INTRODUCTION TO SPRING & SPRING BOOT

Carlos Barragan, Matthias Häussler, Jonas Grundler
- NovaTec Consulting GmbH -

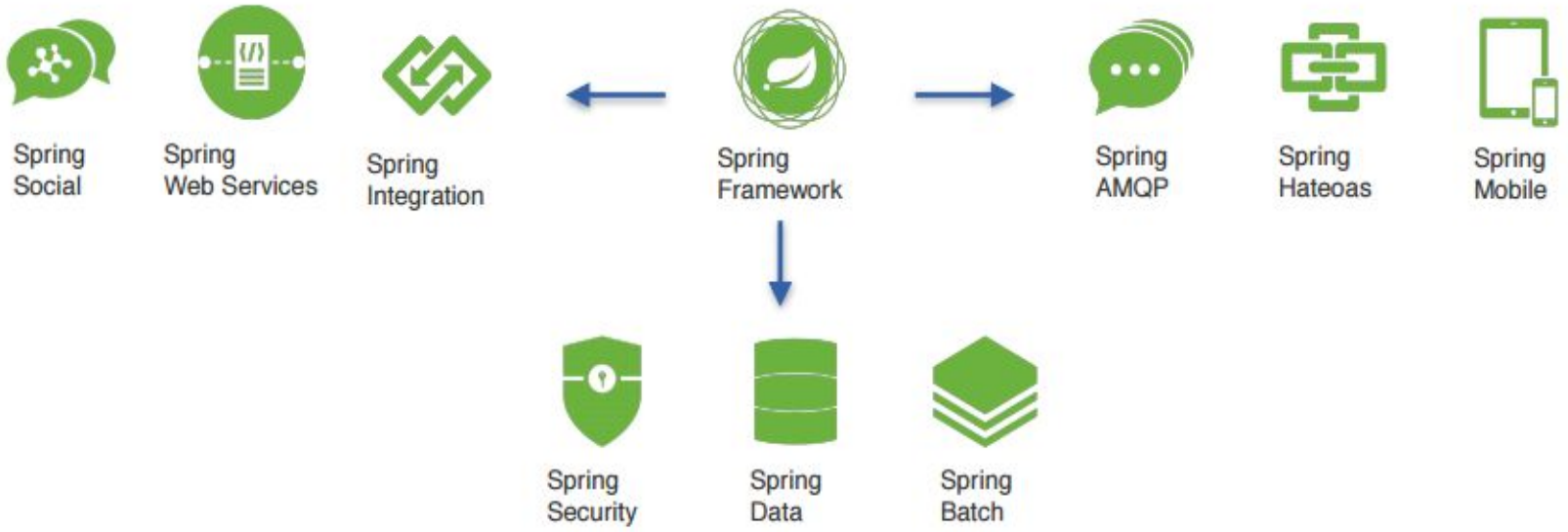
AGENDA

- **Spring Framework**
- **Spring Boot**
- **Spring Initializr**
- **Spring Tool Suite**

The Spring Framework

- Introduced as open project in 2003
- Version 1.0 in 2004
- JAX Innovation Award 2005
- Taken over by VMWare 2009
- Current version 5.1.3
- Current pre-release Version 5.2.x

Spring Framework



Initial idea

- Lightweight alternative to Java EE
- Simple approach to Enterprise Java Dev using
  - Dependency Injection
  - Aspect-Oriented Programming
  - Capabilities of EJBs with POJOs
- Light in code <-> Heavy in configuration
- Java configured through XML (a lot of XML!)

Ideas for improvement

Any time spent writing configuration is time spent not writing application logic!

Hello World web application in Spring

- A project structure
  - Maven or Gradle
  - Required Dependencies
    - Spring MVC
    - Servlet API
- A web.xml
- Spring configuration to enable Spring MVC
- A Controller class to respond to HTTP requests with „Hello, World!“
- An application server (e.g. Tomcat) to deploy the application

Spring Boot idea

- Many configurations re-occur in different deployments
- The only difference is the application logic itself
- Re-Use Spring configuration possibilities and „boilerplate“ them
- If all (at least most ☺) of Spring web applications need the same config, why should the developer have to provide it?

The Spring Framework

Spring Boot

Misconceptions – What Spring Boot is not

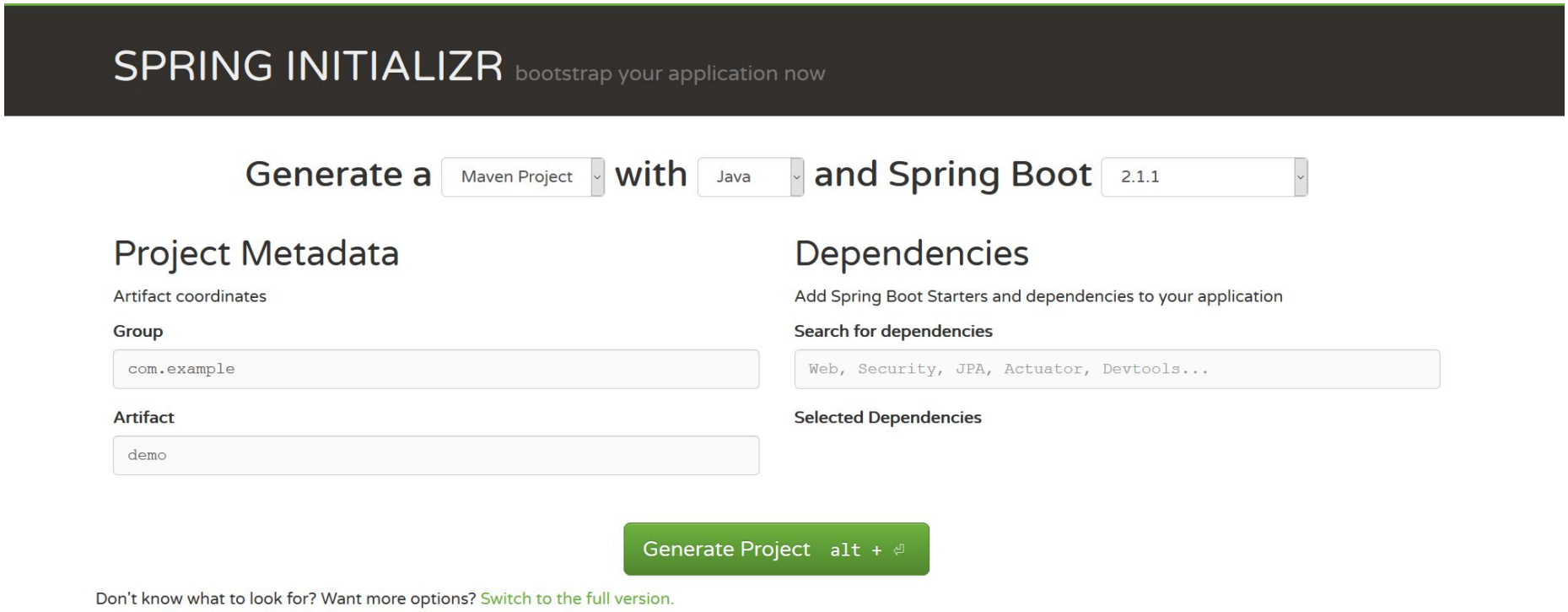
- **Spring Boot is not an application server**
  - It is possible to create a full-functional self-contained JAR that embeds a Tomcat (or Jetty, Undertow..) application server, but it does not provide this logic by itself
- **Spring Boot does not implement any enterprise Java specification**
  - It can leverage the implementation
  - E.g. There is no Spring Boot JPA implementation, but Beans can be auto-configured to use a JPA implementation (like Hibernate)
- **Spring Boot does not employ any form of code generation**
  - It uses the configuration features from Spring 5
  - It uses dependency resolution from Maven or Gradle
- **On the inside it is still simply Spring. Spring Boot is simply taking over the configuration effort**

