09

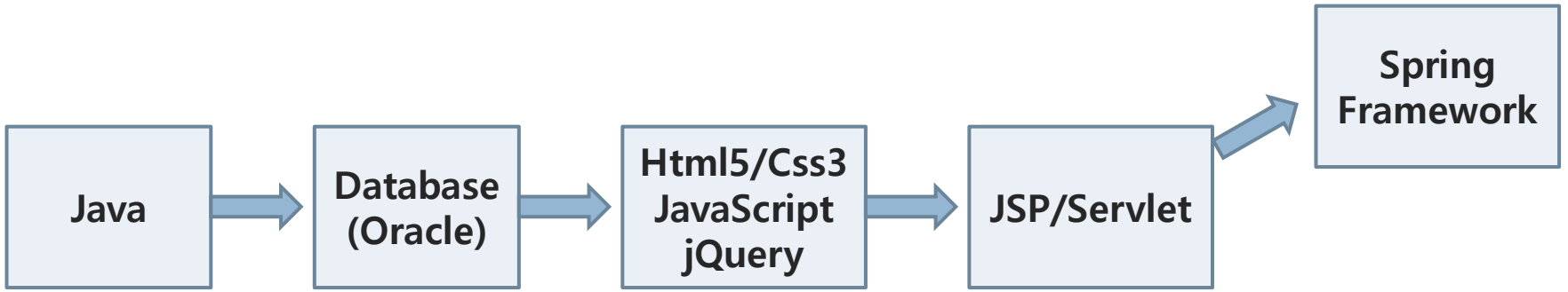
Spring Boot 개요 및 환경 구축

2

Spring Boot의 개요

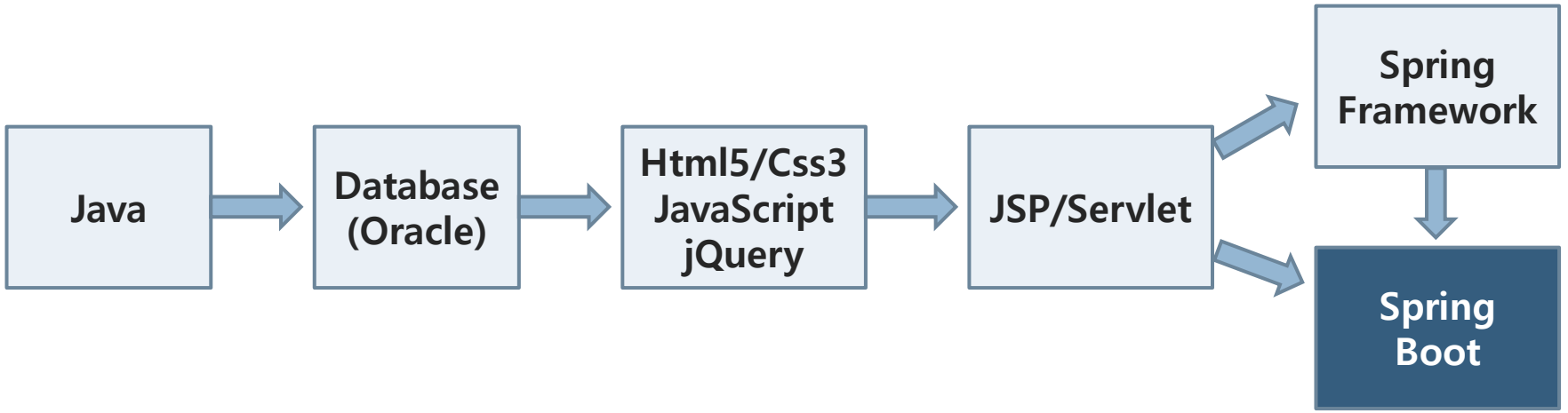
자바 웹 애플리케이션

- 자바 웹 애플리케이션 개발자 학습 로드맵



자바 웹 애플리케이션

- 자바 웹 애플리케이션 개발자 학습 로드맵



스프링 프레임워크와 스프링 부트

- **스프링 프레임워크(Spring Framework)**
  - **스프링 프레임워크의 등장**
    - 점점 복잡해지는 엔터프라이즈 애플리케이션
      - 대규모의 복잡한 데이터를 관리하는 애플리케이션
      - 예) 금융 시스템 – 서버 성능, 안정성, 보안 기능 등이 매우 중요
    - 엔터프라이즈 애플리케이션을 위한 개발 환경을 제공해 줄 도구 필요
  - **스프링 프레임워크의 특징**
    - 복잡한 애플리케이션 개발을 위한 여러 기능을 제공
    - 장점이 많은 개발 도구이지만 설정이 매우 복잡

스프링 프레임워크와 스프링 부트

- **스프링 부트(Spring Boot)**
  - **개요**
    - 스프링 프레임워크를 더 쉽고 빠르게 이용할 수 있도록 만들어주는 도구
    - **빠르게 스프링 프로젝트를 설정**
    - 의존성 세트라고 불리는 스타터를 사용해 **간편하게 의존성을 사용**
    - **개발자**가 조금 더 **비즈니스 로직 개발에 집중**할 수 있도록 하는 도구

스프링 프레임워크와 스프링 부트

스프링 부트(Spring Boot)

특징

- 톰캣과 같은 **웹 애플리케이션 서버(Web Application Server, WAS)가 내장**되어 있어서 따로 설치할 필요가 없음 – **독립적 실행 가능**
- **내장 데이터베이스(H2 DB)**
- 빌드 구성을 단순화하는 **스프링 부트 스타터**를 제공
- XML 설정을 하지 않고 **자바 코드로 모두 작성 가능**
- **JAR를 이용**해서 자바 옵션만으로도 **배포가 가능**
- 애플리케이션의 모니터링 및 관리 도구인 스프링 액츄에이터를 제공

스프링 프레임워크와 스프링 부트

□ 스프링과 스프링 부트 비교

▼ 스프링과 스프링 부트 특징 비교

	스프링	스프링 부트
목적	엔터프라이즈 애플리케이션 개발을 더 쉽게 만들기	스프링의 개발을 더 빠르고 쉽게 하기
설정 파일	개발자가 수동으로 구성	자동 구성
XML	일부 파일은 XML로 직접 생성하고 관리	사용하지 않음
인메모리 데이터베이스 지원	지원하지 않음	인메모리 데이터베이스 자동 설정 지원
서버	프로젝트를 띄우는 서버(예 : 톰캣, 제티)를 별도로 수동 설정	내장형 서버를 제공해 별도의 설정이 필요 없음

스프링 부트는 스프링 프레임워크를 개선한 것

스프링 프레임워크 컨셉 정리

- **IoC(Inversion of Control)**
  - **제어의 역전**
  - 필요한 다른 객체를 직접 생성하거나 제어하는 것이 아니라 **외부에서 관리하는 객체**를 가져와서 **사용**하는 것
  - 클래스 A에서 클래스 B 객체 생성 예

```
public class A {
    B b = new B();
}
```

클래스 A에서 new 키워드로 클래스 B의 객체 생성

  - **스프링 컨테이너가 객체를 관리하는 방식** 예

```
public class A {
    private B b;
}
```

코드에서 객체를 생성하지 않음. 어디선가 받아온 **객체를 b에 할당**

스프링 프레임워크 컨셉 정리

- **IoC(Inversion of Control)**
  - **제어의 역전**
  - 필요한 다른 객체를 직접 생성하거나 제어하는 것이 아니라 **외부에서 관리하는 객체**를 가져와서 **사용**하는 것
  - 클래스 A에서 클래스 B 객체 생성 예

```
public class A {
    B b = new B();
}
```

객체 생성에 대한 제어권을 개발자가 아니라 스프링 프레임에서 관리

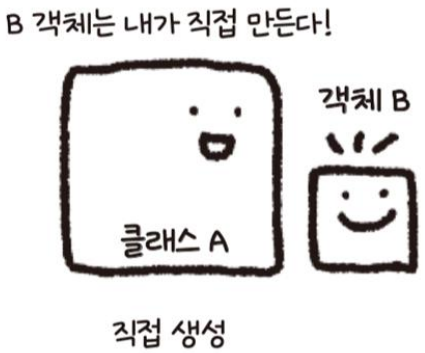
  - **스프링 컨테이너가 객체를 관리하는 방식** 예

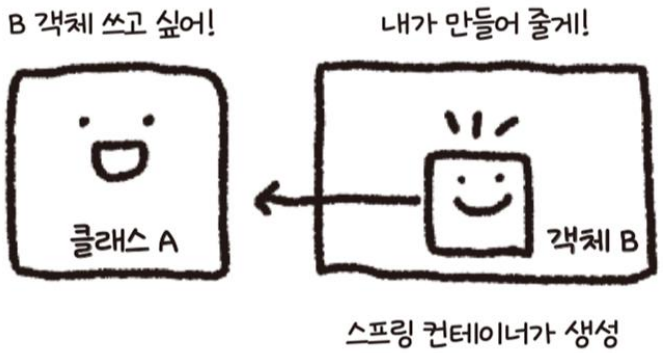
```
public class A {
    private B b;
}
```

코드에서 객체를 생성하지 않음. 어디선가 받아온 **객체를 b에 할당**

스프링 프레임워크 컨셉 정리

- **DI(Dependency Injection)**
  - **의존성 주입(=의존 객체 조립)**
  - 제어의 역전을 구현하기 위해 사용하는 방법
  - DI는 어떤 클래스가 다른 클래스에 의존한다는 뜻






```
public class A {
  // A에서 B를 주입받음
  @Autowired
  B b;
}
```

```
public class A {
    B b = new B();
}
```

