


```
FileService() {&#xA; }&#xA; &#xA; public Path save(String fileName, InputStream is) throws
IOException {&#xA; byte[] buffer = new byte[1 &amp;lt;&amp;lt; 20];&#xA; int bytesRead =
-1;&#xA; int bytesWritten = 0;&#xA; Path out = Path.of(rootPath, fileName);&#xA; try
(OutputStream os = Files.newOutputStream(out)) {&#xA; while((bytesRead = is.read(buffer))
&amp;gt; 0) {&#xA; os.write(buffer, 0, bytesRead);&#xA; bytesWritten += bytesRead;&#xA;
}&#xA; }&#xA; logger.info(&#34;file &#34; + fileName + &#34; written to path &#34; + out + &#34;,
bytes written &#34; + bytesWritten);&#xA; return out;&#xA; }&#xA; &#xA;}Der File Service speichert
die uploaded Datei relativ zum Pfad definiert durch das Property &#34;upload.service.path&#34;.
Solches definieren wir in der application.properteies Datei:upload.service.path=/tmp
```

Upload Controller Integration Test

Das folgende Listing zeigt den Integration Test für den UploadController upload REST Endpoint:

```
package ch.std.demo.upload.rest.test;&#xA;&#xA;import static
org.junit.jupiter.api.Assertions.assertEquals;&#xA;&#xA;import org.junit.jupiter.api.Test;&#xA;import
org.junit.jupiter.api.extension.ExtendWith;&#xA;import
org.springframework.beans.factory.annotation.Autowired;&#xA;import
org.springframework.boot.test.context.SpringBootTest;&#xA;import
org.springframework.boot.test.context.SpringBootTest.WebEnvironment;&#xA;import
org.springframework.boot.test.web.client.TestRestTemplate;&#xA;import
org.springframework.http.HttpEntity;&#xA;import org.springframework.http.HttpHeaders;&#xA;import
org.springframework.http.HttpStatus;&#xA;import org.springframework.http.MediaType;&#xA;import
org.springframework.http.ResponseEntity;&#xA;import
org.springframework.test.context.junit.jupiter.SpringExtension;&#xA;import
org.springframework.util.LinkedMultiValueMap;&#xA;&#xA;import
ch.std.demo.upload.dto.RequestUploadDTO;&#xA;import
ch.std.demo.upload.dto.ResponseUploadDTO;&#xA;&#xA;@ExtendWith(SpringExtension.class)&#
xA;@SpringBootTest(webEnvironment = WebEnvironment.RANDOM_PORT)&#xA;public class
UploadServiceIntegrationTests {&#xA;&#xA; @Autowired&#xA; private TestRestTemplate
restTemplate;&#xA;&#xA; @Test&#xA; public void contextLoads() {&#xA; }&#xA;&#xA; @Test&#xA;
public void testUploadController() throws Exception {&#xA; var multipart = new
LinkedMultiValueMap&amp;lt;&amp;gt;();&#xA; multipart.add(&#34;file&#34;, new
org.springframework.core.io.ClassPathResource(&#34;dummy.pdf&#34;));&#xA;&#xA; final
ResponseEntity&amp;lt;ResponseUploadDTO&amp;gt; post =
this.restTemplate.postForEntity(&#34;/rest/upload&#34;,&#xA; new
HttpEntity&amp;lt;&amp;gt;(multipart, headers()), ResponseUploadDTO.class);&#xA; &#xA;
assertEquals(HttpStatus.OK, post.getStatusCode());&#xA; System.out.println(&#34;response =
&#34; + post.getBody());&#xA; }&#xA;&#xA; private HttpHeaders headers() {&#xA; HttpHeaders
headers = new HttpHeaders();&#xA;
headers.setContentType(MediaType.MULTIPART_FORM_DATA);&#xA; return headers;&#xA;
}&#xA;}Der Integration Test verwendet die Datei dummy.pdf, welche im Pfad
&#34;src/test/resources&#34; abgelegt ist. Die Datei dummy.pdf können Sie via Hyperlink
downloaden und in das Projekt integrieren. Der Integration Unit Test sollte korrekt funktionieren.
```

Upload Controller mit DTO

Das folgende Listing zeigt einen REST Endpoint mit zusätzlichem Request DTO Objekt:

```
@PostMapping(path = &#34;uploaddto&#34;, produces = {
MediaType.APPLICATION_JSON_VALUE })&#xA;public ResponseUploadDTO
uploadDokumentWithDTO(@RequestPart(&#34;dto&#34;) RequestUploadDTO dto,
@RequestPart(&#34;file&#34;) MultipartFile file) {&#xA; String fileName = dto.getFileName();&#xA;
ResponseUploadDTO responseUploadDTO = new ResponseUploadDTO();&#xA; try {&#xA; Path
out = this.fileService.save(fileName, file.getInputStream());&#xA;
responseUploadDTO.setSuccess(&#34;file uploaded&#34;);&#xA;
responseUploadDTO.setPath(out);&#xA; } catch (Exception e) {&#xA;
responseUploadDTO.setFailed(e.getMessage());&#xA; }&#xA; return
responseUploadDTO;&#xA;}Den zugehörigen Integration Test sehen Sie im folgenden
Listing:&#xA;@Test&#xA;public void testUploadDTOController() throws Exception {&#xA; var multipart =
new LinkedMultiValueMap&amp;lt;&amp;gt;();&#xA; RequestUploadDTO requestUploadDTO = new
RequestUploadDTO();&#xA; requestUploadDTO.setFileName(&#34;dummydto.pdf&#34;);&#xA;
multipart.add(&#34;dto&#34;, requestUploadDTO);&#xA; multipart.add(&#34;file&#34;, new
```

```
org.springframework.core.io.ClassPathResource("#34;dummy.pdf#34;));&#xA;&#xA; final
ResponseEntity<ResponseUploadDTO> post =
this.restTemplate.postForEntity("#34;/rest/uploaddto#34;,&#xA; new
HttpEntity<&#xA;>(multipart, headers(), ResponseUploadDTO.class);&#xA; &#xA;
assertEquals(HttpStatus.OK, post.getStatusCode());&#xA; System.out.println("#34;response =
#34; + post.getBody());&#xA;}&#xA;Die RequestDTO Klasse finden Sie hier:package
ch.std.rest.upload.dto;&#xA;&#xA;public class RequestUploadDTO {&#xA; &#xA; private String
fileName;&#xA; &#xA; public RequestUploadDTO() {&#xA; }&#xA; &#xA; public String
getFileName() {&#xA; return fileName;&#xA; }&#xA;&#xA; public void setFileName(String fileName)
{&#xA; this.fileName = fileName;&#xA; }&#xA;}&#xA;Der Test sollte korrekt funktionieren.
```

Full Sample Download

Das gesamte Beispiel finden Sie unter dem Link [springbootrestupload.zip](#).

Feedback

War dieser Blog für Sie wertvoll. Wir danken für jede Anregung und Feedback

Kontakt

Simtech AG
Finkenweg 23
3110 Münsingen
Schweiz

Impressum

Das Copyright für sämtliche Inhalte dieser Website liegt bei Simtech AG, Schweiz.
Beachten Sie auch unsere Hinweise zum Urheberrecht, Datenschutz und Haftungsausschluss.
Jeder Hinweis auf Fehler nehmen wir gerne entgegen.

Copyright

2024 Simtech AG, All rights reserved, Powered by stack.ch written in Golang by Daniel Schmutz

<https://www.simtech-ag.ch/uploaded>