Stored on: {Phylum}<Lisp>Library>SINGLEFILEINDEX.Tedit

Program: {Phylum}<Lisp>Library>SINGLEFILEINDEX.DCOM

Documentation revised: February 27, 1984; and May 16, 1984

SINGLEFILEINDEX

Christopher Tong and Jon L. White

SINGLEFILEINDEX is a Lisp Library package for generating indexed listings for files. Its features include:

1. Items that are indexed by type (e.g. FNS, VARS, etc.) and the page number on which their definitions appear in the listing;
2. The ability to define the types to be indexed, as well as the means for locating definitions of a particular type;
3. In INTERLISP-D, the listing is created in a background process;
4. Ihe creation of an multi-type index for multiple files.

1. Creating indexed file listings.

SINGLEFILEINDEX lists a file, producing indices for different types of items that appear on that file. These types are usually file package types, but they needn't be.[1] A default set of type specifications is provided by the value of the variable, **DEFAULTINDEXEDTYPESLST**:

```
((VAR (RPAQ ADDTOVAR) TestForVar)
 (MACRO PUTPROPS TestForMacro)
 (CONSTANTS CONSTANTS TestForCOnstants)
 (RECORD  {indirects through variable CLISPRECORDTYPES})
 (RESOURCE PUTDEF TestForGenericDefinition)
 (BITMAP "RPAQ " TestForBitmap)
 (CLASS "DEFCLASS ")
 (INSTANCE DEFINST TestForInstance)
 (METHOD METH TestForMethod)
)
```

The user can specify the types he wants to index with the variable **INDEXEDTYPESLST**, which is initially set to the value of **DEFAULTINDEXEDTYPESLST**.

Functions are *always* indexed, and should not be included in **INDEXEDTYPESLST**. See section 2 for more on specifying types.

The two functions responsible for creating indexed listings are SINGLEFILEINDEX and MERGEDFILEINDEX.

(SINGLEFILEINDEX *FILE OUTPUTFILE MERGEDINDEXFLG*) [Function]

> *FILE* is the lisp source file. *OUTPUTFILE* is the destination file. If *OUTPUTFILE*= **NIL**, then the value of **PRINTER** (initially **LPT:**) is used. *MERGEDINDEXFLG*= **T** means a single, alpabetically sorted index will be created for all the types (instead of a several type-specific indices). The value of **FILELINELENGTH** determines the position of the index numbers, as well as the placement of the columns. The value of **LINESPERPAGE** (initially 65 on D machines, 58 otherwise) determines the number of lines per page.

SINGLEFILEINDEX first prints the file, with the filename and page number at the top of every page, as well as the name of any definition whose listing has been interrupted by the page break. After the file has been printed, the subsequent pages list the index or indices (depending on the value of *MERGEDINDEXFLG*). The first page provides the filename, time of creation, and time of listing. If *MERGEDINDEXFLG*= T, the order in which the indices appear is determined by the order of the types in INDEXEDTYPESLIST. If there are no items of a particular type on the file, no index is generated. [1] The index for items of a particular type provides the names of those items, as well as the page number on which their definition begins. The index is sorted by item name.

When the SINGLEFILEINDEX package is first loaded, it redefines LISTFILES1 (see the INTERLISP manual) so that all files listed by LISTFILES will be listed using the function SINGLEFILEINDEX. [2] The file being indexed needn't be loaded, or even noticed (in the file package sense).

(MERGEDFILEINDEX *FILES*) [Function]

> MERGEDFILEINDEX creates a single index for a set of files, *FILES*. The index contains the name of a file item, its type, and the file on which it appears. The index is alphabetically sorted by item name. If the same item of a particular type

appears on several files, a separate index entry will be generated for each occurence.

2. Specifying types

For the purpose of this package, a *type* is defined as a list of four items:

1. The name of the type (e.g. **BITMAP**);
2. A candidate-generating string or list of strings (e.g. **RPAQ** and **"RPAQ "**);
3. A function for ascertaining whether a line defines an item of this type;
4. A non-null flag if the type definition is ambiguous with another type;

The type name should be recognizable by the file package; if it isn't recognized, then the assumption is made that items of this type are declared in the FILECOMS of any file to be processed by a word that is the plural of the typename (e.g. the filepkgcoms command to output a **BITMAP** is just **BITMAPS**, and that to output a LOOPS **CLASS** is just **CLASSES**).

