

# **IBUpload**

# **개발자가이드**

## IBUpload 개발자 가이드

초판발행/ 2016 년 4 월 1 일

지은이/ 함범중, 이연서

펴낸이/ 정광천

펴낸곳/  **아이비리더스**  
IB Leaders Co., Ltd.

08394 서울시 구로구 디지털로 242(구로동) 한화비즈메트로 1501호

전화/ 02) 2621 - 2080

팩스/ 02) 2621 - 2088

편집/ 함범중 인쇄/ C&G

※ 기술상담 및 내용문의는 (주)아이비리더스로 해주십시오.

<http://www.ibleaders.co.kr>

# Contents

<b>IBUpload</b>	<b>1</b>
<b>개발자가이드</b>	<b>1</b>
<b>Contents</b>	<b>3</b>
<b>1 개요</b>	<b>9</b>
<b>1.1. IBUpload의 개요</b>	<b>10</b>

1.2. IBUpload의 주요기능	10
---------------------	----

## 2 시작하기 13

2.1 설치	14
--------	----

2.2 환경설정	18
----------	----

2.3 IBUpload 개체 삽입 및 초기화	21
--------------------------	----

2.4 업로드기능 구현하기	28
----------------	----

2.5 다운로드 기능 구현하기	34
------------------	----

2.6 오류 처리	36
-----------	----

2.7 조회 기능 구현하기	38
----------------	----

## 3 API 레퍼런스 40

<b>A. 속성</b>	<b>41</b>
addTrigger 속성	42
autoUpload 속성	43
contextMenuItems 속성	45
downloadFileName 속성	47
downloadServerUrl 속성	49
exceptDuplicated 속성	50
exceptZeroFile 속성	51
files 속성	52
fileSeparator 전역속성	55
headerText 속성	56
iconMode 속성	59
limitFileCount 속성	60
limitFileCountOnce 속성	61
limitFileSize속성	62
limitFileTotalSize속성	63
limitFileExt 속성	64
limitFileExtMode 속성	66
overWriteFile 속성	68
prependExtendParam 속성	69
submitNoData 속성	70
target 속성	71
theme 속성	73
uploadServerUrl 속성	74
useDragDrop 속성	75
locale 전역속성	76

msgLevel 전역속성	78
useDownloadCookie 전역속성	80
userAgent 전역속성	82
userInputName 전역속성	84
<b>B. 함수</b>	<b>87</b>
add 메소드	88
addDropFile 메소드	90
addFiles 메소드	92
allFiles 메소드	94
delete 메소드	96
deleteAll 메소드	98
deleteFiles 메소드	99
deleteIndex 메소드	101
deleteReserved 메소드	103
deleteServer 메소드	105
downloadAll 메소드	107
download 메소드	108
downloadFileName 메소드	110
downloadServerUrl 메소드	111
dropSheetRow 메소드	112
dropTarget 메소드	114
extendParamDownload 메소드	116
extendParamUpload 메소드	118
fileCount 메소드	120
fileUploadedCount 메소드	121

iconMode 메소드	122
isModified 메소드	123
files 메소드	124
fileList 메소드	126
limitFileCount 메소드	128
limitFileCountOnce 메소드	129
limitFileSize 메소드	130
limitFileTotalSize 메소드	131
linkDown 메소드	132
reset 메소드	134
selectedIndex 메소드	135
selectAll 메소드	137
upload 메소드	138
uploadServerUrl 메소드	141
version 메소드	142

## C. 이벤트 143

onAddFile 이벤트	144
onBeforeAddFile 이벤트	146
onBeforeDeleteFile 이벤트	148
onContextMenu 이벤트	150
onDeleteFile 이벤트	151
onDragSheetRow 이벤트	152
onDbClick 이벤트	154
onMessage 이벤트	156
onSetFiles이벤트	158

onUploadData 이벤트	160
onUploadFinish 이벤트	162
onUploading 이벤트	164
<b>D. 자바스크립트 유틸</b>	<b>166</b>
<b>E. 메시지 코드</b>	<b>166</b>
정보 메시지	166
오류 메시지	166
<b>F. 브라우저 버전별 제약사항</b>	<b>168</b>
한번에 여러개의 파일 선택 기능	168
업로드 파일 용량 제한 기능	168



# CHAPTER

# *1* 개요

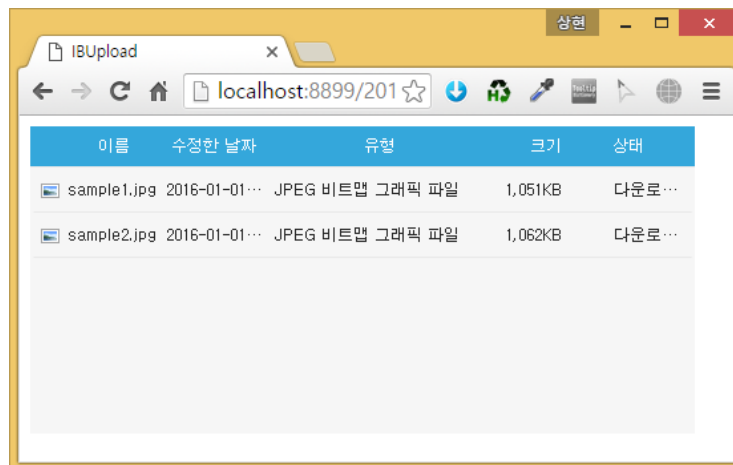
## 1.1. IBUpload의 개요

웹 환경에서 파일의 업로드, 다운로드, 삭제 등과 같은 파일 첨부나 파일 처리와 관련된 기능들을 제공하는 자바스크립트 기반의 웹 컨트롤이며, 이와 같은 파일 처리들을 좀더 쉽게 개발하고, 운영할 수 있도록 돕는 컨트롤 도구이다.

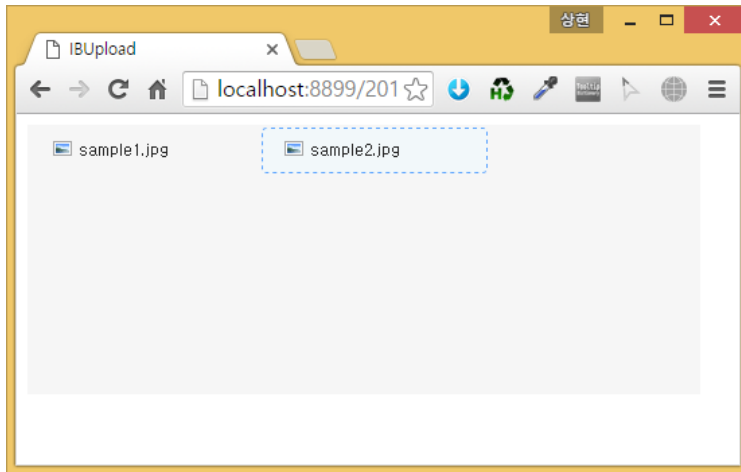
## 1.2. IBUpload의 주요기능

- 다양한 표시 기능

탐색기에서 파일을 표시할 때 "큰 아이콘", "작은 아이콘", "간단히", "자세히" 형태로 파일 아이콘들이 표시되는 것 처럼, 다운로드할 또는 업로드된 파일들을 표시하는 다양한 방법을 제공해 주고 있다.



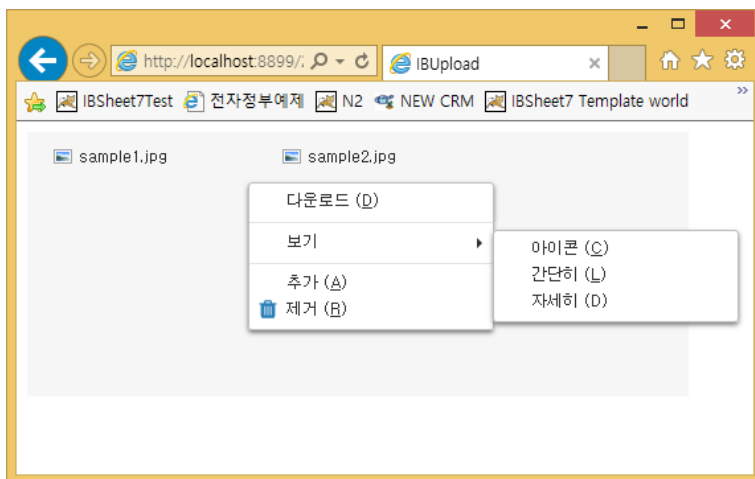
[자세히 (detail) 화면]



[간단히 (list) 화면]

- 다양한 키보드와 마우스 사용자 경험 지원

키보드나 마우스를 이용하여 제품 안에 있는 파일을 선택할수 있고, 키보드 또는 마우스 오른쪽 버튼을 이용한 컨텍스트 메뉴 기능 등을 제공한다.



- 파일 제한 기능

업로드가 가능한 파일 확장자들 또는 불가능한 확장자들을 설정하여 특정 파일들을 첨부을 제한하는 기능을 구현할 수 있으며, 파일 개수 제한 기능과 파일별 용량 제한 기능, 전체 파일의 총 용량을 제한 하는 기능 등이 제공된다.

- 다양한 기능 사용 방법 제공

파일 추가, 파일 삭제 등 대부분의 기능들을 API 함수를 이용하여 직접 호출 및 제어하거나, UI 에서 직접 사용자가 마우스 오른쪽 버튼의 Context 메뉴를 이용하여 동작할 수 있다.

- 메시지 기능

제품에서 발생하는 모든 동작과 관련된 이벤트들이 제공되며, 처리 결과에 따른 정보나 오류 등의 다양한 결과 메시지들을 제공하고 있다. 또한 모든 메시지 들은 고객이 원하는 메시지로 언제든지 변경 설정이 가능하다.

- 진행 율 표시 기능

파일이 업로드 되는 동안, 진행율을 표시할 수 있다.

- 대용량 전송 기능

1 GB ~ 2 GB 이상의 대용량 파일도 안정적으로 업로드 할 수 있는 기능이 제공된다.

# CHAPTER

## 2 시작하기

## 2.1 설치

### IBUpload 의 설치

IBUpload 를 이용한 개발 환경을 구축하기 위해서 아래의 파일 목록들을 웹서버의 특정 폴더에 복사하여 설치한다.

위치는 다음을 기준으로 한다.

#### 배포 파일 목록

- 기본 엔진 파일들

/js/

/js/ibleaders.js

/js/ibupload.js

/js/ibuploadinfo.js

/js/Main (폴더)

- 관련 라이브러리 (jquery)

/jquery/jquery-1.7.2.min.js

/jquery/jquery-ui-1.10.4.min.js

/jquery/jquery.contextMenu.min.js

/jquery/jquery.contextMenu.min.css

/jquery/font

- 서버 파트 jar 파일들 ( 반영시 서버 재기동 필요 )
  - /jar/commons-fileupload-1.3.1.jar ( JDK 1.5 이상 권장 )
  - /jar/commons-io-2.4.jar ( JDK 1.5 이상 권장 )
  - /jar/ant.jar ( JDK 1.5 이상 권장 )
  - /jar/json-simple-1.1.1.jar ( JDK 1.5 이상 권장 )
- 서버 파트 템플릿 jsp 파일들 (ibuploadinfo.js 에서 URL 변경 설정 가능)
  - /jsp/download\_sample.jsp
  - /jsp/upload\_sample.jsp

## IBUpload 의 파일 내용 (역할)

구분	파일명	내 용
엔진 (js)	ibleaders.js	아이비리더스 제품 관련 라이선스 정보
	ibupload.js	IBUpload 엔진 파일
	ibuploadinfo.js	IBUpload 기본 설정 정보 및 메세지 정의, 유틸 함수 모음
	Main	파일별 icon 이미지와 디자인 css를 포함한 폴더
라이브 러리 (jquery )	jquery-1.7.2.min.js	필수 사용 라이브러리. 이미 사용되는 jQuery 버전이 있을 경우 기 존 버전을 사용해도 무관함 ( 1.7.2 버전 이 상. <a href="http://jquery.com">http://jquery.com</a> 참조 )
	jquery-ui-1.10.4.min.js	필수 사용 라이브러리. 이미 사용되는 jQuery-ui 버전이 있을 경우 기존 버전을 사용해도 무관함 ( 1.10 버전 이상 – Interactions – Selectable 체크 필요. <a href="http://jqueryui.com">http://jqueryui.com</a> 참조 )
	jquery.contextMenu.mi n.js	필수 사용 라이브러리 ( <a href="http://swisnl.github.io/jQuery-contextMenu/">http://swisnl.github.io/jQuery- contextMenu/</a> 참고 )
	jquery.contextMenu.mi n.css	
	jquery.contextMenu.mi n.js.map	디버깅 용도. ( 개발 단계에서 필요하나 운 영단계에서 F12 개발자 모드를 지원하지



		않을 경우에는 배포할 필요가 없음 )
	font	contextMenu 관련 font
서버 모듈	commons-fileupload-1.3.1.jar	업로드 파일의 수신 모듈
	commons-io-2.4.jar	업로드 파일의 수신 관련 모듈
	ant.jar	한글 파일명을 지원하는 압축 모듈 (7.3.0.11 버전부터 jazzlib.jar 대신 ant.jar 사용)
	json-simple-1.1.1.jar	JSON 라이브러리
서버 서비스 (jsp)	download_sample.jsp	서버파트 - 파일 다운로드시 처리를 위한 템플릿 ( 예제 소스코드 제공 )
	upload_sample.jsp	서버파트 - 파일 업로드시 처리를 위한 템플릿 ( 예제 소스코드 제공 )

## 2.2 환경설정

### 환경 설정

IBUpload 컨트롤의 기본 적인 환경 설정은 ibuploadinfo.js 파일을 통해 설정할 수 있다.

- 각종 API 속성들에 대한 기본 값의 정의
- Client 의 각종 메시지에 대한 정의
- 공통 이벤트에 대한 로직 구성
- 팝업메뉴의 구성 내용

ibuploadinfo.js 파일 안에서 제공되는 ibleaders.ibupload 개체의 각종 설정 값들은 IBUpload 컨트롤이 기본적으로 참고하여 사용하는 환경설정 항목이다.

설치시 반드시 변경 또는 점검해야 하는 필수 변경항목과 필요한 경우 한하여 운영환경에 따라 변경할 수 있는 옵션 항목이 존재한다.

