عنوان: **خواندن اطلاعات API فیدبرنر**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۱/۰۵ ۱۱:۳۲:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: xml

مطلبی را در سایت رادیکال 2 در مورد نمایش تعداد خواننده یک فید دیدم که پیاده سازی آن با سی شارپ و xml serialization به صورت زیر است:

```
using System;
using System.Xml;
using System.Xml.Serialization;

namespace Test
{
  /// <summary>
  /// کلاسی جهت نمایش تعداد خواننده فید وبلاگ شما
  /// <example>CFeedBurner data = new CFeedBurner { FeedID = "fhphjt61bueu08k93ehujpu234" };
  /// MessageBox.Show(data.Circulation().ToString());</example>
  /// </summary>
  class CFeedBurner
  {
      /// <summary>
      /// آی دی فید شما زمانیکه به فید برنر لاگین کرده‌اید در تایتل صفحه مربوطه
      /// </summary>
      public string FeedID { get; set; }

      /// <summary>
      /// نگاشت فید به یک کلاس
      /// </summary>
      /// <returns>کلاس متناظر با فید</returns>
      /// <exception cref="Exception">لطفا شماره شناسایی فید را وارد کنید</exception>
      rsp deserializeFromXML()
      {
          if (FeedID == null)
              throw new Exception("لطفا شماره شناسایی فید را وارد کنید");

          XmlSerializer deserializer =
              new XmlSerializer(typeof(rsp));
          using (XmlReader reader = XmlReader.Create(
              string.Format("https://feedburner.google.com/api/awareness/1.0/GetFeedData?id={0}",
FeedID)))
          {
              return (rsp)deserializer.Deserialize(reader);
          }
      }

      /// <summary>
      /// دریافت تعداد خواننده فید
      /// </summary>
      /// <returns>آمار فید</returns>
      /// <exception cref="Exception">اطلاعات فید شما قابل دریافت نیست</exception>
      public int Circulation()
      {
          rsp data = deserializeFromXML();
          if (data == null || data.feed == null || data.feed.Length == 0)
              throw new Exception("اطلاعات فید شما قابل دریافت نیست");

          if (data.feed[0].entry == null || data.feed[0].entry.Length == 0)
              throw new Exception("اطلاعات فید شما قابل پردازش نیست");

          return int.Parse(data.feed[0].entry[0].circulation);
      }
  }
}
```

کلاس rsp از فایل xml فید استاندارد سایت فیدبرنر درست شده است. روش تولید آن‌را قبلا توضیح داده بودم. به سادگی اجرای دو سطر زیر است:

```
xsd.exe GetFeedData.xml
xsd.exe GetFeedData.xsd /c
```

دریافت
فایل‌های کلاس‌های نامبرده شده.

عنوان: **استفاده از Google Analytics API در دات نت فریم ورک**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۰۸ ۲۱:۴۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: xml

بالاخره گوگل کار تهیه API مخصوص ابزار Analytics خود را به پایان رساند و اکنون برنامه نویس‌ها می‌توانند همانند سایر سرویس‌های گوگل از این ابزار گزارشگیری نمایند.
خلاصه کاربردی این API ، دو صفحه تعاریف پروتکل (+) و ریز مواردی (+) است که می‌توان گزارشگیری نمود.
هنوز کتابخانه google-gdata جهت استفاده از این API به روز رسانی نشده است؛ بنابراین در این مقاله سعی خواهیم کرد نحوه کار با این API را از صفر بازنویسی کنیم.
مطابق صفحه تعاریف پروتکل، سه روش اعتبارسنجی جهت دریافت اطلاعات API معرفی شده است که در اینجا از روش ClientLogin که مرسوم‌تر است استفاده خواهیم کرد.
مطابق مثالی که در آن صفحه قرار دارد، اطلاعاتی شبیه به اطلاعات زیر را باید ارسال و دریافت کنیم:

```
POST /accounts/ClientLogin HTTP/1.1
User-Agent: curl/7.15.1 (i486-pc-linux-gnu) libcurl/7.15.1
OpenSSL/0.9.8a zlib/1.2.3 libidn/0.5.18
Host: www.google.com
Accept: */*
Content-Length: 103
Content-Type: application/x-www-form-urlencoded
accountType=GOOGLE&Email=userName@google.com&Passwd=myPasswrd&source=curl-tester-1.0&service=analytics

HTTP/1.1 200 OK
Content-Type: text/plain
Cache-control: no-cache
Pragma: no-cache
Date: Mon, 02 Jun 2008 22:08:51 GMT
Content-Length: 497
SID=DQ...
LSID=DQAA...
Auth=DQAAAG8...
```

در دات نت فریم ورک، این‌کار را به صورت زیر می‌توان انجام داد:

```
string getSecurityToken()
       {
           if (string.IsNullOrEmpty(Email))
               throw new NullReferenceException("Email is required!");

           if (string.IsNullOrEmpty(Password))
               throw new NullReferenceException("Password is required!");

           WebRequest request = WebRequest.Create("https://www.google.com/accounts/ClientLogin");
           request.Method = "POST";

           string postData = "accountType=GOOGLE&Email=" + Email + "&Passwd=" + Password +
"&service=analytics&source=vahid-testapp-1.0";
           byte[] byteArray = Encoding.ASCII.GetBytes(postData);

           request.ContentType = "application/x-www-form-urlencoded";
           request.ContentLength = byteArray.Length;

           using (Stream dataSt = request.GetRequestStream())
           {
               dataSt.Write(byteArray, 0, byteArray.Length);
           }

           string auth = string.Empty;
           using (WebResponse response = request.GetResponse())
           {
               using (Stream dataStream = response.GetResponseStream())
               {
                   using (StreamReader reader = new StreamReader(dataStream))
                   {
                       string responseFromServer = reader.ReadToEnd().Trim();
```

```
                        string[] tokens = responseFromServer.Split('\n');
                        foreach (string token in tokens)
                        {
                            if (token.StartsWith("SID="))
                                continue;

                            if (token.StartsWith("LSID="))
                                continue;

                            if (token.StartsWith("Auth="))
                            {
                                auth = token.Substring(5);
                            }
                            else
                            {
                                throw new AuthenticationException("Error authenticating Google user " +
Email);
                            }
                        }
                    }
                }
            }

            return auth;

        }
```

همانطور که ملاحظه می‌کنید به آدرس https://www.google.com/accounts/ClientLogin ، اطلاعات postData با متد POST ارسال شده (دقیقا مطابق توضیحات گوگل) و سپس از پاسخ دریافتی، مقدار نشانه Auth را جدا نموده و در ادامه عملیات استفاده خواهیم کرد. وجود این نشانه در پاسخ دریافتی به معنای موفقیت آمیز بودن اعتبار سنجی ما است و مقدار آن در طول کل عملیات باید نگهداری شده و مورد استفاده مجدد قرار گیرد.
سپس مطابق ادامه توضیحات API گوگل باید لیست پروفایل‌هایی را که ایجاد کرده‌ایم پیدا نمائیم:

```
string getAvailableProfiles(string authToken)
        {
            return fetchPage("https://www.google.com/analytics/feeds/accounts/default", authToken);
        }
```

متد fetchPage را از پیوست این مقاله می‌توانید دریافت نمائید. خروجی یک فایل xml است که با انواع و اقسام روش‌های موجود قابل آنالیز است، از کتابخانه‌های XML دات نت گرفته تا Linq to xml و یا روش serialization که من روش آخر را ترجیح می‌دهم.
مرحله بعد، ساخت URL زیر و دریافت مجدد اطلاعات مربوطه است:

```
string url =
string.Format("https://www.google.com/analytics/feeds/data?ids={0}&metrics=ga:pageviews&start-
date={1}&end-date={2}", id, from, to);
            return fetchPage(url, auth);
```

و سپس آنالیز اطلاعات xml دریافتی، جهت استخراج تعداد بار مشاهده صفحات یا pageviews استفاده شده در این مثال. لیست کامل مواردی که قابل گزارشگیری است، در صفحه Dimensions & Metrics Reference گوگل ذکر شده است.

فایل‌های کلاس‌های مورد استفاده را از اینجا دریافت نمائید.

مثالی در مورد نحوه استفاده از آن:

```
CGoogleAnalytics cga = new CGoogleAnalytics
                                        {
                                            Email = "username@gmail.com",
                                            Password = "password",
                                            From = DateTime.Now.Subtract(TimeSpan.FromDays(1)),
                                            To = DateTime.Now.Subtract(TimeSpan.FromDays(1))
                                        };
```

```
            List<CGoogleAnalytics.SitePagePreviews> pagePreviews =
                cga.GetTotalNumberOfPageViews();

            foreach (var list in pagePreviews)
            {
                //string site = list.Site;
                //int pw = list.PagePreviews;
            }
```

نظرات خوانندگان

نویسنده: نیما
تاریخ: ۱۳۸۸/۰۲/۱۲ ۱۰:۵۸:۰۰

سلام استاد نصیری
با تشکر از مطلب مفید شما.آیا میشه مثلا اون قسمت Map گزارش گوگل رو گذاشت توی سایت؟
ممنون از شما

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۱۲ ۱۲:۳۹:۰۰

سلام،
API فوق امکان استخراج اطلاعات مربوط به کشورهای مختلف و بازدید کننده‌های مختلف را نیز می‌دهد. (به صفحه Dimensions & Metrics Reference مراجعه کنید)
پس از استخراج داده‌ها، رسم نمودار و نقشه با خودتان است. فقط اطلاعات و آمار خام را به شما می‌دهد.

عنوان: **نگاشت JSON به کلاس‌های معادل آن**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۵/۱۷ ۲۲:۳۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: xml

یکی از مواردی که عموما در برنامه نویسی با آن سر و کار داریم، parse اطلاعات با فرمت‌های مختلف است. از CSV تا XML تا ... JSON .
در مورد کار با XML در دات نت فریم ورک، فضاهای نام مرتبط زیادی وجود دارند؛ برای مثال System.Xml.Linq و System.Xml . همچنین یک روش دیگر هم برای کار با اطلاعات XML ایی در دات نت وجود دارد. می‌شود کلاس معادل یک فایل XML را تولید و سپس اطلاعات آن‌را به این کلاس نگاشت کرد. اطلاعات بیشتر : (^). این برنامه کار خود مایکروسافت است.
در مورد JSON از دات نت سه و نیم به بعد کارهایی صورت گرفته مانند : (^). اما آنچنان دلچسب نیست. جهت رفع این خلاء کتابخانه‌ی سورس باز و بسیار کاملی در این زمینه به نام JSON.NET تهیه شده که از این آدرس قابل دریافت است: (^)
و خبر خوب اینکه امکان تهیه کلاس‌های معادل اطلاعات JSON ایی هم مدتی‌است توسط برنامه نویس‌های مستقل تهیه شده است. یا می‌توان از امکانات توکار دات نت استفاده کرد یا از کتابخانه‌هایی مانند JSON.NET یا از هیچکدام! می‌توان یک راست کل اطلاعات JSON ایی دریافتی را به یک یا چند کلاس معادل آن نگاشت کرد:
پروژه سورس باز JSON C# Class Generator
و یا یک ابزار آنلاین مشابه: json2csharp

نظرات خوانندگان

نویسنده: shahin kiassat
تاریخ: ۱۳۹۰/۰۵/۱۸ ۰۰:۲۸:۰۷

آقای نصیری من فقط وقتی که می خواهم با jQuery یک AJAX Post بزنم از JSON استفاده می کنم که آن هم خیلی عملا در گیرش نیستم (از JSON.Stringfy که معرفی کردید استفاده می کنم.)
JSON چه کاربردها و استفاده هایی دارد ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۵/۱۸ ۰۰:۵۵:۱۸

