MST TECHNOLOGY
THE LEADER IN DIGITAL IMAGING AND DOCUMENT MANAGEMENT SOLUTIONS

eViewer HTML5 Integration Guide

A MS Technology Product

Digital Imaging and Document Management Solution

Version 2.8

Licensing and Copyright Information

The software described in this guide is furnished under license agreement and is used in terms of accordance only.

The information provided in this document is used as a guide only and is subject to change without any notice. MS Technology reserves the rights to change and update their product or make changes in the context without any obligation to notify any person for such changes.

MS Technology,
P.O. Box 471843,
Charlotte, NC 28247
USA

Tel: 704-544-3403
Fax: 704-544-0262
Email: info@ms-technology.com
Website: www.ms-technology.com

If you find a typographical error in this guide, or if you have thought of a way to make this guide better, we would love to hear, feel free to share your thoughts with us at **info@ms-technology.com**.

Table of Contents

Preface

This developer's guide provides comprehensive information about the exposed APIs available in the eViewer.

What is in this Guide?

This guide contains descriptions of APIs to assist developers with the eViewer application for development to build connectors, customize the application, and develop automated processes.

Upload Document

uploadDocument (viewerURL, documentUrl, password, isHTML)

Summary:

Upload document for conversion. Although you will receive a response to an upload request quickly, the actual communication process is asynchronous. Therefore, receiving a successful response to an upload request does not imply the document. This has finished converting or that it is immediately available for viewing.

Parameters:

viewerURL*:* URL to eViewer(HTML5). The format of the URL will be:
<protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

documentUrl: URL of a document to be converted.

Password: This is to open the password protected file. It is by default, the value of this parameter must be NULL.

The password will go in the format: PositionOfTheFile_PasswordOfFile, in case of single file and PositionOfTheFile_itsPassword_separator_PositionOfTheFile_itsPassword, in case of multiple password protected files. For example, if you are trying to open three files, with first and third file is password protected, the format of the password will be 1_test_separator_3_test1, where test and test1 are the passwords and 1 and 3 are the positions of the file.

isHTML: Boolean variable is to check the request whether it is for eViewer (HTML5) or eViewer AJAX.

Returns:

UID assigned to the document. For ex: UID: 00000100-0000-0010-8000-00AA006D2EA4.

You should be sure to keep the document's UID since it is required for other document actions such as viewing and deleting.

In case of password protected file, it will return string containing TRUE/FALSE separated by semicolon (;). It is TRUE, if file is password protected, otherwise FALSE.

uploadDocumentByStream (viewerURL,fileListArr,password,isHTML)

Summary:

Upload document from file chooser into the viewer.

Parameters:

Viewer URL: URL to eViewer (HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstp2010:156.

fileListArr: The javascript file chooser object, having byte stream and file information.

password: It is by default carrying the value null, but in case of password protected file, it carries the assigned value.

isHTML: Boolean variable to check the request, whether it is for eViewer (HTML5) or eViewer AJAX.

Returns:
UID assigned to the document. For ex: UID: 00000100-0000-0010-8000-00AA006D2EA4.

You should be sure to keep the document's UID, since it is required for other document actions such as viewing and deleting.

In case of password protected file, it will return string containing TRUE/FALSE separated by semicolon (;). It is TRUE, if file is password protected, otherwise FALSE.

uploadCM8Document (documentID, viewer URL)

Summary:

Upload document from ICM into the viewer for conversion.

Parameters:

viewerURL*:* URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

documentID: ID of the document to be converted.

Returns:

UID assigned to the document. For ex: UID: 00000100-0000-0010-8000-00AA006D2EA4.

uploadSharepointDocument (viewerURL, documentID, isSharePointUploadByUrl, isHTML, userName)

Summary:

Upload document from SharePoint into the viewer for conversion.

Parameters:

viewerURL*:* URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

documentID: ID of a SharePoint document to be converted.

isSharePointUploadByUrl: Boolean value that must be TRUE.

isHTML: Boolean variable to check the request, whether it is for eViewer (HTML5) or eViewer AJAX.

userName: SharePoint user name.