설정 항목	설정 내용
uploadServerUrl	upload 함수 호출시 전송될 서버 URL (upload_sample.jsp 참고)
downloadServerUrl	download 함수 호출시 전송될 서버 URL (download_sample.jsp 참고)
addTrigger	IE9 환경에서 파일 추가 다이얼로그 창을 클라이언트에서 생성
autoUpload	파일 추가시 즉시 업로드를 수행할 지의 여부 (default:true)
exceptDuplicated	중복 파일 업로드 방지 여부

exceptZeroFile	크기가 0byte인 파일 업로드 방지 여부
limitFileCount	전송할 파일 개수 제한
limitFileCountOnce	한번에 전송할 파일 개수 제한
limitFileSize	한번에 전송할 개별 파일 사이즈 제한
limitFileTotalSize	한번에 전송할 전체 파일 사이즈 제한
limitFileExt	제약조건 파일 확장자
limitFileExtMode	limitFileExt 기술된 확장자를 허용/거부 설정
limitFileExtServer	limitFileExt 와 상관없이 서버에서 실행되면 안되는 서버 보안상 꼭 막아야할 파일들에 대한 정의. 이 설정은 Client 에서 1차적으로막을 뿐, 서버 파트에서 도 2중으로 다시 한번 더 막아야 완벽한 서버 보안이 가 능함.
theme	초기에 표시할 테마 설정
viewType	icon : 아이콘 업로드 뷰어 ibsheet : ibsheet 연동 뷰어 (default:icon)
iconMode	아이콘의 정렬 형태 ( icon, list, detail) (default:icon)
uploadEncoding	한글 파일명, 한글 데이터의 업로드시 인코딩 방식 (default:utf-8)
contextMenuItems	마우스 우측버튼 클릭시 보여질 컨텍스트메뉴 설정
message	정보 및 오류 메세지, 화면에 보이는 한/영 텍스트 정의. INFO-XXX : 오류 아닌 일반 메세지에 대한 정의 ERR-XXX : 오류 메세지 TEXT-XXXX : 화면에 보이는 한글 텍스트를 다른 언어의

	텍스트로 교체 설정 가능함.
supportIcon	지원되는 아이콘 확장자 중 아이콘 모양이 file.gif 로 대체되어 표시됨. 지원되는 파일 아이콘을 실제로 js/Main/icon16 폴더와 js/Main/icon32 에 추가한 경우 여기에 추가로 기입해 줘야 제대로 표시된다.
fileType	아이콘의 확장자별 설명글
userAgent	업로드 전송시 HTTP 헤더에 추가할 내용
IBSheetLoadPage_Main (예시)	ibsheet사용시 초기화 관련 구성. 테마별로 별도 함수 추가하여 사용 가능
useDragDrop	DragDrop기능 사용 여부
overWriteFile	중복 파일 덮어쓸지 여부
submitNoData	업로드할 파일이 없을 때 서버에 request 전송 여부

## 2.3 IBUpload 개체 삽입 및 초기화

### 2.3.1 필요 파일 인클루드.

먼저 제품과 같이 배포되는 jquery, jquery-ui,css를 인클루드 하고, ibuploadinfo.js와 ibupload.js 파일을 인클루드 한다. jQuery 의 최소 권장 버전은 1.7.2 이다.

```
<!-- 필수 jquery library -->
<script type="text/javascript" src="/jquery/jquery-1.7.2.min.js"></script>
<script type="text/javascript" src="/jquery/jquery-ui-1.10.4.min.js"></script>

<!-- contextMenu 관련 jquery 추가 모듈 -->
<link href="/jquery/jquery.contextMenu.min.css" rel="stylesheet" type="text/css" />
<script type="text/javascript" src="/jquery/jquery.contextMenu.min.js"></script>

<!-- ibupload 모듈 -->
<script type="text/javascript" src="js/ibleaders.js"></script>
<script type="text/javascript" src="js/ibuploadinfo.js"></script>
<script type="text/javascript" src="js/ibupload.js"></script>
```

**[javascript,css 파일 인클루드]**

### 2.3.2 div 객체 삽입.

body 안에 IBUpload 컨트롤을 표시 할 div 객체를 생성한다. 이때 IBUpload객체는 상위 객체의 크기(너비,높이)를 따르기 때문에 상위 객체가 반드시 크기를 갖고 있어야 한다.

```
<div class="content">
  <div class="btn_div">
    <button type="button" onclick="doAction(add)">파일추가</button>
    <button type="button" onclick="doAction('download')">다운로드</button>
    <button type="button" onclick="doAction('delete')">파일삭제</button>
```

```

        <button type="button" onclick="doAction(save)">저장</button>
    </div>
    <hr>
    <!-- 업로드 상위 객체의 크기를 따른다 -->
    <div style="width:750px;height:400px">
        <div id="#myUpload"></div>
    </div>
</div>

```

[ div를 통한 업로드 객체 삽입]

### 2.3.3 업로드 객체 초기화.

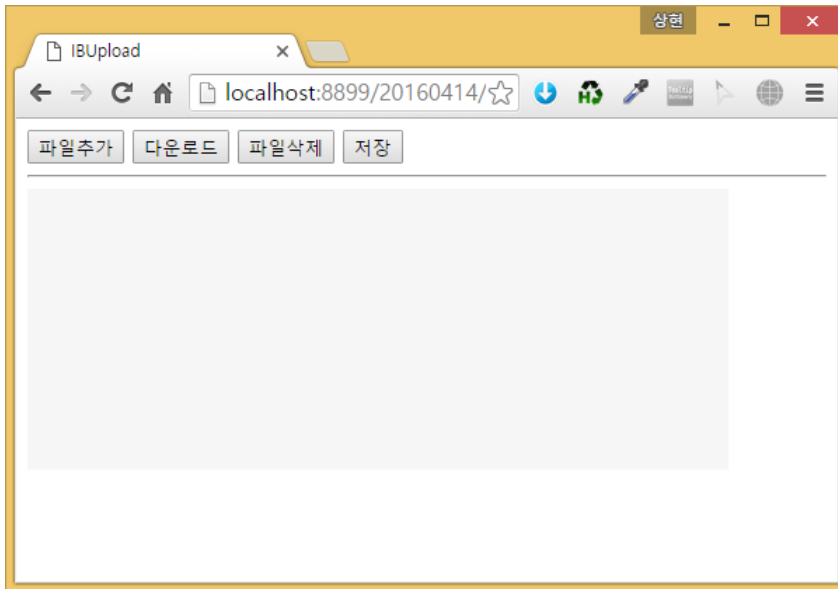
웹페이지의 로딩이 완료된 순간(body onload)에 IBUpload 개체를 생성 및 초기값을 설정할 수 있다.

```

<script>
$(document).ready(function(){
    //객체생성
    $("#myUpload").IBUpload();
});
</script>

```

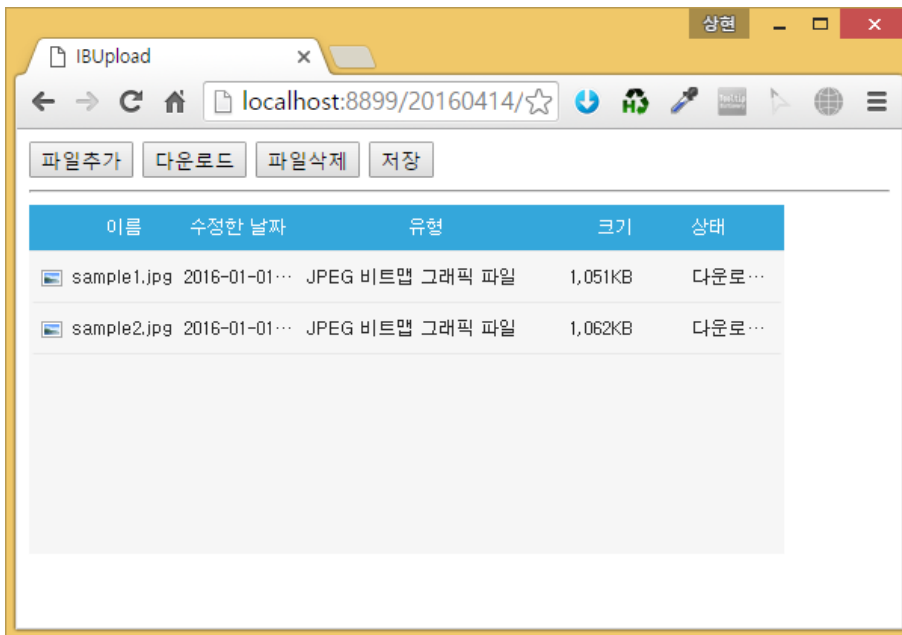
[기본 형태 객체 생성]



[다른 설정이 없이 객체만 생성했을 경우 모습]

```
<script>
$(document).ready(function(){
    //객체생성
    $("#myUpload").IBUpload({
        //디자인 변경
        iconMode:"detail",
        //자동 업로드 여부
        autoUpload:false,
        //초기 로드 파일
        files: [
            {name: "sample1.jpg", size: "1075761", date: "20160101125959",
            url: "20160126_180801"},
            {name: "sample2.jpg", size: "1087426", date: "20160101125959",
            url: "20160126_180801_64356079"}
        ]
    });
});
</script>
```

[객체 생성 및 기본값 설정]



[디자인 변경 및 초기 파일이 설정된 모습]

### 2.3.4 화면 버튼 기능 연결.

화면에 추가한 버튼과 연결된 기능을 정의한다.

```
<script>
function doAction(str){
  switch(str){
    case "add"://파일추가
      $("#myUpload").IBUpload("add");
      break;
    case "download"://선택 파일 삭제
      $("#myUpload").IBUpload("download");
      break;
    case "delete"://선택파일 다운로드
      $("#myUpload").IBUpload("delete");
      break;
    case "save"://변경 내용 저장
      $("#myUpload").IBUpload("upload");
      break;
  }
}
```



```

//ibupload 보기모양 동적 변경
case "icon":
case "list":
case "detail":
    $("#myUpload").IBUpload("iconMode",str); //함수호출
    break;
} //end switch
}
</script>
<body>
<button type="button" onclick="doAction('add')">파일추가</button>
<button type="button" onclick="doAction('download')">다운로드</button>
<button type="button" onclick="doAction('delete')">파일삭제</button>
<button type="button" onclick="doAction(save)">저장</button>
</body>

```

[기본 형태 객체 생성]

### 2.3.5 컨텍스트메뉴 설정 및 기능 연결

업로드 컨트롤 상에서 마우스 우측버튼 클릭시 보여질 컨텍스트 메뉴를 설정하고, 사용자가 메뉴를 선택했을때 행할 기능을 연결한다.

ibsheetinfo.js 파일에 기본 컨텍스트 메뉴 내용이 설정되어 있다.

```

//초기 기본값 설정
ibleaders.ibupload = {

    /* 파일 추가시 자동 업로드 여부 */
    autoUpload: true,
    ..... 중략 .....

    /* 팝업메뉴 항목 구성 */
    contextMenuItems: {
        "download": {name: "다운로드 (D)", icon: "", accesskey: "d"},
        "sep1": "-----",
        "viewtype": {

```

```

        "name": "보기",
        "items": {
            "icon": {"name": "아이콘 (C)", accesskey: "c"},
            "list": {"name": "간단히 (L)", accesskey: "l"},
            "detail": {"name": "자세히 (D)", accesskey: "d"}
        },
        "sep2": "-----",
        "add": {name: "추가 (A)", icon: "", accesskey: "a"},
        "delete": {name: "제거 (R)", icon: "delete", accesskey: "r"}
    },
    ..... 후략 .....
}

```

[ibsheetinfo.js 파일 내용]

contextMenuItems 의 각 속성은 아래와 같이 객체별 name, name, accesskey 로 구성된다.

메뉴 항목	추가 처리 내용
객체별 name	팝업 메뉴항목 의 고유 ID 위의 예 에서 download, icon, add,delete 등에 해당된다. ( onContextMenu 이벤트 참고 )
Name	실제로 메뉴에 보이는 글 내용
Accesskey	팝업메뉴에서의 키보드를 누르면 바로 선택되는 바로가기 키를 설정한다. ( ※ 지원되는 브라우저에 한하여 지원된다. firefox 와 모바일은 지원불가 )

업로드 초기화시 해당 컨텍스트메뉴와 연결될 이벤트를 정의하면 된다.

```

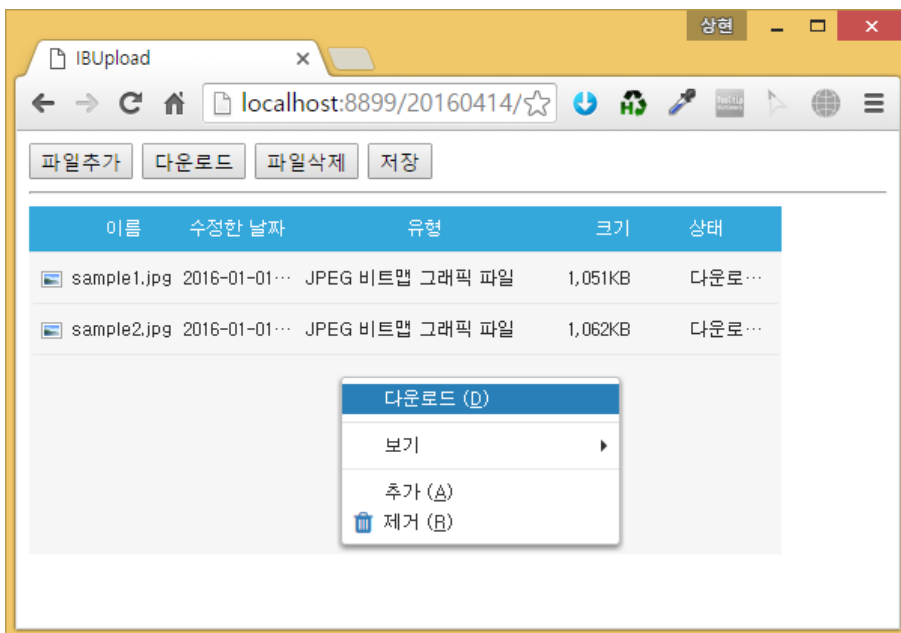
<script>
$(document).ready(function(){
    //업로드 초기화
    $("#myUpload").IBUpload({
        viewType:"icon",
        iconMode : "detail",

        ..... 중략.....

        //컨텍스트 메뉴 연결
        onContextMenu:function(key){
            doAction(key);
        }
    });});
</script>

```

[onContextMenu 이벤트 추가 (2.3.4장 doAction함수 참고) ]



[컨텍스트 메뉴 추가]

## 2.4 업로드기능 구현하기

### 2.4.1 자동/수동 업로드 설정

업로드 방식에는 **autoUpload** 속성에 따라 아래와 같이 나뉜다.

autoUpload값		기능
자동업로드	true	add 메소드의 호출시 보여지는 파일선택창에서 사용자가 파일을 선택하는 즉시 서버로 전송된다.
수동업로드	false	add 메소드의 호출로 사용자가 선택한 파일은 실제 서버로 전송되지 않고 관리되다가, upload 메소드를 호출하는 순간에서야 서버로 전송된다.

Client 파트에서 IBUpload 의 업로드 기능을 구현하기 위해서는 아래와 같이 링크 버튼을 제공하고, 링크 버튼을 사용자가 클릭했을 때 add 메소드를 호출하도록 구현한다.

#### IBUpload 삽입 내용 ( 자동 업로드 방식 )

```
<head>
  <script>
    $(function(){
      //업로드 객체 초기화
      $("#myUpload").IBUpload({
        viewType:"icon",
        iconMode : "detail",
        autoUpload:true //즉시업로드
      });
    });
```

```

    });

function doAction(sAction) {
    switch (sAction) {
        case 'add':
            $('#myUpload').IBUpload('add');
            break;
    }
}
</script>
</head>
<body>
<a href="javascript:doAction('add');">파일첨부</a>
<hr>
<!-- 업로드 상위 객체의 크기를 따른다 -->
<div style="width:750px;height:400px">
    <div id="myUpload"></div>
</div>
</body>

```

[autoUpload:true 모드]

## IBUpload 삽입 내용 ( 수동 업로드 방식 )

```

<head>
<script>
    $(function(){
        $("#myUpload").IBUpload({
            viewType:"icon",
            iconMode : "detail",
            autoUpload:false //수동 업로드
        });
    });

```

```

function doAction(sAction) {
  switch (sAction) {
    case 'add'://파일 선택
      $('#myUpload').IBUpload('add');
      break;
    case 'upload'://서버 전송
      $('#myUpload').IBUpload(upload);
      break;
  }
}
</script>
</head>
<body>
  <a href="javascript:doAction('add');">파일추가</a>
  <a href="javascript:doAction('upload');">업로드</a>
</body>

```

[autoUpload:false모드 (업로드 버튼 클릭시 서버 전송)]

Server 파트에서는 IBUpload 의 업로드 파일의 수신 기능을 구현하기 위해서 서버측으로 전송 된 파일을 저장 처리할 서버측 로직이 필요하다.

ibupload는 **upload\_sample.jsp** 파일을 통해 파일을 저장하고 어떤 결과(json)을 리턴해야 하는지 **템플릿**을 제공하고 있다. 이 파일을 참고하여 실제 프로젝트에 적당한

### 2.4.2 업로드시 파라미터 값들을 함께 전송하기

업로드 시에 `extendParamUpload` 메소드를 통하여 파일 외에 다양한 문자열 값들을 함께 전송할 수 있다.

아래의 예제와 같이 `autoUpload=false` 상태에서 `upload` 메소드를 호출하기 전에 `extendParamUpload` 메소드를 미리 호출하고 `upload` 메소드를 호출하는 순간, 서버 파트에서 해당 파라미터들의 `name` 과 `value` 를 함께 얻어볼 수 있다.

참고로, 파일 추가 없이 `upload` 메소드만 반복적으로 호출하더라도 `extendParamUpload` 메소드에 설정한 파라미터 값들은 계속 서버로 전송된다.

