

Changes Introduced != "release notes"

Please note, this document is intended to "go beyond" the basic release notes and give more details where needed. Such as Release notes may say "new feature - added single page view to WDV" where the Changes Introduced will tell you more useful information such as "to enable this feature you need to set the singlepage config option to true here..."

For basic Release notes, please see

[INFO: 11.5 Full Release Notes](#)

Issues / Fixes in FixPacks

KNOWN ISSUE: WDV Annotations from older versions loading as ReadOnly in 11.5

The readonly property of annotations was added to WebDocumentViewer in a previous release, thus annotations saved to XMP before that feature do not have the setting present.

There is a known issue in DotImage 11.5 (but not in 11.4) where XMP read in from those old versions which is missing the readonly tag are defaulting to True instead of False.

This issue is expected to be addressed in a fixpack soon.

In the meantime, the workaround is that after you finish loading, you can iterate through all annotations on all pages and ensure the readonly flag is false.

Example Code:

```
/// The following code requires that you pass in the viewer object you're using /// example
usage assumes your WebDocumentViewer is named "_viewer" /// Example //
ToggleAllAnnosReadOnly(_viewer, false); // This code loops through all pages and calls the
ToggleReadOnlyOnPage with the // passed in readOnlyValue (defaults to false if null) // the
ToggleReadOnlyOnPage will do the actual heavy lifting for each page's annos function
ToggleAllAnnosReadOnly(wdv, readOnlyValue) { if (readOnlyValue == null) { readOnlyValue =
false; } var pageCount = wdv.getDocumentInfo().count; if (pageCount != null && pageCount > 0)
```

INFO: Changes Introduced in DotImage 11.5

```
{ for (var i = 0; i < pageCount; i++) { ToggleReadOnlyOnPage(wdv, i, readOnlyValue); } } //  
this is to set the readonly flag of all annos on a specific page to the // readOnlyValue  
provided function ToggleReadOnlyOnPage(wdv, pageIndex, readOnlyValue) { var annos =  
wdv.getAnnotationsFromPage(pageIndex); if (annos != null) { for (var i = 0; i < annos.length;  
i++) { var anno = annos[i]; anno.readonly = readOnlyValue; anno.update(); } } }
```

The fix is expected (but not guaranteed) for 11.5.0.3 which has an ETA of early October 2024

Added LimitsPageCount to EmlDecoder, HtmlDecoder & TxtDecoder

In 11.5.0.2 and forward, there is now a new property called `LimitsPageCount` which you can use to increase the default page count limit.

NOTE: the page counts are limited for memory/performance reasons. Adjusting this value in production should only be done after testing thoroughly in your app/deployment to ensure it does not cause memory issues

Missing jquery-ui-1.13.1.min.css in NuGet distribution

The initial 11.5.0.156 release is missing a css file

The file in question is `jquery-ui-1.13.1.min.css`

It should have been installed as part of `Atalasoftware.Web.Document.Viewer` package. but somehow got missed

We're working on ensuring this gets included in the 11.5.0.1 fixpack release

In the meantime, you can either get it from installing the SDK from the main download site, where it can be found in

`C:\Program Files (x86)\Atalasoftware\DotImage 11.5\bin\WebResources\WebDocViewer`

or you can grab a copy here : [Download](#)

[DotImage Clientside Web Resources v11.5.0.0.zip](#)

NEW Features

PdfDecoder Multiprocessing

Overview

The PDF viewer has been upgraded with multiprocessing to support concurrent rendering of pages, aimed at reducing delays with complex and large pages. Performance impact may be affected as follows:

- Documents with vector content will experience performance improvements.
- High page-count documents with rasterized content may experience performance improvements.
- High page-count documents with vector content, especially if the vector quantity per page is small, may not show substantial performance improvement.
- Some PDF processing use case types may experience a degradation in performance as a result of parallel orchestration and system resource constraints. These use case types include the following:
 - o Using the multiprocessing feature for single page PDF documents.
 - o Using the multiprocessing feature for PDF documents without any graphics.
 - o Using the multiprocessing feature in one-core CPU environment

How To: Out of the Box

If you're using a default WebDocumentRequestHandler (not custom handling), all you need to do is set the WebDocumentViewer (and WebDocumentThumbnailer if you are using it) config multiprocessing setting to true)

multiprocessing: true

```
et _viewer = new Atalasoftware.Controls.WebDocumentViewer({ parent: $('#_containerViewer'),
toolbarparent: $('#_toolbar1'), serverurl: _serverUrl, allowannotations: true,
multiprocessing: true, forcepagefit: true, fitting: Atalasoftware.Utils.Fitting.Width, }); let
_thumbs = new Atalasoftware.Controls.WebDocumentThumbnailer({ viewer: _viewer, parent:
$('#_containerThumbs'), serverurl: _serverUrl, documenturl: _docUrl, allowannotations: true,
multiprocessing: true });
```

INFO: Changes Introduced in DotImage 11.5

For full information please see [HOWTO: Enable PdfMultiprocessing In WDV \(new in 11.5\)](#)

NON-BREAKING Changes

These are important changes but ones that don't directly require code changes or other large rewrites

New Decoders

We've added several new decoders in DotImage 11.5.

