عنوان: **خواندن فید گزارش آب و هوای یاهو با استفاده از روش Xml serialization**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۸/۲۱ ۱۹:۳۲:۳۹
آدرس: www.dotnettips.info
برچسب‌ها: RSS

در مطلب قبلی (در مورد کتابخانه anti-xss مایکروسافت) از روش xml serialization برای خواندن فایل xml حملات استفاده کردیم.
ایجاد این کلاس و نگاشت اشیاء با توجه به ساختار ساده آن به صورت دستی و به‌سادگی انجام شد. اکنون به مثال زیر دقت بفرمائید:
سرویس آب و هوای یاهو برای شهرهای مختلف ایران از طریق لینک زیر قابل استفاده است:
http://weather.yahoo.com/regional/IRXX.html
اگر به صفحات شهرهای مختلف مراجعه نمائید، یک فید rss هم مشاهده خواهید کرد، برای مثال در مورد تهران داریم:
http://weather.yahooapis.com/forecastrss?p=IRXX0018&u=c
ساختار این فایل xml تا حدودی با یک rss استاندارد تطابق دارد. اما اگر به سورس xml آن دقت کنیم تگ‌های دیگری را نیز مشاهده خواهیم کرد که برای مثال دما ، تاریخ و شرایط جوی را به صورت دقیقی و با استفاده از اصول xml ارائه می‌دهند.

```
<yweather:condition  text="Partly Cloudy"  code="29"  temp="10"  date="Tue, 11 Nov 2008 5:30 pm IRT" />
```

خوب، برای دریافت این اطلاعات چه باید کرد؟ یکی از روش‌های متداول برای کار با این نوع داده‌ها، استفاده از کلاس DataSet در دات نت و فراخوانی متد ReadXml آن است (یک آدرس اینترنتی را هم می‌تواند دریافت کند). سپس مطابق روش‌های معمول ADO.Net می‌توان به تگ‌ها ومقادیر آنها دسترسی داشت.
روش بالا هر چند مشکلی ندارد اما به زیبایی کار با خواص یک کلاس متناظر با آن فایل xml نیست. اما در اینجا برای استفاده از روش xml serialization یک مشکل وجود دارد! ایجاد دستی این کلاس که بیانگر عملکرد آن فایل xml است کار ساده‌ای نیست.
خوشبختانه به همراه SDK دات نت فریم ورک 2، برنامه‌ای به نام xsd.exe نیز همراه است که کار ایجاد یک کلاس cs یا vb را از یک فایل xml جهت این منظور انجام می‌دهد (این برنامه برای مثال در مسیر C:\Program Files\Microsoft.NET\SDK\v2.0\Bin قرار دارد).

برای ایجاد فایل کلاس به صورت خودکار از روی یک فایل xml موجود باید به ترتیب زیر عمل کرد:
الف) ایجاد فایل xsd متناظر (XML Schema Definition)
برای اینکار در خط فرمان تایپ کنید:

```
xsd.exe file.xml
```

نکته 1:
روش دیگر انجام این کار : فایل xml را در VS.net باز کنید، از منوی بالای صفحه گزینه xml را انتخاب نموده و بر روی دکمه Create Schema کلیک کنید.

ب) ایجاد فایل cs یا vb از روی فایل(های) xsd ایجاد شده
در اینجا برای فید آب و هوای یاهو سه فایل xsd تولید خواهد شد. برای تبدیل آنها به کلاس cs باید دستور زیر را در خط فرمان اجرا کرد:

```
Xsd.exe file_1.xsd file_2.xsd file_3.xsd /c
```

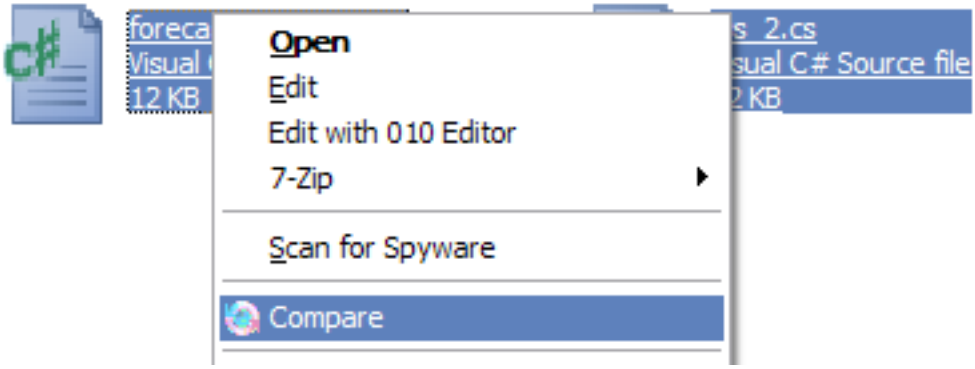
این مورد نکته مهمی است و تنها اگر یکی از فایل‌ها اینجا ذکر شوند، کلاس ناقصی تشکیل خواهد شد. (برای نمونه فایل xssAttacks.xml مطلب قبلی، ساختار ساده‌ای داشته و تنها به یک فایل xsd ختم خواهد شد)

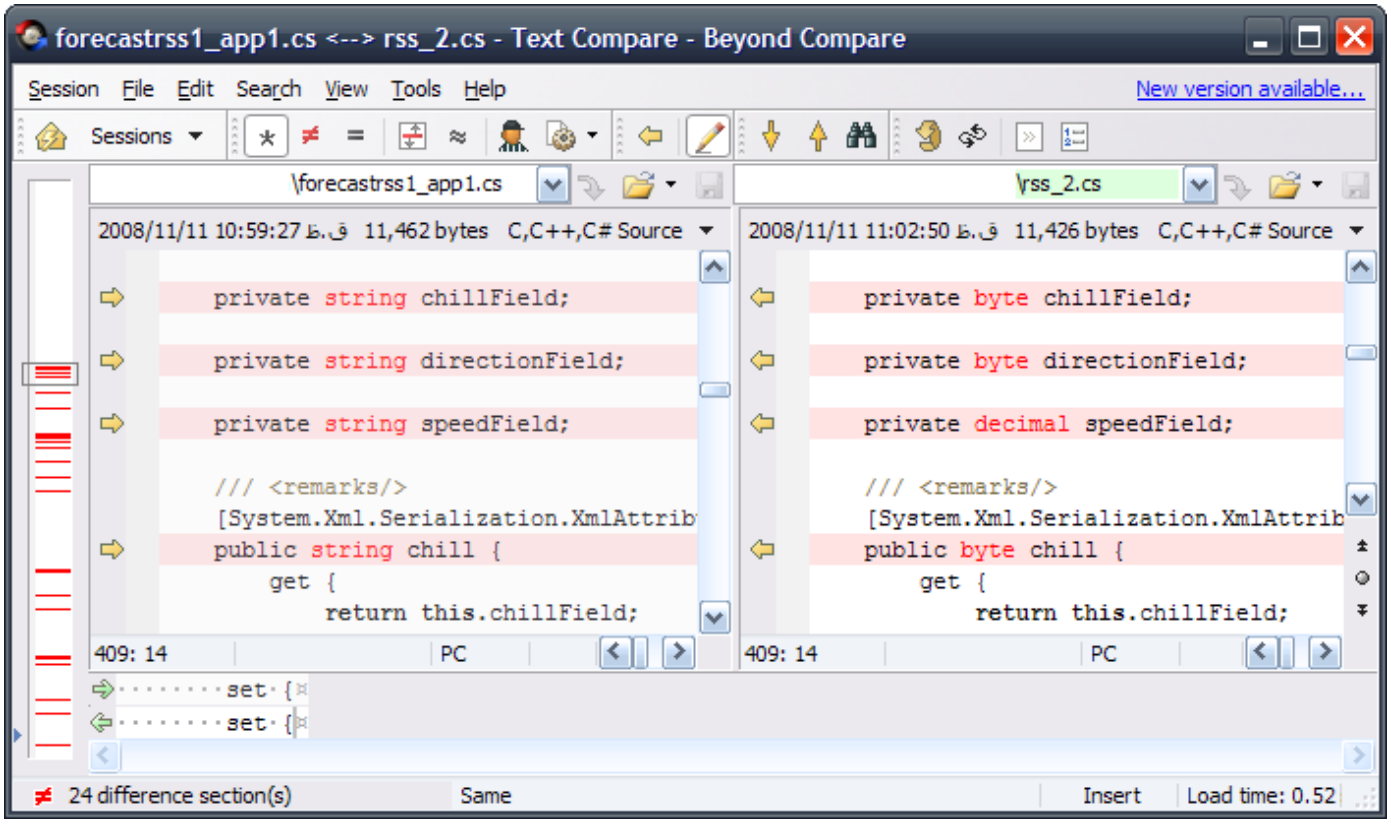
نکته 2:
برای انتخاب زبان VB (با توجه به این‌که پیش فرض آن CS است) می‌توان به صورت زیر عمل کرد:

```
xsd.exe file.xsd /c  /l:vb
```

نکته 3:
برای تولید فایل xsd ، از برنامه Infer.exe نیز می‌توان استفاده کرد (خروجی نهایی دقیق‌تری را ارائه می‌دهد). این برنامه را از اینجا دریافت کنید.

تصاویر زیر مقایسه دو فایل کلاس نهایی تولید شده از xsd های این دو برنامه است:





پس از طی این مراحل فایل کلاس ما برای xml serialization آماده خواهد شد. مرحله بعد دریافت اطلاعات و نگاشت آن به این کلاس تولید شده است:

```
public static rss DeserializeFromXML()
      {
          XmlSerializer deserializer =
              new XmlSerializer(typeof(rss));
          using (XmlReader reader =
XmlReader.Create("http://weather.yahooapis.com/forecastrss?p=IRXX0018&u=c"))
          {
              return (rss)deserializer.Deserialize(reader);
          }
      }
```

کلاس rss

از فید xml و فایل‌های xsd آن که تولید کردیم به صورت خودکار ایجاد شده است.

اکنون برای مثال خواندن وضعیت فعلی جوی از فید دریافتی به سادگی زیر است:

```
rss data = DeserializeFromXML();
MessageBox.Show(data.channel.item.condition.text);
```

نظرات خوانندگان

نویسنده: مهدی یوسفی
تاریخ: ۱۳۸۷/۰۸/۲۵ ۰۷:۲۶:۰۰

واقعا عالی بود.

