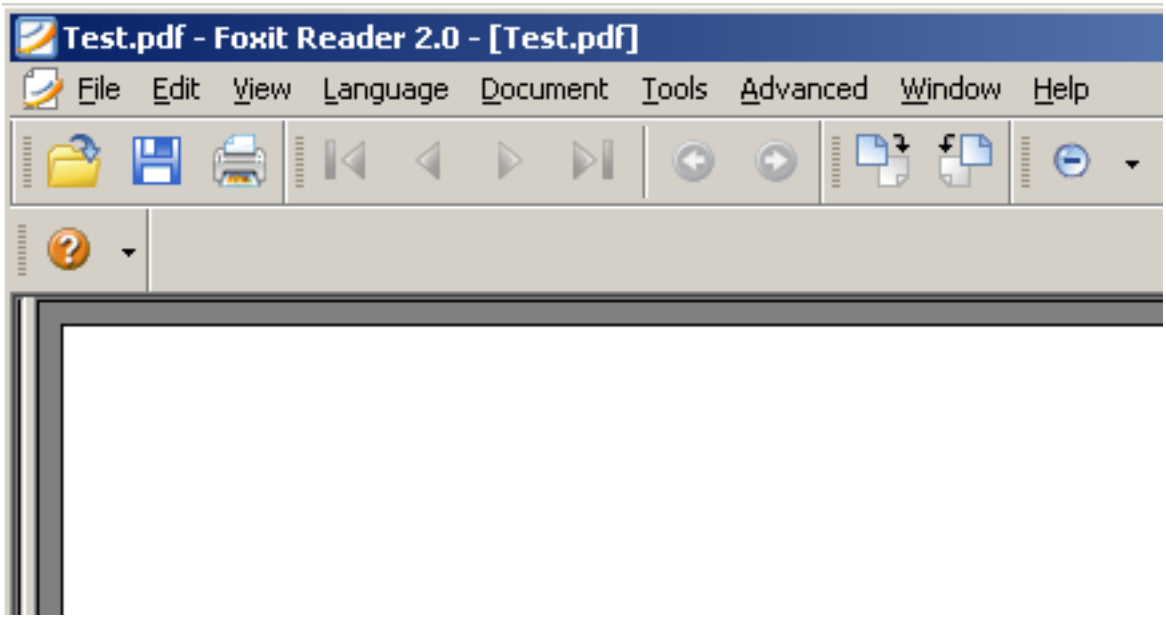
عنوان: **فارسی نویسی و iTextSharp**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۴/۱۰ ۱۰:۴۴:۰۰
آدرس: www.dotnettips.info
گروه‌ها: iTextSharp

شرح یک سری سعی و خطا!
سعی اول:

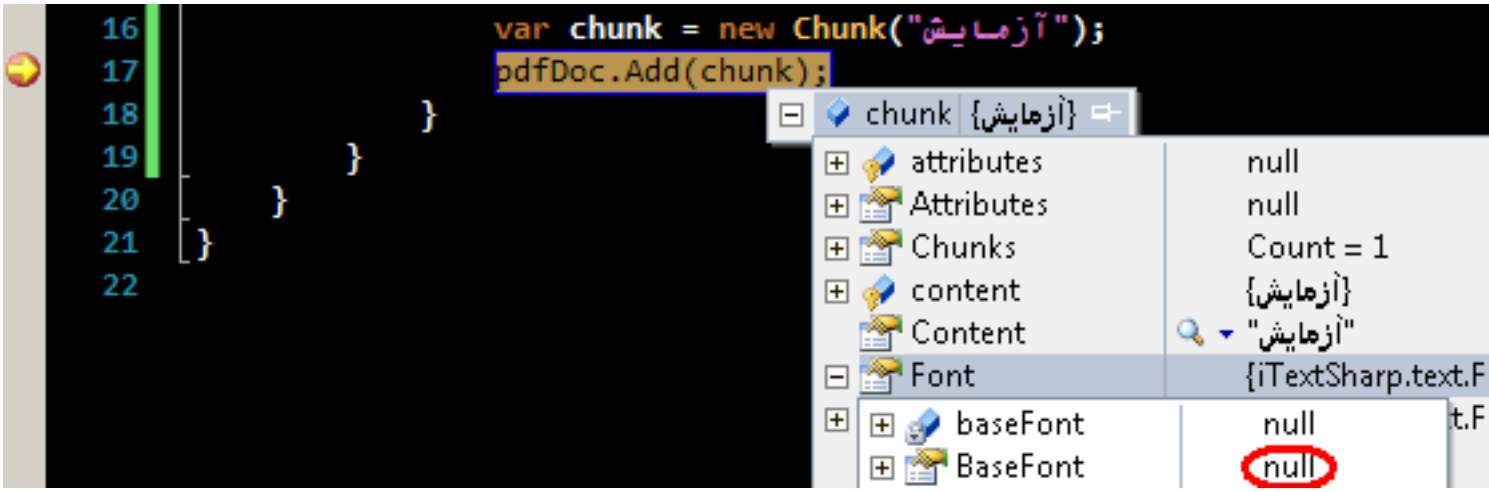
```
using System.IO;
using iTextSharp.text;
using iTextSharp.text.pdf;

namespace iTextSharpTests
{
   class Program
   {
       static void Main(string[] args)
       {
           using (var pdfDoc = new Document(PageSize.A4))
           {
               PdfWriter.GetInstance(pdfDoc, new FileStream("Test.pdf", FileMode.Create));
               pdfDoc.Open();

               var chunk = new Chunk("آزمایش");
               pdfDoc.Add(chunk);
           }
       }
   }
}
```

نتیجه:



بله! هیچی!

مشکل از کجاست؟
در iTextSharp بر اساس نوع فونت انتخابی و encoding مرتبط، نحوه‌ی رندر سازی حروف مشخص می‌شود:



همانطور که ملاحظه می‌کنید، فونت پایه متنی که قرار است اضافه شود، null است.

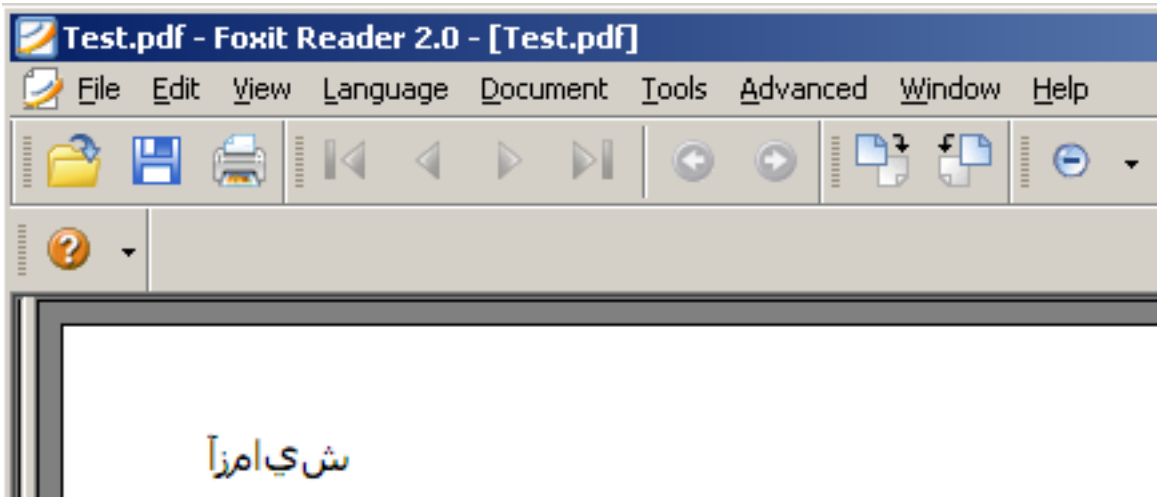
سعی دوم:
اینبار فونت را تنظیم می‌کنیم:

```
using System;
using System.IO;
using iTextSharp.text;
using iTextSharp.text.pdf;

namespace iTextSharpTests
{
   class Program
   {
       static void Main(string[] args)
       {
           using (var pdfDoc = new Document(PageSize.A4))
           {
               PdfWriter.GetInstance(pdfDoc, new FileStream("Test.pdf", FileMode.Create));
               pdfDoc.Open();

               var fontPath = Environment.GetEnvironmentVariable("SystemRoot") + "\\fonts\\tahoma.ttf";
               var baseFont = BaseFont.CreateFont(fontPath, BaseFont.IDENTITY_H, BaseFont.EMBEDDED);
               var tahomaFont = new Font(baseFont, 10, Font.NORMAL, BaseColor.BLACK);

               var chunk = new Chunk("آزمایش",tahomaFont);
               pdfDoc.Add(chunk);
           }
       }
   }
}
```

توضیحات:
متد BaseFont.CreateFont می‌تواند مسیری از فونت مورد نظر را دریافت کند. این حالت خصوصا برای برنامه‌های وب که ممکن است فونت مورد نظر آن‌ها در سرور نصب نشده باشد، بسیار مفید است و لزومی ندارد که الزاما فونت مورد استفاده در پوشه fonts ویندوز نصب شده باشد.

نکات مهم دیگر بکار گرفته شده در این متد، استفاده از BaseFont.IDENTITY_H و BaseFont.EMBEDDED است. به این صورت encoding متن، جهت نوشتن متون غیر Ansi تنظیم می‌شود و در این حالت حتما باید فونت را در فایل، مدفون (embed) نمود. از این لحاظ که عموما این نوع فونت‌ها در سیستم‌های کاربران نصب نیستند.

نتیجه:



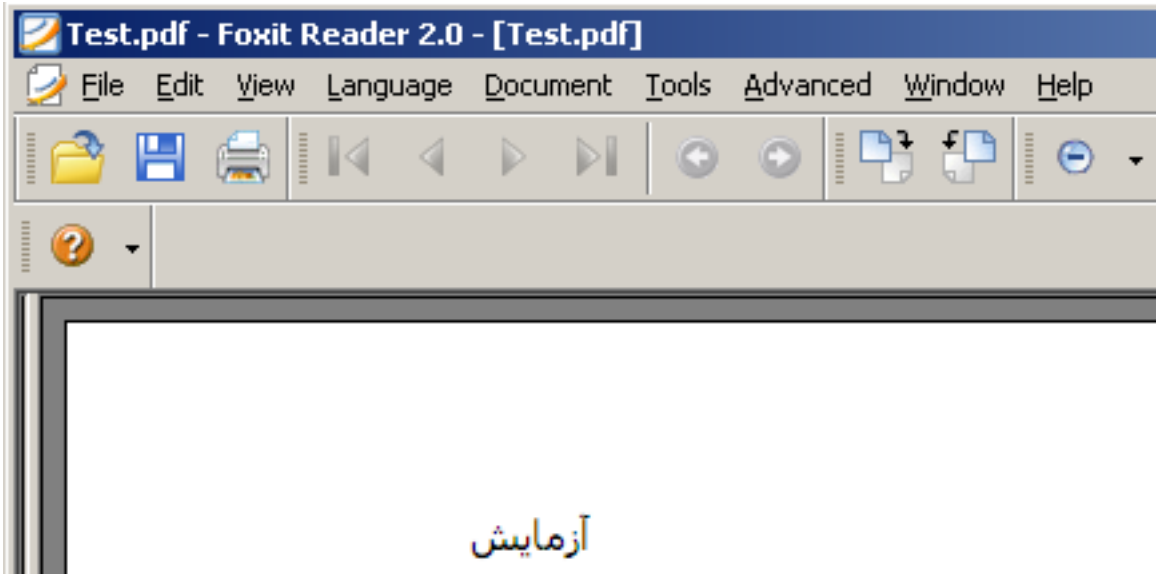
بد نیست! حداقل حروف نمایش داده شدند؛ اما نیاز است تا چرخانده یا معکوس شوند. برای انجام خودکار آن حداقل دو کار را می‌توان انجام داد.

