



Spring
Boot

12

Lombok 및 JPA로 DataBase 사용하기

2

롬복(Lombok) 사용하기

롬복 사용하기

3

□ 롬복(Lombok)

- ▣ 자바 클래스를 만들 때 항상 만들게 되는 **Getter, Setter** 메서드 등을 어노테이션을 이용해서 **자동으로 만들어주는 유틸리티 라이브러리**

□ 롬복을 사용하기 위해

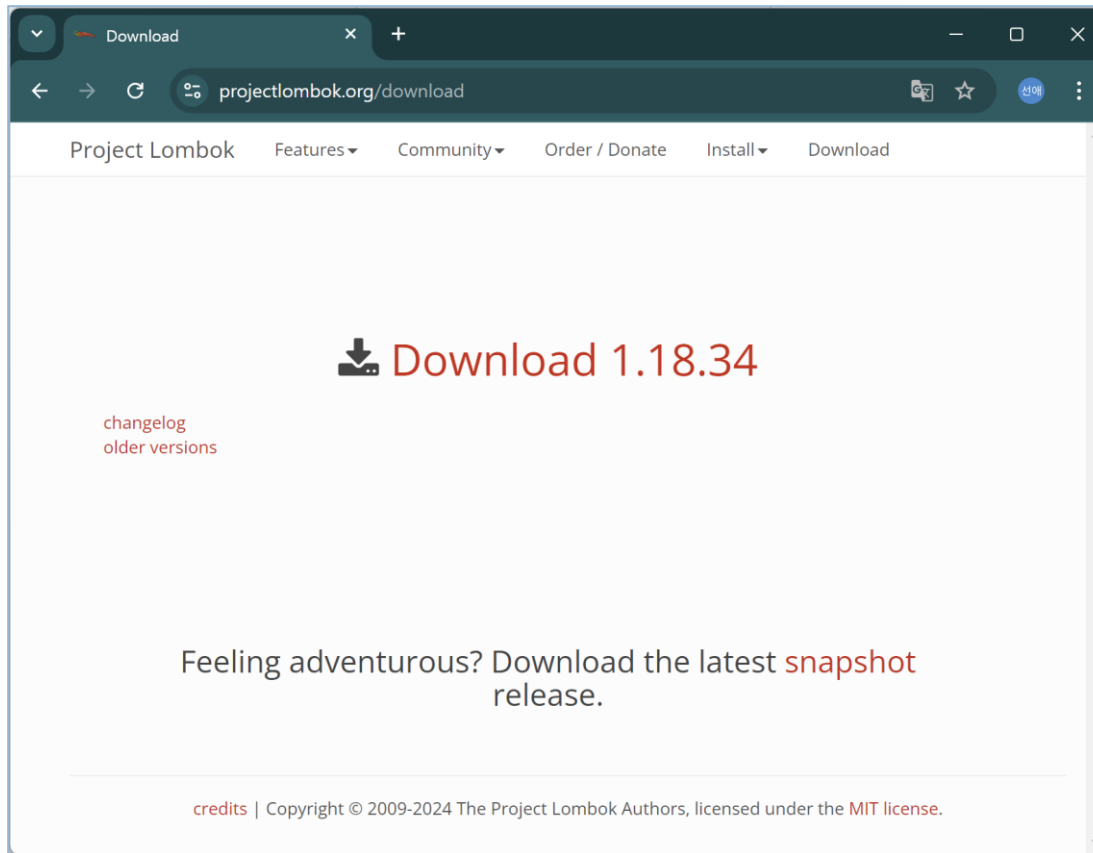
- ▣ **롬복 플러그인 설치**
- ▣ 프로젝트 생성 시 **디펜던시 추가**

롬복 사용하기

4

□ 롬복 다운로드

▣ <https://projectlombok.org/download>

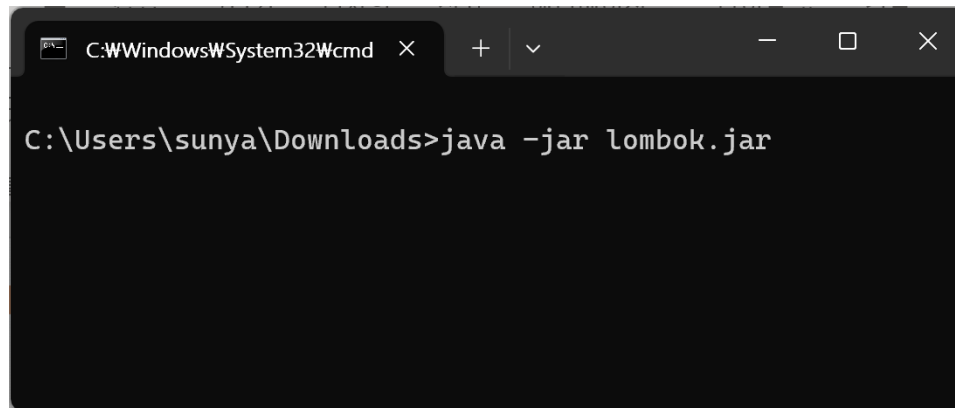


롬복 사용하기

5

□ 롬복 설치

- ▣ 명령 프롬프트 창 > **java -jar lombok.jar**



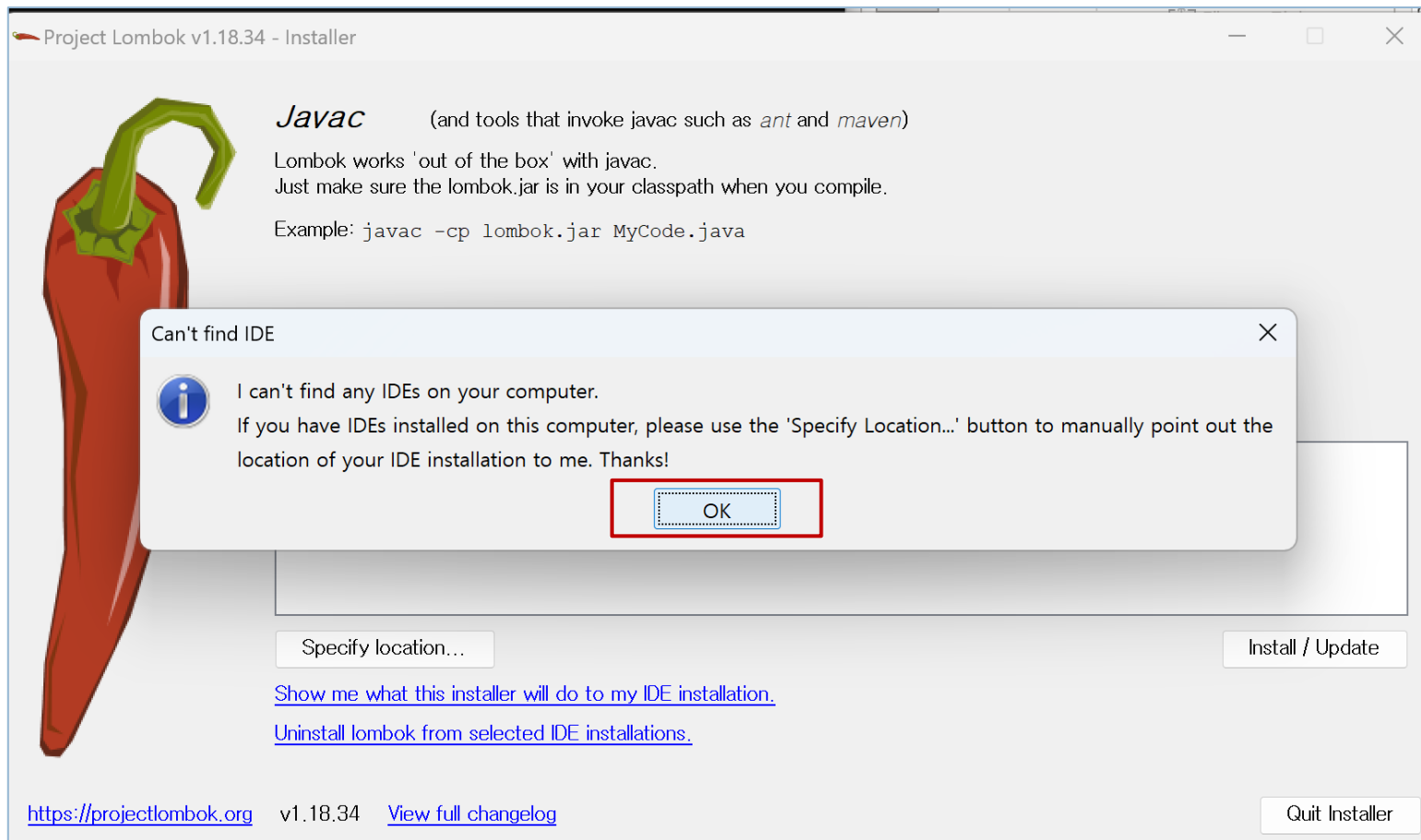
```
C:\Windows\System32\cmd.exe
C:\Users\sunya\Downloads>java -jar lombok.jar
```