Spring Boot Essentials

- **Automatic Configuration**
  - For application functionality of many Spring apps
- **Starter Dependencies**
  - Libraries and Dependencies will automatically be build based on what you tell Spring Boot
- **The Actuator**
  - Insight of what is going on inside your Spring Boot app
- **Command-Line Interface**
  - Optional, will not be covered in this module

Getting Started

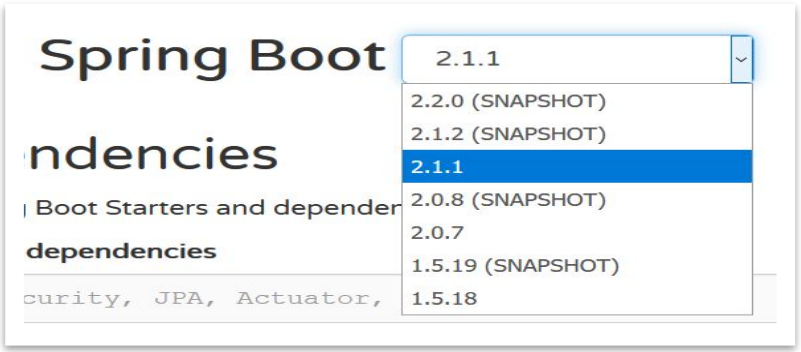
- Spring Initializr
- https://start.spring.io/



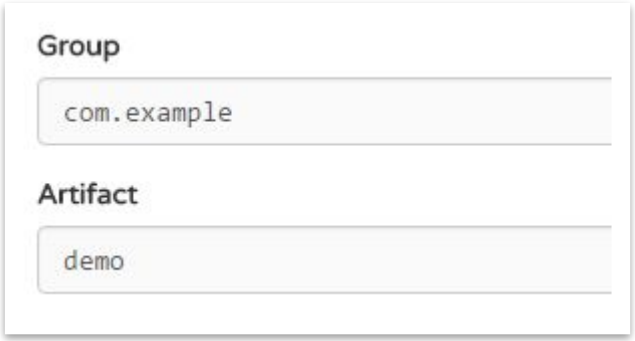
Options

- Maven vs. Gradle

- Version

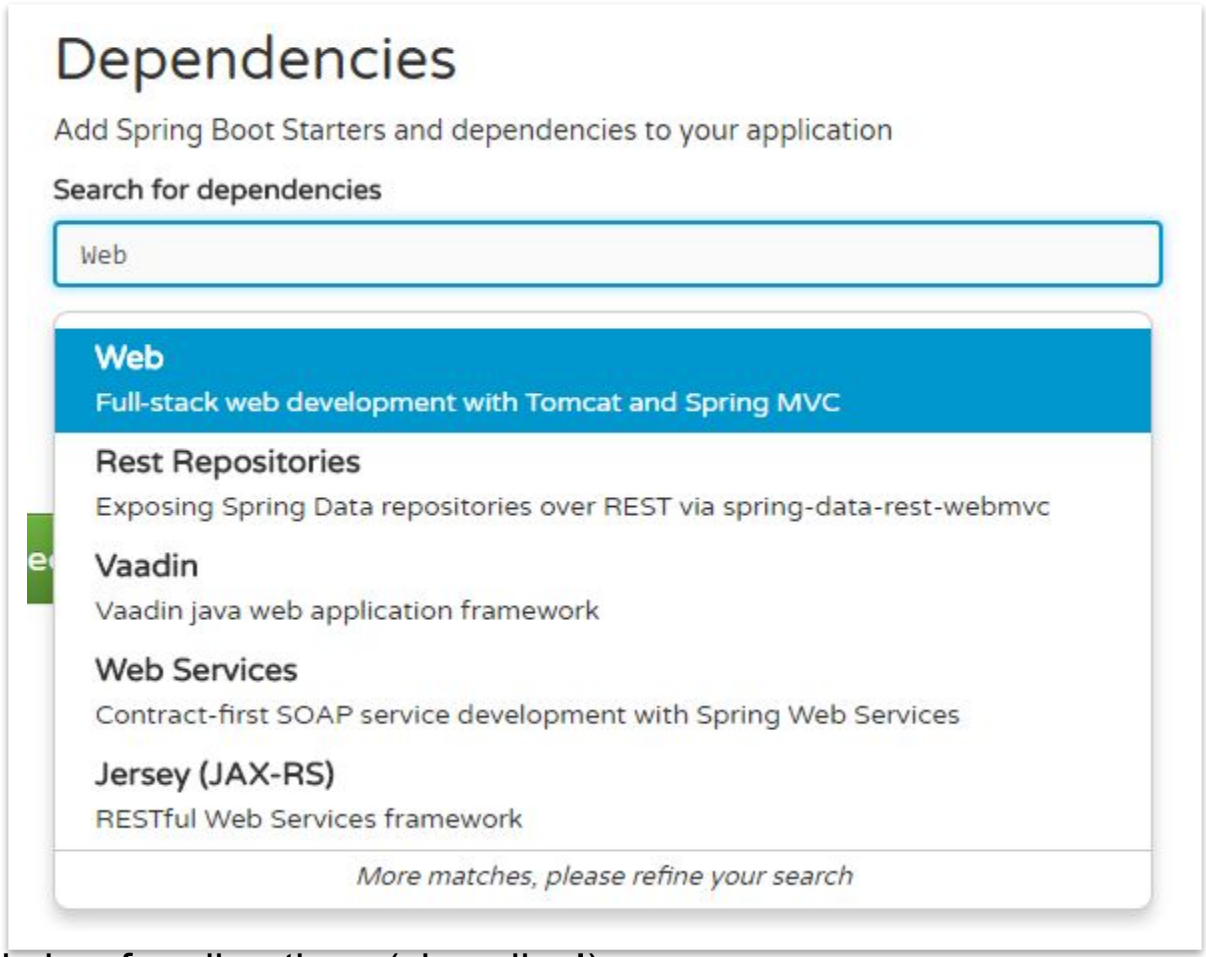


- Metadata



Dependencies

- Shows options based on what you type



- Switch to full view for all options (show live!)

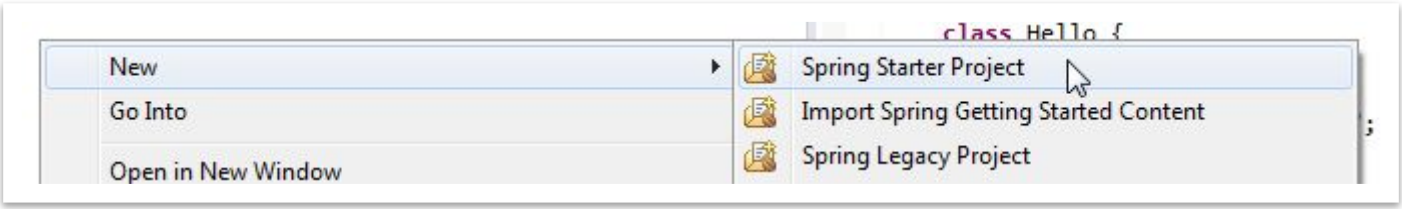
What You Get…

- Will automatically download the artifact within browser
- Artifact contains
  - Spring Boot project structure
  - No application code, only a skeleton
    - Main class
    - Test class
  - application.properties file
  - Maven or Gradle build specifications

