How to find and highlight links in PDF documents

Written by Apitron Documentation Team

Introduction

If you create a custom viewer for PDF documents, then you could try Apitron PDF Rasterizer library that has unique features designed specifically to meet the needs of custom viewer writers. In addition to its core rendering functionality, it offers text search and navigation API which help you to create a perfect viewer.

We won't be focusing on search this time, because it was well described in several articles (see this or this). Instead, we'll show how to find and highlight existing links that are often used to implement quick navigation between chapters of redirect users to external web resource.

Links in PDF are defined using special objects called LinkAnnotations. These objects can point to direct destination within the same document or have an action assigned. There are lots of actions defined in specification e.g. GoToAction, RemoteGoToAction, JavaScriptAction etc. Apitron PDF Rasterizer provides support only for navigation-specific actions and if you need support for all annotation and action types then you could take a look at our Apitron PDF Kit product. You can imagine the link annotation object as a "hot" area on PDF page with predefined target or action. It doesn't contain any text and usually becomes positioned over the text block providing a hint for where it redirects.

In addition, Adobe reader often parses document text and detects external links to web resources. It then allows you to click on these links, as there were link annotation objects created while actually it's just a plain text. So you could also use text search with regular expressions to find the links on page as a second option.

In this article we'll show how to use Apitron PDF Rasterizer and its API to convert PDF page to image and highlight all links on this page.