منهای کاربردهای Ajax ایی، هستند برنامه‌هایی که از JSON برای تبادل اطلاعات استفاده می‌کنند (این وب سرویس‌های غنی دات نت که به این سهولت در زبان‌های دیگر در دسترس نیست). مثلا برنامه utorrent یک API مبتنی بر JSON دارد (utorrent web-api). به این ترتیب می‌شود به آن فایل اضافه کرد، کم کرد، درخواست دریافت داد، آمار درصد دریافت فایل‌ها رو گرفت و غیره.
اینجا است که نیاز پیدا می‌کنید بتونید JSON رو دقیق Parse کنید.
یا مثلا گوگل یک سری API خاص خودش را دارد و بعضی از این‌ها فقط خروجی JSON دارند: google data

نویسنده: hosnoddinov
تاریخ: ۱۳۹۰/۰۵/۱۸ ۰۸:۰۰:۴۱

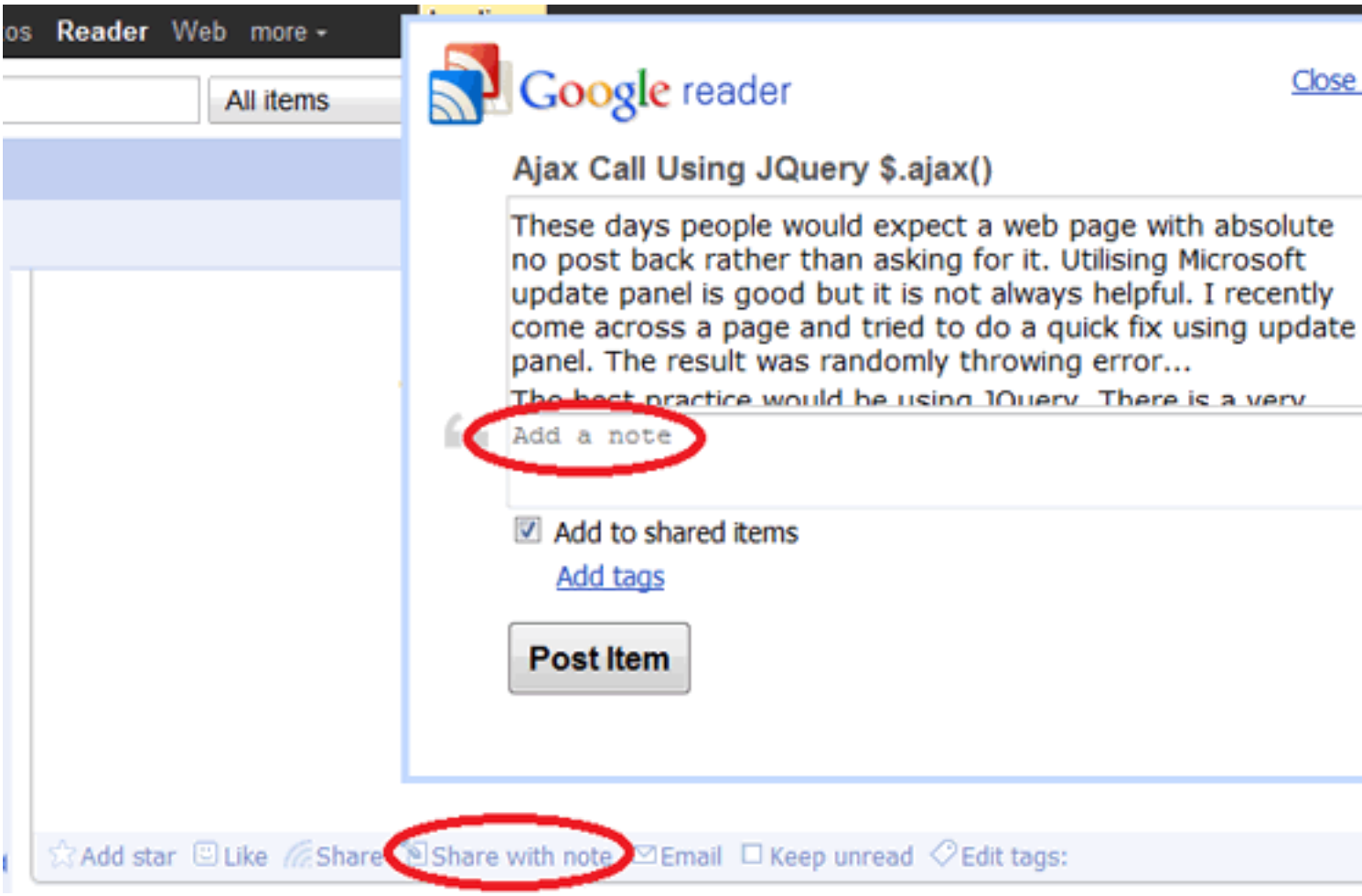
ممنون
استفاده کردیم
(;

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۵/۱۸ ۱۶:۳۰:۲۲

همچنین Raven DB هم مبتنی بر JSON است.

عنوان: **گوگل ریدر و افزودن توضیحات**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۷/۰۱ ۲۳:۰۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: xml

اگر به گوگل ریدر دقت کرده باشید، دو گزینه‌ی به اشتراک گذاری دارد: share و share with note .



اگر گزینه‌ی share with note را انتخاب کرده و توضیحی را ارسال یا اضافه کنیم، این توضیحات، به فید از نوع Atom اشتراک‌ها هم اضافه می‌شود. مثلا:

```
<?xml version="1.0"?>
<feed xmlns:media="http://search.yahoo.com/mrss/"
      xmlns:gr="http://www.google.com/schemas/reader/atom/"
   xmlns:idx="urn:atom-extension:indexing"
   xmlns="http://www.w3.org/2005/Atom"
   idx:index="no"
   gr:dir="ltr">

 ...

  <entry gr:crawl-timestamp-msec="1316627782108">
    ...
    <gr:annotation>
```

```
      <content type="html">text-text-text</content>
      <author>
        <name>Vahid</name>
      </author>
    </gr:annotation>
 ...
  </entry>

    ...

</feed>
```

این افزونه استاندارد نیست و همانطور که در قسمت xmlns:gr اطلاعات فوق مشخص است، در فضای نام http://www.google.com/schemas/reader/atom/ معنا پیدا می‌کند. از دات نت سه و نیم به بعد هم کلاسی جهت خواندن فیدهای استاندارد وجود دارد (تعریف شده در فضای نام System.ServiceModel.Syndication). اما چگونه می‌توان این افزونه‌ی غیر استاندارد را با کمک امکانات توکار دات نت خواند؟

روش کار با استفاده از ElementExtensions هر آیتم یک فید است؛ به صورت زیر :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.ServiceModel.Syndication;
using System.Xml;
using System.Xml.Linq;

namespace Linq2Rss
{
    public class RssEntry
    {
        public string Title { set; get; }
        public string Description { set; get; }
        public string Link { set; get; }
        public DateTime PublicationDate { set; get; }
        public string Author { set; get; }
        public string BlogName { set; get; }
        public string BlogAddress { set; get; }
        public string Annotation { set; get; }
    }

    public static class AtomReader
    {
        private static string getAtomAnnotation(this SyndicationElementExtensionCollection items)
        {
            if (!items.Any()) return string.Empty;
            var item = items.Where(x => x.OuterName.ToLowerInvariant() ==
"annotation").FirstOrDefault();
            if (item == null) return string.Empty;

            var element = item.GetObject<XElement>();
            var content = element.Element("{http://www.w3.org/2005/Atom}content");
            return content == null ? string.Empty : content.Value;
        }

        public static IList<RssEntry> GetEntries(string feedUrl)
        {
            using (var reader = XmlReader.Create(feedUrl))
            {
                var feed = SyndicationFeed.Load(reader);
                if (feed == null) return null;

                return feed.Items.Select(x =>
                    new RssEntry
                    {
                        Title = x.Title.Text,
                        Author = x.Authors.Any() ? x.Authors.First().Name : string.Empty,
                        Description = x.Content == null ? string.Empty :
((TextSyndicationContent)x.Content).Text,
                        Link = x.Links.Any() ? x.Links.First().Uri.AbsoluteUri : string.Empty,
                        PublicationDate = x.PublishDate.UtcDateTime,
                        BlogName = x.SourceFeed.Title.Text,
```

```
                        BlogAddress = x.SourceFeed.Links.Any() ?
x.SourceFeed.Links.First().Uri.AbsoluteUri : string.Empty,
                        Annotation = x.ElementExtensions.getAtomAnnotation()

                    }).ToList();
            }
        }
    }
}
```

در این مثال به کمک متد الحاقی getAtomAnnotation، مجموعه‌ی SyndicationElementExtensionCollection هر آیتم یک فید بررسی شده، در بین این‌ها، موردی که از نوع annotation باشد انتخاب و سپس content آن استخراج می‌گردد.

نکته‌ای دیگر:

اکثر کلاس‌های موجود در فضاهای نام مرتبط با XML در دات نت امکان خواندن اطلاعات را از یک Uri هم دارند؛ مانند مثال فوق و متد XmlReader.Create بکارگرفته شده در آن. اما اگر بخواهیم حین خواندن اطلاعات، یک پروکسی را نیز به پروسه جاری اضافه کنیم، به نظر خاصیت یا متدی جهت انجام اینکار وجود ندارد. برای رفع این مشکل می‌توان یک پروکسی سراسری را تعریف کرد. تنها کافی است خاصیت System.Net.WebRequest.DefaultWebProxy مقدار دهی شود. پس از آن به صورت خودکار بر روی کل برنامه تاثیر خواهد گذاشت.

عنوان: آموزش فایرباگ - #3 JavaScript Development
نویسنده: احمد احمدی
تاریخ: ۱۳۹۱/۰۴/۳۱ ۲:۲۱
آدرس: www.dotnettips.info
برچسب‌ها: JavaScript, xml, FireBug

توابع توسعه جاوا اسکریپت در فایرباگ به دو بخش تقسیم می‌شوند :
توابع خط فرمان - Command Line API
توابع کنسول - Console API

توابع خط فرمان توابعی هستند که فقط در خط فرمان قابل استفاده هستند و توابع کنسول هم توابعی هستند که خارج از محیط خط فرمان (، در بین کدهای جاوا اسکریپت برنامه) هم قابل استفاده هستند .
در این قسمت توابع خط فرمان را بررسی خواهیم کرد و در قسمت بعدی با توابع کنسول آشنا خواهیم شد .

توابع خط فرمان - Command Line API :

این توابع حدود 14 تا هستند که بوسیله آنها می‌توانیم در حین اجرای برنامه تست‌های مختلفی انجام داده یا اطلاعاتی از قسمت‌های مختلف بدست آوریم .
توجه : برای همراه شدن با تست‌های انجام شده در این مقاله می‌توانید کد صفحه‌ی زیر را ذخیره کنید و برای اجرای کدها ، آن‌ها را در قسمت خط فرمان (در تب کنسول) قرار بدهید و دکمه‌ی Run (یا Ctrl + Enter) را بزنید .

```
<div id="first" class="content">Content1 with css class and id</div>
<div class="content">
    Content2 with css class
    <a class="links" href="#">Link1</a>
    <a href="#">Link2</a>
</div>
<div>
    Content3 without css class and id
</div>
<input type="button" onclick="myFunc()" value="Run myFunc" />
<input type="text" id="myInput" />

<script type="text/javascript">
    function myFunc() {
        loop(1000);
        loop(50000);
    }
    function loop(number) {
        for (var i = 0; i < number; i++) { }
    }
</script>
```

قبل از توضیح این توابع ، یک مثال ساده می‌زنیم :
فرض کنید می‌خواهید همه‌ی المنت هایی که با یک selector مطابقت دارند را مشاهده کنید . (آرایه ای از المنت‌ها دریافت کنید)

```
$$("div.content");
```

نتیجه :

```
>>> $$("div.content");
[ div#first.content, div.content ]
```

می خواهید یکی از توابع جاوا اسکریپت برنامه را اجرا و آن را از لحاظ سرعت ، تعداد فراخوانی شدن و ... بررسی کنید .

