عنوان: **تبدیل PDF به تصویر با استفاده از API توکار Window 8.1 در برنامه‌های غیر مترو دات نت**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۰۱ ۱۴:۵۵
آدرس: www.dotnettips.info
گروه‌ها: WinRT, PdfReport, PDF

ویندوز 8.1 دارای امکانات و API توکاری جهت نمایش و خواندن فایل‌های PDF در برنامه‌های مترو است. در ادامه قصد داریم از این امکانات در یک برنامه‌ی متداول دات نت، برای مثال یک برنامه‌ی کنسول غیر مترو استفاده کنیم.

آماده سازی برنامه‌های دات نت برای دسترسی به API مترو ویندوز 8.1

ابتدا یک برنامه‌ی کنسول دات نت 4.5.1 را آغاز کنید. برای دسترسی به API ویندوز 8.1 حتما نیاز است که حداقل از دات نت 4.5.1 شروع کرد. سپس برنامه را در VS.NET بسته و فایل پروژه آن‌را در یک ادیتور متنی باز کنید.
در ابتدای فایل csproj، نیاز است سطر TargetPlatformVersion ذیل اضافه شود.

```
<PropertyGroup>
    <TargetFrameworkVersion>v4.5.1</TargetFrameworkVersion>
    <TargetPlatformVersion>8.1</TargetPlatformVersion>
  </PropertyGroup>
```

سپس در همین فایل، ارجاعات زیر را نیز اضافه نمائید:

```
<ItemGroup>
    <Reference Include="System" />
    <Reference Include="System.ComponentModel.DataAnnotations" />
    <Reference Include="System.Core" />
    <Reference Include="System.ObjectModel" />
    <Reference Include="System.Xml.Linq" />
    <Reference Include="System.Data.DataSetExtensions" />
    <Reference Include="Microsoft.CSharp" />
    <Reference Include="System.Data" />
    <Reference Include="System.Xml" />
    <Reference Include="System.Threading" />
    <Reference Include="System.Threading.Tasks" />
  </ItemGroup>
  <ItemGroup>
    <Reference Include="Windows" />
    <Reference Include="System.Runtime" />
    <Reference Include="System.Runtime.WindowsRuntime" />
  </ItemGroup>
```

مواردی مانند System.Runtime، System.Runtime.WindowsRuntime امکان دسترسی به API ویندوز 8 را در برنامه‌های دات نت میسر می‌کنند.

یک نکته

اگر می‌خواهید این فرآیند را ساده و خودکار کنید، از قالب‌های پروژه‌ی مخصوص DesktopWinRT.Templates.vsix استفاده نمائید.
DesktopWinRT.Templates.vsix

افزودن ارجاعی به Nito.AsyncEx

چون برنامه‌ی مورد استفاده کنسول است و API ویندوز 8 کاملا async طراحی شده‌است، نیاز است با کمک AsyncContext موجود در کتابخانه‌ی Nito.AsyncEx بتوان از امکانات async و await در متد Main برنامه استفاده کرد. البته اگر از سایر برنامه‌های دسکتاپ استفاده می‌کنید، فقط کافی است امضای متد رخدادن گردان را به async تغییر دهید.

```
install-package Nito.AsyncEx
```

