Scintillating! A Modernized Text Editor for VA Smalltalk

Seth Berman
Software Engineer
Instantiations, Inc.

Requirements

- Provide a modern text editor
  - Additional visual cues and styling
  - Take advantage of the latest technologies
- Minimize change to our existing system
  - Maintain full API compatibility with existing editor
  - Structural compatibility with our widget frameworks
- Must be inline with our cross-platform roadmap
  - GTK
- New capability must be made accessible to our customers

instantiations
VA Smalltalk

Scintilla

- Free library providing functionality to help build text editors
- Provides numerous features specific to source code editing
- Initial release in 1999
- Active community
- Used by Code::Blocks, Notepad++, TortoiseSVN
- Cross-Platform support

Scintilla Integration in VAST

- Integrated into our Common Widgets Framework
- New CwScintillaEditor widget integrated
- Offers full API support for Scintilla 3.3.3 (the latest)
- Compatibility methods implemented to provide 100% API capability with our existing legacy editor components

Direct2D/DirectWrite

- Microsoft’s technology to provide higher quality font rendering
- Hardware-Accelerated Rendering
  - Offloads many aspects of rendering to the GPUs
- Windows 7 and above
- How noticeable a difference? Depends on
  - Font type and style
  - Monitor type and size
  - Your eyes and/or attention to detail

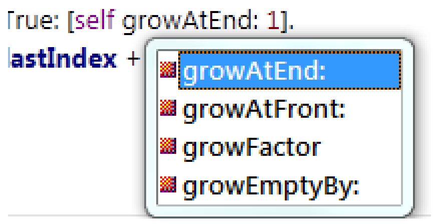
Text Editor Basics

- Auto-Indent
- Keyboard Shortcuts
  - <Tab> to indent, <Shift+Tab> to unindent
  - <Alt+Up/Down> to move blocks of selected text one line at a time
- DragNDrop to relocate blocks of selected text
- Margin Area

Multiple Undo/Redo

- Finally!
- No hard limit (only memory)
- Supports coalescing
  - Combine contiguous insertions and deletions into single undo operations
- APIs for user-defined coalescing

Code Completion



```
True: [self growAtEnd: 1].
lastIndex +
```

- Scintilla offers a code completion popup and API
- We hooked it up to our code completion engine
- Users have the option to select which popup they prefer
  - Basic (Minimal styling capability)
  - Extended (Maximum styling capability)
  - Scintilla (Somewhere in the middle)

instantiations VA Smalltalk

Syntax Color Highlighting

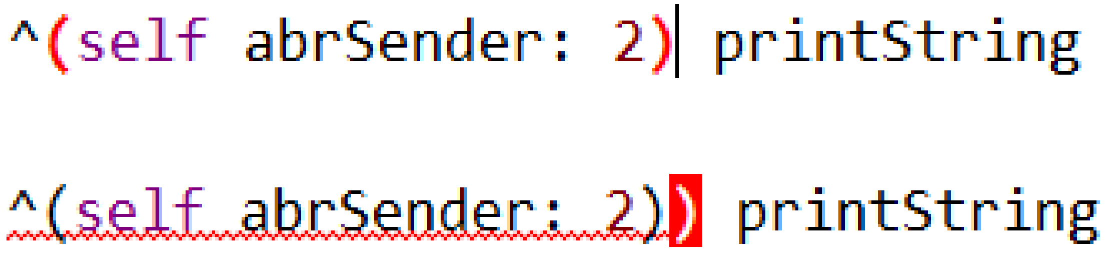
- Scintilla Lexer defines how a specified range of text is to be colored
  - Has lexer support for 80+ languages
  - Smalltalk is one of them, but it was too simplistic
  - Provides hooks to allow us to write our own custom lexer in Smalltalk (Container-Defined Styling)
- Container-Defined Styling
  - Scintilla specifies what needs to be styled via events
  - VAST's custom styler defines how the character range is to be styled

VAST Custom Styler

- Comes in 2 flavors
  - Method styler to style text in browsers and debuggers
    - Optimized to style methods
  - Snippet styler to style text in inspectors and workspaces
    - Optimized to style snippets of code
    - Adds some fuzzy logic rules
    - Now we can offer color in our inspectors and workspaces
- Styler uses a custom token scanner instead of parse trees
  - Enables real-time coloring
  - Much more flexible then our previous parse-tree implementation

