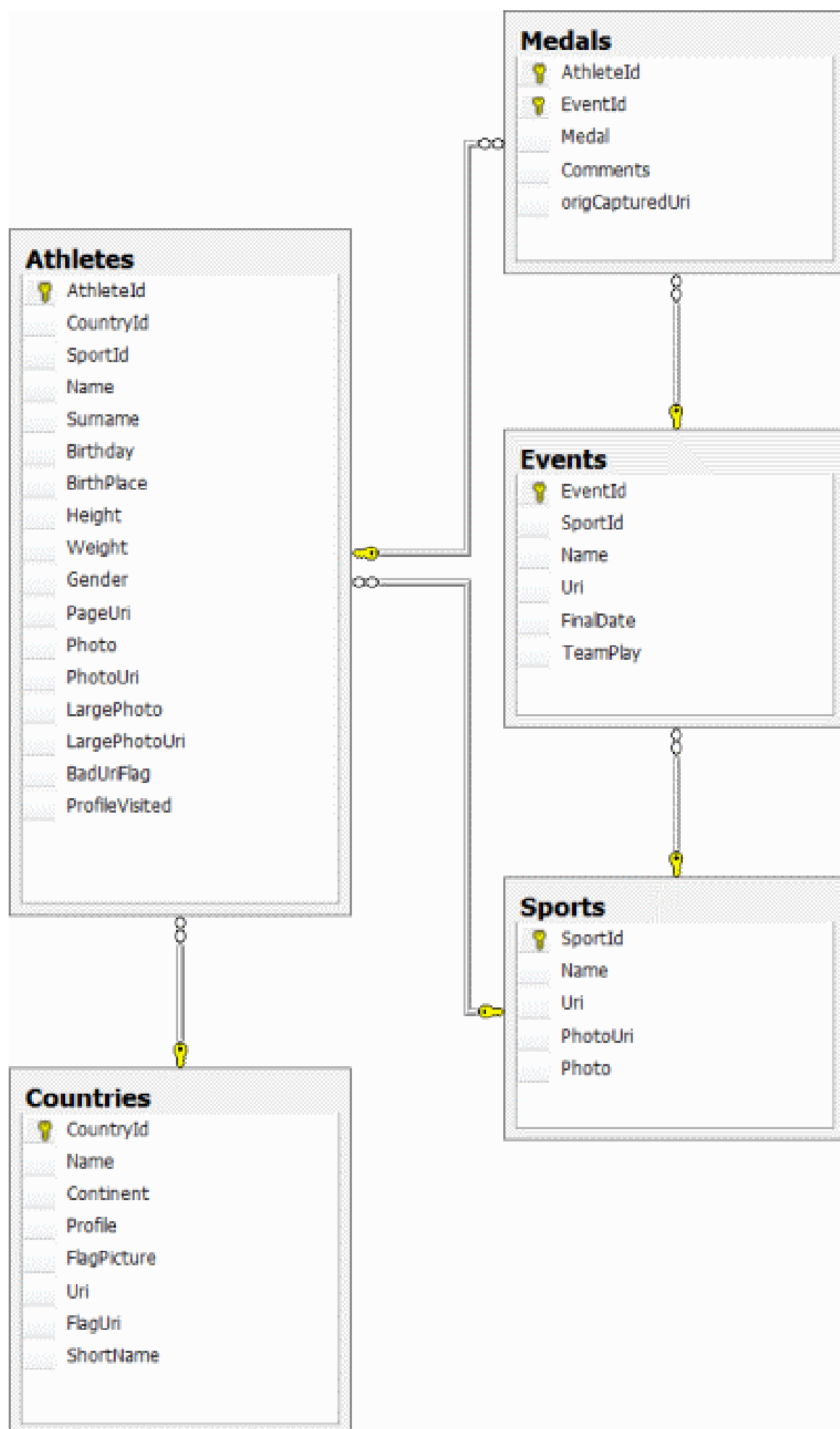


چند مدت پیش موقعی که تب المپیک بود و جدول <http://www.london2012.com/medals/medal-count> رو زیاد نگاه می‌کردم به نظرم رسید که کاشکی به اطلاعاتی مثل اینکه چند نفر از مدال آورها خانم و یا آقا هستند و یا اینکه در روزهای مختلف تعداد مدال‌ها چطور توزیع می‌شند و بشه با یک jQuery UI Slider روزهای مختلف رو انتخاب کرد و جدول رو دید. برای این کار اولین چیزی که لازم بود دریافت و ذخیره اطلاعات بود که من برای این کار از Entity framework 4.1 Database- first و کتابخانه HAP - [htmlagilitypack](http://htmlagilitypack.com) استفاده کردم. طراحی دیتابیس نهایی به این صورت شد



خوب در تلاش اول و مبتدیانه و بدون استفاده از این کتابخانه مفید چون اکثر صفحات وب XHTML نیستند و بالاخره چند تگ درست بسته نشده دارند و شما اگر بخواهید در آبجکت XmlDocument این htmlهای به ظاهر سالم رو لود کنید فوراً با استثنای زیر مواجه می‌شوید

```
XmlException Was unhandled  
The 'img' start tag on line 1 position 1604 does not match the end tag of 'a'. Line 1, position 1766
```

راه حل ساده اینه که این کتابخونه رو با کمک NuGet نصب کنید

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



و از اینجا به بعد با کدی مثل این میتونید از کلاس XmlDocument و مشابه XmlDocument ولی بدون ارور استفاده کنید. مثلاً با کد زیر میشه تاریخ تولد یک ورزشکار رو بدست آورد. توابع دیگه ای که خیلی جاها میتونه بدرد خورد GetAttributeValue و ChildNodes هست که یک نمونه نحوه استفادشو در ادامه میبینید


```
HtmlDocument xhtml = Crawler.GetXHtmlFromUri("http://www.london2012.com/athlete/hadadi-ehsan-1077408/");  
HtmlNode tempNode = xhtml.DocumentNode.SelectSingleNode("//table[@class='athleteBio']/tbody/tr[4]");  
  
string temp = tempNode.FirstChild.FirstChild.InnerText.Replace("&nbsp;", "").Trim();  
athlete.Birthday = DateTime.Parse(temp.Substring(0, 10), new CultureInfo("en-GB"));  
  
tempNode = xhtml.DocumentNode.SelectSingleNode("//div[@class='athletePhotoMedals']/div/div/img");  
athlete.LargePhotoUri = tempNode.GetAttributeValue("src", "");
```

البته تابع GetXHtmlFromUri رو جدا باید با کمک HttpRequest بنویسید و توی خود HAP متاسفانه چنین تابعی توکار نشده نکته اصلی هم پیدا کردن محل دقیق اطلاعاته که با ابزاری مثل Firebug خیلی راحت تر میشه این کارو انجام داد. کافیه روی تاریخ تولد راست کلیک و inspect element by Firebug رو بزنید و حالا اگر تویه dom روی هر المنت html نگه دارید بهتون XPath کامل رو میده که میتونید تویه تابع DocumentNode.SelectSingleNode ارزش استفاده کنید.

Ehsan Hadadi

[Overview](#) | [Events and results](#) |

Country		
 Islamic Republic of Iran		
Birth date	Age	
20/01/1985 - Tehran (IRI)	27	
Height	Weight	Gender
192 cm / 6'4"	110 kg / 243 lbs	M
Sport		
 Athletics Men's Discus Throw		


0 1 0

```
<table class="athleteBio">
  <tbody>
    <tr>
      <td colspan="2">
        </html/body/div/div/div[4]/div[6]/div/div/div/div/div/div/table/tbody/tr[4]>
    <tr>
```

برای درک بهتر XPath هم این 2 تا صفحه `xpath_syntax` و `xpath_examples` خیلی میتونه کمکتون بکنه.

نظرات خوانندگان

نویسنده: مهدی پایروند
تاریخ: ۱۳:۵۳ ۱۳۹۱/۰۶/۰۴

این روش برای استخراج مطالب همین سایت خیلی مفیده!

نویسنده: محسن کریمی
تاریخ: ۱۳:۵۷ ۱۳۹۱/۰۶/۰۴

با تشکر از مطلبتون
نکته: این Htmlagilitypack متاسفانه با صفحات فارسی و UTF8 مشکلات زیادی داره و واقعا همیشه ارزش استفاده کرد.

