Spring Framework 7 Upgrade — Client Technical Brief

Document type: Client-facing preparation guide
Distribution: Ahead of release cycle; publish to designated client-accessible location after internal approval
Product / platform: Fusion / SDMX Core stack (representative BOM: `sdmx-core-bom`, `mt-core-bom`)

Document control

Item	Detail
Version	1.0 DRAFT — replace after stakeholder session
Intended audience	Client technical owners, integration teams, operations, security
Related session	Alignment with subject-matter experts (e.g. session with Graham) to validate deployment assumptions and extension scenarios
Approvals required before publication	Development, QA, Product Owner, Release Management (RM)

---

1. Executive summary

This initiative upgrades the platform from **Spring Framework 5** to **Spring Framework 7**, together with aligned upgrades to **Spring Security**, **Jakarta EE** components (Servlet, Persistence), **Hibernate ORM**, testing frameworks, and numerous supporting libraries.

Clients should prepare for:

- A **minimum JDK requirement** (see §3): applications and integrations must run on a supported Java version consistent with this release.
- **Jakarta namespaces** (`jakarta.*`) instead of legacy `javax.*` for Servlet, JPA, injection, and related APIs wherever client-owned code touches the container or persistence layer.
- **Servlet 6.1–capable runtimes** and ecosystem alignment described below if clients host their own deployments or embed components.
- **Regression and integration validation** against their environments before production cutover.

This document explains implications at the **framework and dependency** level. Product-specific functional changes are communicated separately via release notes for each deliverable.

---

2. Spring Framework 7 — what changed vs earlier generations

Spring Framework 7 is a **major generation** beyond Spring 5.x. Compared to Spring 5 (and the intermediate Spring 6 line clients may have read about elsewhere), version 7:

Theme	Plain-language explanation
Baseline platform	Raises minimum levels for Servlet, JPA, and related Jakarta specifications (see §4). Expect closer alignment with **Jakarta EE 11** concepts in the ecosystem.
JDK	**Minimum JDK 17**; Spring also tests against current LTS versions (e.g. 21, 25). See Spring Framework 7.0 Release Notes.

Logging (JCL)	`spring-jcl` removed in favor of **Apache Commons Logging** 1.3.0 — typically transparent if you use standard logging facades (SLF4J / Logback).
Removed `javax.annotation` / `javax.inject`	Annotations such as `@Resource`, `@PostConstruct`, `@Inject` must use **`jakarta.annotation`** and **`jakarta.inject`** equivalents if still present in client code.
Removed MVC path-mapping options	Long-deprecated options (e.g. certain suffix patterns, trailing-slash behaviors, extension-based content negotiation) are **removed**. Custom URL mapping logic may need review.
Undertow	Spring's Undertow-specific integrations are **removed** until Undertow supports Servlet 6.1. Clients on Undertow must plan a **different Servlet 6.1 container** or wait for a compatible Undertow release.
HTTP API surface	`HttpHeaders` no longer implements `MultiValueMap`-style contracts in full; header handling code may need adjustment in rare custom cases.
Testing	JUnit Jupiter / `SpringExtension` behavior around **`@Nested` tests** and test context scope may affect custom test listeners; JUnit 4 support in Spring Test is **deprecated**.
Jackson	Spring documents **deprecation of Jackson 2.x support** in favor of Jackson 3.x in a future alignment — monitor release notes if you customize JSON binding heavily.
Other	Various removals (e.g. legacy `ListenableFuture`, older WebJars locator coordinates, OkHttp3 support in framework).

For authoritative detail, use the official **Spring Framework 7.0 Release Notes**.

---

3. JDK / Java requirements

3.1 Minimum supported version

Requirement	Value
Minimum JDK for Spring Framework 7	**Java SE 17**
Project compile / bytecode target (this codebase)	**17** (`maven.compiler.source` / `target` in parent POMs)

Clients **must** deploy on **JDK 17 or newer** that remains in security support for their organization.

3.2 Recommended / validated JDKs

Spring Framework 7 is tested on **LTS JDK releases** (17, 21, 25 per Spring documentation). Non-LTS JDKs may work on a **best-effort** basis only.

Client actions:

1. Confirm **runtime JDK** (containers, VMs, application servers) is **≥ 17**.
2. Align **CI/CD build agents** and developer workstations with the same minimum.
3. Retire **JDK 8 / 11** for builds and deployments of this release line.

---

4. Dependency matrix (aligned with Spring 7 initiative)

The following table reflects **central versions** managed in `sdmx-core-bom` (and `mt-core-bom` for MT-specific overrides). Exact patch versions may be updated in maintenance releases; clients should rely on **published BOM version** accompanying each release artifact.

