Textcavator

Name:	Textcavator
Author:	Mike Pasini
Version:	1.0b1a
Last Update:	31 March 2018
Requires:	Keyboard Maestro v8, CocoaDialog 3

Table of Contents

INTRO

There are many ways to scan file content and the most powerful are free beginning with **`grep`** itself.

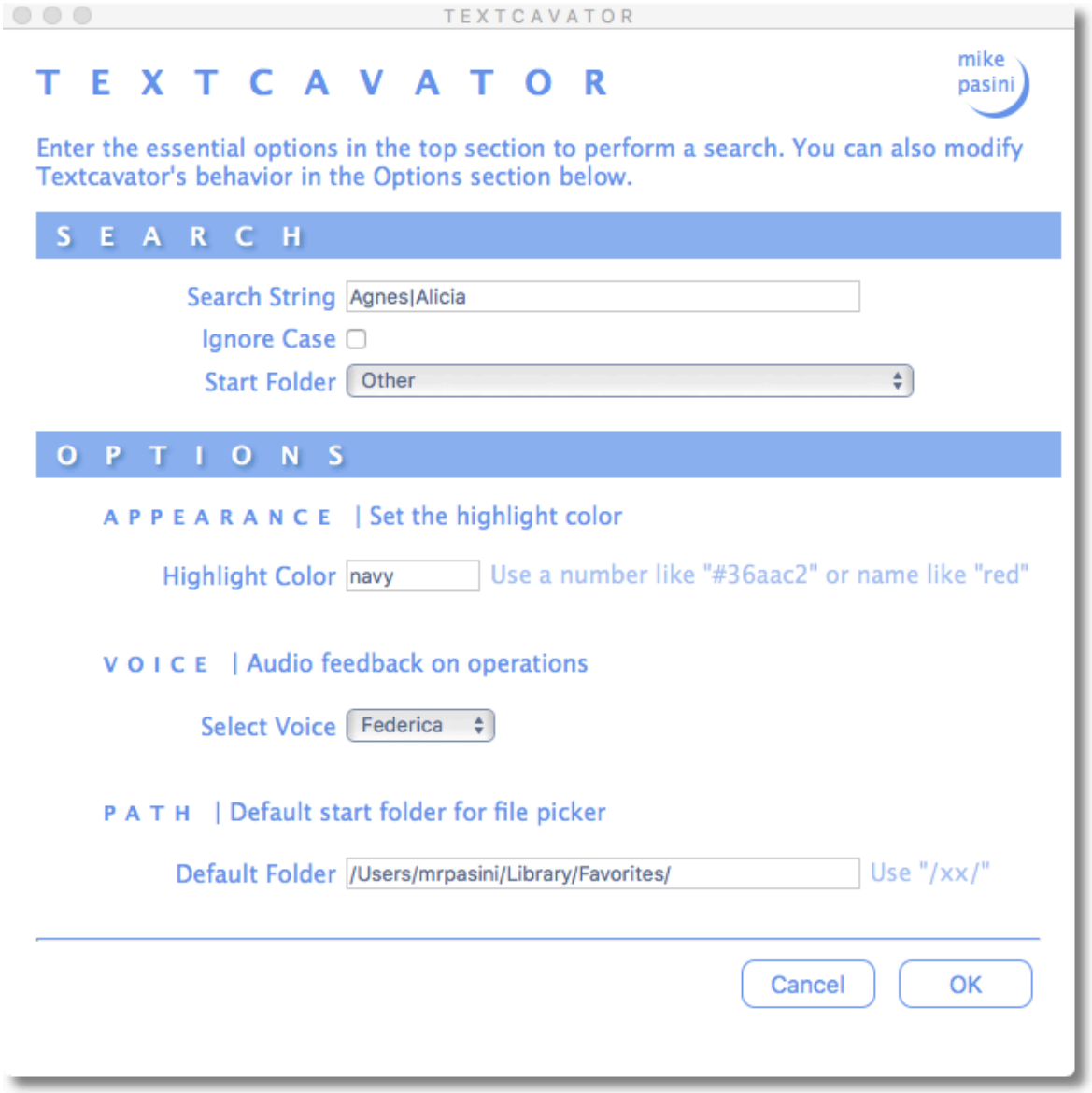
And, if you prefer, you can even find powerful, free tools with a graphical user interface. In fact, for many years, I've relied on DEVONtechnologies' free EasyFind to scan file contents that included the several thousand files in a local copy of my Web site, decades of 20-plus page monthly journals in page layout programs, PDF documentation, code repositories and more.