نویسنده: جمال
تاریخ: ۱۴:۱۹ ۱۳۹۱/۰۶/۰۴

Crawler.GetXHtmlFromUri
شی Crawler از چه نوعیه؟ موقع اجرا خطا میگیره؟

نویسنده: وحید نصیری
تاریخ: ۱۴:۲۸ ۱۳۹۱/۰۶/۰۴

شروع کار به این صورت هم می‌تواند باشد:

```
var doc = new HtmlDocument
{
    OptionCheckSyntax = true,
    OptionFixNestedTags = true,
    OptionAutoCloseOnEnd = true,
    OptionDefaultStreamEncoding = Encoding.UTF8
};
doc.LoadHtml(content);
```

OptionDefaultStreamEncoding رو به UTF8 تنظیم کنید.

نویسنده: محسن کریمی
تاریخ: ۱۵:۴۳ ۱۳۹۱/۰۶/۰۴

با تشکر

ولی طبق تجربه خود من کد بالا هم کمک نمی‌کنه و با تنظیم OptionDefaultStreamEncoding به UTF8 مشکل حل نمیشه ولی یه راه که قبلا من پیدا کرده بودم تو خوندن کد صفحات اینه که شما صفحات رو به صورت استریم دریافت کرده بعد توسط متد LoadHtml بخونید به این صورت مشکل برطرف میشه! (البته این تو سایت فارسی که من قصد خوندشو داشتم، بود با سایتهای دیگه تست نکردم)

```
var request = (HttpWebRequest)WebRequest.Create("آدرس سایت");
request.Method = "GET";
using (var response = (HttpWebResponse)request.GetResponse())
{
    using (var stream = response.GetResponseStream())
    {
        htmlDoc.Load(stream, Encoding.UTF8);
    }
}
```

نویسنده: وحید نصیری
تاریخ: ۱۶:۴۹ ۱۳۹۱/۰۶/۰۴

بستگی داره content نظر قبلی رو به چه فرمتی (چه Encoding ایی) از وب دریافت کردید. مابقی آن توسط این کتابخانه بدون مشکل پردازش می‌شود.

```
using System.Net;
//...
var content = new WebClient { Encoding = Encoding.UTF8 }.DownloadString(url);
```

نویسنده: ایمان عبیدی
تاریخ: ۲۰:۴۶ ۱۳۹۱/۰۶/۰۴

Crawler همونطور که در متن هم نوشته شده دست سازه و مهم نیست و تابع GetXHtmlFromUri میتونه مثل نمونه زیر باشه و دقت کنید خالی نبودن UseAgent خيله مهمه وگرنه ارور (409) Conflict The remote server returned an error: رو میده. من با همین تابع یک سایت فارسی رو چک کردم و اروری نداد و متن فارسی قابل کوئری گرفتن بود. کامل تر و با ارور هندلینگ بهترش رو میتونید در برنامه مفید [plrip](#) آقای وحید نصیری ببینید

```
private static HtmlDocument GetXHtmlFromUri(string uri) {
    HtmlDocument htmlDoc = new HtmlDocument()
    {
        OptionCheckSyntax = true,
        OptionFixNestedTags = true,
        OptionAutoCloseOnEnd = true,
        OptionDefaultStreamEncoding = Encoding.UTF8
    };

    var request = (HttpWebRequest)WebRequest.Create(uri);
    request.Method = "GET";

    //important
    request.UserAgent = "Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0)";
    request.Accept = "text/html";
    requestAutomaticDecompression = DecompressionMethods.GZip | DecompressionMethods.Deflate;

    using (var response = request.GetResponse() as HttpWebResponse)
    {
        using (var stream = response.GetResponseStream())
        {
            htmlDoc.Load(stream, Encoding.UTF8);
        }
    }
    return htmlDoc;
}
```

اینم روش دوم که بازم UserAgent باید اضافه بشه

```
private static HtmlDocument GetXHtmlFromUri2(string uri) {
    WebClient client = new WebClient() { Encoding = Encoding.UTF8 };
    client.Headers.Add("user-agent", "Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0)");

    HtmlDocument htmlDoc = new HtmlDocument()
    {
        OptionCheckSyntax = true,
        OptionFixNestedTags = true,
        OptionAutoCloseOnEnd = true,
        OptionDefaultStreamEncoding = Encoding.UTF8
    };

    htmlDoc.LoadHtml(client.DownloadString(uri));

    return htmlDoc;
}
```

نویسنده: افشار محبی
تاریخ: ۹:۳۷ ۱۳۹۱/۰۶/۰۵

من هم از این ابزار در کارهایم استفاده کرده‌ام. خیلی خوب جواب می‌دهد.

نویسنده: ایمان عبیدی
تاریخ: ۰:۴ ۱۳۹۱/۰۶/۰۶

برای مشاهده نتایج بدست آمده رده بندی المپیک 2012 لندن به همراه اطلاعات جنسیت مدال گیرها و همچنین وضعیت جدول در روز هایه مختلف میتونید به لینک هایه زیر مراجعه کنید.

تویه این صفحات از پلاگین tableSorter و یکم جاوا اسکریپت هم در لینک اول برای کش کردن اطلاعات json استفاده کردم .
<http://olympics2012.iabidi.ir/Home/DailyMedals>
<http://olympics2012.iabidi.ir/Home/RankingsFull>

نویسنده: پریسا
تاریخ: ۲۰:۵۸ ۱۳۹۲/۰۲/۰۲

چرا وقتی XPath از Firebug استفاده می‌کنم با استفاده از SelectSingleNode جواب دقیقی نمیده یا هیچی نیاره

نویسنده: وحید نصیری
تاریخ: ۱۴:۱۴ ۱۳۹۲/۰۲/۰۳

علت این است که html ایی که در فایرباگ بررسی میشه عموما به دلیل یک سری از نرمال سازی‌ها توسط موتور فایرفاکس و همچنین خودش، با html اصلی یک سایت متفاوت است. به همین جهت XPath استخراجی از آن روی سایت اصلی کار نخواهد کرد.
[یک برنامه کمکی](#) برای یافتن XPathها به همان نحوی که هستند.

نویسنده: mahsan
تاریخ: ۱۰:۳۸ ۱۳۹۲/۰۳/۲۳

این متد رو در چه صفحه ای باید بنویسم؟ آیا باید در همون صفحه ای که میخوام اطلاعات لود بشه بنویسم؟
uri مقدارش را باید داخل تابع بگیره؟ در page load فرمم چی بنویسم؟

نویسنده: وحید نصیری
تاریخ: ۱۱:۲۹ ۱۳۹۲/۰۳/۲۳

- تفاوتی نمی‌کنه [کجا فراخوانی بشه](#) ؛ در page load یا در یک روال رخداد گردان کلیک و یا در یک سرویس مستقل.
- بهتره نتیجه رو بعد از فراخوانی برای مدتی کش کنی، تا هربار اطلاعات از وب درخواست نشود.

نویسنده: mahsan
تاریخ: ۱۲:۱۰ ۱۳۹۲/۰۳/۲۳

بخشین که دوباره مزاحمتون شدم: وقتی من کد بالا رو در page load کپی میکنم زیر بعضی کلمات مانند Crawler , xhtml, XmlNode, Parse, Birthday, LargePhotoUri, GetAttributeValue یک سند html که من باید ایجادش میکردم؟ فقط همین کدها رو بنویسم؟ جواب میگیرم با باید چیز دیگه ای هم اضافه بشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۲۳ ۱۲:۴۱

- شما باید از طریق نیوگت با دستور Install-Package HtmlAgilityPack این بسته رو نصب کنید. یا اینکه فایل DLL اون رو [از سایتش دریافت](#) و به ارجاعات پروژه اضافه کنید.
- کدهای کلاس Crawler چند کامنت بالاتر ارسال شدن. تابع GetXHtmlFromUri که [ملاحظه می کنید](#).
- مواردی مانند Birthday, LargePhotoUri یک سری متغیر هستند که از طرف نویسنده مقاله تعریف شدن. مهم نیستند. حذفشون کنید.
- [یک مثال دیگر در مورد استفاده از کتابخانه HtmlAgilityPack با کد قابل دریافت](#).