The candidate-generating string must follow a "(" or "[" at the beginning of a line on the file (litatoms are acceptable as strings). If a list of strings is provided (e.g. for **RECORD**s), a match against any one of them will produce a candidate line. (Note: the candidate-generating string will be used for partial matching; thus **RPAQ** will match **RPAQ**, **RPAQQ**, and **RPAQ?**, but **"RPAQ "** will match only **RPAQ**.) SINGLEFILEINDEX reads the file, one line at a time, and compares each line against the candidate-generating string or strings for each type; if a candidate line is found, the test function is applied with arguments.

The test function has arguments: (***string typeName***), and returns either **NIL** (i.e. the candidate line is not a legitimate instance of this kind of definition) or the name of the item being defined. The argument ***string*** consists of the characters of the current line, beginning where the candidate-generating string was found, up to and including the end-of-line character(s).

The ambiguity flag means that a given line could test positive for two or more types; ordinarily, at most one type will test positive. The tests for **CONSTANTS** and **VARS** are specifically trying to exclude one another, so that one won't get a constant indexed both as a **VAR** and as a **CONSTANT**. Although there are no instances of this flag in the

DEFAULTINDEXEDTYPESLST, it may arise in user-defined types.

As an example of test functions etc., suppose there is a line in the file:

```
(RPAQ Foo NIL)
```

Then the **RPAQ** following an initial "(" makes this line a candidate for the beginning of a bitmap definition. However, when **TestForBitmap** is applied to "**RPAQ Foo NIL)<*CR*>**" it rejects this candidate because the second expression after the **RPAQ** is not **(READBITMAP)**. Additionally, this line is a candidate for for a **VAR** definition, but the test function **TestForVar** *might* reject it if **Foo** is not contained in the FILECOMS as a **VARS**. (This however is not the kind of ambiguity signalled by the fourth item of a *type*; it is merely ambiguity at the candidate level).

Two functions are provided to aid in some very common paradigms for definitional formats:

(1) when the definition is made by a call to **PUTDEF** in the file, then the test string should just be **PUTDEF** and the test function should be **TestForGenericDefinition**; this test is a bit more complicated than would ordinarly be required, because it may require scanning the next couple of lines to find the arguments to the **PUTDEF** (i.e., the ***string*** argument to the test function doesn't contain enough of the file to make the decision).

(2) when the name of the item being defined appears as the first s-expression after the candidate-generating string, the the test function **TestForType** is adequate; in fact, if no test function is provided in the *type*, then **TestForType** will be used.

A few variables are bound as **SPECVARS** so that users' test functions may access important parts of state. The file being indexed is opened for input and **FULLS** is a stream into it. It is often useful to screen candidates of a particular type against the list of all items of that type as found on the FILECOMS of the file -- for this purpose, the variable, **typeNames**, is bound to a list whose members have the form:

(***typeName itemList***)

where ***itemList*** is the list of names as would be returned by **INFILECOMS?**.

Note 1: Both SINGLEFILEINDEX and MERGEDFILEINDEX run as background processes in INTERLISP-D.

Note 2: The files being indexed and/or listed needn't be loaded.

Note 3: The current version of the SINGLEFILEINDEX package replaces an older version which only indexed functions, used relative labels instead of absolute page numbers, and printed the function index before listing the file. Setting **RELATIVEINDEXFLG** to **T** will restore this behavior, if so desired (e.g. if you are printing on fanfold paper rather than cut sheets, you may want the index to appear before the file listing). If **RELATIVEINDEXFLG** is **NIL** (the current default), indexing will be as described above, and the indices will be printed *after* the file is listed. If, for some reason, both kinds of indexing are desired, give **RELATIVEINDEXFLG** the value **BOTH**.

Note 4: The SINGLEFILEINDEX package uses the following variables:

FILELINELENGTH to do the right-justification and columnating;

PRINTER (default value is **LPT:**) as the file to open to print the indexed version of the file;

LINESPERPAGE (initially 65 on D machines, 58 otherwise) as the number of lines on the printer.

The default values are only used if the variables were not bound at the time SINGLEFILEINDEX was loaded.

[1]If the type is a file package type, and there are no items of that type on the file (according to the FILECOMS), then the file is never searched for items of that type.

[2]with *OUTPUTFILE* and *MERGEDINDEXFLG* set to **NIL**.