st13.파일업로드

목차

1. build.gradle 의 dependency 에 라이브러리 추가

- commons-io
- commons-fileupload

2. servlet-context.xml 에 아래 코드 추가

```
<!—step6. 파일 업로드를 위한 MultipartResolver 설정 -->
<!-- 단위는 byte 로 100,000,000byte 이기 때문에 100MB 로 설정 -->
<beans:bean id="multipartResolver"
            class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
    <beans:property name="maxUploadSize" value="100000000" />
    <beans:property name="maxInMemorySize" value="100000000" />
</beans:bean>
```

3. 파일 업로드

3.1 테이블 생성

3.1.1 oracle 용 ddl 스크립트

```
BEGIN EXECUTE IMMEDIATE 'DROP TABLE TB_Upload_File CASCADE CONSTRAINTS PURGE'; EXCEPTION
WHEN OTHERS THEN NULL; END;
/

CREATE TABLE TB_Upload_File (
      uploadFileNo    NUMBER(10)      generated as identity
    , fileNameOrig    VARCHAR2(100)   NOT NULL  -- 실제 파일명
    , fileNameTemp    VARCHAR2(100)   NOT NULL  -- 서버에 저장되는 임시 파일명
    , fileSize        NUMBER(10)      NOT NULL
    , contentType     VARCHAR2(30)    NOT NULL

    , PRIMARY KEY(uploadFileNo)
    , UNIQUE INDEX(fileNameTemp)
)
;
```

3.1.2 mysql 용 ddl 스크립트

```
DROP TABLE IF EXISTS TB_Upload_File;
CREATE TABLE TB_Upload_File (
      uploadFileNo    INT UNSIGNED   NOT NULL AUTO_INCREMENT
    , fileNameOrig    NVARCHAR(100)  NOT NULL  -- 진짜 파일명
    , fileNameTemp    NVARCHAR(100)  NOT NULL  -- 서버에 저장되는 임시 파일명
    , fileSize        INT UNSIGNED   NOT NULL
    , contentType     NVARCHAR(30)   NOT NULL

    , PRIMARY KEY(uploadFileNo)
    , UNIQUE INDEX(fileNameTemp)
)
ENGINE=InnoDB
AUTO_INCREMENT=1
DEFAULT CHARACTER SET utf8
COLLATE utf8_general_ci
;
```

3.2 모델 클래스 만들기

3.2.1 ModelUploadFile

```
public class ModelUploadFile {

    private Integer uploadFileNo   ;
    private String  fileNameOrig   ;
    private String  fileNameTemp   ;
    private Long    fileSize       ;
    private String  contentType    ;

    // getter & setter
    public Integer getUploadFileNo() {
        return uploadFileNo;
    }
    public void setUploadFileNo(Integer uploadFileNo) {
        this.uploadFileNo = uploadFileNo;
    }
    public String getFileNameOrig() {
        return fileNameOrig;
    }
    public void setFileNameOrig(String fileNameOrig) {
        this.fileNameOrig = fileNameOrig;
    }
    public String getFileNameTemp() {
        return fileNameTemp;
    }
    public void setFileNameTemp(String fileNameTemp) {
        this.fileNameTemp = fileNameTemp;
    }
    public Long getFileSize() {
        return fileSize;
    }
    public void setFileSize(Long fileSize) {
        this.fileSize = fileSize;
    }
    public String getContentType() {
        return contentType;
    }
    public void setContentType(String contentType) {
        this.contentType = contentType;
    }

    // toString
```

```
    @Override
    public String toString() {
        return "ModelUploadFile [uploadFileNo=" + uploadFileNo + ", fileNameOrig="
                + fileNameOrig + ", fileNameTemp=" + fileNameTemp + ", fileSize="
                + fileSize + ", contentType=" + contentType + "]";
    }

    // constructor
    public ModelUploadFile() {
        super();
    }
}
```

3.3 mapperUpload.xml 파일 생성(Mybatis mapper)

3.3.1 mysql 용 mapper 파일

```
<mapper namespace="mapper.mapperUpload">

    <select id="getUploadFile" parameterType="ModelUploadFile"
 resultType="ModelUploadFile">
        SELECT *
          FROM TB_Upload_File
         WHERE 1 = 1
            <if test=" fileNameTemp != null and fileNameTemp != '' ">
            and fileNameTemp = #{fileNameTemp}
            </if>
            <if test=" uploadFileNo != null ">
            and uploadFileNo = #{uploadFileNo}
            </if>
    </select>

    <!-- useGeneratedKeys="true" keyProperty="uploadFileNo" 은
inserted 된 primary key 값을 반환하기 위한 설정 -->
    <insert id="insertUploadFile" parameterType="ModelUploadFile"
                                  useGeneratedKeys="true" keyProperty="uploadFileNo">
        INSERT INTO
        TB_Upload_File (   fileNameOrig ,   fileNameTemp ,   contentType ,   fileSize )
        VALUES         ( #{fileNameOrig}, #{fileNameTemp}, #{contentType}, #{fileSize})
    </insert>

    <!-- mysql multi insert
        INSERT INTO t (col1, col2, col3)
               VALUES ('val1_1', 'val1_2', 'val1_3')
, ('val2_1', 'val2_2', 'val2_3')
, ('val3_1', 'val3_2', 'val3_3')  ;
     -->
    <insert id="insertUploadFileList" parameterType="java.util.List">
        INSERT INTO
        TB_Upload_File (   fileNameOrig ,   fileNameTemp ,   contentType ,   fileSize )
        VALUES
        <foreach collection="list" item="item" index="index" open="(" separator="), ("
close=")">
        #{item.fileNameOrig}, #{item.fileNameTemp}, #{item.contentType}, #{item.fileSize}
        </foreach>
    </insert>
```

```
    <delete id="deleteUploadFile" parameterType="ModelUploadFile">
        DELETE FROM TB_Upload_File
         WHERE 1 = 1
            <if test=" fileNameTemp != null and fileNameTemp != '' ">
            and fileNameTemp = #{fileNameTemp},
            </if>
            <if test=" uploadFileNo != null ">
            and uploadFileNo = #{uploadFileNo}
            </if>
    </delete>

</mapper>
```

