

**초격차 패키지 Online.**

# 버전별 변천사

## **PART1** | 버전별 변천사 훑어보기: 1 vs 2

무엇이 달라졌나요?

## **PART2** | 버전별 변천사 훑어보기: 2.1 ~ 2.3

2.1, 2.2, 2.3 지나온 길 돌아보기

## **PART3** | 버전별 변천사 훑어보기: 2.4

2.4 에서 더 좋아진 것들

## **PART4** | 버전별 변천사 훑어보기: 2.5

2.5 에서 더 좋아진 것들

# 버전별 변천사

## 1 버전별 변천사 훑어보기: 1 vs 2

## Spring Boot 1 vs. 2

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

### 주요 변경 사항을 체크하는 경로

- Spring Official Blog: <https://spring.io/blog/category/releases>
- Spring Boot Github: <https://github.com/spring-projects/spring-boot/wiki/Spring-Boot-2.0-Release-Notes>
- SpringOne Platform Presentation, 2017: <https://youtu.be/TasMZsZxLCA>
- Baeldung: <https://www.baeldung.com/new-spring-boot-2>
- Quora
- StackOverflow
- 각종 기술 블로그들

## Spring Boot 1 vs. 2

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

### 주요 변경 사항

- Java 8 (+Java 9) + Spring Framework 5
- 써드파티 라이브러리 업그레이드
- Reactive Spring
- Functional APIs
- Kotlin 지원
- Configuration properties
- Gradle 플러그인
- Actuator 변경점
- Spring Security

## Spring Boot 1 vs. 2

**1.**  
버전별 변경사항  
확인하기: 1 vs 2

### 기타 변경 사항

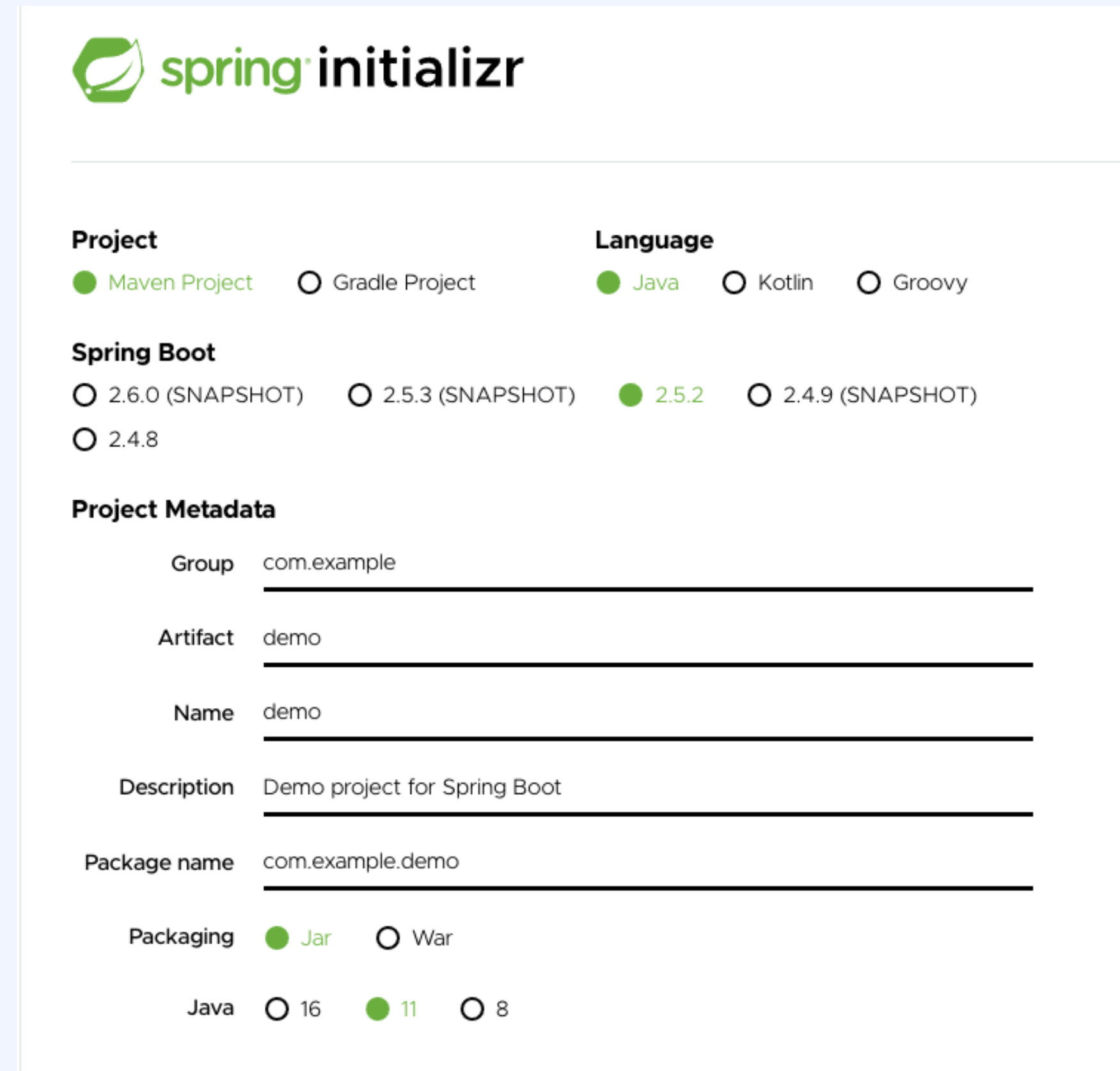
- Spring Boot Properties 변경사항
- Jackson 시간 표시 기본값
- MySQL auto\_increment
- HikariCP
- JOOQ
- GIF banner