Returns:

UID assigned to the document. For ex: UID: 00000100-0000-0010-8000-00AA006D2EA4. In case of multiple documents, returns UIDs separated by semicolon (;).

uploadSharepointDocument (viewerURL, documentID, isHTML, userName)

Summary:

Upload document from FileNet into the viewer for conversion.

Parameters:

viewerURL*:* URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

documentID: ID of a FileNet document - to be converted.

isHTML: Boolean variable to check the request, whether it is for eViewer (HTML5) or eViewer AJAX.

userName: FileNet user name.

Returns:

UID assigned to the document. For ex: UID: 00000100-0000-0010-8000-00AA006D2EA4. In case of multiple documents, returns UIDs, separated by semicolon (;).

View Document

viewDocument (viewerURL, sessionId, userId, userName, redirect, isHTML, isFileNet)

Summary:

Generate a URL for opening the viewer with input document UIDs.

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

sessionId: UID of a document generated during uploading a document or UID's of the multiple documents, separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

userId: User ID of the user. This ID will be used internally to manage annotation layers.

userName: User name of the user. This user name is stored with every annotation.

redirect: Boolean variable.

isHTML: Boolean variable to check the request, whether it is for eViewer (HTML5) or eViewer AJAX.

isFileNet: The Boolean value is TRUE, if the uploaded document is a FileNet document, otherwise FALSE.

Annotation Stream

getAnnotationStream (viewerURL, documentSessionId, isHTML)

Summary:

Retrieve annotation XML stream for an input document UID.

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

documentSessionId: The UID of a document generated during uploading a document or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

isHTML: Boolean variable to check the request whether it is for eViewer (HTML5) or eViewer AJAX.

setAnnotationStream (viewerURL, documentSessionId, annotationUrl, isHTML)

Summary:

Upload an XML annotation stream for a document to view it with the document within the eViewer**.**

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

sessionId: UID of a document generated during uploading a document or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

annotationUrl: URL of annotation XML document.

isHTML: Boolean variable to check the request whether it is for eViewer (HTML5) or eViewer AJAX.

setAnnotationStreamFromCM8 (viewerURL, documentSessionId, docID, isHTML)

Summary:

Set the annotation stream**.**

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

docId: ID of the document to be converted.

documentSessionId: UID of a document generated during uploading a document - or UID's of the multiple documents, separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

isHTML: Boolean variable to check the request, whether it is for eViewer (HTML5) or eViewer AJAX.

Markup Stream

getMarkupStream (viewerURL, documentSessionId, isMarkUp, isHTML)

Summary:

Retrieve Markup XML file location URL.

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

documentSessionId: The UID of a document generated during uploading a document - or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

isMarkUp: Takes Boolean value. In case of strikeout isMarkup is false and in case of checkpoint is true, otherwise null.

isHTML: Boolean variable to check the request whether it is for eViewer (HTML5) or eViewer AJAX.

setMarkupStream (viewerURL, documentSessionId, annotationUrl, isMarkUp)

Summary:

Upload an XML markup stream for a document to view it with the document within the eViewer**.**

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

sessionId: UID of a document generated during uploading a document or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

annotationUrl: URL of annotation, XML document.

isMarkUp: Takes Boolean value. If true, mark up XML file is added, otherwise NULL.

Drag and Drop Stream

getDragAndDropStream (viewerURL, documentSessionId)

Summary:

Retrieve drag and drop XML stream for an input document UID.

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

documentSessionId: UID of a document generated during uploading a document or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

setDragAndDropStream (viewerURL, annotationUrl, documentSessionId)

Summary:

Upload an XML drag and drop stream for a document to view it with the document within the eViewer**.**

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

documentSessionId: The UID of a document generated during uploading a document - or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

annotationUrl: URL of annotation XML document.

Thumbnail Document

genarateThumbnail (viewerURL, sessionId, userId, height, width, pageNo, download)

Summary:

This API generates the thumbnail image of the document. Thumbnails maintain the proper aspect ratio of the page and are bounded by the dimensions, specified in the parameters.

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

sessionId: The UID of a document generated during uploading a document - or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

height: Height of the thumbnail image that has to be created.

width: Width of the thumbnail image that has to be created.

pageno: Page number of the document of which the thumbnail image has to be created.

download: Boolean variable.