CommonDecoders

The new "common Decoders" (non MS_Office) do not require additional licensing outside of DotImage Document Imaging, they are

- **WebpDecoder**

.webp files: Decoding for newer web image formats.

- **TxtDecoder**

For text and other types of files

Can be used for any plain text file including xml

NOTE there is no "pretty printing" / formatting of code. If you wish to have source code or XML, YAML etc properly indented, you'll want to run such files through the relevant pretty printing/lint utilities before passing them to TxtDecoder

- **HtmlDecoder**

.html and .txt files: Expanded web and text file decoding.

- **EmlDecoder**

.eml files: Enhanced email decoding capabilities.

In order to implement them you must follow these steps:

If you're using NuGet packages, everything you need should be in the

INFO: Changes Introduced in DotImage 11.5

Atalasoftware.CommonDecoders.x64

or

Atalasoftware.CommonDecoders.x86

package depending on the "bitness" you need

If you're adding references manually from the installed SDK:

Add a reference to Atalasoftware.CommonDecoders.dll to your project (NOT available for .NET Framework 3.5)

Add the Perceptive Filters (64 or 32 bit depending on your platform target for your app) to your project as ALWAYS COPY or COPY IF NEWER or place them directly in the bin directory of the app

or if you installed the SDK the files are:

```
SYS11df.dll ISYSreaders.dll ISYSreadershd.dll Perceptive.DocumentFilters.dll
```

They can be found in

C:\Program Files (x86)\Atalasoftware\DotImage 11.5\bin\PerceptiveDocumentFilters\intel-32\

or

C:\Program Files (x86)\Atalasoftware\DotImage 11.5\bin\PerceptiveDocumentFilters\intel-64\

depending on which bitness you're targeting

Finally, Add the relevant decoder(s) to the RegisteredDecoders.Decoders collection in a STATIC CONSTRUCTOR for your class/program (do this once and only once)

```
RegisteredDecoders.Decoders.Add(new TxtDecoder() { Resolution = 200 });  
RegisteredDecoders.Decoders.Add(new HtmlDecoder() { Resolution = 200 });  
RegisteredDecoders.Decoders.Add(new EmlDecoder() { Resolution = 200 });  
RegisteredDecoders.Decoders.Add(new WebpDecoder());
```

Documentation / Help Files

INFO: Changes Introduced in DotImage 11.5

In previous versions, the ApiReference.chm and DevelopersGuide.pdf were installed to a folder named help to:

C:\Program Files (x86)\Atalasoftware\DotImage 11.x\Help\

These files are now provided as download links in the DotImage 11.5 start menu folder

Developers Guide

START->All Programs->Atalasoftware DotImage 11.5->DevelopersGuide

[Direct Download](#)

.NET API Reference

START->All Programs->Atalasoftware DotImage 11.5->ApiReference

[Direct Download](#)

WebDocumentViewer (clientside) API Reference

<https://atalasoftware.github.io/web-document-viewer/>

WebCapture (Clientside) API Reference

<https://atalasoftware.github.io/web-capture-service/>

WDV Setting e.FilePath in DocumentInfoRequested now Persists to other events

Previously, there was a bug where setting e.FilePath in the DocumentInfoRequested event did not persist the changed file name to ImageRequested and other events

This has been fixed in 11.5.

INFO: Changes Introduced in DotImage 11.5

This is very useful for situations where you might for instance call `openUrl` from the client and the request fetches a file from another location and you want to update the viewer to point to a copy of the file you've grabbed (cached)... say the user did `openurl(Document1234)` where `Document1234` was the ID of a document to be fetched from a remote server

You could write code to fetch `Document1234` and place it in say

```
/images/Document1234_someRandomNumber
```

Then set `e.FilePath` to `"/images/Document1234_someRandomNumber"`

With no further handling, now the viewer will now look for `Document1234_someRandomNumber` at that path instead of for just `Document1234` as the original open

This can be incredibly useful too for simple "cache breaking"

You request `Document1234` and it's local but you copy it to `Document1234_SomeGUID` and set that file path to the new name - thus breaking any caching done by file name by server or browser

BREAKING Changes

DLL Restructuring

In 11.5 you may notice some DLLs are "missing" versus previous versions. This is by design.

