عنوان: **تبدیل HTML فارسی به PDF با استفاده از افزونه‌ی XMLWorker کتابخانه‌ی iTextSharp**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۲۴ ۱۵:۱۰
آدرس: www.dotnettips.info
گروه‌ها: iTextSharp, PdfReport, PDF

پیشتر مطلبی را در مورد « تبدیل HTML به PDF با استفاده از کتابخانه‌ی iTextSharp » در این سایت مطالعه کرده‌اید. این مطلب از افزونه HTMLWorker کتابخانه iTextSharp استفاده می‌کند که ... مدتی است توسط نویسندگان این مجموعه منسوخ شده اعلام گردیده و دیگر پشتیبانی نمی‌شود.
کتابخانه جایگزین آن‌را افزونه XMLWorker معرفی کرده‌اند که توانایی پردازش CSS و HTML بهتر و کاملتری را نسبت به HTMLWorker ارائه می‌دهد. این کتابخانه نیز همانند HTMLWorker پشتیبانی توکاری از متون راست به چپ و یونیکد فارسی، ندارد و نیاز است برای نمایش صحیح متون فارسی در آن، نکات خاصی را اعمال نمود که در ادامه بحث آن‌ها را مرور خواهیم کرد.

ابتدا برای دریافت آخرین نگارش‌های iTextSharp و افزونه XMLWorker آن به آدرس‌های ذیل مراجعه نمائید:
http://sourceforge.net/projects/itextsharp/files/itextsharp
http://sourceforge.net/projects/itextsharp/files/xmlworker

تهیه یک UnicodeFontProvider

Encoding پیش فرض قلم‌ها در XMLWorker مساوی BaseFont.CP1252 است؛ که از حروف یونیکد پشتیبانی نمی‌کند. برای رفع این نقیصه نیاز است یک منبع تامین قلم سفارشی را برای آن ایجاد نمود:

```
public class UnicodeFontProvider : FontFactoryImp
    {
        static UnicodeFontProvider()
        {
            // روش صحیح تعریف فونت
            var systemRoot = Environment.GetEnvironmentVariable("SystemRoot");
            FontFactory.Register(Path.Combine(systemRoot, "fonts\\tahoma.ttf"));
            // ثبت سایر فونت‌ها در اینجا
            //FontFactory.Register(Path.Combine(Environment.CurrentDirectory, "fonts\\irsans.ttf"));
        }

        public override Font GetFont(string fontname, string encoding, bool embedded, float size, int
style, BaseColor color, bool cached)
        {
            if (string.IsNullOrWhiteSpace(fontname))
                return new Font(Font.FontFamily.UNDEFINED, size, style, color);
            return FontFactory.GetFont(fontname, BaseFont.IDENTITY_H, BaseFont.EMBEDDED, size, style,
color);
        }
    }
```

قلم‌های مورد نیاز را در سازنده کلاس به نحوی که مشاهده می‌کنید، ثبت نمائید.
مابقی مسایل آن خودکار خواهد بود و هر زمانیکه نیاز به قلم خاصی از طرف XMLWorker وجود داشت، به متد GetFont فوق مراجعه کرده و اینبار قلمی با BaseFont.IDENTITY_H را دریافت می‌کند. IDENTITY_H در استاندارد PDF، جهت مشخص ساختن encoding قلم‌هایی با پشتیبانی از یونیکد بکار می‌رود.

تهیه منبع تصاویر

در XMLWorker اگر تصاویر با http شروع نشوند (دریافت تصاویر وب آن خودکار است)، آن تصاویر را از مسیری که توسط پیاده سازی کلاس AbstractImageProvider مشخص خواهد شد، دریافت می‌کند که نمونه‌ای از پیاده سازی آن‌را در ذیل مشاهده می‌کنید:

```
public class ImageProvider : AbstractImageProvider
    {
        public override string GetImageRootPath()
        {
```

```
            var path = Environment.GetFolderPath(Environment.SpecialFolder.MyPictures);
            return path + "\\"; // مهم است که این مسیر به بک اسلش ختم شود تا درست کار کند
        }
    }
```

