8. PROGRAMMER'S INTERFACE TO TEDIT

Using the Top-Level TEdit Function

The top-level entry to TEdit is:

(TEDIT *TEXT WINDOW DONTSPAWN PROPS*) [Function]

TEXT may be a file name, an open STREAM, a string, or an arbitrary (MKSTRING-able) Lisp object. The text is displayed in an editing window and may be edited there. If *TEXT* is other than a file name, a STREAM, or a string, TEDIT will call MKSTRING on it and let you edit the result.

If *WINDOW* is NIL, you will be prompted to create a window. If *WINDOW* is non-NIL, TEDIT will use it as the window to edit in. If *WINDOW* has a title, TEDIT will preserve it; otherwise, TEDIT will provide a descriptive title for the window.

TEDIT will normally spawn a new process to run the edit, so you can edit in parallel with other work; indeed, it is possible to have several editing windows active on the screen. You can have the editing done in your process—and have TEdit return the result of the edit—by calling TEDIT with *DONTSPAWN* set to T.

PROPS

PROPS is a collection of properties in property-list format that control the editing session. The following options are possible:

LOOKS — The default character looks (font, size, etc.) to be used in the edit window if the text does not yet have explicit character looks. This can be a FONTDESCRIPTOR or a property list of character looks properties such as TEDIT.LOOKS would accept, or a CHARLOOKS data structure describing the character looks.

FONT — The default font to be used in the edit window if the text does not yet have explicit character looks. This can be a FONTDESCRIPTOR, or any form of font description that is acceptable to FONTCREATE. The FONT property is respected only if the LOOKS property is not specified.

PAGEFORMAT — The initial page layout specifications (see the functions TEDIT.PAGEFORMAT, TEDIT.SINGLE.PAGEFORMAT and TEDIT.COMPOUND.PAGEFORMAT, below)

QUITFN — A function (or list of functions) to call when the user Quits. (See "Using TEdit's User-Function 'Hooks' " for details.)

LOOPFN — A function to be called each time through the character-read loop. (See "Using TEdit's User-Function 'Hooks' " for details.)

CHARFN — A function to be called for each character typed in. (See "Using TEdit's User-Function 'Hooks' " for details.)

SELFN — A function to be called each time a mouse selection is made in this edit window. (See "Using TEdit's User-Function 'Hooks' " for details.)

TERMTABLE	If you want characters displayed other than TEdit's default way, set this to a terminal table with the correct settings.
READTABLE	If you want command characters that are local to this edit session, set this to a readtable with the appropriate settings.(See "Using TEdit's Terminal Table and Readtables" for details.)
BOUNDTABLE	If you want word breaks to happen other than the default way, set this to a readtable with the appropriate settings. (See TEDIT.WORDGET and TEDIT.WORDSET for details.)
READONLY	If the value of this property is non-NIL, then the edit window will be read-only, i.e., you can copy-select from it, but can't type in it.
CACHE	If the value of this property is non-NIL, then the file being edited will be cached locally instead of being read as needed from the remote server.
COPYBYBKSYSBUF	Affects copy selection: If the value of this property is non-NIL, TEdit will insert copied text into the keyboard buffer and read it as though it had been typed. The default is NIL, which means TEdit will copy text from window to window internally
SEL	Tells what text should be selected initially. This can be a SELECTION (see below) describing the selected text, a character number, a two-element list of first character number and number of characters to select, or DON'T, meaning that nothing is to be selected initially.
MENU	Describes the menu to be displayed when the left or middle mouse button is pressed in the edit window's title region. If it is a MENU, that menu will appear. If it is a list of menu items, a new menu will be constructed.
AFTERQUITFN	A function to be called *after* TEdit has quit. This can be used for cleanup of side effects by TEdit client programs. (See "Using TEdit's User-Function 'Hooks' " for details.)
TITLEMENUFN	A function to get called instead of bringing up the usual TEdit command menu when the user left- or middle-buttons in the edit window's title region.
PARALOOKS	The default paragraph looks to be used for paragraphs in this document. This can be either a FMTSPEC data structure or a property list of paragraph-formatting information such as TEDIT.PARALOOKS would accept.
CARETLOOKSFN	A function that is called (with the candidate looks as an argument) whenever new caret looks are being set. If it returns NIL, the looks will be changed. If the function returns a CHARLOOKS, it will be used in place of the candidate. This function is useful for applications that need to control the appearance of typed-in text.
LEAVETTY	If this is non-NIL, TEdit will not take control of the keyboard when it is started. Instead, it will wait until you first button in the editing window with the mouse.
PROMPTWINDOW	A window that is to be used for unscheduled user interactions, in place of the prompting window that TEdit usually provides. If this is the atom DON'T, no window will be provided, and the main prompt window will be used.

PROMPTWINDOWHEIGHT	The height of the (TEdit-created) prompting window, in lines.
OVERFLOWFN	A function called whenever the TEdit screen updater is about to move text off the bottom of the window. (See "Using TEdit's User-Function 'Hooks'" for details.)
CLEARGET	If this is non-NIL, TEdit will read the file as though it were plain text—regardless of whether it is a formatted file or not.
CLEARPUT	If this is non-NIL, TEdit will not produce any formatting information when the file is PUT.
NOEXTENT	If this is non-NIL, TEdit will not update the EXTENT of the editing window. This is for the use of applications that run TEdit in part of the window.
NOTITLE	If this is non-NIL, TEdit will never change the title on the editing window.
TTYWINDOW	If specified, this will be the window used to back the TEdit process's TTYDISPLAYSTREAM. Normally, TEdit makes a closed window that serves the purpose, which is a path for copy-selected items to get to the edit window.
INTERRUPTS	A list of Lisp interrupts, in the same form as TEDIT.INTERRUPTS, that will be enabled while this TEdit is running.
PUTFN	A function, called both before and after the TEdit PUT command is performed. (See "Using TEdit's User-Function 'Hooks'" for details.)
GETFN	A function, called both before and after the TEdit GET command is performed. (See "Using TEdit's User-Function 'Hooks'" for details.)
TEDIT.TENTATIVE	If this is non-NIL, TEdit will keep track of text that is "new" during the session and will record that fact as part of the file at PUT time. Any *PROPS* specified will be appended to the front of whatever is the value of TEDIT.DEFAULT.PROPS; respecified properties will override anything in the defaults. This provides client applications with a way to set default edit properties.

