

Looking to “Spring Clean” Your Business Data? Use a Search Engine Instead

Article contributed
by [dtSearch®](#)

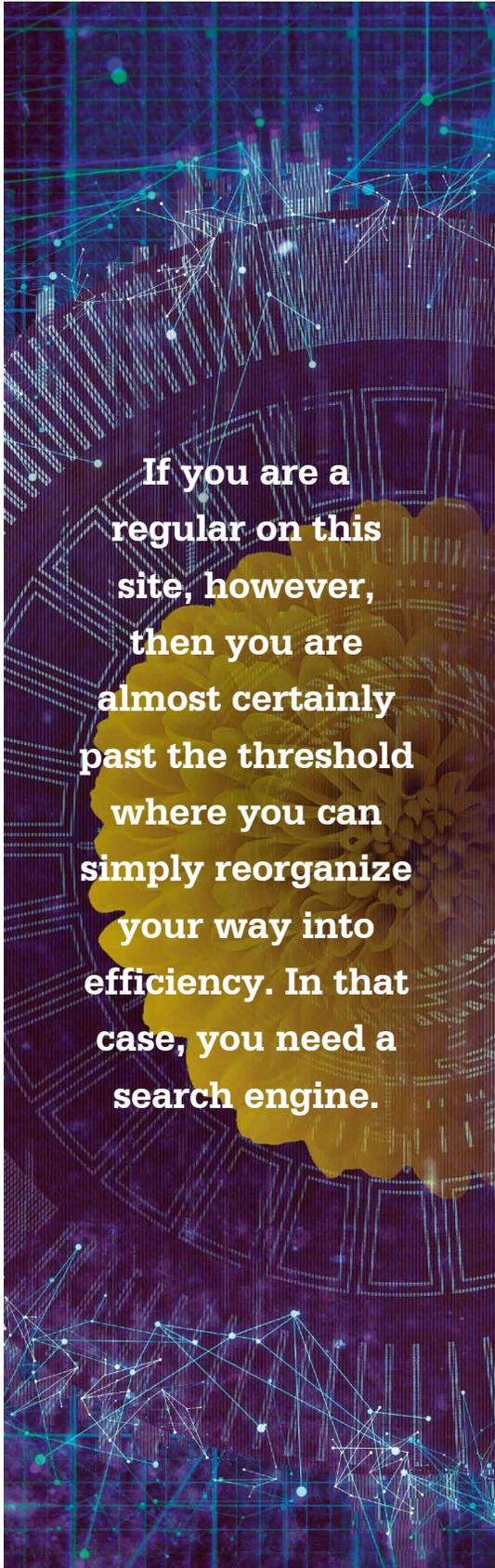
If you are working with a small number of files and emails, “cleaning out” your business data may let you locate what you are looking for reasonably quickly. If you are a regular on this site, however, then you are almost certainly past the threshold where you can simply reorganize your way into efficiency. In that case, you need a search engine.

A search engine (like [dtSearch®](#), [dtSearch.com](#)) can instantly search terabytes of files and emails on your own computer. It can also instantly search terabytes concurrently for multiple users across a network or a web server hosted locally or remotely on Azure or AWS, for example. In fact, with federated searching, there are no limits on the number of data sources end-users can concurrently search.

A search engine operates by first building an index covering all the data. Building an index is effortless for you the end-user. All you have to do is point to the file directories, emails, online repositories, etc. to index and the search engine will on its own do the rest.

A key point in the indexing process is the way in which a search engine approaches content. When you edit a Word document in Microsoft Word or browse a PDF file in Adobe Reader or view an email in Outlook, you are working with these files in their associated applications. To efficiently index terabytes, however, a search engine cannot individually open each file in its associated application. Such a process would take way too long. Rather, a search engine needs to approach files and other data as they sit on a PC, network or web repository in their resting binary format.

If you look at a file in its binary format, you will typically see a mishmash of binary codes. You may be hard-pressed to read any of the general text that readily pops up when you view the file in its associated application. To sift through such binary formats, the document filters component of a search engine needs, as an initial step, to figure out what type of file it is. The specifications for parsing a OneNote file are very different from the specifications for parsing a PDF which are in turn very different for the specifications for parsing an email.



If you are a regular on this site, however, then you are almost certainly past the threshold where you can simply reorganize your way into efficiency. In that case, you need a search engine.

After recognizing the relevant binary format, the search engine's document filters must locate all text and accompanying metadata inside the binary file. The text of most files today are in Unicode. In addition to international alphabets, the Unicode standard covers numbers as well as numeric and other symbols. More in the hearts and minds of most of the world, the Unicode Consortium defines the world's emojis, with a new crop of emojis typically out each year.

Once a search engine finishes indexing, it can instantly search terabytes, displaying search results for one end-user or multiple concurrent end-users with highlighted hits. For the search itself, over 25 different search features help you find exactly what you need. The search engine can even sift through typographical or OCR errors that may appear in text.

Beyond searching for text, a search engine can also offer other search features like the ability to identify any credit card numbers in data. A search engine can also make available forensics-oriented search techniques like generating and locating hash values. A search engine's developer package can leverage metadata from structured databases and other sources to enable easy user interface "drill down" faceted search as well as backend data classification for security and other purposes.

We can leave it at that illustrating the benefits of a search engine as an alternative to spring cleaning. Or you can read on for a "deeper dive" into some points relevant to how a search engine sees your data.

