

Spring Boot Validation Cheat Sheet

What is Spring Boot Validation?

Spring Boot Validation is a sophisticated validation framework for Spring Boot applications based on JSR-303 and JSR-380.

Dependencies

Mandatory Spring Boot Validation Starter

```
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-validation</artifactId>
<version>2.5.1</version>
</dependency>
```

Optional URI Validator

```
<dependency>
<groupId>de.schegge</groupId>
<artifactId>uri-validator</artifactId>
<version>0.2.2</version>
</dependency>
```

Usage

Spring MVC Controller

The parameter validation of controller endpoint methods is integrated. All accordingly annotated parameters are validated.

```
@RestController
public class ExampleController {

    @PostMapping("/validate/{id}")
    Input validate(
        @PathVariable @Positive Long id,
        @RequestParam("query") @NotEmpty String query,
        @Valid @RequestBody Input input) {
        return input;
    }
}
```

Injected Validator

A Validator instance can be injected in each service.

```
@Service
public class UserAccountService {
    @Autowired
    private final Validator validator;

    public boolean checkAccount(UserAccount useraccount) {
        Set<ConstraintViolation<UserAccount>> violations =
            validator.validate(useraccount);
        if (!violations.isEmpty()) {
            throw new ConstraintViolationException(violations);
        }
        return true;
    }
}
```

Annotated POJO

Attributes in POJOs can be marked with validation annotations. Validation groups can be used to add validations for different scenarios

```
class Input {
    @Min(1) @Max(10)
    private int numberBetweenOneAndTen;

    @Email(regexp = "^.+@.+$", groups = Create.class)
    private String email;
}
```

Annotations

The validation of parameters and class attributes can be defined using annotations. The entries in the *type* column mean:

any any type
text CharSequence, String, StringBuilder, ...
date all standard Java time and date classes
uri URI, URL or text
collection Collection, Map or Array

JSR-303, JSR-380

annotation	validates	type
@NotNull @NotEmpty	the property is not null. the property is not null or empty	any collection, text
@NotBlank @AssertTrue @AssertFalse	the property is not null or whitespace. the property is true. the property is false.	text boolean boolean
@Size	the property has a size between the attributes min and max	collection, text
@Min	the property is no smaller than the value attribute.	number
@Max	the property is no larger than the value attribute.	number
@Positive @PositiveOrZero @Negative @NegativeOrZero	the property value is strictly positive, respectively positive including zero. the property is strictly negative, respectively negative including zero.	number
@Past @PastOrPresent @Future @FutureOrPresent	the property is in the past, respectively including the present. the property is in the future, respectively including the present.	date
@Pattern @Email @Valid	the property matches the attribute regex. the property is a valid email address. Marks a property, method parameter or method return type for validation cascading.	text text any

Spring

annotation	validates	type
@Validated	Variant of JSR-303's @Valid annotation, supporting validation groups.	any

Hibernate

annotation	validates	type
@EAN @ISBN @Length	the property is an EAN13 code. the property is an ISBN code the property has a length between the attributes min and max	text text text
@Range	the property has a numeric value between the attributes min and max	number, text
@URL	the property is a URL.	text

URI Validator

annotation	validates	type
@Localhost @NotLocalhost	the host is localhost or not.	uri
@Host	the host matches the attribute regex.	uri
@Http @Https @Schema	the schema is HTTP or HTTPS. the schema matches the attribute regex.	uri uri

Constraint composition

Validators can be easily implemented. You can also compose your own constraints from existing constraints. @HttpNotLocalhost is composed of @Http and @NotLocalhost.

```
@Retention(RUNTIME)
@Target({ METHOD, FIELD, ANNOTATION_TYPE, CONSTRUCTOR, PARAMETER,
        TYPE_USE })
@Constraint(validatedBy = {})
@Http @NotLocalhost
public @interface HttpNotLocalhost {
    String message() default "{cheatsheet.message}";
    Class<?>[] groups() default {};
    Class<? extends Payload>[] payload() default {};
}
```