Using the TEdit Data Structures

Using the "Text Stream" Data Structure

TEdit keeps a STREAM that describes the current state of the text you're editing. You can use most of the usual stream operations on that stream: BIN, SETFILEPTR, GETFILEPTR, and GETEOFPTR do the usual things. BOUT inserts a character in the stream just in front of the next character you'd read if you BINned. You can get the stream by calling (TEXTSTREAM *EDITWINDOW*).

If you need to save the state of an edit, you can save this stream. Calling TEDIT with the stream as the *TEXT* argument will let you continue from where you left off.

There is a data type called TEXTSTREAM that defines two fields that are of interest within the stream, TEXTOBJ and PIECE.

TEXTOBJ: The TEXTOBJ that describes the edit session.

PIECE: The PIECE that describes the text at the file pointer.

Using the "Text Object" Data Structure

TEdit keeps a variety of other information about each edit window, in a data structure called a TEXTOBJ. Field TEXTOBJ of a text STREAM points to the associated TEXTOBJ, which contains the following fields of interest.

WINDOW — A list of the edit windows that contains the text. If this is NIL, there is no edit window for this text. There will be more than one window if the main editing window has been split into several panes.

SEL — The most recent selection made in this text.

SCRATCHSEL — A scratch SELECTION, used by the mouse handler for the edit window, but otherwise available for scratch use.

TEXTLEN — The current length of the edited text, in characters.

STREAMHINT — Points to the text STREAM that describes the text.

EDITFINISHEDFLG — If this is non-NIL, TEdit will halt after the next time through the keyboard-polling loop. No check will be made for unsaved changes. Unless it is T, the value of EDITFINISHEDFLG will be returned as the result of TEDIT if TEdit was called with *DONTSPAWN* = T.

Using the "Selection" Data Structure

The selected text is described by an object of type SELECTION, whose fields are as follows:

CH# — The character number of the first character in the selection. (The first character in the text being edited is numbered one.)

CHLIM — The character number of the character just *past* the last selected character. Must be at least as large as CH#.

DCH — The number of characters in the selection. If DCH is zero, then no characters are selected, and the selection can be used only to describe a place to insert text.

Any one of the above three fields may be derived from the other two.

ONFLG — Tells whether the selection is highlighted (or would be, if the selected text appeared within the edit window). If T, it is; if NIL, it's not.

\TEXTOBJ — The TEXTOBJ that describes the selected text. You can use this to get to the stream itself.

X0 — The X position (edit-window-relative) of the left edge of the first selected character.

Y0	The Y position of the bottom of the first selected character (not the character's baseline, the bottom of its descent).
XLIM	The X position of the right edge of the last character selected. If DCH is zero (a "point" selection), XLIM = X0.
YLIM	The bottom of the last character in the selection.
DX	The width of the selection. If DCH is zero, this will be also.
SELOBJ	This is for a future object-oriented editing interface.
POINT	Tells which side of the selection the caret should appear on. It will be one of the atoms LEFT and RIGHT.
SETT	If this selection is currently valid, NIL if it is obsolete or has never been set.
SELKIND	What kind of selection this is. One of the atoms CHAR, WORD, LINE, or PARA.
HOW	A TEXTURE, which will be used to highlight the selection.
HOWHEIGHT	How high the highlighting is to extend. A selection's highlight starts at the bottom of the lowest descender and extends upward for HOWHEIGHT pixels. To always get highlighting a full line tall, set this to 16384.
HASCARET	T if this selection should have a caret flashing next to it, NIL otherwise.

Using the TEdit Interface Functions

TEdit exports the following functions for use in custom interfaces:

(OPENTEXTSTREAM *TEXT WINDOW START END PROPS*) [Function]

Creates a text STREAM describing *TEXT,* and returns it. If *WINDOW* is specified, the text will be displayed there, and any changes to the text will be reflected there as they happen. You will also be able to scroll the window and select things there as usual. *TEXT* may be an existing TEXTOBJ or text STREAM. *PROPS* is the same as for TEDIT.

If *START* and *END* are given, only the section of *TEXT* delimited by them is edited (if that portion of the file looks itself like a TEdit-structured file, then TEdit will honor the font, paragraph, and IMAGEOBJ information. Otherwise, it will be treated as a plain-text file). *START* is the file pointer of the first character to be included (NIL defaults to the beginning of the file; *END* is a file pointer pointing at the character just *past* the final character to be included (NIL defaults to the end of the file).

Given the STREAM, you can use a number of functions to change the text in an edit window, under program control. The edit window gets updated as the text is changed.

(TEDIT.SETSEL *STREAM CH# LEN POINT PENDINGDELFLG LEAVECARETLOOKS OPERATION*) [Function]

Sets the selection in *STREAM.* If *CH#* is a SELECTION, it is used as-is. Otherwise, *CH#* is the first character in the selection, and

LEN is the number of characters to select (zero is allowed, and gives just an insertion point). Note that character numbers start at one, unlike file pointers, which start at zero. *POINT* tells which side of the selection the caret should come on. It must be one of the atoms LEFT or RIGHT.

If *PENDINGDELFLG* is non-NIL, the selection will be a pending-delete selection—the selected text will be deleted at the next type-in (or if text is copied or moved there). Otherwise, the selection will be a normal selection.

Normally, the act of making a selection sets the "caret" looks—the looks for any characters typed at the caret. This can be suppressed by passing in a non-NIL *LEAVECARETLOOKS*.

Selections made for different purposes look different. You can specify what kind of selection this is to be with the *OPERATION* argument. It may be one of the atoms NORMAL, MOVE, COPY, PENDINGDEL, or DELETE (to make the selection look like that kind of selection), and in addition, can be INVERTED, which just makes the selection black, but leaves the caret flashing.

(TEDIT.GETSEL *STREAM*) [Function]

Returns a copy of the current selection in the edit window described by *STREAM*.

(TEDIT.SHOWSEL *STREAM ONFLG SEL*) [Function]

Lets you turn the highlighting of the selection *SEL* on and off. If *ONFLG* is T, the selection *SEL* in *STREAM* will be highlighted in the edit window; if NIL, any highlighting will be turned off. If *SEL* is NIL, it defaults to the current selection in *STREAM*.

