19 Spring MVC应用拓展Resource应用于文件下载

更新时间：2020-06-24 17:59:51

> 不要问你的国家能够为你做些什么，而要问你可以为国家做些什么。——林肯

背景

在 SpringMVC 的开发过程中，有时需要实现文档的下载功能。Word、Excel、PDF 作为大家最常用的办公程序，它的文件传输就显得尤为重要，我们本节就以这些类型为例。

下载的方法有很多种：

1.常见的方式就是**流**：

```
private static final String FILE_PATH = "/tmp/example.pdf";
private static final String APPLICATION_PDF = "application/pdf";

@RequestMapping(value = "/a", method = RequestMethod.GET, produces = APPLICATION_PDF)
public @ResponseBody void downloadA(HttpServletResponse response) throws IOException {
    File file = getFile();
    InputStream in = new FileInputStream(file);

    response.setContentType(APPLICATION_PDF);
    response.setHeader("Content-Disposition", "attachment; filename=" + file.getName());
    response.setHeader("Content-Length", String.valueOf(file.length()));
    FileCopyUtils.copy(in, response.getOutputStream());
}
```

1 输入文件流

2 文件的展示形式

3 输入流copy到response的输出流中。

先将文件转成输入流形式，然后将输入流写入到 response 的输出流中。

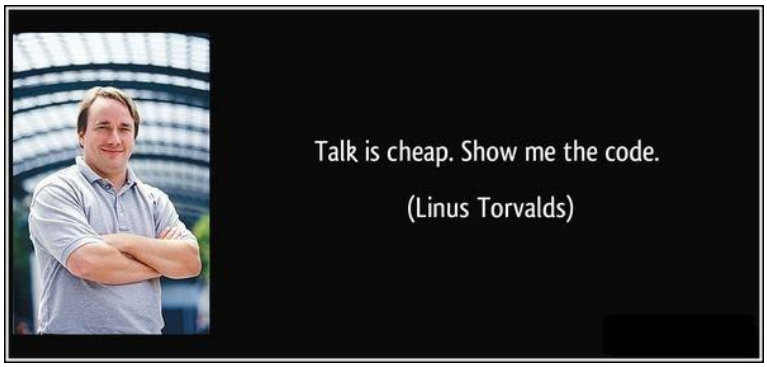
2.也可以通过 **HttpEntity** 来实现：

```
@RequestMapping(value = "/b", method = RequestMethod.GET, produces = APPLICATION_PDF)
public @ResponseBody HttpEntity<byte[]> downloadB() throws IOException {
    File file = getFile();
    byte[] document = FileCopyUtils.copyToByteArray(file);

    HttpHeaders header = new HttpHeaders();
    header.setContentType(new MediaType("application", "pdf"));
    header.set("Content-Disposition", "inline; filename=" + file.getName());
    header.setContentLength(document.length);

    return new HttpEntity<byte[]>(document, header);
}
```