#### **extendParamUpload 의 예**

```
function doAction(sAction) {
    switch (sAction) {

        case 'add':
            $('#myUpload').IBUpload('add');
            break;

        case 'upload':
            $('#myUpload').IBUpload("extendParamUpload","userid=user-  
id10012&content=올해의 신청자료");
            $('#myUpload').IBUpload('upload');
            break;

    }
}
```

### 2.4.3 업로드 완료시 서버에서 응답한 파라미터 값 들을 얻기

업로드가 완료되면 서버에서 리턴한 JSON 구문은 onUploadData 이벤트에서 받아볼수 있다.

따라서 저장 결과 json안에 업로드가 필요로 하는 결과 내용 외에 클라이언트 측으로 전송하고자 하는 내용이 있다면 별도로 추가해도 된다.

```
{ "ibupload": //전송한 파일에 대한 정보
  [{"size":32927, "id":"myUpFile0_0_0","url":"myUploadedFiles#/20180823162727"}
  ....
]
, "etcddata": //서버측에서 임의의로 만들어진 데이터
  { "name" : "실험 데이터" ,
    "content" : "실험 데이터입니다." ,
    "content2" : "실험 데이터2입니다."
  }
}
```

#### [서버의 응답문의 예]

위와 같이 서버에서 "ibupload" 속성을 통해 업로드가 필요로 하는 정보 외에 etcddata라는 임의의 내용을 추가하여 전송할수 있다. 이 내용은 onUploadData 이벤트를 통해 얻어 사용할 수 있다.

```
$(function(){
  $("#myUpload").IBUpload({
    viewType:"icon",
    iconMode : "icon",
```



```
theme : "Main",  
onUploadData: function(serverObject, serverText){  
    frm.sa_name.value = serverObject.etcdata.name;  
    frm.sa_contents.value = serverObject.etcdata.content;  
return serverObject; //반드시 리턴해줘야 함.  
}  
});  
});
```

[onUploadData 이벤트의 수신 예]

## 2.5 다운로드 기능 구현하기




Client 파트에서 IBUpload 의 다운로드 기능을 구현하기 위해서는 아래와 같이 링크 버튼을 구현하고, 버튼을 클릭했을 때 IBUpload의 **download** 메소드를 호출하도록 구현한다.

```
<head>
<script>
function doAction(sAction) {
  switch (sAction) {
    case 'download':
      $('#myUpload').IBUpload('download');
  }
}
</script>
</head>

<a href="javascript:doAction('download');">다운로드</a>
```

[ download 메서드 기능 추가]

download 메서드 호출시 IBUpload 컨트롤 안에 **선택된 파일목록**이 서버로 전송된다. 서버에서는 파일정보를 얻어 클라이언트측으로 파일을 리턴하면 된다.

파일명	업로드날짜	파일종류	파일용량	업로드상태
 20180718_회의내용.txt	2018-08-23 오후 5:46	텍스트 파일	3 KB	다운로드 가능
 clipCopyMode 기능 분석, ...	2018-08-23 오후 5:46	마이크로소프트 엑셀 파일	13 KB	다운로드 가능
 기술지원팀 9월 월간회의 ...	2018-08-23 오후 5:46	마이크로소프트 엑셀 파일	13 KB	다운로드 가능

[두개 파일이 선택된 ibupload]

download 함수 호출시 선택한 파일명과 실제 파일에 대한 경로는 다음과 같이 묶여 서버로 전송된다.

```
<input type="hidden" name="file" value=" myUploadedFiles/20180823174741|20180718_회의내용.txt[줄넘김문자]myUploadedFiles/20180823174741_13945606|clipCopyMode 기능 분석.xlsx"/>
```

자세히 보면 다음과 같은 구조이다.

```
물리적 파일명1[저장시 변환한 임의의 파일명]|실제 파일명1[사용자가 아는 파일명]줄넘김
물리적 파일명2|실제 파일명2줄넘김
.....
```

#### [file input 객체의 value]

따라서 서버측에서는 줄넘김 문자(Wn)를 통해 파일과 파일은 나누고 "|"를 통해 다시 물리적 파일명과 실제 파일명을 얻어 클라이언트로 리턴해 주면 된다.

ibupload에서는 **download\_sample.jsp** 파일을 통해 다운로드 시 서버측 로직에 대한 템플릿을 제공하고 있다.

## 2.6 오류 처리

### Client 오류 메시지 처리

업로드, 다운로드 시 다양한 오류 (또는 정보) 들이 발생할 수 있는데 이와 관련한 이벤트가 onMessage 이벤트이다. (※자세한 오류 코드는 API 레퍼런스 - 메시지 코드 를 참고 )

오류 메시지로는 ERR-001 과 같이 오류 형태의 코드가 있고, INFO-001 과 같이 정보 메시지가 있으므로 오류만 필터링하여 아래와 같이 alert 를 띄울 수 있다.

```
<head>
  <script>
    $(function(){
      $("#myUpload").IBUpload({
        viewType:"icon",
        iconMode : "detail",
        onMessage: function(msgID, msg){
          if (msgID.substring(0, 3) == "ERR") {
            // 오류는 메시지로 표시
            alert(msg + "\n\n" + "error : " + msgID);
          } else {
            //alert(msg); // 일반 정보 메시지는 숨김
          }
        }
      });
    });
  </script>
</head>
```

[오류와 정보 메시지를 분리하여 alert 띄우기]

### Servelet Exception 처리

Servelet에서 Exception 처리 하기 위해서는 아래와 같이 구현할 수 있다.

Exception 관련 에러 메시지는 Json 형식으로 아래와 같이 구성된다.

- Json 형식 : {"error":"오류가 발생하였습니다."}

```
try {

} catch (Exception ex) {

    Map msg = new HashMap();
    msg.put("error","오류가 발생하였습니다.");
    //오류에 대한 자세한 내역이 필요하다면 다음과 같이 사용하세요.
    //String exMsg = ex.getMessage();
    //msg.put("error",exMsg);
    out.print(org.json.simple.JSONValue.toJSONString(msg));
}
```

[Servelet에서 Exception 처리] (jsonSimple lib 사용)

## 2.7 조회 기능 구현하기

IBUpload 의 files 속성을 이용하면, 로딩되는 순간부터 즉시 다운로드 받을 파일들을 컨트롤에 추가하여 표시할 수 있다.

```
<head>
  <script>
    $(function(){
      $("#myUpload").IBUpload({
        viewType:"icon",
        iconMode : "icon",
        theme : "Main",
        files: [
          {name: "오렌지.jpg", size: "1075761", date:
            "20160101125959", url: "20160126_180801"},
          {name: "커피2랑.jpg", size: "1087426", date:
            "20160101125959", url: "20160126_180801_64356079"}
        ]
      });
    });
  </script>
</head>
```

[**IBUpload** 초기 화면부터 다운로드 받을 파일들을 삽입하기]

업로드에 조회 데이터를 표시하기 위해서는 json 구조에 다음과 같은 속성이 필요하다.

name	실제 파일명
size	파일 크기

date	업로드 일자 (yyyyMMddHHmmss)
url	서버에 저장된 실제파일명

ibupload가 생성된 이후에 파일을 조회 할때는 files 메서드를 이용하면 된다.

```
<script>
$.ajax({
  url:"thisPageSearch.do"
  ,data:"pid=12342&user_id=928123"
  ,success:function(jsonData){
    //서버에서 조회한 json 데이터를 로딩한다.
    $("#myUpload").IBUpload("files" , jsonData);
  }
});
</script>
```

files 속성에 대한 구체적인 사항은 files 속성이나 files 메소드를 참고한다.

# CHAPTER 3.

## 3 API 레퍼런스



## A. 속성

속성 값은 IBUpload 초기화 과정에서 사용할 수 있다.

따라서 업로드 객체가 생성된 이후에는 속성을 변경할 수 없다. 단 downloadServerUrl, uploadServerUrl 과 같이 함수를 통해 다시 설정할 수 있는 속성은 변경할 수 있다.

```
//IBUpload 초기화 과정에서 속성 사용
$("#myUpload").IBUpload({
  iconMode : "icon", //보기 모드 (icon,list,detail)
  theme : "Main", //기본 테마 폴더
  autoUpload: false, //자동 업로드 기능 사용 여부
  files:[
    {name: "관심과집중.mp4", size: "11417124", date: "20160101125959", url:
"20160126_180337_82754651"}
    ,{name: "오렌지.jpg", size: "1075761", date: "20160101125959", url: "20160126_180801"}
  ]
});
```

초기화 과정에서 설정하지 않은 속성에 대한 default 값은 `ibuploadinfo.js` 안에 `ibleaders.ibupload` 변수의 내용을 따른다.

## addTrigger 속성

IE9 환경에서 파일 추가 다이얼로그 창을 클라이언트에서 생성한다.

### ➤ 문법

```
$(Selector).IBUpload({
    addTrigger : "String" // 파일 추가 버튼 DOM id
});
```

### ➤ 상세설명

IE9에서 기존 서버와 iframe통신을 활용한 파일 추가 다이얼로그 창을 호출하는 방식과 달리 오직 클라이언트를 통해 생성한다.

### ➤ 정보

이름	자료형	기본값	설명
addTrigger	String	addItem	파일 추가 버튼의 DOM id

### ➤ 버전

7.3.0.56

### ➤ 관련기능

- add 메소드

## autoUpload 속성

파일을 추가하는 즉시 자동으로 파일 업로드를 수행할지 여부를 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    autoUpload : booleanValue
});
```

### ➤ 상세설명

파일을 추가하는 즉시 자동으로 파일을 업로드할 지의 여부를 설정한다. 값에 따라 아래와 같이 자동 업로드 와 수동 업로드 로 나뉜다.

### ➤ 정보

이름	자료형	기본값	설명
autoUpload	boolean	true	<p>true일 경우 add 메소드를 호출하여 사용자가 파일을 선택하는 즉시 서버로 파일들이 업로드 된다.</p> <p>false일 경우 add 메소드를 호출하여 사용자가 파일을 선택 하더라도 일단 대기 상태로 있고, 반복적으로 add 를 추가 호출하여 파일들을 계속 추가 누적할 수 있다. upload 메소드를 호출하는 순</p>

			간, 서버로 파일들이 한꺼번에 업로드 된다.
--	--	--	--------------------------

➤ 버전

7.3.0.0

## contextMenuItems 속성

마우스 커서가 IBUpload 컨트롤 위에 있을때 마우스 오른쪽 버튼 클릭을 통해 보여질 팝업메뉴를 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
  contextMenuItems: {
    "download": {name: "다운로드 (D)", icon: "", accesskey: "d"},
    "sep1": "-----",
    "viewtype": {
      "name": "보기",
      "items": {
        "icon": {"name": "아이콘 (C)", accesskey: "c"},
        "list": {"name": "간단히 (L)", accesskey: "l"},
        "detail": {"name": "자세히 (D)", accesskey: "d"}
      }
    },
    "sep2": "-----",
    "add": {name: "추가 (A)", icon: "", accesskey: "a"},
    "delete": {name: "제거 (R)", icon: "delete", accesskey: "r"}
  }
});
```

### ➤ 상세설명

마우스 오른쪽 버튼을 통해 보여질 내용에 대하여 구성한다.

json 형식으로 구성되며 내용은 아래와 같다.

```
contextMenuItems :{
  key1 :{name:"고양이과 동물"
    ,items:{
      key2:{name:"호랑이",icon:"add",accesskey:"i"}
      ,key3:{name:"표범",icon:"delete",accesskey:"i"}
    }
  }
}
```

```

    } //end key,
}

```

### ➤ 정보

이름	자료형	설명
key	String	onContextMenu 이벤트를 통해 리턴되는 값.
name	String	마우스 우측버튼 클릭시 보여지는 타이틀
icon	String	add,delete 에 대한 이미지 아이콘 표시 여부
items	JSON	내부 메뉴 표현시 사용
accesskey	String	단축키(알파벳) 설정

### ➤ 버전

7.3.0.0

## downloadFileName 속성

2건 이상의 파일 다운로드 시 zip로 파일을 받게 된다. 이때의 zip파일의 이름을 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    downloadFileName: stringValue
});
```

### ➤ 상세설명

2건 이상의 파일을 다운로드시 파일의 이름을 설정할 수 있다.

### ➤ 예제 #1

```
<script type="text/javascript">
$(function(){
    $("#myUpload").IBUpload({
        viewType:"icon",
        autoUpload:false,
        downloadFileName :
            (function() {
                var yyyy = new Date().getFullYear().toString();
                var mm = (new Date().getMonth()+1).toString();
                var dd = new Date().getDate().toString();
                return yyyy + (mm[1]?mm:"0"+mm[0]) + (dd[1]?dd:"0"+dd[0]);})(); + "_다운",
        theme : "Main"
    });
});
```

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

➤ 정보

이름	자료형	기본값	설명
downloadFileName	stringValue	download.zip	Zip 파일 이름을 설정한다.

➤ 버전

7.3.0.0



## downloadServerUrl 속성

다운로드 기능을 제공하는 서버 쪽 jsp 프로그램의 경로를 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    uploadServerUrl:URL,
    downloadServerUrl:URL
});
```

### ➤ 상세설명

다운로드 기능을 제공하는 서버 쪽 jsp 프로그램의 경로를 설정한다.

ibuploadinfo.js 안에서 ibleaders.ibupload.downloadServerUrl 전역 속성으로 최초에 1회만 설정할 수 있고, downloadServerUrl 메소드를 통해서도 다운로드 직전에 해당 경로를 변경 설정할 수 있다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- uploadServerUrl 메소드

## exceptDuplicated 속성

중복 파일 추가 방지 여부를 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({  
    exceptDuplicated: numberValue,  
});
```

### ➤ 상세설명

중복 파일이 컨트롤 내에 추가하는 것이 방지 가능한지 여부를 설정한다.

0으로 설정한 경우 중복파일 추가가 가능하며, 1로 설정시 불가능하다. 기본값은 1이다.

### ➤ 버전

7.3.0.50

### ➤ 관련기능

- add 메소드

## exceptZeroFile 속성

크기가 0byte인 파일 추가 방지 여부 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({  
    exceptZeroFile: numberValue,  
});
```

### ➤ 상세설명

크기가 0byte인 파일을 컨트롤 내에 추가 할지 여부를 설정한다.

0으로 설정한 경우 0byte 크기 파일 추가를 시도하며, 1로 설정시 불가능하다. 기본값은 0이다.