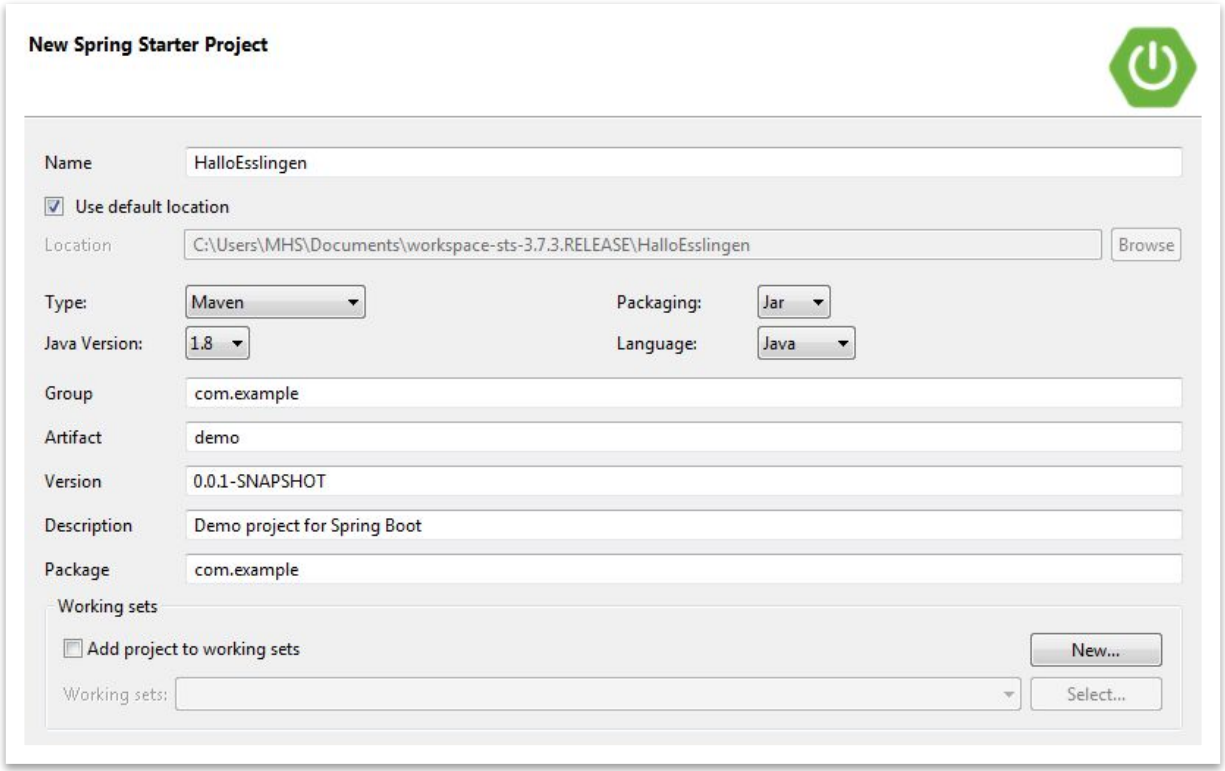

```
MHS@nbmhsH730 MINGW64 ~/Downloads
$ unzip HelloWorld.zip
Archive:  HelloWorld.zip
   creating: HelloWorld/
  inflating: HelloWorld/mvnw
   creating: HelloWorld/.mvn/
   creating: HelloWorld/.mvn/wrapper/
   creating: HelloWorld/src/
   creating: HelloWorld/src/main/
   creating: HelloWorld/src/main/java/
   creating: HelloWorld/src/main/java/com/
   creating: HelloWorld/src/main/java/com/example/
   creating: HelloWorld/src/main/resources/
   creating: HelloWorld/src/main/resources/static/
   creating: HelloWorld/src/main/resources/templates/
   creating: HelloWorld/src/test/
   creating: HelloWorld/src/test/java/
   creating: HelloWorld/src/test/java/com/
   creating: HelloWorld/src/test/java/com/example/
  inflating: HelloWorld/.gitignore
  inflating: HelloWorld/.mvn/wrapper/maven-wrapper.jar
  inflating: HelloWorld/.mvn/wrapper/maven-wrapper.properties
  inflating: HelloWorld/mvnw.cmd
  inflating: HelloWorld/pom.xml
  inflating: HelloWorld/src/main/java/com/example/HelloWorldApplication.java
  inflating: HelloWorld/src/main/resources/application.properties
  inflating: HelloWorld/src/test/java/com/example/HelloWorldApplicationTests.java
```

Leveraging through Spring Tool Suite

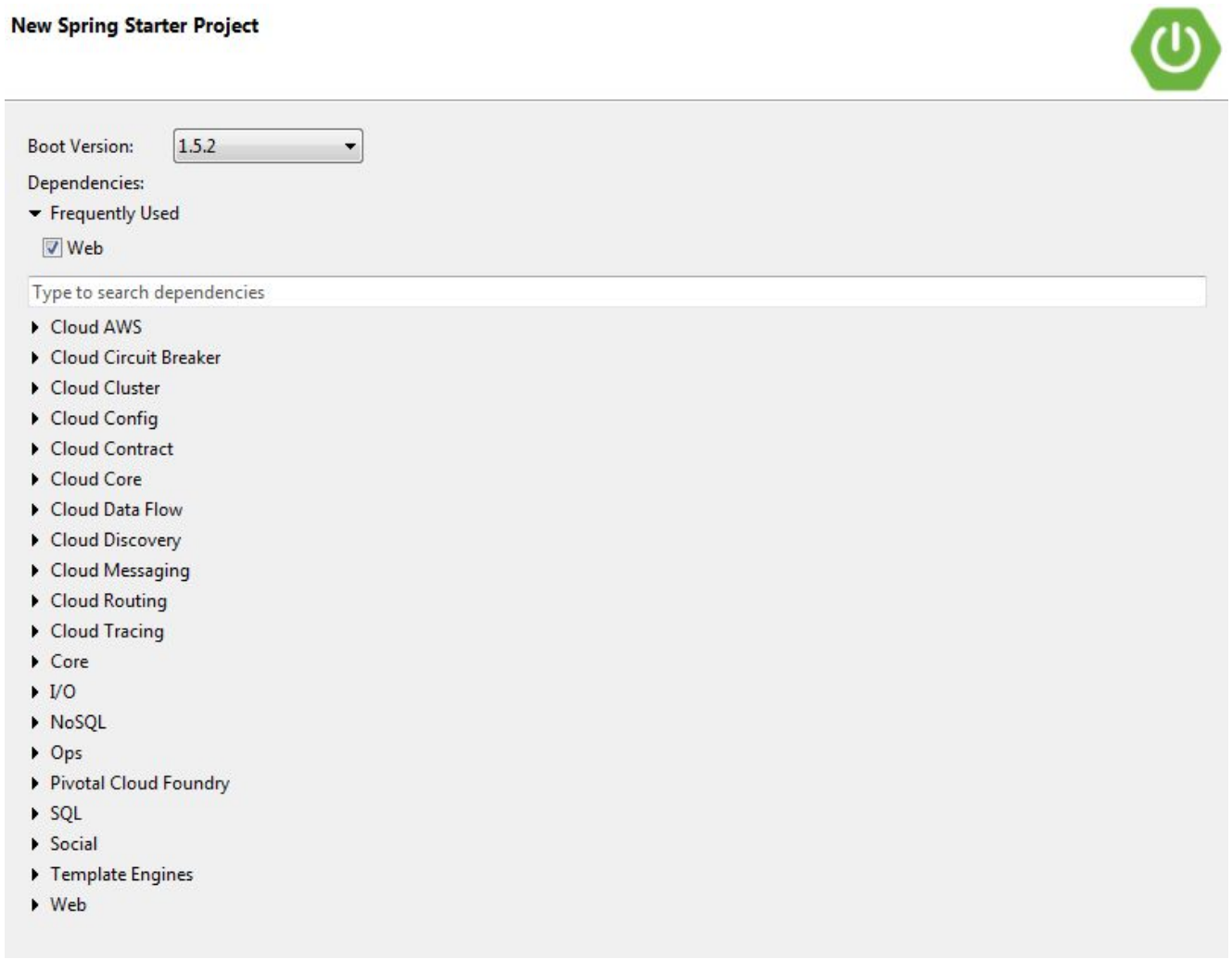
- Create a new Spring Starter Project (Online connection required)