تبدیل استریم‌های دات نت به استریم‌های WinRT

اکثر متدهای WinRT با استریم‌هایی از نوع IRandomAccessStream کار می‌کنند. برای اینکه بتوان استریم استاندارد دات نت را به این نوع تبدیل کرد، می‌توان از کلاس‌های ذیل کمک گرفت:

```
using System;
using System.IO;
using Windows.Storage.Streams;

namespace ConsoleWin81PdfApiTest
{
    public static class MicrosoftStreamExtensions
    {
        public static IRandomAccessStream AsRandomAccessStream(this Stream stream)
        {
            return new RandomStream(stream);
        }

    }

    class RandomStream : IRandomAccessStream
    {
        readonly Stream _internstream;

        public RandomStream(Stream underlyingstream)
        {
            _internstream = underlyingstream;
        }

        public IInputStream GetInputStreamAt(ulong position)
        {
            _internstream.Position = (long)position;
            return _internstream.AsInputStream();
        }

        public IOutputStream GetOutputStreamAt(ulong position)
        {
            _internstream.Position = (long)position;
            return _internstream.AsOutputStream();
        }

        public ulong Size
        {
            get
            {
                return (ulong)_internstream.Length;
            }
            set
            {
                _internstream.SetLength((long)value);
            }
        }

        public bool CanRead
        {
            get { return _internstream.CanRead; }
        }

        public bool CanWrite
        {
            get { return _internstream.CanWrite; }
        }

        public IRandomAccessStream CloneStream()
        {
            throw new NotSupportedException();
        }

        public ulong Position
        {
            get { return (ulong)_internstream.Position; }
        }

        public void Seek(ulong position)
        {
            _internstream.Seek((long)position, SeekOrigin.Begin);
        }
```

```
        public void Dispose()
        {
            _internstream.Dispose();
        }

        public Windows.Foundation.IAsyncOperationWithProgress<IBuffer, uint> ReadAsync(IBuffer buffer,
uint count, InputStreamOptions options)
        {
            return GetInputStreamAt(Position).ReadAsync(buffer, count, options);
        }

        public Windows.Foundation.IAsyncOperation<bool> FlushAsync()
        {
            return GetOutputStreamAt(Position).FlushAsync();
        }

        public Windows.Foundation.IAsyncOperationWithProgress<uint, uint> WriteAsync(IBuffer buffer)
        {
            return GetOutputStreamAt(Position).WriteAsync(buffer);
        }
    }
}
```

تا اینجا به یک متد الحاقی جدیدی به نام AsRandomAccessStream می‌رسیم که امکان تبدیل استریم استاندارد دات نت را به IRandomAccessStream مخصوص WinRT دارد. از آن می‌توان برای باز کردن یک فایل و ارسال استریم آن به توابع WinRT و یا ثبت استریم WinRT در یک فایل استفاده کرد.

خواندن فایل‌های PDF و تبدیل صفحات آن‌ها به تصویر

در ادامه کد کامل استفاده از API جدید ویندوز 8.1 را جهت خواندن فایل‌های PDF ملاحظه می‌کنید. این امکانات جدید در فضای نام Windows.Data.Pdf قرار دارند و صرفا امکان خواندن فایل‌های PDF را تدارک دیده‌اند.

```
using System;
using System.IO;
using System.Threading.Tasks;
using Windows.Data.Pdf;
using Nito.AsyncEx;

namespace ConsoleWin81PdfApiTest
{
    class Program
    {
        static void Main(string[] args)
        {
            AsyncContext.Run(async () =>
            {
                await test();
            });
        }

        private static async Task test()
        {
            using (var randomAccessStream = File.Open("PieChartPdfReport.pdf",
FileMode.Open).AsRandomAccessStream())
            {
                var pdfDocument = await PdfDocument.LoadFromStreamAsync(randomAccessStream);
                for (uint i = 0; i < pdfDocument.PageCount; i++)
                {
                    using (var page = pdfDocument.GetPage(i))
                    {
                        /*var renderOptions = new PdfPageRenderOptions
                        {
                            BackgroundColor = Colors.LightGray,
                            DestinationHeight = (uint) (page.Size.Height*10)
                        };*/

                        using (var stream = File.Open(string.Format("page-{0}.png", i + 1),
FileMode.OpenOrCreate).AsRandomAccessStream())
                        {
                            await page.RenderToStreamAsync(stream/*, renderOptions*/);
                            await stream.FlushAsync();
                        }
                    }
```

```
                }
            }
        }
    }
}
```

توضیحات:

- متد AsyncContext.Run جزو امکانات Nito.AsyncEx است و امکان نوشتن کدهای await دار را در متد Main یک برنامه‌ی کنسول فراهم می‌کند.