(TEDIT.SET.SEL.LOOKS *SEL OPERATION*) [Function]

Changes the highlighting of *SEL* to be appropriate to the kind of selection specified in *OPERATION*. *OPERATION* may be one of the atoms NORMAL, MOVE, COPY, PENDINGDEL, or DELETE (to make the selection look like that kind of selection), and in addition, can be INVERTED, which just makes the selection black, but leaves the caret flashing.

(TEDIT.SEL.AS.STRING *STREAM SEL*) [Function]

Returns the currently selected text as a string. If *SEL* is non-NIL, the text it describes will be returned.

(TEDIT.GETPOINT *STREAM SEL*) [Function]

Returns the character number within *STREAM* that the next character typed would be inserted in front of (the "insertion point"). If *SEL* is given, it determines the result.

(COERCETEXTOBJ *STREAM TYPE OUTPUTSTREAM*) [Function]

Converts the text stream, *STREAM*, or edit window into another form, specified by *TYPE*. The possible values for *TYPE* are STRINGP, FILE, STREAM, and SPLIT.

STRINGP COERCETEXTOBJ will return a string (with any formatting and font information stripped out).

FILE COERCETEXTOBJ will return a file containing the document's text (complete with formatting and font information).

STREAM COERCETEXTOBJ will return a stream from which you may BIN or otherwise read the document.

SPLIT COERCETEXTOBJ will return a list of two files. The first contains the text for the document, and the second contains the formatting information. If these files are concatenated, they make a complete, legal TEdit file.

(TEDIT.INSERT *STREAM TEXT CH#orSEL LOOKS DONTSCROLL*) [Function]

Inserts the string *TEXT* into *STREAM,* as though it had been typed in. *CH#orSEL* tells where to insert the text: If it's NIL, the text goes in where the caret is. If it's a number, the text is inserted in front of the corresponding character (the first character in the stream is numbered one). If it's a SELECTION, the text is inserted accordingly.

If the *LOOKS* argument is provided, it must be a font descriptor. The inserted text will appear in that font.

Normally, TEdit scrolls the editing window so that each change is visible as it is made. If you want the window left where it is instead, the *DONTSCROLL* argument should be non-NIL.

(TEDIT.DELETE *STREAM SEL LEN*) [Function]

Deletes text from *STREAM.* If *SEL* is a SELECTION, the text it describes will be deleted; if *SEL* is a number, it is the character number of the first character to delete. In that case, *LEN* must also be present; it is the number of characters to be deleted.

(TEDIT.INCLUDE *STREAM FILE START END*) [Function]

Performs the TEdit Include command, inserting the text from file *FILE* into *STREAM.* If *START* and *END* are supplied, only the specified portion of the file is included.

(TEDIT.RAW.INCLUDE *STREAM INFILE START END*) [Function]

Enables you to quickly include characters from unformatted files. It is the same as TEDIT.INCLUDE, except that the specified characters from *INFILE* are included without checking to see if *INFILE* is a TEdit file or any other format that would be converted to TEdit format (e.g., Bravo format).

(TEDIT.GET *STREAM FILE UNFORMATTED?*) [Function]

Performs the TEdit Get command, loading the text from *FILE* onto the editing stream *STREAM*—replacing the text that is being edited currently. If *FILE* is not supplied, the user will be asked for a file name. If *UNFORMATTED?* is non-NIL, *FILE* is treated as a plain-text document, and all of its contents are included—even TEdit formatting information.

(TEDIT.PUT *STREAM FILE FORCENEW UNFORMATTED? OLDFORMAT?*) [Function]

Performs the TEdit Put command, saving the text from the text stream *STREAM* onto the file named *FILE.* If *FILE* is NIL, the user will be prompted for a file name. In this case, if *FORCENEW* is NIL, the user is offered the old file name as a default; if non-NIL, no default is given, forcing the user to specify a file name. If *UNFORMATTED?* is non-NIL, only characters are put in the file—no formatting. If *OLDFORMAT?* is non-NIL, the file will be

written in the format used by the previous version of TEdit, for backward compatibility.

In order to store a TEdit document as part of another file, call TEDIT.PUT, passing an open stream on the file as the *FILE* argument. The stream should be open for output and positioned at the place you want TEdit to store the document (call this file pointer START). When TEDIT.PUT returns, the stream's end-of-file pointer will be just after the last byte in the newly-inserted document. Call this file pointer END. To subsequently retrieve the document from the middle of this other file, call OPENTEXTSTREAM on the file, passing the START and END pointers as the *START* and *END* arguments.

Note: When TEDIT.PUT returns, the stream will be open for INPUT.

(TEDIT.FIND *STREAM TARGETSTRING START# END# WILDCARDS?*) [Function]

Searches for the next occurrence of *TARGETSTRING* inside *STREAM*. If *START#* is present, the search starts there; otherwise, the search starts from the caret. If *END#* is present, the search will end at that character; otherwise, it ends at the end of the text. If a match is found, TEDIT.FIND returns the character number of the first character in the matching text. If no match is found, it returns NIL.

If *WILDCARDS?* is non-NIL, the search pattern can contain wild card characters: "#" matches any single character, "*" matches any sequence of characters, and "'" can be used to quote one of the wild cards. When wild cards are enabled, TEDIT.FIND returns a list consisting of the character numbers of the first and last characters in the matching text.

(TEDIT.HARDCOPY *STREAM FILE DONTSEND BREAKPAGETITLE SERVER PRINTOPTIONS*) [Function]

Sends the text contained in *STREAM* to the printer. If a file name is given in *FILE,* the press file will be left there for you to use. If *DONTSEND* is non-NIL, the file will not be sent to the printer; use this if you only want to create a press file for later use.

If *BREAKPAGETITLE* is non-NIL, it is used as the title on the "break page" printed before the text.

You can specify the print server where the hardcopy is to be sent, using the *SERVER* argument; if it is NIL, TEdit uses DEFAULTPRINTINGHOST.

You may also specify printing options (number of copies, whether to print on both sides of the paper, etc.) using *PRINTOPTIONS*. It is a "property list" in the form accepted by SEND.FILE.TO.PRINTER (see the *Interlisp-D Reference Manual*).

FILE may also be an open image stream. If so, the hardcopy output will be appended to the already open stream, and the stream will be left open when TEdit is finished.

