Version 3.2.1 June 4, 2019

Change List:

BUG FIXES:

* Fixed issue where all StyleRuns after a hyperlink will also be marked as hyperlink

* Issue #4074: RTF clipboard ignores hyperlinks with umlauts and cuts them

* Issue #4087: missing RTF space after "\kerning0" keyword

* Issue #4130: FTXojoScriptField: Left part of the text is hidden by the section of the line number.

* Issue #4135: Migrating texts from a TextArea to the FTC (Ich migrieren Texte aus einer TextArea in das FTC)

* Issue #4196: Remove GOTO calls from FTC

Version 3.2.0 February 12, 2019

Change List:

NEW ITEMS:

* Contains code from the ImagePlay Effects Library - http://imageplay.sourceforge.net

* Tab Leaders implemented

* Issue #3832: Implement Control + Up/Down Arrow Key Handling

* Issue #3833: Implement paragraph moving via keyboard handling

* Issue #3700: Drop Shadow available on Windows and Linux

* Issue #3795: The FTDocument properties characterStyles and paragraphStyles need to be accessible to subclasses.

Added CharacterStyles and ParagraphStyles getter/setters so developers can create their own style editor.

* Character Spacing implemented (requires Xojo >= 2015.4)

BUG FIXES:

* Issue #3650: CustomObjects Demo: Added FTFile.Clone method so it works with FTIterator propertly.

* Issue #3653: macOS: "getDoubleClickTime" - Cocoa replacement for old CarbonLib call

* Issue #3699: Add Win64 Declares

* Issue #3654: Hi-DPI Mode: FTPicture - Handles drawn incorrect and matches wrong.

* Issue #3702: DragRect is half height in HiDPI

* Fixed an issue with FTParagraph clones that have a different ID after going through the FTIterator

* Issue #2007/3605: Candidate Window Shows up in Wrong Location

* Issue #3646: FTParagraph.getXML saves wrong TabStop Properties

* Issue #3709: FTPicture ignores Nudge property

* Issue #3748: FTDocument.selectionBold,Italic, shadow, strikethrough, subscript, superscript, underline now works properly

* Issue #3752: FTRBScript: Fix for printing line numbers

* Issue #3763: RTFReader/FtcRtfReader ignored the left/right margin and first indent of paragraphs.

* Issue #3764: RTFReader ignored "\sb" SpaceBefore of paragraphs.

* Win64 Issue: Modified FTCUndoPaste.constructor and FTCUndoManager.savePaste to allow pasting (Xojo Compiler Issue. Feedback 53967)

* Issue #3750: Suggestion & comparison of the Office applications for the display optimization of the paragraph background color

* Issue #3775: Implement Corner Color Property in Formatted Text (Switch between Light/DarkMode - Xojo Version >= 2018.3)

* Issue #3777: Implement Page Shadow Property in Formatted Text (Switch between Light/DarkMode - Xojo Version >= 2018.3)

* Issue #3755: Printing has the side-effect of scrolling the editing window to the top

* Issue #3794: FTDocument.getSelectedParagraphs returns first paragraph of multi-paragraph doc regardless of insertion point

* Issue #3796: FTStyleRun.copyStyleData does not copy background color

* Issue #3820: RTFWriter missed "\par" after the last Paragraph

* Issue #3774: FormattedText — changeParagraphMargins method name changeParagraphMarigns

* Issue #3762: FTDocument.selectionFontColor returns sameColor false when it should be true

* Issue #3842: FTStyleRun.updateWith Style sets textColor black when undefined

* Issue #3806: FTDocument.addParagraphStyle and addCharacterStyle bypassed by insertXML and cannot be overridden in subclass

* Issue #3818: Print RB Code Editor: Unwanted positive y-offset of the content compared to its line number

* Issue #3637: FTCParagraphStyle.backgroundColor won't set correctly

* Issue #3910: RTF-Reader ignores local value for first line indent in paragraphs

* Issue #3914: First Paragraph on Page with Background Color ignores Before Space

* Issue #3917: Paragraphs Background Color ignores Hanging Indent

CHANGES:

* Renamed FormattedText.xDisplayPosition to FormattedText.hScrollValue to better reflect what it is.

* Renamed FormattedText.lastXDisplayPosition to FormattedText.lastHScrollValue

* Renamed FormattedText.yDisplayPosition to FormattedText.vScrollValue to better reflect what it is.

* Renamed FormattedText.lastYDisplayPosition to FormattedText.lastVScrollValue

* Renamed FormattedText.GetResolution to FormattedText.GetMonitorPixelsPerInch to better reflect what it does.

* Renamed FTDocument.SetResolution to FTDocument.setMonitorPixelsPerInch

* Renamed FTDocument.GetResolution to FTDocument.getMonitorPixelsPerInch

* Renamed FTDocument.Resolution to FTDocument.MonitorPixelsPerInch

* FormattedText.DISPLAY_ALIGN_CENTER/LEFT/RIGHT replaced with Display_Alignment enum

* FormattedText.MODE_PAGE/NORMAL/EDIT/Single replaced with Display_Mode enum

* FTLine.FIRST_LINE/MIDDLE/LAST/BOTH replaced with Line_Type enum

* FTPicture.Handle constants converted to FTPicture.Handle_Type enum

* FTParagraph.Alignment constants converted to FTParagraph.Alignment_Type enum

* Removed Alignment constants from RTFConstants module

* Changed how hyperlinks are drawn in FTObject.drawHyperLink

* Changed how the gradient shadow is drawn in FTPage.drawPageViewMode

* Changed how strikethroughs are drawn in FTObject.drawStrikethrough

* Changed how underline is drawing in FTObject.drawUnderline

* Can now read/write shadow properties from/to RTF Documents

* Added AcceptFileDrop to test window for testing custom objects. Added FTFile into project.

* Changed how Styles are saved and read in XML format.

* Issue #3802/1456: Command + (Shift Key) + Arrow Keys doesn't move insertion point correctly

* Changed RB Script classes to Xojo Script.

* Now have TabLeaders (WORK IN PROGRESS)

* Embedded FTC demo now does text formatting with keyboard shortcuts.

* Removed unused FTDocument.DocOffset method.

* Fixed TargetCocoa and TargetWin32 constants with more appropriate ones for 2018 R4.

