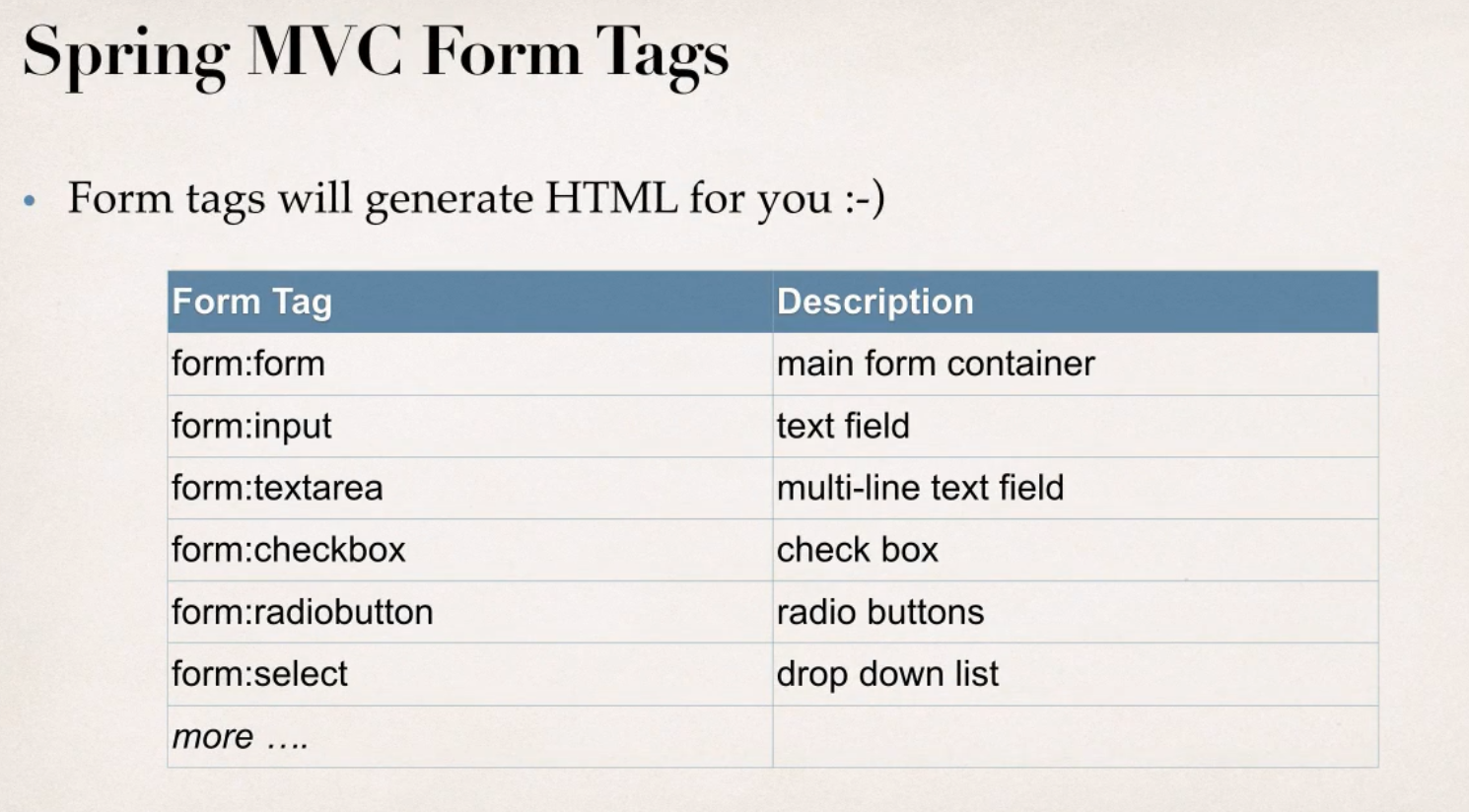
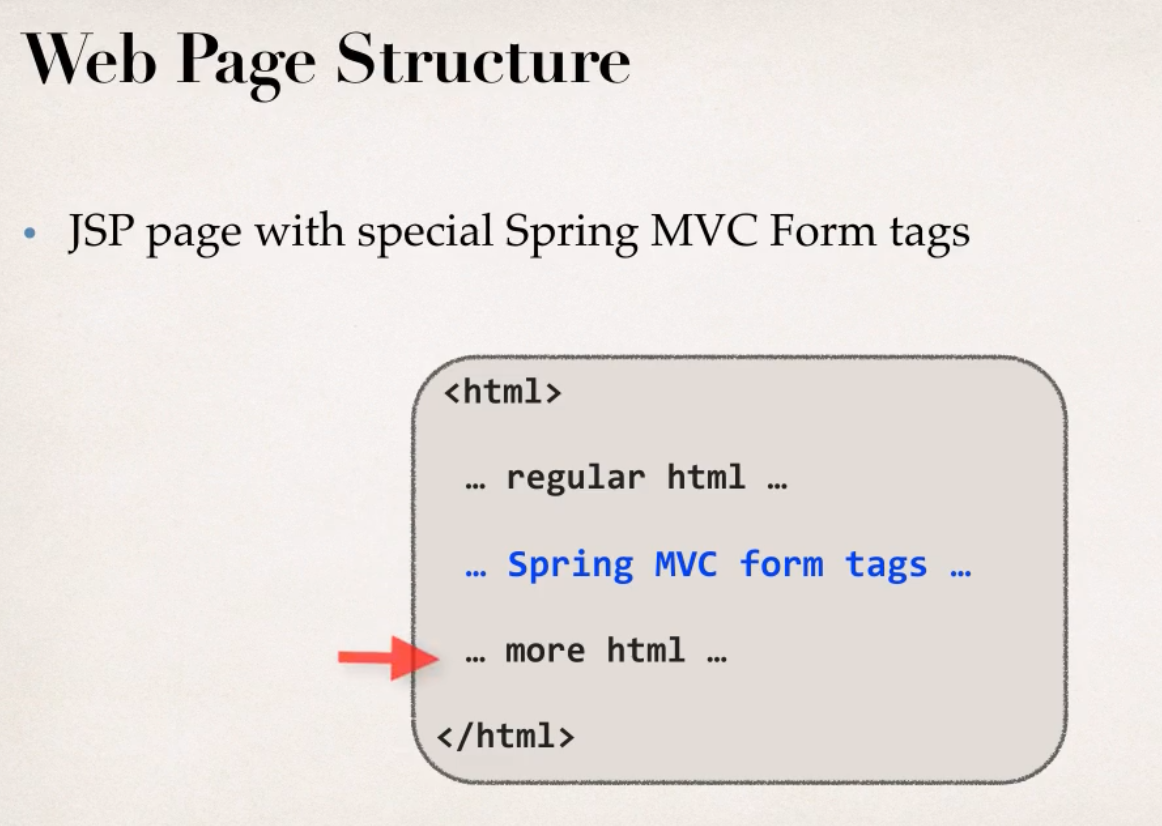
**Form Tags:** Spring has form tags that can make use of data binding. This allows us to automatically set/retrieve data from a java object or bean.