عنوان: **دو نکته کوتاه در مورد RSS های فارسی**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۰۷ ۰۹:۱۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: RSS

اولین نکته مربوط به تاریخ هر مدخل (entry) می‌شود. این تاریخ نباید شمسی باشد! این تاریخ باید حتما استاندارد باشد. عموما یکی از دو استاندارد زیر باید مورد استفاده قرار گیرد:

RFC #822
http://www.ietf.org/rfc/rfc0822.txt
(Standard for ARPA Internet Text Messages (Date and Time Specification

RFC #3339
http://www.ietf.org/rfc/rfc3339.txt
(Date and Time on the Internet (Timestamps

برای مثال در دات نت برای تولید این فرمت استاندارد می‌توان به صورت زیر عمل کرد:

```
DateTime.Now.ToUniversalTime().ToString("r")
```

متاسفانه بسیاری از برنامه نویس‌های هم وطن این نکته را رعایت نمی‌کنند و برنامه‌های فیدخوان را دچار مشکل می‌کنند. (برای نمونه برنامه معروف feedDemon تاریخ چند سال پیش را ثبت خواهد کرد و در به روز رسانی و دنبال کردن مطالب سایت مورد نظر دچار مشکل خواهد شد)

چند مثال از این دست: (سورس صفحه را در مرورگر مطالعه نمائید)
http://www.faradade.com/Xml/RSS.xml
و یا
http://www.ayande.ir/atom.xml
و یا
http://www.tci-sk.ir/Rss.aspx
(ایشان بهتر است علاوه بر این مورد، از XmlTextWriter استفاده کنند و خروجی را به صورت یک فایل xml و نه html در مرورگر Flush کنند)

و یا بدتر از این بعضی از سایت‌ها آموزش‌های غلطی را هم ارائه می‌دهند:
http://www.faradade.com/Article.aspx?code=a726ae6a-f8e1-4b29-88b4-8e7a04e6d06d
به قسمت pubDate دقت کنید.
مطابق معمول این آموزش الان در 200 سایت کپی و پیست شده! عنوان آموزش را در گوگل جستجو کنید!
این کد آموزش داده شده یک ایراد دیگر هم دارد. آیا الزامی دارد که حتما قسمت con.Close به همین ترتیب نوشته شده اجرا شود؟ اگر این بین خطایی رخ دهد تکلیف این کانکشن باز و سایر موارد چه خواهد شد؟ کلا استفاده از try و finally و یا استفاده از using را برای چه هدفی اختراع کرده‌اند؟

و یا بعضی از سایت‌ها این مورد را رعایت می‌کنند اما به صورت نصفه و نیمه. برای مثال: (تاریخ ارائه شده کامل نیست. بنابراین استاندارد تلقی نخواهد شد)

http://www.srco.ir/Articles/RSSArticles.xml

برای آزمایش میزان استاندارد بودن خروجی فید خود می‌توان از سرویس زیر استفاده کرد:
/http://validator.w3.org/feed

مطلب دیگر ایراد نیست بلکه نکته‌ای است که حداقل از IE7 به بعد رعایت می‌شود:
لطفا زبان فید را مشخص کنید! بله، اگر این مورد را مشخص کنید، از IE7 به بعد فید فارسی به صورت خودکار از راست به چپ نمایش داده می‌شود و این امر سبب سهولت خواندن مطالب فارسی سایت شما خواهد شد.
مشاهده اصل مطلب که توسط یکی از اعضای تیم مربوطه مایکروسافت نوشته شده:
مشاهده

اصلاحیه برای RSS فارسی:

```
<language>fa-IR</language>
```

و برای Atom فارسی:

```
<feed xml:lang="fa">
```

و اگر می‌خواهید خروجی استانداردی داشته باشید، کتابخانه سورس باز زیر توصیه می‌شود:
http://www.codeplex.com/Argotic

با تشکر از همکاری شما!

نظرات خوانندگان

نویسنده: Afshar Mohebbi
تاریخ: ۱۳۸۷/۰۸/۱۳ ۰۸:۵۰:۰۰

مطلب خوبی بود...

نویسنده: چالیست
تاریخ: ۱۳۸۸/۰۴/۱۱ ۱۳:۵۱:۱۹

سپاس بزرگوار
خواندنی بود

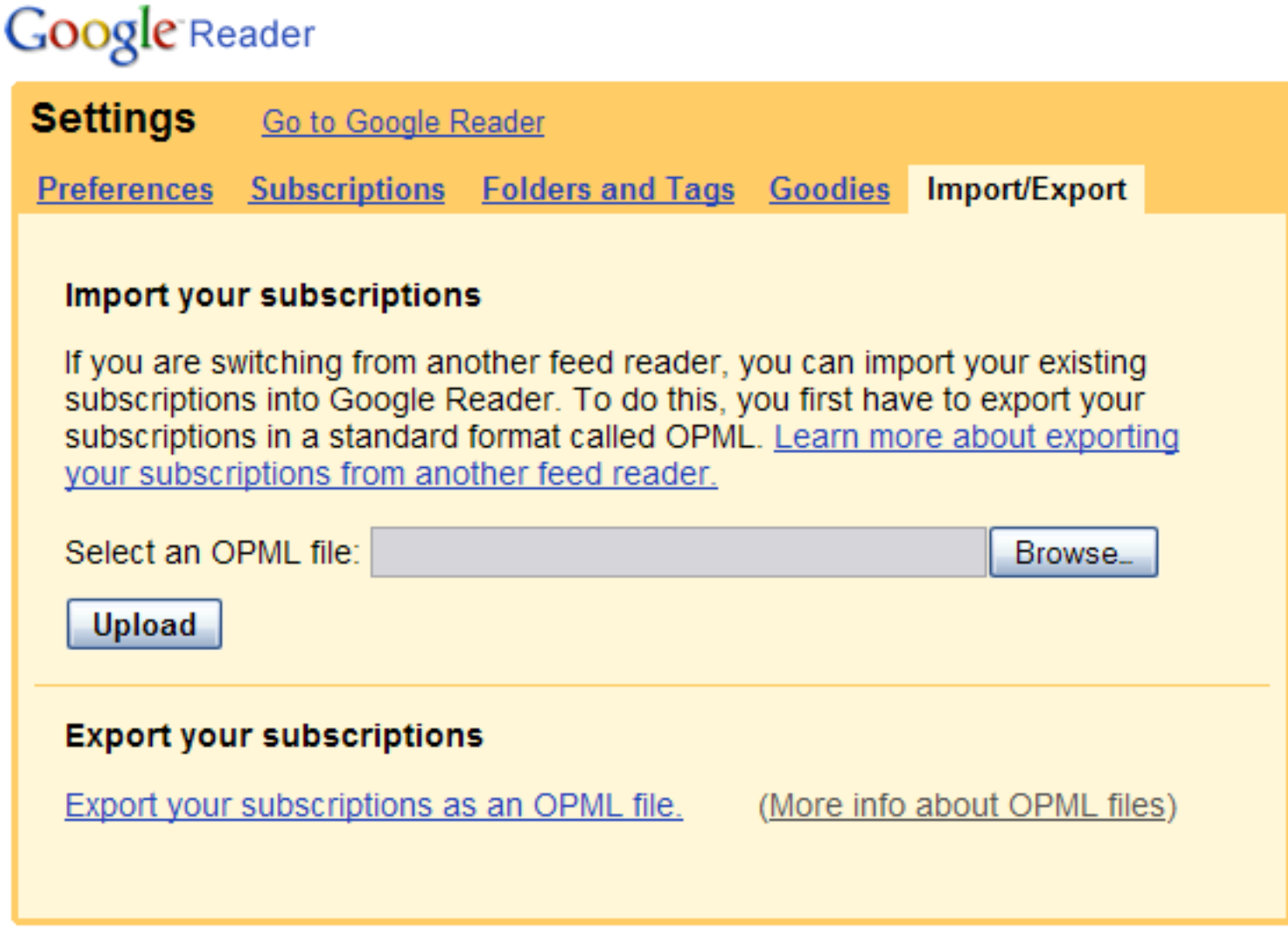
عنوان: **آشنایی با فرمت OPML**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۲۶ ۲۳:۱۹:۱۵
آدرس: www.dotnettips.info
برچسب‌ها: RSS

OPML یا Outline Processor Markup Language اساسا فایلی است مبتنی بر XML که امروزه بیشتر جهت توزیع لینک‌های تغذیه خبری سایت‌ها (RSS/Atom و امثال آن) مورد استفاده قرار می‌گیرد.
به بیانی ساده‌تر، بجای این‌که بگویند ما به این 100 وبلاگ علاقمند هستیم و لینک تک تک آنها را به شما ارائه بدهند، یک فایل OPML استاندارد از آن‌ها درست کرده و در اختیار شما قرار می‌دهند. به این صورت با چند کلیک ساده، این فایل در نرم افزار فیدخوان شما import شده و آدرس‌ها بلافاصله قابل استفاده خواهند بود.
نمونه‌ای از این فرمت:

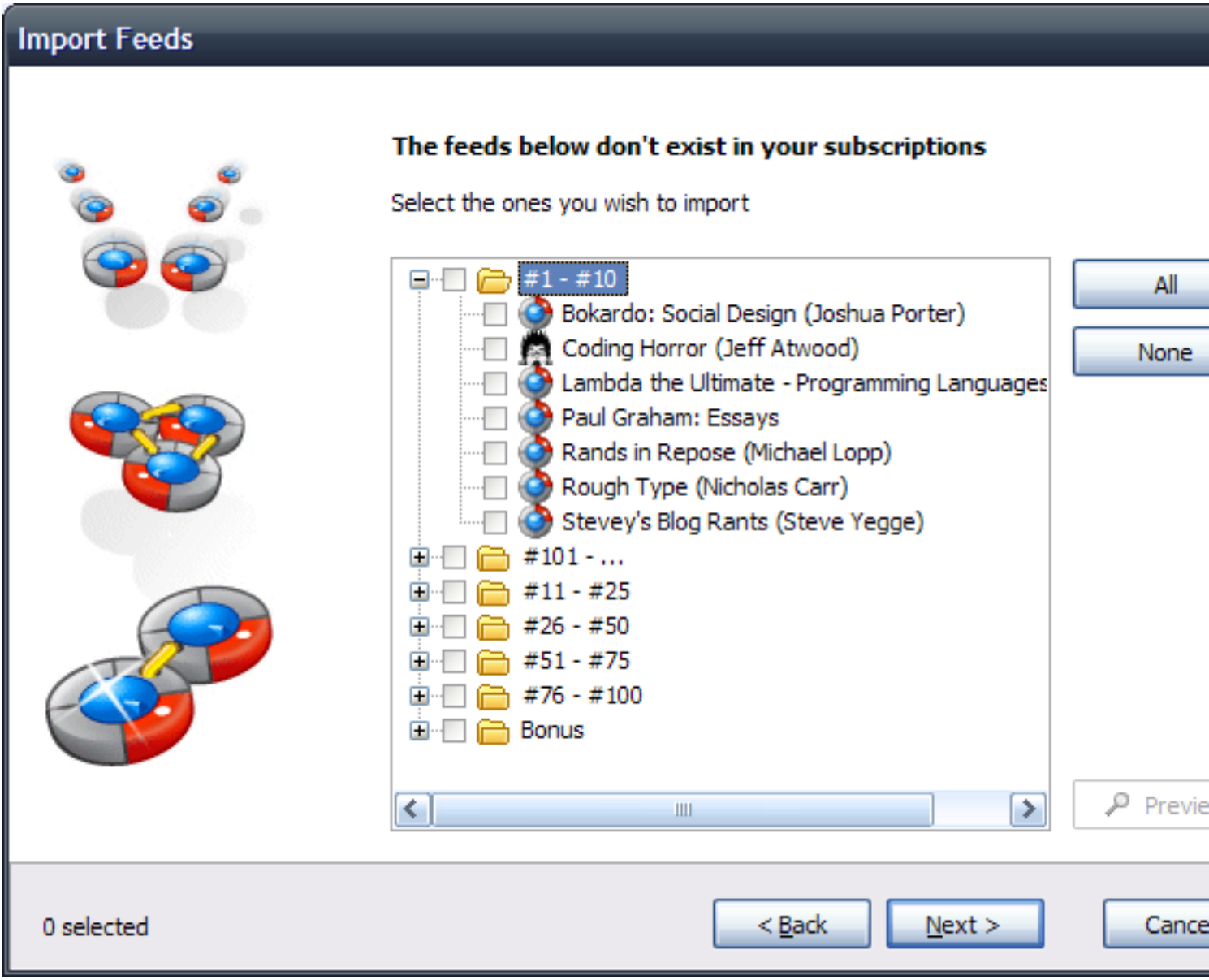
```
<?xml version="1.0" encoding="UTF-8"?>
<opml version="1.0">
  <head>
      <title>Subscriptions in Google Reader</title>
  </head>
  <body>
      <outline title="Programming">
          <outline
               text="Vahid's Blog"
               title="Vahid's Blog"
               type="atom"
               xmlUrl="http://feeds.feedburner.com/vahidnasiri"
               htmlUrl="http://www.dotnettips.info/"/>
.
.
.
```

چند نمونه فایل OPML مرتبط با برنامه نویسی را از سایت‌های مختلف جمع آوری کرده‌ام که آنها را از این آدرس می‌توانید دریافت کنید.