الف) استفاده از ColumnText و اعمال تنظیمات راست به چپ آن

```
using System;
using System.IO;
using iTextSharp.text;
using iTextSharp.text.pdf;

namespace iTextSharpTests
{
   class Program
   {
       static void Main(string[] args)
       {
           using (var pdfDoc = new Document(PageSize.A4))
           {
               var pdfWriter = PdfWriter.GetInstance(pdfDoc, new FileStream("Test.pdf",
FileMode.Create));
               pdfDoc.Open();

               var fontPath = Environment.GetEnvironmentVariable("SystemRoot") + "\\fonts\\tahoma.ttf";
               var baseFont = BaseFont.CreateFont(fontPath, BaseFont.IDENTITY_H, BaseFont.EMBEDDED);
               var tahomaFont = new Font(baseFont, 10, Font.NORMAL, BaseColor.BLACK);

               ColumnText ct = new ColumnText(pdfWriter.DirectContent);
               ct.RunDirection = PdfWriter.RUN_DIRECTION_RTL;
               ct.SetSimpleColumn(100, 100, 500, 800, 24, Element.ALIGN_RIGHT);

               var chunk = new Chunk("آزمایش", tahomaFont);

               ct.AddElement(chunk);
               ct.Go();
           }
       }
   }
}
```

توضیحات:
در اینجا یک ColumnTex جدید تعریف و سپس خصوصیات این ستون تنظیم شده، به همراه RunDirection آن که اصل قضیه است. سپس chunk تعریف شده را به این ستون اضافه کرده‌ایم.

نتیجه:



بله! کار کرد!

ب) استفاده از PdfTable و اعمال تنظیمات راست به چپ آن

```
using System;
using System.IO;
using iTextSharp.text;
using iTextSharp.text.pdf;

namespace iTextSharpTests
{
   class Program
   {
       static void Main(string[] args)
       {
           using (var pdfDoc = new Document(PageSize.A4))
           {
               var pdfWriter = PdfWriter.GetInstance(pdfDoc, new FileStream("Test.pdf",
FileMode.Create));
               pdfDoc.Open();

               var fontPath = Environment.GetEnvironmentVariable("SystemRoot") + "\\fonts\\tahoma.ttf";
               var baseFont = BaseFont.CreateFont(fontPath, BaseFont.IDENTITY_H, BaseFont.EMBEDDED);
               var tahomaFont = new Font(baseFont, 10, Font.NORMAL, BaseColor.BLACK);

               PdfPTable table = new PdfPTable(numColumns: 1);
               table.RunDirection = PdfWriter.RUN_DIRECTION_RTL;
               table.ExtendLastRow = true;

               PdfPCell pdfCell = new PdfPCell(new Phrase("آزمایش", tahomaFont));
               pdfCell.RunDirection = PdfWriter.RUN_DIRECTION_RTL;
```

```
                table.AddCell(pdfCell);
                pdfDoc.Add(table);
            }
        }
    }
}
```

در حین استفاده از PdfTable هم لازم است تا RunDirection مربوط به خود جدول و همچنین هر سلول اضافه شده به آن به RTL تنظیم شوند.

این نکات در هر جایی که با این کتابخانه سر و کار داریم باید اعمال شوند. برای مثال:

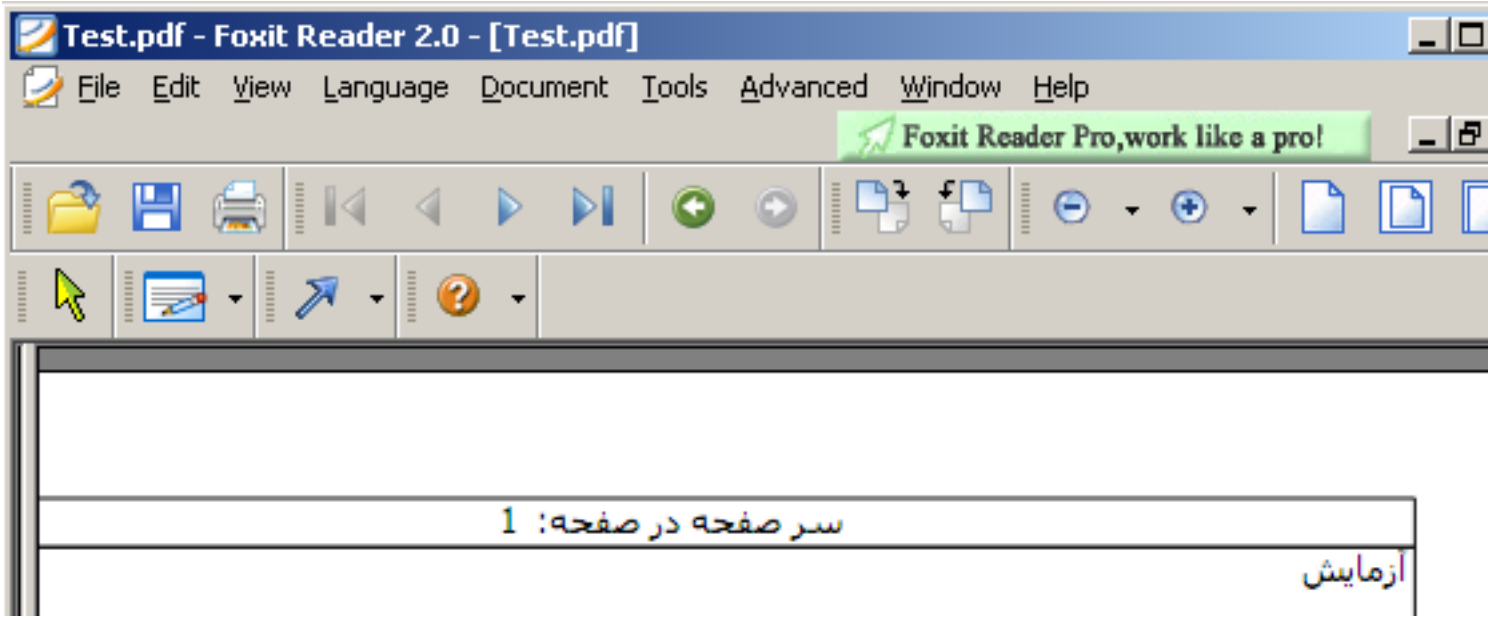
افزودن Header به صفحات Pdf :

افزودن header در نگارش‌های جدید iTextSharp شامل نکته استفاده از کلاس PdfPageEventHelper به شرح زیر است (و مثال‌هایی را که در وب پیدا خواهید کرد، هیچکدام با آخرین نگارش موجود iTextSharp کار نمی‌کنند):

```
using System;
using System.IO;
using iTextSharp.text;
using iTextSharp.text.pdf;

namespace iTextSharpTests
{
   public class PageEvents : PdfPageEventHelper
   {
       Font _font;
       public PageEvents()
       {
           var fontPath = Environment.GetEnvironmentVariable("SystemRoot") + "\\fonts\\tahoma.ttf";
           var baseFont = BaseFont.CreateFont(fontPath, BaseFont.IDENTITY_H, BaseFont.EMBEDDED);
           _font = new Font(baseFont, 10, Font.NORMAL, BaseColor.BLACK);
       }

       public override void OnStartPage(PdfWriter writer, Document document)
       {
           base.OnStartPage(writer, document);

           PdfPTable table = new PdfPTable(numColumns: 1);
           table.RunDirection = PdfWriter.RUN_DIRECTION_RTL;

           PdfPCell pdfCell = new PdfPCell(new Phrase("سر صفحه در صفحه:  " + writer.PageNumber,
_font));
           pdfCell.RunDirection = PdfWriter.RUN_DIRECTION_RTL;
           pdfCell.HorizontalAlignment = Element.ALIGN_CENTER;

           table.AddCell(pdfCell);
           document.Add(table);
       }
   }

   class Program
   {
       static void Main(string[] args)
       {
           using (var pdfDoc = new Document(PageSize.A4))
           {
               var pdfWriter = PdfWriter.GetInstance(pdfDoc, new FileStream("Test.pdf",
FileMode.Create));
               pdfWriter.PageEvent = new PageEvents();
               pdfDoc.Open();

               var fontPath = Environment.GetEnvironmentVariable("SystemRoot") + "\\fonts\\tahoma.ttf";
               var baseFont = BaseFont.CreateFont(fontPath, BaseFont.IDENTITY_H, BaseFont.EMBEDDED);
               var tahomaFont = new Font(baseFont, 10, Font.NORMAL, BaseColor.BLACK);

               PdfPTable table = new PdfPTable(numColumns: 1);
               table.RunDirection = PdfWriter.RUN_DIRECTION_RTL;
               table.ExtendLastRow = true;

               PdfPCell pdfCell = new PdfPCell(new Phrase("آزمایش", tahomaFont));
               pdfCell.RunDirection = PdfWriter.RUN_DIRECTION_RTL;
```

```
                table.AddCell(pdfCell);
                pdfDoc.Add(table);

                pdfDoc.NewPage();
            }
        }
    }
}
```

نتیجه:



تنها نکته‌ای که اینجا اضافه شده، تعریف کلاس PageEvents است که از کلاس PdfPageEventHelper مشتق شده است. در این کلاس می‌توان یک سری متد کلاس پایه را تحریف کرد و header و footer و غیره را اضافه نمود. سپس جهت اضافه کردن آن، pdfWriter.PageEvent باید مقدار دهی شود.

در اینجا هم اگر نوع فونت، encoding و PdfTable به همراه RunDirection آن اضافه نمی‌شد، یا چیزی در header صفحه قابل مشاهده نبود یا متن مورد نظر معکوس نمایش داده می‌شد.

نظرات خوانندگان

نویسنده: Alex Kh58
تاریخ: ۱۳۹۰/۰۴/۱۱ ۱۳:۵۶:۱۴

ممنون از مطلب خوبتون مثل همیشه عالی.

