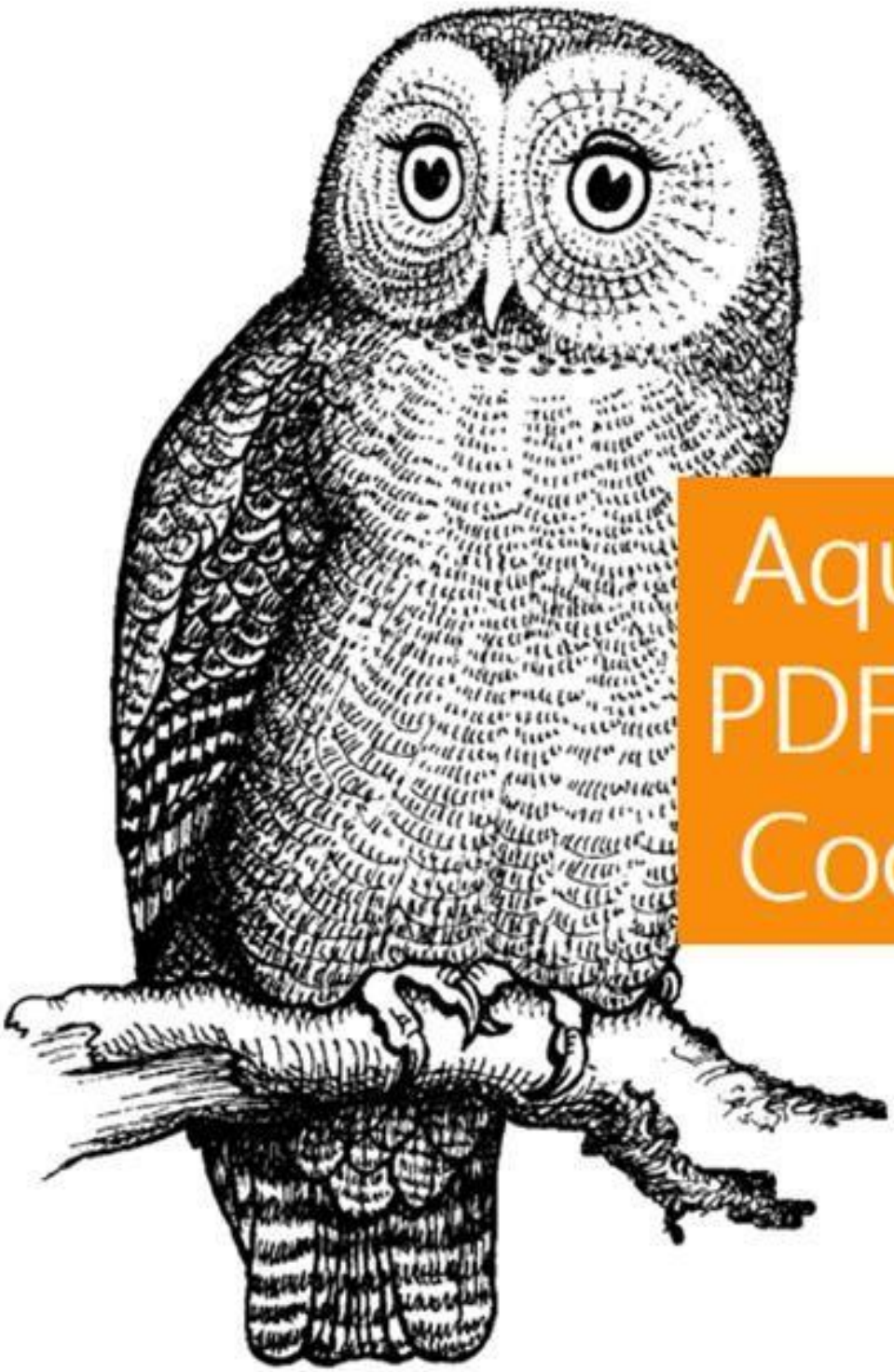
PDF Solutions for C# and Microsoft.Net Developers



Aquaforest PDF Toolkit Cookbook

Aquaforest

Aquaforest PDF Toolkit

Table of Contents

1 Extract Text from PDF

1.1 Requirement

Extract text from a PDF file.