Bracket Highlighting

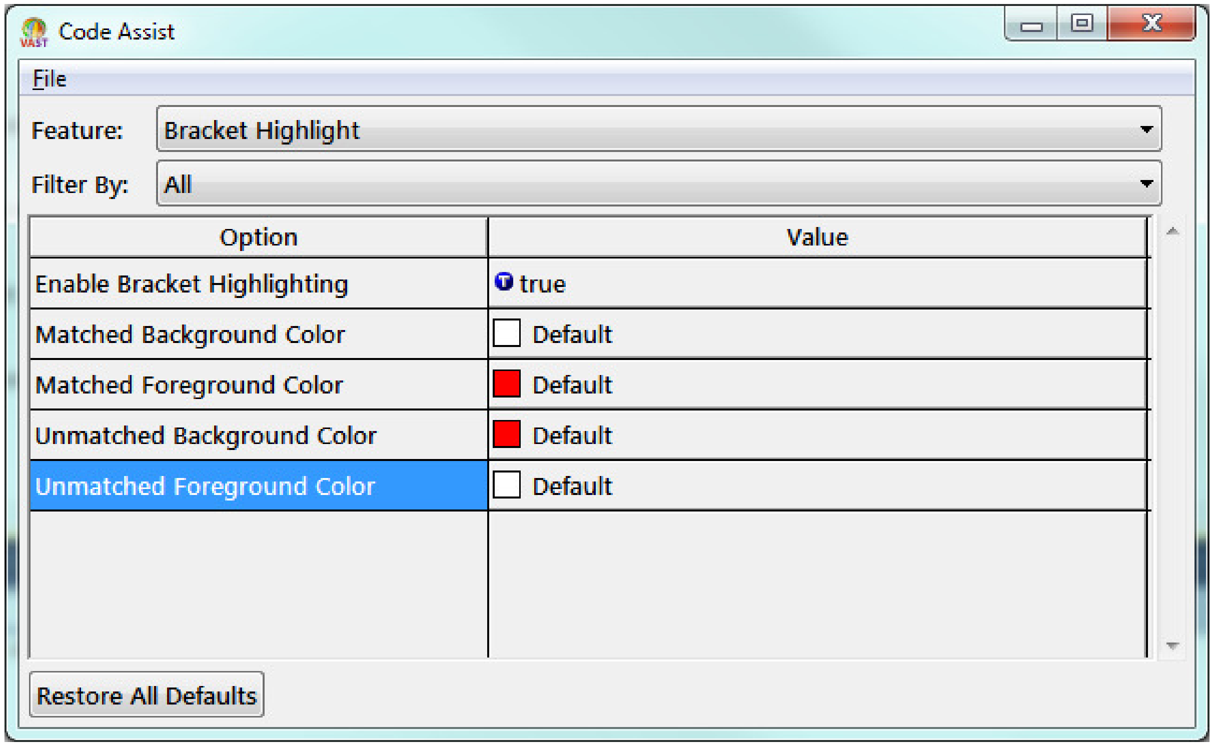
- Stylization used to indicate matching characters for ()[]{}
- Separate style used to highlight unmatched characters



```
^(self abrSender: 2)| printString

^(self abrSender: 2)) printString
```

Bracket Highlighting Cont...

- Bracket Highlighting is configurable



Feature: Bracket Highlight

Filter By: All

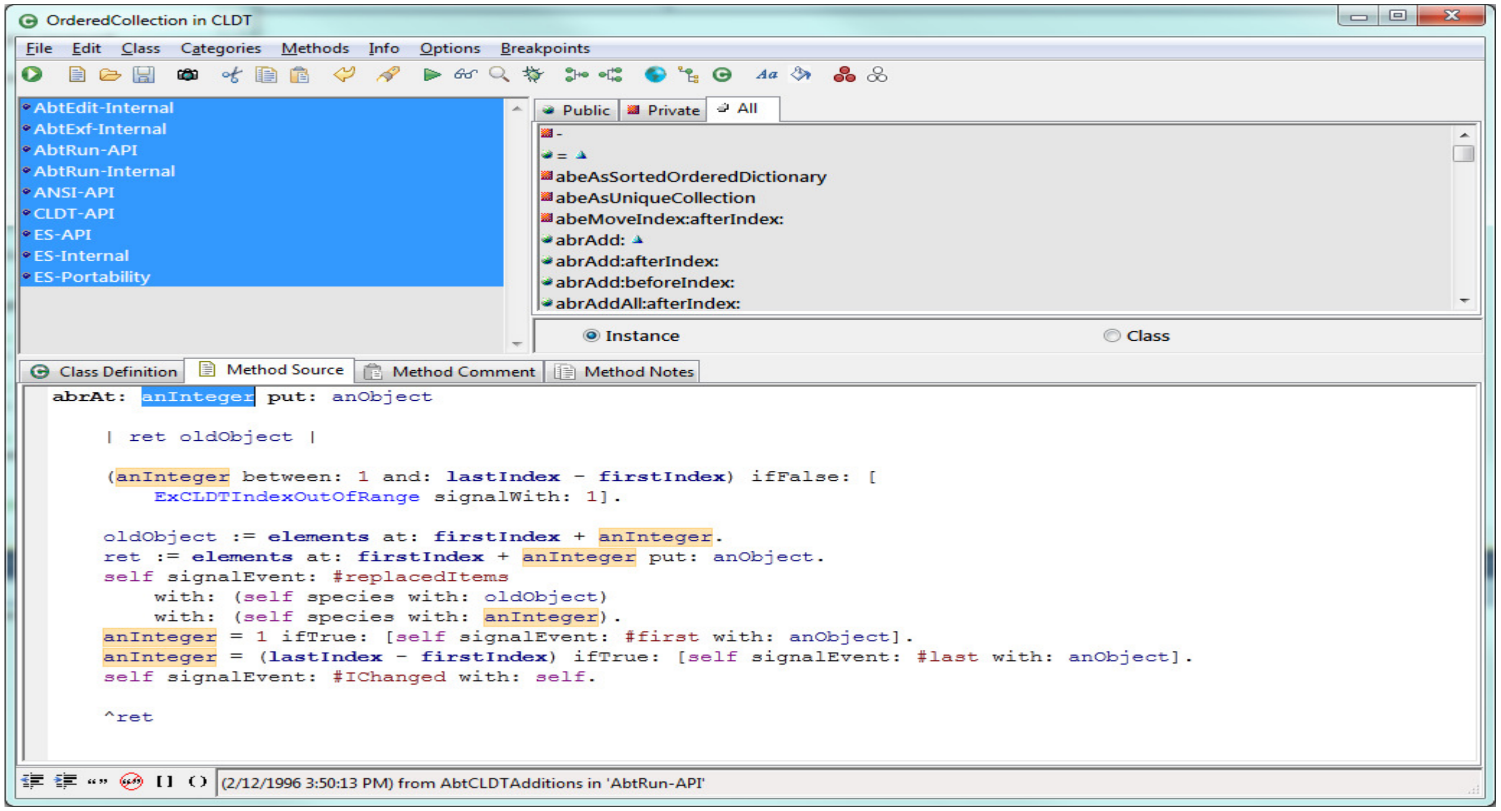
Option	Value
Enable Bracket Highlighting	true
Matched Background Color	☐ Default
Matched Foreground Color	🟥 Default
Unmatched Background Color	🟥 Default
Unmatched Foreground Color	☐ Default

Smart Highlighting

- Adding stylization to a selected word and any matching word in the source
- Useful for seeing local variable usage
- Adds extra decoration to highlighting browsers
- Added logic to handle block argument highlighting
- Stylization applied behind text so syntax color highlighting shows through

Smart Highlighting Cont…

- Where is the argument, *anInteger*, used?