نویسنده: mahsan
تاریخ: ۱۳۹۲/۰۳/۲۵ ۹:۰۶

سلام ممنون که جوابم رو دادین
من کلاسی بنام Crawler ایجاد کردم ولی حالا زیر

OptionDefaultStreamEncoding

OptionAutoCloseOnEnd OptionFixNestedTags OptionCheckSyntax

و

LoadHtml

خط قرمز میکشه: (لطف میکنید بگید دلیلش چیه و باید چیکار کنم؟
و در فرم هم زیر این کلمه GetXHtmlFromUri
مینویسه
generate metod stub for 'GetXHtmlFromUri in "crawel

وقتی من متد GetXHtmlFromUri را در کلاس crawl تعریف کردم چرا باید این پیغام رو بده ؟ آیا باید این گزینه رو انتخاب کنم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۲۵ ۹:۴۳

پیشنهاد من این است که یک دوره سی شارپ مقدماتی رو بگذرونید. با مباحثی مانند نحوه تعریف فضای نام و روش فراخوانی یک متد استاتیک از کلاس متناظر با آن آشنا شوید.
[یک دوره خوب مقدماتی سی شارپ](#)

نویسنده: ایمان توکلی
تاریخ: ۱۳۹۲/۱۱/۲۶ ۱۷:۱۰

با سپاس
در صورتی که مقداری در querystring مربوط به صفحه اضافه شود و در هر درخواست این مقدار تغییر کند چطور می توان صفحه را خواند مثال: <http://website.com?log=1731004>
سایت برای هر بازدید یک لاگ جدید می نویسد یعنی هر درخواست جدید در صورتی که لاگ معتبر نباشد به صفحه دیگر ارسال می کند. حال اطلاعات این صفحه را چطور می توان خواند ؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۲۶ ۱۷:۲۱

این نوع مسایل رو نمی‌تونید با HtmlAgilityPack حل کنید. HtmlAgilityPack فقط کارش آنالیز اطلاعات ثابتی هست که بهش می‌دید و نهایتاً خواندن نودهای HTML نهایی. موردی که عنوان کردید نیاز به آنالیز همزمان هدرهای دریافتی به همراه جاوا اسکریپت سایت مورد نظر داره.

یک سری از دوره‌های پلورال‌سایت دارای زیرنویس هستند که تحت عنوان [Transcript](#) در کنار آن‌ها قرار گرفته‌اند:



Building Applications with ASP.NET MVC 4

This course is a comprehensive introduction to ASP.NET MVC 4, and will give you the essentials you need to start building applications with Microsoft's MVC framework.


[Table of Contents](#)
[Description](#)
[Transcript](#)
[Exercise Files](#)
[Assessment](#)
[Discussion](#)

این زیرنویس‌ها فرمت ویژه‌ای دارند:

```
<li class="transcript-module">
  Introduction to ASP.NET MVC 4
  <ul>
    <li class="transcript-clip" data-p="author=scott-allen&name=mvc4-
building-m1-intro&mode=live&clip=0&course=mvc4-building"><a href="javascript:void(0)"
onclick="LaunchPlayerWindow('http://pluralsight.com/training', 'author=scott-allen&name=mvc4-
building-m1-intro&mode=live&clip=0&course=mvc4-building');">Introduction</a><br />
    <div>
      <a href="javascript:void(0)" onclick="p(this);" data-
s="1.636">Hi, this is Scott Allen and this is the first module in the course design</a>
    </div>
    </li>
    <li class="transcript-clip" data-p="author=scott-allen&name=mvc4-
building-m1-intro&mode=live&clip=1&course=mvc4-building"><a href="javascript:void(0)"
onclick="LaunchPlayerWindow('http://pluralsight.com/training', 'author=scott-allen&name=mvc4-
building-m1-intro&mode=live&clip=1&course=mvc4-building');">Web Platform Installer</a><br
/>
    <div>
      ...
    </div>
  </ul>
</li>
```

در آن، هر li که دارای کلاسی به نام transcript-clip است، حاوی یک div می‌باشد و این div دارای تعدادی لینک است. این لینک‌ها توسط ویژگی datas آن‌ها که بیانگر زمان شروع گفتگو است، مشخص می‌شوند و همین‌طور الی آخر. بنابراین اگر بخواهیم برای آن‌ها ساختاری را تهیه کنیم، به کلاس‌های ذیل خواهیم رسید:

```
public class TranscriptClip
{
    public string Title { set; get; }
    public IList<TranscriptItem> TranscriptItems { set; get; }
}

public class TranscriptItem
{
    public double StartTime { set; get; }
    public string Text { set; get; }
}
```

هر li دارای کلاس transcript-clip، یک شیء TranscriptClip را تشکیل می‌دهد. هر شیء TranscriptClip می‌تواند دارای چندین TranscriptItem باشد.

برای استخراج این اطلاعات، یکی از بهترین ابزارها، کتابخانه [HTML Agility pack](#) است که توسط آن می‌توان به liهای یاد شده دسترسی یافت:

```
var nodes = doc.DocumentNode.SelectNodes("//li[@class='transcript-clip']/div");
```

و سپس اطلاعات آن‌ها را استخراج نمود.

```
using System;
using System.Collections.Generic;
using System.Globalization;
using System.IO;
using System.Linq;
using System.Text;
using System.Text.RegularExpressions;
using System.Web;
using HtmlAgilityPack;

namespace PluralsightTranscripts
{
    public class TranscriptClip
    {
        public string Title { set; get; }
        public IList<TranscriptItem> TranscriptItems { set; get; }
    }

    public class TranscriptItem
    {
        public double StartTime { set; get; }
        public string Text { set; get; }
    }

    public class ExtractSubtitle
    {
        public static void ConvertToSrt(string fileName)
        {
            var transcriptClips = extractItems(fileName);
            var itemNumber = 1;
            foreach (var item in transcriptClips)
            {
                transcriptClipToSrt(item, itemNumber);
                itemNumber++;
            }
        }

        private static void transcriptClipToSrt(TranscriptClip item, int itemNumber)
        {
            var count = item.TranscriptItems.Count;
            var srtFileContent = transcriptItemsToSrt(item.TranscriptItems, count);
            var fileName = removeIllegalCharacters(string.Format("{0}-{1}.srt",
itemNumber.ToString("00"), item.Title));
            File.WriteAllText(fileName, srtFileContent);
        }

        private static string transcriptItemsToSrt(IList<TranscriptItem> items, int count)
        {
            var lineNumber = 1;
            var sb = new StringBuilder();
            for (int row = 0; row < count; row++)
            {
                sb.AppendLine(lineNumber.ToString(CultureInfo.InvariantCulture));
                sb.AppendLine(getTimeLine(items, count, row));
                sb.AppendLine(items[row].Text);
                sb.AppendLine(string.Empty);
                lineNumber++;
            }
            return sb.ToString();
        }

        private static string getTimeLine(IList<TranscriptItem> items, int count, int row)
        {
            var startTs = TimeSpan.FromSeconds(items[row].StartTime);
            var endTs = row + 1 < count ? TimeSpan.FromSeconds(items[row + 1].StartTime) :
TimeSpan.FromSeconds(items[row].StartTime + 5);
            return string.Format("{0} --> {1}", timeSpanToString(startTs), timeSpanToString(endTs));
        }

        private static string timeSpanToString(TimeSpan lineTs)
        {

```

```

    {
        return string.Format("{0}:{1}:{2},{3}", lineTs.Hours.ToString("D2"),
lineTs.Minutes.ToString("D2"), lineTs.Seconds.ToString("D2"), lineTs.Milliseconds.ToString("D3"));
    }

    private static string removeIllegalCharacters(string fileName)
    {
        string regexSearch = string.Format("{0}{1}",
                                                    new string(Path.GetInvalidFileNameChars()),
                                                    new string(Path.GetInvalidPathChars()));
        var r = new Regex(string.Format("[{0}]", Regex.Escape(regexSearch)));
        return r.Replace(fileName, ".");
    }

    private static IList<TranscriptClip> extractItems(string fileName)
    {
        var htmlContent = File.ReadAllText(fileName);
        var results = new List<TranscriptClip>();

        var doc = new HtmlDocument
        {
            OptionCheckSyntax = true,
            OptionFixNestedTags = true,
            OptionAutoCloseOnEnd = true,
            OptionDefaultStreamEncoding = Encoding.UTF8
        };
        doc.LoadHtml(htmlContent);

        var nodes = doc.DocumentNode.SelectNodes("//li[@class='transcript-clip']/div");
        foreach (var node in nodes)
        {
            var itemsList = new List<TranscriptItem>();
            var title = node.ParentNode.ChildNodes.First(x => x.Name == "a").InnerText;

            foreach (var childNode in node.ChildNodes)
            {
                if (childNode.Name != "a") continue;

                var dataS = childNode.Attributes.First(x => x.Name == "data-s");
                itemsList.Add(new TranscriptItem
                {
                    StartTime = double.Parse(dataS.Value),
                    Text = HttpUtility.HtmlDecode(childNode.InnerText.Trim())
                });
            }

            results.Add(new TranscriptClip { TranscriptItems = itemsList, Title = title });
        }

        return results;
    }
}