1.2 Solution

```
using Aquaforest.PDF;
using System;
using System.Collections.Generic;
using System.Drawing;
using System.IO;
namespace ExtractTextFromPDF
{
  class ExtractTextFromPDF
  {
    static void Main(string[] args)
    {
      string inputFile = @"..\documents\source\cookbook.pdf";
      string outputFile = @"..\documents\output\hocr_ouput";

      //Get and print text from page 1
      string pageOne = GetTextFromPage(inputFile,1);
      Console.WriteLine(pageOne);
      Console.WriteLine();

      //Get and print text from whole document
      string wholeDocumet = GetTextFromPDFFile(inputFile);
      Console.WriteLine(wholeDocumet);
      Console.WriteLine();

      //Get and print text from page 2 by rect
      Rectangle rect = new Rectangle {Height=200,Width=200,X=10,Y=10 };
      string rectText = GetTextFromAreaByRect(inputFile, rect,2);
      Console.WriteLine(rectText);
      Console.WriteLine();

      //Get and print text from page 2 by coords
      string coordText = GetTextFromAreaByCordinates(inputFile,10,10,200,200, 2);
      Console.WriteLine(coordText);
      Console.WriteLine();

      //Get Document Word Data
      var worData = GetTextWithCordinatesAndFonts(inputFile);

      //Save Text in file as hocr file
      SaveTextAsHOCRFiles(inputFile,outputFile);

    }
    private static string GetTextFromPage(string doc,int pageNumber)
    {
      string text = string.Empty;
      try
      {
        PDFDocument pdfDoc = new PDFDocument(doc);
        text = pdfDoc.GetText(pageNumber);
      }
      catch(Exception)
```

```
        {

        }
        return text;
    }
    private static string GetTextFromPDFFile(string doc)
    {
        string text = string.Empty;
        try
        {
            PDFDocument pdfDoc = new PDFDocument(doc);
            text = pdfDoc.GetText();
        }
        catch (Exception)
        {

        }
        return text;
    }
    private static string GetTextFromAreaByCordinates(string doc,double x, double y, double width, double height,
int pageNumber)
    {
        string text = string.Empty;
        try
        {
            PDFDocument pdfDoc = new PDFDocument(doc);
            text = pdfDoc.GetTextByArea(x,y,width,height,pageNumber);
        }
        catch (Exception)
        {

        }
        return text;
    }
    private static string GetTextFromAreaByRect(string doc, Rectangle rect,int pageNumber)
    {
        string text = string.Empty;
        try
        {
            PDFDocument pdfDoc = new PDFDocument(doc);
            text = pdfDoc.GetTextByArea(rect, pageNumber);
        }
        catch (Exception)
        {

        }
        return text;
    }
    private static void SaveTextAsHOCRFiles(string doc,string output)
    {
        try
        {
            PDFDocument pdfDoc = new PDFDocument(doc);
            pdfDoc.GenerateHocrFromText(output,false);
        }
        catch (Exception)
        {

        }
```

```
    }
    private static List<HocrPageModel> GetTextWithCordinatesAndFonts(string doc)
    {
      try
      {
        PDFDocument pdfDoc = new PDFDocument(doc);
        return pdfDoc.GetDocumentWordData();
      }
      catch(Exception)
      {
        return null;
      }

    }
  }
}
```

1.3 Comments

The PDF Toolkit provide different ways for users to extract textual information, the cord snippet above shows 6 different ways that you can extract text and text information from a PDF file.

1. Extracts a string from a page and print it on the console.
2. Extracts text from a whole document and print it on the console.
3. Extracts text from a particular region on a PDF page using a .Net Rectangle object representing the region.
4. Extracts text from a particular region on a PDF page using the X,Y,Height,Width values of the region.
5. Extracts all the words in a document as a List<HocrPageModel>, each word is represented by a WordData object with members such as.
    - XCord: The x the beginning of the word.
    - XCord1: The X coordinate of the end of the word.
    - YCord: The Y coordinate of the top of the word.
    - YCord1: The Y coordinate of the bottom of the word.
    - FontName
    - FontSize
    - WordHieght
    - WordLength
    - Word
    - SpaceWidth
    - PageNumber
    - LineID
6. Saves the information Extracted in the above step and saves it a Hocr File.

2 Using the PDF Toolkit with Aquaforest's OCR SDK

2.1 Requirement

Make use of the PDF Toolkit in conjunction Aquaforest's OCR SDK in order to:

- Check whether a file already has text
- Add a stamp on the output file
- Split or Merge results files