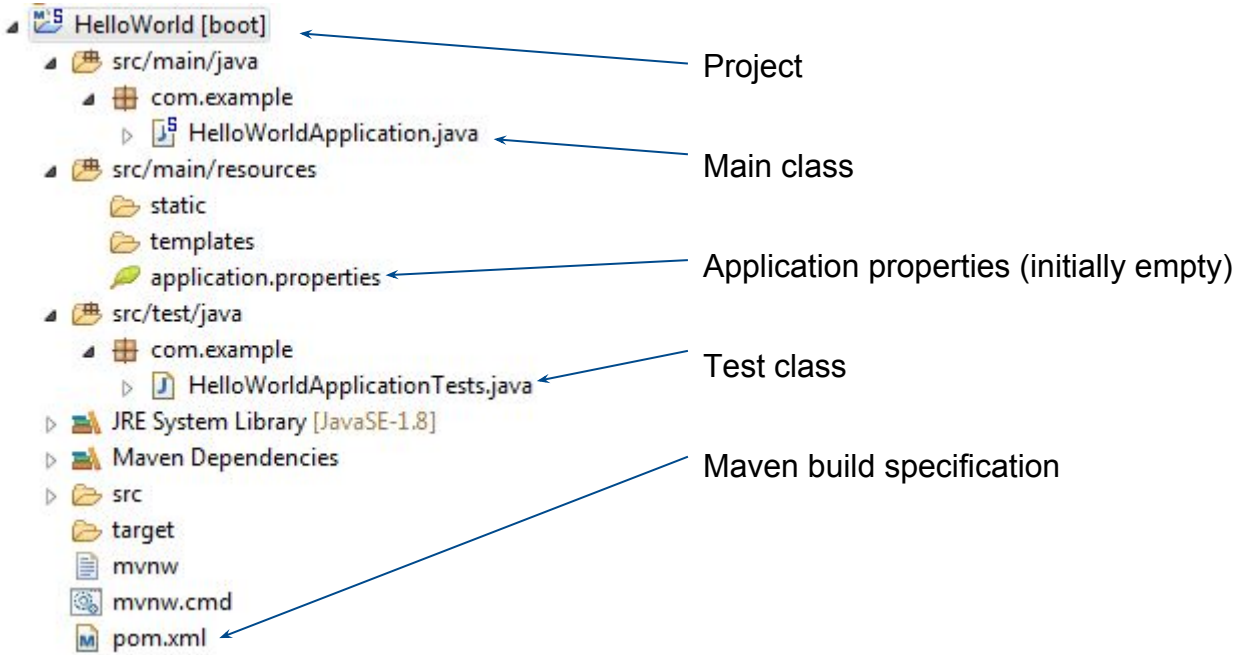
- Same options as on Initializr web page



Dependencies (again)



Analyzing the Project Structure



The Initial Java Code Classes

- Main class

```java
package com.example;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class HelloWorldApplication {

    public static void main(String[] args) {
        SpringApplication.run(HelloWorldApplication.class, args);
    }
}
```

main method delegates to Spring Boot's SpringApplication class by calling run.

SpringApplication will bootstrap our application, starting Spring, which will in turn start the auto-configured Tomcat web server.

We need to pass HelloWorldApplication.class as an argument to the run method, to tell SpringApplication which is the primary Spring component.

- Test class

```java
package com.example;

import org.junit.Test;

@RunWith(SpringRunner.class)
@SpringBootTest
public class HelloWorldApplicationTests {

    @Test
    public void contextLoads() {
    }

}
```

Maven Build Specification

- No version information!
- Spring Boot will sort it out for you

```
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
        </plugin>
    </plugins>
</build>
```

Adding the Application Logic

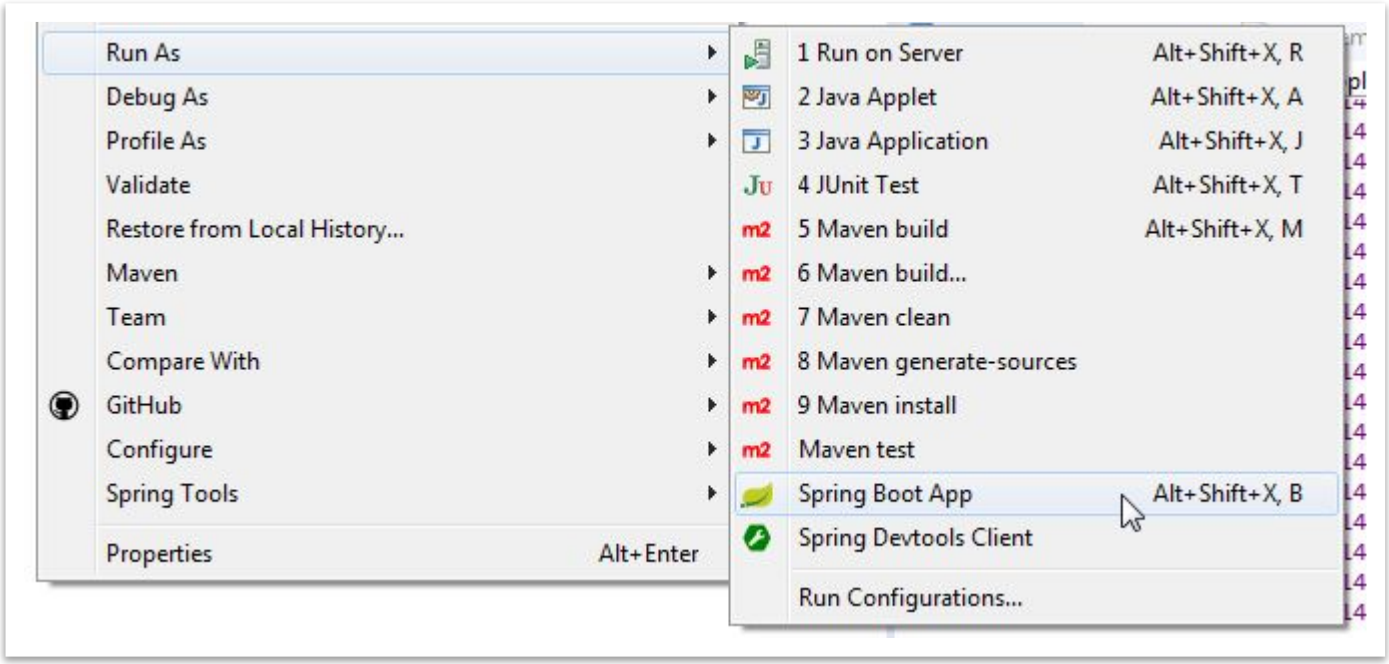
```java
package com.example;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@SpringBootApplication
public class HelloWorldApplication {

    public static void main(String[] args) {
        SpringApplication.run(HelloWorldApplication.class, args);
    }

    @RestController
    class Hello{
        @RequestMapping("/")
        String greeting() {
            return "Hallo Esslingen!";
        }
    }
}
```

New imports

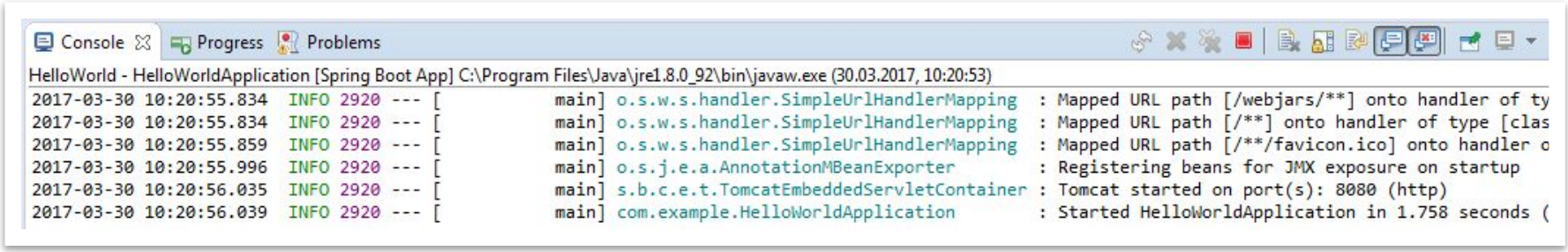
New application logic
(copied from twitter..)

