HOWTO: Use Many Framed Images Efficiently

When opening many separate AtalaImages to process and save into another file, it is generally best to use the FileSystemImageSource object. It efficiently loads individual frames and save out on a single active stream. It also tries to keep as little additional information in physical memory as possible. In most situations this will result in much less physical memory usage. Here is an example of opening two image files to save to a single Pdf:

C#

```
rivate void MergeAndConvertToPdf(string FirstFile,string SecondFile, string OutFilePath) { string[] s = new string[2]; s[0] = FirstFile; s[1] = SecondFile; FileSystemImageSource fsis = new FileSystemImageSource(s,true); //true does all frames in both files PdfEncoder encoder = new PdfEncoder(); using(FileStream OutStream = new FileStream(OutFilePath, FileMode.OpenOrCreate)) { encoder.Save(OutStream,fsis,null); } }
```

VB.NET

```
rivate Sub MergeAndConvertToPdf(ByVal FirstFile As String, ByVal SecondFile As String, ByVal OutFilePath As String) Dim s As String() = New String(1) {} s(0) = FirstFile s(1) = SecondFile Dim fsis As New FileSystemImageSource(s, True) 'True does all frames in both files Dim encoder As New PdfEncoder() Using OutStream As New FileStream(OutFilePath, FileMode.OpenOrCreate) encoder.Save(OutStream, fsis, Nothing) End Using End Sub
```

If additional processing needs to be done on images being controlled by a FileSystemImageSource, this article explains how to overload the FileSystemImageSource to have it process each frame individually:

HOWTO: Efficiently Process Images as they are Saved Into a PDF

Original Article:

Q10257 - HOWTO: Use Many Framed Images Efficiently

Atalasoft Knowledge Base
https://www.atalasoft.com/kb2/KB/50212/HOWTO-Use-Many-Framed-Images-Efficie...