نویسنده: Hosein Mohtasham
تاریخ: ۱۳۹۰/۰۵/۰۹ ۱۵:۱۸:۲۲

خدا خیرت بده مشکل جماعتی رو حل کردی

نویسنده: Hossein Hariri
تاریخ: ۱۳۹۰/۰۶/۲۱ ۱۳:۴۱:۴۸

ممنونم، فقط این کار در زمانی که از html استفاده میکنم جواب نمیده.
یعنی با html هم کار میکنه؟ مشکل کار من کجاست؟ باید کار اضافه ای کنم که نمیکنم؟؟؟
لطفا راهنمایی بفرمایید

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۲۱ ۱۶:۱۱:۳۲

در مورد html to pdf فارسی، یک سری نکات خاص خودش وجود دارد که مفصل توضیح دادم: (+)

نویسنده: سید مجتبی
تاریخ: ۱۳۹۱/۰۴/۱۵ ۳:۱۰

ممنون آقا وحید، عالی

نویسنده: محسن
تاریخ: ۱۳۹۱/۰۶/۱۲ ۱:۹

با سلام ، من کد زیر رو نوشتم

```
var m = new ITModel.ITModelContainer();
        var list = (from pp in m.PERSONNELs
                    select pp).ToList();

        string pdfpath = Server.MapPath("PDFs");
        using (var pdfDoc = new Document(PageSize.A4))
        {
            var pdfWriter = PdfWriter.GetInstance(pdfDoc, new FileStream(pdfpath + "/Personnel2.pdf",
FileMode.Create));
            pdfDoc.Open();
            var fontPath = Environment.GetEnvironmentVariable("SystemRoot") + "\\fonts\\tahoma.ttf";
            var baseFont = BaseFont.CreateFont(fontPath, BaseFont.IDENTITY_H, BaseFont.EMBEDDED);
            var tahomaFont = new Font(baseFont, 10, Font.NORMAL, BaseColor.BLACK);
            float[] widths = new float[] { 1f, 2f };

            PdfPTable table = new PdfPTable(2)
                {
                    TotalWidth = 216f,
                    LockedWidth = true,
                    HorizontalAlignment = 0,
                    SpacingBefore = 20f,
                    SpacingAfter = 30f
                };

            table.SetWidths(widths);
            PdfPCell cell = new PdfPCell(new Phrase("لیست پرسنل", tahomaFont)) { RunDirection =
PdfWriter.RUN_DIRECTION_RTL, Colspan = 2, Border = 0, HorizontalAlignment = 1 };
```

```
            table.AddCell(cell);
            foreach (var item in list)
            {
                PdfPCell cell2 = new PdfPCell(new Phrase(item.PERSON_ID.ToString(), tahomaFont)) {
RunDirection = PdfWriter.RUN_DIRECTION_RTL };
                PdfPCell cell3 = new PdfPCell(new Phrase(item.FIRST_NAME + " " + item.LAST_NAME,
tahomaFont)) { RunDirection = PdfWriter.RUN_DIRECTION_RTL };
                table.AddCell(cell2);
                table.AddCell(cell3);
            }
            pdfDoc.Add(table);
        }
```

آیا امکانش هست که ما به جای اینکه بیایم در هر Cell کد زیر را بنویسیم یک بار برای کد جدول اینو تعریف کنیم؟

```
{ RunDirection = PdfWriter.RUN_DIRECTION_RTL };
```

چون اگه بخواهیم کدهای مربوط به تنظیم فوت رو برای هر Cell بنویسیم یه خورده کدها زیاد میشه.
ممنون میشم راهنمایی کنید
مرسی

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۲ ۸:۵۰

نه. لازم است به ازای هر سلول اینکار انجام شود.
ضمنا یک نکته کلی در مورد PDF وجود دارد و آن هم این است که ساختار PDF یک canvas است (یک تابلو نقاشی برداری). یعنی مفاهیمی مانند جدول، سلول، پاراگراف و غیره در پشت صحنه آن وجود خارجی ندارند و فقط کتابخانه‌های تولید PDF است که این نوع امکانات را جهت سهولت کار اختراع کرده‌اند. بنابراین به ازای هر شیءایی که اضافه می‌شود باید اطلاعات دقیق آن نیز درج شود.

نویسنده: بهاره قدمی
تاریخ: ۱۳۹۲/۰۴/۰۱ ۱۵:۵

با عرض خسته نباشید
اگر در داده‌ها ترکیبی از حروف فارسی و انگلیسی باشه با این روش کار نمیکنه.مثلا با فونت B Lotus حروف انگلیسی نمایش داده نمیشه.در بهترین حالت من فونت Tahoma رو استفاده میکنم که بازم اعدادم انگلیسی هست.
آیا اشکال از فونته؟ نمیتونم دوتا فونت براش بفرستم ؟

یا اینکه باید فونتی بسازم که همه جور حروفی رو داشته باشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۰۱ ۱۶:۲۹

« iTextSharp و استفاده از قلم‌های محدود فارسی »

نویسنده: samin
تاریخ: ۱۳۹۲/۰۹/۱۱ ۱۰:۲۸

باسلام
ممنون از راهنمایی تون عالی بود
اما برای راست چین کردن متون باید چیکار کرد؟
من این دو خط رو اضافه کردم

```
cell.RunDirection = PdfWriter.RUN_DIRECTION_RTL;
cell.RunDirection = PdfWriter.DirectionR2L;
```

اما باز هم کار نمیکنه !

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۱۱ ۱۰:۴۰

- به تصویر آخر و کدهای آن دقت کنید. کلمه آزمایش از سمت راست شروع شده.
- DirectionR2L کاربردی ندارد در اینجا. PdfWriter.RUN_DIRECTION_RTL باید باشد.

نویسنده: رسول
تاریخ: ۱۳۹۲/۱۱/۱۵ ۱۷:۱۳

سلام؛ آیا امکانش هست که برای کلمه آزمایش بطور مثال مختصات تعریف کرد تا در اون مختصات توی صفحه نمایش داده بشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۵ ۲۰:۱۷

بله. باید از متد ColumnText.ShowTextAligned استفاده کنید:

```
ColumnText.ShowTextAligned(
                        canvas: pdfWriter.DirectContent,
                        alignment: Element.ALIGN_RIGHT,
                        phrase: new Phrase("لیست پرسنل", tahomaFont),
                        x: 40,
                        y: 30,
                        rotation: 0,
                        runDirection: PdfWriter.RUN_DIRECTION_RTL,
                        arabicOptions: 0);
```

عنوان: **روش صحیح تعریف قلم در iTextSharp**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۵/۳۱ ۱۱:۵۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: iTextSharp

روش متداول تعریف فونت در iTextSharp به صورت زیر است:

```
public static iTextSharp.text.Font Tahoma()
{
     var fontPath = Environment.GetEnvironmentVariable("SystemRoot") + "\\fonts\\tahoma.ttf";
     var baseFont = BaseFont.CreateFont(fontPath, BaseFont.IDENTITY_H, BaseFont.EMBEDDED);
     return new Font(baseFont);
}
```

از آنجائیکه خصوصا برای متون فارسی نیاز است تا به ازای هر المان کوچکی این فونت تنظیم شود و در غیر اینصورت متنی نمایش داده نخواهد شد، با سربار بالایی مواجه خواهیم شد. بنابراین به نظر می‌رسد که بهتر باشد این تولید اشیاء فونت را کش کنیم. خوشبختانه iTextSharp سیستم کش کردن تعریف قلم‌های متفاوت را هم به صورت توکار دارا است:

```
public static iTextSharp.text.Font GetTahoma()
{
    var fontName = "Tahoma";
    if (!FontFactory.IsRegistered(fontName))
    {
         var fontPath = Environment.GetEnvironmentVariable("SystemRoot") + "\\fonts\\tahoma.ttf";
         FontFactory.Register(fontPath);
    }
    return FontFactory.GetFont(fontName, BaseFont.IDENTITY_H, BaseFont.EMBEDDED);
}
```

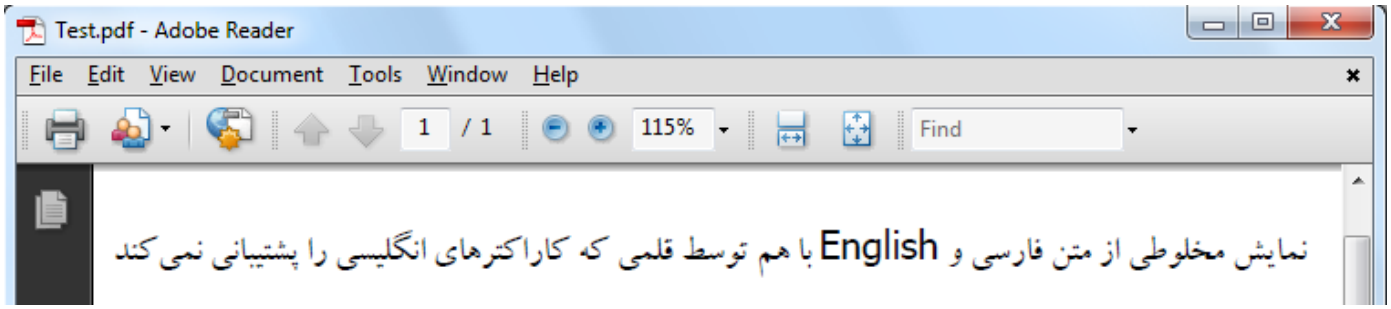
کلاس FontFactory کار ثبت و بازیابی قلم‌های متفاوت را به عهده دارد. تنها کافی است یکبار قلمی در آن ثبت شود (FontFactory.Register)، بار دیگر اطلاعات قلم به سادگی از کش FontFactory خوانده خواهد شد (FontFactory.GetFont).

عنوان: **iTextSharp و استفاده از قلم‌های محدود فارسی**
نویسنده: وحید نصیری
تاریخ: ۰۹:۲۴:۰۰ ۱۳۹۰/۰۶/۲۲
آدرس: www.dotnettips.info
برچسب‌ها: iTextSharp

عموما قلم‌های فارسی، خصوصا مواردی که با B شروع می‌شوند مانند B Zar و امثال آن، فاقد تعاریف حروف مرتبط با glyphs الفبای انگلیسی است. نتیجه این خواهد شد که اگر متن شما مخلوطی از کلمات و حروف فارسی و انگلیسی باشد، فقط قسمت فارسی نمایش داده می‌شود و از قسمت انگلیسی صرفنظر خواهد شد. مرورگرها در این حالت هوشمندانه عمل می‌کنند و به یک قلم پیش فرض مانند Times و همانند آن جهت نمایش اینگونه متون مراجعه خواهند کرد؛ اما اینجا چنین اتفاقی نخواهد افتاد.
برای حل این مشکل، کلاسی به نام FontSelector در کتابخانه‌ی iTextSharp وجود دارد. مثالی در این رابطه:

```
using System.Diagnostics;
using System.IO;
using iTextSharp.text;
using iTextSharp.text.pdf;

namespace HeadersAndFooters
{
    class Program
    {
        static void Main(string[] args)
        {
            using (var pdfDoc = new Document(PageSize.A4))
            {
                PdfWriter.GetInstance(pdfDoc, new FileStream("Test.pdf", FileMode.Create));
                pdfDoc.Open();

                FontFactory.Register("c:\\windows\\fonts\\bzar.ttf");
                Font bZar = FontFactory.GetFont("b zar", BaseFont.IDENTITY_H);

                FontFactory.Register("c:\\windows\\fonts\\tahoma.ttf");
                Font tahoma = FontFactory.GetFont("tahoma", BaseFont.IDENTITY_H);

                FontSelector fontSelector = new FontSelector();

                //قلم اصلی
                if (bZar.Familyname != "unknown")
                {
                    fontSelector.AddFont(bZar);
                }

                //قلم پیش فرض در صورت نبود تعاریف مناسب در قلم اصلی
                if (tahoma.Familyname != "unknown")
                {
                    fontSelector.AddFont(tahoma);
                }

                var table1 = new PdfPTable(1);
                table1.WidthPercentage = 100;
                table1.RunDirection = PdfWriter.RUN_DIRECTION_RTL;

                var pdfCell = new PdfPCell { RunDirection = PdfWriter.RUN_DIRECTION_RTL, Border = 0 };
                pdfCell.Phrase = fontSelector.Process("با هم توسط English نمایش مخلوطی از متن فارسی و
قلمی که کاراکترهای انگلیسی را پشتیبانی نمی‌کند");

                table1.AddCell(pdfCell);
                pdfDoc.Add(table1);

            }

            //open the final file with adobe reader for instance.
            Process.Start("Test.pdf");
        }
    }
}
```

در این مثال از قلم B Zar استفاده شده است. اولین قلمی که به یک FontSelector اضافه می‌شود، قلم اصلی خواهد بود. قلم بعدی اضافه شده، قلم پیش فرض نام خواهد گرفت؛ به این معنا که در مثال فوق اگر قلم B Zar توانایی نمایش حرف جاری را داشت که خیلی هم خوب، در غیراینصورت به قلم بعدی مراجعه خواهد کرد و همینطور الی آخر. بنابراین این ترتیب اضافه کردن قلم‌ها به FontSelector مهم است. نحوه استفاده نهایی از FontSelector تعریف شده هم در قسمت pdfCell.Phrase = fontSelector.Process مشخص است.