스프링 프레임워크 컨셉 정리

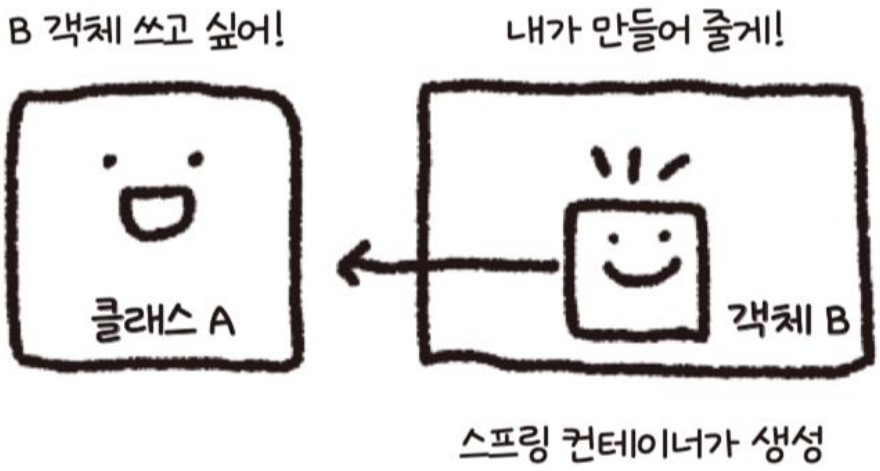
빈(Bean)과 스프링 컨테이너

스프링 컨테이너

- **빈(bean)**이 생성되고 소멸되기까지의 생명주기를 스프링 컨테이너가 관리
- 개발자가 **@Autowired**와 같은 어노테이션을 사용해서 필요한 빈을 주입받음

빈(Bean)

- 스프링 컨테이너가 생성하고 관리하는 객체
- 예) 객체 B가 빈에 해당됨



스프링 프레임워크 컨셉 정리

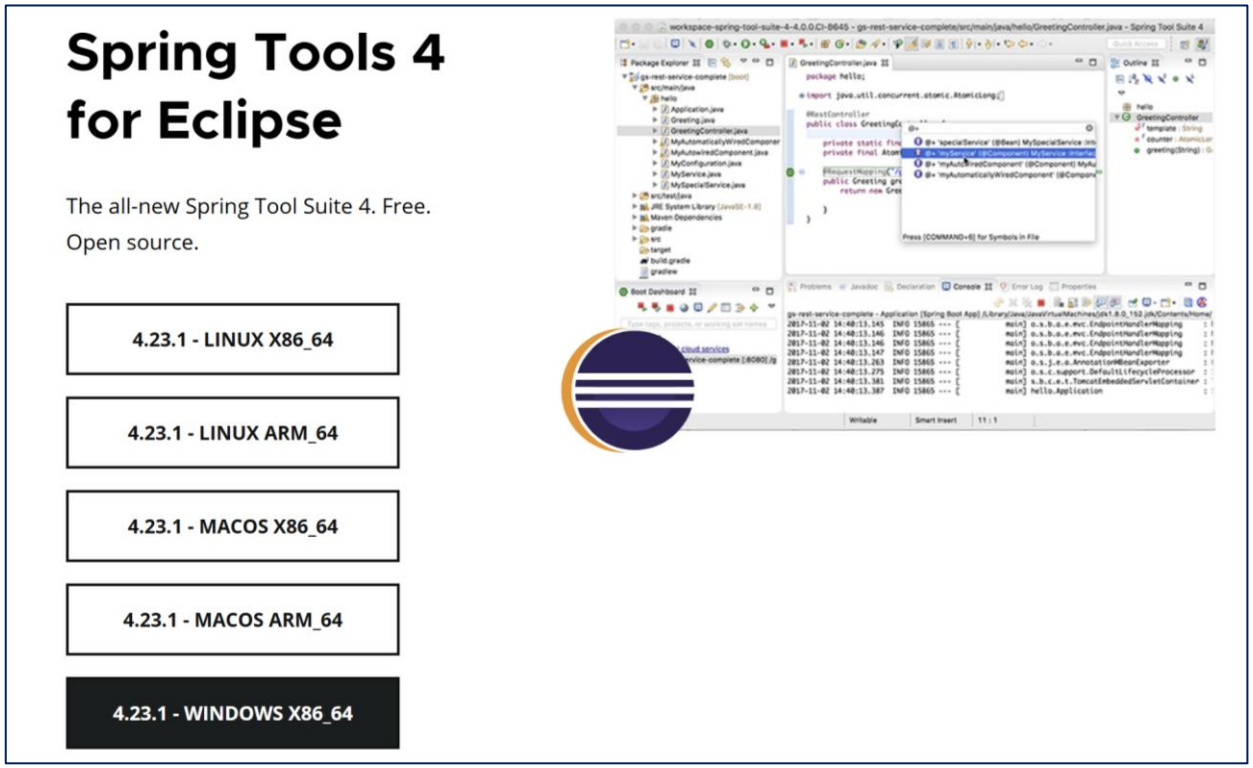
- **빈과 스프링 컨테이너**
  - **스프링 컨테이너에 빈을 등록하는 방법**
    - XML 파일 설정
    - 어노테이션 추가 등
  - **어노테이션 @Component로 빈 등록**
    - 클래스를 빈으로 등록하는 예

```
@Component
public class MyBean {

}
```

클래스 MyBean을 빈(bean)으로 등록

스프링 부트 환경 구축

- **스프링 전용 STS(Spring Tool Suite) 다운로드**
  - **https://spring.io/tools**



스프링 부트 환경 구축

- **STS 설치**
  - **다운로드 한 압축 파일 풀기**
  - **SpringToolSuite4.exe 실행**

스프링 부트 환경 구축

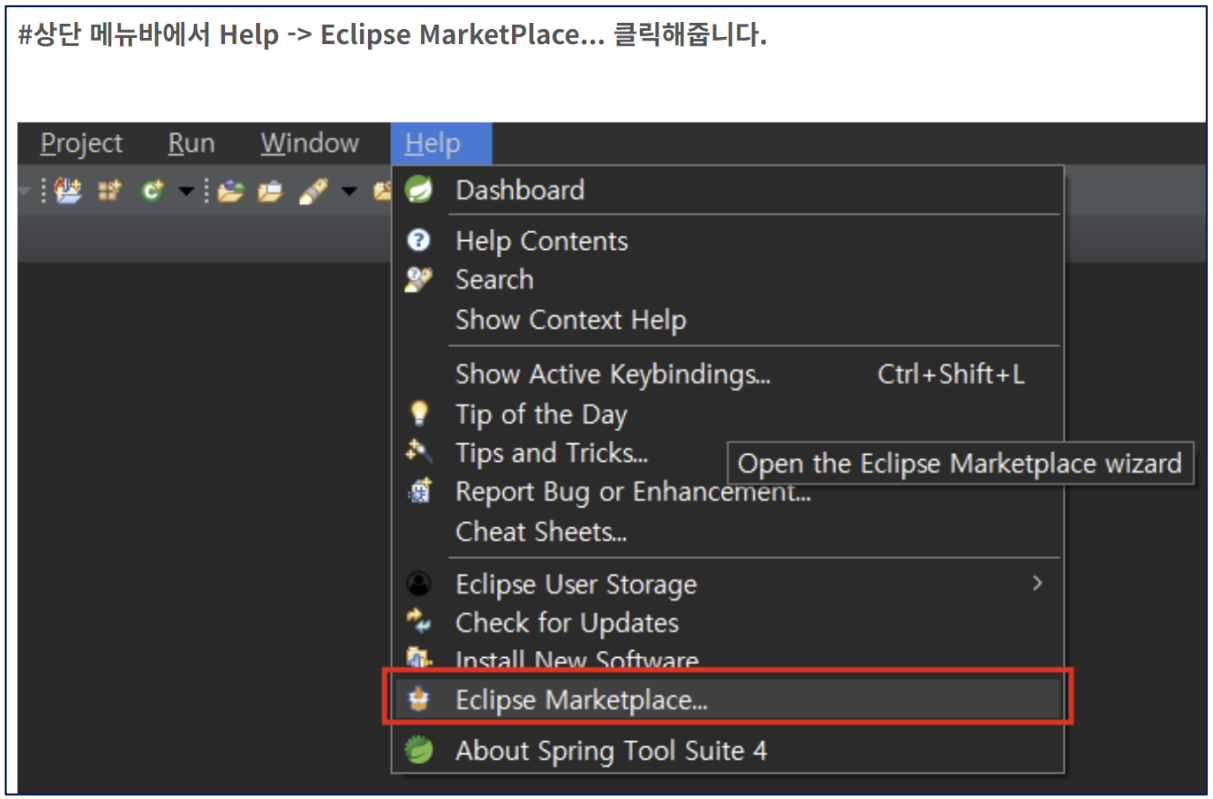
- **이클립스 환경 설정**
  - **인코딩 설정 > Window 메뉴 > Preferences**
    - General > Workspace > Text file encoding > Other > UTF-8
    - Web > CSS Files > Encoding > UTF-8
    - Web > HTML Files > Encoding > UTF-8
    - Web > JSP Files > Encoding > UTF-8
    - XML > XML Files > Encoding > UTF-8
    - JSON > JSON Files > Encoding > UTF-8
  - 브라우저 설정
    - General > Web Browser > Chrome

스프링 부트 환경 구축

- **이클립스 환경 설정**
  - **인코딩 설정 > Window**
    - General > Workspace >
    - Web > CSS Files > Encoding > UTF-8