نحوه تعریف یک فایل CSS خارجی

```
public static class XMLWorkerUtils
    {
        /// <summary>
        /// نحوه تعریف یک فایل سی اس اس خارجی
        /// </summary>
        public static ICssFile GetCssFile(string filePath)
        {
            using (var stream = new FileStream(filePath, FileMode.Open, FileAccess.Read,
FileShare.ReadWrite))
            {
                return XMLWorkerHelper.GetCSS(stream);
            }
        }
    }
```

برای مسیردهی یک فایل CSS در کتابخانه XMLWorker می‌توان از کلاس فوق استفاده کرد.

تبدیل المان‌های HTML پردازش شده به یک لیست PDF ایی

تهیه مقدمات فارسی سازی و نمایش راست به چپ اطلاعات در کتابخانه XMLWorker از اینجا شروع می‌شود. در حالت پیش فرض کار آن، المان‌های HTML به صورت خودکار Parse شده و به صفحه اضافه می‌شوند. به همین دلیل دیگر فرصت اعمال خواص RTL به المان‌های پردازش شده دیگر وجود نخواهد داشت و به صورت توکار نیز این مسایل درنظر گرفته نمی‌شود. به همین دلیل نیاز است که در حین پردازش المان‌های HTML و تبدیل آن‌ها به معادل المان‌های PDF، بتوان آن‌ها را جمع آوری کرد که نحوه انجام آن‌را با پیاده سازی اینترفیس IElementHandler در ذیل مشاهده می‌کنید:

```
/// <summary>
    /// معادل پی دی افی المان‌های اچ تی ام ال را جمع آوری می‌کند
    /// </summary>
    public class ElementsCollector : IElementHandler
    {
        private readonly Paragraph _paragraph;

        public ElementsCollector()
        {
            _paragraph = new Paragraph
            {
                Alignment = Element.ALIGN_LEFT  // سبب می‌شود تا در حالت راست به چپ از سمت راست
صفحه شروع شود
            };
        }

        /// <summary>
        /// این پاراگراف حاوی کلیه المان‌های متن است
        /// </summary>
        public Paragraph Paragraph
        {
            get { return _paragraph; }
        }

        /// <summary>
        /// بجای اینکه خود کتابخانه اصلی کار افزودن المان‌ها را به صفحات انجام دهد
        /// قصد داریم آن‌ها را ابتدا جمع آوری کرده و سپس به صورت راست به چپ به صفحات نهایی اضافه کنیم
        /// </summary>
        /// <param name="htmlElement"></param>
        public void Add(IWritable htmlElement)
        {
            var writableElement = htmlElement as WritableElement;
            if (writableElement == null)
                return;
```

```
            foreach (var element in writableElement.Elements())
            {
                fixNestedTablesRunDirection(element);
                _paragraph.Add(element);
            }
        }

        /// <summary>
        /// نیاز است سلول‌های جداول تو در توی پی دی اف نیز راست به چپ شوند
        /// </summary>
        private void fixNestedTablesRunDirection(IElement element)
        {
            var table = element as PdfPTable;
            if (table == null)
                return;

            table.RunDirection = PdfWriter.RUN_DIRECTION_RTL;
            foreach (var row in table.Rows)
            {
                foreach (var cell in row.GetCells())
                {
                    cell.RunDirection = PdfWriter.RUN_DIRECTION_RTL;
                    foreach (var item in cell.CompositeElements)
                    {
                        fixNestedTablesRunDirection(item);
                    }
                }
            }
        }
    }
```

این کلاس کلیه المان‌های دریافتی را به یک پاراگراف اضافه می‌کند. همچنین اگر به جدولی در این بین برخورد، مباحث RTL آن‌را نیز اصلاح خواهد نمود.