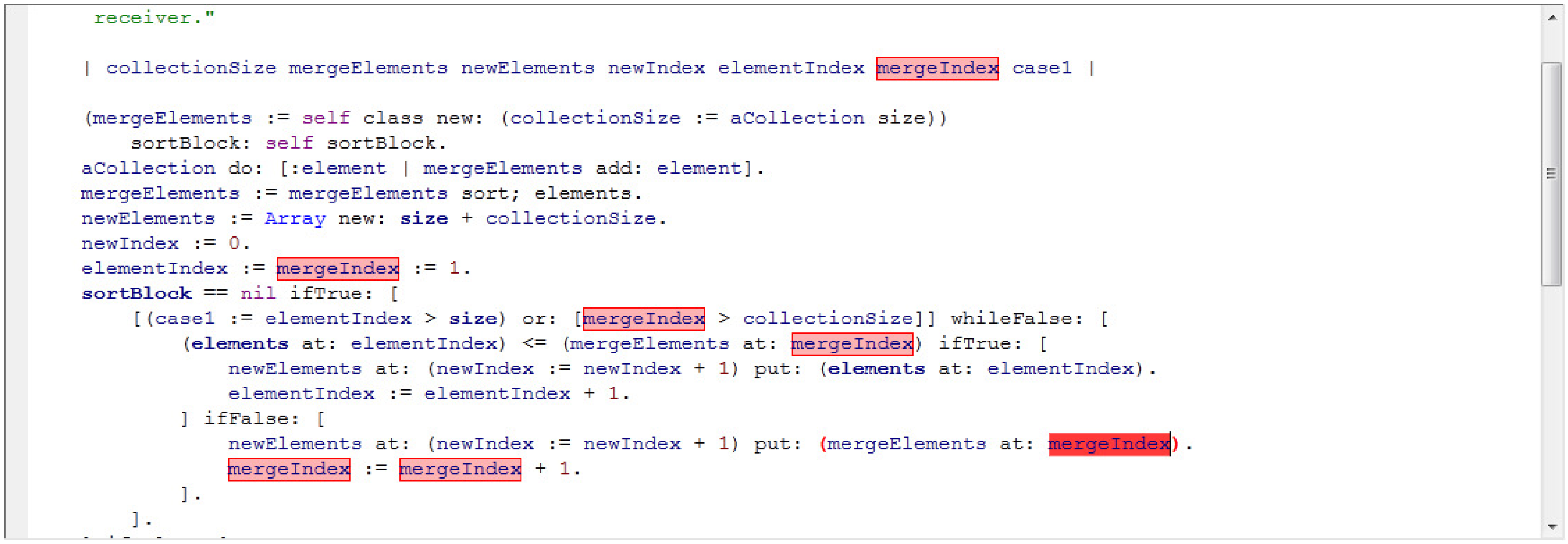
```
abrAt: anInteger put: anObject

    | ret oldObject |

    (anInteger between: 1 and: lastIndex - firstIndex) ifFalse: [
        ExCLDTIndexOutOfRange signalWith: 1].

    oldObject := elements at: firstIndex + anInteger.
    ret := elements at: firstIndex + anInteger put: anObject.
    self signalEvent: #replacedItems
        with: (self species with: oldObject)
        with: (self species with: anInteger).
    anInteger = 1 ifTrue: [self signalEvent: #first with: anObject].
    anInteger = (lastIndex - firstIndex) ifTrue: [self signalEvent: #last with: anObject].
    self signalEvent: #IChanged with: self.

    ^ret
```

(2/12/1996 3:50:13 PM) from AbtCLDTAdditions in 'AbtRun-API'

instantiations VA Smalltalk

Smart Highlighting Cont...