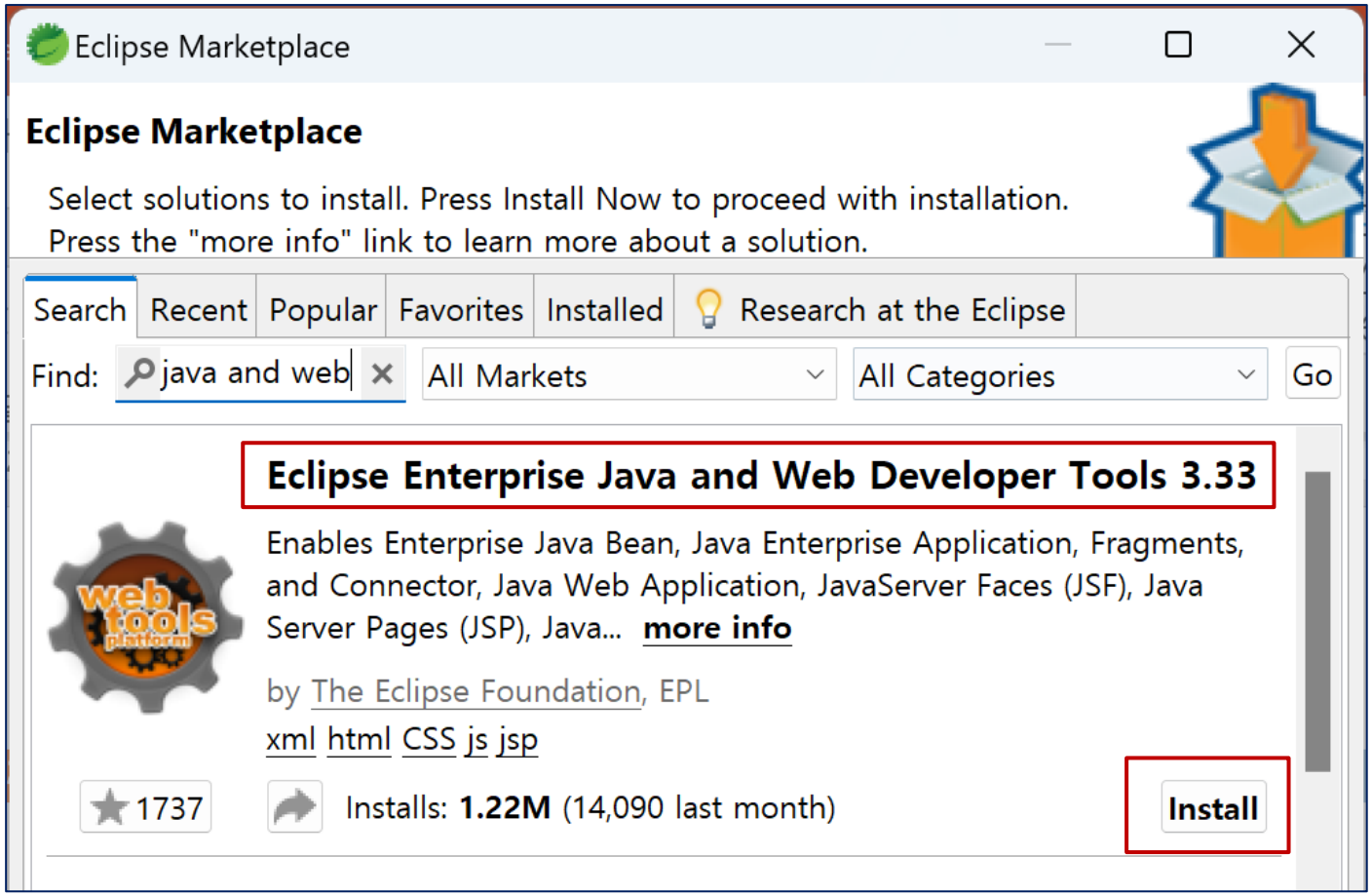
Preferences > Web 속성이 없는 경우

#상단 메뉴바에서 Help -> Eclipse MarketPlace... 클릭해줍니다.

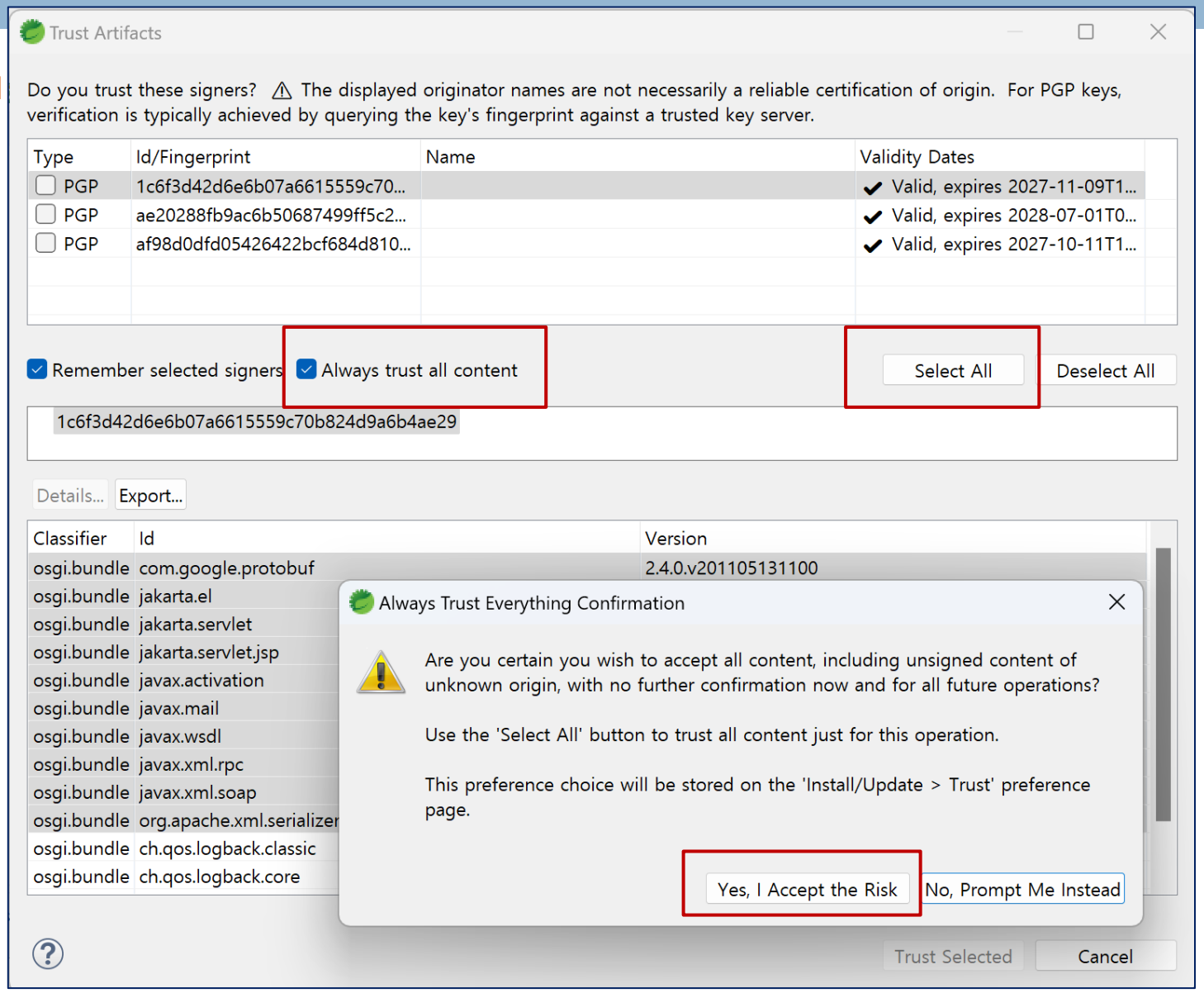


스프링 부트 환경 구축

- **이클립스 환경 설정**
  - Eclipse Marketplace > '**web**' 검색



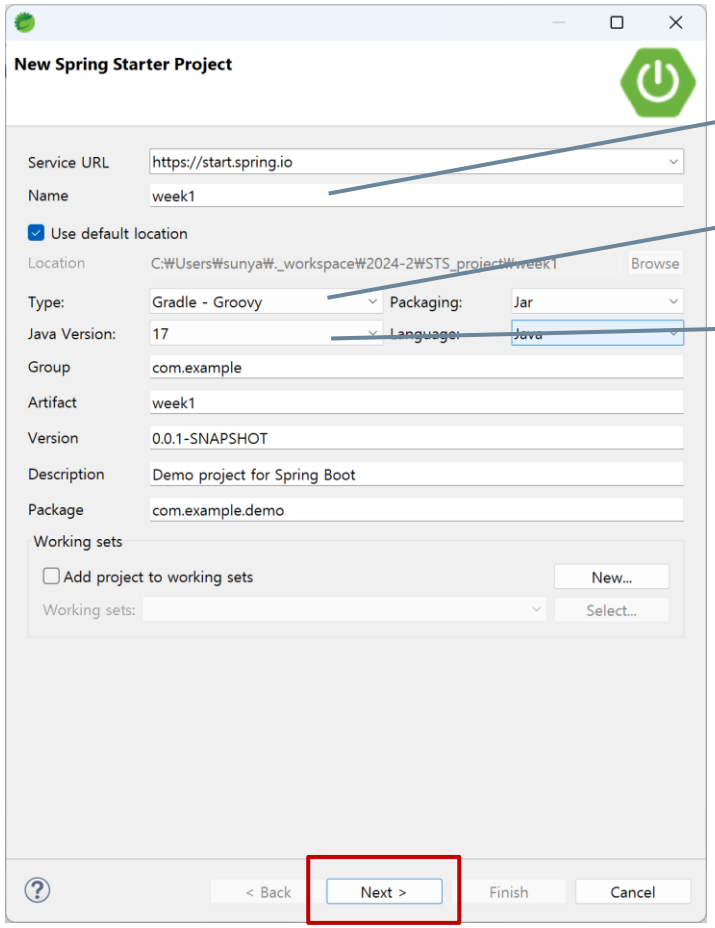
스프링 부트 환경 구축


Trust Artifacts
Do you trust these signers? ⚠ The displayed originator names are not necessarily a reliable certification of origin. For PGP keys, verification is typically achieved by querying the key's fingerprint against a trusted key server.
Type
Id/Fingerprint
Name
Validity Dates
PGP 1c6f3d42d6e6b07a6615559c70... Valid, expires 2027-11-09T1...
PGP ae20288fb9ac6b50687499ff5c2... Valid, expires 2028-07-01T0...
PGP af98d0dfd05426422bcf684d810... Valid, expires 2027-10-11T1...
Remember selected signers
Always trust all content
Select All
Deselect All
1c6f3d42d6e6b07a6615559c70b824d9a6b4ae29
Details...
Export...
Classifier
Id
Version
osgi.bundle com.google.protobuf 2.4.0.v201105131100
osgi.bundle jakarta.el
osgi.bundle jakarta.servlet
osgi.bundle jakarta.servlet.jsp
osgi.bundle javax.activation
osgi.bundle javax.mail
osgi.bundle javax.wsdl
osgi.bundle javax.xml.rpc
osgi.bundle javax.xml.soap
osgi.bundle org.apache.xml.serializer
osgi.bundle ch.qos.logback.classic
osgi.bundle ch.qos.logback.core
Always Trust Everything Confirmation
Are you certain you wish to accept all content, including unsigned content of unknown origin, with no further confirmation now and for all future operations?
Use the 'Select All' button to trust all content just for this operation.
This preference choice will be stored on the 'Install/Update > Trust' preference page.
Yes, I Accept the Risk
No, Prompt Me Instead
Trust Selected
Cancel