* Removed duplicated code line in RTFReader.SetupKnownCommands

* Updated ParagraphWindow to easy format First Line and Hanging Indents

PARTIAL IMPLEMENATION:

* Issue #3753: PageBreaks won't load from RTF

*

Version 3.1.9 October 30, 2017

Changes:

- Fixed issue with view scale < 1.0 setting up the clipping page incorrectly (#3500)
- Fixed spelling replace on TestWindow (#3581)
- Fixed Mac text AutoComplete positioning (#3574)
- Fixed Tabstop Widths (#3500, #3576)
- Updated Linux declare for doubleClickTime to include GTK3 in newer Xojo versions (#3575)
- Added SpellCheck configuration testing to contextual menu on TestWindow

- Added SpellCheck configuration testing to App.Close event
- Fixed resources directory location for the the Spell Checker initialization

Version 3.1.8 March 22, 2017

Changes:

- Fixed issue with getText returning the character before the selection start on occasion
- Added FTDocument.AppendXML helper method
- Added FTDocument.Speak helper method.
- Selections style queries now default to false when selecting the space between paragraphs
- Styling reset for \pard (specific to Microsoft generated RTF)
- Creating and saving a document as xml the reloading it and continuing to edit may drop style runs that initially are correctly set
- Fixed issue in Page Layout mode where white area of page displayed at twice required width
- Removed Regression testing suite since they never worked in Xojo and tries to update UI from within a thread

Version 3.1.7

Changes:

- Fixed issues with Unicode Hyperlinks
- Fixed Windows Compiler Issue
- Fixed HIDPT coordinates issue issue with ConstructContextualMenu event handler
- Added ability to detect Windows HiDPI (requires Xojo 2016 R2 or better.
- TextInputCanvas plugin rebuilt to be 64 bit compatible.
- Spell Checker demo plugin rebuilt to be 64 bit compatible.

Version 3.1.6

Changes:

- Updated printing to that is works properly on Windows and Mac OS X - particularly with the number of copies
- Updated copyright information in many of the comments

- Changed some properties so they don't conflict with MacOSLib
- Added OSVersionInfo module to replace System.Gestalt which is now deprecated
- Fixed a couple of RTFReader issues
- Minor code cleanup

Version 3.1.5

Changes:

- The TextAreaRTF.GetSelectedRTF function now returns the last StyleRun.
- RTFLexer now reads upper and lowercase hexadecimal values properly now.
- Fixed Mac OS X issue where the Candidate Window didn't show properly if the cursor was at the end of the last paragraph.
- Fixed a couple of deprecated items.
- Prepared for Xojo 2014 R3

Version 3.1.4

Changes:

- Added DRAGTIMEOUT to have a timeout so that when a text block was selected there it is no longer draggable. It will change the insertion point again. Fixes an issue where if user did a Select All on the text the mouse would always and forevermore be in drag mode.
- Made an adjustment in RTFReader.Parse method. Commented out the EOF and Token Error case statements since there's nothing we can do about them at that point.
- Fixed an error in the RTFReader that would cause the 2nd use of the same Font to not work.
- Fixed TextAreaRTF functions that would cause exception when there was only one paragraph
- Added new TextAreaRTF example in the main demo.

Version 3.1.3

Changes:

- Add support for the new BKS Spell Checker Plugin
- Changed Spell Checking Demo

- Changed the main demo to use BKS Spell Checker Plugin
- Fixed an issue with Retina display
- Fixed an issue causing an out of bounds error in the FTParagraph
- Fixed some issues with the RTF Reader
- Fixed some issues with the Embedded FTC

Version 3.1.2 beta 3

Changes:

- Uses the new RTF Lexer for the RTF Reader class
- Fixed a hard crash issue when running in Linux (still unsupported however)
- Fixed error that kept text from being processed

Version 3.1.2

Changes:

- The Candidate Window (Cocoa only) now displays in the proper location.

Version 3.1.1

Changes:

- Text handling more complete with special operations from the Text Input Canvas plugin
- Changed Embedded demo to work properly with Text Input Canvas
- Fixed Embedded demo to eliminate flicker in Windows
- Fixed double character text input in Windows
- Tab handling is now properly setup on initialization
- Now handle Forward Delete properly
- HTML export now handles strikethrough
- Embedded FTC now works properly in Cocoa

Version 3.1.0 alpha 1

Changes:

- Save in Xojo binary format.

- Requires the use of the Text Input Canvas plugin from Xojo Inc.
- Will not compile for carbon. Period.
- Uses a new event structure for Cocoa/Windows (no Linux support)

Version 3.0.1

Bug Fixes:

- Fixed Nil Object Exception on Mac's running 10.6.8
- Fixed out of bounds exception when adding a tab
- Added check of NIL in FTDocument
- Slight change in FTCustom Object Demo
- Change in documentation on how to report bugs and link to Mantis

Version 3.0.0

Known Issues

- Cocoa accented text is not working. This is a Real Software issue that will be addressed when they include support for Cocoa text handling for Canvas objects. This is scheduled for Real Studio 2013 Release 1.

New Features and Enhancements:

- Added Hyperlink Support
- Added Retina Display Support (Cocoa Only and when the app plist is configured properly)
- Added Autocomplete Support
- Added Text Shadow Support (Contributed by Paul Levine)
- Added Text Opacity Support (Contributed by Paul Levine)
- Added FTCustom double click event in parent control
- Greatly improved HTML Export
- Rewritten RTF Reader
- Rewritten Undo Manager
- Removed Spell Check Utilities Plugin requirement. Look at the FTConfiguration constant SSCE_SPELLCHECK_ENABLED

- Removed FTCharts
- New, smaller demo projects
- Added MakeTextBigger/MakeTextSmaller functions
- FTIterator class now passes FTCustom objects (if any) instead of the FTPicture base class

Bug Fixes:

- Redo capacity no longer set to one
- Tab support is now automatic (Contributed by Marco Bambini)
- Fixed the Date Mask in the FTEdit Field Demo
- Drag text locations now work properly
- Fixed Windows printing bug for Real Studio versions less than 2012 R1
- Better Cocoa support
- DragItem now uses TrueWindow in case the parent window is a Container
- Shift Down Arrow can now select the last line of the document
- Arrow keys now fire the ChangeInsertionPoint event
- Fixed drawing issue with cursor drawing in the wrong spot in the first paragraph when there's a Space Before Paragraph
- Many others!

Version 2.4.3

Bug Fixes:

- Fixed an Illegal Cast Exception in FTRBScriptField.colorCodeLine (Contributed by Aaron Andrew Hunt)

Version 2.4.2

Bug Fixes:

- Fixed insertion point problem when changing lines.

