NAME

index − SWISH++ indexer

SYNOPSIS

index [*options*] *directory... file...*

DESCRIPTION

index is the SWISH++ file indexer. It indexes the specified files and files in the specified directories; files in subdiretories of specified directories are also indexed by default (unless either the **−r** option or the **RecurseSubdirs** variable is given). Files are indexed either only if their filename extension is among the set specified with either the **−e** option or the **IncludeExtension** variable (unless standard input is used; see next paragraph) or is not among the set specified with either the **−E** option or the **ExcludeExtension** variable.

If there is a single filename of '`-`', the list of directories and files to index is instead taken from standard input (one per line). In this case, filename extensions of files to index need not be specified explicitly: all files, regardless of extension (unless it is among the set not to index specified with either the **−E** option or the **ExcludeExtension** variable), are indexed, i.e., **index** assumes you know what you're doing when specifying filenames in this manner.

In any case, care must be taken not to specify files or subdirectories in directories that are also specified: since directories are recursively indexed by default (unless either the **−r** option or the **RecurseSubdirs** variable is given), explicitly specifying a subdirectory or file in a directory that is also specified will result in those files being indexed more than once.

Character Entities

Both HTML character and numeric (decimal and hexadecimal) entity references are converted to their ASCII character equivalents according to the ISO 8859-1 (Latin 1) character set before further examination and indexing. For example, ''r&eacute;sum&#233;'' becomes ''resume'' before indexing.

Word Determination

Stop words, words that occur too frequently or have no information content, are not indexed. (There is a default compiled-in set of a few hundred such words.) Additionally, several heuristics are used to determine which words should not be indexed.

First, a word is checked to see if it looks like an acronym. A word is considered an acronym only if it starts with a capital letter and is composed exclusively of capital letters, digits, and punctuation symbols, e.g., ''AT&T.'' If a word looks like an acronym, it is indexed and no further checks are done.

Second, there are several other checks that are applied. A word is not indexed if it:

1. Starts with a capital letter, is of mixed case, and contains more than a third capital letters, e.g., ''BizZARE.''
2. Contains a capital letter other than the first, e.g, ''weIrd.''
3. Is less than `Word_Min_Size` letters. (Default is 4.)
4. Contains less than `Word_Min_Vowels` vowels. (Default is 1.)
5. Contains more than `Word_Max_Consec_Same` of the same character consecutevely (not including digits). (Default is 2.)
6. Contains more than `Word_Max_Consec_Consonants` consecutive consonants. (Default is 5.)
7. Contains more than `Word_Max_Consec_Vowels` consecutive vowels. (Default is 4.)
8. Contains more than `Word_Max_Consec_Puncts` consecutive punctuation characters. (Default is 1.)

HTML Files

Additional processing is done for HTML files. (HTML files are recognized as such only if their filename extension is one of: `htm`, `html`, or `shtml`.) The additional processing is:

1. If a matched set of `<TITLE>` ... `</TITLE>` tags is found within the first `Title_Lines` lines of the file (default is 12), then the text between the tags is stored in the generated index file as the file's title rather than the file's name. (Every non-space whitespace character in the title is converted to a space; leading and trailing spaces are removed.)
2. If an HTML element contains a `CLASS` attribute whose value is among the set of class names specified as those not to index (via one or more of either the **−C** option or the **ExcludeClass** variable), then all the text up to the tag that ends the element will not be indexed.

   For an element that has an optional end tag, ''the tag that ends the element'' is either the element's end tag or a tag of another element that implicitly ends it; for an element that does not have an end tag, ''the tag that ends the element'' is the element's start tag. (See the EXAMPLES.) All elements from the HTML 4.0 specification (including deprecated elements), plus common, browser-specific elements are recognized; unrecognized elements are ignored. (See the **−H** option.)
3. If an HTML element contains a `TITLE` attribute, then the words specified as the value of the `TITLE` attribute are indexed.
4. If an `AREA`, `IMG`, or `INPUT` element contains an `ALT` attribute, then the words specified as the value of the `ALT` attribute are indexed.
5. If a `META` element contains both a `NAME` and `CONTENT` attribute, then the words specified as the value of the `CONTENT` attribute are indexed associated with the meta name specified as the value of the `NAME` attribute. (See also the **−m** and **−M** options or the **IncludeClass** or **ExcludeClass** variables.) Meta names can later be queried against specifically using **search**(1).
6. If a `TABLE` element contains a `SUMMARY` attribute, then the words specified as the value of the `SUMMARY` attribute are indexed.
7. If an `OBJECT` element contains a `STANDBY` attribute, then the words specified as the value of the `STANDBY` attribute are indexed.
8. All other HTML tags and comments (anything between `<` and `>` characters) are discarded.

In compliance with the HTML specification, any one of no quotes, single quotes, or double quotes may be used to contain attribute values and attributes can appear in any order. Values containing whitespace, however, must be quoted. The specification is vague as to whether whitespace surrounding the `=` is legal, but **index** allows it.