```

اگر این اطلاعات را کنار هم قرار دهیم، به کلاس کمکی فوق خواهیم رسید. کار با گره‌های li شروع می‌شود. سپس در این گره‌ها، کلیه گره‌های a یا لینک‌ها، یافت شده و سپس dataS و متن آن‌ها استخراج می‌شوند. اگر این‌ها را نهایتاً کنار هم قرار دهیم، می‌توان به فرمت SRT متداول که اکثر پخش‌کننده‌های فایل‌های تصویری قادر به پردازش آن‌ها هستند، رسید. فرمت SRT ساختار ساده‌ای دارد. هر گفتگوی آن حداقل از سه سطر تشکیل می‌شود. سطر اول یک شماره خود افزایشی است. سطر دوم زمان شروع و پایان گفتگو را مشخص می‌کند و سطر سوم بیانگر متن گفتگو است. برای مثال:

```

1
00:00:01,636 --> 00:00:05,616
Hi, this is Scott Allen and this is the first module in the course design

```

دریافت پروژه کامل این مطلب

[PluralsightTranscripts.zip](#)

نظرات خوانندگان

نویسنده: الهام
تاریخ: ۲۲:۲۱ ۱۳۹۲/۰۱/۰۹

سلام آقای نصیری

آیا شما عضو سایت پلورال سایت هستید؟ چقدر پرداخت کردید و این مبلغ را چطور با توجه به وضع ایران واریز کردید؟ آیا ارزش عضو شدن رو داره؟

بخشید چون من دانشجو هستم و بدنبال یادگیری حرفه ای برنامه نویسی و زبانم هم خوبه میخوامم از منابع انگلیسی استفاده کنم.

با تشکر

نویسنده: وحید نصیری
تاریخ: ۲۲:۳۴ ۱۳۹۲/۰۱/۰۹

سلام! من عضو نیستم.

نویسنده: حسین
تاریخ: ۰:۱۳ ۱۳۹۲/۰۱/۱۰

این زیرنویس‌ها فقط برای اعضای اون سایت در دسترسه.از کجا میشه بهشون دسترسی پیدا کرد ؟

نویسنده: وحید نصیری
تاریخ: ۰:۱۶ ۱۳۹۲/۰۱/۱۰

لطفا روی لینک مطرح شده در سطر اول مطلب فوق [کلیک کنید](#) . کل بحث جاری در مورد استخراج اطلاعات و تبدیل فرمت خاص صفحه وبی بود که ملاحظه می‌کنید. این صفحه هم عمومی است (هر چند ظاهر ساده‌ای دارد، اما پشت صحنه و سورس آن، متن زمانبندی شده کل دوره است).

نویسنده: علیرضا جهانشاهلو
تاریخ: ۲:۴۰ ۱۳۹۲/۰۱/۱۴

اتفاقا سایت Lynda هم از همین روش استفاده میکنه و من با کمی تغییر موفق شدم که فایل‌های Transcript آموزشی هاشو استخراج کنم.

ممنون مهندس

نویسنده: سیروان عقیفی
تاریخ: ۱۵:۴۳ ۱۳۹۲/۰۷/۲۷

ظاهراً ساختار عوض شده به این شکل (البته در اینجا data-s حذف شده و مقدار آن به صورت رشته ایی در انتهای مقدار ng-click اضافه شده است به صورت start=39.796 :

```
<li class="transcript-clip">
<a href="javascript:void(0)" ng-click="launchPlayerWindow('http://pluralsight.com/training',
'author=scott-allen&name=mvc4-building-m1-intro&mode=live&clip=0&course=mvc4-
building');">Introduction</a><br>
```

```
<div>
<a href="javascript:void(0)" ng-click="launchPlayerWindow('http://pluralsight.com/training',
'author=scott-allen&name=mvc4-building-m1-intro&mode=live&clip=0&course=mvc4-
building&start=39.796');">and also have an understanding of the design goals of the MVC
framework.</a>
    <a href="javascript:void(0)" ng-click="launchPlayerWindow('http://pluralsight.com/training',
'author=scott-allen&name=mvc4-building-m1-intro&mode=live&clip=0&course=mvc4-
building&start=43.796');">So, let's get started.</a>
</div>
</li>
```

نویسنده: وحید نصیری
تاریخ: ۱۷:۲۱ ۱۳۹۲/۰۷/۲۷

- البته من عضو نیستم و به نظر جدیداً عنوان کردند «Sorry, transcripts are only available to subscribers».
- در کدهای فوق، فقط این چند سطر باید تغییر کنند:

```
//var dataS = childNode.Attributes.First(x => x.Name == "data-s");
var dataS = childNode.Attributes.First(x => x.Name == "ng-click");
var startTime = new Regex("(?s)start=(.+)").Matches(dataS.Value)
    .OfType<Match>()
    .First()
    .Groups[1]
    .Value;

itemsList.Add(new TranscriptItem
{
    StartTime = double.Parse(startTime),
    Text = HttpUtility.HtmlDecode(childNode.InnerText.Trim())
});
```

نویسنده: پویان
تاریخ: ۱۶:۲۴ ۱۳۹۳/۰۱/۲۴

با سلام
الان که حتماً باید در سایت plural sight عضو باشیم راهی نیست تا زیرنویس بگیریم ؟
ایا شما زیرنویس بعضی از فیلم‌ها را دارید ؟

نویسنده: وحید نصیری
تاریخ: ۱۶:۳۸ ۱۳۹۳/۰۱/۲۴

[فایل تورنت](#) پیوست شده حاوی مثال‌ها و زیرنویس‌های 52 دوره هست:
[srt_only.zip](#)

اولین قدم کار کردن با کتابخانه قدرتمند HtmlAgilityPack، داشتن [XPath](#) معتبر و متناظر با یک گره خاص می‌باشد. هرچند به ظاهر تعدادی از مرورگرها با کمک افزونه‌های خود امکان استخراج این XPath ها را فراهم کرده‌اند اما ... عموماً این مقادیر ارائه شده، نادرست هستند و بر روی محتوای HTML اصلی یک سایت قابل اجرا نیستند؛ علت هم به نرمال سازی‌های انجام شده بر روی محتوای یک سایت، توسط موتور مرورگر بر می‌گردد.

خود کتابخانه HtmlAgilityPack به ازای هر XmlNode ای که ارائه می‌دهد، خاصیت XPath معتبری را نیز به همراه دارد. در ادامه قصد داریم از این امکان توکار استفاده کرده و کلیه XPath های یک محتوای HTML ای را استخراج کنیم.

پردازش تگ‌های تو در توی یک HTML به کمک کتابخانه HtmlAgilityPack

```
using System;
using System.Linq;
using System.Net;
using System.Text;
using HtmlAgilityPack;

namespace HapTests
{
    public class HtmlReader
    {
        public Action<string> ParseError { set; get; }

        public Func<XmlNode, bool> ParserXmlNode { set; get; }

        public void StartParsingHtml(Uri url)
        {
            using (var client = new WebClient { Encoding = Encoding.UTF8 })
            {
                client.Headers.Add("user-agent", "Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; WOW64; Trident/5.0)");
                StartParsingHtml(client.DownloadString(url));
            }
        }

        public void StartParsingHtml(string htmlContent)
        {
            if (string.IsNullOrEmpty(htmlContent))
                throw new ArgumentNullException("content");

            var doc = new HtmlDocument
            {
                OptionCheckSyntax = true,
                OptionFixNestedTags = true,
                OptionAutoCloseOnEnd = true,
                OptionDefaultStreamEncoding = Encoding.UTF8
            };
            doc.LoadHtml(htmlContent);

            if (doc.ParseErrors != null && doc.ParseErrors.Any())
            {
                foreach (var error in doc.ParseErrors)
                {
                    if (ParseError != null)
                        ParseError(error.Code + " - " + error.Reason);
                }
            }

            if (!doc.DocumentNode.HasChildNodes)
                return;

            handleChildren(doc.DocumentNode.ChildNodes);
        }

        private void handleChildren(XmlNodeCollection nodes)
        {
            foreach (var itm in nodes)
            {
                if (itm.NodeType == XmlNodeType.Element)
                {
                    if (ParserXmlNode(itm) == true)
                    {
                        // ... 
                    }
                }
            }
        }
    }
}
```

```

        {
            if (itm.Name.ToLower().Equals("html"))
            {
                if (itm.Element("body") != null)
                    handleChildren(itm.Element("body").ChildNodes);
            }
            else
                handleHtmlNode(itm);
        }
    }

    private void parserChildNodes(HtmlNode content)
    {
        foreach (var item in content.ChildNodes)
        {
            handleHtmlNode(item);
        }
    }

    private void handleHtmlNode(HtmlNode htmNode)
    {
        switch (htmNode.Name.ToLower())
        {
            case "html":
            case "body":
                handleChildren(htmNode.ChildNodes);
                break;

            default:
                if (ParserHtmlNode == null)
                    throw new ArgumentNullException("ParserHtmlNode");

                if (ParserHtmlNode(htmNode))
                    parserChildNodes(htmNode);

                break;
        }
    }
}

```

در اینجا کدهایی را ملاحظه می‌کنید که علاوه بر ارائه تنظیمات اولیه HtmlAgilityPack (خصوصاً با در نظر گرفتن مباحث ورودی یونیکد)، به صورت بازگشتی (با توجه به اینکه الزاماً مسیر یا Node خاصی مدنظر نیست)، کلیه گره‌های یک HTML را بررسی و ارائه می‌دهند.

