Spring Framework 5.0 Preview & Roadmap

Juergen Hoeller
Spring Framework Lead
Pivotal

On our way to 5.0
First up: 4.3

spring by Pivotal

Spring Framework 4.3

- **Last 4.x feature release!**
- **4.3 RC1: April 6th, 2016**
- **4.3 GA: June 10th, 2016**

- **Extended support life until 2019**
  - on JDK 6, 7, 8
  - on Tomcat 6, 7, 8.0, 8.5
  - on WebSphere 7, 8.0, 8.5 and 9

- **Programming model refinements brought forward to JDK 6+**
  - DI & MVC refinements, composed annotations

spring by Pivotal

The State of the Art: Component Classes

```
@Service
@Lazy
public class MyBookAdminService implements BookAdminService {

    // @Autowired
    public MyBookAdminService(AccountRepository repo) {
        ...
    }

    @Transactional
    public BookUpdate updateBook(Addendum addendum) {
        ...
    }
}
```

Configuration Classes with Autowired Constructors

```
@Configuration
public class MyBookAdminConfig {

    private final DataSource bookAdminDataSource;

    // @Autowired
    public MyBookAdminService(DataSource bookAdminDataSource) {
        this.bookAdminDataSource = bookAdminDataSource;
    }

    @Bean
    public BookAdminService myBookAdminService() {
        MyBookAdminService service = new MyBookAdminService();
        service.setDataSource(this.bookAdminDataSource);
        return service;
    }
}
```

Annotated MVC Controllers

```
@Controller
@CrossOrigin
public class MyRestController {

    @RequestMapping(path="/books/{id}", method=GET)
    public Book findBook(@PathVariable long id) {
        return this.bookAdminService.findBook(id);
    }

    @RequestMapping(path="/books/new", method=POST)
    public void newBook(@Valid Book book) {
        this.bookAdminService.storeBook(book);
    }
}
```

Precomposed Annotations for MVC Controllers

```
@RestController
@CrossOrigin
public class MyRestController {

    @GetMapping("/books/{id}")
    public Book findBook(@PathVariable long id) {
        return this.bookAdminService.findBook(id);
    }

    @PostMapping("/books/new")
    public void newBook(@Valid Book book) {
        this.bookAdminService.storeBook(book);
    }
}
```

Themes for 5.0:
JDK 9, HTTP/2, Reactive

spring by Pivotal

Spring Framework 5.0

- **A new framework generation for 2017+**
- **5.0 M1: July 2016**
- **5.0 RC1: December 2016**

- **Major baseline upgrade**
  - JDK 8+, Servlet 3.0+, JMS 2.0+, JPA 2.1+, JUnit 5

- **Key infrastructure themes**
  - JDK 9 and Jigsaw modules
  - Servlet 4.0 and HTTP/2
  - Reactive architectures

spring by Pivotal

Comprehensive JDK 9 Support

- **Spring 5 schedule is close to JDK 9 schedule**
  - JDK 9 intends to go GA in March 2017 (but may get delayed still)
- **Jigsaw – a new module system for applications**
  - symbolic module names and requires/exports metadata for jar files
  - currently no versioning, just structuring plus visibility enforcement
  - module path as alternative to class path
- **New HTTP client and general support for HTTP/2**
  - superseding the outdated java.net.HttpURLConnection
  - TLS extension for ALPN

Using Jigsaw with Spring (ideally)

- **Spring Framework jars coming with Jigsaw metadata out of the box**
  - internally declaring module-info for each jar
- **Separate module namespace, following Maven Central jar naming**
  - spring-context, spring-jdbc, spring-webmvc
- **An application's module-info.java can then look as follows...**

```
module my.app.db {
    requires java.sql;
    requires spring.jdbc;
}
```


spring by Pivotal

The Importance of HTTP/2 (RFC 7540)

- **Enormous benefits over HTTP 1.1 (which dates back to 1996)**
  - binary protocol
  - TLS (SSL) everywhere
  - connection multiplexing
  - headers compression
  - request prioritization
  - push of correlated resources

