xexcel5 사용 가이드

Last update: 2025/06/02

www.softbase.co.kr
Copyright © SOFTBase Inc. All rights reserved.

목차

xexcel5 사용 가이드 1
lib 구성 기본 정보 1
lib 지원기능 5
설정방법 5
xFrame5 설정 5
web.xml 설정 6
Lib 복사 6
Excel API 7
그리드 » Api 7
팩토리 » Api 11
XLoggerManager 13
Package 구성 13
설정방법 14
Log4j 설정 정보 15
Logger 설정 동적 변경 방법 16
DRM 파일처리 16
엑셀 업로드시 DRM 처리 class 파라미터 지정 방식 16
엑셀 다운로드시 DRM 처리 class 파라미터 지정 방식 18
엑셀 DRM 처리시 추가 파라미터 전달하는 경우 19
텍스트 및 CSV 파일 로드시 DRM 처리 class 파라미터 지정 방식 26
텍스트 및 CSV 파일 다운로드시 DRM 처리 class 파라미터 지정 방식 29
배경 이미지 설정 32
엑셀 다운로드시 워터마크 이미지 설정 32

xexcel5 사용 가이드

본 장에서는 xFrame5 에서 제공하는 Excel 다운로드, 업로드 기능을 수행하는 java lib 에 대한 내용을 설명합니다.

- lib구성 기본 정보
- lib 지원기능
- 엑셀 가이드 pdf

lib 구성 기본 정보

xFrame5 에서 기본으로 제공하는 Excel 관련 기능은 java 라이브러리 파일인 jar 파일 형태로 제공한다.

lib 제공위치는 xFrame5 홈 디렉터리 아래 jar 디렉터리 안에 존재한다.

제공하는 Excel 관련 라이브러리는 아래와 같다.

구분	xExcel lib	xDataset lib
3.14 버전	• xexcel5-3.14.jar • poi-3.14.jar • poi-ooxml-3.14.jar • poi-ooxml-schemas-3.14.jar • poi-scratchpad-3.14.jar • commons-beanutils-1.8.3.jar • commons-codec-1.10.jar • commons-fileupload-1.2.1.jar • commons-io-1.4.jar • commons-logging-1.2.jar • curvesapi-1.03.jar • hamcrest-core-1.1.jar • log4j-core-2.3.2.jar • log4j-api-2.3.2.jar • log4j-1.2-api-2.3.2.jar • stax-api-1.0.1.jar • xmlbeans-2.6.0.jar	• xdataset5.jar • json-simple-1.1.1.jar

구분	xExcel lib	xDataset lib
3.16 버전	• xexcel5-3.16.jar • poi-3.16.jar • poi-ooxml-3.16.jar • poi-ooxml-schemas-3.16.jar • poi-scratchpad-3.16.jar • commons-beanutils-1.8.3.jar • commons-codec-1.10.jar • commons-collections4-4.1.jar • commons-fileupload-1.2.1.jar • commons-io-1.4.jar • commons-logging-1.2.jar • curvesapi-1.04.jar • hamcrest-core-1.1.jar • log4j-core-2.3.2.jar • log4j-api-2.3.2.jar • log4j-1.2-api-2.3.2.jar • stax-api-1.0.1.jar • xmlbeans-2.6.0.jar	

구분	xExcel lib	xDataset lib
4.1.2 버전	• xexcel5_412.jar • poi-4.1.2.jar • poi-examples-4.1.2.jar • poi-excelant-4.1.2.jar • poi-ooxml-4.1.2.jar • poi-ooxml-schemas-4.1.2.jar • poi-scratchpad-4.1.2.jar • activation-1.1.1.jar • commons-codec-1.13.jar • commons-collections4-4.4.jar • commons-compress-1.19.jar • commons-logging-1.2.jar • commons-math3-3.6.1.jar • jaxb-api-2.3.1.jar • jaxb-core-2.3.0.1.jar • jaxb-impl-2.3.2.jar • junit-4.12.jar • log4j-core-2.17.2.jar • log4j-api-2.17.2.jar • log4j-1.2-api-2.17.2.jar • SparseBitSet-1.2.jar • curvesapi-1.06.jar • xmlbeans-3.1.0.jar • commons-io-1.4commons-fileupload-1.2.1	

구분	xExcel lib	xDataset lib
5.2.2 버전	• xexcel5_522.jar • batik 관련.jar • bcpkix-jdk15on-1.70.jar • bcprov-jdk15on-1.70.jar • bcutil-jdk15on-1.70.jar • commons-codec-1.15.jar • commons-collections4-4.4.jar • commons-io-2.11.0.jar • commons-math3-3.6.1.jar • commons-compress-1.21.jar • commons-fileupload-1.2.1.jar • commons-logging-1.2.jar • curvesapi-1.07.jar • fontbox-2.0.25.jar • graphics2d-0.35.jar • poi-5.2.2.jar • poi-examples-5.2.2.jar • poi-excelant-5.2.2.jar • poi-javadoc-5.2.2.jar • poi-ooxml-5.2.2.jar • poi-ooxml-full-5.2.2.jar • poi-ooxml-lite-5.2.2.jar • poi-scratchpad-5.2.2.jar • slf4j-api-1.7.36.jar • SparseBitSet-1.2.jar • xmlbeans-5.0.3.jar • pdfbox-2.0.25.jar • xml-apis-1.4.01.jar • xml-apis-ext-1.3.04.jar • xmlgraphics-commons-2.6.jar • xmlsec-2.3.0.jar • log4j-core-2.17.2.jar • log4j-api-2.17.2.jar • log4j-1.2-api-2.17.2.jar	

- XExcel5.jar 라이브러리는 Excel 97 – 2003 통합문서 .xls 와 Excel 통합문서 .xlsx 파일을 모두 지원합니다.
- XExcel5.jar 사용하기 위해서는 JDK 1.8 이상의 버전이 필요합니다.
- POI 3.17 또는 4.X 버전의 xexcel5.jar 파일은 별도로 요청 하시길 바랍니다.
- Apache poi lib관련 자세한 내용은 https://poi.apache.org/ 를 참고하세요.
- Log4j 버전별 필요한 Java 버전은 Apache Log4j 공식 웹사이트의 릴리스 노트(https://logging.apache.org/log4j/2.x/release-notes.html)에서 확인하실 수 있습니다.

lib 지원기능

구분	XLS	XLSX
EXCEL 불러오기, 저장	O	O
CSV 불러오기, 저장	O	O
EXCEL 다운로드 최대 행 수	65,536	1,048,576
엑셀다운로드 옵션 : 파일 암호지정, 파일이름 지정, 수평헤더 제목설정, 패턴적용, 통계 데이터적용 설정, 경계선적용, 정렬, 폰트, 배경색, 전경색, 아이템별 스타일 지원, 데이터 컬럼 지정 다운로드		

설정방법

본 장에서는 excel 관련 기능을 사용하기 위한 설정방법에 대한 내용을 정리합니다.

