

# Thymeleaf with Spring Boot





# What is Thymeleaf?

- Thymeleaf is a Java templating engine



[www.thymeleaf.org](http://www.thymeleaf.org)

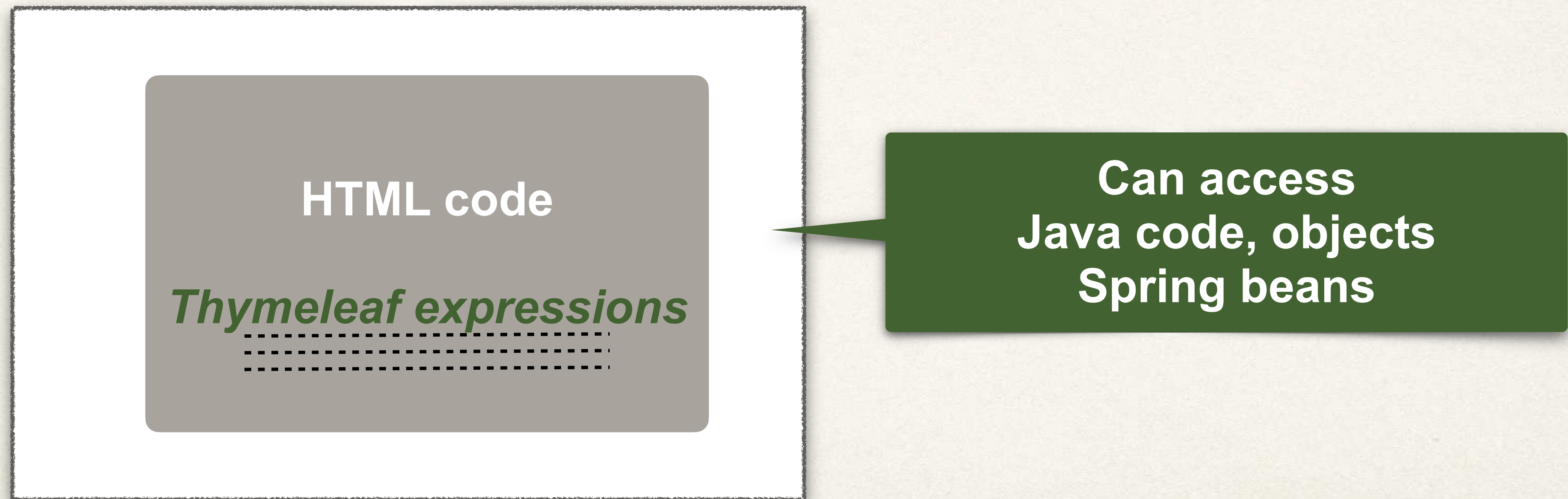
**Separate project  
Unrelated to spring.io**

- Commonly used to generate the HTML views for web apps
- However, it is a general purpose templating engine
- Can use Thymeleaf outside of web apps (more on this later)