یک مثال کامل از نحوه کنار هم قرار دادن پیشنیازهای تهیه شده

خوب؛ تا اینجا یک سری پیشنیاز را تهیه کردیم، اما XMLWorker از وجود آن‌ها بی‌خبر است. برای معرفی آن‌ها باید به نحو ذیل عمل کرد:

```
using (var pdfDoc = new Document(PageSize.A4))
            {
                var pdfWriter = PdfWriter.GetInstance(pdfDoc, new FileStream("test.pdf",
FileMode.Create));
                pdfWriter.RgbTransparencyBlending = true;
                pdfDoc.Open();


                var html = @"<span style='color:blue; font-family:tahoma;'><b>آزمایش</b></span>
                                کتابخانه <i>iTextSharp</i> <u>جهت بررسی فارسی نویسی</u>
                            <table style='color:blue; font-family:tahoma;'
border='1'><tr><td>eeمتن</td></tr></table>
                            <code>This is a code!</code>
                            <br/>
                            <img src='av-13489.jpg' />
                            ";

                var cssResolver = new StyleAttrCSSResolver();
                // cssResolver.AddCss(XMLWorkerUtils.GetCssFile(@"c:\path\pdf.css"));
                cssResolver.AddCss(@"code
                                    {
                                       padding: 2px 4px;
                                       color: #d14;
                                       white-space: nowrap;
                                       background-color: #f7f7f9;
                                       border: 1px solid #e1e1e8;
                                    }",
                                    "utf-8", true);

                // کار جمع آوری المان‌های ترجمه شده به المان‌های پی دی اف را انجام می‌دهد
                var elementsHandler = new ElementsCollector();

                var htmlContext = new HtmlPipelineContext(new CssAppliersImpl(new
UnicodeFontProvider()));
```

```
                htmlContext.SetImageProvider(new ImageProvider());
                htmlContext.CharSet(Encoding.UTF8);

htmlContext.SetAcceptUnknown(true).AutoBookmark(true).SetTagFactory(Tags.GetHtmlTagProcessorFactory());
                var pipeline = new CssResolverPipeline(cssResolver,
                                                       new HtmlPipeline(htmlContext, new
ElementHandlerPipeline(elementsHandler, null)));
                var worker = new XMLWorker(pipeline, parseHtml: true);
                var parser = new XMLParser();
                parser.AddListener(worker);
                parser.Parse(new StringReader(html));

                // با هندلر سفارشی که تهیه کردیم تمام المان‌های اچ تی ام ال به المان‌های پی دی اف تبدیل
شدند
                // الان تنها کافی کافی است تا این‌ها را در یک جدول راست به چپ محصور کنیم تا درست
نمایش داده شوند
                var mainTable = new PdfPTable(1) { WidthPercentage = 100, RunDirection =
PdfWriter.RUN_DIRECTION_RTL };
                var cell = new PdfPCell
                {
                    Border = 0,
                    RunDirection = PdfWriter.RUN_DIRECTION_RTL,
                    HorizontalAlignment = Element.ALIGN_LEFT
                };
                cell.AddElement(elementsHandler.Paragraph);
                mainTable.AddCell(cell);

                pdfDoc.Add(mainTable);
            }

            Process.Start("test.pdf");
```

نحوه تعریف inline css یا نحوه افزودن یک فایل css خارجی را نیز در ابتدای این مثال مشاهده می‌کنید.
UnicodeFontProvider باید به HtmlPipelineContext شناسانده شود.
ImageProvider توسط متد SetImageProvider به HtmlPipelineContext معرفی می‌شود.
ElementsCollector سفارشی ما در قسمت CssResolverPipeline باید به سیستم تزریق شود.
پس از آن XMLWorker را وادار می‌کنیم تا HTML را Parse کرده و معادل المان‌های PDF ایی آن‌را تهیه کند؛ اما آن‌ها را به صورت خودکار به صفحات فایل PDF نهایی اضافه نکند. در این بین ElementsCollector ما این المان‌ها را جمع آوری کرده و در نهایت، پاراگراف کلی حاصل از آن‌را به یک جدول با RUN_DIRECTION_RTL اضافه می‌کنیم. حاصل آن نمایش صحیح متون فارسی است.

