PAGEHOLD

By: Jon L White
Currently maintained by: Bill van Melle (vanMelle.pa@Xerox.com)

INTRODUCTION

Loading PAGEHOLD.LCOM redefines the function `PAGEFULLFN` to alter the behavior that occurs when a tty window fills. Rather than inverting the window and waiting indefinitely for type-in, the PAGEHOLD module indicates the hold by an independent notification, and waits for only a specified interval before continuing. Thus, filling the window is no longer a cause for a program to hang indefinitely.

The default behavior of the PAGEHOLD module is to raise a "button" at the corner of the tty window flashing a message, alternating between

and

indicating that output to the window is being held. While in this state, you can release the hold by any of the following means:

Typing any character (of course, the window must own the tty process). This is the same as the old behavior;

Depressing the CTRL key;

Depressing and releasing either SHIFT key;

Clicking with LEFT on the button that announces the hold (clicking instead with MIDDLE gets a menu of options);

Waiting until the timeout has passed (initially, 20 seconds).

When you depress one of the SHIFT keys, the button stops flashing. Output will continue to be held indefinitely as long as one of the SHIFT keys is depressed, even if the timeout passes. If while holding down SHIFT, you depress the CTRL key for a second or so, the button will start flashing again; you may now release CTRL and then SHIFT, and the hold will be maintained without your needing to hold down SHIFT. You can release the hold by any of the means listed above.

If the CTRL key is down when a window fills, output is not held at all. Depressing the CTRL key immediately releases any hold in progress.

The remainder of this document describes ways of tailoring the behavior further.

Controlling the timeout

One of the primary motivations for the PAGEHOLD module is so that printout to a TTY window does not hang indefinitely when one "page" has filled up. The default release time is in the global variable PAGE.WAIT.SECONDS, which comes initialized to 20 seconds; a value of 0 causes immediate release (unless a SHIFT key is already depressed). If a window being held has a PAGE.WAIT.SECONDS property, then that value is used instead of the global default.

However, if PAGE.WAIT.SECONDS is set to STOP, then the hold will not be released by any automatic timeout, nor will it be sensitive to the SHIFT or CTRL key actions This mode most closely approximates the current Lisp design, except that a pop-up button signals the hold rather than a video inversion (mousing the button will, nevertheless, still effect a release). The message

"Scrolling Stopped"

appears in the button rather than one of the several "holding" messages.

The Pop-up "Buttons"

A secondary motivation for this facility is to have a pop-up "button" that interactively signals the user of a holding condition on a particular window without obscuring the window's contents, as video inversion does. In addition, the button permits the selective release of a particular window by mousing the button (note that holding down SHIFT, on the other hand, would affect *all* windows currently being held). There are three styles of buttons—WINKING, FLASHING, and NIL—and the selection is determined by the value of the global variable PAGE.WAIT.ACTIVITY, which comes initialized to WINKING. If a window has a PAGE.WAIT.ACTIVITY property, then that value is used instead of the global default, thus allowing different types of buttons on different windows.

A WINKING button is a fairly hefty pad—approximately 1/2" by 2 1/2"—which pops up just over the right side of the window's title bar; it will alternately print and clear two short holding messages: one in the upper half of the "button" and one in the lower half. A FLASHING button is about the same width, but half the height, and will alternately print the two holding messages. A NIL button merely shows the message "Release SHIFT for more".

LEFT-mousing any button causes an immediate release of the hold; MIDDLE-mousing the button brings up a menu offering several options. One of these is "Release this hold!", same as using LEFT. Other menu options permit conversion of the hold to indefinite "hold" or to STOP mode; additionally, five options are offered for setting the window's specific PAGE.WAIT.SECONDS property.

The WINKING button has a different pattern of activity when the hold is placed into indefinite hold mode, but the other button styles do not visibly distinguish this state. If there isn't room to place the button down over the right side of the title bar (because, for example, the window is too close to the screen top), then it will be placed over another corner of the window.

Keyboard Input and Typeahead

Consistent with Lisp's current action, there will be no holding on a window which is its process's TtyDisplayStream *and for which there is typeahead in that process's TTY input buffer*. This action can be overridden by setting PAGE.WAIT.IGNORETYPEAHEAD to a non-NIL value: typeahead does not inhibit the hold, character input does not release the hold, and no input is ever discarded (note that depressing the SHIFT and/or the CTRL keys does not generate character input). This feature is intended for those who dislike not knowing whether a keystroke will be consumed by the PAGEFULLFN—under the default behavior, if the TTY input buffer is empty, then the first character you

type will either (a) release a hold already in progress and be discarded, or (b) prevent subsequent holds and be retained, all depending on when exactly you type the character.