Returns:

Thumbnail image of the page.

Swap Pages

swapPages (viewerURL, sessionId, swapPages, rotation)

Summary:

Interchange pages of the document.

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

sessionId: The UID of a document generated during uploading a document - or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

swapPages: Page Numbers to be swapped separated by hyphen (-).

rotation: Value that rotate all the pages of the document.

Export Document

exportDocument (viewerURL, sessionId, userId, download)

Summary:

Exports the document in PDF format.

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

sessionId: The UID of a document generated during uploading a document - or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

userId: User ID of the user. This ID will be used internally to manage annotation layers per user.

download: Boolean variable.

Returns:

PDF file stream.

Remove Document

removeDocument (viewerURL, sessionId, isHTML)

Summary:

This API permanently deletes all content and the links associated with the document.

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

sessionId: The UID of a document generated during uploading a document - or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

isHTML: Boolean variable to check the request, whether it is for eViewer (HTML5) or eViewer AJAX.

Returns:

True, if the document is successfully removed.

Document Status

getDocumentStatus (viewerURL, sessionId)

Summary:

Used to determine the current status of the uploaded document, if it has been successfully converted or not.

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

sessionId: The UID of a document generated during uploading a document - or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

Returns:

Status of the document. It can be from one of the following:

DOWNLOADING: The document is being downloaded from the provided document URL.

DOWNLOADED: The document has been downloaded successfully.

CONVERTING: The document is being converted to PDF.

UPLOADED: The document has been converted to PDF successfully.

ERROR_DURING_CONVERSION: There was an error converting document to PDF.

UNAVAILABLE: For any other unknown error.

Extract Text from Document

extractTextFromDocument (viewerURL, sessionId, userId, pageNo)

Summary:

To extract text from pages of PDF and MS Office documents.

Parameters:

viewerURL: URL to eViewer(HTML5). The format of the URL will be: <protocol>://<IPaddress>:<port number>. For ex: http://mstsp2010:156.

sessionId: The UID of a document generated during uploading a document - or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

userId: User ID of the user. This ID will be used internally to manage annotation layers.

pageNo: Page number of PDF or Office document, whose text is to be extracted.

Returns:

The extracted text is from 'PDF' or 'MS Office' document.

Hide Blank Pages

showOrHideBlankPages (hideBlankPages, viewerFrameId)

Summary:

This API hides all the blank pages present in the document tab in focus.

Parameters:

hideBlankPages*: Set 'true' to hide all blank pages or 'false' to display all pages.*

viewerFrameId: ID of the iframe hosting the eViewer.

Search Annotations

searchAnnotations (searchText, matchcase, searchBookmarks, searchAnnotations, viewerFrameId)

Summary:

This API searches the annotations and sticky notes/bookmarks present on a document. All saved & currently unsaved annotations are searched.

Parameters:

searchText: Text to search.

matchCase: Set to 'true' to match case while searching or else 'false'.

searchBookmarks: Set to 'true' to search bookmarks/sticky note or else 'false'.

searchAnnotations: Set to 'true' to search all remaining annotations or else 'false'.

viewerFrameId: ID of the iframe hosting the eViewer.

Event Handling

The callback functions are implemented in customEvent.js placed in the web folder of the release, that is, eViewer(HTML5) v1.1.51\web\customEvent.js.

The custom event handling on some viewer features:

- Save document - invoked when the user saves the document.
- Page navigation – invoked on traversing the page.
- Zooming – invoked on zoom in or zoom out the page.
- Hide Annotation – invoked on hiding or showing the annotations.
- Split – invoked on splitting the document.
- Thumbnails – invoked on hiding or showing the thumbnails.
- TreeView – invoked on hiding or showing the tree view.
- Annotation Selected – invoked on selecting the annotation.
- Annotation Property Change – invoked on changing the property of the annotation.
- SideNote – invoked on creating a side note.
- Text Markups – invoked on creating a text mark up.
- WaterMark – invoked on applying watermark on the page of the document.
- Layer Manager – invoked on selecting or deselecting a user from the layer manager component.