- Made the MouseDown click handling a bit more efficient by eliminating duplicate code.
- Removed direct manipulation of the Graphics object.
- Added support for Real Studio 2012 R1.
- Modified a bit of Regression testing code to make it more like BKeeney coding style.

Version 2.4.1

Bug Fixes:

- Fixed a drag and drop problem with the first click on a selection.
- Fixed a problem with the wrong picture size being written out to the RTF stream for scaled pictures.

Version 2.4

New Features and Enhancements:

- Added the methods changeToLowerCase, changeToUpperCase, changeToTitleCase to allow you change the case of selected text. These method support undo operations. You can also call them at the FTDocument level.
- Updated the RTFParser code to the latest version 1.0.3.
- Added support for RS2011R4.

Bug Fixes:

- Fixed a problem inserting plain text where tab characters were ignored. Tabs are now properly inserted into the text.
- Fixed a problem with shift-select not properly selecting all the intended characters.
- Fixed a couple of more problems with shift-arrow up and down selections.
- Fixed scrollbar placement for Lion.

Version 2.3

New Features and Enhancements:

- Added support for the AltGr key on foreign keyboards.
- Optimized the selection algorithms so that they now constant in their execution time no matter how long the selection.

- Made selection more responsive when doing a mouse drag.
- Added the method FTDocument.insertParagraph that allows you to directly insert paragraphs from one FTC to another. It will clone the paragraph, change the parent and then insert it into the new control.

Bug Fixes:

- Fixed a problem when inserting text when the current paragraph contains a page break and the page break is carried forward to each subsequent paragraphs in the inserted text.
- Fixed a condition where selecting a picture and trying to apply strikethrough, mark or underline would not apply the attribute.
- Fixed a problem with saving pictures in RTF where the strikethrough, underline and text background color commands were not written to the RTF stream.
- Fixed a problem where a picture that has a background color assigned to it would not properly display it when in resize mode.
- When the RTF "\plain" command was processed, it incorrectly reset the unicode skip count to 1. This has been fixed.

Version 2.2

New Features and Enhancements:

- Removed the requirement for graphic conversion plugins from MBS or Einhugur. The FTC now uses recently added RS framework methods to accomplish task and therefore the minimum RS IDE is 2010R3.
- Added support for the RTF tags bullet, emdash, endash, emspace, enspace, qmspace.
- Added a method to FTPage called getPageLineCount that will tell you how many lines are on the page.
- Added two design time properties called VScrollbarAutoHide and HScrollbarAutoHide. These properties will cause the scrollbar to hide when the length of the content can be fully displayed without scrolling.

Bug Fixes:

- Fixed a problem with parsing RTF on Windows with non-existent text encoding in RB encoding. It now just returns a space.
- Updated the demo project to be compatible with RS2010R5.
- Updated the SCU code within the demo to the latest version.

- Changed the depth size in scratch graphics in the FTUtilites module to 32 bits for compatibility with Cocoa.
- Fixed a problem in the RTF parser under Windows where the RS framework wasn't returning an encoding for charset 0. An encoding is now returned which will allow a lot of non-ascii character to be properly shown.
- Fixed a problem in the RTF parser where "/" was appended to the output text.
- Fixed a problem where pictures weren't retaining user changed size settings when dragged or saved.
- Removed configuration constants USE_EINHUGUR_MBS_CONVERTERS, USE_EINHUGUR_PLUGINS and USE_MBS_PLUGINS.
- Fixed a problem with a click on picture which would not be handled when the display is in center alignment.

Version 2.1

New Features and Enhancements:

- A new chart infrastructure based off the FTCustom class has been created. The first chart based on this is a vertical bar chart called FTVerticalBarChart. In the future more chart types and variations will be added.
- Added the classes FTIterator and FTIteratorBase. These two classes allow you to easily walk through the FTC paragraphs and style runs. In the demo application this is used to generate some basic HTML output (see the menu item "File->Export->HTML"). The class FTInterator takes care of the mechanics of walking through the data structures and FTIteratorBase is the base class you use to create your own handlers. The FTInterator class will call the handlers in your subclass.
- The FTRBScriptField can now soft wrap and change wrapping mode on the fly.
- Added the event DrawLine. This is called when a composed line (FTLine) is drawn. You can use this to line specific drawing like line numbers.
- Added convenience function getText to the FTTextField.
- The events PrePageDraw and PostPageDraw are now called in all modes.
- Improved the scrolling performance when dragging a selection past the end of the control.
- Implemented some minor speed optimizations.
- Added the constants MODE_PAGE, MODE_NORMAL, MODE_EDIT and MODE_SINGLE for setting the mode property.

Bug Fixes:

- Fixed a problem with certain combinations of backspacing and then typing characters which were then inserted in the wrong spot.
- Fixed a problem with showing the insertion caret when it when it need scrolled into view.
- Refined the behavior of the FTTextField to handle continuous pressing of the arrow keys and scroll properly.
- Fixed a problem with the scroll to caret functionality where it tried to scroll to the caret horizontally in edit mode and that doesn't make sense. This fixes the problem with the embedded FTC slightly shifting unpredictably in the display.
- The RTF parser under Windows would crash when a field instruction (a specialized way of representing a character) was encountered. It now just returns a space when this happens.
- In version 2.0 the format for encoding pictures in the XML output was changed to optimize for space. This means pictures read in from the 1.x series would not be decoded correctly. The FTC now detects this difference and decodes the picture accordingly. This gives backwards compatibility with 1.x. Note, the FTC will encode the picture in the 2.x format and that format is not compatible with 1.x releases.
- Fixed a problem with clicking in a word that is marked as misspelled and clicking outside of the word and the original word is marked spelled correctly.

Version 2.0

New Features and Enhancements:

- You can now change the alignment of the display where the page is oriented in the main display (left, center, right) for the page and normal modes.
- Implemented search and replace functionality. You can search and replace interactively or programatically. See the "SearchWindow" window for an example of how to implement interactive search and replace it in your application.

  ***search**(target as string, searchStart as FTInsertionOffset, searchEnd as FTInsertionOffset, ByRef selectStart as FTInsertionOffset, ByRef selectEnd as FTInsertionOffset) as boolean*

  Search allows you to specify the starting and ending locations for the search. If you pass in nil for searchStart or searchEnd then the beginning or end of the document will be used respectively.

  scrollToSelection

  The scrollToSelection method allows you to make sure the current selection is visible in the display.

