Apple II
Technical Notes

Developer Technical Support

Apple IIGS
#92: Twisted Tales of TextEdit

Revised by: Dave Lyons
December 1991
Written by: C.K. Haun <TR> and Dave Lyons
September 1990

This Technical Note discusses some undocumented features and some bugs in the TextEdit tool set through System Software 5.0.4.
Changes since November 1990: Noted that a non-control `TENew` creates a Text Edit record for the current port.

TENew

TextEdit records you create with `TENew` are always tied to the current port at the time of the `TENew` call, whether or not the `fNotControl` bit is set. (For TextEdit controls, `NewControl2` is the preferred call.)

TEInsert

Using the `TEInsert` call on an invisible TextEdit record causes the screen to scroll, exactly as if the TextEdit record were visible.

If you use `LETextBox2` style text as input for a `TEInsert` call, any style change information contained at the end of the `LETextBox2` text is ignored. To ensure that the style change is not ignored, append an additional character at the end of the block, then delete (with `TESetSelect` and `TEDelete`) the extra character after the `TEInsert` call.

TEGetText

The documentation for `TEGetText` says that a `dataFormat` value of $4 returns the text as "Formatted for input to LineEdit `LETextBox2`". This is not a reliable return method—this call may or may not succeed. Greater chance for success occurs with less than 4,000 characters in the TextEdit record.

`TEGetText` also supports getting just the text of the current selection range. Adding `$0020` (`onlyGetSelection`) to the number passed in `bufferDescriptor` returns the text of the current selection. This technique does **not** work with data format `LETextBox2`, but does work with all other formats. Also, there is no corresponding bit for the associated style record, so you cannot get the style for just the current selection this way, if you request style information you get a `styleRef` for the entire TextEdit record.

TEClick

Using `TEClick` or `TestControl` on an inactive record currently causes that record to activate.

TERuler

Pixel tabbing values must all be greater than zero or TextEdit loops infinitely on a tab.

TEGetRuler & TESetRuler

`TERuler`, for the default ruler or any ruler that uses a `tabType` value of $1 returns a ruler four bytes longer than described in the documentation. The extra four bytes are all $FF, and they are the terminator characters for `tabType` $2 rulers. Expand your buffers by four bytes to prevent overwriting any data. TextEdit also expects the additional information on a `TESetRuler` call, so you should pad your ruler with four $FF bytes if you are using a type $1 ruler.

TESetText

Passing a zero-length class one input string (a word length string with the word set to zero) to `TESetText` causes TextEdit to crash.

TEPaintText

`TEPaintText` currently prints colored text in only four colors.

It's Not Dirty, It's Text

There has been some confusion about determining if a TextEdit record has been changed. The documentation has been a little vague, and the process itself has mislead some people. Here is The Truth: there **is** a TextEdit dirty flag, and you can use it and rely on it to tell you when a TextEdit record has changed.

The TextEdit dirty flag is bit 6 (`fRecordDirty` in the E16.TextEdit interface file) of the `ctlFlag` byte. This has caused some confusion because the `ctlFlag` byte is at offset `$12` in the control definition **template**, and it is at offset `$10` in the TextEdit or Control **record**. Just remember that it is **not** in the same place in the record as it is in the template.

If it is set, then the TextEdit or Control record has been changed since the last time the dirty bit was cleared. The dirty bit is clear initially when you create the TextEdit or Control record. Anytime after that, if the user enters text into the TextEdit record, TextEdit sets the dirty flag. It is up to your application to clear the dirty flag; TextEdit has no way of knowing when you've saved or cleared data.

Further Reference

- *Apple IIGS Toolbox Reference*, Volume 3