- xFrame5 설정
- web.xml 설정
- lib 복사

xFrame5 설정

xFrame5 Engine 실행 환경 정보를 설정하는 xframe5.js 파일에 아래의 내용을 설정한다.

```
// 설정확인
GRID_EXCELDOWNLOAD_ITEMSTYLE: true,
GRID_EXCELDOWNLOAD_FORECOLOR: true,

// 설정확인
XEXCEL_UPLOAD_URL: 'http://127.0.0.1:8080/xexcel5/XExcelUpload',
XEXCEL_DOWNLOAD_URL: 'http://127.0.0.1:8080/xexcel5/XExcelDownload',
```

프로퍼티	비고
XEXCEL_UPLOAD_URL	엑셀 업로드 URL • ex) XEXCEL_UPLOAD_URL: 'http://127.0.0.1:8080/xexcel5/XExcelUpload'
XEXCEL_DOWNLOAD_URL	엑셀 다운로드 URL • ex) XEXCEL_DOWNLOAD_URL: 'http://127.0.0.1:8080/xexcel5/XExcelDownload'

프로퍼티	비고
GRID_EXCELDOWNLOAD_ITEMSTYLE	엑셀다운로드 시 아이템 별 스타일 적용여부 표시옵션 • ex) true
GRID_EXCELDOWNLOAD_FORECOLOR	엑셀다운로드 시 폰트 색상 적용여부 표시옵션. • ex) true)

web.xml 설정

엑셀 업, 다운로드기능을 처리할 Web 프로젝트 web.xml 파일에 아래의 내용을 추가한다.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:web="http://xmlns.jcp.org/xml/ns/javaee"
    xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
    http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd
http://xmlns.jcp.org/xml/ns/javaee
    http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd" id="xadmin"
version="2.4">

  <!-- 추가 -->
    <servlet>
        <servlet-name>XExcelDownload</servlet-name>
        <servlet-class>xexcel5.servlet.XExcelDownload</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>XExcelDownload</servlet-name>
        <url-pattern>/XExcelDownload</url-pattern>
    </servlet-mapping>
    <servlet>
        <servlet-name>XExcelUpload</servlet-name>
        <servlet-class>xexcel5.servlet.XExcelUpload</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>XExcelUpload</servlet-name>
        <url-pattern>/XExcelUpload</url-pattern>
    </servlet-mapping>
  <!-- 추가 종료 -->

</web-app>
```

Lib 복사

적용할 xexcel5 lib 파일을 프로젝트의 WEB-INF/lib 디렉토리에 복사한다.

Excel API

본 장에서는 xFrame5 에서 지원하는 API 에 대한 내용을 정리합니다.

- 그리드 API
- 팩토리 API

그리드 » Api

xFrame5 GRID 오브젝트에서 사용하는 API 는 다음과 같습니다.

downloadexcel()

그리드의 내용을 엑셀 파일로 다운로드 한다.

Parameters

Parameters	Type	Description
sFileName	STRING	엑셀 파일 이름

Return Value

Type	Description
void	

Example

```
function btn_downloadexcel_on_mouseup(objInst)
{
    grdList.downloadexcel("download.xlsx");
}
```

downloadexcelex()

그리드의 내용을 엑셀 파일로 다운로드 한다.

Parameters

Parameters	Type	Description
sFileName	STRING	엑셀 파일 이름
bShowOption	BOOL	옵션창 표시 여부
bSaveOneRow	BOOL	한줄로 저장 여부

Parameters	Type	Description
bIncPattern	BOOL	패턴 포함 여부
bIncHeader	BOOL	헤더 포함 여부
bIncStatData	BOOL	통계 데이터 포함 여부
bIncBorder	BOOL	테두리 포함 여부
bIncAlign	BOOL	정렬 포함 여부
bIncBackColor	BOOL	배경색 포함 여부
bIncFont	BOOL	폰트 포함 여부
bIncForeColor	BOOL	[옵션] 전경색 포함 여부 (기본값: false)
bIncItemStyle	BOOL	[옵션] 아이템별 스타일 적용 여부 (기본값: false)
bIncItemMerge	BOOL	[옵션] 아이템별 병합 적용 여부 (기본값: false)
bIncLineNumber	BOOL	[옵션] 순번 데이터 포함 여부 (기본값: false)
bIncMasking	BOOL	[옵션] 순번 데이터 포함 여부 (기본값: false)
nFileDownType	short	파일 다운로드 유형 (기본값: 0)
nConvNumberType	short	[옵션] 숫자 변환 타입 (기본값: 0)
bCheckRowOnly	BOOL	[옵션] 체크된 행만 저장 여부 (기본값: false)

Return Value

Type	Description
void	

Example

```
function btn_downloadexcel_on_mouseup(objInst)
{
    grdList.downloadexcelex("download.xlsx", true, false, true, true, true,
true, true, false,false);
}
```

nFileDownType 파라미터

nFileDownType 파라미터를 1 로 설정하면 엑셀 파일을 서버에만 저장하고 사용자 PC 에는 저장하지 않을 수 있다. 이때 엑셀 파일의 기본 저장 경로는 XExcelDownload 서블릿이 실행되는 Context Root 하위의 temp 디렉토리이다.

엑셀 파일 저장 경로는 web.xml 에 TEMP_DIR 파라미터를 설정하여 변경 할 수 있다. TEMP_DIR 을 설정하면 TEMP_DIR 파라미터값 하위의 temp 디렉토리에 엑셀 파일이 저장된다. 예를 들어 다음과 같이 설정시 C 드라이브 > temp 디렉토리 하위로 저장된다.

```
<servlet>
    <servlet-name>XExcelDownload</servlet-name>
    <servlet-class>xexcel5.servlet.XExcelDownload</servlet-class>
     <!-- 추가 -->
```

```
        <init-param>
            <param-name>TEMP_DIR</param-name>
            <param-value>C:\</param-value>
        </init-param>
     <!-- 추가 종료 -->
</servlet>
<servlet-mapping>
    <servlet-name>XExcelDownload</servlet-name>
    <url-pattern>/XExcelDownload</url-pattern>
</servlet-mapping>
```

---

uploadexcel()

엑셀 파일 내용을 그리드에 로드한다.

parameters

Type	Type	Description
bAppendMode	BOOL	[옵션] 데이터 Append 모드 여부(기본값: false)

Return Value

Type	Description
void	

Example

```
function btn_loadexcel_on_mouseup(objInst)
{
    grdList.uploadexcel();
}
```

uploadexcelex()

엑셀 파일 내용을 그리드에 로드한다. **uploadexcel API 와 다른 점은 다이얼로그의 옵션을 파라미터로 셋팅할 수 있는 점이다.**

Parameters

Parameters	Type	Description
nStartSheetNo	short	시작 시트 번호 (1 부터 시작)
nEndSheetNo	short	끝 시트 번호
nStartRowNo	long	시작 행 번호 (1 부터 시작)

Parameters	Type	Description
nExcelRowCountPerGridRow	STRING	시작 열 이름 (A 부터 시작)
nExcelRowCountPerGridRow	short	그리드 한행에 대한 엑셀 행 갯수
bAppendMode	BOOL	[옵션] 데이터 Append 모드 (기본값: false)

Return Value

Type	Description
void	

Example

```
function btn_loadexcelex_on_mouseup(objInst)
{
    grdList.uploadexcelex(1, 1, 1, "A", 1);
}
```

*** TEMP_DIR 파라미터**

엑셀을 서버에 업로드 할 시 서버에 엑셀 처리에 사용되는 임시 폴더가 생성되며, 임시 폴더 위치는 web.xml 에 TEMP_DIR 파라미터를 설정하여 변경할 수 있다.

TEMP_DIR 파라미터를 설정하면 TEMP_DIR 의 하위 경로에 temp 및 upload 디렉토리가 생성되어 임시파일이 저장되게 된다.

예를들어 다음과 같이 설정시 경로가 C 드라이브 > excel_test 디렉토리로 변경된다.

임시파일은 업로드 작업 완료 후 삭제된다.