نحوه استفاده از آنها در Google reader
بعد از ورود به قسمت تنظیمات Google reader ، با استفاده از قسمت import/export می‌توان یک فایل OPML را به آن معرفی کرد (شکل زیر):



و یا با استفاده از برنامه باکیفیت و رایگان FeedDemon و قسمت import feeds آن می‌توان یک فایل OPML را به برنامه وارد کرد.
البته این‌جا امکانات بیشتری را نسبت به Google reader دراختیار شما قرار می‌دهد و می‌توانید از لیست دریافتی، موارد مورد نظر را انتخاب کنید و نه تمامی آنها را.



اگر علاقمند بودید که این فایل‌ها را در برنامه‌های دات نت خود import کنید، کتابخانه سورس باز Syndication [Argotic](#) Framework این امکان را در اختیار شما قرار می‌دهد.

نظرات خوانندگان

نویسنده: بهروز راد
تاریخ: ۱۳۸۷/۰۹/۲۶ ۱۳:۳۵:۰۰

آقا این Argotic چیز کاملیه. منم در تیم توسعش هستم.
http://www.codeplex.com/Argotic/People/ProjectPeople.aspx

حتماً استفاده کنید. سوالی هم داشتید هستیم در خدمتتون.

نویسنده: salarblog
تاریخ: ۱۳۸۷/۰۹/۲۶ ۲۳:۰۷:۰۰

این لیستی که گذاشته خیلی بدرد بخور بود
بسیار ممنون

بنظرم بهتره که برجسته ترش کنی تا دید بزنه

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۲۶ ۲۳:۲۱:۰۰

با تشکر از دوستان.
کمی ضخیمش کردم تا مشخص‌تر باشد :)

عنوان: **سایت‌های خبری ایرانی و RSS های بی‌خاصیت آن‌ها!**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۴/۱۱ ۱۲:۱۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: RSS

در طی این چند وقت اخیر هر قدر به سایت‌های خبری داخل کشور مراجعه کردم بیشتر نا امید شدم. آیا واقعا این بزرگواران فکر می‌کنند مردم فرصت این را دارند که روزانه به چند صد سایت خبری سر بزنند؟ این سایت‌ها یا RSS فید ندارند و یا این مشکلات را به همراه دارند:
استاندارد نبودن تاریخ فیدها. (عزیزان برنامه نویس این تاریخ شمسی نیست و نباید باشد! RSS یک فرمت استاندارد است.)
نکته مربوطه

استاندارد نبودن محتوای XML تولید شده (قابل parse نیست!)
چگونه کیفیت فید خروجی را بررسی کنیم؟

بعضی از آن‌ها RSS فید دارند اما باید چند دقیقه در سایت جستجو کنید تا یک لینک را بتوانید در این زمینه پیدا کنید!
چگونه آدرس فید سایت را قابل تشخیص برای IE و فایرفاکس کنیم (Feed autodiscovery)؟

از همه بدتر اینکه خروجی RSS آن‌ها یا چند لینک است بدون توضیح یا چند لینک است بعلاوه یک سطر توضیح. (هدف یک مشترک RSS این است که دیگر به سایت شما مراجعه نکند و مشروح مطالب را از طریق فید دنبال کند. بنابراین یک لینک کافی نیست. یک سطر توضیح هم کم لطفی است. لطفا کل متن خبر را نیز ارائه دهید.)
تعداد در نظر گرفته شده ناکافی مداخل مربوطه. مثلا امروز 20 خبر در سایت درج شده اما فید RSS آن فقط 10 خبر آخر را نمایش می‌دهد. این فید هم تقریبا بدون استفاده است چون حداقل یک روز کامل را پوشش نمی‌دهد.

نظرات خوانندگان

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۴/۱۱ ۲۳:۲۷:۴۳

ba salam va tashakor az matalebe khoobetoon ,sharmande chon ba systemi minivisam ke farsi nadara ru keybordesh age mishe linke matlabi ro dar morede sakhte feed rss baraye site neveshte shode ba c# va mssql , va php va mysql bedin chon man kheyli vagte donbalesh migardam vali matlabe khoobi peyda nemikonam va chon poste shomaro khundam dardham taze shod

mamnoo az tavajohetoon

pishapish

عنوان: **تهیه خروجی RSS در برنامه‌های ASP.NET MVC**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۰ ۱۷:۵۱
آدرس: www.dotnettips.info
گروه‌ها: MVC, RSS, ASP.Net MVC

در این مطلب با کتابخانه تهیه شده جهت تولید فیدهای RSS سایت جاری آشنا خواهید شد. در این کتابخانه مسایل زیر لحاظ شده است:

1) تهیه یک ActionResult جدید به نام FeedResult برای سازگاری و یکپارچگی بهتر با ASP.NET MVC
2) اعمال زبان فارسی به خروجی نهایی (این مورد حداقل در IE محترم شمرده می‌شود و فید را، راست به چپ نمایش می‌دهد)
3) اعمال جهت‌های rtl و ltr به متون فارسی یا انگلیسی به صورت خودکار؛ به نحوی که خروجی نهایی در تمام فیدخوان‌ها یکسان به نظر می‌رسد.
4) اعمال کاراکتر یونیکد RLE به صورت خودکار به عناوین فارسی (این مساله سبب می‌شود تا عنوان‌های ترکیبی متشکل از حروف و کلمات فارسی و انگلیسی، در فیدخوان‌هایی که متون را، راست به چپ نمایش نمی‌دهند، صحیح و بدون مشکل نمایش داده شود.)
5) نیازی به کتابخانه اضافی خاصی ندارد و پایه آن فضای نام System.ServiceModel.Syndication دات نت است.
6) تنظیم صحیح ContentEncoding و ContentType جهت نمایش بدون مشکل متون فارسی

سورس کامل این کتابخانه به همراه یک مثال استفاده از آن را از اینجا می‌توانید دریافت کنید:
MvcRssApplication.zip

توضیحاتی در مورد نحوه استفاده از آن

کتابخانه کمکی MvcRssHelper به صورت یک پروژه Class library جدا تهیه شده است. بنابراین تنها کافی است ارجاعی را به اسمبلی آن به پروژه خود اضافه کنید. البته این پروژه برای MVC4 کامپایل شده است؛ اما با MVC3 هم قابل کامپایل است. فقط باید ارجاع به اسمبلی System.Web.Mvc.dll را حذف و نمونه MVC3 آن‌را جایگزین کنید.
پس از اضافه کردن ارجاعی به اسمبلی آن، اکشن متد فید شما یک چنین امضایی را باید بازگشت دهد:

```
FeedResult(string feedTitle, IList<FeedItem> rssItems, string language = "fa-IR")
```

آیتم اول، نام فید است. مورد دوم، لیست عناوینی است که قرار است در فید ظاهر شوند. برای مثال، هر بار 20 آیتم آخر مطالب سایت را گزارش‌گیری کرده و به فرمت لیستی از FeedItemها باید ارائه دهید. FeedItem هم یک چنین ساختاری دارد و در اسمبلی MvcRssHelper قرار گرفته:

```
using System;

namespace MvcRssHelper
{
    public class FeedItem
    {
        public string Title { set; get; }
        public string AuthorName { set; get; }
        public string Content { set; get; }
        public string Url { set; get; }
        public DateTime LastUpdatedTime { set; get; }
    }
}
```

دو نکته در اینجا حائز اهمیت است:
الف) تاریخ استاندارد یک فید میلادی است نه شمسی. به همین جهت DateTime در اینجا ظاهر شده است.
ب) Url آدرسی است به مطلب متناظر در سایت و باید یک آدرس مطلق مثلا شروع شده با http باشد.

یک مثال از استفاده آن

فرض کنید مدل مطالب سایت ما به نحو زیر است:

```
using System;

namespace MvcRssApplication.Models
{
    public class Post
    {
        public int Id { set; get; }
        public string Title { set; get; }
        public string AuthorName { set; get; }
        public string Body { set; get; }
        public DateTime Date { set; get; }
    }
}
```

و یک منبع داده فرضی (کوئری از بانک اطلاعاتی یا استفاده از یک ORM یا ... موارد دیگر) نهایتا تعدادی رکورد را در اختیار ما خواهد گذاشت:

```
using System;
using System.Collections.Generic;
using MvcRssApplication.Models;

namespace MvcRssApplication.DataSource
{
    public static class BlogItems
    {
        public static IList<Post> GetLastBlogPostsList()
        {
            var results = new List<Post>();
            for (int i = 1; i < 21; i++)
            {
                results.Add(new Post
                {
                     AuthorName = "شخص " + i,
                     Body = "مطلب " + i,
                     Date = DateTime.Now.AddDays(-i),
                     Id = i,
                     Title = "عنوان "+i
                });
            }
            return results;
        }
    }
}
```

اکنون برای نمایش این اطلاعات به صورت یک فید، تنها کافی است به صورت زیر عمل کنیم:

```
using System.Collections.Generic;
using System.Web.Mvc;
using MvcRssApplication.DataSource;
using MvcRssApplication.Models;
using MvcRssHelper;

namespace MvcRssApplication.Controllers
{
    public class HomeController : Controller
    {
        const int Min15 = 900;

        [OutputCache(Duration = Min15)]
        public ActionResult Index()
        {
            var list = BlogItems.GetLastBlogPostsList();
            var feedItemsList = mapPostsToFeedItems(list);
            return new FeedResult("فید مطالب سایت ما", feedItemsList);
        }

        private List<FeedItem> mapPostsToFeedItems(IList<Post> list)
        {
            var feedItemsList = new List<FeedItem>();
            foreach (var item in list)
```

```
            {
                feedItemsList.Add(new FeedItem
                {
                    AuthorName = item.AuthorName,
                    Content = item.Body,
                    LastUpdatedTime = item.Date,
                    Title = item.Title,
                    //این آدرس باید مطلق باشد به نحو زیر
                    Url = this.Url.Action(actionName: "Details", controllerName: "Post", routeValues:
new { id = item.Id }, protocol: "http")
                });
            }
            return feedItemsList;
        }
    }
}
```

توضیحات

BlogItems.GetLastBlogPostsList منبع داده فرضی ما است. در ادامه باید این اطلاعات را به صورت لیستی از FeedItemها در آوریم. می‌توانید از AutoMapper استفاده کنید یا در این مثال ساده، نحوه انجام کار را در متد mapPostsToFeedItems ملاحظه می‌کنید.

نکته مهم بکارگرفته شده در متد mapPostsToFeedItems، استفاده از Url.Action برای تولید آدرس‌هایی مطلق متناظر با کنترلر نمایش مطالب سایت است.