```
profile("myFunc Testing");
myFunc();
profileEnd();
```

نتیجه :

```
>>> profile("myFunc Testing"); myFunc(); profileEnd();
```

myFunc Testing (0.291ms, 3 calls)

Function	Calls	Percent	Own Time	Time	Avg	Min	Max	File
loop	2	95.19%	0.277ms	0.277ms	0.139ms	0.123ms	0.154ms	1.htm (line 26)
myFunc	1	4.81%	0.014ms	0.291ms	0.291ms	0.291ms	0.291ms	1.htm (line 22)

اکنون با همه‌ی توابع خط فرمان آشنا می‌شویم :

$(id)

معادل دستور document.getElementById است که یک المنت با id داده شده بر می‌گرداند .

```
$("first");
```

نتیجه :

```
>>> $("first")
<div id="first" class="content">
```

$$(selector)

آرایه ای از المنت‌های مطابق با selector داده شده بر می‌گرداند .

```
$$("div.content")
```

نتیجه :

```
>>> $$("div.content")
[ div#first.content, div.content ]
```

به تفاوت دو دستور توجه کنید . خروجی دستور اول ، یک المنت است و خروجی دستور دوم یک آرایه از المنت که بین [و] قرار گرفته اند .

برای آشنایی بیشتر با CSS Seletorها به این لینک مراجعه کنید : http://www.w3.org/TR/css3-selectors

$x(xPathExpression)

آرایه ای از المنت هایی را بر می‌گرداند که با xPath داده شده مطابقت داشته باشند .

```
var objects = $x("html/body/div[2]/a")

for(var i = 0; i < objects.length; i++) {
   console.log(objects[i]);
}
```

نتیجه :

```
>>> var objects = $x("html/body/div[2]/a")  for(var ...cts.length; i++) {
console.log(objects[i]); }
<a class="links" href="#">
<a href="#">
```

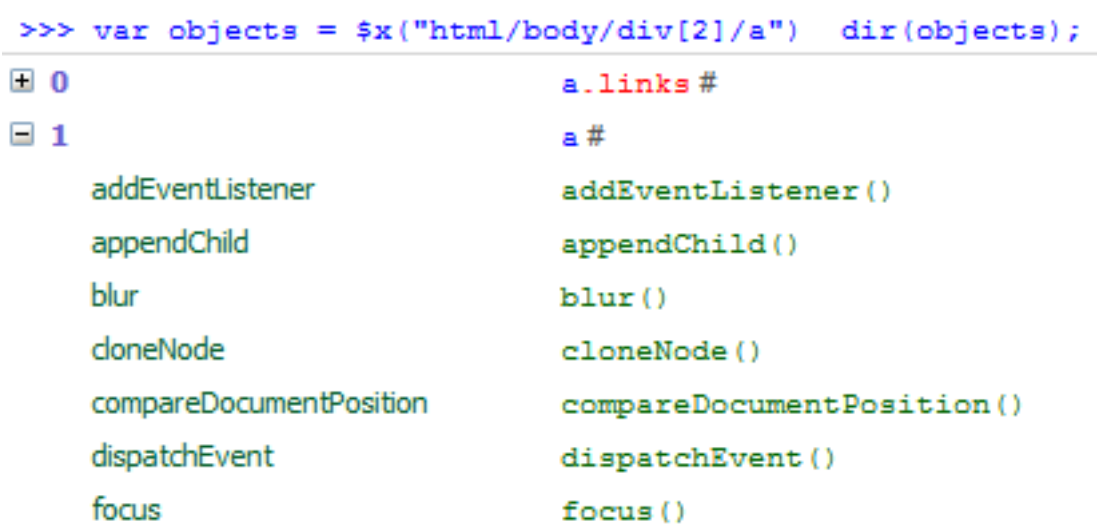
برای آشنایی بیشتر با عبارات xPath به این لینک مراجعه کنید : http://www.w3schools.com/xpath

dir(object)

تمام خصوصیات شیء ارسال شده را لیست می‌کند .

```
var objects = $x("html/body/div[2]/a")

dir(objects);
```

نتیجه :

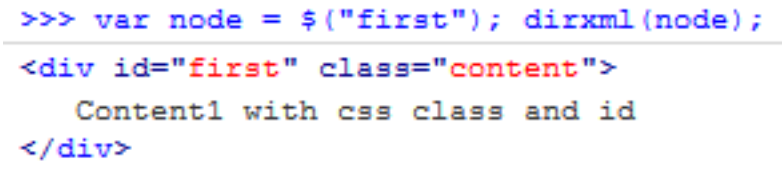


(dirxml(node

سورس یک المنت را بصورت درختواره (tree) پرینت می‌کند . همچنین با کلیک بروی هر node ، فایرباگ آن node را در تب html نمایش می‌دهد .

```
var node = $("first");
dirxml(node);
```

نتیجه :



توجه کنید که این دستور فقط یک node دریافت می‌کند . برای همین اگر از دستور $$("first#") استفاده می‌کنید ، چون این دستور یک آرایه بر می‌گرداند ، باید اولین عضو آرایه را دریافت و ارسال کنید .
یعنی :

```
var node = $$("#first")[0];
dirxml(node);
```

()clear

این دستور محیط console را خالی می‌کند . عملکرد این دستور معادل کلیک دکمه‌ی Clear (در بالا - چپ تب کنسول) است .

([inspect(object[,tabName

توسط این دستور می‌توانید یک شیء را در مناسب‌ترین تب فایرباگ یا یکی از تب‌های مورد نظر خود ، Inspect کنید .

```
var node = $("first");

inspect(node); // inspect in html tab
inspect(node,'dom'); // inspect in dom tab
```

(keys(object

آرایه ای از "نام" تمام خصوصیات شیء ارسال شده بر می‌گرداند .

```
var obj = $("first");
keys(obj)
```

(values(object

آرایه ای از "مقدار" تمام خصوصیات شیء ارسال شده بر می‌گرداند .

```
var obj = $("first");
values(obj)
```

(debug(fn) and undebug(fn

این متدها یک BreakPoint در ابتدای تابع مشخص شده اضافه/حذف می‌کنند . (در تب Script) . به همین ترتیب هنگامی که تابع مورد نظر فراخوانی شود ، در نقطه ای که BreakPoint قرار داده شده توقف خواهد کرد .
البته می‌شود BreakPoint را دستی هم قرار داد . در اصل این تابع ، این عملیات را ساده‌تر می‌کند .

```
debug(myFunc);
myFunc();
undebug(myFunc);
```

(monitor(fn) and unmonitor(fn

این متدها برای فعال/غیرفعال کردن Logging فراخوانی‌های یک تابع استفاده می‌شوند .
در حالت عادی برای پی بردن به اینکه یک تابع اجرا می‌شود یا نه ، در تابع مورد نظر یک alert قرار می‌دهیم و تست می‌کنیم . که این روش در برنامه برنامه‌های بزرگ صحیح نیست . زیرا در این حالت باید بین حجم زیادی کد به دنبال تابع مورد نظر بگردیم و سپس alert را قرار بدهیم و بعد اطمینان از صحت عملکرد تابع مجدد آن را حذف کرد ، که با اتلاف زمان و به خطر انداختن کدها همراه است .
اما با استفاده از این متدها ، تنها نیاز به داشتن اسم تابع داریم (و نه مکان تابع در کدهای برنامه) .
تست monitor :

```
monitor(myFunc);
// now click on "Run myFunc" button
```

تست unmonitor :

```
unmonitor(myFunc);
// now click on "Run myFunc" button
```

([monitorEvents(object[, types]) and unmonitorEvents(object[, types

این متدها عملیات Event Logging برای یک شیء را فعال/غیرفعال می‌کنند . در کنار شیء مورد نظر ، می‌توان نوع رویداد را هم به

متد ارسال کرد . در این صورت عملیات Logging فقط برای همان گروه رویداد/رویداد ، فعال/غیرفعال می‌شود .
منظور از گروه رویداد ، مجموعه رویدادهای یک شیء است . مثلا mousedown , moseover , mousemove , ... در گروه mouse قرار می‌گیرند . یعنی می‌توانید با ارسال کلمه‌ی mouse فقط رویدادهای mouse را تحت نظر بگیرید یا اینکه فقط یک رویداد را مشخص کنید ، مثل mousedown .
راه ساده‌تر فعال کردن Event Logging ، رفتن به تب Html ، راست کلیک کردن بروی المنت مورد نظر و فعال کردن گزینه‌ی Log Events می‌باشد .

```
var obj = $("myInput");
monitorEvents(obj,'keypress');
```

نتیجه پس از فشردن چند دکمه‌ی کیبورد در myInput :

keypress charCode=0, keyCode=38
keypress charCode=0, keyCode=38
keypress charCode=0, keyCode=38
keypress charCode=0, keyCode=39
keypress charCode=0, keyCode=13
keypress charCode=0, keyCode=13
keypress charCode=115, keyCode=0
keypress charCode=100, keyCode=0
keypress charCode=97, keyCode=0
keypress charCode=97, keyCode=0

توضیحات بیشتر : http://getfirebug.com/wiki/index.php/MonitorEvents

profile([title]) and profileEnd()

این متدها ، JavaScript Profiler را فعال/غیرفعال می‌کنند . هنگام فعال کردن می‌توانید یک عنوان هم برای پروفایل مشخص کنید . در قسمت قبلی مقاله در مورد این قابلیت توضیحاتی ارائه شد .
سه را برای اجرای Profiler وجود دارد :
1 - کلیک بروی دکمه‌ی Profiler در بالای تب کنسول .
2 - استفاده از کد console.profile("ProfileTitle") در کدهای جاوا اسکریپت .
3 - استفاده از متد profile("Profile Title") در خط فرمان .

```
profile("myFunc Testing");
myFunc();
profileEnd();
```

نتیجه :

