Native Support in Spring Boot 3

Stéphane Nicoll
@snicoll[@mastodon.online]
stephane.nicoll@broadcom.com

Touraine Tech, February 2024

vmware

Agenda

- Why compile to Native?
- Support strategies for Frameworks
- AOT processing in Spring
- Will this work with my app?
- Developer cookbook

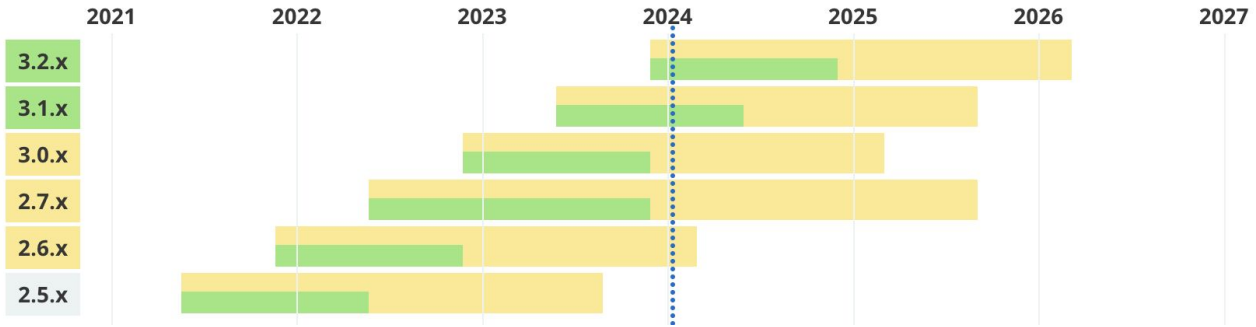
spring

Support Timeline

Spring Boot 2.7.x support

Branch	Initial Release	End of Support	End Commercial Support *
3.2.x	2023-11-23	2024-11-23	2026-02-23
3.1.x	2023-05-18	2024-05-18	2025-08-18
3.0.x	2022-11-24	2023-11-24	2025-02-24
2.7.x	2022-05-19	2023-11-24	2025-08-24
2.6.x	2021-11-17	2022-11-24	2024-02-24
2.5.x	2021-05-20	2022-05-19	2023-08-24

More



spring

Migration pre-steps

1. Upgrade to the latest 2.7.x release (2.7.18).
2. Upgrade to Java 17 at least (why not 21?).
3. Check deprecations in code.
4. Check deprecations in application properties (properties migrator).
5. Using Spring Security? Upgrade to Spring Security 5.8 first.
6. Check for warnings in your logs!
7. Check for dependency version overrides / Use our BOM

Migrate to Spring Boot 3

You are now ready to get started!

- Upgrade to the **latest** 3.**0**.x (3.0.13).
- Each feature release has dedicated release notes.
- Each major release has a specific migration guide.
- The release notes link to more specific release notes and/or migration guides for the Spring Modules you might be using.
- Upgrade to the latest 3.**1**.x (3.1.8 at this time) and follow the dedicated release notes.
- One more time and you are on 3.2.x!

Why compile to Native?

spring

GraalVM advantages

Instant startup

Milliseconds for native instead of seconds for the JVM

No warmup

Peak performance available immediately

Low resource usage

Lower memory footprint and no JIT compilation

Reduced surface attack

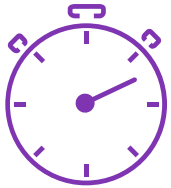
Closed world of dependencies with explicit reflection and serialization