스프링 부트 시작하기

스프링 부트로 프로젝트 만들기

- File > New > Spring Starter Project



week9_01Hello : 프로젝트 이름

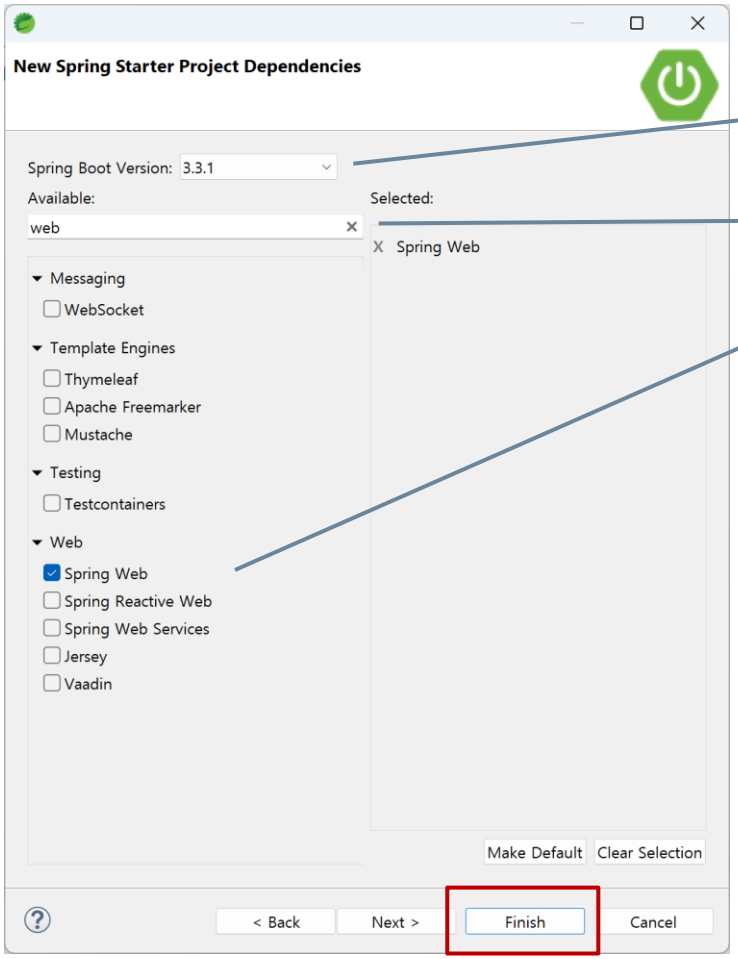
Gradle-Groovy : 프로젝트 관리 도구 선택

17 : 자바 버전 선택

스프링 부트 시작하기

스프링 부트로 프로젝트 만들기

스프링 부트 의존성 추가



3.3.1 : 스프링 부트 버전 선택

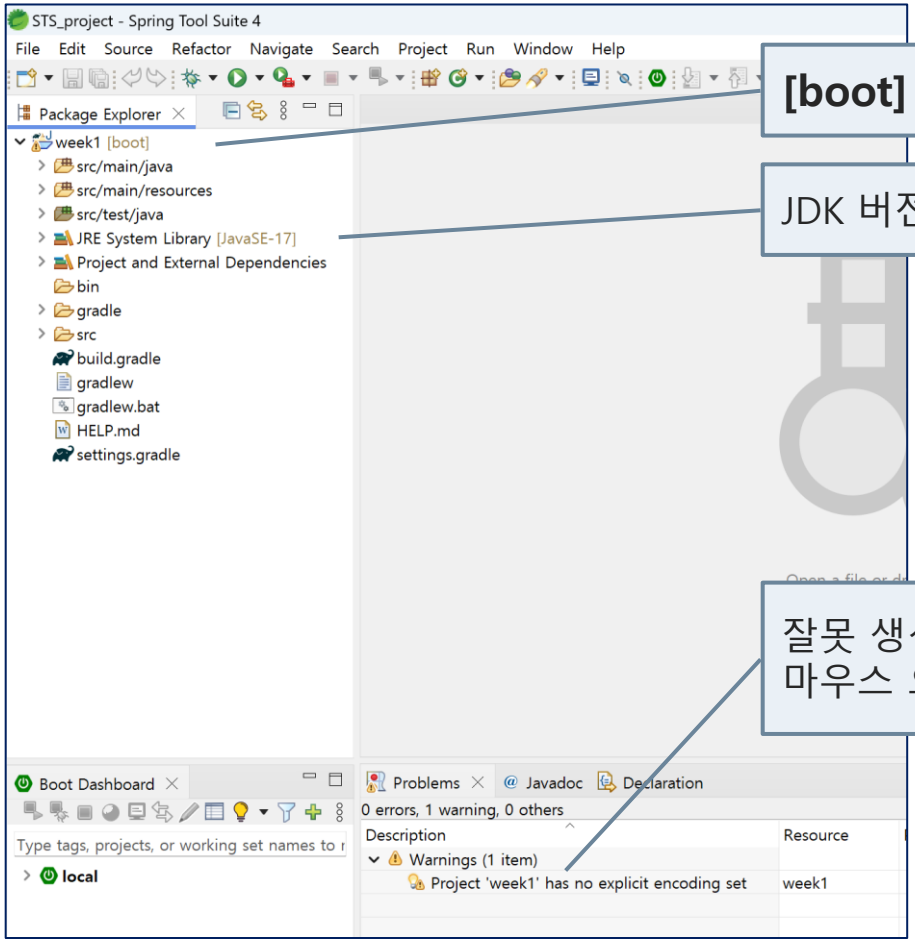
web : 검색어 입력

Spring Web 의존성 선택

스프링 부트 시작하기

스프링 부트로 프로젝트 만들기

스프링 프로젝트 생성 확인



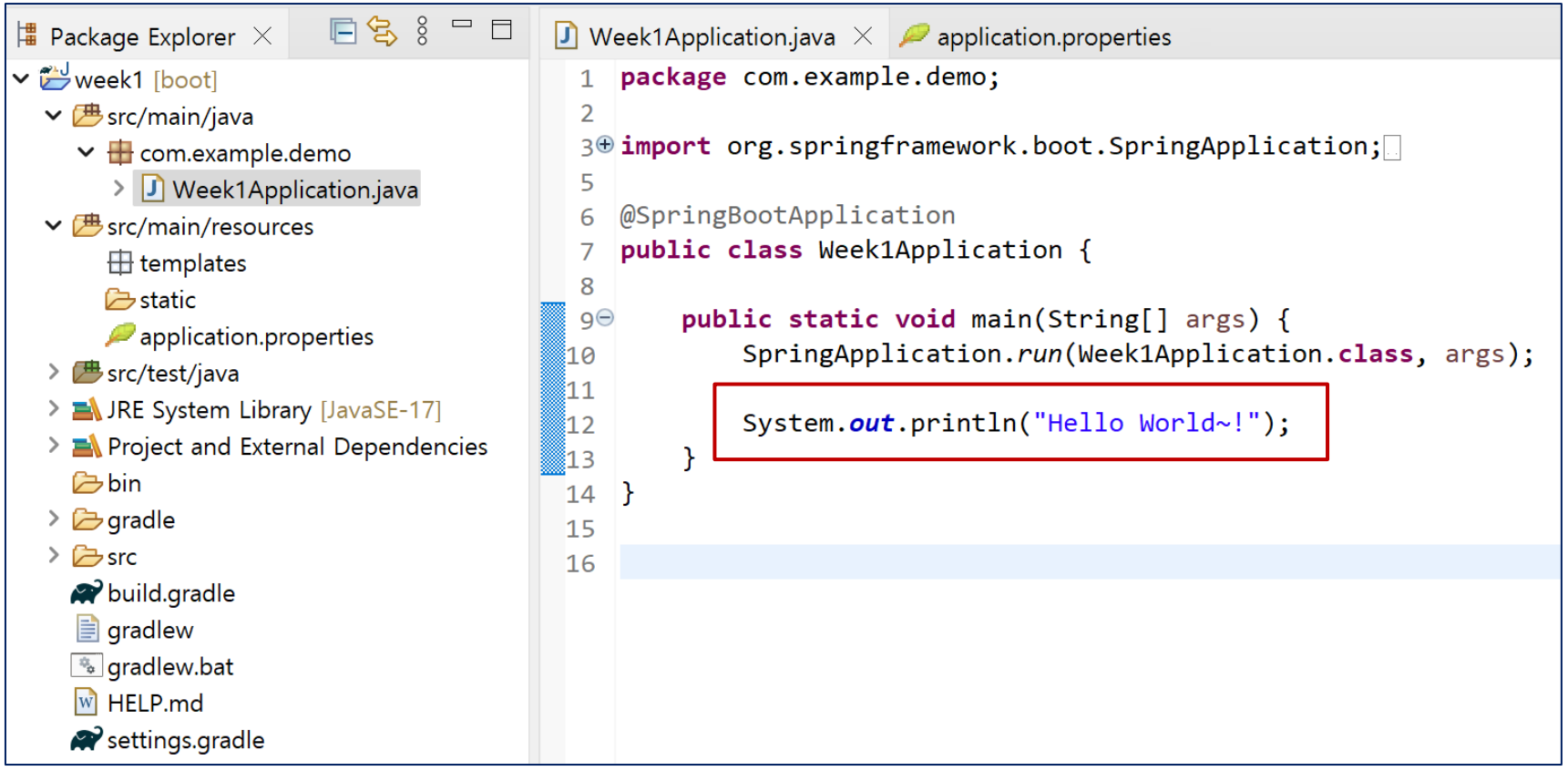
[boot] : 프로젝트 생성이 완료됨

JDK 버전 표시

잘못 생성된 경우 오류나 경고 표시