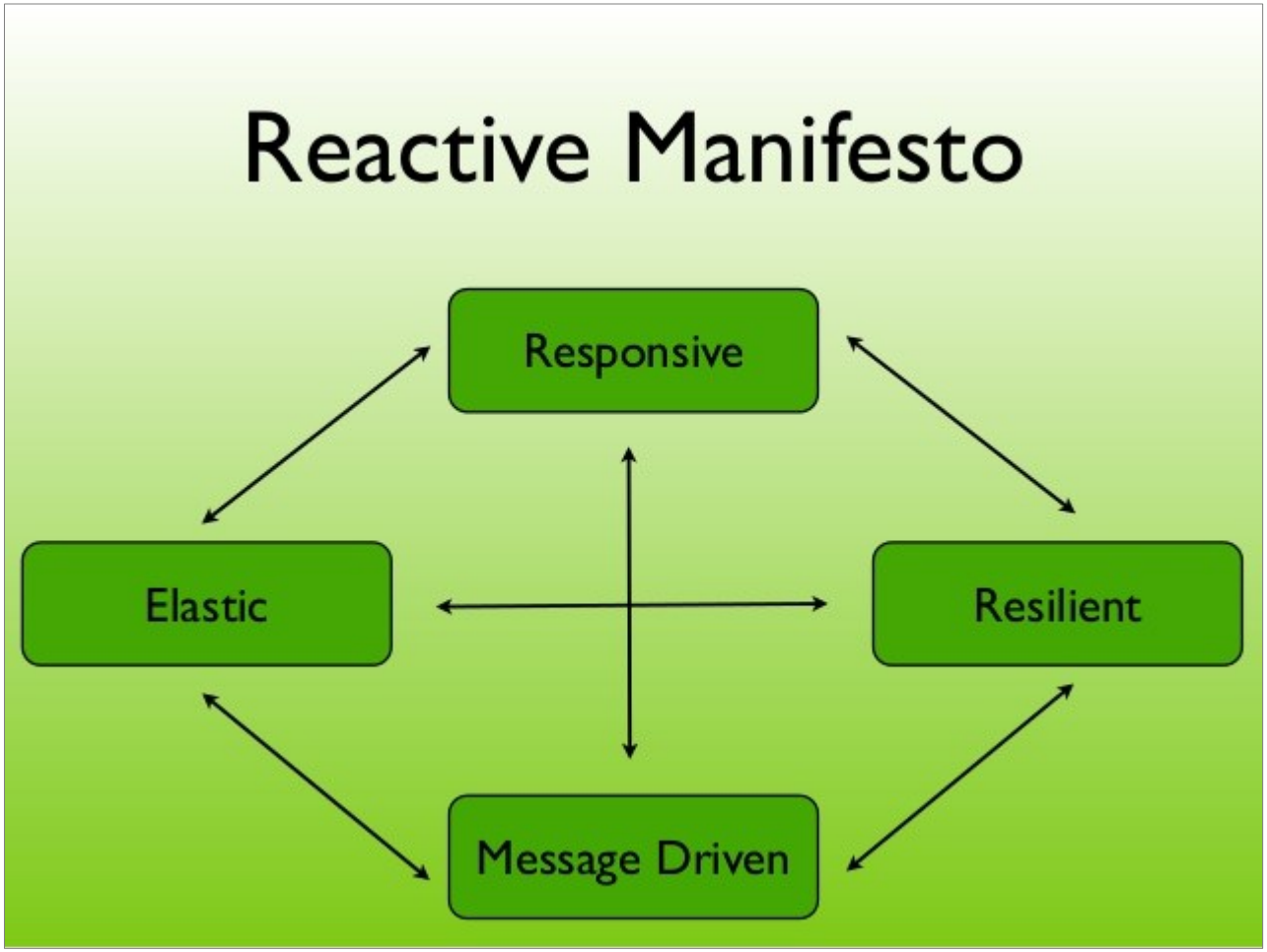
- **Browsers already implement HTTP/2 over TLS**
  - major websites work with HTTP/2 already: Google, Twitter, etc
  - *We need to embrace it in Java land as well!*

Spring 5 and HTTP/2

- **Servlet 4.0 – mid 2017**
  - enforces support for HTTP/2 in Servlet containers
  - API features for stream prioritization and push resources
- **Tomcat / Jetty / Undertow**
  - native HTTP/2 support available in current Servlet 3.1 containers
  - Tomcat 8.5 / 9.0, Jetty 9.3, Undertow 1.3
- **Spring Framework 5.0 will ship dedicated Servlet 4.0 support**
  - as well as dedicated support for the new JDK 9 HTTP client
  - but like 4.3, it focuses on native HTTP/2 on top of Tomcat / Jetty / Undertow

spring by Pivotal

The Importance of Reactive Architectures



Reactive Streams Specification

- **Focus on infrastructure interoperability**
  - web servers, datastore drivers
  - and of course: web frameworks!

- **Minimal API**
  - Publisher + Subscriber/Subscription for backpressure support
  - repackaged into JDK 9 as java.util.concurrent.Flow

- **Operators left up to composition libraries**
  - map, flatMap, take, subscribe, ...
  - Reactor, RxJava, Akka Streams

Reactive Web Endpoints in Spring

- **A Spring MVC like endpoint model based on a reactive foundation**
  - reusing the common Spring MVC programming model style
  - but accepting and returning reactive streams

- **A new HTTP endpoint engine on top of a non-blocking runtime**
  - Netty, Jetty, Tomcat, Undertow
  - not based on the Servlet API but adaptable to a Servlet container

- **Currently developed as a public R&D project**
  - https://github.com/spring-projects/spring-reactive/
  - to be merged into Spring Framework master in June (for 5.0 M1)

Reactive Web Controller with Reactor Flux

```
@Controller
public class MyReactiveWebController {

    @RequestMapping("/capitalize")
    public Flux<Person> capitalize(Flux<Person> persons) {
        return persons.map(person -> {
            person.setName(person.getName().toUpperCase());
            return person;
        }
    }
}
```


spring by Pivotal

Reactive Web Controller with Repository Interop

```
@Controller
public class MyReactiveWebController {

    @Autowired
    private MyRepository<Person> repository;

    @RequestMapping("/insert")
    public Mono<Void> insert(Flux<Person> persons) {
        return this.repository.insert(persons);
    }
}
```

Reactive Infrastructure All Around

- **Reactive datastore drivers becoming available**
  - Postgres, Mongo, Couchbase
- **Reactive HTTP clients**
  - Netty, Jetty, OkHttp
- **Reactive Streams Commons project**
  - Servlet adapters: by default against Servlet 3.1 async I/O
  - native container SPI for more efficiency at runtime
  - currently a collaboration between Spring and Jetty / Tomcat

Summary

Spring Framework 4.3 (June 2016)
Programming model refinements on JDK 6/7/8

Spring Framework 5.0 (early 2017)
JDK 8+9, Jigsaw, HTTP/2, Reactive Streams

spring by Pivotal