Model Validation Exercise

Model Validation

Model Validation is a way of performing validation automatically by using annotations on the Form Object.

Read about model validation here:

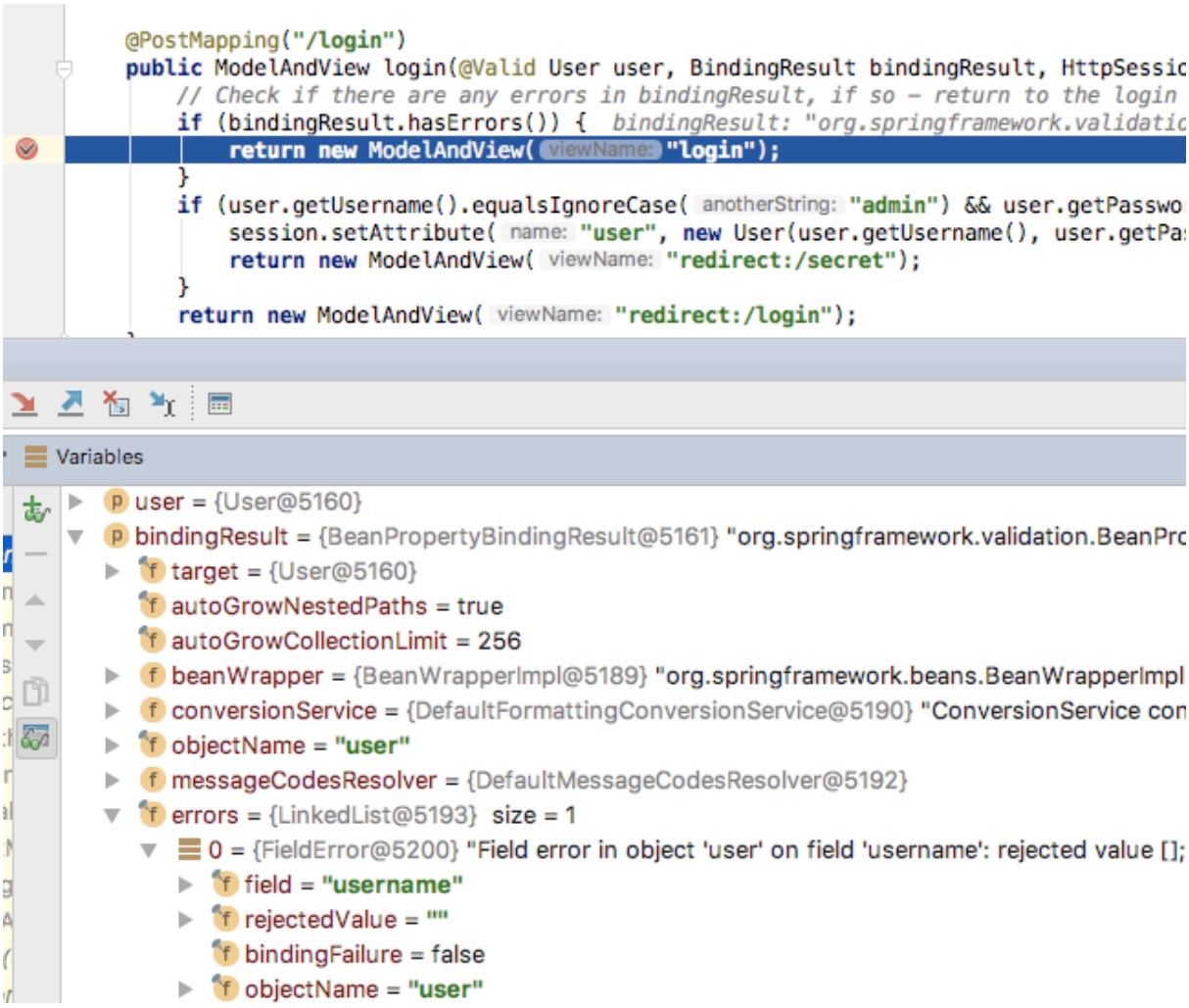
https://spring.io/guides/gs/validating-form-input

We will add a simple form validation to the Login application, only to validate that the username field in the html form must be at least 2 characters and at most 30 characters.

We will use the solution for the Form Object exercise in module 10. You can use your own solution or copy the Solution to the exercise that have been provided for you as a download. Look at the exercise mentioned above in the link, and try to add Model Validation to the application from the Login and cookies exercise by doing this:

- Add the correct annotation on the username variable in the **User** class.
- Change the signature of the **@PostMapping("/login")** method so that you now get a **@Valid User user**, and also a **BindingResult** object.
- Check if there are any errors in the **BindingResult** object, and if so return to the **login** template again.

Now you can try out the code to see if you have any errors. Try to submit an empty username field. You will not see any change since the method already returned to the login form when the user enters an incorrect username, but try instead to debug and set a breakpoint in the **@PostMapping** method in where you check if **BindingResult** has any errors, like this:



Expand the **BindingResult** variable and you will see there is one error on the username field, and that the rejected value is an empty string.

You should let the user see why there is an error by displaying an error message in the login form. Check the example at https://spring.io/guides/gs/validating-form-input again and add an error message in the html form. It is just one line of code. You can put it in a <p> tag instead of a <td> tag as in the example since we are not using tables here.

The following expression will only create the html element if there are errors on the username field:

```
th:if="${#fields.hasErrors('username')}"
```

The following expression will print out a default error message depending on the annotation that was broken:

```
th:errors="*{username}"
```

If you leave this expression out, your own text in the html element will be displayed instead of the default error message. Try it out to delete this expression and be sure to have some custom text inside the and tag.

Now you should get an error message if you try to submit the login form with a username that doesn't follow the rules!

Stretch Tasks - If you have time

1. Try the exercise in the guide mentioned above:

https://spring.io/guides/gs/validating-form-input

2. Try to change the forms for adding a new blog and adding a new comment in the Solution of module 8 – Java Web Jdbc Hacking Exercise.

Add validation to check that the blog title is not empty.