3.3.2 oralce 용 mapper 파일

```
<mapper namespace="mapper.mapperUpload">
    <select id="getUploadFile" parameterType="ModelUploadFile"
resultType="ModelUploadFile">
        SELECT *
          FROM TB_Upload_File
         WHERE 1 = 1
            <if test=" fileNameTemp != null and fileNameTemp != '' ">
            and fileNameTemp = #{fileNameTemp}
            </if>
            <if test=" uploadFileNo != null ">
            and uploadFileNo = #{uploadFileNo}
            </if>
    </select>

    <insert id="insertUploadFile" parameterType="ModelUploadFile"
                                  useGeneratedKeys="true" keyProperty="uploadFileNo">
        INSERT INTO
        TB_Upload_File (   fileNameOrig ,   fileNameTemp ,   contentType ,   fileSize )
        VALUES         ( #{fileNameOrig}, #{fileNameTemp}, #{contentType}, #{fileSize})
    </insert>

<insert id="insertUploadFileList" parameterType="java.util.List">
        INSERT INTO
        TB_Upload_File (   fileNameOrig ,   fileNameTemp ,   contentType ,   fileSize )

    </insert>

    <delete id="deleteUploadFile" parameterType="ModelUploadFile">
        DELETE FROM TB_Upload_File
         WHERE 1 = 1
            <if test=" fileNameTemp != null and fileNameTemp != '' ">
            and fileNameTemp = #{fileNameTemp},
            </if>
            <if test=" uploadFileNo != null ">
            and uploadFileNo = #{uploadFileNo}
            </if>
    </delete>
    <select id="getImageList" parameterType="int" resultType="ModelUploadImage">
        SELECT uploadImageNo, fileName, contentType, fileSize, imageBytes, imageBase64
        FROM TB_Upload_Image
    </select>
</mapper>
```

3.4 Dao 만들기

3.4.1 IDaoUploadFile 인테이스 만들기

```
public interface IDaoUploadFile {
List<ModelUploadFile> getUploadFile( ModelUploadFile file );
    int insertUploadFile( ModelUploadFile file );
    int insertUploadFileList( List<ModelUploadFile> files );
    int deleteUploadFile( ModelUploadFile file );
}
```

3.4.2 DaoUploadFile 클래스 만들기

```
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Repository;

import com.spring67.upload.model.*;

@Repository
public class DaoUploadFile implements IDaoUploadFile {

    @Autowired
    @Qualifier("sqlSession")
    private org.apache.ibatis.session.SqlSession session;


@Override
public List<ModelUploadFile> getUploadFile(ModelUploadFile fileinfo) {
        return session.selectList("mapper.mapperUpload.getUploadFile", fileinfo );
}

@Override
    public int insertUploadFile(ModelUploadFile fileinfo) {

       /* // Oracle 인 경우
        Map<String, Object> map = new  HashMap<String, Object>();
        map.put("fileinfo"  , fileinfo);
        map.put("result", null);

        session.insert("mapper.mapperUpload.insertUploadFile", map);

// inserted 된 primary key 값이 반환됨
        int result = map.get("result") != null ? (int) map.get("result") : -1;
```

```
        return result;
        */

        // mysql 인 경우
        session.insert("mapper.mapperUpload.insertUploadFile",fileinfo);
// inserted 된 primary key 값이 반환됨
        return fileinfo.getUploadFileNo();
    }

@Override
public int insertUploadFileList(List<ModelUploadFile> files) {
        return session.insert("mapper.mapperUpload.insertUploadFileList", files);
}

@Override
public int deleteUploadFile(ModelUploadFile fileinfo) {
        return session.delete("mapper.mapperUpload.deleteUploadFile", fileinfo);
}

}
```

3.5 Service 만들기

3.5.1 IServiceUploadFile 인터페이스 만들기

```
public interface IServiceUploadFile extends IDaoUploadFile {
}
```

3.5.2 ServiceUpload 클래스 만들기

```
import java.util.List;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class ServiceUpload implements IServiceUploadFile {

    // SLF4J Logging
    private Logger logger = LoggerFactory.getLogger(this.getClass());

    @Autowired
    private IDaoUploadFile uploaddao;


      @Override
      public List<ModelUploadFile> getUploadFile(ModelUploadFile file) {

            List<ModelUploadFile> result = null;
        try {
            result = uploaddao.getUploadFile(file);
        } catch (Exception e) {
            logger.error("getUploadFile " + e.getMessage() );
        }
        return result;
      }

    @Override
    public int insertUploadFile(ModelUploadFile file) {

        int result = -1;
        try {
```

```
            result = uploaddao.insertUploadFile(file);
        } catch (Exception e) {
            logger.error("insertUploadFile " + e.getMessage() );
        }
        return result;
    }


    @Override
    public int insertUploadFileList(List<ModelUploadFile> files) {

        int result = -1;
        try {
            result = uploaddao.insertUploadFileList(files);
        } catch (Exception e) {
            logger.error("insertUploadFileList " + e.getMessage() );
        }
        return result;
    }


    @Override
    public int deleteUploadFile(ModelUploadFile file) {

        int result = -1;
        try {
            result = uploaddao.deleteUploadFile(file);
        } catch (Exception e) {
            logger.error("deleteUploadFile " + e.getMessage() );
        }
        return result;
    }

}
```

