HOWTO: Combine Multiple PDF Files With Fillable Forms (AcroForms)

Customers frequently use our PdfDocument class to combine multiple PDF files together, However, if you've ever tried combining two or more PDFs that have PDF fillable forms (AcroForms) you may have seen this error:

```
PdfException::{"source pages include multiple AcroForm objects that conflict"}
```

This is because PDF forms (AcroForms) are a complex mix of form objects and rely on embedded resources in the containing PDF. There is no way for the combine to merge these resources and combine two forms. Our PdfDocument component takes a "above all else, do no harm" approach which means any time an operation would result in loss of data or fundamental change, it should throw an exception rather than continue.

The intent is to give the programmer a means to gracefully handle the exception. Be that to gracefully fail or to take corrective action.

In the case of this AcroForm object conflict, there used to be no way to automatically handle the conflict in a way that would preserve the integrity and intent of the original form. So, this lead to the question "so, what can be done to successfully combine the PDFs"

Importing Pages With Conflict Resolution

PdfGeneratedDocument now has a new ImportPages feature designed to easily allow for importing pages from additional external PDFs.

PdfDocument can not combine PDFs where more than one of the documents has AcroForm elements. The new ImportPages feature of PdfGeneratedDocument includes conflict resolution events to allow for handling such cases

Quick Example:

```
rivate void MergePdfWithFormsConflicResolution(string baseFile, string importFile, string
outFile) { using (FileStream baseFileStream = new FileStream(baseFile, FileMode.Open,
FileAccess.Read, FileShare.Read)) { using (PdfGeneratedDocument genDoc = new
PdfGeneratedDocument(baseFileStream)) { using (FileStream importFileStream = new
FileStream(importFile, FileMode.Open, FileAccess.Read, FileShare.Read)) {
genDoc.ImportPages(importFileStream, new ImportOptions { //// These are additional options
available //InsertIndex = -1, //PagesToImport = new int[] {1,3,5,7}, //OwnerPassword = "",
//UserPassword = "", //RepairOptions = new Atalasoft.PdfDoc.Repair.RepairOptions(), //// This
```

```
option is the key one for merging with acroforms conflict resolution
FormFieldsConflictHandler = ResolveFormFieldsConflict }); using (FileStream outFileStm = new
FileStream(outFile, FileMode.Create)) { genDoc.Save(outFileStm); } } } } } /// /// Example
conflict resolution where we'll let the forms just bring in their fields normally, and only
rename fields if needed /// the rename here is VERY SIMPLE and may not be complex enough (we
just pre-end "new" to the field name of any conflicting field" /// You could do something
involving random numbers/letters , GUID etc.. /// /// /// private static void
ResolveFormFieldsConflict(object sender, FormFieldsConflictEventArgs args) { if
(!args.AreFieldTypesEqual) { // no conflict really so just let it ride
args.ConflictResolution = FormFieldsConflictResult.KeepCurrentFieldAndMergeChildren; } else {
// generate new name for field args.ExternalField.FieldName = "new" +
args.ExternalField.FieldName; args.ConflictResolution = FormFieldsConflictResult.KeepBoth; }
}
```

Please see attachments for a sample application that implements this solution (MergePdfFormsExample_v11.0.zip)

Original Workarounds

The following are the original workarounds provided before the conflict resolution was introduced in 11.0 for PdfGeneratedDocument. These are still valid methods but will result in stripping form content/"flattening" forms. For true merging of multiple PDFs with AcroForm objects, please consider using 11.0 or higher with the new PdfGeneratedDocument.ImportPages technique covered above

There is a way forward which involves deliberately stripping out the PDF Fillable form in such a way that the PDF is no longer interactively fillable but the filled in information is still present visually.

There are actually two practical approaches - each has slightly different licensing requirements, and results in slightly different output.

DotPdf (PdfGeneratedDocument)

If you have a license for the Atalasoft "DotPdf" product (a serial number beginning with PDFG (SDK) or PDFX2 (Server) which provides a license named Atalasoft.PdfDoc.lic)

That solution is to use our PdfGeneratedDocument class to open each PDF and strip the Form object out. This causes the actual "AcroForm" to no longer be a PDF fillable form, but leaves the "Widget Annotation" objects on their pages with their existing content.

You prepare the PDF along these lines:

```
PdfGeneratedDocument genDoc = new PdfGeneratedDocument(streamWithPdf);

genDoc.Form = null;

genDoc.Save(newStreamWithModifiedPdf);
```

You do this for each PDF you'll be combining, then you can use the standard PdfDocument.Combine(...) to combine the PDFs.

The sample solution PdfForms_DotPdfCombine which is attached to this article demonstrates first the original error from combining, then using the PdfGeneratedDocument to strip the forms and then combining the PDFs in this manner.

This specific sample was designed to require only a DotPdf license.

PdfImageSource (DotImage Document Imaging and PdfReader)

An alternative for those who do not have license for DotPdf, but do have licensing for DotImage Document Imaging and PdfReader is to simply rasterize the PDFs. This will "flatten" out any annotations present and will provide an image-based PDF of the form as it currently appears.

Our MergedImageSource can take an array of ImageSource objects (ImageSource[]) so we open each PDF that is to be merged with PdfImageSource and then pass these in the order we want them to be in the merged PDF into a MergedImageSource.

Then, we just need to pass that image source along with a stream which will be saved to into the PdfEncoder

```
pdfEncoder.Save(targetStream, mergedImageSource, null);
```

The sample application PdfForms_PdfReaderCombine implements this with proper management of streams and ImageSources (keeping track of the streams and sources so they can be properly closed/disposed when done.

Original Article

Q10428 - HOWTO: Combine Multiple PDF Files With Fillable Forms (AcroForms)

Last Update

2026-09-22 - TD - (Minor Spelling)

Atalasoft Knowledge Base
https://www.atalasoft.com/kb2/KB/50072/HOWTO-Combine-Multiple-PDF-Files-Wit...