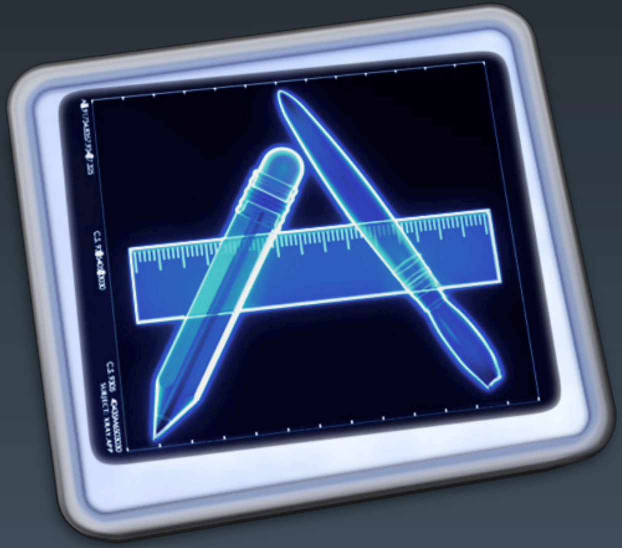
What’s New in Instruments



Steve Lewallen
Engineering Manager for Performance Tools

What You’ll Learn

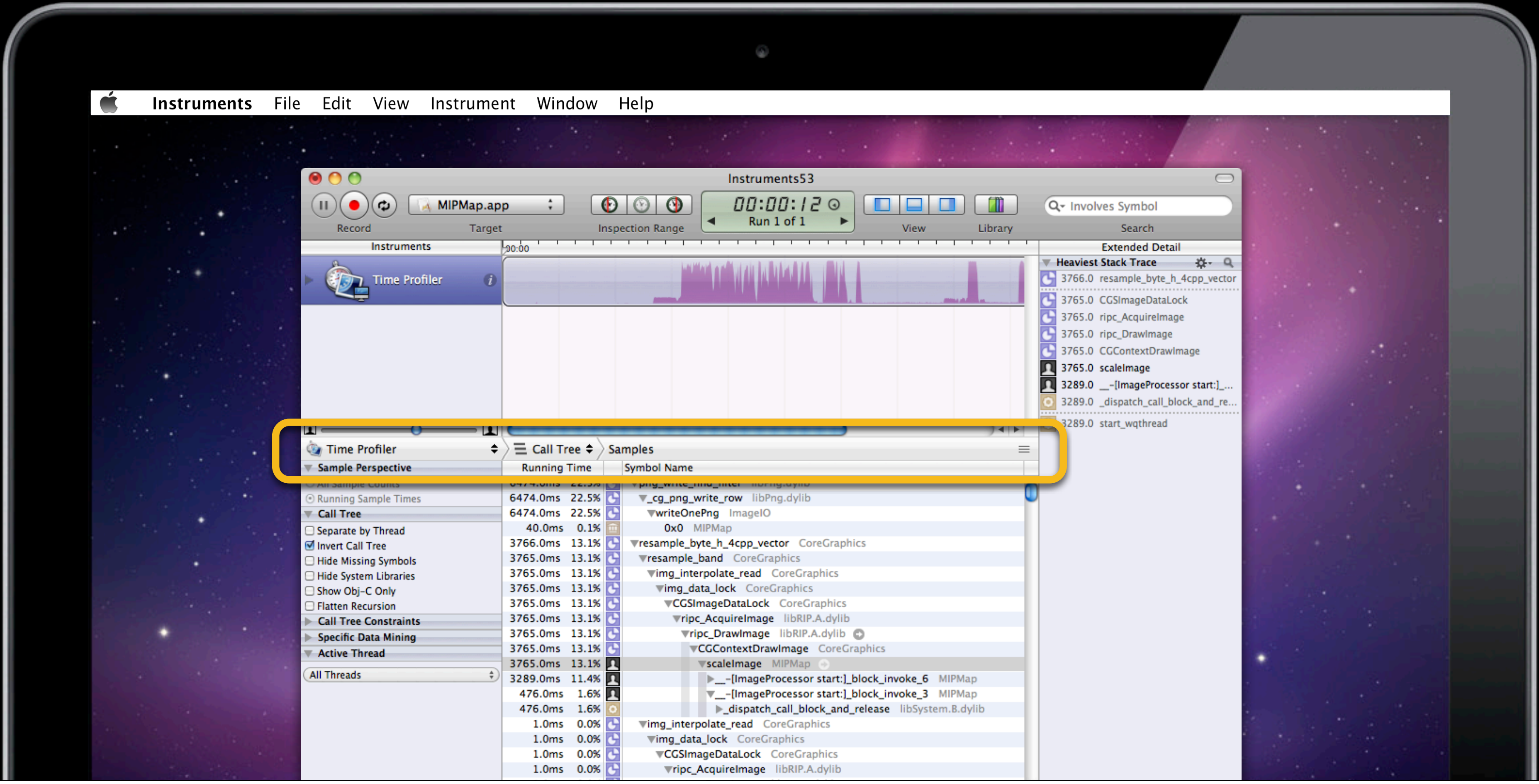
- Navigating the improved user interface
- Significant recording techniques
- Advancements to existing Instruments
- Significant new Instruments

Instruments User Interface

An easier to understand, more capable, streamlined workflow

The Jump Bar

Where you are and how you got there



```
3766.0  resample_byte_h_4cpp_vector
3765.0  CGSImageDataLock
3765.0  ripc_AcquireImage
3765.0  ripc_DrawImage
3765.0  CGContextDrawImage
3765.0  scaleImage
3289.0  __-[ImageProcessor start:]_...
3289.0  _dispatch_call_block_and_re...
3289.0  start_wqthread
```

Time Profiler 〉 Call Tree 〉 Samples

Sample Perspective

○ All Sample Counts

● Running Sample Times

Call Tree

- □ Separate by Thread
- ☑ Invert Call Tree
- □ Hide Missing Symbols
- □ Hide System Libraries
- □ Show Obj-C Only
- □ Flatten Recursion

Call Tree Constraints

Specific Data Mining

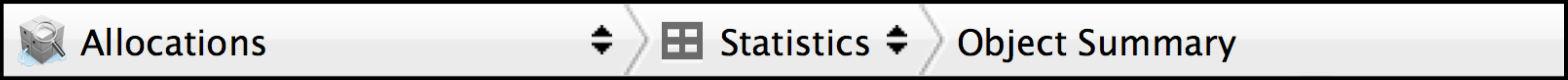
Active Thread

All Threads

```
Running Time      Symbol Name
6474.0ms  22.5%   ▼png_write_find_filter  libPng.dylib
6474.0ms  22.5%     ▼_cg_png_write_row  libPng.dylib
6474.0ms  22.5%       ▼writeOnePng  ImageIO
  40.0ms   0.1%          0x0  MIPMap
3766.0ms  13.1%   ▼resample_byte_h_4cpp_vector  CoreGraphics
3765.0ms  13.1%     ▼resample_band  CoreGraphics
3765.0ms  13.1%       ▼img_interpolate_read  CoreGraphics
3765.0ms  13.1%         ▼img_data_lock  CoreGraphics
3765.0ms  13.1%           ▼CGSImageDataLock  CoreGraphics
3765.0ms  13.1%             ▼ripc_AcquireImage  libRIP.A.dylib
3765.0ms  13.1%               ▼ripc_DrawImage  libRIP.A.dylib
3765.0ms  13.1%                 ▼CGContextDrawImage  CoreGraphics
3765.0ms  13.1%                   ▼scaleImage  MIPMap
3289.0ms  11.4%                     ▶__-[ImageProcessor start:]_block_invoke_6  MIPMap
 476.0ms   1.6%                     ▼__-[ImageProcessor start:]_block_invoke_3  MIPMap
 476.0ms   1.6%                       ▶_dispatch_call_block_and_release  libSystem.B.dylib
   1.0ms   0.0%     ▼img_interpolate_read  CoreGraphics
   1.0ms   0.0%       ▼img_data_lock  CoreGraphics
   1.0ms   0.0%         ▼CGSImageDataLock  CoreGraphics
   1.0ms   0.0%           ▼ripc_AcquireImage  libRIP.A.dylib
```