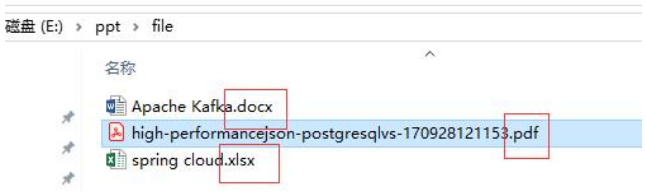
其中，HttpEntity 包含了报文头和报文体。

我们一直强调：一切皆资源。我们如何通过 Resource 来实现实现 docx、xlsx、pdf 文件下载功能呢？

Resource 实现文件下载功能



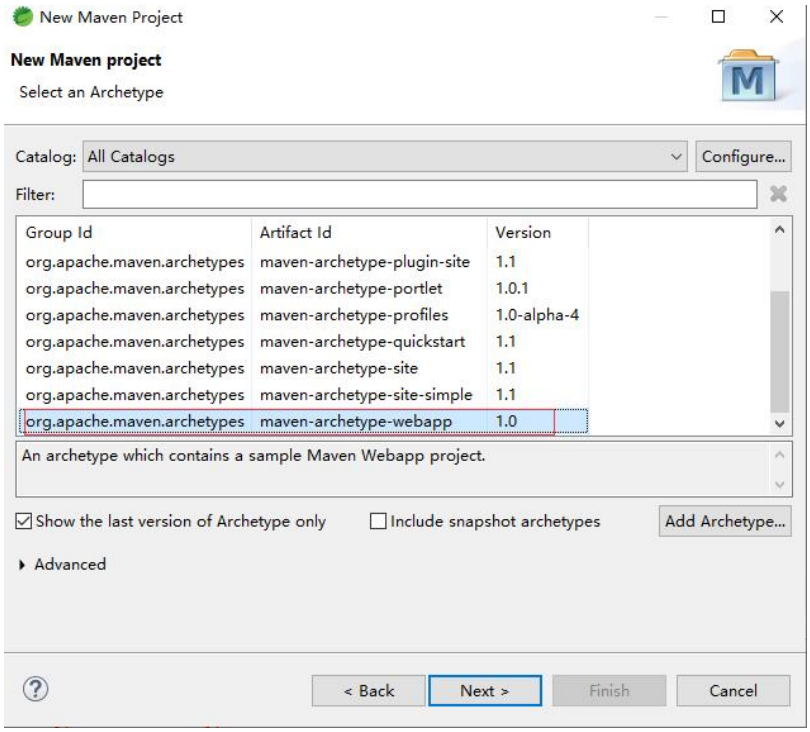
提前准备好文件



三种格式的文件：Word、Excel、PDF。

其中 PDF 可以直接展示，Word 和 Excel 直接作为附件下载。

1.1 创建 Maven 项目，选择 Web 项目



1.2 添加依赖

```
<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>3.8.1</version>
        <scope>test</scope>
    </dependency>
    <!-- Spring dependencies -->
    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-core</artifactId>
        <version>${spring.version}</version>
    </dependency>

    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-web</artifactId>
        <version>${spring.version}</version>
    </dependency>

    <dependency>
        <groupId>org.springframework</groupId>
        <artifactId>spring-webmvc</artifactId>
        <version>${spring.version}</version>
    </dependency>

            <!-- Servlet Dependency -->
    <dependency>
         <groupId>javax.servlet</groupId>
         <artifactId>javax.servlet-api</artifactId>
         <version>3.1.0</version>
         <scope>provided</scope>
    </dependency>
</dependencies>
```

1.3 修改或者添加配置文件

web.xml 替换类

```java
/**
 * Abstract base for exceptions related to media types. Adds a list of supported
 * {@link MediaType MediaTypes}.
 *
 * @author Arjen Poutsma
 * @since 3.0
 */
public class AppInitializer extends AbstractAnnotationConfigDispatcherServletInitializer {
    @Override
    protected Class<?>[] getRootConfigClasses() {
        return new Class<?>[] { AppConfig.class };
    }

    @Override
    protected Class<?>[] getServletConfigClasses() {
        return null;
    }

    @Override
    protected String[] getServletMappings() {
        return new String[] { "/" };
    }
}
```

dispatcher-servlet.xml 替换类

```java
package org.framework.davidwang456.controller;

import org.springframework.context.MessageSource;

/**
 * Abstract base for exceptions related to media types. Adds a list of supported {@link MediaType MediaTypes}.
 *
 * @author Arjen Poutsma
 * @since 3.0
 */
@SuppressWarnings("deprecation")
@Configuration
@EnableWebMvc
@ComponentScan
public class AppConfig extends WebMvcConfigurerAdapter {

    @Bean
    public MessageSource messageSource () {
        ResourceBundleMessageSource messageSource =
                new ResourceBundleMessageSource();
        messageSource.setBasenames("locale/messages");
        messageSource.setDefaultEncoding("UTF-8");
        return messageSource;
    }

    @Override
    public void addFormatters (FormatterRegistry registry) {
        registry.addConverter(new StringToDateTimeConverter());
    }

    @Bean
    public ViewResolver beanNameViewResolver() {
        BeanNameViewResolver resolver = new BeanNameViewResolver();
        return resolver;
    }
}
```

