

# SPRING FORM

**Tags and Validation**

# Review HTML Forms

HTML form sử dụng các thẻ input để tạo form



**Sign In**

Email Address:

Password:

☐ Remember me

# Spring MVC Form Tags

- Spring MVC Form Tags là được xây dựng thành 1 khối form cho trang web page view.
- Các thẻ tags của spring mvc form là được sử dụng lại

# Spring MVC Form Tags

- Cần khai báo ở file JSP

```
<%@ taglib prefix="form" uri="http://www.springframework.org/tags/form" %>
```

# Data Binding

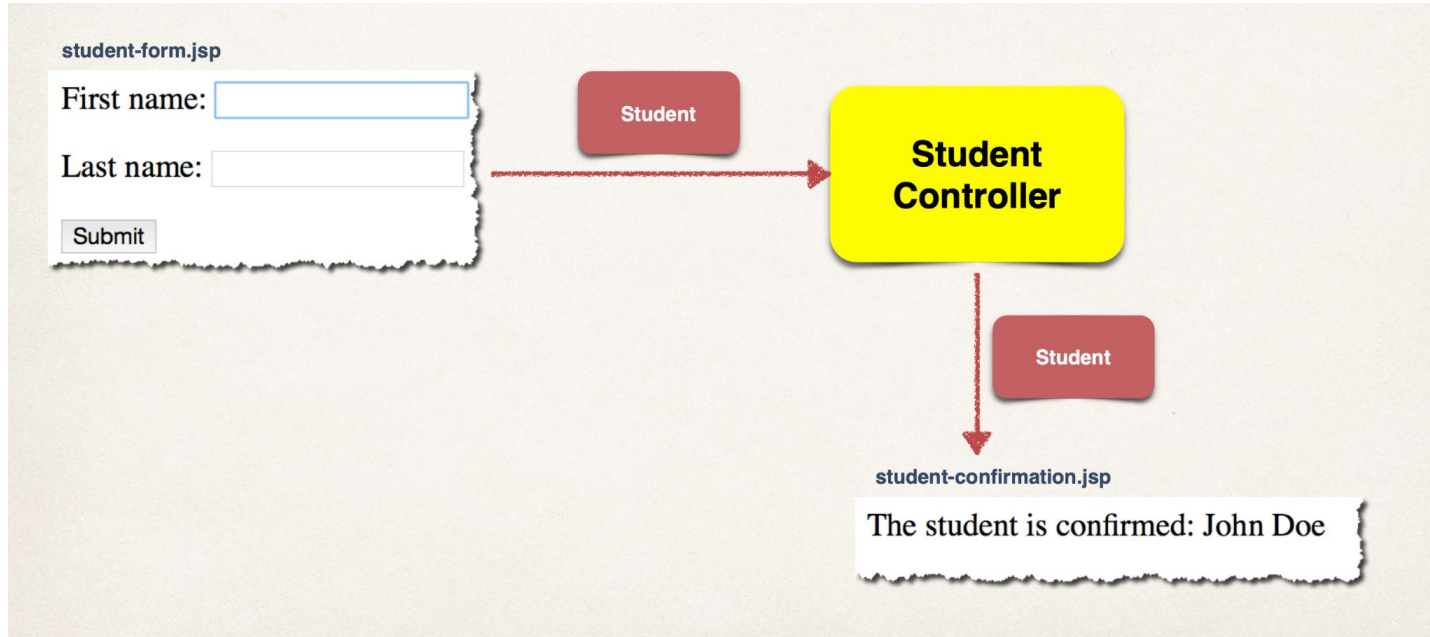
- Spring MVC Form tags có thể được điền dữ liệu theo model
- Tự động lấy/điền dữ liệu từ các model/bean

# Spring MVC Form Tags

- Spring MVC Form Tags sẽ tự sinh mã HTML theo chuẩn HTML Form

Form Tag	Description
form:form	main form container
form:input	text field
form:textarea	multi-line text field
form:checkbox	check box
form:radiobutton	radio buttons
form:select	drop down list
<i>more ....</i>	