(1) Files saved with non-conforming file extensions. As mentioned, for a search engine, parsing a binary file is a multistep process: determining the correct file format; applying the correct parsing specification; and "following the Unicode" throughout the file. One way, it might seem, to undermine this process is to give a file a non-conforming file extension, such as saving a Microsoft Word document with a .PDF ending.

However, current file formats include information inside the binary format indicating the document type. That way, a Microsoft Access file can have a OneNote extension and a PowerPoint can have an Excel spreadsheet extension and the search engine can parse the files regardless.

Article contributed
by [dtSearch®](#)

Once a search engine finishes indexing, it can instantly search terabytes, displaying search results for one end-user or multiple concurrent end-users with highlighted hits.



Article contributed
by [dtSearch®](#)

(2) **“Invisible” text.** Sometimes people use black text against a black background, white text against a white background, red text against a red background, etc. to avoid prying eyes. Even if someone is not personally trying to obscure text, an editing or redaction program can sometimes mask text as part of the editing or redaction process.

The caution here is that while masked text may look invisible, the text can sometimes reappear with copy and paste. Further, while prying eyes may miss black on black or white on white text inside of a file's associated application, such text is fully available in a file's binary format and hence readily accessible to a search engine.

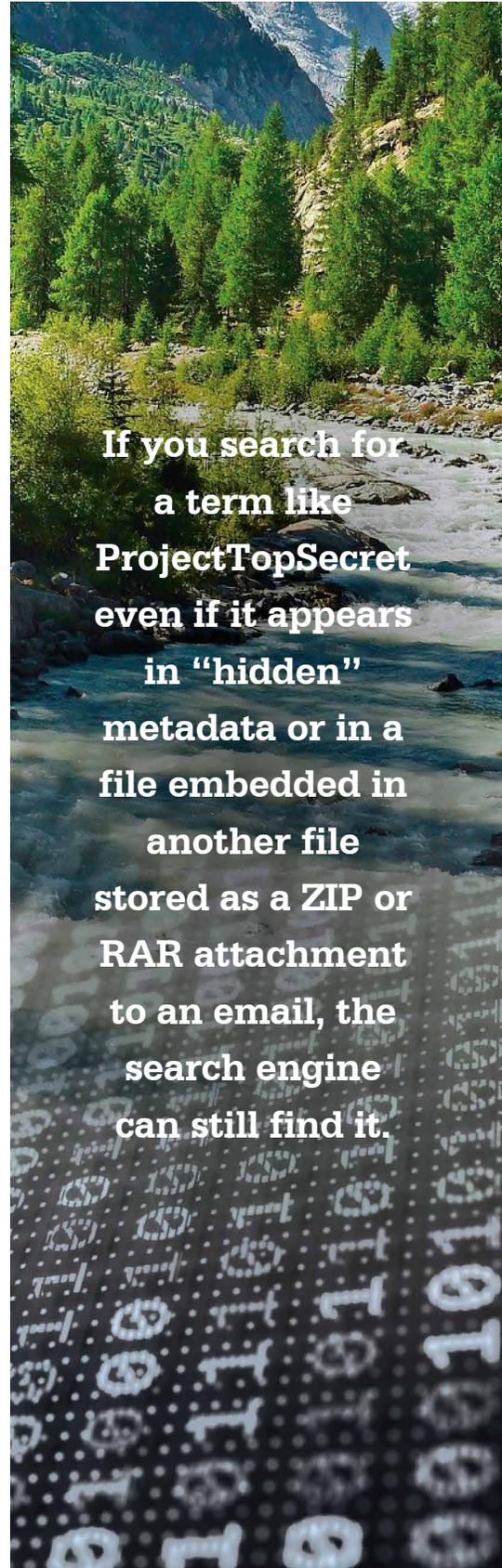
(3) **“Faux” text.** With the previous point, you may think that text is gone even if it really isn't. Here, you may see what looks like standard text but is not text at all. The most common example of this is in PDFs. Have you ever looked at a PDF, tried to copy and paste some text, and been unable to do so? What you were looking at was probably an “image only” PDF.

The resolution here is to OCR the document using an OCR program like Adobe Acrobat. That OCR process can turn the image into standard Unicode text. (As an aside, if you are running dtSearch, there is a feature which can flag “image only” PDFs which may be accidentally or potentially even intentionally mixed in with regular PDFs.)

(4) **“Buried” text.** Present-day file formats can be really complex. They can store metadata that only appears in very specific file views. Attached to an email, you can have containers like RAR or ZIP embedding multiple files. You can even have files fully embedded inside other files. For example, you could have a Microsoft Word document with an Excel spreadsheet embedded, where by default you may only see a portion of the spreadsheet inside your Word display.

While the application view can obscure the full extent of such embedded text, in binary format, for a search engine set up to look for embedded content, all text remains fully indexable. If you search for a term like ProjectTopSecret even if it appears in “hidden” metadata or in a file embedded in another file stored as a ZIP or RAR attachment to an email, the search engine can still find it.

If you want to try a search engine to instantly search terabytes of your own data, a fully-functional 30-day evaluation version awaits you at [dtSearch.com](#)



**If you search for
a term like
ProjectTopSecret
even if it appears
in “hidden”
metadata or in a
file embedded in
another file
stored as a ZIP or
RAR attachment
to an email, the
search engine
can still find it.**