# What is a Thymeleaf template?

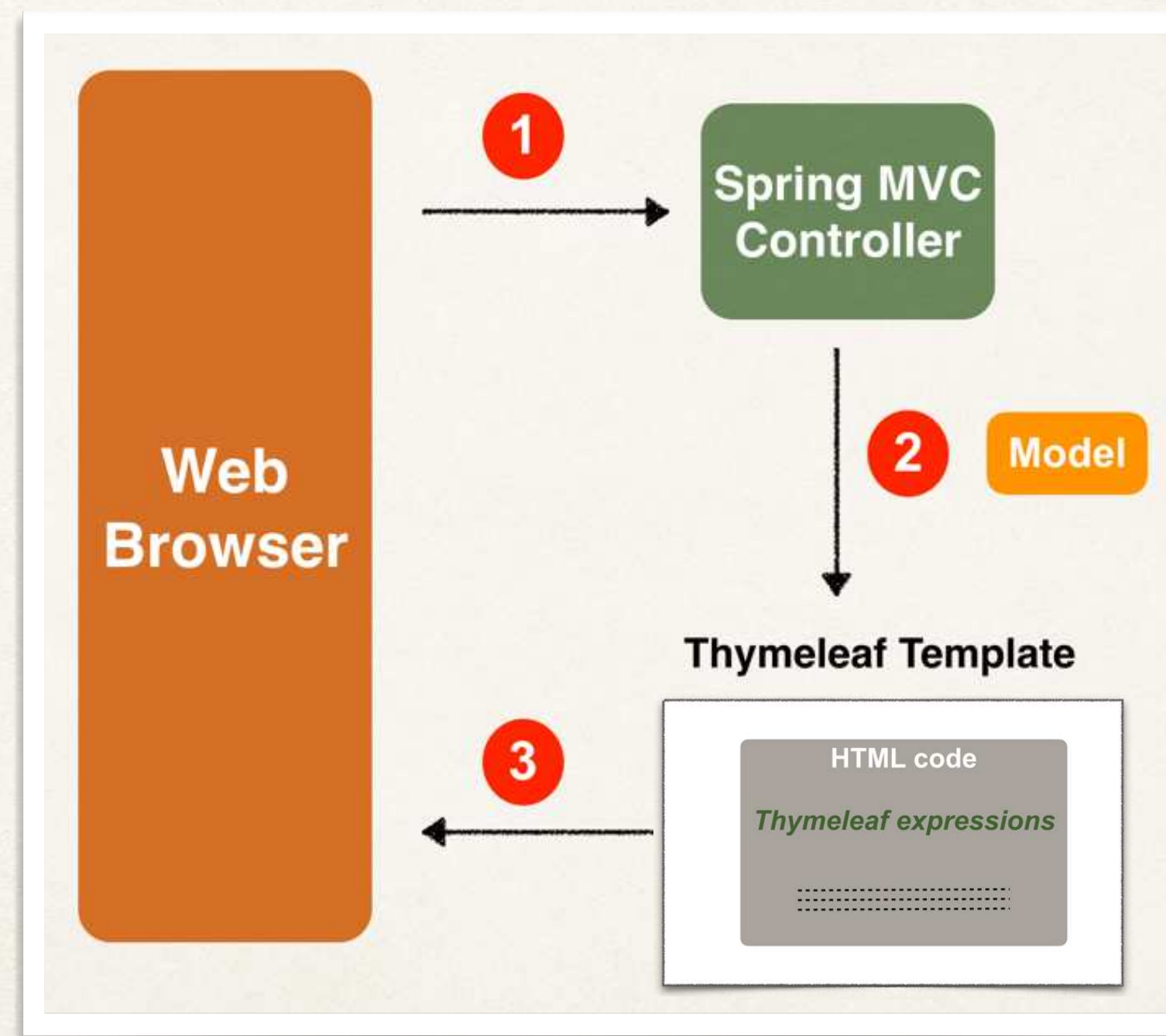
- Can be an HTML page with some Thymeleaf expressions
- Include dynamic content from Thymeleaf expressions





