JSR-303

Summary

The validation method was studied through Screen Processing: validation. With JSR-303, a single javax.validation.Validator instance typically validates *all* model objects that declare validation constraints.

In addition, validation can be prformed using the annotation in the model object field using the empty validation method of JSR-303 licensed as a standard specification.

Description

Autovalidation using @Valid

The existing validation method was changed to autovalidation method. The method adds the @Valid annotation to the @ModelAttribute parameter of the controller method. Rather than running validate() method, the validation is run automatically in the binding process.

```
@Controller
public class ExampleController {

	@Autowired ExampleValidator validator;

	@InitBinder
	public void initBinder(WebDataBinder dataBinder){
		dataBinder.setValidator(this.validator);
	}


	@RequestMapping("/insertMember.do")
	public String insertMember(@ModelAttribute("memberVO") @Valid MemberVO memberVO,
BindingResult bindingResult, ..) {
		//..
	}
}
```

JSR-303 bean validation function

Above method is the method that changed existing validation method to the autovalidation method. The validation method is the method that configures limitations to the bean.

First of all, dependency library should be added to class pad. If the Maven is in use, add the following dependency library to the project.

```
<dependency>
	<groupId>javax.validation</groupId>
	<artifactId>validation-api</artifactId>
	<version>1.0.0.GA</version>
</dependency>
<dependency>
	<groupId>org.hibernate</groupId>
	<artifactId>hibernate-validator</artifactId>
	<version>4.0.0.GA</version>
</dependency>
```

Following is the example of model object applied with JSR-303 restriction condition annotation.

```
public class MemberVO{

	@NotNull
```

```
	@Size(min = 1, max = 50, message="Enter the name.")
	private String name;

	@Pattern(regexp=".+@.+\\.[a-z]+", message= "E-mail format is wrong.")
	private String email;

	//..
}
```

Since @NotNull cannot validate empty character string, empty character string should be checked using @Size(min=1).

To perform validation using restriction conditino annotation as shown above, it is required to register LocalValidatiorFactoryBean as a bean. LocalValidatiorFactoryBean is a type of adapter that enables to use validation function of JSR-303 as Validator of spring. If you register LocalValidatiorFactoryBean as a bean, receive DI as validator type in controller, set in WebDataBinder at @InitBinder or directly use at the code like Validator.

```
<bean id="validator" class="org.springframework.validation.beanvalidation.LocalValidatorFactoryBean"
/>
```

This is the controller example that uses the validator as the bean validation function.

```
@Controller
public class ExampleController {

	@Resource
	Validator validator;

	@InitBinder
	public void initBinder(WebDataBinder dataBinder){
		dataBinder.setValidator(this.validator);
	}

	@RequestMapping("/insertMember.do")
	public String insertMember(@ModelAttribute("memberVO") @Valid MemberVO memberVO,
BindingResult bindingResult, ..) {
		//..
	}
}
```

Reference

- [Validator Example]