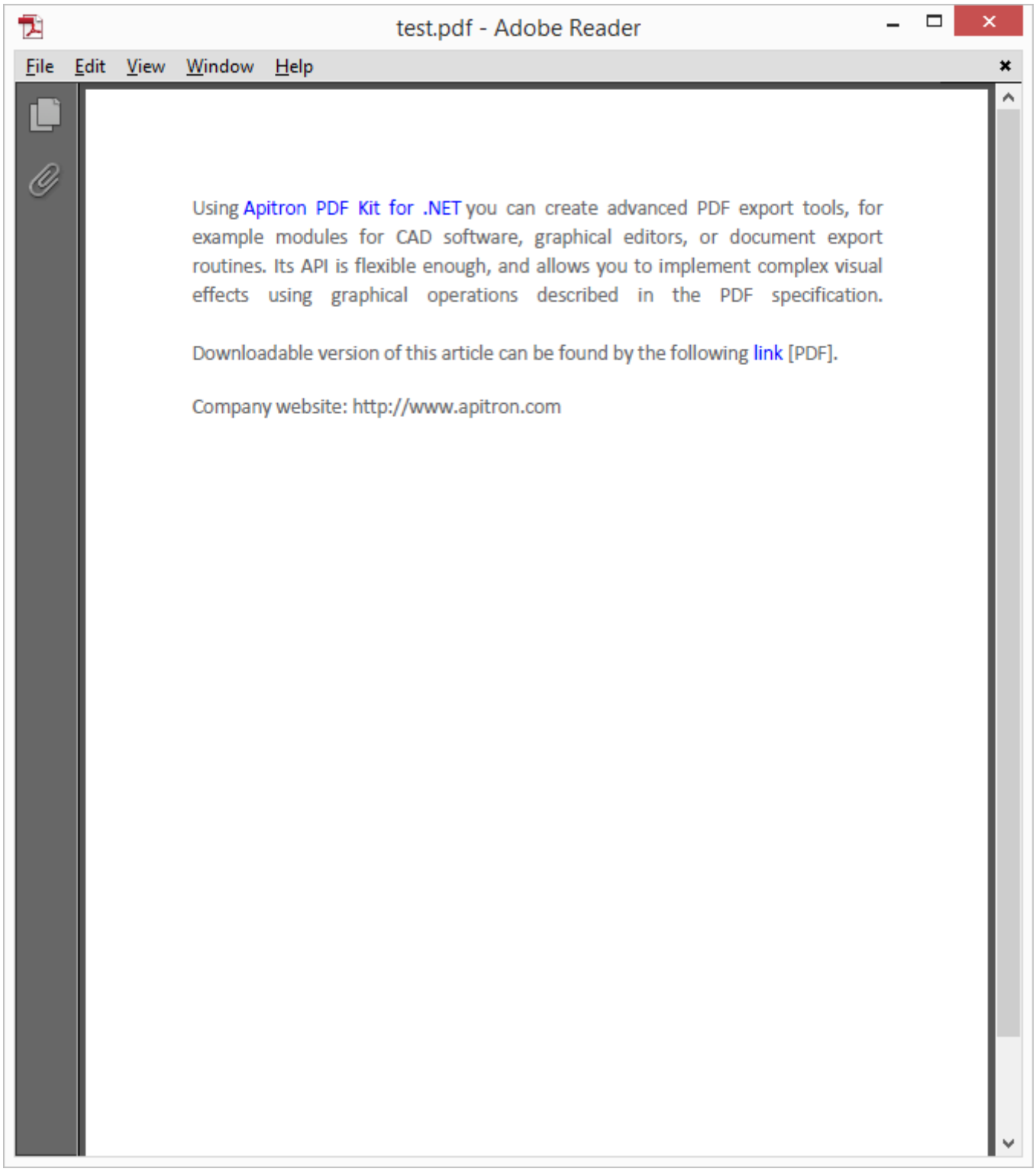
The code

```
class Program
{
    static void Main(string[] args)
    {
        // prepare graphics objects
        Bitmap renderedPage = null;
        Brush highlightBrush = new SolidBrush(Color.FromArgb(126, 255, 255, 0));

        //  store rendering settings
        Resolution renderingResolution = new Resolution(144, 144);
        RenderingSettings renderingSettings = new RenderingSettings();
        Page firstPage = null;

        // a list of rects to highlight
        IList<RectangleF> highlightRects = new List<RectangleF>();

        // open PDF document
        using (FileStream fs = new FileStream("../../files/test.pdf",
              FileMode.Open, FileAccess.Read,
              FileShare.ReadWrite))
        {
            using (Document doc = new Document(fs))
            {
                firstPage = doc.Pages[0];

                renderedPage = firstPage.Render(renderingResolution, renderingSettings);

                // parse links and store highlight rects
                foreach (Link link in firstPage.Links)
                {
                    if (link.IsUriLink)
                    {
                        Apitron.PDF.Rasterizer.Rectangle locationRect =
                            link.GetLocationRectangle(renderingResolution, renderingSettings);

                        highlightRects.Add(TransformToGDIRect(locationRect, renderedPage.Height));

                        Console.WriteLine(link.DestinationUri);
                    }
                }
            }
        }

        // search text in the same document using regular expression matching URLs
        using (SearchIndex search = new SearchIndex(new FileStream("../../files/test.pdf",
               FileMode.Open, FileAccess.Read,
               FileShare.ReadWrite)))
        {
            search.Search(handlerArgs =>
            {
                // first page only
                if (handlerArgs.PageIndex > 0)
                {
                    handlerArgs.CancelSearch = true;
                    return;
                }

                // add highlight rects by processing found items
                foreach (SearchResultItem item in handlerArgs.ResultItems)
                {
                    SearchResultRegion searchResultRegion = firstPage.TransformRegion(item.Region,
                      renderingResolution, renderingSettings);
```

```
                    foreach (double[] block in searchResultRegion.Blocks)
                    {
                        float xMin = float.MaxValue;
                        float yMin = float.MaxValue;
                        float xMax = float.MinValue;
                        float yMax = float.MinValue;
                        for (int i = 0; i < block.Length;)
                        {
                            xMin = (float)Math.Min(xMin, block[i]);
                            xMax = (float)Math.Max(xMax, block[i++]);

                            yMin = (float)Math.Min(yMin, block[i]);
                            yMax = (float)Math.Max(yMax, block[i++]);
                        }

                        highlightRects.Add(new RectangleF(xMin, yMin, xMax-xMin, yMax-yMin));
                    }

                    Console.WriteLine(item.Title);
                }
            },
            new Regex(@"(http|ftp|https):\/\/[\w\-_]+(\.[\w\-_]+)+([\w\-\.,@?^=%&amp;:/~\+#]*[\w\-
                \@?^=%&amp;/~\+#])?"));
        }

        // render hightllight rects
        HighlightRects(renderedPage, highlightRects, highlightBrush);

        renderedPage.Save("renderedPage.png");
        Process.Start("renderedPage.png");

        Console.WriteLine("Press any key to continue...");
        Console.ReadLine();
    }

    /// <summary>
    /// Highlights a list of rects.
    /// </summary>
    private static void HighlightRects(Bitmap renderedPage, IList<RectangleF> highlightRects,
       Brush highlightBrush)
    {
        using (Graphics g = Graphics.FromImage(renderedPage))
        {
            foreach (RectangleF rect in highlightRects)
            {
                g.FillRectangle(highlightBrush, rect);
            }
        }
    }

    /// <summary>
    /// Transforms PDF rect to GDI rect.
    /// </summary>
    /// <param name="locationRect">Rect to transform.</param>
    /// <param name="height">The height of the page.</param>
    /// <returns>Transformed GDI rect.</returns>
    private static RectangleF TransformToGDIRect(Rectangle locationRect, double height)
    {
        return new RectangleF((float)locationRect.Left, (float)(height - locationRect.Top),
            (float)locationRect.Width, (float)locationRect.Height);
    }
}
```

The complete code sample can be found in our [github](#) repository.