Filters

Via the **FilterExtension** configuration file variable, files having particular extensions can be filtered prior to indexing. (See FILTERS in **swish++.conf**(4).)

OPTIONS

−**c***config_file*	The name of the configuration file to use. The default is `swish++.conf` in the current directory. A configuration file is not required: if none is specified and the default does not exist, none is used; however, if one is specified and it does not exist, then this is an error.
−**C***class_name*	For HTML files only, a class name of an HTML element whose text is not to be indexed. Multiple **−C** options may be specified.
−**e***extension*	A filename extension of files to index *without* the ''dot.'' Case is significant. Multiple **−e** options may be specified.
−**E***extension*	A filename extension of files *not* to index *without* the ''dot.'' Case is significant. Multiple **−E** options may be specified.
−**f***file_max*	The maximum number of files a word may occur in before it is discarded as being too frequent. The default is infinity.
−**F***file_reserve*	Reserve space for this number of files to start. More space will be allocated as necessary, but with a slight performace penalty. The default is 1000.

−H Dump the built-in set of recognized HTML elements to standard output and exit.

−i*index_file* The name of the generated index file. The default is `swish++.index` in the present working directory.

−l Follow symbolic links during indexing. The default is not to follow them. (This option is not available under Microsoft Windows since it doesn't support symbolic links.)

−m*meta_name* For HTML files only, the value of a meta `NAME` attribute for which the words in the value of the associated `CONTENT` attribute should be indexed. Case is irrelevant. Multiple **−m** options may be specified.

By default, words in the value of the `CONTENT` attribute for all meta names are indexed. Specifying at least one meta name via this option changes that so that only the words in the value of the `CONTENT` attribute associated with a member of the set of meta names explicitly specified via one or more **−m** options are indexed.

−M*meta_name* For HTML files only, the value of a meta `NAME` attribute for which the words in the value of the associated `CONTENT` attribute should not be indexed. Case is irrelevant. Multiple **−M** options may be specified.

−p*percent_max* The maximum percentage of files a word may occur in before it is discarded as being too frequent. The default is 100. If you want to keep all words regardless, specify 101.

−r Do not recursively index the files in subdirectories, that is: when a directory is encountered, all the files in that directory are indexed (modulo the filename extensions specified via either the **−e** or **−E** options or the **IncludeExtension** or **ExcludeExtension** variables) but subdirectories encountered are ignored and therefore the files contained in them are not indexed. (This option is most useful when specifying the directories and files to index via standard input.) The default is to index the files in subdirectories recursively.

−s*stop_word_file* The name of a file containing the set of stop-words to use instead of the built-in set. Whitespace, including blank lines, and characters starting with `#` and continuing to the end of the line (comments) are ignored.

−S Dump the built-in set of stop-words to standard output and exit.

−t*title_lines* For HTML files only, the maximum number of lines into a file to look at for HTML `<TITLE>` tags. The default is 12. Larger numbers slow indexing.

−T*temp_dir* The path of the directory to use for temporary files. The directory must exist. The default is `/tmp`.

If your OS mounts swap space on `/tmp`, as indexing progresses and more files get created in `/tmp`, you will have less swap space, indexing will get slower, and you may run out of memory. If this is the case, you can specify a directory on a real filesystem, i.e., one on a physical disk.

−v*verbosity* Print additional information to standard output during indexing. The verbosity levels, 0-4, are:

0	No output is generated except for errors. This is the default.
1	Only run statistics (elapsed time, number of files, word count) are printed.
2	Directories are printed as indexing progresses.
3	Directories and files are printed with a word-count for each file.
4	Same as 3 but also prints all files that are not indexed and why.

−V Print the version number of **SWISH++** to standard output and exit.

CONFIGURATION FILE

The following variables can be set in a configuration file. Variables and command-line options can be mixed, the latter taking priority.

ExcludeClass	Same as the −**C** option.
ExcludeExtension	Same as the −**E** option.
ExcludeMeta	Same as the −**M** option.
FilesReserve	Same as the −**F** option.
FilterExtension	(See FILTERS in **swish++.conf**(4).)
FollowLinks	Same as the −**l** option.
IncludeExtension	Same as the −**e** option.
IncludeMeta	Same as the −**m** option.
IndexFile	Same as the −**i** option.
RecurseSubdirs	Same as the −**r** option.
StopWordFile	Same as the −**s** option.
TempDirectory	Same as the −**T** option.
TitleLines	Same as the −**t** option.
Verbosity	Same as the −**v** option.
WordFilesMax	Same as the −**f** option.
WordPercentMax	Same as the −**p** option.