این کد برای نوشتن مبدل‌های HTML به XYZ بسیار مناسب است. برای مثال اگر بخواهید یک مبدل HTML به PDF را تهیه کنید، کدهای ابتدایی آن همین موارد است:

```

new HtmlReader
{
    ParseError = error => Console.WriteLine(error),
    ParserHtmlNode = htmlNode =>
    {
        //switch(htmlNode.Name) { }
        return true; //it's a nested node.
    }
}.StartParsingHtml(html);

```

نمونه‌ای از نحوه استفاده از کدهای کلاس HtmlReader را ملاحظه می‌کنید.

در اینجا html، محتوای HTML ای در حال بررسی است. ParserHtmlNode یک callback است. هر زمانیکه به یک گره HTML برخورد، آن را در اختیار شما قرار می‌دهد. در ادامه فرصت خواهید داشت تا برای نمونه یک swich را تهیه کرده و مثلاً به ازای تگ hr یک خط رسم کنید، به ازای تگ br یک سطر جدید را در نظر بگیرید و الی آخر. اگر خروجی این Func را true در نظر بگیرید، فرض بر این خواهد بود که گره جاری تو در تو است (حالت دنیای واقعی)؛ در غیر این صورت، یک سطح این گره، بیشتر بررسی نخواهد شد.

در این کلاس، ParseError نیز یک callback است و اگر کتابخانه HtmlAgilityPack، در حین آنالیز کدهای HTML دریافتی به خطایی برخورد، آن را گزارش خواهد داد.

در کلاس فوق، دو حالت برای متد StartParsingHtml در نظر گرفته شده است. در حالت اول، یک Uri یا آدرس اینترنتی دریافت


```

class='l3_3'>1,200,400</td><td class='h3_3'>1,228,100</td><td class='z3_3 fa'>17:50</td>
</tr><tr><td>انسى نقره</td><sup>دلار</sup></td><td class='s0_5'>21.83</td><td
class='c0_5'>(0.00%) 0.00</td><td class='l0_5'>21.67</td><td class='h0_5'>21.96</td>
<td class='z0_5 fa'>17:53</td></tr></tbody></table><br><table
id='coin_tbl'><tbody><tr><th>سكه</th><th>زنده قيمت</th><th>تغيير</th><th>كمترين</th>
<th>بيشترين</th><th>طلارش</th><th>زمان</th></tr><tr><td>بهار
آزادى</td><td class='s3_10'>12,650,000</td><td class='c3_10 pos'>(2.68%) 330,000</td>
<td class='l3_10'>12,320,000</td><td class='h3_10'>12,650,000</td><td
class='z4_10'>11,918,400</td><td class='z3_10 fa'>16:07</td></tr><tr><td>امامى</td>
<td class='s3_11'>12,960,000</td><td class='c3_11 pos'>(2.61%)
330,000</td><td class='l3_11'>12,630,000</td><td class='h3_11'>13,050,000</td><td
class='z4_11'>11,918,400</td>
<td class='z3_11 fa'>17:43</td></tr><tr><td>نيم</td><td
class='s3_12'>6,880,000</td><td class='c3_12 pos'>(2.69%) 180,000</td><td class='l3_12'>6,700,000</td>
<td class='h3_12'>6,900,000</td><td class='z4_12'>5,959,200</td><td
class='z3_12 fa'>16:08</td></tr><tr><td>ربع</td><td class='s3_13'>4,250,000</td><td class='c3_13
pos'>(2.41%) 100,000</td>
<td class='l3_13'>4,150,000</td><td class='h3_13'>4,300,000</td><td
class='z4_13'>2,978,100</td><td class='z3_13 fa'>17:42</td></tr><tr><td>گر مى</td><td
class='s3_14'>2,940,000</td>
<td class='c3_14 pos'>(3.16%) 90,000</td><td
class='l3_14'>2,850,000</td><td class='h3_14'>2,940,000</td><td class='z4_14'>1,465,400</td><td
class='z3_14 fa'>17:40</td></tr></tbody></table></div></td>
</tr>
</tbody></table>
";

extractXPath(html);
test(html);
}

/// <summary>
/// Converts /#comment[1] to /comment()[1]
/// or /#text[1] to /text()[1]
/// </summary>
private static string GetValidXPath(string xpath)
{
    var index = xpath.LastIndexOf("/");
    var lastPath = xpath.Substring(index);

    if (lastPath.Contains("#"))
    {
        xpath = xpath.Substring(0, index);
        lastPath = lastPath.Replace("#", "");
        lastPath = lastPath.Replace("[", "()[");
        xpath = xpath + lastPath;
    }

    return xpath;
}

private static void extractXPath(string html)
{
    var sb = new StringBuilder();
    new HtmlReader
    {
        ParseError = error => Console.WriteLine(error),
        ParserHtmlNode = htmlNode =>
        {
            if (htmlNode is HtmlTextNode)
            {
                sb.AppendLine("Text NodeName: " + htmlNode.Name.Trim());
                sb.AppendLine("InnerText: " + htmlNode.InnerText.Trim());
            }
            else
            {
                sb.AppendLine("NodeName: " + htmlNode.Name.Trim());
                var nodeText = new StringBuilder();
                for (int i = 0; (i < htmlNode.OuterHtml.Length && htmlNode.OuterHtml[i] !=
'>'); i++)
                    nodeText.Append(htmlNode.OuterHtml[i]);

                nodeText.Append(">");

                sb.AppendLine("Node Start: " + nodeText.ToString());
            }

            sb.AppendLine("XPath: " + GetValidXPath(htmlNode.XPath.Trim()));
            sb.AppendLine(Environment.NewLine);

            return true; //it's a nested node.
        }
    };
}

```

```

    }
    }.StartParsingHtml(html);

    File.WriteAllText("xpath.txt", sb.ToString());
    Process.Start("xpath.txt");
}

private static void test(string html)
{
    var doc = new HtmlDocument
    {
        OptionCheckSyntax = true,
        OptionFixNestedTags = true,
        OptionAutoCloseOnEnd = true,
        OptionDefaultStreamEncoding = Encoding.UTF8
    };
    doc.LoadHtml(html);
    var node =
doc.DocumentNode.SelectSingleNode("/table[1]/tbody[1]/tr[7]/td[1]/div[1]/table[2]/tbody[1]/tr[6]/td[7]/text()[1]");
    Console.WriteLine(node.InnerText);
}
}
}

```

در این مثال html مقداری است که از یک سایت عمومی دریافت شده است. سپس نمونه‌ای دیگر از نحوه استفاده از کلاس HtmlReader قسمت قبل را در ادامه، در متد extractXPath ملاحظه می‌کنید. در اینجا کلاس HtmlReader در یک عملیات بازگشتی، کلیه گره‌های تو در توی HTML مورد نظر را آنالیز کرده و توسط callback ای به نام ParserHtmlNode در اختیار ما قرار می‌دهد. اکنون که این htmlNode را داریم، خاصیت XPath آن دقیقاً مقداری است که به دنبالش هستیم.

در اینجا چند نکته حائز اهمیت هستند:

- با بررسی HtmlTextNode، به نودهایی خواهیم رسید که دارای مقدار متنی هستند. در غیراینصورت این گره، خود ابتدای یک سری گره تو در توی دیگر است.

- XPath بازگشتی توسط کتابخانه HtmlAgilityPack نیاز به کمی تمیز سازی دارد. اینکار در متد GetValidXPath انجام شده است.

- در متد test انتهایی، نمونه‌ای از نحوه استفاده از XPath های استخراجی را ملاحظه می‌کنید.

```

Text NodeName: #text
InnerText: 17:40
XPath: /table[1]/tbody[1]/tr[7]/td[1]/div[1]/table[2]/tbody[1]/tr[6]/td[7]/text()[1]

```

برای نمونه سه سطر فوق، یکی از مداخل فایل نهایی تولیدی مثال جاری است. اکنون که XPath را داریم، استفاده از آن جهت استخراج مقدار InnerText مدنظر، ساده خواهد بود.

