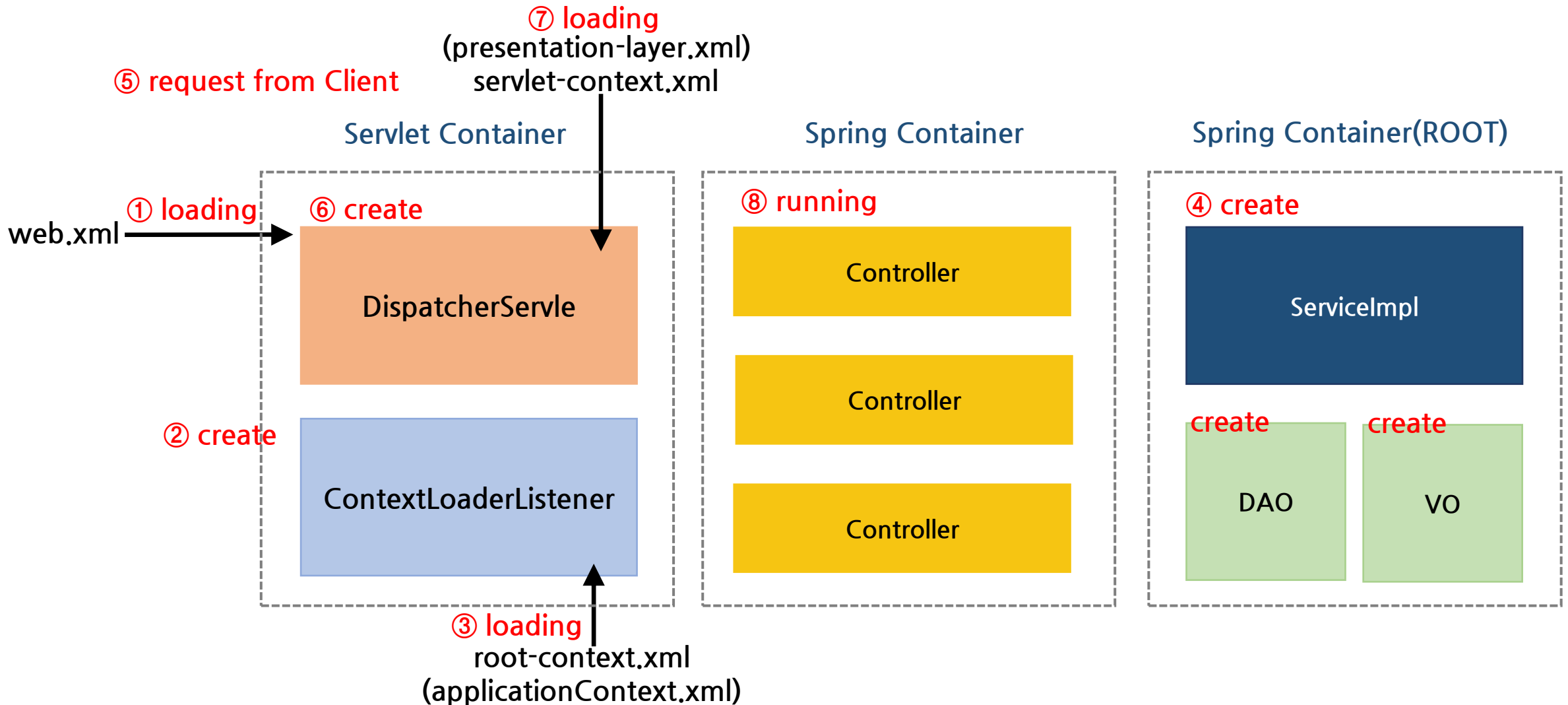


❖ Spring MVC 실행 순서

- ① DispatcherServlet이 요청을 수신한다.
 - 단일 Front Controller Servlet.
 - 요청을 수신하여 처리를 다른 컴포넌트에 위임한다.
 - 어느 Controller에 요청을 전송할지 결정한다.
- ② DispatcherServlet은 Handler Mapping에 어느 Controller를 사용할 것인지 문의한다.
 - URL과 Mapping.
- ③ DispatcherServlet은 요청을 Controller에게 전송하고 Controller는 요청을 처리한 후 결과 리턴 한다.
 - Business Logic 수행 후 결과 정보(Model)가 생성되어 JSP와 같은 view에서 사용된다.
- ④ ModelAndView Object에 수행결과가 포함되어 DispatcherServlet에 리턴 한다.
- ⑤ ModelAndView는 실제 JSP정보를 갖고 있지 않다.

