Encodings

Why?

Problems related to them are common

Important when speaking with external system

- Files
- FFI
- Database

They are Easy!

Provided you:

- Remember to deal with them
- Know what is expected by whom you communicate with
- Handle them consistently

Definitions

- (Abstract) Character Reportoire?
- Character Map ?
- (Coded) Character Set ?
- Character Encoding (Scheme / Form) ?

http://www.unicode.org/reports/tr17/

Files

- No specified filename encoding on Unix
- Impossible to know which single-byte encoding is used without external knowledge

http://www.dwheeler.com/essays/fixing-unix-linux-filenames.html#utf8

FFI

String format is dependent on library

- UTF16 for Windows
- UTF8 for Unix/Mac
- ??? for custom libraries

Databases

- Each have their own way of doing it

Take Oracle :

- Client fetches NLS_LANG from environment
- Or the registry (on windows)
- Or you set encoding manually

Did I mention they use non-standard encoding names?

Coding time...

```
apiGetEnvironmentVariable: lpName with: lpBuffer with: nSize
    <apicall: ulong 'GetEnvironmentVariableA' (char*  byte* ulong) module: 'kernel32.dll'>
    ^self externalCallFailed
```

Problems?

- Doesn't accept WideStrings
- Fails for values outside system code page
- Returns invalid results where ISO-8859-1 different from encoding
- Same on Unix, but with utf8 results

Solutions

- Always use method wrapper which handles encoding
- Use W version of interface for Windows APIs

```
apiGetEnvironmentVariable: lpName with: lpBuffer with: nSize
    <apicall: ulong 'GetEnvironmentVariableW' (char*  byte* ulong) module: 'kernel32.dll'>
    ^self externalCallFailed

getEnvironmentVariable: aString
"Windows 2000 or later required"
| buffer size utf16string systemConverter |
 “Windows unicode-encoding is UTF16-LE”
systemConverter := UTF16TextConverter new useLittleEndian: true.
utf16string := aString  convertToWithConverter: systemConverter.
size := self apiGetEnvironmentVariable: utf16string with: nil with: 0.
"size is in wchar_t, so double that for size in bytes"
buffer := ByteString new: size * 2.
size :=  self apiGetEnvironmentVariable: utf16string with: buffer with: size.
“Returned string is 0-terminated”
 ^(buffer convertFromWithConverter: systemConverter) allButLast: 1
```

Optimization

Why?

- Always have a need
- Optimization costs both time and complexity

How?

- Quantify a stop-condition
- Less code != Faster code
- Do it smarter

What is smart?

- Better algorithms
- More suited data structures
- Avoid redundancy
- Use domain/precondition knowledge

OK... But seriously, where do I start?

- TimeProfileBrowser onBlock: [myThingThatGoesToSlow]
- World menu -> System -> Start profiling...

Spy

- Framework for writing runtime program monitors (profilers)