Run it!

- Project Context Menu

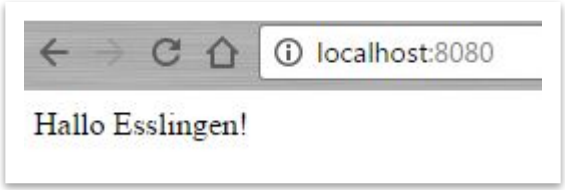


- Console out



q.e.d.

- It was not that hard ☺



Review – What has Just Happened

```
@SpringBootApplication
public class HelloWorldApplication {
```

- @SpringBootApplication combines 3 important annotations:
  - @Configuration – Designates a configuration class using Spring‘s Java-based configuration
  - @ComponentScan – Enables auto-discovery of web controller classes and other components as beans in the Spring application context
  - @EnableAutoConfiguration – This is where the magic happens. Builds the configuration which it thinks you need for your application without having you doing it

```
/**
 * Enable auto-configuration of the Spring Application Context, attempting to guess and
 * configure beans that you are likely to need. Auto-configuration classes are usually
 * applied based on your classpath and what beans you have defined. For example, If you
 * have {@code tomcat-embedded.jar} on your classpath you are likely to want a
 * {@link TomcatEmbeddedServletContainerFactory} (unless you have defined your own
 * {@link EmbeddedServletContainerFactory} bean).
 * <p>
 * Auto-configuration tries to be as intelligent as possible and will back-away as you
 * define more of your own configuration. You can always manually {@link #exclude()} any
 * configuration that you never want to apply (use {@link #excludeName()} if you don't
 * have access to them). You can also exclude them via the
 * {@code spring.autoconfigure.exclude} property. Auto-configuration is always applied
 * after user-defined beans have been registered.
```

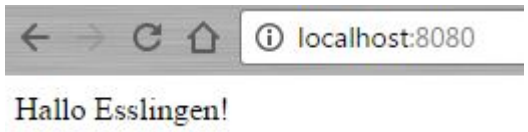
Review – What has Just Happened

```
@RestController
class Hello{
    @RequestMapping("/")
    String greeting() {
        return "Hallo Esslingen!";
    }
}
```

Discoverable through Component Scan, tells Spring to consider it

when handling incoming web requests

Annotation for routing web requests onto specific handler classes and/or handler methods.



Alternative Path to Build and Run

- Go to main directory of maven project

```
MHS@nbmhsH730 MINGW64 ~/Documents/workspace-sts-3.7.3.RELEASE/HelloWorld
$ ls
mvnw*  mvnw.cmd  pom.xml  src/  target/
```

- Execute „mvn package“

```
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO]
[INFO] --- maven-jar-plugin:2.6:jar (default-jar) @ demo ---
[INFO] Building jar: C:\Users\MHS\Documents\workspace-sts-3.7.3.RELEASE\HelloWorld\target\demo-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:1.5.2.RELEASE:repackage (default) @ demo ---
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 6.202 s
[INFO] Finished at: 2017-04-02T12:25:28+02:00
[INFO] Final Memory: 20M/362M
[INFO] ------------------------------------------------------------------------
```

- Will provide a self-contained executable jar file in the target folder
- Execute via

```
MHS@nbmhsH730 MINGW64 ~/Documents/workspace-sts-3.7.3.RELEASE/HelloWorld
$ java -jar target/demo-0.0.1-SNAPSHOT.jar
```

```
2017-04-02 12:32:10.798  INFO 11060 --- [           main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat started on port(s): 9001 (http)
2017-04-02 12:32:10.803  INFO 11060 --- [           main] o.s.c.support.DefaultLifecycleProcessor  : Starting beans in phase 0
2017-04-02 12:32:10.863  INFO 11060 --- [           main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat started on port(s): 9000 (http)
2017-04-02 12:32:10.867  INFO 11060 --- [           main] com.example.HelloWorldApplication        : Started HelloWorldApplication in 4.343 seconds
.751)
```

Dependency Resolution

- Maven transitive dependencies resolved

```xml
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>1.5.2.RELEASE</version>
    <relativePath/> <!-- lookup parent from repository -->
</parent>

<properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <project.reporting.outputEncoding>UTF-8</project.reporting.outputEncoding>
    <java.version>1.8</java.version>
</properties>

<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>

<build>
    <plugins>
        <plugin>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-maven-plugin</artifactId>
        </plugin>
    </plugins>
</build>
```

```
MHS@nbmhsH730 MINGW64 ~/Documents/workspace-sts-3.7.3.RELEASE/HelloWorld
$ mvn dependency:tree
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building HelloWorld 0.0.1-SNAPSHOT
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-dependency-plugin:2.10:tree (default-cli) @ demo ---
[INFO] com.example:demo:jar:0.0.1-SNAPSHOT
[INFO] +- org.springframework.boot:spring-boot-starter-web:jar:1.5.2.RELEASE:compile
[INFO] |  +- org.springframework.boot:spring-boot-starter:jar:1.5.2.RELEASE:compile
[INFO] |  |  +- org.springframework.boot:spring-boot:jar:1.5.2.RELEASE:compile
[INFO] |  |  +- org.springframework.boot:spring-boot-autoconfigure:jar:1.5.2.RELEASE:compile
[INFO] |  |  +- org.springframework.boot:spring-boot-starter-logging:jar:1.5.2.RELEASE:compile
[INFO] |  |  |  +- ch.qos.logback:logback-classic:jar:1.1.11:compile
[INFO] |  |  |  |  \- ch.qos.logback:logback-core:jar:1.1.11:compile
[INFO] |  |  |  +- org.slf4j:jcl-over-slf4j:jar:1.7.24:compile
[INFO] |  |  |  +- org.slf4j:jul-to-slf4j:jar:1.7.24:compile
[INFO] |  |  |  \- org.slf4j:log4j-over-slf4j:jar:1.7.24:compile
[INFO] |  |  \- org.yaml:snakeyaml:jar:1.17:runtime
[INFO] |  +- org.springframework.boot:spring-boot-starter-tomcat:jar:1.5.2.RELEASE:compile
[INFO] |  |  +- org.apache.tomcat.embed:tomcat-embed-core:jar:8.5.11:compile
[INFO] |  |  +- org.apache.tomcat.embed:tomcat-embed-el:jar:8.5.11:compile
[INFO] |  |  \- org.apache.tomcat.embed:tomcat-embed-websocket:jar:8.5.11:compile
[INFO] |  +- org.hibernate:hibernate-validator:jar:5.3.4.Final:compile
[INFO] |  |  +- javax.validation:validation-api:jar:1.1.0.Final:compile
[INFO] |  |  +- org.jboss.logging:jboss-logging:jar:3.3.0.Final:compile
[INFO] |  |  \- com.fasterxml:classmate:jar:1.3.3:compile
[INFO] |  +- com.fasterxml.jackson.core:jackson-databind:jar:2.8.7:compile
[INFO] |  |  +- com.fasterxml.jackson.core:jackson-annotations:jar:2.8.0:compile
[INFO] |  |  \- com.fasterxml.jackson.core:jackson-core:jar:2.8.7:compile
[INFO] |  +- org.springframework:spring-web:jar:4.3.7.RELEASE:compile
[INFO] |  |  +- org.springframework:spring-aop:jar:4.3.7.RELEASE:compile
[INFO] |  |  +- org.springframework:spring-beans:jar:4.3.7.RELEASE:compile
[INFO] |  |  \- org.springframework:spring-context:jar:4.3.7.RELEASE:compile
[INFO] |  \- org.springframework:spring-webmvc:jar:4.3.7.RELEASE:compile
[INFO] |     \- org.springframework:spring-expression:jar:4.3.7.RELEASE:compile
[INFO] \- org.springframework.boot:spring-boot-starter-test:jar:1.5.2.RELEASE:test
[INFO]    +- org.springframework.boot:spring-boot-test:jar:1.5.2.RELEASE:test
[INFO]    +- org.springframework.boot:spring-boot-test-autoconfigure:jar:1.5.2.RELEASE:test
[INFO]    +- com.jayway.jsonpath:json-path:jar:2.2.0:test
[INFO]    |  +- net.minidev:json-smart:jar:2.2.1:test
[INFO]    |  |  \- net.minidev:accessors-smart:jar:1.1:test
[INFO]    |  |     \- org.ow2.asm:asm:jar:5.0.3:test
[INFO]    |  \- org.slf4j:slf4j-api:jar:1.7.24:compile
[INFO]    +- junit:junit:jar:4.12:test
[INFO]    +- org.assertj:assertj-core:jar:2.6.0:test
[INFO]    +- org.mockito:mockito-core:jar:1.10.19:test
[INFO]    |  \- org.objenesis:objenesis:jar:2.1:test
[INFO]    +- org.hamcrest:hamcrest-core:jar:1.3:test
[INFO]    +- org.hamcrest:hamcrest-library:jar:1.3:test
[INFO]    +- org.skyscreamer:jsonassert:jar:1.4.0:test
[INFO]    |  \- com.vaadin.external.google:android-json:jar:0.0.20131108.vaadin1:test
[INFO]    +- org.springframework:spring-core:jar:4.3.7.RELEASE:compile
[INFO]    \- org.springframework:spring-test:jar:4.3.7.RELEASE:test
[INFO] ------------------------------------------------------------------------
```