We distribute our SDK in sets based on Framework

our dlls for targeting .NET Framework 3.5 are in

```
C:\Program Files (x86)\Atalasoftware\DotImage 11.5\bin\3.5
```

Our DLLs for targeting .NET Framework 4.6.2 through 4.8.x are in

```
C:\Program Files (x86)\Atalasoftware\DotImage 11.5\bin\4.6.2
```

INFO: Changes Introduced in DotImage 11.5

Our DLLs for .NET 6 and p

C:\Program Files (x86)\Atalasoft\DotImage 11.5\bin\6.0

In older dotImage we had 3.5, 4.5.2, and 5.0 respectively

.NET Framework Minimum

The DLL restructure has a specific impact on the .NET Framework minimum, Any project you have that targets .NET Framework < 4.6.2 must be updated to .NET Framework 4.6.2 or later (but NOT .NET 5 or higher - stay with the 4.6.2-4.8.x (4.9 maybe someday?) for .NET Framework. .NET 5 and up is a totally different beast - see below

.NET 6+ Minimum

For those using our .NET 5 dlls previously, we have dropped .NET 5 and you must target .NET 6 or up (we support .NET 6, .NET 7, and .NET 8 currently - we will drop 6 and 7 in 11.6)

Please see [FAQ: Support for ASP.NET Core / .NET Core / .NET 5 / .NET 6 / .NET 7 / .NET 8](#)

WebDocumentRequestHandler - Hard Dependencies

WebDocumentRequestHandler (in support of our new CommonDecoders and Multithreading) has 2 hard dependencies at runtime. (Your solution will build but you'll get a crash if the following have not been installed:

Microsoft.CodeAnalysis.CSharp 4.5.0

Please note our dependency is on the very specific 4.5.0 version, and NOT on the latest - you must install the specific 4.5.0 version (for now, we're working on possibly being able to set it to 4.5.0 minimum in future)

NOTE That if you install the NuGet package

Atalasoft.dotImage.PdfReader.Multithreading.x64 or

Atalasoft.dotImage.PdfReader.Multithreading.x86, this should get installed for you

INFO: Changes Introduced in DotImage 11.5

automatically

We have also noticed that IISExpress (and possibly IIS) caches the install so once it's been installed once, you may not find that it complains)

The symptom of this dependency being missing will not show up at compile - but at runtime the WebDocumentRequestHandler will throw an exception and if you look at the network trace and error text is on it will mention that it's missing Microsoft.CodeAnalysis.CSharp 4.5.0 specifically

Perceptive Filter DLLs

These are supposed to be available in a NuGet package, but the PerceptiveFilter packages currently available are not correct. This is being investigated

For now, you'll need to grab them from the SDK (you would need to download and install it. The files are:

```
SYS11df.dll ISYSreaders.dll ISYSreadershd.dll Perceptive.DocumentFilters.dll
```

They can be found in

C:\Program Files (x86)\Atalasoftware\DotImage 11.5\bin\PerceptiveDocumentFilters\intel-32\

or

C:\Program Files (x86)\Atalasoftware\DotImage 11.5\bin\PerceptiveDocumentFilters\intel-64\

depending on which bitness you're targeting

Alternately, for your convenience you can download them here:

[PerceptiveDocumentFilters_11.5.0.0.zip](#)

Adding the Perceptive Filters

Once we get a NuGet package, you can just add that.

INFO: Changes Introduced in DotImage 11.5

However, the NuGet package DLLs can not be referenced, so you can't just add them as references. Instead, you need to add the DLLs to your project then set them to "COPY ALWAYS" or "COPY IF NEWER"

Dropped Support

Windows Server 2012 R2 and older

Microsoft has end of life'd Windows server 2012 R2 and older - minimum server OS is now Windows Server 2016 and up

Windows 8.1 and Older

Microsoft has end of life'd Windows 8.1 and older - minimum desktop OS is now Windows 10 and up

MacOS Catalina and older dropped

This applies to the Web Capture client-side component. The installable MacOS package that clients using pages with WebCapture need to install. The package no longer supports MacOS Catalina and older.

Atalasoft Knowledge Base

<https://www.atalasoft.com/kb2/KB/50434/INFO-Changes-Introduced-in-DotImage-...>