마우스 오른쪽 > 'Quick Fix' 적용 가능

스프링 부트 시작하기

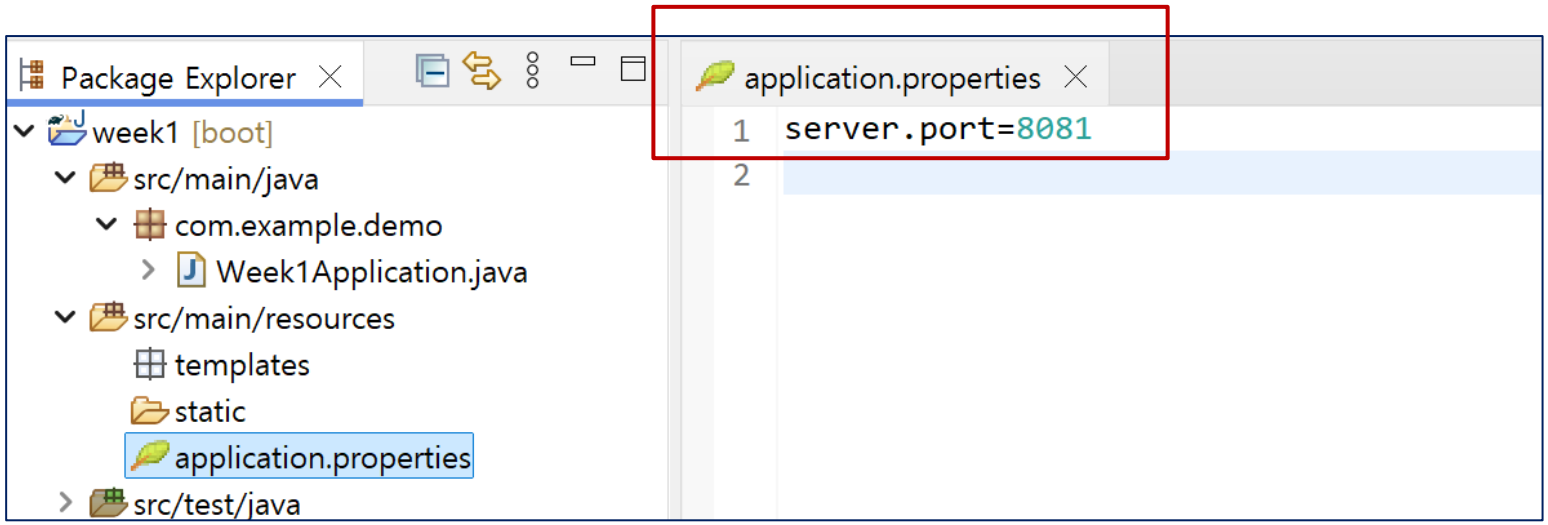
- **스프링 부트로 프로젝트 만들기**
  - 스프링 프로젝트 실행하기

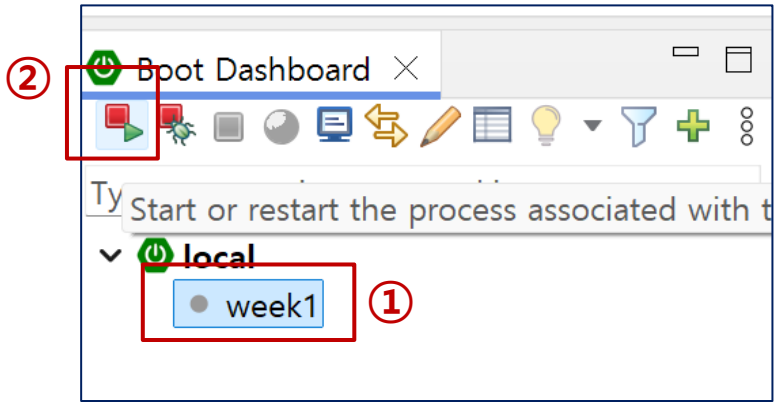


스프링 부트 시작하기

스프링 부트로 프로젝트 만들기

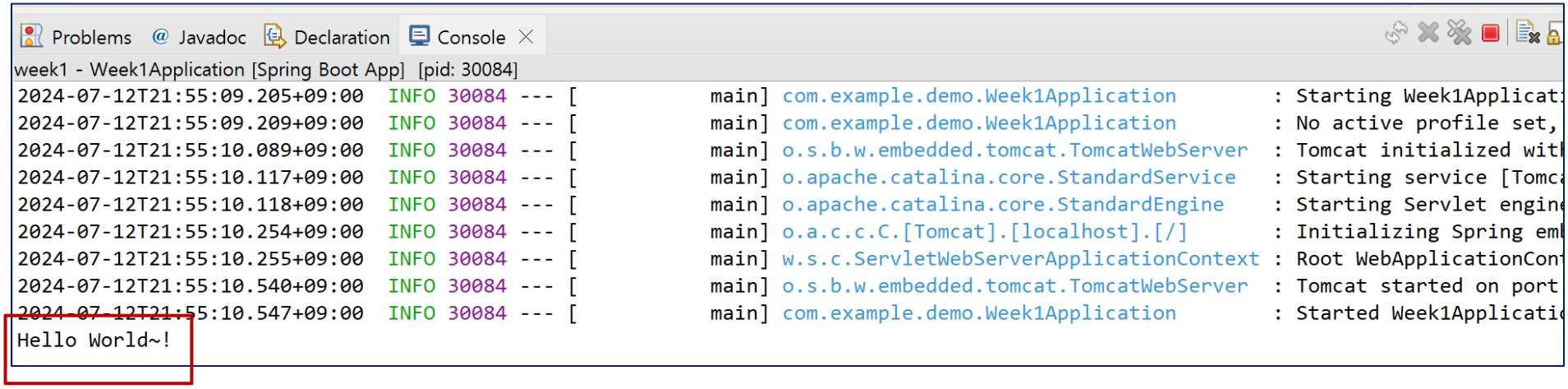
스프링 프로젝트 실행하기





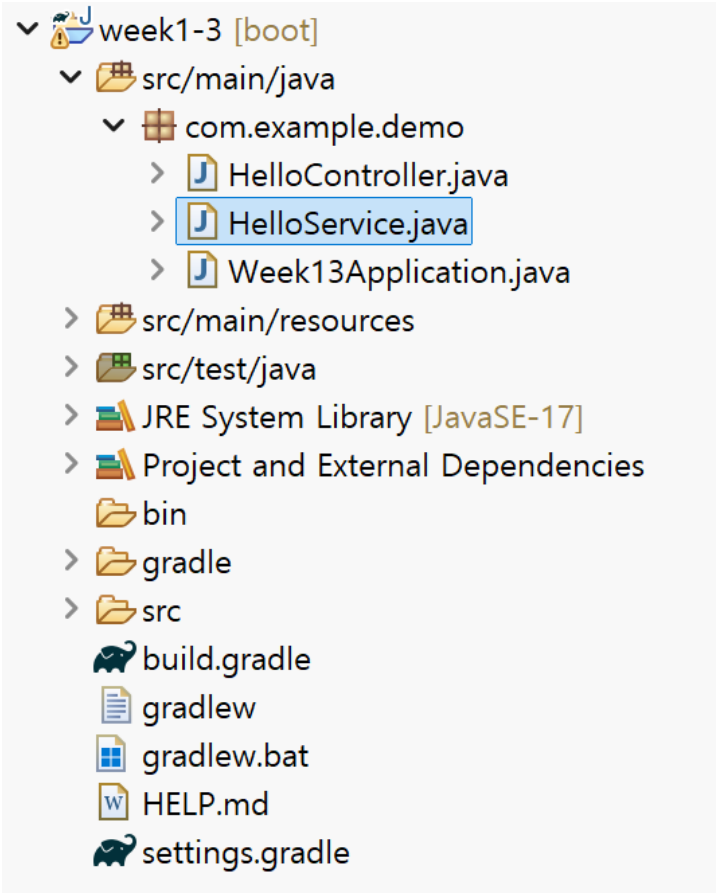
스프링 부트 시작하기

- **스프링 부트로 프로젝트 만들기**
  - 스프링 프로젝트 실행하기
    - 콘솔창 확인



어노테이션 @Component로 빈 등록 맛보기

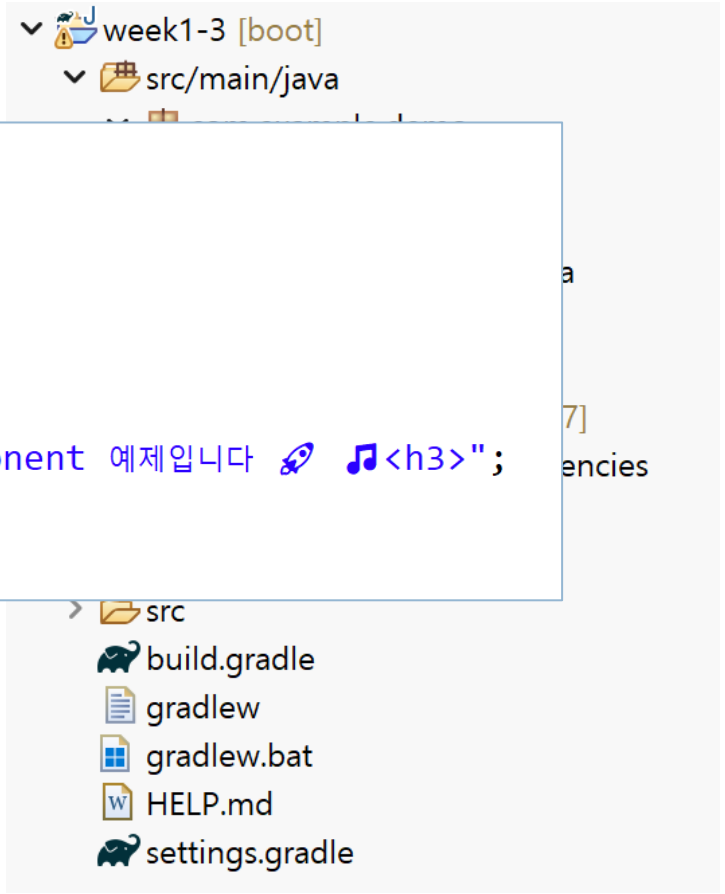
- **스프링 부트로 새 프로젝트 만들기**
  - spring web 의존성 추가