So why build a new tool?

EasyFind, like many other text search applications, suffers an annoying limitation. They report filenames where the content has been found but don't show the content in context. Which obliges you to open a dozen or so files to find the reference you want.

To display results of a search in context, I wrote a Perl CGI for the Web that I adapted for my site (http://mikepasini.com/corners/), which I called Mike's Method, based on Perl code by Kevin Reid from 2000. That has run very nicely for many years.

Why can't I have it search my hard drive, too? With Keyboard Maestro I can.



DESIGN ISSUES

The HTML version on my site simply asks for a regexp (which can also be literal text) before going to work. It already knows which directories to look through. And it ignores case because who, searching for text, bothers with the Shift key?

But a hard drive has a lot more files on it than a Web site. And different kinds of files are in different places on the drive. Some of these targets will be searched frequently. But you still need to be able to wing it. And it would be nice to remember what you winged.

So I wanted a list of locations I could easily modify (well, add to). And I wanted an option to select a starting folder, too, if none of the favorites was the target.

Searches would be recursive, digging through any subfolders for files. I had hoped to be able to exclude certain folders as well by using multiple selection in my folder picker but that wasn't supported by either CocoaDialog's or Keyboard Maestro's picker.

As for the regexp, I first thought that if I entered a word in caps the search should be case sensitive. "Agnes" for example should not return a hit on "magnesium." And if the search string was lowercase, the search could ignore case.

But then I thought I would prefer to make that explicit so searching for "new" would not return "Newman." I needed a checkbox. So Keyboard Maestro's Custom HTML Prompt won the day. It would also allow me to create a popup of favorite locations and, unlike the user prompt, allow me to organize the prompts into the essentials and some options.

The code itself was, for the most part, taken from the Perl CGI I use on my site, although it was seeded with a lot of variables where there had been constants. The highlight color is just one example.

But how was I going to display the results? The CGI wrote HTML. And I liked the tabular format and typography of the solution I had devised for the site.

Fortunately, Keyboard Maestro has the option of writing HTML to a window. It's the Custom HTML Prompt but it works just as well to display HTML.

Because some searches traversing multiple directories might take a long time, I wanted to display a progress bar as the Perl code executes. Keyboard Maestro provides a number of ways to display a progress bar but CocoaDialog works within Perl. And its file picker lets you set a default location, which Keyboard Maestro's does not.

Solving those issues was enough to get me started on Textcavator, where I focused on a clean user interface, efficient code and simplifying the installation.

INSTALLATION

The .zip file contains the following files:

- CocoaDialog 3
- Textcavator macro with embedded Perl code
- Textcavator.pdf

To install Textcavator:

1. Copy **CocoaDialog** to your Applications folder
2. Import the **Textcavator macro**

You will then have to set up the macro to reflect your environment.

To configure the macro:

1. Set the **hot key trigger** you prefer

2. Change the Default Values for the Variable "**TC Start Folder**" in the first Custom HTML Prompt to a list of your favorite locations, each ending in a forward slash ("/"). The first line provides a simple example of the HTML syntax:

   ```
   <option value="Other">Other</option>
   ```

   Here's how to point to your Desktop:

   ```
   <option value="/Users/[username]/Desktop/">/Users/
   [username]/Desktop/</option>
   ```

   Substitute "[username]" with your own user name.

   There's no obligation to use the whole path in the descriptive part of the HTML. If you prefer, you can simply write:

   ```
   <option value="/Users/[username]/Desktop/">Desktop</
   option>
   ```

   That goes for any option you want to list. The `description` between the > and `<` delimiters can be anything. The form passes the actual path listed in the `value` to the Perl script for processing.

   If you want a rule (or a space if you omit the line) between groups of options use this code:

   ```
   <option disabled>──────────</option>
   ```

   Alternately, you can use the `<optgroup label="[your label]"></optgroup>` tags around your options, adding a descriptive word for the label.