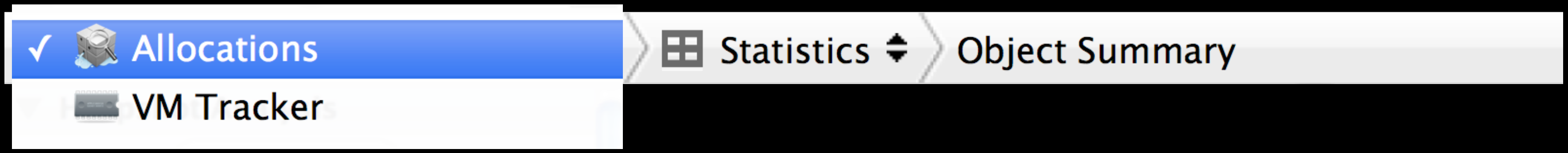
The Jump Bar

Where you are and how you got there



The Jump Bar

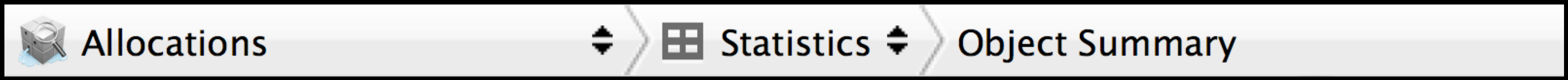
Where you are and how you got there



Navigate Between Instruments

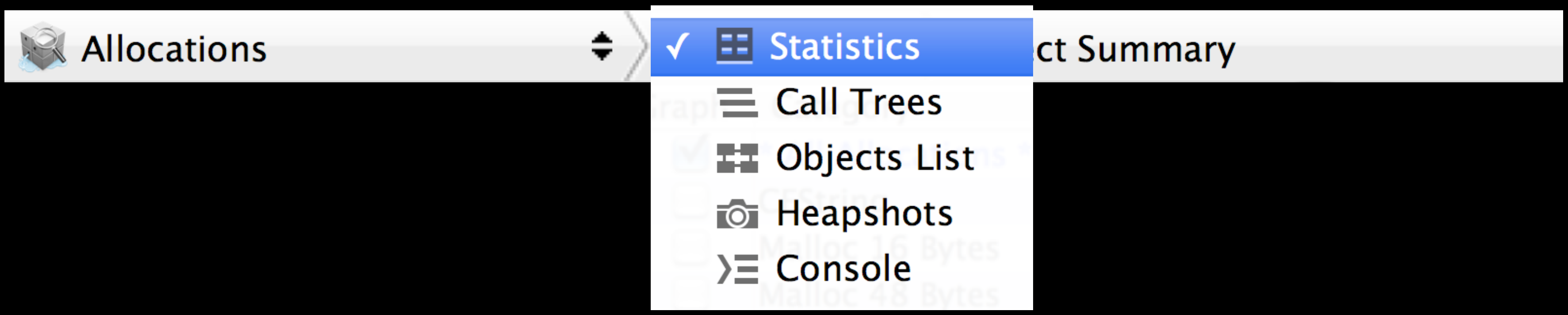
The Jump Bar

Where you are and how you got there



The Jump Bar

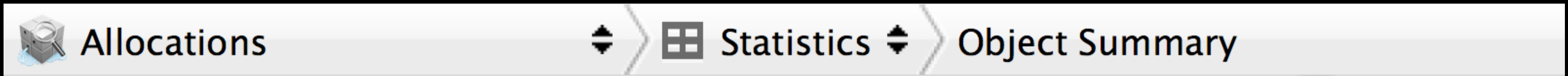
Where you are and how you got there



Navigate Between Detail Views

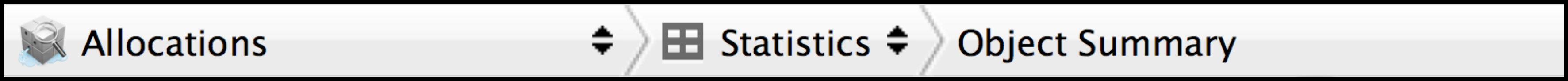
The Jump Bar

Where you are and how you got there



The Jump Bar

Where you are and how you got there

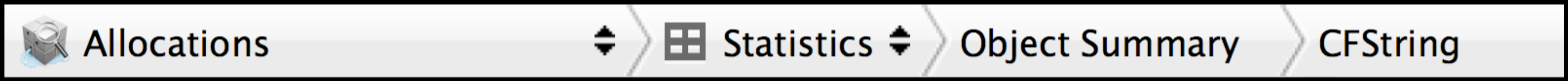


CFString ➔

And after focusing on an item…

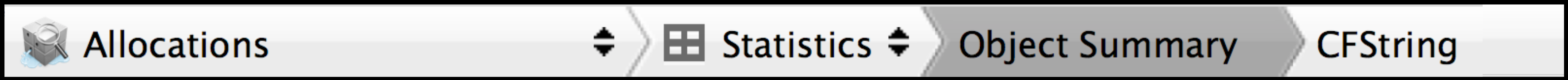
The Jump Bar

Where you are and how you got there



The Jump Bar

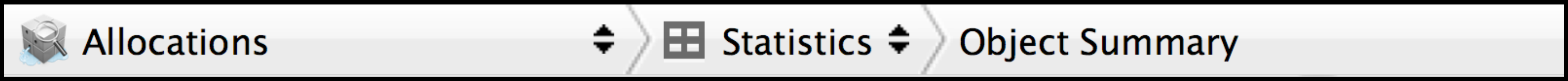
Where you are and how you got there



…navigate back to previous detail view

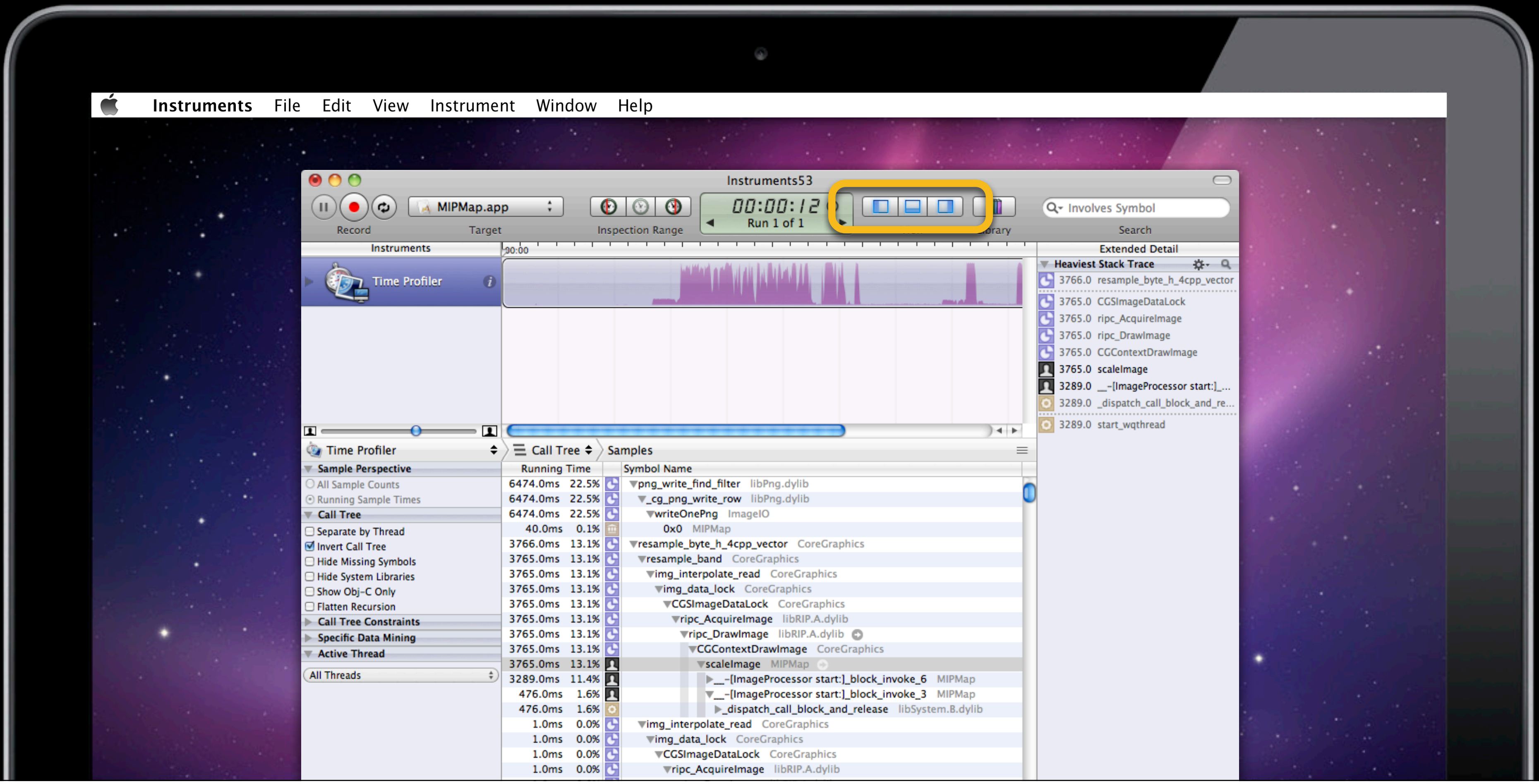
The Jump Bar

Where you are and how you got there



Improved View Access

See what you want, not what you don’t



Time Profiler

Sample Perspective

○ All Sample Counts
● Running Sample Times

Call Tree

□ Separate by Thread
☑ Invert Call Tree
□ Hide Missing Symbols
□ Hide System Libraries
□ Show Obj-C Only
□ Flatten Recursion

Call Tree Constraints

Specific Data Mining

Active Thread

All Threads

Call Tree › Samples

Running Time		Symbol Name
6474.0ms	22.5%	▼png_write_find_filter libPng.dylib
6474.0ms	22.5%	▼_cg_png_write_row libPng.dylib
6474.0ms	22.5%	▼writeOnePng ImageIO
40.0ms	0.1%	0x0 MIPMap
3766.0ms	13.1%	▼resample_byte_h_4cpp_vector CoreGraphics
3765.0ms	13.1%	▼resample_band CoreGraphics
3765.0ms	13.1%	▼img_interpolate_read CoreGraphics
3765.0ms	13.1%	▼img_data_lock CoreGraphics
3765.0ms	13.1%	▼CGSImageDataLock CoreGraphics
3765.0ms	13.1%	▼ripc_AcquireImage libRIP.A.dylib
3765.0ms	13.1%	▼ripc_DrawImage libRIP.A.dylib
3765.0ms	13.1%	▼CGContextDrawImage CoreGraphics
3765.0ms	13.1%	▼scaleImage MIPMap
3289.0ms	11.4%	▶__-[ImageProcessor start:]_block_invoke_6 MIPMap
476.0ms	1.6%	▼__-[ImageProcessor start:]_block_invoke_3 MIPMap
476.0ms	1.6%	▶_dispatch_call_block_and_release libSystem.B.dylib
1.0ms	0.0%	▼img_interpolate_read CoreGraphics
1.0ms	0.0%	▼img_data_lock CoreGraphics
1.0ms	0.0%	▼CGSImageDataLock CoreGraphics
1.0ms	0.0%	▼ripc_AcquireImage libRIP.A.dylib