```
>>> profile("myFunc Testing"); myFunc(); profileEnd();
```

myFunc Testing (0.335ms, 3 calls)

Function	Calls	Percent	Own Time	Time	Avg	Min	Max	File
loop	2	95.22%	0.319ms	5.545ms	2.773ms	2.662ms	2.883ms	1.htm (line 26)
myFunc	1	4.78%	0.016ms	5.561ms	5.561ms	5.561ms	5.561ms	1.htm (line 22)

ستون‌های Profiler :

Function : نام تابع اجرا شده .
Calls : تعداد دفعات فراخوانی تابع .
Percent : زمان اجرای تابع در زمان کل ، به درصد .
Own Time : زمان اجرای تابع به تنهایی . برای مثال در کد ما ، زمان اجرای تابع myFunc به تنهایی تقریبا صفر است زیرا عمیاتی در خود انجام نمی‌دهد و زمان صرف شده در این تابع ، برای اجرای 2 با تابع loop است . این زمان (Own Time) زمان اجرای تابع ، منهای زمان صرف شده برای فراخوانی توابع دیگر است .
Time : زمان اجرای تابع از نقطه‌ی آغاز تا پایان . مجموع زمان اجرای خود تابع به همراه زمان اجرای توابع فراخوانی شده . در کد ما ، این زمان ، مجموع زمان اجرای خود تابع به همراه دو بار فراخوانی تابع loop است .
Avg : میانگین زمان اجرای هربار تابع . فرمول : Avg = Time / Calls
Min & Max : حداقل و حداکثر زمان اجرای تابع .
File : نام فایل و شماره خطی که تابع در آن قرار دارد .

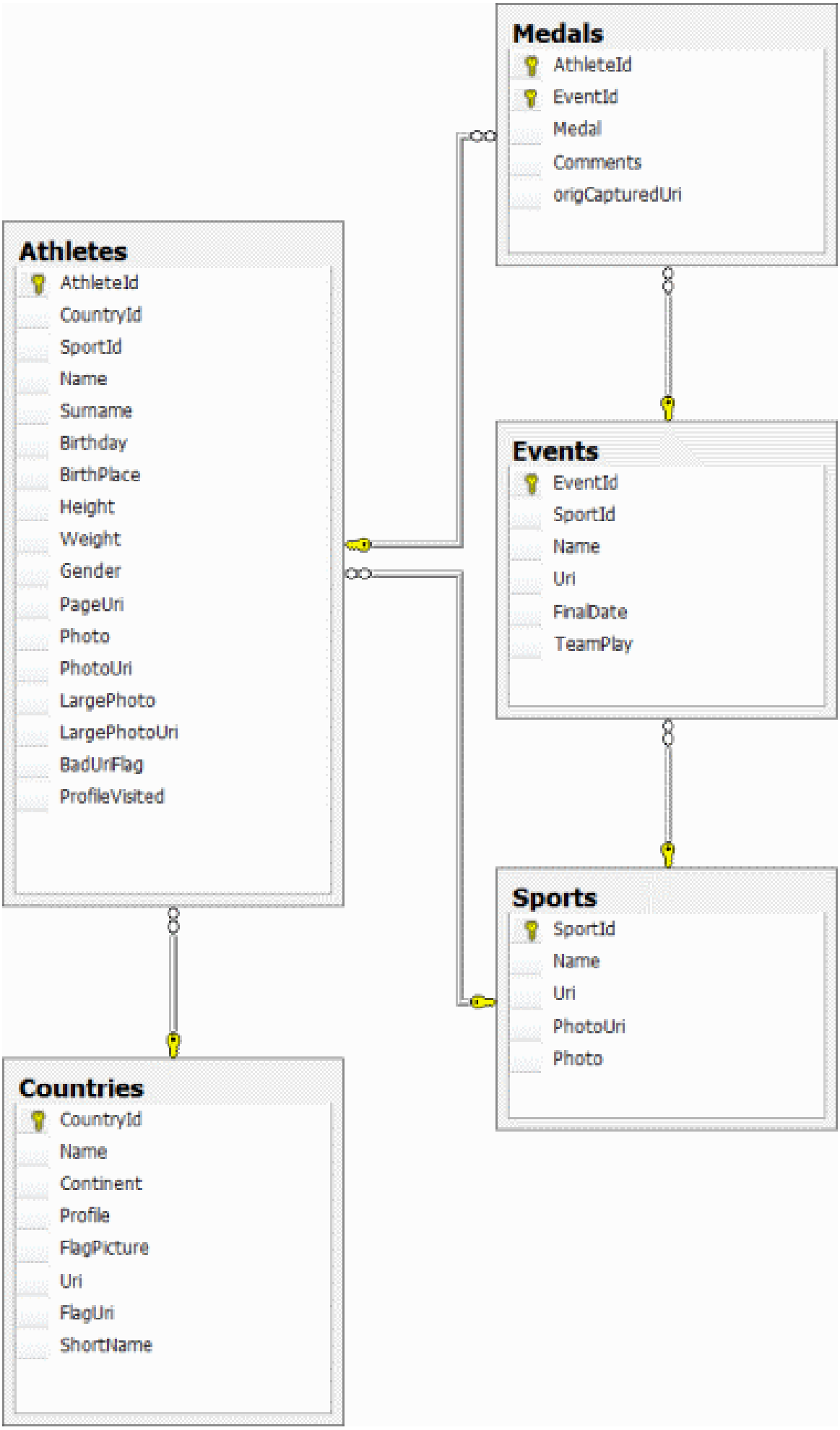
در قسمت بعد با توابع کنسول آشنا خواهیم شد .

منابع :
http://michaelsync.net/2007/09/15/firebug-tutorial-commandline-api
http://michaelsync.net/2007/09/09/firebug-tutorial-logging-profiling-and-commandline-part-i
http://getfirebug.com/wiki/index.php/Console_Panel
http://getfirebug.com/wiki/index.php/MonitorEvents

Packtpub.Firebug.1.5.Editing.Debugging.and.Monitoring.Web.Pages.Apr.2010

عنوان: **استخراج اطلاعات از صفحات وب با کمک HtmlAgilityPack**
نویسنده: ایمان عبیدی
تاریخ: ۱۳۹۱/۰۶/۰۳ ۲۳:۵۸
آدرس: www.dotnettips.info
گروه‌ها: XPATH, xml, Html Agility Pack, FireBug

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 استفاده کردم . طراحی دیتابیس نهایی به این صورت شد


Medals
AthleteId
EventId
Medal
Comments
origCapturedUri
Athletes
AthleteId
CountryId
SportId
Name
Surname
Birthday
BirthPlace
Height
Weight
Gender
PageUri
Photo
PhotoUri
LargePhoto
LargePhotoUri
BadUrlFlag
ProfileVisited
Events
EventId
SportId
Name
Uri
FinalDate
TeamPlay
Sports
SportId
Name
Uri
PhotoUri
Photo
Countries
CountryId
Name
Continent
Profile
FlagPicture
Uri
FlagUri
ShortName

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 فورا با استثنای زیر مواجه می‌شوید

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

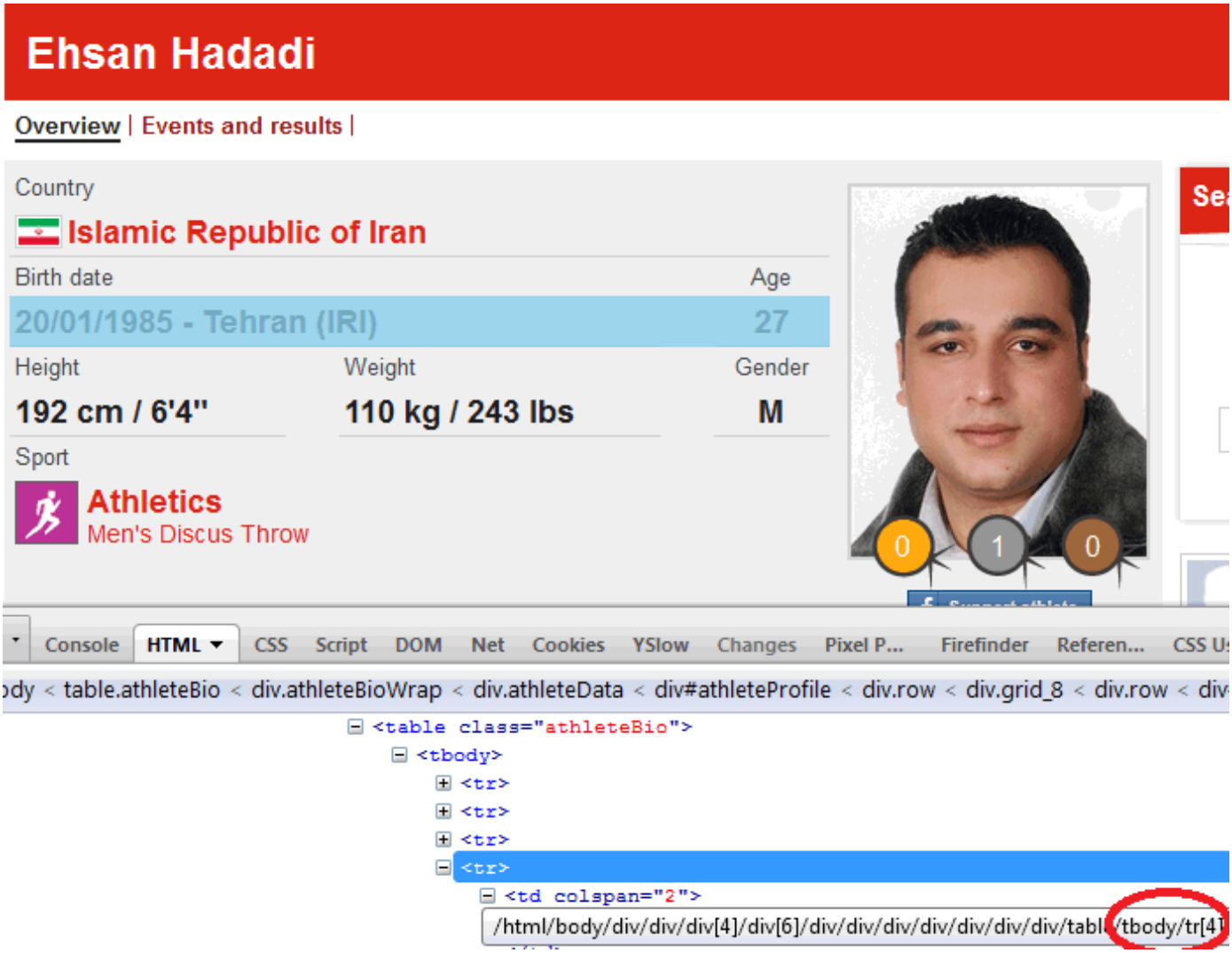
راه حل ساده اینه که این کتابخونه رو با کمک NuGet نصب کنید

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

و از اینجا به بعد با کدی مثل این میتونید از کلاس HtmlDocument و مشابه XmlDocument ولی بدون ارور استفاده کنید.
مثلا 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(" ", "").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 HttpWebRequest بنویسید و توی خوده HAP متاسفانه چنین تابعی توکار نشده نکته اصلی هم پیدا کردن محل دقیق اطلاعاته که با ابزاری مثل Firebug خیلی راحت‌تر 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 ازش استفاده کنید.



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

نظرات خوانندگان

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۶/۰۴ ۱۲:۵۳

این روش برای استخراج مطالب همین سایت خیلی مفیده!

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

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

نویسنده: جمال
تاریخ: ۱۳۹۱/۰۶/۰۴ ۱۴:۱۹

Crawler.GetXHtmlFromUri
شی Crawler ازچه نوعیه؟موقع اجرا خطا میگیره؟

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

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: ۱۳۹۱/۰۶/۰۴ ۱۵:۴۳

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: ۱۳۹۱/۰۶/۰۴ ۱۶:۴۹

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: ۱۳۹۱/۰۶/۰۴ ۲۰:۴۶

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 خیله مهمه وگرنه ارور The remote server returned an error: (409) Conflict رو میده.
من با همین تابع یک سایت فارسی رو چک کردم و اروری نداد و متن فارسی قابل کوئری گرفتن بود.
کامل‌تر 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";
            request.AutomaticDecompression = 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: ۱۳۹۱/۰۶/۰۵ ۹:۳۷

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

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

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.
توبه 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: ۱۳۹۲/۰۲/۰۲ ۲۰:۵۸

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

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

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: ۱۳۹۲/۰۳/۲۳ ۱۰:۳۸

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: ۱۳۹۲/۰۳/۲۳ ۱۱:۲۹

- 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: ۱۳۹۲/۰۳/۲۳ ۱۲:۱۰

ببخشین که دوباره مزاحمتون شدم: وقتی من کد بالا رو در page load کپی میکنم زیر بعضی کلمات مانند , Crawler
xhtml,HtmlNode,Parse, Birthday, LargePhotoUri, GetAttributeValueخط قرمز میکشه ! چرا؟ آیا HtmlDocument 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 می‌کنید .
- 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 ۹:۶

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 فرمم هم زیر این کلمه GetXHtmlFromUri
مینویسه
generate metod stub for 'GetXHtmlFromUri in "crawel

وقتی من متد GetXHtmlFromUri را در کلاس crawel تعریف کردم جرا 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 سی‌شارپ

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 می‌توان صفحه را خواند مثال: **http://website.com?log=1731004**
سایت برای هر بازدید یک لاگ جدید می‌نویسد یعنی هر درخواست جدید در صورتی که لاگ معتبر نباشد به صفحه دیگر ارسال می‌کند.حال اطلاعات این صفحه را چطور می‌توان 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 نهایتا خواندن نودهای HTML نهایی. موردی که عنوان کردید نیاز به آنالیز همزمان هدرهای دریافتی به همراه جاوا اسکریپت سایت مورد نظر داره.

عنوان: **گرفتن خروجی XML از جداول در SQL Server 2012**
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۰۹/۱۹ ۰:۰
آدرس: www.dotnettips.info
گروه‌ها: SQL Server, xml, SQL Server 2012, T-SQL

فرض کنید که می‌خواهیم خروجی از جدول خود را به صورت XML نمایش یا از طریق وب سرویس در برنامه مان استفاده نماییم. اولین راهی که به ذهنمان می‌رسد خودمان رشته xml را با حلقه ای ایجاد نماید یا استفاده از فضای نام System.Xml و کلاس‌های نوشته شده برای این کار . اما خود Sql Server امکانات ویژه ای برای کار با ساختار xml مهیا نموده که براحتی می‌توانید خروجی xml از داده هایتان ایجاد نمایید.

برای این کار از عبارت For XML در Select می‌توان استفاده نمود. برای مثال برای بدست آوردن ساختار ساده از For Xml Auto استفاده نمایید