# 주요 변경 사항

## Java 8 (+ Java 9) + Spring Framework 5

1.  
버전별 변천사  
훑어보기: 1 vs 2

- Java 8 이 이제 최소 사양
- Java 9 공식 최초 지원 - 부트 1.x 는 미지원
- Spring Framework 5



The image shows the Spring Initializr web form for creating a new project. The form is titled "spring initializr" and includes the following sections:

- Project:** Radio buttons for "Maven Project" (selected) and "Gradle Project".
- Language:** Radio buttons for "Java" (selected), "Kotlin", and "Groovy".
- Spring Boot:** Radio buttons for "2.6.0 (SNAPSHOT)", "2.5.3 (SNAPSHOT)", "2.5.2" (selected), and "2.4.9 (SNAPSHOT)". There is also a "2.4.8" option without a radio button.
- Project Metadata:** Text input fields for "Group" (com.example), "Artifact" (demo), "Name" (demo), "Description" (Demo project for Spring Boot), and "Package name" (com.example.demo).
- Packaging:** Radio buttons for "Jar" (selected) and "War".
- Java:** Radio buttons for "16", "11" (selected), and "8".

## 써드파티 라이브러리 업그레이드

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

- Tomcat 8.5
- Flyway 5
- Hibernate 5.2
- Thymeleaf 3
- Elasticsearch 5.6
- Gradle 4
- Jetty 9.4
- Mockito 2.x



## Reactive Spring

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

### 무엇을 위해 존재하는가?

- 한정된 자원 (thread pool) 으로
- 비동기(asynchronous) 논블록킹(non-blocking) 알고리즘을 이용해
- 다수의 요청에도 빠르고 예측 가능한 응답 성능(반응)을 실현

### 리액티브 지원 모듈

- Spring WebFlux
- Reactive Spring Data
- Reactive Spring Security
- Embedded Netty Server