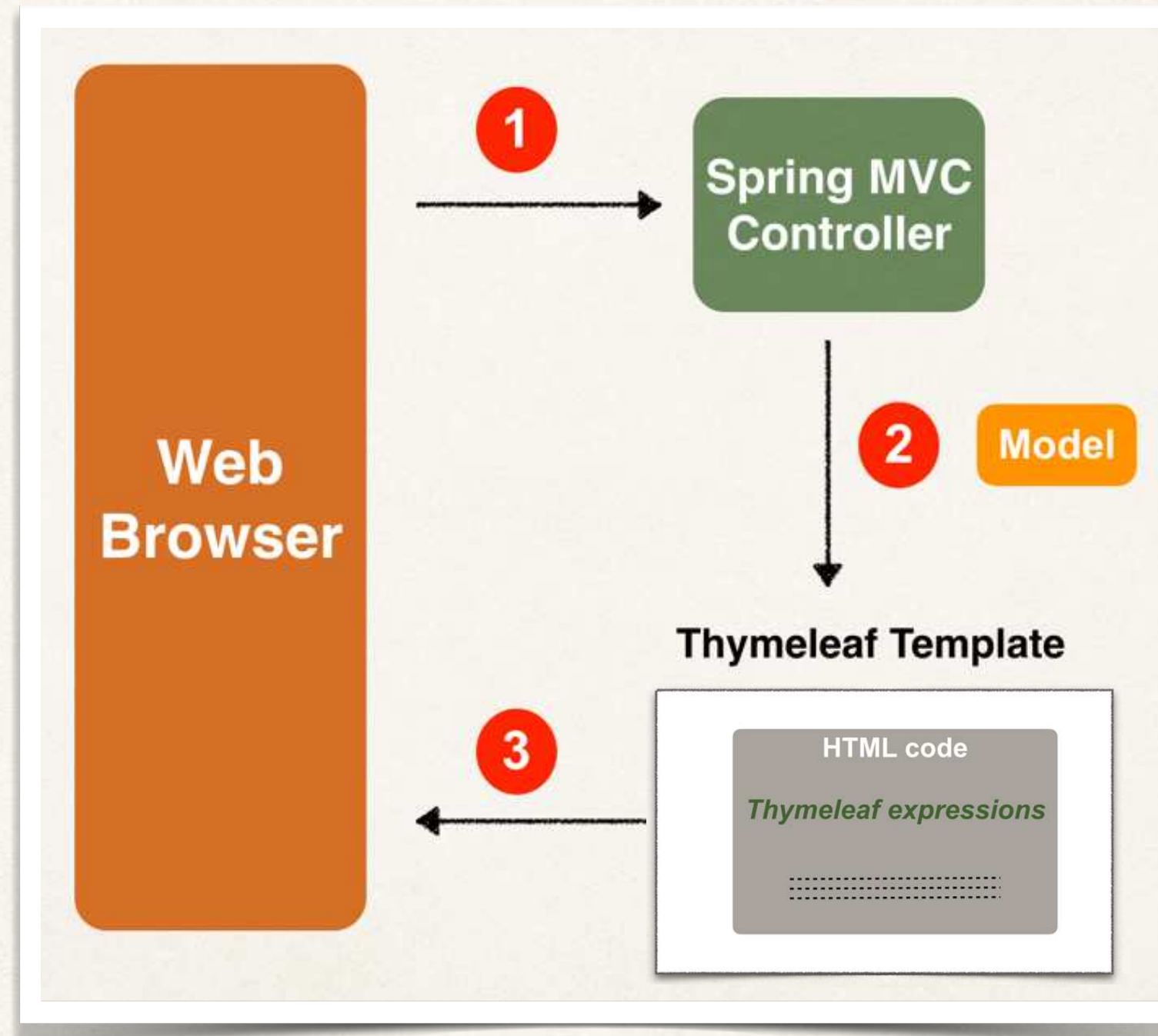
# Where is the Thymeleaf template processed?

- In a web app, Thymeleaf is processed on the server
- Results included in HTML returned to browser





# Thymeleaf Demo



Output

Time on the server is 20xx-03-30T11:27:52.297247



# Development Process


Step-By-Step

1. Add Thymeleaf to Maven POM file
2. Develop Spring MVC Controller
3. Create Thymeleaf template



# Step 1: Add Thymeleaf to Maven pom file

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



Based on this,  
Spring Boot will auto configure to  
use Thymeleaf templates

## Dependencies

### Spring Web WEB

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

### Thymeleaf TEMPLATE ENGINES



A modern server-side Java template engine for both web and standalone environments. Allows HTML to be correctly displayed in browsers and as static prototypes.