پس از اینکه feedItemsList نهایی به صورت پویا تهیه شد، تنها کافی است return new FeedResult را به نحوی که ملاحظه می‌کنید فراخوانی کنیم تا خروجی حاصل به صورت یک فید RSS نمایش داده شود و قابل استفاده باشد. ضمنا جهت کاهش بار سرور می‌توان از OutputCache نیز به مدتی مشخص استفاده کرد.

نظرات خوانندگان

نویسنده: علیرضا اسم‌رام
تاریخ: ۱۳۹۱/۰۶/۲۰ ۱۹:۸

سلام. بسیار کاربردی و عالی. سپاس.

نویسنده: احمدعلی شفیعی
تاریخ: ۱۳۹۱/۰۶/۲۱ ۰:۳۳

یک‌سری از وبگاه‌ها عادت دارن از description به‌جای content استفاده کنند.آیا این کتابخانه این قابلیت رو داره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۱ ۰:۴۳

خروجی کتابخانه فوق، RSS استاندارد است و description دارد. مقادیر خواص کلاس FeedItem نهایتا به این خروجی نگاشت می‌شود.
برای نمونه این خروجی رو می‌تونید در سورس نهایی فید سایت مشاهده کنید.

نویسنده: محمد صافدل
تاریخ: ۱۳۹۱/۰۶/۲۱ ۱۰:۳۰

ممنون آقای نصیری. بسیار عالی بود و ابهامات زیادی در مورد Rss برای من برطرف شد.

نویسنده: محمد صافدل
تاریخ: ۱۳۹۱/۰۶/۲۳ ۱۵:۲۴

در صورت امکان در مورد نحوه استفاده از کتابخانه MvcRssHelper در پروژه‌های ASP.NET و تغییراتی که باید در این کتابخانه انجام شود، راهنمایی کنید. با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۳ ۱۶:۲۲

آنچنان تفاوتی نداره. فقط اینبار باید از فایل‌های ashx استفاده کنید. روال ProcessRequest آن معادل روال ExecuteResult موجود در فایل FeedResult.cs است.

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۷/۲۲ ۱۹:۹

سلام
ممنوم بابت مطلب خوبتون
ولی فایل که برای دانلود گذاشتید مشکل داره
پروژه ام وی سی توش نیست ... ممنونم می‌شم اگر اصلاح کنید

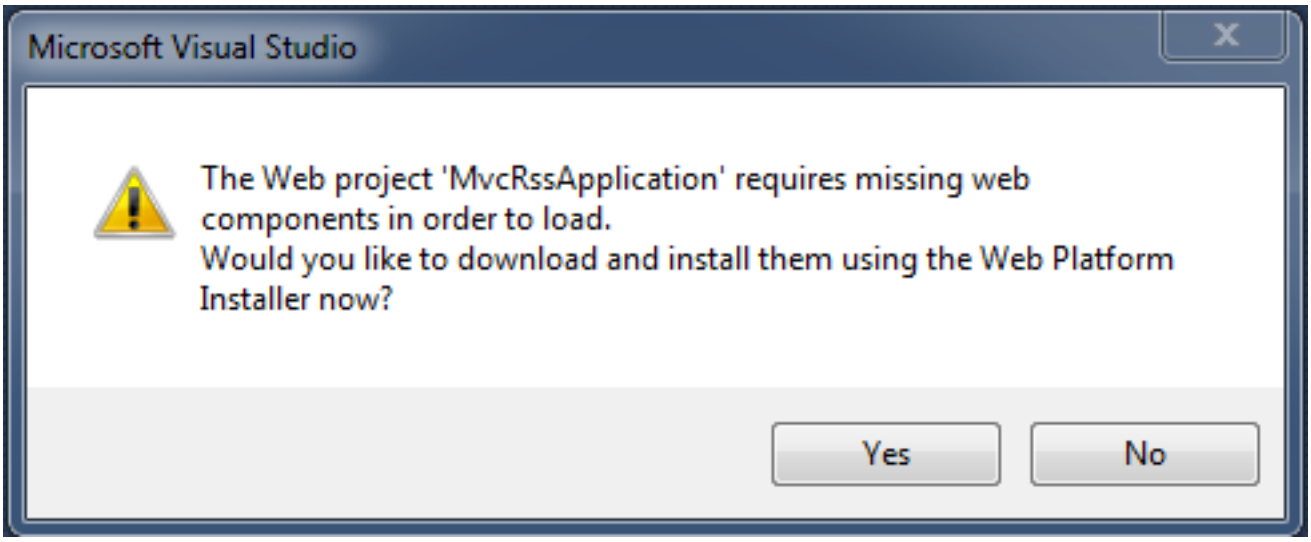
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۲۲ ۱۹:۴۱

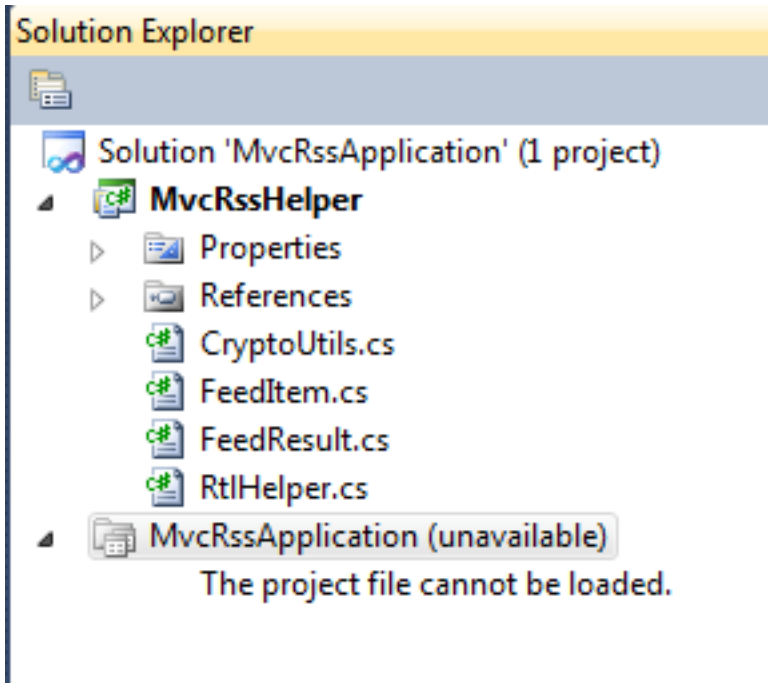
داخل فایل MvcRssApplication.zip فوق این موارد هست:
پوشه MvcRssApplication : یک پروژه مثال

پوشه MvcRssHelper : کتابخانه اصلی

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۷/۲۳ ۱۵:۲۲

سلام، من هم توی لود پروژه مشکل دارم، البته همه نسخه‌های mvc رو نصب شده داشتم ولی:





نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۲۳ ۱۶:۲۶

- این یک پروژه MVC4 است. همچنین فرض هم بر این است که IIS Express بر روی سیستم شما نصب است (^).

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۳۰ ۰:۱۱

نسخه به روز شده این پروژه:

MvcRssApplication.zip

تغییرات:
پیشتر فقط تاریخ به روز رسانی را داشت:

```
<a10:updated>2012-10-20T16:09:13+03:30</a10:updated>
```

الان تاریخ انتشار هم به آن اضافه شده:

```
<pubDate>Sat, 30 Jan 2010 02:26:32 -0800</pubDate>
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۸/۰۲ ۲۳:۷

به روز رسانی سوم:

MvcRssApplication3.zip

تغییرات (بر اساس نظرات W3C Feed Validation Service):
- channel link به آن اضافه شد.
- فضای نام a10 ایی که تولید می‌شد با atom جایگزین شد.

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۸/۱۳ ۱۷:۳

سلام

از این روش می‌شه برای ساخت sitemap هم استفاده کرد؟
اگر نه چطور؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۴/۰۱/۱۰ ۱:۵۵

یک نکته‌ی تکمیلی

فید سایت امروز از کار افتاده بود. علت آن وجود یک سری کاراکتر غیرمجاز XML در متن بود که باید به نحو ذیل پاک شوند:

```
private static readonly Regex _matchHexadecimalSymbols =
            new Regex("[\x00-\x08\x0B\x0C\x0E-\x1F]", RegexOptions.IgnoreCase | RegexOptions.Compiled);

        /// <summary>
        /// there are a lot of symbols which can't be in xml code.
        /// </summary>
        public static string RemoveHexadecimalSymbols(this string txt)
        {
            return string.IsNullOrWhiteSpace(txt) ?
                string.Empty : _matchHexadecimalSymbols.Replace(txt, string.Empty);
        }
```

در کدهای مطلب فوق، مقادیر content و title باید پیش از استفاده، توسط این متد پاکسازی شوند.

نویسنده: سانی
تاریخ: ۱۳۹۴/۰۱/۳۰ ۲۳:۲۱

خیلی متشکر، کدها بسیار کار امد بود و بدون دردسر در پروژه جواب داد
فقط این که مثلا باید ادرس http://localhost:40252/feed/rss را وارد کرد ، حال این شکل rss (مربع نارنجی قرمز بالا چپ ادرس بار در گوگل کروم)که در نوار ابزار نمایش داده میشود رو چه تغییراتی باید انجام بدهم تا در ادرس بار نمایش داده شود ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۴/۰۱/۳۱ ۰:۲۷

به هدر صفحه باید این مورد جهت فعال سازی auto discovery اضافه شود:

```
<link title="فید آخرین تغییرات سایت" href="/rss.xml" type="application/rss+xml" rel="alternate" />
```

محدودیتی هم ندارد. هر تعدادی که نیاز است.

عنوان: **روشی سریع برای ایجاد RSS و Sitemap در ASP.NET MVC**
نویسنده: مرتضی دلیل
تاریخ: ۱۳۹۳/۰۴/۲۵ ۲۱:۲۰
آدرس: www.dotnettips.info
گروه‌ها: MVC, RSS, sitemap

برای مطالعه روش‌های بدست آوردن خروجی xml مربوط به Rss و Sitemap، میتوانید از مقالات مشخص شده استفاده کنید .[اینجا] و [اینجا].
در صورتیکه طراحی شما بر اساس MVC صورت گرفته است، در کمتر از چند دقیقه و در سه مرحله میتوانید پرونده Rss و Sitemap را برای همیشه ببندید.

پیش از تشریح مراحل، به ساختار این دو فایل توجه کنید.

مراحل کار :

مرحله 1. ایجاد نوع(Type) مورد نیاز برای ایجاد Xml های فوق

مرحله 2 . ایجاد کنترلر XML

مرحله 3. ایجاد مسیریابی(Routing)

مرحله 1 : ابتدا یک کلاس به منظور شکل دهی به اطلاعات، بر اساس خواسته‌های xml مرتبط با RSS و Sitemap تشکیل دهید:

```
public class PostToXml
    {

        public int PostId { get; set; }

        public string title { get; set; }

        public string link { get; set; }

        public string description { get; set; }

        public Nullable<DateTime> pubDate { get; set; }
```

```
    }
```

مرحله 2 : یک کنترلر به نام xml ایجاد کنید و اکشن متدهای زیر را درون آن قرار دهید :