롬복 사용하기

6

□ 롬복 설치

▣ 명령 프롬프트 창 > `java -jar lombok.jar`

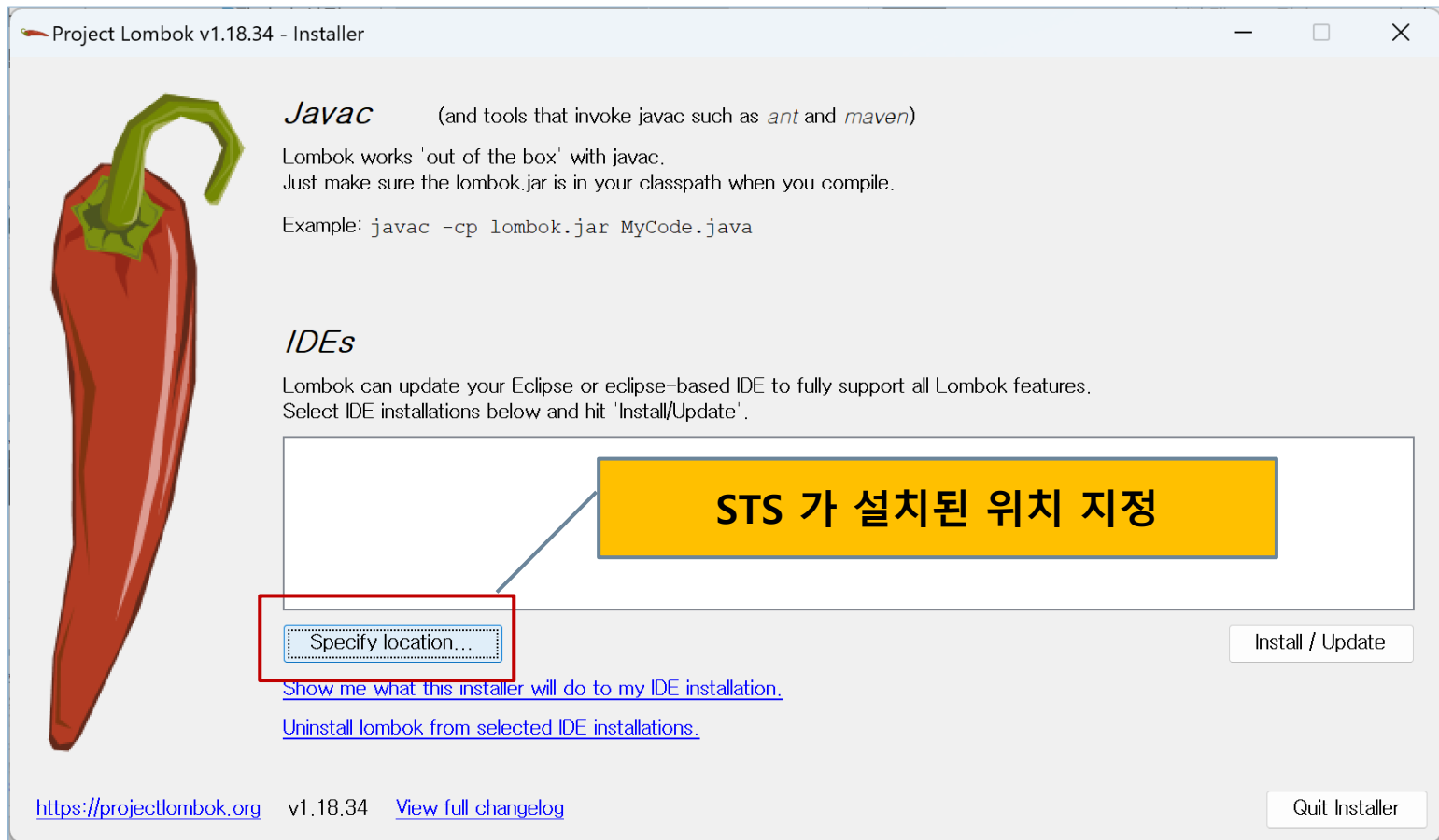


롬복 사용하기

7

□ 롬복 설치

- ▣ 명령 프롬프트 창 > `java -jar lombok.jar`

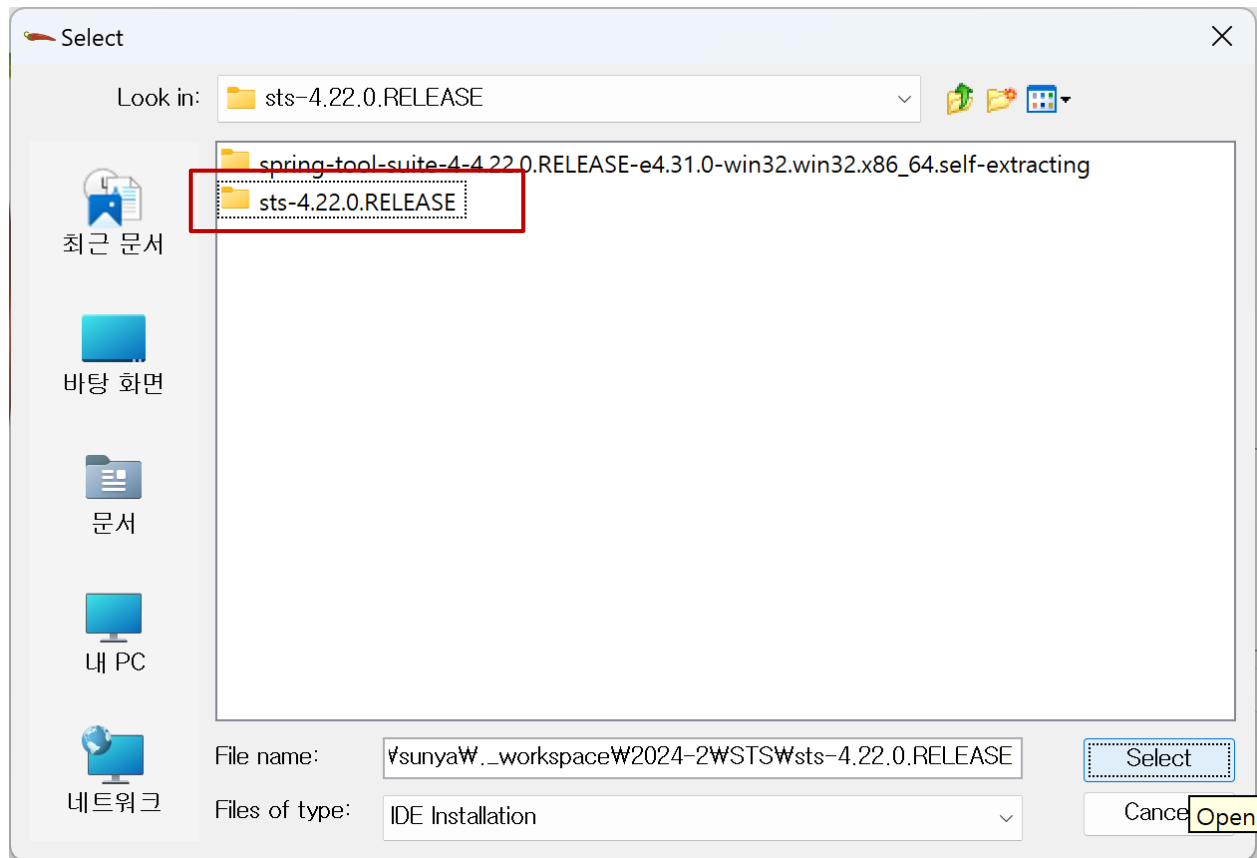


롬복 사용하기

8

□ 롬복 설치

- ▣ 명령 프롬프트 창 > `java -jar lombok.jar`

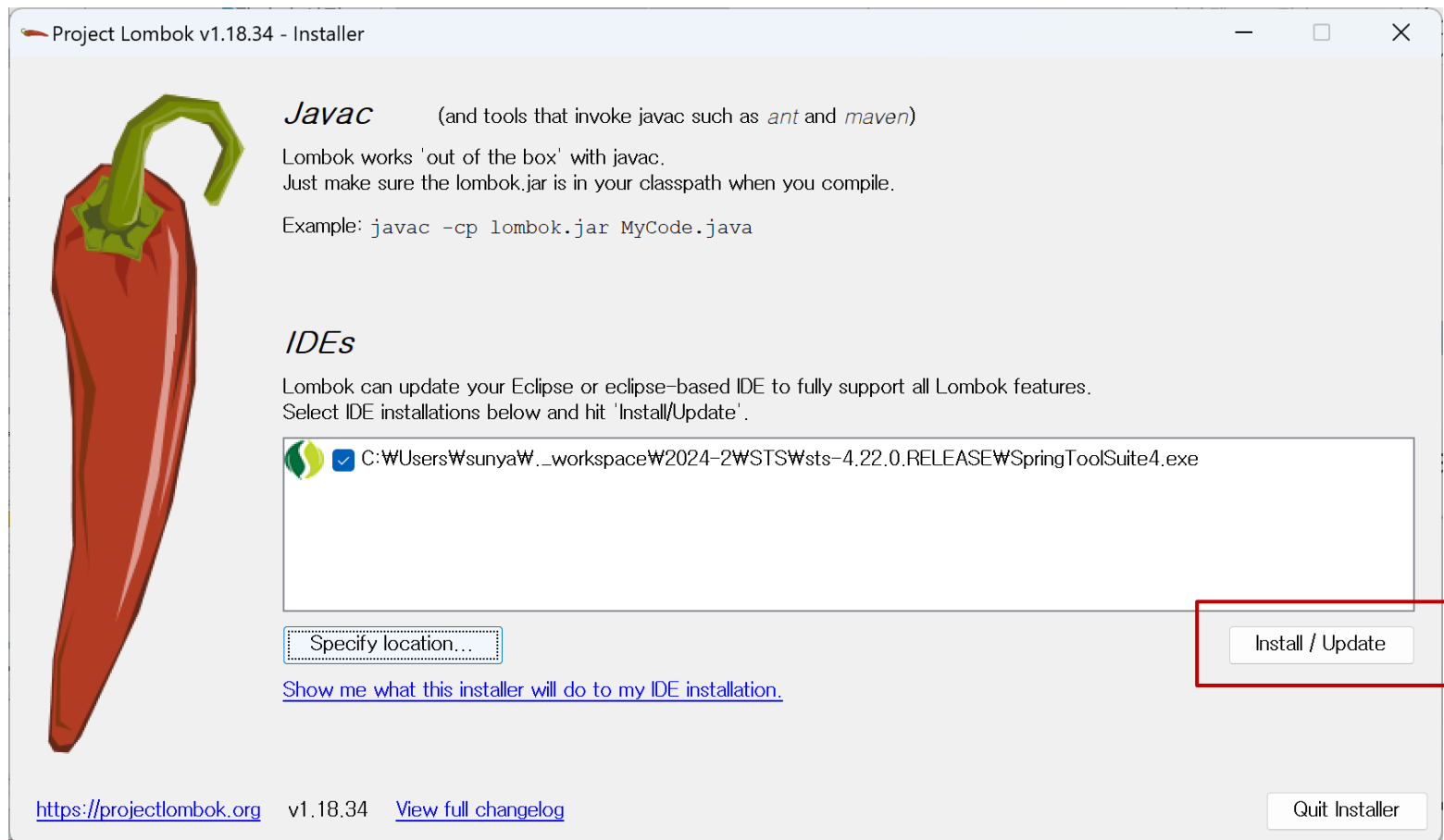


롬복 사용하기

9

□ 롬복 설치

- ▣ 명령 프롬프트 창 > `java -jar lombok.jar`

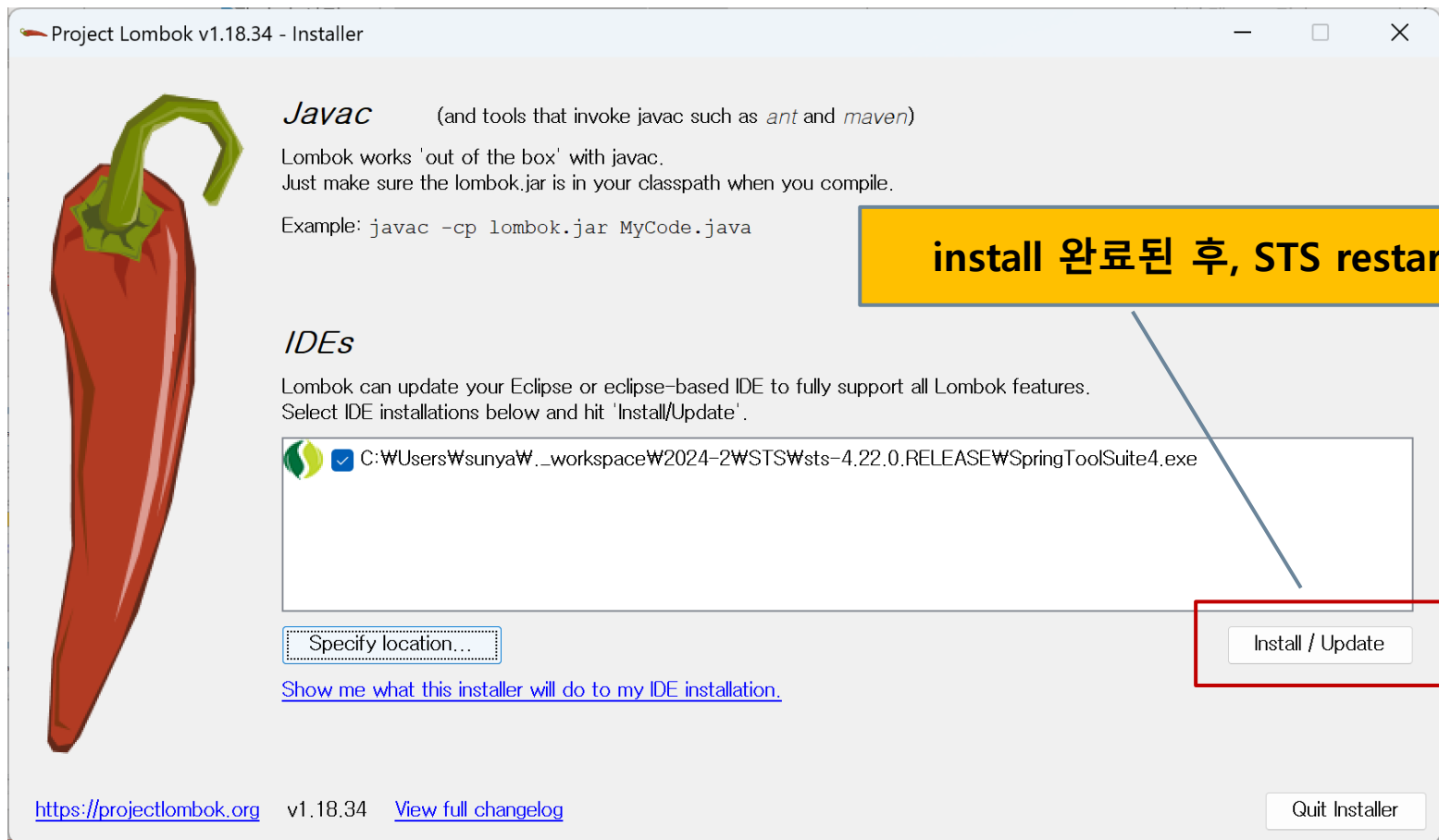


롬복 사용하기

10

□ 롬복 설치

- ▣ 명령 프롬프트 창 > `java -jar lombok.jar`



롬복 사용하기

11

- 롬복 사용 프로젝트 생성(**week12_01Lombok**)
 - ▣ 디펜던시 설정
 - SpringWeb
 - **Lombok** (implementation 'org.projectlombok:lombok')
 - ▣ build.gradle
 - JSP 설정 / 버전 수정
 - Refresh Gradle Project
 - ▣ application.properties
 - 포트 설정
 - JSP 파일 경로 설정
 - ▣ File > Properties
 - Project Facets > Dynamic Web Module 버전 > 5.0으로 수정
 - ▣ JSP 사용을 위한 폴더 구성
 - src/main/webapp/WEB-INF/views

로ombok 사용하기

12

- 커맨드(Command) 객체 만들기
 - ▣ Member 클래스

The screenshot displays an IDE window with two panes. The left pane shows the source code of `Member.java`, and the right pane shows the `Outline` of the project.

Member.java Source Code:

```
1 package com.study.springboot;
2
3 import lombok.Data;
4
5 @Data
6 public class Member {
7     private String id;
8     private String name;
9 }
10
11
12
13
14
15
```

Outline:

- com.study.springboot
 - Member
 - getId() : String
 - getName() : String
 - setId(String) : void
 - setName(String) : void
 - equals(Object) : boolean
 - canEqual(Object) : boolean
 - hashCode() : int
 - toString() : String
 - Member()
 - id : String
 - name : String

롬복 사용하기

13

□ 리퀘스트 매핑

▣ MyController

```
MyController.java ×
1 package com.study.springboot;
2
3 import org.springframework.stereotype.Controller;
4 import org.springframework.web.bind.annotation.GetMapping;
5 import org.springframework.web.bind.annotation.RequestMapping;
