1. 파일을 첨부하는 자료실 기능만들기(게시판+파일첨부=>자료실이 된다.)

1. web.xml을 이용하는 경우의 첨부파일 설정

```
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns="http://xmlns.jcp.org/xml/ns/javaee"
        xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
        id="WebApp_ID" version="3.1">

1. Spring Legacy Project 에서는 서블릿 버전이 2.5이기 때문에 xml 네임스페이스를 2.5에서 3.1로 수정한다.

2. 톰캣 7.0 버전 이후부터는  서블릿 3.0이상을 지원하므로 3.0이상부터는 pom.xml에 이진 업로드 라이브러리를 추가하지 않고, 서블릿 자체적인 파일업로드 api를 사용한다.
```

2. web.xml 의 <servlet> 태그 내에는 <multipart-config> 태그를 추가한다.

```
<servlet>
                <servlet-name>appServlet</servlet-name>

<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
                <init-param>
                        <param-name>contextConfigLocation</param-name>

<param-value>/WEB-INF/spring/appServlet/servlet-context.xml</param-value>
                </init-param>
                <load-on-startup>1</load-on-startup>

                <!-- 이진파일 업로드 설정 -->
                <multipart-config>
                        <location>C:₩₩upload₩₩temp</location><!-- 업로드 되는 경로. 첨부파일은 실제 서버가 동작하는 컴퓨터 내에 있는 폴더에
                        업로드 시켜야 하므로 c드라이브 밑에 upload폴더와  임시 업로드 파일을 저장할 temp 폴더를 각각 생성 -->
                        <max-file-size>20971520</max-file-size> <!--1MB * 20 =>업로드 되는 파일 최대크기 -->
                        <max-request-size>41943040</max-request-size><!-- 40MB => 한번에 올릴 수 있는 최대크기 -->
                        <file-size-threshold>20971520</file-size-threshold>    <!-- 20MB => 메모리 사용 크기 -->
                </multipart-config>
        </servlet>
```

3. 스프링에서 업로드 처리는 MultipartResolver 라는 타입의 객체를 빈으로 등록해야 한다. Web과 관련된 설정이므로 servlet-context.xml에서 다음과 같이 설정한다.

```
<!-- 스프링 이진파일 업로드 api 설정 => 첨부파일을 처리하는 빈 id 설정.-->
	<beans:bean id="multipartResolver"
class="org.springframework.web.multipart.support.StandardServletMultipartResolver" />
```

2. <form> 방식의 파일 업로드

1. 서버상에서 첨부파일 처리는 컨트롤러에서 이루어지므로, 실습을 위해서 org.zerock.controller 패키지에 UploadController.java 컨트롤러 클래스를 만든다. 이 클래스에는 get방식으로 첨부파일을 업로드 할 수 있는 화면을 처리하는 메서드와 post 방식으로 첨부파일을 업로드해서 처리해주는 메서드를 추가한다.

```
package org.zerock.controller;

import java.io.File;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.multipart.MultipartFile;

@Controller
public class UploadController {



	@GetMapping("uploadForm")	//get으로 접근하는 매핑주소 처리.
//@GetMapping 애노테이션은 스프링 4.3에서 추가됨.
	public void uploadForm() {
		//리턴타입이 없는 void형이면 매핑주소가 jsp파일명이 된다.
	}//uploadForm


	@PostMapping("/uploadFormAction")	//post로 접근하는 매핑주소 처리.
//@PostMapping 애노테이션도 스프링 4.3에서 추가됨.
	public void uploadFormAction(MultipartFile[] uploadFile, Model model) {
		//스프링에서 MultipartFile 타입을 제공해서 업로드 되는 파일 데이터
//를 쉽게 처리.다중 업로드 파일은 배열로 받는다. <input type="file"
		//name="uploadFile" />네임피라미터이름(네임속성명)을 매개변수명으
//로 지정해서 처리한다.
```

```
		String uploadFolder = "C:₩₩upload";//이진파일 업로드 경로

		for (MultipartFile multipartFile : uploadFile) {

			System.out.println("-------------------------------------");
			System.out.println("Upload File Name: " +
multipartFile.getOriginalFilename());//원래 업로드 원본파일명
			System.out.println("Upload File Size: " +
multipartFile.getSize());//업로드 파일크기

			File saveFile = new File(uploadFolder,
multipartFile.getOriginalFilename());

			try {
				multipartFile.transferTo(saveFile);//업로드 되는 원
//래 파일명으로 c드라이브 upload폴더에 저장
			} catch (Exception e) {
				e.printStackTrace();
			} // end catch
		} // end for

	}//uploadFormAction()
}
```

2. uploadForm.jsp 파일첨부 뷰페이지

가. 파일첨부한 파일을 이진파일이라 한다. 이 이진파일을 서버에 업로드 할려면 반드시 method=post방식이어야 한다. get 방식은 안된다. form태그내에서 method속성을 생략하면 기본값이 get이다. 그러므로 파일을 첨부하기 위해서는 method 속성을 생략하면 안된다.