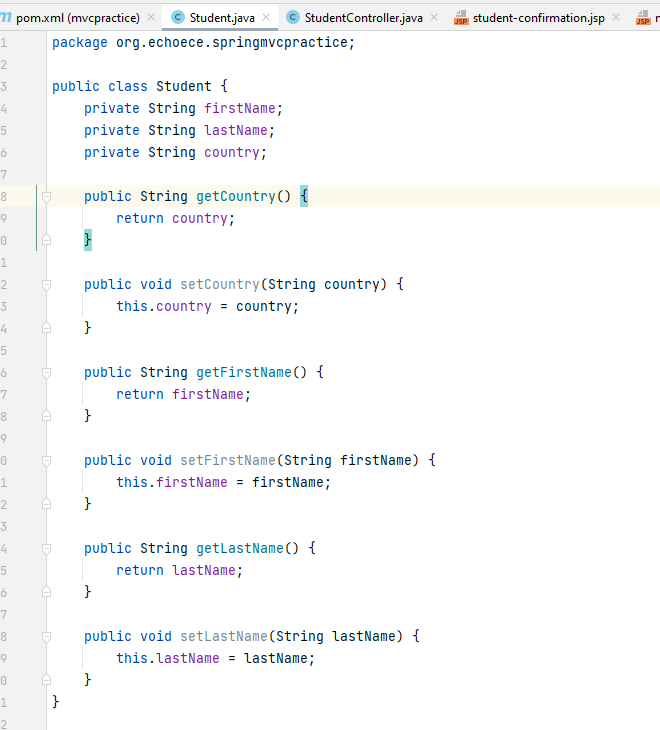
So, we don’t need to manually get the form data from the parameters or query string or whatever, we can just use form tags to bind the form data to a model.