www.squeaksource.org/Spy

http://www.esug.org/data/ESUG2010/IWST/2010-ESUG-ProfilingBlueprint.pdf

Reading a time profile

- Identify loops
- Leave leaf nodes for last

A Profile

Time Profile

```
--------------------------------
Process: (40s)  291241984: nil
--------------------------------
59.1% {9975ms} WorldState>>doOneCycleFor:
  58.0% {9781ms} WorldState>>doOneCycleNowFor:
    |53.7% {9065ms} WorldState>>displayWorldSafely:
    | |53.7% {9065ms} PasteUpMorph>>displayWorld
    | | 53.7% {9065ms} PasteUpMorph>>privateOuterDisplayWorld
    | |  53.7% {9065ms} WorldState>>displayWorld:submorphs:
    | |   42.9% {7237ms} WorldState>>drawWorld:submorphs:invalidAreasOn:
    | |    |42.6% {7184ms} FormCanvas(Canvas)>>fullDrawMorph:
    | |    | 42.6% {7184ms} FormCanvas(Canvas)>>fullDraw:
    | |    |  42.6% {7184ms} SystemWindow(Morph)>>fullDrawOn:
    | |    |   41.8% {7048ms} SystemWindow(Morph)>>drawSubmorphsOn:
    | |    |    41.5% {7007ms} FormCanvas(Canvas)>>fullDrawMorph:
    | |    |     41.5% {7004ms} FormCanvas(Canvas)>>fullDraw:
    | |    |      41.5% {7000ms} PluggableTextMorph(Morph)>>fullDrawOn:
    | |    |       37.0% {6233ms} PluggableTextMorph>>drawSubmorphsOn:
    | |    |        |37.0% {6233ms} PluggableTextMorph(Morph)>>drawSubmorphsOn:
    | |    |        | 37.0% {6233ms} FormCanvas(Canvas)>>fullDrawMorph:
    | |    |        |  36.9% {6232ms} FormCanvas(Canvas)>>fullDraw:
    | |    |        |   36.9% {6232ms} TransformMorph(Morph)>>fullDrawOn:
    | |    |        |    36.3% {6118ms} TransformMorph>>drawSubmorphsOn:
    | |    |        |     36.2% {6098ms} FormCanvas(Canvas)>>fullDrawMorph:
    | |    |        |      36.2% {6098ms} FormCanvas(Canvas)>>fullDraw:
    | |    |        |       36.2% {6098ms} TextMorphForEditView(Morph)>>fullDrawOn:
    | |    |        |        36.1% {6093ms} FormCanvas(Canvas)>>drawMorph:
    | |    |        |         36.1% {6093ms} FormCanvas(Canvas)>>draw:
    | |    |        |          36.1% {6091ms} TextMorphForEditView>>drawOn:
    | |    |        |           27.5% {4643ms} TextMorphForEditView(TextMorph)>>drawOn:
    | |    |        |            |27.5% {4641ms} FormCanvas>>paragraph:bounds:color:
    | |    |        |            | 27.5% {4637ms} MultiNewParagraph(NewParagraph)>>displayOn:using:at:
    | |    |        |            |  27.3% {4607ms} MultiDisplayScanner>>displayLine:offset:leftInRun:
    | |    |        |            |   24.9% {4198ms} MultiDisplayScanner(MultiCharacterScanner)>>scanChar
    | |    |        |            |    |24.8% {4191ms} MultiDisplayScanner(MultiCharacterScanner)>>basicSc
    | |    |        |            |    | 17.4% {2942ms} StrikeFont(AbstractFont)>>widthAndKernedWidthOfL
    | |    |        |            |    |  |10.1% {1711ms} StrikeFont>>widthOf:
    | |    |        |            |    |  | |3.7% {620ms} StrikeFont>>characterToGlyphMap
    | |    |        |            |    |  | |3.4% {579ms} Character>>charCode
    | |    |        |            |    |  | |3.0% {512ms} primitives
    | |    |        |            |    |  |4.0% {667ms} primitives
    | |    |        |            |    |  |3.3% {564ms} StrikeFont(AbstractFont)>>kerningLeft:right:
    | |    |        |            |    | 4.4% {748ms} TextStopConditions>>at:
    | |    |        |            |    | 3.0% {501ms} primitives
    | |    |        |            |   1.7% {282ms} MultiDisplayScanner>>setStopConditions
    | |    |        |            |    1.6% {269ms} MultiDisplayScanner>>setFont
    | |    |        |           8.6% {1447ms} MultiNewParagraph>>displayExtraSelectionOn:
    | |    |        |            8.5% {1439ms} MultiNewParagraph>>buildSelectionBlocksFrom:to:
```