   Make sure `Other` remains in the list grouped with the long item just below it with the `value` "Last." That's where your last `Other` is recorded. The order isn't important to the code so make it convenient for yourself. `Other` with `Last` may be last if you don't tend to search random places, for example.

3. Confirm that `$CD` is set to the location you copied **CocoaDialog** in the **Execute Shell Script** action to the location you copied Textcavator.pl if you did not embed it.

That's it. The macro should now be set to respond to your trigger with a list of your preferred targets.

USAGE

When you first trigger the macro, some of its fields may be unpopulated. Subsequent runs will remember your entries from last time.

The initial form has two parts: the essential Search parameters and a separate section for Options.

The top Search section of the form presents a prompt for three things:

1. A Search String
2. The option to Ignore Case
3. A Start Folder

Let's look at each of them to see what you Textcavator expects:

The **Search String** can be a literal like "Sam's vitamins" or a regexp like "(Sam|Ethyl)'s vitamins" to look for vitamins for either Sam or Ethyl. If you want to use a metacharacter like "?" literally, escape it with the backslash ("\"). Other metacharacters that require a backslash in a literal search include **`[ ] { } ( ) . | ^ $ * \`**.

If enabled, **Ignore Case** will match regardless of case. So "sam" will match both "Samantha" and "samovar." If you uncheck the option to disable it, only "samovar" would match "sam."

Start Folder sets the root folder to begin the recursive search down through all the folders it contains. Every file will be opened and read by Textcavator as long as it has one of these extensions: **`txt html? indd? qxp php pdf pl pdf rtf`**. Here "**`html?`**" is a regexp that includes files ending in both "**`htm`**" and "**`html`**."

If you select **`Other`**, Textcavator will present the CocoaDialog file picker pointed at the default directory you specified in the **`Options`** section of the prompt. Textcavator will remember which directory you picked and show it under **`Other`** on the popup list. If it was the last directory used, it (rather than **`Other`**) will be the default shown next time you run Textcavator.

Textcavator can't be pointed at an individual file (just open the file and use the application's **`Find`** tool if that's what you want). It does, however, look at all files of the legal types in a directory and its subdirectories.

OPTIONS

The Custom HTML Prompt also allows you to set some options in addition to the basic search parameters. Those include:

HIGHLIGHT COLOR

You can set the highlight color for each or every run of Textcavator by entering either its RGB code (like "#36aac2" or HTML name (like "navy") in the prompt provided.

Use any color you like that will both contrast with the white background of the report and the light blue color of the text.

If you use **cornflowerblue**, the effect will be as if no highlighting has been done (because that's the color of the text itself). Try “**navy**” or “**red**” but avoid light colors like “**yellow**.”

VOICE

Textcavator can provide audio feedback. I find it useful for longer running searches, allowing me to flip to another screen and work on something else. I particularly like hearing Federica talk to me in Italian.

But Federica is a large, high-quality, optional voice that must be downloaded and installed. And Italian is an acquired taste.

A selection of standard male and female voices is available in the popup. And they all speak English. Think of them as a staff of assistants.

You can disable the audio by simply selecting "**None**" from the list.

PATH

If you select **Other** as your Start Folder, the file picker uses the location you enter in **Path** as its default. I find it handy to default a Custom search to my Favorites folder, for example, so I don't have to add all of the paths in it to the popup menu.

CUSTOMIZATION

You can customize Textcavator beyond setting options by editing the macro or KM-Textcavator.pl.

FILE TYPES

As shipped, Textcavator will only open files whose extensions include **txt**, **doc**, **docx**, **htm**, **html**, **ind**, **indd**, **qxp**, **php**, **pl**, **pages**, **pdf** and **rtf**.

You can include other extensions by adding them to the list in the subroutine "**wanted**" at the very end of KM-Textcavator.pl.

Word files are preprocessed using Apple’s textutil software (included with OS X since version 10.4) into text files. HTML files are stripped of code, RTF files are stripped of their own codes and any file not listed explicitly as a text file is stripped of unprintable characters. To add a text file format to the exclusions list, search for "**# preprocessing**" in KM-Textcavator.pl and add it to list.

It's safe to add the extension for any text file but proprietary files are all different and may cause issues even with just printable characters. If you don’t add the extension to the preprocessing section, it will be treated like a proprietary file.

Saving the output file should help diagnose any problem. In fact any runtime error encountered will show up in the HTML report.