# Step 2: Develop Spring MVC Controller

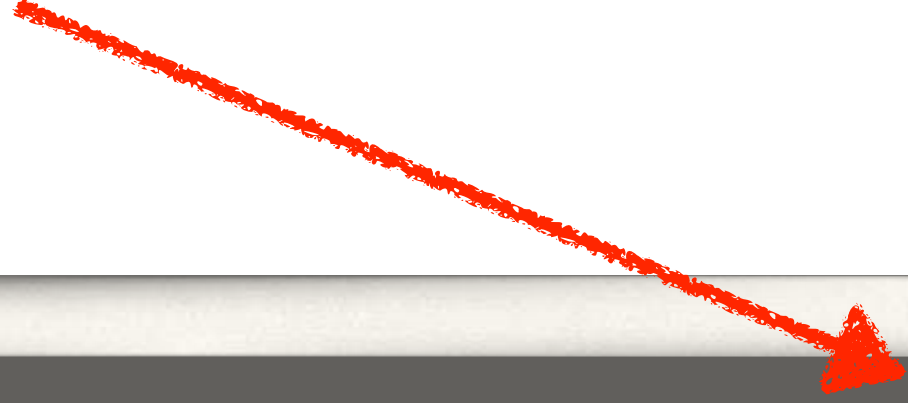
File: DemoController.java

```
@Controller
public class DemoController {

    @GetMapping("/")
    public String sayHello(Model theModel) {

        theModel.addAttribute("theDate", java.time.LocalDateTime.now());

        return "helloworld";
    }
}
```



src/main/resources/templates/helloworld.html



# Where to place Thymeleaf template?

- In Spring Boot, your Thymeleaf template files go in
  - **src/main/resources/templates**
- For web apps, Thymeleaf templates have a **.html** extension



# Step 3: Create Thymeleaf template

Thymeleaf accesses "theDate" from the Spring MVC Model

File: src/main/resources/templates/helloworld.html

```
<!DOCTYPE HTML>
<html xmlns:th="http://www.thymeleaf.org">
<head> ... </head>

<body>
  <p th:text="'Time on the server is ' + ${theDate}" />
</body>

</html>
```

Thymeleaf expression

File: DemoController.java

```
@Controller
public class DemoController {

    @GetMapping("/")
    public String sayHello(Model theModel) {

        theModel.addAttribute("theDate", java.time.LocalDateTime.now());

        return "helloworld";
    }
}
```

1

2

Time on the server is 20xx-03-30T11:27:52.297247



# Step 3: Create Thymeleaf template

Thymeleaf accesses "theDate" from the Spring MVC Model

File: src/main/resources/templates/helloworld.html

```
<!DOCTYPE HTML>
<html xmlns:th="http://www.thymeleaf.org">
<head> ... </head>

<body>
  <p th:text="'Time on the server is ' + ${theDate}" />
</body>

</html>
```

Thymeleaf expression

File: DemoController.java

```
@Controller
public class DemoController {

    @GetMapping("/")
    public String sayHello(Model theModel) {

        theModel.addAttribute("theDate", java.time.LocalDateTime.now());

        return "helloworld";
    }
}
```

1

2

Time on the server is 20xx-03-30T11:27:52.297247



# Additional Features

- Looping and conditionals
- CSS and JavaScript integration
- Template layouts and fragments

[www.thymeleaf.org](http://www.thymeleaf.org)



# CSS and Thymeleaf





# Let's Apply CSS Styles to our Page

## Before



## After



```
font-style: italic;  
color: green;
```



# Using CSS with Thymleaf Templates

- You have the option of using
  - Local CSS files as part of your project
  - Referencing remote CSS files
- We'll cover both options in this video