```
SELECT BusinessEntityID, PersonType, Title, FirstName, MiddleName, LastName
FROM Person
WHERE BusinessEntityID = 10001
FOR XML AUTO
```

که خروجی بصورت **node attribute** زیر می‌باشد:

```
<Person.Person BusinessEntityID="10001" PersonType="IN" FirstName="Carolyn" LastName="Alonso" />
```

اما اگر بخواهیم خروجی به صورت node Elements باشد کافیست از پارامتر Elements استفاده نمایید

```
SELECT BusinessEntityID, PersonType, Title, FirstName, MiddleName, LastName
FROM Person
WHERE BusinessEntityID = 10001
FOR XML AUTO, ELEMENTS
```

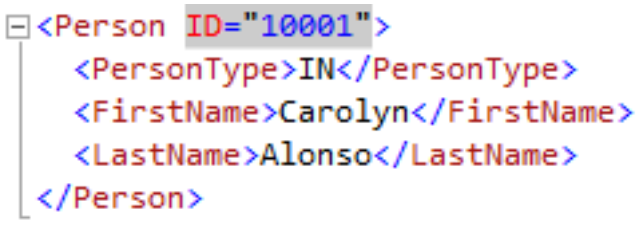
خروجی بصورت زیر می‌باشد:

```
<Person.Person>
  <BusinessEntityID>10001</BusinessEntityID>
  <PersonType>IN</PersonType>
  <FirstName>Carolyn</FirstName>
  <LastName>Alonso</LastName>
</Person.Person>
```

اگر بخواهیم node attributes و node elements با هم ترکیب کنیم بصورت زیر عمل می‌کنیم:

```
SELECT BusinessEntityID AS '@ID', PersonType, Title,  FirstName, MiddleName, LastName
FROM Person
WHERE BusinessEntityID = 10001
FOR XML ELEMENTS
```

خروجی بصورت زیر است:



حال می‌خواهیم همه nodeها را یک node ریشه قرار دهیم برای این کار از پارامتر ROOT در کنار AUTO به صورت زیر استفاده نمایید:

```
SELECT *
FROM Person
WHERE BusinessEntityID = 15291
FOR XML AUTO , ROOT('Persons')
```

اما اگر بخواهیم نام جدول را با نام دلخواه خود تغییر دهیم از پارامتر PATH به جای AUTO به صورت زیر استفاده نمایید:

```
SELECT *
FROM Person
WHERE BusinessEntityID = 15291
FOR XML PATH('P') , ROOT('Persons')
```

نظرات خوانندگان

نویسنده: احسان میرسعیدی
تاریخ: ۱۳۹۱/۰۹/۱۹ ۰:۲۶

بسیار تکنیک کارامدی بود. واقعا متشکرم

نویسنده: فرهاد فرهمندخواه
تاریخ: ۱۳۹۱/۰۹/۱۹ ۷:۲۹

سلام
مرسی از مطلب مفید تان

نویسنده: بهراد
تاریخ: ۱۳۹۱/۰۹/۱۹ ۱۱:۸

بسیار عالی بود ، خیلی برام مفید واقع شد
راه مشابه ای برای خروجی Json نیست؟
ممنون

نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۰۹/۱۹ ۱۱:۵۷

ممنون .هنوز به صورت native نه اما از تیم microsoft در این لینک خواسته شده که For Json را هم اضافه کند.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۹/۱۹ ۱۱:۵۹

« استفاده از JSON در SQL Server »

نویسنده: مهدی ناظم السادات
تاریخ: ۱۳۹۲/۰۹/۱۷ ۱۸:۵۸

حالا دستور Insert نداریم که بشه یه فایل xml رو مثل فایل backup روی دیتابیس restore کنیم؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۹/۱۸ ۰:۵۹

- می‌تونی با کدنویسی اینکار رو انجام بدی:

```
var reportData = new DataSet();
reportData.ReadXml("yourfile.xml");
var connection = new SqlConnection("DB ConnectionSTring");
var sbc = new SqlBulkCopy(connection);
sbc.DestinationTableName = "yourXMLTable";
```

- یا می‌تونی از import و export خود SQL Server استفاده کنی.
- و یا از OPENXML میشه استفاده کرد:

```
INSERT Customers
SELECT *
FROM OPENXML ...
```

نویسنده: ناصرنیازی
تاریخ: ۱۳۹۲/۱۱/۲۹ ۲۱:۵۱

تشکر فراوان از این مطلب فوق العاده کاربردی. یه نکته کوچک به ذهنم رسید. سومین قطعه کدی که نوشتید در جدول من کار نکرد به نظرم اومد که شاید اینچنین بوده باشه :
کد شما

```
FOR XML ELEMENTS
```

کدی که من مد نظرم هست و در جدول من کار می‌کنه :

```
for xml auto, ELEMENTS
```

عنوان: **تهیه خروجی XML از یک بانک اطلاعاتی، توسط EF Code first**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۱۰ ۱۱:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: Entity framework, xml, LINQ to XML

نگارش کامل SQL Server امکان تهیه خروجی XML از یک بانک اطلاعاتی را دارد . اما اگر بخواهیم از سایر بانک‌های اطلاعاتی که چنین توابع توکاری ندارند، استفاده کنیم چطور؟ برای تهیه خروجی XML توسط Entity framework و مستقل از نوع بانک اطلاعاتی در حال استفاده، حداقل دو روش وجود دارد:

الف) استفاده از امکانات Serialization توکار دات نت

```
using System.IO;
using System.Xml;
using System.Xml.Serialization;

namespace DNTViewer.Common.Toolkit
{
    public static class Serializer
    {
        public static string Serialize<T>(T type)
        {
            var serializer = new XmlSerializer(type.GetType());
            using (var stream = new MemoryStream())
            {
                serializer.Serialize(stream, type);
                stream.Seek(0, SeekOrigin.Begin);
                using (var reader = new StreamReader(stream))
                {
                    return reader.ReadToEnd();
                }
            }
        }
    }
}
```

در اینجا برای نمونه، لیستی از اشیاء مدنظر خود را تهیه کرده و به متد Serialize فوق ارسال کنید. نتیجه کار، تهیه معادل XML آن است.
امکانات سفارشی سازی محدودی نیز برای XmlSerializer درنظر گرفته شده است؛ برای نمونه قرار دادن ویژگی‌هایی مانند XmlIgnore بالای خواصی که نیازی به حضور آن‌ها در خروجی نهایی XML نمی‌باشد.

ب) استفاده از امکانات LINQ to XML دات نت

روش فوق بدون مشکل کار می‌کند، اما اگر بخواهیم قسمت Reflection خودکار ثانویه آن‌را (برای نمونه جهت استخراج مقادیر از لیست دریافتی) حذف کنیم، می‌توان از LINQ to XML استفاده کرد که قابلیت سفارشی سازی بیشتری را نیز در اختیار ما قرار می‌دهد (کاری که در سایت جاری برای تهیه خروجی XML از بانک اطلاعاتی آن انجام می‌شود).

```
private string createXmlFile(string dir)
        {
            var xLinq = new XElement("ArrayOfPost",
                        _blogPosts
                            .AsNoTracking()
                            .Include(x => x.Comments)
                            .Include(x => x.User)
                            .Include(x => x.Tags)
                            .OrderBy(x => x.Id)
                            .ToList()
                            .Select(x => new XElement("Post", postXElement(x)))
                            );

            var xmlFile = Path.Combine(dir, "dot-net-tips-database.xml");
            xLinq.Save(xmlFile);
```

```
            return xmlFile;
        }

        private static XElement[] postXElement(BlogPost x)
        {
            return new XElement[]
            {
                new XElement("Id", x.Id),
                new XElement("Title", x.Title),
                new XElement("Body", x.Body),
                new XElement("CreatedOn", x.CreatedOn),
                tagElement(x),
                new XElement("User",
                                new XElement("Id", x.UserId.Value),
                                new XElement("FriendlyName", x.User.FriendlyName))
            }.Where(item => item != null).ToArray();
        }

        private static XElement tagElement(BlogPost x)
        {
            var tags = x.Tags.Any() ?
                            x.Tags.Select(y =>
                                        new XElement("Tag",
                                                new XElement("Id", y.Id),
                                                new XElement("Name", y.Name)))
                                    .ToArray() : null;
            if (tags == null)
                return null;

            return new XElement("Tags", tags);
        }
```

خلاصه‌ای از نحوه تبدیل اطلاعات لیستی از مطالب را به معادل XML آن در کدهای فوق مشاهده می‌کنید. یک سری نکات ریز نیز باید در اینجا رعایت شوند:

1) کار با یک new XElement که دارای متد Save با فرمت XML نیز هست، شروع می‌شود. مقدار آن‌را مساوی یک کوئری از بانک اطلاعاتی قرار می‌دهیم. این کوئری چون قرار است تنها اطلاعاتی را از بانک اطلاعاتی دریافت کند و نیازی به تغییر در آن‌ها نیست، با استفاده از متد AsNoTracking ، حالت فقط خواندنی پیدا کرده است.

2) اطلاعاتی را که نیاز است در فایل نهایی XML وجود داشته باشند، تنها کافی است در قسمت Select این کوئری با فرمت new XElementهای تو در تو قرار دهیم. به این ترتیب قسمت Relection خودکار XmlSerializer روش مطرح شده در ابتدای بحث دیگر وجود نداشته و عملیات نهایی بسیار سریعتر خواهد بود.

3) چون در این حالت، کار انجام شده دستی است، باید نام‌های گره‌های صحیحی را انتخاب کنیم تا اگر قرار است توسط همان XmlSerializer مجددا کار serializer.Deserialize صورت گیرد، عملیات با شکست مواجه نشود. بهترین کار برای کم شدن سعی و خطاها، تهیه یک لیست اطلاعات آزمایشی و سپس ارسال آن به روش ابتدای بحث است. سپس می‌توان با بررسی خروجی آن مثلا دریافت که روش serializer.Deserialize به صورت پیش فرض به دنبال ریشه‌ای به نام ArrayOfPost برای دریافت لیستی از مطالب می‌گردد و نه Posts یا هر نام دیگری.

4) در کوئری LINQ to Entites نوشته شده، پیش از Select، یک ToList قرار دارد. متاسفانه EF اجازه استفاده مستقیم از Select هایی از نوع XElement را نمی‌دهد و باید ابتدا اطلاعات را تبدیل به LINQ to Objects کرد.

5) در حین تهیه XElementها اگر قرار است عنصری نال باشد، باید آن‌را در خروجی نهایی ذکر نکرد. به این ترتیب serializer.Deserialize بدون نیاز به تنظیمات اضافه‌تری بدون مشکل کار خواهد کرد. در غیراینصورت باید وارد مباحثی مانند تعریف یک فضای نام جدید برای خروجی XML به نام XSI رفت و سپس به کمک ویژگی‌ها، xsi:nil را به true مقدار دهی کرد. اما همانطور که در متد postXElement ملاحظه می‌کنید، برای وارد نشدن به مبحث فضای نام xsi، مواردی که null بوده‌اند، اصلا در آرایه نهایی ظاهر نمی‌شوند و نهایتا در خروجی، حضور نخواهند داشت. به این ترتیب متد ذیل، بدون مشکل و بدون نیاز به تنظیمات اضافه‌تری قادر است فایل XML نهایی را تبدیل به معادل اشیاء دات نتی آن کند.