Compact packaging

Smaller container easier to deploy

spring

GraalVM trade-offs



Very slow compilation

Minutes instead of seconds

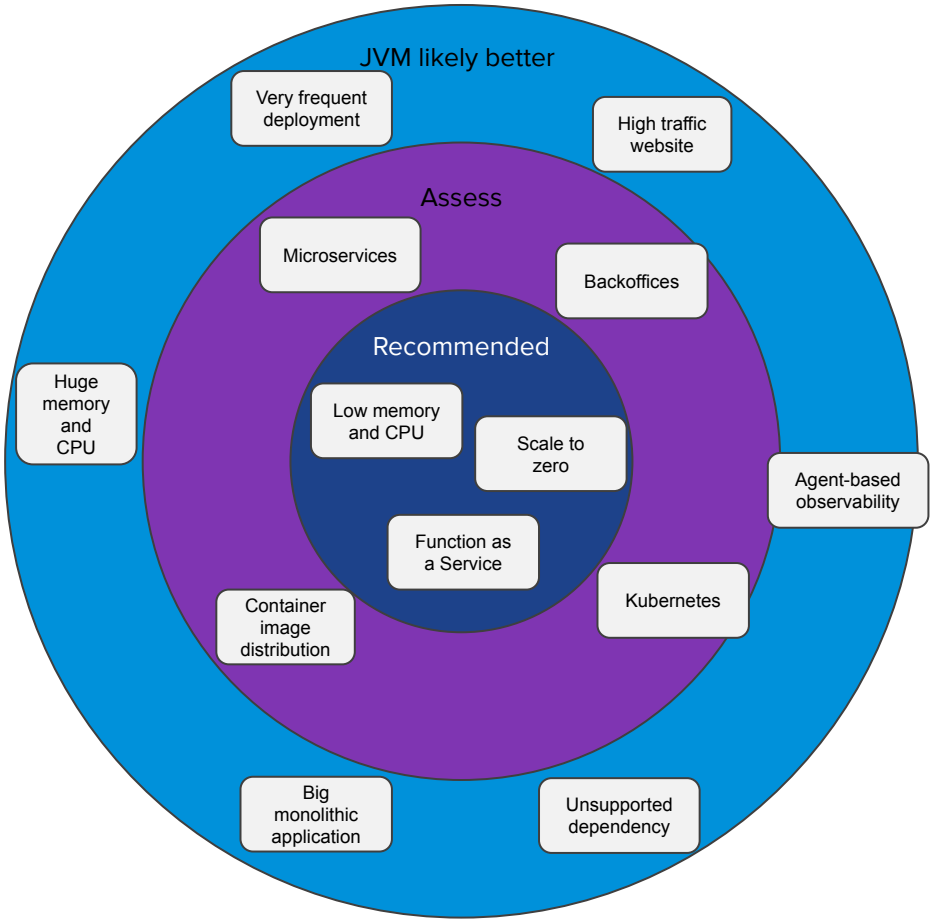
Compatibility

Additional metadata required for reflection, proxies, resources

Closed-world Assumptions

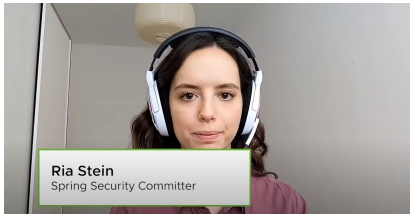
Bean conditions fixed at build time
No dynamic class loading

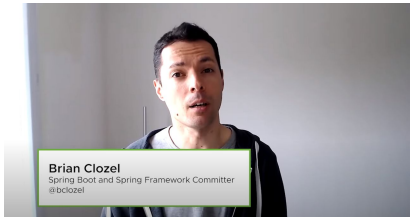
Use cases for native images

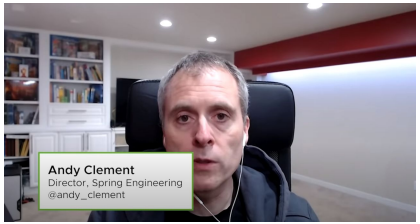


spring

Started with “Spring Native”