### ➤ 버전

7.3.0.52

### ➤ 관련기능

- add 메소드

## files 속성

초기 로딩시 컨트롤 내에 추가할 파일 목록들을 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    files : fileArrayObject or stringValue
});
```

### ➤ 상세설명

초기 로딩시 컨트롤 내에 추가할 파일 목록들을 설정한다.

### ➤ 예제 #1

```
<script type="text/javascript">
$(function(){
    // 엔진 생성

    $("#myUpload").IBUpload({
        viewType:"icon",
        iconMode:"icon",
        autoUpload:false,
        files: [
            {name: "관심과집중.mp4", size: "11417124", date: "20160101125959", url:
"20160126_180337_82754651"},
            {name: "오렌지.jpg", size: "1075761", date: "20160101125959", url:
"20160126_180801"}
```

```

    ],
    theme : "Main"
  });
});
</script>

```

## ➤ 예제 #2

```

<script type="text/javascript">
$(function(){
    // 엔진 생성
    $("#myUpload").IBUpload({
        viewType:"icon",
        iconMode:"icon",
        autoUpload:false,
        files: '{name: "관심과집중.mp4", size: "11417124", date: "20160101125959",
url: "20160126_180337_82754651"},{name: "오렌지.jpg", size: "1075761", date:
"20160101125959", url: "20160126_180801"}',
        theme : "Main"
    });
});
</script>

```

File 객체의 구성 요소

이름	자료형	기본값	설명
date	String		YYYYMMDDhhmmss(년월일시분)

			초) 형식으로 구성된 문자열 업로드했던 날짜와 시간
name	String		업로드 했던 당시의 파일 원본 명칭. 폴더 명이 제외된 순수한 파일명.
size	Number		업로드 파일의 용량 (Byte 단위)
url	String		파일 업로드가 완료된 경우 서버로부터 다운로드 가능한 URL

➤ 버전

7.3.0.0

➤ 관련기능

- files 메소드

## fileSeparator 전역속성

여러개 파일을 선택하여 다운로드시 서버로 전달되는 파일 정보와 파일명 사이의 구분자를 설정한다.

### ➤ 문법

```
ibleaders.ibupload = {
    fileSeparator : "|"
};
```

### ➤ 상세설명

파일 다운로드시 선택한 파일 명은 file이라는 이름으로 서버로 전송되는데, 설정이 없는 경우 파일명과 파일명 사이에 구분자로 " "(공백)문자가 들어간다.

파일명 안에 공백이 있는 경우 문제가 되기 때문에 이 문자를 공백 대신에 다른 문자로 변경할 수 있다.

ibuploadinfo.js에서 설정가능하다. 1개의 문자만 설정 가능하고 /,\는 사용할 수 없다.

특별한 경우가 아니면 그냥 ibsheetinfo.js 에 정의된 "|"를 사용할 것을 권장함.

### ➤ 버전

7.3.0.49

### ➤ 관련기능

- download

## headerText 속성

viewType이 detail 모드에서 보여지는 컬럼의 순서를 바꾸거나 숨긴다.

### ➤ 문법

```
$(Selector).IBUpload({
  headerText : fileArrayObject
});
```

### ➤ 상세설명

detail 모드에서 보여지는 컬럼의 순서를 바꾸거나 숨긴다.

전역으로 설정시 모든 IBUpload 컨트롤에 컬럼 숨김과 순서가 동일하게 적용되며, 지역으로 특정 IBUpload 에서만 설정하면 해당 컨트롤의 디자인만 적용된다.

모두 선언하지 않으면 ibuploadinfo.js 안에 있는 ibleaders.ibupload.message 의 값들로 적용된다.

해당 속성들이 곧 표시되는 컬럼의 순서이며, 일부 속성을 지우면 화면에서도 해당 컬럼이 숨겨진다.

### ➤ 예제 #1 - ibuploadinfo.js 의 전역속성에서 선언시 모든 IBUpload 컨트롤에 적용된다.

```
ibleaders.ibupload = {
  headerText : {"icon32":"","icon16":"","name" : "파일명", "size": "파일 용량", "date": "날짜", "state" : "상태", "type" : "파일 유형"},
```



```

:
:
}

```

➤ **예제 #2 – 지역설정시 해당 IBUpload 컨트롤에만 적용**

```

<script type="text/javascript">
$(function(){
    // 엔진 생성
    $("#myUpload").IBUpload({
        headerText : {"icon32":"","icon16":"","name" : "파일명", "size": "파일 용량"},
        :
        :
    });
});
</script>

```

headerText 의 구성 요소

이름	자료형	기본값	설명
icon32	String	undefined	viewType 속성에서 icon / list 모드와 관련하여 큰 아이콘들을 표시하려면 icon32 를 반드시 선언해야 한다. 아이콘을 숨기고 싶으면 선언을 하지 않는다. (name 설정 자체가 중요하며, 값은 의미 없음)

icon16	String	undefined	detail 모드에서 아이콘을 표시하려면 icon16 를 선언해야 한다. 아이콘을 숨기고 싶으면 선언을 하지 않는다. (name 설정 자체가 중요하며, 값은 의미 없음)
name	String	undefined	파일명 컬럼 - 파일명을 표시하는 컬럼의 헤더글자와 보임여부 설정 ( 값은 눈에 보이는 헤더 글자이며, 선언시 컬럼이 보이고, 선언을 하지 않으면 해당 컬럼은 감춰진다. 이하 동일 )
size	String	undefined	파일크기 컬럼 - 파일 크기를 표시하는 컬럼의 헤더글자와 보임여부 설정
date	String	undefined	업로드 날짜 컬럼 - 파일을 업로드하는 컬럼의 헤더글자와 보임여부 설정
state	String	undefined	파일상태 컬럼 - 파일의 상태를 표시하는 컬럼의 헤더글자와 보임여부 설정
type	String	undefined	파일유형 컬럼 - 파일의 종류를 표시하는 컬럼의 헤더글자와 보임여부 설정

## ➤ 버전

7.3.0.20

## iconMode 속성

컨트롤 내부의 생성 모양을 설정한다.

### ➤ 문법

```
$(Selector).IBUpload(  
    iconMode: stringValue  
);
```

### ➤ 상세설명

아이콘 타입의 업로드 컨트롤 생성시 타입을 설정한다. 제공되는 모드는 icon, list, detail 형태이며 각 모드별 사용 변경은 css를 통해 컨트롤 가능하다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- viewType 속성

## limitFileCount 속성

컨트롤 내에 추가할 파일의 최대 개수를 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    limitFileCount : numberValue
});
```

### ➤ 상세설명

컨트롤 내에 추가할 파일의 최대 개수를 설정한다.

동시에 N 개의 파일을 추가 시도 중에 최대 개수를 초과하게 되면 해당 N 개의 멀티 파일 선택 목록 전체가 추가 되지 않고 취소 된다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- limitFileCountOnce 속성
- limitFileExtMode 속성

## limitFileCountOnce 속성

컨트롤 내에 한번에 추가할 파일의 최대 개수를 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({  
    limitFileCountOnce: numberValue  
});
```

### ➤ 상세설명

컨트롤 내에 한번에 추가할 파일의 최대 개수를 설정한다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

## limitFileSize속성

컨트롤 내에 추가할 파일의 최대 제한 용량을 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    limitFileSize: numberValue
});
```

### ➤ 상세설명

컨트롤 내에 추가할 1개 파일의 최대 제한 용량을 Byte 단위로 얻거나 설정한다.

동시에 N 개의 파일을 추가 시도 중에 최대 제한 용량을 초과하는 파일이 1개라도 발생하면 해당 N 개의 멀티 파일 선택 목록 전체가 추가 되지 않고 취소 된다.

※ 파일의 사이즈,확장자를 제어하는 것은 사용자의 편의를 돕기 위한 기능이며, 보안적인 측면에서 파일에 대한 필터링은 서버측 로직에 반드시 구현되어야 함

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- limitFileTotalSize 속성

## limitFileTotalSize속성

컨트롤 내에 추가할 모든 파일들의 총 합계의 최대 제한 용량을 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    limitFileTotalSize: numberValue
});
```

### ➤ 상세설명

컨트롤 내에 추가할 모든 파일들의 총 합계의 최대 제한 용량을 Byte 단위로 설정한다. 동시에 N 개의 파일을 추가 시도 중에 총 합계의 최대 제한 용량을 초과하는 파일이 1개라도 발생하면 해당 N 개의 멀티 파일 선택 목록 전체가 추가 되지 않고 취소 된다.

※ 파일의 사이즈,확장자를 제어하는 것은 사용자의 편의를 돕기 위한 기능이며, 보안적인 측면에서 파일에 대한 필터링은 서버측 로직에 반드시 구현되어야 함

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- limitFileSize 속성
- limitFileCount 속성

## limitFileExt 속성

컨트롤 내에 제약할 수 있는 파일의 확장자를 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    limitFileExt: "파일 확장자 [,파일 확장자]",
    limitFileExtMode : "allow" 또는 "deny"
});
```

### ➤ 상세설명

컨트롤 내에 허용 또는 거부할 파일의 확장자들을 설정한다.

1 개 이상의 확장자를 나열하려면 , (coma구분자) 를 이용하여 나열한다.

※ 파일의 사이즈,확장자를 제어하는 것은 사용자의 편의를 돕기 위한 기능이며, 보안적인 측면에서 파일에 대한 필터링은 서버측 로직에 반드시 구현되어야 함

### ➤ 예제

```
$(Selector).IBUpload({
    limitFileExt: "jpg, png, gif", //제약 조건 확장자
    limitFileExtMode : "allow" // jpg, gif, png 파일만 업로드를 허용한다.
});
```

### ➤ 버전

7.3.0.0

### ➤ 관련기능



- limitFileExtMode 속성

## limitFileExtMode 속성

limitFileExt 로 나열한 확장자 파일들만 허용할 지, 거부할 지를 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    limitFileExt: "파일확장자 [,파일확장자]",
    limitFileExtMode : "allow" 또는 "deny" 또는 "accept"
});
```

### ➤ 상세설명

limitFileExt에 설정한 확장자를 허용(allow)할지, 거부(deny)할지를 설정한다.

accept를 설정 할 경우 파일 선택창에서 limitFileExt에 설정한 목록만 보이고, 모든 파일(\*.\*)을 선택 할 경우 allow로 동작한다.

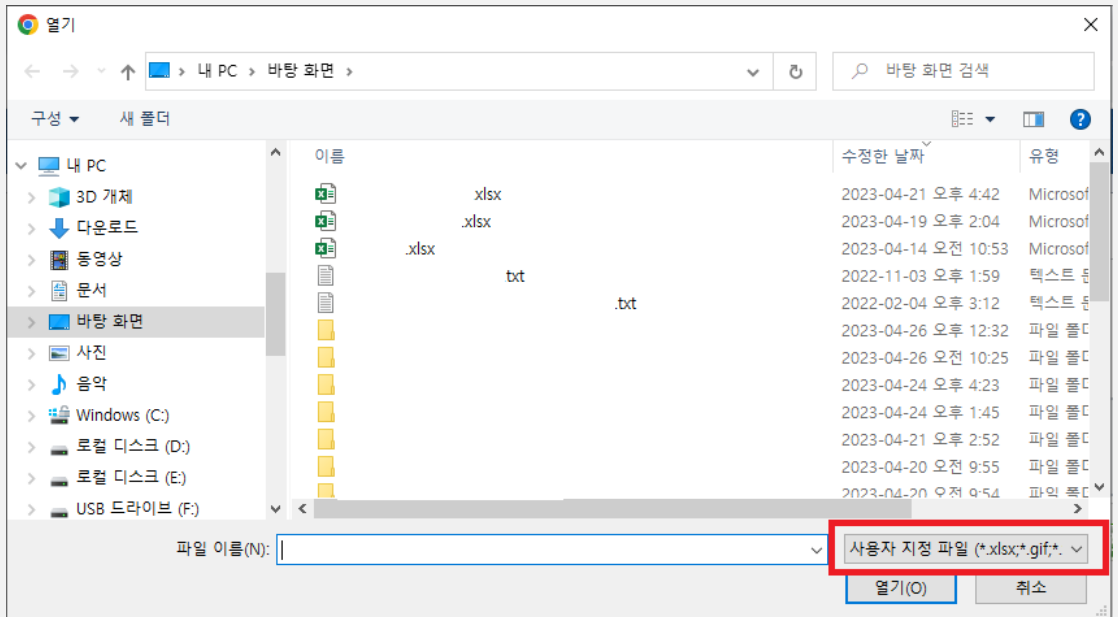
<input type="file" accept=".gif, .jpg, .png">와 같은 기능이므로 Internet Explorer 10 브라우저 이상에서 사용가능하다.

※ 파일의 사이즈,확장자를 제어하는 것은 사용자의 편의를 돕기 위한 기능이며, 보안적인 측면에서 파일에 대한 필터링은 서버측 로직에 반드시 구현되어야 함

### ➤ 예제

```
$(Selector).IBUpload({
    limitFileExt: "jsp, asp, php, class",
    limitFileExtMode : "deny" // jsp, asp, php, class 파일을 제외하고 업로드 한다.
});
```

```
$(Selector).IBUpload({
    limitFileExt: "xlsx, gif, txt"
    limitFileExtMode : "accept" //파일 선택창에서 xlsx, gif, txt 파일만 필터링 된다.
});
```



## ➤ 버전

7.3.0.0

7.3.0.70 accept 기능추가

## ➤ 관련기능

- limitFileExt 속성

## overWriteFile 속성

중복 파일 덮어쓸지 여부를 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({  
    overWriteFile: numberValue  
});
```

### ➤ 상세설명

컨트롤 내에 추가할 파일의 중복 여부를 설정한다.

1로 설정 시, 기존 파일을 삭제하고 파일 추가를 실행한다.

### ➤ 버전

7.3.0.52

### ➤ 관련기능

- add 메소드

## prependExtendParam 속성

extendParamUpload 메소드로 설정된 추가 파라미터들이 폼 데이터의 첫 부분에 생성되어 전송된다.

### ➤ 문법

```
$(Selector).IBUpload({
    prependExtendParam: numberValue
});
```

### ➤ 상세설명

prependExtendParam 속성을 true로 설정 시 extendParamUpload 메소드로 설정된 추가 파라미터들이 폼 데이터의 첫 부분에 생성되어 전송된다.

해당 속성이 false거나 미설정 시 추가 파라미터들은 폼 데이터의 마지막에 생성되어 전송된다.

### ➤ 버전

7.3.0.63

### ➤ 관련기능

- extendParamUpload 메소드

## submitNoData 속성

업로드할 파일이 없는 경우 서버에 request 전송 여부.

### ➤ 문법

```
$(Selector).IBUpload({  
    submitNoData: numberValue  
});
```

### ➤ 상세설명

업로드할 파일이 없는 경우 서버에 request 전송 여부를 설정합니다.

0으로 설정 시 업로드 또는 삭제할 파일이 없는 경우 서버로의 upload 명령을 무시합니다. 기본값은 1이다.

### ➤ 버전

7.3.0.52

### ➤ 관련기능

- upload 메소드

## target 속성

폼(form)을 전송할 타겟을 지정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    target: stringValue
});
```

### ➤ 상세설명

IBUpload 컨트롤은 내부적으로 별도의 IFRAME을 사용하여 파일 업로드를 처리하므로 target이 특정 IFRAME의 name으로 설정되어 있다. 만약 화면 전체를 reload 되도록 작업할 경우 target을 \_self로 지정할 수 있다.

만약, IBUpload 컨트롤과 다른 폼 항목을 동시에 작업을 원할 경우 form.submit() 호출 대신 \$(Selector).IBUpload("upload")를 호출하여 해당 페이지 처리해야 한다.

### ➤ 예제

```
$( "#myUpload").IBUpload({
    target: "_self",
});

function doAction(sAction, arg) {
    switch (sAction) {
        case 'save':
            $('#myUpload').IBUpload('extendParamUpload',FormQueryStringEnc
```

```
(document.myForm,false));  
$('#myUpload').IBUpload('upload');  
    break;  
}
```

- 버전  
7.3.0.0
- 관련기능



## theme 속성

IBUpload의 viewType이 icon일 경우 테마 디자인을 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    theme: stringValue
});
```

### ➤ 상세설명

이름	자료형	기본값	설명
theme	string	Main	설정하기 위해서는 세부사항 작업 필요 (viewType이 ibsheet일 경우 info에서 sheet생성시 테마 추가 및 수정 필요)

1. Main 폴더를 하나 다른 이름으로 복사한다.
2. 폴더안에 각종 버튼 이미지를 개발하는 프로젝트 테마에 맞게 변경한다.
3. ibupload.css파일 참고하여 작성해야 한다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

## uploadServerUrl 속성

업로드 기능을 제공하는 서버 쪽 jsp 프로그램의 경로를 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    uploadServerUrl:URL(stringValue),
    downloadServerUrl:URL(stringValue)
});
```

### ➤ 상세설명

업로드 기능을 제공하는 서버 쪽 jsp 프로그램의 경로를 설정한다.

uploadServerUrl 속성은 제품 초기화로 최초에 1회만 실행되지만, uploadServerUrl 메소드는 업로드 직전에 해당 경로를 변경할 수 있다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- uploadServerUrl 메소드

## useDragDrop 속성

윈도우즈 탐색기로부터 Drag-Drop 기능을 사용하여 업로드 컨트롤에 파일을 포함할 수 있는지의 여부를 설정한다.

### ➤ 문법

```
$(Selector).IBUpload({
    useDragDrop:Boolean });
```

### ➤ 상세설명

윈도우즈 탐색기로부터 Drag-Drop 기능을 사용하여 업로드 컨트롤에 파일을 포함할 수 있는지의 여부를 설정한다.

단, IE10 이상에서만 가능하다.

또한, 윈도우 탐색기와 브라우저에서의 권한이 서로 다른 경우에는 사용자의 윈도우즈 환경적, 보안적 특성상의 이유에 따라 드래그 드랍이 불가능할 수도 있다.

### ➤ 버전

7.3.0.17

### ➤ 관련기능

- add 메소드

## locale 전역속성

다국어 설정을 변경한다.

### ➤ 문법

