Implementación de la subida de ficheros

Desarrollo de un API REST con Spring Boot

Código base

- Hemos incluido todos los elementos descritos en la lección anterior.

Método de subida

- Nuestro método nuevoProducto con algunas modificaciones
  - Aunque no es obligatorio, añadimos el tipo mime de los datos que consume *MediaType.MULTIPART_FORM_DATA_VALUE*
  - Ya no podemos obtener nuestros datos a partir de *@RequestBody*, ya que el cuerpo de la petición tiene varias partes.
  - Necesitamos utilizar otras anotaciones, como *@RequestParam* o ***@RequestPart***.

@RequestParam vs @RequestPart

- *@RequestParam*: asocia un parámetro de la petición a un argumento de un método de un controlador.
  - Puede usarse en peticiones multiparte.
  - Válido para anotar *MultipartFile*
  - Inconveniente: si no es de tipo String o MultipartFile, necesita un Converter registrado.
  - Nosotros transformamos JSON - Objeto Java vía un HttpMessageConverter (no es lo mismo).

@RequestParam vs @RequestPart

- *@RequestPart*: asocia una parte de una petición multiparte a un argumento de un método de un controlador.
  - Análogo a *@RequestBody* (que se usa en peticiones no-multiparte).
  - Puede usarse con MultipartFile o cualquier otro tipo de dato que tenga asociado un HttpMessageConverter
- Es probable que @RequestParam se use con campos de formulario clave-valor, mientras que @RequestPart se use con partes que contengan contenido más complejo: JSON, XML.

Firma del método

- La firma del método quedará entonces así:

```
@PostMapping(value = "/producto",
                consumes= MediaType.MULTIPART_FORM_DATA_VALUE)
public ResponseEntity<?> nuevoProducto(
     @RequestPart("nuevo") CreateProductoDTO nuevo,
     @RequestPart("file") MultipartFile file)
```

- La petición que realicemos deberá incluir dos partes
  - Una, llamada *nuevo*, de tipo mime *application/json*
  - Otra, llamada *file*, de tipo *application/octet-stream*

Subida de la imagen

- Subimos la imagen al almacenamiento a través de nuestro servicio.
- Obtenemos el nombre del fichero.
- A partir del nombre del fichero, obtenemos la URI del mismo.
- Asignamos la URI al producto
- *Nota*: también modificamos el DTO para que se incluya la imagen en el listado de productos.

En la próxima lección vamos a comprobar cómo podemos utilizar este servicio.

Reto

- Te propongo que trates de implementar la subida múltiple de ficheros, es decir, que una petición multiparte pueda incluir más de uno.
- Te dejo las pistas para las modificaciones a incluir

Reto

- Necesitas recibir como argumento más de un MultipartFile, aunque todos deben llamarse igual (es decir, igual @RequestPart).
- A la hora de procesar la subida, necesitarás un bucle, para repetir el procesamiento que hemos hecho en esta lección.
- Necesitas incluir cambios en el modelo de Producto, para que soporte más de una imagen. ¿Qué tal con @ElementCollection?
- La petición GET de todos los productos sólo debería incluir la primera imagen, mientras que la petición GET de un producto, sí que debería devolver todas las imágenes.

Reto (2)

- Otro posible reto sería separar la subida de imágenes de la subida de un producto.
- Esto nos permitiría subir imágenes de forma independiente.
- Vendría bien por ejemplo para un formulario con drag&drop que permita subir varios ficheros.
- Para ello, necesitas implementar un nuevo método en el controlador.
- Sería bueno crear también un DTO para responder la petición.