نظرات خوانندگان

نویسنده: Ramin
تاریخ: ۱۳۹۰/۰۶/۲۲ ۲۳:۲۸:۱۶

سلام آقای مهندس
شما برای نمایش PDF در سایت چه روشی رو پیشنهاد میدید؟ نمایش در مرورگر ، دانلود فایل و...

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۲۲ ۲۳:۴۱:۱۱

روش متداول، نصب Adobe reader در سمت کاربر است. اکتیوایکس آن سال‌ها است که با اکثر مرورگرها کار می‌کند و امکان مشاهده فایل pdf را درون خود مرورگر به صورت یکپارچه میسر کرده.
http://www.dotnettips.info/2011/07/pdf-winforms-wpf.html

نویسنده: امیر بختیاری
تاریخ: ۱۳۹۲/۰۳/۱۱ ۱۵:۸

با سلام
می‌خواستم برای حل این مشکل در RDLC چه راهی وجود داره
چون من تمام گزارشات سیستم رو با این ساختم و کلی دردسر برای ساختش کشیدم
البته یه راه حل گیر آوردم و این بود که اومدم با یه نرم افزار فونت هایی که می‌خواستم را ویرایش کردم و مثلا قسمت‌های تعاریف حروف مرتبط با glyphs الفبای انگلیسی را خودم از یه فونت دیگه مثل تایم اضافه کردم عملی هم بود ولی ساخت هر فونت با مشتقاتش 5-6 ساعت وقت میگیره
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۱۱ ۱۶:۳۴

با توجه به غیرسورس باز بودن PDF سازی که یاد کردید، بجز ویرایش فونت و افزودن دستی glyphs مفقود در آن‌ها، راه دیگری وجود ندارد. در iTextSharp برای اینکار FontSelector طراحی شده. طراحان گزارش ساز مدنظر شما هم باید چنین کاری رو انجام بدن و اضافه کنند. ضمن اینکه در iTextSharp هم اگر کسی این نکته رو ندونه، به صورت پیش فرض از FontSelector استفاده نمیشه و مدتی سردرگم خواهد بود.
در PdfReport این مسایل به صورت توکار در همه جا اعمال شده و استفاده کننده با خیلی از جزئیات و نکات ریز درگیر نخواهد شد.

نویسنده: محسن نجف زاده
تاریخ: ۱۳۹۲/۰۳/۲۱ ۸:۱۶

با سلام
چون من از HTMLWorker استفاده می‌کنم و به کمک کد زیر فونت BNazanin را بکار گرفتم

```
styles.LoadTagStyle(HtmlTags.BODY, HtmlTags.FONTFAMILY, "BNazanin");
```

لازم بود تا کلمات انگلیسی هم نمایش داده شوند در نتیجه از فونت کامل tahoma بایستی استفاده می‌کردم.اما این فونت مورد پسند نیست. در نتیجه از روش زیر (که شاید هم نا متعارف باشد) استفاده کردم

1.ابتدا استایل زیر را اضافه نمودم

```
styles.LoadStyle("english", HtmlTags.FONTFAMILY, "tahoma");
```

2.و سپس تمامی کلمات انگلیسی را به کمک کد زیر یافته و استایل english را به آن نسبت دادم

```
var cleanTagsContent = Regex.Replace(content, @"<[^>]*>", String.Empty);
            var regex = new Regex("[a-zA-Z0-9]*");
            foreach (var word in cleanTagsContent.Split(' '))
                if (regex.Match(word).Value == word && word.Length > 0)
                {
                    content = content.Replace("<" + word, "#!#");
                    content = content.Replace(word + ">", "#^#");

                    content = content.Replace(word, string.Format("<span class='english'>{0}</span>",
word));

                    content = content.Replace("#!#", "<" + word);
                    content = content.Replace("#^#", word + ">");
                }
```

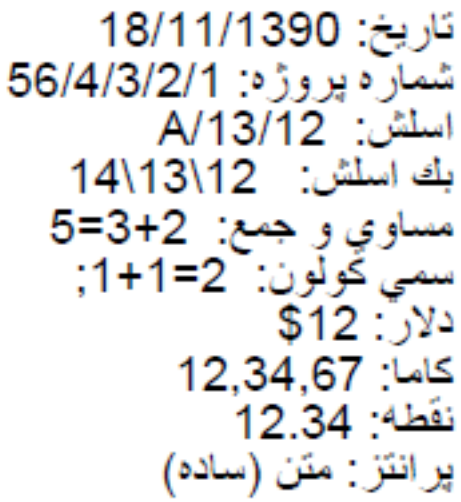
نکته : کدهای به صورت زیر را برای زمانی گذاشتیم که کلمه انگلیسی شامل td,table,div,... باشد

```
content = content.Replace("<" + word, "#!#");
```

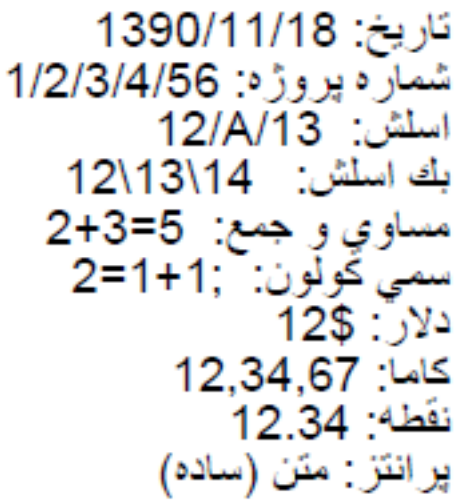
باز هم مرا به خاطر این کار نامتعارف ببخشید :)

عنوان: **iTextSharp و نمایش صحیح تاریخ در متنی راست به چپ**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۱/۱۹ ۱۹:۴۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: iTextSharp

خروجی PDF زیر را در نظر بگیرید:



مشکلی را در آن مشاهده می‌کنید؟ اصل آن یا صحیح آن باید به شکل زیر باشد:



و این وارونه نمایش دادن‌ها، دقیقا مشکلی است که حین کار با iTextSharp برای نمایش متنی مثلا به همراه یک تاریخ شمسی وجود دارد. البته این مشکل هم اساسا به خود استاندارد یونیکد برمی‌گرد که یک سری کاراکتر را « کاراکتر ضعیف » معرفی کرده؛ برای مثال کاراکتر اسلش بکار رفته در یک تاریخ هم از این دست است. بنابراین PDF تولیدی توسط iTextSharp از دید استاندارد

یونیکد مشکلی ندارد، زیرا یک «نویسه ضعیف» مثل اسلش نمی‌تواند جهت را تغییر دهد؛ مگر اینکه از یک « نویسه قوی » برای دستکاری آن استفاده شود. برای مثال این نویسه‌ها قوی هستند:

```
U+202A:   LEFT-TO-RIGHT EMBEDDING (LRE)
U+202B:   RIGHT-TO-LEFT EMBEDDING (RLE)
U+202D:   LEFT-TO-RIGHT OVERRIDE (LRO)
U+202E:   RIGHT-TO-LEFT OVERRIDE (RLO)
U+202C:   POP DIRECTIONAL FORMATTING (PDF)
```

برای رسیدن به تصویر صحیح نمایش داده شده در بالا، متد FixWeakCharacters زیر را تهیه کرده‌ام که حداقل با iTextSharp جواب می‌ده:

```
using System;
using System.Diagnostics;
using System.IO;
using iTextSharp.text;
using iTextSharp.text.pdf;

namespace RleTests
{
    class Program
    {
        const char RightToLeftEmbedding = (char)0x202B;
        const char PopDirectionalFormatting = (char)0x202C;

        static string FixWeakCharacters(string data)
        {
            if (string.IsNullOrWhiteSpace(data)) return string.Empty;
            var weakCharacters = new[] { @"\", "/", "+", "-", "=", ";", "$" };
            foreach (var weakCharacter in weakCharacters)
            {
                data = data.Replace(weakCharacter, RightToLeftEmbedding + weakCharacter +
PopDirectionalFormatting);
            }
            return data;
        }

        static void Main(string[] args)
        {
            using (var pdfDoc = new Document(PageSize.A4))
            {
                PdfWriter.GetInstance(pdfDoc, new FileStream("Test.pdf", FileMode.Create));
                pdfDoc.Open();

                FontFactory.Register("c:\\windows\\fonts\\Arial.ttf");
                Font tahoma = FontFactory.GetFont("Arial", BaseFont.IDENTITY_H);

                var table1 = new PdfPTable(1);
                table1.WidthPercentage = 100;

                var pdfCell = new PdfPCell
                {
                    RunDirection = PdfWriter.RUN_DIRECTION_RTL,
                    Border = 0,
                    Phrase = new Phrase(FixWeakCharacters(
                        "18/11/1390" + " :تاریخ" + Environment.NewLine +
                        "56/4/3/2/1" + " :شماره پروژه" + Environment.NewLine +
                        "12 " + " :اسلش/A/13 " + Environment.NewLine +
                        "14\\13\\12  " + " بک اسلش: " + Environment.NewLine +
                        "5=3+2 " + " مساوی و جمع: " + Environment.NewLine +
                        "1+1=2 " + " ;سمی کولون: " + Environment.NewLine +
                        "12" + " :دلار$" + Environment.NewLine +
                        "12,34,67" + " :کاما" + Environment.NewLine +
                        "12.34" + " :نقطه" + Environment.NewLine +
                        "متن (ساده)" + " :پرانتز)"
                        ),
                        tahoma)
                };

                table1.AddCell(pdfCell);
                pdfDoc.Add(table1);
```

```
            }
            Process.Start("Test.pdf");
        }
    }
}
```

از این نوع مشکلات حین کار با HTML هم هست؛ وارونه نمایش داده شدن تاریخ فارسی در بین یک متن راست به چپ. البته در آنجا راه حل زیر هم توصیه شده (بدون نیاز به دستکاری نویسه‌ها):

```
<span dir="ltr" style="display:inline">1390/11/19</span>
```

نظرات خوانندگان

نویسنده: فرهاد یزدان پناه
تاریخ: ۱۳۹۰/۱۱/۱۹ ۲۱:۱۶:۴۲

ممنون. جناب نصیری.
آقای حاجلو هم مطلبه بسیار مفیدی در همین موضوع دارند.
/http://hajloo.wordpress.com/2009/03/02/persian-text-problem-in-ltr-forms

نویسنده: linux
تاریخ: ۱۳۹۰/۱۱/۲۲ ۲۳:۱۹:۴۵

برای جدا سازی اجزای تاریخ شمسی، ماه ، روز و سال نباید از / استفاده کرد در unicode برای این کار از Unicode Character ('ARABIC DATE SEPARATOR' (U+060D استفاده کنید برای دیدن جزییات بیشتر
http://www.fileformat.info/info/unicode/char/60d/index.htm

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۱/۲۳ ۰۰:۰۰:۴۲

شاید در زبان عربی اینطور باشه. حتما می‌دونید که نحوه نمایش و نویسه‌های اعداد 4 و 6 عربی و فارسی یکی نیست. ک و ی عربی و فارسی هم یکی نیست. حتی ممیز فارسی هم شیوه خاص خودش را دارد و کلا بحث من اینجا در مورد نحوه متداول ورود اطلاعات در زبان فارسی است؛ در مورد هزاران هزار سطر موجود. ضمن اینکه اگر به مثال دقت کرده باشید یک شماره پروژه‌ای هم این وسط هست که الگویی شبیه به تاریخ ندارد؛ به علاوه یک سری نویسه ضعیف دیگر مثل مساوی و جمع و منها و غیره. به علاوه بحث من در مورد کتابخانه تولید PDF ذکر شده است و راه حلی که با آن جواب بدهد.
راه حل بالایی که من مطرح کردم در نمایش هیچ تغییری ایجاد نمی‌کنه. این حرف بکارگرفته شده، نامرئی هستند. PDF هم یک لایه Presentation است. بنابراین زمانیکه اطلاعاتی را درست نمایش می‌دهد، یعنی هدف اصلی خودش را برآورده کرده.

