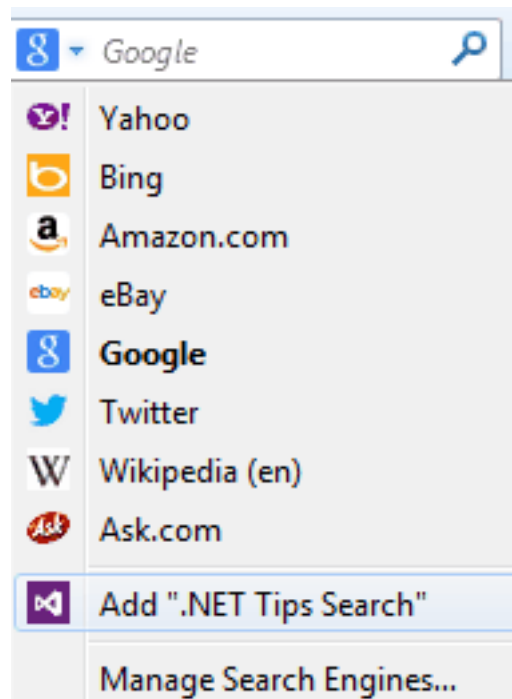
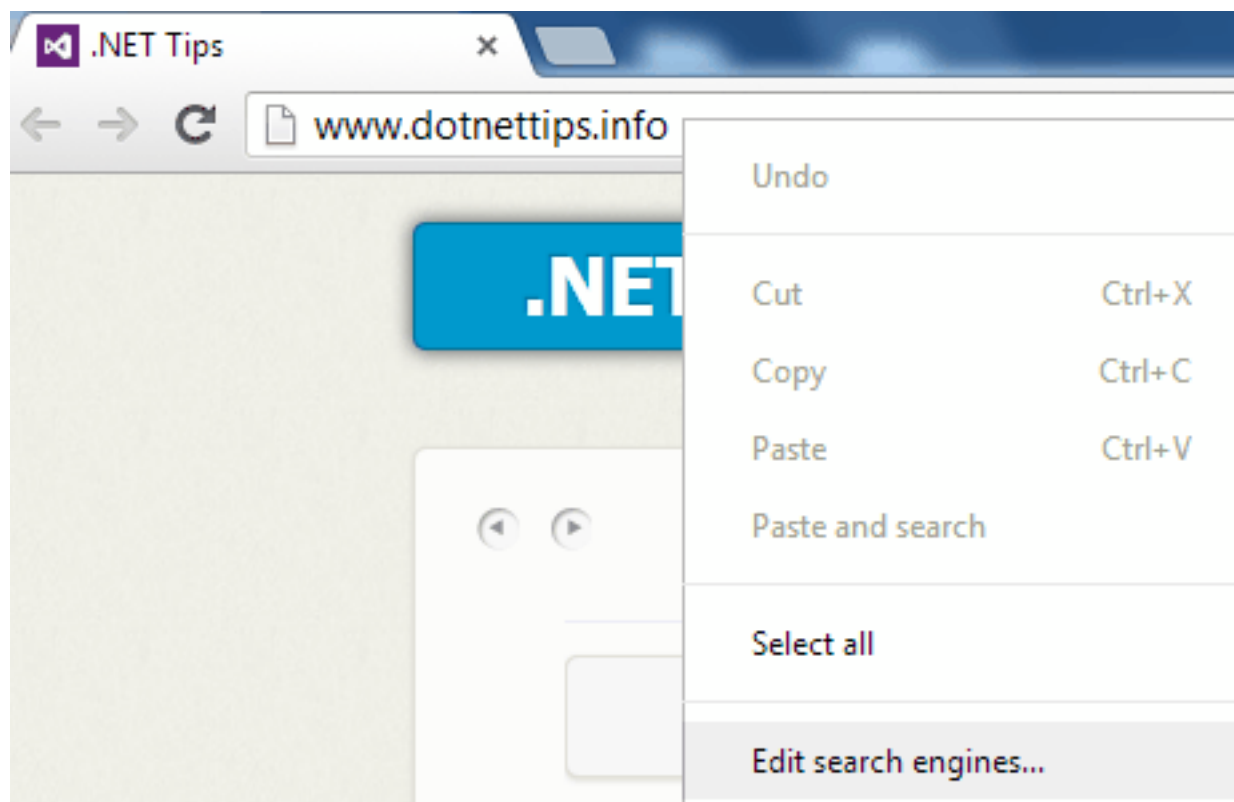




اگر به امکانات مرورگرهای جدید دقت کرده باشید، امکان تعریف منبع جستجوی جدید، نیز برای آن‌ها وجود دارد. برای نمونه تصاویر ذیل مرتبط به مرورگرهای فایرفاکس و کروم هستند:







Search engines

Default search settings

 Google (Default)	google.com	{google:baseURL}search?q=%s&{google:RLZ}{go...
 Yahoo!	yahoo.com	http://search.yahoo.com/search?ei={inputEncodi...

Other search engines

 .NET Tips	dotnettips.info	http://www.dotnettips.info/sea... Make default 
---	-----------------	--

این مرورگرها در صورتیکه پیاده سازی پروتکل [Open Search](#) را در سایت شما پیدا کنند، به صورت خودکار امکان افزودن آنرا به عنوان منبع جستجوی جدیدی جهت جعبه متنی جستجوی خود ارائه می‌دهند. در ادامه قصد داریم با جزئیات پیاده سازی آن آشنا شویم.