6 import org.springframework.web.bind.annotation.ResponseBody;
7
8 @Controller
9 public class MyController {
10     @RequestMapping("/")
11     public @ResponseBody String root() {
12         return "롬복 사용하기";
13     }
14
15     @GetMapping("/test3")
16     public String test3(Member member) {
17         // 파라미터와 일치하는 빈을 만들어서 사용할 수 있다.
18         // View 페이지에서 model을 사용하지 않고 member를 사용한다.
19         return "test3";
20     }
21 }
```

로복 사용하기

14

- 리퀘스트 매핑
 - ▣ test3.jsp

```
test3.jsp ×
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2      pageEncoding="UTF-8"%>
3  <!DOCTYPE html>
4  <html>
5  <head>
6      <meta charset="UTF-8">
7      <title>Insert title here</title>
8  </head>
9  <body>
10 <%
11     out.println("<h1>#03 : Hello World </h1>");
12 %>
13 <br>
14 <h3>
15     당신의 아이디는 ${member.id }입니다. <br>
16     당신의 이름은 ${member.name }입니다.
17 </h3>
18 </body>
19 </html>
```

15

JPA로 데이터베이스 사용하기

JPA로 데이터베이스 사용하기

16

□ ORM과 JPA 이해하기

▣ ORM(Object Relational Mapping)

- Java 문법으로도 데이터베이스를 다룰 수 있는 도구
- SQL을 직접 작성하지 않아도 데이터베이스의 데이터 처리가 가능

▣ JPA(Java Persistence API)

- 스프링 부트는 JPA를 사용하여 데이터베이스를 관리
- 스프링 부트는 JPA를 **ORM의 기술 표준**으로 사용

JPA로 데이터베이스 사용하기

17

□ ORM과 JPA 이해하기

▣ ORM(Object Relational Mapping)

- Java 문법으로도 데이터베이스를 다룰 수 있는 도구
- SQL을 직접 작성하지 않아도 데이터베이스의 데이터 처리가 가능

▣ ORM의 장점

- MySQL, Oracle DB, MS SQL 과 같은 **DBMS의 종류에 관계없이** 일관된 **자바 코드 사용이 가능**
- **프로그램 유지, 보수, 관리가 편리**
- **코드 내부에서 안정적인 SQL 쿼리문을 생성**
- 개발자가 달라져도 통일된 쿼리문 작성이 가능
- 오류 발생률도 감소

JPA로 데이터베이스 사용하기

18

□ ORM과 JPA 이해하기

▣ JPA(Java Persistence API)

- 스프링 부트는 JPA를 사용하여 데이터베이스를 관리
- 스프링 부트는 JPA를 ORM의 기술 표준으로 사용

▣ JPA 특징

- JPA는 인터페이스 모음으로 이 인터페이스를 구현한 실체 클래스가 필요
- 대표적인 JPA 실체 클래스 => 하이버네이트(Hibernate)
- **Hibernate**
 - JPA의 인터페이스를 구현한 실체 클래스
 - 자바의 ORM 프레임워크로 스프링 부트에서 데이터베이스 관리를 쉽게 하는 도구

JPA로 데이터베이스 사용하기

19

□ ORM과 JPA 이해하기

▣ ORM(Object Relational Mapping)