3.6 RepositoryFiles 클래스 작성

- RepositoryFiles 클래스는 한번에 여러 개의 파일을 업로드하는 경우에 사용된다.
- MultipartFile 클래스는 commons-fileupload 라이브러리에 들어있다. import 를 하면 사용 할 수 있다.
- 클라이언트에서 배열로 넘오는 변수를 자바의 List 로 변한하기 위한 클래스

```
import java.util.List;

import org.springframework.stereotype.Repository;
import org.springframework.web.multipart.MultipartFile;

// 클라이언트에서 배열로 넘오는 변수를 자바의 List 로 변한하기 위한 클래스
@Repository
public class RepositoryFiles {

    // 필드명으로는 클라이언트에서 넘어오는 변수명,
    // 즉, input 태그의 name 속성 값을 필드명으로 사용해야 한다.
    // <input type="file" name="files[0]" />
    private List<org.springframework.web.multipart.MultipartFile> files;

    public List<MultipartFile> getFiles() {
        return files;
    }

    public void setFiles(List<MultipartFile> files) {
        this.files = files;
    }

    public RepositoryFiles() {
        super();
    }
```

3.7 테스트 코드 만들기

```
import static org.junit.Assert.*;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.List;

import org.junit.BeforeClass;
import org.junit.Test;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.springframework.mock.web.MockMultipartFile;
import org.springframework.web.multipart.MultipartFile;

import com.spring67.upload.model.ModelUploadFile;
import com.spring67.upload.repository.RepositoryFiles;
import com.spring67.upload.service.IServiceUploadFile;


public class TestUploadFile {
    // SLF4J Logging
    private Logger logger = LoggerFactory.getLogger(this.getClass());

    private static ApplicationContext context;
    private static IServiceUploadFile service;

    @BeforeClass
    public static void setUpBeforeClass() throws Exception {

        context = new ClassPathXmlApplicationContext("file:src/main/webapp/WEB-
INF/spring/appServlet/servlet-context.xml");
        service = context.getBean(IServiceUploadFile.class);
    }

    @Test
    public void test01_InsertUploadFile() throws IOException {
        int result = -1;

        // 테스트용 변수 설정 : client 로 부터 받게 되는 값들.
        File uploadfile = new File("src/test/resources/image.png");
        FileInputStream inputFile = new FileInputStream( uploadfile.getAbsolutePath() );
```

```
        MockMultipartFile clinetfile = new MockMultipartFile( uploadfile.getName(),
uploadfile.getName(), "image/png", inputFile);
        String UPLOAD_PATH = "C:\\"; // 클라이언트에서 올라온 image.png 를 저장할 서버의
폴더 위치.

        // step2. 클라이언트의 첨부 파일을 서버로 올리기 위한 코드
        if( !clinetfile.getOriginalFilename().isEmpty() ){

            // 서버의 업로드 폴더 존재 여부 체크. 없으면 폴더 생성.
            java.io.File uploadDir = new java.io.File( UPLOAD_PATH  );
            if( !uploadDir.exists() && !uploadDir.isDirectory() ) uploadDir.mkdir();

            // 클라이언트의 첨부 파일을 서버로 복사
            String fileName  = clinetfile.getOriginalFilename();
            String tempName  =
LocalDateTime.now().format(DateTimeFormatter.ofPattern("yyyyMMddHHmmss"));
            String filepath  = UPLOAD_PATH + tempName;
            java.io.File serverfile   = new java.io.File( filepath );
            clinetfile.transferTo( serverfile ); // 실제로 파일 카피 발생.

            // DB insert 처리를 위한 코드
            ModelUploadFile attachfile = new ModelUploadFile();
            attachfile.setFileNameOrig( fileName );
            attachfile.setFileNameTemp( tempName );
            attachfile.setFileSize( (long)serverfile.length() );
            attachfile.setContentType( clinetfile.getContentType() ); // 파일 확장자

            // DB insert
            result = service.insertUploadFile(attachfile);
        }

        assertTrue( result > 0 );
    }

    @Test
    public void test02_InsertUploadFileList() throws IOException {
        int result = -1;

        // step1. 테스트용 변수 설정 : client 로 부터 받게 되는 값들.
        List<MultipartFile> localfiles = new ArrayList<>();
        for(int i=0; i<10; i++){
            File uploadfile           = new File("src/test/resources/image.png");
            FileInputStream inputFile = new
FileInputStream( uploadfile.getAbsolutePath() );
            MockMultipartFile file    = new MockMultipartFile( uploadfile.getName(),
uploadfile.getName(), "image/png", inputFile);
```

```
            localfiles.add(file);
        }
        RepositoryFiles uploadForm = new RepositoryFiles();
        uploadForm.setFiles(localfiles);
        String UPLOAD_PATH = "C:\\aaaaaa";


        // step2. 로컬 첨부 파일을 서버로 올리기 위한 코드
        if( uploadForm != null && uploadForm.getFiles().size()>0 ){

            // 업로드 폴더 존재 여부 체크. 없으면 폴더 생성.
            java.io.File uploadDir = new java.io.File( UPLOAD_PATH  );
            if( !uploadDir.exists() && !uploadDir.isDirectory() ) uploadDir.mkdir();

            // DB 에 insert 할 데이터를 닫는 list
            List<ModelUploadFile> listfile = new ArrayList<>();

            List<MultipartFile> files = uploadForm.getFiles();
            if( files != null && files.size() > 0 ){

                int index = 0;
                for (MultipartFile clientfile : files) {

                    // 클라이언트의 첨부 파일을 서버로 복사
                    String fileName = clientfile.getOriginalFilename();
                    String tempName =
LocalDateTime.now().format(DateTimeFormatter.ofPattern("yyyyMMddHHmmss")) +
String.format("%04d", ++index);
                    String filepath = UPLOAD_PATH + "/" + tempName;
                    java.io.File serverfile   = new java.io.File( filepath );
                    clientfile.transferTo( serverfile );

                    // DB insert 처리를 위한 코드
                    ModelUploadFile attachfile = new ModelUploadFile();
                    attachfile.setFileNameOrig( fileName );
                    attachfile.setFileNameTemp( tempName );
                    attachfile.setFileSize( (long)serverfile.length() );
                    attachfile.setContentType( clientfile.getContentType() ); // 확장자

                    //
                    listfile.add(attachfile);
                }
            }

            // DB insert
```

```
            result = service.insertUploadFileList(listfile);
        }

        assertTrue( result > 0 );
    }


    @Test
    public void test03_GetUploadFile() {
        ModelUploadFile file = new ModelUploadFile();
        file.setUploadFileNo(1);

        // DB select
        List<ModelUploadFile> result = service.getUploadFile(file);

        assertNotNull(result);
        assertSame( 1, result.get(0).getUploadFileNo() );
        assertTrue( !result.get(0).getFileNameTemp().isEmpty() );



        result = service.getUploadFile( null );

        assertNotNull(result);
        assertSame( 1, result.get(0).getUploadFileNo() );
        assertTrue( !result.get(0).getFileNameTemp().isEmpty() );
    }


    @Test
    public void test04_DeleteUploadFile() {
        ModelUploadFile file = new ModelUploadFile();
        file.setUploadFileNo(1);

        // DB delete
        int result = service.deleteUploadFile(file);

        assertSame(1, result);
    }
}
```

