```
                    Date : 25/02/2021
                    Spring Boot 9AM
                      Mr. RAGHU
              ------------------------
                    Spring Boot WEB MVC
                       Thymeleaf UI
```

*) Thymeleaf UI is a light weight (less memory) technology
*) Easy to code, works faster.
*) Only Dynamic content is converted to java code.
*) It is Java Based UI. Can't use for non-java apps.
*) Spring boot supports integration with WEB MVC
   prefix = /templates/ folder
   suffix = .html
----code------------------------------------
$ --- Read data from Model memory

<div th:text="${sid}"> => Read sid from Model memory
  and print at browser. [th:text --> print data]

@ -- Location/Path/URL

<a th:href="@{/controllerPath/methodPath}" > ___
<form th:action="@{/controllerPath/methodPath}" > ...</form>
<script th:src="@{/js/myvalidate.js}" ...
<link th:href="@{/css/myui.css}" ...
<img th:src="@{/images/mydesign.png}" ..

*) Thymeleaf files are placed in templates folder
  and images/css/javascript..etc are placed in static folder

       Form Using Thymeleaf (Bi-Directional Form)

HTML Code :
<input ------- name="empName" id="empName" />

Thymeleaf Code:
<input ------- th:field="*{variableName}" />
<input ------- th:field="*{empName}" />

HTML Code:
 <form action="___" method="POST" > ...</form>

Thymeleaf Code:
<form th:action="@{/fullpath}" method="POST"
   th:object="${objectNameSentFromControllerUsingModel}">

</form>
----
*) To fill data in Form we should pass object using
Controller Model memory, else Exception:
IllegalStateException: Neither BindingResult nor
   plain target object for bean name .

S#1 Create one Map at Controller (or read data from DB)

Map<Integer,String> myprjs =
Map.of(10,"P1",11,"P2",12,"P3",13,"P4",14,"P5");

S#2 Send Map to UI using Model memory
model.addAttribute("myprjs", myprjs);

S#3 Display Map as DropDown
   MapValue--Label Text, mapKey as Value

<option th:each="ob:${myprjs}"
        th:text="${ob.value}"
        th:value="${ob.key}">

----------------Full Code----------------------------
1. Name: SpringBoot2ThymeleafForm
   Dep : Web, Lombok, Devtools, Thymeleaf

2. Model class
package in.nareshit.raghu.model;

import java.util.List;

import lombok.Data;
```

```java
@Data
public class Employee {

        private Integer empId;
        private String empName;
        private Double empSal;
        private String empGen;
        private String empProj;
        private String empAddr;
        private List<String> empLangs;
}
```

3. Controller
```java
package in.nareshit.raghu.controller;

import java.util.List;
import java.util.Map;

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

import in.nareshit.raghu.model.Employee;

@Controller
@RequestMapping("/employee")
public class EmployeeController {

        //1. To show form with data

        @GetMapping("/edit")
        public String showEditPage(Model model) {
                Employee e = new Employee();
                e.setEmpId(500);
                e.setEmpName("ABC");
                e.setEmpSal(2500.0);
                e.setEmpGen("Male");
                e.setEmpProj("11");
                e.setEmpAddr("5-99/A, HYD");
                e.setEmpLangs(List.of("ENG","SPN"));

                //DropDown Data (comes from DB)
                Map<Integer,String> myprjs =
Map.of(10,"P1",11,"P2",12,"P3",13,"P4",14,"P5");
                model.addAttribute("myprjs", myprjs);

                model.addAttribute("employee", e);
                return "EmployeeEdit";
        }

        //2. To read form data on submit
        @PostMapping("/update")
        public String doUpdate(
                        @ModelAttribute Employee employee,
                        Model model
                        )
        {

                //model.addAttribute("employee", employee);
                //return "EmployeeEdit";
                model.addAttribute("emp", employee);
                return "EmployeeUpdate";
        }

}
```

4. UI:
--EmployeeEdit.html---------------
```html
<html xmlns:th="https://www.thymeleaf.org/">
<body>
<h2>WELCOME TO EMPLOYEE EDIT FORM</h2>
<form th:action="@{/employee/update}" method="POST"
th:object="${employee}">
<pre>
EMPLOYEE  ID : <input type="text" th:field="*{empId}" readOnly/>
EMPLOYEE NAME : <input type="text" th:field="*{empName}"/>
EMPLOYEE SAL  : <input type="text" th:field="*{empSal}"/>
EMPLOYEE GEN  :
        <input type="radio" th:field="*{empGen}" value="Male" /> Male
        <input type="radio" th:field="*{empGen}" value="Female" />
```

```
Female
EMPLOYEE PROJ :
        <select th:field="*{empProj}">
                       <option value="">-SELECT-</option>
                       <option th:each="ob:${myprjs}"
                                        th:text="${ob.value}"
                                        th:value="${ob.key}"/>

                       <!-- <option value="P1">P1</option>
                       <option value="P2">P2</option>
                       <option value="P3">P3</option> -->
                </select>
EMPLOYEE ADDRESS:
                <textarea th:field="*{empAddr}"></textarea>
EMPLOYEE LANGUAGES:
        <input type="checkbox" th:field="*{empLangs}" value="ENG" /> ENG
        <input type="checkbox" th:field="*{empLangs}" value="FRE" /> FRE
        <input type="checkbox" th:field="*{empLangs}" value="SPN" /> SPA

        <input type="submit" value="Update"/>
</pre>
</form>
</body>
</html>


--------EmployeeUpdate.html------------
<html xmlns:th="https://www.thymeleaf.org/">
<body>
<h3>
Your Form Data is : <span th:text="${emp}"></span>
</h3>
</body>
</html>
=================================================
Redirection in controller:-
 If request is made to one controller#method,
 then after execution of M#1 moving to M#2 execution
 finally M#2 is prodiving output.
    Such process is called redirection.

Ex Flow:
Req--> Path#1 ---> Method#1 ---> Method#2 ---> View/Resp

Syntax: use method return type as "redirect:path";

--Basic Ex--
@Controller
class _____ {
  @GetMapping("/path1")
  public String m1() {
    //logic...
    return "redirect:path2";
  }

  @GetMapping("/path2")
  public String m2() {
     ...
    return "ViewName";
  }

}
-----------Full code---------------------
*)Controller
package in.nareshit.raghu.controller;


import java.util.List;
import java.util.Map;

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

import in.nareshit.raghu.model.Employee;

@Controller
@RequestMapping("/employee")
public class EmployeeController {

        @GetMapping("/delete")
        public String doDelete() {
```

```java
        public String doDelete() {
                System.out.println("DELETE DONE!");
                return "redirect:all";
        }

        @GetMapping("/all")
        public String showAll() {
                System.out.println("FETCHING DATA FROM DB!");
                return "EmployeeData";
        }
}
```

UI:
--EmployeeData.html---
```html
<html xmlns:th="https://www.thymeleaf.org/">
<body>
<h3>THIS IS DATA PAGE!!</h3>
</body>
</html>
```
----
*)Req URL:
http://localhost:8080/employee/delete