* **Creating Student Class:** This is basically the model, here we create getter and setter methods for the properties. These methods are called when forms are loaded and submitted.



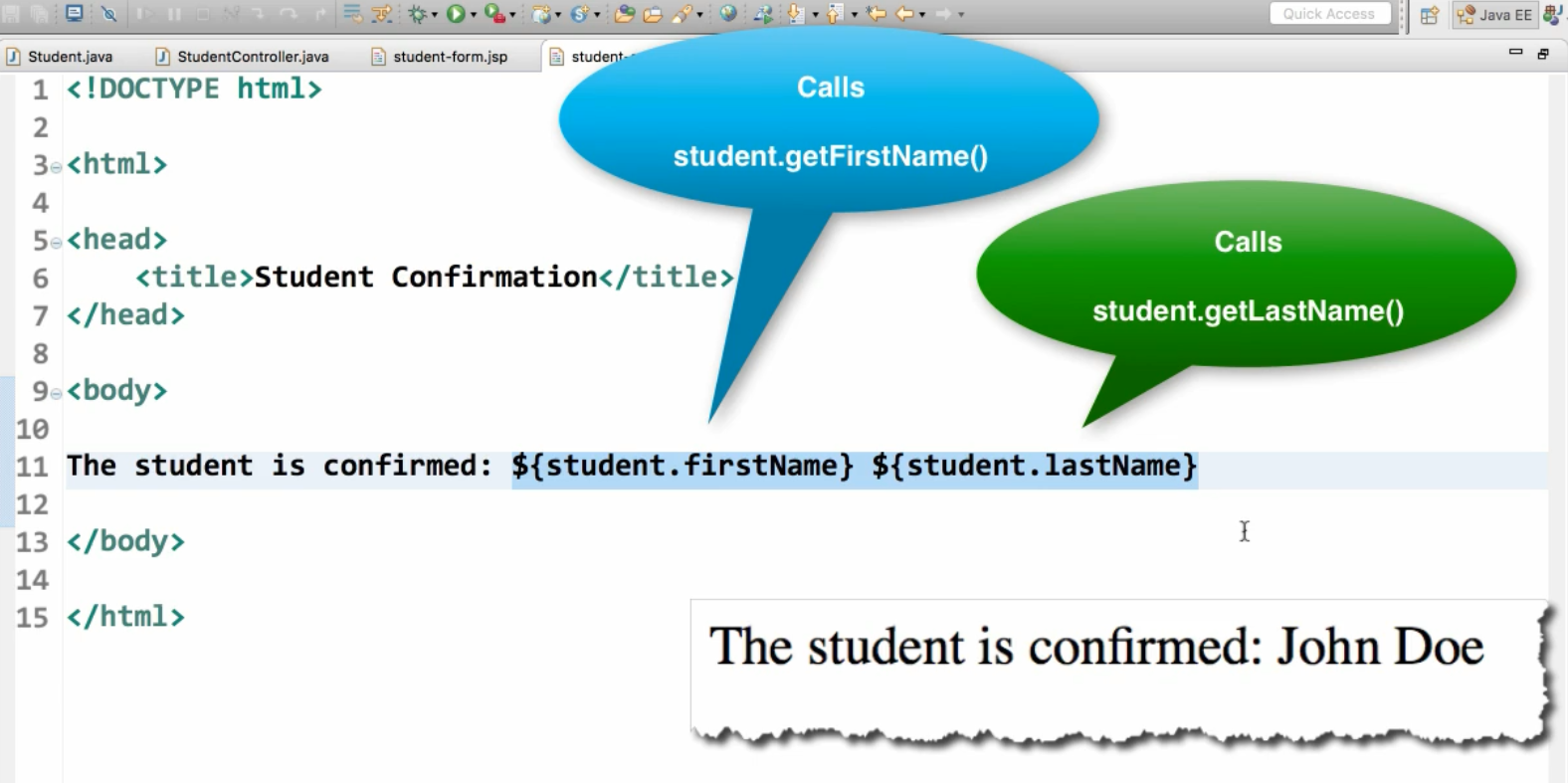
* **Creating Student Controller Class:** This is the controller class, we put annotation @Controller and Controller level request mapping as well. Then we create two methods to process and show form with their respective mapping.