- متد File.Openدات نت، خروجی از نوع استریم دارد. برای تبدیل آن به نوع IRandomAccessStream، از متد الحاقی AsRandomAccessStream که پیشتر تهیه کردیم، می‌توان استفاده کرد.

- در ادامه متد PdfDocument.LoadFromStreamAsync این استریم خاص را دریافت کرده و امکان دسترسی به API ویندوز 8.1 را میسر می‌کند.

- توسط متد pdfDocument.GetPage می‌توان به صفحات مختلف فایل PDF باز شده دسترسی یافت. در اینجا متد page.RenderToStreamAsync، سبب رندر شدن صفحه با فرمت PNG می‌شود. این خروجی نهایتا باید در یک استریم از نوع IRandomAccessStream ثبت شود. در اینجا نیز می‌توان از متد File.Open در حالت FileMode.OpenOrCreate استفاده کرد.

- اگر می‌خواهید ابعاد تصویر نهایی و ویژگی‌های آن‌را تغییر دهید، می‌توان از پارامتر دوم متد page.RenderToStreamAsync استفاده کرد که شیءایی از نوع PdfPageRenderOptions را می‌پذیرد.

کدهای کامل این پروژه را از اینجا می‌توانید دریافت کنید

MicrosoftStreamExtensions.zip

برای مطالعه بیشتر

How to use specific WinRT API from Desktop apps

How to call WinRT APIs from .NET desktop apps

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۰۳ ۱۰:۴۹

یک نکته‌ی تکمیلی

اگر با استفاده از مطالب فوق قصد داشته باشید در WPF یک PDF Viewer درست کنید، می‌توان از متد ذیل استفاده کرد:

```
private async Task<List<System.Windows.Media.Imaging.BitmapImage>> getPdfPageImages()
        {
            var results = new List<System.Windows.Media.Imaging.BitmapImage>();
            using (var randomAccessStream = File.Open("PieChartPdfReport.pdf",
FileMode.Open).AsRandomAccessStream())
            {
                var pdfDocument = await PdfDocument.LoadFromStreamAsync(randomAccessStream);
                for (uint i = 0; i < pdfDocument.PageCount; i++)
                {
                    using (var memoryStream = new MemoryStream())
                    {
                        using (var stream = memoryStream.AsRandomAccessStream())
                        {
                            using (var page = pdfDocument.GetPage(i))
                            {
                                // Set render options
                                var renderOptions = new PdfPageRenderOptions
                                {
                                    BackgroundColor = Colors.LightGray,
                                    DestinationHeight = (uint)(page.Size.Height * 10)
                                };

                                await page.RenderToStreamAsync(stream); //, renderOptions);
                                await stream.FlushAsync();

                                var bitmapImage = new System.Windows.Media.Imaging.BitmapImage();
                                bitmapImage.BeginInit();
                                //Without this, BitmapImage uses lazy initialization by default and the
stream will be closed by then.
                                bitmapImage.CacheOption =
System.Windows.Media.Imaging.BitmapCacheOption.OnLoad;
                                bitmapImage.StreamSource = memoryStream;
                                bitmapImage.EndInit();

                                results.Add(bitmapImage);
                            }
                        }
                    }
                }
            }
            return results;
        }
```

بعد برای استفاده از BitmapImage های حاصل از آن، برای مثال نمایش اولین صفحه در یک کنترل Image استاندارد، می‌توان نوشت:

```
private async void Button_Click(object sender, RoutedEventArgs e)
        {
            var images = await this.getPdfPageImages();
            ImagePdf.Source = images.First();
        }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۰۹ ۱۴:۵۱

استفاده از این نکته برای ساخت یک PDF Viewer ساده در WPF.