ViewResolver가 논리적 이름을 실제 JSP이름으로 변환한다.
- ⑥ View는 결과정보를 사용하여 화면을 표현한다.

4. Web Application 동작원리

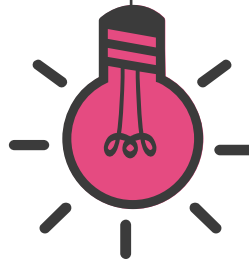


❖ Web Application 실행 순서

1. 웹 어플리케이션이 실행되면 Tomcat(WAS)에 의해 web.xml이 loading.
2. web.xml에 등록되어 있는 ContextLoaderListener (Java Class)가 생성.
ContextLoaderListener class는 ServletContextListener interface 를 구현하고 있으며, ApplicationContext를 생성하는 역할을 수행.
3. 생성된 ContextLoaderListener는 root-context.xml을 loading.
4. root-context.xml에 등록되어 있는 Spring Container가 구동.
이 때 개발자가 작성한 Business Logic(Service)에 대한 부분과 Database Logic(DAO), VO 객체들이 생성.
5. Client로 부터 요청(request)가 들어옴.
6. DispatcherServlet(Servlet)이 생성. DispatcherServlet은 FrontController의 역할을 수행.
Client로부터 요청 온 메시지를 분석하여 알맞은 PageController에게 전달하고 응답을 받아 요청에 따른 응답을 어떻게 할 지 결정. 실질적인 작업은 PageController에서 이루어 진다.
이러한 클래스들을 HandlerMapping, ViewResolver Class라고 함.
7. DispatcherServlet은 servlet-context.xml을 loading.
8. 두번째 Spring Container가 구동되며 응답에 맞는 PageController들이 동작.
이 때 첫번째 Spring Container가 구동되면서 생성된 DAO, VO, ServiceImpl 클래스들과 협업하여 알맞은 작업을 처리.

07

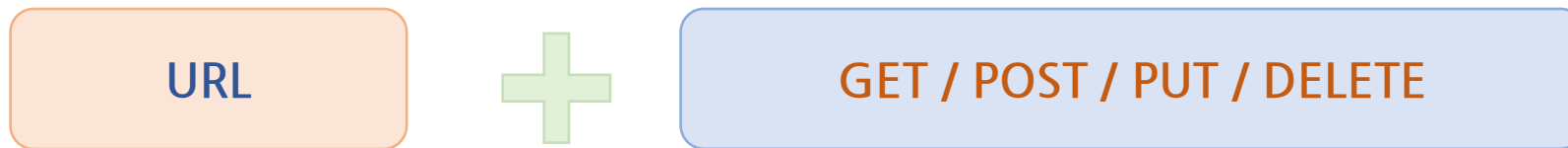
Rest API



1. Rest(Representational State Transfer) API

- **Open API (Application Programming Interface)**는 프로그래밍에서 사용할 수 있는 개방되어 있는 상태의 Interface이다.
- 2000년에 처음 로이 필딩(Roy Fielding)의 박사학위 논문에 소개되었다.
- REST는 'Representational State Transfer'의 약어로 하나의 URI는 하나의 고유한 리소스(Resource)를 대표하도록 설계된다는 개념에 전송방식을 결합해서 원하는 작업을 지정한다.
- 웹의 장점을 최대한 활용할 수 있는 아키텍처(설계구조)로써 REST를 발표하였다.

❖ Rest 특징



- HTTP URI를 통해 제어할 자원(Resource)를 명시하고, HTTP Method(GET, POST, PUT, DELETE)을 통해 해당자원(Resource)를 제어하는 명령을 내리는 방식의 아키텍처 이다.

