عنوان: **تهیه پردازنده‌های سفارشی برای افزونه XMLWorker کتابخانه iTextSharp**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۲۴ ۱۳:۵
آدرس: www.dotnettips.info
برچسب‌ها: iTextSharp

پیشتر مطلب « تهیه پردازنده‌های سفارشی برای HTMLWorker کتابخانه iTextSharp » را در این سایت مطالعه کرده‌اید. از آنجائیکه افزونه HTMLWorker منسوخ شده است و دیگر پشتیبانی نخواهد شد، باید کدهای فعلی را به افزونه XMLWorker منتقل کرد. مقدمه‌ای را در این زمینه در مطلب « تبدیل HTML فارسی به PDF با استفاده از افزونه‌ی XMLWorker کتابخانه‌ی iTextSharp » می‌توانید مطالعه نمائید.
در ادامه قصد داریم همان امکان پشتیبانی از تصاویر base64 مدفون شده در صفحات HTML را به کتابخانه XMLWorker نیز اضافه کنیم.

تهیه پردازنده‌های سفارشی تگ‌های HTML جهت افزونه XMLWorker

در اینجا کار با ارث بری از کلاس AbstractTagProcessor شروع می‌شود. سپس کافی است تا متد End این کلاس پایه را override کرده و تگ سفارشی خود را پردازش کنیم. نمونه‌هایی از این نوع پردازنده‌ها را در پوشه itextsharp.xmlworker\iTextSharp\tool\xml\html سورس‌های iTextSharp می‌توانید ملاحظه کنید و جهت ایده گرفتن بسیار مناسب می‌باشند.
یکی از این پردازنده‌های پیش فرض موجود در افزونه XMLWorker کار پردازش تگ‌های img را انجام می‌دهد و در کلاس iTextSharp.tool.xml.html.Image قرار گرفته است. این پردازنده به صورت پیش فرض تصاویر base64 را پردازش نمی‌کند. برای رفع این مشکل می‌توان به نحو ذیل عمل کرد:

```
public class CustomImageTagProcessor : iTextSharp.tool.xml.html.Image
    {
        public override IList<IElement> End(IWorkerContext ctx, Tag tag, IList<IElement>
currentContent)
        {
            IDictionary<string, string> attributes = tag.Attributes;
            string src;
            if (!attributes.TryGetValue(HTML.Attribute.SRC, out src))
                return new List<IElement>(1);

            if (string.IsNullOrEmpty(src))
                return new List<IElement>(1);

            if (src.StartsWith("data:image/", StringComparison.InvariantCultureIgnoreCase))
            {
                // data:[<MIME-type>][;charset=<encoding>][;base64],<data>
                var base64Data = src.Substring(src.IndexOf(",") + 1);
                var imagedata = Convert.FromBase64String(base64Data);
                var image = iTextSharp.text.Image.GetInstance(imagedata);

                var list = new List<IElement>();
                var htmlPipelineContext = GetHtmlPipelineContext(ctx);
                list.Add(GetCssAppliers().Apply(new
Chunk((iTextSharp.text.Image)GetCssAppliers().Apply(image, tag, htmlPipelineContext), 0, 0, true), tag,
htmlPipelineContext));
                return list;
            }
            else
            {
                return base.End(ctx, tag, currentContent);
            }
        }
    }
```

با ارث بری از کلاس پردازنده پیش فرض تگ‌های تصاویر یا iTextSharp.tool.xml.html.Image شروع و سپس متد End آن‌را تحریف کرده‌ایم.
در اینجا اگر src یک تگ img با الگوی تصاویر base64 شروع شده باشد، تصویر آن استخراج و تبدیل به وهله‌ای از تصاویر استاندارد iTextSharp می‌شود. سپس این تصویر در اختیار htmlPipelineContext قرار داده خواهد شد و یا اگر این تصویر از نوع base64 نباشد، همان متد base.End کلاس پایه، جهت پردازش آن کفایت می‌کند.