Overriding Dependencies

- Removed Jackson

```
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
        <exclusions>
            <exclusion>
                <groupId>com.fasterxml.jackson.core</groupId>
                <artifactId>jackson-core</artifactId>
            </exclusion>
            <exclusion>
                <groupId>com.fasterxml.jackson.core</groupId>
                <artifactId>jackson-annotations</artifactId>
            </exclusion>
            <exclusion>
                <groupId>com.fasterxml.jackson.core</groupId>
                <artifactId>jackson-databind</artifactId>
            </exclusion>
        </exclusions>

    </dependency>
```

```
MHS@nbmhsH730 MINGW64 ~/Documents/workspace-sts-3.7.3.RELEASE/HelloWorld
$ mvn dependency:tree
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building HelloWorld 0.0.1-SNAPSHOT
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-dependency-plugin:2.10:tree (default-cli) @ demo ---
[INFO] com.example:demo:jar:0.0.1-SNAPSHOT
[INFO] +- org.springframework.boot:spring-boot-starter-web:jar:1.5.2.RELEASE:compile
[INFO] |  +- org.springframework.boot:spring-boot-starter:jar:1.5.2.RELEASE:compile
[INFO] |  |  +- org.springframework.boot:spring-boot:jar:1.5.2.RELEASE:compile
[INFO] |  |  +- org.springframework.boot:spring-boot-autoconfigure:jar:1.5.2.RELEASE:compile
[INFO] |  |  +- org.springframework.boot:spring-boot-starter-logging:jar:1.5.2.RELEASE:compile
[INFO] |  |  |  +- ch.qos.logback:logback-classic:jar:1.1.11:compile
[INFO] |  |  |  |  \- ch.qos.logback:logback-core:jar:1.1.11:compile
[INFO] |  |  |  +- org.slf4j:jcl-over-slf4j:jar:1.7.24:compile
[INFO] |  |  |  +- org.slf4j:jul-to-slf4j:jar:1.7.24:compile
[INFO] |  |  |  \- org.slf4j:log4j-over-slf4j:jar:1.7.24:compile
[INFO] |  |  \- org.yaml:snakeyaml:jar:1.17:runtime
[INFO] |  +- org.springframework.boot:spring-boot-starter-tomcat:jar:1.5.2.RELEASE:compile
[INFO] |  |  +- org.apache.tomcat.embed:tomcat-embed-core:jar:8.5.11:compile
[INFO] |  |  +- org.apache.tomcat.embed:tomcat-embed-el:jar:8.5.11:compile
[INFO] |  |  \- org.apache.tomcat.embed:tomcat-embed-websocket:jar:8.5.11:compile
[INFO] |  +- org.hibernate:hibernate-validator:jar:5.3.4.Final:compile
[INFO] |  |  +- javax.validation:validation-api:jar:1.1.0.Final:compile
[INFO] |  |  +- org.jboss.logging:jboss-logging:jar:3.3.0.Final:compile
[INFO] |  |  \- com.fasterxml:classmate:jar:1.3.3:compile
[INFO] |  +- org.springframework:spring-web:jar:4.3.7.RELEASE:compile
[INFO] |  |  +- org.springframework:spring-aop:jar:4.3.7.RELEASE:compile
[INFO] |  |  +- org.springframework:spring-beans:jar:4.3.7.RELEASE:compile
[INFO] |  |  \- org.springframework:spring-context:jar:4.3.7.RELEASE:compile
[INFO] |  \- org.springframework:spring-webmvc:jar:4.3.7.RELEASE:compile
[INFO] |     \- org.springframework:spring-expression:jar:4.3.7.RELEASE:compile
[INFO] \- org.springframework.boot:spring-boot-starter-test:jar:1.5.2.RELEASE:test
[INFO]    +- org.springframework.boot:spring-boot-test:jar:1.5.2.RELEASE:test
[INFO]    +- org.springframework.boot:spring-boot-test-autoconfigure:jar:1.5.2.RELEASE:test
[INFO]    +- com.jayway.jsonpath:json-path:jar:2.2.0:test
[INFO]    |  +- net.minidev:json-smart:jar:2.2.1:test
[INFO]    |  |  \- net.minidev:accessors-smart:jar:1.1:test
[INFO]    |  |     \- org.ow2.asm:asm:jar:5.0.3:test
[INFO]    |  \- org.slf4j:slf4j-api:jar:1.7.24:compile
[INFO]    +- junit:junit:jar:4.12:test
[INFO]    +- org.assertj:assertj-core:jar:2.6.0:test
[INFO]    +- org.mockito:mockito-core:jar:1.10.19:test
[INFO]    |  \- org.objenesis:objenesis:jar:2.1:test
[INFO]    +- org.hamcrest:hamcrest-core:jar:1.3:test
[INFO]    +- org.hamcrest:hamcrest-library:jar:1.3:test
[INFO]    +- org.skyscreamer:jsonassert:jar:1.4.0:test
[INFO]    |  \- com.vaadin.external.google:android-json:jar:0.0.20131108.vaadin1:test
[INFO]    +- org.springframework:spring-core:jar:4.3.7.RELEASE:compile
[INFO]    \- org.springframework:spring-test:jar:4.3.7.RELEASE:test
```

Own Version of Dependencies

- Specifying own version

```
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-databind</artifactId>
    <version>2.7.4</version>
</dependency>
```