نظرات خوانندگان

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۲/۰۳/۲۷ ۸:۸

ممنون بابت مطلب مفیدتون، برای پردازش محتوای جاوا اسکریپت هم میشه از این کتابخانه استفاده کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۲۷ ۸:۴۴

- برای استخراج هر نوع تگ قرار گرفته شده داخل HTML نهایی، میشه از این کتابخانه استفاده کرده. فقط کافی است در switch htmlNode.Name مطلب فوق، scriptها را تحت نظر قرار داد.
- برای اجرای کدهای جاوا اسکریپت در دات نت، [یک سری موتور ویژه برای اینکار هست](#).

نویسنده: علی
تاریخ: ۱۳۹۲/۰۳/۲۷ ۱۰:۳۹

سلام جناب نصیری
آیا سرعت این کتابخانه از کتابخانه [LINQ To HTML](#) بیشتر هست؟ اگر مقایسه اس هم انجام بدهید عالی می شود

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۲۷ ۱۱:۳۵

برای انتخاب یک کتابخانه صرفا به سرعت آن نباید توجه کرد. این موارد برای انتخاب کتابخانه های ثالث، مهم هستند:
- آیا این کتابخانه محلی برای بحث و رفع اشکال دارد؟
- آیا سورس باز است؟ (غیر الزامی؛ اما یک امتیاز مثبت)
- آیا به همراه مثال های کاربردی است؟
- آیا مستندات قابل قبولی دارد؟
- آیا در جستجویی که انجام شده، کسی از آن در پروژه های خودش استفاده می کند؟
- آیا هر از چندگاهی به روز می شود؟ آخرین باری که به روز شده چه زمانی بوده؟
- آیا استفاده از آن در انواع و اقسام پروژه ها مجاز است؟ مجوز استفاده از آن به چه نحوی است؟

نویسنده: افشین
تاریخ: ۱۳۹۲/۰۳/۲۸ ۲:۹

بسیار مفید بود جناب نصیری.. ممنون.

یک مشکلی که هست، وقتی متن [\(این\)](#) صفحه رو با این روش پردازش می کنم، کاراکترهای نامفهومی نمایش داده می شه..
Encoding رو چطور تنظیم کنم، یا مشکل از جای دیگه ای هست؟

برای مثال InnerText این XPath:

```
/html[1]/body[1]/table[1]/tr[1]/td[1]/table[1]/tr[1]/td[2]/table[1]/tr[1]/td[1]/font[1]/td[2]/font[1]/td[1]/map[1]/tr[3]/td[1]/table[1]/tr[1]/td[1]/table[1]/tr[3]/td[1]/table[1]/tr[1]/td[1]/table[1]/tr[1]/td[1]/table[1]/tr[1]/td[2]/a[1]/td[1]/a[1]/table[3]/tr[2]/td[2]/table[1]/tr[1]/td[1]/div[1]/span[1]/span[1]/html[1]/head[1]/title[1]/text()[1]
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۲۸ ۹:۳۱

این صفحه 1256 است.

```
<meta http-equiv="Content-Type" content="text/html; charset=windows-1256">
```

در کدهای فوق به این شکل باید تنظیم شود:

```
using (var client = new WebClient { Encoding = Encoding.GetEncoding("windows-1256") })
```

نویسنده:

صابر فتح الهی

تاریخ:

۱۸:۳۸ ۱۳۹۲/۰۵/۱۶

سلام؛ من از این کتابخانه استفاده کردم اما وقتی سایت دالود شد که روش پردازش انجام بدم درخواست های نامتقارنی که بعدا لود میشن و محتوی صفحه تغییر میدن وجود ندارن چطور به اون محتویات دسترسی داشته باشم؟

نویسنده:

وحید نصیری

تاریخ:

۱۸:۵۴ ۱۳۹۲/۰۵/۱۶

- این کتابخانه پردازشگر جاوا اسکریپتی نداره (همزمان و یا حالت های دیگری مانند Ajax ای). فرضش بر این است که محتویات کامل رو در اختیارش قرار دادید.
- یک راه این است که از Web Control دات نت (موجود در WinForms و همچنین WPF) که در پشت صحنه از موتور کامل IE استفاده می کند، کمک بگیرید و زمانیکه Document آن [کاملا load شد](#) ، نتیجه آنرا به این کتابخانه ارسال کنید.

نویسنده:

صابر فتح الهی

تاریخ:

۱۹:۲۹ ۱۳۹۲/۰۵/۱۶

استفاده کردم حق با شما بود سایت کامل لود می کنه اما در خواست های Ajax پردازش نمی شه، یعنی این کامل شدن لود ربطی به درخواست های ای جکس نداره، خروجی ای جکسی وجود داره توی صفحه ولی در سورس html وجود ندارد

عنوان:	دریافت زمانبندی شده به روز رسانی‌های آنتی ویروس Symantec به کمک کتابخانه‌های Quartz.NET و Html Agility Pack
نویسنده:	سیروان عقیقی
تاریخ:	۸:۳۰ ۱۳۹۲/۰۴/۰۳
آدرس:	www.dotnettips.info
برچسب‌ها:	Regular expressions, Quartz.NET, Html Agility Pack, Asynchronous Programming

در این رابطه آقای راد در دو قسمت به صورت مختصر و مفید این کتابخانه قدرتمند رو همراه با ارائه چندین مثال کاربردی معرفی کردند:

[قسمت اول](#)

[قسمت دوم](#)

در تکمیل قسمت‌های فوق بنده می‌خواهم مثالی رو در این رابطه براتون بذارم، هدف از ارائه این مثال اتوماتیک سازی یک فرآیند روتین می‌باشد، به این صورت که در جایی که بنده مشغول به کار هستم یک سری لایسنس آنتی ویروس برای کلاینت‌ها در یک شبکه با مقیاس متوسط تهیه گردیده است، حال یک نسخه رایگان نیز برای کاربرانی که قصد دارند آنتی ویروس را برای سیستم شخصی خود نصب کنند نیز موجود می‌باشد که نیاز به آپدیت دارد معمولاً آپدیت‌ها هر چند روز یکبار یا هر هفته در دو نسخه 64 و 32 بیتی ارائه می‌شوند، روال معمول برای دریافت آپدیت مراجعه به سایت و دانلود نسخه‌های مربوطه می‌باشد.

حال توسط کتابخانه قدرتمند [Quartz.NET](#) این فرآیند روتین را به صورت اتوماتیک می‌خواهیم انجام دهیم، استفاده از کتابخانه ذکر شده سخت نیست همانطور که در دو مطلب قبلی مرتبط ذکر گردیده، تنها پیاده سازی چندین اینترفیس است و بس.

```
namespace SymantecUpdateDownloader
{
    using System;
    using System.IO;
    using Quartz;
    using Quartz.Impl;
    using System.Globalization;
    public class TestJob : IJob
    {
        public void Execute(IJobExecutionContext context)
        {
            new Download().Scraping();
        }
    }
    public interface ISchedule
    {
        void Run();
    }
    public class TestSchedule : ISchedule
    {
        public void Run()
        {
            DateTimeOffset startTime = DateBuilder.FutureDate(2, IntervalUnit.Second);

            IJobDetail job = JobBuilder.Create<HelloJob>()
                .WithIdentity("job1")
                .Build();

            ITrigger trigger = TriggerBuilder.Create()
                .WithIdentity("trigger1")
                .StartAt(startTime)
                .WithDailyTimeIntervalSchedule(x =>
                    x.OnEveryDay().StartingDailyAt(new TimeOfDay(7, 0)).WithRepeatCount(0))
                .Build();

            ISchedulerFactory sf = new StdSchedulerFactory();
            IScheduler sc = sf.GetScheduler();
            sc.ScheduleJob(job, trigger);

            sc.Start();
        }
    }
}
```

در این کد که همانند کدهای پیشنهادی مطلب است، در خط 33 از متد `WithDailyTimeIntervalSchedule` استفاده شده است

و همانطور که مشخص است وظیفه تعیین شده و هر روز ساعت 7 اجرا میشود.

مورد بعدی عملیات دانلود فایل می‌باشد که در ادامه مشاهده خواهید کرد، [صفحه ایی](#) که لینک فایل‌های دانلود را ارائه داده است دو نسخه مد نظر ما را در ابتدا لیست کرده است و با استفاده از web scraping می‌توانیم موارد تعیین شده را استخراج کنیم برای این منظور از کتابخانه [htmlagilitypack](#) استفاده میکنیم، تطبیق دو مورد (لینک) اول جهت دریافت نسخه‌های 32 و 64 بیتی به کمک Regular Expression میسر است و همانطور که در شکل زیر مشاهده میکنید از سمت چپ تاریخ به صورت 8 رقم، سه رقم قسمت دوم و ارقام و حروف قسمت سوم است به اضافه پسوند فایل مشخص است :



