MIRRORING

Searchable Web Content on Portable Media — *dtSearch®*

WEB SEARCH OVERVIEW

Web data typically falls into three categories: static browser-ready content, like HTML, PDF, and XML/XSL; dynamic browser-ready content, like SharePoint and ASP.NET; and browser- incompatible content, like MS Office or OpenOffice files.

A single search request should be able to span all of these content categories. A single search request should also be able to span multiple Internet and Intranet sites. The Internet part of a search might query a competitor's website for a specific combination of keywords. The Intranet part of a search might look for a different combination of keywords across an enterprise's own local and remote sites.

> The dtSearch Spider works by building a search index that stores the location of words in the spidered data.

Operating either as a user-interfacedriven software component or through a .NET API, the dtSearch Spider can meet all of these criteria. In both cases, the dtSearch Spider can expand the scope of a search beyond a site's own data to content on a remote site, whether public or private.

The dtSearch Spider displays static and dynamic browser-ready content WYSIWYG, including display of images, formatting and links, with the sole addition of highlighted hits. The Spider can convert browser-incompatible content such as MS Office or OpenOffice "on the fly" to HTML for browser display with highlighted hits.

The dtSearch Spider works by building a search index that stores the location of words in the spidered data. Indexing with the Spider involves simply selecting a URL or URLs and indicating how many vertical or horizontal links to follow. The Spider automatically figures out the format of the content, so there is no need to tell the Spider whether a retrieved web page contains, for example, an MS Office document or a PDF file.

Indexed search time is typically less than a second, even over a terabyte or more of data. For convenient offline access, the dtSearch Spider also includes a caching option, to store the full spidered content along with the index. (Without caching, the Spider has to return to the relevant URL to display the full content with highlighted hits.)

MIRRORING WEB DATA ON PORTABLE MEDIA

Suppose you want to provide offline access to web searchable content. Including a mini-HTTP or HTTPS server on portable media offers similar data access without a web connection. The user need only insert a disk, and the disk itself can run with zero footprint.

This approach, however, has a major drawback. The local HTTP server does not actually access anything outside of the local machine. But because it communicates with the web browser using HTTP, some

firewall software treats it as if it were accessing the Internet. So, instead of seamless execution, the user sees security alerts and error messages.

Simulating an HTTP server avoids the firewall issue, while still allowing zero-footprint execution from removable media. The HTTP simulation approach relies on a viewer program embedding a web browser control. The control displays content just as it would online with a web browser.

Simulating an HTTP server is the approach dtSearch Publish takes. With dtSearch Publish, all web pages and search forms continue to appear exactly as they would on the web. Features like JavaScript and CSS also work identically to online.

The viewer program displays static browser-ready content (HTML, PDF and XSL/XML) the same as online, i.e. WYSIWYG with highlighted hits. The viewer program will also act like the online application in

converting browser-incompatible content (MS Office, OpenOffice, etc.) to HTML for display with highlighted hits.

STARTING WITH PAPER

Suppose that a project requires going one step farther, starting with actual paper instead of web content. In that case, dtSearch recommends and supports the PDF "image with hidden text" format.

The "image with hidden text" format stores the complete original image of a scanned document, along with the OCR'ed text. The format hides the text in the sense that opening the PDF file displays only the scanned image, not the underlying OCR'ed text. But following a search, the search can highlight a word in the OCR'ed text itself, over the actual image of the document.

Publishing OCR'ed documents online or to portable media works the same as with any other

documents. One search feature, however, is essential with OCR'ed text — or for that matter, any text that might contain typographical errors, such as email correspondence. To sift through OCR'ed or other text containing typographical errors requires a degree of fuzzy searching.

With a fuzziness level of 1, dtSearch would look for an exact word spelling, as well as a slight deviation in letters: traveler as well as traveter. With a fuzziness of 2, the algorithm also looks for additional deviations in letters, finding not only traveler and traveter, but also trameller. The fuzzy search that dtSearch uses is not hardwired into the index, so it is fully adjustable at the time of search.

For maximum precision searching, fuzzy searching also works in connection with all other full-text and fielded data search options. And fuzzy searching does not substantially slow down a search — indexed search time is still typically less than a second even over a terabyte or more of data. ■