```
public ContentResult RSS()
        {

            var items = GetRssFeed();
            var rss = new XDocument(new XDeclaration("1.0", "utf-8", "yes"),
              new XElement("rss",
                new XAttribute("version", "2.0"),
                  new XElement("channel",
                    new XElement("title", "آخرین مطالب سایت"),
                    new XElement("link", "http://" + Request.Url.Host+"/rss"),
                    new XElement("description", "آخرین مطالب سایت من"),
                    new XElement("copyright","(c)" + DateTime.Now.Year + ", نام سایت من.تمامی حقوق
محفوظ است"),
                  from item in items
                  select
                  new XElement("item",
                    new XElement("title", item.title),
                    new XElement("description", item.description),
                    new XElement("link", item.link),
                    new XElement("pubDate", item.pubDate)

                  )
                )
              )
            );
            return Content(rss.ToString(), "text/xml");
        }



        public ContentResult Sitemap()
        {
            XNamespace ns = "http://www.sitemaps.org/schemas/sitemap/0.9";
            var items = GetLinks();
            var sitemap = new XDocument(new XDeclaration("1.0", "utf-8", "yes"),
                new XElement(ns + "urlset",
                    from item in items
                    select
                    new XElement("url",
                      new XElement("loc", item.link),
                      new XElement("changefreq", "monthly"),
                      new XElement("priority", "0.5")
                      )
                    )
                  );
            return Content(sitemap.ToString(), "text/xml");
        }


        public IEnumerable<PostToXml> GetRssFeed()
        {
          // یک کوئری که لیستی از تایپ مشخص شده به ما بدهد
        }

         public IEnumerable<PostToXml> GetLinks()
        {
            // یک کوئری که لیستی از تایپ مشخص شده به ما بدهد
        }
```

این کنترلر دارای دو اکشن متد Rss و Sitemap است و این اکشن‌ها وظیفه‌ی ایجاد فایل‌های Xml را به عهده دارند. مواد اولیه این xml ها از دو متد GetRssFeed و GetLinks تهیه می‌شوند. ما این مواد را در تمپلیت Rss و Sitemap جایگذاری خواهیم کرد. (به کمک دو اکشن متد Rss و Sitemap)

کافیست لیستی از مواردی را که می‌خواهیم در Rss یا Sitemap ثبت شوند، تهیه کنیم. این لیست بر اساس شکل تنظیم دیتابیس و مسیریابی سایت، می‌تواند پیچیده و یا ساده باشد. (به کمک کوئری گرفتن با linq و یا اضافه کردن مستقیم آدرس‌ها به لیست و یا

ترکیبی از هر دو مورد) برای درک بهتر موضوع، لطفا تصویر موجود در ابتدای مقاله را مشاهده نمایید.

مرحله 3 : در مرحله آخر کافیست دو مورد زیر را به فایل RoutConfig.cs بیافزایید:

```
routes.MapRoute(
              "Sitemap", "sitemap",
              new { controller = "XML", action = "Sitemap" });
 routes.MapRoute(
              "RSS", "rss",
               new { controller = "XML", action = "RSS" });
```

به کمک آدرس‌های زیر می‌توانید به آنچه که تهیه کرده‌اید دسترسی داشته باشید :

```
http://domain.com/rss
http://domain.com/sitemap
```

فایل پروژه را دریافت کنید :

MVC_RSS_Sitemap-43ad3c6681734b34b91deaaabcdba871.rar

نظرات خوانندگان

نویسنده: محمد دلیری
تاریخ: ۱۳۹۳/۰۴/۲۶ ۰:۲۴

لطفا فایل‌های پروژه را هم اضافه کنید.

نویسنده: مرتضی دلیل
تاریخ: ۱۳۹۳/۰۴/۲۶ ۱۵:۳۵

اضافه شد.

نویسنده: میثم کریمی
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۷:۳۹

خیلی ممنون از شما که دو تا از مشکلات من رو حل کردید .
فقط یک سوال اینکه این مورد SiteMap اگر بخواهیم از چندین جدول باشد و اینکه یکسری آدرس رو هم که استاتیک هستند دستی اضافه کنیم چه کار باید بکنیم ؟ می‌شه یک مثال بزنید ؟ خیلی ممنون می‌شم
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۸:۲۶

این متدها در نهایت با لیستی از PostToXML کار می‌کنند. یعنی برای استفاده از آن‌ها باید اطلاعات خودتان را به فرمت لیستی از PostToXML تبدیل کنید؛ برای مثال توسط مباحث LINQ Projection که نمونه‌ای از آن در مثال پیوستی ذکر شده:

```
List<PostToXML> sampleposts = (from p in PostsFromDb
                select new PostToXML()
                {
                    description = p.description,
                    link = "http://" + Request.Url.Host+"/news/"+p.postname,
                    pubDate = p.pubDate,
                    title = p.title
                }).ToList();
```

به این صورت از چندین جدول، چندین لیست PostToXML خواهید داشت. مهم هم نیست که اطلاعات این لیست از جدولی تهیه می‌شود یا صرفا با متد Add اضافه شده‌اند (استفاده کننده از آن کاری به منبع داده ندارد).
نهایتا برای یکی کردن چندین لیست PostToXML به یک لیست PostToXML جهت استفاده در این متدها، از متد Concat و یا Union استفاده کنید:

```
List<PostToXML> total = list1.Concat(list2);
```

نویسنده: میثم سام
تاریخ: ۱۳۹۳/۱۱/۲۱ ۲۲:۴۶

سلام؛ من یه سایت دارم مینویسم که همش دینامیک و یه سری شرکت توش هست. میخوام مثلا اسم این شرکت‌ها و لینک هاشونو از توی دیتابیس بخونم و توی سایت مپ قرار بدم. لازم به ذکر که این شرکت‌ها هر کدومشون توی همین سایت یه صفحه برای خودشون دارن. من کل مبحث سایت مپ متوجه اش شدم ولی یه چیز اصلا نفهمیدم. کجا باید این تابع فرخوانی کرد. مثلا وقتی این شرکت داره اضافه میشه داخل دیتابیس . یا وقتی که میخوایم این صفحه خود شرکت باز بشه. یا نه وقتی خود لینک مثلا : http://www.domain.com/sitemap
میزنیم همون موقع باید از دیتابیس اسم شرکت‌ها رو بخونیم و لینکشونو وارد سایت مپ کنیم و هیچ کاری به کنترل‌های دیگه نداره

و این که توی سایت خوبتون چند تا روش برای سایت مپ گفته شده به نظر خودتون کدوم یک بهتره
خیلی ممنونم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۱/۲۱ ۲۲:۵۷

- «کجا باید این تابع فرخوانی کرد»
دقیقا مطابق مثالی که پیوست کردند، در اکشن متدی به همین نام. کار بیشتری هم لازم نیست انجام شود. هر زمانیکه یک موتور جستجو به سایت شما رسید، این آدرس را واکشی می‌کند (اینکه از کجا باید بداند این آدرس را نیاز است واکشی کند، مرتبط است به مباحث فایل robots.txt و قید صریح آدرس آن). پس از واکشی خودکار آن، از بانک اطلاعاتی گزارش تهیه کرده و لیستی از PostToXML را مطابق این مطلب بازگشت می‌دهید.
- « به نظر خودتون کدوم یک بهتره »
فرقی نمی‌کنند. خروجی نهایی یک استاندارد بیشتر ندارد.

عنوان: **نوشتن افزونه برای مرورگرها: قسمت دوم : کروم**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۱/۰۴ ۱:۱۵
آدرس: www.dotnettips.info
گروه‌ها: RSS, Plugin, Google Chrome, Extension, Addon

در مقاله پیشین ما ظاهر افزونه را طراحی و یک سری از قابلیت‌های افزونه را معرفی کردیم. در این قسمت قصد داریم پردازش پس زمینه افزونه یعنی خواندن RSS و اعلام به روز آوری سایت را مورد بررسی قرار دهیم و یک سری قابلیت هایی که گوگل در اختیار ما قرار داده است.

خواندن RSS توسط APIهای گوگل

گوگل در تعدادی از زمینه‌ها و سرویس‌های خودش apiهایی را ارائه کرده است که یکی از آن ها خواندن فید است و ما از آن برای خواندن RSS یا اتم وب سایت کمک می‌گیریم. روند کار بدین صورت است که ابتدا ما بررسی می‌کنیم کاربر چه مقادیری را ثبت کرده است و افزونه قرار است چه بخش هایی از وب سایت را بررسی نماید. در این حین، صفحه پس زمینه شروع به کار کرده و در هر سیکل زمانی مشخص شده بررسی می‌کند که آخرین بار چه زمانی RSS به روز شده است. اگر از تاریخ قبلی بزرگتر باشد، پس سایت به روز شده است و تاریخ جدید را برای دفعات آینده جایگزین تاریخ قبلی کرده و یک پیام را به صورت نوتیفیکیشن جهت اعلام به روز رسانی جدید در آن بخش به کاربر نشان می‌دهد.
اجازه دهید کدها را کمی شکیل‌تر کنیم. من از فایل زیر که یک فایل جاوااسکریپتی است برای نگه داشتن مقادیر بهره می‌برم تا اگر روزی خواستم یکی از آن‌ها را تغییر دهم راحت باشم و در همه جا نیاز به تغییر مجدد نداشته نباشم. نام فایل را (const.js) به خاطر ثابت بودن آن‌ها انتخاب کرده‌ام.

```
برای ذخیره مقادیر از ساختار نام و مقدار استفاده می‌کنیم که نام‌ها را اینجا ثبت کرده ام//
var Variables={
 posts:"posts",
 postsComments:"postsComments",
 shares:"shares",
 sharesComments:"sharesComments",
}

برای ذخیره زمان آخرین تغییر سایت برای هر یک از مطالب به صورت جداگانه نیاز به یک ساختار نام و مقدار//
است که نام‌ها را در اینجا ذخیره کرده ام
var DateContainer={
 posts:"dtposts",
 postsComments:"dtpostsComments",
 shares:"dtshares",
 sharesComments:"dtsharesComments",
 interval:"interval"
}

//برای نمایش پیام‌ها به کاربر
var Messages={
SettingsSaved:"تنظیمات ذخیره شد",
SiteUpdated:"سایت به روز شد",
PostsUpdated:"مطلب ارسالی جدید به سایت اضافه شد",
CommentsUpdated:"نظری جدیدی در مورد مطالب سایت آرسال شد",
SharesUpdated:"اشتراک جدید به سایت ارسال شد",
SharesCommentsUpdated:"نظری برای اشتراک‌های سایت اضافه شد"
}
//لینک‌های فید سایت
var Links={
 postUrl:"http://www.dotnettips.info/feeds/posts",
 posts_commentsUrl:"http://www.dotnettips.info/feeds/comments",
 sharesUrl:"http://www.dotnettips.info/feed/news",
 shares_CommentsUrl:"http://www.dotnettips.info/feed/newscomments"
}
//لینک صفحات سایت
var WebLinks={
Home:"http://www.dotnettips.info",
 postUrl:"http://www.dotnettips.info/postsarchive",
 posts_commentsUrl:"http://www.dotnettips.info/commentsarchive",
 sharesUrl:"http://www.dotnettips.info/newsarchive",
 shares_CommentsUrl:"http://www.dotnettips.info/newsarchive/comments"
}
```

موقعی که اولین بار افزونه نصب می‌شود، باید مقادیر پیش فرضی وجود داشته باشند که یکی از آن‌ها مربوط به مقدار سیکل زمانی

است (هر چند وقت یکبار فید را چک کند) و دیگری ذخیره مقادیر پیش فرض رابط کاربری که قسمت پیشین درست کردیم؛ پروسه پس زمینه برای کار خود به آن‌ها نیاز دارد و بعدی هم تاریخ نصب افزونه است برای اینکه تاریخ آخرین تغییر سایت را با آن مقایسه کند که البته با اولین به روزرسانی تاریخ فید جای آن را می‌گیرد. جهت انجام اینکار یک فایل init.js ایجاد کرده‌ام که قرار است بعد از نصب افزونه، مقادیر پیش فرض بالا را ذخیره کنیم.

