Spring MVC

Creating a project using Spring MVC:

1. Create a new Web dynamic project named "`classspring`".
2. In the folder `WEB-INF/lib` add the jar files from Spring.
3. In the folder `WEB-INF` add the files `spring-context.xml` and `web.xml`.
4. Create a package `br.edu.ufabc.classspring.controller` and, inside it, a class named `HelloWorldController` containing the following:

```
@Controller
public class HelloWorldController {
	@RequestMapping("/helloWorldController")
	public String execute() {
		System.out.println("Executing the logic with Spring MVC");
		return "ok";
	}
}
```

5. Don't forget to add the *imports* (make it automatically using ctrl + 1, for it's a good opportunity to verify if the Spring libraries are already being "seen"):

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

6. Create a folder inside `WEB-INF` named `views`.
7. In the folder views, create a file `ok.jsp` with the following:

```
<html>
	<body>
		First class of Spring MVC!
	</body>
</html>
```

8. You can see the result in:

`http://localhost:8080/classspring/helloWorldController.`

Creating a CRUD application for students using Spring MVC

1. Modify the project named "classspring", adding the packages:

   br.edu.ufabc.classspring.dao

   br.edu.ufabc.classspring.jdbc

   br.edu.ufabc.classspring.model

2. Add to each package the files StudentDAO, Student and ConnectionFactory. Download and also add the database progweb and the H2 jar file.

3. Let us start to better organize our view files. Inside the folder views, create a folder students to store the JSP files related to the model Student.

4. Create a JSP file named form.jsp with a form to insert students:

```
<form action="insertStudent" method="post">
 Name: <br /> <input type="text" name="name" /><br />
 Email: <br /> <input type="text" name="email" /><br />
 Address: <br /> <input type="text" name="address" /><br />
 <input type="submit" value="Insert">
</form>
```

5. Create a class StudentController with an insertion method in the package br.edu.ufabc.classspring.controller:

```
@Controller
public class StudentController {
	@RequestMapping("insertStudent")
	public String insert(Student student) {
		StudentDAO dao = new StudentDAO();
		dao.insert(student);
		return "student/added";
	}
}
```

6. In WEB-INF/views/student, create a file added.jsp to confirm and insert a new record:

```
<html>
  <body>
    New student successfully added!
  </body>
</html>
```

7. Using Spring, the direct access to the files in WEB-INF is not allowed and, consequently, we cannot access the form directly. Therefore, in the class StudentController we create a method to call the form:

```
@RequestMapping("newStudent")
public String form() {
	return "student/form";
}
```

8. Access the form in the url: http://localhost:8080/classspring/newStudent

Exercise 1. In the "classspring", implement a list of students that must be updated after insertions, updates, and removals. Each line of the table must have a link to "Change" and a link to "Remove". This page must also have a link to the insertion page for new students that we already made.

INSTRUCTIONS:

- **First Step:** to implement the list of students, you must modify the class `StudentController.java`, adding the following:

```
@RequestMapping("listStudents")
public String list(Model model) {
	StudentDAO dao = new StudentDAO();
	model.addAttribute("students", dao.getList());
	return "student/list";
}
```

Now, create the file `list.jsp` with the following:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="f"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
	pageEncoding="UTF-8"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>List of Students</title>
</head>
<body>
	<table border=1>
		<tr>
			<td><b>Id</b></td>
			<td><b>Name</b></td>
			<td><b>E-mail</b></td>
			<td><b>Address</b></td>
		</tr>
		<c:forEach items="${students}" var="student">
			<tr>
				<td>${student.id}</td>
				<td>${student.name}</td>
				<td>${student.email}</td>
				<td>${student.address}</td>
			</tr>
		</c:forEach>
	</table>
</body>
</html>
```

- **Second Step:** to redirect a request back to the list of students, you must modify the class `StudentController.java`, modifying the methods of each operation as in the following example:

```
@RequestMapping("insert")
public String insert(Student student) {
	StudentDAO dao = new StudentDAO();
	dao.insert(student);
	return "redirect:listStudents";
}
```

- **Third Step:** to create links for "Remove" in the listing, besides of implementing the operation in the class `StudentController.java`, you need the following code:

```
<td><a href="remove?id=${student.id}">Remove</a></td>
```

- **Fourth Step:** to create links for "Change" in the listing, besides implementing the operation in the class `StudentController.java`, you need the following code:

```
<td><a href="showStudent?id=${student.id}">Change</a></td>
```

Also, create an action for the URL `showStudent` in the class `StudentController.java`. This action must get the student data according to the given id, and then put the data in fields in a JSP file so that the user may be able to modify it:

```
@RequestMapping("showStudent")
public String show(Long id, Model model) {
	StudentDAO dao = new StudentDAO();
	model.addAttribute("student", dao.searchById(id));
	return "student/show";
}
```

The page `show.jsp` contains:

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
	pageEncoding="UTF-8"%>
<html>
	<head>
		<title> Change Student</title>
	</head>
	<body>
		<form action="changeStudent" method="POST">
		<input type="hidden" id="id" name="id" value="${student.id }"/><br />
		Name: <input type="text" id="name" name="name"
			value="${student.name }"/><br />
		Email: <input type="text" id="email" name="email"
			value="${student.email }"/><br />
		Address: <input type="text" id="address" name="address"
			value="${student.address }" /><br />
		<input type="submit" value="Change"/>
		</form>
	</body>
</html>
```