# Development Process

**Step-By-Step**

1. Create CSS file
2. Reference CSS in Thymeleaf template
3. Apply CSS style

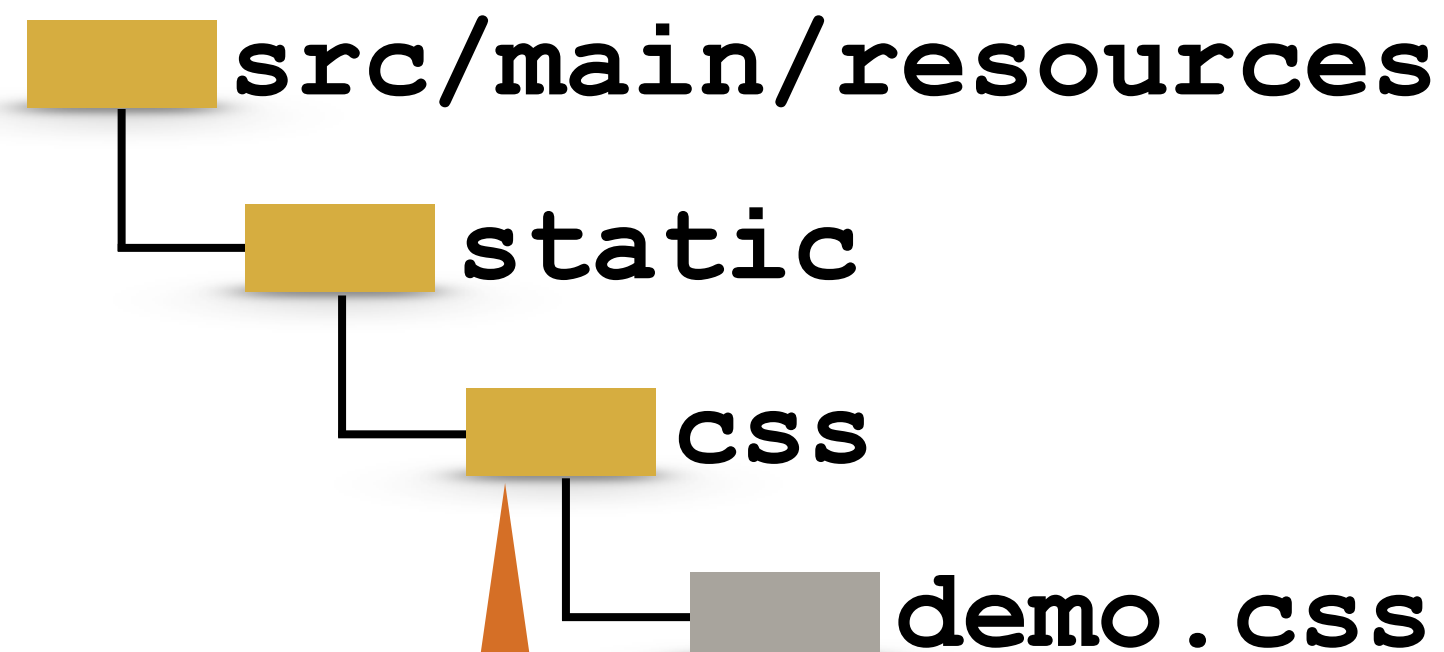


# Step 1: Create CSS file

- Spring Boot will look for static resources in the directory

- **src/main/resources/static**

You can create your own custom sub-directories  
`static/css`  
`static/images`  
`static/js`  
etc ...



Can be any sub-directory name

File: demo.css

```
.funny {  
    font-style: italic;  
    color: green;  
}
```