To programmatically replace all occurrences of a string with another string, use the following method.

replaceAll*(target as string, replacement as string, searchStart as FTInsertionOffset, searchEnd as FTInsertionOffset)*

The same behavior applies to the searchStart and searchEnd in the replaceAll method as does in the search method.

- Created a new control called FTTextField which is the equivalent of the RB TextField except that it has all the advantages of the FTC like cross platform spell checking, undo capabilities and many more. As part of the control, implemented the LimitText property that limits the number of characters that can be typed into the control. Also implemented a mask for the control. Shown below are the symbols you can use in a mask. Note, the mask is case insensitive where an "a" is the equivalent of an "A".

  # - The single digit placeholder (0-9).

  A - Alphabetic (a-z A-Z).

  B - Alphanumeric (a-z A-Z 0-9).

  C - Any printable character.

  [ ] - Character set. Allows you to define a set of characters for that position. For example "[-+]" allows you to have a plus or minus sign for that particular character position.

  . - Decimal placeholder. Derived from the international settings.

  , - Thousands separator. Derived from the international settings.

  / - Date separator. Derived from the international settings.

  **** - Mask escape character. Used to specify literal characters that conflict with reserved mask characters.

  > - Convert all the characters that follow to uppercase.

  < - Convert all the characters that follow to lowercase.

  All other symbols are displayed as literals for formatting purposes.

- Added a new KeyPress event to allow you to change the actual text that will be inserted when a user types on the key board. Extraneous keys like arrow keys are filtered out before this event is called.

- Implemented some minor speed optimizations.

Bug Fixes:

- Fixed the getRTF method to ignore non-existent character styles references.
- Fixed a problem with changing the style (bold, italic, underline, etc.) and once turned on could not be turned off with the change methods. Also fixed a related problem where the displayed selection was incorrect after repeated style changes.
- Fixed a cosmetic problem with text background color not matching the selection width and height when they cover the same text.
- Fixed a problem with horizontal scrolling in single view mode where there would be a slight telescoping of the line in relation to the scroll position.
- Fixed a problem with inaccurate insertion caret position calculation to determine which paragraph and line the caret should be drawn.
- Fixed a problem with pasting text that has multiple style elements to the end of a paragraph. The individual style elements were pasted out of order. They now are pasted in the correct order.
- Fixed a problem with undoing deletions crossing paragraph boundaries where the previous selections wasn't properly deselected.
- Fixed a problem with FTPicture hex encoding where it would cause a crash when used with RBScript (in this case regression testing of the FTC).
- The insertion caret is now drawn through the descent of the line (it is now longer).
- Made some cosmetic display tweaks to the selection and background drawing.
- Fixed a problem with dragging an internal selection to past the end of a paragraph where the selection was offset after the drag.
- Fixed the handling of "\\<return>" sequence in an RTF stream.
- Fixed the hasCharacterStyle method to search for the style name instead of comparing against the index.

Version 1.8.3

Bug Fixes:

- Fixed a problem with the RTF parser not properly handling returns in the input stream under certain circumstances.
- Fixed an obscure undo bug that affected undoing disjointed typed segments.
- Fixed the changing of style attributes of empty paragraphs so that it does all the of them.
- Fixed a problem with empty styles drawing erroneous marks and text background colors.

- Fixed a problem with internal segments not being valid which in turn caused nil object exceptions under certain conditions.
- Fixed a problem with changing style characteristics on a single selected empty paragraph.
- Fixed a problem with deleting selected text where a pilcrow is the first selected item and would cause the insertion caret to be off by one and affect undo.

Version 1.8.2

New Features and Enhancements:

- Optimized the RTF parser to be 30% faster!
- Added support for the "plain" tag in the RTF parser.
- Added support in the RTF parser for symbols that are encoded in the field instruction tag.
- Added an event called PasteAction to the FormattedText class. This event will allow you to handle the paste event. For example, perhaps you want to strip off any formatting and just paste plain text. If you return false from this event, the built in functionality will be done. If you return true, it means you handled it. In either case the undo functionality is handled for you.

Bug Fixes:

- Updated version handling in the demo project.
- For custom objects, the GotFocus and LostFocus events were called multiple times when clicking on the object with the mouse. The focus events are now only called when they should be called.
- Fixed a problem where triple clicking on a picture selects an entire paragraph but left the picture in resize mode.
- Fixed a problem with the selectionLineSpacing method returning incorrect results.
- Calling addToSelection when nothing selected caused a crash. It now uses the current insertion point when nothing is selected.
- Fixed a problem with escaped character in the RTF stream messing up internal encodings.
- Added a check in the insertion offset code to prevent out of bounds insertion offsets for empty paragraphs.
- Fixed a problem with the RTF parser not stripping out comment groups and processing the commands within those groups.

- Rewrote the font parsing section of the RTF parser to be more robust.
- Plugged a hole in the RTF font parsing where a nil object exception occurred.
- Fixed a problem with getting the selected text as plain text where the pilcrow is selected and it returned an empty string. It now return the selected text.
- Fixed a problem with the RTF parser not retaining the current style settings when adding empty paragraphs.
- Fixed a problem with the generation of RTF for empty paragraphs so they retain their basic style attributes.
- Fixed a problem with the undo stack not being cleared when you call the FormattedText class methods setText, setRTF and setXML.

Version 1.8.1

Bug Fixes:

- Fixed a problem with the FTRBScriptField calling a general utilities method and thus causing an unintended coupling.
- Under Windows, fixed a problem with not being able to drag a picture within the control.

Version 1.8

New Features and Enhancements:

- Added support for pilcrows (the backwards P character at the end of a paragraph). Also added the ShowPilcrows flag to allow you control the appearance of the pilcrow character. If this is set to false, it will still display the selected background of the pilcrow. This is useful for specialized editors.
- Added line numbers to the FTRBScriptField. You can manipulate the display and drawing characteristics at design time and programatically.
- Implemented a major speed up in the FTRBScriptField with indenting lines.
- Added lookupErrorCode to the FTRBScriptField that will look up the error code produced when compiling the script and give you the english version of the error for display.
- Add support for the RTF command \txN (tab stops).
- Added tab stop support into the XML output.
- Changed the behavior when you specify specific tab stops, the rest of the line will be filled in with default the tab stops.
- Added the insertTab convenience method to make it easier to programatically add tabs.