(TEDIT.LOOKS *STREAM NEWLOOKS SELORCH# LEN*) [Function]

Changes the character looks of selected characters, e.g., the font, character size, etc. *SELORCH#* can be a SELECTION, an integer, or NIL. If *SELORCH#* is a SELECTION, the text it describes will be

changed; if it is a FIXP, it is the character number of the first character to be changed. In that case, *LEN* must also be present; it is the number of characters to be changed. A *SELORCH#* of NIL will use the current selection.

NEWLOOKS is a property-list-like description of the changes to be made. The property names tell what to change, and the property values describe the change. Any property that isn't changed explicitly retains its old value. Thus, it is possible to make a piece of text all bold without changing the fonts the text is in. The possible list entries are as follows:

FAMILY — The name of the font family (e.g., Modern). All the selected text is changed to be in that font.

FACE — The face for the new font. This may be in either of the two forms acceptable to FONTCREATE: a list such as (BOLD ITALIC REGULAR), or an atom such as MRR.

WEIGHT — The new weight for the font. This must be one of LIGHT, MEDIUM, or BOLD. Specifying this *disables* the FACE parameter.

SLOPE — The new slope for the font. This must be one of REGULAR or ITALIC. Specifying this *disables* the FACE parameter.

EXPANSION — The new weight for the font. This must be one of CONDENSED, REGULAR, or EXPANDED. Specifying this *disables* the FACE parameter.

SIZE — The new point size.

UNDERLINE — The value for this property must be one of the atoms ON or OFF. The text will be underscored or not, accordingly.

OVERLINE — The value for this property must be one of the atoms ON or OFF. The text will be overscored or not, accordingly.

STRIKEOUT — The value for this property must be one of the atoms ON or OFF. The text will be struck through with a single line or not, accordingly.

SUPERSCRIPT — A distance, in points. The text will be raised above the normal baseline by that amount. This is mutually exclusive with SUBSCRIPT.

SUBSCRIPT — A distance, in points. The text will be raised above the normal baseline by that amount. This is mutually exclusive with SUPERSCRIPT.

PROTECTED — The value for this property must be one of the atoms ON or OFF. If it is ON, the text will be protected from mouse selection and from deletion.

SELECTPOINT — The value for this property must be one of the atoms ON or OFF. If a character has this property, the user can make a point selection just after it, even if the character is also PROTECTED.

INVISIBLE — The value for this property must be one of the atoms ON or OFF. If a character has this property, the character will not appear on the screen or on hardcopy.

INVERTED — The value for this property must be one of the atoms ON or OFF. If set to ON, the text with that look will appear in inverse video on the screen, but not on hardcopy.

(TEDIT.GET.LOOKS *STREAM CH#ORCHARLOOKS*) [Function]

Returns a property list describing the character looks of the specified character(s). This list is suitable for passing to TEDIT.LOOKS. *CH#ORCHARLOOKS* can be a SELECTION, an integer, or NIL (meaning the current selection in *STREAM*). If *CH#ORCHARLOOKS* describes more than one character, the character looks of the first character are returned.

(TEDIT.COPY.LOOKS *STREAM SOURCE DEST*) [Function]

Makes the characters described by *DEST* have the same character looks as those described by *SOURCE* in the TEXTSTREAM *STREAM*. *SOURCE* and *STREAM* may both be SELECTIONs or integers. Integers are interpreted as character positions in *STREAM*. If *DEST* is a SELECTION, it must be in *STREAM*, whereas *SOURCE* may be a SELECTION from any TEXTSTREAM. When *SOURCE* describes characters with multiple looks, the looks are taken from the first character.

(TEDIT.PARALOOKS *STREAM NEWLOOKS SEL LEN*) [Function]

Changes the paragraph looks of selected paragraphs, e.g., the margins, line leading, etc. *SEL* can be a SELECTION, an integer, or NIL. If *SEL* is a SELECTION, the text it describes will be changed; if it is a FIXP, it is the character number of the first character to be changed. In that case, *LEN* must also be present; it is the number of characters to be changed. A *SEL* of NIL will use the current selection. In all cases, TEDIT.PARALOOKS operates on *whole paragraphs*. If any portion of a paragraph is included in the selection, the entire paragraph's looks will be changed.

NEWLOOKS is a property-list-like description of the changes to be made. The property names tell what to change, and the property values describe the change. Any property that isn't changed explicitly retains its old value. Thus, it is possible to make a paragraph indented without changing its tab stops. The possible list entries are as follows:

QUAD — One of LEFT (for flush-left, ragged-right), CENTERED (for centered lines), RIGHT (for flush-right, ragged-left), or JUSTIFIED (for flush-left and -right).

1STLEFTMARGIN — The left margin for the first line of the paragraph, in points.

LEFTMARGIN — The left margin for the rest of the paragraph, in points.

RIGHTMARGIN — The right margin for all lines of the paragraph, in points. If this value is zero, one gets a "floating" right margin, which adjusts to the width of the edit window or paper.

SPECIALX — The *X* position of a "specially positioned" paragraph, in picas. Specifying a value of zero turns off special positioning.

SPECIALY — The *Y* position of a "specially positioned" paragraph, measured from the bottom of the paper in picas. Specifying a value of zero turns off special positioning.

HEADINGKEEP — If set to ON, the paragraph so set will be kept with the beginning of the next paragraph, so that a title or heading will stay with the text it heads.

TABS — This is a CONS pair, *(DefaultTabWidth . TabStops)*. A tab advances the cursor to the next absolute tab stop to the right of the current position. Should there be no absolute tab stop to the right of the cursor, the cursor is advanced to the next even multiple of *DefaultTabWidth*. If *DefaultTabWidth* is NIL, the cursor moves to the next multiple of $\frac{1}{2}$ inch (36 points).

TabStops is a list of the tab stops for this paragraph. Each tab stop is represented as (*Location* . *Type*), where *Location* is the tab's location in points from the left margin. Entries must be in ascending order of *Location*s. *Type* is one of:

LEFT	Left tab
RIGHT	Right tab
CENTERED	Centered tab
DECIMAL	Decimal tab
DOTTEDLEFT	Left tab and dotted leader
DOTTEDRIGHT	Right tab and dotted leader
DOTTEDCENTERED	Centered tab and dotted leader
DOTTEDDECIMAL	Decimal tab and dotted leader

LINELEADING — The space to be left before each line of the paragraph, in points.