```
<servlet>
    <servlet-name>XExcelUpload</servlet-name>
    <servlet-class>xexcel5.servlet.XExcelUpload</servlet-class>
     <!-- 추가 -->
        <init-param>
            <param-name>TEMP_DIR</param-name>
            <param-value>C:\excel_test</param-value>
        </init-param>
     <!-- 추가 종료 -->
</servlet>
<servlet-mapping>
    <servlet-name>XExcelUpload</servlet-name>
    <url-pattern>/XExcelUpload</url-pattern>
</servlet-mapping>
```

*** CORS_HEADER 파라미터**

엑셀을 서버에 업로드 할 시 응답 헤더에 기본적으로 추가되는 CORS(Cross-Origin Resource Sharing) 관련 헤더값을 추가하지 않는 경우 설정하는 파라미터이다.

false 로 설정할 시 Access-Control-Allow-Credentials, Access-Control-Allow-Headers, Access-Control-Allow-Origin 이 설정되지 않는다.

```
<servlet>
    <servlet-name>XExcelUpload</servlet-name>
    <servlet-class>xexcel5.servlet.XExcelUpload</servlet-class>
     <!-- 추가 -->
        <init-param>
            <param-name>CORS_HEADER</param-name>
            <param-value>false</param-value>
        </init-param>
     <!-- 추가 종료 -->
</servlet>
<servlet-mapping>
    <servlet-name>XExcelUpload</servlet-name>
    <url-pattern>/XExcelUpload</url-pattern>
</servlet-mapping>
```

팩토리 » Api

setexceldownloadurl()

exceldownload 함수 실행시 호출될 URL 를 설정한다.

Parameters

Parameters	Type	Description
url	STRING	exceldownload 함수 실행시 호출될 URL

Return Value

Type	Description
void	

Example

```
function btn_setexceldownloadurl_on_mouseup(objInst)
{
    var sExcelDownloadUrl =
"http://127.0.0.1:8080/xexcel5_app/XExcelDownload";
    factory.setexceldownloadurl(sExcelDownloadUrl);
}
```

setexceluploadurl()

excelupload 함수 실행시 호출될 URL를 설정한다.

Parameters

Parameters	Type	Description
url	STRING	exceldownload 함수 실행시 호출될 URL)

Return Value

Type	Description
void	

Example

```
function btn_setexceluploadurl_on_mouseup(objInst)
{
    var sExcelUploadUrl = "http://127.0.0.1:8080/xexcel5_app/XExcelUpload";
    factory.setexceluploadurl(sExcelUploadUrl);
}
```

getexceldownloadurl()

exceldownload 함수 실행시 호출될 URL 를 리턴한다.

Return Value

Type	Description
STRING	exceldownload 함수 실행시 호출될 URL

Example

```
function btn_getexceldownloadurl_on_mouseup(objInst)
{
    var sExcelDownloadUrl;
    sExcelDownloadUrl = factory.getexceldownloadurl();
    factory.consoleprint("sExcelDownloadUrl = " + sExcelDownloadUrl);
}
```

getexceluploadurl()

excelupload 함수 실행시 호출될 URL 를 리턴한다.

Return Value

Type	Description
STRING	excelupload 함수 실행시 호출될 URL

Example

```
function btn_getexceluploadurl_on_mouseup(objInst)
{
    var sExcelUploadUrl;
    sExcelUploadUrl = factory.getexceluploadurl();
    factory.consoleprint("sExcelUploadUrl = " + sExcelUploadUrl);
}
```

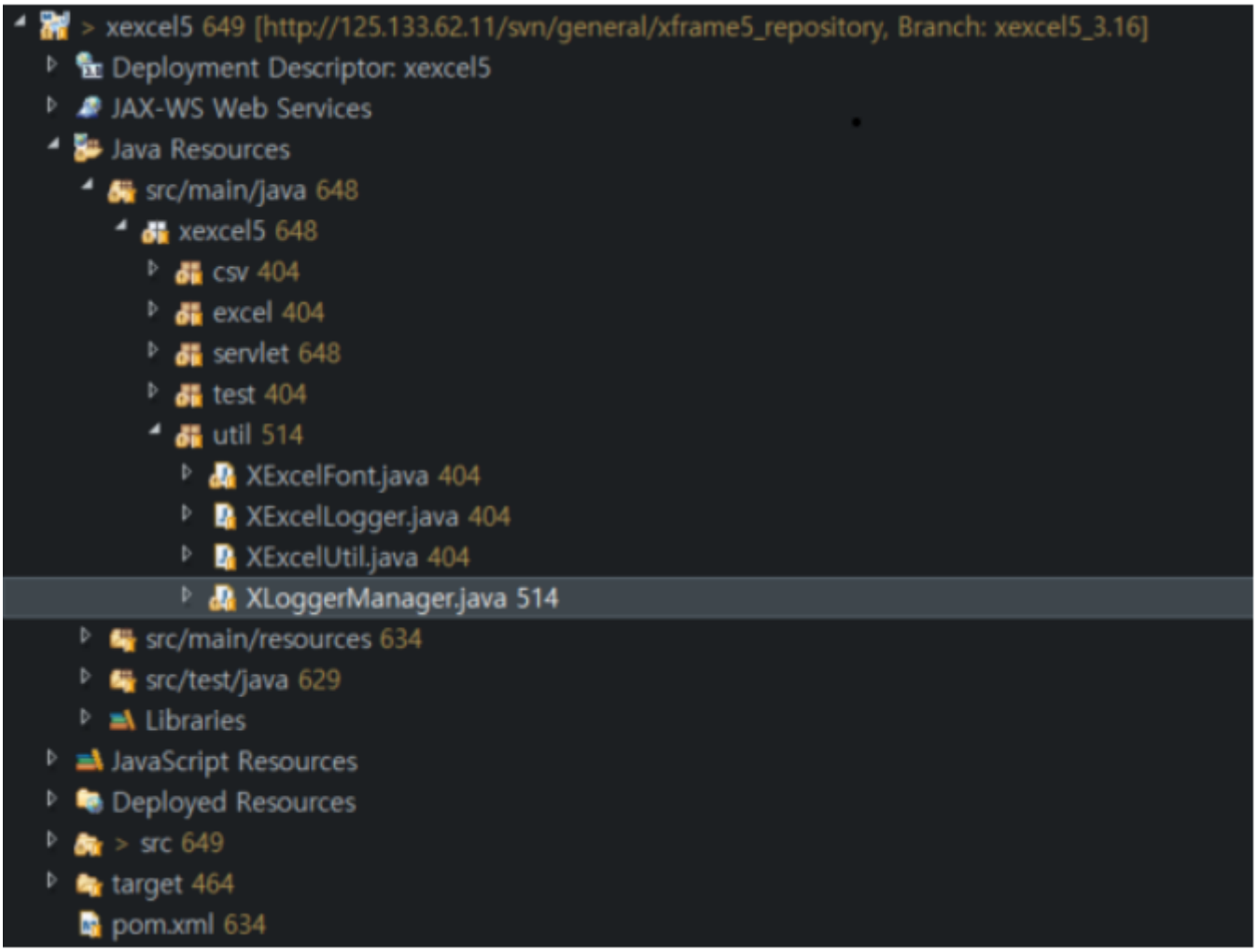
XLoggerManager

XLoggerManager Class 는 xFrame5 에서 엑셀 업로드, 다운로드 기능을 제공하는 Xexcel5 라이브러리 사용시에 로그를 처리 하는 클래스 입니다.

- Package 구성
- 외부 Lib 정보
- 설정방법
- Log4j 설정 정보
- Logger 설정 동적 변경 방법

Package 구성

XLoggerManager 클래스가 포함된 Xexcel5 라이브러리의 package 구성정보는 다음과 같다.



XLoggerManager 클래스 는 xexcel5.util package 안에 포함되어 제공된다.

설정방법

XLoggerManager 를 사용하기 위해 web.xml 에 다음의 내용을 추가한다.