- When you show invisibles it now includes a dot for tabs and pilcrows characters.
- Added InvisiblesColor property to allow you to change the colors of invisible characters.
- Made horizontal and vertical scrolling via mouse drags more responsive.
- Added a property called "associatedFolderItem" to FTPicure that allows you to associate a FolderItem with the picture. This will be automatically populated when you drop a picture file on the control.
- Added the method insertCustomObject to allow you to insert and handle the undo chores in one call when you want to insert a custom object. See the insert Hyperlink menu item in the demo for an example.
- Implemented automatic regression testing facility. To run the regression tests, you need the MBS plugin "MBS REALbasic ComputerControl Plugin.rbx." Also added the constant FTC_REGRESSION_TEST to control the inclusion of the MBS methods. You must set this to true to in order to run the tests.
- Added a "VERSION" constant to the FormattedText class to show the build version of the FTC.
- Added some compatibility code for versions prior to RB2010r1.

Bug Fixes:

- Updated deprecated items that generated warnings with RB2010r1.
- Added the method isValidParagraphMargins that allows you to test to see if the specified margins are valid. The changeParagraphMarigns methods have also been enhanced to call this validity check and do nothing if it fails the check.
- Fixed a problem with inserting page breaks at the end of a paragraph and the display isn't updated properly.
- Fixed a problem with repagination when deleting paragraphs.
- Fixed a problem with the style run width cache that caused occasional wrong width values to be returned. This would show up when you were selecting something.
- Fixed a problem with tabs causing exceptions when changing the the misspelled state.
- Fixed a problem with RTF parsing where character attributes were improperly reset when when the “pard” tag was processed.
- Fixed a problem with parsing tabs where double tabs would be inserted.
- Fixed a problem with the wrong style being confused with the default style when loading in RTF.
- Fixed a problem with tab widths when printing.

- Converted all the threads in the FTC to timers and did some optimization on when they are called. You should notice no difference in behavior from previous versions of the FTC. This fixes the problem with the REALServer barfing on threads that are suspended.
- Refined the selection behaviors.
- Fixed a problem with setting the insertion point between lines when the current insertion point is logically the same but resides on the opposite side of the line.
- Removed the RTF optimizer since it was causing problems.
- Fixed a problem with selecting an entire paragraph (the pilcrow was selected) and changing style characteristics.
- Fixed a problem with setting the insertion caret to the end of a paragraph and then typing a couple characters where the second character ends up in the next paragraph.
- Made the internal drag and drop of a selection more robust.
- In edit view mode, fixed a crash when setting the width of the control to zero and also fixed the composer to handle small control width situations.
- Fixed selecting to the end of the paragraph with the mouse when the mouse is on the right side of the paragraph.
- Fixed a problem when you type a tab at the end of line and then type some characters where the new character don't pick up the previous style run characteristics.
- Fixed various situations where typing characters at the end of the line will cause the character to end up on the beginning of the next line.
- Fixed a parser error in single line mode.
- Fixed a quirk with dropping files from the finder onto the control.
- Fixed a problem with aligning text after tabs and the display was scaled or when printing. The text should now accurately line up after a tab.
- Enhanced the dragging and dropping onto the FTC so that the display doesn't overwrite the dark tinge drop selection border.
- Enhanced the RTF color parser to be more robust.
- Fixed a paragraph optimization problem where selections were not properly preserved when combining style runs with the same characteristics.
- Fixed a problem with border cases when internally splitting style runs when applying style changes.

- Fixed a problem when a picture is in resize mode and you select a range of text and the picture doesn't exit resize mode. For example, click on a picture and then do a select all.
- Fixed a bug in the XML parser in handling the loading of character styles.
- Added "read only" checks to the appropriate edit action methods.
- Added range checking for setting paragraph margins.
- Fixed a problem with tab selection drawing past the right margin.
- Fixed a problem with the RBScript indentation parser handling single line if-then constructs.
- Fixed a situation where deleting a selection and the first or last item in the selection was a custom object and wasn't deleted.
- Eliminated a problem with custom objects under Windows where the custom object could flicker under certain situations. Specifically, the OpenXML event is now called first so you can extract the data before drawing is done.

Version 1.7

New Features and Enhancements:

- Implemented an optimization with calculating partial widths of strings in style runs.
- Added a DrawControlBorder design time property that will control if a border is to be drawn around the control.
- Increased the cache hit ratio for partial and full width and height measurements. This should increase the speed of the FTC in general.
- Optimized the mouse drag event so that is only called when something is changed. Kind of like turning down the idle speed.
- Added some fuzzy logic to the highlight drawing method to prevent the highlight color from "wiggling" at the end of style run because of the way the RB calculates string widths that causes slight inaccuracies. This only affects OS X.
- Optimized the getting of text from a paragraph. This improves the internal composing speed of the FTC.
- Put in logic to prevent live spell checking from being invoked when the user is in the middle of selecting text. This will make text selection more responsive.
- Spell checking is now invoked when you do things like pasting.
- Under Windows, eliminated flash in the display when undoing or redoing.

Bug Fixes:

- Fixed a problem with printing when the page range is manually specified in code.
- Made the selection algorithm smarter when selecting the beginning or end of a paragraph.
- Corrected a problem with the insertion caret being drawn with the wrong pen height which would cause a fuzzy looking caret under Mac OS X.
- Made minor adjustments to the undo manager.
- Fixed a problem with the scaled width and height of large pictures not being correctly written out to the XML stream.
- Fixed a problem with recreating an internal picture buffer when it shouldn't and it would cause crashing under the right circumstances. This fix will also speed up the code since it now only creates the buffer when needed and not all the time.
- Added some internal range checking.
- Fixed a problem with pasting RTF data that would cause an exception.
- Fixed a problem with typing in normal letters and hitting the return key simultaneously would cause erroneous duplication of chunks of text.
- Rewrote some of the selection algorithm.
- Removed some extraneous code.
- Fixed a problem with deleting a character at the end of a paragraph where the insertion point is placed in the wrong position after the deletion.
- Fixed a problem with live spell checking after performing deletions of material.
- Fixed a problem with the scaled width and height for pictures was not properly set in the XML output.
- Fixed a problem with dragging pictures where a picture could be inserted into the wrong spot after the drop.
- Fixed a problem with the caret would stop blinking after deleting selected text.
- Fixed a problem with spell checking being invoked at inappropriate times while typing.
- Fixed a problem with tabs incorrectly displaying a background selection when it was not selected and something else was selected in the paragraph.
- Fixed a problem with cloning the demo FTGraph object.
- Fixed a problem when resizing a picture when the display is scaled and then undoing the resize.