- Identify where variables are declared with a glance

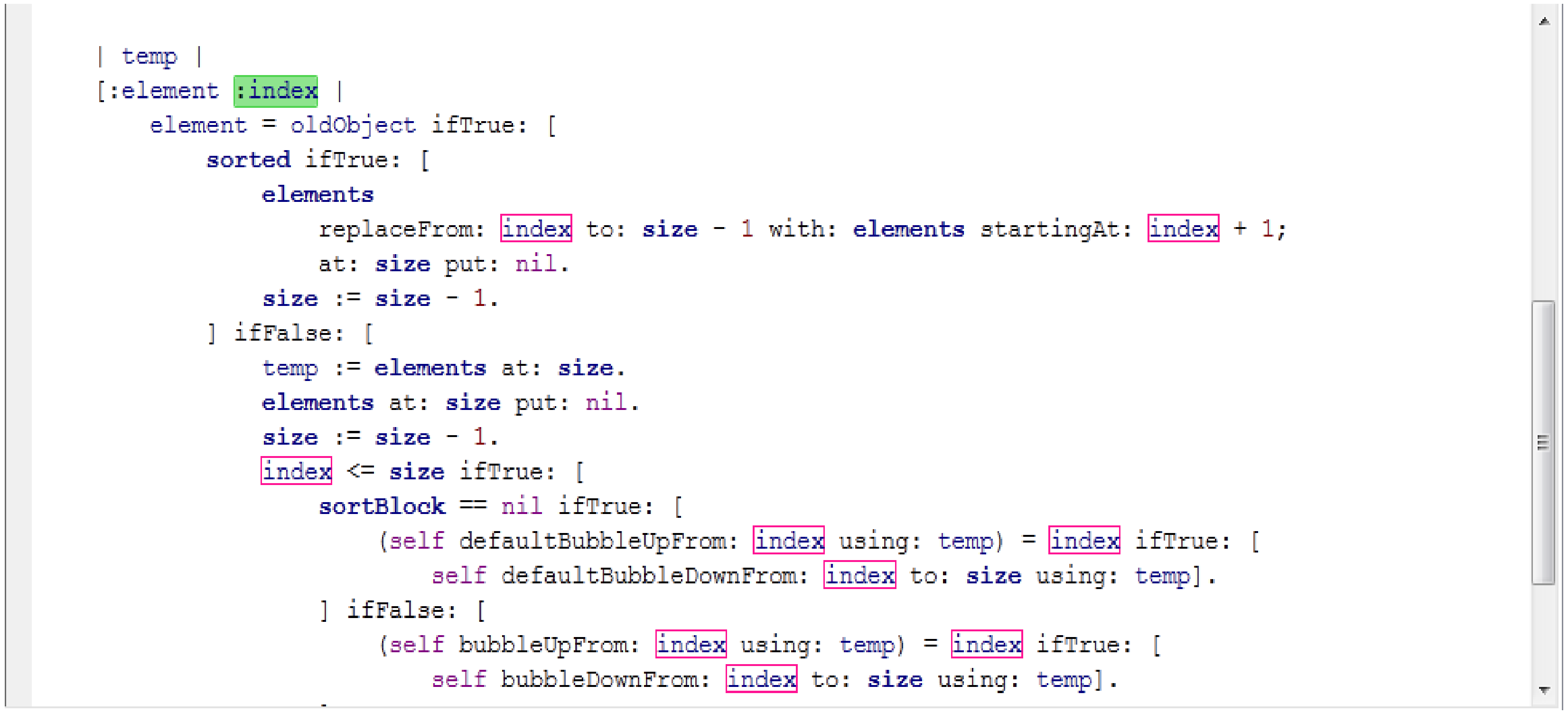


```
 receiver."

| collectionSize mergeElements newElements newIndex elementIndex mergeIndex case1 |

(mergeElements := self class new: (collectionSize := aCollection size))
    sortBlock: self sortBlock.
aCollection do: [:element | mergeElements add: element].
mergeElements := mergeElements sort; elements.
newElements := Array new: size + collectionSize.
newIndex := 0.
elementIndex := mergeIndex := 1.
sortBlock == nil ifTrue: [
    [(case1 := elementIndex > size) or: [mergeIndex > collectionSize]] whileFalse: [
        (elements at: elementIndex) <= (mergeElements at: mergeIndex) ifTrue: [
            newElements at: (newIndex := newIndex + 1) put: (elements at: elementIndex).
            elementIndex := elementIndex + 1.
        ] ifFalse: [
            newElements at: (newIndex := newIndex + 1) put: (mergeElements at: mergeIndex).
            mergeIndex := mergeIndex + 1.
        ].
    ].
```

instantiations VA Smalltalk

Smart Highlighting Cont…

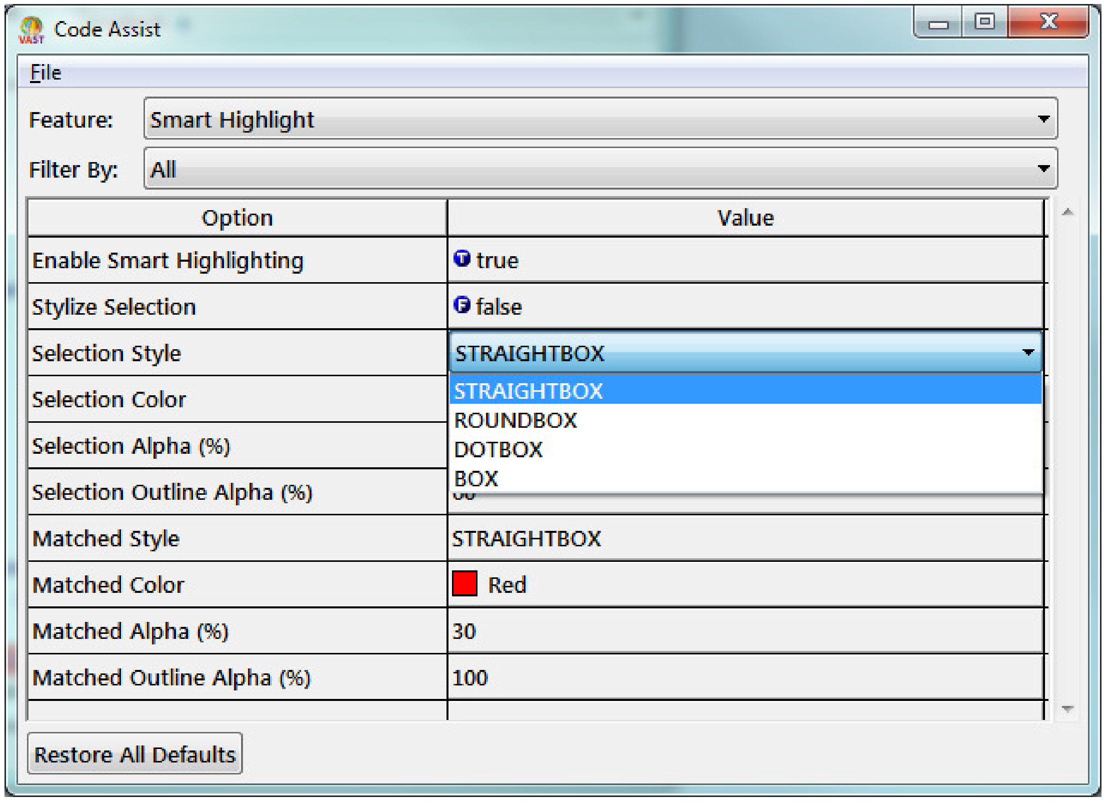
- Block-args included…



```
| temp |
[:element :index |
    element = oldObject ifTrue: [
        sorted ifTrue: [
            elements
                replaceFrom: index to: size - 1 with: elements startingAt: index + 1;
                at: size put: nil.
            size := size - 1.
        ] ifFalse: [
            temp := elements at: size.
            elements at: size put: nil.
            size := size - 1.
            index <= size ifTrue: [
                sortBlock == nil ifTrue: [
                    (self defaultBubbleUpFrom: index using: temp) = index ifTrue: [
                        self defaultBubbleDownFrom: index to: size using: temp].
                ] ifFalse: [
                    (self bubbleUpFrom: index using: temp) = index ifTrue: [
                        self bubbleDownFrom: index to: size using: temp].
```