# Step 2: Reference CSS in Thymeleaf template

File: helloworld.html

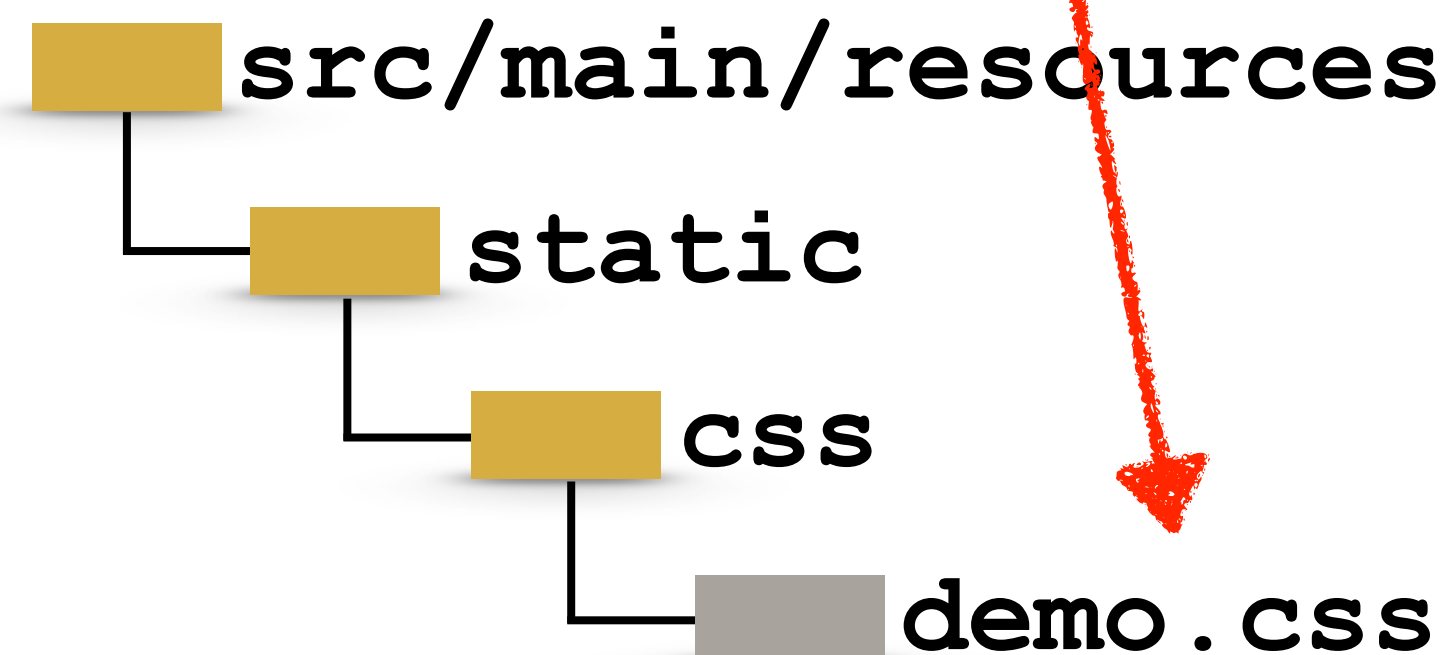
```
<head>
  <title>Thymeleaf Demo</title>

  <!-- reference CSS file -->
  <link rel="stylesheet" th:href="@{/css/demo.css}" />

</head>
```

@ symbol

Reference context path of your application  
(app root)





# Step 3: Apply CSS

File: helloworld.html

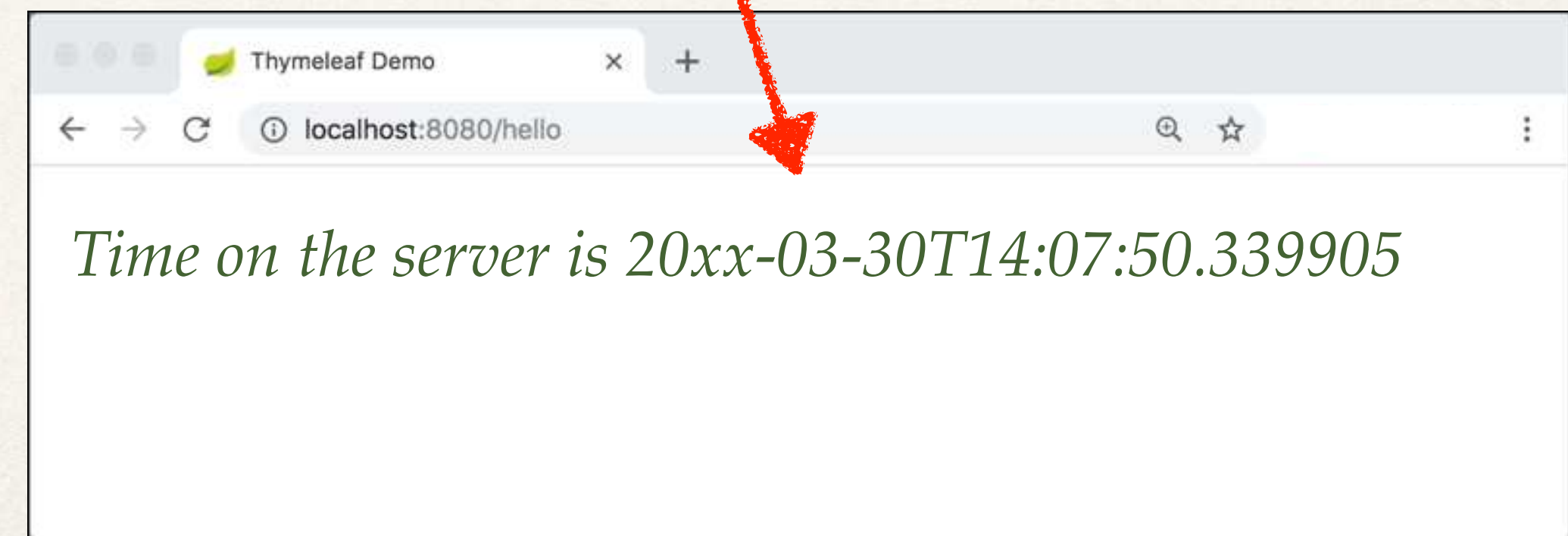
```
<head>
  <title>Thymeleaf Demo</title>

  <!-- reference CSS file -->
  <link rel="stylesheet" th:href="@{/css/demo.css}" />
</head>

<body>
  <p th:text="'Time on the server is ' + ${theDate}" class="funny" />
</body>
```