```
// ibuploadinfo.js 안에서 설정 (only 전역설정)
ibleaders.ibupload = {
    locale: stringValue,
    :
};
```

### ➤ 상세설명

IBUpload 컨트롤의 표시되는 텍스트 들과 동작 중에 발생하는 모든 정보, 메세지, 오류 들에 대하여 표시할 언어를 설정한다.

기본값은 Korean 이며, English 로 변경할 수 있다.

전역 속성으로만 설정이 가능하지만, IBUpload 컨트롤을 로딩하기 전에 ibleaders.ibupload.locale 값을 변경하면 지역 속성을 설정한 것처럼 해당 컨트롤만 일시적으로 언어설정을 변경하여 표시할 수도 있다.

만약 중국어나 일본어등 언어를 추가하려면 ibupload.message와 ibupload.fileType 안에 각각 Korean 과 English 외에 더 추가하고 싶은 항목들을 정해서 추가하고 이 이름을 locale 값에 설정하면 된다.

참고로, 팝업메뉴를 사용하는 경우에는 IBUpload\_GetContextMenu() 함수의 로직에도 추가할 언어 항목을 반영해야 하며, ibsheet 모드를 사용할 경우에는 IBSheetLoadPage\_Main() 함수 안에서도 헤더에 표시될 내용을 분기처리 해 주어야 한

다.

➤ **예제**

```
ibleaders.ibupload = {  
    locale: "english", // 한글에서 영문으로 전환한다.  
};
```

➤ **버전**

7.3.0.22

## msgLevel 전역속성

onMessage 이벤트를 발생시킬 메세지 레벨을 설정한다.

### ➤ 문법

```
// ibuploadinfo.js 안에서 설정 (only 전역설정)
ibleaders.ibupload = {
  msgLevel: "INFO" //INFO or ERR
};
```

### ➤ 상세설명

IBUpload에서 파일을 추가하거나 삭제, 혹은 서버로 전송하고, 결과는 받는 모든 과정에 대해서 onMessage 이벤트가 발생하게 되는데, 이 이벤트를 발생시키는 수준을 결정한다.

"INFO"로 설정시에는 위 모든 경우에 대해 onMessage 이벤트가 발생하고, "ERR"로 설정하면 오류가 발생하는 경우(업로드 불가능한 파일을 선택했다거나 지정된 파일보다 많은 파일을 추가하는 등)에만 onMessage 이벤트가 발생한다.

### ➤ 예제

```
ibleaders.ibupload = {
  msgLevel: "ERR", //오류가 발생했을 경우에만 onMessage 이벤트가 발생한다.
  ...
};
```

### ➤ 버전

### 7.3.0.X

➤ **관련기능**

[onMessage 이벤트](#)

[msgLevel 전역속성](#)

[E. 메세지 코드](#)

## useDownloadCookie 전역속성

파일에 다운로드 완료 후, onMessage 이벤트를 발생시킨다..

### ➤ 문법

```
ibleaders.ibupload = {
    useDownloadCookie:true
};
```

### ➤ 상세설명

ibuploadinfo.js에 useDownloadCookie를 설정하면 파일이 다운로드 되는 과정에서 서버에서 cookie값을 변경시켜, 화면에 다운로드 완료이벤트를 발생시킨다.

이 기능을 사용하려면 서버에서 cookie 사용이 가능해야 한다.

(jeus 서버의 경우 cookie사용이 안되는 경우가 많으니 확인 필요)

**onMessage**이벤트의 발생여부는 **msgLevel** 전역속성의 값을 따릅니다.

### ➤ 예제 - ibuploadinfo.js 파일의 설정 내용

```
//다운로드 완료시 onMessage 이벤트 발생.
ibleaders.ibupload={
    useDownloadCookie : true
};
$("#myUpload").IBUpload({
    onMessage:function(id,msg,org){
        console.log(id,msg,org);
```



```
}  
});
```

➤ **결과**

콘솔에 출력된 내용

INFO-012 다운로드가 완료되었습니다.

➤ **버전**

7.3.0.?

➤ **관련기능**

msgLevel 전역속성

## userAgent 전역속성

조회나 저장시 request header 정보에 포함되어 넘어가는 값을 설정한다.

### ➤ 문법

```
ibleaders.ibupload = {
    userAgent:[{name:value},{name2:value2}...]
};
```

### ➤ 상세설명

ibuploadinfo.js에 userAgent를 설정하면 서버로 파일 전송시 헤더 정보에 지정한 문자가 포함되어 전달된다.

### ➤ 예제 - ibuploadinfo.js 파일의 설정 내용

```
// 컨트롤의 파일들 중, 선택한 파일들만 제거한다.

ibleaders.ibupload={
userAgent : [{ "AjaxComponent":"IBUpload" },{ "Say":"Hello!!!" } ]
};
```

➤ 예제 - 서버파트(jsp 예)

```
// 헤더 정보를 가져온다.  
System.out.println(request.getHeader("Say")); //Hello!!!
```

➤ 버전

7.3.0.4

➤ 관련기능

## userInputName 전역속성

서버로 전송하는 파일의 name 을 기본값 (file) 대신 다른 이름으로 설정하여 전송한다.

### ➤ 문법

```
// ibuploadinfo.js 안에서 설정 (only 전역설정)
ibleaders.ibupload = {
    userInputName: stringValue,
    :
};
```

### ➤ 상세설명

ibuploadinfo.js에 userInputName 을 설정하면 서버로 파일을 업로드할 때에 아래와 같이 <input> 태그의 이름이 각각 다르게 변경되어 전송된다.

아래의 예제는 사용자가 총 두 차례로 파일 추가를 실행 (첫 번째에는 1개의 파일만 선택, 두 번째에는 멀티로 2개를 동시 선택하여 추가하여 총 3개의 파일을 추가 하였음) 한 뒤에 서버로 업로드 했을 때 전송된 자료의 모습이다.

( userInputName 미설정시 - name 이 항상 'file' 로 name 이 고정적이다. )

```
<input type="file" name="file">
<input type="file" name="file">
<input type="file" name="file">
```

(userInputName = "myFile" 설정시 - 설정한 값대로 'myFile' 로 고정적이다.

```
<input type="file" name="myfile">
<input type="file" name="myfile">
<input type="file" name="myfile">
```

(userInputName = "myFile#" 설정시 - name 마다 각각 고유한 숫자가 붙는다.)

```
<input type="file" name="myfile0_0_0">
<input type="file" name="myfile0_1_0">
<input type="file" name="myfile0_1_1">
```

본 속성은 버전 7.3.0.19 이후의 버전에서 제공되는 기능이며, 7.3.0.18 이전 버전에서는 userInputName 를 사용할 수 없다.

7.3.0.1 ~ 7.3.0.18 버전에서 제공된 ibupload.jsp 로 운영되는 사이트에서는 ibuploadinfo.js 안에서 userInputName 이 선언된 경우 주석으로 가려야만 정상적으로 동작한다.

- 버전  
7.3.0.19
- 관련기능



## B. 함수

함수는 IBUpload 컨트롤이 생성된 이후 파일을 추가하거나 다운로드 등의 기능을 수행하기 위해 호출된다.

함수를 사용하는 경우 속성을 통해 정의한 일부 내용을 변경할 수도 있다.

```
//파일 추가 함수 호출
$('#myUpload').IBUpload('add');

//iconMode 변경 함수 호출
$('#myUpload').IBUpload('iconMode','detail');
```

## add 메소드

업로드할 파일 선택 창을 띄워서 파일을 선택하면 업로드 처리를 진행한다.

### ➤ 문법

```
$(Selector).IBUpload("add" [,object ]); // Method
```

### ➤ 상세설명

업로드할 파일 선택 창을 띄워서 파일을 선택하면 업로드 처리를 진행한다.

브라우저의 지원 여부에 따라 파일을 동시에 1 개 또는 여러 개를 동시에 선택하여 업로드 할 수 있다.

add 메소드와 upload 메소드는 첫번째 인자로 IBUpload 컨트롤의 초기 생성 객체를 재설정하는 기회를 제공한다. 즉, add 또는 upload 메소드가 호출되고 난 직후, IBUpload 의 모든 속성과 모든 이벤트들 중 일부를 재설정 할 수 있다는 뜻이다.

(단, 초기화 로딩 시점에서 화면에 표시와 관련되는 iconMode 나 theme 속성은 여기에서 재설정한다고 즉시 적용되는 것은 아니며, 이 설정 시점 이후로 동작에 영향을 주는 limitFileCount 등의 모든 속성과 onMessage 이벤트등의 모든 이벤트들은 재설정이 가능하다.)

또한, 첫번째 인자 안에 callback 속성에 함수를 선언하면 함수가 호출된 직후의 로직을 추가할 수 있으며 다음과 같은 순서로 콜백함수와 관련 이벤트가 발생한다.



순서	처리 내용	설명
1	onAddFile 이벤트 발생	가장 먼저 onAddFile 이벤트가 발생한다
2	callback 함수 실행	callback 함수를 실행한다.
3	업로드 실행	autoUpload=T 인 경우 즉시 업로드 전송 호출을 한다.
4	onUploadData 이벤트 발생	서버로부터 응답이 온다.
5	onUploadFinish 이벤트 발생	업로드가 모두 완료된다.

### ➤ 예제

```

$('#myUpload').IBUpload('add', {
  onAddFile: function(files){
    alert('파일 추가 로직이 변경되었습니다 ');
  },
  callback: function(){
    alert('파일을 추가 합니다. ');
  }
});

```

### ➤ 버전

7.3.0.0

7.3.0.21 – 속성과 이벤트의 재정의 기능

## addDropFile 메소드

드래그-드랍 이벤트를 이용하여 IBUpload 컨트롤로 파일들을 추가한다.

### ➤ 문법

```
$(Selector).IBUpload("addDropFile", event); // Method
```

### ➤ 상세설명

드래그-드랍 이벤트를 이용하여 IBUpload 컨트롤로 파일들을 추가한다.

특정 DIV 태그에 drop 이벤트를 걸어두어 사용자가 해당 영역에 드래그-드랍을 통한 파일들을 추가하게 되면, addDropFile 의 메소드에 event 를 인자로 전달해 주어, 사용자가 드랍한 파일들을 모두 IBUpload 컨트롤로 추가하게 된다.

useDragDrop 속성을 true 로 사용하면 이미 자체 컨트롤에서 이러한 기능이 동작하게 되므로 파일이 중복 추가 현상이 나타날 수 있다. 따라서 useDragDrop 속성은 반드시 false 로 설정해야 정상 동작한다.

아래의 예제와 같이 특정 DIV 태그와 관련해서 최소한 3개의 이벤트(dragenter, dragover, drop )는 반드시 걸어두어야 한다.

## ➤ 예제

```

$("#userDropDiv").on('dragenter', function(e) {
    e.stopPropagation();// 필수
    e.preventDefault();// 필수

    $("#userDropDiv").css('border', '5px dotted #ff6666'); // 옵션. 드래그시 테두리 점선
});
$("#userDropDiv").on('dragover', function(e) {
    e.stopPropagation();
    e.preventDefault();
});
$("#userDropDiv").on('drop', function(e) {
    e.preventDefault();

    $("#myUpload").IBUpload("addDropFile", e); // 핵심 한 줄!!

    $("#userDropDiv").css('border', '1px solid black'); // 디자인 옵션 (동작에 필수 아님 -
    드랍시 테두리를 원래대로 표시한다 )
});

```

## ➤ 버전

7.3.0.20

## ➤ 관련기능

- add 메소드
- useDragDrop 속성

## addFiles 메소드

컨트롤 내에 추가된 파일 목록들을 데이터 객체 얻는다.

### ➤ 문법

```
result = $(Selector).IBUpload("addFiles"); // GET
```

### ➤ 상세설명

컨트롤 내에 추가된 파일 목록들을 데이터 객체로 얻는다.

구성요소 멤버

이름	자료형	기본값	설명
name	String		업로드 했던 파일 원본 명칭. 폴더 명이 제외된 순수한 파일명.
date	String		YYYYMMDDhhmmss(년월일시분초) 형식으로 구성된 문자열 업로드했던 날짜와 시간
size	Number		업로드 파일의 용량 (Byte 단위)
url	String		파일 업로드가 완료된 경우 서버로부터 다운로드 가능한 URL

### ➤ 버전

7.3.0.3

### ➤ 관련기능

- files 속성

## allFiles 메소드

컨트롤 내에 파일 목록들을 JSON 형태의 문자열로 얻는다.

### ➤ 문법

```
result = $(Selector).IBUpload("allFiles"); // GET
```

### ➤ 상세설명

컨트롤 내에 파일 목록들을 JSON 형태의 문자열로 얻는다.

구성요소 멤버

이름	자료형	기본값	설명
name	String		업로드 했던 파일 원본 명칭. 폴더 명이 제외된 순수한 파일명.
date	String		YYYYMMDDhhmmss(년월일시분초) 형식으로 구성된 문자열 업로드했던 날짜와 시간
size	Number		업로드 파일의 용량 (Byte 단위)
url	String		파일 업로드가 완료된 경우 서버로부터 다운로드 가능한 URL

### ➤ 버전

7.3.0.4

### ➤ 관련기능

- files 속성
- addFiles 메소드
- Files 메소드

## delete 메소드

업로드 컨트롤에 존재하는 파일들 중에서 선택한 파일들만 제거한다.

### ➤ 문법

```
$(Selector).IBUpload("delete", boolean); // Method
```

### ➤ 상세설명

업로드 컨트롤에 담겨 있는 파일들 중, 선택된 파일들만 제거한다.

이 메소드로 파일들이 제거된 후에 upload 메소드를 호출하면 제거했던 파일들이 서버에서도 제거된다.

### ➤ 정보

인자명	자료형	기본값	설명
<b>boolean</b>	boolean	true	삭제할 파일 존재 여부 체크 및 메세지 이용 여부(INFO-041)

### ➤ 예제 1

```
// 컨트롤의 파일들 중, 선택한 파일들만 제거한다.
$('#myUpload').IBUpload("delete");
```



➤ 버전

7.3.0.0	
7.3.0.14	boolean 인자 추가

➤ 관련기능

- deleteAll 메소드
- deleteIndex 메소드
- deleteServer 메소드
- deleteReserved 메소드

## deleteAll 메소드

업로드 컨트롤에 담겨 있는 전체 파일들을 제거한다.

### ➤ 문법

```
$(Selector).IBUpload("deleteAll");
```

### ➤ 상세설명

업로드 컨트롤에 담겨 있는 전체 파일들을 제거한다.

upload 메소드를 호출하는 순간 서버쪽 저장된 파일도 삭제된다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- delete 메소드
- deleteIndex 메소드
- deleteServer 메소드
- deleteReserved 메소드

## deleteFiles 메소드

업로드 컨트롤에 담겨 있는 파일들 중, 특정 파일들 n 개를 제거한다.

### ➤ 문법

```
$(Selector).IBUpload("deleteFiles", deleteFileObject); // Method
```

### ➤ 상세설명

업로드 컨트롤에 담겨 있는 파일들 중, 특정 파일들 n 개를 제거한다.

files 메소드에 입력하는 파일 목록과 같이 삭제하려는 파일 목록을 지정하면 해당 파일의 id(파일 추가시 자동 발급 받음) 나 url(서버 전송 후 자동 수신 받음) 이 일치하는 파일들을 제거한다.

아래는 모두 컨트롤의 전체 파일을 삭제하는 기능을 동일하게 수행한다.