```
chrome.runtime.onInstalled.addListener(function(details) {
var now=String(new Date());

var params={};
params[Variables.posts]=true;
params[Variables.postsComments]=false;
params[Variables.shares]=false;
params[Variables.sharesComments]=false;

params[DateContainer.interval]=1;

params[DateContainer.posts]=now;
params[DateContainer.postsComments]=now;
params[DateContainer.shares]=now;
params[DateContainer.sharesComments]=now;

 chrome.storage.local.set(params, function() {
  if(chrome.runtime.lastError)
   {
       /* error */
       console.log(chrome.runtime.lastError.message);
       return;
   }
        });
});
```

chrome.runtime شامل رویدادهایی چون onSuspend ، onStartup ، onInstalled و ... است که مربوطه به وضعیت اجرایی افزونه میشود. آنچه ما اضافه کردیم یک listener برای زمانی است که افزونه نصب شده است و در آن مقادیر پیش فرض ذخیره می‌شوند. اگر خوب دقت کنید می‌بینید که روش دخیره سازی ما در اینجا کمی متفاوت از مقاله پیشین هست و شاید پیش خودتان بگویید که احتمالا به دلیل زیباتر شدن کد اینگونه نوشته شده است ولی مهمترین دلیل این نوع نوشتار این است که متغیرهای بین {} آنچنان فرقی با خود string نمی‌کنند یعنی کد زیر:

```
chrome.storage.local.set('mykey':myvalue,....
```

با کد زیر برابر است:

```
chrome.storage.local.set(mykey:myvalue,...
```

پس اگر مقداری را داخل متغیر بگذاریم آن مقدار حساب نمی‌شود؛ بلکه کلید نام متغیر خواهد شد.
برای معرفی این دو فایل const.js و init.js به manifest.json می‌توانید به صورت زیر عمل کنید:

```
"background": {
    "scripts": ["const.js","init.js"]
}
```

در این حالت خود اکستنشن در زمان نصب یک فایل html درست کرده و این دو فایل js را در آن صدا میزند که البته خود ما هم می‌توانیم اینکار را مستقیما انجام دهیم. مزیت اینکه ما خودمان مسقیما این کار را انجام دهیم این است که در صورتی که فایل‌های js ما زیاد شوند، فایل manifest.jason زیادی شلوغ شده و شکل زشتی پیدا می‌کند و بهتر است این فایل را تا آنجا که می‌توانیم خلاصه نگه داریم. البته روش بالا برای دو یا سه تا فایل js بسیار خوب است ولی اگر به فرض بشود 10 تا یا بیشتر بهتر است یک فایل جداگانه شود و من به همین علت فایل background.htm را درست کرده و به صورت زیر تعریف کرده‌ام:
نکته:نمی توان در تعریف بک گراند هم فایل اسکریپت معرفی کرد و هم فایل html

```
"background": {
    "page": "background.htm"
}
```

background.htm

```
<html>
  <head>
  <script type="text/javascript" src="const.js"></script>
    <script type="text/javascript" src="https://www.google.com/jsapi"></script>
    <script type="text/javascript" src="init.js"></script>
<script type="text/javascript" src="omnibox.js"></script>
<script type="text/javascript" src="rssreader.js"></script>
<script type="text/javascript" src="contextmenus.js"></script>
  </head>
  <body>
  </body>
</html>
```

لینک‌های بالا به ترتیب معرفی ثابت‌ها، لینک api گوگل که بعدا بررسی می‌شود، فایل init.js برای ذخیره مقادیر پیش فرض، فایل ominibox که در مقاله پیشین در مورد آن صحبت کردیم و فایل rssreader.js که جهت خواندن rss در پایینتر در موردش بحث می‌کنیم و فایل contextmenus که این را هم در مطلب پیشین توضیح دادیم.
جهت خواندن فید سایت ما از Google API استفاده می‌کنیم؛ اینکار دو دلیل دارد:
کدنویسی راحت‌تر و خلاصه‌تر برای خواندن RSS
استفاده اجباری از یک پروکسی به خاطر Content Security Policy و حتی CORS

قبل از اینکه manifst به ورژن 2 برسد ما اجازه داشتیم کدهای جاوااسکریپت به صورت inline در فایل‌های html بنویسیم و یا اینکه از منابع و آدرس‌های خارجی استفاده کنیم برای مثال یک فایل jquery بر روی وب سایت jquery ؛ ولی از ورژن 2 به بعد، گوگل سیاست امنیت محتوا Content Security Policy را که سورس و سند اصلی آن در اینجا قرار دارد، به سیستم Extension خود افزود تا از حملاتی قبیل XSS و یا تغییر منبع راه دور به عنوان یک malware جلوگیری کند. پس ما از این به بعد نه اجازه داشتیم inline بنویسیم و نه اجازه داشتیم فایل jquery را از روی سرورهای سایت سازنده صدا بزنیم. پس برای حل این مشکل، ابتدا مثل همیشه یک فایل js را در فایل html معرفی می‌کردیم و برای حل مشکل دوم باید منابع را به صورت محلی استفاده می‌کردیم؛ یعنی فایل jquery را داخل دایرکتوری extension قرار می‌دادیم.
برای حل مشکل مشکل صدا زدن فایل‌های راه دور ما از Relaxing the Default Policy استفاده می‌کنیم که به ما یک لیست سفید ارائه می‌کند و در این لیست سفید دو نکته‌ی مهم به چشم میخورد که یکی از آن این است که استفاده از آدرس هایی با پروتکل Https و آدرس لوکال local host/127.0.0.1 بلا مانع است و از آنجا که api گوگل یک آدرس Https است، می‌توانیم به راحتی از API آن استفاده کنیم. فقط نیاز است تا خط زیر را به manifest.json اضافه کنیم تا این استثناء را برای ما در نظر بگیرد.

```
"content_security_policy": "script-src 'self' https://*.google.com; object-src 'self'"
```

در اینجا استفاده از هر نوع subdomain در سایت گوگل بلامانع اعلام می‌شود.
بنابراین آدرس زیر به background.htm اضافه می‌شود:

```
<script type="text/javascript" src="https://www.google.com/jsapi"></script>
```

استفاده از این Api در rssreader.js

فایل rssreader.js را به background.htm اضافه می‌کنیم و در آن کد زیر را می‌نویسیم:

```
google.load("feeds", "1");
google.setOnLoadCallback(alarmManager);
```

آدرسی که ما از گوگل درخواست کردیم فقط مختص خواندن فید نیست؛ تمامی apiهای جاوااسکریپتی در آن قرار دارند و ما تنها نیاز داریم قسمتی از آن لود شود. پس اولین خط از دستور بالا بارگذاری بخش مورد نیاز ما را به عهده دارد. در مورد این دستور این صفحه را مشاهده کنید.
در خط دوم ما تابع خودمان را به آن معرفی می‌کنیم تا وقتی که گوگل لودش تمام شد این تابع را اجرا کند تا قبل از لود ما از توابع

آن استفاده نکنیم و خطای undefined دریافت نکنیم. تابعی که ما از آن خواستیم اجرا کند alarmManager نام دارد و قرار است یک آلارم و یک سیکل زمانی را ایجاد کرده و در هر دوره، فید را بخواند. کد تابع مدنظر به شرح زیر است:

```
function alarmManager()
{
chrome.storage.local.get(DateContainer.interval,function ( items) {
period_time==items[DateContainer.interval];
chrome.alarms.create('RssInterval', {periodInMinutes: period_time});
});


chrome.alarms.onAlarm.addListener(function (alarm) {
console.log(alarm);
    if (alarm.name == 'RssInterval') {

var boolposts,boolpostsComments,boolshares,boolsharesComments;
chrome.storage.local.get([Variables.posts,Variables.postsComments,Variables.shares,Variables.sharesComm
ents],function ( items) {
boolposts=items[Variables.posts];
boolpostsComments=items[Variables.postsComments];
boolshares=items[Variables.shares];
boolsharesComments=items[Variables.sharesComments];


chrome.storage.local.get([DateContainer.posts,DateContainer.postsComments,DateContainer.shares,DateCont
ainer.sharesComments],function ( items) {

var Vposts=new Date(items[DateContainer.posts]);
var VpostsComments=new Date(items[DateContainer.postsComments]);
var Vshares=new Date(items[DateContainer.shares]);
var VsharesComments=new Date(items[DateContainer.sharesComments]);

if(boolposts){var result=RssReader(Links.postUrl,Vposts,DateContainer.posts,Messages.PostsUpdated);}
if(boolpostsComments){var
result=RssReader(Links.posts_commentsUrl,VpostsComments,DateContainer.postsComments,Messages.CommentsUp
dated); }
if(boolshares){var
result=RssReader(Links.sharesUrl,Vshares,DateContainer.shares,Messages.SharesUpdated);}
if(boolsharesComments){var
result=RssReader(Links.shares_CommentsUrl,VsharesComments,DateContainer.sharesComments,Messages.SharesC
ommentsUpdated);}

});
});
    }
});
}
```

خطوط اول تابع alarmManager وظیفه‌ی خواندن مقدار interval را که در init.js ذخیره کرده‌ایم، دارند که به طور پیش فرض 10 ذخیره شده است تا تایمر یا آلارم خود را بر اساس آن بسازیم. در خط chrome.alarms.create یک آلارم با نام rssinterval می‌سازد و قرار است هر 10 دقیقه وظایفی که بر دوشش گذاشته می‌شود را اجرا کند **(استفاده از api جهت دسترسی به آلارم نیاز به مجوز "alarms" دارد)** . وظایفش از طریق یک listener که بر روی رویداد chrome.alarms.onAlarm گذاشته شده است مشخص می‌شود. در خط بعدی مشخص می‌شود که این رویداد به خاطر چه آلارمی صدا زده شده است. البته از آنجا که ما یک آلارم داریم، نیاز چندانی به این کد نیست. ولی اگر پروژه شما حداقل دو آلارم داشته باشد نیاز است مشخص شود که کدام آلارم باعث صدا زدن این رویداد شده است. در مرحله بعد مشخص می‌کنیم که کاربر قصد بررسی چه قسمت‌هایی از سایت را داشته است و در تابع callback آن هم تاریخ آخرین تغییرات هر بخش را می‌خوانیم و در متغیری نگه داری می‌کنیم. هر کدام را جداگانه چک کرده و تابع RssReader را برای هر کدام صدا می‌زنیم. این تابع 4 پارامتر دارد:

آدرس فیدی که قرار است از روی آن بخواند

آخرین به روزسانی که از سایت داشته متعلق به چه تاریخی است.

نام کلید ذخیره سازی تاریخ آخرین تغییر سایت که اگر بررسی شد و مشخص شد سایت به روز شده است، تاریخ جدید را روی آن ذخیره کنیم.

در صورتی که سایت به روز شده باشد نیاز است پیامی را برای کاربر نمایش دهیم که این پیام را در اینجا قرار می‌دهیم.

کد تابع rssreader

