

# API Workshop

Build REST APIs with Spring Boot

O'REILLY®

Software  
Architecture

# AGENDA

- Introduction to Spring & Spring Boot
- Web services, APIs and REST

# WHAT IS SPRING?

“Make the right thing easy to do”

Rod Johnson

# WHAT IS SPRING?

- Originally a very simple **dependency injection** framework for Java
- **Abstractions** for common patterns
- Now consists of many projects and components for building applications
- **Data Access** - transaction support, JDBC, ORM
- **Integration** with external concerns e.g. messaging and caching
- **Spring MVC** and the newer **WebFlux**
- **Multiple language support** for the JVM

# BUILDING SERVICES USING JAVA

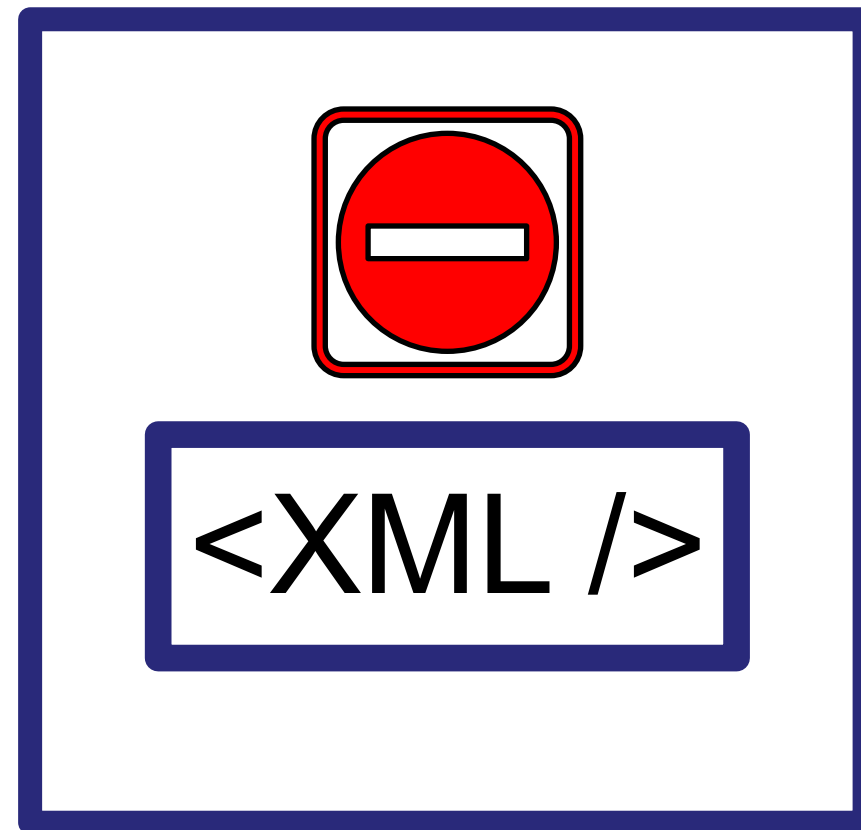
## IN THE OLD(ISH) DAYS

<XML />

<XML />

<XML />

# WHAT IS SPRING BOOT?

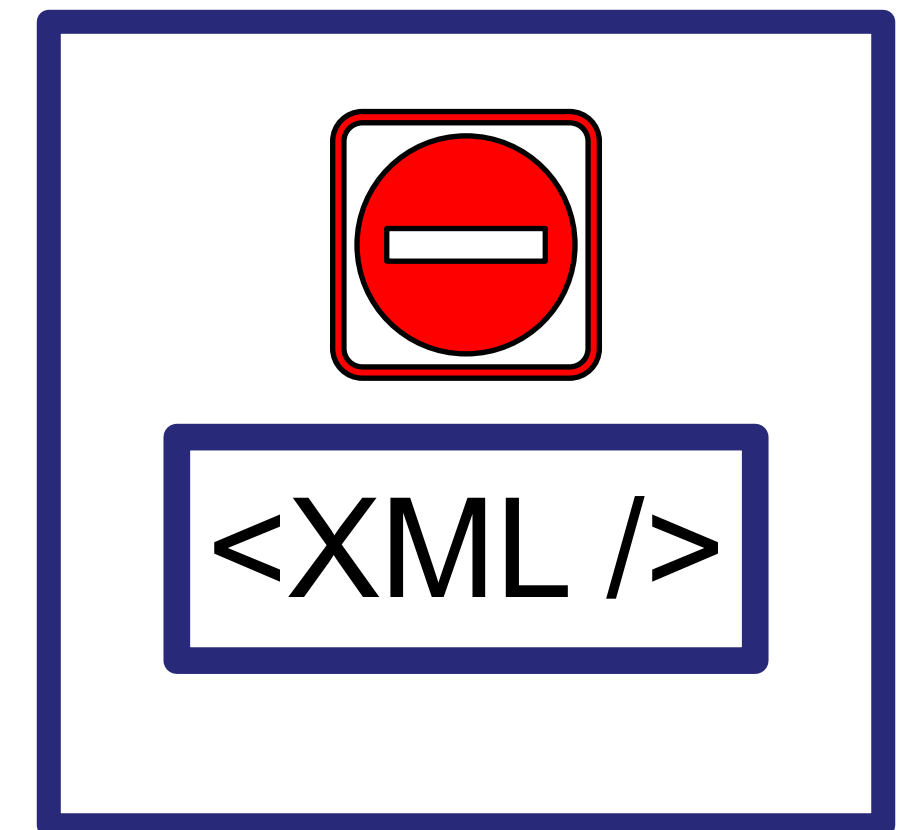


Stand Alone



# AUTO CONFIGURATION

- Including a library dependency will usually also include auto configuration
  - This is facilitated by Spring Boot Starters
- Driven by the `@EnableAutoConfiguration` annotation
- The configuration of a `@Bean` can be overridden



# STANDALONE

- Previously with Java applications we had to
  - **Package** the application up into a WAR file or similar
  - **Download** the package onto a webserver
  - **Configure** that webserver to run the application
  - **Deploy and start** the webserver
- Spring Boot is simply **package** and **run**



Stand Alone




# OPINIONATED

- Closely tied to AutoConfiguration
- Spring will make opinions about how to configure your application
- Convention over configuration
- Using Spring's opinion means you can be up and running in minutes
- **WARNING: hidden complexity!**



<http://start.spring.io>

 **Spring Initializr**  
Bootstrap your application

Project

Language

Spring Boot

Project Metadata

Dependencies

Maven Project

**Gradle Project**

**Java**

Kotlin

Groovy

2.2.0 M6

2.2.0 (SNAPSHOT)

2.1.9 (SNAPSHOT)

**2.1.8**

Group

com.jpgough

Artifact

apiworkshop

> Options

Q

☰

2 selected

Search dependencies to add

Web, Security, JPA, Actuator, Devtools...

Selected dependencies