❖ Rest 구성

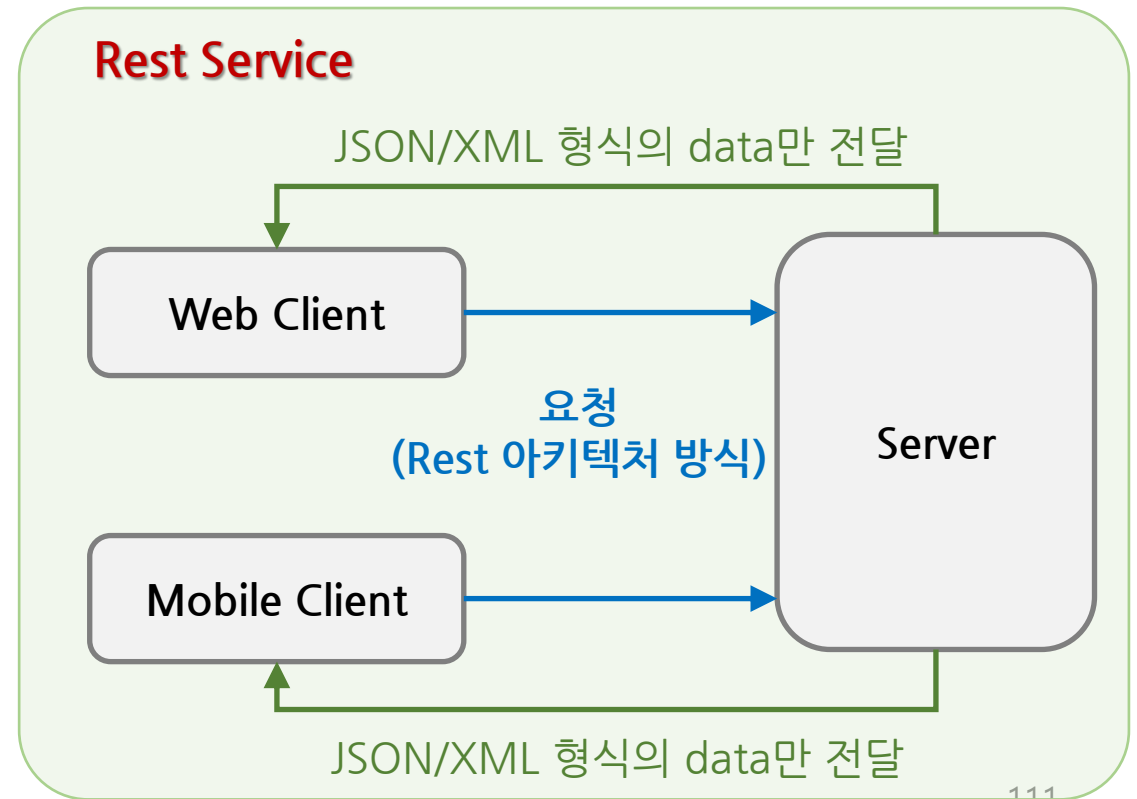
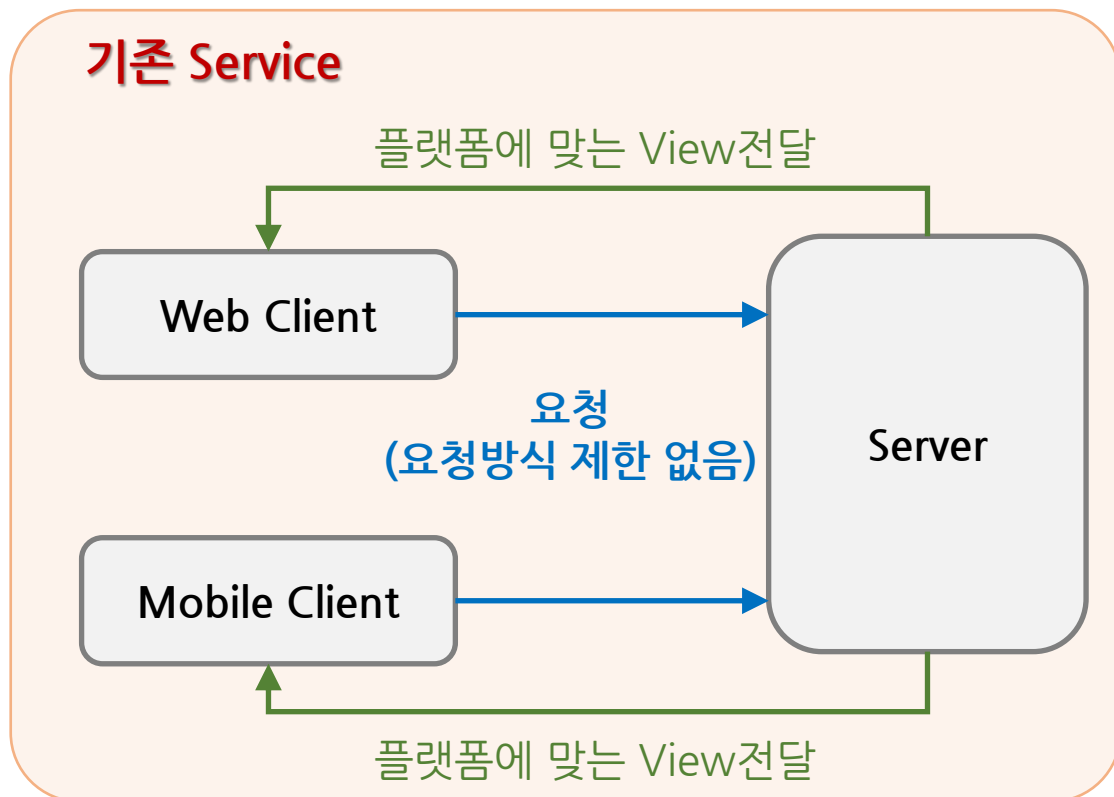
- 자원(Resource) - URI
- 행위(Verb) - HTTP Method
- 표현(Representations)

