Spring ile Validasyon İşlemleri

Spring ve Metot Düzeyinde Validasyon

- Controller/Servis metotlarının **input parametreleri** ve **return değeri** Spring vasıtası ile dekleratif biçimde validate edilebilir

```
@Validated
@Service
public class UserService {
    public @NotNull Long createUser(@Valid User user) {
    ...
    }
}
```

Validate edilecek metotları içeren servis bean'ı @Validated annotasyonu ile işaretlenmelidir

Bunun yanında JSR-303 validasyon constraint'leri doğrundan input paremetre ve return değerinde de kullanılabilir

Validasyona tabi tutulacak input parametre veya return tipi içerisinde JSR 303 constraint'leri barındıran bir domain sınıf ise @Valid anotasyonu ile şaretlenir

Spring ve Metot Düzeyinde Validasyon

```
public class User {
    @NotEmpty
    private String username;
    @Email
    private String email;
    @Min(18) @Max(64)
    private int age;
}
```

Buradaki @NotEmpty, @Email, @Min, @Max anotasyonları JSR-303 constraint'leridir Spring'den bağımsızdır

JSR Bean Validation API ve Spring

- Spring validasyon işlemini JSR-303/JSR-349 **Bean Validation API' üzerine** bina etmiştir
- Servis metotlarına ve domain model sınıflarına **validasyon annotasyonları** yerleştirilir
- Runtime'da da otomatik tespit edilip yüklenen **JSR Validator bean**'i ile bu constraint'ler denetlenir

Servis Metot Düzeyinde Validasyon Konfigürasyonu

JAVA Eğitimleri

@**Validated** anotasyonuna sahip bean'ların metot parametrelerini ve return değerlerini validate eden proxy bean'lar üretir

```
<bean
class="org.springframework.validation.beanvalidation.
MethodValidationPostProcessor">
      <property name="validator" ref="validator"/>
</bean>
```

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

classpath'deki JSR-303 bean validation provider'ı devreye sokar

JSR-303 Validator API'nin Doğrudan Kullanılması

JAVA Eğitimleri

```
public class FooService {
    private javax.validation.Validator validator;

     public void setValidator(Validator validator) {
        this.validator=validator
    }

    public void doWork() {
        validator.validate(...);
    }
}
```

JSR-303 implementasyonu Spring managed bean olarak uygulama içindeki diğer bean'lere **Validator nesnesi** olarak enjekte edilerek doğrudan da kullanılabilir

```
<bean id="fooService" class="x.y.FooService">
    <property name="validator" ref="validator"/>
</bean>

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

JSR-303 Custom Validation Constraints ve Spring Ent.

- Spring JSR-303 API ile **custom validation constraint** yazımı için de kolaylık sağlar
- Örneğin @**CheckIfApproved** gibi bir constraint yazılabilir
- Validation constraint anotasyonunu process eden **ConstraintValidator** sınıfına Spring dependency injection yapabilir
- Böylece ConstraintValidator sınıfları **Spring ekosistemindeki imkanlar**dan faydalanabilir

JSR-303 Custom Validation Constraints ve Spring Ent.

```
public class VisaRequest {

    @CheckIfApproved
    private Document doc;

    //...
}
```

Custom validation constraint'dir

Custom validation constraint'i process eden **ConstraintValidator** sınıfıdır

```
@Target({ElementType.METHOD, ElementType.FIELD})
@Retention(RetentionPolicy.RUNTIME)
@Constraint(validatedBy=ApprovedConstraintValidator.class)
public @interface CheckIfApproved {

}
```

JSR-303 Custom Validation Constraints ve Spring Ent.

```
import javax.validation.ConstraintValidator;

public class ApprovedConstraintValidator
                                    implements ConstraintValidator {

    @Autowired
    private ApprovalCheckingService approvalCheckingService;

    //...
}
```

JSR Validator bean'ini tanımlamak için **LocalValidatorFactoryBean** kullanıldığı takdirde **ConstraintValidator** nesnelerine dependency injection yapmak mümkündür

Validasyon Grupları

- **@Validated** anotasyonu **metot düzeyinde** de kullanılabilir
- Bu durumda validasyon sürecinde **validasyon gruplarını** belirtmek için kullanılır

```
public interface OnCreate {
}
```

```
public interface OnUpdate {
}
```

```
@Entity
public class User {
    @Id
    @GeneratedValue
    @Null(groups=OnCreate.class)
    @NotNull(groups=OnUpdate.class)
    private Long id;
}
```

Validasyon Grupları

```
@Validated
@Service
public class UserService {

    @Validated(OnCreate.class)
    public @NotNull Long createUser(@Valid User user) {
        //...
    }

    @Validated(OnUpdate.class)
    public void updateUser(@Valid User user) {
        //...
    }
}
```

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com