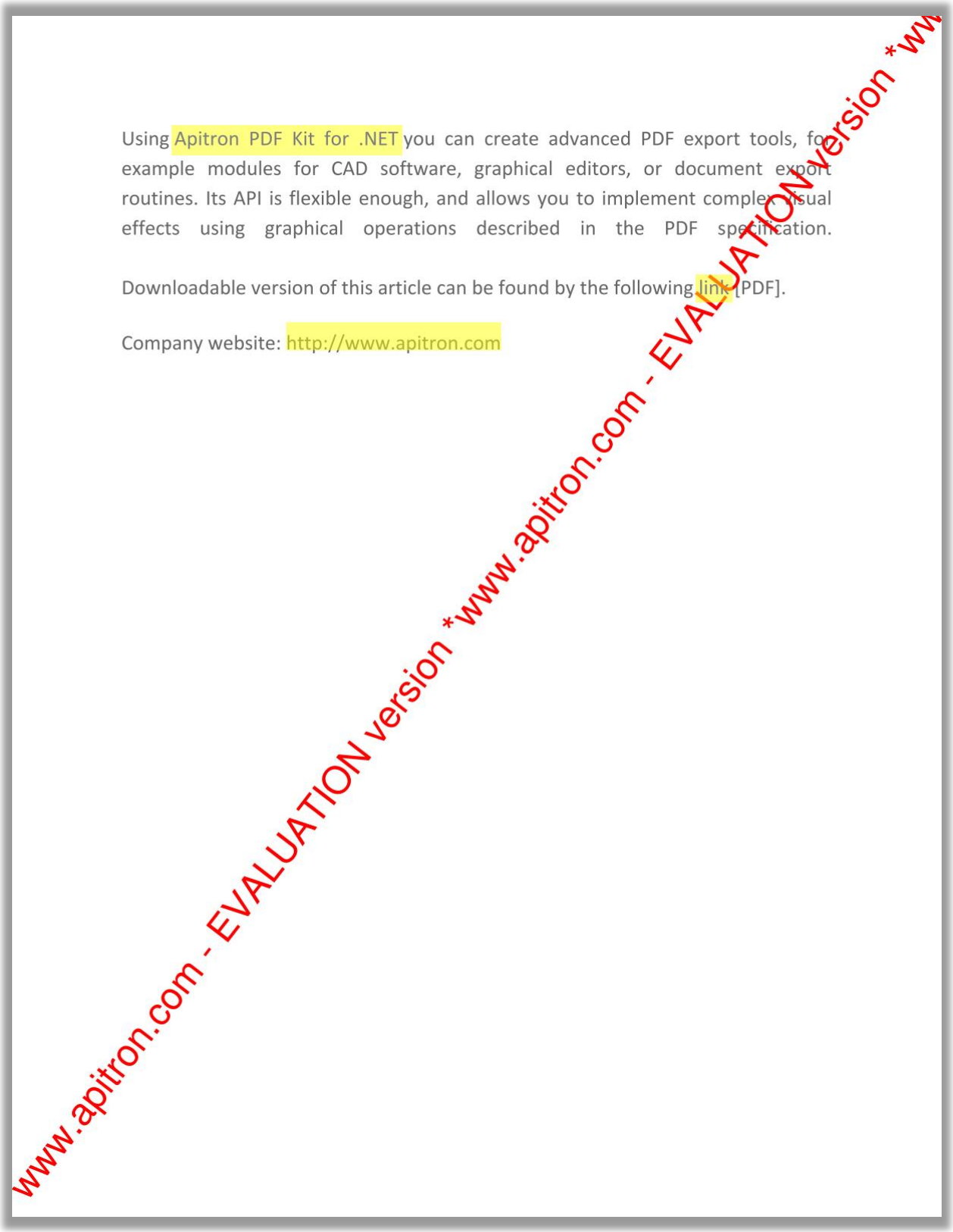
Original document:



Pic. 1 Original PDF document with links

You can see that this document has three links, two of them are marked with link annotation object and the last one is a part of the text.

Resulting image looks as follows:



Pic. 2 PDF page with highlighted links

You can see that all links are highlighted including the link that was found in text and doesn’t have a link annotation created for it.

Summary

Apitron PDF Rasterizer can be used to convert PDF to image or create custom PDF viewers, and provides comprehensive API that helps you to get the job done. It can also be used for searching text and dumping font names used in PDF document. It's cross platform and works on many modern platforms that makes your code reusable and easily portable.