

Outdated Content Notice

This article has been flagged by support for review/rewrite. The general concepts are likely sound but there may be better ways to approach the subject in more recent DotImage. Please contact support if you have questions

Article Content

A common task in the document imaging world is to scan many pages into a single file. DotTwain, along with DotImage can easily be used to accomplish this task. The two most popular multi page image formats are TIFF and PDF, so this article will concentrate on those.

To start off, lets first look at what it takes to scan a single page. It is important to know that the entire acquisition process ('acquisition' refers to the process of scanning a page and transferring it into your application) is event driven. This means that anything which happens during this time, must be dealt with in an event handler, including saving the image. In DotTwain, the acquisition is controlled by an instance of the Acquisition class. For any one application, there should be exactly one Acquisition object. This is because the Acquisition object connects to the TWAIN interface, and TWAIN does not allow more than one connection. The following acquisition events are of interest to us:

AcquireCanceled: Raised if the user cancels the acquisition process.

AcquireFinished: Raised when the scanner has finished scanning the last page.

ImageAcquired: Raised each time the scanner finishes scanning a page.

C#

```
acquisition myAcquisition = new Acquisition(this);
```

VB.NET

```
im WithEvents myAcquisition As New Acquisition(Me)
```

HOWTO: Scan a Multi-Page Document Into a PDF or TIFF

The reason we must pass the constructor a reference to the current window is that TWAIN uses its parent window's message loop for interacting with our program. At this point, we can choose a device to use. Since we don't know what type or how many scanners the user will have, let's let them choose. The ShowSelectSource method will display a list of detected scanners and let the user pick one.

C#

```
Device activeDevice = myAcquisition.ShowSelectSource();
```

VB.NET

```
Dim activeDevice As Device = myAcquisition.ShowSelectSource()
```

The Device object holds a reference to the scanner that was selected, which is controlled through our acquisition object. In this way, we can use the activeDevice object to control the scan. Before we start off the actual scan, we need to set the event handlers. This is done with delegates in C# and declared as 'Handles' in VB:

C#

```
myAcquisition.AcquireCanceled += new EventHandler(OnAcquireCanceled);  
myAcquisition.AcquireFinished += new EventHandler(OnAcquireFinished);  
myAcquisition.ImageAcquired += new ImageAcquiredEventHandler(OnImageAcquired); ... private  
void OnImageAcquired(object sender, AcquireEventArgs e) {} private void  
OnAcquireCanceled(object sender, EventArgs e) {} private void OnAcquireFinished(object  
sender, EventArgs e) {}
```

VB.NET

```
Private Sub OnImageAcquired(ByVal sender As Object, ByVal e As AcquireEventArgs) Handles  
myAcquisition.ImageAcquired End Sub Private Sub OnAcquireCanceled(ByVal sender As Object,  
ByVal e As EventArgs) Handles myAcquisition.AcquireCanceled End Sub Private Sub  
OnAcquireFinished(ByVal sender As Object, ByVal e As EventArgs) Handles  
myAcquisition.AcquireFinished End Sub
```

Now that all the events are set up, we can start the scan. We do this using the Device object that we created earlier:

HOWTO: Scan a Multi-Page Document Into a PDF or TIFF

C#

```
ctiveDevice.Acquire();
```

VB.NET

```
ctiveDevice.Acquire()
```

We can now turn our attention the meat of the process: actually saving the images. There are two main ways to save a multi page image file using DotImage. The first way is to have all of the images loaded into memory, and pass them all to the image encoder. The second way is to save the images one by one (append, in the case of TIFF) or to supply a reference to a file on disk rather than an image in memory (as in the PDF encoder). For more information on this topic, please see the article describing conversion between TIFF and PDF. In this article we will use the second method.

The PdfEncoder class does not have the ability to append to an existing file, which means that we must supply the encoder with handles to each of the images. To do this, we will first need to save the aquired images as a TIFF image. Because the TiffEncoder has the ability to append, this is easy:

C#

```
private void OnImageAcquired(object sender, AcquireEventArgs e) { if (e.Image != null) {  
TiffEncoder enc = new TiffEncoder(TiffCompression.Default, true); FileStream fs = new  
FileStream("outputTiff.tif", FileMode.OpenOrCreate, FileAccess.ReadWrite); enc.Save(fs,  
AtalaImage.FromBitmap(e.Image), null); fs.Close(); } }
```

VB.NET

```
private Sub OnImageAcquired(ByVal sender As Object, ByVal e As AcquireEventArgs) Handles  
myAcquisition.ImageAcquired If Not e.Image Is Nothing Then Dim enc As TiffEncoder = New  
TiffEncoder(TiffCompression.Default, True) Dim fs As FileStream = New  
FileStream("outputTiff.tif", FileMode.OpenOrCreate, FileAccess.ReadWrite)  
enc.Save("outputTiff.tif", AtalaImage.FromBitmap(e.Image), Nothing) fs.Close() End If End Sub
```

This will append each new image to "outputTiff.tif". The next step is to convert this file to

HOWTO: Scan a Multi-Page Document Into a PDF or TIFF

PDF. We can do this inside the AcquireFinished event handler because we know that all of the images have been saved into the temporary tiff file.

