« Backwards-Compatible Spam and Delete Buttons for...

Valid, SEO-Friendly Post Translation Links »

Custom OpenSearch Functionality for Your Website

I recently added OpenSearch functionality to Perishable Press. Now, OpenSearch-enabled browsers such as Firefox and IE 7 alert users with the option to customize their browser's built-in search feature with an exclusive OpenSearch-powered search option for Perishable Press. The autodiscovery feature of supportive browsers detects the custom search protocol and enables users to easily add it to their collection of readily available site-specific search options. Now, users may search the entire Perishable Press domain with the click of a button.

And you can do it too! Adding customized OpenSearch-powered search functionality to your own site is a great way to foster site awareness and reinforce brand identity, while providing a tool that will benefit your visitors and improve the usability of your site. Even better, implementing OpenSearch functionality is extremely easy, completely free, and requires zero maintenance. In this article, I provide an easy, 3-step tutorial on how to add OpenSearch functionality to your site in less than five minutes. After the tutorial, we will look at the many different ways to customize your OpenSearch implementation, including examples, search options, and much more.

The Tutorial: How to add OpenSearch to your site in less than five minutes

In this tutorial, we will implement basic OpenSearch functionality to your site. This implementation will enable visitors to use the autodiscovery feature of Firefox (or other OpenSearch-enabled browser) to easily add your site's OpenSearch engine to their list of custom search tools. Visitors will then be able to search your site quickly and easily using the incredibly powerful Google search engine. Of course, you may customize this technique to use any search-engine/protocol you wish, including that of your own site. After the tutorial, we examine these alternate search possibilities and explore further configurational options that will enable you to fully customize your OpenSearch engine to meet your specific needs. Ready? Go!

Step 1: Create the XML file

Copy and paste the following code into an empty XML file, name it whatever you would like, and upload the file to a convenient location on your server:

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/">
        <ShortName>Perishable Press</ShortName>
        <Description>Use Google to search Perishable Press!</Description>
        <Url type="application/rss+xml" template="http://www.google.com/search?q=site:perishablepress.com {searchTerms}"/>
</OpenSearchDescription>
```

Replace the two instances of "`Perishable Press`" and the one instance of "`perishablepress.com`" with the name and URL of your own site (note that the site name must be less than 17 characters in length). For more OpenSearch description elements, such as `LongName` and `Tags`, check out the additional configurational options provided after this tutorial.

Step 2: Add the autodiscovery link

Once the required XML file is available on your server, you need to add the following `<link>` element to the `<head>` region of your web page(s):

```
<link rel="search" type="application/opensearchdescription+xml" title="Search Perishable Press" href="https://perishablepress.com/opensearc
```

Replace the `title` and `href` attribute values with your site's name and XML-file location, respectively.

Once in place, this autodiscovery link will alert browsers to the availability of your new OpenSearch-powered search protocol, enabling visitors to quickly and easily add your custom search engine to their browser. Note that this code must be present on all pages for which you would like to enable the autodiscovery feature.

Step 3: Autodiscover & search!

Once the file and link are both in place, load (and refresh) one of your OpenSearch-enabled web pages in Firefox (or your favorite OpenSearch-enabled browser). If

everything is in place, you will see the browser's search field flash and/or glow (depending on browser) to indicate the presence of a new custom search engine. You can then add your search engine by clicking on its name in the list. Now, anytime you need to bust a quick search of your site, just select your OpenSearch engine from the drop-down menu and search! Even better, this same functionality will be available to all of your visitors as well ;)

Of course, this is a very basic implementation that is designed to get OpenSearch up and running on your site quickly and easily. There are many great ways to customize the functionality of your OpenSearch protocol, including the tagging of search results, use of alternate search engines and protocols, display of custom images and icons, and much more. In the next section, we explore the many different descriptive elements available to your OpenSearch implementation and examine different ways to use them to customize your OpenSearch engine.

The Details: Advanced configurational options to customize OpenSearch

There are many ways to customize the functionality of your OpenSearch search engine. In addition to the descriptive elements used in the previous tutorial, the OpenSearch protocol provides a versatile set of descriptive elements that enable successful integration of virtually any OpenSearch configuration. In this section, we will examine these various elements and consider some of the functional possibilities enabled by their individual manipulation. A good way to make use of this information is to copy and paste the desired elements into the description document for your own OpenSearch engine and then edit as needed to suit your specific needs. Then, in the next section, we will dig into the formatting of the autodiscovery link, and then wrap things up with a more complete example of an OpenSearch implementation.

`OpenSearchDescription`

The `OpenSearchDescription` element serves as the root node of the OpenSearch XML description document. This element must include a XML Namespaces URI, which for the currently employed specification (1.1) is this:

`http://a9.com/-/spec/opensearch/1.1/`

Here is the code used for this element (replace the dots with your descriptive elements):