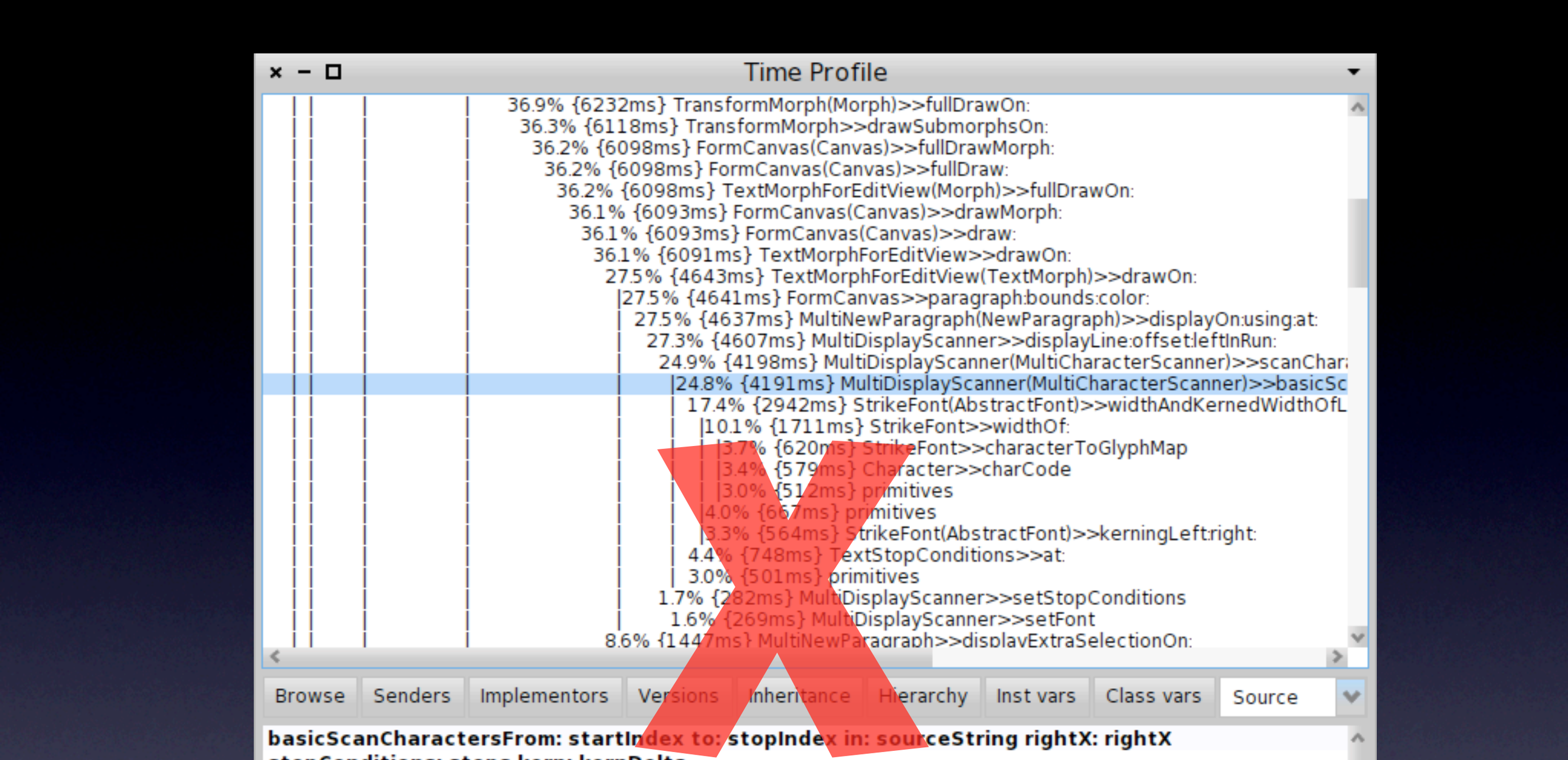
Time Profile

```
36.9% {6232ms} TransformMorph(Morph)>>fullDrawOn:
 36.3% {6118ms} TransformMorph>>drawSubmorphsOn:
  36.2% {6098ms} FormCanvas(Canvas)>>fullDrawMorph:
   36.2% {6098ms} FormCanvas(Canvas)>>fullDraw:
    36.2% {6098ms} TextMorphForEditView(Morph)>>fullDrawOn:
     36.1% {6093ms} FormCanvas(Canvas)>>drawMorph:
      36.1% {6093ms} FormCanvas(Canvas)>>draw:
       36.1% {6091ms} TextMorphForEditView>>drawOn:
        27.5% {4643ms} TextMorphForEditView(TextMorph)>>drawOn:
         27.5% {4641ms} FormCanvas>>paragraph:bounds:color:
          27.5% {4637ms} MultiNewParagraph(NewParagraph)>>displayOn:using:at:
           27.3% {4607ms} MultiDisplayScanner>>displayLine:offset:leftInRun:
            24.9% {4198ms} MultiDisplayScanner(MultiCharacterScanner)>>scanChar
             24.8% {4191ms} MultiDisplayScanner(MultiCharacterScanner)>>basicSc
              17.4% {2942ms} StrikeFont(AbstractFont)>>widthAndKernedWidthOfL
               10.1% {1711ms} StrikeFont>>widthOf:
                3.7% {620ms} StrikeFont>>characterToGlyphMap
                3.4% {579ms} Character>>charCode
                3.0% {512ms} primitives
               4.0% {667ms} primitives
               3.3% {564ms} StrikeFont(AbstractFont)>>kerningLeft:right:
              4.4% {748ms} TextStopConditions>>at:
              3.0% {501ms} primitives
            1.7% {282ms} MultiDisplayScanner>>setStopConditions
            1.6% {269ms} MultiDisplayScanner>>setFont
          8.6% {1447ms} MultiNewParagraph>>displayExtraSelectionOn:
```

