

VALIDACIÓN

- ▶ Spring utiliza el estándar definido en la JSR-303
- ▶ No solamente sirve para MVC
 - Gestión de beans
 - Spring Data
 - ...

VALIDACIÓN

- ▶ Nos permite usar anotaciones
- ▶ Nos permite extender el modelo creando validadores propios.
- ▶ Alta flexibilidad

DEPENDENCIAS MAVEN

```
<!-- https://mvnrepository.com/artifact/javax.validation/validation-api -->
<dependency>
  <groupId>javax.validation</groupId>
  <artifactId>validation-api</artifactId>
  <version>1.1.0.Final</version>
</dependency>
<!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-validator -->
<dependency>
  <groupId>org.hibernate</groupId>
  <artifactId>hibernate-validator</artifactId>
  <version>5.3.1.Final</version>
</dependency>
```

ANOTACIONES PERMITIDAS

- ▶ @NotNull: no nulo
- ▶ @Pattern: expresión regular
- ▶ @Past: fecha en el pasado
- ▶ @Min: valor mínimo
- ▶ @Max: valor máximo
- ▶ @NotBlank: la cadena tiene más de 0 caracteres
- ▶ @Email: email válido

CUSTOMIZANDO MENSAJES

Fichero de *properties* (key=value)

`Validador.Objeto.Atributo=Mensaje`

```
NotEmpty.empleadoForm.nombre=Por favor introduzca el ...  
Size.empleadoForm.nombre=El nombre de un empleado debe...  
NotNull.empleadoForm.id=Por favor, introduzca el ID del ...  
NotEmpty.empleadoForm.email=Por favor, introduzca el ...  
Email.empleadoForm.email=El email es incorrecto
```

CUSTOMIZANDO MENSAJES

Bean para uso de *properties*

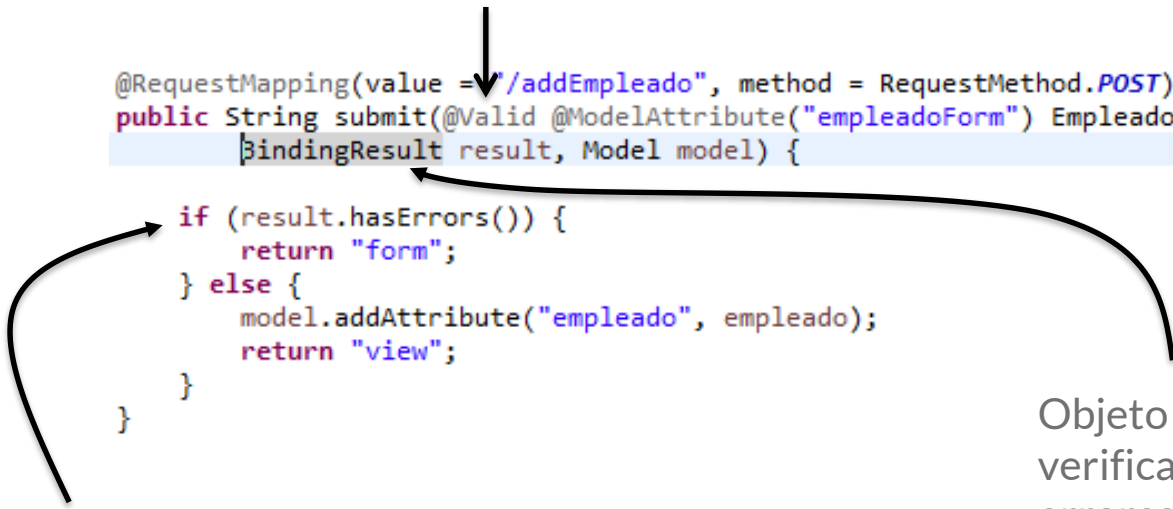
```
<!-- Este bean nos permitirá usar properties -->
<bean id="messageSource"
      class="org.springframework.context.support.ReloadableResourceBundleMessageSource">

    <property name="basename" value="/WEB-INF/messages" />

</bean>
```

CAMBIOS EN EL CONTROLADOR

Forzamos que el objeto recibido sea válido



```
@RequestMapping(value = "/addEmpleado", method = RequestMethod.POST)
public String submit(@Valid @ModelAttribute("empleadoForm") Empleado empleado,
    BindingResult result, Model model) {
```

```
    if (result.hasErrors()) {
        return "form";
    } else {
        model.addAttribute("empleado", empleado);
        return "view";
    }
}
```

En caso de haberlos,
volvemos al formulario

Objeto que nos permite
verificar si ha habido
errores de validación

CAMBIOS EN EL FORMULARIO

```
<tr>
  <td><form:label path="nombre">Nombre</form:label></td>
  <td><form:input path="nombre"/></td>
  <td><form:errors path="nombre" cssClass="error"></form:errors></td>
</tr>
```

De esta forma
podemos mostrar el
mensaje de error

Podemos aplicarle
una clase CSS