Spring Boot 4

What's new, and how to migrate from Spring Boot 3 to 4

The first major release since 2022 brings one module per technology, Jackson 3, JSpecify null safety and built-in API versioning. Most of the upgrade is renaming. Watch out for the few changes that compile fine but behave differently.

pom.xml — 3.5 to 4.0

```
  <parent>
    <artifactId>spring-boot-starter-parent</artifactId>
-   <version>3.5.5</version>
+   <version>4.0.0</version>
  </parent>

- <artifactId>spring-boot-starter-web</artifactId>
+ <artifactId>spring-boot-starter-webmvc</artifactId>
- <artifactId>flyway-core</artifactId>
+ <artifactId>spring-boot-starter-flyway</artifactId>
```

November 2025
Spring Boot 4.0.0
Released 20 November, a week after Spring Framework 7.0, alongside Spring Data 2025.1 and Spring Cloud 2025.1. [3, 4]

January 2026
Spring Cloud 2025.1.1
Use it with Spring Boot 4.0.1 and later. Spring Cloud 2025.1.0 only works with 4.0.0. [11]

June 2026
Spring Boot 4.1.0 [13]

June 2026
Last free 3.5 release
Spring Boot 3.5.16 is the last free 3.x release. Later 3.x fixes need commercial support. [15]

November 2026
Spring Boot 4.2 RC [14]
Final release not published yet.

What's new

One module per technology

The old auto-configuration jar had grown to 2 MiB, and every class and property in it showed up in your IDE. Now each technology gets its own module, starter and test starter, all following the naming scheme below. [1, 4]



Module	`spring-boot-flyway`
Starter	`spring-boot-starter-flyway`
Test starter	`spring-boot-starter-webmvc-test`
Package	`org.springframework.boot.flyway`

Jackson 3 by default

New group ID and package (`tools.jackson`), immutable mappers, unchecked exceptions, and support for `java.time` without an extra module. [1]

```
// Spring Boot 3.x
import com.fasterxml.jackson.databind.ObjectMapper;
// Spring Boot 4
import tools.jackson.databind.json.JsonMapper;
import com.fasterxml.jackson.annotation.JsonProperty; // unchanged
```

- The auto-configured mapper is now a `JsonMapper`. A custom `ObjectMapper` bean no longer replaces it, so declare your own `JsonMapper` bean instead.
- Every Jackson module on the classpath is registered automatically. Turn this off with `spring.jackson.find-and-add-modules=false`.
- Not ready to switch? The `spring-boot-jackson2` module lets you stay on Jackson 2 while you migrate. It is deprecated and will be removed in a later release.

API versioning

Map one path to several versions. The version can come from a header, a query parameter, the path or the media type. [7]

```
@GetMapping(path = "/orders", version = "1.0")
Order1 order1(...)

@GetMapping(path = "/orders", version = "1.1+")
Order2 order2(...)
```

```
spring.mvc.apiversion.use.header=X-Version
spring.mvc.apiversion.default=1.0.0
```

`"1.1+"` matches 1.1 and every later version. WebFlux uses `spring.webflux.apiversion.*`. On the client side, `RestClient`, `WebClient`, `WebTestClient` and MockMvc can send a version with each request.

HTTP service clients

Declare an interface with `@HttpExchange` methods and Spring generates the implementation. Put clients into groups and set the base URL and timeouts per group in properties. [8]

```
@HttpExchange("/orders")
interface OrderClient {
  @GetExchange("/{id}")
  Order get(@PathVariable String id);
}

@ImportHttpServices(group = "orders",
                    types = OrderClient.class)
```

```
spring.http.serviceclient:
  orders:
    base-url: https://orders.example.com
    read-timeout: 2s
```

Retry and concurrency limits

Retries and concurrency limits are now part of Spring Framework 7. In most apps you can drop the Spring Retry dependency. [9]

```
@Configuration
@EnableResilientMethods
class ResilienceConfig {}

@Retryable(includes = TimeoutException.class,
           maxRetries = 5, delay = 100,
           multiplier = 2)
public Quote fetchQuote() { ... }

@ConcurrencyLimit(10)
public void callLegacySystem() { ... }
```

Import from `org.springframework.resilience.annotation`, not Spring Retry's package of the same name. For programmatic retries, use `RetryTemplate` with a `RetryPolicy`. Spring Kafka and Spring AMQP now use this retry support instead of Spring Retry.