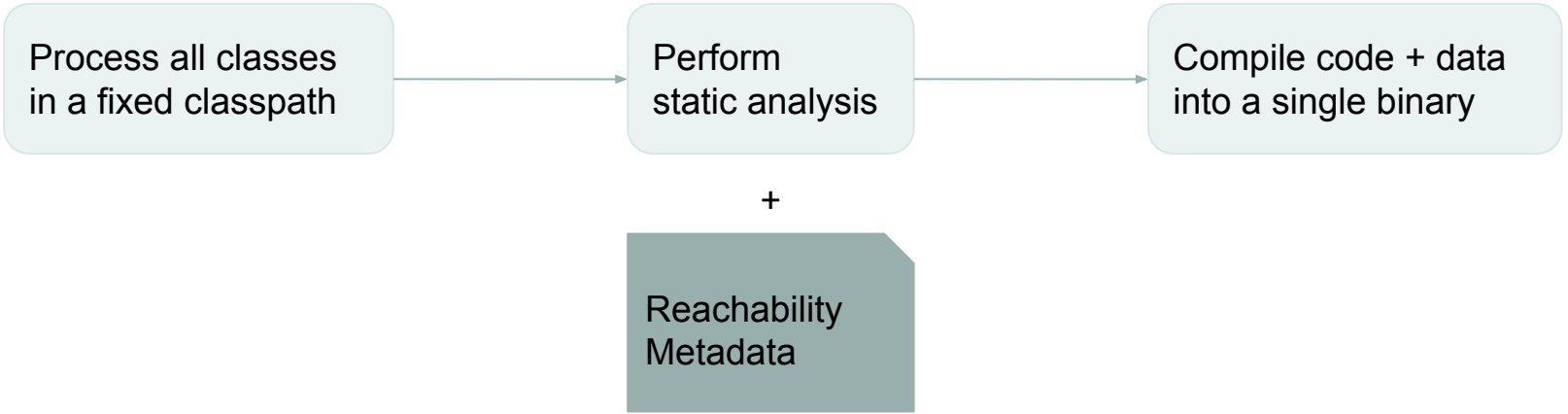
spring

Support Strategies for Frameworks

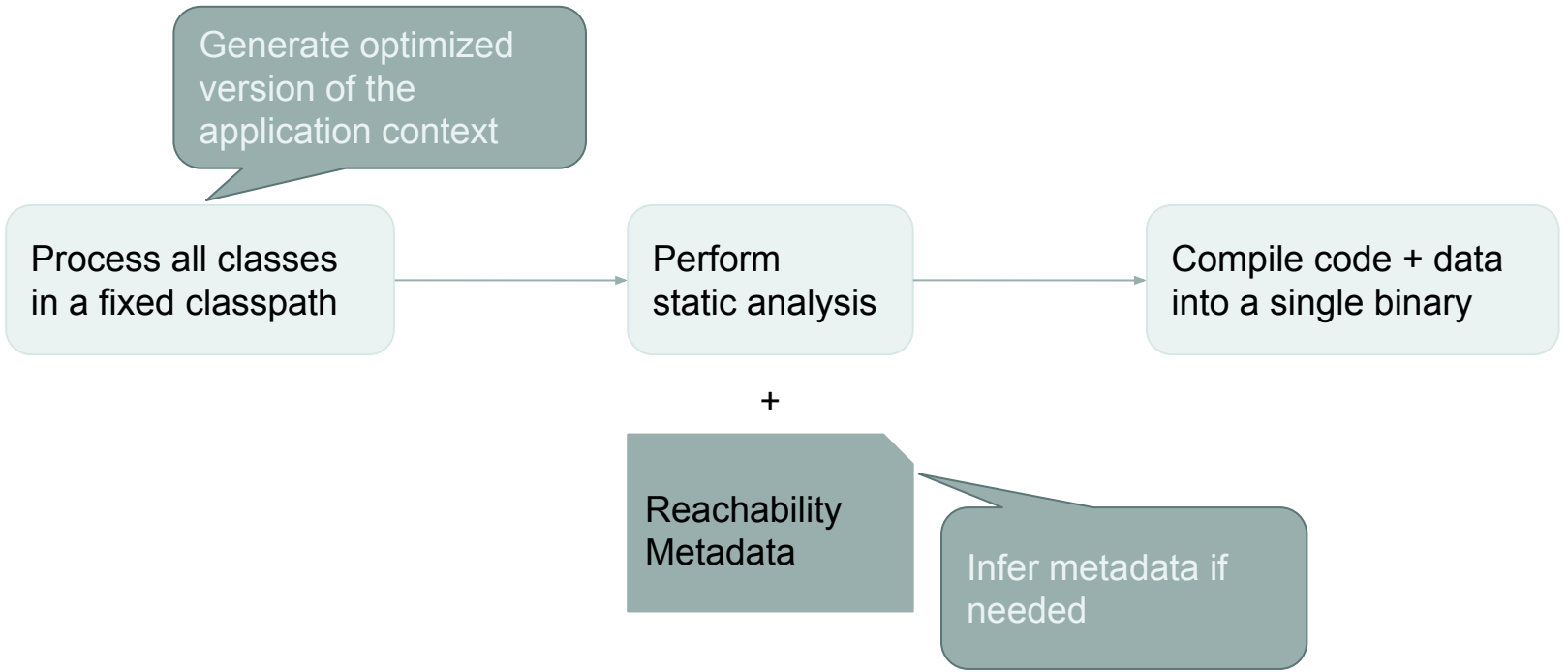
spring

Building a native image with GraalVM