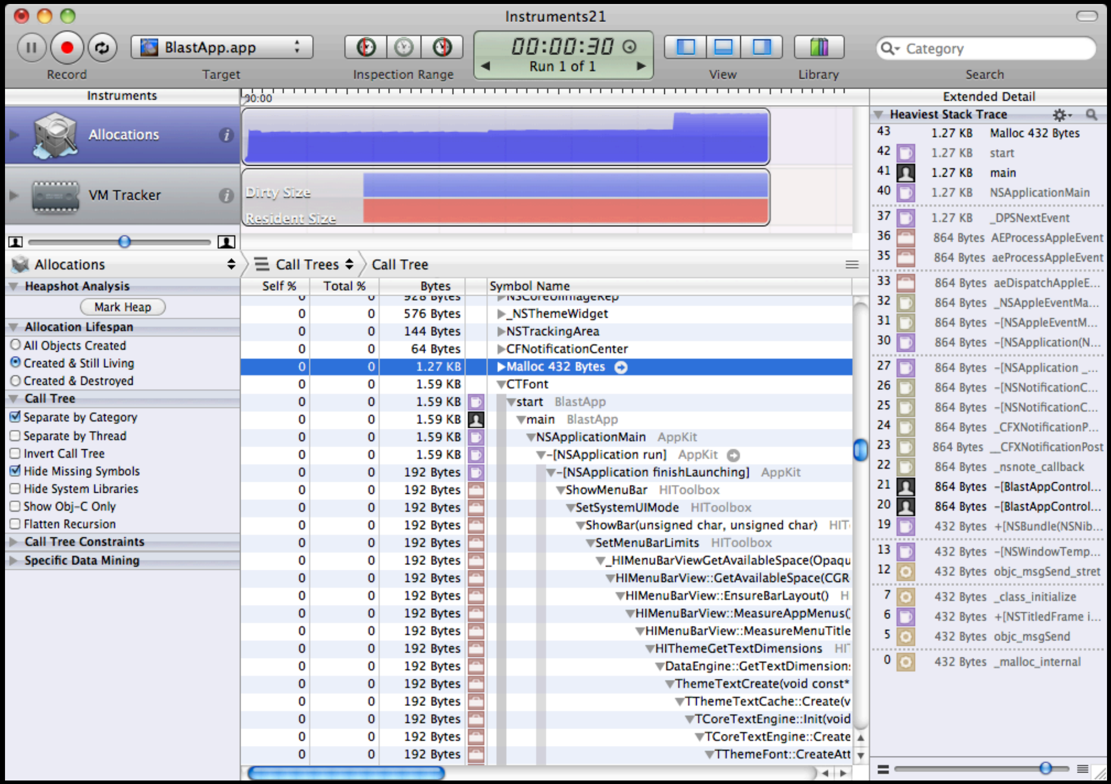
Heaviest Stack Trace

3766.0	resample_byte_h_4cpp_vector
3765.0	CGSImageDataLock
3765.0	ripc_AcquireImage
3765.0	ripc_DrawImage
3765.0	CGContextDrawImage
3765.0	scaleImage
3289.0	__-[ImageProcessor start:]_...
3289.0	_dispatch_call_block_and_re...
3289.0	start_wqthread

Improved View Access

See what you want, not what you don't

- Collapsible views



Allocations

Heapshot Analysis

Mark Heap

Allocation Lifespan

○ All Objects Created
● Created & Still Living
○ Created & Destroyed

Call Tree

☑ Separate by Category
□ Separate by Thread
□ Invert Call Tree
☑ Hide Missing Symbols
□ Hide System Libraries
□ Show Obj-C Only
□ Flatten Recursion

Call Tree Constraints

Specific Data Mining

Self %	Total %	Bytes	Symbol Name
0	0	576 Bytes	▶_NSThemeWidget
0	0	144 Bytes	▶NSTrackingArea
0	0	64 Bytes	▶CFNotificationCenter
0	0	1.27 KB	▶Malloc 432 Bytes
0	0	1.59 KB	▼CTFont
0	0	1.59 KB	▼start BlastApp
0	0	1.59 KB	▼main BlastApp
0	0	1.59 KB	▼NSApplicationMain AppKit
0	0	1.59 KB	▼-[NSApplication run] AppKit
0	0	192 Bytes	▼-[NSApplication finishLaunching] AppKit
0	0	192 Bytes	▼ShowMenuBar HIToolbox
0	0	192 Bytes	▼SetSystemUIMode HIToolbox
0	0	192 Bytes	▼ShowBar(unsigned char, unsigned char) HIT
0	0	192 Bytes	▼SetMenuBarLimits HIToolbox
0	0	192 Bytes	▼_HIMenuBarViewGetAvailableSpace(Opaqu
0	0	192 Bytes	▼HIMenuBarView::GetAvailableSpace(CGR
0	0	192 Bytes	▼HIMenuBarView::EnsureBarLayout() H
0	0	192 Bytes	▼HIMenuBarView::MeasureAppMenus(
0	0	192 Bytes	▼HIMenuBarView::MeasureMenuTitle
0	0	192 Bytes	▼HIThemeGetTextDimensions HI
0	0	192 Bytes	▼DataEngine::GetTextDimension:
0	0	192 Bytes	▼ThemeTextCreate(void const*
0	0	192 Bytes	▼TThemeTextCache::Create(v
0	0	192 Bytes	▼TCoreTextEngine::Init(void
0	0	192 Bytes	▼TCoreTextEngine::Create
0	0	192 Bytes	▼TThemeFont::CreateAtt

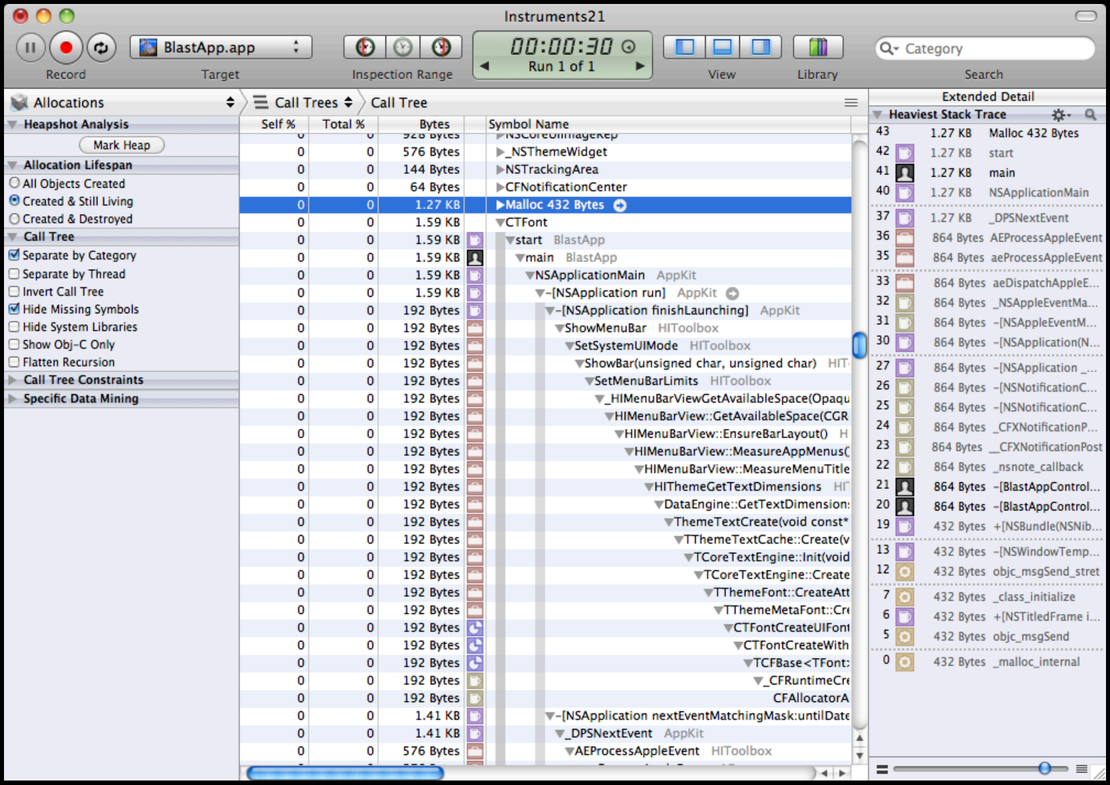
Heaviest Stack Trace

43	1.27 KB	Malloc 432 Bytes
42	1.27 KB	start
41	1.27 KB	main
40	1.27 KB	NSApplicationMain
37	1.27 KB	_DPSNextEvent
36	864 Bytes	AEProcessAppleEvent
35	864 Bytes	aeProcessAppleEvent
33	864 Bytes	aeDispatchAppleE...
32	864 Bytes	_NSAppleEventMa...
31	864 Bytes	-[NSAppleEventM...
30	864 Bytes	-[NSApplication(N...
27	864 Bytes	-[NSApplication _...
26	864 Bytes	-[NSNotificationC...
25	864 Bytes	-[NSNotificationC...
24	864 Bytes	_CFXNotificationP...
23	864 Bytes	__CFXNotificationPost
22	864 Bytes	_nsnote_callback
21	864 Bytes	-[BlastAppControl...
20	864 Bytes	-[BlastAppControl...
19	432 Bytes	+[NSBundle(NSNib...
13	432 Bytes	-[NSWindowTemp...
12	432 Bytes	objc_msgSend_stret
7	432 Bytes	_class_initialize
6	432 Bytes	+[NSTitledFrame i...
5	432 Bytes	objc_msgSend
0	432 Bytes	_malloc_internal

Improved View Access

See what you want, not what you don’t

- Collapsable views
  - Track



Heapshot Analysis

Mark Heap

Allocation Lifespan

- ○ All Objects Created
- ● Created & Still Living
- ○ Created & Destroyed

Call Tree

- ☑ Separate by Category
- □ Separate by Thread
- □ Invert Call Tree
- ☑ Hide Missing Symbols
- □ Hide System Libraries
- □ Show Obj-C Only
- □ Flatten Recursion

