TTY Package

File: <lispusers>TTY.press
Revised: December 7, 1982 by Mark Stefik

The TTY package is a set of functions for interacting with the TTY. It is much simpler to use than ASKUSER. The input functions also provide a technique for suppressing prompts that makes it easy to create programs that respond to "commands" or sequenced entry with a varied amount of prompting for input.

These functions were originally written by Mark Stefik and Peter Friedland in 1978 for use with the UNITS package. They have been renamed and documented here.

INTTY (promptStr goodList helpStr noShiftFlg)

INTTY interacts with a user at the TTY and returns an atom. In its simplest usage, it prompts the user using the *promptStr*. The user then types a response followed by a carriage return. INTTY returns the user's response which is a member of *goodList*. If there is a sequence of calls to INTTY, the user can separate the responses by spaces and follow the last one with a carriage return.

More formally, the arguments to INTTY are as follows:

promptStr. A prompt typed to the TTY when INTTY is invoked. This prompt is suppressed if the user has typed ahead.

goodList. Optional argument. This is a list of acceptable answers. If the characters typed by the user correspond to the first characters of any element of *goodList*, that member is returned as the value of INTTY. If the leading characters are ambiguous, then user is warned. If *goodList* is supplied and the user's reply matches no entry, INTTY attempts spelling correction. On failure of spelling correction, all type ahead is flushed and the user is warned and reprompted. If *goodList* is NIL, no checking is performed.

helpStr. A secondary prompt if the user types a ? to INTTY. Intended to allow short prompts, and longer more explanatory prompts as needed.

noShiftFlg. Normally, INTTY converts the user's response to all upper case. If *noShiftFlg* is T, then the user's response is passed along as typed.

Example

The following statements:

```
(SETQ name (INTTY "Name: " NIL "Please type your first name"))
(SETQ age (INTTY "Age: " "Your age in years"))
(SETQ color (INTTY "Paint color: " '(RED BLUE YELLOW WHITE)))
```

could yield the following dialog (prompts in boldface):

Name: John <return>
Age: 17 <return>
Paint color: Green <return>
Response not recognized. Type ? for help.
Paint color: ? <return>
Expecting one of (RED BLUE YELLOW WHITE)
Paint color: blue <return>

which illustrates the help facility, or the following dialog

Name: John 17 yell <return>

which illustrates the suppression of type ahead and name completion. Note that no actual reading of the typing takes place until the <return> is typed. Successive calls to INTTY suppress the prompt, and read through the separating spaces. At the conclusion of execution, the LISP variables *name*, *age*, and *color* have all been set to JOHN, 17, and YELLOW respectively.

INTTYL (promptStr goodList helpStr noShiftFlg)

Similar to INTTY except that it always returns a list of the items typed. INTTYL reads successive entries until a carriage return is typed.

WRITE (arg1 arg2 ...)

Prints the arguments *arg1, arg2*, etc. to the TTY (using PRINT1). Prints a carriage return at the end. Output goes to the primary output device.

PrintStatus (arg1 arg2 ...)

Similar to WRITE except that PrintStatus allocates its own window and prints to that. The window is title "Status Window" and is made to scroll as writing proceeds.