2.2 Solution

```
using Aquaforest.OCR.Api;
using Aquaforest.PDF;
using System;
using System.IO;
using System.Drawing;

namespace PDFToolkitWithOCRSDK
{
   class Program
   {
      // NOTE: In order for this example to work you will need to download the OCR SDK
(http://www.aquaforest.com/en/ocrsdk.asp) first and then reference Aquaforest.OCR.Api.

      static void Main(string[] args)
      {
         DirectoryInfo directory = new DirectoryInfo(@"..\..\..\documents\source\tree");

         foreach (var file in directory.GetFiles("*.pdf", SearchOption.AllDirectories))
         {
            Console.WriteLine("Processing PDF file: {0}", file.FullName);
            PDFDocument doc = new PDFDocument(file.FullName);
            string docText = doc.GetText().Replace("\r\n", "").Replace("\n", "").Replace("\r", "");
            int numberOfPages = doc.NumberOfPages;
            doc.Close();

            // OCR document only if it is image only
            if (docText.Length == 0)
            {
               Console.WriteLine("PDF file is image-only");

               string outputDirectory = Path.Combine(@"..\..\..\documents\output", file.Directory.Name);
               if (!Directory.Exists(outputDirectory))
                  Directory.CreateDirectory(outputDirectory);

               string output = Path.Combine(outputDirectory, file.Name);

               bool ocrSuccessful = OCRPDF(file.FullName, output);

               if (ocrSuccessful)
               {
                  AddStamp(output);

                  if (numberOfPages > 1)
                  {
                     SplitFile(output);
                  }
               }
```

```
            }
            else
            {
                Console.WriteLine("PDF file contains text");
            }

            Console.WriteLine("");
        }
    }

    static bool OCRPDF(string source, string output)
    {
        bool success = false;

        try
        {
            Ocr ocr = new Ocr();
            PreProcessor = new PreProcessor();

            string OCRFiles = Path.GetFullPath(@"..\..\lib\");
            Environment.SetEnvironmentVariable("PATH", Environment.GetEnvironmentVariable("PATH") + ";" +
OCRFiles);
            ocr.ResourceFolder = OCRFiles;
            ocr.EnableConsoleOutput = true;
            ocr.Language = SupportedLanguages.English;
            ocr.EnablePdfOutput = true;
            ocr.ReadPDFSource(source);

            preProcessor.Deskew = true;
            preProcessor.Autorotate = false;

            if (ocr.Recognize(preProcessor))
            {
                ocr.SavePDFOutput(output, true);
            }
            ocr.DeleteTemporaryFiles();

            success = true;
        }
        catch (Exception e)
        {
            Console.WriteLine("Error in OCR Processing :" + e.Message);
            success = false;
        }

        return success;
    }

    static void AddStamp(string output)
    {
        Console.WriteLine("Adding stamp...");

        PDFDocument doc = new PDFDocument(output);

        PDFStamper stamper = new PDFStamper(doc, doc.FilePath);
        stamper.FontSize = 12;
        stamper.StampOpacity = 100;
        stamper.StampColor = Color.Black;
        stamper.StampPDFText("THIS IS A TEST STAMP", 200, 200);
```

```
    }

    static void SplitFile(string output)
    {
      Console.WriteLine("Splitting file...");

      PDFDocument doc = new PDFDocument(output);

      PDFSplitter splitter = new PDFSplitter(doc);
      splitter.OutputFileName = Path.GetFileNameWithoutExtension(output);
      splitter.OutputFilePath = Path.GetDirectoryName(output);
      splitter.SplitByRepeatingNumber(1, 5, 1);
    }
  }
}
```

2.3 Comments

The above code will loop through all the PDF documents in a directory and check if they either contain text or are image-only. The image-only PDFs are OCRed and stamped. If the OCRed documents have more than one page, they are split into single pages.

In order for this example to work you will need to download the OCR SDK (http://www.aquaforest.com/en/ocrsdk.asp) first and then reference Aquaforest.OCR.Api.

3 Split PDF

3.1 Requirement

Split PDF files.

3.2 Solution

```
using Aquaforest.PDF;

namespace SplitPDF
{
  class Program
  {
    static void Main(string[] args)
    {
      PDFDocument doc = new
PDFDocument(@"..\..\..\documents\source\cookbook.pdf");
      PDFSplitter splitter = new PDFSplitter(doc);        splitter.OutputFileName = "split";
      splitter.OutputFilePath = @"..\..\..\documents\output\";

      splitter.SplitByRepeatingNumber(1, 5, 1);
    }
  }
}
```

3.3 Comments

The PDFSplitter class provides different methods to split PDF files namely:

- SplitByRepeatingNumber
  Split PDF document with each split containing a particular number of pages

- SplitByTopLevelBookmarks
  Split the document by the top level bookmarks in the PDF.

- SplitByPageRanges
  Split PDF document by page ranges. E.g. 1, 3-7, 10

- SplitByRepeatingPageRanges
  Apply the page range to each set of "Page Ranges" pages within the document. For example if 2-4 is specified for page ranges, and 4 is specified as the repeating range, then the range is re-applied every 4 pages. Hence the file is split such that the first output file contains pages 2-4 from the original document; the second contains pages 6-8 and so on.

4 Merge PDF

4.1 Requirement

Merge PDF files.

4.2 Solution

```
using Aquaforest.PDF;
using System.Collections.Generic;

namespace MergePDFs
{
   class Program
   {
      static void Main(string[] args)
      {
         PDFMerger merger = new PDFMerger();

         List<string> pdfList = new List<string>()
         {
            @"..\..\..\documents\source\image_pdf.pdf",
            @"..\..\..\documents\source\releasenotes.pdf",
            @"..\..\..\documents\source\pdf_with_images.pdf"
         };
          merger.MergePDFs(pdfList,
@"..\..\..\documents\output\merge.pdf");
      }
   }
}
```

4.3 Comments

The PDFMerger class provides 3 different methods of merging PDF files:

- Select and add individual PDF files to merge to a List

- Merge all PDF files in a particular folder
- Merge each folder in a folder tree separately

5 Create a PDF document

5.1 Requirement

Create a PDF document from scratch and add some text and metadata in it.