EXAMPLES

Command-lines

To index all HTML and text files on a web server:

```
cd /home/www/htdocs
index -v3 -e html -e shtml -e txt .
```

To index all files not under directories named `SCCS`:

```
cd /home/www/htdocs
find . -name SCCS -prune -o -type f -a -print | index -
```

Using `CLASS` attributes to index selectively

In an HTML document, there may be sections that should not be indexed. For example, if every page of a web site contains a navigation menu such as:

```
<SELECT NAME="menu">
<OPTION>Home
<OPTION>Automotive
<OPTION>Clothing
<OPTION>Hardware
</SELECT>
```

or a common header and footer, then, ordinarily, those words would be indexed for every page and therefore be discarded because they would be too frequent. However, via either the −**C** option or the **ExcludeClass** variable, one or more class names can be specified and then HTML elements belonging to one of those classes will not have the text up to the tag that ends them indexed. Given a class name of, say, `no_index`, the above menu can be changed to:

```
<SELECT NAME="menu" CLASS="no_index">
```

and then everything up to the `</SELECT>` tag will not be indexed.

For an HTML element that has an optional end tag (such as the `<P>` element), the text up to the tag that ends it will not be indexed, which is either the element's own end tag or a tag of some other element that implicitly ends it. For example, in:

```
<P CLASS="no_index">
This was the poem that Alice read:
<BLOCKQUOTE>
<B>Jabberwocky</B><BR>
'Twas brillig, and the slithy toves<BR>
Did gyre and gimble in the wabe;<BR>
All mimsy were the borogoves,<BR>
And the mome raths outgrabe.
</BLOCKQUOTE>
```

the `<BLOCKQUOTE>` tag implicitly ends the `<P>` element (as do all block-level elements) so the only text that is not indexed above is: ‘‘This was the poem that Alice read.’’

For an HTML element that does not have an end tag, only the text within the start tag will not be indexed. For example, in:

```
<IMG SRC="home.gif" ALT="Home" CLASS="no_index">
```

the word ‘‘Home’’ will not be indexed even though it ordinarily would have been if the `CLASS` attribute were not there.

Filters

(See Filters under EXAMPLES in **swish++.conf**(4).)

EXIT STATUS

Exits with one of the values given below:

0	Success.
1	Error in configuration file.
2	Error in command-line options.
10	Unable to open temporary file.
11	Unable to write index file.
12	Unable to write temporary file.
30	Unable to read stop-word file.

CAVEATS

1. Files without extensions can not be indexed.
2. Generated index files are machine-dependent (size of data types and byte order).
3. The character encoding always used is ISO 8859-1 (Latin 1). Character encodings specified either in `META` elements or via the `charset` attribute in other elements are ignored.

FILES

`swish++.conf`	default configuration file name
`swish++.index`	default index file name

SEE ALSO

extract(1), **find**(1), **search**(1), **swish++.conf**(4)

International Standards Organization. ‘‘ISO 8859-1: Information Processing -- 8-bit single-byte coded graphic character sets -- Part 1: Latin alphabet No. 1.’’ 1987.

−−. ‘‘ISO 8879: Information Processing -- Text and Office Systems -- Standard Generalized Markup Language (SGML)’’ 1986.

Dave Raggett, Arnaud Le Hors, and Ian Jacobs. ‘‘On SGML and HTML: SGML constructs used in HTML: Entities,’’ *HTML 4.0 Specification, section 3.2.3,* World Wide Web Consortium, April 1998.
http://www.w3.org/TR/PR-html40/intro/sgmltut.html#h-3.2.3

`——`. ‘‘The global structure of an HTML document: The document head: The `title` attribute,’’ *HTML 4.0 Specification, section 7.4.3,* World Wide Web Consortium, April 1998.
http://www.w3.org/TR/REC-html40/struct/global.html#adef-title

`——`. ‘‘The global structure of an HTML document: The document head: Meta data,’’ *HTML 4.0 Specification, section 7.4.4,* World Wide Web Consortium, April 1998.
http://www.w3.org/TR/PR-html40/struct/global.html#h-7.4.4

`——`. ‘‘The global structure of an HTML document: The document body: Element identifiers: the `id` and `class` attributes,’’ *HTML 4.0 Specification, section 7.5.2,* World Wide Web Consortium, April 1998.
http://www.w3.org/TR/PR-html40/struct/global.html#h-7.5.2

`——`. ‘‘Tables: Elements for constructing tables: The `TABLE` element,’’ *HTML 4.0 Specification, section 11.2.1,* World Wide Web Consortium, April 1998.
http://www.w3.org/TR/REC-html40/struct/tables.html#adef-summary

`——`. ‘‘Objects, Images, and Applets: Generic inclusion: the `OBJECT` element,’’ *HTML 4.0 Specification, section 13.3,* World Wide Web Consortium, April 1998.
http://www.w3.org/TR/REC-html40/struct/objects.html#adef-standby

`——`. ‘‘Objects, Images, and Applets: How to specify alternate text,’’ *HTML 4.0 Specification, section 13.8,* World Wide Web Consortium, April 1998.
http://www.w3.org/TR/REC-html40/struct/objects.html#h-13.8

`——`. ‘‘Index of Elements,’’ *HTML 4.0 Specification,* World Wide Web Consortium, April 1998.
http://www.w3.org/TR/REC-html40/index/elements.html

AUTHOR

Paul J. Lucas <*pjl@best.com*>