- Java 문법으로도 데이터베이스를 다룰 수 있는 도구
- SQL을 직접 작성하지 않아도 데이터베이스의 데이터 처리가 가능

▣ JPA(Java Persistence API)

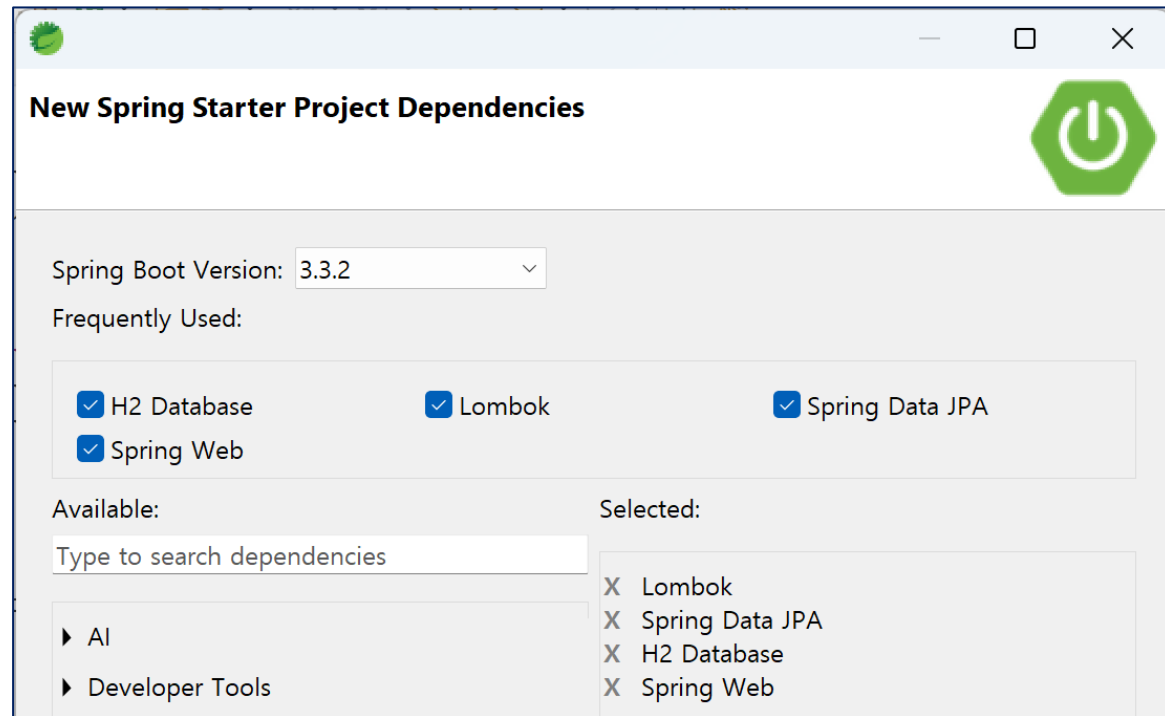
데이터베이스 관리 = JPA + Hibernate

- 스프링 부트는 JPA를 사용하여 데이터베이스를 관리
- 스프링 부트는 JPA를 ORM의 기술 표준으로 사용
- JPA는 인터페이스 모음으로 이 인터페이스를 구현한 실체 클래스가 필요
- 대표적인 JPA 실체 클래스 => 하이버네이트(Hibernate)
- Hibernate
 - JPA의 인터페이스를 구현한 실체 클래스
 - 자바의 ORM 프레임워크로 스프링 부트에서 데이터베이스 관리를 쉽게하는 도구

JPA로 데이터베이스 사용하기

20

- H2 데이터베이스 사용
 - ▣ 주로 개발 환경에서 사용하는 자바 기반의 경량 DBMS
 - ▣ 스프링 부트 프로젝트 생성(**week12_02JPADemo**)
 - ▣ 디펜던시 설정
 - **Spring Web**
 - **Lombok**
 - **Spring Data JPA**
 - **H2 Database**

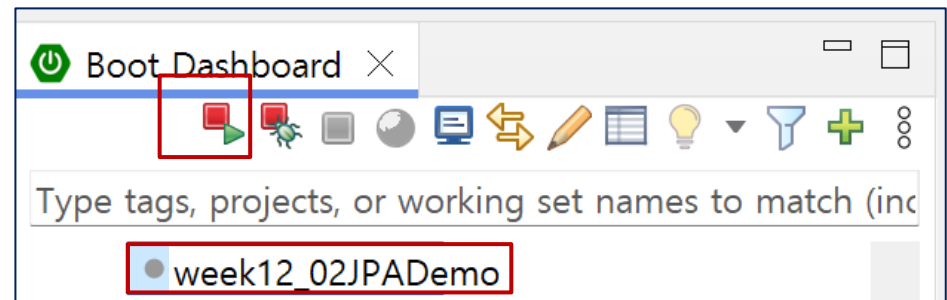
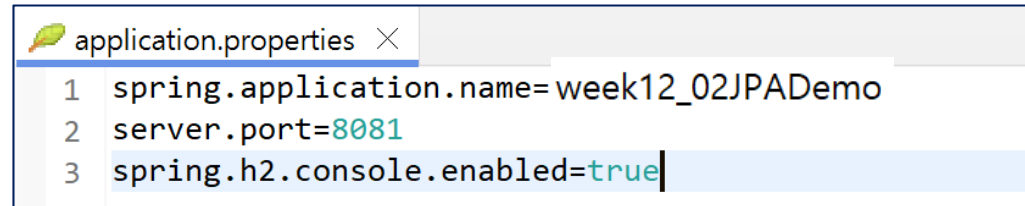
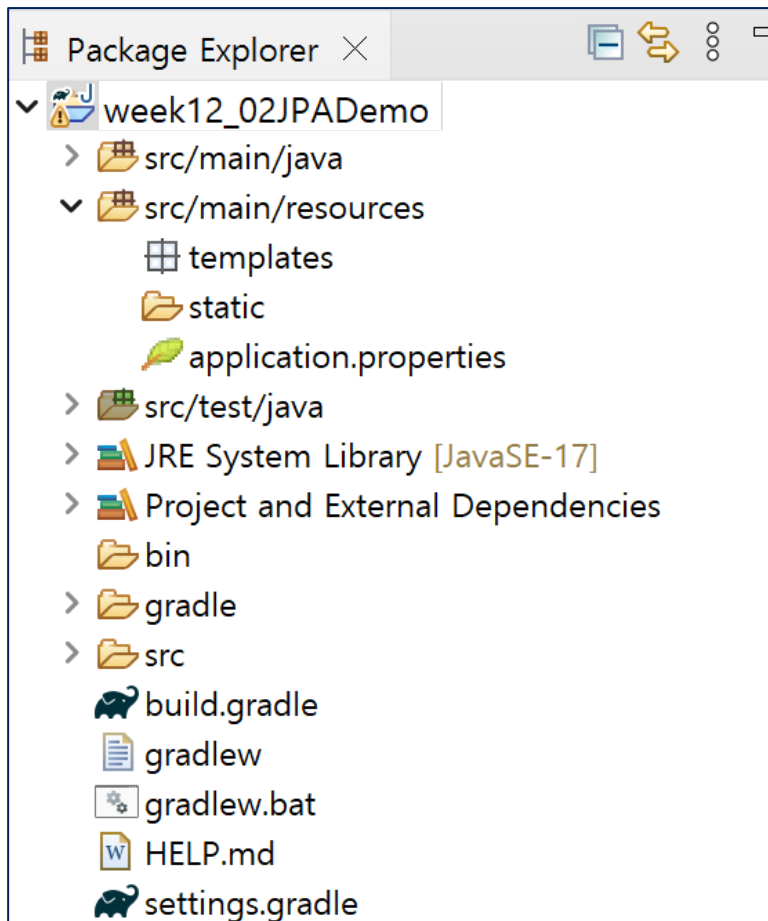


JPA로 데이터베이스 사용하기

21

□ H2 데이터베이스 사용

▣ application.properties



JPA로 데이터베이스 사용하기

22

- H2 데이터베이스 사용
 - ▣ 프로젝트 실행 후 콘솔 창

```
main] c.s.springboot.Ex12JpaDemoApplication : Starting Ex12JpaDemoApplication using Java 17.0.10 with PID 13324 (C:\Users\sunya\._workspace\2024-2\S
main] c.s.springboot.Ex12JpaDemoApplication : No active profile set, falling back to 1 default profile: "default"
main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories in DEFAULT mode.
main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning in 16 ms. Found 0 JPA repository interfaces.
main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8081 (http) with context path '/'
main] o.apache.catalina.core.StandardService : Starting
main] o.apache.catalina.core.StandardEngine : Starting
main] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1306 ms
main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
main] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - Added connection conn0: url=jdbc:h2:mem:1efa9784-88af-4876-95ae-270ffc9e591b user=SA
main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
main] o.s.b.a.h2.H2ConsoleAutoConfiguration : H2 console available at '/h2-console'. Database available at 'jdbc:h2:mem:1efa9784-88af-4876-95ae-270f
main] o.hibernate.jpa.internal.util.LogHelper : HHH0000204: Processing PersistenceUnitInfo [name: default]
main] org.hibernate.Version : HHH0000412: Hibernate ORM core version 6.5.2.Final
main] o.h.c.internal.RegionFactoryInitiator : HHH000026: Second-level cache disabled
main] o.s.o.j.p.SpringPersistenceUnitInfo : No LoadTimeWeaver setup: ignoring JPA class transformer
main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH0000489: No JTA platform available (set 'hibernate.transaction.jta.platform' to enable JTA platform
main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database queries may be performed during vie
main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8081 (http) with context path '/'
main] c.s.springboot.Ex12JpaDemoApplication : Started Ex12JpaDemoApplication in 3.659 seconds (process running for 8.982)
```

H2 DB 연결 url => 복사

JPA로 데이터베이스 사용하기

23

- H2 데이터베이스 연결
 - ▣ 브라우저 창 > **http://localhost:8081/h2-console**
 - ▣ 복사 한 url 주소 => **JDBC URL** 붙여넣기

한국어 ▼ 설정 도구 도움말

로그인

저장한 설정: Generic H2 (Embedded) ▼

설정 이름: Generic H2 (Embedded) 저장 삭제

드라이버 클래스: org.h2.Driver

JDBC URL: jdbc:h2:mem:f7d79527-4384-4a00-9de9-d3b081049466

사용자명: sa

비밀번호:

연결 연결 시험

JPA로 데이터베이스 사용하기

24

□ H2 데이터베이스

The screenshot shows the H2 console web interface in a browser. The address bar shows the URL: `localhost:8081/h2-console/login.do?sessionId=719c41e7aef04b88a115f75e5b248a9d`. The interface includes a sidebar with the database structure (jdbc:h2:~/local, INFORMATION_SCHEMA, 사용자, SA, Admin) and a main area with a toolbar (실행, Run Selected, 자동 완성, 지우기, SQL 문:). Below the toolbar, there is a section titled "중요 명령" (Important Commands) with a table of shortcuts and actions.