5.2 Solution

```
using Aquaforest.PDF; using
Aquaforest.PDF.Font; using System;

namespace CreatePDF
{
   class Program
   {
      // Create a PDF document       static void
Main(string[] args)
      {
         PDFDocument doc = new PDFDocument();
         PDFPage page = new PDFPage();

         // Add text to page
         PDFPageContentStream contents = new PDFPageContentStream(doc, page);
         contents.BeginText();
         contents.SetFont(PDFType1Font.COURIER_BOLD, 12);
contents.MoveText(100, 700);          contents.DrawString("Hello World!");
contents.EndText();          contents.Close();

         // Add metadata
         PDFDocumentInformation info = new PDFDocumentInformation()          {
            Author = "Name Surname",
            Subject = "Test",
            Title = "New PDF",
            Keywords = "PDF, OCR, SDK",
            CreationDate = new DateTime(2013, 9, 9),
            Producer = "Aquaforest"
         };
         info.SetCustomMetadataValue("AQUAFOREST_PDF_TOOLKIT", "1.01");
doc.SetDocumentInformation(info);

         // Add page to PDF document          doc.AddPage(page);
          doc.Save(@"..\..\..\documents\output\pdf_out.pdf");          doc.Close();
      }
   }
}
```

6 Apply Stamps to PDF Files

6.1 Requirement

Apply various types of stamp / watermark to PDF files.

6.2 Solution

```
using Aquaforest.PDF; using
System.Drawing; namespace StampPDF
{
   class Program
   {
      static void Main(string[] args)
      {
         PDFDocument doc = new
PDFDocument(@"..\..\..\documents\source\releasenotes.pdf");

         PDFStamper stamper = new PDFStamper(doc,
@"..\..\..\documents\output\stamp.pdf");          stamper.FontSize = 12;
stamper.StampOpacity = 100;          stamper.StampColor = Color.Black;
         stamper.StampPDFText("This is a test stamp", 200, 200);
      }
   }
}
```

6.3 Comments

The PDFStamper provides different types of stamping methods namely:

- StampPDFTextAsImage - Stamp in the output PDF will be in image format
- StampTextAsString - Stamp in the output PDF will be in text format
- StampPageNumber - Add a number to each page starting from a particular number and incrementing by 1 for each page
- StampPageNumberBates - Stamp each page with a bate number
- StampVariable – Stamp each page with one of the stamp variables provided in the PDF Toolkit.

The table below shows all the acceptable variables:

Variable	**Stamp**
%a	Short Day (Mon)
%A	Long Day (Monday)
%b	Short Month (Jan)
%B	Long Month (January)
%c	Date and time (30 October 2013 17:21)
%C	Date and Time With seconds (30 October 2013 17:21:50)
%d	Month and Year (October 2013)
%D	Day and Month (30 October)

%e	Short Year (13)
%E	Long Year (2013)
%f	Short Time of Day (17:21)
%F	Time of Day With Seconds (17:21:20)
%G	Full Date and time (Wed, 30 October 2013 17:21:50 GMT)
%Y	File Name

7 Convert an Image file to PDF

7.1 Requirement

Convert an image file (BMP, PNG, JPEG, TIFF…) to PDF.

7.2 Solution

```
using Aquaforest.PDF;

namespace ImageToPDF
{
   class Program    {
      static void Main(string[] args)
      {
         ImageToPDFConverter imageConverter = new
ImageToPDFConverter(@"..\..\..\documents\source\image.tif",
@"..\..\..\documents\output\output.pdf");          imageConverter.ConvertImageToPDF();
      }
   }
}
```

8 Extract Images from PDF

8.1 Requirement

Extract all images from a PDF document.

8.2 Solution

```
using Aquaforest.PDF;

namespace ExtractImagesFromPDF
{
   class Program
   {
      static void Main(string[] args)
      {
         PDFDocument doc = new
PDFDocument(@"..\..\..\documents\source\pdf_with_images.pdf");          PDFImageExtractor extractor = new
PDFImageExtractor(doc);          extractor.ExtractImages(@"..\..\..\documents\output", "img", false);       }
   }
}
```

8.3 Comments

Note that the images extracted from the PDF file will be saved in the same format that they were added to the PDF.

9 Overlay a PDF

9.1 Requirement

Add a PDF page as an overlay to another PDF.

9.2 Solution

```
using Aquaforest.PDF;

namespace AddOverlay
{
   class Program
   {
      static void Main(string[] args)
      {
         PDFDocument overlay = new
PDFDocument(@"..\..\..\documents\source\overlay.pdf");
         PDFDocument overlayDestination = new
PDFDocument(@"..\..\..\documents\source\image_pdf.pdf");
         PDFOverlay o = new PDFOverlay(overlay, overlayDestination);

o.ApplyOverlay(@"..\..\..\documents\output\overlay_output.pdf");
      }
   }
}
```

9.3 Comments

Note that the page being added as an overlay should not be an image PDF or else it will cover the original page in the destination PDF completely.