PARALEADING — Additional space to be left before the first line of the paragraph, in points.

POSTPARALEADING — Additional space to be left after the last line of the paragraph, in points.

BASETOBASE — The distance, in points, between the baselines of adjacent lines in this paragraph. This overrides the LINELEADING specification. However, the PARALEADING and POSTPARALEADING still affect this paragraph's relationship to other paragraphs. Specifying BASETOBASE as NIL turns off base-to-base leading and makes LINELEADING work again.

(TEDIT.GET.PARALOOKS *TEXTSTREAM SELORCH#*) [Function]

Returns a property list describing the paragraph looks of the specified character(s). This property list is suitable for passing to TEDIT.PARALOOKS. *SELORCH#* can be a SELECTION, an integer, or NIL (meaning the current selection in *STREAM*). If *SELORCH#* describes more than one character, the paragraph looks of the first character are returned.

(TEDIT.COPY.PARALOOKS *STREAM SOURCE DEST*) [Function]

Makes the characters described by *DEST* have the same paragraph looks as those described by *SOURCE* in the TEXTSTREAM *STREAM*. Details are analagous to TEDIT.COPY.LOOKS.

(TEDIT.PAGEFORMAT *STREAM FORMAT*) [Function]

This function sets the page format specifications for the given text stream. *FORMAT* is either a format specification for a single

page, which will be used for all pages in the document, or a "compound specification," which gives individual pages specifications for the first page, all other right (recto) pages, and all left (verso) pages. To create a "single page" format specification, use the function TEDIT.SINGLE.PAGEFORMAT (described below). To create a compound specification, generate three single-format specifications, and combine them into one with the function TEDIT.COMPOUND.PAGEFORMAT (described below).

(TEDIT.SINGLE.PAGEFORMAT *PAGE#S? PX PY PFONT PQUAD LEFT RIGHT TOP BOTTOM COLS COLWIDTH INTERCOL HEADINGS UNITS PAGEPROPS PAPERSIZE*) [Function]

PAGE#S? T if you want page numbers on this kind of page, else NIL.

PX The horizontal location of the page number, measured from the left edge of the paper. Negative values are measured from the paper's right edge.

PY The vertical location of the baseline for the page numbers, measured from the bottom of the paper. Negative values are measured from the top of the paper.

PFONT The font to be used to display the page numbers. This can be any specification that is acceptable to TEDIT.LOOKS.

PQUAD An atom that tells how the page number is to be aligned on the location specified by *PX* and *PY*. LEFT means the location is the lower-left corner of the page number. RIGHT means the location is the lower-right corner. CENTERED means the page number will be centered around the *PX* you specified.

LEFT The left margin—the distance from the left edge of the paper to the left edge of the first text column.

RIGHT The right margin—the distance from the right edge of the rightmost text column to the right edge of the paper.

TOP The top margin of the page—the distance from the top of the paper to the top of the first line of body text.

BOTTOM The bottom margin—the distance from the bottom of the last line of body text to the bottom of the paper.

COLS Number of columns (default is one).

COLWIDTH The column width (default is to evenly divide the available space among the *#COLS* columns).

INTERCOL The space between the right edge of one column and the left edge of the next column. Defaults to evenly divide the space left after the columns are set up. If there is more than one column, one or the other of *COLWIDTH* and *INTERCOLSPACE must* be specified.

HEADINGS A list of lists in the form of ((HEADINGNAME$_1$ XLOCATION$_1$ YLOCATION$_1$) (HEADINGNAME$_2$ XLOCATION$_2$ YLOCATION$_2$) . . .(HEADINGNAME$_n$ XLOCATION$_n$ YLOCATION$_n$)).

UNITS The units used in setting the values you specified. May be one of the atoms PICAS, IN, INCHES, CM, POINTS. Default is POINTS.

PAGEPROPS A property list of extra information. Properties are STARTINGPAGE#, FOLIOINFO, and LANDSCAPE?.

STARTINGPAGE# is the first page's number; it is ignored if this isn't the first page.

FOLIOINFO is a list of information about page numbers, (FORMAT TEXTBEFORE TEXTAFTER). FORMAT can be one of ARABIC, LOWERROMAN, UPPERROMAN, or NIL (i.e., ARABIC). TEXTBEFORE is the text preceding the number, and TEXTAFTER is the text following the number.

LANDSCAPE? determines if the document is printed in the usual vertical format or printed in landscape format (horizontally). If NIL the document is printed vertically, if non-NIL the document is printed landscape. Defaults to NIL.

PAPERSIZE Is one of LETTER, LEGAL, the metric paper sizes (A0, A1, A2 A3, A4, A5, B0, B2, B3, B4), or NIL (which defaults to letter size).

(TEDIT.COMPOUND.PAGEFORMAT *FIRST VERSO RECTO*) [Function]

Creates and returns a "compound specification," which gives individual page specifications for the first page, all other right (recto) pages, and all other left (verso) pages. *FIRST, VERSO,* and *RECTO* should be PAGEREGION objects created by TEDIT.SINGLE.PAGEFORMAT.

(TEDIT.QUIT *STREAM VALUE*) [Function]

STREAM must be the text stream associated with a running TEdit. TEDIT.QUIT causes the editing session to end, without executing the QUITFN in the property list. If *VALUE* is given, it is returned as TEdit's result; otherwise, TEdit will return the usual result. The user is not asked to confirm his or her desire to stop editing.

(TEDIT.KILL *STREAM*) [Function]

STREAM must be the text stream, TEXTOBJ, or edit window associated with a running TEdit. TEDIT.KILL kills the TEdit process and cleans up its data structures, without executing the QUITFN in the property list. It does not cause TEdit to return a result.

(TEDIT.ADD.MENUITEM *MENU ITEM*) [Function]

Adds a menu *ITEM* to *MENU.* This will update the menu's image so that the newly added item will appear the next time the menu pops up. This is only guaranteed to work right with pop-up menus that aren't visible.

(TEDIT.REMOVE.MENUITEM *MENU ITEM*) [Function]

Removes a menu *ITEM* from *MENU.* This will update the menu's image so that the newly added item will appear the next time the menu pops up. This is only guaranteed to work right with pop-up menus that aren't visible. *ITEM* may be either the whole menu item, or just the indicator that appears in the menu's image.