Call Tree Constraints

Specific Data Mining

```
Self %  Total %       Bytes  Symbol Name
     0        0   576 Bytes  ▶_NSThemeWidget
     0        0   144 Bytes  ▶NSTrackingArea
     0        0    64 Bytes  ▶CFNotificationCenter
     0        0     1.27 KB  ▶Malloc 432 Bytes
     0        0     1.59 KB  ▼CTFont
     0        0     1.59 KB    ▼start   BlastApp
     0        0     1.59 KB     ▼main   BlastApp
     0        0     1.59 KB      ▼NSApplicationMain   AppKit
     0        0     1.59 KB       ▼-[NSApplication run]   AppKit
     0        0   192 Bytes        ▼-[NSApplication finishLaunching]   AppKit
     0        0   192 Bytes         ▼ShowMenuBar   HIToolbox
     0        0   192 Bytes          ▼SetSystemUIMode   HIToolbox
     0        0   192 Bytes           ▼ShowBar(unsigned char, unsigned char)   HIT
     0        0   192 Bytes            ▼SetMenuBarLimits   HIToolbox
     0        0   192 Bytes             ▼_HIMenuBarViewGetAvailableSpace(Opaqu
     0        0   192 Bytes              ▼HIMenuBarView::GetAvailableSpace(CGR
     0        0   192 Bytes               ▼HIMenuBarView::EnsureBarLayout()   H
     0        0   192 Bytes                ▼HIMenuBarView::MeasureAppMenus(
     0        0   192 Bytes                 ▼HIMenuBarView::MeasureMenuTitle
     0        0   192 Bytes                  ▼HIThemeGetTextDimensions   HI
     0        0   192 Bytes                   ▼DataEngine::GetTextDimension:
     0        0   192 Bytes                    ▼ThemeTextCreate(void const*
     0        0   192 Bytes                     ▼TThemeTextCache::Create(v
     0        0   192 Bytes                      ▼TCoreTextEngine::Init(void
     0        0   192 Bytes                       ▼TCoreTextEngine::Create
     0        0   192 Bytes                        ▼TThemeFont::CreateAtt
     0        0   192 Bytes                         ▼TThemeMetaFont::Cre
     0        0   192 Bytes                          ▼CTFontCreateUIFont
     0        0   192 Bytes                           ▼CTFontCreateWith
     0        0   192 Bytes                            ▼TCFBase<TFont:
     0        0   192 Bytes                             ▼_CFRuntimeCre
     0        0   192 Bytes                               CFAllocatorA
     0        0     1.41 KB        ▼-[NSApplication nextEventMatchingMask:untilDate
     0        0     1.41 KB         ▼_DPSNextEvent   AppKit
     0        0   576 Bytes          ▼AEProcessAppleEvent   HIToolbox
```

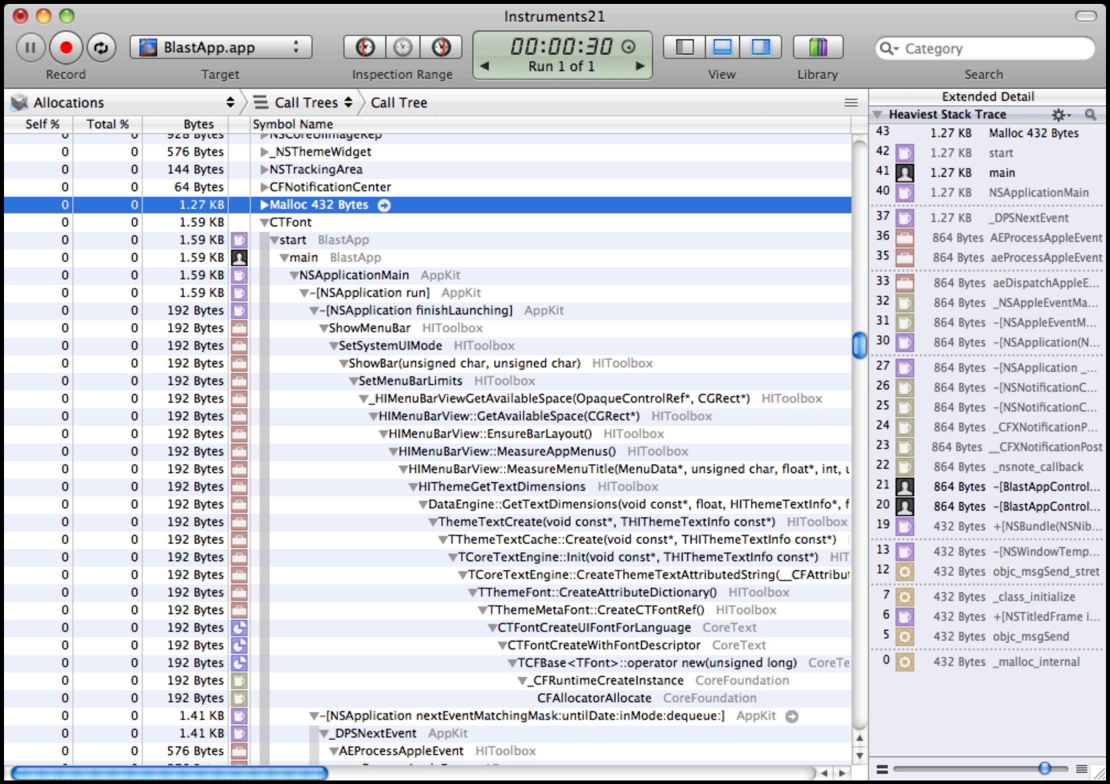
Heaviest Stack Trace

```
43   1.27 KB     Malloc 432 Bytes
42   1.27 KB     start
41   1.27 KB     main
40   1.27 KB     NSApplicationMain
37   1.27 KB     _DPSNextEvent
36   864 Bytes   AEProcessAppleEvent
35   864 Bytes   aeProcessAppleEvent
33   864 Bytes   aeDispatchAppleE...
32   864 Bytes   _NSAppleEventMa...
31   864 Bytes   -[NSAppleEventM...
30   864 Bytes   -[NSApplication(N...
27   864 Bytes   -[NSApplication _...
26   864 Bytes   -[NSNotificationC...
25   864 Bytes   -[NSNotificationC...
24   864 Bytes   _CFXNotificationP...
23   864 Bytes   __CFXNotificationPost
22   864 Bytes   _nsnote_callback
21   864 Bytes   -[BlastAppControl...
20   864 Bytes   -[BlastAppControl...
19   432 Bytes   +[NSBundle(NSNib...
13   432 Bytes   -[NSWindowTemp...
12   432 Bytes   objc_msgSend_stret
 7   432 Bytes   _class_initialize
 6   432 Bytes   +[NSTitledFrame i...
 5   432 Bytes   objc_msgSend
 0   432 Bytes   _malloc_internal
```

Improved View Access

See what you want, not what you don’t

- Collapsable views
  - Track
  - Instrument list



```
Self %  Total %      Bytes   Symbol Name
     0        0  576 Bytes   ▶_NSThemeWidget
     0        0  144 Bytes   ▶NSTrackingArea
     0        0   64 Bytes   ▶CFNotificationCenter
     0        0    1.27 KB   ▶Malloc 432 Bytes
     0        0    1.59 KB   ▼CTFont
     0        0    1.59 KB     ▼start  BlastApp
     0        0    1.59 KB      ▼main  BlastApp
     0        0    1.59 KB       ▼NSApplicationMain  AppKit
     0        0    1.59 KB        ▼-[NSApplication run]  AppKit
     0        0  192 Bytes         ▼-[NSApplication finishLaunching]  AppKit
     0        0  192 Bytes          ▼ShowMenuBar  HIToolbox
     0        0  192 Bytes           ▼SetSystemUIMode  HIToolbox
     0        0  192 Bytes            ▼ShowBar(unsigned char, unsigned char)  HIToolbox
     0        0  192 Bytes             ▼SetMenuBarLimits  HIToolbox
     0        0  192 Bytes              ▼_HIMenuBarViewGetAvailableSpace(OpaqueControlRef*, CGRect*)  HIToolbox
     0        0  192 Bytes               ▼HIMenuBarView::GetAvailableSpace(CGRect*)  HIToolbox
     0        0  192 Bytes                ▼HIMenuBarView::EnsureBarLayout()  HIToolbox
     0        0  192 Bytes                 ▼HIMenuBarView::MeasureAppMenus()  HIToolbox
     0        0  192 Bytes                  ▼HIMenuBarView::MeasureMenuTitle(MenuData*, unsigned char, float*, int, u
     0        0  192 Bytes                   ▼HIThemeGetTextDimensions  HIToolbox
     0        0  192 Bytes                    ▼DataEngine::GetTextDimensions(void const*, float, HIThemeTextInfo*, f
     0        0  192 Bytes                     ▼ThemeTextCreate(void const*, THIThemeTextInfo const*)  HIToolbox
     0        0  192 Bytes                      ▼TThemeTextCache::Create(void const*, THIThemeTextInfo const*)
     0        0  192 Bytes                       ▼TCoreTextEngine::Init(void const*, THIThemeTextInfo const*)  HIT
     0        0  192 Bytes                        ▼TCoreTextEngine::CreateThemeTextAttributedString(__CFAttribut
     0        0  192 Bytes                         ▼TThemeFont::CreateAttributeDictionary()  HIToolbox
     0        0  192 Bytes                          ▼TThemeMetaFont::CreateCTFontRef()  HIToolbox
     0        0  192 Bytes                           ▼CTFontCreateUIFontForLanguage  CoreText
     0        0  192 Bytes                            ▼CTFontCreateWithFontDescriptor  CoreText
     0        0  192 Bytes                             ▼TCFBase<TFont>::operator new(unsigned long)  CoreTe
     0        0  192 Bytes                              ▼_CFRuntimeCreateInstance  CoreFoundation
     0        0  192 Bytes                                CFAllocatorAllocate  CoreFoundation
     0        0    1.41 KB         ▼-[NSApplication nextEventMatchingMask:untilDate:inMode:dequeue:]  AppKit
     0        0    1.41 KB          ▼_DPSNextEvent  AppKit
     0        0  576 Bytes           ▼AEProcessAppleEvent  HIToolbox
```