Icon	Command
?	이 도움말 페이지 보기
📄	명령 이력 보기
▶	Ctrl+엔터: 현재의 SQL 문 실행
▶	Shift+엔터: Executes the SQL statement defined by the text selection
⌨	Ctrl+Space: 자동 완성
🔌	데이터베이스 연결 끊기

Below the commands table, there is a section titled "샘플 SQL 스크립트" (Sample SQL Scripts) with a table of sample SQL statements and their descriptions.

Description	SQL Statement
데이터베이스가 존재하는 경우 삭제하기	DROP TABLE IF EXISTS TEST;
새 테이블 만들기	CREATE TABLE TEST(ID INT PRIMARY KEY, NAME VARCHAR(255));
컬럼은 ID와 NAME	INSERT INTO TEST VALUES(1, 'Hello');
행 추가	INSERT INTO TEST VALUES(2, 'World');
행 추가	SELECT * FROM TEST ORDER BY ID;
테이블 질의	UPDATE TEST SET NAME='Hi' WHERE ID=1;
행 데이터 변경	

JPA로 데이터베이스 사용하기

25

□ H2 데이터베이스

▣ url 고정

```
application.properties ×  
1 spring.application.name= week12_02JPADemo  
2 server.port=8081  
3 spring.h2.console.enabled=true  
4 spring.datasource.url= jdbc:h2:mem:f7d79527-4384-4a00-9de9-d3b081049466
```

한국어 ▼ 설정 도구 도움말

로그인

저장한 설정: Generic H2 (Embedded) ▼

설정 이름: Generic H2 (Embedded) 저장 삭제

드라이버 클래스: org.h2.Driver

JDBC URL: jdbc:h2:mem:f7d79527-4384-4a00-9de9-d3b081049466

사용자명: sa

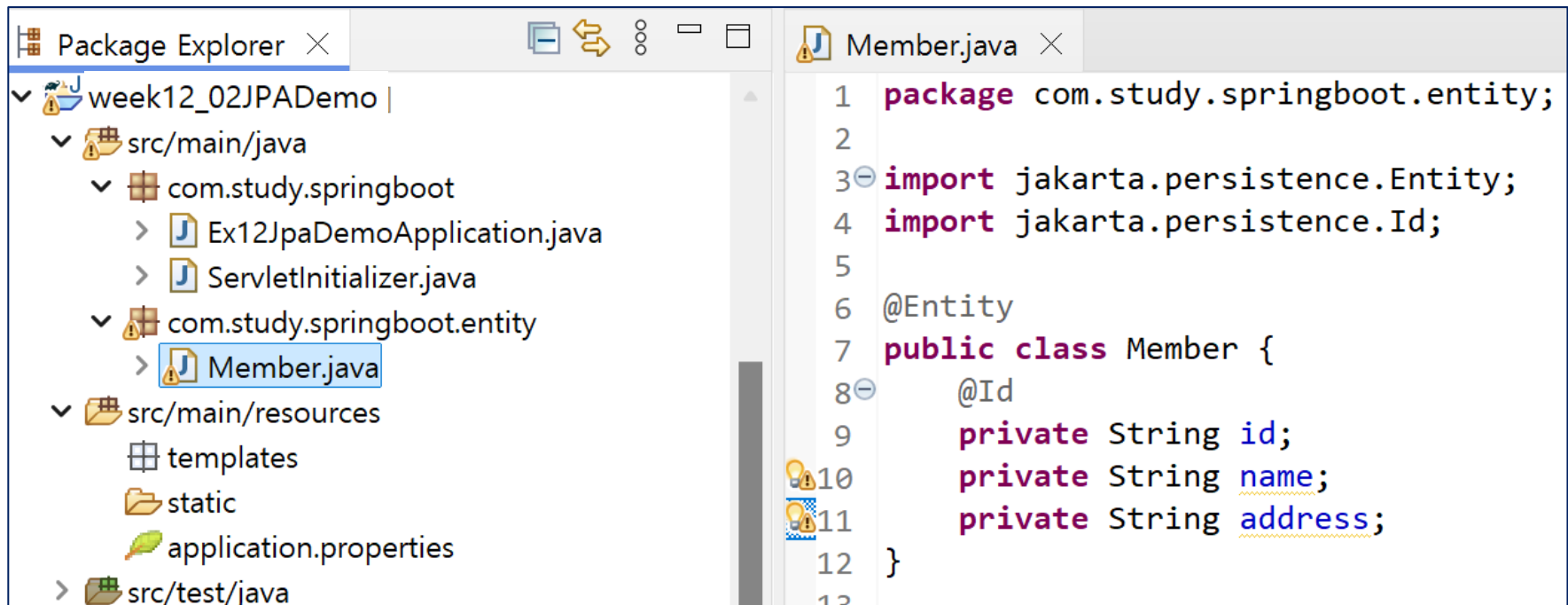
비밀번호:

연결 연결 시험

JPA로 데이터베이스 사용하기

26

- Entity로 테이블 매핑하기
 - ▣ Member 클래스



JPA로 데이터베이스 사용하기

27

- Entity로 테이블 매핑하기
 - ▣ 프로젝트 재실행 후 > H2 콘솔 확인

H2 콘솔

localhost:8081/h2-console/login.do?jsessionid=08d2185bff00568a58b6850ee403ec4a

자동 커밋 ☒ 최대 행 수: 1000 자동 완성 안함 Auto select On

jdbc:h2:mem:demo

MEMBER

- ADDRESS CHARACTER VARYING(255)
- ID CHARACTER VARYING(255)
- NAME CHARACTER VARYING(255)

인덱스

INFORMATION_SCHEMA

사용자

H2 2.2.224 (2023-09-17)

실행 Run Selected 자동 완성 지우기 SQL 문:

중요 명령

JPA로 데이터베이스 사용하기

28

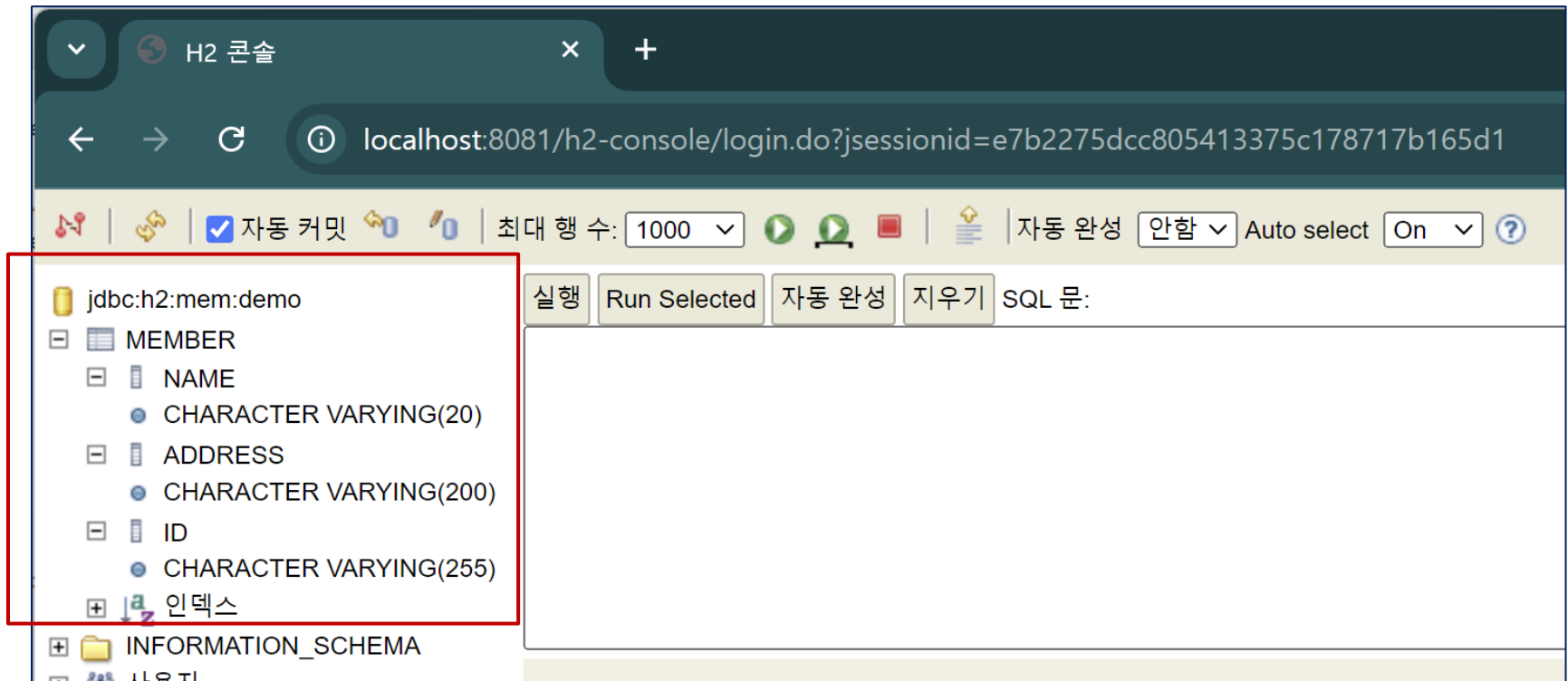
- Entity로 테이블 매핑하기
 - ▣ 프로젝트 재실행 후 > H2 콘솔 확인

```
Member.java ×
1 package com.study.springboot.entity;
2
3 import jakarta.persistence.Column;
4 import jakarta.persistence.Entity;
5 import jakarta.persistence.Id;
6
7 @Entity
8 public class Member {
9     @Id
10     private String id;
11
12     @Column(length = 20)
13     private String name;
14
15     @Column(length = 200)
16     private String address;
17 }
```