10 Set PDF Security

10.1 Requirement

Set security and other access permissions to a PDF document.

10.2 Solution

```
using Aquaforest.PDF;

namespace SetPDFPermissions
{
   class Program    {
      static void Main(string[] args)
      {
         PDFDocument doc = new
PDFDocument(@"..\..\..\documents\source\releasenotes.pdf");

         PDFPermission permission = new PDFPermission()
         {
            AllowExtractContents = false,
            AllowModifyContents = false,
            AllowAssembly = true,
            AllowDegradedPrinting = true,
            AllowExtractForAccessibility = true,
            AllowFillInForm = false,
            AllowModifyAnnotations = false,
            AllowPrinting = true
         };

         PDFSecurity encryptor = new PDFSecurity()
         {
            Permission = permission,
            OwnerPassword = "myOwnerPassword",
            UserPassword = "myUserPassword"
         };
          encryptor.SecurePDF(doc,
@"..\..\..\documents\output\secure_pdf.pdf");
      }
   }
}
```

11 Get and Set XMP metadata

11.1 Requirement

Get XMP metadata from a PDF file and set XMP metadata to a PDF file.

11.2 Solution

```
using Aquaforest.PDF; using
System; using System.Xml;

namespace SetAndGetXMPMetadata
{
   class Program
   {
      static PDFDocument doc;

      static void Main(string[] args)
      {
         SetXMP();

         GetXMP();
      }

      static void SetXMP()
      {
         XmlDocument xmp = new XmlDocument();
         xmp.Load(@"..\..\..\documents\source\xmp.xml");

         doc = new
PDFDocument(@"..\..\..\documents\source\releasenotes.pdf");          doc.SetXMPMetadata(xmp);
         doc.Save(@"..\..\..\documents\output\releasenotes_xmp.pdf");       }

      static void GetXMP()
      {
         doc = new
PDFDocument(@"..\..\..\documents\source\pdf_with_images.pdf");          string xmp =
doc.GetXMPMetadata();
         Console.WriteLine(xmp);
      }
   }
}
```

12 Get and Set PDF Metadata

12.1 Requirement

Get metadata from a PDF file and set metadata to a PDF file.

12.2 Solution

```
using Aquaforest.PDF;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace SetAndGetPDFMetadata
{
   class SetGetMetadata
   {
      static void Main(string[] args)
      {
         string inputFile = @"..\documents\source\releasenotes.pdf";
         string outputFile = @"..\documents\output\metadata.pdf";
         PDFDocumentInformation info = new PDFDocumentInformation()
         {
            Author = "Name Surname",
            Subject = "Test",
            Title = "New PDF",
            Keywords = "PDF, OCR, SDK",
            CreationDate = new DateTime(2013, 9, 9),
            Producer = "Aquaforest"
         };
         info.SetCustomMetadataValue("AQUAFOREST_PDF_TOOLKIT", "1.01");
         SetPDFMetadata(inputFile,outputFile,info);
         ReadDocumentInformation(outputFile);
      }
      public static void ReadDocumentInformation(string input)
      {
         try
         {
            PDFDocument doc = new PDFDocument(input);
            PDFDocumentInformation docInfo = doc.GetDocumentInformation();
            Console.WriteLine("Author = {0}, CreationDate = {1}, Creator = {2} .", docInfo.Author,
docInfo.CreationDate, docInfo.Creator);
            doc.Close();
         }
         catch(Exception)
         {     }
      }
      public static void SetPDFMetadata(string input,string output,PDFDocumentInformation metadata)
      {
         try
         {
            PDFDocument doc = new PDFDocument(input);
            doc.SetDocumentInformation(metadata);
            doc.Save(output);
            doc.Close();
         }
         catch (Exception)
         {         }
      }
   }
}
```

13 Read and Set Viewer Preferences

13.1 Requirement

Read the PDF viewer preferences of PDF file and set the PDF viewer preferences of a PDF file.

13.2 Solution

```
using Aquaforest.PDF;
using System;

namespace SetAndGetPDFOpensettings
{
   internal class Opensettings
   {
      private static void Main(string[] args)
      {
         string inputFile = @"..\documents\source\releasenotes.pdf";
         string outputFile = @"..\documents\output\setoptions.pdf";
         SetOpenOptions(inputFile, outputFile);
         ReadOpenOptions(outputFile);
      }

      public static void SetOpenOptions(string input, string output)
      {
         try
         {
            PDFDocument doc = new PDFDocument(input);
            //Create the opensettings object
            PDFDocumentOpenSetting opensetting = new PDFDocumentOpenSetting(doc);
            opensetting.PDFPageMode = PageModeOptions.FULL_SCREEN;
            opensetting.NonFullScreenPageMode = NonFullScreenModeOptions.UseThumbs;
            opensetting.HideWindowUI = true;
            opensetting.CenterWindow = true;
            opensetting.PDFPageLayout = PageLayoutOptions.ONE_COLUMN;
            opensetting.HideToolbar = true;
            opensetting.HideMenubar = true;
            opensetting.SaveOpenSettings(output);
            doc.Close();
         }
         catch (Exception)
         {
         }
      }

      public static void ReadOpenOptions(string input)
      {
         PDFDocument doc = new PDFDocument(input);
         PDFDocumentOpenSetting opensetting = new PDFDocumentOpenSetting(doc);
         Console.WriteLine("PageLayout = {0}, PageMode = {1}, HideWindowUI = {2} .",
opensetting.PDFPageLayout, opensetting.PDFPageMode, opensetting.HideWindowUI);
         doc.Close();
      }
   }
}
```