```
public class Download
{
    static WebClient wc = new WebClient();
    static ManualResetEvent handle = new ManualResetEvent(true);

    private DateTime myDate = new DateTime();
    public void Scraping()
    {
        using (WebClient client = new WebClient())
        {
            client.Encoding = System.Text.Encoding.UTF8;
            var doc = new HtmlAgilityPack.HtmlDocument();
            ArrayList result = new ArrayList();

            doc.LoadHtml(client.DownloadString("https://www.symantec.com/security_response/definitions/download/detail.jsp?gid=savce"));
            var tasks = new List<Task>();
            foreach (var href in doc.DocumentNode.Descendants("a").Select(x =>
                x.Attributes["href"]))
            {
                if (href == null) continue;
                string s = href.Value;
                Match m = Regex.Match(s, @"http://definitions.symantec.com/defs/(\d{8}-\d{3}-v5i(32|64)\.exe)");
                if (m.Success)
                {
                    Match date = Regex.Match(m.Value, @"(\d{4})(\d{2})(\d{2})");
                    Match filename = Regex.Match(m.Value, @"(\d{8}-\d{3}-v5i(32|64)\.exe)");
                    int year = Int32.Parse(date.Groups[0].Value);
                    int month = Int32.Parse(date.Groups[1].Value);
                    int day = Int32.Parse(date.Groups[3].Value);

                    myDate = new DateTime(
                        Int32.Parse(date.Groups[1].Value),
                        Int32.Parse(date.Groups[2].Value),
                        Int32.Parse(date.Groups[3].Value));
                    if (myDate == DateTime.Today)
                    {
                        tasks.Add(DownloadUpdate(m.Value, filename.Value));
                    }
                    else
                    {
                        MessageBox.Show("امروز آپدیت موجود نیست");
                    }
                }
            }
            DownloadTask = Task.WhenAll(tasks);
        }
    }
    private static Task DownloadTask;
    private Task DownloadUpdate(string url, string fileName)
    {
        var wc = new WebClient();
        return wc.DownloadFileTaskAsync(new Uri(url), @"\\10.1.0.15\SymantecUpdate\\" + fileName);
    }
}
```

```
}
```

توضیح کدهای فوق :

ابتدا توسط متد LoadHtml خط 14 صفحه مورد نظر که حاوی لینک‌ها می‌باشد رو Load میکنیم، سپس توسط یک حلقه foreach خط 16 مقدار خصوصیت href تمام لینک‌های موجود در صفحه را استخراج میکنیم مثلا مقدار خصوصیت href در لینک‌ها به صورت زیر می‌باشد :

```
http://definitions.symantec.com/defs/20130622-007-v5i32.exe
```

```
http://definitions.symantec.com/defs/20130622-007-v5i64.exe
```

همانطور که مشخص است در دو مورد فوق تنها نام فایل متفاوت می‌باشد، همانطور که بحث شد برای نام فایل‌ها هم می‌توانیم یک Pattern را به صورت زیر داشته باشیم :

```
(\d{8}-\d{3}-v5i(32|64)\.exe)
```

در خط 20 نیز عملیات تطبیق تمام hrefهای موجود در صفحه را توسط Regular Expression فوق تطبیق می‌دهیم، اگر تطبیق با موفقیت انجام پذیرفت باید نام فایل و همچنین تاریخ موجود در نام فایل را نیز توسط دو Regular Expression استخراج کنیم(خط 23 و 24) در ادامه برای جدا کردن مقادیر سال ، ماه ، روز از امکان Groups در RegEx استفاده کرده ایم:

```
int year = Int32.Parse(date.Groups[0].Value);  
int month = Int32.Parse(date.Groups[1].Value);  
int day = Int32.Parse(date.Groups[3].Value);
```

در ادامه تاریخ استخراج شده را با تاریخ روز جاری مقایسه می‌کنیم اگر مساوی بود عملیات دانلود فایل‌ها توسط یک [Task](#) تعریف شده به صورت [همزمان](#) بر روی سرور مربوطه دانلود می‌شوند. البته لازم به ذکر است که کدهای فوق مسلما نیاز به Refactoring دارند منتها هدف از ارائه این مثال آشنایی بیشتر با کتابخانه‌های فوق می‌باشد.

نکته آخر اینکه برنامه فوق به حالت‌های مختلفی می‌تواند اجرا گردد مثل یک برنامه وب یا یک سرویس ویندوزی و ... ، بهترین حالت یک سرویس ویندوز می‌باشد، ولی در حالت خام در حال حاضر یک ویندوز اپلیکیشن ساده می‌باشد که بر روی سرور RUN شده است که در آینده به صورت یک سرویس ویندوز ارائه خواهد شد.

نظرات خوانندگان

نویسنده: افشین
تاریخ: ۱۳۹۲/۰۴/۱۵ ۸:۱

یه سؤال دارم که همیشه ذهنم رو مشغول کرده
مگه اینترفیس فقط امضا روال‌ها رو نداره؟ پس یک کلاس نیاز داره که بتونه اون متدها رو پیاده سازی کنه و ما ازش استفاده کنیم
غیر از اینه؟
پس در کد زیر

```
IJobDetail job = JobBuilder.Create<HelloJob>()
```

مجبوریم از اینترفیس به عنوان متغیر استفاده کنیم؟

نویسنده: سیروان عقیفی
تاریخ: ۱۳۹۲/۰۴/۱۵ ۱۲:۴۲

بله به همین صورته، این مطلب رو درباره [اینترفیس](#) و این مطلب رو درباره متدهای [Generic](#) بخونید،
متد Create یک متد Generic است که نام کلاسی رو که اینترفیس IJob و Implement کرده را قبول میکند، و در نهایت مقدار بازگشتی این متد از نوع IJobDetail است.

[Pingback](#) یکی از روش‌های اطلاع رسانی به سایت‌های دیگر در مورد لینک دادن به آن‌ها در سایت خود است. برای مثال من لینکی از یکی از مطالب شما را در متن جاری خودم قرار می‌دهم. سپس به وسیله‌ی ارسال یک ping، در مورد انجام اینکار به شما اطلاع رسانی می‌کنم. حاصل آن عموماً قسمت معروف ping-backs سایت‌ها است. این مورد نیز یکی از روش‌های مؤثر SEO در گرفتن [backlink](#) است و تبلیغ محتوا.

کار کردن با پروتکل Ping-back آنچنان ساده نیست؛ از این جهت که تبادل ارتباطات آن با پروتکل [XML-RPC](#) انجام می‌شود. XML-RPC نیز توسط PHP کارها بیشتر مورد استفاده قرار می‌گیرد؛ بجای استفاده از پروتکل‌های استاندارد وب سرویس‌ها مانند Soap و امثال آن. پیاده‌سازی‌های ابتدایی Pingback نیز مرتبط است به Wordpress معروف که با PHP تهیه شده‌است. در ادامه نگاهی خواهیم داشت به جزئیات پیاده‌سازی ارسال ping-back توسط برنامه‌های ASP.NET.

یافتن آدرس وب سرویس سایت پذیرای Pingback

اولین قدم در پیاده‌سازی Pingback، یافتن آدرسی است که باید اطلاعات مورد نظر را به آن ارسال کرد. این آدرس عموماً به دو طریق ارائه می‌شود:

الف) در هدری به نام x-pingback و یا pingback
ب) در قسمتی از کدهای HTML صفحه به شکل

```
<link rel="pingback" href="pingback server">
```

برای مثال اگر به وبلاگ‌های MSDN دقت کنید، هدر x-pingback را می‌توانید در خروجی وب سرور آن‌ها مشاهده کنید:

Latest Developments in General Purpose GPU Programming with F#



dsyme 23 Apr 2014 4:06 AM

0

RATI
★★★★★

Console HTML CSS Script DOM **Net** Cookies YSlow Pixel P... Refere... Changes F

Clear Persist **All** HTML CSS JavaScript XHR Images Plugins Media Fonts

URL	Status	Domain	Size	Remote IP	Timelin
GET latest-developments-in-gp	200 OK	blogs.msdn.com	19.4 KB	0.0.0.0:80	

Headers Response HTML Cache Cookies

Response Headers [view source](#)

- Cache-Control** no-cache, no-store
- Content-Encoding** gzip
- Content-Length** 19875
- Content-Type** text/html; charset=utf-8
- Date** Sun, 27 Apr 2014 18:41:08 GMT
- Expires** -1
- P3P** CP="ALL IND DSP COR ADM CONo CUR CUSo IVAo IVDo PSA PSD TAI TELo OUR SAMo CNT COM PUR UNI", CP="DSP CUR OTPi IND OTRi ONL FIN", CP="DSP CUR OTPi IND OTRi ONL FIN"
- Pragma** no-cache
- Server** Microsoft-IIS/7.5
- Set-Cookie** msdn=L=1033; domain=.microsoft.com; expires=Tue, 27-May-2014 18:41:08 GMT; path=
- Telligent-Evolution** 5.6.50428.7875
- Vary** Accept-Encoding
- X-AspNet-Version** 2.0.50727
- X-Frame-Options** SAMEORIGIN
- X-Pingback** http://blogs.msdn.com/b/dsyme/pingback.aspx
- X-Powered-By** ASP.NET

همانطور که ملاحظه می‌کنید، نیاز است Response header را آنالیز کنیم.