نویسنده: Nasser Mansouri
تاریخ: ۱۳۹۰/۱۲/۰۸ ۱۵:۲۹:۴۷

سلام، می خواستم اگر ممکن باشه من رو راهنمایی کنید، من می خوام با استفاده از itextsharp محتوای یک فایل پی دی اف رو به صورت txt ذخیره کنم، با زبانهای چپ به راست خیلی آسون هست ولی در زبان راست به چپ، کلمات از اخر به اول نوشته می شن مثلا کلمه ارزیابی به صورت "یبایزرا" خوانده می شه، ممنون می شم من رو راهنمایی بکنید.

```
()private static string ParsPDFToString
}
;("PdfReader reader = new PdfReader("c:/2v.pdf
;(PdfReaderContentParser parser = new PdfReaderContentParser(reader
;()StringBuilder sb = new StringBuilder
;(((Console.WriteLine("reader.NumberOfPages : " + reader.NumberOfPages.ToString
;()Console.ReadKey
(++for (int i = 1; i <= reader.NumberOfPages; i
}
)ITextExtractionStrategy strategy = parser.ProcessContent
()i , new SimpleTextExtractionStrategy
;(
;(((sb.Append(strategy.GetResultantText
{
;()return sb.ToString
```

{

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۲/۰۸ ۲۰:۰۶:۵۱

بستگی داره نرم افزار تولید PDF از چه روشی استفاده کرده باشه. بعضی‌ها از چرخاندن حروف استفاده می‌کنند و این روش بسیار متداولی هست. یعنی مشکل از iTextSharp نیست. در اصل به همین ترتیب حروف ذخیره شدن. الگوریتم اولیه به همین صورت بوده.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۲/۰۹ ۱۲:۲۹:۱۸

دو مورد تکمیلی:
- کار این چرخاندن‌ها توسط دو کلاس ArabicLigaturizer و BidiLine در iTextSharp انجام می‌شود. سورس کتابخانه را دریافت و این دو کلاس را مطالعه کنید (ضمن اینکه PDF های فارسی هم وجود دارند که اصلا با این الگویتم‌ها تهیه نشده‌اند و خلاصه راه سختی را پیش رو دارید).iTextSharp انجمنی نداره ولی یک mailing list فعال داره:
https://lists.sourceforge.net/lists/listinfo/itext-questions

عنوان: **تبدیل HTML به PDF با استفاده از کتابخانه‌ی iTextSharp**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۱۲ ۰۹:۵۲:۰۰
آدرس: www.dotnettips.info
گروه‌ها: iTextSharp

روش متداول کار با کتابخانه‌ی iTextSharp ، ایجاد شیء Document ، سپس ایجاد PdfWriter برای نوشتن در آن، گشودن سند و افزودن اشیایی مانند PdfPCell ، PdfPTable ، Paragraph و غیره به آن است و در نهایت بستن سند. راه میانبری هم برای کار با این کتابخانه وجود دارد و آن هم استفاده از امکانات فضای نام iTextSharp.text.html.simpleparser آن می‌باشد. به این ترتیب می‌توان به صورت خودکار، یک محتوای HTML را تبدیل به فایل PDF کرد. ...

مثال : نمایش یک متن HTML ساده انگلیسی

```
using System.Diagnostics;
using System.IO;
using iTextSharp.text;
using iTextSharp.text.html.simpleparser;
using iTextSharp.text.pdf;

namespace HeadersAndFooters
{
    class Program
    {
        static void Main(string[] args)
        {
            using (var pdfDoc = new Document(PageSize.A4))
            {
                PdfWriter.GetInstance(pdfDoc, new FileStream("Test.pdf", FileMode.Create));
                pdfDoc.Open();

                var html = @"<span style='color:blue'><b>Testing</b></span>
                             <i>iTextSharp's</i> <u>HTML to PDF capabilities</u>";
                var parsedHtmlElements = HTMLWorker.ParseToList(new StringReader(html), null);

                foreach (var htmlElement in parsedHtmlElements)
                {
                    pdfDoc.Add(htmlElement);
                }
            }

            //open the final file with adobe reader for instance.
            Process.Start("Test.pdf");
```

```
        }
    }
}
```

نکته‌ی جدید کد فوق، استفاده از متد HTMLWorker.ParseToList است. به این ترتیب parser کتابخانه‌ی iTextSharp وارد عمل شده و html تعریف شده را به معادل المان‌های بومی خودش تبدیل می‌کند؛ مثلا تبدیل به chunk یا pdfptable و امثال آن. در نهایت در طی یک حلقه، این عناصر به صفحه اضافه می‌شوند.
البته باید دقت داشت که HTMLWorker امکان تبدیل عناصر پیچیده، تودرتو و چندلایه HTML را ندارد؛ اما بهتر از هیچی است!

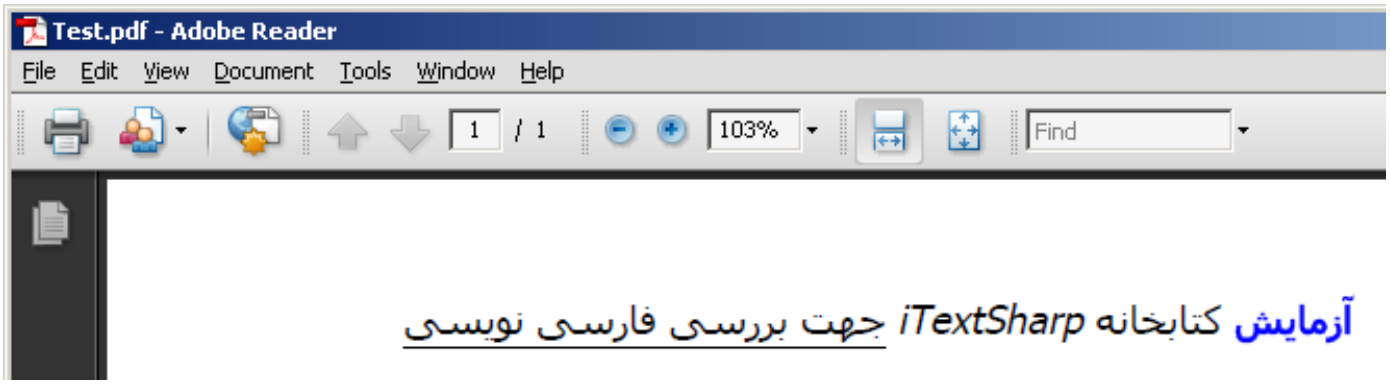
همه‌ی این‌ها خوب! اما به درد ما فارسی زبان‌ها نمی‌خورد. همین متغیر html فوق را با یک متن فارسی جایگزین کنید، چیزی نمایش داده نخواهد شد. البته این هم نکته دارد که در ادامه ذکر خواهد شد.
جهت نمایش متون فارسی نیاز است تا نکات ذکر شده در مطلب « فارسی نویسی و iTextSharp » رعایت شوند که شامل:
- تعیین صریح قلم
- تعیین encoding
- استفاده از عناصر دربرگیرنده‌ای است که خاصیت RunDirection را پشتیبانی می‌کنند؛ مانند PdfPCell و غیره

به این ترتیب خواهیم داشت:

```
using System.Diagnostics;
using System.IO;
using iTextSharp.text;
using iTextSharp.text.html.simpleparser;
using iTextSharp.text.pdf;
using iTextSharp.text.html;

namespace HeadersAndFooters
{
    class Program
    {
        static void Main(string[] args)
        {
            using (var pdfDoc = new Document(PageSize.A4))
            {
                PdfWriter.GetInstance(pdfDoc, new FileStream("Test.pdf", FileMode.Create));
                pdfDoc.Open();

                روش صحیح تعریف فونت//
                FontFactory.Register("c:\\windows\\fonts\\tahoma.ttf");

                StyleSheet styles = new StyleSheet();
                styles.LoadTagStyle(HtmlTags.BODY, HtmlTags.FONTFAMILY, "tahoma");
```

```
                styles.LoadTagStyle(HtmlTags.BODY, HtmlTags.ENCODING, "Identity-H");

                var html = @"<span style='color:blue'><b>آزمایش</b></span>
                                کتابخانه <i>iTextSharp</i> <u>جهت بررسی فارسی نویسی</u>";
                var parsedHtmlElements = HTMLWorker.ParseToList(new StringReader(html), styles);

                PdfPCell pdfCell = new PdfPCell { Border = 0 };
                pdfCell.RunDirection = PdfWriter.RUN_DIRECTION_RTL;

                foreach (var htmlElement in parsedHtmlElements)
                {
                    pdfCell.AddElement(htmlElement);
                }

                var table1 = new PdfPTable(1);
                table1.AddCell(pdfCell);
                pdfDoc.Add(table1);
            }

            //open the final file with adobe reader for instance.
            Process.Start("Test.pdf");
        }
    }
}
```

همانطور که ملاحظه می‌کنید ابتدا قلمی در cache قلم‌های این کتابخانه ثبت می‌شود (FontFactory.Register). سپس نوع قلم و encoding آن توسط یک StyleSheet تعریف شده و به HTMLWorker.ParseToList ارسال می‌گردد و در نهایت به کمک یک المان دارای RunDirection، در صفحه نمایش داده می‌شود.



نکته:
ممکن است که به متغیر html ، یک table ساده html را نسبت دهید. در این حالت پس از تنظیم style یاد شده، در هر سلول این html table ، متون فارسی به صورت معکوس نمایش داده خواهند شد که این هم یک نکته‌ی کوچک دیگر دارد:

```
foreach (var htmlElement in parsedHtmlElements)
                {
                    if (htmlElement is PdfPTable)
                    {
                        var table = (PdfPTable)htmlElement;
                        table.RunDirection = PdfWriter.RUN_DIRECTION_RTL;
                        foreach (var row in table.Rows)
                        {
                            foreach (var cell in row.GetCells())
                            {
                                cell.RunDirection = PdfWriter.RUN_DIRECTION_RTL;
                            }
                        }
                    }

                    pdfCell.AddElement(htmlElement);
                }
```

در قسمتی که قرار است المان‌های معادل به pdfCell اضافه شوند، آن‌ها را بررسی کرده و RunDirection آن‌ها را RTL خواهیم کرد.

کاربردها:
بدیهی است این حالت برای تهیه گزارشات پیشرفته‌تر برای مثال تهیه قالب‌هایی که در حین تهیه PDF ، قسمت‌هایی از آن‌ها توسط برنامه نویس Replace می‌شوند، بسیار مناسب است.
همچنین مطلب « بارگذاری یک یوزرکنترل با استفاده از جی‌کوئری » و متد RenderUserControl مطرح شده در آن که در نهایت یک قطعه کد HTML را به صورت رشته به ما تحویل می‌دهد، می‌تواند جهت تهیه گزارش‌های پویایی که برای مثال قسمتی از آن یک GridView بایند شده حاصل از یک یوزر کنترل است، مورد استفاده قرار گیرد.

نظرات خوانندگان

نویسنده: A.Karimi
تاریخ: ۱۳۹۰/۰۶/۱۳ ۱۶:۴۳:۲۶

بسیار عالی!

