INFO: Proper use of ForceRebuildCrossReferenceTable in RepairOptions

Repairing PDFs can be done in two ways, explicit and implicit. Explicit repair is when you use PdfDocument.Repair(...) and is used when you have a PDF you know is damaged and needs repair. Implicit repair is when you pass in RepairOptions to a PdfDocument or PdfGeneratedDocument in the constructor .. the idea is that repair will not be performed unless something goes wrong / it's needed

The general idea is that instead of attempting an operation on the PDF and using PdfDocument.Repair explicitly, you can just provide RepairOptions in the constructor and it will repair only if needed... saving you try/catch blocks and retrying manually

To use this, you would change code like this:

```
PdfDocument doc = new PdfDocument(InputFileHere);

// some operations on doc

doc.Save(OUTPUTFileOrStreamHere);

doc.Close();
```

to this:

```
PdfDocument doc = new PdfDocument(null, null, InputFileHere, new RepairOptions());

// some operations on doc

doc.Save(OUTPUTFileOrStreamHere);

doc.Close();
```

This invokes the "repair if needed" logic.. the repair will only be engaged if there is a problem opening, modifying, or saving the PDF

This works OK with the default RepairOptions. However, some options .. specifically RepairOptions.StructureOptions.ForceRebuildCrossReferenceTable change this behavior

When that option is set to true, then repair will ALWAYS run on the PDF and the entire crossreference table will be rebuilt. This is a processor, memory, and time intensive operation.. it is a "brute force" option that totally drops the "repairs only if needed" light-touch approach

INFO: Proper use of ForceRebuildCrossReferenceTable in RepairOptions

The ForceRebuildCrossReferenceTable should only ever be used on PDF that attempted repair failed

This is the recommended way to use this feature:

```
try {
    PdfDocument doc = new PdfDocument(null, null, InputFileHere, new RepairOptions());
    // some operations on doc
    doc.Save(OUTPUTFileOrStreamHere);
    doc.Close();
} catch (PdfException pdfEx) {
    // possibly log pdfEx and indicate the error in case needed for later analysis
    try {
        PdfDocument doc = new PdfDocument(null, null, InputFileHere, new RepairOptions(){
StructureOptions.ForceRebuildCrossReferenceTable = true });
        // some operations on doc
        doc.Save(OUTPUTFileOrStreamHere);
        doc.Close();
    } catch (PdfException pdfEx) {
        // something really went wrong, set aside the file and log
        // so you can investigate further
        throw pdfEx;  // or just gracefully handle etc...
    }
}
```

Conclusion

Setting StructureOptions.ForceRebuildCrossReferenceTable = true is a powerful but "brute force" instrument. When used it changes the normal "light touch" behavior of passing in RepairOptions.. it costs processing time and memory and means that every single PDF will

have its entire crossreference table rebuilt from scratch whether it needs it or not

Support recommends only using this option after normal repair/operations have failed.

Original Article:

Q10489 - INFO: Proper use of ForceRebuildCrossReferenceTable in RepairOptions

Atalasoft Knowledge Base
https://www.atalasoft.com/kb2/KB/50015/INFO-Proper-use-of-ForceRebuildCross...