```
using System.IO;
using System.Xml;
using System.Xml.Serialization;

namespace DNTViewer.Common.Toolkit
{
    public static class Serializer
```

```
    {
        public static T DeserializePath<T>(string xmlAddress)
        {
            using (var xmlReader = new XmlTextReader(xmlAddress))
                {
                    var serializer = new XmlSerializer(typeof(T));
                    return (T)serializer.Deserialize(xmlReader);
                }
        }
    }
}
```

عنوان: **اطلاع از بروز رسانی نرم افزار ساخته شده**
نویسنده: رضابیات
تاریخ: ۱۳۹۲/۰۵/۱۶ ۱۴:۵۵
آدرس: www.dotnettips.info
برچسب‌ها: Windows forms, xml, Software deployment, Check update application

برای شما هم پیش آمده که نرم افزاری را تهیه و منتشر کرده باشید و تمایل داشته باشید که استفاده کنندگان از وجود نسخه بروز شده مطلع شوند. یک راه ساده این است که اطلاعات نسخه جدید نرم افزار را داخل فایلی ذخیره کنیم و در وب سایت پشتیبانی نرم افزار قرار دهیم. حال بایستی اطلاعات این فایل را در زمان اجرای برنامه بررسی کنیم و در صورت وجود نسخه جدید از نرم افزار به کاربر اطلاع رسانی کنیم.
ابتدا فایل اطلاعات بروز رسانی نرم افزار را تهیه می‌کنیم و در وب سایت پشتیبانی نرم افزار قرار میدهیم. در اینجا از قالب Xml استفاده شده. که در آن Vertsion نسخه در دسترس نرم افزار است و URL هم مسیر وب سایت و یا فایل بروز رسانی است.

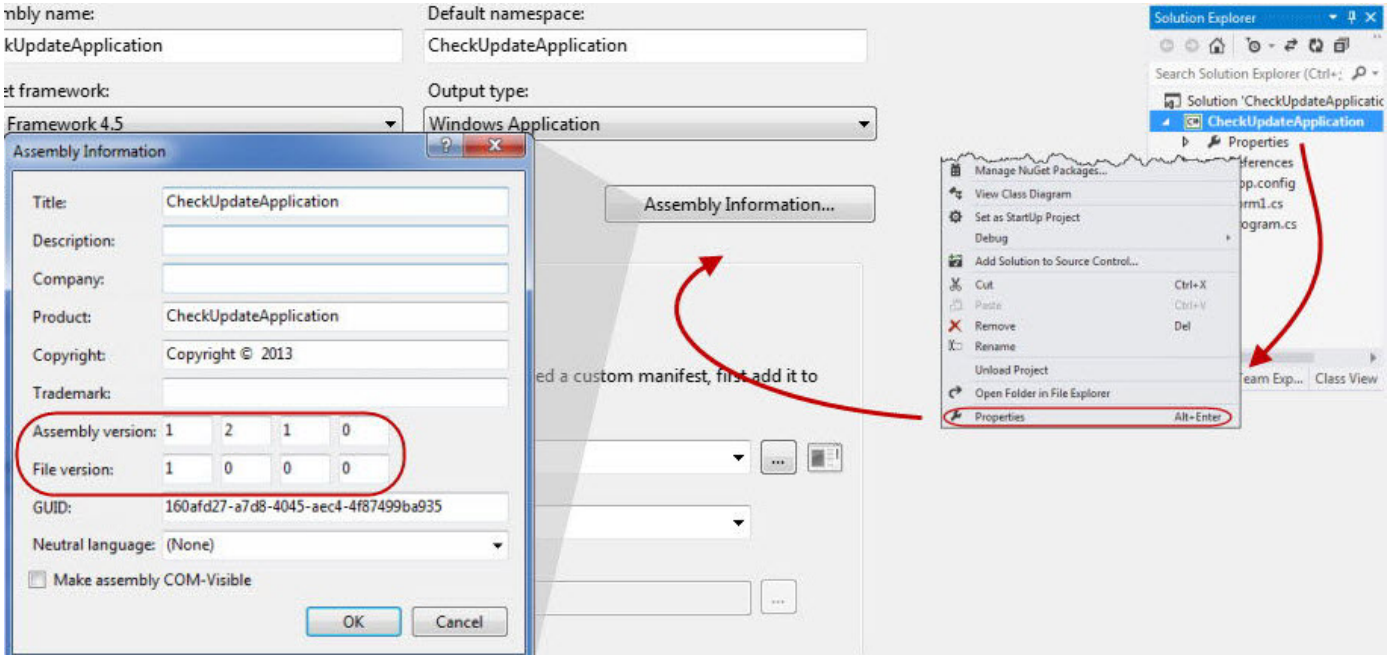
```
<?xml version="1.0" encoding="utf-8"?>
<AccountingApplication>
  <Version>1.5.2</Version>
  <URL>http://www.myappsupport.ir</URL>
</AccountingApplication>
```

نرم افزار را ساخته و کد زیر را در محل مناسبی کد نویسی می‌کنیم. این کد در ابتدا فایل Xml را خوانده و اطلاعات مورد نیاز را از آن دریافت می‌کند. سپس با استخراج نسخه اسمبلی برنامه و مقایسه این دو با هم از وجود نسخه جدید نرم افزار مطلع میشود.

```
...
using System.Xml;
namespace CheckUpdateApplication
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private void CheckUpdate_Click(object sender, EventArgs e)
        {
            Version NewVersion = null;
            string DownloadPath = "";
            try
            {
                XmlTextReader xmlRead = new
XmlTextReader("http://www.myappsupport.ir/AccUpdateVersion.xml");
                xmlRead.MoveToContent();
                string elmName = "";
                if ((xmlRead.NodeType == XmlNodeType.Element) && (xmlRead.Name ==
"AccountingApplication"))
                {
                    while (xmlRead.Read())
                    {
                        if (xmlRead.NodeType == XmlNodeType.Element)
                        {
                            elmName = xmlRead.Name;
                        }
                        else
                        {
                            if ((xmlRead.NodeType == XmlNodeType.Text) && (xmlRead.HasValue))
                            {
                                switch (elmName)
                                {
                                    case "Version":
                                        NewVersion = new Version(xmlRead.Value);
                                        break;
                                    case "URL":
                                        DownloadPath = xmlRead.Value;
                                        break;
                                }
                            }
                        }
                    }
                }
                Version AppVertion =
```

```
System.Reflection.Assembly.GetExecutingAssembly().GetName().Version;
                if (AppVertion.CompareTo(NewVersion) < 0)
                {
                    DialogResult Result = MessageBox.Show("نسخه " +
                        NewVersion.Major.ToString() + "." +
                        NewVersion.Build.ToString() + " نسخه" ,"؟در دسترس میباشد مایل به دانلود هستید
",جدید
                        MessageBoxButtons.YesNo,MessageBoxIcon.Question);
                    if (Result == DialogResult.Yes)
                    {
                        System.Diagnostics.Process.Start(DownloadPath);
                    }
                }
                else
                {
                    MessageBox.Show("نرم افزار بروز میباشد");
                }
            }
            catch (Exception E)
            {
                MessageBox.Show(E.Message);
            }
        }
    }
}
```

به روش زیر هم نسخه اسمبلی برنامه را می‌شود تغییر داد.



سورس برنامه نمونه CheckUpdateApplicationSample.rar

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۱۶ ۱۸:۹

ممنون از شما. یک روش برای اینکه مستقیما با XML Reader کار نکنیم می‌تونه استفاده از روش‌های سریالایز کردن کلاس‌ها باشه. دردسرش کمتره.

یک سؤال: این فلش‌های انحنا دار رو با چه برنامه‌ای ایجاد کردید؟

نویسنده: رضابیات
تاریخ: ۱۳۹۲/۰۵/۱۷ ۰:۳۴

با تشکر از توجه و راهنمایی شما. نرم افزارهای زیادی برای این کار وجود داره ولی من خیلی وقته از Snagit و Snagit Editor استفاده میکنم. این نرم افزار بیشتر برای فیلم و عکس گرفتن از دسکتاب استفاده میشه ولی امکانات فراوانی دیگری هم در این نرم افزار وجود داره. من خودم نسخه Snagit 11.2.1 را از سایت http://www.softgozar.com دانلود کردم.

نویسنده: مصطفی
تاریخ: ۱۳۹۲/۰۶/۰۹ ۹:۳۴

سلام
میخواستم بدونم فرق File Version و Assembly Version چیه؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۶/۰۹ ۲۲:۱۲

Assembly Version برای مصرف کنندگان اسمبلی شما مهمه (و فقط در دنیای CLR دارای اهمیت هست). مثلا شخصی ارجاعی به اسمبلی نگارش خاصی داره. AssemblyFileVersion در قسمت خواص فایل در ویندوز قابل مشاهده است و بیشتر برای برنامه‌های ست آپ مفیده. اطلاعات بیشتر

عنوان: **ثبت لینک‌های غیرتکراری**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۱۰ ۲۰:۱۵
آدرس: www.dotnettips.info
گروه‌ها: xml, LINQ to XML, OPML

ثبت لینک‌های مختلف در یک سیستم (مثلا قسمت به اشتراک گذاری لینک‌ها) در ابتدای کار شاید ساده به نظر برسد؛ خوب، هر صفحه‌ای که یک آدرس منحصربفرد بیشتر ندارد. ما هش این لینک را محاسبه می‌کنیم و بعد روی این هش، یک کلید منحصربفرد را تعریف خواهیم کرد تا دیگر رکوردی تکراری ثبت نشود. همچنین چون این هش نیز طول کوتاهی دارد، جستجوی آن بسیار سریع خواهد بود. واقعیت این است که خیر! این روش ناکارآمدترین حالت پردازش لینک‌های مختلف است.

برای مثال لینک‌های http://www.site.com و http://www.site.com/index.htm دو هش متفاوت را تولید می‌کنند اما در عمل یکی هستند. نمونه‌ی دیگر، لینک‌های http://www.site.com/index.htm و http://www.site.com/index.htm#section1 هستند که فقط اصطلاحا در یک fragment با هم تفاوت دارند و از این دست لینک‌هایی که باید در حین ثبت یکی درنظر گرفته شوند، زیاد هستند و اگر علاقمند به مرور آن‌ها هستید، می‌توانید به صفحه‌ی URL Normalization در ویکی‌پدیا مراجعه کنید.

اگر نکات این صفحه را تبدیل به یک کلاس کمکی کنیم، به کلاس ذیل خواهیم رسید:

```
using System;
using System.Web;

namespace OPMLCleaner
{
    public static class UrlNormalization
    {
        public static bool AreTheSameUrls(this string url1, string url2)
        {
            url1 = url1.NormalizeUrl();
            url2 = url2.NormalizeUrl();
            return url1.Equals(url2);
        }

        public static bool AreTheSameUrls(this Uri uri1, Uri uri2)
        {
            var url1 = uri1.NormalizeUrl();
            var url2 = uri2.NormalizeUrl();
            return url1.Equals(url2);
        }

        public static string[] DefaultDirectoryIndexes = new[]
            {
                "default.asp",
                "default.aspx",
                "index.htm",
                "index.html",
                "index.php"
            };

        public static string NormalizeUrl(this Uri uri)
        {
            var url = urlToLower(uri);
            url = limitProtocols(url);
            url = removeDefaultDirectoryIndexes(url);
            url = removeTheFragment(url);
            url = removeDuplicateSlashes(url);
            url = addWww(url);
            url = removeFeedburnerPart(url);
            return removeTrailingSlashAndEmptyQuery(url);
        }

        public static string NormalizeUrl(this string url)
        {
            return NormalizeUrl(new Uri(url));
        }

        private static string removeFeedburnerPart(string url)
        {
            var idx = url.IndexOf("utm_source=", StringComparison.Ordinal);
            return idx == -1 ? url : url.Substring(0, idx - 1);
        }

        private static string addWww(string url)
        {
            if (new Uri(url).Host.Split('.').Length == 2 && !url.Contains("://www."))
```