3.8 jsp 파일 만들기

3.8.1 uploadfile.jsp 작성

src/main/webapp/WEB-INF/views/upload 폴더 아래에 uploadfile.jsp 파일을 만든다.

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<html>
<head>
    <meta charset="utf-8" />
    <title>file  upload demo</title>
    <script src="http://code.jquery.com/jquery-latest.js"></script>
    <script>
        <!-- jquery 로 파일 첨부 input 태그 추가 -->
        $(document).ready(function() {

            //add more file components if Add is clicked
            $('#addFile').click(function() {

                var fileIndex = $('#fileview tr').children().length;
                $('#fileview').append(
                '<tr><td>' +
                ' <span class="deletefile">X</span> ' +
                '   <input type="file" name="files['+ fileIndex +']" />' +
                '</td></tr>');
            });

            $('#fileview').on('click', '.deletefile' , function(e){
                  $(this).parent().remove();
            });
        });
    </script>
</head>
<body>
    <h3>Spring Single File Upload</h3>
    <form method="post" action="./uploadfileone" enctype="multipart/form-data">

        Upload Directory :
        <input type="text" name="upDir" value="c:/upload/" />
        <br />
        <br />
        <input type="file"  name="file" />
```

```
        <br />
        <input type="submit" id="uploadone"   value="Upload One" />
    </form>

    <hr>

    <h3>Spring Multi File Upload</h3>
    <form method="post" action="./uploadfilemulti" enctype="multipart/form-data">

        Upload Directory :
        <input type="text" name="upDir" value="c:/upload/" />
        <br />
        <br />
        <input type="button" id="addFile" value="File Add" />

        <table id="fileview">
            <tr>
                <td>
                    <span class="deletefile">X</span>
                    <input type="file"  name="files[0]" />
                </td>
            </tr>
        </table>
        <br />
        <input type="submit" id="uploadmulti" value="Upload Multi" />
    </form>

    <hr>

    <h3>Drag &amp; Drop Multi File Upload</h3>
    <form method="post" action="" modelAttribute="uploadForm" enctype="multipart/form-
data">

    </form>
</body>
</html>
```

3.8.2 uploadfilelist.jsp 작성

src/main/webapp/WEB-INF/views/upload 폴더 아래에 uploadfilelist.jsp 파일을 만든다.

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<html>
<head>
    <meta charset="utf-8" />
    <style>
        table, td, th { border: 1px solid green; }
        th { background-color: green; color: white; }
    </style>
    <script>
        var download = function download(filenametemp, filenameorig) {

              var f = document.createElement('form');

              f.style.visibility = "hidden";

              f.setAttribute('method','post');
              f.setAttribute('action','/download');
              f.setAttribute('enctype','application/x-www-form-urlencoded');
              document.body.appendChild(f);

              var i = document.createElement('input'); //input element, text
              i.type = 'text';
              i.name = 'filenametemp';
              i.value = filenametemp;
              f.appendChild(i);

              var i = document.createElement('input'); //input element, text
              i.type = 'text';
              i.name = 'filenameorig';
              i.value = filenameorig;
              f.appendChild(i);

              f.submit();
          };
    </script>
</head>
<body>
```

```
    <table>
        <tr>
            <th>no</th>
            <th>fileNameOrig</th>
            <th>fileNameTemp</th>
            <th>download    </th>
        </tr>

        <!--  반복 구간 시작 -->
        <c:forEach var="file" items="${files }" varStatus="status">
        <tr>
            <td>${file.uploadFileNo}</td>
            <td>${file.fileNameOrig}</td>
            <td>${file.fileNameTemp}</td>
            <td><a href="javascript:download('${file.fileNameTemp }',
'${file.fileNameOrig }')">download</a></td>
        </tr>
        </c:forEach>
        <!--  반복 구간 끝 -->
    </table>

</body>
</html>
```

3.8.3 download.jsp 작성

- Http Response 메시지 포맷

Http 의 Response 는 "상태라인", "헤더", "본문"으로 구성되며 파일 다운로드시

✓ header 에는 파일명, 파일사이즈, MIME 타입의 정보가 담겨진다.

✓ Body 부분에는 실제 파일이 담겨서 클라이언트로 넘어오게 된다.

st13.파일업로드

src/main/webapp/WEB-INF/views/upload 폴더 아래에 download.jsp 파일을 만든다.