4.1 Core Spring and security

Component	Managed version (indicative)	Notes
Spring Framework (`spring-*` artifacts)	**7.0.2**	Core upgrade driver
Spring Security	**7.0.2**	Aligned with Spring 7

4.2 Jakarta EE & web

Component	Managed version (indicative)	Notes
Jakarta Servlet API	**6.1.0**	Matches Spring 7 baseline for Servlet **6.1**
Jakarta Persistence API	**3.2.0**	JPA **3.2**

Spring's own upgrade notes cite **Servlet 6.1** (e.g. Tomcat 10+ Jetty 12.1 class of containers). Clients must ensure **embedded or external servlet containers** support Servlet **6.1**.

4.3 Persistence & data

Component	Managed version (indicative)	Notes
Hibernate ORM (`hibernate-core`)	**7.2.0.Final**	Aligns with JPA 3.2 / Spring 7 expectations

4.4 Integration & messaging

Component	Managed version (indicative)	Notes
Spring for Apache Kafka (`spring-kafka`)	**4.0.1**	
Spring Integration AMQP (MT BOM)	**7.0.0-M3**	**Milestone** — validate messaging in staging

4.5 JSON, HTTP client, JAX-RS

Component	Managed version (indicative)	Notes
Jackson (`jackson-core` / `jackson-databind`)	**2.20.1**	

Apache HttpComponents (httpclient / httpcore)	**4.5.14** / **4.4.16**	
Jersey (JAX-RS)	**4.0.0-M2**	**Milestone** — review if client code uses JAX-RS alongside Spring

4.6 Testing

Component	Managed version (indicative)	Notes
JUnit (BOM import + `junit-jupiter`)	**6.0.0**	JUnit **6** on the test classpath
Mockito	**5.23.0**	

4.7 Other notable libraries (non-exhaustive)

Examples from the same BOMs: **Guava** 33.4.8-jre, **Apache Commons** (IO, Lang3, DBCP2, FileUpload), **Log4j-to-SLF4j** 2.25.2, **Logback** 1.2.13, **Apache POI** 5.4.0, **Lucene** 8.11.4, **Antlr** 4.13.2, **MongoDB driver** 3.12.x (legacy coordinate), **H2 MVStore**, **Kryo**, etc.

Client takeaway: Custom extensions or classpath shading must remain **compatible** with these generations; duplicated older Spring / Hibernate / Servlet JARs on the classpath are a common source of failures.

---

5. Compatibility considerations, deprecations, and breaking changes (client-relevant)

5.1 High impact

Topic	Impact	Mitigation
JDK < 17	Application may not start or may fail bytecode verification	Upgrade runtime to JDK **17+**
Servlet container < 6.1	Deployment failures or missing APIs	Upgrade Tomcat / Jetty / etc. per vendor guidance for Servlet **6.1**
`javax.*` in client code (servlets, JPA, injection)	Compile or runtime errors	Migrate to `jakarta.*`
Undertow as primary container	Unsupported Undertow-specific Spring integrations removed	Migrate to Tomcat / Jetty / other Servlet 6.1 server, or await Undertow support
Custom Spring MVC path / content negotiation	Possible 404s or negotiation changes	Retest all URL patterns; remove reliance on removed mapping options

5.2 Medium impact

Topic	Impact	Mitigation
Custom `HttpHeaders` usage	Subtle behavioral differences	Refactor per Spring 7 Javadoc / release notes

JUnit / Spring tests (`@Nested`, custom listeners)	Test failures	Apply `@SpringExtensionConfig` or documented properties; update listeners
CORS	Empty CORS config may change pre-flight behavior	Retest browser clients and API gateways
GraalVM native images	Metadata format changes	Update reachability metadata if using native compilation

5.3 Lower impact (awareness)

- `RestTemplate` — long-term direction is **REST clients** documented in Spring; plan migrations for new code.
- **JUnit 4** Spring test support — **deprecated**; move to JUnit Jupiter.
- **Jackson 2** — framework signals move toward **Jackson 3** over time; watch release notes.

---

6. Client impact assessment

Client profile	Typical impact	Preparation
Consumes only public HTTP/REST APIs	Low	Run **contract / regression tests**; verify auth, headers, CORS
Embeds vendor JARs or uses shared classpath	Medium	Resolve **duplicate** Spring/Hibernate/Jackson versions; enforce BOM
Custom servlets, filters, JSPs, WAR deployment	Medium–High	**Jakarta** packages, Servlet **6.1**, server upgrade
Custom JPA entities, native queries, Hibernate APIs	Medium–High	Validate against **Hibernate 7** / JPA 3.2
LDAP / Spring Security extensions	Medium	Retest auth flows; review Spring Security **7** config
Kafka / AMQP integrations	Medium	End-to-end tests; note **milestone** Spring Integration AMQP in MT BOM
CI pipelines	Low–Medium	JDK 17+, updated Surefire / reporting

---

7. Configuration, deployment, and migration steps

7.1 Mandatory client actions

1. **Runtime:** Install and certify **JDK 17 or newer** on all target environments.
2. **Application server / Kubernetes images:** Upgrade base images and ingress stacks to versions supporting **Servlet 6.1** where applicable.
3. **Code:** Replace remaining `javax.servlet`, `javax.persistence`, `javax.annotation`, `javax.inject` imports with `jakarta.*` in **client-maintained** code.
4. **Dependencies:** Align internal libraries with the **published BOM** for the release; avoid mixing Spring **5** or **6** artifacts with Spring **7**.