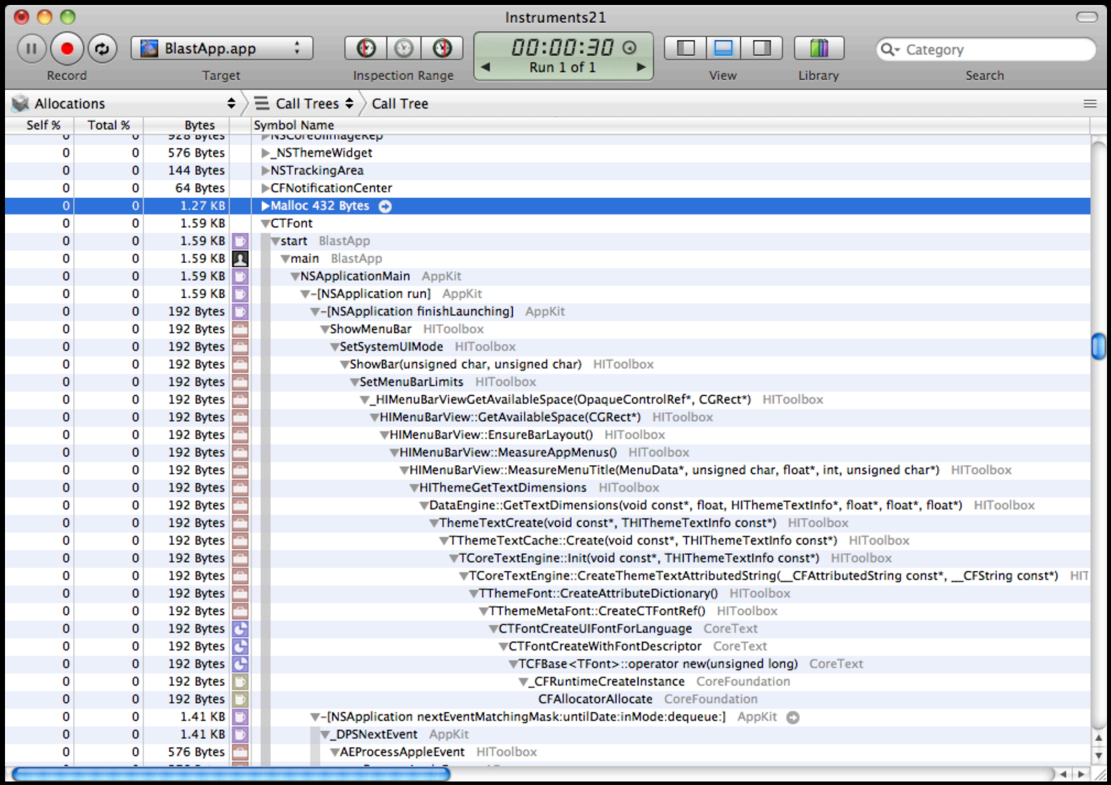
▼ Heaviest Stack Trace

43	1.27 KB	Malloc 432 Bytes
42	1.27 KB	start
41	1.27 KB	main
40	1.27 KB	NSApplicationMain
37	1.27 KB	_DPSNextEvent
36	864 Bytes	AEProcessAppleEvent
35	864 Bytes	aeProcessAppleEvent
33	864 Bytes	aeDispatchAppleE...
32	864 Bytes	_NSAppleEventMa...
31	864 Bytes	-[NSAppleEventM...
30	864 Bytes	-[NSApplication(N...
27	864 Bytes	-[NSApplication _...
26	864 Bytes	-[NSNotificationC...
25	864 Bytes	-[NSNotificationC...
24	864 Bytes	_CFXNotificationP...
23	864 Bytes	__CFXNotificationPost
22	864 Bytes	_nsnote_callback
21	864 Bytes	-[BlastAppControl...
20	864 Bytes	-[BlastAppControl...
19	432 Bytes	+[NSBundle(NSNib...
13	432 Bytes	-[NSWindowTemp...
12	432 Bytes	objc_msgSend_stret
7	432 Bytes	_class_initialize
6	432 Bytes	+[NSTitledFrame i...
5	432 Bytes	objc_msgSend
0	432 Bytes	_malloc_internal

Improved View Access

See what you want, not what you don’t

- Collapsable views
  - Track
  - Instrument list
  - Extended detail view

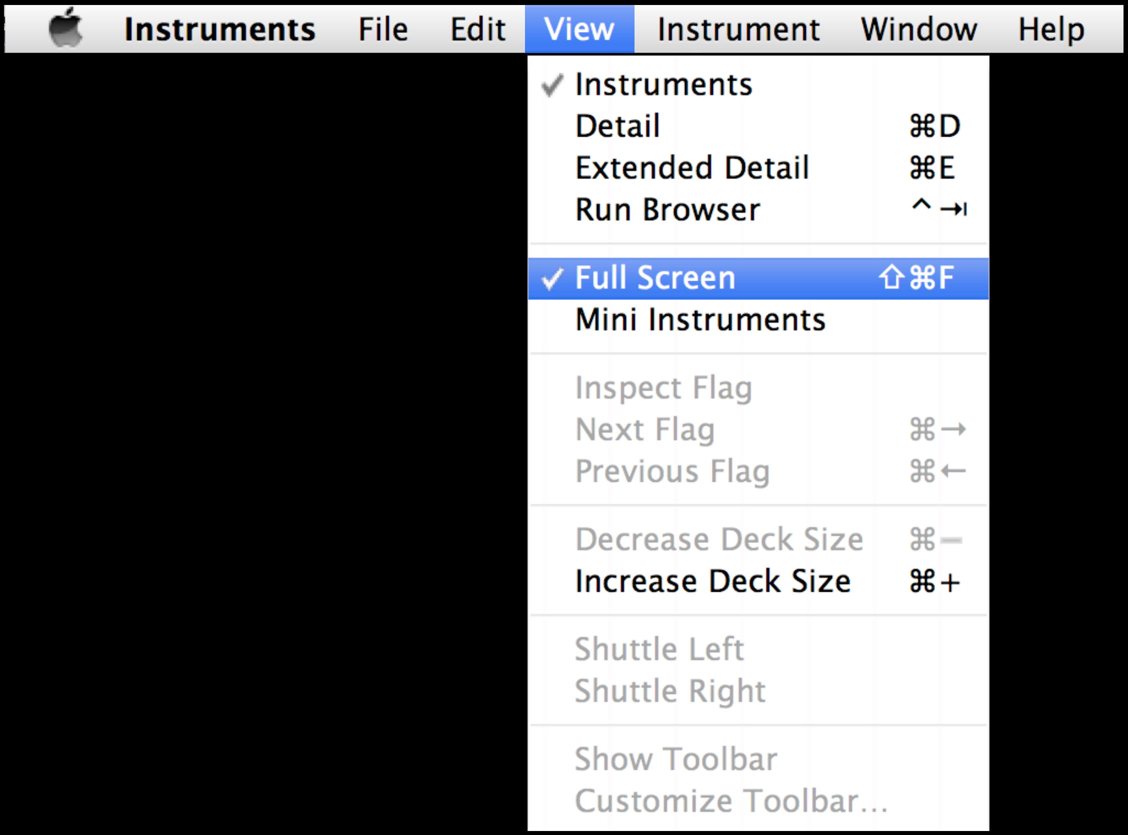


Self %	Total %	Bytes	Symbol Name
0	0	576 Bytes	▶_NSThemeWidget
0	0	144 Bytes	▶NSTrackingArea
0	0	64 Bytes	▶CFNotificationCenter
0	0	1.27 KB	▶Malloc 432 Bytes
0	0	1.59 KB	▼CTFont
0	0	1.59 KB	▼start BlastApp
0	0	1.59 KB	▼main BlastApp
0	0	1.59 KB	▼NSApplicationMain AppKit
0	0	1.59 KB	▼-[NSApplication run] AppKit
0	0	192 Bytes	▼-[NSApplication finishLaunching] AppKit
0	0	192 Bytes	▼ShowMenuBar HIToolbox
0	0	192 Bytes	▼SetSystemUIMode HIToolbox
0	0	192 Bytes	▼ShowBar(unsigned char, unsigned char) HIToolbox
0	0	192 Bytes	▼SetMenuBarLimits HIToolbox
0	0	192 Bytes	▼_HIMenuBarViewGetAvailableSpace(OpaqueControlRef*, CGRect*) HIToolbox
0	0	192 Bytes	▼HIMenuBarView::GetAvailableSpace(CGRect*) HIToolbox
0	0	192 Bytes	▼HIMenuBarView::EnsureBarLayout() HIToolbox
0	0	192 Bytes	▼HIMenuBarView::MeasureAppMenus() HIToolbox
0	0	192 Bytes	▼HIMenuBarView::MeasureMenuTitle(MenuData*, unsigned char, float*, int, unsigned char*) HIToolbox
0	0	192 Bytes	▼HIThemeGetTextDimensions HIToolbox
0	0	192 Bytes	▼DataEngine::GetTextDimensions(void const*, float, HIThemeTextInfo*, float*, float*, float*) HIToolbox
0	0	192 Bytes	▼ThemeTextCreate(void const*, THIThemeTextInfo const*) HIToolbox
0	0	192 Bytes	▼TThemeTextCache::Create(void const*, THIThemeTextInfo const*) HIToolbox
0	0	192 Bytes	▼TCoreTextEngine::Init(void const*, THIThemeTextInfo const*) HIToolbox
0	0	192 Bytes	▼TCoreTextEngine::CreateThemeTextAttributedString(__CFAttributedString const*, __CFString const*) HIT
0	0	192 Bytes	▼TThemeFont::CreateAttributeDictionary() HIToolbox
0	0	192 Bytes	▼TThemeMetaFont::CreateCTFontRef() HIToolbox
0	0	192 Bytes	▼CTFontCreateUIFontForLanguage CoreText
0	0	192 Bytes	▼CTFontCreateWithFontDescriptor CoreText
0	0	192 Bytes	▼TCFBase<TFont>::operator new(unsigned long) CoreText
0	0	192 Bytes	▼_CFRuntimeCreateInstance CoreFoundation
0	0	192 Bytes	CFAllocatorAllocate CoreFoundation
0	0	1.41 KB	▼-[NSApplication nextEventMatchingMask:untilDate:inMode:dequeue:] AppKit
0	0	1.41 KB	▼_DPSNextEvent AppKit
0	0	576 Bytes	▼AEProcessAppleEvent HIToolbox