```
<OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/">
        .
        .
        .
</OpenSearchDescription>
```

`ShortName`

The `ShortName` element is a brief description of your search engine. As mentioned in the tutorial, this element must contain fewer than 17 characters of plain text — no markup allowed. This is a required element and is written as follows:

```
<ShortName>Perishable Press</ShortName>
```

`Description`

The `Description` element provides a complete description of your search engine. Use fewer than 1025 plain-text characters with no markup. This is a required element and is written as follows:

```
<Description>Use Google to search Perishable Press!</Description>
```

`Url`

The `Url` element describes the request protocol(s) available to the search client. In our tutorial, we use the following `Url` element to instruct OpenSearch-enabled browsers to search your site via Google:

```
<Url type="application/rss+xml"
     template="http://www.google.com/search?q=site:perishablepress.com {searchTerms}"/>
```

This `Url` element must be included at least once, with multiple elements describing multiple request protocols. For example, here is how to use your own site's search function as the OpenSearch engine:

```
<Url type="text/html"
     template="https://perishablepress.com/press/?s={searchTerms}"/>
```

For more information about the many wonderful things made possible with the `Url` element, check out official documentation.

`Contact`

The `Contact` element is used to share your email address with the whole wide world. This element is certainly *not* required, but if you simply must use it, I recommend using a junk email account that already receives a ton of spam. Otherwise, it's a great way to ruin a spam-free email address.

```
<Contact>spam-me-please@wasted-email-account.com</Contact>
```

`Tags`

The `Tags` element contains a set of keywords that describe and categorize your site. Use single words only and separate each tag with a blank space (`' '`). This element is not required, and must include less than 257 characters of plain text (no markup allowed!). Here is an example:

```
<Tags>web design development tutorials wordpress htaccess php xhtml css</Tags>
```

`LongName`

The `LongName` element provides the complete title or name of your custom search engine. As this element is not required, search clients (e.g., OpenSearch-enabled browsers) should use the value of the `ShortName` element. Use fewer than 49 plain-text characters (no markup!) for this bad boy:

```
<LongName>Custom Google Search for Perishable Press</LongName>
```

`Image`

The `Image` element specifies the URL of an image that may be used in association with the search content. Any specified image sizes are treated as recommendations and may or may not be used by the search client in question when rendering your image. For this reason, if you are planning on providing an image for your search engine, you should provide differently sized versions so that search clients may choose the optimal size. It is recommended that images be formatted with square aspect ratios, and that the following image formats be made available:

- 16×16 (px) image of MIME type `"image/x-icon"` (genral icon format)
- 16×16 (px) image of MIME type `"image/vnd.microsoft.icon"` (Microsoft ICON format)
- 64×64 (px) image of MIME type `"image/jpeg"` or `"image/png"`

```
<Image height="16" width="16" type="image/x-icon">https://perishablepress.com/favicon.ico</Image>
<Image height="16" width="16" type="image/vnd.microsoft.icon">https://perishablepress.com/favicon.ico</Image>
<Image height="64" width="64" type="image/png">https://perishablepress.com/favicon.png</Image>
```

`Query`

The `Query` element specifies a search query that may be performed by search clients. Although not required, it is a good idea to include at least one working example of a search query that will return results, thereby enabling clients to validate the proper functionality of the search engine. For example, if the search query "`chimpanzee`" returns valid results on your site, you may use the following example for your `Query` element:

```
<Query role="example" searchTerms="chimpanzee"/>
```

For more information on this element, check out the OpenSearch Query element specification.

`Developer`

The `Developer` element specifies the identity of the creator of the OpenSearch XML description document. This element is not required, and doesn't necessarily reflect the owner, author, or copyright holder of the search content itself. Use fewer than 65 characters of plain text with no markup:

```
<Developer>Monzilla Media: Obsessive Web Design @ monzillamedia.com</Developer>
```

`Attribution`

The `Attribution` element lists those responsible for the search content. Use fewer than 257 characters of plain text with no markup:

```
<Attribution>All Searched Content Copyright 2050 Perishable Press</Attribution>
```

`SyndicationRight`

The `SyndicationRight` element specifies the degree to which search results may be queried, displayed, and redistributed. Not required, but if included, must employ one of the following attribute values:

- `open` (default) — search clients may request, display, and redistribute search results
- `limited` — search clients may request and display, but not redistribute search results
- `private` — search clients may request, but not display or redistribute search results
- `closed` — search clients may not request, display, or redistribute search results

And here it is, in all its glory:

```
<SyndicationRight>open</SyndicationRight>
```

`AdultContent`

The `AdultContent` element is either "`true`" or "`false`" depending on whether or not search queries may return adult material. Use common sense here. If your site contains stuff that you wouldn't want your kids to see, use a value of "`true`"; otherwise, if your site content is suitable for audiences of all ages, use "`false`"

```
<AdultContent>false</AdultContent>
```

`Language`

The `Language` element indicates the language(s) that are supported by the search engines. Include one of these bad boys for each supported langauge. Use a wildcard value (`*`) for robust search engines such as Google. Note that the `Language` value must conform to the XML 1.0 Language Identification, as specified by RFC 3066. Here are two different examples, one for English-only search engines (like the one used at this site), and another showing use of the wildcard character:

```
<Language>en-us</Language>
<Language>*</Language>
```

`InputEncoding`

The `InputEncoding` element specifies the character encoding(s) that your search engine supports for search requests. Although not required, multiple instances of this element may appear according to the number of different character encodings supported for search requests. Note that the attribute value for this element must conform to the XML 1.0 Character Encodings, as specified by the IANA Character Set Assignments. The default value is `UTF-8`, as used in the following example:

```
<InputEncoding>UTF-8</InputEncoding>
```

`OutputEncoding`

The `OutputEncoding` element specifies the character encoding(s) that your search engine supports for search responses. Although not required, multiple instances of this element may appear according to the number of different character encodings supported for search responses. Note that the attribute value for this element must conform to the XML 1.0 Character Encodings, as specified by the IANA Character Set Assignments. The default value is `UTF-8`, as used in the following example:

```
<OutputEncoding>UTF-8</OutputEncoding>
```

Clearly, this wide assortment of descriptive elements provide your search engine with all the configurational options needed to meet the specific functional requirements of your site. Now that we have examined the available components of the OpenSearch XML description document, let us walk through the specifics of the second part of any OpenSearch implementation: the autodiscovery link.

The Link: A closer look at the OpenSearch autodiscovery link

As seen in the tutorial, an autodiscovery link is required to alert search clients such as web browsers to the presence and availability of your custom OpenSearch engine. This autodiscovery link is included via the `<link>` element and may be included in both web pages (HTML/XHTML) and web feeds (RSS/Atom). Ideally, OpenSearch-enabled search clients will recognize the autodiscovery link and alert its user to the availability of the custom search engine. Further, search engines that support OpenSearch should include a reference to the associated OpenSearch description document on each page of the search results. Here are a few examples, the first showing a complete (X)HTML implementation of two autodiscovery links (one for content and one for comments); the second showing the same implementation within the Atom format; and the third showing the implementation within the RSS format.

Autodiscovery links for (X)HTML-formatted documents

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en" dir="ltr">
        <head profile="http://a9.com/-/spec/opensearch/1.1/">

                <link rel="search"
                      type="application/opensearchdescription+xml"
                      href="http://domain.tld/content-search.xml"
                      title="Content search" />
                <link rel="search"
                      type="application/opensearchdescription+xml"
                      href="http://domain.tld/comment-search.xml"
                      title="Comments search" />

        </head>
        <body>
                <!-- ... -->
        </body>
</html>
```

Autodiscovery links for Atom-formatted documents

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom"
      xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/">

        <link rel="search"
              type="application/opensearchdescription+xml"
              href="http://domain.tld/content-search.xml"
              title="Content Search" />
        <link rel="search"
              type="application/opensearchdescription+xml"
              href="http://domain.tld/comment-search.xml"
              title="Comments Search" />
```

```
</feed>
```

Autodiscovery links for RSS-formatted documents

```
<?xml version="1.0" encoding="UTF-8"?>
<rss version="2.0"
     xmlns:atom="http://www.w3.org/2005/Atom">
        <channel>

                <atom:link rel="search"
                           type="application/opensearchdescription+xml"
                           href="http://domain.tld/content-search.xml"
                           title="Content Search" />
                <atom:link rel="search"
                           type="application/opensearchdescription+xml"
                           href="http://domain.tld/comment-search.xml"
                           title="Comments Search" />