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>

<%@ page import="java.io.File" %>
<%@ page import="java.net.URLEncoder" %>
<%@ page import="java.io.OutputStream" %>
<%@ page import="java.io.FileInputStream" %>
<%@ page import="java.io.IOException" %>
<%@ page import="org.springframework.util.FileCopyUtils" %>

<%
    //request.setCharacterEncoding("UTF-8");//이 작업은 필터가 한다.
    String filenameorig = request.getParameter("filenameorig");
    String filenametemp = request.getParameter("filenametemp");

    File file = new File("c:\\upload\\" + filenametemp);

    String filetype = filenameorig.substring(filenameorig.indexOf(".") + 1,
filenameorig.length());

    if (filetype.trim().equalsIgnoreCase("txt")) {
        response.setContentType("text/plain");
    } else {
        response.setContentType("application/octet-stream");
    }

    response.setContentLength((int) file.length());

    boolean ie = request.getHeader("User-Agent").indexOf("MSIE") != -1;

    if (ie) {
        filenameorig = URLEncoder.encode(filenameorig, "UTF-8").replaceAll("\\+", " ");
    } else {
        filenameorig = new String(filenameorig.getBytes("UTF-8"), "8859_1");
    }

    // response header 에 쓰는 부분
    response.setHeader("Content-Disposition", "attachment; filename=\"" + filenameorig +
"\"");

    // response body 에 쓰는 부분
    OutputStream outputStream = response.getOutputStream();
    FileInputStream fis = null;
```

```
    try {
        fis = new FileInputStream(file);
        FileCopyUtils.copy(fis, outputStream);
    } finally {
        if (fis != null) {
            try {
                fis.close();
            } catch (IOException e) {
            }
        }
    }

    out.flush();
%>
```

3.9 UploadController 컨트롤러 작성

```
import java.io.*;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.*;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpHeaders;
import org.springframework.http.HttpStatus;
import org.springframework.http.MediaType;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.multipart.MultipartFile;

@Controller
public class UploadController  {

    private static final Logger logger = LoggerFactory.getLogger(UploadController.class);

    @Autowired
      private IServiceUploadFile svrupload;


    /**
     * http://localhost/upload/uploadfile
     */
    @RequestMapping(value = "/upload/uploadfile", method = RequestMethod.GET)
    public String uploadfile(Model model)  {

        logger.info("uploadfile");

        return "upload/uploadfile";
    }
```

```
    /**
     * http://localhost/upload/uploadone
     */
    @RequestMapping(value = "/upload/uploadfileone", method = RequestMethod.POST)
    public String uploadfileone(Model model
            , @RequestParam(value="file") MultipartFile file
            , @RequestParam(value="upDir") String upDir
            ) throws IllegalStateException, IOException  {

        logger.info("uploadfile");

        // 폴더 존재 여부 검사
        java.io.File dir = new java.io.File(upDir );
        if (!dir.exists() && !dir.isDirectory() )  dir.mkdir();


        // 로컬 파일을 서버로 올리기 위한 코드
        String fileName = file.getOriginalFilename();
        String tempName =
LocalDateTime.now().format(DateTimeFormatter.ofPattern("yyyyMMddHHmmss"));
        String filepath = upDir + "/" + tempName;
        java.io.File f   = new java.io.File( filepath );
        file.transferTo( f );

        // DB insert 처리를 위한 코드
        ModelUploadFile attachfile = new ModelUploadFile();
        attachfile.setFileNameOrig( fileName );
        attachfile.setFileNameTemp( f.getName() );
        attachfile.setFileSize( (Long)f.length() );
        attachfile.setContentType( file.getContentType() ); // 확장자

        // DB insert
        svrupload.insertUploadFile(attachfile);


        // uploadsuccess.jsp 에 출력할 파일이름  저장
        List<String> filelist = new ArrayList<String>();
        filelist.add(fileName);
        model.addAttribute("files", filelist);
        model.addAttribute("upDir", upDir);

        model.asMap().clear(); // Redirect without parameters being added to my url
        return "redirect:uploadfilelist";
    }
```

```
    /**
     * http://localhost/upload/uploadfilemulti
     */
    @RequestMapping(value = "/upload/uploadfilemulti", method = RequestMethod.POST)
    public String uploadfilemulti( Model model
            , @ModelAttribute RepositoryFiles uploadForm
            , @RequestParam(value="upDir") String upDir ) throws IllegalStateException,
IOException  {

        logger.info("uploadfilemulti");

        // 폴더 존재 여부 검사
        java.io.File dir = new java.io.File( upDir );
        if (!dir.exists() && !dir.isDirectory() )  dir.mkdir();

        List<MultipartFile> files = uploadForm.getFiles();
        if( files != null && files.size() > 0 ){

            for (MultipartFile file : files) {

                // 클라이언트의 첨부 파일을 서버로 복사
                String fileName = file.getOriginalFilename();
                String tempName =
LocalDateTime.now().format( DateTimeFormatter.ofPattern("yyyyMMddHHmmss") );
                String filepath = upDir + "/" + tempName;
                java.io.File f   = new java.io.File( filepath );
                file.transferTo( f );

                // DB insert 처리를 위한 코드
                ModelUploadFile attachfile = new ModelUploadFile();
                attachfile.setFileNameOrig( fileName );
                attachfile.setFileNameTemp( f.getName() );
                attachfile.setFileSize( (Long)f.length() );
                attachfile.setContentType( file.getContentType() ); // 확장자

                // DB insert
                svrupload.insertUploadFile(attachfile);
            }
        }


        // uploadsuccess.jsp 에 출력할 파일이름 저장
        List<String> filelist = new ArrayList<String>();
        for (MultipartFile file : files) {
            filelist.add( file.getOriginalFilename() );
```

```
        }
        model.addAttribute("files", filelist);
        model.addAttribute("upDir", upDir);


        model.asMap().clear(); // Redirect without parameters being added to my url
        return "redirect:uploadfilelist";
    }


    /**
     * http://localhost/upload/uploadfilelist
     */
      @RequestMapping(value="/upload/uploadfilelist", method=RequestMethod.GET)
      public String uploadfilelist(Model model) {

        logger.info("uploadfilelist");

