NAME

swish++.index − SWISH++ index file format

SYNOPSIS

```
long            num_words;
off_t           word_offset[ num_words ];
long            num_stop_words;
off_t           stop_word_offset[ num_stop_words ];
long            num_directories;
off_t           directory_offset[ num_directories ];
long            num_files;
off_t           file_offset[ num_files ];
long            num_meta_names;
off_t           meta_name_offset[ num_meta_names ];
```

word index
stop-word index
directory index
file index
meta-name index

DESCRIPTION

The index file format used by SWISH++ is as shown above. Every `word_offset` is an offset into the *word index* pointing at the first character of a word entry; similarly, every `stop_word_offset` is an offset into the *stop-word index* pointing at the first character of a stop-word entry; similarly, every `directory_offset` is an offset into the *directory index* pointing at the first character of a directory entry; similarly, every `file_offset` is an offset into the *file index* pointing at the first byte of a file entry; finally, every `meta_name_offset` is an offset into the *mete-name index* pointing at the first character of a meta-name entry.

The index file is written as it is so that it can be mapped into memory via the **mmap**(2) Unix system call enabling ''instantaneous'' access.

Encoded Integers

All integers in an index file are stored in an encoded format for compactness. An integer is encoded to use a varying number of bytes. For a given byte, only the lower 7 bits are used for data; the high bit, if set, is used to indicate whether the integer continues into the next byte. The encoded integer is in big-endian byte order. For example, the integers 0−127 are encoded as single bytes of `\x00`−`\x7F`, respectively; the integer 128 is encoded as the two bytes of `\x8100`.

Note that the byte `\x80` will never be the first byte of an encoded integer (although it can be any other byte); therefore, it can be used as a *marker* to embed extra information into an encoded integer byte sequence.

Encoded Integer Lists

Because the high bit of the last byte of an encoded integer is always clear, lists of encoded integers can be stored one right after the other with no separators. For example, the byte sequence `\x010203` encodes the three separate integers 1, 2, and 3; the byte sequence `\x81002A8101` encodes the three integers 128, 42, and 129.

Word Entries

Every word entry in the *word index* is of the form:

word`0`**{***data*}**...**

that is: a null-terminated word followed by one or more *data* entries where a *data* entry is:

{*F*}**{***O*}**{***R*}**[{***list*}**...]{***marker*}

that is: a file-index (*F*) followed by the number of occurrences in the file (*O*) followed by a rank (*R*)

followed by zero or more *lists* of integers followed by a *marker* byte. The *file-index* is an index into the `file_offset` table; the *marker* byte is one of:

`00` Another *data* entry follows.
`80` No more *data* entries follow.

A *list* is:

{type}{I}...`x80`

that is: a *type* followed by one or more integers (*I*) followed by an `\x80` byte where *type* defines the type of list, i.e., what the integers mean. Currently, there is only one type of list:

`01` Meta-ID list. Meta-IDs are unique integers identifying which meta name(s) a word is associated with in the meta-name index.

`02` Word-position list. Every word in a file has a position: the first word is 1, the second word is 2, etc. Each word position is stored as a delta from the previous position for compactness.

Stop-Word Entries

Every stop-word entry in the *stop-word index* is of the form:

stop-word`0`

that is: every word is null-terminated.

Directory Entries

Every directory entry in the *directory index* is of the form:

directory-path`0`

that is: a null-terminated full pathname of a directory (not including the trailing slash). The pathnames are relative to where the indexing was performed (unless absolute paths were used).

File Entries

Every file entry in the *file index* is of the form:

{D}*file-name*`0`***{S}{W}****file-title*`0`

that is: the file's directory index (*D*) followed by a null-terminated file name followed by the file's size in bytes (*S*) followed by the number of words in the file (*W*) followed by the file's null-terminated title.

For an HTML or XHTML file, the title is what is between `<TITLE>` ... `</TITLE>` pairs; for an MP3 file, the title is the value of title field; for a mail or news file, the title is the value of the `Subject` header; for a LaTeX file, the title is the argument of the `\title` command; for a Unix manual page file, the title is the contents of the first line within the `NAME` section. If a file is not one of those types of files, or is but does not have a title, the title is simply the file (not path) name.

Meta-Name Entries

Every meta-name entry in the *meta-name index* is of the form:

meta-name`0`***{I}***

that is: a null-terminated meta-name followed by the ID (*I*).

CAVEATS

Generated index files are machine-dependent (size of data types and byte-order).

SEE ALSO

index(1), **search**(1)

AUTHOR

Paul J. Lucas <*pauljlucas@mac.com*>