Servlet API

Upload File as Stream

- **POST / InsertByteArray**

  This API will accept byte array of the file as input and return the UUID of the document.

  HTTP Method

  POST

  Request Body Parameters

Name	Description	Details
File data	Byte array of the File	ByteArray

  Response Body

  If successful, this method returns UUID of the file.

Name	Description	Details
UUID	UUID of the file.	String

Scanner APIs

isScannerInstalled ()

Summary:

Check whether MSTScanner is installed on the computer or not.

Returns:

Boolean TRUE or FALSE. Returns TRUE, if MSTScanner is installed, otherwise FALSE.

installScanner()

Summary:

Install MSTScanner on the computer.

Viewer Callback Events

Document saved notification (SaveNotificationJavaCallback(sessionID, dataURL))

Summary: This event is invoked by the viewer to notify the integrating web page, where document has been saved by the user. Viewer passes updated document, annotations, textmarkup, checkpoint, sidenote data, in the form of HTTP/HTTPS URL to the parent web page.

The event called to proceed further is as follows:

Parameters:

Session ID: The UID of a document generated during uploading a document - or UID's of the multiple documents separated by comma (,). For Example: "00000100-0000-0010-8000-00AA006D2EA4, 00000105-0000-0010-8000-00AA006D2EA4".

Data URL: The object array data are documentUrl, annotationUrl, strikeOutUrl, checkpointUrl, and sessionID. For example:

```
var dataURL={
        document_Url:documentUrl,
        annotation_Url:annotationUrl,
        strikeout_Url:strikeOutUrl,
        checkpoint_Url:checkpointUrl,
        sessionID:sessionID
  }
```

Sample Code:

```
function SaveNotificationJavaCallback(sessionID, dataURL)
{
}
```

Viewer Callback Messages

eViewer messages can be handled by attaching an event listener (i.e.

window.addEventListener("message", handleViewerMessages, true);).

- **SearchResults**
  This message is dispatched as a response to “searchAnnotations” API call.

  Action: “search”
  Data: Search Results

- **ViewerFileReloaded**
  This message is dispatched as a response to a user action that caused the viewer to reload. (e.g. Inserting a document into viewer).

  Action: “viewerLoadedAgain”
  Data: None

  e.g.

```
function handleViewerMessages(event) {
    if(event.data.action=="search")
    {
        //console.log( JSON.stringify(event.data)));
    }else if(event.data.action == "viewerLoadedAgain"){
        //ToDo
    }
}
```

- **Document Switch Notification**
  This message is dispatched when a user switches between documents in eViewer.

  Action: “tabSwitchNotification”
  Data: An object containing the data related to the document being switched to. E.g. Status of blank pages being displayed in the document i.e. hideBlankPages: true/false.

FileNet Authentication

Viewer provides an interface `FileNetSubjectGetter` for authentication. The interface provides a `getData` method which returns a hash table containing below parameters in a set of key value pair.

Key	Value	Type
subject	An authenticated object of JAAS subject	javax.security.auth.Subject
FileNetServerURL	URL to FileNet server in the format. http://<server_name>:<port_number>	string
Object_Store_Name	Name of the object store	string

Viewer provides a default implementation to this interface in "**MSTServer\web-inf\classes\com\filenet\FileNet\MSTFileNetSubjectGetter.class**".

Steps to replace default implementation:

1. Implement `FileNetSubjectGetter` interface.
2. Place the implementing class files at "**MSTServer\web-inf\classes\com\filenet\FileNet**".
3. Update the implementing class name in `FileNetSubjectClassName` parameter of "**MSTServer\web-inf\classes\com\FileNetConfigIP.properties**" file.

FileNetSubjectGetter Interface

```
package com.filenet;
import java.util.Hashtable;

public interface FileNetSubjectGetter
{
public abstract Hashtable getData(String userName);
}
```

For example, if you have implemented your class by name `MyFileNetSubjectGetter` and placed `MyFileNetSubjectGetter.class` in **MSTServer\web-inf\classes\com\filenet** folder then provide `FileNetSubjectClassName=com.filenet.MyFileNetSubjectGetter` in `FileNetConfigIP.properties` file.