Null safety

All Spring projects now declare nullness with JSpecify. Spring's own `@Nullable` and `@NonNullApi` are deprecated. [1, 5]

```
// package-info.java
@NullMarked
package com.example.orders;
```

Switch to `org.jspecify.annotations.Nullable`. Kotlin reads these annotations, so Kotlin code that calls Spring APIs may need extra null handling to compile.

OpenTelemetry starter

`spring-boot-starter-opentelemetry` sets up the OpenTelemetry SDK and exports metrics and traces over OTLP. [2, 17]

Micrometer has its own modules now, so you can record metrics without pulling in the full Actuator.

Testing

`RestTestClient` is a blocking counterpart to `WebTestClient`. It works against MockMvc or a running server. [2]

Spring Boot now manages JUnit 6. Cached test contexts that sit idle are paused and restarted when a test needs them again. [6]

Spring Security 7

Adds multi-factor authentication. Spring Authorization Server is now part of Spring Security. [9]

The `and()` chaining method is removed. Configure each part with a lambda, such as `http.csrf(csrf -> ...).authorizeRequests` and `AntPathRequestMatcher` give way to `authorizeHttpRequests` and `PathPatternRequestMatcher`. The OAuth2 password grant is removed.

Also in 4.0

- `JmsClient`, a fluent JMS API like `JdbcClient` [2]
- Clients built on the JDK `HttpClient` use virtual threads when `spring.threads.virtual.enabled=true` [2]
- Spring Data repository code generated at build time (AOT) for faster startup [10]
- Multiple `TaskDecorator` beans are applied together, in `@Order` order [2]
- `spring-boot-starter-kotlin-serialization` for JSON with kotlinx.serialization [2]
- A `logging.console.enabled` switch to turn console logging off [2]

Migrating from 3 to 4

1. **Get to the latest 3.5.x** — Fix every deprecation warning. Anything deprecated in 3.x is gone in 4.0.
2. **Check your toolchain and dependencies** — Java 17 is still the minimum, and Java 25 is supported. Kotlin needs 2.2+, Gradle 8.14+. Spring Cloud needs 2025.1 (Oakwood). Confirm every other library supports Spring Boot 4.
3. **Switch to Spring Boot 4** — Add `spring-boot-properties-migrator` as a runtime dependency. At startup it logs renamed properties and temporarily maps them to the new names.
4. **Compile with classic starters** — `spring-boot-starter-classic` and `-test-classic` pull in every module. Fix moved imports and the Jackson 3 changes until it compiles.
5. **Move to focused starters** — Swap the classic starters for one starter and one test starter per technology.
6. **Test for changed behavior** — Run the full test suite, work through the dark box below, then remove the properties migrator.

OpenRewrite automates much of steps 1 to 5 with the recipe `org.openrewrite.java.spring.boot4.UpgradeSpringBoot_4_0`. [16]

Find and replace

The renames most projects hit. For properties, the migrator reports any others. [1]

Starters

`spring-boot-starter-web`	`spring-boot-starter-webmvc`
`spring-boot-starter-web-services`	`spring-boot-starter-webservices`
`spring-boot-starter-aop`	`spring-boot-starter-aspectj`
`spring-boot-starter-oauth2-client`	`spring-boot-starter-security-oauth2-client` Same pattern for resource-server and authorization-server

Annotations and classes

`@MockBean`, `@SpyBean`	`@MockitoBean`, `@MockitoSpyBean`
`Jackson2ObjectMapperBuilderCustomizer`	`JsonMapperBuilderCustomizer`
`@JsonComponent`, `@JsonMixin`	`@JacksonComponent`, `@JacksonMixin`
`HttpMessageConverters` bean (Spring Boot)	`ServerHttpMessageConvertersCustomizer`, `ClientHttpMessageConvertersCustomizer`
`org.springframework.boot.env.EnvironmentPostProcessor`	`org.springframework.boot.EnvironmentPostProcessor`
`hibernate-jpamodelgen`	`hibernate-processor`