        ModelUploadFile file = null;
        List<ModelUploadFile> files = svrupload.getUploadFile(file);
        model.addAttribute("files", files);

        return "upload/uploadfilelist";
      }


    /**
     * http://localhost/download
     */
      @RequestMapping(value="/download", method=RequestMethod.POST)
      public String download(Model model
                  , @RequestParam String filenametemp
                  , @RequestParam String filenameorig ) {

            logger.info("download");

            model.addAttribute( "filenametemp", filenametemp);
            model.addAttribute( "filenameorig", filenameorig);

            return "upload/download";

      }

}
```

4. 테이블 컬럼에 이미지 저장하기

이미지를 DB 테이블에 저장하는 방법에는 크게 2 가지가 존재한다.

1. BLOB 컬럼에 바이너리 형태로 저장하는 방식
2. CLOB(또는 TEXT) 컬럼에 문자열 형태로 저장하는 방식

지금부터 이미지 파일을 LOB 형식의 컬럼에 등록을 해보는 방법과 LOB 형식의 데이터를 이미지태그에 출력하는 방법을 해보도록 하겠습니다.

4.1 이미지 저장할 테이블 생성

4.1.1 oracle ddl 스크립트

```
CREATE TABLE TB_Upload_Image (
      uploadImageNo  NUMBER(10)     generated as identity

    , fileName       VARCHAR2(50)   NOT NULL
    , fileSize       NUMBER(10)     NOT NULL
    , contentType    VARCHAR(50)    NOT NULL

    , imageBytes     BLOB  -- 사진 저장 컬럼. 바이너리로 이미지 저장
    , imageBase64    CLOB  -- 사진 저장 컬럼. BASE64 로 이미지 저장

    , PRIMARY KEY(uploadImageNo)
);
```

4.1.2 mysql ddl 스크립트

```
DROP TABLE IF EXISTS TB_Upload_Image;
CREATE TABLE TB_Upload_Image (
      uploadImageNo  INT UNSIGNED   NOT NULL AUTO_INCREMENT
    , fileName       NVARCHAR(50)   NOT NULL
    , fileSize       INT UNSIGNED   NOT NULL
    , contentType    NVARCHAR(30)   NOT NULL

    , imageBytes     LONGBLOB  -- 사진 저장 컬럼. 바이너리로 이미지 저장
    , imageBase64    LONGTEXT  -- 사진 저장 컬럼. BASE64 로 이미지 저장

    , PRIMARY KEY(uploadImageNo)
)
ENGINE=InnoDB
AUTO_INCREMENT=1
DEFAULT CHARACTER SET utf8
COLLATE utf8_general_ci
;
```

4.2 모델 클래스 생성

```
public class ModelUploadImage {

    private Integer uploadImageNo   ;
    private String  fileName       ;
    private Long    fileSize       ;
    private String  contentType    ;
    private byte[]  imageBytes     ; // BLOB 컬럼
    private String  imageBase64    ;  // CLOB(TEXT) 컬럼

    private CommonsMultipartFile image;


    // getter & setter 만들기

    // 기본 생성자 만들기

    // toString() 만들기

}
```

4.3 mapperUpload.xml 파일 생성(Mybatis mapper)

4.3.1 oracle용

```
<mapper namespace="mapper.mapperUpload">

    <select id="getImageByte" parameterType="int" resultType="ModelUploadImage">
        SELECT uploadImageNo, fileName, contentType, fileSize, imageBytes, imageBase64
          FROM TB_Upload_Image
         WHERE uploadImageNo = #{uploadImageNo}
    </select>

    <insert id="insertPhoto" parameterType="map" statementType="CALLABLE">
        declare
            s2  number := 0;
        begin
            INSERT INTO
            TB_Upload_Image( fileName, fileSize, contentType, imageBytes, imageBase64 )
            VALUES( #{file.fileName}, #{file.fileSize}, #{file.contentType},
#{file.imageBytes}, #{file.imageBase64} )
            RETURNING uploadImageNo INTO s2;

            #{result, jdbcType=INTEGER, mode=OUT} := s2;
        end;
    </insert>

    <update id="updateAttachImage" parameterType="ModelUploadImage">
        update TB_Upload_Image
        <set>
            <if test="photo.originalFilename != null">
fileName     = #{photo.originalFilename},
</if>
            <if test="photo.ContentType      != null">
contentType  = #{photo.contentType}     ,
</if>
            <if test="photo.bytes            != null">
imageBytes   = #{photo.bytes}           ,
</if>
            <if test="photo.bytes            != null">
imageBytes   = #{photo.bytes}           ,
</if>
        </set>
        where uploadImageNo = #{uploadImageNo}
    </update>

</mapper>
```

4.3.2 mysql 용

```
<mapper namespace="mapper.mapperUpload">

    <select id="getImageByte" parameterType="int" resultType="ModelUploadImage">
        SELECT uploadImageNo, fileName, contentType, fileSize, imageBytes, imageBase64
          FROM TB_Upload_Image
         WHERE uploadImageNo = #{uploadImageNo}
    </select>

<insert id="insertPhoto" parameterType="ModelUploadImage"
useGeneratedKeys="true" keyProperty="uploadImageNo">
        INSERT INTO
        TB_Upload_Image(fileName, fileSize, contentType,imageBytes,imageBase64)
        VALUES( #{fileName}, #{fileSize}, #{contentType}, #{imageBytes}, #{imageBase64} )
    </insert>

    <update id="updateAttachImage" parameterType="ModelUploadImage">
        update TB_Upload_Image
        <set>
            <if test="photo.originalFilename != null">
fileName     = #{photo.originalFilename},
</if>
            <if test="photo.ContentType      != null">
contentType  = #{photo.contentType}     ,
</if>
            <if test="photo.bytes            != null">
imageBytes   = #{photo.bytes}           ,
</if>
            <if test="photo.bytes            != null">
imageBytes   = #{photo.bytes}           ,
</if>
        </set>
        where uploadImageNo = #{uploadImageNo}
    </update>

