

23rd & Walnut

Document Viewer jQuery Plugin

Documentation and help file

Saleem El-Amin
2/23/2012

Thank you for purchasing the Document Viewer jQuery plugin.

If you have any questions that are beyond the scope of this help file, please feel free to email via my profile page contact form.

<http://codecanyon.net/user/23andwalnut/profile>

Thanks so much!

Contents

- Introduction..... 3
- Quick Start 4
- Using The Plugin 5
- IMPORTANT NOTES: 7
- Options 9
- Dependency Loader..... 10
- Debugging..... 11
- FAQs..... 12
- FREEBIES 13

Introduction

Document Viewer is a jQuery plugin that allows you to load several file formats into a web page using a number of open source utilities.

The file formats that Document Viewer supports are:

1. **PDF Files**
2. **Text Files**
3. **Code** - bsh, c, cc, cpp, cs, csh, css, cyc, cv, htm, html, java, js, m, mxml, perl, php, pl, pm, py, rb, sh, xhtml, xml, xsl, sql, vb
4. **Images** - png, jpg, jpeg, gif
5. **Audio** - mp3, m4a, oga, webma, fla
6. **Video** - m4v, ogv, ogg, webmv, flv, mpg, mpeg, mov, divx, avi, wmv

Quick Start

1. Make sure you have included the following in your page:

- a. jQuery
- b. The yepnope plugin
- c. The document viewer plugin
- d. The document viewer css file

```
<link rel="stylesheet" href="../../documentViewer/css/style.css">
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.7.1/jquery.min.js"></script>
<script type="text/javascript" src="../../documentViewer/libs/yepnope.1.5.3-min.js"></script>
<script type="text/javascript" src="../../documentViewer/ttw-document-viewer.min.js"></script>
```

2. Initialize the document viewer plugin. The viewer will load in the html element you use to load the plugin. In the code below, the document viewer will load on the element with an id of 'document-preview'

```
var documentViewer = $('#document-preview').documentViewer();
```

3. Load a document. Be sure to use an absolute url or a relative file system path (for text and code)

```
documentViewer.load('http://absolute/url/to/file.pdf');
documentViewer.load('../relative/path/to/file.txt');
```

4. If you are not linking directly to the file, you can specify the file extension in the options object.

```
documentViewer.load('http://absolute/url/to/1', {extension: 'pdf'});
```

Using The Plugin

Using the plugin is very simple. It has three methods: load, close, and getDocumentType. See below for examples.

```
//initialize the document viewer. The viewer will load in the html element you
//use to load the plugin. In the code below, the document viewer will load on
//the element with an id = document-preview
var documentViewer = $('#document-preview').documentViewer();
```

```
//load a document
documentViewer.load('http://absolute/url/to/file.pdf');
```

```
//load a document with a url that doesn't contain an extension
documentViewer.load('http://absolute/url/to/1', {extension:'pdf'});
```

```
//load a document passing all of the options
documentViewer.load('http://absolute/url/to/file.pdf', {
    height:600;
    width:500;
    extension:'pdf',
    autoplay:true,
    autoLoadDependencies:true,
    debug:false,
    callback:function(){
        alert('document loaded');
    },
    jPlayer:{
        //you can play any jPlayer options here
        warningAlerts:false
    }
});
```

```
//close the currently open document
documentViewer.close();
```

(continued on next page)

```
//getDocumentType has two uses.  
//1. Get the document type. The script will return 'pdf', 'txt', 'code', 'video', 'audio',  
//or 'image'  
//2. Determine if a file can be opened by the document viewer (based on it's type)  
  
//get the document type. This code will return 'video' as the document type  
documentViewer.getDocumentType('http://absolute/url/to/file.m4v');  
  
//determine if we can open a file  
if(documentViewer.getDocumentType('http://absolute/url/to/file.m4v') !== false){  
    //this document is a type that can be loaded, so let's load it now  
    documentViewer.load('http://absolute/url/to/file.m4v')  
}  
else{  
    //we can't open this file.  
}
```

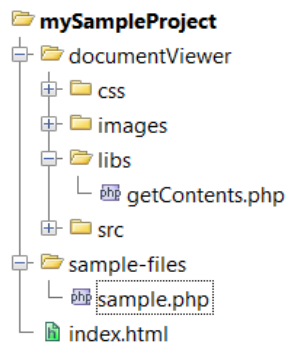
IMPORTANT NOTES:

1. **If you run into issues. Your first step should be to turn on the plugin's debug option.** This will display debugging messages to your browser's console.

```
var documentViewer = $('#document-preview').documentViewer({  
    debug: true  
});
```
2. **You MUST use absolute URLs for any document that is not text or code. For text and code you MUST use relative file system paths.** This means that the documents must exist on your server. The relative file system path should be the relative path FROM the getContents.php file in the documentViewer/libs folder TO the location to the document.

EXAMPLE:

If you use the following directory structure:



The relative path FROM getContents TO sample.php is: `'../../sample-files/sample.php'`.

This means that a valid value for a pdf might look like: `'http://www.mysite.com/sales-forcast.pdf'`
and valid value for a php file might be: `'../../sample-files/sample.php'`

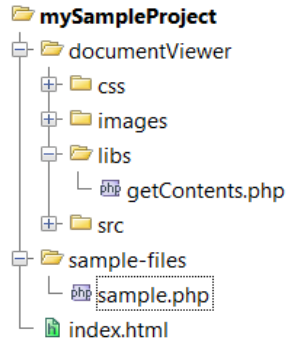
(continued on next page)

IMPORTANT NOTES (continued):

3. If you are using the dependency loader, you may need to change the `options.path` value. This value tells the plugin where it is located relative to the file you are using it in.