- Fixed a composing problem when undoing a delete across paragraph boundaries.
- Fixed mathematical rounding errors when displaying the position of paragraphs when the display is scaled. This would manifest when you select paragraphs and you see small gaps between the selected paragraphs.
- Made marks hug a little closer to the base of a line so that it isn't drawn over by the next line's selection.
- Fixed a problem with hitting the shift-home/end keys in certain combinations which could incorrectly select an extra line.
- Fixed some issues with handling the before paragraph space.
- Fixed an issue with linux and spell checking.
- Fixed a cosmetic issue with the placement of scrollbars under linux.

Version 1.6.2

Bug Fixes:

- Fixed a problem with the demo application where it would fail under linux with a run time error because it couldn't find the library "libstdc+ +.so.5". The cause of this problem was the Einhugur WindowSplitter plugin. I have eliminated the use of this plugin and now the demo works under linux.
- Corrected a problem with mouse wheel scrolling.
- Fixed a problem with printing paragraph background colors. It is now properly scaled.
- Fixed a problem with position updating which would cause various errors.
- Fixed a problem with contextual menu click positions not being interpreted correctly.

Version 1.6.1

Bug Fixes:

- Fixed a problem with threads that could lead to a random crash.

Version 1.6

New Features and Enhancements:

- Added spell checking as you type infrastructure functionality. There is a property called SpellCheckWhileTyping that allows you to turn it on and off. You also must fill in the SpellCheckWord and SpellCheckParagraph events do the actual spell checking.

- Added new parameter to the InsertionPointChanged event to show you the previous and current insertion offsets.
- Enhanced the internals for calling the InsertionPointChanged event so that it is call only when there is an actual change.
- Implemented some internal speed optimizations.
- Made minor improvements to the demo.

Bug Fixes:

- Fixed a problem with printing pictures when the original display was scaled and then the print functionality was invoked. Pictures should now print at the proper size. This fix also increases the speed of printing documents with pictures.
- Removed bit depth property and replaced it with a constant with a 32 bit depth.
- Sorted out some issues with picture handling.
- Changed the addObject method by removing the optional optimize parameter.
- Fixed a problem where strikethroughs, underlines, and marks were inconsistently handled across different print drivers.
- Added support for the RTF tags picw and pich.
- Fixed the situation where double clicking on a picture would select text outside of the picture.
- Fixed a couple of problems with dragging and dropping pictures when the display is scaled.
- Fixed a picture deletion problem.

Version 1.5.2

Bug Fixes:

- Fixed a boundary problem with the findWordBoundaryAtInsertion method.
- Added a generic findWordBoundaryAtOffset method to help find a word at an arbitrary offset.
- Fixed a problem with the demo spell checking code. If you used this as your template for your own code, please look over the spellCheckParagraph event to see the changes I made.
- When printing a picture, the original picture is used to maximize the resolution.

- Put in more checking in the compositions methods for lines that end up being taller than the page. This would typically happen when you print from edit mode and you have a large picture that is bigger than the printed page size.
- Enhanced to the code to handle really big pictures.
- The contextual menu click code inadvertently would change the insertion point. It now preserves the insertion point.
- Added the methods changeFontSizeAll and changeFontAll to the FTRBScriptField class to allow you to change the look and feel of the editor with one method call.
- In the FTRBScriptField class, set the AcceptTabs property to false filtered out all tabs when setting the text into the control.
- Fixed a problem with the insertion caret disappearing in an empty FTRBScriptField.
- Fixed a problem with unbalanced show/hide cursor which would cause the cursor to stay hidden at certain points.
- Fixed a problem where contextual menu code wasn't reseting internal flags correctly.

Version 1.5.1

Bug Fixes:

- Fixed the getText method to not append a return on the last paragraph.
- Fixed a display problem with inserting text into the FTRBScriptField and making sure everything gets indented properly.
- Enhanced the setXML method to handle empty strings.
- Removed the "import" method. Use appendFTC instead.
- Added a call to the resize method in open event to make sure the content and scrollbars are up to date.
- Added support to insert the FTSegment class into a document. This allows you to build sections of text using FTParagraphs and FTObjects and then insert them in to a document.
- Fixed a cosmetic bug in the placement of scrollbars.

Version 1.5

New Features and Enhancements:

- Implemented FTRBScriptField. This is an editor for REALbasic language that supports syntax coloring and line indention.

- Added a single line mode to the parser where it takes an entire paragraph and displays it as one line. This is useful for code editors.
- Implemented scrolling to the caret when you type in the horizontal direction so that the insertion caret will always be visible after you type.
- When selecting text with the mouse, it now tracks the horizontal mouse movement and scrolls the display accordingly.
- Refined the find insertion point functionality to consider on what part of the character the point resides and if it is more than halfway, it moves the insertion point after the character.
- Removed a restriction where a click off the page in the control wouldn't reset the insertion point to the right side of the affected line.
- Added support for single and double quote RTF commands.
- Reworked the naming of the edit view mode to setEditViewMode to make it more clear. Also overloaded it to allow passing in individual margin values so that you can have more control over the look and feel.
- Updated FTCStrings module to add the "&" symbol to the "undo" and "redo" strings so that they appear with underlines in the menu under Windows.

Bug Fixes:

- Added more range checking to the adjust display functionality.
- Added more nil checking to the change selection methods.
- When moving the insertion caret with the arrow keys it now correctly sets the line bias when in between lines.
- Removed the superfluous command-shift-left/right arrow combination code.
- Fixed a problem with parsing lines that would cause text to disappear and appear under certain circumstances.
- Fixed a problem with undoing a picture deletion via a backspace.
- Corrected a problem with printing scaled images and custom objects where they would not print at the correct size.
- Fixed a problem with strikethrough, marks, and underlines would draw to long when printing.
- Fixed a NilObjectException that would occur in the open event when manually setting the mode to Edit.

- Enhanced the deletion and update methods to more properly coordinate the updates of pages when in page mode.
- Fixed a proportional picture sizing problem.
- Fixed a problem with resizing pictures at different scales in various modes.
- Fixed a mathematical creep in the dimension of a picture when scaling.
- Fixed small gaps between paragraphs when a paragraph background color is specified.

Version 1.4

New Features and Enhancements:

- Implemented the ability to embed the FTC directly into a canvas control. This allows you to have the full capabilities of the FTC within a custom canvas.
- Added the method FTDocument.findXYOffset to return the FTInsertionOffset that coresponds to the X and Y coordinates. See the TestWindow.updateCoordinates method to see how to use it.