```
            {
                return url.Replace("://", "://www.");
            }
            return url;
        }

        private static string removeDuplicateSlashes(string url)
        {
            var path = new Uri(url).AbsolutePath;
            return path.Contains("//") ? url.Replace(path, path.Replace("//", "/")) : url;
        }

        private static string limitProtocols(string url)
        {
            return new Uri(url).Scheme == "https" ? url.Replace("https://", "http://") : url;
        }

        private static string removeTheFragment(string url)
        {
            var fragment = new Uri(url).Fragment;
            return string.IsNullOrWhiteSpace(fragment) ? url : url.Replace(fragment, string.Empty);
        }

        private static string urlToLower(Uri uri)
        {
            return HttpUtility.UrlDecode(uri.AbsoluteUri.ToLowerInvariant());
        }

        private static string removeTrailingSlashAndEmptyQuery(string url)
        {
            return url
                    .TrimEnd(new[] { '?' })
                    .TrimEnd(new[] { '/' });
        }

        private static string removeDefaultDirectoryIndexes(string url)
        {
            foreach (var index in DefaultDirectoryIndexes)
            {
                if (url.EndsWith(index))
                {
                    url = url.TrimEnd(index.ToCharArray());
                    break;
                }
            }
            return url;
        }
    }
}
```

از این روش برای تمیز کردن و حذف فیدهای تکراری در فایل‌های OPML تهیه شده نیز می‌شود استفاده کرد. عموما فیدخوان‌های نه‌چندان با سابقه، نکات یاد شده در این مطلب را رعایت نمی‌کنند و به سادگی می‌شود در این سیستم‌ها، فیدهای تکراری زیادی را ثبت کرد.

برای مثال اگر یک فایل OPML چنین ساختار XML ایی را داشته باشد:

```
<?xml version="1.0" encoding="utf-8"?>
<opml xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema" version="1.0">
  <body>
    <outline text="آی تی ایرانی">
    <outline type="rss"
                          text="فید کلی آخرین نظرات، مطالب، اشتراک‌ها و پروژه‌های .NET Tips"
                          title="فید کلی آخرین نظرات، مطالب، اشتراک‌ها و پروژه‌های .NET Tips"
                          xmlUrl="http://www.dotnettips.info/Feed/LatestChanges"
                          htmlUrl="http://www.dotnettips.info/" />
    </outline>
  </body>
</opml>
```

هر outline آن‌را به کلاس زیر می‌توان نگاشت کرد:

```
using System.Xml.Serialization;
```

```
namespace OPMLCleaner
{
    [XmlType(TypeName="outline")]
    public class Opml
    {
        [XmlAttribute(AttributeName="text")]
        public string Text { get; set; }

        [XmlAttribute(AttributeName = "title")]
        public string Title { get; set; }

        [XmlAttribute(AttributeName = "type")]
        public string Type { get; set; }

        [XmlAttribute(AttributeName = "xmlUrl")]
        public string XmlUrl { get; set; }

        [XmlAttribute(AttributeName = "htmlUrl")]
        public string HtmlUrl { get; set; }
    }
}
```

برای اینکار فقط کافی است از LINQ to XML به نحو ذیل استفاده کنیم:

```
var document = XDocument.Load("it-92-03-01.opml");
var results = (from node in document.Descendants("outline")
                            where node.Attribute("htmlUrl") != null && node.Parent.Attribute("text") !=
null
                             && node.Parent.Attribute("text").Value == "آی تی ایرانی"
                            select new Opml
                            {
                                HtmlUrl = (string)node.Attribute("htmlUrl"),
                                Text = (string)node.Attribute("text"),
                                Title = (string)node.Attribute("title"),
                                Type = (string)node.Attribute("type"),
                                XmlUrl = (string)node.Attribute("xmlUrl")
                            }).ToList();
```

در این حالت لیست کلیه فیدهای یک گروه را چه تکراری و غیرتکراری، دریافت خواهیم کرد. برای حذف موارد تکراری نیاز است از متد Distinct استفاده شود. به همین جهت باید کلاس ذیل را نیز تدارک دید:

```
using System.Collections.Generic;

namespace OPMLCleaner
{
    public class OpmlCompare : EqualityComparer<Opml>
    {
        public override bool Equals(Opml x, Opml y)
        {
            return UrlNormalization.AreTheSameUrls(x.HtmlUrl, y.HtmlUrl);
        }

        public override int GetHashCode(Opml obj)
        {
            return obj.HtmlUrl.GetHashCode();
        }
    }
}
```

اکنون با کمک کلاس OpmlCompare فوق که از کلاس UrlNormalization برای تشخیص لینک‌های تکراری استفاده می‌کند، می‌توان به لیست بهتر و متعادل‌تری رسید:

```
var distinctResults = results.Distinct(new OpmlCompare()).ToList();
```

عنوان: **ایجاد سیستم وضعیت آب و هوا مانند گوگل (بخش اول)**
نویسنده: شقایق اشتری
تاریخ: ۱۳۹۲/۱۲/۱۰ ۱۱:۳۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, C#, ASP.Net MVC, MVC, xml

در این آموزش قصد دارم چگونگی ایجاد یک سیستم اعلام وضعیت آب و هوا را مشابه آنچه که در سایت گوگل می‌بینید برای شما توضیح دهم. باید توجه داشت من این آموزش را با ASP.NET MVC نوشتم ولی شما می‌توانید با اندک تغییراتی در کدها، آنرا در ASP.NET وب فرمز نیز استفاده کنید. برای گرفتن آب و هوای هر شهر از Rssهای اعلام وضعیت آب و هوای یاهو استفاده می‌کنم و توضیح خواهم داد که چگونه با Rss آن کار کنید.
Rss آب و هوای هر شهر در یاهو به صورت یک لینک یکتا می‌باشد؛ به شکل زیر :

```
http://weather.yahooapis.com/forecastrss?w=WOEID&u=c
```

حال می‌خواهم کوئری استرینگ‌های این لینک را برای شما توضیح دهم. هر شهری بر روی کره‌ی زمین یک WOEID یکتا و منحصر بفرد دارد که شما به پارامتر w عدد WOEID شهر موردنظر خود را می‌دهید. بعد از مقداردهی پارامتر w، وقتی این لینک را در آدرس بار مرورگر خود می‌زنید، RSS مربوط به آب و هوای آن شهر را به شما می‌دهد. مثلا WOEID تهران عدد 28350859 می‌باشد.
و این لینک http://weather.yahooapis.com/forecastrss?w=28350859&u=c اطلاعات آب و هوای تهران را در قالب یک RSS به شما نمایش خواهد داد.

خوب، حالا پارامتر دوم یعنی پارامتر u چکاری را انجام می‌دهد؟
* چنانچه مقدار پارامتر u برابر c باشد، یعنی شما دمای آب و هوای شهر مد نظر را بر اساس سانتیگراد می‌خواهید.
* اگر مقدار پارامتر u برابر f باشد، یعنی شما دمای آب و هوای آن شهر مورد نظر را بر اساس فارنهایت می‌خواهید.

برای گرفتن WOEID شهرها هم به این سایت بروید http://woeid.rosselliot.co.nz و اسم هر شهری که می‌خواهید بزنید تا WOEID را به شما نمایش دهد.

در این مثال من از یک DropDown استفاده کردم که کاربر با انتخاب هر شهر از DropDown، آب و هوای آن شهر را مشاهده می‌کند.
Action مربوط به صفحه‌ی Index به صورت زیر می‌باشد :

```
[HttpGet]
        public ActionResult Index()
        {
           ViewBag.ProvinceList = _RPosition.Positions;
            ShowWeatherProvince(8);
            return View();
        }
```

در اینجا من لیست شهرها را از جدول می‌خوانم که البته این جدول را چون بخش مهمی نبود و فقط شامل ID و نام شهرها بود در فایل ضمیمه قرار ندادم و نام شهرها و ID آنها را بر عهده‌ی خودتان گذاشتم.
حال تابعی را که آب و هوای مربوط به هر شهر را نمایش می‌دهد، به شرح زیر است:

```
public ActionResult ShowWeatherProvince(int dpProvince)
        {
            XDocument rssXml=null;
            CountryName CountryName = new CountryName();
            if (dpProvince != 0)
            {
                switch (dpProvince)
                {
                    case 1:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345768&u=c");
```

```
                            CountryName = new CountryName() { Country = "Iran", City = "Azarbayejan-e
Sharqhi" };
                            break;
                        }
                    case 2:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345767&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Azarbayejan-e
Qarbi" };
                            break;
                        }
                    case 3:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2254335&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Ardebil" };
                            break;
                        }
                    case 4:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=28350859&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Alborz" };
                            break;
                        }
                    case 5:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345787&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Esfahan" };
                            break;
                        }
                    case 6:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345775&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Ilam" };
                            break;
                        }
                    case 7:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2254463&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Bushehr" };
                            break;
                        }
                    case 8:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=28350859&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Tehran" };
                            break;
                        }
                    case 9:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345769&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Chahar Mahall
va Bakhtiari" };
                            break;
                        }
                    case 10:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=56189824&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Razavi
Khorasan" };
                            break;
                        }
                    case 11:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345789&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Shomali
Khorasan" };
                            break;
                        }
                    case 12:
                        {
                            rssXml =
```

```
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345789&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Jonubi
Khorasan" };
                            break;
                        }
                    case 13:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345778&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Khuzestan" };
                            break;
                        }
                    case 14:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2255311&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Zanjan" };
                            break;
                        }
                    case 15:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345784&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Semnan" };
                            break;
                        }
                    case 16:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345770&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Sistan va
Baluchestan" };
                            break;
                        }
                    case 17:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345772&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Fars" };
                            break;
                        }
                    case 18:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=20070200&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Qazvin" };
                            break;
                        }
                    case 19:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2255062&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Qom" };
                            break;
                        }
                    case 20:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345779&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Kordestan" };
                            break;
                        }
                    case 21:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2254796&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Kerman" };
                            break;
                        }
                    case 22:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2254797&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Kermanshah" };
                            break;
                        }
                    case 23:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345771&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Kohgiluyeh va
```

```
Buyer Ahmad" };
                            break;
                        }
                    case 24:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=20070201&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Golestan" };
                            break;
                        }
                    case 25:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345773&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Gilan" };
                            break;
                        }
                    case 26:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345782&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Lorestan" };
                            break;
                        }
                    case 27:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345783&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Markazi" };
                            break;
                        }
                    case 28:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345780&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Mazandaran" };
                            break;
                        }
                    case 29:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2254664&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Hamedan" };
                            break;
                        }
                    case 30:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345776&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Hormozgan" };
                            break;
                        }
                    case 31:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2253355&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Yazd" };
                            break;
                        }
                }
                ViewBag.Location = CountryName;
                XNamespace yWeatherNS = "http://xml.weather.yahoo.com/ns/rss/1.0";
                List<YahooWeatherRssItem> WeatherList = new List<YahooWeatherRssItem>();
                for (int i = 0; i < 4; i++)
                {
                    YahooWeatherRssItem YahooWeatherRssItem = new YahooWeatherRssItem()
                    {
                        Code = Convert.ToInt32(rssXml.Descendants("item").Elements(yWeatherNS +
"forecast").ElementAt(i).Attribute("code").Value),
                        Day = rssXml.Descendants("item").Elements(yWeatherNS +
"forecast").ElementAt(i).Attribute("day").Value,
                        Low = rssXml.Descendants("item").Elements(yWeatherNS +
"forecast").ElementAt(i).Attribute("low").Value,
                        High = rssXml.Descendants("item").Elements(yWeatherNS +
"forecast").ElementAt(i).Attribute("high").Value,
                        Text = rssXml.Descendants("item").Elements(yWeatherNS +
"forecast").ElementAt(i).Attribute("text").Value,
                    };

                    WeatherList.Add(YahooWeatherRssItem);
                }
```

```
                ViewBag.FeedList = WeatherList;
            }

                return PartialView("_Weather");