```
<!-- 추가 -->
<servlet>
    <servlet-name>XLoggerManager</servlet-name>
    <servlet-class>xexcel5.util.XLoggerManager</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>XLoggerManager</servlet-name>
    <url-pattern>/XLoggerManager</url-pattern>
</servlet-mapping>
<!-- 추가 종료 -->
```

사용하는 프로젝트 환경에 따라 구성 방법이 다를 수 있습니다.

Log4j 설정 정보

XLoggerManager 가 사용하는 log4j 설정은 다음과 같다.

Log4j2.xml 설정파일

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration>

<!-- 콘솔 로그 (옵션 수정가능) -->
<Appenders>
    <Console name="xloggerconsole" target="SYSTEM_OUT">
    <PatternLayout pattern="[%5p] %d (%F\:%L) %c{1}.%M \: %m%n"/>
    </Console>

   <!-- 파일 로그 (옵션 수정가능)-->
    <!-- name: Appender명, fileName: target파일명, filePattern: history파일명 -
->
    <RollingFile name="xloggerfile" fileName="/excel5Logs/excel5log.log"
filePattern="/excel5Logs/excel5log.%i.log">
        <PatternLayout pattern="[%5p] %d (%F\:%L) %c{1}.%M \: %m%n"/>
        <Policies>
        <!-- size 단위: Byte(default), KB, MB, or GB -->
        <SizeBasedTriggeringPolicy size="5000kb" />
        </Policies>
     <!-- 최대 생성 파일수 (50개가 넘으면 순차적으로 파일이 삭제되도록 설정) -->
        <DefaultRolloverStrategy max="50" fileIndex="min" />
    </RollingFile>
</Appenders>

<Loggers>
    <!-- xexcel5 로거(수정 및 삭제 금지) -->
    <Logger name="xexcel5" level="DEBUG" additivity="false">
        <AppenderRef ref="xloggerconsole" />
    </Logger>
    <!-- xexcel5 템플릿 로거(수정 및 삭제 금지) -->
    <Logger name="xexcel5TempLoger" level="ERROR" additivity="false">
        <AppenderRef ref="xloggerconsole" />
        <AppenderRef ref="xloggerfile" />
    </Logger>
</Loggers>

</Configuration>
```

사용하는 프로젝트 환경에 맞게 xml 파일을 구성할 수 있다.

- Logger 정보
    - Logger 명 : xexcel5

- Logger Level : ERROR
- Appender : xloggerconsole(default) : 콘솔출력, xloggerfile(option) : 파일출력

Logger 설정 동적 변경 방법

WAS 서버를 재 시작 하지 않고 logger 설정을 변경 하기 위해 XLoggerManager 서블릿을 사용한다.

- Parameter
  - logname : 로거명(xexcel5)
  - loglevel : 로그레벨[DEBUG > INFO > WARN > ERROR > FATAL]
  - appender : 로그 appender [all, xloggerfile, xloggerconsole]
  - logprint(옵션) : 로그 설정 출력 여부 [true, false]
- 로그레벨을 DEBUG로 변경하고 CONSOLE에 출력하도록 설정

http://projectpath/XLoggerManager?logname=xexcel5&loglevel=debug&appender=xloggerconsole

- 로그레벨을 DEBUG로 변경하고 FILE과 CONSOLE에 출력하도록 설정

http://projectpath/XLoggerManager?logname=xexcel5&loglevel=debug&appender=all

- 현재 로그정보를 콘솔에 출력

http://projectpath/XLoggerManager?logname=xexcel5&logprint=true

- 로그레벨을 INFO로 변경하고 CONSOLE에 출력하도록 설정하고 현재 로그정보를 콘솔에 출력

http://projectpath/XLoggerManager?logname=xexcel5&loglevel=info&appender=xloggerconsole&logprint=true

DRM 파일처리

본 장에서는 xFrame5 에서 xexcel5 lib 를 사용하여 엑셀업로드 수행 시 DRM 적용된 파일 처리 방안에 대해 기술 합니다.

- 엑셀 업로드시 DRM 처리 class 파라미터 지정 방식
- 엑셀 다운로드시 DRM 처리 class 파라미터 지정 방식
- 엑셀 DRM 처리시 추가 파라미터 전달하는 경우
- 텍스트 및 CSV파일 로드시 DRM 처리 class 파라미터 지정 방식

엑셀 업로드시 DRM 처리 class 파라미터 지정 방식

엑셀 업로드 기능을 사용하는 프로젝트 web.xml 파일의 XExcelUpload 서블릿안에 다음의 내용을 추가 한다.

```
<servlet>
    <servlet-name>XExcelUpload</servlet-name>
```

```
    <servlet-class>xexcel5.servlet.XExcelUpload</servlet-class>
  <!-- 추가 -->
    <init-param>
        <param-name>DRM_CLASS_PATH</param-name>
        <param-value>softbase.sample.XExcelUploadDRMClassSample</param-
value>
    </init-param>
  <!-- 추가 종료-->
</servlet>
<servlet-mapping>
    <servlet-name>XExcelUpload</servlet-name>
    <url-pattern>/XExcelUpload</url-pattern>
</servlet-mapping>
```

<init-param> 안에 <param-value>는 실제 적용할 DRM 클래스 위치를 기술한다.

DRM 복호화 작업을 수행할 XExcelUploadDRMClassSample 샘플 클래스의 내용은 다음과 같다.

```
import xexcel5.util.XExcelDrmInterface;

public class XExcelUploadDRMClassSample implements XExcelDrmInterface {
    public XExcelUploadDRMClassSample() {
    }

    public boolean initialize() throws Exception {
        return true;
    }

    public String execute(String saveFileAbsolutePath) throws Exception {
        // saveFileAbsolutePath : 원본 엑셀 파일 저장 위치
        String decFileAbsolutePath = saveFileAbsolutePath;
        try {

            // TODO: DRM이 걸려 있는 경우에 대한 처리 수행

            // ##### END

            // DRM 복호화 처리 후 복호화한 엑셀 파일 경로를 넘겨준다.
       // 원본 파일에 덮어썼을경우 원본경로 리턴.
            decFileAbsolutePath = saveFileAbsolutePath;
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            return saveFileAbsolutePath;
        }

        return decFileAbsolutePath;
    }
}
```

Line	Description
3	클래스 생성시 XExcelDrmInterface 를 implements 한다.
4	클래스 생성자를 생성한다.
11 ~ 27	public String execute(String saveFileAbsolutePath) 실제 엑셀 DRM 복호화 기능을 처리할 함수 saveFileAbsolutePath : xexcel lib 에서 넘겨주는 엑셀 파일 저장 PATH
29	decFileAbsolutePath : 복호화 처리후 복호화된 파일 PATH 를 넘겨준다.