کدهای مثال فوق را از آدرس ذیل نیز می‌توانید دریافت کنید:

XMLWorkerRTLsample.cs

به روز رسانی

کلیه نکات مطلب فوق را به همراه بهبودهای مطرح شده در نظرات آن، در پروژه‌ی ذیل می‌توانید به صورت یکجا دریافت و بررسی کنید:

XMLWorkerRTLsample.zip

نظرات خوانندگان

نویسنده: ح م
تاریخ: ۱۳۹۲/۰۸/۱۶ ۹:۱۸

همه‌ی فونت‌ها و استایل‌ها را هم که پیوست می‌کنم، برای برخی از کدهای HTML ، خروجی pdf سفید(خالی) است.
راهی وجود دارد که خطاهای پارسر دست کم در حالت Debug نشان داده شوند؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۱۶ ۱۰:۱۱

بله. یک کلاس لاگر سفارشی درست کنید:

```
public class CustomLogger : iTextSharp.text.log.ILogger
    {
        public iTextSharp.text.log.ILogger GetLogger(Type klass)
        {
            return this;
        }

        public iTextSharp.text.log.ILogger GetLogger(string name)
        {
            return this;
        }

        public bool IsLogging(iTextSharp.text.log.Level level)
        {
            return true;
        }

        public void Warn(string message)
        {
            System.Diagnostics.Trace.TraceWarning(message);
        }

        public void Trace(string message)
        {
            System.Diagnostics.Trace.TraceInformation(message);
        }

        public void Debug(string message)
        {
            System.Diagnostics.Trace.TraceInformation(message);
        }

        public void Info(string message)
        {
            System.Diagnostics.Trace.TraceInformation(message);
        }

        public void Error(string message)
        {
            System.Diagnostics.Trace.TraceError(message);
        }

        public void Error(string message, Exception e)
        {
            System.Diagnostics.Trace.TraceError(message + System.Environment.NewLine + e);
        }
    }
```

بعد در ابتدای اجرای برنامه آن‌را ثبت کنید:

```
iTextSharp.text.log.LoggerFactory.GetInstance().SetLogger(new CustomLogger());
```

خروجی‌ها در پنجره دیباگ VS.NET نمایش داده می‌شوند.

نویسنده: مهرداد
تاریخ: ۱۳۹۲/۰۸/۲۴ ۲۱:۱۳

سلام دوست عزیز
من کد بالا رو تست کردم ظاهرا تگ‌های div رو نشون نمیده و از بین می‌بره !

تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۲۴ ۲۱:۳۱

- نام این کتابخانه Worker **XML** هست. یعنی HTML شما باید معتبر باشد و تگ‌های آن همانند یک فایل XML درست تشکیل و باز و بسته شده باشند؛ چیزی مثل XHTML ها.
- می‌توانید از کتابخانه HTML Agility pack برای درست کردن XHTML استفاده کنید:

```
var sb = new StringBuilder();
var stringWriter = new StringWriter(sb);
var doc = new HtmlDocument
            {
                OptionOutputAsXml = true,
                OptionCheckSyntax = true,
                OptionFixNestedTags = true,
                OptionAutoCloseOnEnd = true,
                OptionDefaultStreamEncoding = Encoding.UTF8
            };
doc.LoadHtml(htmlContent);
doc.Save(stringWriter);
var xhtml = sb.ToString();
```

خاصیت OptionOutputAsXml آن‌را true کنید تا در حد توانش مشکلات HTML شما را برطرف و یک خروجی XHTML را تولید کند.
- سایر مشکلات آن‌را بهتر است در mailing لیست آن‌ها به همراه ارائه مثال قابل بازتولیدی ارسال کنید.