- 잘 표현된 HTTP URI로 리소스를 정의하고 HTTP method로 리소스에 대한 행위를 정의한다.
- 리소스는 JSON, XML과 같은 여러 가지 언어로 표현할 수 있다.

- Naver, kakao 등 포털 서비스 사이트나 통계청, 기상청, 우체국 등과 같은 관공서, 공공 데이터 포털
- (<https://www.data.go.kr>)이 가지고 있는 데이터를 외부 응용 프로그램에서 사용할 수 있도록 OPEN API를 제공하고 있다.
- Open API와 함께 거론되는 기술이 Rest이며, 대부분의 Open API는 Rest방식으로 지원된다

❖ Rest 구조

- 기존 Service : 요청에 대한 처리를 한 후 가공된 data를 이용하여 특정 플랫폼에 적합한 형태의 View로 만들어서 반환한다.
- REST Service : data 처리만 한다 거나, 처리 후 반환될 data가 있다면 JSON이나 XML 형식으로 전달한다.
- View에 대해서는 신경 쓸 필요가 없기 때문에 Open API 에서 많이 사용한다.



2. Rest API 작업 방식

- 기존의 블로그 등은 GET과 POST만으로 자원에 대한 CRUD를 처리하며, URI는 액션을 나타냈다.
- REST로 변경할 경우 4가지 method를 모두 사용하여 CRUD를 처리하며, URI는 제어하려는 자원을 나타낸다.