استفاده از یک پردازنده تگ سفارشی HTML افزونه XMLWorker

تا اینجا یک پردازنده جدید تصاویر را ایجاد کرده‌ایم. برای اینکه XMLWorker را از وجود آن مطلع کنیم، نیاز است آن‌را به درون htmlPipeline این افزونه تزریق نمائیم که کدهای کامل آن‌را در ذیل مشاهده می‌کنید:

```
using (var doc = new Document(PageSize.A4))
            {
                var writer = PdfWriter.GetInstance(doc, new FileStream("test.pdf", FileMode.Create));
                doc.Open();
                var html = @"<img
src='
U29mdHdhcmUAQWRvYmUgSW1hZ2VSZWFkeXHJZTwAAAGAUExURdSmeJp2SHlbQIRoSUg2J499a8KebqeHZuGufBEVJPz7+3NWPVxGMdu
whPXEktnX1mtROLq7t5WDc2VMNv3LmKB8TMSidMbFxLGlmXlhSMSddpJUL+y8i3VlVqedlOzr6gUIF2lXRLCLY4ZyXLyYaYhtUYiJhJ
FyU1dBLLiVZnlwZrWRY/Hx8b+2rbySaJh9YqeooDw4NygnKvvJlpyblzksIUhGRryYckc7MPjGlKODX5x8VVA8K+azgM3FvDInHK2JW
2ZbUOHh4Xt2cFpaWKeAUM6kel1RRJmUjo5vSrWzrJJ1WFhLQCQmMuK1iJiMgmthWPPCkOm3hEtBOunm5LCNXnJtZquEXmNkYvG+i7Ct
q+y5hrWRbKqSeaN/WqmFVYFgQh8aGOa4isWkd8mcby4vONDNy0AwI5h2U19JMxkdLzIuL1JBMjQ3P5Z6Ve6/j93c2+Xi34KAfJ5/Xvj
4+O/u7sSKVJd4Wo6QjXE+IeOwfQcNJoBeQ8Gdbf/Mmf///5GX6NEAAAcrSURBVHja3JbpX9pIGMchiWkgEaOBtaGinBLEyopFBeMqtY
KI4kGt2lILFsUoXa3WdZcc/dd3JheHAvaz7/Z5Ec2Q7/yeaw7Lz/9klv8rfnM+Orz5cXLjZsL+67h9eCq9Vaxvzc6v3W6+/TX85kN6i
xdokkQQCaE5vrg28Qv4a2yFQcpSi/HzH6efi+/UaEAwWAtepuvv3tw/B//hqZGQqDFSmyHC7v0z8EldlZQQEgTfMgF23h8/T+gEhQGr
cQYrMBKVtvfDb4qU/j3DMK3SdIKWsNs++M1iS8R8W/gULyG1771w+/stQWpTpFpzByb09MRHEwaoxUxToGtaZiBrE72cXzMyhcDiIRg
CHxJPIxKt5aF23gMf0iquz8BJmAAFpUStxvG0xIA3arcHPsvrJM1wvFTDeEGQeKCewCo1jgRDwKuJrrh9C3osIfyiz+NboZFKxU0xJE
YmeJbBhPoKiKyMDXfHd0mJWSETnoKiKCmgSioFDKFr4T1lbn/fgkHf+PGu+A+A12imMqdAqzNUXlFCFP+gOD41CKJBcCB4bKSnOmitB
5VWSgnMrSjhCnu8D1hoS1xP/KcH1BhZdGi4c4VNAh/I5PGyRjdQqje+A6YXPIpup/DhHlMUh44f1hAJ6x77z3OwVjG/0ml7Ot4gOWnx
vkfbALw+2EnPGc43ojWk3qNt7hdpiSp0ajcMukHQPB/4o3vPf8TKQgc+pqXdkpEtgGewE7THel/j66dtdBLA1XAYRXK8AGbxC/6RHvj
bCuOE0Kklk8lcg/+OicaJcOhfTflTVYCHuYvX3XH7QCxcUAol9i6VursLha+VfcLPHwamZjfSAgxi6QId6oFnC5awsjdoWYjFPrOlB3
QONAtJjrwsetiq2jkzgfc9nPdklJBDyXvGj+Zf+jIKe7pPoNFoOHwyoyaQKFcD9z3wzbwSGnT6fCMB9u5UmWMLYwTJQo5QC2AB6r122
ukBJeVWnA6HIwlLnp/bI/w5wI3tJR3LjcZMbvVzL/xHwOG+M6s2mFeSjRm0QRyDYnyCOEv/0fOYGM/vha4N3J1S5hoZhCAcYBro/AwV
63NIjafuzL4rLSjOZYKeIT45j9XUnQTs/Y7Inbqp/pABeIPBqsTystr0/pd9T9jprZIGO9CHa4gTPHairxr/eP/rwai+YdzlWQfALSH
u4qTxfHxiQKVTaBINvfCjDFo1Fmzjor/zP+0BNXdgxSTdqRe5w0bT2hq+293mdWDOSJ5DWbgwd4uGpSPxXW5WGzGddhYWHsDRguqpO5
x9jjq4HY3BnjtcRRGGe/Xqn38YC6SraVt84jnXwo0FgC8kOK7s+mv91St6RhVnZ72Vqeln4EM+cFY43SHgdj584c9ormdFbx3Jbk73v
9PuvNCCvx67ntPzlmG2xUvUhQpZz9roxHdwXx4e7Yb/fdXc7o81PFcUxW2ry+Wy5miM4gQkEAh0uxKfXWbdLXs1XGxZURRnXZpZrVbX
egT/rUvm571itnncQPctWZso2hAdd61GIzIuf32y5zduL0VxtwQPWG2vB7QP0OKKVaejOI7L8lP4+S3r+wY+zSZfGPvGPlFlt8FQ3BC
PQPYpfOjWs3QHtMVLJqmU0NLe9XVhsBpOwyER0+D1oE534t8Hsn/KctwLokxUgeunD6FwCA2xMGtAPAdhjkr55afwoaksGpHlAKTnWU
K9ZIAt15k/U+mK5voSuoI9Vre/fZPOBcFQKg4+PXsXg7urVra0Stvqmud4mTp4hN/s+lAIy8ErIC7Oz8aITzqegYkUL4tawQ+ivEvud
P7Gt6SPpCpewJ8BfN+pb/aq71dG2kjayLuJ3/vC+gB+EBe9Xm/8KEQs67hShMmgIRsNylFuFe9UL1IGHXHNAtr77ZYN7htNB8LxJmCn
yaBZULpJ6/g4ZZQCX83FAS1u3675xnTaX/GKFdLl+gIaDZeFpU78rS9oDnzZEmHstqPJKc9n90LJPThyBUZIVRtMv8Q1v9Xx8bzxigd
dWo1t7yZ//zgSCwRiK6CO0PUD2OR4hMnhHfiPtYiJr4a8Jj4MbHNe7UC4RtTfc5wsd+DD6RbxxTZ8chtkrcJGIlqX41GqTVzFp3wmfm
CNi5rNT74Z3nwHi2BjZW11AtdzgvxIfSBl4l/Klzr+bfLvzSNYA1u9xTfmz8f4lLmA5HWfgV8eTa7BEohxox1xeZ1F5Ef4fTrYnL4oG
jb7QZ3JVgk2W4KJPMZvmWbo9KWJ27QsXKHm3DkhJT/Gs6z55lo0abV5wCSL5txL/CMa4PYPUXN+5qwTj68aXwa5MP4Efj/VDA4TW3BV
3PQMp7Wlgnfg555mcPFO8RbXMbXv8Oh6pG3J7IRM8bq3Q/zKLFqUQ3GteNYvbepG1XG57O0Qt9Hmd1bOKC1qbZH/zbK78FWzYMJ2aZo
XPq7kr8ZvORr+iUSjJzQb/Gpa5l8BBgBZTppAyfsf0wAAAABJRU5ErkJggg==' width='62' height='80' style='float:
left; margin-right: 28px;' />";

                var tagProcessors = (DefaultTagProcessorFactory)Tags.GetHtmlTagProcessorFactory();
                tagProcessors.RemoveProcessor(HTML.Tag.IMG); // remove the default processor
                tagProcessors.AddProcessor(HTML.Tag.IMG, new CustomImageTagProcessor()); // use our new
processor

                var cssResolver = new StyleAttrCSSResolver();
                cssResolver.AddCss(@"code { padding: 2px 4px; }", "utf-8", true);
                var charset = Encoding.UTF8;
                var hpc = new HtmlPipelineContext(new CssAppliersImpl(new XMLWorkerFontProvider()));
                hpc.SetAcceptUnknown(true).AutoBookmark(true).SetTagFactory(tagProcessors); // inject
the tagProcessors
                var htmlPipeline = new HtmlPipeline(hpc, new PdfWriterPipeline(doc, writer));
                var pipeline = new CssResolverPipeline(cssResolver, htmlPipeline);
                var worker = new XMLWorker(pipeline, true);
                var xmlParser = new XMLParser(true, worker, charset);
                xmlParser.Parse(new StringReader(html));
            }
            Process.Start("test.pdf");
```