```
$("#myUpload").IBUpload("deleteFiles",$("#myUpload").IBUpload("fileList"));
$("#myUpload").IBUpload("deleteAll");
$("#myUpload").IBUpload("reset");
```

단, removeAll 과 reset 에는 미묘한 차이가 있다.

onBeforeDeleteFile 에서 제거를 거부해도 reset 은 초기상태로 돌아가므로 모두 제거되며, removeAll 은 모든 파일을 삭제시도하는 중에 onBeforeDeleteFile 이벤트에서 제거를 거부하면 여전히 파일이 남게 된다.

### ➤ 예제

```
// 특정 셀에 첨부된 파일 목록들을 모두 지운다.
```

```
$('#myUpload').IBUpload("deleteFiles", mySheet.GetCellValue(mySheet.GetSelectRow(),  
"fileData"));
```

➤ 버전

7.3.0.27

➤ 관련기능

- delete 메소드
- deleteAll 메소드
- reset 메소드
- deleteFiles 메소드
- deleteReserved 메소드
- deleteServer 메소드
- onBeforeDeleteFile 이벤트

## deleteIndex 메소드

업로드 컨트롤에 담겨 있는 파일들 중, 특정 위치의 파일 1개만 제거한다.

### ➤ 문법

```
$(Selector).IBUpload("deleteIndex", number); // Method
```

### ➤ 상세설명

업로드 컨트롤에 담겨 있는 파일들 중, 특정 위치의 파일 1개만 제거한다.

0 베이스 인덱스 체계 이므로, 인덱스를 0 으로 주면 컨트롤의 첫번째 파일만 제거된다. fileList 메소드를 이용하여 전체 배열을 얻어서 반복적으로 제거할 경우에는 루프를 거꾸로 역순으로 돌면서 제거를 해야 전체적으로 올바른 결과를 얻을 수 있다.

### ➤ 예제

```
var i = 0;
var fileList = [];

fileList = $('#myUpload').IBUpload('fileList'); // 전체 파일 목록을 얻는다.

for (i = fileList.length-1; i >= 0; i--) { // 반드시 역순으로 돌아야 제거 직후에 자료가 밀리거나
  당겨지는 일이 없다.

  if (fileList[i].name.indexOf(".jpg") > -1 ) { // jpg 파일만 모두 제거
    $('#myUpload').IBUpload('deleteIndex', i);
```

```
}  
}
```

➤ 버전

7.3.0.5

➤ 관련기능

- delete 메소드
- deleteAll 메소드
- deleteReserved 메소드
- deleteServer 메소드
- onBeforeDeleteFile 이벤트

## deleteReserved 메소드

서버 쪽에서도 제거해야 할 파일 목록들을 얻는다.

### ➤ 문법

```
var myFiles = "";
myFiles = $(Selector).IBUpload("deleteReserved"); // Get StringValue
```

### ➤ 상세설명

서버 쪽에서도 제거해야 할 파일 목록들을 얻는다.

files 속성(또는 메소드) 을 이용하여 업로드 컨트롤에 파일들을 채워서 조회한 시점 이후로 사용자가 파일을 제거하면, autoUpload=true 인 경우 서버에서도 즉시 제거되지만, autoUpload=false 인 경우에는 upload 메소드를 호출할 때 까지 서버에서는 아무런 반응을 하지 않는다.

사용자가 제거한 파일은 upload 메소드를 호출하여 서버 쪽으로 제거해야 할 파일 목록들이 전달되는데, 이와 같이 화면에서는 제거되었으나 서버에서는 아직 제거되지 않은 상태로 있는 동안에 이 메소드를 호출하면 제거하기로 예약되어 있는 모든 파일들의 목록을 얻을 수 있다.

upload 메소드를 통하여 서버에서도 지워야 할 항목들을 제거하게 되면, 이 메소드에서는 이후로 널 값이 리턴된다.

리턴 값에는 files, linkDown 메소드에서 사용하는 리턴 값이나 설정 값과 그 형식이 동일하다.

➤ 버전

7.3.0.7

➤ 관련기능

- delete 메소드
- deleteIndex 메소드
- deleteAll 메소드
- deleteServer 메소드
- files 메소드
- linkDown 메소드
- onBeforeDeleteFile 이벤트



## deleteServer 메소드

서버 쪽에 저장된 파일을 제거하도록 요청한다.

### ➤ 문법

```
$(Selector).IBUpload("deleteServer", objectArray ); // Method
```

### ➤ 상세설명

서버 쪽에 저장된 파일을 제거하도록 요청한다.

delete 메소드는 컨트롤에 존재하는 파일들 중에서 선택한 파일들만 제거하고, 이후로 upload 메소드가 호출되는 순간에는 서버에서도 해당 파일을 자동으로 제거하게 되는데, deleteServer 메소드는 컨트롤에 존재하지 않더라도 유효한 url 을 가진 files 정보만 인자로 설정하여 호출하면, 이후로 upload 메소드가 호출되는 순간에 서버에서도 해당 파일을 자동으로 제거하게 된다.

인자의 형식은 files 메소드나 linkDown 메소드와 동일하다.

지정된 name 이나 size 속성들은 모두 무시하고 url 값만 참조하여 서버로 \_\_delList 라는 명칭으로 upload 메소드 호출시 함께 전송하게 된다.

### ➤ 예제

```
// 컨트롤에 존재하지 않더라도 서버쪽까지 해당 파일정보를 통하여 특정 파일을 제거하기 위하
```

여 인자로 제공한다.

```
$("#myUpload").IBUpload("deleteServer", "{ 'name': '연습.hwp',  
'url': '20160126_180337_82754651' }, { 'name': '삶.ppt', 'url': '20160126_180801' }");
```

## ➤ 버전

7.3.0.6

## ➤ 관련기능

- delete 메소드
- deleteIndex 메소드
- deleteAll 메소드
- deleteServer 메소드
- deleteReserved 메소드
- files 메소드
- linkDown 메소드
- onBeforeDeleteFile 이벤트

## downloadAll 메소드

전체 파일을 다운로드 받는다.

### ➤ 문법

```
$(Selector).IBUpload("downloadAll"); // Method
```

### ➤ 상세설명

IBUpload 컨트롤에 추가된 모든 파일들을 다운로드 받는다.

1개의 파일을 다운로드 받을 경우 해당 파일을 직접 다운로드 받지만, 2개 이상의 파일을 다운로드 받을 경우에는 zip 으로 압축하여 다운로드 받는다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- download 메소드
- deleteAll 메소드

## download 메소드

1개 이상의 URL 파일들을 다운로드 받는다.

### ➤ 문법

```
$(Selector).IBUpload("download"); // Method
```

### ➤ 상세설명

1개 이상의 URL 파일들을 다운로드 받는다.

업로드 컨트롤에 담겨 있는 파일들 중, 선택한 특정 파일들을 서버로부터 다운로드 받는다.

1개의 파일을 다운로드 받을 경우 해당 파일을 직접 다운로드 받지만, 2개 이상의 파일을 다운로드 받을 경우에는 zip 으로 압축하여 다운로드 받게 된다.

### ➤ 예제

```
$('#myUpload').IBUpload("download");
```

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- downloadAll 메소드

- deleteAll 메소드

## downloadFileName 메소드

2건 이상의 파일 다운로드 시 zip로 파일을 받게 된다. 이때의 zip파일의 이름을 설정하거나 확인한다.

### ➤ 문법

```
$(Selector).IBUpload("downloadFileName", stringValue); //SET  
$(Selector).IBUpload("downloadFileName"); //GET
```

### ➤ 상세설명

2건 이상의 파일을 다운로드시 파일의 이름을 설정하거나 확인 할 수 있다.

### ➤ 버전

7.3.0.5

## downloadServerUrl 메소드

다운로드 기능을 제공하는 서버 쪽 jsp 프로그램의 경로를 재설정 한다.

### ➤ 문법

```
$(Selector).IBUpload("downloadServerUrl",stringValue);
```

### ➤ 상세설명

다운로드 기능을 제공하는 서버 쪽 jsp 프로그램의 경로를 재설정 한다.

downloadServerUrl 속성은 제품 초기화 시, 최초에 1회만 실행되지만,  
downloadServerUrl 메소드는 다운로드 직전에 해당 경로를 수시로 변경할 수 있다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- downloadServerUrl 속성
- updateServerUrl 메소드

## dropSheetRow 메소드

IBSheet 의 특정 Cell 로 파일들을 Drop 한 경우, 특정 Cell 의 위치에 해당되는 Row 값을 얻는다.

### ➤ 문법

```
$(Selector).IBUpload("dropSheetRow");
```

### ➤ 상세설명

IBSheet 의 특정 Cell 로 파일들을 Drop 한 경우, 특정 Cell 의 위치에 해당되는 Row 값을 얻는다.

onDragSheetRow 이벤트를 사용하여 드래그 중에 IBSheet 의 SelectCell 를 매번 마우스의 위치로 설정한다면, IBSheet의 GetSelectRow 값으로도 동일한 값을 얻을 수 있으므로 이 경우에는 본 메소드가 필요하지 않다.

### ➤ 예제

```
onAddFile: function(files) {

    if (ibleaders.ibupload.explicitJson == true) {
        files = JSON.stringify(files); // 객체 => 문자열
    } else {
        files = "[" + files + "]";
    }
}
```



```

//var iRow =mySheet.GetSelectRow();
var iRow = $('#myUpload').IBUpload("dropSheetRow");

// 셀파일제거 ver 73024 이상에서만 지원됨.
$('#myUpload').IBUpload("deleteFiles", mySheet.GetCellValue(iRow, "fileData"));

mySheet.SetCellValue(iRow, "fileData", files);
mySheet.SetCellValue(iRow, "fileIcon", fildData_2_IconTextList(files));

},

```

#### ➤ 버전

7.3.0.28

## dropTarget 메소드

IBSheet 의 Cell 로 파일 첨부을 연동할 경우, 어느 IBSheet 와 연동할지 ID 를 지정한다.

### ➤ 문법

```
$(Selector).IBUpload("dropTarget", StringValue );
```

### ➤ 상세설명

IBSheet 의 Cell 로 파일 첨부을 연동할 경우, 어느 IBSheet 와 연동할지 ID 를 지정한다.

많은 IBSheet 중에서 어떤 IBSheet 로 파일을 드래그 & 드랍 할때에 IBUpload 에서 파일첨부 기능을 연동할지 ID 를 설정해야 그 이후로 드래그 & 드랍을 통한 연동이 시작된다. IBUpload 와 IBSheet 간의 연동이므로 두 컨트롤 모두 생성이 완료된 후에, 본 메소드를 호출해야 정상 동작한다.

### ➤ 예제

```
$('#myUpload').IBUpload('dropTarget','mySheet');
```

### ➤ 버전

7.3.0.28

➤ **관련기능**

- dropSheetRow 메소드
- onDragSheetRow 이벤트

## extendParamDownload 메소드

**downloadServerUrl** 속성에 설정한 다운로드 전용 서버쪽 프로그램으로 파일 다운로드 요청 정보를 전송할 때 폼안에 추가 데이터를 함께 보낼 수 있다.

### ➤ 문법

```
$(Selector).IBUpload("extendParamDownload","prop=value&id2=value2");

$(Selector).IBUpload("extendParamDownload", "inputName", stringValue);
```

### ➤ 상세설명

**downloadServerUrl** 속성에 설정한 다운로드 전용 서버쪽 프로그램으로 파일 다운로드 요청 정보를 전송할 때 폼안에 추가 데이터를 함께 보낼 수 있다.

아래의 예제와 같이 작성하여 추가 데이터를 전송할 수 있으며, 데이터의 개수만큼 <input> 태그들이 함께 전송 되므로 서버 쪽에서는 request.getParameter() 같은 방법으로 해당 값들을 전송한 그대로 얻을 수 있다.

### ➤ 예제 1

```
$(Selector).IBUpload("extendParamDownload", "userid=user-id10012&content=올해의 신청자료");
```

아래와 같이 폼 안에 추가되어 전송된다.

```
<input type="hidden" name="user-id" value="user-id10012">
<input type="hidden" name="content" value="올해의 신청자료 ">
```

## ➤ 예제 2

```
$(Selector).IBUpload("extendParamDownload","myReq","uestuserid=user-id10012&content=올해의 신청자료");
```

아래와 같이 폼 안에 추가되어 전송된다.

```
<input type="hidden" name="myReq" value="uestuserid=user-id10012&content=올해의  
신청자료">
```

## ➤ 버전

7.3.0.30

## extendParamUpload 메소드

**uploadServerUrl** 속성에 설정한 업로드 전용 서버쪽 프로그램으로 파일을 전송할때 폼안에 추가 데이터를 함께 보낼 수 있다.

### ➤ 문법

```
$(Selector).IBUpload("extendParamUpload", "prop=value&id=value2");

$(Selector).IBUpload("extendParamUpload", "inputName", stringValue);
```

### ➤ 상세설명

**uploadServerUrl** 속성에 설정한 업로드 전용 서버쪽 프로그램으로 업로드 파일과 함께 전송할 추가 데이터들을 함께 보낸다.

아래의 예제와 같이 작성하여 추가 데이터를 전송할 수 있으며, 데이터의 개수만큼 `<input>` 태그들이 함께 전송 되므로 서버 쪽에서는 `request.getParameter()` 같은 방법으로 해당 값들을 전송한 그대로 얻을 수 있다.

### ➤ 예제 1

```
$(Selector).IBUpload("extendParamUpload", "userid=user-id10012&content=올해의 신청
자료");
```

아래와 같이 폼 안에 추가되어 전송된다.

```
<input type="hidden" name="user-id" value="user-id10012">
<input type="hidden" name="content" value="올해의 신청자료 ">
```

## ➤ 예제 2

```
$(Selector).IBUpload("extendParamUpload","myReq","uestuserid=user-id10012&content=올해의 신청자료");
```

아래와 같이 폼 안에 추가되어 전송된다.

```
<input type="hidden" name="myReq" value="uestuserid=user-id10012&content=올해의  
신청자료">
```

## ➤ 버전

7.3.0.0

## fileCount 메소드

컨트롤 내에 추가된 파일의 개수를 얻는다.

### ➤ 문법

```
result = $(Selector).IBUpload("fileCount"); // GET
```

### ➤ 상세설명

컨트롤 내에 추가된 파일의 개수를 얻는다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- limitFileCount 메소드
- fileUploadedCount 메소드



## fileUploadedCount 메소드

컨트롤 내에 업로드 완료된 파일의 개수를 얻는다.

### ➤ 문법

```
result = $(Selector).IBUpload("fileUploadedCount"); // GET
```

### ➤ 상세설명

컨트롤 내에 추가된 전체 파일 중 대기중이 아니라, 서버로 업로드가 완료된 개수를 얻는다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- fileCount 메소드
- limitFileCount 메소드

## iconMode 메소드

컨트롤 내부의 생성 모양을 설정한다.

### ➤ 문법

```
$(Selector).IBUpload("iconMode", stringValue); //SET  
$(Selector).IBUpload("iconMode"); //GET
```

### ➤ 상세설명

아이콘 타입의 업로드 컨트롤 타입을 설정한다. 제공되는 모드는 icon, list, detail 형태이며 각 모드별 사용 변경은 css를 통해 컨트롤 가능하다.

### ➤ 버전

7.3.0.0

## isModified 메소드

컨트롤에서 새롭게 추가 또는 제거한 파일이 있는지 확인한다.

### ➤ 문법

```
var result = $(Selector).IBUpload("isModified"); //GET
```

### ➤ 상세설명

컨트롤에서 새롭게 추가 또는 제거한 파일이 있는지 확인한다.

기존의 파일을 1개 이상 제거 또는 추가를 하여 "대기중" 인 상태로 있을 경우, upload 메소드를 호출하기 까지 isModified 메소드는 true 를 리턴한다.

파일을 1개 이상 추가 또는 제거하여 upload 메소드를 호출하였으나, 서버에서 여러 가지 원인에 따라 오류가 발생한 경우에는 여전히 isModified 는 true 로 유지된다.

autoUpload=true 인 경우에는 추가 및 제거시 즉각적인 upload 메소드가 자동으로 호출되므로 isModified 는 항상 false 를 리턴하게 된다.

### ➤ 버전

7.3.0.7

### ➤ 관련기능

- deleteReserved 메소드

## files 메소드

컨트롤 내에 추가된 파일 목록 중 파일상태가 다운로드 가능한 목록을 데이터 객체로 얻거나 전체파일 목록을 설정한다.

### ➤ 문법

