New Technical Notes

Macintosh

Developer Support

TextEdit Bugs in System 4.2

Text **M.TE.TextEditBugs**

Revised by: March 1988
Written by: Chris Derossi June 1987

This note formerly described the known bugs with the version of Styled TextEdit that was provided with System 4.1. Many of these bugs were fixed in System 4.2. This updated Technical Note describes the remaining known problems.

TEStylInsert

Calling `TEStylInsert` while the TextEdit record is deactivated causes unpredictable results, so make sure to only call `TEStylInsert` when the TextEdit record is active.

TESetStyle

When using the `doFace` mode with `TESetStyle`, the style that you pass as a parameter is `ORed` into the style of the currently selected text. If you pass the empty set (no styles) though, `TESetStyle` is supposed to remove all styles from the selected text. But `TESetStyle` checks an entire word instead of just the high-order byte of the `tsFace` field. The style information is contained completely in the high-order byte, and the low-order byte may contain garbage.

If the low-order byte isn't zero, `TESetStyle` thinks that the `tsFace` field isn't empty, so it goes ahead and ORs it with the selected text's style. Since the actual style portion of the `tsFace` field is zero, no change occurs with the text. If you want to have `TESetStyle` remove all styles from the text, you can explicitly set the `tsFace` field to zero like this:

```
VAR
   myStyle  : TextStyle;
   anIntPtr : ^Integer;

BEGIN
   ...
   anIntPtr := @myStyle.tsFace;
   anIntPtr^ := 0;
   TESetStyle(doFace, myStyle, TRUE, textH);
   ...
```

```
END;
```

TEStylNew

The line heights array does not get initialized when `TEStylNew` is called. Because of this, the caret is initially drawn in a random height. This is easily solved by calling `TECalText` immediately after calling `TEStylNew`. Extra calls to `TECalText` don't hurt anything anyway, so this will be compatible with future Systems.

An extra character run is placed at the beginning of the text which corresponds to the font, size, and style which were in the grafPort when `TEStylNew` was called. This can cause the line height for the first line to be too large. To avoid this, call `TextSize` with the desired text size before calling `TEStylNew`. If the text's style information cannot be determined in advance, then call `TextSize` with a small value (like 9) before calling `TEStylNew`.

TEScroll

The bug documented in M.TE.TEScrollBug remains in the new TextEdit. `TEScroll` called with zero for both vertical and horizontal displacements causes the insertion point to disappear. The workaround is the same as before; check to make sure that `dV` and `dH` are not both zero before calling `TEScroll`.

Growing TextEdit Record

TextEdit is supposed to dynamically grow and shrink the `LineStarts` array in the `TERec` so that it has one entry per line. Instead, when lines are added, TextEdit expands the array without first checking to see if it's already big enough. In addition, TextEdit never reduces the size of this array.

Because of this, the longer a particular TextEdit record is used, the larger it will get. This can be particularly nasty in programs that use a single `TERec` for many operations during the program's execution.

Restoring Saved TextEdit Records

Applications have used a technique for saving and restoring styled text which involves saving the contents of all of the TextEdit record handles. When restoring, `TEStylNew` is called and the TextEdit record's handles are disposed. The saved handles are then loaded and put into the TextEdit record. This technique should not be used for the `nullStyle` handle in the style record.

Instead, when `TEStylNew` is called, the `nullStyle` handle from the style record should be copied into the saved style record. This will ensure that the fields in the null-style record point to valid data.