14 Convert a CSV file to PDF

14.1 Requirement

Take a Comma Separated Value (CSV) file and convert it to a table in a PDF file.

14.2 Solution

```
using Aquaforest.PDF; using
Aquaforest.PDF.Font;

namespace CSVToPDF
{
   class Program
   {
      static void Main(string[] args)
      {
         CSVToPDFConverter csv = new
CSVToPDFConverter(@"..\..\..\documents\source\csv_input.csv");

         // The page settings of the output PDF
         PDFPageSettings pageSettings = new PDFPageSettings();          pageSettings.Font =
PDFType1Font.HELVETICA;          pageSettings.FontSize = 12;
         pageSettings.SetMargin(50, 50, 50, 50);          pageSettings.SetSize(PageSize.A4);

         // The settings of the table in the output PDF
         PDFTableSettings tableSettings = new PDFTableSettings()
         {
            CellMargin = 5,
            RowHeight = 20
         };

         csv.ConvertToPDF(@"..\..\..\documents\output\csv_output.pdf", pageSettings, tableSettings);
      }
   }
}
```

15 Add Bookmarks to PDF File

15.1 Requirement

Add and retrieve bookmarks from a PDF document

15.2 Solution

```
using Aquaforest.PDF;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;


namespace PDFBookmarks
{
   class PDFBookmarks
   {
      static void Main(string[] args)
      {
         string input = @"..\documents\source\cookbook.pdf";
         string output = @"..\documents\output\bookmark.pdf";
         //Insert bookmark
         PDFBookmarkItem bookmark1 = new PDFBookmarkItem(1, "Page  1");
         PDFBookmarkItem bookmark2 = new PDFBookmarkItem(4, "Page 2");
         PDFBookmarkItem bookmark3 = new PDFBookmarkItem(14, "Page 14");
         PDFBookmarkItem bookmark4 = new PDFBookmarkItem(4, "Page 2");
         PDFBookmarkItem bookmark5 = new PDFBookmarkItem(14, "Page 14 second");
         PDFBookmark bookmark = new PDFBookmark();
         bookmark.BookmarkHeader = "Exampl";
         bookmark1.BookmarkItems.Add(bookmark4);
         bookmark1.BookmarkItems.Add(bookmark5);
         bookmark2.BookmarkItems.Add(bookmark4);
         bookmark2.BookmarkItems.Add(bookmark5);
         bookmark.BookmarkItems.Add(bookmark1);
         bookmark.BookmarkItems.Add(bookmark2);
         bookmark.BookmarkItems.Add(bookmark3);
         AddBookmarks( input, output,bookmark);

         //retrive bookmarks
         var bookmarks = GetBookmarks(input);
         if(bookmarks.BookmarkItems.Count>0)
         {
            foreach(var book in bookmarks.BookmarkItems)
            {
               Console.WriteLine("Tile = "+book.BookmarkTitle+" , page number = "+book.BookmarkTitle);
               if(book.BookmarkItems.Count>0)
               {
                  // you can do some thing recursive if three is a bookmark tree
               }
            }
         }
      }




   public static void AddBookmarks(string input,string output,PDFBookmark bookmark)
```

```
    {
      try
      {
        PDFDocument document = new PDFDocument(input);

        document.AddBookmarks(bookmark);
        document.Save(output);
      }
      catch(Exception)
      {

      }
    }
    public static PDFBookmark  GetBookmarks(string input)
    {
      PDFDocument document = new PDFDocument(input);
      return document.GetBookmarks();
    }
  }
}
```