Smart Highlighting Cont…

- Smart Highlighting is configurable



Feature: Smart Highlight

Filter By: All

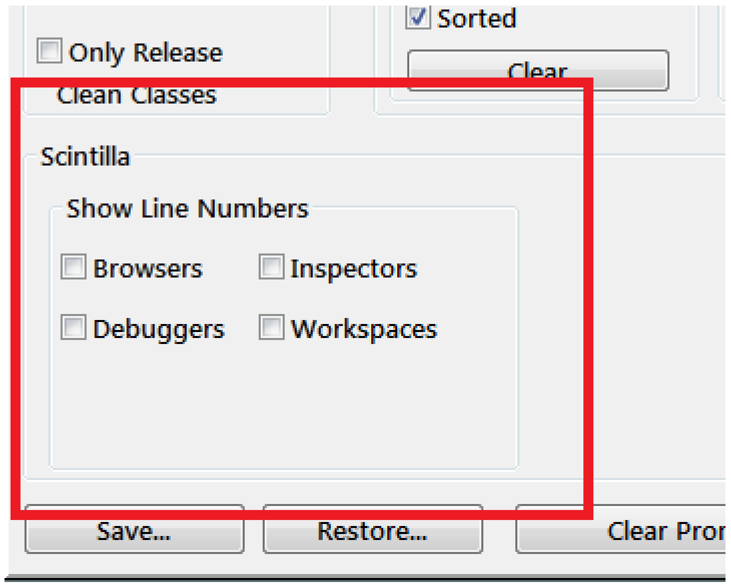
Option	Value
Enable Smart Highlighting	true
Stylize Selection	false
Selection Style	STRAIGHTBOX
Selection Color	
Selection Alpha (%)	
Selection Outline Alpha (%)	
Matched Style	STRAIGHTBOX
Matched Color	🟥 Red
Matched Alpha (%)	30
Matched Outline Alpha (%)	100

STRAIGHTBOX
ROUNDBOX
DOTBOX
BOX

instantiations VA Smalltalk

Line Numbers

- Available in Browser/Debuggers/Workspaces/Inspectors
- Line Number margins are dynamically sized
- Configurable across different window types



[✓] Sorted

[ ] Only Release Clean Classes

Scintilla

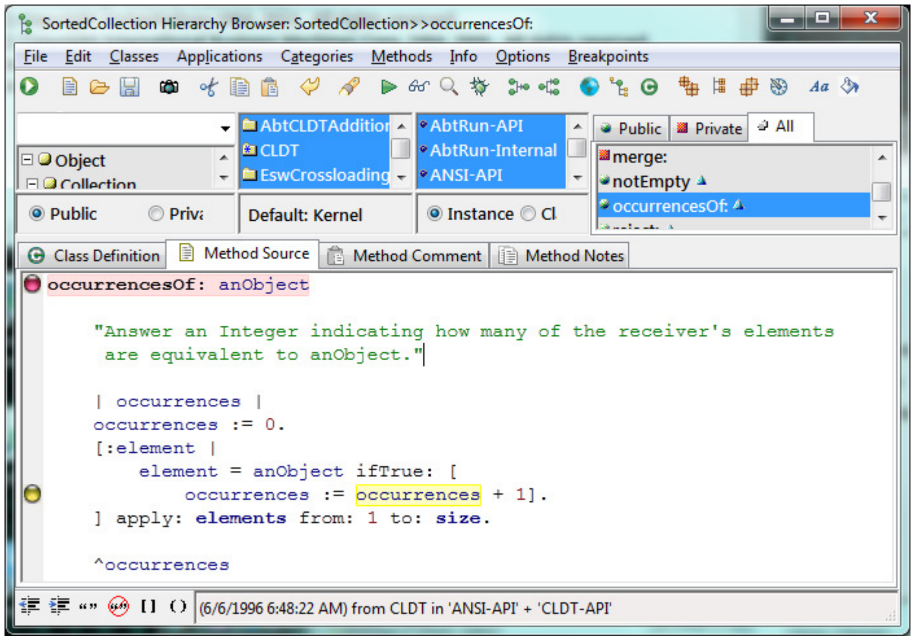
Show Line Numbers

[ ] Browsers [ ] Inspectors

[ ] Debuggers [ ] Workspaces

Breakpoint Management

- Persistent Breakpoint margin
- New Breakpoint icons
- Multiple Breakpoints / Line support
- Stylizes Breakpoint regions