DRM 연동 엑셀 업로드 처리 흐름

1. xFrame5 UI 에서 엑셀 파일을 선택하여 파일을 서버로 전송 (XExcelUpload 서블릿)
2. xexcel5 lib 에서 엑셀 파일을 서버 디렉토리에 생성. (XExcelUpload class)
3. 저장된 엑셀파일 경로를 DRM class 에 전달. (XExcelUpload class)
4. DRM class에서 복호화 처리 후 복호화된 파일 경로 리턴 (DRM class 개발 대상)
5. 전달 받은 경로의 해당 파일 내용을 읽어온 후 xFrame5 UI로 데이터 전달. (XExcelUpload class)

엑셀 다운로드시 DRM 처리 class 파라미터 지정 방식

엑셀 다운로드 기능을 사용하는 프로젝트 web.xml 파일의 XExcelDownload 서블릿안에 다음의 내용을 추가한다.

```
<servlet>
    <servlet-name>XExcelDownload</servlet-name>
    <servlet-class>xexcel5.servlet.XExcelDownload</servlet-class>
   <!-- 추가 -->
    <init-param>
        <param-name>DRM_ENCRYPT_CLASS_PATH</param-name>
        <param-value>softbase.sample.XExcelDownloadDRMClassSample</param-
value>
    </init-param>
   <!-- 추가 종료-->
</servlet>
<servlet-mapping>
    <servlet-name>XExcelDownload</servlet-name>
    <url-pattern>/XExcelDownload</url-pattern>
</servlet-mapping>
```

<init-param> 안에 <param-value>는 실제 적용할 DRM 클래스 위치를 기술한다.

DRM 암호화 작업을 수행할 XExcelDownloadDRMClassSample 샘플 클래스의 내용은 다음과 같다.

```


import xexcel5.util.XExcelDrmEncryptInterface;

public class XExcelDownloadDRMClassSample implements
```

```
XExcelDrmEncryptInterface {
     public XExcelDownloadDRMClassSample () {
     }

     public boolean initialize() throws Exception {
        return true;
     }

     public String execute(String saveFileAbsolutePath) throws Exception {
        // saveFileAbsolutePath : 원본 엑셀 파일 저장 위치
        String encFileAbsolutePath = saveFileAbsolutePath;

        try {
            // TODO: 생성한 엑셀파일에 대한 DRM 처리 수행

            // ##### END

            // DRM 암호화 처리 후 암호화한 엑셀 파일 경로를 넘겨준다.
       // 원본 파일에 덮어썼을경우 원본경로 리턴.
            encFileAbsolutePath = saveFileAbsolutePath;
        } catch (Exception e) {
            // TODO: handle exception
           e.printStackTrace();
            return saveFileAbsolutePath;
       }

     return encFileAbsolutePath;
   }
}
```

Line	Description
3	클래스 생성시 XExcelDrmEncryptInterface 를 implements 한다.
4	클래스 생성자를 생성한다.
11~27	public String execute(String saveFileAbsolutePath) 실제 엑셀 DRM 암호화 기능을 처리할 함수 saveFileAbsolutePath : xexcel lib 에서 넘겨주는 엑셀 파일 저장 PATH
29	encFileAbsolutePath : 암호화 처리후 암호화된 파일 PATH 를 넘겨준다.

DRM 연동 엑셀 다운로드 처리 흐름

1. xFrame5 UI 에서 그리드의 데이터를 서버로 전송 (XExcelDownload 서블릿)
2. xexcel5 lib 에서 엑셀 파일을 서버 디렉토리에 생성. (XExcelDownload class)
3. 저장된 엑셀파일 경로를 DRM class 에 전달. (XExcelDownload class)
4. DRM class에서 암호화 처리 후 암호화된 파일 경로 리턴 (DRM class 개발 대상)
5. 전달 받은 경로의 해당 엑셀 파일을 xFrame5 UI로 전송. (XExcelUpload class)

엑셀 DRM 처리시 추가 파라미터 전달하는 경우

엑셀 업로드 및 다운로드시 추가로 파라미터를 전달해야 하는 경우 다음처럼 factory 의 setexcelglobalinfo API 를 사용하여 전달할 파라미터를 설정한다.

```
function btn_setexcelglobalinfo_on_moeseup (object)
{
      factory.setexcelglobalinfo("EMP_ID#EMP_NAME") ;
}
```

개별 그리드 마다 다르게 설정하는 파라미터이면 그리드의 setexcelgridinfo API로 설정한다.

DRM 복호화 작업을 수행할 XExcelUploadDRMClassSampleEx 샘플 클래스의 내용은 다음과 같다.

```
import xexcel5.util.XExcelDrmInterfaceEx;

public class XExcelUploadDRMClassSampleEx implements XExcelDrmInterfaceEx {
     public XExcelUploadDRMClassSample() {
     }

     public boolean initialize() throws Exception {
          return true;
     }

     public String execute(String saveFileAbsolutePath, String
strGlobalInfo,
                String strGridInfo) throws Exception {

        // saveFileAbsolutePath : 원본 엑셀 파일 저장 위치
        String decFileAbsolutePath = saveFileAbsolutePath;

        try {
            // TODO: DRM이 걸려 있는 경우에 대한 처리 수행

            // ##### END

            // DRM 복호화 처리 후 복호화한 엑셀 파일 경로를 넘겨준다.
       // 원본 파일에 덮어썼을경우 원본경로 리턴.
            decFileAbsolutePath = saveFileAbsolutePath;
     } catch (Exception e) {
        // TODO: handle exception
        e.printStackTrace();
        return saveFileAbsolutePath;
     }

     return decFileAbsolutePath;
   }
}
```

Line	Description
3	클래스 생성시 XExcelDrmInterfaceEx 를 implements 한다.
4	클래스 생성자를 생성한다.

Line	Description
11 ~ 29	public String execute(String saveFileAbsolutePath) 실제 엑셀 DRM 복호화 기능을 처리할 함수 saveFileAbsolutePath : xexcel lib 에서 넘겨주는 엑셀 파일 저장 PATH strGlobalInfo : factory.setexcelglobalinfo API 에서 설정한 데이터 strGridInfo : 그리드의 setexcelgridinfo API 에서 설정한 데이터
31	decFileAbsolutePath : 복호화 처리후 복호화된 파일 PATH 를 넘겨준다.

DRM 암호화 작업을 수행할 XExcelDownloadDRMClassSampleEx 샘플 클래스의 내용은 다음과 같다.

```
import xexcel5.util.XExcelDrmEncryptInterfaceEx;

public class XExcelDownloadDRMClassSampleEx implements
XExcelDrmEncryptInterfaceEx {
    public XExcelDownloadDRMClassSample () {
    }

    public boolean initialize() throws Exception {
        return true;
    }

    public String execute(String saveFileAbsolutePath, String strGlobalInfo,
        String strGridInfo) throws Exception {

        // saveFileAbsolutePath : 원본 엑셀 파일 저장 위치
        String encFileAbsolutePath = saveFileAbsolutePath;

        try {
            // TODO: 생성한 엑셀파일에 대한 DRM 처리 수행

            // ##### END

            // DRM 암호화 처리 후 암호화한 엑셀 파일 경로를 넘겨준다.
       // 원본 파일에 덮어썼을경우 원본경로 리턴.
            encFileAbsolutePath = saveFileAbsolutePath;
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            return saveFileAbsolutePath;
        }

        return encFileAbsolutePath;
    }
}
```

Line	Description
3	클래스 생성시 XExcelDrmEncryptInterfaceEx 를 implements 한다.
4	클래스 생성자를 생성한다.