1.4 控制器 Controller

```java
@RequestMapping(value="/download",method=RequestMethod.GET,produces=MediaType.APPLICATION_PDF_VALUE)
@ResponseBody
public Resource download (HttpServletRequest req,HttpServletResponse resp) {
    String fileName=req.getParameter("file");
    File file=getFile(fileName);
    String suffix=fileName.substring(fileName.lastIndexOf(".")+1);
    if(MediaType.APPLICATION_PDF_VALUE.contains(suffix)) {
        resp.setContentType(MediaType.APPLICATION_PDF_VALUE);
        resp.setHeader("Content-Disposition", "inline;filename="+fileName);
    }else if(suffix.equalsIgnoreCase("xlsx")) {
        resp.setContentType("application/vnd.ms-excel");
        resp.setHeader("Content-Disposition", "attachment;filename="+fileName);
    } if("application/msword".contains(suffix)) {
        resp.setContentType("application/msword ");
        resp.setHeader("Content-Disposition", "attachment;filename="+fileName);
    }

    resp.setHeader("Content-Length", String.valueOf(file.length()));
    return new FileSystemResource(file);
}

private File getFile(String fileName) {
    File file=new File("E:/ppt/file/"+fileName);
    if(!file.exists()) {
        System.out.println("指定路径下的文件不存在");
    }
    return file;
}
```

1.5 测试结果

发送请求：http://localhost:8080/mvc1/hello/download?file=high-performancejson-postgresqlvs-170928121153.pdf

PDF 文件进行展示，Word 和 Excel 直接作为附件下载。

拓展

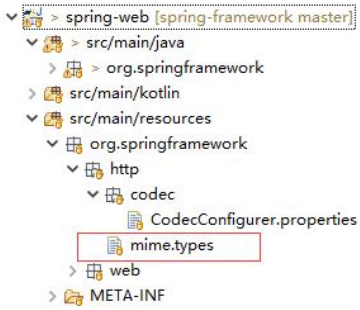
上述示例中 HttpServletResponse 使用了 Content-Type 和报文头 Content-Disposition，他们的用法有什么不同呢？

Content-Type

Content-type 的格式如下：Content-Type = “Content-Type”:" media-type"，它指明了通信使用的类型，示例：

```
Content-Type: text/html; charset=ISO-8859-4
```