C#

```
private void OnAcquireFinished(object sender, EventArgs e) { PdfImageCollection col = new PdfImageCollection(); TifDecoder dec = new TiffDecoder(); FileStream fs = new FileStream("outputTiff.tif", FileMode.Open, FileAccess.Read); int frameCount = dec.GetFrameCount(fs); fs.Close(); for(int i=0; i< frameCount; i++) col.Add(new PdfImage("outputTiff.tif", i, PdfCompressionType.Auto)); FileStream outputStream = new FileStream("outputPdf.pdf", FileMode.OpenOrCreate, FileAccess.ReadWrite); PdfEncoder enc = new PdfEncoder(); enc.Save(outputStream, col, null); }
```

VB.NET

```
private Sub OnAcquireFinished(ByVal sender As Object, ByVal e As EventArgs) Handles myAcquisition.AcquireFinished Dim col As PdfImageCollection = New PdfImageCollection() Dim dec As TifDecoder = New TiffDecoder() Dim fs As FileStream = New FileStream("outputTiff.tif", FileMode.Open, FileAccess.Read) Dim frameCount As Integer = dec.GetFrameCount(fs) fs.Close() Dim i As Integer=0 Dim i As Integer For i = 0 To (frameCount - 1) col.Add(New PdfImage("outputTiff.tif", i, PdfCompressionType.Auto)) Next i Dim outputStream As FileStream = New FileStream("outputPdf.pdf", FileMode.OpenOrCreate, FileAccess.ReadWrite) Dim enc As PdfEncoder = New PdfEncoder() enc.Save(outputStream, col, Nothing) End Sub
```

The end result is a pdf file that contains all of the scanned images. The source code is attached in a zip file. The full code listing is shown below:

C#

```
public class AcquisitionClass { private bool _acquireCanceled; private Acquisition myAcquisition = new Acquisition(this); public void ScanImages() { _acquireCanceled = false; myAcquisition.AcquireCanceled += new EventHandler(OnAcquireCanceled); myAcquisition.AcquireFinished += new EventHandler(OnAcquireFinished); myAcquisition.ImageAcquired += new ImageAcquiredEventHandler(OnImageAcquired); Device activeDevice = myAcquisition.ShowSelectSource(); activeDevice.Acquire(); } private void OnImageAcquired(object sender, AcquireEventArgs e) { if (e.Image != null) { TiffEncoder enc = new TiffEncoder(TiffCompression.Default, true); FileStream fs = new FileStream("outputTiff.tif",FileMode.OpenOrCreate, FileAccess.ReadWrite); enc.Save(fs, AtalaImage.FromBitmap(e.Image), null); fs.Close(); } } private void OnAcquireCanceled(object sender, EventArgs e) { _acquireCanceled = true; } private void OnAcquireFinished(object sender, EventArgs e) { if (_acquireCanceled) { return; } PdfImageCollection col = new PdfImageCollection(); TiffDecoder dec = new TiffDecoder(); FileStream fs = new FileStream("outputTiff.tif", FileMode.Open, FileAccess.Read); int frameCount =
```

HOWTO: Scan a Multi-Page Document Into a PDF or TIFF

```
dec.GetFrameCount(fs); fs.Close(); for(int i=0; i< frameCount; i++) { col.Add(new PdfImage("outputTiff.tif", i, PdfCompressionType.Auto)); } FileStream outputStream = new FileStream("outputPdf.pdf", FileMode.OpenOrCreate, FileAccess.ReadWrite); PdfEncoder enc = new PdfEncoder(); enc.Save(outputStream, col, null); }
```

VB.NET

```
Public Class AcquisitionClass Private _acquireCanceled As Boolean Private myAcquisition As Acquisition = New Acquisition(Me) Public Sub ScanImages() _acquireCanceled = False Dim activeDevice As Device = myAcquisition.ShowSelectSource() activeDevice.Acquire() End Sub Private Sub OnImageAcquired(ByVal sender As Object, ByVal e As AcquireEventArgs) Handles myAcquisition.ImageAcquired If Not e.Image Is Nothing Then Dim enc As TiffEncoder = New TiffEncoder(TiffCompression.Default, True) Dim fs As New FileStream("outputTiff.tif", FileMode.OpenOrCreate, FileAccess.ReadWrite) enc.Save("outputTiff.tif", AtalaImage.FromBitmap(e.Image), Nothing) fs.Close() End If End Sub Private Sub OnAcquireCanceled(ByVal sender As Object, ByVal e As EventArgs) Handles myAcquisition.AcquireCanceled _acquireCanceled = True End Sub Private Sub OnAcquireFinished(ByVal sender As Object, ByVal e As EventArgs) Handles myAcquisition.AcquireFinished If _acquireCanceled Then Return End If Dim col As PdfImageCollection = New PdfImageCollection() Dim dec As TiffDecoder = New TiffDecoder() Dim fs As FileStream = New FileStream("outputTiff.tif", FileMode.Open, FileAccess.Read) Dim frameCount As Integer = dec.GetFrameCount(fs) fs.Close() Dim i As Integer For i = 0 To (frameCount - 1) col.Add(New PdfImage("outputTiff.tif", i, PdfCompressionType.Auto)) Next i Dim outputStream As FileStream = New FileStream("outputPdf.pdf", FileMode.OpenOrCreate, FileAccess.ReadWrite) Dim enc As PdfEncoder = New PdfEncoder() enc.Save(outputStream, col, Nothing) End Sub End Class
```

Original Article:

Q10129 - HOWTO: Scan a Multi-Page Document Into a PDF or TIFF

Atalasoft Knowledge Base

<https://www.atalasoft.com/kb2/KB/50310/HOWTO-Scan-a-MultiPage-Document-Into...>