Improved View Access

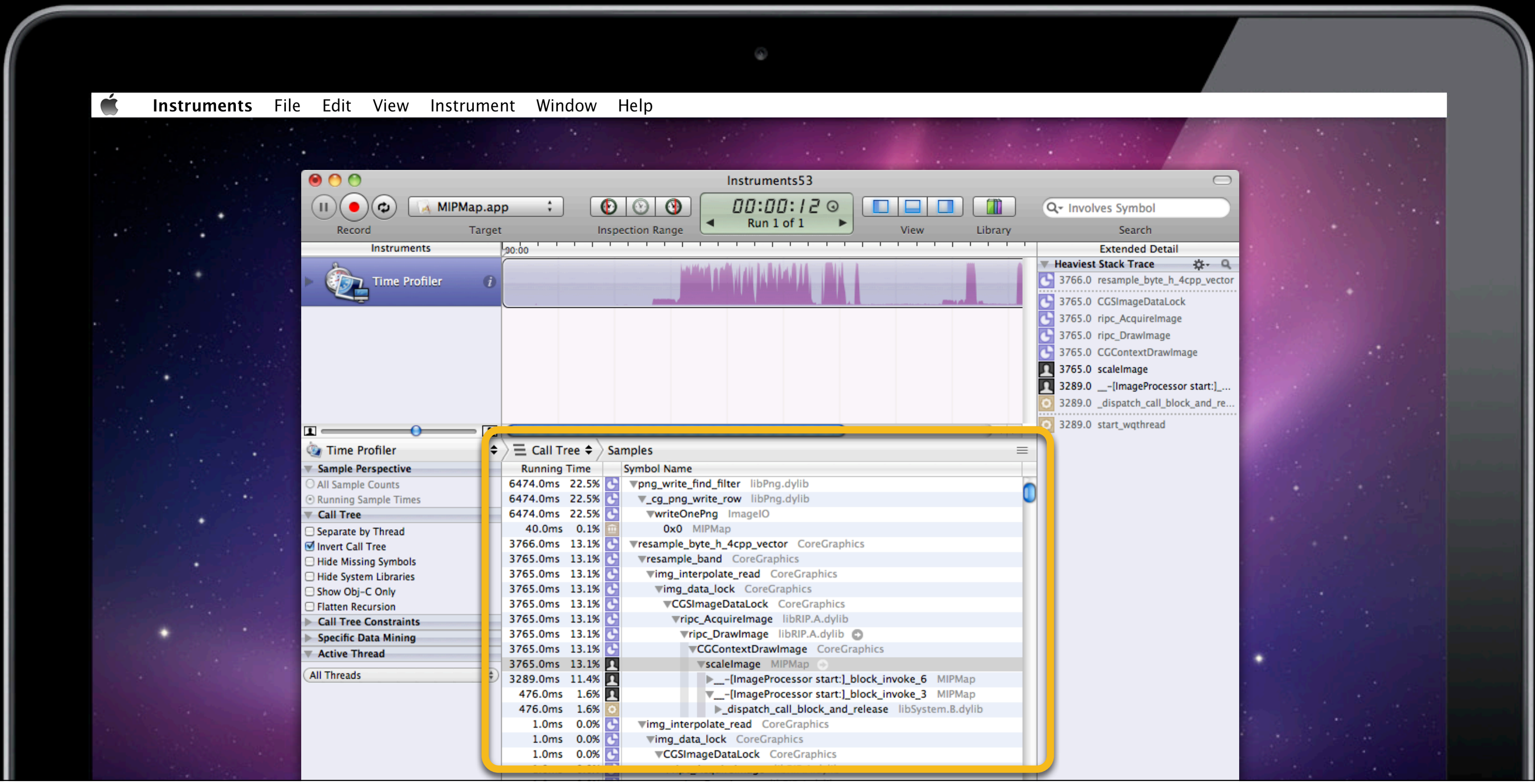
See what you want, not what you don’t

- Full screen
  - Now with menu bar



The Call Tree

The coolest call tree in the universe



Time Profiler

Sample Perspective
- ○ All Sample Counts
- ● Running Sample Times

Call Tree
- □ Separate by Thread
- ☑ Invert Call Tree
- □ Hide Missing Symbols
- □ Hide System Libraries
- □ Show Obj-C Only
- □ Flatten Recursion

Call Tree Constraints

Specific Data Mining

Active Thread

All Threads

Call Tree › Samples

```
Running Time        Symbol Name
6474.0ms  22.5%     ▼png_write_find_filter  libPng.dylib
6474.0ms  22.5%       ▼_cg_png_write_row  libPng.dylib
6474.0ms  22.5%         ▼writeOnePng  ImageIO
  40.0ms   0.1%             0x0  MIPMap
3766.0ms  13.1%     ▼resample_byte_h_4cpp_vector  CoreGraphics
3765.0ms  13.1%       ▼resample_band  CoreGraphics
3765.0ms  13.1%         ▼img_interpolate_read  CoreGraphics
3765.0ms  13.1%           ▼img_data_lock  CoreGraphics
3765.0ms  13.1%             ▼CGSImageDataLock  CoreGraphics
3765.0ms  13.1%               ▼ripc_AcquireImage  libRIP.A.dylib
3765.0ms  13.1%                 ▼ripc_DrawImage  libRIP.A.dylib
3765.0ms  13.1%                   ▼CGContextDrawImage  CoreGraphics
3765.0ms  13.1%                     ▼scaleImage  MIPMap
3289.0ms  11.4%                       ▶__-[ImageProcessor start:]_block_invoke_6  MIPMap
 476.0ms   1.6%                       ▼__-[ImageProcessor start:]_block_invoke_3  MIPMap
 476.0ms   1.6%                         ▶_dispatch_call_block_and_release  libSystem.B.dylib
   1.0ms   0.0%       ▼img_interpolate_read  CoreGraphics
   1.0ms   0.0%         ▼img_data_lock  CoreGraphics
   1.0ms   0.0%           ▼CGSImageDataLock  CoreGraphics
```

Heaviest Stack Trace

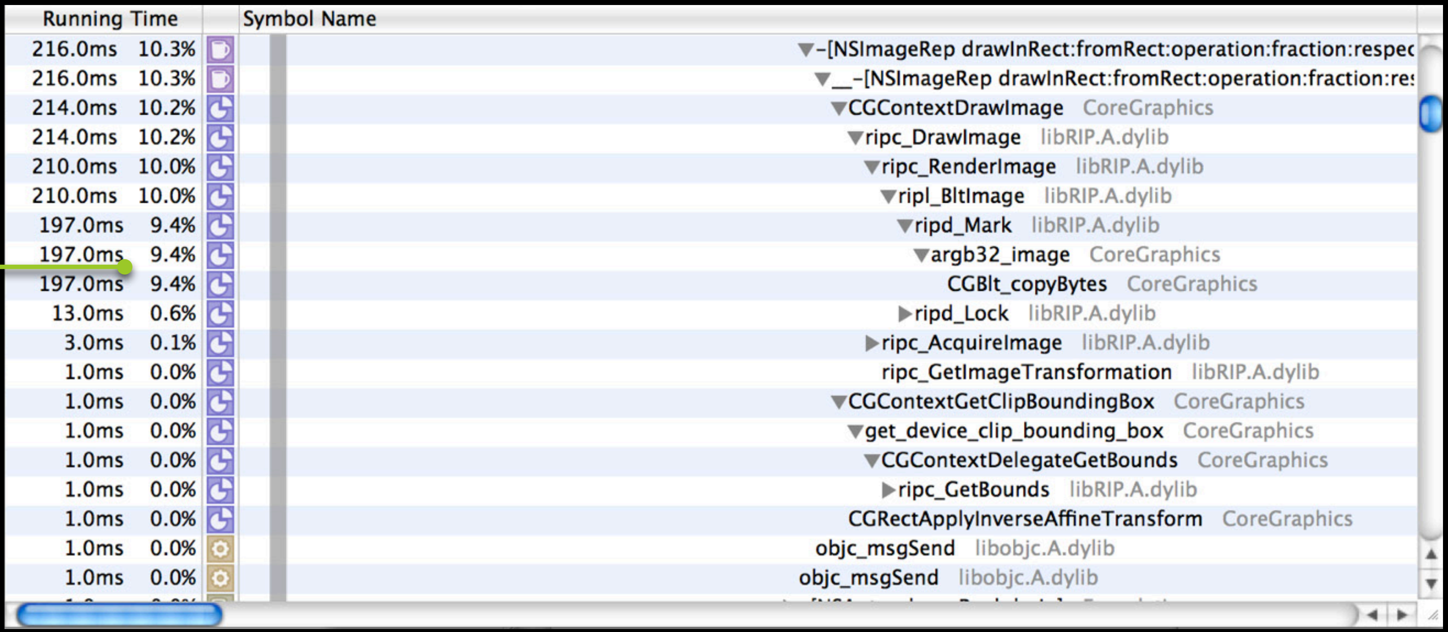
```
3766.0  resample_byte_h_4cpp_vector
3765.0  CGSImageDataLock
3765.0  ripc_AcquireImage
3765.0  ripc_DrawImage
3765.0  CGContextDrawImage
3765.0  scaleImage
3289.0  __-[ImageProcessor start:]_...
3289.0  _dispatch_call_block_and_re...
3289.0  start_wqthread
```