```
private Uri findPingbackServiceUri()
{
    var request = (HttpWebRequest)WebRequest.Create(_targetUri);
    request.UserAgent = UserAgent;
    request.Timeout = Timeout;
    request.ReadWriteTimeout = Timeout;
    request.Method = WebRequestMethods.Http.Get;
    request.AutomaticDecompression = DecompressionMethods.GZip | DecompressionMethods.Deflate;
    using (var response = request.GetResponse() as HttpWebResponse)
    {
        if (response == null) return null;

        var url = extractPingbackServiceUriFormHeaders(response);
        if (url != null)
            return url;

        if (!isResponseHtml(response))
            return null;

        using (var reader = new StreamReader(response.GetResponseStream()))
        {
            return extractPingbackServiceUriFormPage(reader.ReadToEnd());
        }
    }
}

private static Uri extractPingbackServiceUriFormHeaders(WebResponse response)
{
    var pingUrl = response.Headers.AllKeys.FirstOrDefault(header =>
        header.Equals("x-pingback", StringComparison.OrdinalIgnoreCase) ||
        header.Equals("pingback", StringComparison.OrdinalIgnoreCase));
}
```

```

        return getValidAbsoluteUri(pingUrl);
    }

    private static Uri extractPingbackServiceUriFromPage(string content)
    {
        if (string.IsNullOrEmpty(content)) return null;
        var regex = new Regex(@"(?s)<link\srel=""pingback""\shref=""(.+?)""",
        RegexOptions.IgnoreCase);
        var match = regex.Match(content);
        return (!match.Success || match.Groups.Count < 2) ? null :
        getValidAbsoluteUri(match.Groups[1].Value);
    }

    private static Uri getValidAbsoluteUri(string url)
    {
        Uri absoluteUri;
        return string.IsNullOrEmpty(url) || !Uri.TryCreate(url, UriKind.Absolute, out
        absoluteUri) ? null : absoluteUri;
    }

    private static bool isResponseHtml(WebResponse response)
    {
        var contentTypeKey = response.Headers.AllKeys.FirstOrDefault(header =>
        header.Equals("content-type",
        StringComparison.OrdinalIgnoreCase));
        return !string.IsNullOrEmpty(contentTypeKey) &&
        response.Headers[contentTypeKey].StartsWith("text/html",
        StringComparison.OrdinalIgnoreCase);
    }
}

```

نحوه‌ی استخراج آدرس سرویس Pingback یک سایت را در کدهای فوق ملاحظه می‌کنید.

targetUri، آدرسی است از یک سایت دیگر که در سایت ما درج شده‌است. زمانیکه این صفحه را درخواست می‌کنیم، response.Headers.AllKeys حاصل می‌تواند حاوی کلید x-pingback باشد یا خیر. اگر بلی، همینجا کار پایان می‌یابد. فقط باید مطمئن شد که این آدرس مطلق است و نه نسبی. به همین جهت در متد getValidAbsoluteUri، بررسی بر روی UriKind.Absolute انجام شده‌است.

اگر هدر فاقد کلید x-pingback باشد، قسمت ب را باید بررسی کرد. یعنی نیاز است محتوای HTML صفحه را برای یافتن link rel=pingback بررسی کنیم. همچنین باید دقت داشت که پیش از اینکار نیاز است حتما بررسی isResponseHtml صورت گیرد. برای مثال در سایت شما لینکی به یک فایل 2 گیگابایتی SQL Server درج شده‌است. در این حالت نباید ابتدا 2 گیگابایت فایل دریافت شود و سپس بررسی کنیم که آیا محتوای آن حاوی link rel=pingback است یا خیر. اگر محتوای ارسالی از نوع text/html بود، آنگاه کار دریافت محتوای لینک انجام خواهد شد.

ارسال Ping به آدرس سرویس Pingback

اکنون که آدرس سرویس pingback یک سایت را یافته‌ایم، کافی است ping ایی را به آن ارسال کنیم:

```

public void Send()
{
    var pingUrl = findPingbackServiceUri();
    if (pingUrl == null)
        throw new NotSupportedException(string.Format("{0} doesn't support pingback.",
        _targetUri.Host));

    sendPing(pingUrl);
}

private void sendPing(Uri pingUrl)
{
    var request = (HttpWebRequest)WebRequest.Create(pingUrl);
    request.UserAgent = UserAgent;
    request.Timeout = Timeout;
    request.ReadWriteTimeout = Timeout;
    request.Method = WebRequestMethods.Http.Post;
    request.ContentType = "text/xml";
    request.ProtocolVersion = HttpVersion.Version11;
    makeXmlRpcRequest(request);
    using (var response = (HttpWebResponse)request.GetResponse())
    {
        response.Close();
    }
}

```

```

    }
}

private void makeXmlRpcRequest(WebRequest request)
{
    var stream = request.GetRequestStream();
    using (var writer = new XmlTextWriter(stream, Encoding.ASCII))
    {
        writer.WriteStartDocument(true);
        writer.WriteStartElement("methodCall");
        writer.WriteElementString("methodName", "pingback.ping");
        writer.WriteStartElement("params");

        writer.WriteStartElement("param");
        writer.WriteStartElement("value");
        writer.WriteElementString("string", Uri.EscapeUriString(_sourceUri.ToString()));
        writer.WriteEndElement();
        writer.WriteEndElement();

        writer.WriteStartElement("param");
        writer.WriteStartElement("value");
        writer.WriteElementString("string", Uri.EscapeUriString(_targetUri.ToString()));
        writer.WriteEndElement();
        writer.WriteEndElement();

        writer.WriteEndElement();
        writer.WriteEndElement();
    }
}

```

اینبار `HttpWebRequest` تشکیل شده از نوع `post` است و نه `get`. همچنین مقداری را که باید ارسال کنیم نیاز است مطابق پروتکل XML-RPC باشد. برای کار با XML-RPC در دات نت یا می‌توان از کتابخانه‌ی [Cook Computing's XML-RPC.Net](http://CookComputing's XML-RPC.Net) استفاده کرد و یا مطابق کدهای فوق، دستورات آنرا توسط یک `XmlTextWriter` کنار هم قرار داد و نهایتاً در درخواست `Post` ارسالی درج کرد. در اینجا `sourceUri` آدرس صفحه‌ای در سایت ما است که `targetUri` ایی (آدرسی از سایت دیگر) در آن درج شده‌است. در یک `pinback`، صرفاً این دو آدرس به سرویس دریافت کننده‌ی `pingback` ارسال می‌شوند. سپس سایت دریافت کننده‌ی `ping`، ابتدا `sourceUri` را دریافت می‌کند تا عنوان آنرا استخراج کند و همچنین بررسی می‌کند که آیا `targetUri`، در آن درج شده‌است یا خیر (آیا spam است یا خیر)؟ تا اینجا اگر این مراحل را کنار هم قرار دهیم به کلاس `Pingback` ذیل خواهیم رسید:

[Pingback.cs](#)

نحوه‌ی استفاده از کلاس `Pingback` تهیه شده

کار ارسال `Pingback` عموماً به این نحو است: هر زمانیکه مطلبی یا یکی از نظرات آن، ثبت یا ویرایش می‌شوند، نیاز است `Pingback`‌های آن ارسال شوند. بنابراین تنها کاری که باید انجام شود، استخراج لینک‌های خارجی یک صفحه و سپس فراخوانی متد `Send` کلاس فوق است.

یافتن لینک‌های یک محتوا را نیز می‌توان مانند متد `extractPingbackServiceUriFormPage` فوق، توسط یک `Regex` انجام داد و یا حتی با استفاده از کتابخانه‌ی معروف [HTML Agility Pack](#) :

```

var doc = new HtmlWeb().Load(url);
var linkTags = doc.DocumentNode.Descendants("link");
var linkedPages = doc.DocumentNode.Descendants("a")
    .Select(a => a.GetAttributeValue("href", null))
    .Where(u => !String.IsNullOrEmpty(u));

```