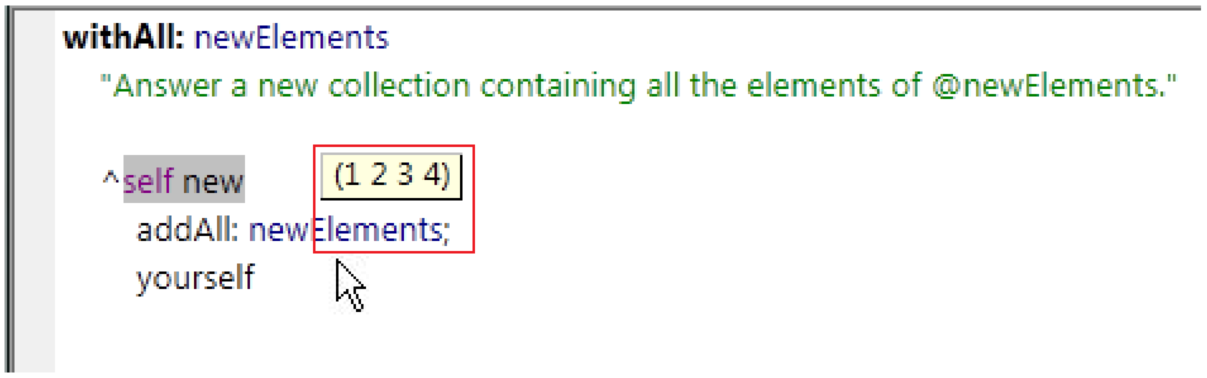


```
occurrencesOf: anObject

    "Answer an Integer indicating how many of the receiver's elements
     are equivalent to anObject."

    | occurrences |
    occurrences := 0.
    [:element |
        element = anObject ifTrue: [
            occurrences := occurrences + 1].
    ] apply: elements from: 1 to: size.

    ^occurrences
```

(6/6/1996 6:48:22 AM) from CLDT in 'ANSI-API' + 'CLDT-API'

Debugger Call Tips

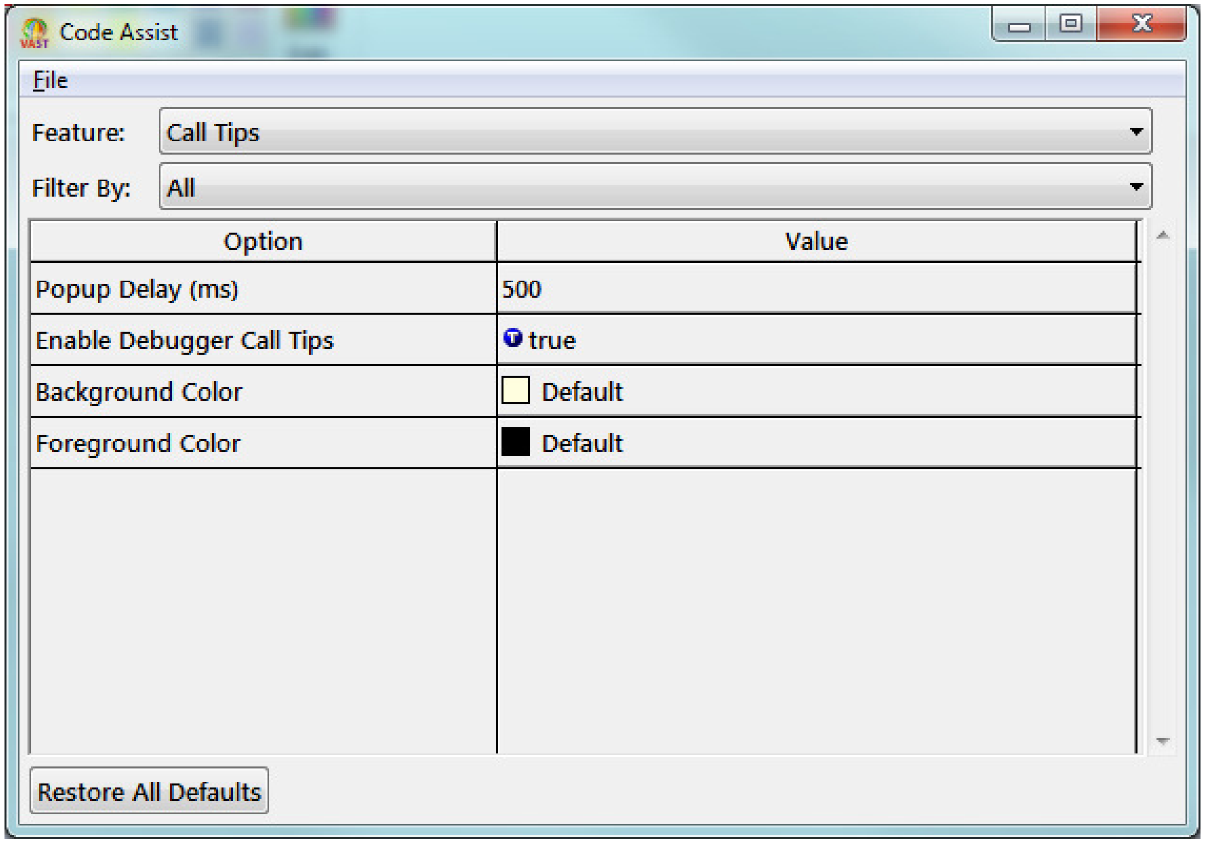
- Hover mouse over variables and globals displays a calltip showing it's value
- Clicking a calltip will bring up an inspector
- Long calltips are formatted to single-line



```
withAll: newElements
	"Answer a new collection containing all the elements of @newElements."

	^self new
		addAll: newElements;
		yourself
```

Debugger Call Tips Cont...