EXAMPLE:

If you use the following directory structure and the plugin is used in `index.html`



The `options.path` value will simply be `'documentViewer/'`. Please note the trailing slash on the end. This is very important.

```
var documentViewer = $('#document-preview').documentViewer({
  path: 'documentViewer/'
});
```

Options

There are several options that you can pass into the document viewer plugin. The full list of options is below.

Option	Default	Valid Values	Description
height	auto	Integer	The height of the document viewer
width	500	Integer	The width of the document viewer
extension		String (any extension recognized by the plugin)	The extension of the file being loaded. This is not required unless the path to the file does not include the file extension.
callback		A function	A callback function that will be run when the document is loaded.
autoplay	true	True, false	Determines if audio/video files start playing immediately after they are loaded
autoLoadDependencies	true	True, false	Determines if the plugin should auto load the 3 rd party plugins (pdf.js, jPlayer, flowplayer, prettify). If this is set to false, the dependencies have to be manually added to the page.
debug	false	True, false	Determines if the plugin should display debug messages (in the console, except for jPlayer messages which are alerts)
jPlayer	{}	jPlayer Options	This allows you to override any of the default jPlayer options.
path	<code>`documentViewer/'</code>	String	This is required for the dependency loader and tells the plugin where it is located relative to the file it is used in.
enableTextAndCode	true	Boolean	Whether to allow .txt and code documents to be loaded. If set to true, the plugin will use the server side script <code>getContents.php</code> in the <code>libs</code> folder

Dependency Loader

OVERVIEW

- The plugin uses several 3rd party plugins to provide its functionality. These plugins are [pdf.js](#), [iPlayer](#), [Flowplayer](#), and [Google Code Prettify](#)
- The dependency loader will auto load these plugins when they are required.
- The dependency loader uses [yepnope](#)
- Using the dependency loader is optional, but it is turned on by default.

If you are using the dependency loader, then the only resources you need to include in your page are jQuery, YepNope, and the Document Viewer plugin.

```
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.7.1/jquery.min.js"></script>
<script type="text/javascript" src="../../documentViewer/libs/yepnope.1.5.3-min.js"></script>
<script type="text/javascript" src="../../documentViewer /ttw-document-viewer.min.js"></script>
```

If you decide not to use the dependency loader, you will need to manually include the 3rd party scripts in your page:

```
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.7.1/jquery.min.js"></script>
<script type="text/javascript" src="../../documentViewer/libs/yepnope.1.5.3-min.js"></script>
<script type="text/javascript" src="../../documentViewer/libs/pdfjs/pdf.js"></script>
<script type="text/javascript" src="../../documentViewer/libs/google-code-
prettify/src/prettify.js"></script>
<script type="text/javascript" src="../../documentViewer/libs/flowplayer/flowplayer-3.2.6.min.js"></script>
<script type="text/javascript" src="../../documentViewer/libs/jquery-jplayer/jquery.jplayer.js"></script>
<script type="text/javascript" src="../../plugin/ttw-document-viewer.min.js"></script>
```

Debugging

If you run into issues. Your first step should be to turn on the plugin's debug option. This will display debugging messages to your browser's console.

```
var documentViewer = $('#document-preview').documentViewer({  
    debug: true  
});
```

Possible debugging messages:

1. Invalid File Type - This means you are attempting to open a file type that the document viewer plugin does not support.
2. Invalid File Type. Please set enableTextAndCode option to true – This means you are attempting to open a text or code file, but you have enableTextAndCode = false. You must change this to true to open these types of files.
3. jPlayer Ready – This means that jPlayer is ready. If you are attempting to open a file format that is supported by jPlayer and you do not see this message in the console, then something is wrong with your jPlayer configuration.
4. Error loading file ('some/file/name '). Please make sure that the path is correct – This means that the path you supplied to the file is incorrect or that the plugin can't access the location of the file.
5. There was an error loading the dependency – This means that the dependency loader was unable to load one of the dependencies. This is most likely due to an incorrect value for options.path. Please see the Important Notes section for more information.

FAQs

This section will be updated with more information soon.

1. The plugin doesn't work.

- a. Make sure you have read the documentation.
- b. Turn on the debug option. See section on debugging if you are unsure how to do this.
- c. Open the developer console, check for errors.
- d. If you are unsure how to open your browsers developer tools, see the links below

2. I don't know how to use my browser's developer tools

As you begin to write more code, these tools will become invaluable to you. It is best to get familiar with them asap.

- a. Google Chrome - <http://code.google.com/chrome/devtools/docs/overview.html#access>
- b. Firefox - <http://getfirebug.com/>
- c. IE9 (Press F12) - [http://msdn.microsoft.com/library/gg589507\(VS.85\).aspx](http://msdn.microsoft.com/library/gg589507(VS.85).aspx)
- d. IE 6, 7, and 8 - <http://getfirebug.com/firebuglite>

3. My PDF isn't displaying correctly.

As mentioned in the description on Codecanyon, pdf.js is still in beta and there are still some instances in which pdfs may not display correctly. Unfortunately, this is not something that I have any control over. Fortunately, pdf.js is being actively developed by a very talented team and the plugin will improve over time.

FREEBIES

Visit www.codebasehero.com for free files