The Call Tree

The coolest call tree in the universe

- More room for symbol tree

Pinned Statistics Columns

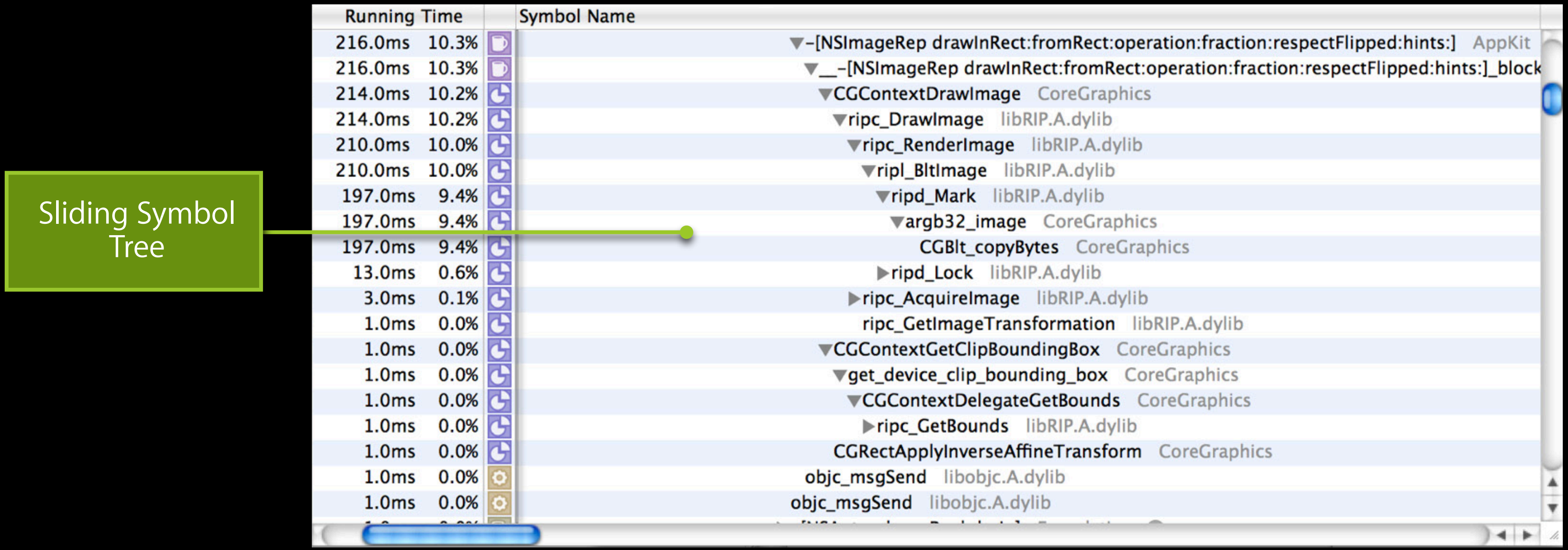


```
Running Time          Symbol Name
216.0ms  10.3%   ▼-[NSImageRep drawInRect:fromRect:operation:fraction:respec
216.0ms  10.3%    ▼__-[NSImageRep drawInRect:fromRect:operation:fraction:re:
214.0ms  10.2%     ▼CGContextDrawImage  CoreGraphics
214.0ms  10.2%      ▼ripc_DrawImage  libRIP.A.dylib
210.0ms  10.0%       ▼ripc_RenderImage  libRIP.A.dylib
210.0ms  10.0%        ▼ripl_BltImage  libRIP.A.dylib
197.0ms   9.4%         ▼ripd_Mark  libRIP.A.dylib
197.0ms   9.4%          ▼argb32_image  CoreGraphics
197.0ms   9.4%             CGBlt_copyBytes  CoreGraphics
 13.0ms   0.6%         ▶ripd_Lock  libRIP.A.dylib
  3.0ms   0.1%       ▶ripc_AcquireImage  libRIP.A.dylib
  1.0ms   0.0%        ripc_GetImageTransformation  libRIP.A.dylib
  1.0ms   0.0%     ▼CGContextGetClipBoundingBox  CoreGraphics
  1.0ms   0.0%      ▼get_device_clip_bounding_box  CoreGraphics
  1.0ms   0.0%       ▼CGContextDelegateGetBounds  CoreGraphics
  1.0ms   0.0%        ▶ripc_GetBounds  libRIP.A.dylib
  1.0ms   0.0%      CGRectApplyInverseAffineTransform  CoreGraphics
  1.0ms   0.0%    objc_msgSend  libobjc.A.dylib
  1.0ms   0.0%   objc_msgSend  libobjc.A.dylib
```

The Call Tree

The coolest call tree in the universe

- More room for symbol tree
  - “Sliding” symbol tree

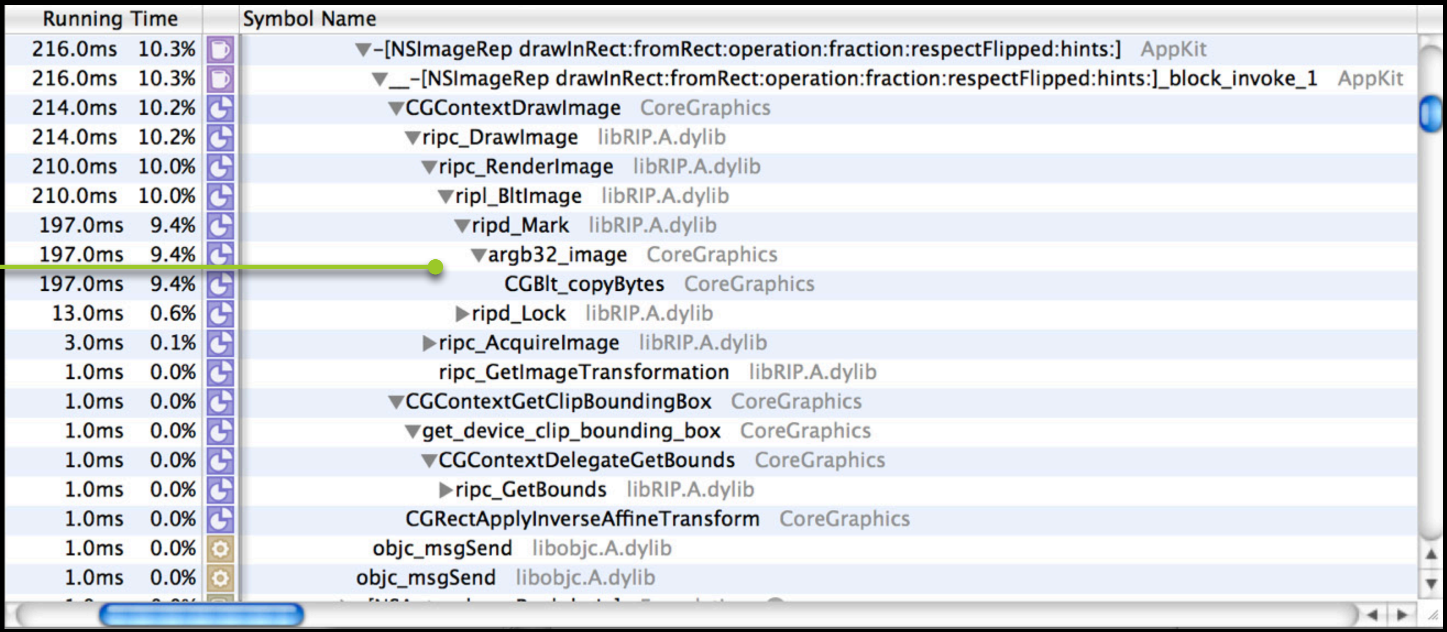


Running Time		Symbol Name
216.0ms	10.3%	▼-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:] AppKit
216.0ms	10.3%	▼__-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]_block
214.0ms	10.2%	▼CGContextDrawImage CoreGraphics
214.0ms	10.2%	▼ripc_DrawImage libRIP.A.dylib
210.0ms	10.0%	▼ripc_RenderImage libRIP.A.dylib
210.0ms	10.0%	▼ripl_BltImage libRIP.A.dylib
197.0ms	9.4%	▼ripd_Mark libRIP.A.dylib
197.0ms	9.4%	▼argb32_image CoreGraphics
197.0ms	9.4%	CGBlt_copyBytes CoreGraphics
13.0ms	0.6%	▶ripd_Lock libRIP.A.dylib
3.0ms	0.1%	▶ripc_AcquireImage libRIP.A.dylib
1.0ms	0.0%	ripc_GetImageTransformation libRIP.A.dylib
1.0ms	0.0%	▼CGContextGetClipBoundingBox CoreGraphics
1.0ms	0.0%	▼get_device_clip_bounding_box CoreGraphics
1.0ms	0.0%	▼CGContextDelegateGetBounds CoreGraphics
1.0ms	0.0%	▶ripc_GetBounds libRIP.A.dylib
1.0ms	0.0%	CGRectApplyInverseAffineTransform CoreGraphics
1.0ms	0.0%	objc_msgSend libobjc.A.dylib
1.0ms	0.0%	objc_msgSend libobjc.A.dylib