        }
```

قسمت SwitchCase، مقدار و Value مربوط به هر آیتم DropDown را که شامل یک اسم شهر است، میگیرد و RSS مربوط به آن شهر را بر می‌گرداند.
حالا کد مربوط به خواندن فایل Rss را برایتان توضیح می‌دهم : حلقه‌ی for 0 تا 4 (که در کد بالا مشاهده می‌کنید)یعنی اطلاعات 4 روز آینده را برایم برگردان.
من تگ‌های High ، Low ، Day ، Code و text فایل RSS را در این حلقه For می‌خوانم که البته مقادیر این 4 روز را در لیستی اضافه می‌کنم که نوع این لیست هم از نوع YahooWeatherRssItem می‌باشد. من این کلاس را در فایل ضمیمه قرار دادم. اکنون هر کدام از این تگ‌ها را برایتان توضیح می‌دهم:

code : هر آب و هوا کدی دارد .مثلا آب و هوای نیمه ابری یک کد ، آب و هوای آفتابی کدی دیگر و ...
Low: حداقل دمای آن روز را به ما می‌دهد .
High: حداکثر دمای آن روز را به می‌دهد .
day: نام روز از هفته را بر می‌گرداند مثلا شنبه ، یکشنبه و ....
text: که توضیحاتی می‌دهد مثلا اگر هوا آفتابی باشد مقدار sunny را بر می‌گرداند و ...

خوب، تا اینجا ما Rss مربوط به هر شهر را خواندیم حالا در قسمت Design باید چکار کنیم .
کدهای html صفحه‌ی Index ما شامل کدهای زیر است :

```
@{

    ViewBag.Title = "Weather";
}

<link href="~/Content/User/Weather/Weather.css" rel="stylesheet" />
@section scripts{
    <script src="@Url.Content("~/Scripts/jquery-1.6.2.min.js")" type="text/javascript"></script>
        <script src="@Url.Content("~/Scripts/jquery.unobtrusive-ajax.min.js")"
type="text/javascript"></script>
       <script type="text/javascript">
           $("#dpProvince").change(function () {
               $(this).parents("form").submit();
           });
    </script>
}
<h2>Weather</h2>
<div id="Progress">
    <img src="~/Images/User/Other/ajax-loader.gif" />
</div>
<div id="BoxContent"> @Html.Partial("_Weather")</div>

  @using (Ajax.BeginForm(actionName: "ShowWeatherProvince", ajaxOptions: new AjaxOptions {
UpdateTargetId = "BoxContent", LoadingElementId = "Progress", InsertionMode = InsertionMode.Replace }))
                {
<div style="padding-top:15px;">
        <div style="float:left; width:133px; ">Select Your Province</div>
        <div style="float:left">   @Html.DropDownList("dpProvince", new
SelectList(ViewBag.ProvinceList, "Id", "Name"),"Select Your Province", new { @class =
"webUserDropDown", @style = "width:172px" })</div>
    </div>
  }
```

و کدهای _Weather که Partial است به صورت زیر است:

```
@{
    List<Weather.YahooWeatherRssItem> Feeds = ViewBag.FeedList;
}
<div>
```

```
    @{
        HtmlString StartTable = new HtmlString("<table class='WeatherTable' cellspacing='0'
cellpadding='0'><tbody><tr>");
        HtmlString EndTable = new HtmlString("</tr></tbody></table>");
        HtmlString StartTD = new HtmlString("<td>");
        HtmlString EndTD = new HtmlString("</td>");
    }
    <div style="width: 300px;">
        @{
            @StartTable
            foreach (var item in Feeds)
            {
            @StartTD
            <div>@item.Day</div>
            <div>
                @{
                string FileName = "";
                switch (item.Code)
                {
                    case 0: { FileName = "/Images/User/Weather/Tornado.png"; break; }
                    case 1: { FileName = "/Images/User/Weather/storm2.gif"; break; }
                    case 2: { FileName = "/Images/User/Weather/storm2.gif"; break; }
                    case 3: { FileName = "/Images/User/Weather/storm2.gif"; break; }
                    case 4: { FileName = "/Images/User/Weather/15.gif"; break; }
                    case 5: { FileName = "/Images/User/Weather/29.gif"; break; }
                    case 6: { FileName = "/Images/User/Weather/29.gif"; break; }
                    case 7: { FileName = "/Images/User/Weather/29.gif"; break; }
                    case 8: { FileName = "/Images/User/Weather/26.gif"; break; }
                    case 9: { FileName = "/Images/User/Weather/drizzle.png"; break; }
                    case 10: { FileName = "/Images/User/Weather/26.gif"; break; }
                    case 11: { FileName = "/Images/User/Weather/18.gif"; break; }
                    case 12: { FileName = "/Images/User/Weather/18.gif"; break; }
                    case 13: { FileName = "/Images/User/Weather/19.gif"; break; }
                    case 14: { FileName = "/Images/User/Weather/19.gif"; break; }
                    case 15: { FileName = "/Images/User/Weather/19.gif"; break; }
                    case 16: { FileName = "/Images/User/Weather/22.gif"; break; }
                    case 17: { FileName = "/Images/User/Weather/Hail.png"; break; }
                    case 18: { FileName = "/Images/User/Weather/25.gif"; break; }
                    case 19: { FileName = "/Images/User/Weather/dust.png"; break; }
                    case 20: { FileName = "/Images/User/Weather/fog_icon.png"; break; }
                    case 21: { FileName = "/Images/User/Weather/hazy_icon.png"; break; }
                    case 22: { FileName = "/Images/User/Weather/2017737395.png"; break; }
                    case 23: { FileName = "/Images/User/Weather/32.gif"; break; }
                    case 24: { FileName = "/Images/User/Weather/32.gif"; break; }
                    case 25: { FileName = "/Images/User/Weather/31.gif"; break; }
                    case 26: { FileName = "/Images/User/Weather/7.gif"; break; }
                    case 27: { FileName = "/Images/User/Weather/38.gif"; break; }
                    case 28: { FileName = "/Images/User/Weather/6.gif"; break; }
                    case 29: { FileName = "/Images/User/Weather/35.gif"; break; }
                    case 30: { FileName = "/Images/User/Weather/7.gif"; break; }
                    case 31: { FileName = "/Images/User/Weather/33.gif"; break; }
                    case 32: { FileName = "/Images/User/Weather/1.gif"; break; }
                    case 33: { FileName = "/Images/User/Weather/34.gif"; break; }
                    case 34: { FileName = "/Images/User/Weather/2.gif"; break; }
                    case 35: { FileName = "/Images/User/Weather/freezing_rain.png"; break; }
                    case 36: { FileName = "/Images/User/Weather/30.gif"; break; }
                    case 37: { FileName = "/Images/User/Weather/15.gif"; break; }
                    case 38: { FileName = "/Images/User/Weather/15.gif"; break; }
                    case 39: { FileName = "/Images/User/Weather/15.gif"; break; }
                    case 40: { FileName = "/Images/User/Weather/12.gif"; break; }
                    case 41: { FileName = "/Images/User/Weather/22.gif"; break; }
                    case 42: { FileName = "/Images/User/Weather/22.gif"; break; }
                    case 43: { FileName = "/Images/User/Weather/22.gif"; break; }
                    case 44: { FileName = "/Images/User/Weather/39.gif"; break; }
                    case 45: { FileName = "/Images/User/Weather/thundershowers.png"; break; }
                    case 46: { FileName = "/Images/User/Weather/19.gif"; break; }
                    case 47: { FileName = "/Images/User/Weather/thundershowers.png"; break; }
                    case 3200: { FileName = "/Images/User/Weather/1211810662.png"; break; }
                }
                }
                <img alt='@item.Text' title='@item.Text' src='@FileName'>
            </div>
            <div>
                <span>@item.High°</span>
                <span>@item.Low°</span>
            </div>
            @EndTD
            }
        }
        @EndTable
    </div>
```

```
</div>
```

من عکس‌های مربوط به وضعیت آب و هوا را در فایل ضمیمه قرار دادم.
چنانچه در مورد RSS وضعیت آب و هوای یاهو اطلاعات دقیق‌تری را می‌خواهید بدانید به این لینک بروید.
در آموزش بعدی قصد دارم برایتان این بخش را توضیح دهم که بر اساس IP بازدید کننده سایت شما، اطلاعات آب و هوایی شهر بازدید کننده را برایش در سایت نمایش دهد.

Files-06bf65bac63d4dd694b15fc24d4cb074.zip

موفق باشید

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۱ ۰:۵

ممنون از شما. چند نکته جزئی در مورد بهبود کیفیت کدها. در Viewهای MVC باید کدنویسی صرفا به نمایش اطلاعات View خلاصه شود. یعنی switch داخل آن بهتر است تبدیل به یک extension method شده و نهایتا استفاده شود (برای تمیز کردن View). همچنین استفاده از switch 1 و 2 و 10000 اصطلاحا به magic code مشهور هستند. یعنی مشخص نیست معنای این اعداد چی هست. این‌ها را عموما بهتر است تبدیل به enum کرد و بعد استفاده نهایی.

نویسنده: شقایق اشتری
تاریخ: ۱۳۹۲/۱۲/۱۲ ۸:۳۹

ممنونم از شما که مشکل کد نویسی من را گفتید . extension methodها منظورتون همون helperها هست ؟ لینکی دارید که در مورد این extension methodها آموزشی داده باشد چون من می‌خوام کدهامو تا جایی که می‌شود بهینه بنویسم .

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۱۲/۱۲ ۹:۱۵

یک مثال در اینجا
قسمت آخر (**Ifهای شرطی را در Viewها را در متدهای کمکی کپسوله کنید)**

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۲ ۹:۵۱

قسمت switch ایی را که در View نوشتید، تبدیل کنید به یک متد کمکی در کلاسی خارج از View: (مهم نیست متد الحاقی باشد یا خیر؛ فقط داخل View نباشد)

```
public static string GetFileName(int code)
{
    switch (code)
    {
       case 0: return "/Images/User/Weather/Tornado.png";
  //...
}
```

بعد یک خاصیت محاسباتی به نام FileName به مدل مورد استفاده اضافه کنید:

```
public class YahooWeatherRssItem
    {
        public int Code { get; set; }
        //...
        public string FileName
        {
          get { return Util.GetFileName(Code); }
        }
    }
```

به این صورت View از کدهای محاسبات یافتن FileName خالی می‌شود.

نویسنده: علی
تاریخ: ۱۳۹۲/۱۲/۱۲ ۱۰:۷

یه راه دیگه هم استفاده از Yahoo Query Language هست، می تونید با کوئری زیر اطلاعات آب و هوای شهر مورد نظر رو در قالب JSON دریافت کنید

```
select * from weather.forecast where woeid in (select woeid from geo.placefinder where text="CityName")
```

نویسنده: شقایق اشتری
تاریخ: ۱۳۹۲/۱۲/۱۲ ۱۰:۱۳

یه سوال دیگه که داشتم اگر بین کدهایمان ، کد html وجود داشت باز هم باید آنها را از view جدا کنیم و به صورت extension method بنویسیم؟

یا اینکه جداسازی کدها از View را زمانی انجام می‌دهیم که چندین خط کد پشت سر هم نوشته باشیم و بین آنها کد html یی نباشد ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۲ ۱۰:۴۱

- نه الزاما. اگر چندین جا استفاده و تکرار می‌شود یا منطق طولانی دارد، روش « نوشتن HTML Helpers ویژه، به کمک امکانات Razor » می‌تواند مفید باشد.
- منطق قرار گرفته در View فقط باید کار «نهایی» نمایشی را انجام دهد. اگر در View مثلا مستقیما با دیتابیس کار می‌کنید، محاسبات مرتبط با فیلد خاصی را انجام می‌دهید و کلا هر منطقی که «نهایی» بودن نمایش اطلاعات را زیر سؤال ببرد، باید از View جدا شده و به کنترلر و زیرساخت آن منتقل شود.

نویسنده: محسن تقی پور
تاریخ: ۱۳۹۲/۱۲/۲۰ ۰:۸

میشه راجبه این روش کمی بیشتر توضیح بدین

نویسنده: Anaritius
تاریخ: ۱۳۹۲/۱۲/۲۱ ۰:۴۴

سلام
ممنون از مقاله خوبتون
امکانش هست سورس کد webForm ش رو هم بذارید واسه دانلود؟
ممنون

نویسنده: علی
تاریخ: ۱۳۹۲/۱۲/۲۱ ۱۰:۸

اینجا یک مثال از نحوه کار با YQL رو توضیح داده