Bug Fixes:

- Fixed a problem with the tag field not being copied for FTParagraphs when they were cloned internally.
- Fixed a problem with the RTF software appending an extraneous empty paragraph to the output.
- Fixed a problem with cutting and pasting over selected text.
- Fixed a problem with deleting a selection of text and the paragraphs weren't composed properly.
- Fixed a problem with dragging and dropping a picture where the drop contained extraneous text.
- Fixed a problem with mark, underlined, or strikethrough not being applied to a picture at the end of the paragraph.

Version 1.3

New Features and Enhancements:

- Implemented the following key combinations.

  option-home: Jump to the beginning of the document.
  option-end: Jump to the end of the document.
  shift-home: Selects from the cursor position to the beginning of the line.

shift-end: Selects from the cursor position to the end of the line.

option-shift-home: Selects from the cursor position to the beginning of the document.

option-shift-ends: Selects from the cursor position to the end of the document.

- Implemented option-delete to delete the entire word preceding the insertion caret or if in the middle of a word, the preceding part of that word.
- Implemented option-forward-delete to delete the entire word following the insertion caret or if in the middle of a word, the last part of that word.
- Increase the triple click time to make it more responsive.
- Enhanced the insertion caret height to reflect the current font ascent in which it resides.
- Added FTCString module to hold constants for items that will be displayed. In version 1.3 it holds all the undo/redo strings that are displayed to the user.
- Separated the configuration files into a separate folder to make it easier to upgrade your projects when a new FTC release comes out. The configuration files are now kept in the "FTC Config" folder and if you change them for your project, you can keep them separate and not have to fiddle with updating them each time. So 99% of the time you will just install the "FTC Core" folder.
- Rewrote the code for handling multiple clicks covering double and triple clicks. It should be more responsive and act correctly.
- Added the funtion getDisplayPicture to the FormattedText class to access the display picture.
- Added support for encoding and decoding tags into the XML stream. Two new events were added to the FormattedText class to allow you to encode the tags. In the TextWindow in the demo project there is some sample code you can use as a template.

XmlEncodeTag(obj as FTBase) as string

Return an encoded string representing the obj.tag field.

XmlDecodeTag(data as string, obj as FTBase)

Take the data passed in and decode it and assign it to the obj.tag field.

Bug Fixes:

- Fixed a problem with hitting the home or end key when there is a selection. It now deselects the selected text and then moves the insertion caret.

- Fixed a problem where the encoding was not set properly for text that was pasted coming from the the FTC.
- Fixed a crash in an optimization where typing in several spaces at the end of a paragraph would cause bad input internally. This could manifest in several ways. This fix should cover all those occurrences.
- Fixed a problem with adding spaces at the end of a centered line of text.
- Removed the homeEndKeyBehavior property (see the added option-home/end key combinations).
- Restored the data optimization of paragraph data structures to clean up empty style runs in a paragraph when it is composed.
- Fixed a problem with triple clicking on an empty paragraph and the insertion caret keeps blinking.
- Fix a problem with composition of lines in certain situations where it wasn't properly looking ahead for spaces.
- Fixed a problem with limiting resizing a picture below the minimum values.
- Rewrote the right alignment code to be more consistent.
- Fixed a problem with setting the insertion point to the beginning of the document where the bias was not correctly handled.
- Rewrote the code handling leading and trailing spaces with underlines.
- Fixed a problem with normal view mode not having the correct scroll length when scaled.
- Under windows fixed a problem with the control-delete key combination not being invoked properly.
- Fixed a problem with paragraph background color not scaled to the right width when the display is scaled.

Version 1.2.1

Bug Fixes:

- Fixed a problem with tabs displaying an incorrect background color when being cut and pasted.

Version 1.2

New Features and Enhancements:

- Implemented the following key combinations.

option-up arrow: Move to the beginning of the paragraph.
option-down arrow: Move to the end of the paragraph.
option-left arrow: Move to the beginning of the current word or previous word.
option-right arrow: Move to the end of the current word or next word.
option-shift-up arrow: Select to the beginning of the paragraph.
option-shift-down arrow: Select to the end of the paragraph.
option-shift-left arrow: Select to the beginning of the current word or previous word.
option-shift-right arrow: Select to the end of the current word or next word.

- Isolated the edit menu functionality into the following methods so that you can call them independently. Use the following methods to invoke this functionality.

  editClearAction
  editCopyAction
  editCutAction
  editPasteAction
  editRedoAction
  editUndoAction

  Also implemented methods to check to see if this functionality is available.

  isClearActionAvailable
  isCopyActionAvailable
  isCutActionAvailable
  isPasteActionAvailable
  isRedoActionAvailable
  isUndoActionAvailable

- Added a ScrollChanged event to notify you when the user changes the scrollbars.
- Made line and page step visible in the FTScrollbar so that you can change them at design time.
- Added a PostDisplayDraw event to the FormattedText class. This event gives you access to the display buffer just after the FTC completes drawing its contents and before it blits it to the screen. This event allows you draw any additional elements on the face of the control.
- Added more logic to more intelligently hide and show the cursor as you type, move the mouse, and transition focus.
- Implemented an optimization of the segmentString method.

- Enhanced the selectionXXX methods to take into account the current style binding in determining if a style attribute is currently set.
- Optimized the FTStyleRun.setText method for an overall speed improvement in the FTC of about 5-7%. If you call the setText method in your code, it now takes an optional parameter for the length of the string you are passing in. So if you know the length, passing it in saves the time of calculating the string's length. If you pass in an incorrect length, you will cause problems, so it is up to you to make sure this value is accurate.

Bug Fixes:

- Added initializers to the FTScrollbar class to set the line and page steps to make sure they are set when added to a new project.
- Moved the call to the "open" event to the end of the call sequence to insure everything is ready to go when the open event is called in your instance or subclass.
- Modified the demo application to behave more like a regular editor in terms saving and opening a file.
- Fixed a problem with updating the display after deleting empty paragraphs.
- Updated the demo application to give the internal spell checker properly encoded string.
- Fixed another case that would cause words to flip between lines above where you are typing.
- Did some enhancements on the demo application to make it more like an editor.
- Fixed a bug with shift-command-left arrow with selecting previous lines when it shouldn't.
- Fixed a problem with jumping by word to the very beginning would cause a hang.
- Changed the key mapping from alt to ctrl for handling operations like arrow keys under Windows.
- Fixed a problem with printing the background color of text not matching the text width and height.
- Updated the demo application to use the latest spell check utilities code.
- Fixed an initialization problem with pictures when the scaled height and width were not specified in the XML. This only affects documents created with version 1.0 and moved to a new version.
- Fixed a problem with making sure the insertion caret was visible after typing returns in certain cases.
- The cursor is now hidden when you type and reappears when you move the mouse.