```
result = $(Selector).IBUpload("files"); // GET
$(Selector).IBUpload("files", string or objectValue); // SET
```

### ➤ 상세설명

컨트롤 내에 추가된 파일 목록들 중 파일상태가 다운로드 가능한 목록을 데이터 객체로 얻을때 사용한다. (전체 파일목록을 얻기 위해서는 allFiles 메소드 사용) 컨트롤의 파일을 설정할때는 컨트롤의 파일은 모두 삭제되고 설정된다.

Files 구성요소 멤버

이름	자료형	기본값	설명
name	String		업로드 했던 파일 원본 명칭. 폴더 명이 제외된 순수한 파일명.
date	String		YYYYMMDDhhmmss(년월일시분초) 형식으로 구성된 문자열 업로드했던 날짜와 시간
size	Number		업로드 파일의 용량 (Byte 단위)
url	String		파일 업로드가 완료된 경우 서버로부터 다운로드 가능한 URL

- 버전  
7.3.0.0
- 관련기능
  - files 속성

## fileList 메소드

컨트롤 내에 포함되어 있는 모든 파일들을 하나의 배열로 얻는다.

### ➤ 문법

```
var result = [];
result = $(Selector).IBUpload("fileList"); // GET
```

### ➤ 상세설명

컨트롤 내에 포함되어 있는 모든 파일들을 하나의 배열로 얻는다.

컨트롤에 표시된 파일들이 그대로 배열로 구성되므로, 대기중인 파일들과 이미 서버에 저장되어서 다운로드 가능한 파일들까지 모두 포함하여 리턴되며, 파일의 배치 순서와 개수까지 정확히 일치한다.

컨트롤에 포함된 파일들의 파일명, 파일크기, 경로 등을 얻기 위해서는 이 배열을 루프돌면서 각 요소별로 속성들을 참조한다. ( Files 메소드의 멤버 참조)

### ➤ 예제

```
// 컨트롤에 표시된 모든 파일들의 파일명을 순서대로 표시하기
var i = 0;
var fileList = [];

fileList = $('#myUpload').IBUpload('fileList');
```

```
for (i = 0; i < fileList.length; i++) { // 모든 파일들을 표시하기 위하여 루프를 돈다.  
    alert(fileList[i].name); // 각각의 파일명을 표시한다.  
}
```

➤ 버전

7.3.0.5

➤ 관련기능

- files 메소드

## limitFileCount 메소드

컨트롤 내에 추가할 파일의 최대 개수를 얻거나 설정한다.

### ➤ 문법

```
result = $(Selector).IBUpload("limitFileCount"); // GET
$(Selector).IBUpload("limitFileCount", numberValue); // SET
```

### ➤ 상세설명

컨트롤 내에 추가할 파일의 최대 개수를 얻거나 설정한다.

동시에 N 개의 파일을 추가 시도 중에 최대 개수를 초과하게 되면 해당 N 개의 멀티 파일 선택 목록 전체가 추가 되지 않고 취소 된다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- limitFileCountOnce 메소드
- limitFileSize 메소드
- limitFileTotalSize 메소드



## limitFileCountOnce 메소드

컨트롤 내에 한번에 추가할 파일의 최대 개수를 얻거나 설정한다.

### ➤ 문법

```
result = $(Selector).IBUpload("limitFileCountOnce"); // GET
$(Selector).IBUpload("limitFileCountOnce", numberValue); // SET
```

### ➤ 상세설명

컨트롤 내에 한번에 추가할 파일의 최대 개수를 얻거나 설정한다.

동시에 N 개의 파일을 추가 시도 중에 최대 개수를 초과하게 되면 해당 N 개의 멀티 파일 선택 목록 전체가 추가 되지 않고 취소 된다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- limitFileCount 메소드
- limitFileSize 메소드
- limitFileTotalSize 메소드

## limitFileSize 메소드

컨트롤 내에 추가할 파일의 최대 제한 용량을 얻거나 설정한다.

### ➤ 문법

```
result = $(Selector).IBUpload("limitFileSize"); // GET
$(Selector).IBUpload("limitFileSize", numberValue); // SET
```

### ➤ 상세설명

컨트롤 내에 추가할 1개 파일의 최대 제한 용량을 Byte 단위로 얻거나 설정한다.

동시에 N 개의 파일을 추가 시도 중에 최대 제한 용량을 초과하는 파일이 1개라도 발생하면 해당 N 개의 멀티 파일 선택 목록 전체가 추가 되지 않고 취소 된다.

### ➤ 예제

```
// 초기화 후에 최대 제한 용량을 변경함
$('#myUpload').IBUpload('limitFileSize', 1024 * 1024 * 1024);
```

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- limitFileCount 메소드
- limitFileTotalSize 메소드

## limitFileTotalSize 메소드

컨트롤 내에 추가할 모든 파일들의 총 합계의 최대 제한 용량을 얻거나 설정한다.

### ➤ 문법

```
result = $(Selector).IBUpload("limitFileTotalSize"); // GET
$(Selector).IBUpload("limitFileTotalSize", numberValue); // SET
```

### ➤ 상세설명

컨트롤 내에 추가할 모든 파일들의 총 합계의 최대 제한 용량을 Byte 단위로 얻거나 설정한다. 동시에 N 개의 파일을 추가 시도 중에 총 합계의 최대 제한 용량을 초과하는 파일이 1개라도 발생하면 해당 N 개의 멀티 파일 선택 목록 전체가 추가 되지 않고 취소 된다.

### ➤ 예제

```
// 파일 1개 용량을 1 GB 로 제한한다.
$('#myUpload').IBUpload("limitFileSize", 1024 * 1024 * 1024);
```

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- limitFileSize 메소드
- limitFileCount 메소드

## linkDown 메소드

IBUpload 컨트롤에서 선택된 파일 다운로드가 아닌 별도의 파일목록을 즉시 다운로드 받고 싶을 때 사용할 수 있도록 설정한다.

### ➤ 문법

```
$(Selector).IBUpload("linkDown",stringValue); // SET
```

### ➤ 상세설명

별도의 파일 목록들을 문자열로 지정하여 해당 파일만 즉시 다운로드 할 수 있도록 설정한다. stringValue값에는 files메소드의 인자값과 그 형식이 동일하다.

### ➤ 예제

```
<script type="text/javascript">
    $("#myUpload").IBUpload("linkDown",{name: "관심과집중.mp4", size: "11417124",
date: "20160101125959", url: "20160126_180337_82754651"},{name: "오렌지.jpg",
size: "1075761", date: "20160101125959", url: "20160126_180801"});
</script>
```

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- download 메소드

- downloadAll 메소드
- files 메소드

## reset 메소드

업로드 컨트롤을 초기화한다.

### ➤ 문법

```
$(Selector).IBUpload("reset");
```

### ➤ 상세설명

업로드 컨트롤을 초기화 한다. deleteAll 메소드는 upload메소드를 사용하면 삭제된 파일 내역이 서버에 전송되지만 reset메소드는 서버쪽으로 파일이 전송되지 않는다.

### ➤ 버전

7.3.0.4

### ➤ 관련기능

- deleteAll 메소드

## selectedIndex 메소드

업로드 컨트롤에 담겨 있는 파일들 중, 특정 위치의 파일 1개의 선택 여부를 얻거나 설정한다.

### ➤ 문법

```
$(Selector).IBUpload("selectedIndex", {index:0, isSelected:true}); // SET
var isSelected = $(Selector).IBUpload("selectedIndex", {index:0 }); // GET
```

### ➤ 상세설명

업로드 컨트롤에 담겨 있는 파일들 중, 특정 위치의 파일 1개만 선택여부를 얻거나 설정한다.

0 베이스 인덱스 체계 이므로, 인덱스를 0 으로 주면 컨트롤의 첫번째 파일만 선택 또는 선택 해제할 수 있다.

fileList 메소드를 이용하여 전체 배열을 얻어서 반복적으로 선택 또는 선택 해제 할 경우에는 루프를 거꾸로 역순으로 돌든 정순으로 돌든 상관이 없으나, 제거를 포함할 경우에는 반드시 루프를 거꾸로 돌아야 전체적으로 올바른 결과를 얻을 수 있다.

### ➤ 예제

```
var i = 0;
var fileList = [];

fileList = $('#myUpload').IBUpload('fileList'); // 전체 파일 목록을 얻는다.
```

for (i = fileList.length-1; i >= 0; i--) { // 반드시 역순으로 돌아야 제거 직후에 자료가 밀리거나 당겨지는 일이 없다.

```
    if (fileList[i].name.indexOf(".jpg") > -1
        && $('#myUpload').IBUpload('selectedIndex', {index:i})==true
    ) { // jpg 파일들 중에서 선택된 파일들만 모두 제거
        $('#myUpload').IBUpload('deleteIndex', i);
    }
}
```

## ➤ 버전

7.3.0.16

## ➤ 관련기능

- selectAll 메소드
- fileList 메소드



## selectAll 메소드

업로드 컨트롤에 담겨 있는 모든 파일들을 전체 선택, 또는 전체 선택해제 처리한다.

### ➤ 문법

```
$(Selector).IBUpload("selectAll", booleanValue); // SET
```

### ➤ 상세설명

업로드 컨트롤에 담겨 있는 모든 파일들을 전체 선택, 또는 전체 선택해제 처리한다.

### ➤ 버전

7.3.0.16

### ➤ 관련기능

- selectedIndex 메소드

## upload 메소드

수동 업로드 (autoUpload=false) 방식에서, 대기 상태에 있던 모든 파일들을 즉시 업로드 한다.

### ➤ 문법

```
$(Selector).IBUpload("upload" [, object ] ); // Method
```

### ➤ 상세설명

수동 업로드(autoUpload = false ) 방식에서, 대기 상태에 있던 모든 파일들을 즉시 업로드 한다.

자동 업로드 방식에서는 파일을 추가할 때마다 이 메소드가 내부적으로 자동 호출된다.

파일을 전혀 추가하지 않더라도 upload 를 호출할때에는 폼에 파일만 첨부되지 않을 뿐, extendParamUpload 값들과 함께 자료들은 항상 서버로 전송 된다.

upload 메소드와 add 메소드는 첫번째 인자로 IBUpload 컨트롤의 초기 생성 객체를 재설정하는 기회를 제공한다. 즉, upload 메소드가 호출되고 난 직후, IBUpload 의 모든 속성과 모든 이벤트들중 일부를 재설정 할 수 있다는 뜻이다.

(단, 초기화 로딩 시점에서 화면에 표시와 관련되는 iconMode 나 theme 속성은 여기에서 재설정한다고 즉시 적용되는 것은 아니며, 이 설정 시점 이후로 동작에 영향을 주는 limitFileCount 등의 모든 속성과 onMessage 이벤트등의 모든 이벤트들은 재설정이 가능하다.)

또한, 첫번째 인자 안에 `callback` 속성에 함수를 선언하면 함수가 호출된 직후의 로직을 추가할 수 있으며 다음과 같은 순서로 콜백함수와 관련 이벤트가 발생한다.

순서	처리 내용	설명
1	업로드 실행	<code>upload</code> 메소드를 호출한다.
2	<code>onUploadData</code> 이벤트 발생	서버로부터 응답이 온다.
3	<code>onUploadFinish</code> 이벤트 발생	업로드가 모두 완료된다.
4	<code>callback</code> 함수 실행	<code>callback</code> 함수를 실행한다.

## ➤ 예제

```
//IBSheet와 연동시

$('#myUpload').IBUpload('upload', {
  onUploadData: function(files){
    alert('서버 응답 처리 로직이 변경되었습니다 ');
  },
  callback: function(files){
    alert('서버에 업로드를 처리한 직후 입니다.');
```

```
});
```

➤ 버전

7.3.0.0

7.3.0.21 – 속성과 이벤트의 재정의 기능

➤ 관련기능

- target 속성
- add 메소드

## uploadServerUrl 메소드

업로드 기능을 제공하는 서버 쪽 jsp 프로그램의 경로를 재설정 한다.

### ➤ 문법

```
$(Selector).IBUpload("uploadServerUrl",stringValue);
```

### ➤ 상세설명

업로드 기능을 제공하는 서버 쪽 jsp 프로그램의 경로를 재설정 한다.

uploadServerUrl 속성은 제품 초기화 시, 최초에 1회만 실행되지만, uploadServerUrl 메소드는 업로드 직전에 해당 경로를 수시로 변경할 수 있다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

- uploadServerUrl 속성
- downloadServerUrl 메소드

## version 메소드

업로드 컨트롤의 버전을 얻는다.

문법

```
$(Selector).IBUpload("version");
```

### ➤ 상세설명

업로드 컨트롤의 버전을 얻는다.

### ➤ 버전

7.3.0.0

### ➤ 관련기능

## C. 이벤트

이벤트는 IBUpload 초기화 시점에서 속성 설정과 유사하게 설정하게 된다.

```
$("#myUpload").IBUpload({
  autoUpload:false,
  limitFileCount:5,
  //... 초기화 속성 설정. ...
  onEventName:function(eventParameter1 , eventParameter2 ...){
  },
  ....
});
```

[기본 이벤트 문법]

모든 화면에 들어가는 Upload 컨트롤에 동일한 이벤트를 적용하고자 하는 경우에는 ibuploadinfo.js 파일에서 설정 할수 있다.

```
var ibleaders.ibupload = {
  //.... 전역이벤트 설정..
  onAddFile:function(files){
    if(files.length > 10) {
      alert("too many files select");
    }
  }
};
```

[모든 upload 컨트롤에 onAddFile이벤트 적용]

만약 ibuploadinfo.js와 해당 upload 초기화 구문에 동일한 이벤트가 존재하는 경우에는 upload 초기화 구문에서 설정한 이벤트만 구동된다.

## onAddFile 이벤트

사용자가 파일 선택이 완료되면 해당 이벤트가 발생한다.

### ➤ 문법

```
$(Selector).IBUpload({
    onAddFile:function(files, addFileObject){
        // 사용자 로직
    }
});
```

### ➤ 상세설명

add메소드를 통해 파일을 선택하면 문자열로 리턴된다.

예를 들면 \$("#myUpload").IBUpload('add')메소드를 "공지사항.hwp"라 하면 {name:"공지사항.hwp", size:"104242", date:"20160409150708", url:"20160409150708"} 과 같이 문자열 (explicitJson=true 인 경우 object 로 리턴)이 files로 리턴된다.

### ➤ 예제

```
//IBSheet와 연동시

$('#myUpload').IBUpload({
    onAddFile:
        function(files){
            mySheet.SetCellValue(mySheet.GetSelectRow(),"filedata",files);
            mySheet.SetCellValue(mySheet.GetSelectRow(),
                "filetext",_IBUpload_GetFileNamesFromFiles(files));
        }
});
```



```
});
```

### ➤ 정보

인자명	자료형	기본값	설명
<b>files</b>	string ( or object)	없음	name, size, date, url 문자열 (explicitJson=true 인 경우 object 타입으로 리턴됨)
<b>addFileObject</b>	object	없음	추가된 파일 객체 (브라우저가 제 공하는 파일 객체)

### ➤ 버전

7.3.0.0	
7.3.0.11	addFileObject 인자 추가

## onBeforeAddFile 이벤트

사용자가 파일을 선택하여 IBUpload 에 파일을 추가하기 직전에 이벤트가 발생하여 파일 추가를 취소할 수 있다.

### ➤ 문법

```
$(Selector).IBUpload({
  onBeforeAddFile:function(Reservedfiles, addFileObject){
    // 사용자 로직
  }
});
```

### ➤ 상세설명

사용자가 파일을 선택하여 IBUpload 에 파일을 추가 하기 직전에 이벤트가 발생하여 파일 추가를 취소할 수 있다.

본 이벤트에서는 최종적으로 true 를 리턴하면 더 이상 onAddFile 이벤트가 발생하지 않고, 컨트롤에 파일도 추가되지 않은 채 그냥 취소되고 끝난다. return 문장을 생략하거나 return false 를 리턴하면 정상적으로 파일 추가 기능이 지속된다.

### ➤ 예제

```
//IBSheet와 연동시

$('#myUpload').IBUpload({
  onBeforeAddFile: function(Reserved, tryAddFiles) {
    for (var i = 0; i < tryAddFiles.length; i++) {
```

```

        alert(tryAddFiles[i].name + " 파일을 추가하려고 합니다.");
        if (tryAddFiles[i].name.indexOf(".jpg") != - 1) {
            alert("jpg 파일은 추가를 불허합니다.");
            return true;
        }
    }