Using the “native-image" binary



How Spring adapts for native

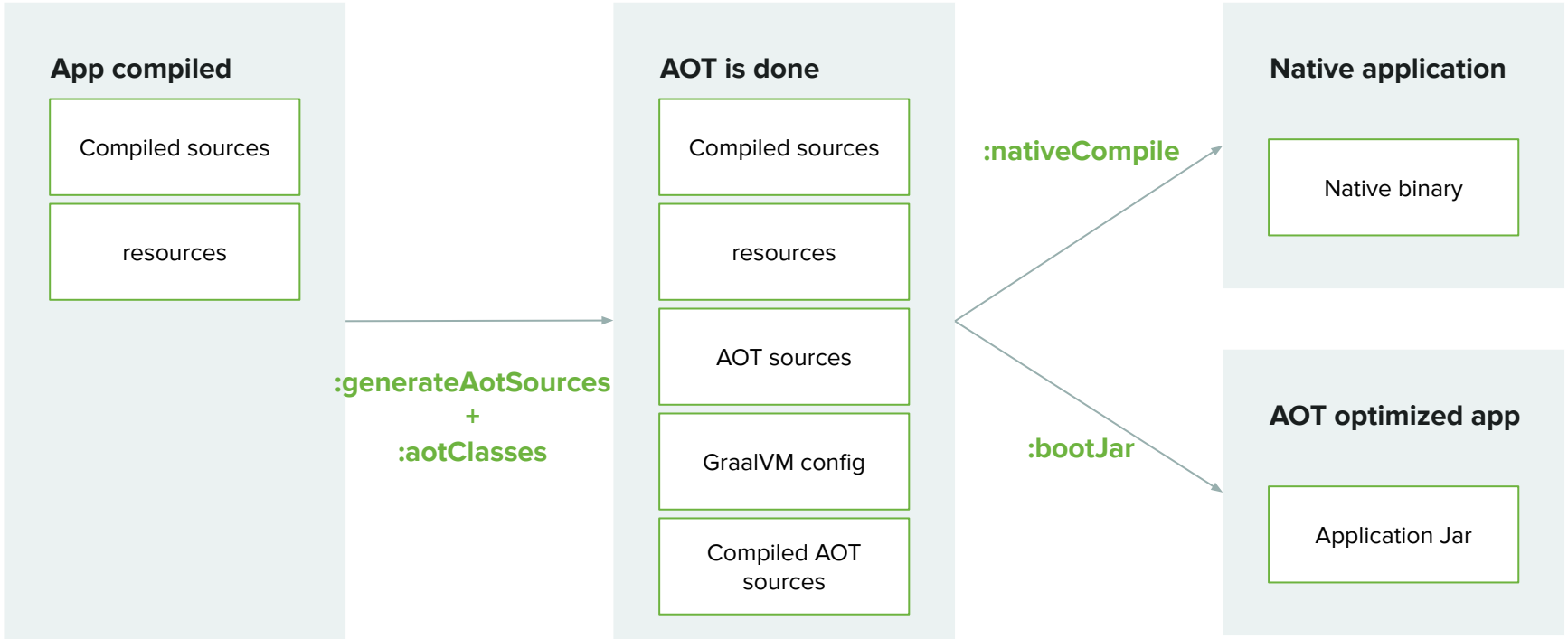


Also applies to resources, JDK proxies... See class metadata features

AOT processing in Spring