(TEXTOBJ *STREAM*) [Function]

Given a text stream, a TEXTOBJ, a running TEdit process, or a TEdit editing window, returns the associated TEXTOBJ.

(TEXTSTREAM *STREAM*) [Function]

Given a stream, a TEXTOBJ, a text stream, a running TEdit process, or a TEdit editing window, returns the associated text stream.

(TEXTPROP *TEXTOBJ/STREAM PROPNAME VALUE*) [Function]

Queries or sets the value of editor properties, such as the ones passed to TEDIT or OPENTEXTSTREAM in their *PROPS* arguments. This can also be used to associate user data with an editing session. If *VALUE* is omitted, the current value associated with *PROPNAME* is returned; if *VALUE* is present, it becomes *PROPNAME*'s associated value.

(TEDIT.FORMATTEDFILEP *STREAM*) [Function]

Tells whether a given text stream is plain text (result is NIL), must be stored as a special TEdit-format file (result is one of the atoms CHARLOOKS, PARALOOKS, or IMAGEOBJ, depending on the amount of formatting information that must be stored), or contains NS characters in other than character set zero (result is NSCHARS). The return NSCHARS is a sign that you may want to save TEdit formatting information. However, it is *not* necessary to save TEdit formatting information for the output file to have valid NS characters (though TEdit won't recognize them without that formatting information).

(TEDIT.CARETLOOKS *STREAM LOOKS*) [Function]

The looks of newly typed characters are controlled by the looks that are "attached to the caret." This function lets you set those looks for a given document. *LOOKS* is either a font descriptor, a CHARLOOKS, or a property list of character looks (such as TEDIT.LOOKS would accept). Any text inserted or typed in thereafter will appear in that font (or with those looks).

(TEDIT.STREAMCHANGEDP *STREAM RESET?*) [Function]

Returns T if the text represented by the *STREAM* has been modified since it was last saved. If *RESET?* is non-NIL, then the change indicator will be reset—i.e., TEdit will then believe that the text is unchanged and will not ask for confirmation of the Quit and Get operations, unless further changes are made.

(TEDIT.NORMALIZECARET *STREAM SEL*) [Function]

Makes sure that the caret is visible in the editing window; if not, the document is scrolled to place the caret on the top line of the window. This is normally controlled by the existing selection for the given text stream. However, if *SEL* is specified, it is used to decide the caret's location.

(COPYTEXTSTREAM *ORIGINAL CROSSCOPY*) [Function]

Makes a fresh copy of the text stream *ORIGINAL*. If *CROSSCOPY* is non-NIL, the new stream will not share structure with the old one—it can be edited without affecting the original stream.

(TEDIT.SELECTED.PIECES *TEXTOBJ SEL CROSSCOPY PIECEMAPFN FNARG1 FNARG2*) [Function]

Returns a list of PIECEs that describe the text selected in the selection *SEL* out of the document *TEXTOBJ*. If *CROSSCOPY* is non-NIL, the pieces will be copied.

PIECEMAPFN, if given, is applied in turn to arguments (*PIECE, TEXTOBJ, FNARG1, and FNARG2*), and the value it returns is used in place of the piece (or its copy).

(TEDIT.SUBLOOKS *TEXTSTREAM OLDLOOKSLIST NEWLOOKSLIST*) [Function]

Replaces all looks in *TEXTSTREAM* that are a superset of the looks specified in *OLDLOOKSLIST* with the looks produced by changing the aspects of the looks specified in *NEWLOOKSLIST.* Both *OLDLOOKSLIST* and *NEWLOOKSLIST* are in the format of TEDIT.LOOKS. e.g., (TEDIT.SUBLOOKS TEXTSTREAM '(FAMILY CLASSIC) '(FAMILY MODERN SIZE 12)) would change all looks in TEXTSTREAM that were in family CLASSIC to 12-point MODERN. All other aspects of the looks, e.g., WEIGHT, remain unchanged.

(TEDIT.PROMPTPRINT *TEXTSTREAM MSG CLEAR?*) [Function]

Prints a message in the TEdit prompting window associated with the given *TEXTSTREAM.* If *CLEAR?* is non-NIL, the window will be cleared first.

(TEDIT.MOVE *FROM TO*) [Function]

FROM and *TO* are SELECTIONs. Moves the text described by *FROM* to the place described by *TO*, within the same text stream or between different text streams. The text described by *FROM* is deleted from its original location.

(TEDIT.COPY *FROM TO*) [Function]

FROM and *TO* are SELECTIONs. Copies the text described by *FROM* to the place described by *TO*, within the same text stream or between different text streams. The text described by *FROM* is not deleted in the *FROM* location.

Using TEdit's User-Function "Hooks"

TEdit provides a number of hooks where a user-supplied function can be called. To supply a function, attach it to the TEXTSTREAM under the appropriate indicator, using TEXTPROP, or specify it in the *PROPS* argument to TEDIT or OPENTEXTSTREAM. Most of these functions are APPLYed to the text STREAM that describes the text, and other arguments specific to the function; the descriptions below contain the details.

QUITFN [TEdit property]

A function (or list of functions) to be called whenever the user ends an editing session: (QUITFN *WINDOW TEXTSTREAM TEXTOBJ*). This function may do any sort of testing or cleanup. If any of the functions is T or returns T, the user will not be asked to confirm the Quit—even if the document has unsaved changes; if it returns the atom DON'T, TEdit will not terminate. Any other result permits TEdit to do its normal cleanup and termination.

AFTERQUITFN [TEdit property]

A function to be called *after* the user ends an editing session: (AFTERQUITFN *TEXTSTREAM*). This may perform any cleanup of side effects that you desire.

LOOPFN [TEdit property]

A function that gets called, for effect only, each time through TEdit's main command loop: (LOOPFN *TEXTSTREAM*).

CHARFN [TEdit property]

A function that gets called once for each character typed into TEdit: (CHARFN *TEXTSTREAM CHARCODE*). The character code of the newly typed character is passed to the function as its second argument. The function may return one of three values: T means that the character should be processed as typed; NIL means that the character is to be ignored; any other numeric value is taken as a character code to be processed *in place of* the typed character.