نویسنده: جلال
تاریخ: ۱۳۹۲/۱۲/۱۵ ۸:۵۰

با سلام؛ من خیلی دنبال کلاس HtmlDocument گشتم، اما نه توی net. پیدا کردم و نه توی سایت خودتون، میتونید راهنماییم کنید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۵ ۹:۲۵

«می‌توانید از کتابخانه HTML Agility pack استفاده کنید»

```
PM> Install-Package HtmlAgilityPack
```

نویسنده: جلال
تاریخ: ۱۳۹۲/۱۲/۱۵ ۱۱:۴

من کدی که فرمودید رو اضافه کردم، همچنین، کد Html هم Valid هستش، و کلا با div ساخته شده، اما pdf خروجی سفید هستش.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۵ ۱۲:۳۵

برای رفع مشکل محو شدن Div، کدهای کلاس ElementsCollector مطلب جاری را به نحو زیر تغییر دهید:

```
public void Add(IWritable htmlElement)
        {
```

```
            var writableElement = htmlElement as WritableElement;
            if (writableElement == null)
                return;

            foreach (var element in writableElement.Elements())
            {
                var div = element as PdfDiv;
                if (div != null)
                {
                    foreach (var divChildElement in div.Content)
                    {
                        fixNestedTablesRunDirection(divChildElement);
                        _paragraph.Add(divChildElement);
                    }
                }
                else
                {
                    fixNestedTablesRunDirection(element);
                    _paragraph.Add(element);
                }
            }
        }
```

نویسنده: سمیه
تاریخ: ۱۳۹۳/۰۱/۱۹ ۱۵:۳۹

سلام؛ ضمن تشکر از مطلب مفیدتان من نمونه کدهایی که در قسمت پایین قرار داده بودید، دانلود کردم. همچنین آخرین نگارش‌های iTextSharp و افزونه XMLWorker را از لینک هایی معرفی شده دانلود و dll هایشان را به پروژه ام اضافه کرده ام، ولی با وجود این به فضای نام iTextSharp.tool خطا می‌دهد و آن را نمی‌شناسد. می‌شه لطفا من راهنمایی کنید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۱۹ ۱۷:۲۰

پروژه شما باید ارجاعاتی را به دو فایل itextsharp.dll و itextsharp.xmlworker.dll داشته باشد.

```
PM> Install-Package iTextSharp
PM> Install-Package itextsharp.xmlworker
```

نویسنده: هیمن صادقی
تاریخ: ۱۳۹۳/۰۲/۲۵ ۱۹:۰

درود
با سپاس از مطالب که در سایت قرار دادید یک مشکل داشتم
من کد رو در پروژه قرار دارم اما کد زیر که قرار متن راست به چپ کنه کار نمی‌کنه

```
_paragraph = new Paragraph
  {
Alignment = Element.ALIGN_LEFT  // سبب می‌شود تا در حالت راست به چپ از سمت راست صفحه شروع شود
  };
```

و کد زیر هم کار نمی‌کنه

```
fixNestedTablesRunDirection(element);
```

اگر لطف کنید من رو راهنمایی کنید

نویسنده: هیمن صادقی
تاریخ: ۱۳۹۳/۰۲/۲۵ ۲۲:۲۸

درود؛ پیوست پیام قبلی که گفتم کد کار نمی‌کنه: rar.1

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۶ ۰:۲۹

- XML Worker از تمام امکانات CSS پشتیبانی نمی‌کند. لیست موارد پشتیبانی شده در اینجا (رنگ‌های سبز)
- در کد شما float: left و float: right دارید که مطابق لینک داده شده فعلا پشتیبانی نمی‌شود.
- نکته‌ی تکمیلی « برای رفع مشکل محو شدن Div، کدهای کلاس ElementsCollector مطلب جاری را به نحو زیر تغییر دهید » را هم اضافه نکرده‌اید.
- کد fixNestedTablesRunDirection مطلب جاری در کدهای شما به نمونه‌ای که cell.RunDirection = PdfWriter.RUN_DIRECTION_RTL ندارد، تغییر پیدا کرده. بنابراین کار نخواهد کرد.