تهیه OpenSearchResult سفارشی

برنامه باید بتواند محتوای XML ایی ذیل را مطابق پروتکل Open Search به صورت پویا تهیه و در اختیار مرورگر قرار دهد:

```
<?xml version="1.0" encoding="UTF-8" ? />
<OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/">
  <ShortName>My Site's Asset Finder</ShortName>
  <Description>Find all your assets</Description>
  <Url type="text/html"
    method="get"
    template="http://MySite.com/Home/Search/?q=searchTerms"/>
  <InputEncoding>UTF-8</InputEncoding>
  <SearchForm>http://MySite.com/</SearchForm>
</OpenSearchDescription>
```

به همین جهت کلاس OpenSearchResult ذیل تهیه شده است تا انجام آنرا با روشی سازگار با ASP.NET MVC سهولت بخشد:

```
using System;
using System.Text;
using System.Web;
using System.Web.Mvc;
using System.Xml;

namespace WebToolkit
{
    public class OpenSearchResult : ActionResult
    {
        public string ShortName { set; get; }
        public string Description { set; get; }
        public string SearchForm { set; get; }
        public string FavIconUrl { set; get; }
        public string SearchUrlTemplate { set; get; }

        public override void ExecuteResult(ControllerContext context)
        {
            if (context == null)
                throw new ArgumentNullException("context");

            var response = context.HttpContext.Response;
            writeToResponse(response);
        }

        private void writeToResponse(HttpResponseBase response)
        {
            response.ContentEncoding = Encoding.UTF8;
            response.ContentType = "application/opensearchdescription+xml";
            using (var xmlWriter = XmlWriter.Create(response.Output, new XmlWriterSettings { Indent =
true })))
            {
                xmlWriter.WriteStartElement("OpenSearchDescription", "http://a9.com/-
/spec/opensearch/1.1/");

                xmlWriter.WriteElementString("ShortName", ShortName);
                xmlWriter.WriteElementString("Description", Description);
                xmlWriter.WriteElementString("InputEncoding", "UTF-8");
                xmlWriter.WriteElementString("SearchForm", SearchForm);

                xmlWriter.WriteStartElement("Url");
                xmlWriter.WriteAttributeString("type", "text/html");
                xmlWriter.WriteAttributeString("template", SearchUrlTemplate);
                xmlWriter.WriteEndElement();

                xmlWriter.WriteStartElement("Image");
                xmlWriter.WriteAttributeString("width", "16");
                xmlWriter.WriteAttributeString("height", "16");
                xmlWriter.WriteString(FavIconUrl);
                xmlWriter.WriteEndElement();

                xmlWriter.WriteEndElement();
                xmlWriter.Close();
            }
        }
    }
}
```

کار این Action Result، تهیه محتوایی XML ایی مطابق نمونه‌ای است که در ابتدای توضیحات ملاحظه نمودید. توضیحات خواص آن، در ادامه مطلب ارائه شده‌اند.

تهیه OpenSearchController

در ادامه برای استفاده از Action Result سفارشی تهیه شده، نیاز است یک کنترلر را نیز به برنامه اضافه کنیم:

```
using System.Web.Mvc;

namespace Readers
{
    public partial class OpenSearchController : Controller
    {
        public virtual ActionResult Index()
        {
            var fullBaseUrl = Url.Action(result: MVC.Home.Index(), protocol: "http");
            return new OpenSearchResult
            {
                ShortName = ".NET Tips",
                Description = ".NET Tips Contents Search",
                SearchForm = fullBaseUrl,
                FavIconUrl = fullBaseUrl + "favicon.ico",
                SearchUrlTemplate = Url.Action(result: MVC.Search.Index(), protocol: "http") +
                "?term={searchTerms}"
            };
        }
    }
}
```

برای استفاده از OpenSearchResult به چند نکته باید دقت داشت:

الف) آدرس‌های مطرح شده در آن باید مطلق باشند و نه نسبی. به همین جهت پارامتر protocol در اینجا ذکر شده است تا سبب تولید یک چنین آدرس‌هایی گردد.

ب) Url.Action ایی که در اینجا استفاده شده است مطابق تعاریف [T4MVC](#) است؛ ولی کلیات آن با نمونه پیش فرض ASP.NET MVC تفاوتی نمی‌کند. توسط T4MVC بجای ذکر نام اکشن متد و کنترلر مد نظر به صورت رشته‌ای، می‌توان به صورت Strongly typed به این موارد ارجاع داد.

ج) تنها نکته مهم این کلاس، خاصیت SearchUrlTemplate است. قسمت انتهایی آن یعنی {searchTerms} همیشه ثابت است. اما ابتدای این آدرس باید به کنترلر جستجوی شما که قادر است پارامتری را به شکل کوئری استرینگ دریافت کند، اشاره نماید. د) FavIconUrl به آدرس یک آیکن در سایت شما اشاره می‌کند. برای نمونه ذکر favicon.ico پیش فرض سایت می‌تواند مفید باشد.

معرفی OpenSearchController به Header سایت

```
<link href="@Url.Action(result: MVC.OpenSearch.Index(), protocol: "http")" rel="search"
      title=".NET Tips Search" type="application/opensearchdescription+xml" />
```

مرحله نهایی افزودن پروتکل Open search به سایت، مراجعه به فایل layout پروژه و افزودن link خاص فوق به آن است. در این لینک، href آن باید به مسیر کنترلر OpenSearch ایی که در قسمت قبل تعریف کردیم، اشاره کند. این مسیر نیز باید مطلق باشد. به همین جهت پارامتر protocol آن مقدار دهی شده است.