19. WHERE DOES ALL THE TIME GO? SPY

SPY is an Lisp library package that shows you where you spend your time when you run your system. It is easy to learn, and very useful when trying to make programs run faster.

How to Use Spy with the SPY Window

The function SPY.BUTTON brings up a small window which you will be prompted to position. Using the mouse buttons in this window controls the action of the SPY program. When you are not using SPY, the window appears as in Figure 19.1.

Figure 19.1. SPY Window When SPY is Not Being Used

To use SPY, click either the left or middle mouse button with the mouse cursor in the SPY window. The window will appear as in Figure 19.2, and means that SPY is accumulating data about your program.

Figure 19.2. SPY Window When SPY is Being Used

To turn off SPY after the program has run, again click a mouse button in the SPY window. The eye closes, and you are asked to position another window. This window contains SPY's results. An example of the resulting window is shown in Figure 19.3.

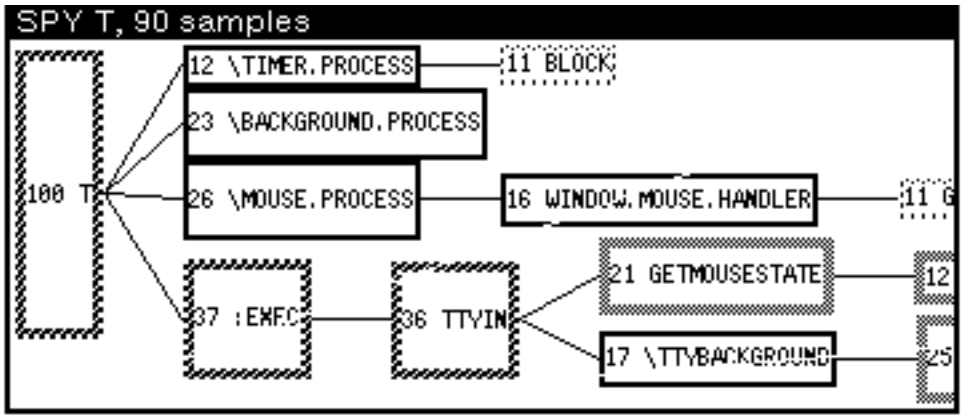

Figure 19.3. Window Produced After Running SPY

This window is scrollable horizontally and vertically. This is useful, since the whole tree does not fit in the window. If a part that you want to see is not shown, you can scroll the window to show the part you want to see.

How to Use SPY from the Lisp Top Level

SPY can also be run while a specific function or system is being used. To do this, type the function WITH.SPY:

```
(WITH.SPY form)
```

The expression used for form should be the call to begin running the function or system that SPY is to watch. If you watch the SPY window, the eye will blink! To see your results, run the function SPY.TREE. To do this, type:

```
(SPY.TREE)
```

The results of the last running of SPY will be displayed. If you do this, and SPY.TREE returns (no SPY samples have been gathered), your function ran too fast for SPY to follow.

Interpreting SPY's Results

Each node in the tree is a box that contains, first, the percentage of time spent running that particular function, and second, the function name. There are two modes that can be used to display this tree.

The default mode is cumulative. In this mode, each percentage is the amount of time that function spent on top of the stack, plus the amount of time spent by the functions it calls. The second mode is individual. To change the mode to individual, point to the title bar of the window, and press the middle mouse button. Choose Individual from the menu that appears. In this mode, the percentage shown is the amount of time the function spent on the top of the stack.

To look at a single branch of the tree, point with the mouse cursor at one of the nodes of the tree, and press the right mouse button. From the menu that appears, choose the

option `SubTree`. Another `SPY` window will appear, with just this branch of the tree in it.

Another way to focus within the tree is to remove branches from the tree. To do this, point to the node at the top of the branch you would like to delete. Press the middle mouse button, and choose `Delete` from the menu that appears.

There are also different amounts of "merging" of functions that can be done in the window. A function can be called by another function more than once. The amount of merging determines where the subfunction, and the functions that it calls, appear in the tree, and how often. (For a detailed explanation of merging, see the *Lisp Library Packages Manual*.)