</mapper>
```

4.4 Dao 만들기

4.4.1 IDaoUploadImage 만들기

```
public interface IDaoUploadImage {
```

```
    ModelUploadImage getImageByte(int attachfileno);
    int insertPhoto(ModelUploadImage attachfile);
}
```

4.4.2 DaoUploadFile 만들기

```
import java.util.List;

import org.apache.ibatis.session.SqlSession;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Repository;

@Repository
public class DaoUploadImage implements IDaoUploadImage {

    @Autowired
    @Qualifier("sqlSession")
    private SqlSession session;

    public int insertPhoto(ModelUploadImage imageinfo) {

       /* // Oracle  인 경우
        Map<String, Object> map = new  HashMap<String, Object>();
        map.put("fileinfo"  , imageinfo);
        map.put("result", null);

        session.insert("mapper.mapperUpload.insertPhoto", map);
        int result = map.get("result") != null ? (int) map.get("result") : -1;

        return result;
        */

        // mysql 인 경우
        session.insert("mapper.mapperUpload.insertPhoto", imageinfo );
        return imageinfo.getUploadImageNo();
    }

    public ModelUploadImage getImageByte(int attachfileno) {
        return session.selectOne("mapper.mapperUpload.getImageByte", attachfileno);
    }
}
```

4.5 Service 만들기

4.5.1 IServiceUploadImage 만들기

```
public interface IServiceUploadImage extends IDaoUploadImage {
}
```

4.5.2 ServiceUploadImage 만들기

```
import java.util.List;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
public class ServiceUploadImage implements IServiceUploadImage {

    // SLF4J Logging
    private Logger logger = LoggerFactory.getLogger(this.getClass());

    @Autowired
    private IDaoUploadImage uploaddao;

    @Override
    public int insertPhoto(ModelUploadImage attachfile) {

        int result = -1;
        try {
            result = uploaddao.insertPhoto(attachfile);
        } catch (Exception e) {
            logger.error("insertPhoto " + e.getMessage() );
        }
        return result;
    }

    @Override
    public ModelUploadImage getImageByte(int attachfileno) {

        ModelUploadImage result = null;
        try {
            result = uploaddao.getImageByte(attachfileno);
        } catch (Exception e) {
            logger.error("getImageByte " + e.getMessage() );
```

```
        }
        return result;
    }

}
```

4.6 테스트 코드 만들기

```
import static org.junit.Assert.*;

import javax.imageio.ImageIO;

import org.apache.commons.io.FilenameUtils;
import org.junit.BeforeClass;
import org.junit.Test;

import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;

import java.awt.image.BufferedImage;
import java.io.*;
import java.nio.file.Files;
import java.util.Base64;

public class TestUploadImage {

    private static ApplicationContext context = null;
    private static IServiceUploadFile service = null;


    @BeforeClass
    public static void setUpBeforeClass() throws Exception {

        context= new ClassPathXmlApplicationContext("file:src/main/webapp/WEB-
INF/spring/appServlet/servlet-context.xml");
        service=context.getBean(IServiceUploadFile.class);
    }

    @Test
    public void test_insertPhoto() {

        // convert image to byte

        try {
```

```
            File file = new File("src/test/resources/image.png");
            BufferedImage originalImage = ImageIO.read(file.getAbsoluteFile());

            ByteArrayOutputStream baos = new ByteArrayOutputStream();
            ImageIO.write(originalImage, "jpg", baos);
            baos.flush();
            byte[] photoBytes = baos.toByteArray();

            ModelUploadImage fileupload = new ModelUploadImage();
            fileupload.setFileName( file.getName() );
            fileupload.setFileSize( file.length() );
            fileupload.setContentType( Files.probeContentType( file.toPath() ) );
            fileupload.setImageBytes(photoBytes);

fileupload.setImageBase64( Base64.getEncoder().encodeToString(baos.toByteArray()) );

            int result = service.insertPhoto(fileupload);

            assertNotSame(-1, result);

        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    @Test
    public void test_getImageByte() {

        // convert byte to image

        try {

            int attachfileno = 1;
            ModelUploadImage result = service.getImageByte(attachfileno);

            // convert byte array back to BufferedImage
            InputStream in = new ByteArrayInputStream( result.getImageBytes() );
            BufferedImage bImageFromConvert = ImageIO.read(in);

            File file = new File( "c:\\" + result.getFileName() );

            File dir = new File(file.getAbsolutePath());
            if (!dir.exists()) {
                dir.mkdir();
            }
```

```
            ImageIO.write(bImageFromConvert, FilenameUtils.getExtension(file.getName()),
file );

            assertSame(1, result.getUploadImageNo() );