Line	Description
11 ~ 29	public String execute(String saveFileAbsolutePath, String strGlobalInfo, String strGridInfo) 실제 엑셀 DRM 복호화 기능을 처리할 함수 saveFileAbsolutePath : xexcel lib 에서 넘겨주는 엑셀 파일 저장 PATH strGlobalInfo : factory.setexcelglobalinfo API 에서 설정한 데이터 strGridInfo : 그리드의 setexcelgridinfo API 에서 설정한 데이터
31	decFileAbsolutePath : 복호화 처리후 복호화된 파일 PATH 를 넘겨준다.

*** 엑셀 파일을 zip 압축하는 예시**

엑셀파일을 zip 압축하는 기능은 xexcel에서 직접 지원하지는 않지만
XExcelDownloadDRMClassSampleEx 클래스에서 압축 후

압축된 파일 경로를 리턴해 주는 방식으로 처리 가능하다.

다음 처럼 화면에서 그리드의 setexcelgridinfo API를 이용하여 엑셀이름과 비밀번호를 설정한다.

```
function btn_on_mouseup(objInst)
{
    // EXCEL_NAME: 저장될 엑셀 파일 이름
    // ZIP_PWD: zip 압축시 설정할 비밀번호
    grdList.setexcelgridinfo("EXCEL_NAME=download.xlsx,ZIP_PWD=123");

  // 엑셀 파일 이름 파라미터를 download.xlsx로 설정시
  // 다운로드되는 zip 파일의 이름은 확장자만 zip으로 변경되어 download.zip이 되고
    // zip 파일에 포함되는 엑셀 파일의 확장자는 xlsx가 된다.
    grdList.downloadexcel("download.xlsx");
}
```

EXCEL_NAME을 설정하지 않으면 엑셀파일 이름은 랜덤한 이름으로 zip 파일에 압축되고

ZIP_PWD을 설정하지 않으면 비밀번호 없이 압축만 수행한다.

zip 압축 기능을 수행할 XExcelDownloadDRMClassSampleEx 샘플 클래스의 내용은 다음과 같으며

압축시 비밀번호를 설정하기 위해 사용하는 zip4j 라이브러리의 버전은 2.11.5 이다.

```
import java.io.File;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.HashMap;
import java.util.Map;
import net.lingala.zip4j.ZipFile;
import net.lingala.zip4j.model.ZipParameters;
import net.lingala.zip4j.model.enums.EncryptionMethod;
import xexcel5.util.XExcelDrmEncryptInterfaceEx;
import org.apache.logging.log4j.LogManager;
```

```
import org.apache.logging.log4j.Logger;

public class XExcelDownloadDRMClassSampleEx implements
XExcelDrmEncryptInterfaceEx
{
    private static final Logger logger =
LogManager.getLogger(XExcelDownloadDRMClassSampleEx.class);

    public XExcelDownloadDRMClassSampleEx () {
    }
    public boolean initialize() throws Exception
    {
        return true;
    }

    @Override
    public String execute(String saveFileAbsolutePath, String strGlobalInfo,
String strGridInf) throws Exception
    {
        String encFileAbsolutePath         = "";
        String zipFilePath                 = "";
        String excelName                 = "";
        String zipPassword                 = "";
        Map<String, String> keyValueMap = null;

        try {
       // 키=값,키=값 형태의 문자열을 파싱하여 맵으로 반환
            keyValueMap = parseKeyValueString(strGridInf);

       // 엑셀 이름과 zip 파일 비밀번호 구하기
            excelName = keyValueMap.get("EXCEL_NAME");
            zipPassword = keyValueMap.get("ZIP_PWD");
            logger.debug("execute > EXCEL_NAME = " + excelName + ",ZIP_PWD =
" + zipPassword);

       // 생성될 zip 파일 경로 구하기
            zipFilePath = getZipFilePath(saveFileAbsolutePath);

       // 파일 압축
            zipFile(saveFileAbsolutePath, zipFilePath, zipPassword,
excelName);

       // 리턴 문자열을 zip 파일 경로로 변경
            encFileAbsolutePath = zipFilePath;
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            return saveFileAbsolutePath;
        }

        return encFileAbsolutePath;
```

```
    }

    /**
     * zip 파일 경로 구하기
     * @param saveFileAbsolutePath 엑셀 경로
     * @return 구한 zip 파일 경로
     */
    public String getZipFilePath(String saveFileAbsolutePath) {
        String fileNameWithoutExtension = "";
        String zipFilePath                 = "";
        String fileName                 = "";
        Path filePath                     = null;
        Path fileNamePath                 = null;
        Path directoryPath                 = null;
        int lastDotIndex                 = -1;

        try {
       // 문자열을 Path로 변환
            filePath = Paths.get(saveFileAbsolutePath);

       // 파일명 얻기
            fileNamePath = filePath.getFileName();

       // 파일명에서 확장자 제외한 이름 얻기
            if (fileNamePath != null) {
                fileName = fileNamePath.toString();

         // 파일명에서 확장자 제외한 부분 얻기
                lastDotIndex = fileName.lastIndexOf('.');
                if (lastDotIndex == -1) {
                    fileNameWithoutExtension = fileName;
                }
                else {
                    fileNameWithoutExtension = fileName.substring(0,
lastDotIndex);
                }
            } else {
                throw new Exception("The provided path is not a valid file
path.");
            }

       // 파일명을 제외한 경로 얻기
            directoryPath = filePath.getParent();

       // 생성될 zip 파일 경로 만들기
            zipFilePath = directoryPath.toString() + File.separator +
fileNameWithoutExtension + ".zip";
        } catch (Exception e) {
            e.printStackTrace();
        }

        return zipFilePath;
```

```
    }

    /**
   * 파일 압축하기
     * @param sourceFilePath 압축될 엑셀 파일 경로
     * @param zipFilePath 만들어질 zip 파일 경로
     * @param password 설정할 비밀번호
     */
    public void zipFile(String sourceFilePath, String zipFilePath, String
password, String excelName) {
        ZipParameters zipParameters = null;
        ZipFile zipFile             = null;

        try {
            // ZipParameters 객체 생성
            zipParameters = new ZipParameters();

       // 엑셀파일 이름을 화면에서 전달한 경우
            if (excelName != null) {
                zipParameters.setFileNameInZip(excelName);
            }

       // 비밀번호 설정 하지 않은 경우
            if (password == null) {
                zipFile = new ZipFile(zipFilePath);
            }
       // 비밀번호 설정 한 경우(비밀번호 설정하면서 ZipFile 생성)
            else {
                // ZipParameters에 암호관련 설정
                zipParameters.setEncryptFiles(true);
zipParameters.setEncryptionMethod(EncryptionMethod.ZIP_STANDARD);

                // ZipFile 객체 생성
                zipFile = new ZipFile(zipFilePath, password.toCharArray());
            }

            // zip 파일에 엑셀 파일 추가
            zipFile.addFile(new File(sourceFilePath), zipParameters);
            zipFile.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    /**
   * 키=값,키=값 형태의 문자열을 파싱하여 맵으로 반환하는 메소드
     *
     * @param inputString 키=값,키=값 형태의 문자열
     * @return 파싱된 키와 값으로 구성된 맵
     */
    public Map<String, String> parseKeyValueString(String inputString) {
        String[] keyValuePairs             = null;
```

```
        Map<String, String> keyValueMap = null;
        String[] keyValue                 = null;
        String key                         = "";
        String value                      = "";
        String pair                       = "";
        int i                              = 0;

        try {
      // 결과를 저장할 맵 생성
            keyValueMap = new HashMap<>();

      // 파라미터 전달하지 않은 경우 리턴
            if (inputString.equals("")) { return keyValueMap; }

      // 문자열을 쉼표로 분리
            keyValuePairs = inputString.split(",");

      // 각 키=값 쌍을 등호로 분리하여 맵에 저장
            for (i = 0; i < keyValuePairs.length; i++) {
                pair = keyValuePairs[i];
                keyValue = pair.split("=");

                if (keyValue.length == 2) {
                    key = keyValue[0];
                    value = keyValue[1];
                    keyValueMap.put(key, value);
                } else {
           // 올바르지 않은 형식의 입력
                    logger.debug("Incorrect format of input: " + pair);
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

        return keyValueMap;
    }
}
```