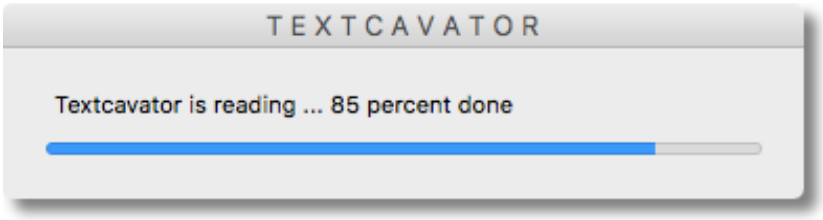
COCOADIALOG LOCATION

If you copy CocoaDialog to your main Applications folder, where it will be accessible to any user on your system, you don't have to edit the **$CD** variable in KM-Textcavator.pl. If you prefer to put it somewhere else, change the **$CD** variable location in the Perl code.

RUNNING TEXTCAVATOR

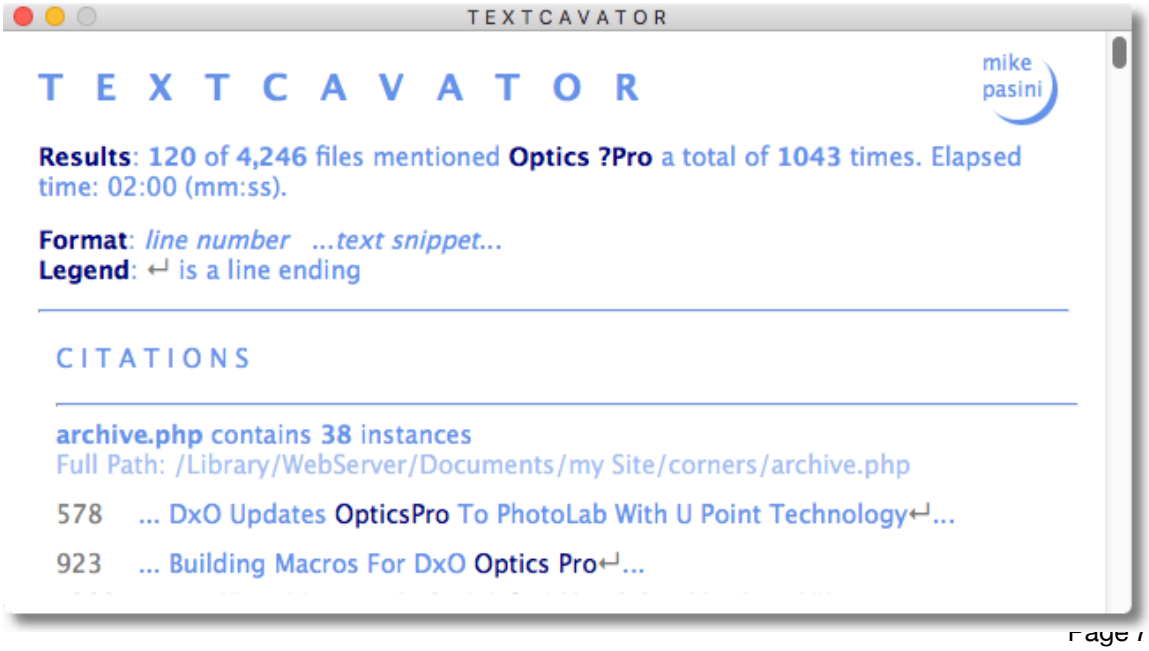
After entering your options, click OK to start Textcavator.

An indeterminate file progress bar (it throbs) is displayed as Textcavator builds a file list. Sometimes this takes a few seconds and you'll see the progress bar and other times you'll just see the window flash (it happens so fast).