JPA로 데이터베이스 사용하기

29

- Entity로 테이블 매핑하기
 - ▣ 프로젝트 재실행 후 > H2 콘솔 확인



Repository로 데이터베이스 사용하기

30

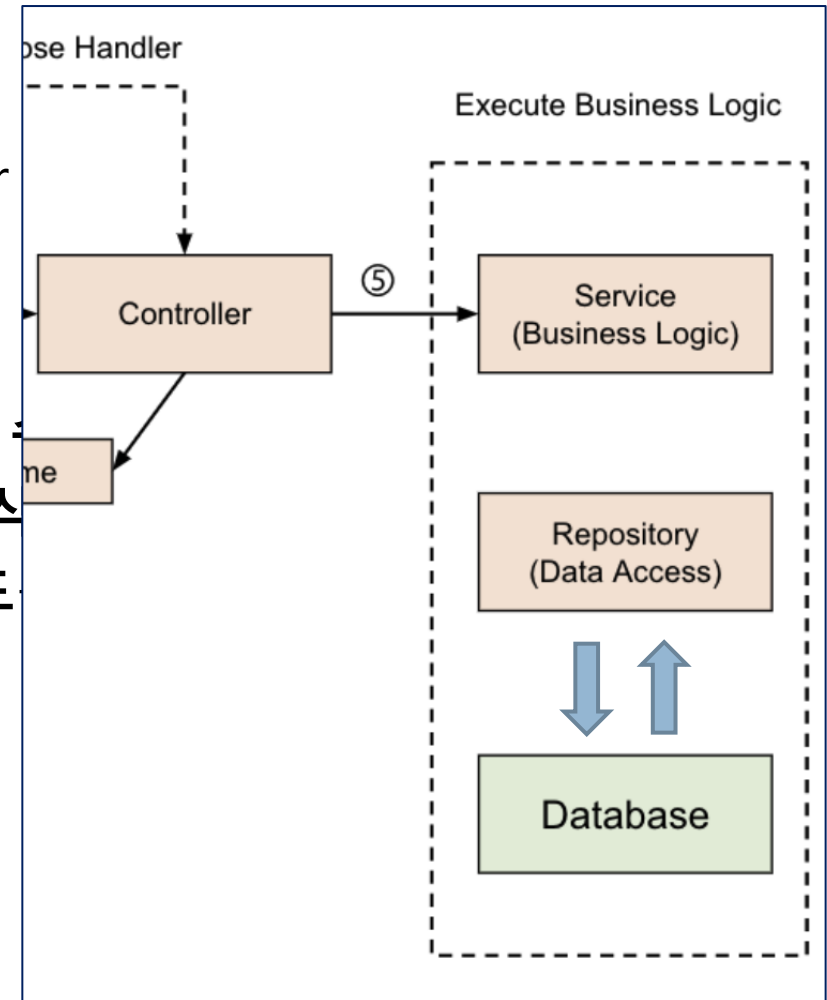
- 테이블을 구성하기 위해
 - ▣ Entity를 사용
 - ▣ 예) Question, Answer, Member 등

- **JPA Repository**
 - ▣ 테이블의 데이터를 저장, 조회, 수정, 삭제 등을 지원
 - ▣ **JpaRepository 인터페이스 형식을 이용**
 - ▣ 데이터를 관리하는 여러 메서드를 제공
 - findAll(), save() 등

Repository로 데이터베이스 사용하기

31

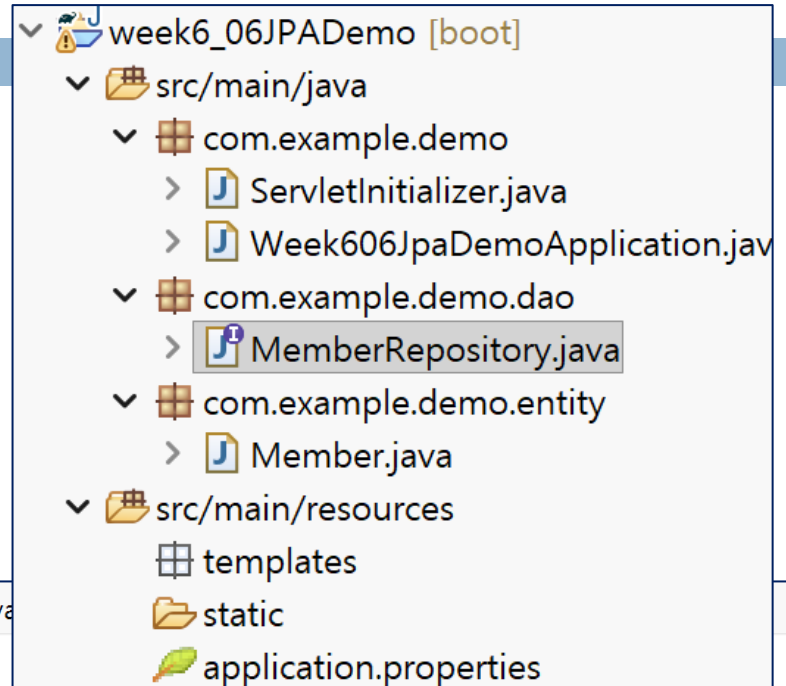
- 테이블을 구성하기 위해
 - ▣ Entity를 사용
 - ▣ 예) Question, Answer, Member
- **JPA Repository**
 - ▣ 테이블의 데이터를 저장, 조회, 삭제
 - ▣ JpaRepository 인터페이스 형식
 - ▣ 데이터를 관리하는 여러 메서드
 - findAll(), save() 등



Repository로 데이터베이스 사용하기

32

- Repository 생성하기
 - ▣ **MemberRepository** 인터페이스



MemberRepository.java × MemberController.java Member.java

```
1 package com.example.demo.dao;
2
3 import org.springframework.data.jpa.repository.JpaRepository;
4
5 import com.example.demo.entity.Member;
6
7 public interface MemberRepository extends JpaRepository<Member, String> {
8
9 }
10
```

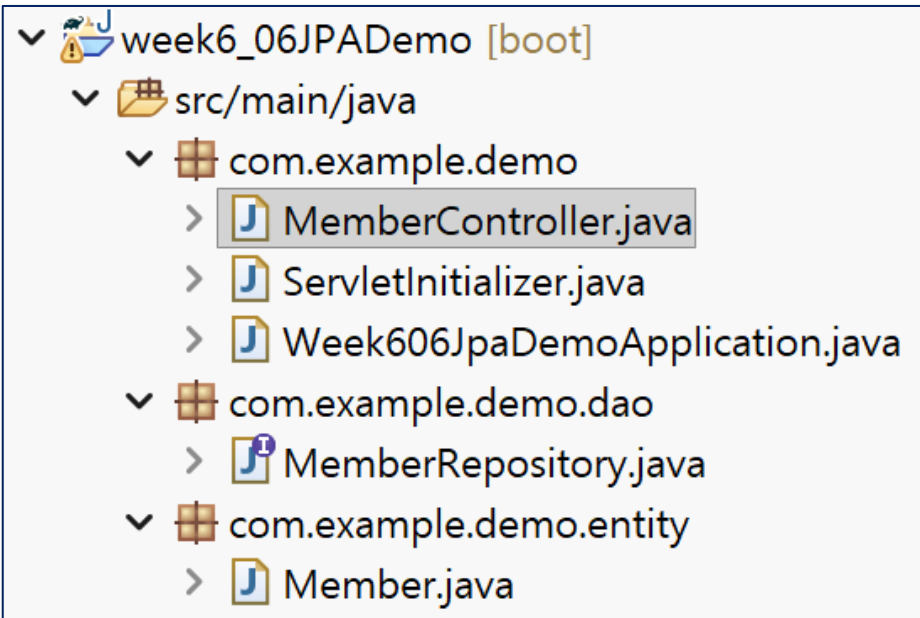
해당 Entity의
키 타입

JPA로 핸들링 할
Entity 명(테이블)

Repository로 데이터베이스 사용하기

33

- Repository를 이용하여 데이터 생성하기
 - ▣ 클라이언트 요청정보("insert") 처리 클래스
 - ▣ **MemberController** 클래스