어노테이션 @Component로 빈 등록 맛보기

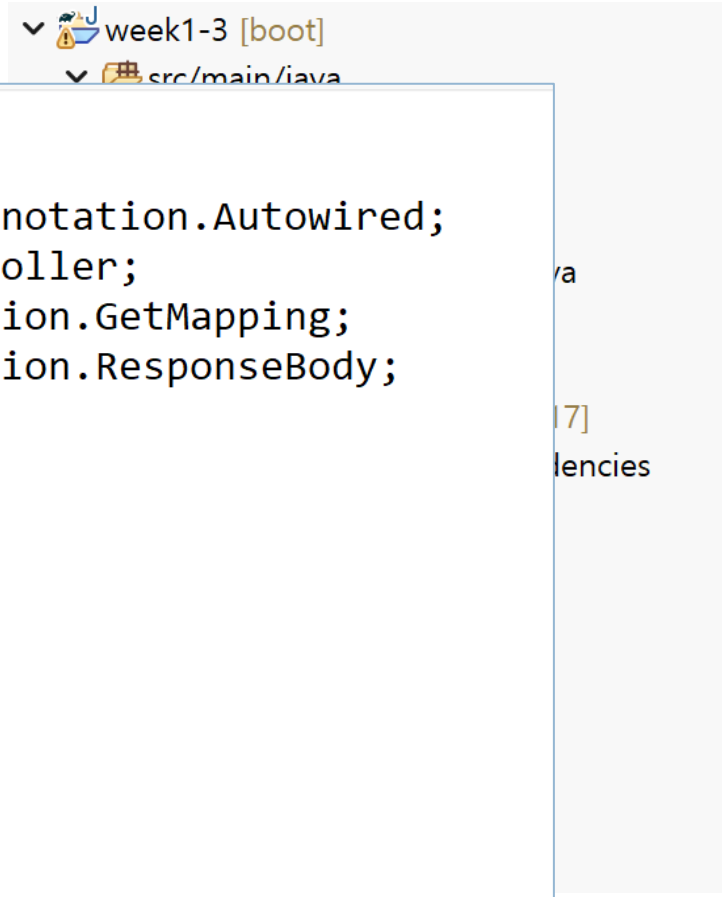
- **스프링 부트로 새 프로젝트 만들기**
  - spring web 의존성 추가

```java
package com.example.demo;

import org.springframework.stereotype.Component;

@Component
public class HelloService {
    public String sayHello() {
        return "<h2>안녕하세요! </h2><h3>스프링 부트 @Component 예제입니다 🚀 🎵<h3>";
    }
}
```



어노테이션 @Component로 빈 등록 맛보기

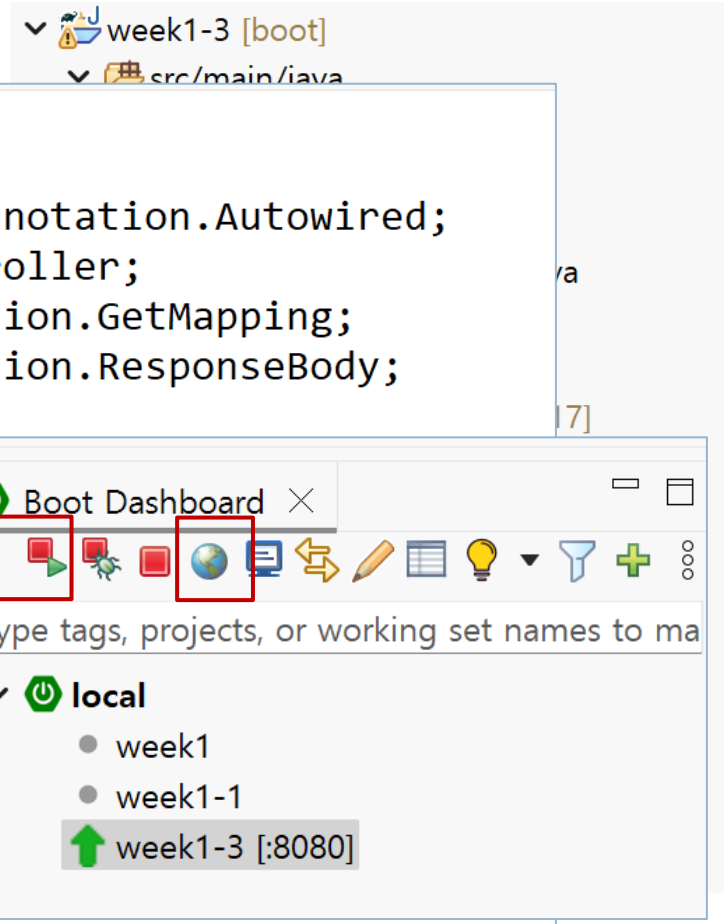
- **스프링 부트로 새 프로젝트 만들기**
  - spring web 의존성 추가




```java
package com.example.demo;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ResponseBody;

@Controller
public class HelloController {
    @Autowired
    private HelloService helloService;

    @GetMapping("/")
    @ResponseBody
    public String hello() {
        return helloService.sayHello();
    }
}
```

어노테이션 @Component로 빈 등록 맛보기

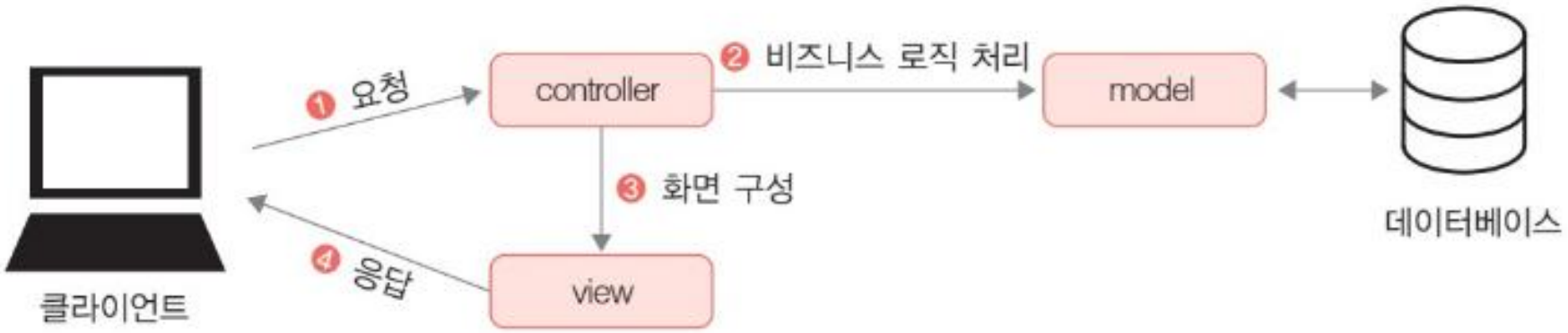
- **스프링 부트로 새 프로젝트 만들기**
  - spring web 의존성 추가




```java
package com.example.demo;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ResponseBody;

@Controller
public class HelloController {
    @Autowired
    private HelloService helloService;

    @GetMapping("/")
    @ResponseBody
    public String hello() {
        return helloService.sayHello();
    }
}
```

MVC 패턴 이해하기

- **MVC 개념**
  - **Model, View, Controller**를 이용해서 프로그래밍하는 설계 방법
  - 구성 요소
    - **Model** - **DB와 밀접한 관계**를 갖고 **비즈니스 로직**을 담당
    - **View** - **클라이언트와 밀접한 관계**를 갖고 비즈니스 로직의 결과를 출력하기 위한 **화면 구성**을 담당
    - **Controller** – **클라이언트의 요청**에 대해 Model과 View를 컨트롤하는 업무를 담당