# Spring MVC Form Tags - Text Field



# Form nhập

## Yêu cầu ở Controller

- Trước khi show form nhập ra , cần có add 1 model attribute để đưa model ra form
- Và bean này sẽ điền data ra form thông qua data binding

```
@RequestMapping("/showForm")  
public String showForm(Model theModel) {  
  
    theModel.addAttribute("student", new Student());  
  
    return "student-form";  
}
```



# Form in view - Data binding

```
<form:form action="processForm" modelAttribute="student">
```

```
  First name: <form:input path="firstName" />
```

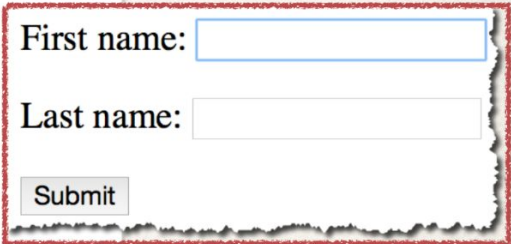
```
  <br><br>
```

```
  Last name: <form:input path="lastName" />
```

```
  <br><br>
```

```
  <input type="submit" value="Submit" />
```

```
</form:form>
```



First name:

Last name:

# Form in view - Data binding - khi form loading

```
<form:form action="processForm" modelAttribute="student">
```

```
    First name: <form:input path="firstName" />
```

```
    <br><br>
```

```
    Last name: <form:input path="lastName" />
```

```
    <br><br>
```

```
    <input type="submit" value="Submit" />
```

```
</form:form>
```

When form is **loaded**,  
Spring MVC will call:

**student.getFirstName()**

...

**student.getLastName**

# Form in view - Data binding - khi form submit

```
<form:form action="processForm" modelAttribute="student">
```

```
    First name: <form:input path="firstName" />
```

```
    <br><br>
```

```
    Last name: <form:input path="lastName" />
```

```
    <br><br>
```

```
    <input type="submit" value="Submit" />
```

```
</form:form>
```

When form is **submitted**,  
Spring MVC will call:

```
student.setFirstName(...)
```

```
...
```

```
student.setLastName(...)
```

# Controller - Data binding - khi form submission

```
@RequestMapping("/processForm")  
public String processForm(@ModelAttribute("student") Student theStudent) {  
  
    // log the input data  
    System.out.println("theStudent: " + theStudent.getFirstName()  
                        + " " + theStudent.getLastName());  
  
    return "student-confirmation";  
}
```

# Page submission thành công

```
@RequestMapping("/processForm")
public String processForm(@ModelAttribute("student") Student theStudent) {

    // log the input data
    System.out.println("theStudent: " + theStudent.getFirstName()
        + " " + theStudent.getLastName());

    return "student-confirmation";
}
```

# Spring MVC Form Tags - Select Tag

<form:select>

First name:

Last name:

Country:

```
<select name="country">  
  <option>Brazil</option>  
  <option>France</option>  
  <option>Germany</option>  
  <option>India</option>  
  ...  
</select>
```



# Spring MVC Form Tags - Select Tag

```
<form:select path="country">
```

```
<form:option value="Brazil" label="Brazil" />
```

```
<form:option value="France" label="France" />
```

```
<form:option value="Germany" label="Germany" />
```

```
<form:option value="India" label="India" />
```

```
</form:select>
```