```
function RssReader(URL,lastupdate,datecontainer,Message) {
            var feed = new google.feeds.Feed(URL);
            feed.setResultFormat(google.feeds.Feed.XML_FORMAT);
                    feed.load(function (result) {
if(result!=null)
{
var strRssUpdate = result.xmlDocument.firstChild.firstChild.childNodes[5].textContent;
var RssUpdate=new Date(strRssUpdate);

if(RssUpdate>lastupdate)
{
SaveDateAndShowMessage(datecontainer,strRssUpdate,Message)
}
}
});
}
```

در خط اول فید توسط گوگل خوانده میشود، در خط بعدی ما به گوگل میگوییم که فید خوانده شده را چگونه به ما تحویل دهد که ما قالب xml را خواسته ایم و در خط بعدی اطلاعات را در متغیری به اسم result قرار میدهد که در یک تابع برگشتی آن را در اختیار ما میگذارد. از آن جا که ما قرار است تگ **lastBuildDate** را بخوانیم که پنجمین تگ اولین گره در اولین گره به حساب می‌آید، خط زیر این دسترسی را برای ما فراهم می‌کند و چون تگ ما در یک مکان ثابت است با همین تکه کد، دسترسی مستقیمی به آن داریم:

```
var strRssUpdate = result.xmlDocument.firstChild.firstChild.childNodes[5].textContent;
```

مرحله بعد تاریخ را که در قالب رشته‌ای است، تبدیل به تاریخ کرده و با lastupdate یعنی آخرین تغییر قبلی مقایسه می‌کنیم و اگر تاریخ برگرفته از فید بزرگتر بود، یعنی سایت به روز شده است و تابع SaveDateAndShowMessage را صدا می‌زنیم که وظیفه ذخیره سازی تاریخ جدید و ایجاد notification را به عهده دارد و سه پارامتر کلید ذخیره سازی و مقدار آن و پیام را به آن پاس می‌کنیم.

کد تابع SaveDateAndShowMesage

```
function SaveDateAndShowMessage(DateField,DateValue,Message)
{
var params={
}
params[DateField]=DateValue;

chrome.storage.local.set( params,function(){

var options={
  type: "basic",
   title: Messages.SiteUpdated,
   message: Message,
   iconUrl: "icon.png"
}
chrome.notifications.create("",options,function(){
chrome.notifications.onClicked.addListener(function(){
chrome.tabs.create({'url': WebLinks.Home}, function(tab) {
});
});
});
});
}
```

خطوط اول مربوط به ذخیره تاریخ است و دومین نکته نحوه‌ی ساخت نوتیفکیشن است. اجرای یک notification نیاز به مجوز "notifications " دارد که مجوز آن در manifest به شرح زیر است:

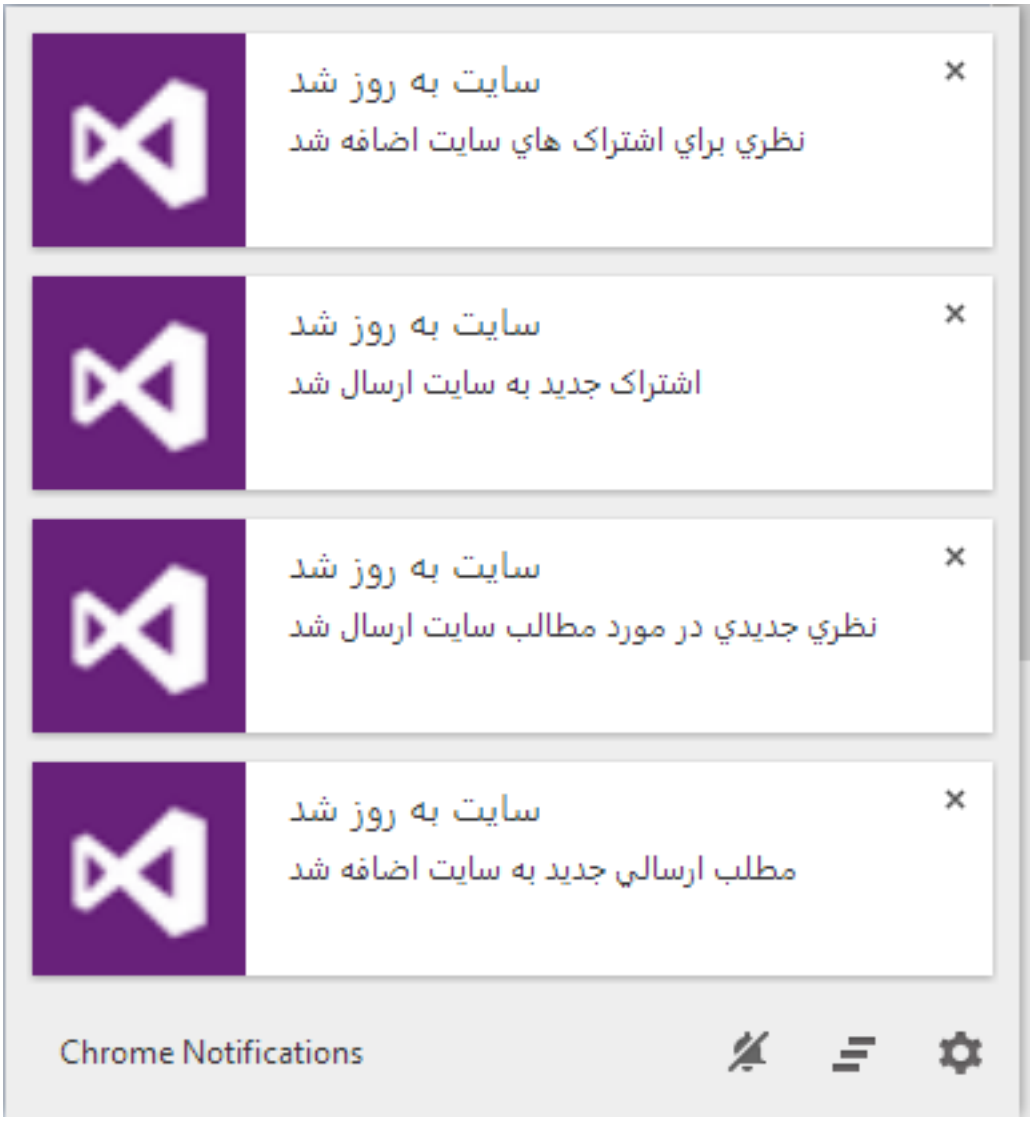
```
"permissions": [
    "storage",
     "tabs",
 "alarms",
 "notifications"
  ]
```

در خطوط بالا سایر مجوزهایی که در طول این دوره به کار اضافه شده است را هم می‌بینید.
برای ساخت نوتیفکیشن از کد chrome.notifications.create استفاده می‌کنیم که پارامتر اول آن کد یکتا یا همان ID جهت ساخت نوتیفیکیشن هست که میتوان خالی گذاشت و دومی تنظیمات ساخت آن است؛ از قبیل عنوان و آیکن و ... که در بالا به اسم options معرفی کرده ایم و در آگومان دوم آن را معرفی کرده ایم و آرگومان سوم هم یک تابع callback است که نوشتن آن اجباری است. options شامل عنوان، پیام، آیکن و نوع notification می‌باشد که در اینجا basic انتخاب کرده‌ایم. برای دسترسی به دیگر خصوصیت‌های options به اینجا و برای داشتن notificationهای زیباتر به عنوان rich notification به اینجا مراجعه کنید. برای اینکه این امکان باشد که کاربر با کلیک روی notification به سایت هدایت شود باید در تابع callback مربوط به notifications.create این کد اضافه گردد که در صورت کلیک یک تب جدید با آدرس سایت ساخته شود:

```
chrome.notifications.create("",options,function(){
chrome.notifications.onClicked.addListener(function(){
chrome.tabs.create({'url': WebLinks.Home}, function(tab) {
});});
});
```

نکته مهم: پیشتر معرفی آیکن به صورت بالا کفایت میکرد ولی بعد از این باگ کد زیر هم باید جداگانه به manifest اضافه شود:

```
"web_accessible_resources": [
    "icon.png"
  ]
```



خوب؛ کار افزونه تمام شده است ولی اجازه دهید این بار امکانات افزونه را بسط دهیم:
من می‌خواهم برای افزونه نیز قسمت تنظیمات داشته باشم. برای دسترسی به options میتوان از قسمت مدیریت افزونه‌ها در مرورگر یا حتی با راست کلیک روی آیکن browser action عمل کرد. در اصل این قسمت برای تنظیمات افزونه است ولی ما به خاطر آموزش و هم اینکه افزونه ما UI خاصی نداشت تنظیمات را از طریق browser action پیاده سازی کردیم و گرنه در صورتی که افزونه شما شامل UI خاصی مثلا نمایش فید مطالب باشد، بهترین مکان تنظیمات، options است. برای تعریف options در manifest.json به روش زیر اقدام کنید:

```
"options_page": "popup.html"
```

همان صفحه popup را در این بخش نشان میدهم و اینبار یک کار اضافه‌تر دیگر که نیاز به آموزش ندارد اضافه کردن input با Type=number است که برای تغییر interval به کار می‌رود و نحوه ذخیره و بازیابی آن را در طول دوره یاد گرفته اید.

جایزگزینی صفحات یا Override Pages

بعضی صفحات مانند بوک مارک و تاریخچه فعالیت‌ها History و همینطور newtab را می‌توانید جایگزین کنید. البته یک اکستنشن میتواند فقط یکی از صفحات را جایگزین کند. برای تعیین جایگزین در manifest اینگونه عمل می‌کنیم:

```
"chrome_url_overrides": {
    "newtab": "newtab.htm"
  }
```

ایجاد یک تب اختصاصی در Developer Tools

تکه کدی که باید manifest اضافه شود:

```
"devtools_page": "devtools.htm"
```

شاید فکر کنید کد بالا الان شامل مباحث ui و ... می‌شود و بعد به مرورگر اعمال خواهد شد؛ در صورتی که اینگونه نیست و نیاز دارد چند خط کدی نوشته شود. ولی مسئله اینست که کد بالا تنها صفحات html را پشتیبانی می‌کند و مستقیما نمی‌تواند فایل js را بخواند. پس صفحه بالا را ساخته و کد زیر را داخلش می‌گذاریم:

```
<script src="devtools.js"></script>
```

فایل devtools.js هم شامل کد زیر می‌شود:

```
chrome.devtools.panels.create(
    "Dotnettips Updater Tools",
    "icon.png",
    "devtoolsui.htm",
    function(panel) {
    }
);
```

خط chrome.devtools.panels.create یک پنل یا همان تب را ساخته و در پارامترهای بالا به ترتیب عنوان، آیکن و صفحه‌ای که باید در آن رندر شود را دریافت می‌کند و پس از ایجاد یک callback اجرا می‌شود. اطلاعات بیشتر

APIها

برای دیدن لیست کاملی از APIها می‌توانید به مستندات آن رجوع کنید و این مورد را به یاد داشته باشید که ممکن است بعضی apiها در بعضی موارد پاسخ ندهند. به عنوان مثال در content scripts نمی‌توانید به chrome.devtools.panels دسترسی داشته باشید یا اینکه در DeveloperTools دسترسی به DOM میسر نیست. پس این مورد را به خاطر داشته باشید. همچنین بعضی apiها از نسخه‌ی خاصی به بعد اضافه شده‌اند مثلا همین مثال قبلی devtools از نسخه 18 به بعد اضافه شده است و به این معنی است با خیال راحت می‌توانید از آن استفاده کنید. یا آلارم‌ها از نسخه 22 به بعد اضافه شده‌اند. البته خوشبختانه امروزه با دسترسی آسانتر به اینترنت و آپدیت خودکار مرورگرها این مشکلات دیگر آن چنان رخ نمی‌دهند.

Messaging

همانطور که در بالا اشاره شد شما نمی‌توانید بعضی از apiها را در بعضی جاها استفاده کنید. برای حل این مشکل می‌توان از messaging استفاده کرد که دو نوع تبادلات پیغامی داریم:

One-Time Requests یا درخواست‌های تک مرتبه‌ای

Long-Lived Connections یا اتصالات بلند مدت یا مصر

درخواست‌های تک مرتبه ای

این درخواست‌ها همانطور که از نامش پیداست تنها یک مرتبه رخ می‌دهد؛ درخواست را ارسال کرده و منتظر پاسخ می‌ماند. به عنوان مثال به کد زیر که در content script است دقت کنید:

```
window.addEventListener("load", function() {
    chrome.extension.sendMessage({
        type: "dom-loaded",
        data: {
            myProperty   : "value"
        }
    });
}, true);
```

کد بالا یک ارسال کننده پیام است. موقعی که سایتی باز می‌شود، یک کلید با مقدارش را ارسال می‌کند و کد زیر در background گوش می‌ایستد تا اگر درخواستی آمد آن را دریافت کند:

```
chrome.extension.onMessage.addListener(function(request, sender, sendResponse) {
    switch(request.type) {
        case "dom-loaded":
            alert(request.data.myProperty   );
        break;
    }
    return true;
});
```

اتصالات بلند مدت یا مصر

اگر نیاز به یک کانال ارتباطی مصر و همیشگی دارید کدها را به شکل زیر تغییر دهید *contentscripts*

```
var port = chrome.runtime.connect({name: "my-channel"});
port.postMessage({myProperty: "value"});
port.onMessage.addListener(function(msg) {
    // do some stuff here
});
```

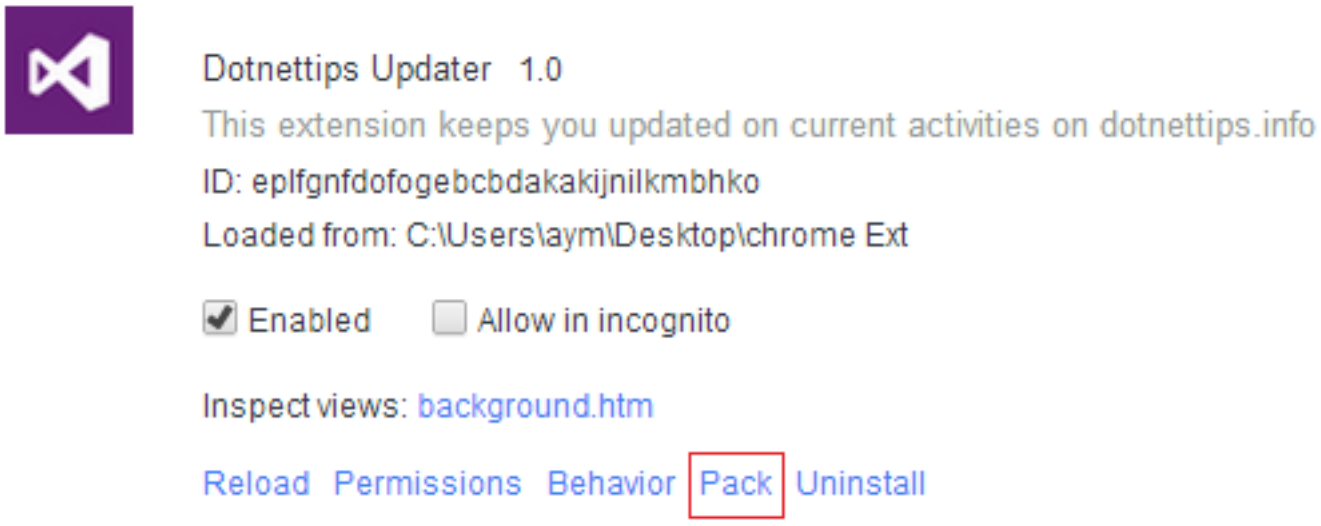
background

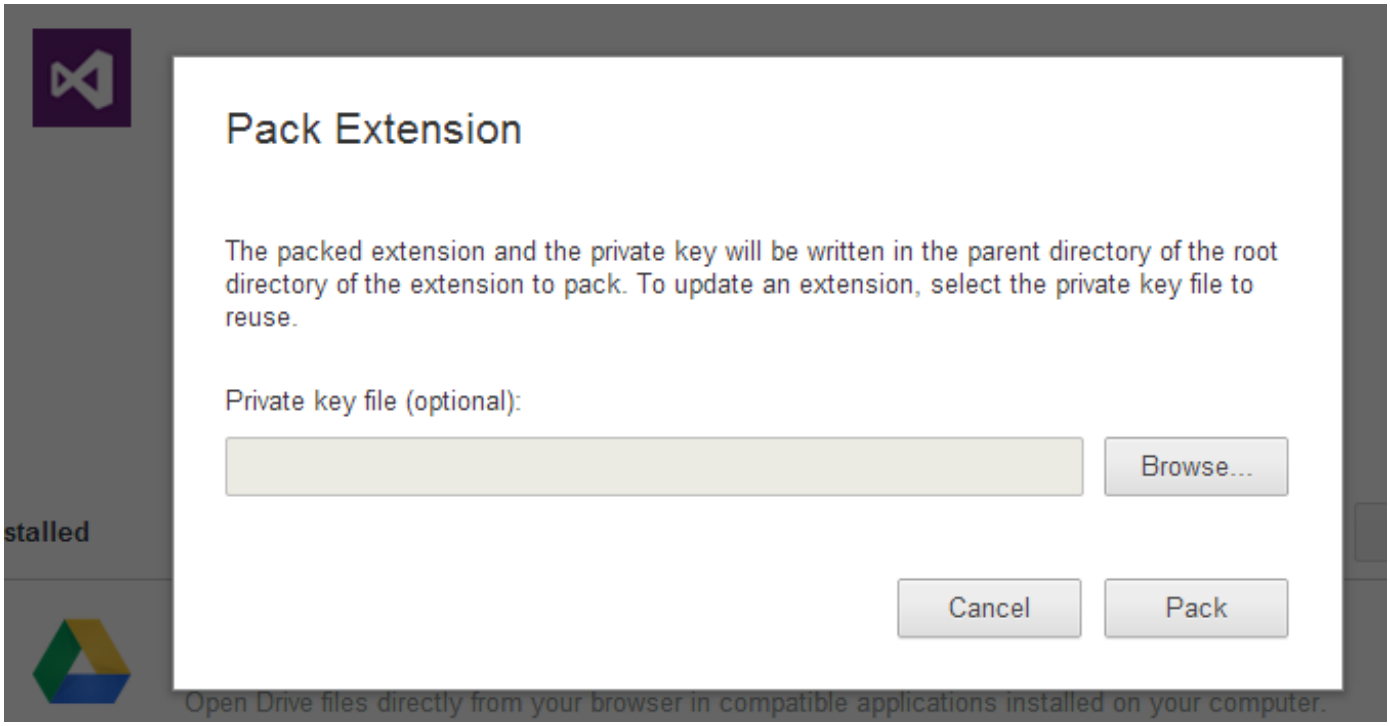
```
chrome.runtime.onConnect.addListener(function(port) {
    if(port.name == "my-channel"){
        port.onMessage.addListener(function(msg) {
            // do some stuff here
        });
    }
});
```

نمونه کد نمونه کدهایی که در سایت گوگل موجود هست می‌توانند کمک بسیاری خوبی باشند ولی اینگونه که پیداست اکثر مثال‌ها مربوط به نسخه‌ی یک manifest است که دیگر توسط مرورگرها پشتیبانی نمی‌شوند و مشکلاتی چون اسکریپت inline و CSP که در بالا اشاره کردیم را دارند و گوگل کدها را به روز نکرده است.

دیباگ کردن و پک کردن فایل‌ها برای تبدیل به فایل افزونه Debugging and packing

برای دیباگ کردن کدها می‌توان از دو نمونه console.log و alert برای گرفتن خروجی استفاده کرد و همچنین ابزار Chrome Apps & Extensions Developer Tool هم نسبتا امکانات خوبی دارد که البته میتوان از آن برای پک کردن اکستنشن نهایی هم استفاده کرد. برای پک کردن روی گزینه pack کلیک کرده و در کادر باز شده گزینه‌ی pack را بزنید. برای شما دو نوع فایل را در مسیر والد دایرکتوری extension نوشته شده درست خواهد کرد که یکی پسوند crx دارد که می‌شود همان فایل نهایی افزونه و دیگری هم پسوند pem دارد که یک کلید اختصاصی است و باید برای آپدیت‌های آینده افزونه آن را نگاه دارید. در صورتی که افزونه را تغییر دادید و خواستید آن را به روز رسانی کنید موقعی که اولین گزینه pack را می‌زنید و صفحه باز می‌شود قبل از اینکه دومین گزینه pack را بزنید، از شما می‌خواهد اگر دارید عملیات به روز رسانی را انجام می‌دهید، کلید اختصاصی آن را وارد نمایید و بعد از آن گزینه pack را بزنید:





آپلود نهایی کار در Google web store

برای آپلود نهایی کار به google web store که در آن تمامی برنامه‌ها و افزونه‌های کروم قرار دارند بروید. سمت راست آیکن تنظیمات را بزنید و گزینه developer dashboard را انتخاب کنید تا صفحه‌ی آپلود کار برای شما باز شود. دایرکتوری محتویات اکستنشن را zip کرده و آپلود نمایید. توجه داشته باشید که محتویات و سورس خود را باید آپلود کنید نه فایل crx را. بعد از آپلود موفقیت آمیز، صفحه‌ای ظاهر می‌شود که از شما آیکن افزونه را در اندازه 128 پیکسل میخواهد بعلاوه توضیحاتی در مورد افزونه، قیمت گذاری که به طور پیش فرض به صورت رایگان تنظیم شده است، لینک وب سایت مرتبط، لینک محل پرسش و پاسخ برای افزونه، اگر لینک یوتیوبی در مورد افزونه دارید، یک شات تصویری از افزونه و همینطور چند تصویر برای اسلایدشو سازی که در همان صفحه استاندارد آن‌ها را توضیح می‌دهد و در نهایت گزینه‌ی جالب‌تر هم اینکه اکستنشن شما برای چه مناطقی تهیه شده است که متاسفانه ایران را ندیدم که می‌توان همه موارد را انتخاب کرد. به خصوص در مورد ایران که آی پی‌ها هم صحیح نیست، انتخاب ایران چنان تاثیری ندارد و در نهایت گزینه‌ی publish را می‌زنید که متاسفانه بعد از این صفحه درخواست می‌کند برای اولین بار باید 5 دلار آمریکا پرداخت شود که برای بسیاری از ما این گزینه ممکن نیست.

سورس پروژه را می‌توانید از اینجا ببینید و خود افزونه را از اینجا دریافت کنید.