نویسنده: Hossein Hariri
تاریخ: ۱۳۹۰/۰۶/۲۱ ۱۲:۱۲:۵۸

با سلام و تشکر
من از روش شما برای تولید pdf استفاده کردم. ولی متاسفانه align متنهای ایجاد شده Left است و هر کاری میکنم درست نمیشود. لطفا راهنمایی بفرمایید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۲۱ ۱۵:۲۵:۲۱

برای اینکار نیاز است تا style تعریف شده را کمی تغییر داد. به صورت زیر:

```
(styles.LoadTagStyle(HtmlTags.BODY, HtmlTags.ALIGN, HtmlTags.ALIGN_LEFT
```

نویسنده: Alisfard
تاریخ: ۱۳۹۰/۰۶/۲۲ ۲۳:۴۰:۳۸

لطفا میشه نمونه کد را برای دانلود قرار دهید چون من هر چی سعی کردم نتونستم اجرا کنم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۶/۲۲ ۲۳:۵۴:۰۷

از اینجا قابل دریافت است: [^]

نویسنده: hamidnch2007
تاریخ: ۱۳۹۰/۰۷/۲۹ ۰۰:۳۵:۰۲

من همین روش رو برای گریدویو تودرتو که داخل یه div هستند و آن div بصورت runat=server میباشد و آن را با استفاده از متد RenderControl داخل یه String Writer ریختم و اونو بعنوان پارامتر StreamReader پاس دادم.نتیجه این شد که فیلدهای گرید بیرونی درست نمایش داده شدند ولی گریدویو داخلی متنهای آن بصورت برعکس و نچسبیده نمایان شدندممنون یشم راهنمایی کنید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۷/۲۹ ۰۸:۴۰:۳۴

parsedHtmlElements تولیدی را دیباگ و سپس نکته آخری رو که در متن بالا عنوان شد باید اعمال کنید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۷/۲۹ ۱۳:۳۶:۰۶

چند نکته عمومی:
- اینجا انجمن نیست. مشکلات عمومی خودتون رو در انجمن‌ها پیگیری کنید.
- اگر خواستید جایی کدی طولانی را ارسال کنید حداقل از سایت http://pastebin.com استفاده کنید.
- در مورد این مثال جاری، تا دسترسی به html نهایی تولیدی شما نباشد، دیباگ کردن آن بی‌معنا است.

نویسنده: hamidnch2007
تاریخ: ۱۳۹۰/۰۷/۲۹ ۱۵:۰۳:۵۰

بابت اشتباهم عذرخواهی میکنم.
من فقط میخواستم اگر برایتان امکان پذیر است یه مثال کوچک از یه html پارس شده که شامل گریدویو تودرتو هست برایم بزنید و روال را بگویید.جواب شما خیلی کلی بود و بنده متوجه نشدم.باز هم ببخشید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۷/۲۹ ۲۱:۲۰:۴۶

با گرید تو در تو (جداول تو در تو) هم کار می‌کنه؛ فقط اینبار باید اون حلقه‌ای رو که به عنوان نکته گفتم، تبدیل به یک تابع بازگشتی کنید. به این صورت: http://pastebin.com/zpMsPmMa

نویسنده: hamidnch2007
تاریخ: ۱۳۹۰/۰۷/۲۹ ۲۱:۳۳:۳۳

خدا یک در دنیا و هزار در آخرت بهت بده.الهی خیر ببینی.کارت درسته.آقای نصیری.
یه سوال دیگه بکنم.اگه من تو گریدویو چک باکس داشته باشم اون رو هم میشه به pdf ارسال کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۷/۲۹ ۲۲:۰۴:۵۰

تعداد تگ‌هایی که iTextSharp ساپورت می‌کنه کم هست. لیست این‌ها رو می‌تونید در کلاس HtmlTags فضای نام iTextSharp.text.html مشاهده کنید.

یک توصیه کلی:
اگر به دنبال یک راه حل حرفه‌ای برای کارهای پیچیده‌تر HTML to PDF هستید، باید سراغ این نوع کتابخانه‌ها بروید:
(wkhtmltopdf ,Convert html to pdf using webkit (qtwebkit
برای مثال این مورد از WebKit یا همان موتور گوگل کروم استفاده می‌کند. بنابراین HTML parser آن مانند iTextSharp محدود نیست و فوق العاده حرفه‌ای است.

نویسنده: hamidnch2007
تاریخ: ۱۳۹۰/۰۷/۲۹ ۲۳:۱۲:۳۷

از خدا برای شما بهترین ها را آرزو میکنم.امشب مشکل من را حل کردید.امیدوارم که ثمره آن را در زندگی تان ببینی.اگر روزی بدانم میتوانم برایتان کاری انجام بدم و در توانم باشد دریغ نخواهم کرد.همیشه پیروز باشید.

<> Disqus 2011/10/21

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۷/۲۹ ۲۳:۳۹:۰۶

ممنون. سلامت باشید.

نویسنده: mkpro
تاریخ: ۱۳۹۱/۰۷/۲۵ ۱۳:۲۸

سلام خسته نباشید
من جدیداً سایتتون رو کشف کردم سایت خیلی مفید و پر باری دارید.
من یه مشکل توی تبدیل html به pdf داشتم وقتی تعداد صفحاتم بیشتر از 2 بشه بهم اخطاره Object reference not set to an

instance of an object. و این اخطاره مربوط میشه به pdfdoc.add(Table)
ممنون میشم یه راه حلی ارائه بدید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۲۵ ۱۳:۴۰

HTML worker مطرح شده در این مطلب، مدتی است که از طرف نویسندگان آن منسوخ شده اعلام گردیده و دیگری پشتیبانی یا نگهداری نخواهد شد. بنابراین اگر باگی وجود دارد یا هر مطلبی، کسی دیگر به آن رسیدگی نخواهد کرد.
راه حل جایگزین، استفاده از XML Worker است که بجای HTML worker در حال کار و توسعه می‌باشد .

نویسنده: جواد جعفری
تاریخ: ۱۳۹۱/۰۸/۰۴ ۱۴:۸

بابا دمت گرم خیلی خیلی ممنونم واقعا به دردم خورد

نویسنده: مرتضی موسوی
تاریخ: ۱۳۹۲/۰۴/۱۰ ۱۱:۵۸

آقای مهندس , میتونید راهنمایی کنید من چطور میتونم قابلیت Save a Copy as رو غیر فعال کنم . بعبارتی نباید کاربر بتونه از فایل کپی بگیره .

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۱۰ ۱۲:۱۵

مراجعه کنید به مطلب « رمزنگاری فایل‌های PDF با استفاده از کلید عمومی توسط iTextSharp ». در اینجا توسط مقادیری مانند PdfWriter.ALLOW_COPY و غیره می‌شود روی فایل تولیدی محدودیت ایجاد کرد. ضمنا راه برای برطرف کردن این محدودیت‌ها هم هست .

نویسنده: راد
تاریخ: ۱۳۹۲/۰۸/۰۴ ۱۰:۵۷

باسلام
امکان داره نمونه کد این مثال گذاشته شود
باتشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۰۴ ۱۳:۲۲

- نمونه کد همان مثالی هست که در متن آورده شده. برای اجرا تنها نیاز به کتابخانه iTextSharp دارد. (یک برنامه کنسول ساده را ایجاد کنید. کدهای مثال مطلب فوق را در آن paste کنید و بعد ارجاعی را به اسمبلی iTextSharp به آن اضافه نمائید)
- ضمنا افزونه HTMLWorker این کتابخانه منسوخ شده (مطلب جاری) و به XMLWorker ارتقاء یافته .

نویسنده: فیضی
تاریخ: ۱۳۹۲/۱۱/۱۵ ۹:۴۲

سلام.
من کدهای بالا رو استفاده کردم تا محتویات یک فیلد از جدول که محتویاتش از ادیتور Tinymce پر میشه رو به pdf تبدیل کنم.
در ادیتور عکس هم ممکنه درج بشه.

مشکلی که وجود داره اینه که عکسهایی که در ادیتور در سمت راست قرار داده شدن در pdf در سمت چپ قرار می‌گیرن. این مشکل رو چطور میشه رفع کرد؟
نمونه رو می‌تونید از لینک زیر دانلود کنید.

pdf
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۵ ۹:۵۴

- پردازش CSS کتابخانه HTMLWorker خیلی ضعیف و ابتدایی است. به همین جهت آن‌را کنار گذاشته‌اند و به XMLWorker کوچ کرده‌اند (HTMLWorker هیچ پشتیبانی رسمی دیگر ندارد ؛ به قسمت Deprecated. please switch to XML Worker instead آن دقت کنید). ضمنا HTMLWorker مشکلات دیگری هم دارد. مثلا یک تگ hr در صفحه باشد، کرش می‌کند. پردازش ویژگی‌های مختلف CSS و HTML تقریبا در آن پیاده سازی نشده و ...
- برای کار با ADO.NET بهتر است این روزها از Micro ORMs استفاده کنید.

نویسنده: فیضی
تاریخ: ۱۳۹۲/۱۱/۱۸ ۰:۳

ممنون از پاسخ. بله، من xml worker ای که شما توضیح داده بودید رو هم تست کرده بودم. html ای که شما پاس داده بودید رو به خوبی نشون می‌داد ولی من با همون داده‌های پست قبلی مثال شما رو تست کردم. حروف انگلیسی درست نشون داده میشن ولی فارسی‌ها به شکل نقطه دیده میشن عکسها هم سمت چپ مشاهده می‌شوند
اینم لینک نمونه تستی
Sample
یا شاید من اشتباه فهمیدم که xmlworker اون قدر قوی هست که عناصر html رو می‌تونه با همون استایلی که دارن نشون بده؟
ممنون.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۸ ۰:۳۱

- در مورد فارسی نویسی در iTextSharp یک دیباگ مرحله به مرحله قبلا در سایت مطرح شده. اگر خروجی یونیکد نگرفتید یعنی قلم صحیحی در حال استفاده نیست. کدهایی که قبلا ارسال کرده بودم به این نحو است:

```
// روش صحیح تعریف فونت
var systemRoot = Environment.GetEnvironmentVariable("SystemRoot");
FontFactory.Register(Path.Combine(systemRoot, "fonts\\tahoma.ttf"));
```

در کدهای شما به این نحو:

```
var systemRoot = Environment.GetEnvironmentVariable("SystemRoot");
FontFactory.Register(Path.Combine(systemRoot, "c:\\windows\\fonts\\tahoma.ttf"));
```

با توجه به استفاده از Path.Combine، مسیری را که معرفی کرده‌اید می‌شود چیزی مانند c:\\windows\\c:\\windows\\fonts\\tahoma.ttf . به همین جهت این فونت یافت نشده و ثبت نمی‌شود (چون دوبار system root در آن وجود دارد).
- بله؛ قدرت پردازش CSS در XML Worker آن خیلی بهتر است از HTML Worker.
- در مورد میزان چرخش جدول، RunDirection = PdfWriter.RUN_DIRECTION_RTL را با حالت LTR هم تست کنید (PdfWriter.RUN_DIRECTION_LTR).