나. 파일 첨부 기능을 만들기 위해서는 반드시 form태그내에서 enctype= "multipart/form-data" 속성을 꼭 지정해야 한다.

```
<%@ page contentType="text/html; charset=UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title> </title>
</head>
<body>
```

```
<form method="post" action="uploadFormAction" enctype="multipart/form-data">

<input type='file' name='uploadFile' multiple>
<%-- 최근 브라우저에서는 'multiple'속성을 지원하는데 이를 이용하면 하나의 input type="file"로 다중 이진파일을 동시에
업로드 할 수 있다. 이 속성은 IE10 이상에서만 사용할 수 있다.--%>
<input type="submit" value="파일업로드" />
</form>
</body>
</html>
```

3. Ajax를 이용하는 파일 업로드

1. 첨부파일을 업로드하는 또 다른 방식은 Ajax를 이용해서 파일 데이터만을 전송하는 방식이다. Ajax를 이용하는 첨부파일 처리는 FormData라는 객체를 이용하는 데 IE의 경우 10 버전 이후부터 지원된다.

2. UploadController.java에서 get 방식으로 첨부파일을 업로드하는 뷰페이지를 제작한다.

```
...
 .. 중략
        @GetMapping("/uploadAjax")
        public void uploadAjax() {//리턴타입이 없는 경우 매핑주소가 jsp파일명이 된
        //다. 뷰페이지 경로 => /WEB-INF/views/uploadAjax.jsp 가 된다.
        }//ajax를 이용하는 파일 업로드
..중략
```

3. uploadAjax.jsp

```
<%@ page contentType="text/html; charset=UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title></title>
<script src="./resources/js/jquery.js"></script>
<script>
        $(document).ready(function() {

                $("#uploadBtn").on("click", function(e) {
```

```
                        var formData = new FormData();
                        //첨부파일을 업로드 하는 또 다른 방식은 jQuery ajax를
//이용해서 파일 데이터만을 전송.
//ajax를 이용하는 첨부파일 처리는 FormData라는 객체를 이용하는데 IE의 경우 10
//이후 버전부터 지원되므로 브라우저에 제약이 있을 수 있다.
                        var inputFile = $("input[name='uploadFile']");
//file 객체를 구함
                        var files = inputFile[0].files;
//첫번째 파일객체에서 첨부한 파일을 배열로 구한다.
                        console.log(files);//이클립스 콘솔모드에 첨부한 파일을
//출력

                        //첨부파일을 forData객체에 추가
                        for (var i = 0; i < files.length; i++) {

                                formData.append("uploadFile", files[i]);

                        }

                        $.ajax({
                                url : '/controller/uploadAjaxAction',
                                processData : false,//false로 지정해야 전송
//processData 데이터를 컨텐트 타입에 맞게 변환 여부
                                contentType : false,
//요청 컨텐트 타입
                                data : formData,//Ajax를 통해서 formData
// 자체를 전송
                                type : 'POST',//보내는 방식
                                success : function(result) {
//받아오는 성공시 호출되는 콜백함수. 받아온 데이터는 result매개변수에 저장
                                        alert("Uploaded ok");
                                }
                        }); //$.ajax=>jQuery 아작스

                });
        });
</script>
</head>
<body>
```

```
	<h1>Upload with Ajax</h1>

	<input type='file' name='uploadFile' multiple>
	<%-- 최근 브라우저에서는 'multiple'속성을 지원하는데 이를 이용하면 하나의
input type="file"로 다중 이진파일을 동시에
업로드 할 수 있다. 이 속성은 IE10 이상에서만 사용할 수 있다.--%>
	<hr />
	<button id='uploadBtn'>Upload</button>
</body>
</html>
```

4. UploadController.java의 일부

```
@PostMapping("/uploadAjaxAction")
	public void uploadAjaxPost(MultipartFile[] uploadFile) {

		System.out.println("update ajax post.....");

		String uploadFolder = "C:₩₩upload";
//이진 파일 업로드경로

		for (MultipartFile multipartFile : uploadFile) {

			System.out.println("-------------------------------------");
			System.out.println("Upload File Name: " +
multipartFile.getOriginalFilename());
			System.out.println("Upload File Size: " +
multipartFile.getSize());

			String uploadFileName =
multipartFile.getOriginalFilename();

			//IE 경우 전체 파일 경로가 전송되기 때문에 마지막 '₩'을
기준으로 잘라낸 문자열이 실제 파일명이 된다.
			uploadFileName =
uploadFileName.substring(uploadFileName.lastIndexOf("₩₩") +
					1);
			System.out.println("only file name: " + uploadFileName);

			File saveFile = new File(uploadFolder, uploadFileName);
```

```
                        try {

                                multipartFile.transferTo(saveFile);
                        } catch (Exception e) {
                                e.printStackTrace();
                        } // end catch

                } // end for

        }
```