- Fixed a problem with finding the caret insertion point when doing things like hitting the End key or clicking past the end of the line.
- Fixed a problem with the insertion point bias (when the insertion caret is between lines) where it was set incorrectly when at the beginning of the document.
- Fixed a problem with scrolling the display to the insertion caret. When the caret was at the beginning of a line the scrolling motion would be jumpy. It now scrolls smoothly.
- Fixed a problem with the FTStyleRun.setText optimization where it did not properly account for zero length strings.
- Incorporated the latest SSCE source files in the demo.
- Added a filter for control characters being added to the FTC via the setText or insertString methods. All control characters will be discarded except for tabs.
- The XML output for the text field of the FTStyleRun is now base64 encoded. This avoids problems with illegal characters being introduced into the XML stream. The side affect of this is programs using older versions of the FTC (1.0 to 1.1) will not be able to decode the new format. Version 1.2 will read the new format and the old format, so version 1.2 is backwards compatible.
- Fixed an internal scaling consistency issue when doing actions like inserting a page break at a scaled output.
- Fixed a problem when a picture was in resize mode and you printed the document and the drawing code would attempt to draw the selection rectangle in the print output.
- Fixed a problem with the before and after space of a picture when printing and the background color is set. The background color dimensions now match when printing.
- Fixed a problem with the before space of a paragraph when composing pages under certain circumstances.
- Fixed a problem with a selected picture where the right middle handle being drawn off center. The handle is now draw properly centered.
- Fixed a problem with setting the insertion point to the beginning of a line when the paragraph is in center or right alignment.
- Fixed a problem with displaying the selection of tabs.
- Fixed a problem with the selection showing gaps appearing between paragraphs at odd scales line 125%.
- Fixed a problem when switching display modes and a picture is larger than the margins in the new mode. The picture will be scaled to fit the new mode's margins.
- When in read only mode, the cursor now remains visible when the user types.

- Fixed a problem with comparing style runs that have been marked.
- Misspelled word indicators are now drawn over mark indicators.
- Rewrote the algorithm for tracking proportional picture size changes to account for changes elsewhere in the software.
- Adjusted the drag threshold time from 10 to 12 ticks to accommodate better timing under Windows.
- Fixed a problem with copying or cutting pictures in resize mode.

Version 1.1

New Features and Enhancements:

- Added hyperlinks via a custom object class called FTCustomHyperlink. This custom class is a stop gap measure until full fledged hyperlinks can be implement in version 2.0. The only functional difference between this implementation and the one to come in 2.0 is the ability to edit the link inline. The FTCustomHyperlink will act like a picture in that it will act as one unit.
- Added a setDrawBorder method to the FTPicture class that allows you to control if a border is draw around a picture when it has the focus. This is meant for custom object who want more control over their appearance.
- Added a setNudge method to the FTPicture class that allows you to control where the picture is drawn in relation to the baseline. If you are drawing text in a custom object, this gives you control over where the baseline in the custom object appears in relation to the rest of the drawn text.
- Enhanced the FTCustom class by adding a ScaleChanged event. This event is triggered when the scale of the document changes.
- Made the out of focus selection color a shade darker to make it more distinguishable.
- Made cut an pasting and dragging of pictures more robust.
- Added a bias attribute to insertion points for handling where the insertion point is displayed when it falls between lines. An example situation where you will see this is when you click to the left of a paragraph.
- Selection rectangles now enclose the actual text.
- Added a method called appendFTC that allows you to append the contents of one FTC to another FTC control. This method is present on both the FormattedText and FTDocument classes.
- Added design time property called EditViewPrintMargin to specify the margins for printing when in edit mode.

- Added a getLineCount method to the FTDocument class.
- Enhanced the paint event in custom objects by passing a printing flag and the scale needed for printing. This allows you to do high resolution output.

Bug Fixes:

- Fixed a couple of bugs associated with passing in an empty string into the setText method.
- Enhanced the FormattedText class print method to take first and last pages to print as parameters. If you leave them unspecified, then the entire document will be printed.
- Fixed a problem with file types where the built-in AcceptFileDrop method corrupts the internal RB framework data if you pass in a direct reference. This problem is manifested by allowing you to open a file the first time and the second time it improperly filters files.
- Fixed a problem with paragraphs in right alignment where the selection area didn't match the actual selection.
- Fixed a problem when clicking on a picture that was part of selection and it selected the picture instead of starting a drag.
- Fixed a problem under Windows where the insertion point would be off as you clicked down the page due to a rounding error.
- Fixed a problem with preserving the resize mode of a picture when it is dragged in downward direction.
- Fixed a problem with composing lines with pictures in certain positions on the line.
- Fixed an insertion point problem with setting the insertion point at the end of the line.
- Added a call to make sure the insertion caret is set to visible after each key stroke.
- Fixed some issues with handling read-only mode and custom objects.
- Fixed a problem with the paragraph optimizing method not properly joining style runs together which in turn could cause odd line breaks and flipping of words between lines.
- Fixed a problem with words flipping back and forth between lines above where typing is occurring in a paragraph.
- Fixed a problem where points were translated incorrectly with custom objects in page mode under certain circumstances.
- Fixed a problem with selections becoming disjointed after setting a paragraph style. The selection range is preserved after changing a paragraph style.
- In the demo application, made the character style menu items disabled if nothing is selected.

- Fixed composition problems when printing from normal or edit view modes.
- Fixed a problem when switching to edit mode and the previous mode's scale was not 100%.
- Fixed a problem with making sure the virtual callPaint method is call for custom objects.
- Fixed a problem with clicking in a picture to drag it when it is part of a bigger selection.
- Changed the minimum length for pictures from 1 to 3 pixels when resizing pictures. This makes it easier for the user to click on the picture and grab a resize handle when it is sized to its minimum height and width.
- Fixed a problem with parser occasionally splitting lines one character to far and thus over drawing the margin guides.
- Fixed a problem with underlines and strikethroughs were not being applied to pictures.
- Fixed problems with adding to the selection with the arrow keys.