

Spring Boot 入門

ハンズオン

Java Doでしよう #09

講師役：haruki-ueno



環境セッティング

- ・ 教科書のURL
- ・ ハンズオン[環境のセッティング]の手順URL
- ・ 下記の手順ぐらいまで順次進めてください
 1. https://github.com/java-do/20170422_seminar にブラウザでアクセスしよう

SpringBoot

Mavenでプロジェクトを作成

- ・ 以下のコマンドを実行する
- ・

```
$ mvn -B archetype:generate  
-DgroupId=jp.javado.springboot  
-DartifactId=springboot-handson1  
-Dversion=1.0.0-SNAPSHOT  
-DarchetypeArtifactId=  
maven-archetype-quickstart
```
- ・ 入力する際は改行なしの一行（表示の都合で改行されています）

Mavenのプロジェクト確認

- ・ springboot-handson1 のフォルダができていることを確認
- ・ フォルダ内に、下記のフォルダ/ファイルが存在
 - ・ src
 - ・ pom.xml

作成したプロジェクトを統合 開発環境にインポート

- ・ IntelliJの場合
 - ・ “Project from Existing Sources”
 - > “Import project from external model”
 - > “Maven”
 - ・ “Import Maven projects automatically”をチェック
 - ・ あとはデフォルト設定で作成
- ・ Eclipseの場合

pom.xmlを編集（赤字を追加）

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  ~省略~
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>1.5.2.RELEASE</version>
  </parent>
  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
      <groupId>junit</groupId>
    ~省略~
    </dependency>
  </dependencies>
  <build>
    <plugins>
      <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
      </plugin>
    </plugins>
  </build>
  <properties>
    <java.version>1.8</java.version>
  </properties>
</project>
```

赤字は
SpringBootの設定

SpringBootの起動クラス

jp.javado.springbootパッケージに、
JavaDoSpringApplicationクラスを作成してみましょう

```
package jp.javado.springboot;

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

@SpringBootApplication
public class JavaDoSpringApplication {

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


起動確認

- ・ javaDoSpringApplicationクラスを実行してみましょう
 - ・ コマンドラインの場合
 - ・ `$ mvn spring-boot:run`
 - ・ IntelliJの場合
 - ・ Eclipseの場合

起動確認

- ・ 正常に起動すると以下が出ます

```
dhcp7:springboot-handson1 ueno$ mvn spring-boot:run
```

```
[INFO] Scanning for projects...
```

```
~~中略~~
```

```
[INFO] --- spring-boot-maven-plugin:1.5.2.RELEASE:run (default-cli) @ springboot-handson1 ---
```

```

  .
 / \  / ____ \  _ _   _  ( _ )  _ _   _  \ \ / \  \ \ / \
( ( ) \ ____ | | | | | | | | | | \ /  _/ | \ \ / \  \ \ / \
 \ \ /  ____ ) | | | | | | | | | | ( _ | | ) ) ) )
  '   | ____ | ._| | | | | | | | | \__, | / / / / /
=====|_|=====|_|_/=/ / / / /
:: Spring Boot ::                (v1.5.2.RELEASE)

```

```
2017-04-17 13:38:16.428 INFO 9666 --- [main] j.j.springboot.JavaDoSpringApplication : Starting
JavaDoSpringApplication on dhcp7.kk.pilot.chitose.ac.jp with PID 9666 (/Users/ueno/javado/springboot-
handson1/target/classes started by ueno in /Users/ueno/javado/springboot-handson1)
```

```
~~中略~~
```

```
2017-04-17 13:38:19.008 INFO 9666 --- [main] o.s.j.e.a.AnnotationMBeanExporter : Registering
beans for JMX exposure on startup
```

```
2017-04-17 13:38:19.086 INFO 9666 --- [main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat
started on port(s): 8080 (http)
```

```
2017-04-17 13:38:19.094 INFO 9666 --- [main] j.j.springboot.JavaDoSpringApplication : Started
JavaDoSpringApplication in 3.043 seconds (JVM running for 7.424)
```