Browse | Senders | Implementors | Versions | Inheritance | Hierarchy | Inst vars | Class vars | Source

basicScanCharactersFrom: startIndex to: stopIndex in: sourceString rightX: rightX stopConditions: stops kern: kernDelta

```
"Primitive. This is the inner loop of text display--but see
scanCharactersFrom: to:rightX: which would get the string,
stopConditions and displaying from the instance. March through source
String from startIndex to stopIndex. If any character is flagged with a
non-nil entry in stops, then return the corresponding value. Determine
width of each character from xTable, indexed by map.
If dextX would exceed rightX, then return stops at: 258.
Advance destX by the width of the character. If stopIndex has been
reached, then return stops at: 257. Optional.
See Object documentation whatIsAPrimitive."
| ascii nextDestX char floatDestX widthAndKernedWidth nextChar atEndOfRun |
<primitive: 103>
lastIndex := startIndex.
floatDestX := destX.
widthAndKernedWidth := Array new: 2.
atEndOfRun := false.
```



basicScanCharactersFrom: startIndex to: stopIndex in: sourceString rightX: rightX stopConditions: stops kern: kernDelta

```
"Primitive. This is the inner loop of text display--but see
scanCharactersFrom: to:rightX: which would get the string,
stopConditions and displaying from the instance. March through source
String from startIndex to stopIndex. If any character is flagged with a
non-nil entry in stops, then return the corresponding value. Determine
width of each character from xTable, indexed by map.
If dextX would exceed rightX, then return stops at: 258.
Advance destX by the width of the character. If stopIndex has been
reached, then return stops at: 257. Optional.
See Object documentation whatIsAPrimitive."
| ascii nextDestX char floatDestX widthAndKernedWidth nextChar atEndOfRun |
<primitive: 103>
lastIndex := startIndex.
floatDestX := destX.
widthAndKernedWidth := Array new: 2.
atEndOfRun := false.
```

Time Profile

```
--------------------------------
Process: (40s)  291241984: nil
--------------------------------
59.1% {9975ms} WorldState>>doOneCycleFor:
 58.0% {9781ms} WorldState>>doOneCycleNowFor:
  |53.7% {9065ms} WorldState>>displayWorldSafely:
  | |53.7% {9065ms} PasteUpMorph>>displayWorld
  | | 53.7% {9065ms} PasteUpMorph>>privateOuterDisplayWorld
  | |  53.7% {9065ms} WorldState>>displayWorld:submorphs:
  | |   42.9% {7237ms} WorldState>>drawWorld:submorphs:invalidAreasOn:
  | |    |42.6% {7184ms} FormCanvas(Canvas)>>fullDrawMorph:
  | |    | 42.6% {7184ms} FormCanvas(Canvas)>>fullDraw:
  | |    |  42.6% {7184ms} SystemWindow(Morph)>>fullDrawOn:
  | |    |   41.8% {7048ms} SystemWindow(Morph)>>drawSubmorphsOn:
  | |    |    41.5% {7007ms} FormCanvas(Canvas)>>fullDrawMorph:
  | |    |     41.5% {7004ms} FormCanvas(Canvas)>>fullDraw:
  | |    |      41.5% {7000ms} PluggableTextMorph(Morph)>>fullDrawOn:
  | |    |       37.0% {6233ms} PluggableTextMorph>>drawSubmorphsOn:
  | |    |        |37.0% {6233ms} PluggableTextMorph(Morph)>>drawSubmorphsOn:
  | |    |        | 37.0% {6233ms} FormCanvas(Canvas)>>fullDrawMorph:
  | |    |        |  36.9% {6232ms} FormCanvas(Canvas)>>fullDraw:
  | |    |        |   36.9% {6232ms} TransformMorph(Morph)>>fullDrawOn:
  | |    |        |    36.3% {6118ms} TransformMorph>>drawSubmorphsOn:
  | |    |        |     36.2% {6098ms} FormCanvas(Canvas)>>fullDrawMorph:
  | |    |        |      36.2% {6098ms} FormCanvas(Canvas)>>fullDraw:
  | |    |        |       36.2% {6098ms} TextMorphForEditView(Morph)>>fullDrawOn:
```

Browse | Senders | Implementors | Versions | Inheritance | Hierarchy | Inst vars | Class vars | Source

drawWorld: aWorld submorphs: submorphs invalidAreasOn: aCanvas

"Redraw the damaged areas of the given canvas and clear the damage list. Return a collection of the areas that
were redrawn."

```
| rectList n morphs rects validList |
rectList := damageRecorder invalidRectsFullBounds: aWorld viewBox.
"sort by areas to draw largest portions first"
rectList := rectList asArray sort: [:r1 :r2 | r1 area > r2 area].
damageRecorder reset.
n := submorphs size.
morphs := OrderedCollection new: n * 2.
rects := OrderedCollection new: n * 2.
validList := OrderedCollection new: n * 2.
rectList do:
        [:dirtyRect |
        dirtyRect allAreasOutsideList: validList
```