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)
```

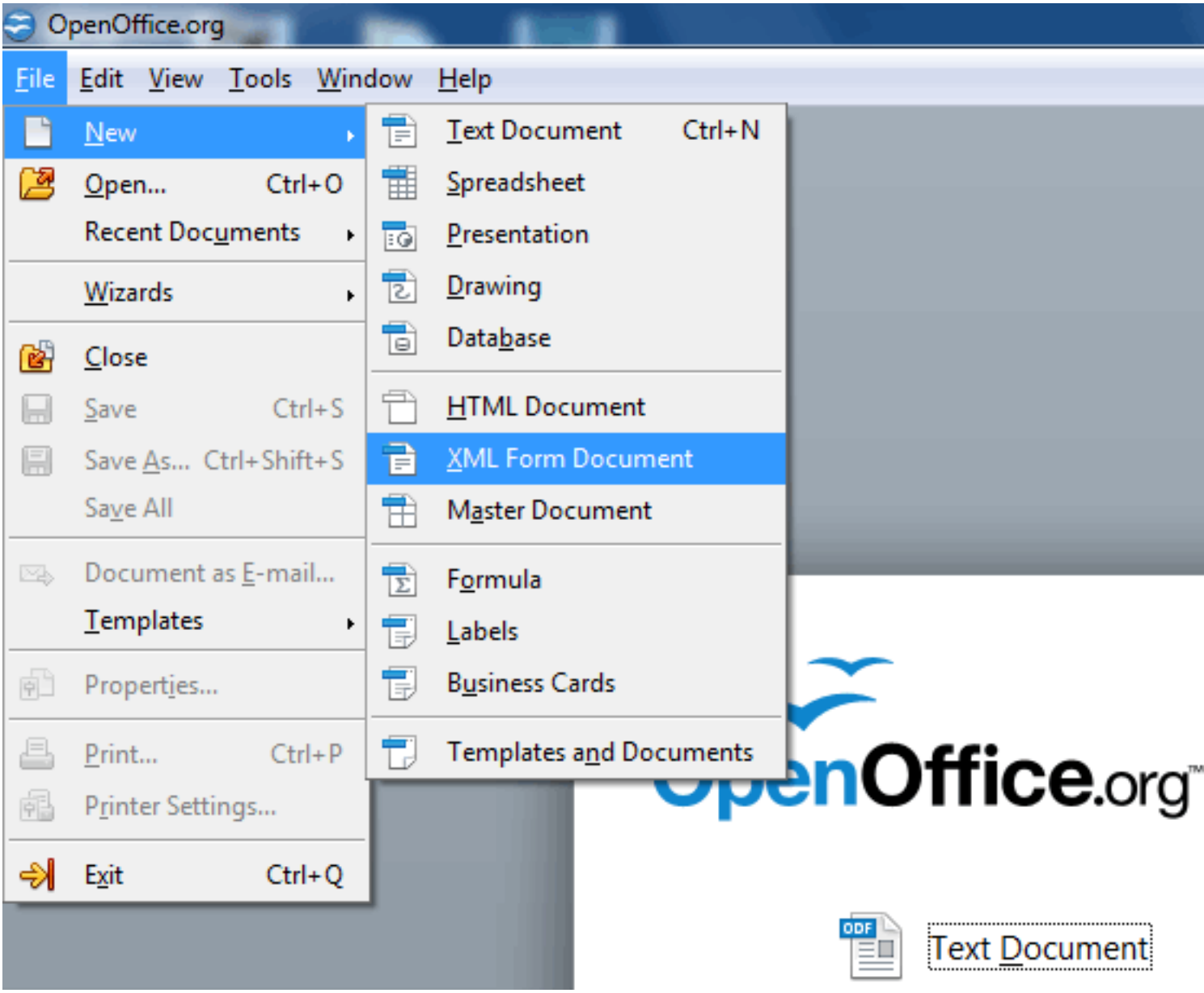
```
        {
            var writableElement = htmlElement as WritableElement;
            if (writableElement == null)
                return;

            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)
```

عنوان: **ساخت یک گزارش ساز به کمک iTextSharp و Open Office**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۹/۰۱ ۰۰:۱۹:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: iTextSharp

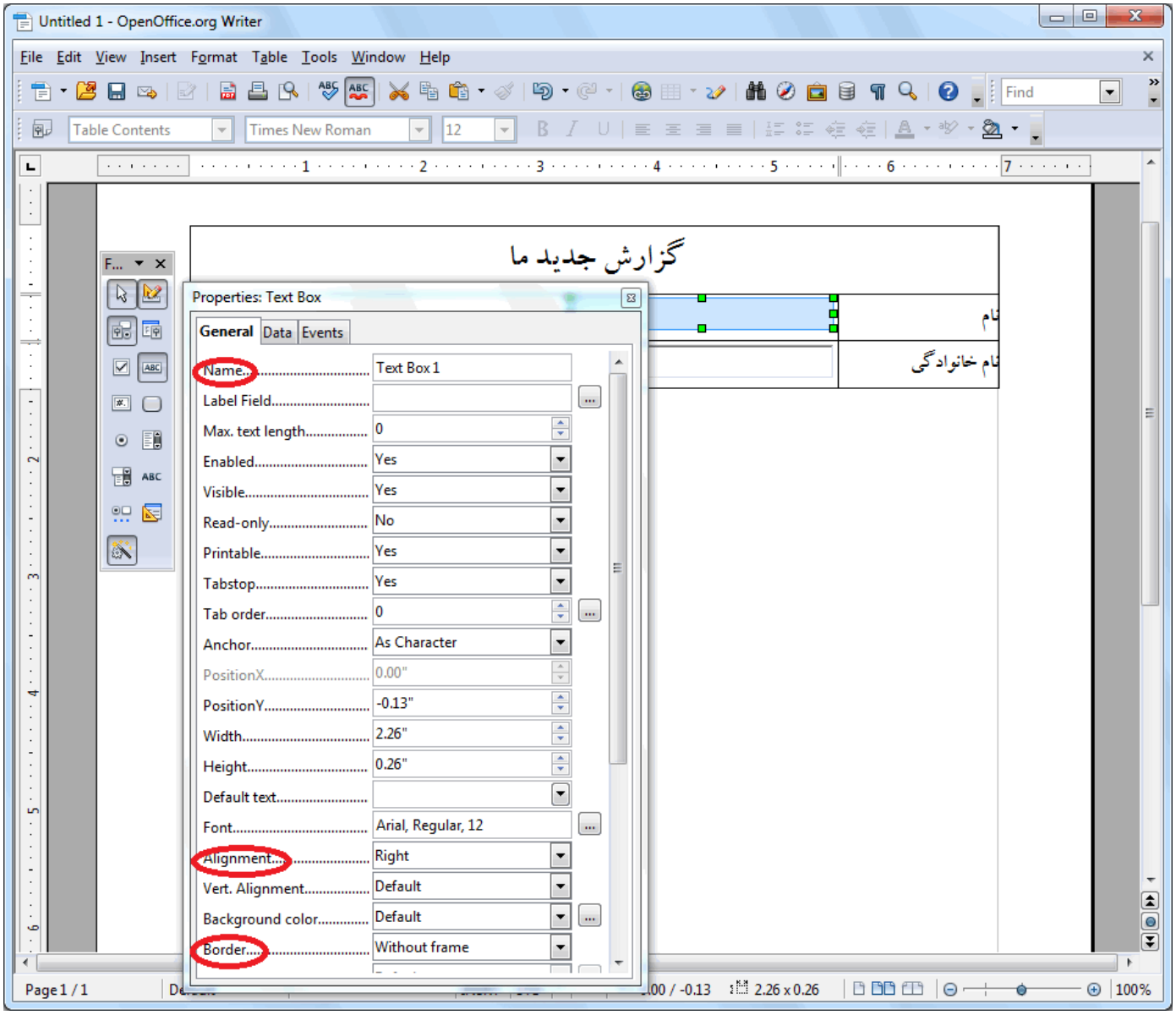
iTextSharp پایه کار با فایل‌های PDF را ارائه می‌دهد اما ابزاری را جهت ساده‌تر سازی تولید فایل‌های PDF به همراه ندارد؛ هر چند مثلا امکان تبدیل HTML به PDF را دارا است اما باید گفت: «تا حدودی البته». اگر نیاز باشد جدولی را ایجاد کنیم باید کد نویسی کرد، اگر نیاز باشد تصویری اضافه شود به همین ترتیب و الی آخر. البته این را هم باید در نظر داشت که کد نویسی انعطاف قابل توجهی را در اختیار برنامه نویس قرار می‌دهد؛ شاید به همین دلیل این روزها مباحث «Code first» بیشتر مورد توجه برنامه نویس‌ها است، تا مباحث «Wizard first» یک دهه قبل!
اما باز هم داشتن یک طراح بد نیست و می‌تواند در کاهش مدت زمان تولید نهایی یک فایل PDF مؤثر باشد. برای این منظور می‌توان از برنامه‌ی رایگان و معروف Open office استفاده کرد. توسط آن می‌توان یک فرم PDF را طراحی و سپس فیلدهای آن‌را (این قالب تهیه شده را) با iTextSharp پر کرد. این مورد می‌تواند برای تهیه گزارش‌هایی که تهیه آن‌ها با ابزارهای متداول گزارش سازی عموما میسر نیست، بسیار مناسب باشد.

طراحی یک فرم PDF با استفاده از برنامه Open Office

آخرین نگارش برنامه Open office را از اینجا می‌توانید دریافت کنید و آنچنان حجمی هم ندارد؛ حدودا 154 مگابایت است.
پس از نصب و اجرای برنامه، حداقل به دو طریق می‌توان یک فرم جدید را شروع کرد:
الف) آغاز یک XML Form document جدید در Open office سبب خواهد شد که نوارهای ابزار طراحی فرم، مانند قرار دادن CheckBox ، TextBox و غیره به صورت خودکار ظاهر شوند.
ب) و یا آغاز یک سند معمولی و سپس مراجعه به منوی View->Toolbars->Form Controls هم همان حالت را به همراه خواهد داشت.

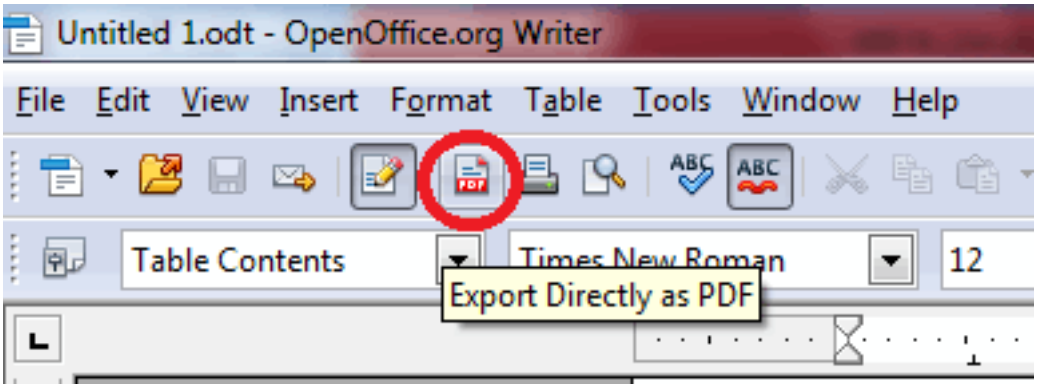


در اینجا برای طراحی یک گزارش یا فرم جدید تنها کافی است همانند روش‌های متداول تهیه یک سند معمولی رفتار کنیم و مواردی را که قرار است توسط iTextSharp مقدار دهی کنیم، با کنترل‌های نوار ابزار Form آن بر روی صفحه قرار دهیم که نمونه‌ی ساده آن‌را در شکل زیر ملاحظه می‌کنید:



برای گزارش‌های فارسی بهتر است Alignment یک کنترل به Right تنظیم شود و Border به حالت Without frame مقدار دهی گردد. نام این کنترل را هم بخاطر بسپارید و یا تغییر دهید. از این نام‌ها در iTextSharp استفاده خواهیم کرد. (صفحه خواص فوق با دوبار کلیک بر روی یک کنترل قرار گرفته بر روی فرم ظاهر می‌شود)

مرحله بعد، تبدیل این فرم به فایل PDF است. کلیک بر روی دکمه تهیه خروجی به صورت PDF در نوار ابزار اصلی آن برای اینکار کفایت می‌کند. این گزینه در منوی File نیز موجود است.



فرم‌های PDF تهیه شده در اینجا، فقط خواندنی هستند. مثلا یک کاربر می‌تواند آن‌ها را پر کرده و چاپ کند. اما ما از آن‌ها در ادامه به عنوان قالب گزارشات استفاده خواهیم کرد. بنابراین جهت ویرایش فرم‌های تهیه شده بهتر است فایل‌های اصلی Open Office مرتبط را نیز درجایی نگهداری کرد و هر بار پس از ویرایش، نیاز است تا خروجی جدید PDF آن‌ها تهیه شود.

استفاده از iTextSharp جهت مقدار دهی فیلدهای یک فرم PDF

در ادامه می‌خواهیم این قالب گزارشی را که تهیه کردیم با کمک iTextSharp پر کرده و یک فایل PDF جدید تهیه کنیم. سورس کامل اینکار را در ذیل مشاهده می‌کنید:

```
using System;
using System.Diagnostics;
using System.IO;
using iTextSharp.text;
using iTextSharp.text.pdf;

namespace PdfForm
{
    class Program
    {
        //روش صحیح ثبت و معرفی فونت در این کتابخانه
        public static iTextSharp.text.Font GetTahoma()
        {
            var fontName = "Tahoma";
            if (!FontFactory.IsRegistered(fontName))
            {
                var fontPath = Environment.GetEnvironmentVariable("SystemRoot") +
"\\fonts\\tahoma.ttf";
                FontFactory.Register(fontPath);
            }
            return FontFactory.GetFont(fontName, BaseFont.IDENTITY_H, BaseFont.EMBEDDED);
        }

        static void Main(string[] args)
        {
            string fileNameExisting = @"form.pdf";
            string fileNameNew = @"newform.pdf";

            using (var existingFileStream = new FileStream(fileNameExisting, FileMode.Open))
            using (var newFileStream = new FileStream(fileNameNew, FileMode.Create))
            {
                var pdfReader = new PdfReader(existingFileStream);
                using (var stamper = new PdfStamper(pdfReader, newFileStream))
                {
                    //نکته مهم جهت کار با اطلاعات فارسی
                    //در غیراینصورت شاهد ثبت اطلاعات نخواهید بود
                    stamper.AcroFields.AddSubstitutionFont(GetTahoma().BaseFont);

                    //form.Fields.Keys = تمام فیلدهای موجود در فرم
                    var form = stamper.AcroFields;

                    //مقدار دهی فیلدهای فرم
                    form.SetField("TextBox1", "مقدار1");
```

```
                    form.SetField("TextBox2", "مقدار2");

                    // "Yes" and "Off" are valid values here
                    form.SetField("Check Box 1", "Yes");

                    // "" and "Off" are valid values here
                    form.SetField("Option Button 1", "");

                    // نحوه مقدار دهی لیست
                    form.SetListOption("ListBox1", new[] { "مقدار یک", "مقدار دو11" }, null);
                    form.SetField("ListBox1", null);

                    // به این ترتیب فرم دیگر توسط کاربر قابل ویرایش نخواهد بود
                    //stamper.PartialFormFlattening --> جهت غیرقابل ویرایش نمودن فیلدی مشخص
                    stamper.FormFlattening = true;

                    stamper.Close();
                    pdfReader.Close();
                }
            }

            Process.Start("newform.pdf");
        }
    }
}
```

توضیحات:

چون در اینجا فایل PDF، از پیش تهیه شده است، پس باید از اشیاء PdfReader و PdfStamper جهت خواندن و نوشتن اطلاعات در آن‌ها استفاده کرد. سپس توسط شیء stamper.AcroFields می‌توان به این فیلدها یا همان کنترل‌هایی که در برنامه‌ی Open office بر روی فرم قرار دادیم، دسترسی پیدا کنیم.

در ابتدا نیاز است فونت این فیلدها توسط متد AddSubstitutionFont مقدار دهی شود. این مورد برای گزارش‌های فارسی الزامی است؛ در غیراینصورت متنی را در خروجی مشاهده نخواهید کرد.

ادامه کار هم مشخص است. توسط متد form.SetField مقداری را به کنترل‌های قرار گرفته بر روی فرم نسبت می‌دهیم. آرگومان اول آن نام کنترل است و آرگومان دوم، مقدار مورد نظر می‌باشد. اگر کنترل CheckBox را بر روی صفحه قرار دادید، تنها مقدارهای Yes و Off را می‌پذیرد (آن هم با توجه به اینکه به کوچکی و بزرگی حروف حساس است). اگر یک Radio button یا در اینجا Option button را بر روی فرم قرار دادید، تنها مقدارهای خالی و Off را قبول خواهد کرد. نحوه‌ی مقدار دهی یک لیست هم در اینجا ذکر شده است.

در پایان چون نمی‌خواهیم کاربر نهایی قادر به ویرایش اطلاعات باشد، FormFlattening را true خواهیم کرد و به این ترتیب، کنترل‌ها فقط خواندنی خواهند شد. البته اگر همانطور که ذکر شد، border کنترل‌ها را در حین طراحی حذف کنید، PDF نهایی تولیدی یکپارچه و یک دست به نظر می‌رسد و اصلا مشخص نخواهد بود که این فایل پیشتر یک فرم قابل پر کردن بوده است.

نظرات خوانندگان

نویسنده: باربد
تاریخ: ۱۳۹۱/۰۷/۱۰ ۱۰:۸

سلام
آیا میشه از طریق همین ادیتور ، پارامتر تصویری هم به فایل اضافه کرد؟
(مثلا عکس شخص ...)
- یک مشکل : وقتی Open office را دانلود میکنم ، موقع اجرا پیغام میده که مشکل داره و کامل دانلود نشده ، در صورتیکه اینجوری نیست . (حجم فایلش هم 130 MB هست)
سپاسگزارم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۱۰ ۱۰:۵۶

- در مورد اجرا نشدن برنامه نصاب نظری ندارم. عموما این فایل‌ها دارای یک امضای دیجیتال md5 یا sha1 منتشر شده در سایت اصلی هم هستند. مقایسه کنید آیا کامل دریافت شده یا نه.
- در مورد تصویر، می‌تونید از روش‌های متداول iTextSharp استفاده کنید. PDF در اصل یک قالب برداری است. شما یک Canvas دارید که می‌تونید در هر جایی از آن هر شیءایی را قرار دهید. برای نمونه در مثال فوق:

```
PdfContentByte content = stamper.GetOverContent(pdfReader.NumberOfPages);
Image image = Image.GetInstance(imagePath);
image.SetAbsolutePosition(450,650);
image.ScaleAbsolute(200,200);
content.AddImage(image);
```

شما به کمک stamper دسترسی به این Canvas پیدا می‌کنید. سپس در هر مختصات دلخواهی مطابق کدهای فوق، تصویر مورد نظر را قرار دهید.

نویسنده: باربد
تاریخ: ۱۳۹۱/۰۷/۱۰ ۱۱:۳۷

سپاس آقای نصیری

نویسنده: M.B
تاریخ: ۱۳۹۱/۰۷/۲۸ ۱۰:۵۱

با سلام، من زمانی که کدهای مربوطه را می‌نویسم و فرم رو اجرا می‌کنم بلافاصله یک فایل PDF برای من باز میشه که در اون کلمه آزمایش نوشته شده است کدهای من به صورت زیر هستند

```
public static Font GetTahoma()
    {
        var fontName = "Tahoma";
        if (!FontFactory.IsRegistered(fontName))
        {
            var fontPath = Environment.GetEnvironmentVariable("SystemRoot") + "\\fonts\\tahoma.ttf";
            FontFactory.Register(fontPath);
        }
        return FontFactory.GetFont(fontName, BaseFont.IDENTITY_H, BaseFont.EMBEDDED);
    }
```

```
string fileNameExisting = @"Test.pdf";
        string fileNameNew = @"newform.pdf";
         using (var existingFileStream = new FileStream(fileNameExisting, FileMode.Open))
            using (var newFileStream = new FileStream(fileNameNew, FileMode.Create))
```

```
{
    var pdfReader = new PdfReader(existingFileStream);
    using (var stamper = new PdfStamper(pdfReader, newFileStream))
    {
        //نکته مهم جهت کار با اطلاعات فارسی
        //در غیراینصورت شاهد ثبت اطلاعات نخواهید بود
        stamper.AcroFields.AddSubstitutionFont(GetTahoma().BaseFont);

        //form.Fields.Keys = تمام فیلدهای موجود در فرم
        var form = stamper.AcroFields;

        //مقدار دهی فیلدهای فرم
        form.SetField("Text1", "مقدار1");
         form.SetField("Text2", "مقدار2");
         form.SetField("Text3", "مقدار3");
         form.SetField("Text4", "مقدار4");
         form.SetField("Text5", "مقدار5");
         form.SetField("Text6", "مقدار6");



        // به این ترتیب فرم دیگر توسط کاربر قابل ویرایش نخواهد بود
        //stamper.PartialFormFlattening --> جهت غیرقابل ویرایش نمودن فیلدی مشخص
        stamper.FormFlattening = true;

        stamper.Close();
        pdfReader.Close();
    }
}

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

و محتوای فایل Test.PDF

شماره پرسنلی		شماره شناسنامه	
نام و نام خانوادگی		کد ملی	
نام پدر		مقطع	

و محتوای فایل جدید که برای من ایجاد می‌کند

آزمایش

ممنون .

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۲۸ ۱۱:۲۱

در سیستم شما تداخل وجود دارد. کلمه «آزمایش» متعلق به فایل قبلی دیگری است که دارید.
فایل‌های آزمایشی خروجی PDF موجود را کلا حذف کنید و بعد کدهای فوق را اجرا کنید.
Process.Start را هم حذف کرده و خروجی را دستی و خارج از VS.NET بررسی کنید.

نویسنده: پویا امینی
تاریخ: ۱۳۹۱/۰۷/۲۸ ۱۱:۴۰

ببخشید آقای نصیری من وقتی Process.Start رو حذف کنم کجا می‌تونمم محتوای فایل NewForm.PDF رو ببینم؟ چون درون Sloution چنین فایلی برای من ایجاد نمی‌شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۲۸ ۱۱:۵۳

معمولا درون پوشه bin\debug یا bin\release تشکیل می‌شود.
اما جهت اطمینان می‌تونید مسیر دهی کامل کنید:

```
string fileNameExisting = @"c:\Test.pdf";
string fileNameNew = @"c:\newform.pdf";
```

نویسنده: پویا امینی
تاریخ: ۱۳۹۱/۰۷/۲۹ ۰:۳۲

خیلی کارت درسته جناب نصیری، من اومدم مسیر دهی خودم رو به صورت زیر انجام دادم

```
string fileNameExisting =HttpRuntime.AppDomainAppPath+ @"TestOpenOffice.pdf";
 string fileNameNew =HttpRuntime.AppDomainAppPath+ @"newform.pdf";
```

و Process.Start

```
Process.Start(HttpRuntime.AppDomainAppPath+"newform.pdf");
```

یه دنیا ممنون جناب نصیری

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۲۹ ۰:۴۵

اگر برنامه ویندوزی است بهتره از Application.StartupPath استفاده کنید (تعریف شده در اسمبلی System.Windows.Forms).
اگر برنامه وب است، از Server.MapPath استفاده کنید.