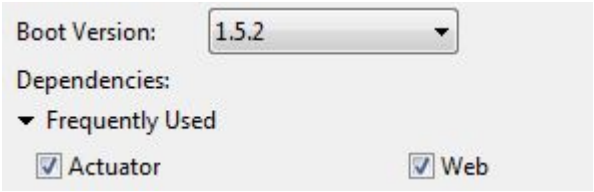
```
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>

    </dependency>

    <dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-databind</artifactId>
    </dependency>
```

Overriding managed version 2.8.7 for jackson-databind

```
$ mvn dependency:tree -Dincludes=com.fasterxml.jackson.core
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building HelloWorld 0.0.1-SNAPSHOT
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] --- maven-dependency-plugin:2.10:tree (default-cli) @ demo ---
[INFO] com.example:demo:jar:0.0.1-SNAPSHOT
[INFO] \- com.fasterxml.jackson.core:jackson-databind:jar:2.7.4:compile
[INFO]    +- com.fasterxml.jackson.core:jackson-annotations:jar:2.8.0:compile
[INFO]    \- com.fasterxml.jackson.core:jackson-core:jar:2.8.7:compile
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 1.505 s
[INFO] Finished at: 2017-03-30T11:23:23+02:00
[INFO] Final Memory: 22M/619M
[INFO] ------------------------------------------------------------------------
```

Adding new Dependencies – Using the Actuator

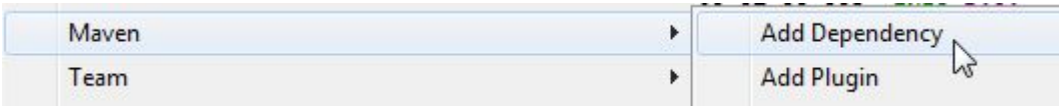
- Create a new Spring Starter Project and select what you need

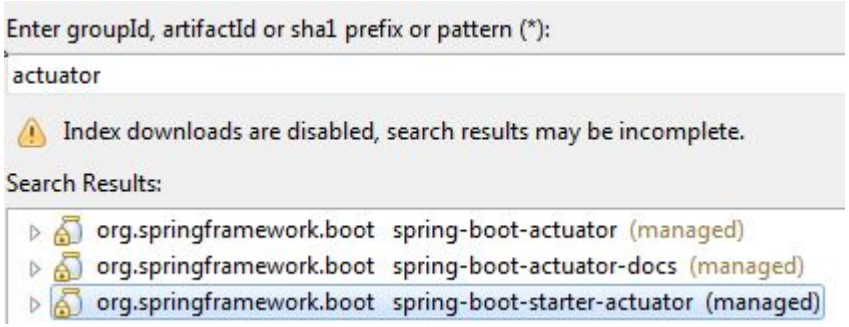


- Validate the new Maven or Gradle configuration (and copy/paste to existing one)

```
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-actuator</artifactId>
    </dependency>
```

- Add the Maven dependency via Eclipse





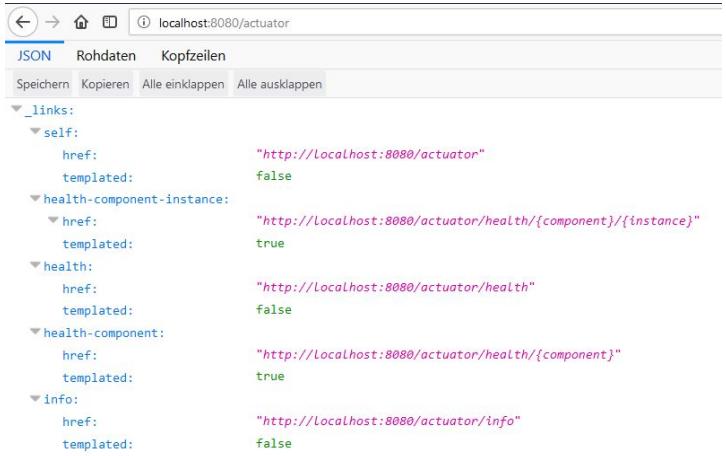
The Actuator

- Spring Boot Actuator is a sub-project / feature that is part of the spring boot framework
  - It is not a separate framework to be added to your applications
- Main purpose of this feature is to provide various useful metrics about the applications.
  - It is very helpful in the production environment to check the various metrics like health of your application, configurations, error page, version details, etc.
- Actuator is supported out of the box within spring boot applications.
  - You just have to add the dependency to enable the actuator
  - The default configurations are enabled if you are not providing any application specific configurations
- Actuator makes the metrics are accessed through different endpoints like /error, /metrics, /beans, /info, etc.
  - End points are HTTP URLs that can be accessed through your browser.

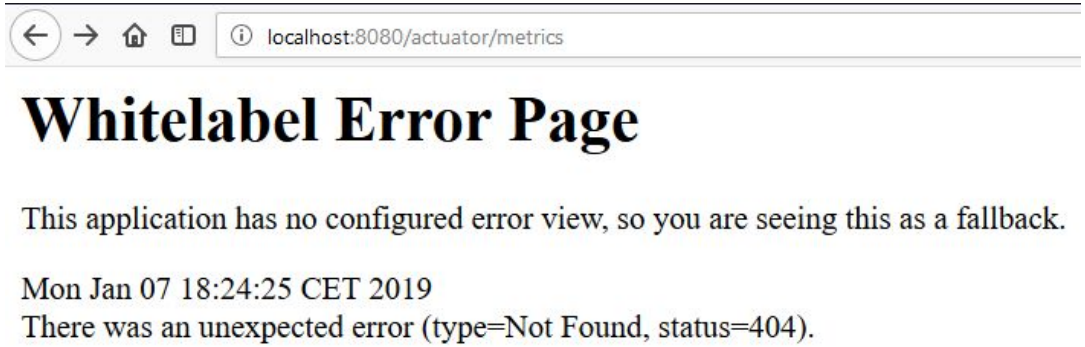
(http://javabeat.net/spring-boot-actuator/)

Actuator

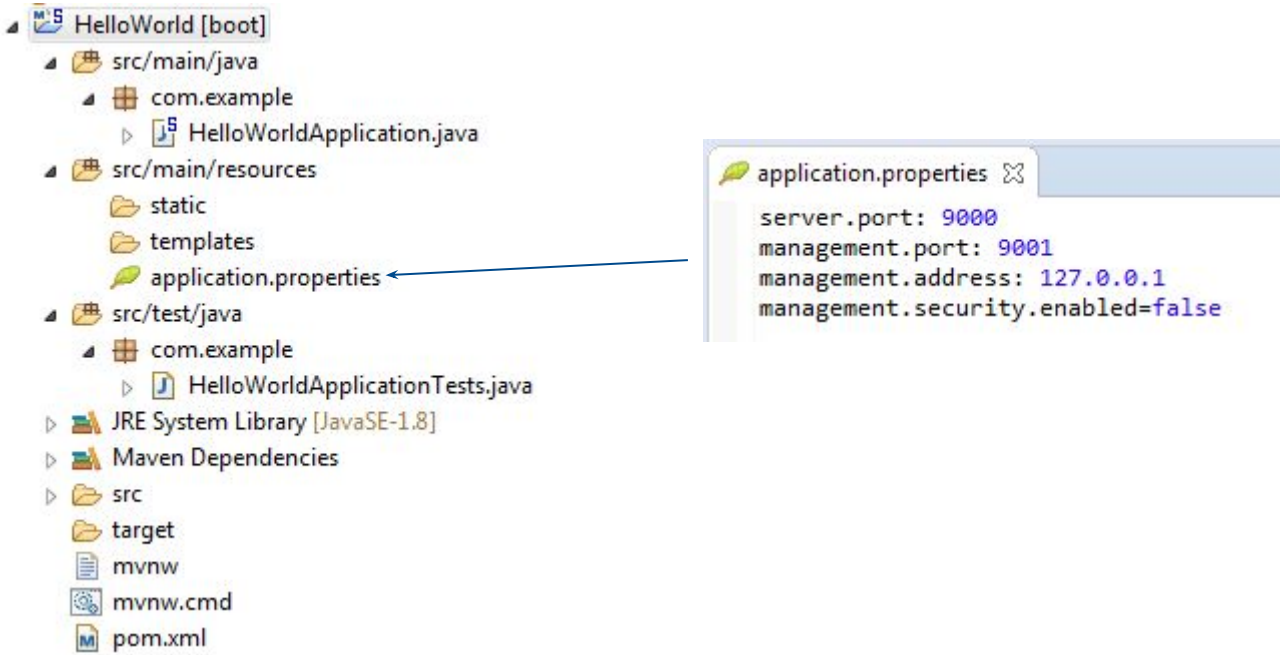
- By default the actuator will listen on the same port as the application itself
- You can invoke the functionality be calling the endpoints
- Most of the endpoints are secured so you will not be able to invoke them directly
- /actuator will show the current available endpoints