Time Profile

```
--------------------------------
Process: (40s)  291241984: nil
--------------------------------
59.1% {9975ms} WorldState>>doOneCycleFor:
 58.0% {9781ms} WorldState>>doOneCycleNowFor:
  |53.7% {9065ms} WorldState>>displayWorldSafely:
  | |53.7% {9065ms} PasteUpMorph>>displayWorld
  | | 53.7% {9065ms} PasteUpMorph>>privateOuterDisplayWorld
  | |  53.7% {9065ms} WorldState>>displayWorld:submorphs:
  | |   42.9% {7237ms} WorldState>>drawWorld:submorphs:invalidAreasOn:
  | |    |42.6% {7184ms} FormCanvas(Canvas)>>fullDrawMorph:
  | |    | 42.6% {7184ms} FormCanvas(Canvas)>>fullDraw:
  | |    |  42.6% {7184ms} SystemWindow(Morph)>>fullDrawOn:
  | |    |   41.8% {7048ms} SystemWindow(Morph)>>drawSubmorphsOn:
  | |    |    41.5% {7007ms} FormCanvas(Canvas)>>fullDrawMorph:
  | |    |     41.5% {7004ms} FormCanvas(Canvas)>>fullDraw:
  | |    |      41.5% {7000ms} PluggableTextMorph(Morph)>>fullDrawOn:
  | |    |       37.0% {6233ms} PluggableTextMorph>>drawSubmorphsOn:
  | |    |        |37.0% {6233ms} PluggableTextMorph(Morph)>>drawSubmorphsOn:
  | |    |        | 37.0% {6233ms} FormCanvas(Canvas)>>fullDrawMorph:
  | |    |        |  36.9% {6232ms} FormCanvas(Canvas)>>fullDraw:
  | |    |        |   36.9% {6232ms} TransformMorph(Morph)>>fullDrawOn:
  | |    |        |    36.3% {6118ms} TransformMorph>>drawSubmorphsOn:
  | |    |        |     36.2% {6098ms} FormCanvas(Canvas)>>fullDrawMorph:
  | |    |        |      36.2% {6098ms} FormCanvas(Canvas)>>fullDraw:
  | |    |        |       36.2% {6098ms} TextMorphForEditView(Morph)>>fullDrawOn:
```

Browse | Senders | Implementors | Versions | Inheritance | Hierarchy | Inst vars | Class vars | Source

drawWorld: aWorld submorphs: submorphs invalidAreasOn: aCanvas

"Redraw the damaged areas of the given canvas and clear the damage list. Return a collection of the areas that
were redrawn."

```
| rectList n morphs rects validList |
rectList := damageRecorder invalidRectsFullBounds: aWorld viewBox.
"sort by areas to draw largest portions first"
rectList := rectList asArray sort: [:r1 :r2 | r1 area > r2 area].
damageRecorder reset.
n := submorphs size.
morphs := OrderedCollection new: n * 2.
rects := OrderedCollection new: n * 2.
validList := OrderedCollection new: n * 2.
rectList do:
        [:dirtyRect |
        dirtyRect allAreasOutsideList: validList
```

Measurements!

- timeToRun
- 1 to: many do:
- Regression testing?

- In-process optimization tools adequate
- Post-process validation tools nonexistent

- I’ve found my hotspots, now what?

Tests!

- "You can make it run as fast as you want, as long as you don't need it to be correct"
- Never start optimizing without them

Some common techniques

- Culling
- Caching
- Loop invariant extraction
- Avoiding allocation
- Change datastructures

Coding time...

```
String >> subStrings: separators
        "Answer an array containing the substrings in the receiver separated
        by the elements of separators."
        | result sourceStream subStringStream |
        (separators isString or: [ separators allSatisfy: [ :element | element isKindOf: Character ] ])
                ifFalse: [ ^ self error: 'separators must be Characters.' ].
        sourceStream := self readStream.
        result := OrderedCollection new.
        subStringStream := String new writeStream.
        [ sourceStream atEnd ] whileFalse: [
                | char |
                char := sourceStream next.
                (separators includes: char)
                        ifTrue: [
                                subStringStream isEmpty ifFalse: [
                                        result add: subStringStream contents.
                                        subStringStream := String new writeStream ] ]
                        ifFalse: [
                                subStringStream nextPut: char ] ].
subStringStream isEmpty ifFalse: [
        result add: subStringStream contents ].
^ result asArray
```