- Call Tips are configurable



Feature: Call Tips

Filter By: All

Option	Value
Popup Delay (ms)	500
Enable Debugger Call Tips	true
Background Color	⬜ Default
Foreground Color	⬛ Default

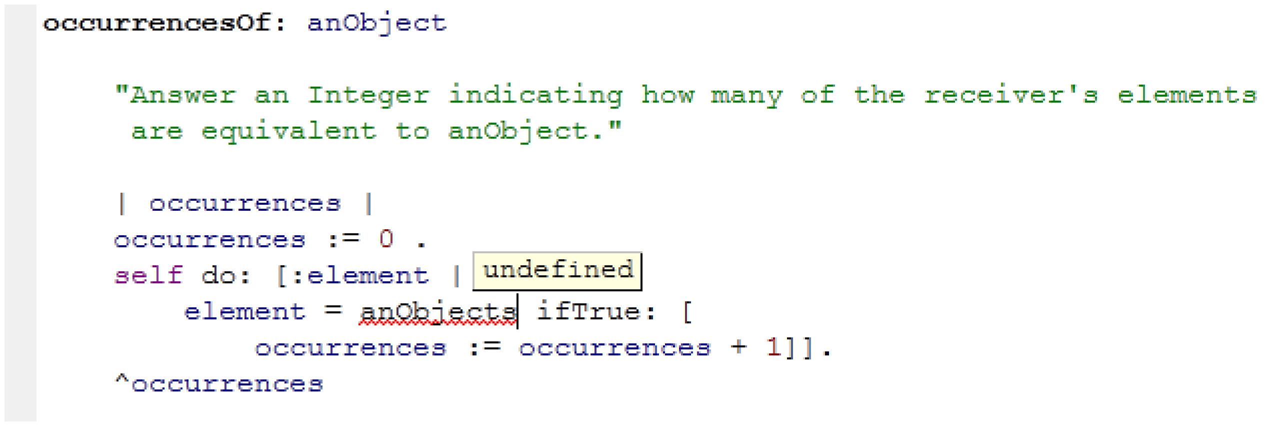
instantiations VA Smalltalk

Error/Warning/Info Indicators

- Stylized Squiggle lines underneath text to indicate
  - Fatal Errors
  - Errors
  - Warnings
  - Info
- Calltips provide information about the indicator
- Incremental compiler runs in the background to identify issues in real-time
  - Collects information from parse trees and the styler

Error/Warning/Info Indicators Cont…

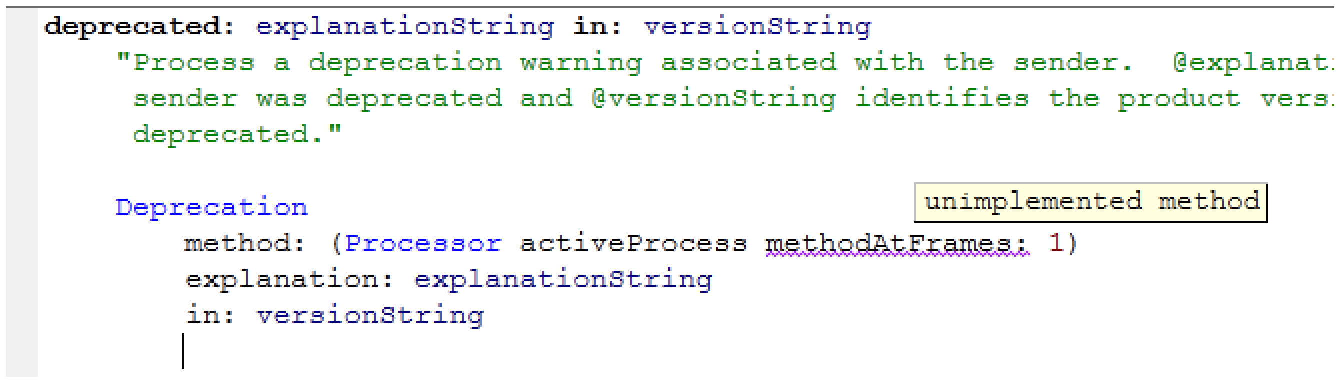
- Instantly identify misspelled variables



```
occurrencesOf: anObject

    "Answer an Integer indicating how many of the receiver's elements
     are equivalent to anObject."

    | occurrences |
    occurrences := 0 .
    self do: [:element | undefined
        element = anObjects ifTrue: [
            occurrences := occurrences + 1]].
    ^occurrences
```

Error/Warning/Info Indicators Cont…

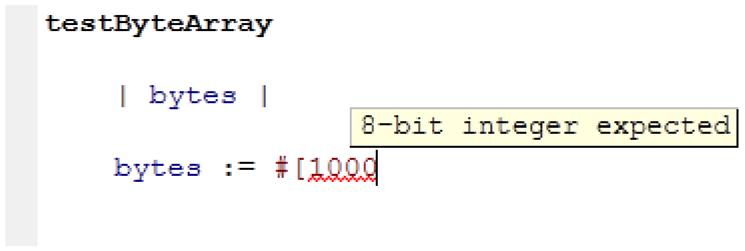
- Identify methods not implemented



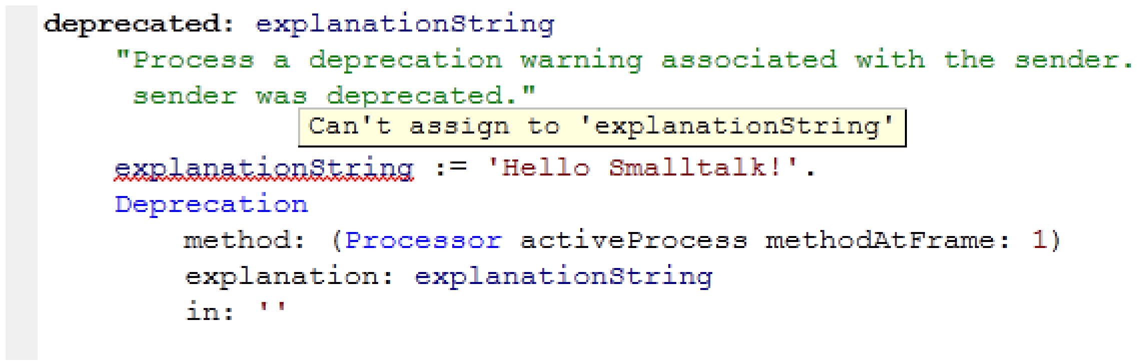
```
deprecated: explanationString in: versionString
    "Process a deprecation warning associated with the sender.  @explanat
     sender was deprecated and @versionString identifies the product vers
     deprecated."

    Deprecation                                          unimplemented method
        method: (Processor activeProcess methodAtFrames: 1)
        explanation: explanationString
        in: versionString
```