در اینجا ابتدا لیست پردازنده‌های پیش فرض افزونه XMLWorker را دریافت و سپس پردازنده تگ img آن‌را حذف و با نمونه جدید خود جایگزین کرده‌ایم. در ادامه این لیست تغییر یافته به درون HtmlPipelineContext تزریق شده‌است تا بجای DefaultTagProcessorFactory اصلی مورد استفاده قرار گیرد.

نظرات خوانندگان

نویسنده: ایمان دارابی
تاریخ: ۱۳۹۲/۰۷/۲۷ ۱۵:۱

با سلام؛ معمولا تگ عکس حاوی مسیر فایل عکس در صورتی که در فایل pdf عکس معمولا embed می‌شه دو تا راه حل به نظر من می‌رسه یکی دانلود عکس‌ها و قرار دادن آنها در پوشه تمپ که پردازنده فوق باید اونو هندل کنه که البته برای افزایش سرعت باید از درخواست‌های غیر هم زمان استفاده کرد چون ممکنه مجبور به دانلود چندین عکس در یک رشته html باشیم و راه حل دوم عدم ذخیره فایل عکس و تبدیل آن به رشته که در مثال شما دیده می‌شه شما کدوم روش را توصیه می‌کنید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۲۷ ۱۵:۱۴

تمام این روش‌ها پشتیبانی می‌شوند. اگر src تصویر
1- مسیر لوکال هست، در مطلب مقدماتی استفاده از XMLWorker از کلاس ImageProvider تهیه شده استفاده کنید.
2- URL و مسیر وب است، خود iTextSharp به صورت خودکار آن‌را دانلود می‌کند.
3- base64 است، از راه حل مطلب جاری استفاده کنید.
از لحاظ سرعت کار، 3 سریعترین است؛ بعد 1 و در آخر 2.

نهایتا در هر سه حالت، عکس در فایل PDF مدفون می‌شود و نیازی به تنظیم خاصی ندارد.