        </channel>
</rss>
```

Regardless of document type, the following restrictions apply to the autodiscovery link:

- The "`type`" attribute must contain the value ldquo;`application/opensearchdescription+xml`".
- The "`rel`" attribute must contain the value "`search`".
- The "`href`" attribute must contain a URI that resolves to an OpenSearch description document.
- The "`title`" attribute may contain a human-readable plain text string describing the search engine.

Had enough? I didn't think so! Let's wrap things up with a couple of examples..

Examples: Minimal and complete renditions of the OpenSearch description document

For the sake of completeness, here are two examples of the OpenSearch XML description document. The first is similar to the description document given in the prvious tutorial, whereas the second example is a complete template featuring the entire collection of available description elements. Each of these examples uses my site, Perishable Press, as the associated website, so please edit each of the elements according to your specific needs.

Minimal OpenSearch Description Document

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/">
        <ShortName>Perishable Press</ShortName>
        <LongName>Google Search of Perishable Press</LongName>
        <Description>Use Google to search Perishable Press!</Description>
        <Tags>digital design development wordpress php htaccess xhtml css javascript</Tags>
        <Url type="application/rss+xml" template="http://www.google.com/search?q=site:perishablepress.com {searchTerms}"/>
</OpenSearchDescription>
```

Complete OpenSearch Description Document

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/">
        <OutputEncoding>UTF-8</OutputEncoding>
        <InputEncoding>UTF-8</InputEncoding>
        <AdultContent>false</AdultContent>
        <Language>en-us</Language>
        <ShortName>Perishable Press</ShortName>
        <LongName>Google Search of Perishable Press</LongName>
        <Description>Use Google to search Perishable Press!</Description>
        <Tags>digital design development wordpress php htaccess xhtml css javascript</Tags>
        <Url type="application/rss+xml" template="http://www.google.com/search?q=site:perishablepress.com {searchTerms}"/>
        <Url type="text/html" template="https://perishablepress.com/press/?s={searchTerms}"/>
        <Image height="16" width="16" type="image/x-icon">https://perishablepress.com/favicon.ico</Image>
        <Image height="16" width="16" type="image/vnd.microsoft.icon">https://perishablepress.com/favicon.ico</Image>
        <Image height="64" width="64" type="image/png">https://perishablepress.com/favicon.png</Image>
        <Query role="example" searchTerms="chimpanzee" />
        <Developer>Monzilla Media: Obsessive Web Design @ monzillamedia.com</Developer>
        <Attribution>Search data Copyright 2050, Perishable Press; All Rights Reserved</Attribution>
        <SyndicationRight>open</SyndicationRight>
        <Contact>spam@gmail.com</Contact>
</OpenSearchDescription>
```

That's all for this fun-filled article! Stay tuned for more high-quality articles from Perishable Press. Thanks for your generous attention! :)

Related posts:

1. Feed your Image via Atom or RSS
2. How to Add Meta Noindex to Your Feeds
3. Pimp Your 404: Presentation and Functionality
4. Important Note for Your Custom Error Pages
5. Choosing the Best Title Separators

About this post

Author: Jeff Starr
Archived: December 07, 2008
Updated: Oct 3rd, 2016
Category: HTML, SEO
Tags: google, reference, rss, search, tips, tricks, websites, xml

Check it out

Comments

Cheryl Beckham

December 7, 2008 at 2:45 pm

This is just a quick comment of the brilliantly designed background on your site, it caught my attention and I just had to comment on it. Good work.
Happiest of Holidays to You
Cheryl Beckham

Jeff Starr Post author

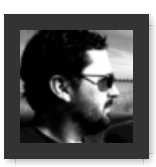
December 8, 2008 at 9:39 pm

Thank you for the kind remarks, Cheryl — they are greatly appreciated :)

jarrett jordaan

May 22, 2009 at 4:15 am

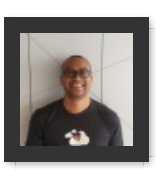
Is it possible to just feed results without OpenSearch autodiscovery?
What if you would like to just supply search results in OpenSearch format?
Can the repository description xml file be left out?



Jeff Starr Post author

May 27, 2009 at 7:21 am

@jarrett jordaan: Haven't tried it — have you seen anything about this in the documentation? http://www.opensearch.org/



jarrett jordaan

May 28, 2009 at 12:13 am

With respect to the format of search results.
What I understand is that OpenSearch specification is merely an extension to the existing Atom and RSS specifications.

So currently, I has a web services interface, to login, and return a session key.
And then every search request needs the returned search key to access search API.

Correct me if I am wrong.

Nonetheless a great article that got me started.

Jeff Starr Post author

May 30, 2009 at 2:14 pm

Glad to hear that you are digging into OpenSearch — I know you will reap the rewards once you get the hang of it. In the meantime, thanks for the feedback and positive remarks — much appreciated.

Andre Luis de Andrade

February 13, 2013 at 12:27 pm

Hello!

Now, the A9.com spec does not exists anymore. What we'll do?

Look: http://a9.com/-/spec/opensearch/1.1/

Thanks for help!

Comments are closed. Contact the author with questions or further information.

« Backwards-Compatible Spam and Delete Buttons for...

Valid, SEO-Friendly Post Translation Links »