Line	Description
35	화면에서 전달한 엑셀이름과 비밀번호 문자열을 map으로 변환한다.
43	생성될 zip 파일 경로를 구한다. 엑셀파일과 동일한 경로와 이름에 확장자만 zip으로 변경된다.
46	zip4j 라이브러리를 사용하여 엑셀파일을 압축한다.
49	반환 문자열을 zip 파일 경로로 변경한다.

텍스트 및 CSV 파일 로드시 DRM 처리 class 파라미터 지정 방식

텍스트 파일 혹은 csv 파일에 DRM 을 해제해야 하는 경우 서버와 연동하여 DRM 해제 후 그리드에 로드할 수 있다.

xframe5.js에 공통속성 설정

xframe5.js 에 다음처럼 서블릿 경로와 업로드 방식을 설정한다(개발툴 설치폴더 doc > 98 번문서 참고)

```
xdatafile_upload_url: './xdatafileupload',
grid_csvupload_methodtype: 2, // csv 파일 업로드 방식(1: SCGIPT(기본값), 2:HTTP)
grid_txtupload_methodtype: 2,  //TEXT 파일 업로드 방식(1: SCGIPT(기본값),
2:HTTP)
```

web.xml 에 서블릿 설정

업로드 기능을 사용하는 프로젝트 web.xml 파일의 XDataFileUpload 서블릿안에 다음의 내용을 추가한다.

```
<servlet>
    <servlet-name>XDataFileUpload</servlet-name>
    <servlet-class>xdatafile5.servlet.XDataFileUpload</servlet-class>
        <init-param>
            <param-name>DRM_CLASS_PATH</param-name>
            <param-
value>softbase.sample.XDataFileUploadDRMClassSample</param-value>
        </init-param>
        <init-param>
            <param-name>DEFAULT_CHARSET</param-name>
            <param-value>EUC-KR</param-value>
        </init-param>
</servlet>
<servlet-mapping>
    <servlet-name>XDataFileUpload</servlet-name>
    <url-pattern>/XDataFileUpload</url-pattern>
</servlet-mapping>
```

DRM_CLASS_PATH 파라미터의 <param-value>는 실제 적용할 DRM 클래스 위치를 기술한다.

DEFAULT_CHARSET는 업로드하는 파일의 BOM(Byte Order Mark)이 없는 경우 사용할 인코딩 값이다. 업로드 하는 파일의 BOM이 있는 경우 BOM값에 따라 파일 내용을 읽고, BOM이 없는 경우 juniversalchardet 라이브러리를 사용하여 파일 내용에서 인코딩을 찾는다.

찾은 인코딩이 US-ASCII, UTF-8, EUC-KR 인 경우 찾은 인코딩으로 파일 내용을 읽는다. 그 외의 값인 경우 DEFAULT_CHARSET에 설정한 인코딩(기본값: EUC-KR)으로 파일 내용을 읽는다.

TEMP_DIR 파라미터를 설정하면 임시 폴더가 생성되는 경로를 변경할 수 있다.

DRM 처리할 클래스 생성

DRM 복호화 작업을 수행할 XDataFileUploadDRMClassSample 샘플 클래스의 내용은 다음과 같다.

```
import xdatafile5.util.XDataFileDrmInterface;

public class XDataFileUploadDRMClassSample implements XDataFileDrmInterface
{
    public XDataFileUploadDRMClassSample () {
    }

    public boolean initialize() throws Exception {
        return true;
    }

    public String execute(String saveFileAbsolutePath) throws Exception {
        // saveFileAbsolutePath : 원본 엑셀 파일 저장 위치
        String decFileAbsolutePath = saveFileAbsolutePath;

        try {
            // TODO: DRM 이 걸려 있는 경우에 대한 처리 수행

            // ##### END

            // DRM 복호화 처리 후 복호화한 엑셀 파일 경로를 넘겨준다.
       // 원본 파일에 덮어썼을경우 원본경로 리턴.
            decFileAbsolutePath = saveFileAbsolutePath;
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            return saveFileAbsolutePath;
        }

        return decFileAbsolutePath;
    }
}
```

Line	Description
3	클래스 생성시 XDataFileDrmInterface 를 implements 한다.
4	클래스 생성자를 생성한다.
11 ~ 27	public String execute(String saveFileAbsolutePath) 실제 엑셀 DRM 복호화 기능을 처리할 함수 saveFileAbsolutePath : xdatafile5 lib 에서 넘겨주는 파일 저장 PATH
29	decFileAbsolutePath : 복호화 처리후 복호화된 파일 PATH 를 넘겨준다.

추가로 전달할 파라미터가 있다면 엑셀과 마찬가지 방식으로 XDataFileDrmInterfaceEx 인터페이스를 상속하고 화면에서 factory의 setexcelglobalinfo나 그리드의 setexcelgridinfo API를 호출하여 설정한다.

화면에서 파일 로드 API 호출

그리드에 텍스트 파일을 로드하는 경우 readtxt 혹은 readtxtex API를 사용한다. CSV을 로드할 경우에는 readcsv혹은 readcsvex API를 사용하여 로드한다.

```
function btn_readcsv_on_mouseup (objInst)
{
     //CSV 파일 1행 부터 읽기 처리 시작
     grd.readcsves(1,1, "", false);
}
```

로그 설정

로거이름은 xdatafile5 이며 log4j2.xml 를 사용할 경우 다음과 같이 설정한다.

```
<Loggers>
    <!-- xdatafile5 로거 -->
    <Logger name="xdatafile5" level="DEBUG" additivity="false">
        <AppenderRef ref="xloggerconsole" />
    </Logger>
</Loggers>
```

사용하는 라이브러리

사용하는 라이브러리는 다음과 같다.