* **Create HTML Form:** Forst we must include the taglib at top. Then when we add the form, we can use the taglibs. Important thing here is,
  + when the form is loaded, it will call the student.getfirstName() and last name methods.
  + And when form is submitted it will call the setter methods with the value we input.



* **Creating Form Processing Code:** Here we add model attribute to the student object same as we did in the form. Spring will then populate all the data in the model object with the form data.
  + **Note:** Here, the **modelAttribute** must be same name as the one we used in the form modalAttribute.
* **Create Confirmation Page:**



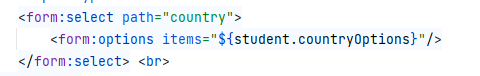
**Drop Down Menu:** We can achieve this in two ways, first is hard coding the options one by one and 2nd is getting the options from our model and spring will populate the options.

1. **Hard Coding:**  The structure is basically like this



1. **Populating from the model:** Here we can populate the data from anyplace (an API, database, file system, hardcoded etc.).

After getting the data, we have to use **form:options** tag (in hardcoded we had option only, no s at last) and it will populate the data.





* The options are hardcoded here, but best to get from DB or API etc.
* Here we are still binding the data to country property of our model, just the options we are getting from different hashmap.

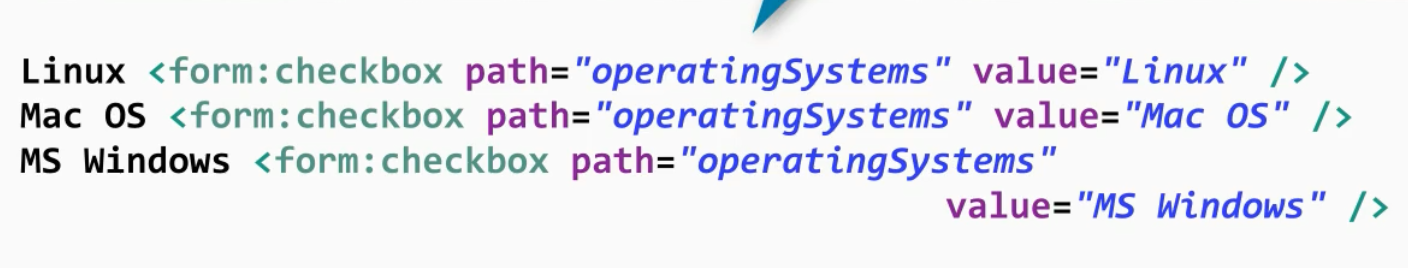
**Radio Button:** This is same as drop down list as well, we can hard code the options or get the options from the model itself.

Hard Coded way:

****

For other way, we can define a similar HashMap in the model, get the data and use tag **form**:**radiobuttons** , then **path** and **items** properties similar to drop down list..

**Check-Box:** Here, we can check multiple items, unlike radio button- or drop-down list. In this case we need a collection data structure to hold the data from checkbox in model.



In this case, to show output we need to add some logic in the jsp page. For this we need the JSTL core tags. We can import them at the top.

**<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>**

Then we can put the logic in,