Properties

`spring.data.mongodb.uri`, `host`, …	`spring.mongodb.uri`, `host`, …
`spring.jackson.read.*`, `write.*`	`spring.jackson.json.read.*`, `json.write.*`
`spring.session.redis.*`	`spring.session.data.redis.*`
`spring.dao.exceptiontranslation.enabled`	`spring.persistence.exceptiontranslation.enabled`
`management.tracing.enabled`	`management.tracing.export.enabled`

Removed in 4.0

Plan a replacement before you upgrade. [1]

- ~~Undertow~~ use Tomcat or Jetty
- ~~Executable jar launch scripts~~ `java -jar` still works
- ~~Classic jar loader~~ remove the `CLASSIC` loader setting
- ~~Spock integration~~ back in 4.1
- ~~Reactive Pulsar client auto-configuration~~
- ~~Spring Session Hazelcast and MongoDB~~ now maintained by the vendors
- ~~Spring Retry version management~~ declare the version yourself

Silent changes

No build error will warn you. Check each one before you ship. [1]

Flyway or Liquibase migrations don't run
The library alone no longer triggers auto-configuration. Add `spring-boot-starter-flyway` or `-liquibase`.

The H2 console is gone
Add its new module, `spring-boot-h2console`.

Batch job history disappears after a restart
`spring-boot-starter-batch` keeps metadata in memory. Use `spring-boot-starter-batch-jdbc`.

MockMvc or TestRestTemplate can't be injected
`@SpringBootTest` no longer adds them. Add `@AutoConfigureMockMvc` or `@AutoConfigureTestRestTemplate`.

`@Mock` fields are null
Spring Boot no longer registers `MockitoTestExecutionListener`, which used to initialize them. Add `@ExtendWith(MockitoExtension.class)`.

JSON reads or writes differently
Jackson 3 changes several serialization defaults. Compare your API responses with the 3.x output. `spring.jackson.use-jackson2-defaults=true` restores the old defaults while you test.

MongoDB UUIDs or decimals fail to map
Spring Boot no longer sets a default storage format for `UUID` and `BigDecimal`. Set `spring.mongodb.representation.uuid` and `spring.data.mongodb.representation.big-decimal`.

Health shows liveness and readiness groups
Liveness and readiness probes are now enabled by default. Turn them off with `management.endpoint.health.probes.enabled=false`.

Optional dependencies missing from the jar
The Spring Boot Maven plugin no longer packages optional dependencies. Set `<includeOptional>true</includeOptional>`.

After 4.0

Spring Boot ships a feature release every six months, and only recent ones get free fixes.

Spring Boot 4.1

Released 10 June 2026 [13]

- Spring gRPC support
- Context propagation for `@Async`
- `InetAddressFilter` for HTTP clients, to block requests to internal addresses (SSRF)
- Removes the APIs deprecated in 4.0, so clear 4.0 deprecation warnings first [12]

Spring Boot 4.2

Release candidate, November 2026 [14]

- Based on Spring Framework 7.1
- `RestTemplate` becomes formally deprecated. Plan the move to `RestClient`. [7]
- Optional OpenTelemetry naming for metrics and traces via `management.observations.conventions=open-telemetry`

Sources

1. Spring Boot 4.0 Migration Guide
2. Spring Boot 4.0 Release Notes
3. Spring Boot 4.0.0 available now
4. Modularizing Spring Boot
5. Spring Framework 7.0 Release Notes
6. Spring Framework reference: Resilience features
7. Spring Boot reference: Servlet web applications
8. Spring Boot reference: Calling REST services
9. What's new in Spring Security 7.0
10. Spring Data 2025.1 goes GA
11. Spring Cloud 2025.1.1 (Oakwood) released
12. Spring Boot 4.1 Release Notes
13. Spring Boot 4.1.0 available now
14. Spring Boot 4.2.0-RC2 Release Notes
15. Spring Boot 3.5.16 available now
16. OpenRewrite: Migrate to Spring Boot 4.0
17. OpenTelemetry with Spring Boot

Compiled in November 2026 from the official Spring documentation listed above. This is a community poster, not affiliated with the Spring team or Broadcom. Spring and Spring Boot are trademarks of Broadcom Inc.

Fonts: Metropolis, Work Sans, JetBrains Mono. Icons: Lucide.