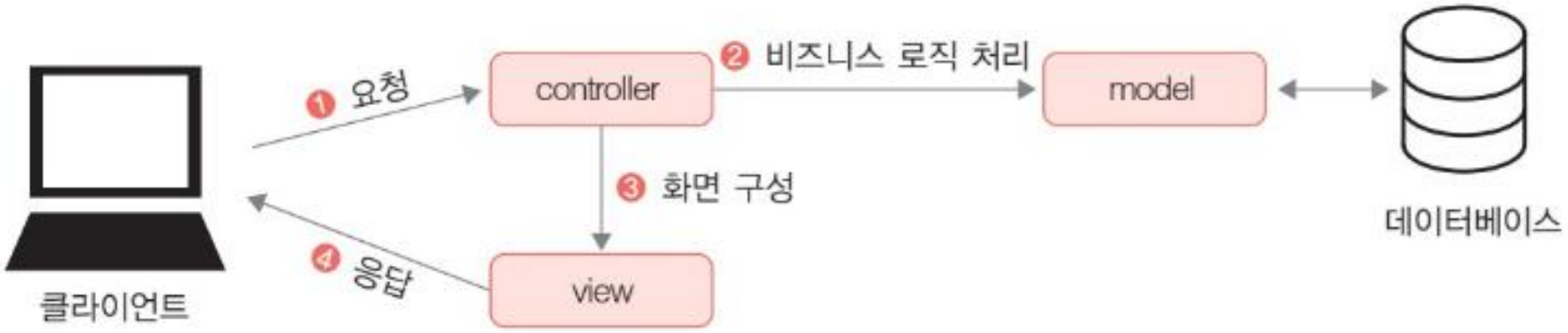


각각 업무를 분업화 하는 MVC 패턴

MVC 패턴 이해하기

- 스프링 부트 프로젝트 => 웹 프로젝트
  - 웹 **브라우저 주소창**에 **URL을 입력**하면(**요청** 단계)
  - **요청 형태에 따라 메서드 호출(로직 처리** 단계)
  - **로직 처리 결과를 보여줄 페이지(화면구성** 단계)가 호출되는 방식 (**응답** 단계)

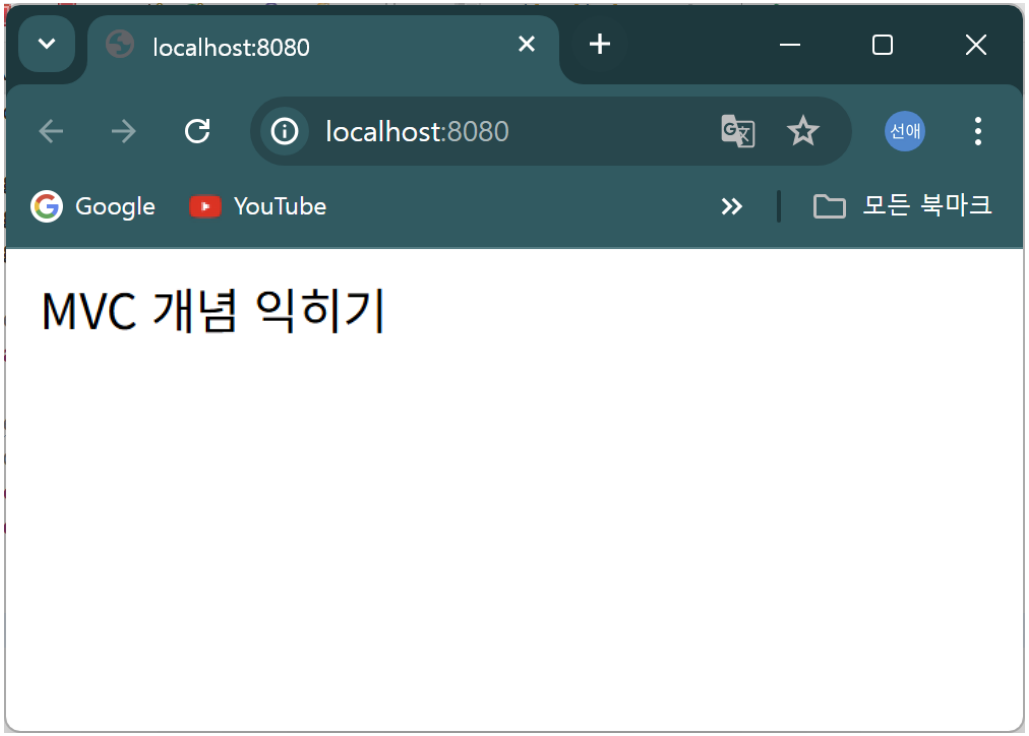
➡ **@Controller 어노테이션**을 적용한 클래스에서 처리



각각 업무를 분업화 하는 MVC 패턴

스프링 부트 프로젝트 실습

- MVC 개념 익히기
  - File > New > Spring Starter Project > **week9_02MVC**
  - **MyController** 클래스 생성
  - 프로젝트 실행 > 브라우저에 'MVC 개념 익히기' 출력



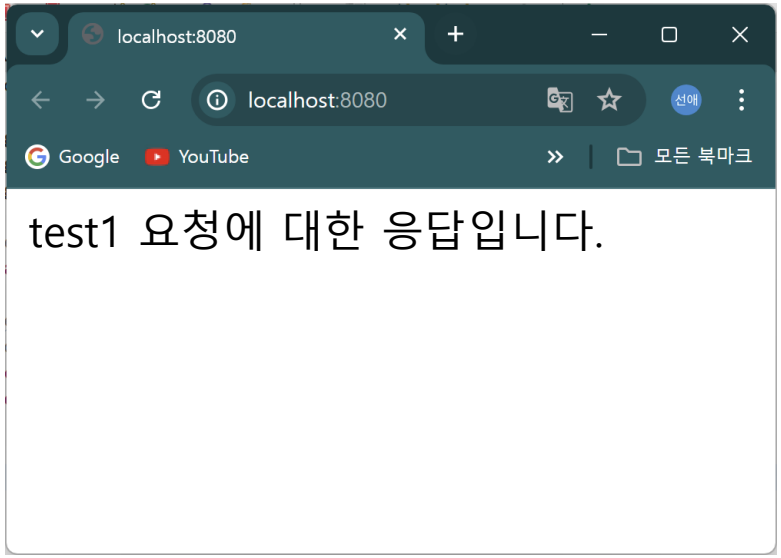
스프링 부트 프로젝트 실습

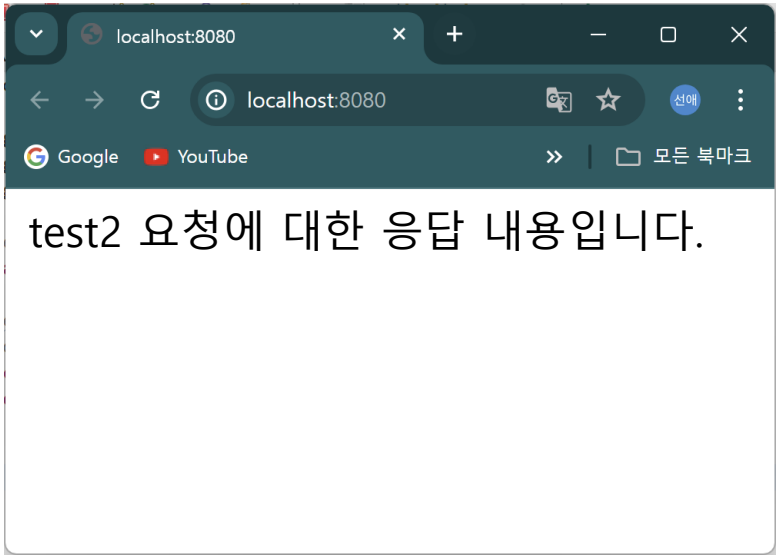
- MVC 개념 익히기
  - File > New > Spring Starter Project > **week9_02MVC**
  - **MyController** 클래스 생성

```
1  package com.example.demo;
2
3  import org.springframework.stereotype.Controller;
4  import org.springframework.web.bind.annotation.RequestMapping;
5  import org.springframework.web.bind.annotation.ResponseBody;
6
7  @Controller
8  public class MyController {
9      @RequestMapping("/")
10     @ResponseBody
11     public String root() {
12         return "MVC 개념 익히기";
13     }
14 }
15
```

스프링 부트 프로젝트 실습

- MVC 개념 익히기
  - File > New > Spring Starter Project > **week9_02MVC**
  - **MyController** 클래스 생성
  - 프로젝트 실행 > 브라우저에서 **'/test1' 요청**
  - 프로젝트 실행 > 브라우저에서 **'/test2' 요청**