❖ Rest 작성규칙

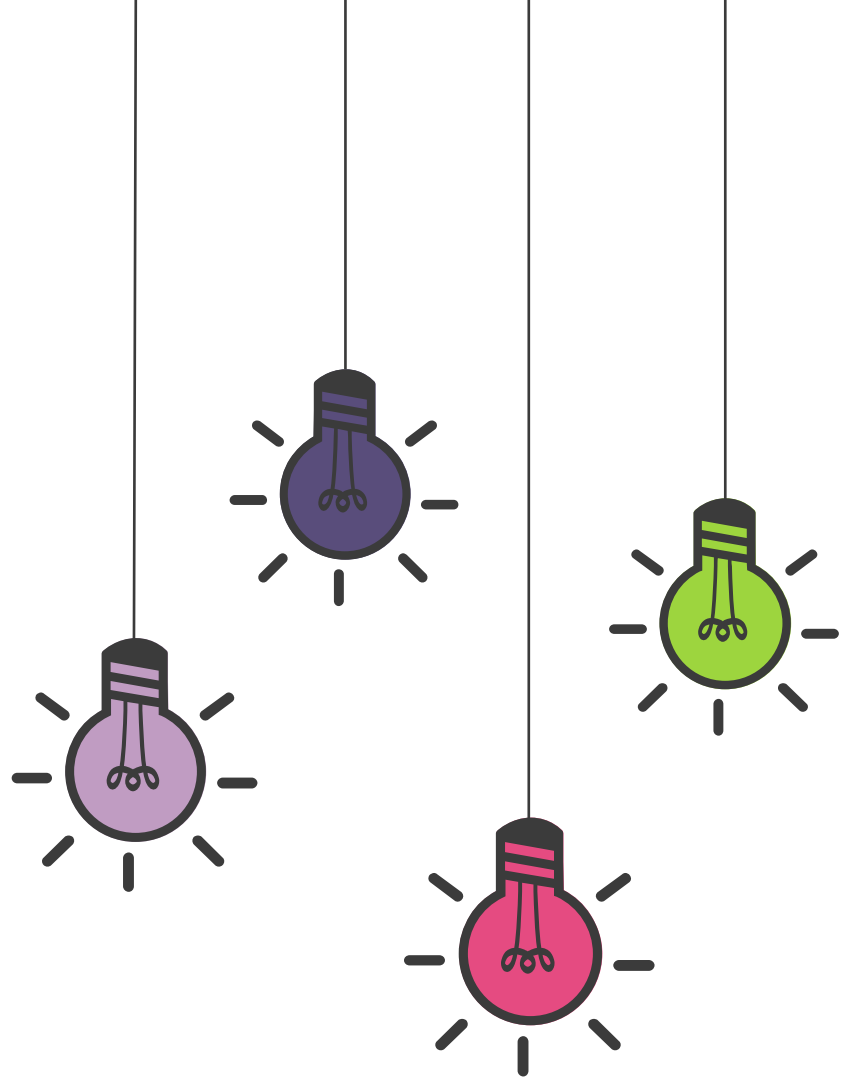
작업		기존 방식		Rest 방식	내용
Create(Insert)	POST	/write.do?id=abcd	POST	/blog/abcd	글쓰기
Read(Select)	GET	/view.do?id=abcd&articleno=1	GET	/blog/abcd/1	글읽기
Update(Update)	POST	/modify.do?id=abcd	PUT	/blog/abcd	글수정
Delete>Delete)	GET	/delete.do?id=qbcd&articleno=10	DELETE	/blog/abcd/10	글삭제

❖ Rest 방식의 작성 주의사항

- ✓ 기존의 전송방식과는 달리 서버는 요청으로 받은 리소스에 대해 순수한 데이터를 전송한다.
- ✓ 기존의 GET/POST 외에 PUT, DELETE 방식을 사용하여 리소스에 대한 CRUD 처리를 할 수 있다.
- ✓ HTTP URI을 통해 제어할 자원(Resource)을 명시하고, HTTP METHOD(GET/POST/PUT/DELETE)를 통해 해당 자원(Resource)를 제어하는 명령을 내리는 방식의 Architecture이다.
- ✓ 자원을 표현할 때 Collection(문서, 객체의 집합)과 Document(하나의 문서, 객체) 사용한다.
 - ➔ <http://www.javavc.com/sports/baseball/players/31>
- ✓ 가장 큰 단점은 딱 정해진 표준이 없어 ‘다들 이렇게 쓰더라~~’ 정도의 암묵적인 표준만 정해져 있다.
 - 하이픈(-)은 사용 가능하지만 언더바(_)는 사용하지 않는다. (가독성)
 - 특별한 경우를 제외하고 대문자 사용은 하지 않는다.(대소문자 구분을 하기 때문, RFC3986 규정.)
 - URI 마지막에 슬래시(/)를 사용하지 않는다.
 - 슬래시(/)로 계층 관계를 나타낸다.
 - 확장자가 포함된 파일 이름을 직접 포함시키지 않는다.
 - URI는 명사를 사용한다.

❖ Rest 관련 Annotation

Annotation	설 명
@RestController	Controller 가 REST 방식을 처리하기 위한 것임을 명시
@ResponseBody	SP 같은 뷰로 전달되는 것이 아니라 데이터 자체를 전달
@PathVariable	URL 경로에 있는 값을 파라미터로 추출
@CrossOrigin	Ajax의 크로스 도메인 문제를 해결
@RequestBoby	JSON 데이터를 원하는 타입으로 바인딩



감사합니다

THANK YOU FOR WATCHING