5. **Reverse proxies / API gateways:** Retest routing, TLS, WebSockets, and CORS after upgrade.

7.2 Recommended actions

1. Refresh **staging** data and run **full regression** plus **integration** suites.
2. Review **logging** configuration (Commons Logging / SLF4J / Logback) if custom appenders assumed JCL internals.
3. If using **native image** (GraalVM), apply **Spring 7 native metadata** guidance from release notes.
4. Schedule a **technical walkthrough** with your account / services team if you have unusual embedding scenarios.

7.3 Deployment considerations

- **Blue/green or canary** deployments are recommended for first production rollout.
- **Rollback plan:** Keep previous release artifacts and JDK configuration documented until the new line is stable in production.
- **Monitoring:** Watch for startup errors mentioning `NoClassDefFoundError`, `javax/`, or version skew between Spring modules.

---

8. Testing guidance

Layer	Guidance
Smoke	Health checks, login, critical user journeys, batch windows
Regression	Full automated suite against **JDK 17+** staging
Integration	Partner systems, SSO, LDAP, Kafka, REST/SOAP consumers, scheduled jobs
Performance	Baseline comparison on representative load; JVM tuning may shift slightly after JDK change
Security	AuthZ paths, CSRF (if applicable), session fixation, headers
Browser / API clients	CORS and preflight if web apps span origins

Coordinate **QA sign-off** per your organization's release policy; this document supports but does not replace formal test plans.

---

9. Timeline and release expectations

> ***Placeholder — replace with program-approved dates.***

Milestone	Target / status
Client brief published (this document)	TBD
Release candidate / early access environment	TBD
General availability (GA) of upgraded deliverables	TBD
End of support for previous major dependency line	TBD (per RM policy)

Expectations:

- Clients receive **advance notice** per contractual or operational agreements.
- **Patch releases** may update dependency patch levels without a full announcement cycle; subscribe to release communications.

---

10. Supporting diagram — logical dependency stack

```
flowchart TB
    subgraph Runtime["Client runtime"]
        JDK["JDK 17+"]
        Container["Servlet 6.1 container"]
    end

    subgraph Platform["Fusion / SDMX platform"]
        SF["Spring Framework 7.x"]
        SS["Spring Security 7.x"]
        HI["Hibernate 7.x / JPA 3.2"]
        Jakarta["Jakarta Servlet / Persistence APIs"]
    end

    JDK --> SF
    JDK --> SS
    Container --> Jakarta
    SF --> Jakarta
    SF --> HI
    SS --> SF
```

---

11. References (external)

- Spring Framework 7.0 Release Notes
- Spring Framework 7.0 General Availability announcement (verify current URL on spring.io)

---

12. Appendix A — Summary for Release Notes (copy-paste block)

Spring Framework 7 upgrade: The platform now targets **Spring Framework 7** and **Spring Security 7**, with **Jakarta Servlet 6.1**, **JPA 3.2**, and **Hibernate ORM 7**. **JDK 17** is the minimum supported Java version. Clients must run on a **JDK 17+** runtime and, where they host the software, use a **Servlet 6.1–compatible** application server. Code using legacy `javax.*` Servlet, Persistence, or injection APIs should migrate to `jakarta.*`. Several Spring 5–era APIs and mapping options are removed; clients with custom Spring MVC, security, or Hibernate integrations should complete **regression and integration testing** before production deployment. See `docs/client-facing/README.md` (Spring Framework 7 Upgrade — Client Technical Brief) for the full dependency matrix and migration checklist.

---

13. Appendix B — Internal publication checklist

Use this checklist before sharing with clients.

Step	Owner	Done

Technical accuracy reviewed against BOM and build	Dev	☐
Test strategy and risk areas reviewed	QA	☐
Product messaging and scope aligned	PO	☐
Session with SME completed (e.g. Graham — deployment edge cases, messaging milestones)	Dev / RM	☐
Release timeline section filled	RM	☐
Final approval for client distribution	RM	☐
Published to **designated client-accessible location**	RM / Ops	☐

---

14. Appendix C — Version lineage snapshot (repository)

For internal traceability (versions may drift slightly in SNAPSHOT BOMs):

- `sdmx-core-bom` property `spring.version` : **7.0.2**
- `sdmx-core-bom` property `spring-security.version` : **7.0.2**
- Parent `sdmx-core` / `mt-core` Java compiler level: **17**

End of document.