ここまでの流れ

SpringでのREST

RestControllerを作ってみよう

- ・ 下記のHelloSpringRestControllerクラスを作ってみましょう

```
package jp.javado.springboot;

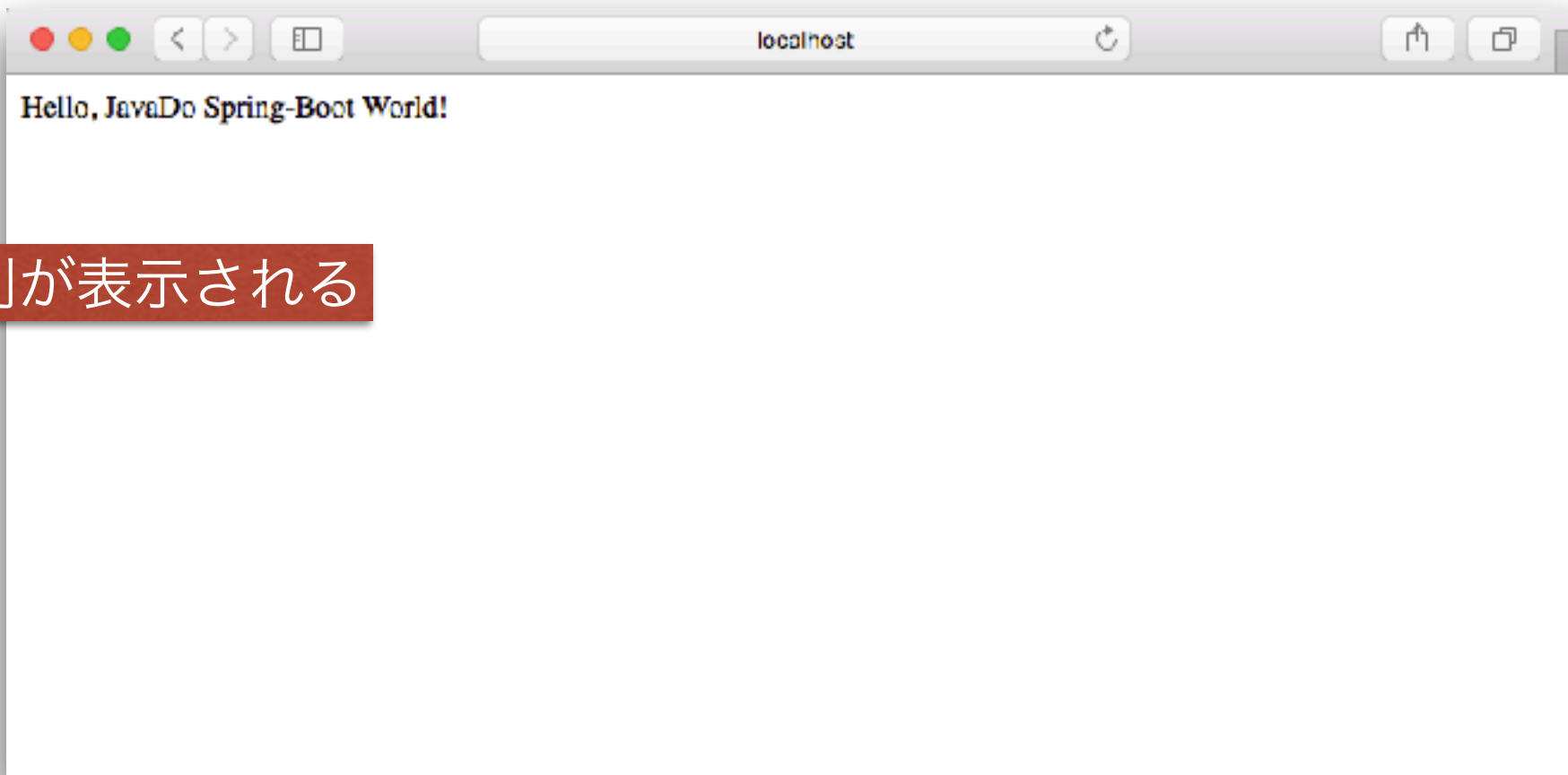
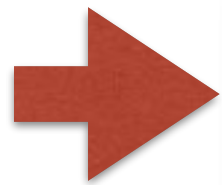
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class HelloSpringRestController {

    @RequestMapping("/hello")
    public String hello() {
        return "Hello, JavaDo Spring-Boot World!";
    }
}
```

ブラウザでアクセス

- ・ コマンドで起動
 - ・ `$ mvn spring-boot:run`
- ・ ブラウザで以下のURLにアクセス
 - ・ <http://localhost:8080/hello>



書いた文字列が表示される

これらの仕組み

画面を作って見ましょう

- ・ pom.xmlに赤字の箇所を追加

```
<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-thymeleaf</artifactId>
  </dependency>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>3.8.1</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```


Web画面用Controller

- ・ 以下のIndexControllerクラスを
jp.javado.springbootパッケージに作成しましょう

```
package jp.javado.springboot;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;

@Controller
public class IndexController {

    @RequestMapping("/web/index")
    public String index() {
        return "index";
    }
}
```

templates フォルダ作成

- ・ `springboot-handson1/src/main/resources` フォルダ配下に `templates` フォルダを作成
- ・ もし、`springboot-handson1/src/main` 以下に `resources` フォルダがない場合は作成しましょう

htmlの作成

- ・ src/main/resources/templates配下にindex.htmlを作成しましょう

```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8"/>
  <title>Index</title>
</head>
<body>
<h1>Hello Web page</h1>
<p>This is Thymeleaf page on Spring Boot.</p>
</body>
</html>
```

Controller側で設定した値を 表示

- ・ では、Controller側とhtmlでの値の受け渡しを作ってみましょう
- ・ IndexControllerを編集
- ・ index.htmlを編集

Controllerの編集

```
package jp.javado.springboot;

import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.servlet.ModelAndView;

@Controller
public class IndexController {

    @RequestMapping("/web/index")
    public ModelAndView index(ModelAndView modelAndView) {
        modelAndView.setViewName("index");
        modelAndView.addObject("message1", "Controllerで設定した文字列");
        return modelAndView;
    }
}
```

修正：赤字の箇所

ModelAndViewクラスを使うことで、
Controllerからhtmlに値を渡すことができる