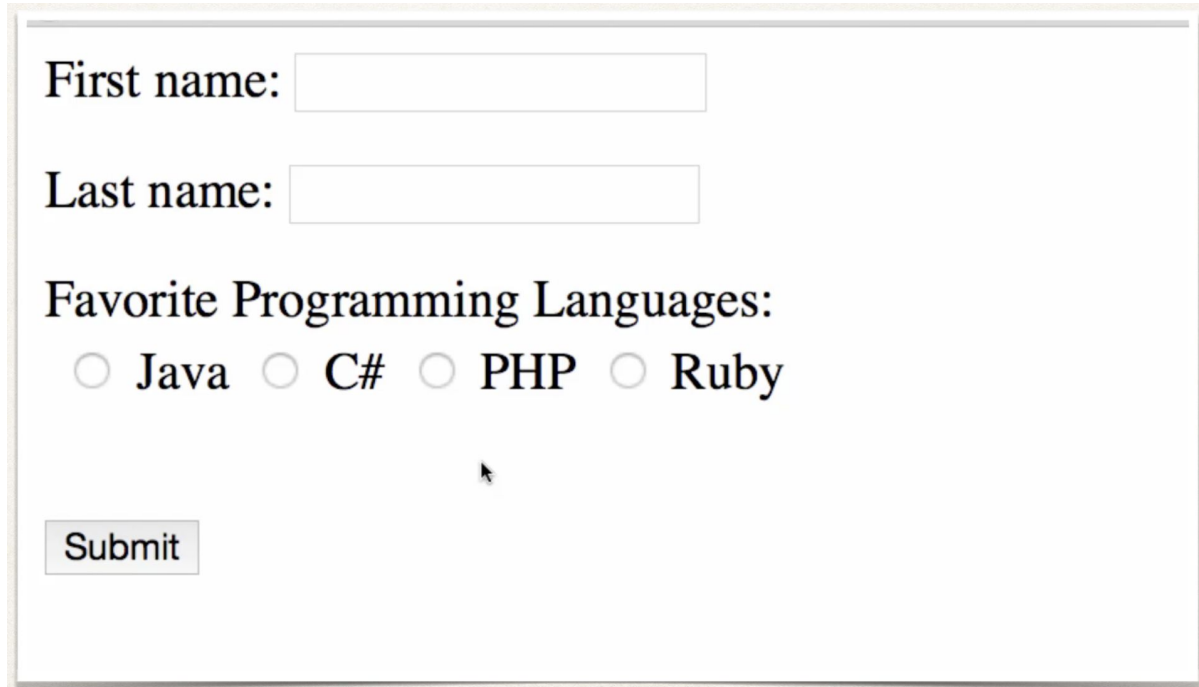
First name:

Last name:

Country:  

# Spring MVC Form Tags - Radio Tag

`<form:radiobutton>`



First name:

Last name:

Favorite Programming Languages:

☐ Java ☐ C# ☐ PHP ☐ Ruby



# Spring MVC Form Tags - Radio Tag

```
Java <form:radio button path="favoriteLanguage" value="Java" />  
C# <form:radio button path="favoriteLanguage" value="C#" />  
PHP <form:radio button path="favoriteLanguage" value="PHP" />  
Ruby <form:radio button path="favoriteLanguage" value="Ruby" />
```

# Spring MVC Form Tags - Checkbox

`<form:checkbox>`

Operating Systems: Linux ☐ Mac OS ☐ MS Windows ☐

Submit

# Spring MVC Form Tags - Checkbox

Linux `<form:checkbox path="operatingSystems" value="Linux" />`

Mac OS `<form:checkbox path="operatingSystems" value="Mac OS" />`

MS Windows `<form:checkbox path="operatingSystems"  
value="MS Windows" />`

# Spring and Validation

Spring version 4 hoặc cao hơn hỗ trợ việc validation thông qua Bean Validation API

Cần add thư viện Validation và để dự án Spring chạy

Các validation của Bean Validation API:

- Required
- Validate length
- Validate numbers
- Validate with regular expressions
- Custom validation

# Spring and Validation

Annotation	Description
<b>@NotNull</b>	Checks that the annotated value is not null
<b>@Min</b>	Must be a number $\geq$ value
<b>@Max</b>	Must be a number $\leq$ value
<b>@Size</b>	Size must match the given size
<b>@Pattern</b>	Must match a regular expression pattern
<b>@Future / @Past</b>	Date must be in future or past of given date
<b><i>others ...</i></b>	

# Spring and Validation

Java's standard Bean Validation API (JSR-303) là API

Nên cần 1 thư viện Implement nó, và Hibernate là 1 trong những bản Validation đơn giản và dễ dùng.