파일명
• xdatafile5.jar • commons-fileupload-1.2.1.jar • commons-io-1.4.jar • commons-logging-1.2.jar • juniversalchardet-2.4.0.jar

텍스트 및 CSV 파일 다운로드시 DRM 처리 class 파라미터 지정 방식

텍스트 파일 혹은 csv 파일에 DRM 을 설정해야 하는 경우 서버와 연동하여 DRM 처리 후, DRM 적용된 파일을 다운로드 할 수 있다.

xframe5.js에 공통속성 설정

xframe5.js 에 다음처럼 서블릿 경로와 다운로드 방식을 설정한다.

```
XDATAFILE_DOWNLOAD_URL: '/XdatafileDownload',
GRID_CSVDOWNLOAD_METHODTYPE: 2,
```

```
GRID_TXTDOWNLOAD_METHODTYPE: 2,
```

web.xml 에 서블릿 설정

다운로드 기능을 사용하는 프로젝트 web.xml 파일의 XDataFileUpload 서블릿안에 다음의 내용을 추가한다.

```
<servlet>
<servlet-name>XDataFileDownload</servlet-name>
    <servlet-class>xdatafile5.servlet.XDataFileDownload</servlet-class>
    <init-param>
        <param-name>DRM_ENCRYPT_CLASS_PATH</param-name>
        <param-
value>softbase.sample.XDataFileDownloadDRMClassSample</paramvalue>
    </init-param>
</servlet>
<servlet-mapping>
    <servlet-name>XDataFileDownload</servlet-name>
    <url-pattern>/XDataFileDownload</url-pattern>
</servlet-mapping>
```

DRM_ENCRYPT_CLASS_PATH 파라미터의 <param-value>는 실제 적용할 DRM 클래스 위치를 기술한다.

TEMP_DIR 파라미터를 설정하면 임시 폴더가 생성되는 경로를 변경할 수 있다.

DRM 처리할 클래스 생성

DRM 암호화 작업을 수행할 XDataFileDownloadDRMClassSample 샘플 클래스의 내용은 다음과 같다.

```
import xdatafile5.util.XDataFileDrmInterface;

public class XDataFileDownloadDRMClassSample implements
cDataFileDrmEncryptInterface {
    public XDataFileDownloadDRMClassSample () {
    }

    public boolean initialize() throws Exception {
        return true;
    }

    public String execute(String saveFileAbsolutePath) throws Exception {
        // saveFileAbsolutePath : 원본 파일 저장 위치
        String encFileAbsolutePath = saveFileAbsolutePath;
        try {
            // TODO: 생성한 파일에 대한 DRM 처리 수행
```

```
            // ##### END

            // DRM 암호화 처리 후 암호화한 파일 경로를 넘겨준다.
        // 원본 파일에 덮어썼을경우 원본경로 리턴.
            encFileAbsolutePath = saveFileAbsolutePath;
          } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
            return saveFileAbsolutePath;
          }

          return encFileAbsolutePath;
    }
}
```

Line	Description
3	클래스 생성시 XExcelDrmEncryptInterface 를 implements 한다.
4	클래스 생성자를 생성한다.
11 ~ 26	public String execute(String saveFileAbsolutePath) 실제 파일 DRM 암호화 기능을 처리할 함수 saveFileAbsolutePath : xdatafile5 lib 에서 넘겨주는 파일 저장 PATH
28	encFileAbsolutePath : 암호화 처리후 암호화된 파일 PATH 를 넘겨준다.

추가로 전달할 파라미터가 있다면 엑셀과 마찬가지 방식으로 XDataFileDrmEncryptInterfaceEx 인터페이스를 상속하고 화면에서 factory의 setexcelglobalinfo나 그리드의 setexcelgridinfo API를 호출하여 설정한다.

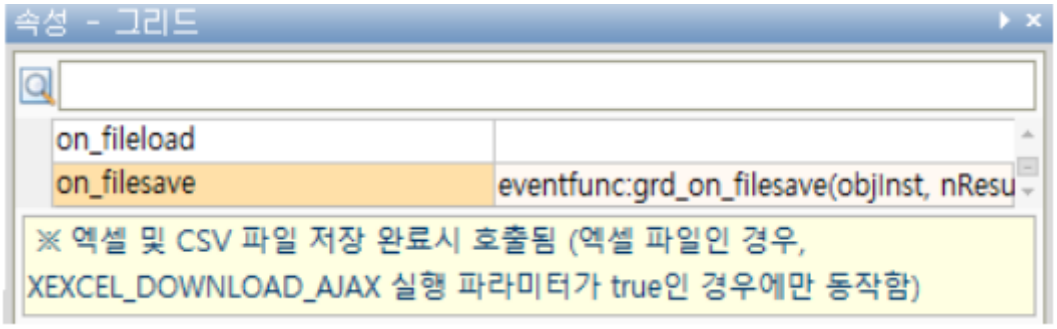
화면에서 파일 다운로드 API 호출

그리드의 내역을 텍스트 파일로 다운로드하는 경우 savetxt 혹은 savetxtex API를 사용한다. CSV파일로 다운로드할 경우에는 savecsv 혹은 savecsvex API를 사용하여 다운로드한다.

```
function btn_savecsv_on_mouseup(objInst)
{
    // csv로 파일로 저장
    grd.savecsv();
}
```

다운로드 완료시 콜백함수

텍스트 및 CSV 파일 다운로드 성공 혹은 실패시 그리드의 on_filesave 이벤트가 호출된다. 실패시 서버로부터 전달된 실패 이유를 결과 메시지로 확인할 수 있다.



배경 이미지 설정

본 장에서는 xFrame5 에서 xexcel5 lib 를 사용하여 엑셀다운로드 수행 시 배경 이미지 설정 방법에 대해 기술 합니다.

- 엑셀 다운로드시 워터마크 이미지 설정

엑셀 다운로드시 워터마크 이미지 설정

엑셀에서 자체적으로 워터마크 기능을 제공하지는 않지만

머리글에 로고 이미지등을 설정하여 워터마크 기능을 수행 할 수 있다.

머리글에 이미지를 설정하면, 인쇄시 설정한 이미지가 표시된다.

엑셀 다운로드 기능을 사용하는 프로젝트 web.xml 파일의 XExcelDownload 서블릿안에 다음의 내용을 추가한다.

```
<servlet>
  <servlet-name>XExcelDownload</servlet-name>
  <servlet-class>xexcel5.servlet.XExcelDownload</servlet-class>
 <!-- 추가 -->
  <init-param>
      <param-name>HEADERIMAGE_PATH</param-name>
      <param-value>C:\\sample.png</param-value>
  </init-param>
 <!-- 추가 종료-->
</servlet>
<servlet-mapping>
  <servlet-name>XExcelDownload</servlet-name>
  <url-pattern>/XExcelDownload</url-pattern>
</servlet-mapping>
```

<init-param> 안에 <param-value>는 적용할 이미지 파일의 위치를 기술한다.

이미지는 png 확장자 이미지를 설정할 수 있다.

.xlsx 파일에서만 동작하는 기능이다.

다운로드 되는 시트가 2개 이상인 경우 모든 시트에 동일한 이미지가 설정된다.

From:
https://technet.softbase.co.kr/wiki/ - **xFrame5 TechNet**

Permanent link:
https://technet.softbase.co.kr/wiki/guide/general/excel_guide

Last update: **2025/06/02 16:27**