其中，media-type 定义在 spring-web 的/org/springframework/http/mime.types 中，种类很多。



它通过 MediaTypeFactory 来产生 MediaType，常用的 MediaType 类型主要有：

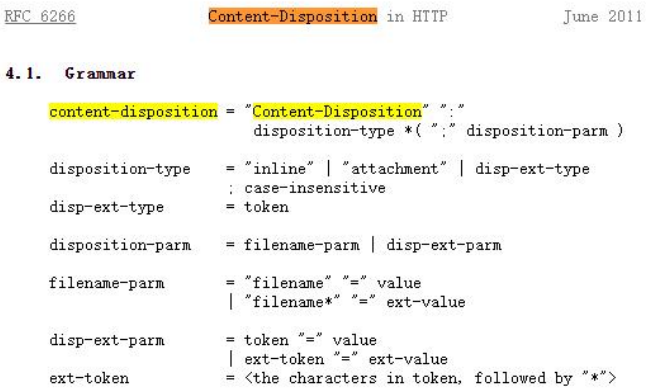
```
static {
    // Not using "valueOf' to avoid static init cost
    ALL = new MediaType("*", "*");
    APPLICATION_ATOM_XML = new MediaType("application", "atom+xml");
    APPLICATION_CBOR = new MediaType("application", "cbor");
    APPLICATION_FORM_URLENCODED = new MediaType("application", "x-www-form-urlencoded");
    APPLICATION_JSON = new MediaType("application", "json");
    APPLICATION_JSON_UTF8 = new MediaType("application", "json", StandardCharsets.UTF_8);
    APPLICATION_OCTET_STREAM = new MediaType("application", "octet-stream");
    APPLICATION_PDF = new MediaType("application", "pdf");
    APPLICATION_PROBLEM_JSON = new MediaType("application", "problem+json");
    APPLICATION_PROBLEM_JSON_UTF8 = new MediaType("application", "problem+json", StandardCharsets.UTF_8);
    APPLICATION_PROBLEM_XML = new MediaType("application", "problem+xml");
    APPLICATION_RSS_XML = new MediaType("application", "rss+xml");
    APPLICATION_STREAM_JSON = new MediaType("application", "stream+json");
    APPLICATION_XHTML_XML = new MediaType("application", "xhtml+xml");
    APPLICATION_XML = new MediaType("application", "xml");
    IMAGE_GIF = new MediaType("image", "gif");
    IMAGE_JPEG = new MediaType("image", "jpeg");
    IMAGE_PNG = new MediaType("image", "png");
    MULTIPART_FORM_DATA = new MediaType("multipart", "form-data");
    MULTIPART_MIXED = new MediaType("multipart", "mixed");
    MULTIPART_RELATED = new MediaType("multipart", "related");
    TEXT_EVENT_STREAM = new MediaType("text", "event-stream");
    TEXT_HTML = new MediaType("text", "html");
    TEXT_MARKDOWN = new MediaType("text", "markdown");
    TEXT_PLAIN = new MediaType("text", "plain");
    TEXT_XML = new MediaType("text", "xml");
}
```

注意：当你在响应类型为 application/octet- stream 情况下（“Content-type: application/octet-stream”）使用了这个头信息的话，那就意味着你不想直接显示内容，而是弹出一个"文件下载"的对话框，接下来就是由你来决定"打开"还是"保存" 了。

Content-disposition

Content-disposition 是 MIME 协议的扩展，MIME 协议指示 MIME 用户代理如何显示附加的文件。Content-disposition 其实可以控制用户请求所得的内容存为一个文件的时候提供一个默认的文件名，文件直接在浏览器上显示或者在访问时弹出文件下载对话框。

Content-Disposition 的格式如下：



RFC 6266 Content-Disposition in HTTP June 2011

4.1. Grammar

```
content-disposition = "Content-Disposition" ":"
                      disposition-type *( ";" disposition-parm )

disposition-type    = "inline" | "attachment" | disp-ext-type
                    ; case-insensitive
disp-ext-type       = token

disposition-parm    = filename-parm | disp-ext-parm

filename-parm       = "filename" "=" value
                    | "filename*" "=" ext-value

disp-ext-parm       = token "=" value
                    | ext-token "=" ext-value
ext-token           = <the characters in token, followed by "*">
```

总结

Resource 集成了 InputStreamSource，在拥有 InputStreamSource 的 getInputStream 的同时，也封装了很多其它功能。Resource 抽象了物理世界中的资源概念：一切皆资源。

它的很多实现类如
UrlResource，FileUrlResource，FileSystemResource，ClassPathResource，ByteArrayResource，InputStreamResource 使它可以操作 URL、文件、URL 形式的文件，classpath 下的资源，byte 数组资源和 InputStream 等资源。Resource 接口封装了各种可能的资源类型，也就是对使用者来说屏蔽了文件类型的不同。对资源的提供者来说，如何把资源包装起来交给其他人用这也是一个问题，我们看到 Resource 接口继承了 InputStreamSource 接口，这个接口中有个 getInputStream 方法，返回的是 InputStream 类。这样所有的资源都被可以通过 InputStream 这个类来获取，所以也屏蔽了资源的提供者。

}

18 Spring IoC容器如何读取应用外部的xml，txt，图形或者属性文件？

20 面试官，请不要再问我@Resource和@Autowire注解的区别了