Hibernate-validator

```
<!-- Hibernate Validator -->  
<dependency>  
  <groupId>org.hibernate</groupId>  
  <artifactId>hibernate-validator</artifactId>  
  <version>5.4.1.Final</version>  
</dependency>
```

# Required

*Fill out the form. Asterisk (\*) means required.*

First name:

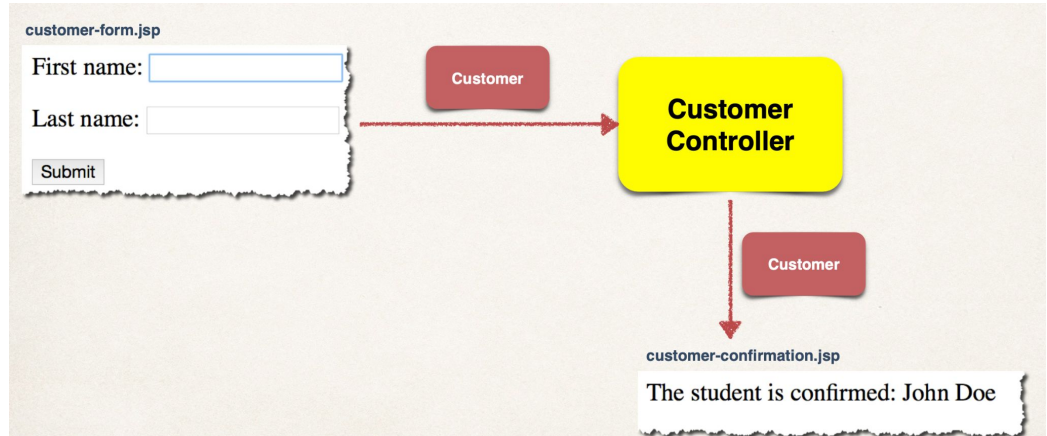
Last name (\*):

→

*Fill out the form. Asterisk (\*) means required.*

First name:

Last name (\*):  is required



# Required

```
import javax.validation.constraints.NotNull;  
import javax.validation.constraints.Size;  
  
public class Customer {  
  
    private String firstName;  
  
    @NotNull(message="is required")  
    @Size(min=1, message="is required")  
    private String lastName;  
  
    // getter/setter methods  
  
}
```



# Required

customer-form.jsp

```
<form:form action="processForm" modelAttribute="customer">
```

```
  First name: <form:input path="firstName" />
```

```
  <br><br>
```

```
  Last name (*): <form:input path="lastName" />  
  <form:errors path="lastName" cssClass="error" />
```

```
  <br><br>
```

```
  <input type="submit" value="Submit" />
```

```
</form:form>
```

First name:

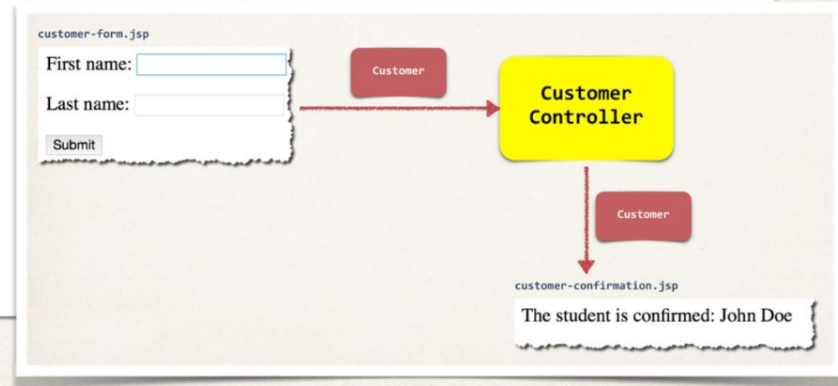
Last name (\*):  is required

Submit

# Required

```
@RequestMapping("/processForm")
public String processForm(
    @Valid @ModelAttribute("customer") Customer theCustomer,
    BindingResult theBindingResult) {

    if (theBindingResult.hasErrors()) {
        return "customer-form";
    }
    else {
        return "customer-confirmation";
    }
}
```



# Number Range: @Min and @Max

```
import javax.validation.constraints.Min;
import javax.validation.constraints.Max;

public class Customer {

    @Min(value=0, message="must be greater than or equal to zero")
    @Max(value=10, message="must be less than or equal to 10")
    private int freePasses;

    // getter/setter methods

}
```

# Regular Expressions

Tham khảo

<https://docs.oracle.com/javase/tutorial/essential/regex/>

```
import javax.validation.constraints.Pattern;

public class Customer {

    @Pattern(regexp="^[a-zA-Z0-9]{5}", message="only 5 chars/digits")
    private String postalCode;

    // getter/setter methods

}
```