        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

4.7 UploadContoller 만들기

```
import java.io.*;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.*;

import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpHeaders;
import org.springframework.http.HttpStatus;
import org.springframework.http.MediaType;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.multipart.MultipartFile;

@Controller
public class UploadController  {

    private static final Logger logger = LoggerFactory.getLogger(UploadController.class);

    @Autowired
    private IServiceUploadFile svrupload;
```

```
    /**
     * 사진 업로드를 위한 화면
     */
    @RequestMapping(value="/upload/imageupload", method = RequestMethod.GET)
    public String imageupload () {
        return "upload/imageupload";
    }

    /**
     * 사진 파일 업로드 후 DB 저장
     */
    @RequestMapping(value="/upload/imageupload", method = RequestMethod.POST)
    public String imageupload (Model model
            , @RequestParam String upDir
            , @ModelAttribute ModelUploadImage vo ) {

        logger.info("imageupload");

        Integer attachfileno = null;

        try {
            vo.setFileName( vo.getImage().getOriginalFilename() );
            vo.setFileSize( (Long)vo.getImage().getSize() );
            vo.setContentType( vo.getImage().getContentType() ); // 확장자
            vo.setImageBytes( vo.getImage().getBytes() );

vo.setImageBase64( Base64.getEncoder().encodeToString( vo.getImage().getBytes() ) );

            attachfileno = svrupload.insertPhoto(vo);

        } catch (Exception e) {
            e.printStackTrace();
        }

        return "redirect:/upload/imageview/"+ Integer.toString( attachfileno );
    }

    /**
     * 임의의 뷰페이지
     */
    @RequestMapping(value="/upload/imageview/{attachfileno}", method = RequestMethod.GET)
    public String imageview(Model model , @PathVariable int attachfileno) {

        logger.info("imageview");

        ModelUploadImage result = svrupload.getImageByte(attachfileno);
```

```
        model.addAttribute("attachfileno", attachfileno);
        model.addAttribute("contentType", result.getContentType() );
        model.addAttribute("imageBase64", result.getImageBase64() );

        return "upload/imageview";
    }

    /** img 태그의 src 에 이미지를 출력하기 위한 메서드 */
    @RequestMapping(value="/upload/getphoto/{attachfileno}", method = RequestMethod.GET)
    public ResponseEntity<byte[]> getImageByte(@PathVariable int attachfileno) {

        logger.info("getImageByte");

        ModelUploadImage result = svrupload.getImageByte(attachfileno);

       byte[] imageContent =  result.getImageBytes();
       final HttpHeaders headers = new HttpHeaders();
       headers.setContentType( MediaType.valueOf( result.getContentType() ) );

       return new ResponseEntity<byte[]>(imageContent, headers, HttpStatus.OK);
    }
}
```

4.8 jsp 파일 만들기

4.8.1 DB 에 저장할 파일처리를 위한 뷰페이지(imageupload.jsp)

```
<%@ page session="false" %>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>

<html>
<head>
    <meta charset="utf-8" />
      <title> 테이블 컬럼에 이미지 저장하는 예제 </title>
</head>
<body>
    <form action="/upload/imageupload" enctype="multipart/form-data" method="post">

        Upload Directory :
        <input type="text" name="upDir" value="c:/upload/" />
        <br />
```

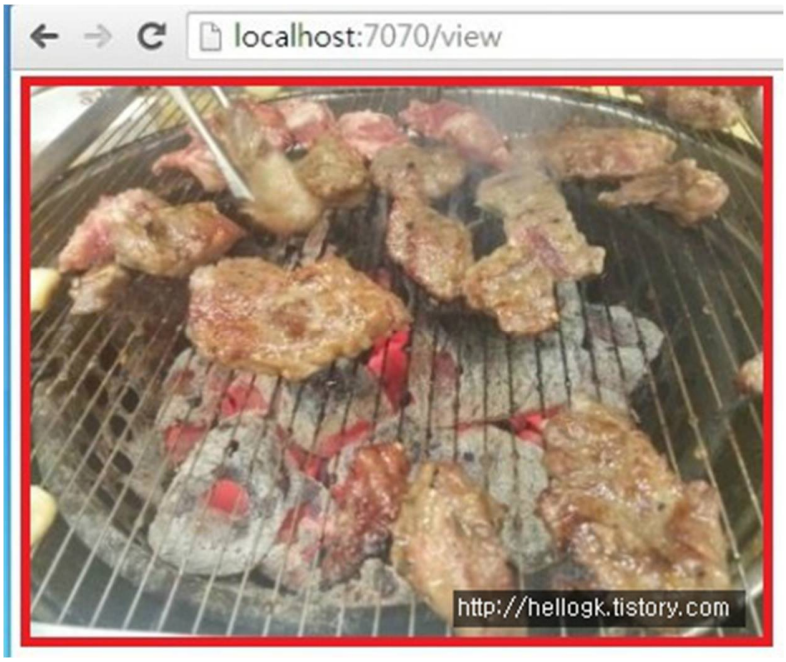
```
        <br />
        <input type="file" name="image" />
        <br />
        <input type="submit" value="이미지저장"/>
    </form>
</body>
</html>
```

4.8.2 뷰페이지(imageview.jsp)

```
<%@ page session="false" %>

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>
<html>
<head>
    <meta charset="utf-8" />
      <title>helloworld</title>
</head>
<body>
    <!-- img 태그의 src 경로는 byte 이미지 가져오는 컨트롤러 호출(/getImageByte) -->
    <img alt="" src="/upload/getphoto/${attachfileno}" />
    <hr />
    <img src="data:${contentType};base64,${imageBase64}" />
</body>
</html>
```

임의로 body 태그내에 작성 후 /view 컨트롤러를 호출해보도록 하겠습니다.



호출결과 정상적으로 byte 데이터를 DB 로부터 가져와서 화면에 출력이 되었습니다.

5. Reference

http://aircook.tistory.com/entry/Spring-Ibatis-프레임워크-구성시-오라클-LOB-타입-사용하기

https://anirbanchowdhury.wordpress.com/2012/05/15/mybatis-upload-retrieve-download-file-or-image/

http://www.databasesql.info/article/534369750/

http://hellogk.tistory.com/129

http://stackoverflow.com/questions/12059872/how-to-select-a-blob-column-from-database-using-ibatis

http://stackoverflow.com/questions/17055558/select-a-blob-column-from-oracle-db-using-mybatis

http://viralpatel.net/blogs/tutorial-save-get-blob-object-spring-3-mvc-hibernate/

http://javahonk.com/save-and-retrieve-image-from-database/

http://forum.spring.io/forum/spring-projects/web/52397-multipart-fileupload-problem-in-fileuploadaction-junit-testing

http://stackoverflow.com/questions/8799378/how-to-write-unit-test-for-commonsmultipartfile-with-mock-in-spring

http://stackoverflow.com/questions/7879620/how-to-unit-test-file-uploads-with-mockhttpservletrequest

convert byte array to base64 string in java

http://stackoverflow.com/questions/2418485/how-do-i-convert-a-byte-array-to-base64-in-java