SELFN [TEdit property]

A function that gets called each time the user tries to select something with the mouse: (SELFN *TEXTOBJ SELECTION SELECTMODE FINAL?*). It is called once for each tentative selection (e.g., while the mouse button is still down, but the mouse gets moved), and once—for effect only—after the selection is finalized. The new SELECTION is passed as the function's second argument, and an atom describing the kind of selection (one of NORMAL, COPY, MOVE, PENDINGDEL (for an extended selection that will be deleted on type-in), or DELETE) as the third. When the function is being called with a candidate selection, *FINAL?* will be the atom TENTATIVE; when being called with the final selection, *FINAL?* is the atom FINAL.

When the function is called with a candidate selection, it may veto that selection by returning the atom DON'T. This can be used to limit selections to items of interest. If a selection is vetoed, the old selection will remain highlighted; the effect is that of the user being unable to move the selection from its old location.

PRESCROLLFN [TEdit property]

A function called just before TEdit scrolls the edit window: (PRESCROLLFN *EDITWINDOW*).

POSTSCROLLFN [TEdit property]

Called just after TEdit scrolls the edit window: (POSTSCROLLFN *EDITWINDOW*).

OVERFLOWFN [TEdit property]

A function called whenever the TEdit screen updater is about to move text off the bottom of the window. Called with the arguments *TEXTSTREAM* and *EDITWINDOW,* the function may perform whatever cleanup or recovery operations are necessary. If it handles the overflow completely, the function should return T; if the function returns NIL, the screen updater will do its usual processing. This function is intended for use with the REGION

property, when an application will be running TEdit in *part* of a window, and needs to handle text overflows.

PUTFN [TEdit property]

Called both before and after the TEdit PUT command is performed, with arguments *TEXTSTREAM FULLFILENAME* and one of the atoms BEFORE or AFTER. When called before the PUT, this function may return the atom DON'T, which aborts the PUT process. Generally, this function is present for TEdit client systems to perform their own cleanup.

GETFN [TEdit property]

Called both before and after the TEdit GET command is performed, with arguments *TEXTSTREAM FULLFILENAME* and one of the atoms BEFORE or AFTER. When called before the GET, this function may return the atom DON'T, which aborts the GET process. Generally, this function is present for TEdit client systems to perform their own initialization and cleanup.

TEDIT.TITLEMENUFN [TEdit property]

Called whenever the user presses the LEFT or MIDDLE mouse button in the edit window's title region, with the editing window as its argument. Normally, this is the function TEDIT.DEFAULT.MENUFN, which brings up the usual TEdit command menu.

CARETLOOKSFN [TEdit property]

Called whenever TEdit is about to set the caret looks for an edit window. This function, called as (CARETLOOKSFN *NEWLOOKS TEXTOBJ*) may perform whatever checking it likes, and then return either the atom DON'T, meaning that the caret looks are not to be changed, NIL, meaning that *NEWLOOKS* should be used as the caret looks, or a new CHARLOOKS that will be used as the caret looks.

Note: If this function returns a new CHARLOOKS, it must *not* be a smashed version of *NEWLOOKS*.

TEdit also saves pointers to its data structures on each edit window. They are available for any user function's use.

TEDIT.MENU.COMMANDS [TEdit property]

A list of menu items used by TEDIT.DEFAULT.MENUFN to create a pop-up menu.

TEXTOBJ [Window property]

The TEXTOBJ that describes the current editing session.

TEXTSTREAM [Window property]

The text STREAM that describes the text of the document.

Changing the TEdit Command Menu

You may replace the middle-button command menu with one of your own. When you press the middle button in an edit window's title region, TEDIT calls the value of the TEDIT.TITLMENUFN window property with the window as its

argument. Normally, what gets called is TEDIT.DEFAULT.MENUFN, but you may change it to anything you like.

TEDIT.DEFAULT.MENUFN brings up a menu of commands. If the edit window has a property TEDIT.MENU, that menu is used. If not, TEdit looks for the window property TEDIT.MENU.COMMANDS (a list of menu items) and constructs a menu from that. Failing that, it uses TEDIT.DEFAULT.MENU.

This means that you can control the command menu by setting the appropriate window properties. Alternatively, you may add your own menu buttons to the default menu, TEDIT.DEFAULT.MENU.

(TEDIT.ADD.MENUITEM TEDIT.DEFAULT.MENU *ITEM*)

Will add *ITEM* to the TEdit menu. Menu items should be in the form (*NAME* (QUOTE *FUNCTION*)), where *NAME* is what appears in the menu, and *FUNCTION* will be applied to the text stream and can perform any operation you desire.

Finally, you may *remove* menu items from the default menu, by doing

(TEDIT.REMOVE.MENUITEM TEDIT.DEFAULT.MENU *ITEM*)

ITEM can be either a complete menu item, or just the text that appears in the menu; either will do the job.

Using TEdit's Global Variables

There are a number of global variables that control TEdit, or that contain state information for editing sessions in progress:

TEDIT.EXTEND.PENDING.DELETE [Variable]

If this is non-NIL, extending a selection makes it into a pending-delete selection. See the selection section. TEdit sets this initially to T.

TEDIT.DEFAULT.FMTSPEC [Variable]

A paragraph-looks description. This contains the default looks for a paragraph.

TEDIT.SELECTION [Variable]

A SELECTION. This is the most recent regular selection made in *any* TEdit window.

TEDIT.SHIFTEDSELECTION [Variable]

A SELECTION. This is the most recent shift-selection made in *any* TEdit window.

TEDIT.MOVESELECTION [Variable]

A SELECTION. This is the most recent control-shift-selection made in *any* TEdit window.

TEDIT.READTABLE [Variable]

A readtable; this is used to translate typed-in characters into TEdit commands. See the section on TEdit readtables. This can be overridden using the READTABLE property argument to TEDIT.

TEDIT.WORDBOUND.READTABLE [Variable]

The readtable that controls TEdit's concept of word boundaries. The syntax classes in this table also determine which characters TEdit thinks are white space (which gets deleted by CONTROL-W along with the preceding word). This can be overridden using the BOUNDTABLE property argument to TEDIT.

TEDIT.DEFAULT.PROPS [Variable]

A default set of PROPS arguments for TEDIT or OPENTEXTSTREAM. Any PROPS the user specifies are appended to the front of a copy of the default. The effect is that any user specifications override the defaults.

