# 4. HOW TO USE FILES

## Types of Files

A program file, or Lisp file, contains a series of expressions that can be read and evaluated by the Lisp interpreter. These expressions can include function or macro definitions, variables and their values, properties of variables, and so on. How to save Interlisp-D expressions on these files is explained in Chapter 7. Loading a file is explained in the Simple Commands for Manipulating Files section below.

Not all files, however, have Lisp expressions stored on them. For example, TEdit files store text; sketches are stored on files made with the package Sketch , or can be incorporated into TEdit files. These files are not loaded directly into the environment, but are accessed with the package used to create them, such as TEdit or Sketch.

When you name a file, there are conventions that you should follow. These conventions allow you to tell the type of file by the extension to its name.

| **If a file contains:** | **Then:** |
|---|---|
| Lisp expressions | It should not have an extension or have the extension `.LISP`. For example, a file called `MYCODE` should contain Lisp expressions. |
| Compiled Code | It should have the extension `.LCOM or .DFASL`. For example, a file called `MYCODE.DFASL` should contain compiled code. |
| A Sketch | Its extension should be `.SKETCH`. For example, a file called `MOUNTAINS.SKETCH` should contain a Sketch. |
| Text | It should have the extension `.TEDIT`. For example, a file called `REPORT.TEDIT` should contain text that can be edited with the editor `TEDIT`. |

## Directories

This section focuses on how you can find files, and how you can easily manipulate files. To see all the files listed on a device, use the function `DIR`. For example, to see what files are stored in your current directory, type:

```
(DIR *.*)
```

Partial directory listings can be gotten by specifying a file name, rather than just a device name. The wildcard character `*` can be used to match any number of unknown characters. For example, the command `(DIR T*)` will list the names of all files that begin with the letter `T`. An example using the wildcard is shown in Figure 4-1.


Figure 4-1. Using DIR with a Wildcard

## Directory Options

Various words can appear as extra arguments to the DIR command. these words give you extra information about the files.

SIZE displays the size of each file in the directory. For example, type:

```
(DIR {DSK} SIZE)
```

DATE displays the creation date of each file in the directory. An example of this is shown in Figure 4-2.

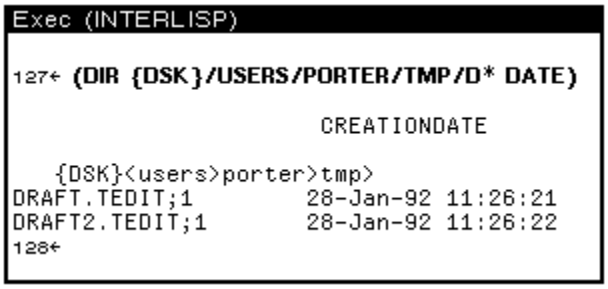


Figure 4-2. Example Using DATE

DEL deletes all the files found by the directory command.

## Subdirectories

Sudirectories are very helpful for organizing files. A set of files that have a single purpose (for example, all the external documentation files for a system) can be grouped together into a subdirectory.

To associate a subdirectory with a filename, simply include the desired subdirectory as part of the name of the file. Subdirectories are specified after the device name and before the simple filename. The first subdirectory should be between less-than and greater-than signs (angle brackets) < >, with nested subdirectory names only followed by a greater than sign >. For example:

```
{DSK}<Directory>SubDirectory>SubSubDirectory>...>filename
```

or use the UNIX convention:

```
{DSK}/Directory/Subdirectory/Subsubdirectory/filename
```

## To See What Files Are Loaded

If you type FILELST<CR>, the names of all the files you loaded will be displayed.

Type SYSFILES<CR> to see what files are loaded to create the sysout.

## Simple Commands for Manipulating Files

When using these functions, always be sure to specify the full filename, including subfile directories if appropriate.

To have the conents of a file displayed in a window:

(SEE '*filename*)

To copy a file (see Figure 4-3):

(COPYFILE '*oldfilename* '*newfilename*)


Figure 4-3. Example Use of COPYFILE

To delete a file (see Figure 4-4):

(DELFILE '*filename*)

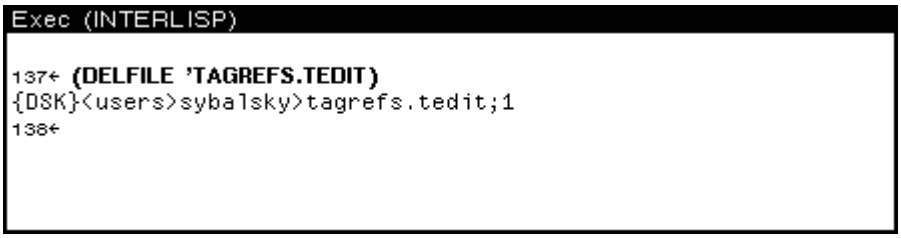


Figure 4-4. Example Use of DELFILE

To rename a file:

(RENAMEFILE '*oldfilename*'*newfilename*)

Files that contain Lisp expressions can be loaded into the environment. That means that the information on them is read, evaluated, and incorporated into the Medley environment. To load a file, type:

(LOAD '*filename*)

## Connecting to a Directory

Often, each person or project has a subdirectory where files are stored. If this is your situation, you will want any files you create to be put into this directory automatically. This means you should "connect" to the directory.

CONN is the Medley command that connects you to a directory. For example, CONN in Figure 4-5 connects you to the subsubdirectory IM, in the subdirectory PRIMER, the directory LISPFILES, on the device DSK. This information—the device and the directory names down to the subdirectory to which you want to be connected—is called the "path" to that subdirectory. CONN expects the path to a directory as an argument.

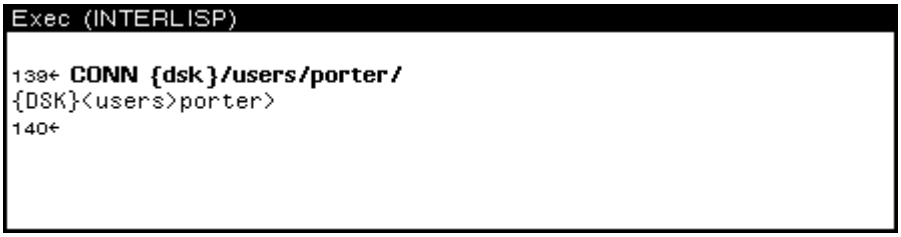


Figure 4-5. CONNecting to Subdirectory Primer Subsubdirectory IM

Once you are connected to a directory, the command DIR will assume you want to see the files in that directory, or any of its subdirectories.

Other commands that require a filename as an argument (e.g., SEE, above) will assume that the file is in the connecteds directory if there is no path specified with the filename. This will often save you typing.

## File Version Numbers

When stored, each filename is fillowed by a semicolon and a number, as shown in this example:

```
MYFILE.TEDIT;1
```

The number is the version number of the file. This is the system's way of protecting your files from being overwritten. Each time the file is written, a new file is created with a version number one greater than the last. This new file will have everything from your previous file, plus all of your changes.

In most cases, you can exclude the version number when referencing the file. When the version is not specified, and there is more than one version of the file on that particular directory, the system generally uses your most recent version. An exception is the function DELFILE, which deletes the oldest version (the one with the lowest version number) if none is specified.

[This page intentionally left blank]