}

});

```

#### ➤ 정보

인자명	자료형	기본값	설명
<b>ReservedFiles</b>	N/A		(아직 제공되지 않음) (추후, json 형태의 객체 제공 예정)
<b>addFileObject</b>	object	없음	추가된 파일 객체 (브라우저가 제공하는 파일 객체)

#### ➤ 버전

7.3.0.27

## onBeforeDeleteFile 이벤트

IBUpload 에서 파일을 사용자가 제거하기 직전에 이벤트가 발생하여 파일 제거를 취소할 수 있다.

### ➤ 문법

```
$(Selector).IBUpload({
  onBeforeDeleteFile:function(tryDeleteFile){
    // 사용자 로직
  }
});
```

### ➤ 상세설명

IBUpload 에서 파일을 사용자가 제거하기 직전에 이벤트가 발생하여 파일 제거를 취소할 수 있다.

본 이벤트에서는 최종적으로 true 를 리턴하면 더 이상 삭제 처리가 진행되지 않고, 컨트롤에서 파일도 더 이상 제거되지 않은 채 그냥 취소되고 끝난다. return 문장을 생략하거나 return false 를 리턴하면 정상적으로 파일 제거 기능이 지속된다.

tryDeleteFile 인자는 IBUpload 의 파일 구성요소와 동일한 형태( size, url, name, date 등으로 구성된) 이다. files 메소드의 [ Files 구성요소 멤버 ] 를 참고.

### ➤ 예제

```

$('#myUpload').IBUpload({
    onBeforeDeleteFile: function(tryDeleteFile) {
        alert(tryDeleteFile.name + " 파일을 제거하려고 합니다.");

        if (tryDeleteFile.name.indexOf(".jpg") > - 1) {
            alert("jpg 파일은 삭제를 불허합니다.");
            return true;
        }
    },
});

```

#### ➤ 정보

인자명	자료형	기본값	설명
<b>tryDeleteFile</b>	FileObject	없음	삭제하려는 파일 1개

#### ➤ 버전

7.3.0.27

## onContextMenu 이벤트

오른쪽 마우스 버튼을 눌러서 표시한 팝업메뉴를 클릭했을 때 이벤트를 발생한다.

### ➤ 문법

```
$(Selector).IBUpload({
  onContextMenu:function(key){
    // 사용자 로직
  }
});
```

### ➤ 상세설명

마우스 오른쪽 버튼을 눌러서 표시한 팝업메뉴를 항목을 선택했을 때 이벤트를 발생한다.

해당 이벤트를 정의하기 전에 반드시 [contextMenuItems](#) 속성에 대한 설정이 해당 페이지나 ibuploadinfo.js에 존재해야 한다.

### 정보

인자명	자료형	기본값	설명
Key	string	없음	ibuploadinfo.jsp 에서 설정한 ibleaders.ibupload.contextMenuItems 개체의 각 항목 명칭

### ➤ 버전

7.3.0.0

## onDeleteFile 이벤트

IBUpload 에서 파일을 사용자가 제거한 후 이벤트가 발생한다.

### ➤ 문법

```
$(Selector).IBUpload({
  onDeleteFile:function(){
    // 사용자 로직
  }
});
```

### ➤ 상세설명

IBUpload 에서 파일을 사용자가 제거한 후 이벤트가 발생한다. 본 이벤트에서 fileCount 메소드로 삭제 직후 남아있는 파일 개수를 확인 할 수 있다.

### ➤ 예제

```
$('#myUpload').IBUpload({
  onDeleteFile: function() {
    var count = $("#myUpload").IBUpload("fileCount");
    $("#total").text(` ${$("#myUpload").IBUpload("fileCount")} 개 `);
  }
});
```

### ➤ 버전

7.3.0.62

## onDragSheetRow 이벤트

IBSheet 로 셀첨부를 위한 Drag & Drop 과정 중에서 파일을 잡아 Drag 하는 동안 발생하는 이벤트이다.

### ➤ 문법

```
$(Selector).IBUpload({
  onDragSheetRow:function(sheetID, sheetRow){
    // 사용자 로직
  }
});
```

### ➤ 상세설명

IBSheet 로 셀첨부를 위한 Drag & Drop 과정 중에서 파일을 잡아 Drag 하는 동안 발생하는 이벤트이다. dropTarget 메소드를 호출하고 나서 IBSheet 위로 파일을 Drag 하면 IBUpload 에서 이 이벤트가 발생한다.

#### 정보

인자명	자료형	기본값	설명
<b>sheetID</b>	string	없음	드래그 & 드랍 하고 있는 IBSheet 의 ID 값. ( = dropTarget 에 설정한 IBSheet 의 id 문자열 값 )
<b>sheetRow</b>	number	없음	드래그 & 드랍 하고 있는 IBSheet 의 Row 값



➤ 예제

```
$('#myUpload').IBUpload({  
  onDragSheetRow: function(sheetID, sheetRow) {  
    // 드래그 할때마다 파일첨부 아이콘 컬럼을 선택해 준다.  
    window[sheetID].SelectCell(sheetRow,  
    window[sheetID].SaveNameCol("fileIcon"));  
  }  
});
```

➤ 버전

7.3.0.28

## onDbClick 이벤트

IBUpload를 더블클릭 했을 경우 이벤트가 발생한다.

### ➤ 문법

```
$(Selector).IBUpload({
  onDbClick: function(uploadid){
    // 사용자 로직
  }
});
```

### ➤ 상세설명

IBUpload를 더블클릭 했을 경우 이벤트가 발생한다.

인자명	자료형	기본값	설명
<b>uploadid</b>	String	업로드 컨트롤 ID	생성된 업로드 컨트롤 ID

### ➤ 예제

```
$('#myUpload').IBUpload({
  onDbClick: function(uploadid) {
    // IBUpload를 더블클릭 했을 경우 다운로드가 실행된다.
    $('#' + uploadid).IBUpload('download');
  }
});
```

➤ 버전  
7.3.0.41

## onMessage 이벤트

컨트롤 내에서 발생하는 모든 메시지 이벤트를 발생한다.

### ➤ 문법

```
$(Selector).IBUpload({
  onMessage : function(msgID, msgDesc, msgOrg){
    // 사용자 로직
  }
});
```

### ➤ 상세설명

컨트롤 내에서 파일추가,업로드,삭제 등 여러가지 작업이 발생할 때마다 관련 ID값과 함께 메시지가 이벤트로 발생한다. msgOrg은 서버에서 받은 request body 전문이다. 제공되는 메시지는 ibuploadinfo.js 파일에 정의 되어있다.

코드에 따른 메시지는 개발자 가이드 문서 "**E. 메시지 코드**" 를 참고한다.

### ➤ 예제

```
$('#myUpload').IBUpload({
  onMessage: function(msgID, msgDesc, msgOrg){
    if (msgID.substring(0, 3) == "ERR") {
      alert(msgDesc + "\n\n" + "error : " + msgID); // 오류만 메시지로 표시
    } //end if
  } // end onMessage
});
```

➤ 버전

7.3.0.0	
7.3.0.12	msgOrg 인자 추가

## onSetFiles이벤트

files 메소드로 IBUpload 컨트롤 내 파일들을 초기화 시 이벤트가 발생한다.

### ➤ 문법

```
$(Selector).IBUpload({
    onSetFiles : function(){
        // 사용자 로직
    }
});
```

### ➤ 상세설명

files 메소드로 IBUpload 컨트롤 내 파일들을 초기화 시 발생하는 이벤트로, 처음에 로딩된 파일 개수를 화면에 표시하고자 할 때 사용할 수 있다.

### ➤ 예제

```
$('#myUpload').IBUpload({
    onSetFiles: function(){
        var count = $("#myUpload").IBUpload("fileCount");
        $("#total").text(` ${$("#myUpload").IBUpload("fileCount")} 개 `);
    }
});
```

➤ 버전  
7.3.0.62

## onUploadData 이벤트

업로드 완료 후 서버에서 응답문을 얻는다.

### ➤ 문법

```
$(Selector).IBUpload(
    onUploadData: function(serverResponseObject, serverResponseText){
        // 사용자 로직
        return serverResponseObject;
    }
});
```

### ➤ 상세설명

업로드 완료 후 서버에서 응답객체와 응답문을 얻는다.

컨트롤 내에 진행되는 모든 업로드가 완료 되면, 서버에서 수신된 파일을 저장한 직후에 응답문을 보내 주는데 그 응답문의 JSON 형태의 객체 와 서버 응답문 전체에 해당되는 문자열 값의 형태도 둘 다 얻을 수 있다.

이 이벤트는 서버로부터 데이터를 받은 직후, 받은 데이터를 확인 및 수정이 가능하며, 암호화 모듈과 연동하는 경우에도 사용할 수 있다.

이 함수를 선언한 경우에는 반드시 파일 객체를 리턴해야 한다. 암호화된 문자열을 복호화 한 경우에도 반드시 그 최종 결과 값을 객체로 변환해서 리턴해야 그 결과를 기반으로 업로드 컨트롤의 정보 들이 최종 업데이트 된다.

이 함수에서 널을 리턴하면 서버에서 수신된 모든 자료가 널로 응답된 것처럼 동작한다.



전달 받은 JSON 객체의 모든 정보를 변경 가능하지만 key 값은 IBUpload 내부적으로 사용되므로 key 값을 수정해서는 안된다.

#### ➤ 예제 #1

**// 서버로부터 데이터를 가공하지 않고 그대로 적용하는 예**

```
$('#myUpload').IBUpload({
  onUploadData: function(serverResponseObject, serverResponseText){
    return serverResponseObject;
  }
});
```

#### ➤ 예제 #2

**// 서버로부터 암호화된 데이터를 복호화 하는 예**

```
$('#myUpload').IBUpload({
  onUploadData: function(serverResponseObject, serverResponseText){
    // 복호화 처리 (복호화 함수가 별도로 있다고 가정)
    var decrypt_data = fnDecryption(serverResponseText);
    // 복호화된 데이터를 객체 변환하여 리턴
    return JSON.parse(decrypt_data);
  }
});
```

#### ➤ 버전

7.3.0.0

## onUploadFinish 이벤트

컨트롤 내에 진행되는 모든 업로드가 완료 되면 이벤트가 발생한다.

### ➤ 문법

```
$(Selector).IBUpload({
  onUploadFinish: function(serverResponseObject , replacedResponseObject){
    // 사용자 로직
  }
});
```

### ➤ 상세설명

파일의 추가, 삭제, 업로드 등 컨트롤에 포함된 파일들의 상태가 변경되는 등의 모든 사항에 대하여 이벤트가 발생한다.

### ➤ 정보

인자명	자료형	기본값	설명
<b>serverResponseObject</b>	string	없음	onUpdateData 에서 가공하기 전에 서버에서 업로드 결과로 직접 받은 파일 객체
<b>replacedResponseObject</b>	string	없음	onUpdateData 에서 가공한 파일 객체

### ➤ 예제

```
$('#myUpload').IBUpload({
  onUploadFinish: myUpload_OnUploadFinish
});

function myUpload_OnUploadFinish(orgObject, newObject) {
  console.log("[서버응답 객체]");
  console.dir(orgObject);

  console.log("[onUpdateData 에서 변경된 최종 결과물의 객체]");
  console.dir(newObject);
}
```

## ➤ 버전

7.3.0.0

## onUploading 이벤트

컨트롤 내에서 서버로 파일을 업로드를 진행하는 동안 전체 진행율의 값을 이벤트로 발생한다.

### ➤ 문법

```
$(Selector).IBUpload({
  onUploading: function(number){
    // 사용자 로직
  }
});
```

### ➤ 상세설명

컨트롤 내에서 서버로 파일을 업로드를 진행하는 동안 전체 진행율의 값을 이벤트로 발생한다.

업로드가 시작되면 0 값 (=0%) 으로 이벤트가 한번 발생하고, 완료된 순간에도 100 (=100%를 의미) 값 으로 이벤트가 발생한다.

파일의 용량이 크면 0~100 사이의 값 들로 수시로 동일한 이벤트가 여러번 발생하지만, IE 9 이하에서는 시작시 0 과 완료시 100 만 발생한다.

여러개의 파일들을 한번에 전송하는 경우에는 그 평균 값으로 진행율이 제공되며, autoUpload 가 true 인 경우에는 이미 전송 중인 파일들이 있을 경우 진행중인 상황과 추가하여 업로드 하는 모든 사항들이 합산된 평균 값으로 반영된다.

100 까지 완료된 직후 서버 측의 사정으로 오류가 발생하는 경우가 있는데, 이 이벤트에

서 제공하는 100 이라는 값은 업로드할 파일 전체를 서버쪽으로 100% 전송을 완료했다는 것이지, 모든 파일을 성공적으로 서버에 저장했다는 것을 의미하지는 않는다.

인자명	자료형	기본값	설명
<b>Percent</b>	number	없음	업로드의 전체 진행율을 숫자로 제공한다. 0 ~ 100 사이의 숫자.

#### ➤ 예제

```
$('#myUpload').IBUpload({
  onUploading: function(iPercent){
    console.log(iPercent + " % 업로드가 진행중입니다.");
  }
});
```

#### ➤ 버전

7.3.0.7

## D. 자바스크립트 유틸

ibuploadinfo.js 파일에 다음과 같은 유틸이 포함되어 있다.

항목	설정 내용
_IBUpload_GetFileNamesFromFiles	files문자열로 부터 간단한 파일명 리스트만 얻는다. 단순한 파일을 나열하여 표시하고 싶을 때 사용한다.
_IBUpload_mFileName2Date	날짜와 시각 설정 얻는다
_IBUpload_mFileProgress	진행율 표시를 설정한다.

## E. 메시지 코드

OnMessage 이벤트에서는 다양한 메시지가 발생한다. 경고메시지, 정보 메시지 등 다양한 메시지에 대한 코드를 정리하면 다음과 같다.

### 정보 메시지

구분	msgID	message
업로드	INFO-001	업로드를 시도하였습니다.
	INFO-002	업로드를 완료하였습니다.
다운로드	INFO-011	다운로드를 시도하였습니다.
	INFO-012	다운로드를 완료하였습니다.

### 오류 메시지

구분	msgID	message
----	-------	---------

파일 제한	ERR-001	업로드할 파일의 개수가 너무 많습니다.
	ERR-002	업로드할 파일의 용량이 너무 큼니다.
	ERR-003	업로드할 파일들의 전체 용량이 너무 큼니다.
	ERR-004	업로드 불가능한 파일 형식 입니다.
	ERR-010	0Byte인 파일이 있습니다.
업로드 오류	ERR-000	전송이 진행중입니다.
	ERR-009	중복된 파일이 있습니다.
	ERR-011	업로드 중에 오류가 발생하였습니다.
	ERR-012	업로드 후에 서버 응답문에 오류가 있습니다.
	ERR-013	서버보안상 업로드가 제한된 파일 형식입니다.
Exception 오류	ERR-998	Exception 오류가 발생하였습니다.
API 사용오류	ERR-999	올바른 API 사용이 아닙니다. 매뉴얼을 확인하 시고 바르게 사용하여 주십시오.

## F. 브라우저 버전별 제약사항

브라우저의 종류나 버전별로 아래와 같은 제약사항이 있다.

### 한번에 여러개의 파일 선택 기능

종류	브라우저 버전	지원 현황
MS브라우저	MS-IE 7	( IE7 은 지원하지 않음 )
	MS-IE 8	※ 동시에 1 개 파일만 선택 가능
	MS-IE 9	※ 동시에 1 개 파일만 선택 가능
	MS-IE 10	가 능
	MS-IE 11	가 능
	MS Edge	가 능
기타	FireFox	가 능
	Chrome	가 능
	Opera	가 능
	Safari	가 능

### 업로드 파일 용량 제한 기능

종류	브라우저 버전	Client 파일용량 제한	Server파트 파일용량 제한
MS브라우저	MS-IE 7	( IE7 은 지원하지 않음 )	-
	MS-IE 8	불가 - 용량이 0 으로 표기	가 능
	MS-IE 9	불가 - 용량이 0 으로 표기	가 능



	MS-IE 10	가 능	가 능
	MS-IE 11	가 능	가 능
	MS Edge	가 능	가 능
기타	FireFox	가 능	가 능
	Chrome	가 능	가 능
	Opera	가 능	가 능
	Safari 5.0.x	가 능	가 능
	Safari 5.1.x	불가 - 용량이 0 으로 표기	가 능