TEDIT.DEFAULT.FOLIO.LOOKS [Variable]

The default page number font. You can set this to be any character-looks specification acceptable to TEDIT.LOOKS. The default (i.e., if you don't specify one in the page layout menu) is taken from there. The default page number font is Modern 10 MRR.

Using TEdit's Terminal Table and Readtables

When TEdit reads a character from the keyboard, the first thing it does is check to see if it's a command character. TEdit first looks at its default readtable, TEDIT.READTABLE, or at the readtable supplied as the READTABLE property.

Failing that, TEdit then looks to the system terminal table. Characters with terminal syntax-classes CHARDELETE, WORDDELETE, or LINEDELETE act as follows:

CHARDELETE Acts as a character-backspace.

WORDDELETE Acts like CONTROL-W (in fact, this is how CONTROL-W is implemented).

LINEDELETE Acts like DEL.

Since the system terminal table is used to implement these functions, you can assign them to other keys at will.

Failing that, TEdit inserts the character at the current insertion point in the document.

The TEdit default readtable is named TEDIT.READTABLE, and it is global. You can use the functions TEDIT.SETSYNTAX and TEDIT.GETSYNTAX to read it and make changes:

(TEDIT.SETSYNTAX *CHAR CLASS TABLE*) [Function]

Sets the readtable syntax of the character whose charcode is *CHAR* to be *CLASS* in the readtable *TABLE*. The possible syntax classes are listed below. *TABLE* may be a readtable, NIL (defaulting to TEDIT.READTABLE), or a TEXTOBJ or text stream, in which case the readtable being used by that TEXTOBJ/stream will be used.

(TEDIT.GETSYNTAX *CH TABLE*) [Function]

Returns the TEdit syntax class of the character whose charcode is *CH*, according to the readtable *TABLE*. The possible syntax classes are listed below. An illegal syntax will be returned as NIL.

The allowable syntax classes are as follows:

CHARDELETE	A character with this syntax acts like backspace.
WORDDELETE	A character with this syntax acts like CONTROL-W.
DELETE	A character with this syntax acts like DEL.
UNDO	A character with this syntax causes an UNDO.
REDO	A character with this syntax acts like the AGAIN key.
NEXT	A character with this syntax acts like the 1108/1186 NEXT key.
FN	A character with this syntax calls a specified function (see below).
NONE or NIL	A character with this syntax simply inserts it in the document.

You can also cause a keystroke to invoke a function for you. To do so, use the function

(TEDIT.SETFUNCTION *CHARCODE FN RTBL*) [Function]

Sets up the TEdit readtable *RTBL* so that typing the character with charcode *CHARCODE* will APPLY *FN* to the text STREAM and the TEXTOBJ for the document being edited. The function may have arbitrary side effects. Setting a *FN* of NIL resets the character's syntax class to NONE.

The abbreviation feature described below is implemented using this function-call facility.

Finally, TEdit uses TEDIT.WORDBOUND.READTABLE to decide where word boundaries are. Whenever two adjacent characters have different syntax classes, there is a word boundary between them. The state of this table can be controlled by the following functions:

(TEDIT.WORDGET *CH TABLE*) [Function]

Returns the syntax class (a small integer) for a given character. *CH* may be either a character or a charcode; *TABLE* defaults to TEDIT.WORDBOUND.READTABLE.

(TEDIT.WORDSET *CHARCODE CLASS TABLE*) [Function]

Sets the syntax class for a character. Again, *CHARCODE* is either a character or a charcode; *TABLE* defaults to TEDIT.WORDBOUND.READTABLE; *CLASS* may be either a small integer as returned by TEDIT.WORDGET, or one of the atoms WHITESPACE, TEXT, or PUNCTUATION. Those represent the syntax classes in the default TEDIT.WORDBOUND.READTABLE.

The initial TEDIT.WORDBOUND.READTABLE assigns every character to one of the above classes, along pretty obvious lines. For purposes of CONTROL-W, white space between the caret and the word being deleted is also removed.

This, too, can be overridden for a specific edit session using the BOUNDTABLE property in the call to TEdit.

Using the TEdit Abbreviation Facility

TEDIT.ABBREVS [Variable]

A list of abbreviations known to TEdit. Each element of the list is a dotted pair (*Abbreviation* . *Expansion*). The *Abbreviation* must be a string (case does not matter). The *Expansion* may be either a string, a symbol, or a list. If it's a string, it is the expansion itself; if it's a symbol, it is then a name of a function to beAPPLY*'d to the text stream and the abbreviation string, and must return a string/charcode value that is the expansion. If the expansion supplied is a list, it will be EVALed and the result used.

To expand an abbreviation, select it and press the EXPAND key (or CONTROL-X on the 1132). It will be replaced by its expansion. If the abbreviation expander doesn't find an abbreviation as typed, it converts it to uppercase and tries again.

The default abbreviations and their expansions are listed in the first part of this document in the section "Producing Special Characters."

Using the TEdit IMAGEOBJ Interface

(TEDIT.INSERT.OBJECT *OBJECT STREAM CH#*) [Function]

Inserts the IMAGEOBJ *OBJECT* into the document *STREAM* at the place specified by *CH#*.

(TEDIT.OBJECT.CHANGED *STREAM OBJECT*) [Function]

Notifies TEdit that the IMAGEOBJ *OBJECT* has changed and the display should be updated. This is called by object-editing functions after they have updated the object's internal information. It is not intended to be called from within an object's WHENOPERATEDFN, which should return one of the values specified in the IMAGEOBJ documentation (see the *Interlisp-D Reference Manual)*. Will accept either a text stream or a text object as its *STREAM* argument.

Using TEdit's Interface for Reading Foreign File Formats

Whenever TEdit does a GET, it first checks to see if the file should be converted from some foreign format. This conversion process is controlled by the variable TEDIT.INPUT.FORMATS:

TEDIT.INPUT.FORMATS [Variable]

A list of two-element lists. Each sublist corresponds to a potential input format. The first element of the sublist is a predicate, which is applied to the open input file; if it returns a non-NIL result, conversion is performed. The second element of the list is applied to the open file, the result from the predicate, and a target TEXTSTREAM into which the converted text is to be placed. The function must return an open TEXTSTREAM containing the converted text. (This means that a conversion function *may* create its own text stream. However, if the conversion includes things like page formatting that are carried in the TEXTOBJ, it must use the TEXTSTREAM provided.)