Date : 24/02/2021
Spring Boot 9AM
Mr. RAGHU

UI: JSP (Java Server Page)

JSP/Servlets are heavy weighted objects. ie they need more memory at runtime in web container, objects created like: servlet object, config, context, request, response, ...etc

=> So, better to writes less Servlet/JSPs in App.
=> To write any web application/web services (works using HTTP protocol)
using Java Tech/F/w, at least one Servlet is must.

=> So, finally Spring WEB MVC has given one Servlet : DispatcherServlet.
=> But for UIs we are using JSPs, they are internally sevlets which needs
more memory.

Thymeleaf UI

-> Java based UI Technology.
-> It is light weight (ie need less memory to run file)
-> Thymeleaf supports both static and dynamic tags.
But only dynamic tags are converted to java code,executed
output rendered.

---

a) Symbols used in Thymeleaf

@ --- URL/Path/Location

$ --- Read Data from Model/Container

* --- Link Form Input <---with--> Model class Variable

--- Read Object given by thymeleaf
(session, number, request, context)

---

*) HTML tags and thymeleaf tags must have difference, using one prefix(th).
*) Those(thymeleaf) tags are executed using Thymeleaf Runtime
by using namespace.

*) Thymeleaf file extension is .html , looks like

```
<html xmlns:th="https://www.thymeleaf.org/">
  <head>____</head>
  <body>

  </body>
</html>
```

*) Spring boot supports integration with thymeleaf.

```
<dependency>
    <groupId>org.springframework.boot</groupId>
```

```
    <artifactId>spring-boot-starter-thymeleaf</artifactId>
</dependency>

*) Spring boot has provided
    prefix=/src/main/resources/templates   [you can modify]
    suffix=.html                           [fiexd one]
 (no need to provide those in application)
-----------------------------------------------------
th:text ="${keyFromModel}"  This is used to read data from
      Controller/Container/Model and print at UI

--example code--
Name : SpringBoot2WebMvcThyemeleafEx
Dep  : Spring web, Thymeleaf

a. application.properties
server.port=80

b. Controller
package in.nareshit.raghu.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;

@Controller
@RequestMapping("/product")
public class ProductController {

        @GetMapping("/info")
        public String showInfo(Model model) {
                model.addAttribute("pid", 10);
                model.addAttribute("pcode", "PEN");

                return "Products";
        }
}

c. UI : Products.html
> Right click on 'templates' folder > new > other > search and select
HTML
> Next > Enter name > finish
----------Products.html--------------------
<html xmlns:th="https://www.thymeleaf.org/">
<body>
<h1> PRODUCT DATA IS </h1>
<span th:text="${pid}"></span>
<span th:text="${pcode}"></span>
</body>
</html>
-----------------------------------------
*)Spring Boot DevTools (Use in Dev Env only)
  It avoid application re-start for small code modification
  while developing application.

--pom.xml--
```

```
<dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-devtools</artifactId>
        <scope>runtime</scope>
</dependency>

*) LiveReload server is running on port 35729
   and updates back to tomcat/or any..etc
-------------------------------------------
*) HyperLink (<a> tag ) in Thymeleaf

  th:href="@{/path/path/path}

Ex:
<a th:href="@{/product/data}">DATA</a>

*) Image tag in Thymeleaf
<img th:src="@{/images/nit.png}" width="200" height="80"/>

*) CSS Link File
<link rel="stylesheet" th:href="@{/css/mydesign.css}">


*) JS Link File
<script type="text/javascript" th:src="@{/js/validate.js}"></script>

*) All CSS/JS/Images must under static folder only.
-------------------------------------------


============full code============
--Products.html--
<html xmlns:th="https://www.thymeleaf.org/">
<head>
<script type="text/javascript" th:src="@{/js/validate.js}"></script>
<link rel="stylesheet" th:href="@{/css/mydesign.css}">
</head>
<body>
<img th:src="@{/images/nit.png}" width="200" height="80"/>
<h1> PRODUCT DATA IS </h1>
<span class="myd" th:text="${pid}"></span>
<span th:text="${pcode}"></span>

<a th:href="@{/product/data}">DATA</a>
</body>
</html>

---static/css/mydesign.css---
.myd{
        color: red;
        font-family: fantasy;
        font-size: 65
}
----static/js/validate.js-------
alert("WELCOME");

*) Place one image : static/images/nit.png
```

*) Controller

```
package in.nareshit.raghu.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;

@Controller
@RequestMapping("/product")
public class ProductController {

	@GetMapping("/info")
	public String showInfo(Model model) {
		model.addAttribute("pid", 96960);
		model.addAttribute("pcode", "PEN");

		return "Products";
	}

	@GetMapping("/data")
	public String showData() {

		return "DataPage";
	}
}
```

Ex URL:

http://localhost/product/info