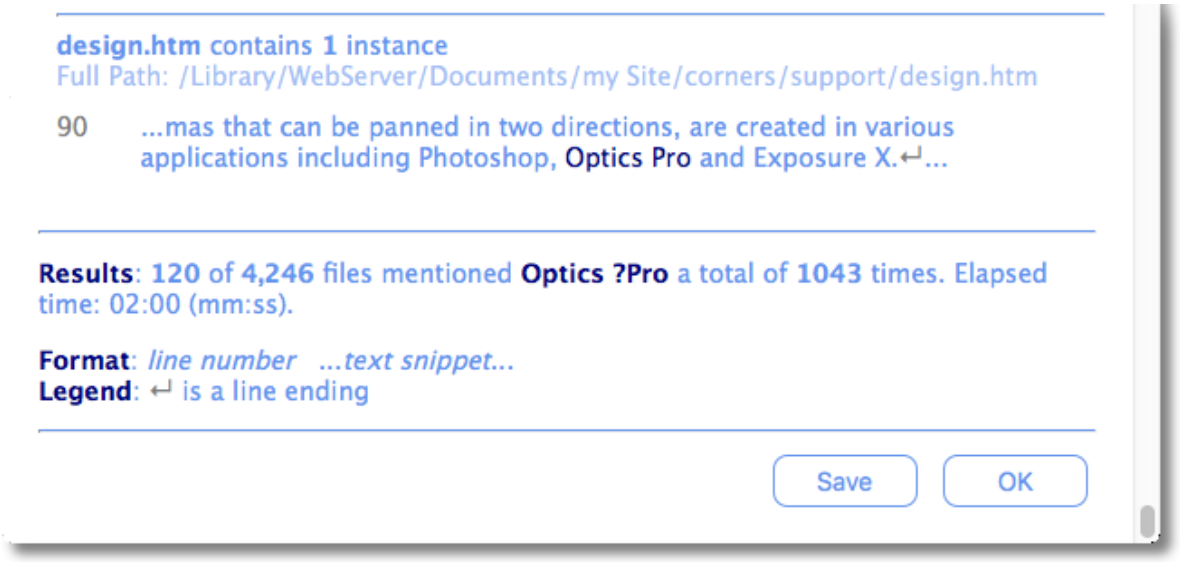
A progress bar indicates what percentage of the files TextCavator has processed.

When it has finished processing the files, it will present a window with its report. The bottom of the report includes the number of files with the search phrase, the total files, the number of total occurrences and the elapsed time.



In testing on a spinning hard disk, Textcavator was able to go through over 4,200 HTML files in just over a minute (1:04) on the first run and 37 seconds on the second. Textcavator took 3:55 minutes to read through 756 long (20+ pages each) QuarkXpress and InDesign documents in multiple directories.

The window also includes a **Save** option to write the report to disk. You'll be prompted for a filename and location.



NOTES

Scanning proprietary formats like QuarkXPress and InDesign documents requires stripping non-printing characters (except for a few) from the text stream (but not the file itself, which is left untouched). Sometimes this doesn't quite work as expected and the HTML window will be truncated. In that case, Saving the file will allow you inspect the data for the offending characters. And usually you can display the saved file in any browser.

The code to strip non-printing characters from propriety files written by QuarkXPress and InDesign is not comprehensive. Mainly what we're trying to accomplish here is to prevent anything from aborting the display. But smart quotes and other niceties are not converted so it wouldn't be wise to quote the in-context display for those finds. But they should be sufficient to show whether or not you want to open the file.

Line numbers for page layout programs are meaninglessly high. They may look more like character counts. Ignore.

Loading CocoaDialog (for the file picker or the progress bars) takes a few seconds, slowing things down. So does audio feedback. Once cached the delay disappears, so second runs are usually much quicker.

Although this is version 1.1a, it's still a work in progress. The macro is solid but the filtering routines will no doubt enjoy some refinement. Meanwhile, though, it's been very helpful to me so I'm passing it around.

One of those refinements has been incorporating Apple's textutil to handle Word files. When a Word file is passed to Textcavator, it calls textutil to convert the Word file into a temporary text file that Textcavator processes normally. Textcavator then deletes the temporary file.

Happy to discuss further customizations with anyone or feedback on the interface. And pleased to see any code revisions you make, particularly for other file formats

RELEASE NOTES

v1.1a on 31 March 2018

- Added support for Microsoft Word doc and docx formats via Apple's textutil.

v1.0b on 1 March 2018

- Corrected the Keyboard Maestro variable name for the default folder that did not have the TC precede.

- Added a tip in this documentation about describing folder targets.

v1.0a on 22 February 2018

- Initial release.

CONTACT

You can reach me at http://mikepasini.com. Or by clicking the Moon icon on any of Textcavator's HTML pages.