16 Add Annotations to PDF

16.1 Requirement

Add annotations to PDF pages

16.2 Solution

```
using Aquaforest.PDF;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;


namespace PDFAnnotations
{
  class PDFAnnotations
  {
    static void Main(string[] args)
    {
      string inputFile = @"..\documents\source\image_pdf.pdf";
      string outputFile = @"..\documents\output\pdfannotations.pdf";
      PDFRectangle rect = new PDFRectangle(10, 10, 300, 50);
      PDFAnnotation annotation = new
PDFAnnotation(PDFAnnotationType.PDFAnnotationRubberStamp,rect,"PDF
Tookit",PDFRubberStampImages.NotForPublicRelease);
      PDFAnnotation annotation1 = new PDFAnnotation(PDFAnnotationType.PDFAnnotationRubberStamp, rect,
"PDF Tookit",PDFRubberStampImages.NotApproved);
      PDFAnnotation annotation3 = new PDFAnnotation(PDFAnnotationType.PDFAnnotationTextMarkup, rect,
"PDF Tookit");
      PDFAnnotation annotation5 = new PDFAnnotation(PDFAnnotationType.PDFAnnotationLink, rect, "PDF
Tookit");
      PDFAnnotation annotation6 = new PDFAnnotation(PDFAnnotationType.PDFAnnotationLine, rect, "PDF
Tookit", PDFAnnotationLineSubtypes.SQUARE);
      Dictionary<int, PDFAnnotation> annotaions = new Dictionary<int, PDFAnnotation>();
```

```
        annotation5.LinkURL = "https://www.aquaforest.com/en/pdftoolkit.asp";
        annotation3.AnnotationColor = PDFColor.Yellow;
        annotaions.Add(1, annotation);
        annotaions.Add(2, annotation1);
        annotaions.Add(3, annotation3);
        annotaions.Add(5, annotation5);
        annotaions.Add(6, annotation6);
        AddAnotations(inputFile,outputFile, annotaions);
    }
    public static void AddAnotations(string file,string output,Dictionary<int,PDFAnnotation> annotaions )
    {
      try
      {
        PDFDocument pdfDoc = new PDFDocument(file);
        foreach(var key in annotaions.Keys)
        {
          pdfDoc.GetPage(key).AddAnnotation(annotaions[key], pdfDoc);
        }

        pdfDoc.Save(output);
        pdfDoc.Close();
      }
      catch(Exception)
      { }

    }
  }
}
```

17 Add/Extract File Attachments from PDF document

17.1 Requirement

Add attachments to a PDF document, extract files attached to a PDF document to a folder.

17.2 Solution

```
using Aquaforest.PDF;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace PDFAttachments
{
  class PDFAttachments
  {
    static void Main(string[] args)
    {
      string inputFile = @"..\documents\source\image_pdf.pdf";
      string outputFolder = @"..\documents\output\extract\\";
      string outputFile = @"..\documents\output\attachment.pdf";
      PDFAttachmentItem item = new PDFAttachmentItem(@"..\documents\source\cookbook.pdf");
      PDFAttachmentItem item1 = new PDFAttachmentItem(@"..\documents\source\csv_input.csv");
      PDFAttachmentItem item2 = new PDFAttachmentItem(@"..\documents\source\stampFile.png");
      PDFAttachmentItem item3 = new PDFAttachmentItem(@"..\documents\source\xmp.xml");
      AddPDFAttchment(item,inputFile, outputFile);
      List<PDFAttachmentItem> allItems = new List<PDFAttachmentItem> {item,item1,item2,item3 };
      AddPDFAttchment(allItems, inputFile, outputFile);
```

```
            GetPDFAttchment(outputFolder, outputFile);
        }
        public static void AddPDFAttchment(List<PDFAttachmentItem> attachments,string inputFile,string ouptut)
        {
            PDFDocument pdfDoc = new PDFDocument(inputFile);
            pdfDoc.EmbedPDFAttachments(attachments);
            pdfDoc.Save(ouptut);
        }
        public static void AddPDFAttchment(PDFAttachmentItem attachments, string inputFile, string ouptut)
        {
            PDFDocument pdfDoc = new PDFDocument(inputFile);
            pdfDoc.EmbedPDFAttachment(attachments);
            pdfDoc.Save(ouptut);
        }
        public static void GetPDFAttchment(string outputFolder,string inputfile)
        {
            PDFDocument pdfDoc = new PDFDocument(inputfile);
            pdfDoc.ExtractAttachment(outputFolder);
        }
    }
}
```