## Functional APIs

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

- WebFlux.fn
- WebMvc.fn (Spring Framework 5.2)
- 기존의 스프링 웹애플리케이션을 함수형 스타일로 작성 가능
- 스프링 기술과 애노테이션에서 분리된 코드
  - 자바 코드 레벨에서 분석 가능
  - 독립적인 유닛 테스트 가능
  - 스프링 컨테이너에서 독립

```
RouterFunction<?> route = route(GET("/person/{id}"),
    request -> {
        Mono<Person> person = Mono.justOrEmpty(request.pathVariable("id"))
            .map(Integer::valueOf)
            .then(repository::getPerson);
        return ServerResponse.ok().body(fromPublisher(person, Person.class));
    })
    .and(route(GET("/person"),
        request -> {
            Flux<Person> people = repository.allPeople();
            return ServerResponse.ok().body(fromPublisher(people, Person.class));
        })
        .and(route(POST("/person"),
            request -> {
                Mono<Person> person = request.body(toMono(Person.class));
                return ServerResponse.ok().build(repository.savePerson(person));
            })
        ));
```

## Kotlin 지원

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

코틀린으로 본격적인 스프링 부트 프로그래밍을 시작



## Configuration properties

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

- 프로퍼티를 쓸 때: Relaxed binding 은 여전히 지원
- 프로퍼티를 읽을 때: 양식이 통일됨
  - 엘리먼트 구분: "."
  - 영어 소문자 + 숫자
  - 단어 구분자로 "-" 사용 가능
- 환경변수(environment variables)에서 컬렉션 데이터의 인덱스 표현 가능
  - MY\_VAR\_1= a -> my.var[1] = "a"
  - MY\_VAR\_1\_2= b -> my.var[1][2] = "b"
- 더 편리한 자료형 인식 (ex: java.time.Duration -> "1s", "2m", "5d")
- Origin 지원: 스프링 부트가 읽은 프로퍼티의 위치를 기억하고, 에러가 나면 알려줌
  - ex: "origin": "class path resource [application.properties]:1:27"

```
my.foo.hello-world    → my.foo.helloworld
my.foo.helloWorld
my.foo.hello_world
```

## Gradle 플러그인

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

- 최소 버전: 4.x
- bootRepackage -> bootJar & bootWar
- dependency management 기능을 사용하려면, 플러그인을 명시해야 함

## Actuator 변경점

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

- 보안성 강화: 1.5 에서 기본으로 보여주던 endpoint 를 더이상 보여주지 않음
- @Endpoints: 커스텀 endpoint 를 환경(MVC, JMX, Jersey..)에 상관 없이 편하게 구현
- 이름 변화
  - /autoconfig -> /conditions
  - /trace -> /httptrace

## Spring Security

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

- OAuth 2.0 통합
- 커스텀 설정이 더 쉬워짐
- WebSecurityConfigurerAdapter 순서 문제 해결
  - 기본 설정이 하나로 통합됨
  - WebSecurityConfigurerAdapter 를 추가하면 기본 설정이 꺼짐
- 보안이 중요한 기능들은 명시적으로 작성하게끔 변경 (ex: actuator)

# 기타 변경 사항



## Spring Boot Properties 변경사항

**1.**  
버전별 변경사항  
확인하기: 1 vs 2

- 이름과 구성에 변화
- spring-boot-properties-migrator
- JdbcTemplate 제어 옵션 추가: spring.jdbc.template.\*
- Redis 제어 옵션 추가: spring.cache.redis.\*

## MySQL auto\_increment

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

- Spring Data JPA, @GeneratedValue strategy 기본 동작이 바뀜
- 기본값: GenerationType.AUTO
  - Spring Boot 1.5: MySQL AUTO == IDENTITY
  - Spring Boot 2.0: MySQL AUTO == TABLE

## HikariCP

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

- Database 커넥션 풀 관리 프레임워크
- Tomcat Pool -> HikariCP

## JOOQ

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

- Java Object Oriented Querying
- Datasource 에 맞게 JOOQ dialect 자동 설정
- @JooqTest 지원
- 국내에서는 QueryDSL 에 밀리는 분위기?..

# GIF banner

**1.**  
버전별 변천사  
훑어보기: 1 vs 2

- 직접 보시죠