How do I run a Web search in the user's default Web browser using their default search provider?

devblogs.microsoft.com/oldnewthing/20150914-00

September 14, 2015

Raymond Chen

More than one customer has asked, "How do I run a Web search in the user's default Web browser using their default search provider?"

Nobody knows for sure.

Windows does define a mechanism for determining the user's default Web browser, or more specifically, default handler for the http protocol. But that's as far as it goes. There is no standard mechanism for invoking the user's default search provider. The concept of the default search provider is internal to the Web browser; Windows itself has no insight into that, any more than it knows which of your bank accounts is in danger of being overdrawn, which is a concept internal to your personal finance program.

But that doesn't stop people from trying. In particular, Process Explorer uses the following technique: It acts as if it's launching a URL, but instead of passing something like `http://www.microsoft.com`, it passes `"? search string"`. That is, a quotation mark, a question mark, a space, and then the search parameters, and then a closing quotation mark.

Experimentation revealed that most Web browsers interpret this as a request to perform a search in the user's default search provider. (Not all Web browsers do this, however. For example, it doesn't work in Internet Explorer 6 because Internet Explorer 6 didn't even have the concept of a *default search provider*.)

Here's some code that tries to follow the above algorithm:

```
using System;
using System.Text;
using System.Runtime.InteropServices;
using System.Diagnostics;

class Program
{
 [DllImport("shlwapi.dll", CharSet=CharSet.Unicode, PreserveSig=false)]
 static extern void AssocQueryString(
    int flags, int str, string assoc, string extra,
    [Out] StringBuilder buffer, ref int bufferSize);

 const int ASSOCF_ISPROTOCOL = 0x00001000;
 const int ASSOCSTR_COMMAND = 1;

 [DllImport("shell32.dll", CharSet=CharSet.Unicode, PreserveSig=false)]
 static extern void SHEvaluateSystemCommandTemplate(
    string template,
    [Out] out string application,
    [Out] out string commandLine,
    [Out] out string parameters);

 public static void Main(string[] args)
 {
  int bufferSize = 260;
  var buffer = new StringBuilder(bufferSize);
  AssocQueryString(ASSOCF_ISPROTOCOL, ASSOCSTR_COMMAND,
                   "http", "open", buffer, ref bufferSize);
  var template = buffer.ToString();

  Console.WriteLine("Template = {0}", template);

  string application, commandLine, parameters;
  SHEvaluateSystemCommandTemplate(template, out application,
                                  out commandLine, out parameters);

  Console.WriteLine("Application = {0}", application);
  Console.WriteLine("Parameters = {0}", parameters);

  parameters = parameters.Replace("%1", "\"? " + args[0] + "\"");

  Console.WriteLine("Parameters after replacement = {0}", parameters);

  Process.Start(application, parameters);
 }
}
```

Most of this program is style points.

First, we call `AssocQueryString` to get the answer to the question "If I wanted to open a URL that begins with http:, what command template should I use?" This will return something like

```
"C:\Program Files\Internet Explorer\iexplore.exe" %1
```

All we have to do is replace the `%1` with `"? search phrase"`, then run the resulting command line.

Now come the style points.

The `SHEvaluateSystemCommandTemplate` function conveniently splits split the command template into the pieces we need in order to pass them to `Process.Start`. The path to the excutable comes out as the `application`, the command line arguments come out as the `parameters`, and we don't use the `commandLine` part.

After splitting the command line into the executable and the parameters, we perform the `%1` substitution on the parameters, and then we're ready to pass it all to `Process.Start`.

And there you have it. Note that there is no guarantee that this code will actually work, but in practice it works with most of the major Web browsers out there.

Raymond Chen

Follow