18 Convert Office Files to PDF

18.1 Requirement

Convert word, Excel, PowerPoint and publisher documents to PDF

18.2 Solution

```
using Aquaforest.PDF;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;


namespace OfficeToPDFConversion
{
    class OfficeConversion
    {
        static void Main(string[] args)
        {
            string wordInput = @"..\documents\source\Skills.docx";
            string excelInput = @"..\documents\source\excel.xlsx";
            string powerPointInput = @"..\documents\source\split.pptx";
            string pubInput = @"..\documents\source\publisher.pub";
            string wordOutput = @"..\documents\output\word.pdf";
            string excelOutput = @"..\documents\output\excel.pdf";
            string powerPointOutput = @"..\documents\output\powerpoint.pdf";
            string pubOutput = @"..\documents\output\pub.pdf";
            DocumentSettings docSettings = new DocumentSettings
{BookmarkOption=AquaforestPDFBookmarks.BOOKMARKS_FROM_HEADINGS,PDFA1a= true};
            ConvertWordToPDF(wordInput,wordOutput,docSettings);
```

```
        ConvertExcelToPDF(excelInput, excelOutput, docSettings);
        ConvertPowerPointToPDF(powerPointInput, powerPointOutput, docSettings);
        ConvertPublisherToPDF(pubInput, pubOutput, docSettings);
    }
    public static void ConvertWordToPDF(string input,string output,DocumentSettings settings)
    {
        try
        {
            WordToPDF wordToPDF = new WordToPDF(Path.GetFullPath(input));
            wordToPDF.ConvertToPDF(settings, Path.GetFullPath(output));
        }
        catch(Exception)
        {

        }

    }
    public static void ConvertExcelToPDF(string input, string output, DocumentSettings settings)
    {
        try
        {
            ExcelToPDF wordToPDF = new ExcelToPDF(Path.GetFullPath(input));
            wordToPDF.ConvertToPDF(settings, Path.GetFullPath(output));
        }
        catch (Exception)
        {

        }
    }
    public static void ConvertPowerPointToPDF(string input, string output, DocumentSettings settings)
    {
        try
        {
            PowerPointToPDF wordToPDF = new PowerPointToPDF(Path.GetFullPath(input));
            settings.OptimizeFor = 0;
            wordToPDF.DocStructureTags =true;
            wordToPDF.FrameSlides = true;

            wordToPDF.PowerPointHandoutOrder = 0;
            wordToPDF.ConvertToPDF(settings, Path.GetFullPath(output));
        }
        catch (Exception)
        {

        }
    }
    public static void ConvertPublisherToPDF(string input, string output, DocumentSettings settings)
    {
        try
        {
            PublisherToPDF wordToPDF = new PublisherToPDF(Path.GetFullPath(input));
            wordToPDF.ConvertToPDF(settings, Path.GetFullPath(output));
        }
        catch (Exception)
        {

        }
    }
```

```
    }
}
```

18.3 Comment

You will need Microsoft Office Installed to use this feature.

19 Convert and Validate PDF/A files

19.1 Requirement

Convert PDF files to PDF/A Complaint files, validate files that claim to be PDF/A complaint.

19.2 Solution

```
using Aquaforest.PDF;
using System;

namespace PDFAConversionAndValidation
{
   internal class PdfaOperations
   {
      private static void Main(string[] args)
      {
         string inputFile = @"..\documents\source\image_pdf.pdf";
         string outputFile = @"..\documents\output\pdfa.pdf";

        //Converting PDF File to PDF
         ConvertToPDFA(inputFile,outputFile,AquaforestPDFAFlavour.PDFA_1_B);

         //Validate PDFa File
         PDFAValidationResult results = ValidatePDFA(AquaforestPDFAFlavour.PDFA_1_B, outputFile);

         //check the result
         if(results.IsValid)
         {
            //Success
            Console.WriteLine("Validated!");
         }
         else
         {
            //Not validated, print out validation errors
            foreach(var error in results.ValidationResult)
            {
               Console.WriteLine(error);
            }
         }
      }

      public static void ConvertToPDFA(string input, string output, AquaforestPDFAFlavour flavour)
      {
         try
         {
            //Create PDFA object
            PDF2PDFaConverter pdfaConverter = new PDF2PDFaConverter(input, output, flavour);
            //Convert File to PDFa
```

```
            pdfaConverter.ConvertToPDFA();
        }
        catch (Exception e)
        {
        }
    }

    public static PDFAValidationResult ValidatePDFA(AquaforestPDFAFlavour flavour, string file)
    {
        try
        {
            //Create PDFAValidator object
            PDFAValidator pdfaValidator = new PDFAValidator();
            // Validate file base on flavour
            return pdfaValidator.ValidatePDFA(file, flavour);
        }
        catch (Exception)
        {
            return null;
        }
    }
  }
}
```

19.3 Comment

Make sure the input PDF has all the fonts embedded before attempting the PDF/A conversion, we can guarantee conversion of image only PDFs too.

20 Validate PDF file

20.1 Requirements

Check to see if your PDF files are valid and can be processed by most of our software.

20.2 Solution

```
using Aquaforest.PDF;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;


namespace PDFValidate
{
  class ValidatePDF
  {
    static void Main(string[] args)
    {
      var files = Directory.GetFiles(@"C:\dev\Source Control\pdf-toolkit\documents\source\ValidatorTest");
      foreach (var file in files)
      {
        PDFValidator validate = new PDFValidator(file);
        if (validate.IsPasswordProtected)
        {
```

```
                Console.WriteLine("{0} is Password Protected", file);
            }
            else
            {
                Console.WriteLine("{0} is not Password Protected", file);
            }
            if (validate.IsValid)
            {
                Console.WriteLine("{0} is Valid", file);
            }
            else
            {
                Console.WriteLine("{0} is In Valid, {1}", file,validate.ErrorMessage);
            }
        }
    }
  }
}
```

20.3 Comment

This does not check for all PDF errors; it just checks for the error that will make the PDF file fail to process in most software.