Working with transparent images in PDF

Written by Apitron Documentation Team

Introduction

Transparency in PDF can be specified using various methods and depends on desired effect, all possible techniques are discussed in details in our free book Apitron PDF Kit in Action in sections:

3.3.1.8 Drawing transparent objects in PDF

3.3.4 Transparency Group XObjects

3.3.5 Images

3.3.6 Softmasks

In this post we'll cover the typical task of using a transparent image in PDF document. One may think that such a simple thing should work of the box and shouldn't require any tricky manipulations. That's true for Apitron PDF Kit, and you'll see how easy it can be done, but underlying PDF specifics might be an interesting read by itself.

PDF standard doesn't have a concept of a transparent image as a whole. Instead, a transparent image gets defined as a combination of raster image data samples (pixels) plus an additional object defining the transparency level for each of them.

These objects may define whether the image will be masked using color key mask, explicit (stencil) mask defined by another monochrome image, or a softmask.

The latter allows for smooth masking, gradually changing the transparency level of the image samples, while other methods can only define fully transparent or opaque pixels. And the softmask is the only way to correctly reproduce the transparent PNG for example. Using Apitron PDF Kit API you can manage image transparency yourself or let the library handle it for you. Let' see the code sample in next section.

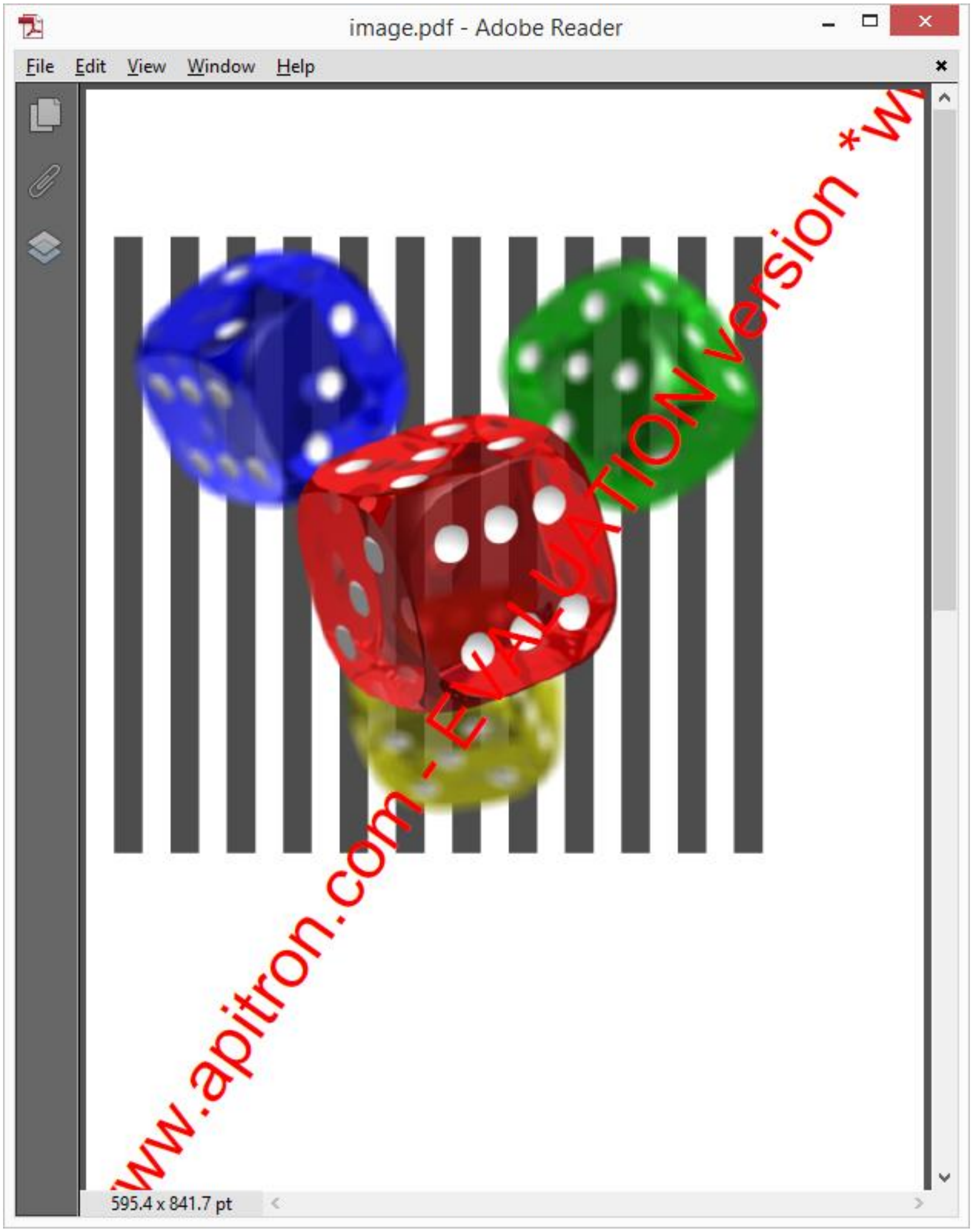
Code sample (C#)

Let’s use the PNG demo image from Wikipedia in our sample, see the code below:

```
using (FileStream outputStream = new FileStream("image.pdf", FileMode.Create, FileAccess.Write))
{
    // create new PDF document
    FixedDocument document = new FixedDocument();
    document.Pages.Add(new Page());

    // create image XObject and set it to use embedded transparency data
    Image maskedImage = new Image("maskedImage", "../../images/cubes.png", true);
    document.ResourceManager.RegisterResource(maskedImage);

    // create page content object
    ClippedContent pageContent = document.Pages[0].Content;

    // define stripe parameters
    double stripeWidth = 20.0;
    double doubleStripeWidth = stripeWidth*2;
    // create stripe rect
    Path path = new Path();
    path.AppendRectangle(0, 0, stripeWidth, maskedImage.Height);

    // set initial translation
    pageContent.Translate(20, 300);
    // save state in order to restore it later for image
    pageContent.SaveGraphicsState();

    // draw gray stripes with a step of 20 points
    for (int i = 0; i < (maskedImage.Width / doubleStripeWidth); ++i)
    {
        pageContent.SetDeviceNonStrokingColor(new double[] {0.3});
        pageContent.FillPath(path);
        pageContent.Translate(doubleStripeWidth, 0);
    }

    pageContent.RestoreGraphicsState();

    // add image
    pageContent.AppendImage(maskedImage.ID, 0, 0, maskedImage.Width, maskedImage.Height);
    document.Save(outputStream);
}
```

This code adds a set of gray stripes to create a background for transparent image and draws it over. Note that when we create the *maskedImage* object we set *useTransparency* flag to ***true***. It indicates that library should automatically handle transparency by creating the corresponding softmask for this image if necessary.

The resulting image is shown below:



Pic. 1 Transparent image in PDF document

Conclusion

Working with transparent images doesn’t need additional coding when you’re using Apitron PDF Kit for .NET component. Read our free book to get the complete understanding on how the transparency works in PDF. It’s an invaluable resource for those interested in PDF internals and advanced processing. Contact us if you have questions or need help, we are always ready to help.