Manipulating Text

The **Script Manager** supplies a variety of routines that help you draw, edit, format, and modify text. New routines available with system software version 7.0, identified by an asterisk (*), allow you to justify, substitute, and truncate text, perform case conversion, and strip diacritical marks.

Drawing and Editing Text

The **Char2Pixel**, **DrawJust**, **FindWord**, **HiliteText**, **MeasureJust**, and **Pixel2Char** routines help you draw and edit text. With system software version 7.0, the **Char2Pixel**, **DrawJust**, **MeasureJust**, and **Pixel2Char** routines handle intercharacter spacing in all scripts, if appropriate. In addition, there are new **NChar2Pixel**, **NDrawJust**, **NMeasureJust**, **NPixel2Char**, and **NPortionText** routines that also allow you to specify additional parameters to improve the handling of fully justified text. Also new with system version 7.0, the **NFindWord** procedure is a more powerful version of **FindWord** that lets you specify word boundaries for more than one script.

Char2Pixel	Finds the screen position of carets and selection points given a text buffer, an offset, and a slop value
NChar2Pixel*	Supplies a more powerful version of **Char2Pixel** that works with intercharacter spacing, lets you indicate the position of a style run within a line for lines with multiple style runs, and accepts scaling parameters
DrawJust	Draws the given text at the current pen location in the current font, style, and size, taking into account the slop value
NDrawJust*	Supplies a more powerful version of **DrawJust** that works with inter-character spacing, lets you indicate the position of a style run within a line for lines with multiple style runs, and accepts scaling parameters
FindWord	Returns two offsets in the array defined by the OffsetTable data type that specify the boundaries of the word defined by the offset parameter and the *leadingEdge* flag
NFindWord*	Supplies a faster and more powerful version of **FindWord** that can replace script-dependent versions of **FindWord** so that script systems must only supply appropriate tables (in the 'itl2' resource)
HiliteText	Finds the characters that should be highlighted between two offsets
MeasureJust	Given a slop value, a pointer to an array of characters, an integer indicating the number of characters in that text, and a pointer to an array of integers, fills each element in the array of integers with the width from the beginning of the string to the corresponding character in the array of characters (the supplied widths take the slop value into account-that is, they will be the widths

necessary to justify the text)

NMeasureJust* Supplies a more powerful version of **MeasureJust** that works with inter-character spacing, lets you indicate the position of a style run within a line for lines with multiple style runs, and accepts scaling parameters

Pixel2Char Finds the nearest character offset within a text buffer corresponding to a given pixel width, taking into account the slop value

NPixel2Char* Supplies a more powerful version of **Pixel2Char** that works with intercharacter spacing, lets you indicate the position of a style run within a line for lines with multiple style runs, and accepts scaling parameters

Formatting Text

The **FindScriptRun**, **PortionText**, **GetFormatOrder**, **StyledLineBreak**, and **VisibleLength** routines allow you to format text. The **NPortionText** function, available with system software version 7.0, provides a more powerful version of **PortionText** that lets you work with scaling and indicate the position of a style run within a line.

FindScriptRun Finds the next block of Roman or native text within a script run; within scripts that contain subscripts, blocks of native text are limited to a subscript

PortionText Indicates the correct proportion of justification to be allocated to given text when compared to other text; used to determine how to distribute the slop of a line among the style runs on the line

NPortionText * Supplies a new version of **PortionText** that lets you indicate the position of a style run within a line for lines with multiple style runs and accepts scaling parameters

GetFormatOrder Tells in what order format runs should be drawn based on line direction for a particular line of text

StyledLineBreak Breaks a line on a word boundary

VisibleLength Returns the length of text, excluding trailing white space and accounting for the script of the text

Modifying Text

The **LowerText**, **Transliterate**, and **UpperText** routines let you localize text into a base form and convert text from lowercase into uppercase, providing for the localizable stripping of diacritical marks. The **StripText** and **StripUpperText** procedures, available with system software version 7.0, provide localizable stripping of diacritical characters and conversion of the characters into uppercase. **LowerText** and **UpperText**, also new with version 7.0, are faster and easier to use than **Transliterate**, but less powerful.

LowerText*	Provides localizable lowercasing of text up to 32 KB in length (LwrText is a synonym for LowerText**)**
StripText*	Provides localizable stripping of diacritical characters for text up to 32 KB in length
StripUpperText*	Provides localizable stripping of diacritical characters for text up to 32 KB in length and converts them to uppercase characters
Transliterate	Converts characters from a set of scripts or subscripts to the closest possible approximation in a different script or subscript, and performs localizable uppercasing and lowercasing
UpperText*	Provides localizable uppercasing for text up to 32 KB in length

Note: **UpperText** is different from **UprText**, which provides nonlocalizable upper-casing using the _UprString trap.

Substituting Text

The **ReplaceText** function, new with system software version 7.0, allows you to substitute text correctly for all scripts. It provides a global way for you to do parameter text replacement in your applications. **ReplaceText** searches specified text for instances of a string specified by the key parameter and replaces each instance with the replacement text supplied.

Truncating Text

The **TruncString** and **TruncText** functions, new with system software version 7.0, let you truncate text at the end or in the middle in order to fit it into a specified pixel width.

TruncString*	Ensures that a string supplied as Str255 fits into the specified pixel width by truncating the string, if necessary, in a manner dependent on the script associated with the font of the active grafPort.
TruncText*	Is similar to **TruncString** except that the string is defined by a pointer and a length

Lexically Interpreting Different Scripts

The **IntlTokenize** function takes arbitrary text and breaks it into tokens like the lexical analyzer of a compiler. **IntlTokenize** allows a program to recognize tokens such as variables, symbols, and quoted literals without making assumptions that depend on a particular script.