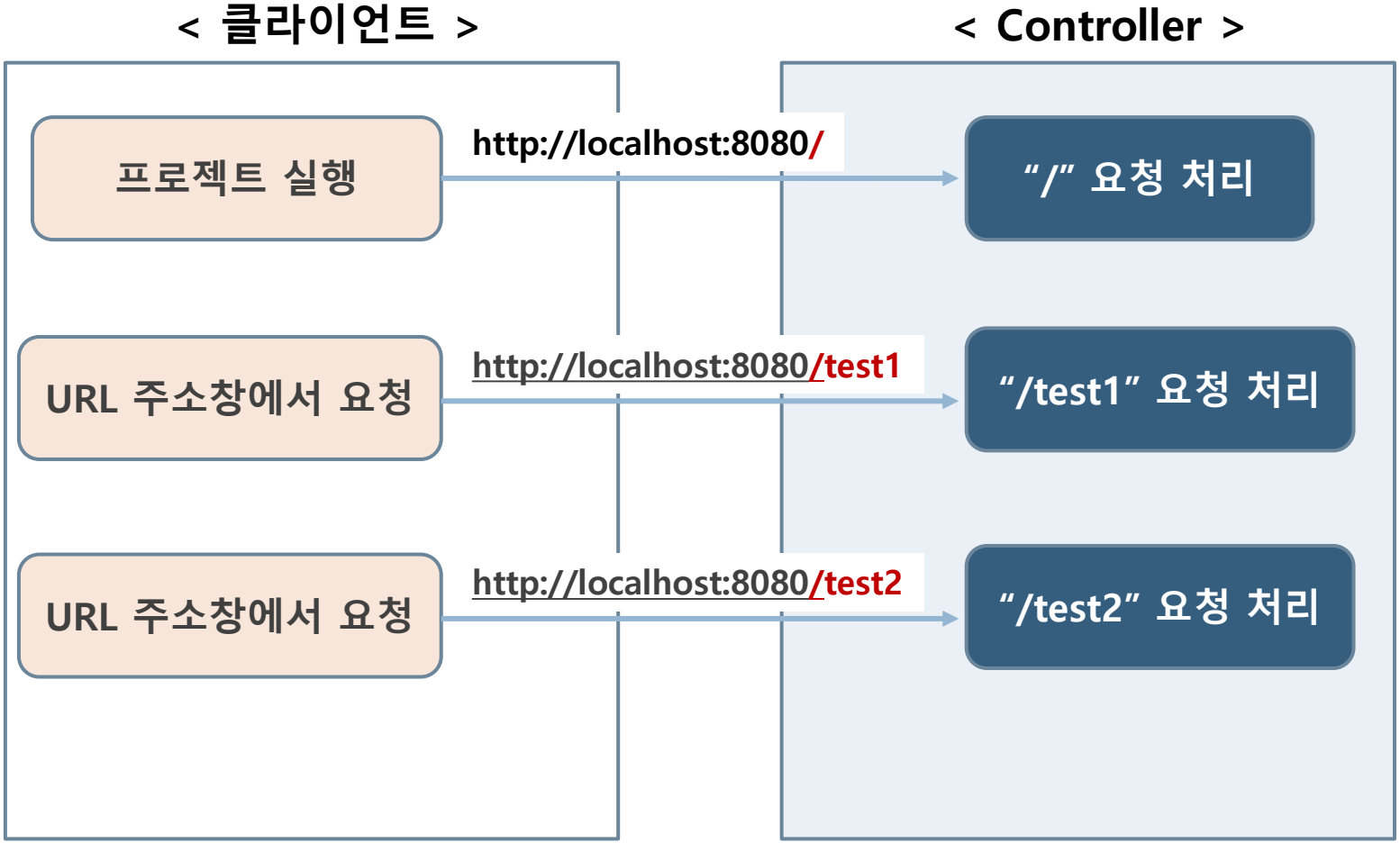
스프링 부트 프로젝트 실습

- MVC 개념 익히기

```
@Controller
public class MyController {
    @RequestMapping("/")
    @ResponseBody
    public String root() {
        return "MVC 개념 익히기";
    }

    @RequestMapping("/test1")
    @ResponseBody
    public String test1() {
        return "test1 요청에 대한 응답입니다.";
    }

    @RequestMapping("/test2")
    @ResponseBody
    public String test2() {
        return "test2 요청에 대한 응답 내용입니다.";
    }
}
```

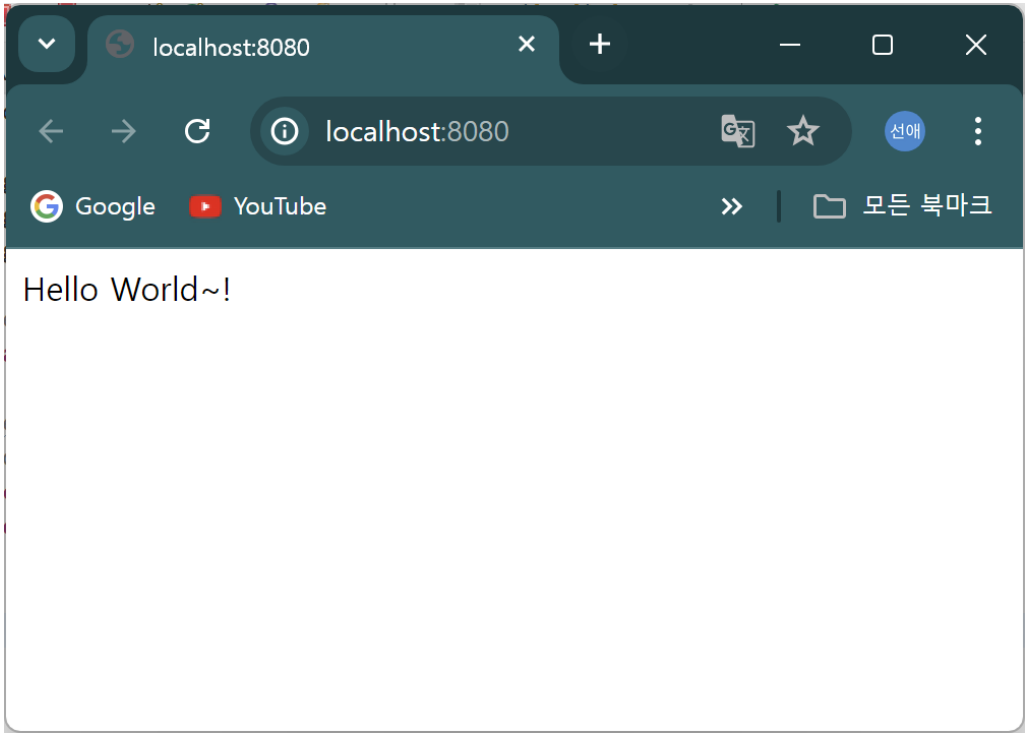
스프링 부트 프로젝트 실행 흐름도

- MVC 개념 익히기



스프링 부트 프로젝트 실습

- ‘Hello World~!’ 출력하기
  - File > New > Spring Starter Project > **week9_03AboutMe**
  - **MyController** 클래스 생성
  - 프로젝트 실행 > 브라우저에 ‘**Hello World~!**’ **출력**



스프링 부트 프로젝트 실습

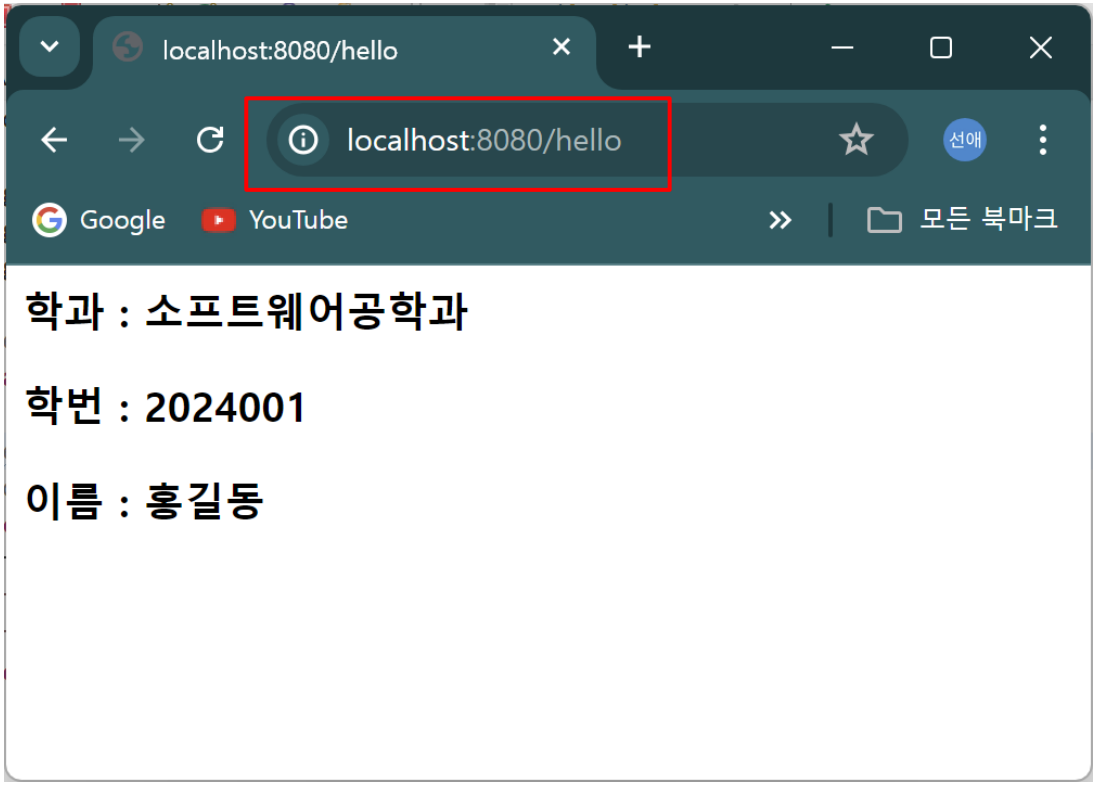
- ‘Hello World~!’ 출력하기
  - File > New > Spring Starter Project > **week9_03AboutMe**
  - **MyController** 클래스 생성
  - 프로젝트 실행 > 브라우저에 ‘Hello World~!’ 출력

MyController.java

```java
package com.study.springboot;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;

@Controller
public class MyController {

    @RequestMapping("/")
    @ResponseBody
    public String hello() {
        return "Hello World~!";
    }
}
```

스프링 부트 프로젝트 실습

- 개인 정보 출력하기
  - File > New > Spring Starter Project > **week9_03AboutMe**
  - **MyController 클래스** 생성
  - 프로젝트 실행 > 브라우저에서 **'/hello' 요청**



스프링 부트 프로젝트 실습

- 개인 정보 출력하기
  - File > New > Spring Starter Project > **week9_03AboutMe**
  - **MyController 클래스** 생성
  - 프로젝트 실행 > 브라우저에서 **'/hello' 요청**

MyController.java

```
1  package com.study.springboot;
2
3  import org.springframework.stereotype.Controller;
4  import org.springframework.web.bind.annotation.RequestMapping;
5  import org.springframework.web.bind.annotation.ResponseBody;
6
7  @Controller
8  public class MyController {
9
10     @RequestMapping("/hello")
11     @ResponseBody
12     public String hello() {
13         String str = "<h3>학과 : 소프트웨어공학과</h3>";
14         str += "<h3>학번 : 2024001</h3>";
15         str += "<h3>이름 : 홍길동</h3>";
16         return str;
17     }
18 }
```

과제물

최종 결과 화면(URL 포함)
이미지 파일제출

자기 소개하기

- File > New > Spring Starter Project > **week9_04Introduce**
- **MyController 클래스** 생성
- 프로젝트 실행 > 브라우저에서 **'/' 요청**
  - **"안녕하세요, 홍길동입니다."**
- 브라우저에서 **'/intro' 요청**
  - **"나는 25살이고, MBTI는 XXXX입니다."**
- 브라우저에서 **'/sports' 요청**
  - **"좋아하는 운동은 XX입니다. "**