Error/Warning/Info Indicators Cont…

- Some more examples…



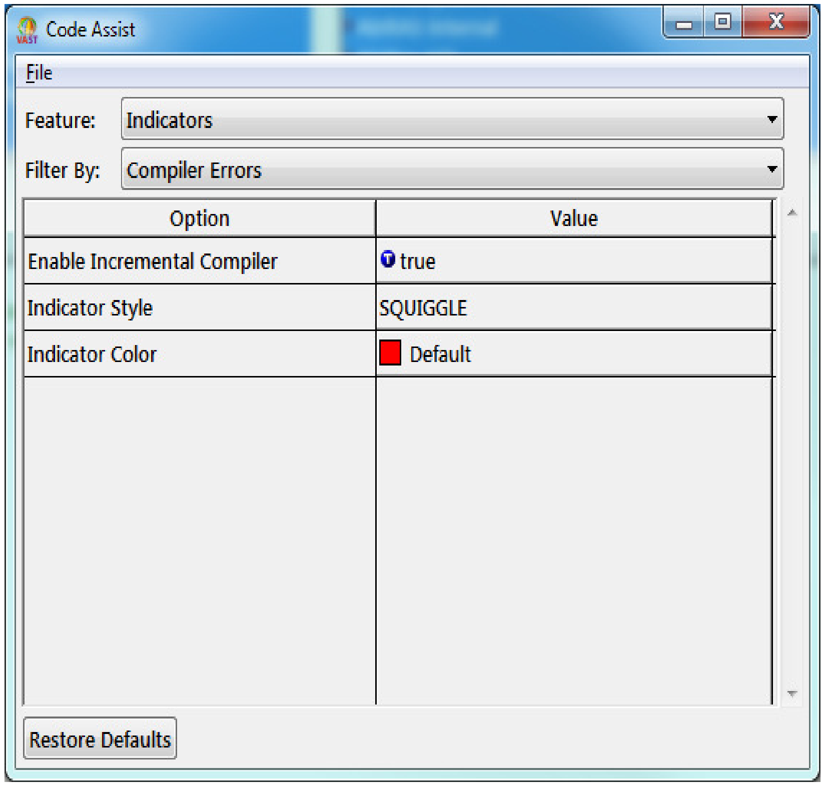
```
testByteArray

    | bytes |
                    8-bit integer expected
    bytes := #[1000
```



```
deprecated: explanationString
    "Process a deprecation warning associated with the sender.
     sender was deprecated."
                  Can't assign to 'explanationString'
    explanationString := 'Hello Smalltalk!'.
    Deprecation
        method: (Processor activeProcess methodAtFrame: 1)
        explanation: explanationString
        in: ''
```

Error/Warning/Info Indicators Cont…

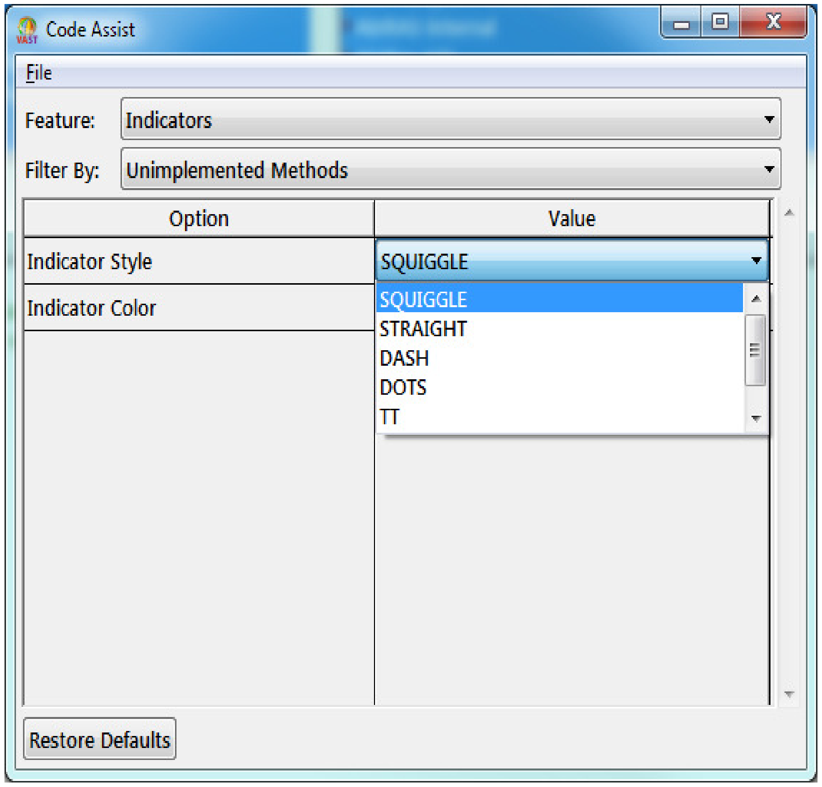
- Indicators are configurable



Feature: Indicators

Filter By: Compiler Errors

Option	Value
Enable Incremental Compiler	true
Indicator Style	SQUIGGLE
Indicator Color	🟥 Default



Feature: Indicators

Filter By: Unimplemented Methods

Option	Value
Indicator Style	SQUIGGLE
Indicator Color	

SQUIGGLE
STRAIGHT
DASH
DOTS
TT

Questions?