نویسنده: هیمن صادقی
تاریخ: ۱۳۹۳/۰۲/۲۶ ۱:۱۹

ممنون از شما

تابع fixNestedTablesRunDirection در خط

```
if (table == null)
                return;
```

خاتمه پیدا می‌کند و کدی را که برداشتم تاثیر بر کد نداره. زمانیکه به صورت دستی کد زیر را به متن اضافه می‌کنیم

```
paragraph.Add("Data")
```

کار میکنه یعنی راست به چپ را درست می‌کند. اما زمانی که فایل html بهش میدم چپ به راست می‌باشد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۶ ۲:۹

متد Add را به این صورت اصلاح کنید تا جهت Paragraph ها را هم درست کند:

```
public void Add(IWritable htmlElement)
        {
            var writableElement = htmlElement as WritableElement;
            if (writableElement == null)
                return;

            foreach (var element in writableElement.Elements())
            {
                if (element is PdfDiv)
                {
                    var div = element as PdfDiv;
                    foreach (var divChildElement in div.Content)
                    {
                        fixNestedTablesRunDirection(divChildElement);
                        _paragraph.Add(divChildElement);
                    }
                }
                else if(element is Paragraph)
                {
                    var paragraph = element as Paragraph;
                    paragraph.Alignment = Element.ALIGN_LEFT;
                    _paragraph.Add(element);
                }
                else
                {
                    fixNestedTablesRunDirection(element);
                    _paragraph.Add(element);
                }
            }
        }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۶ ۲:۱۲

یک نکته‌ی مهم

از خروجی GetBuffer استریم نباید استفاده شود:

```
return File(memoryStream.GetBuffer(), "application/pdf", "Test.pdf");
```

باید از ToArray استفاده کنید تا حاوی اضافات بافر نباشد (نمایش پیغام ذخیره تغییرات در adobe reader به همین دلیل اضافات است):

```
return File(memoryStream.ToArray(), "application/pdf", "Test.pdf");
```

در این حالت حجم فایل نهایی هم نصف خواهد بود.

نویسنده: الیاس سربند
تاریخ: ۱۳۹۳/۰۳/۰۷ ۱۷:۳۸

سلام و خسته نباشید. میشه از این روش توی ASP.Net استفاده کرد؟ اگر آره در مورد دستور آخر Process.Start چه باید کرد؟
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۷ ۱۷:۵۶

- مثال پیوست شده کمی بالاتر یک مثال ASP.NET MVC است.
- Process.Start را حذف کنید؛ نیازی نیست.
- به قسمت new FileStream آن دقت کنید. اینجا مسیر یک فایل را می‌شود مشخص کرد. فایل نهایی تولید شده در این مسیر نوشته می‌شود. از آن مسیر در برنامه‌های وب و ویندوز می‌توان استفاده کرد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۷ ۱۸:۳

به روز رسانی

کلیه نکات مطلب فوق را به همراه بهبودهای مطرح شده در نظرات آن، در پروژه‌ی ذیل می‌توانید به صورت یکجا دریافت و بررسی کنید:

XMLWorkerRTLsample.zip

نویسنده: مصطفی سلطانی
تاریخ: ۱۳۹۳/۰۶/۰۱ ۱۲:۲۳

با سلام
با تشکر از مطلب مفیدتان
من پروژه نمونه شما را دانلود کردم ولی داخل جدول مشکل راست به چپ فارسی را مشاهده می‌کنم. مثلا لغت "متن" به صورت "ن ت م" نشان داده می‌شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۰۱ ۱۳:۱۷

ابتدای متد Add فایل ElementsCollector.cs آن‌را به صورت زیر اصلاح کنید:

```
public void Add(IWritable htmlElement)
```

```
        {
            var writableElement = htmlElement as WritableElement;
            if (writableElement == null)
                return;

            foreach (var element in writableElement.Elements())
            {
                if (element is NoNewLineParagraph)
                {
                    var noNewLineParagraph = element as NoNewLineParagraph;
                    foreach (var item in noNewLineParagraph)
                    {
                        fixNestedTablesRunDirection(item);
                        _paragraph.Add(item);
                    }
                }
                else if (element is PdfDiv)
```