File: demo.css

```
.funny {
  font-style: italic;
  color: green;
}
```





# Other search directories

Spring Boot will search following directories for static resources:

**`/src/main/resources`**

1. `/META-INF/resources`
2. `/resources`
3. `/static`
4. `/public`

Search order: top-down



# 3rd Party CSS Libraries - Bootstrap

- Local Installation
- Download Bootstrap file(s) and add to `/static/css` directory

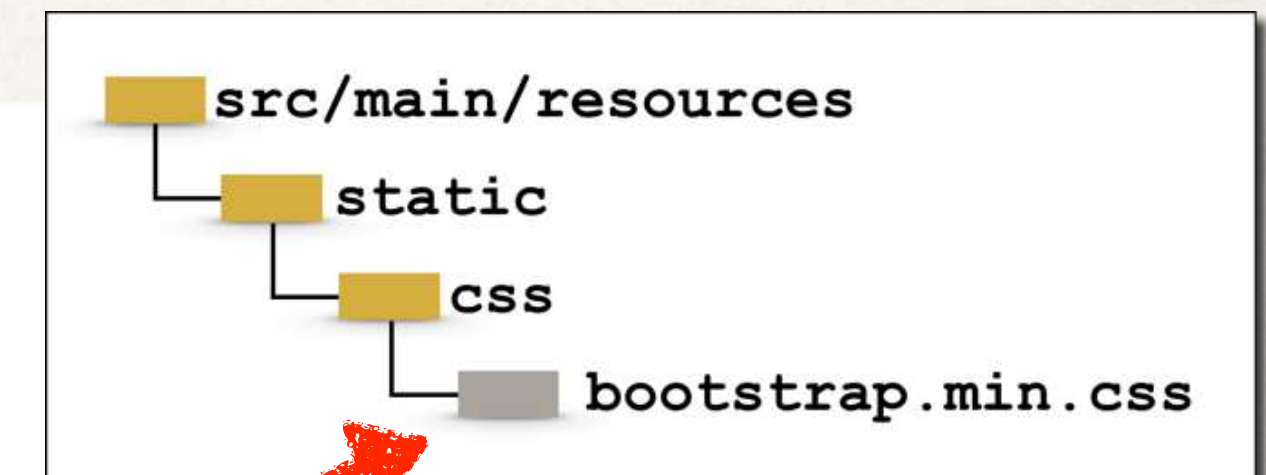
```
<head>
```

```
... ..
```

```
<!-- reference CSS file -->
```

```
<link rel="stylesheet" th:href="@{/css/bootstrap.min.css}" />
```

```
</head>
```





# 3rd Party CSS Libraries - Bootstrap

- Remote Files

```
<head>
... ..

<!-- reference CSS file -->
<link rel="stylesheet"
      href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" />

... ..
</head>
```