Repository로 데이터베이스 사용하기

34

- Repository를 이용하여 데이터 생성하기
 - ▣ 클라이언트 요청정보("insert") 처리 클래스
 - ▣ **MemberController** 클래스

```
@RestController
public class MemberController {
    private MemberRepository memberRepository;

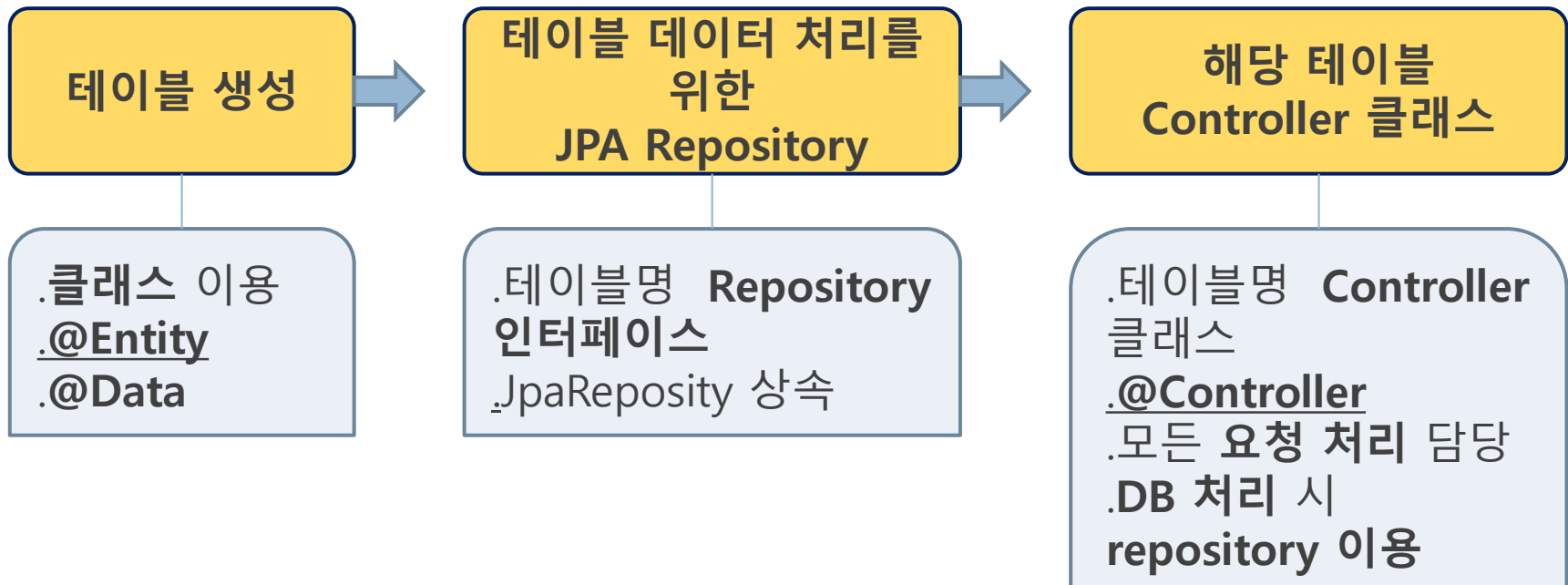
    public MemberController(MemberRepository repository) {
        memberRepository = repository;
    }

    @GetMapping("/insert")
    public void insert() {
        Member member = new Member();
        member.setId("admin");
        member.setName("홍길동");
        member.setAddress("경기도 화성시");
        memberRepository.save(member);
    }
}
```

JPA로 데이터베이스 사용하기

35

- 테이블 생성 후 JPA로 DB 처리를 위한 설정 순서



JPA로 데이터베이스 사용하기

36

□ Entity로 테이블 매핑하기

▣ Question 클래스

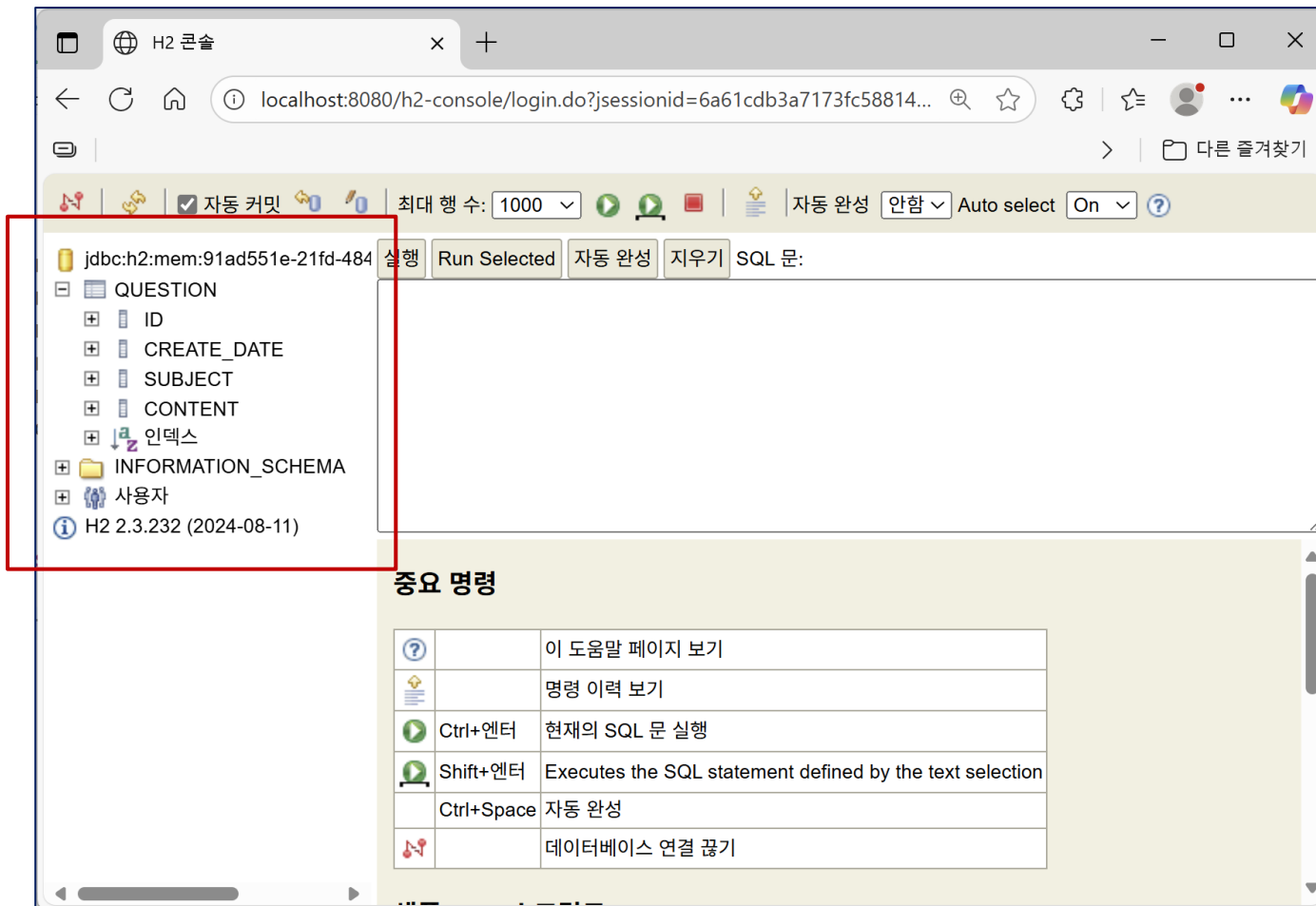
데이터를 저장할 때 해당 속성 값을
자동으로 1씩 증가하여 저장

```
Question.java ×
1 package com.example.demo.entity;
2
3 import java.time.LocalDateTime;
4
5 import jakarta.persistence.Column;
6 import jakarta.persistence.Entity;
7 import jakarta.persistence.GeneratedValue;
8 import jakarta.persistence.GenerationType;
9 import jakarta.persistence.Id;
10 import lombok.Data;
11
12 @Data
13 @Entity
14 public class Question {
15     @Id
16     @GeneratedValue(strategy = GenerationType.IDENTITY)
17     private int id;
18
19     @Column(length = 200)
20     private String subject;
21
22     @Column(columnDefinition = "TEXT")
23     private String content;
24
25     private LocalDateTime createDate;
26 }
```

JPA로 데이터베이스 사용하기

37

- Entity로 테이블 매핑하기
 - ▣ 프로젝트 재실행 후 > H2 콘솔 확인



Repository로 데이터베이스 사용하기

38

- 테이블을 구성하기 위해
 - ▣ Entity를 사용
 - ▣ 예) Question, Answer, Member 등

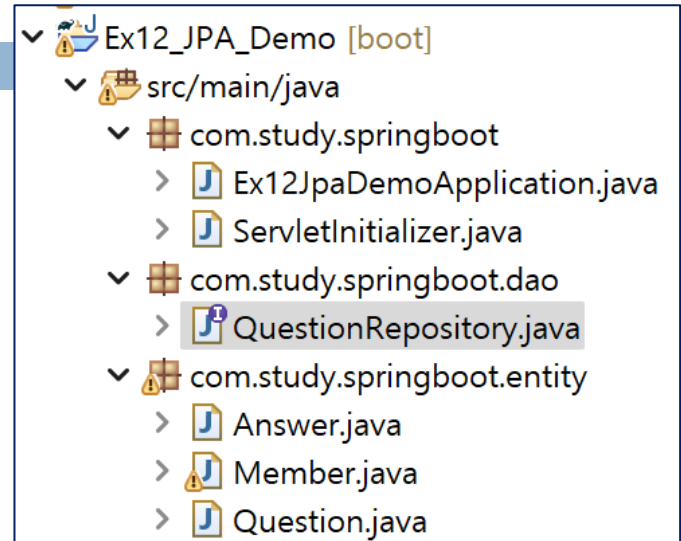
- **JPA Repository**
 - ▣ 테이블의 데이터를 저장, 조회, 수정, 삭제 등을 지원
 - ▣ **JpaRepository 인터페이스 형식을 이용**
 - ▣ 데이터를 관리하는 여러 메서드를 제공
 - findAll(), save() 등