spring

AOT processing with Spring



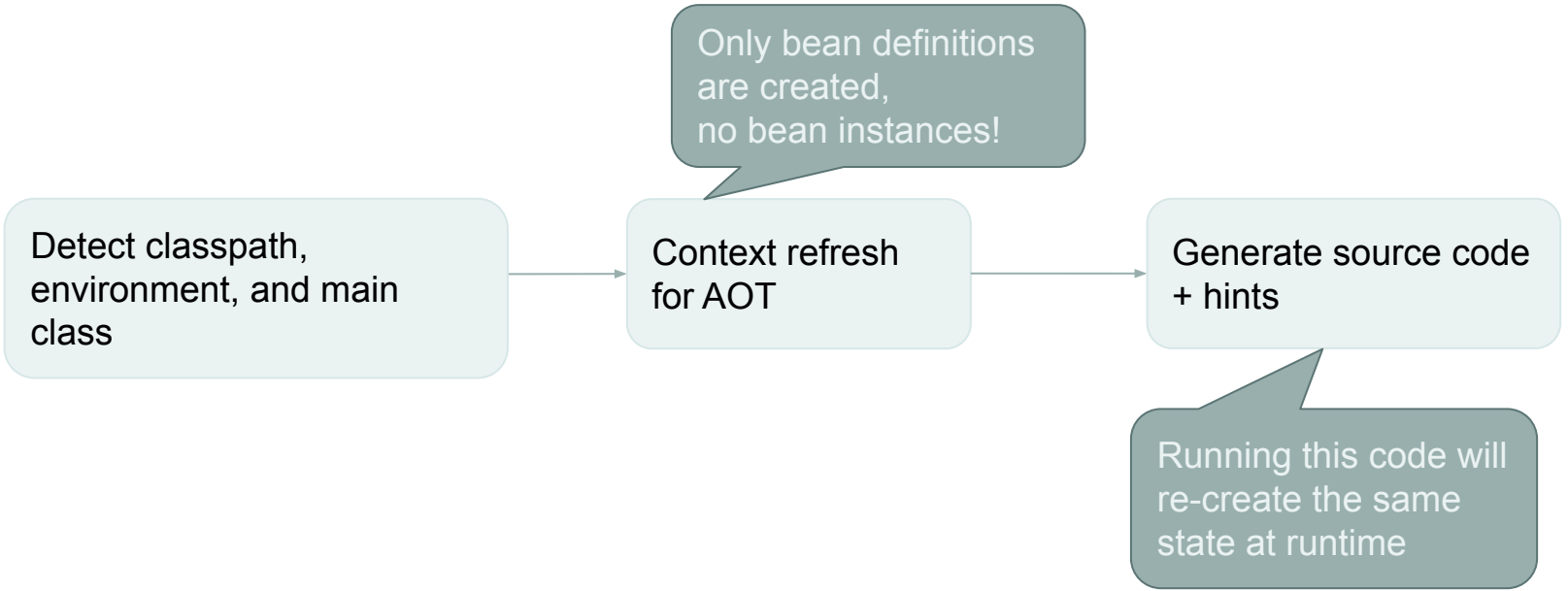
spring

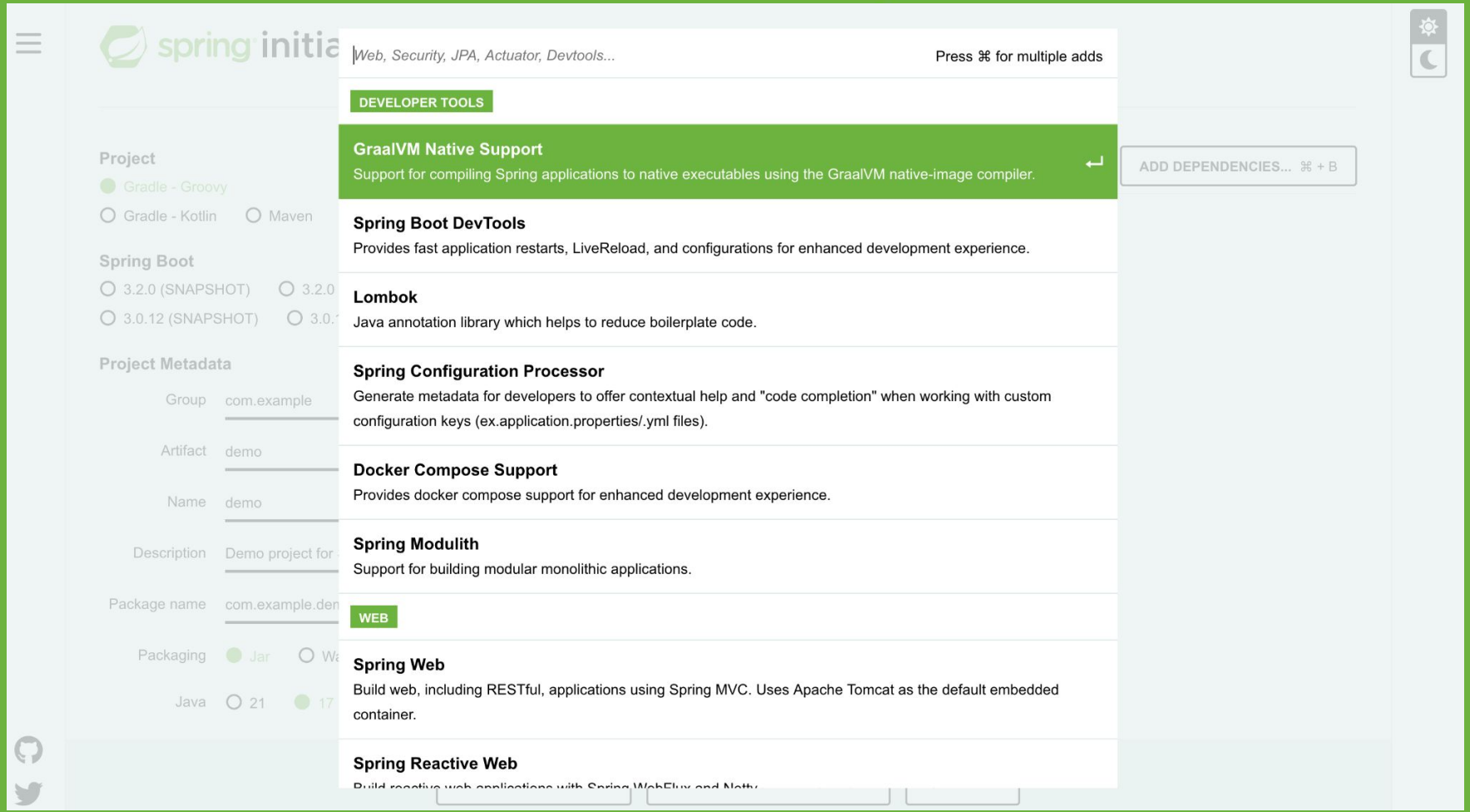
AOT phases in Spring

Generating functional configuration

- Skips the @Configuration model at runtime
- Generate available, debuggable source code
- Perfect fit with GraalVM native image static analysis
- Reachability metadata generated as needed