**Contract Verifier**  
Moves TDD to the level of software architecture by enabling Consumer Driven Contract (CDC) development. ✓

**Spring Web**  
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container. ✓

Generate the project - ⌘ + ↵

Explore the project - Ctrl + Space

Light UI Github Twitter

© 2013-2019 Pivotal Software  
start.spring.io is powered by  
[Spring Initializr](#) and [Pivotal Web Services](#)

# WEB SERVICES, APIS & REST

- SpringBoot makes it relatively easy to build and deploy HTTP based web services
- Web services (or micro services) can be designed to expose business data or capabilities via APIs
- REST can be used over HTTP to provide structure to those APIs

# WHAT IS REST?

- **R**epresentational **S**tate **T**ransfer
- An architectural style, or design pattern, for APIs.
- Resources
- Operations
  - Create, Read, Update, Delete

# REST OVER HTTP

- HTTP as the transport layer
- URLs identify resources
  - <http://www.api-workshop.com/todos>
  - <http://www.api-workshop.com/todos/1>

# REST OVER HTTP

- HTTP verbs for operations
  - POST, GET, PUT, DELETE
- HTTP status codes represent the result of a request
  - HTTP 2xx - Success
  - HTTP 3xx - Redirection
  - HTTP 4xx - Client errors
  - HTTP 5xx - Server errors

# EXAMPLE: GET ALL TODOS

*Request:* **GET** <http://www.api-workshop.com/todos>

HTTP Headers:

- **Accept:** application/json

*Response:* **200 OK**

```
{
  "todos": [
    {
      "id": 1,
      "description": "Attend API workshop",
      "done": false
    }
  ]
}
```

# EXAMPLE: GET WITH QUERY PARAMETERS

*Request:* **GET** <http://www.api-workshop.com/todos?done=false>

HTTP Headers:

- **Accept:** application/json

*Response:* **200 OK**

```
{
  "todos": [
    {
      "id": 2,
      "description": "Learn about SpringBoot",
      "done": true
    }
  ]
}
```



# EXAMPLE: CREATE A TODO

*Request:* **POST** <http://www.api-workshop.com/todos>

HTTP Headers:

- **Content-Type:** application/json

```
{  
  "description": "Learn about REST APIs"  
}
```

*Response:* **201 CREATED**

```
{  
  "id": 3,  
  "description": "Learn about REST APIs",  
  "done": false  
}
```