Repository로 데이터베이스 사용하기

39

- Repository 생성하기
 - ▣ **QuestionRepository** 인터페이스

Repository는 테이블 당 하나씩

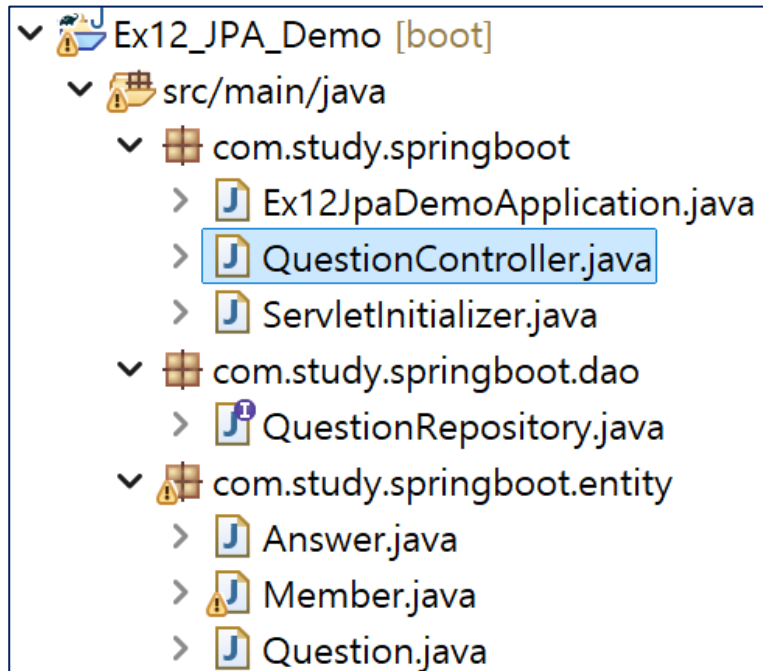


```
QuestionRepository.java ×
1 package com.study.springboot.dao;
2
3 import org.springframework.data.jpa.repository.JpaRepository;
4
5 import com.study.springboot.entity.Question;
6
7 public interface QuestionRepository extends JpaRepository<Question, Integer> {
8
9 }
10
```

Repository로 데이터베이스 사용하기

40

- Repository를 이용하여 데이터 생성하기
 - ▣ 클라이언트 요청정보("insert") 처리 클래스
 - ▣ **QuestionController** 클래스



Repository로 데이터베이스 사용하기

41

```
QuestionController.java ×
1 package com.study.springboot;
2
3 import java.time.LocalDateTime;
4
5 import org.springframework.web.bind.annotation.GetMapping;
6 import org.springframework.web.bind.annotation.RestController;
7
8 import com.study.springboot.dao.QuestionRepository;
9 import com.study.springboot.entity.Question;
10
11 @RestController
12 public class QuestionController {
13     private QuestionRepository repository;
14
15     public QuestionController(QuestionRepository repository) {
16         this.repository = repository;
17     }
18
19     @GetMapping("/insert")
20     public void insert() {
21         Question question = new Question();
22         question.setSubject("sbb가 무엇인가요?");
23         question.setContent("sbb에 대해 알고 싶습니다");
24         question.setCreateDate(LocalDateTime.now());
25         repository.save(question);
26     }
27 }
```

Repository로 데이터베이스 사용하기

42

- Repository를 이용하여 데이터 생성하기
 - ▣ 프로젝트 재실행 > "localhost:8080/insert"
 - ▣ H2 콘솔 창 > 새로고침 > SQL 문장으로 결과 확인

실행

Run Selected

자동 완성

지우기

SQL 문:

```
select * from question
```

select * from question;

ID	CREATE_DATE	SUBJECT	CONTENT
1	2024-08-11 12:53:31.894076	sbb가 무엇인가요?	sbb에 대해 알고 싶습니다

(1 row, 1 ms)

편집

Repository로 데이터베이스 사용하기

43

```
QuestionController.java ×
12
13 @RestController
14 public class QuestionController {
15     private QuestionRepository repository;
16
17     public QuestionController(QuestionRepository repository) {
18         this.repository = repository;
19     }
20
21     @GetMapping("/insert")
22     public void insert() {
23         Question question1 = new Question();
24         question1.setSubject("sbb가 무엇인가요?");
25         question1.setContent("sbb에 대해 알고 싶습니다");
26         question1.setCreateDate(LocalDateTime.now());
27         repository.save(question1);
28
29         Question question2 = new Question();
30         question2.setSubject("스프링 부트 모델 질문입니다.");
31         question2.setContent("id는 자동으로 생성되나요?");
32         question2.setCreateDate(LocalDateTime.now());
33         repository.save(question2);
34     }
35
36     @GetMapping("/display")
37     @ResponseBody
38     public List<Question> display() {
39         List<Question> q = repository.findAll();
40         return q;
41     }
42 }
```

Repository로 데이터베이스 사용하기

44

- Repository를 이용하여 데이터 생성하기
 - ▣ 프로젝트 재실행 > localhost:8081/insert > localhost:8081/display
 - ▣ H2 콘솔 창 > 새로고침 > SQL 문장으로 결과 확인

실행

Run Selected

자동 완성

지우기

SQL 문:

```
select * from question
```

```
select * from question;
```

ID	CREATE_DATE	SUBJECT	CONTENT
1	2024-08-11 13:51:56.017478	sbb가 무엇인가요?	sbb에 대해 알고 싶습니다
2	2024-08-11 13:51:56.107715	스프링 부트 모델 질문입니다.	id는 자동으로 생성되나요?

(2 행, 4 ms)

JSP 기반 웹 프로젝트 구성

45

□ JSP 설정

- ▣ **build.gradle에 디펜던시 추가**
- ▣ **Gradle > Refresh Gradle Project**
- ▣ 프로젝트 > properties > Project Facets
 - **Dynamic Web Module > 5.0**
- ▣ **resources 폴더**
 - **application.properties 수정**
 - **spring.mvc.view.prefix=/WEB-INF/views/**
 - **spring.mvc.view.suffix=.jsp**
 - **JSP 폴더 생성**
 - **src > main > webapp > WEB-INF > views**
 - **jsp 파일 생성**
 - **views 폴더에 생성**

JSP 기반 웹 프로젝트 구성 => JSP 설정

46

QuestionController.java ×

```
5 @Controller
6 public class QuestionController {
7     private QuestionRepository repository;
8
9     public QuestionController(QuestionRepository repository) {
10         this.repository = repository;
11     }
12
13     @GetMapping("/")
14     public String index() {
15         return "index";
16     }
17
18     @GetMapping("/insertForm")
19     public String insertForm() {
20         return "insertForm";
21     }
22 }
```

JSP 기반 웹 프로젝트 구성

47

QuestionController.java ×

```
3  @GetMapping("/insert")
4  public String insert(Question question) {
5      question.setCreateDate(LocalDateDateTime.now());
6      repository.save(question);
7      return "redirect:list";
8  }
9
10 @GetMapping("/list")
11 public String list(Model model) {
12     List<Question> questionList = repository.findAll();
13     model.addAttribute("questionList", questionList);
14     return "question_list";
15 }
```

JSP 기반 웹 프로젝트 구성

48

□ index.jsp

```
index.jsp ×
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6   <meta charset="UTF-8">
7   <title>Insert title here</title>
8   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet"
9     integrity="sha384-QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNlyT2bRjXh0JMHjY6hW+ALEwIH" crossorigin="anonymous">
10 </head>
11 <body>
12 <div class="container" style="padding: 20px;">
13   <h3> ♠ 게시판 프로그램 ♠</h3>
14   <br>
15   <a href="/insertForm" class="btn btn-primary">등록</a>
16   <a href="/list" class="btn btn-primary">조회</a>
17 </div>
18
19 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"
20   integrity="sha384-YvpcrYf0tY3LHB60NNkmXc5s9fDVZLESaAA55NDz0xhy9GkcIdslK1eN7N6jIeHz" crossorigin="anonymous"></script>
21 </body>
22 </html>
```


JSP 기반 웹 프로젝트 구성

49

□ insertForm.jsp

```
insertForm.jsp ×
10 </head>
11 <body>
12
13 <div class="container" style="padding: 20px;">
14     <h3> ♣ 게시판 등록 ♣</h3>
15     <br>
16     <form action="/insert">
17         제 목 : <input type="text" name="subject"> <br>
18         내 용 : <br>
19             <textarea rows="5" cols="30" name="content"></textarea><br><br>
20         <input type="submit" value="등록">
21     </form>
22 </div>
23
```

JSP 기반 웹 프로젝트 구성

50

□ question_list.jsp

```
question_list.jsp ×
3 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
4 <!DOCTYPE html><html><head><meta charset="UTF-8">
5 <title>Insert title here</title>
6 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet"
7 integrity="sha384-QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNlyT2bRjXh0JMhY6hW+ALEwIH" crossorigin="anonymous">
8 </head><body>
9 <div class="container" >
10 <h3> ⬆ 게시판 목록 ⬆</h3>
11 <br>
12 <table border="1" cellpadding="5">
13 <tr>
14 <th>ID</th>
15 <th>Subject</th>
16 <th>Content</th>
17 <th>CreateDate</th>
18 </tr>
19 <c:forEach var="q" items="${questionList }">
20 <tr>
21 <td>${q.id } </td>
22 <td>${q.subject } </td>
23 <td>${q.content } </td>
24 <td>${q.createDate } </td>
25 </tr>
26 </c:forEach>
27 </table>
28
29 <br><br>
30 <a href="/insertForm" class="btn btn-primary">입력화면</a>
31 <a href="/" class="btn btn-primary">초기화면</a>
32 </div>
```

□ 기말 프로젝트

▣ 스프링부트 기반 웹 시스템 개발 프로젝트

▣ 프로젝트 주제 정하기

- 선정 동기
- 벤치마킹 사이트
- 컨셉트 등 그 외 부가사항 포함 가능

▣ 화면 구성도

- 페이지 디자인 및 내용 구성

▣ DB 테이블

- 테이블 구성도

과제물

52

- 제출 형식
 - ▣ 제출 파일 : 학번_이름.pdf
 - ▣ ppt로 작업하고 pdf 형식으로 저장 후 제출