FORMULARIOS CON VALIDACIÓN Y MENSAJES DE ERROR

VALIDACIÓN CON SPRING

- Spring permite usar el estándar *JSR-303/JSR-380 Bean Validation API*.
- Spring Boot configura por defecto la implementación de este estándar realizada por *hibernate*.
- Permite realizar la validación añadiendo anotaciones en nuestras clases modelo.

ANOTACIONES DE VALIDACIÓN

- @NotNull: el atributo no puede ser nulo
- @Min, @Max: mayor o igual (o menor o igual) que un valor determinado.
- @NotEmpty: el atributo no puede estar vacío (Strings, colecciones, arrays, …)
- @Email: el atributo debe ser un email válido.
- @Size: el atributo tiene que tener un tamaño según el indicado.

https://beanvalidation.org/2.0/spec/#builtinconstraints

FORMULARIO CON VALIDACIÓN

- Necesitamos modificar nuestro controlador *@PostMapping*.

El objeto recibido debe ser válido

Este objeto nos permite saber si ha habido errores y cuales.

```
@PostMapping("/nuevo/submit")
public String submitNuevoProducto(@Valid Producto producto, BindingResult bindingResult, Model model) {

    if (bindingResult.hasErrors()) {
        model.addAttribute("categorias", categoriaService.findAll());
        return "admin/form-producto";
    } else {
        productoService.save(producto);
        return "redirect:/admin/producto/";

    }

}
```

En caso de que haya habido errores, volvemos al formulario para gestionarlos allí.

VISUALIZACIÓN DE ERRORES

- Thymeleaf nos ofrece algunos elementos para gestionar los errores
  - Algunas funciones del objeto #fields
    - #fields.hasError('...')
    - #fields.errors('...')
  - Atributos
    - th:errors
    - th:errorclass

VISUALIZACIÓN DE ERRORES

- th:errors
  - Nos permite indicar los errores de un atributo del *command object*

- th:errorclass
  - Nos permite modificar la clase de error de un campo del formulario

VISUALIZACIÓN DE ERRORES

- #fields.hasErrors('...')
  - Nos permite saber si una determinada tiene o no errores.

- #fields.errors('....')
  - Su uso es similar a th:errors
  - Permite utilizar los errores fuera del formulario.

NUESTRA VISUALIZACIÓN DE ERRORES

- Si hay error, aplicar a la capa entera la clase de bootstrap *has-error*.
- Visualizar, debajo el campo correspondiente, el mensaje de error

NUESTRA VISUALIZACIÓN DE ERRORES


```
<div class="form-group"
      th:classappend="${#fields.hasErrors('nombre')} ? 'has-error'">
     <label class="control-label" for="nombre">Nombre</label>
     <input type="text"  class="form-control"
            id="nombre" placeholder="Nombre" th:field="*{nombre}" />
     <span  th:if="${#fields.hasErrors('nombre')}" th:errors="*{nombre}"
                        class="help-block" id="nombre-error">Errores</span>
</div>
```