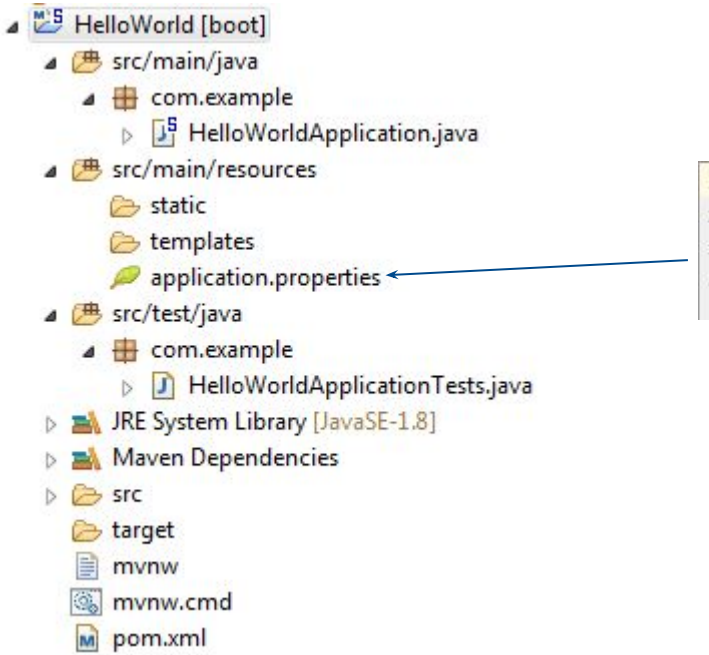
- Unavailable endpoints will return an error page



Disable Security - Deprecated!



Disable Security - Since Spring Boot 2.0




```
1
2  # endpoints are generally available, but not exposed
3  # expose only endpoints you are interested in
4  management.endpoints.web.exposure.include=health, info, metrics, env, beans
```

Actuator – Live Metrics


```
{
    "mem": 711663,
    "mem.free": 549106,
    "processors": 8,
    "instance.uptime": 21448,
    "uptime": 29658,
    "systemload.average": -1,
    "heap.committed": 641536,
    "heap.init": 524288,
    "heap.used": 92429,
    "heap": 7431168,
    "nonheap.committed": 72000,
    "nonheap.init": 2496,
    "nonheap.used": 70128,
    "nonheap": 0,
    "threads.peak": 24,
    "threads.daemon": 17,
    "threads.totalStarted": 27,
    "threads": 20,
    "classes": 8739,
    "classes.loaded": 8739,
    "classes.unloaded": 0,
    "gc.ps_scavenge.count": 6,
    "gc.ps_scavenge.time": 172,
    "gc.ps_marksweep.count": 2,
    "gc.ps_marksweep.time": 224,
    "httpsessions.max": -1,
    "httpsessions.active": 0
}
```

Actuator - Endpoints

ID	Description	Enabled by default
`auditevents`	Exposes audit events information for the current application.	Yes
`beans`	Displays a complete list of all the Spring beans in your application.	Yes
`caches`	Exposes available caches.	Yes
`conditions`	Shows the conditions that were evaluated on configuration and auto-configuration classes and the reasons why they did or did not match.	Yes
`configprops`	Displays a collated list of all `@ConfigurationProperties`.	Yes
`env`	Exposes properties from Spring's `ConfigurableEnvironment`.	Yes
`flyway`	Shows any Flyway database migrations that have been applied.	Yes
`health`	Shows application health information.	Yes
`httptrace`	Displays HTTP trace information (by default, the last 100 HTTP request-response exchanges).	Yes
`info`	Displays arbitrary application info.	Yes
`integrationgraph`	Shows the Spring Integration graph.	Yes
`loggers`	Shows and modifies the configuration of loggers in the application.	Yes
`liquibase`	Shows any Liquibase database migrations that have been applied.	Yes
`metrics`	Shows 'metrics' information for the current application.	Yes
`mappings`	Displays a collated list of all `@RequestMapping` paths.	Yes
`scheduledtasks`	Displays the scheduled tasks in your application.	Yes
`sessions`	Allows retrieval and deletion of user sessions from a Spring Session-backed session store. Not available when using Spring Session's support for reactive web applications.	Yes
`shutdown`	Lets the application be gracefully shutdown.	No
`threaddump`	Performs a thread dump.	Yes

Common Application Properties

- https://docs.spring.io/spring-boot/docs/current/reference/html/common-application-properties.html

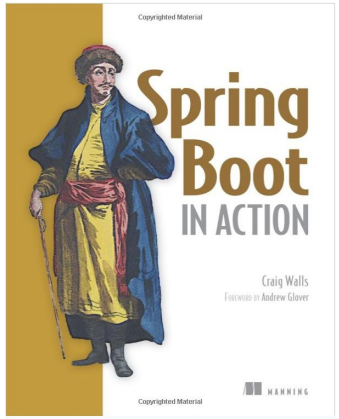
```
# LOGGING
logging.config= # Location of the logging configuration file. For instance `classpath:logback.xml` for Logback
logging.exception-conversion-word=%wEx # Conversion word used when logging exceptions.
logging.file= # Log file name. For instance `myapp.log`
logging.level.*= # Log levels severity mapping. For instance `logging.level.org.springframework=DEBUG`
logging.path= # Location of the log file. For instance `/var/log`
```

```
# EMBEDDED SERVER CONFIGURATION (ServerProperties)
server.address= # Network address to which the server should bind to.
server.compression.enabled=false # If response compression is enabled.
server.compression.excluded-user-agents= # List of user-agents to exclude from compression.
```

```
# HTTP encoding (HttpEncodingProperties)
spring.http.encoding.charset=UTF-8 # Charset of HTTP requests and responses. Added to the "Content-Type" header if not set explicitly.
spring.http.encoding.enabled=true # Enable http encoding support.
spring.http.encoding.force= # Force the encoding to the configured charset on HTTP requests and responses.
spring.http.encoding.force-request= # Force the encoding to the configured charset on HTTP requests. Defaults to true when "force" has not been
spring.http.encoding.force-response= # Force the encoding to the configured charset on HTTP responses.
spring.http.encoding.mapping= # Locale to Encoding mapping.
```

```
# H2 Web Console (H2ConsoleProperties)
spring.h2.console.enabled=false # Enable the console.
spring.h2.console.path=/h2-console # Path at which the console will be available.
spring.h2.console.settings.trace=false # Enable trace output.
spring.h2.console.settings.web-allow-others=false # Enable remote access.
```

Sources



- Spring Docs:
  - https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/
- Tutorial: http://javabeat.net/spring-tutorials/
- Tutorials: http://www.baeldung.com/
- http://docs.spring.io/spring-boot/docs/current/reference/html/getting-started-first-application.html

Questions?