Generating AOT sources




spring initia
Web, Security, JPA, Actuator, Devtools...
Press ⌘ for multiple adds
DEVELOPER TOOLS
GraalVM Native Support
Support for compiling Spring applications to native executables using the GraalVM native-image compiler.
Spring Boot DevTools
Provides fast application restarts, LiveReload, and configurations for enhanced development experience.
Lombok
Java annotation library which helps to reduce boilerplate code.
Spring Configuration Processor
Generate metadata for developers to offer contextual help and "code completion" when working with custom configuration keys (ex.application.properties/.yml files).
Docker Compose Support
Provides docker compose support for enhanced development experience.
Spring Modulith
Support for building modular monolithic applications.
WEB
Spring Web
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.
Spring Reactive Web
ADD DEPENDENCIES... ⌘ + B
Project
Gradle - Groovy
Gradle - Kotlin
Maven
Spring Boot
3.2.0 (SNAPSHOT)
3.0.12 (SNAPSHOT)
Project Metadata
Group
com.example
Artifact
demo
Name
demo
Description
Demo project for
Package name
Packaging
Jar
Java
21

Will this work with my app?

spring

“Closed world” assumptions

Runtime flexibility constraints

- Application Classpath is fixed at build time
- Environment changes impacting the context are not supported
  - “spring.some.feature.enabled=true”
  - Spring profile changes that contribute new beans

Native image constraints

- Java agents are not supported (at runtime)
- Reading/manipulating bytecode: please don’t

JVM test strategies

- Running your app in AOT mode on the JVM “-Dspring.aot.enabled=true”
- Use Spring’s testing utilities for advanced cases (RuntimeHintsAgent, RuntimeHintsPredicates)

Run your test suite in a native image

Maven: `mvn -PnativeTest test`

Gradle: `gradle nativeTest`

```
Test run finished after 62 ms
[         2 containers found      ]
[         0 containers skipped    ]
[         2 containers started    ]
[         0 containers aborted    ]
[         2 containers successful ]
[         0 containers failed     ]
[         2 tests found           ]
[         0 tests skipped         ]
[         2 tests started         ]
[         0 tests aborted         ]
[         2 tests successful      ]
[         0 tests failed          ]
```

Reachability metadata repository

Native image configuration for JVM libraries

- Main goal remains direct inclusion in libraries
- This repository intends to fill the gap*

Automated testing via native build tools + dedicated CI infrastructure

Initially a GraalVM and Spring driven effort

Guidelines on how to craft native configuration

- Runtime initialization by default
- No build-time initialization
- Reachability based native configuration
- Mandatory native testing

* https://github.com/oracle/graalvm-reachability-metadata

Deployment strategies

Buildpacks

Container based native builds based on Buildpacks

Application compilation support

Requires Docker but no local GraalVM installation

Produces a Linux container image

- x64 is supported
- ARM support work in progress, follow buildpacks/lifecycle#435 for updates

Native Build Tools

Started as a collaboration between Spring and GraalVM team, and recently the Micronaut team has joined

Application compilation and **testing support**

Requires local GraalVM native-image compiler

Produces a native executable

- Linux (x64, ARM)
- MacOS (x64, ARM)
- Windows (x64)

Spring apps adoption

Being part of the GraalVM community

🆕 developers and use cases

Q&A

https://github.com/snicoll/demo-aot-native

spring

Thank you

Contact me at @snicoll[@mastodon.online]