The Call Tree

The coolest call tree in the universe

- More room for symbol tree
  - "Sliding" symbol tree



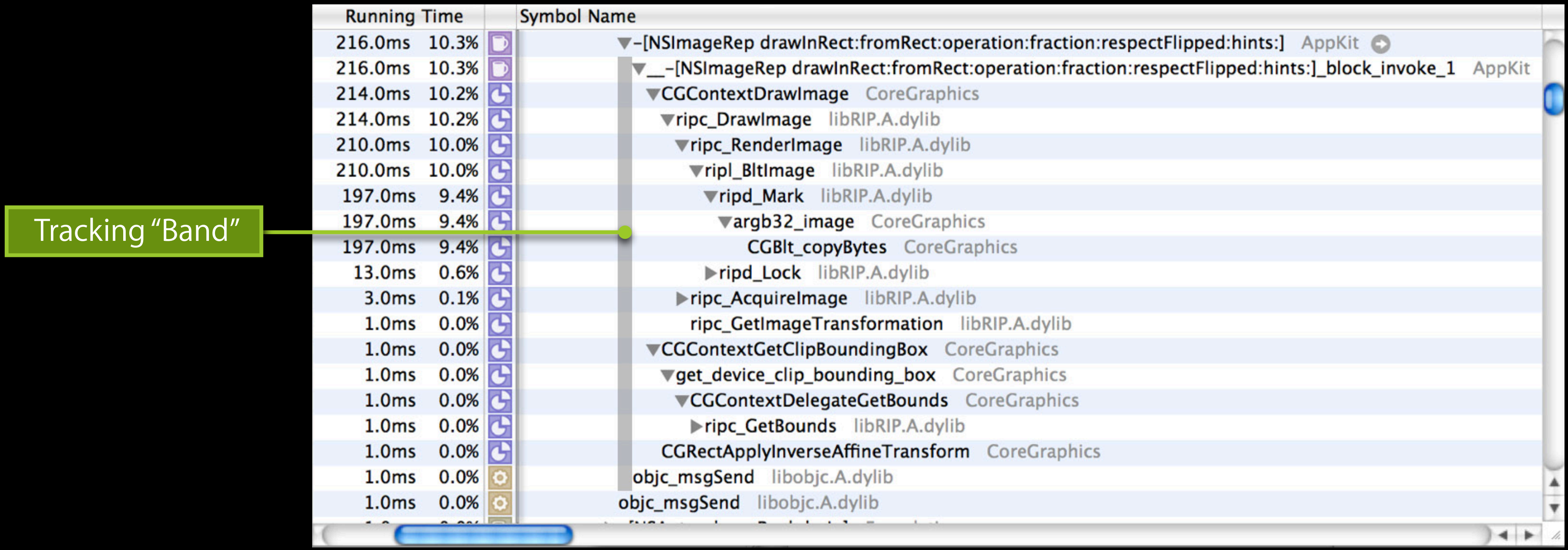
```
Running Time     Symbol Name
216.0ms  10.3%   ▼-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]  AppKit
216.0ms  10.3%     ▼__-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]_block_invoke_1  AppKit
214.0ms  10.2%       ▼CGContextDrawImage  CoreGraphics
214.0ms  10.2%         ▼ripc_DrawImage  libRIP.A.dylib
210.0ms  10.0%           ▼ripc_RenderImage  libRIP.A.dylib
210.0ms  10.0%             ▼ripl_BltImage  libRIP.A.dylib
197.0ms   9.4%               ▼ripd_Mark  libRIP.A.dylib
197.0ms   9.4%                 ▼argb32_image  CoreGraphics
197.0ms   9.4%                     CGBlt_copyBytes  CoreGraphics
 13.0ms   0.6%               ▶ripd_Lock  libRIP.A.dylib
  3.0ms   0.1%           ▶ripc_AcquireImage  libRIP.A.dylib
  1.0ms   0.0%             ripc_GetImageTransformation  libRIP.A.dylib
  1.0ms   0.0%       ▼CGContextGetClipBoundingBox  CoreGraphics
  1.0ms   0.0%         ▼get_device_clip_bounding_box  CoreGraphics
  1.0ms   0.0%           ▼CGContextDelegateGetBounds  CoreGraphics
  1.0ms   0.0%             ▶ripc_GetBounds  libRIP.A.dylib
  1.0ms   0.0%         CGRectApplyInverseAffineTransform  CoreGraphics
  1.0ms   0.0%     objc_msgSend  libobjc.A.dylib
  1.0ms   0.0%   objc_msgSend  libobjc.A.dylib
```

Sliding Symbol Tree

The Call Tree

The coolest call tree in the universe

- More room for symbol tree
  - “Sliding” symbol tree
  - “Sibling Banding”



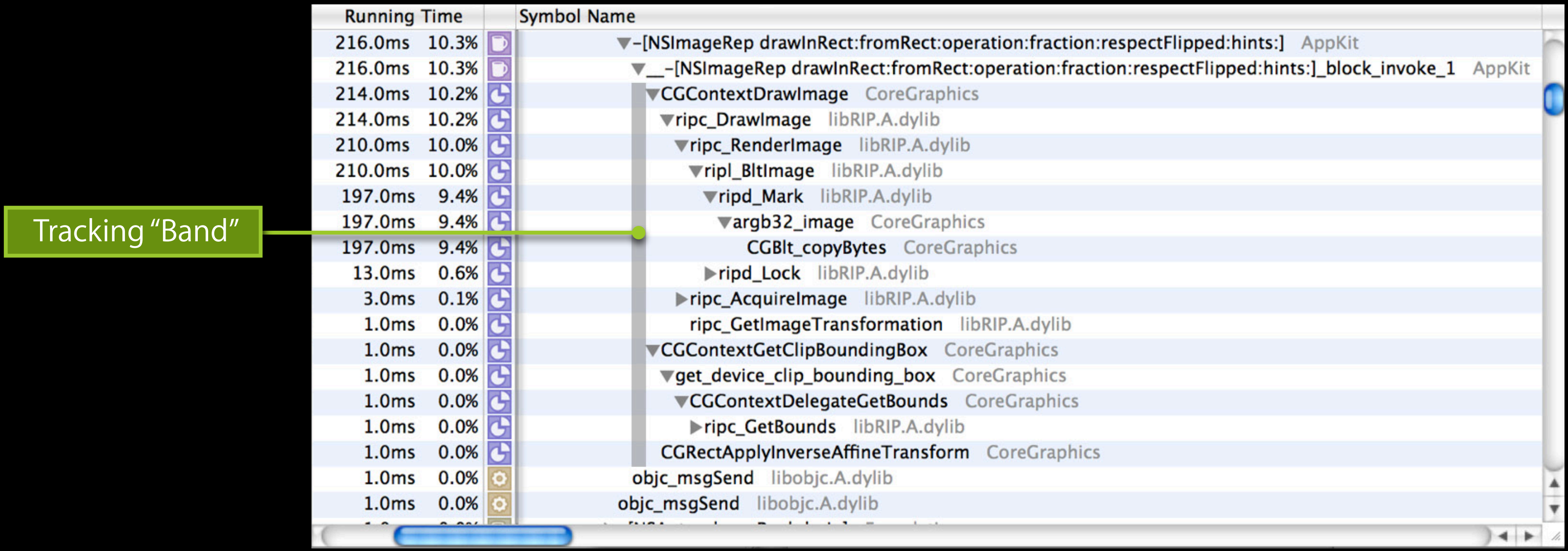
Tracking “Band”

Running Time		Symbol Name	
216.0ms	10.3%	▼-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]	AppKit
216.0ms	10.3%	▼__-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]_block_invoke_1	AppKit
214.0ms	10.2%	▼CGContextDrawImage	CoreGraphics
214.0ms	10.2%	▼ripc_DrawImage	libRIP.A.dylib
210.0ms	10.0%	▼ripc_RenderImage	libRIP.A.dylib
210.0ms	10.0%	▼ripl_BltImage	libRIP.A.dylib
197.0ms	9.4%	▼ripd_Mark	libRIP.A.dylib
197.0ms	9.4%	▼argb32_image	CoreGraphics
197.0ms	9.4%	CGBlt_copyBytes	CoreGraphics
13.0ms	0.6%	▶ripd_Lock	libRIP.A.dylib
3.0ms	0.1%	▶ripc_AcquireImage	libRIP.A.dylib
1.0ms	0.0%	ripc_GetImageTransformation	libRIP.A.dylib
1.0ms	0.0%	▼CGContextGetClipBoundingBox	CoreGraphics
1.0ms	0.0%	▼get_device_clip_bounding_box	CoreGraphics
1.0ms	0.0%	▼CGContextDelegateGetBounds	CoreGraphics
1.0ms	0.0%	▶ripc_GetBounds	libRIP.A.dylib
1.0ms	0.0%	CGRectApplyInverseAffineTransform	CoreGraphics
1.0ms	0.0%	objc_msgSend	libobjc.A.dylib
1.0ms	0.0%	objc_msgSend	libobjc.A.dylib

The Call Tree

The coolest call tree in the universe

- More room for symbol tree
  - “Sliding” symbol tree
  - “Sibling Banding”



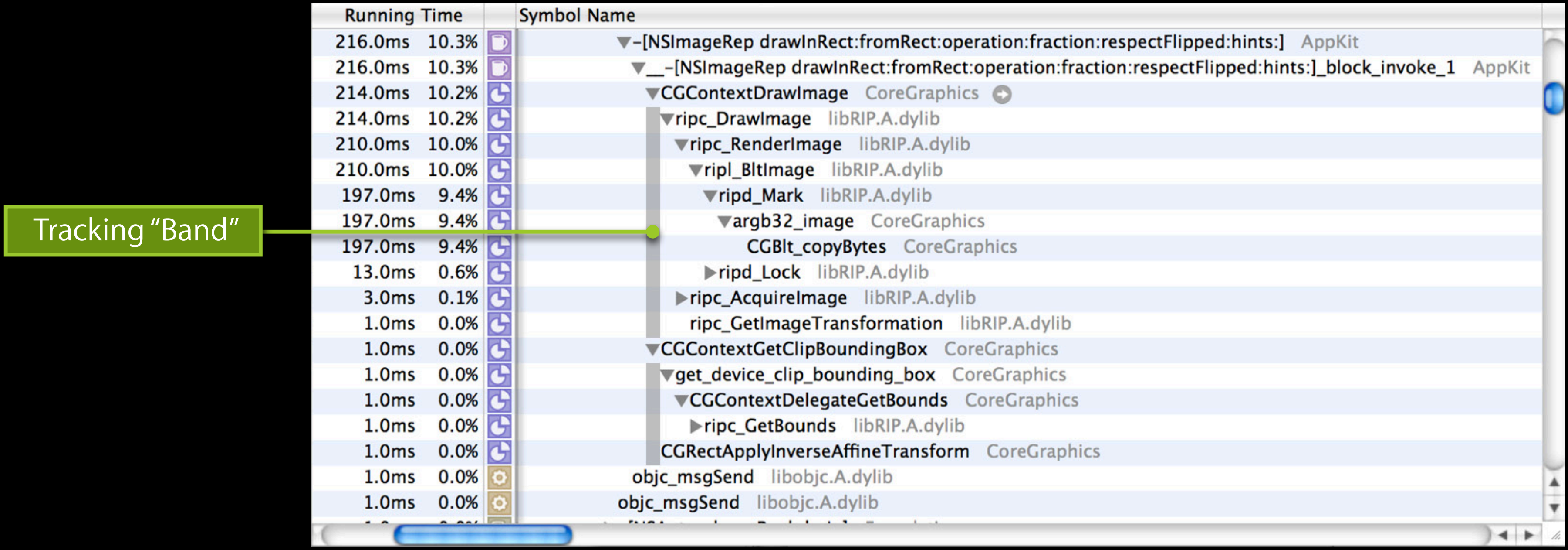
Tracking “Band”

```
Running Time      Symbol Name
216.0ms  10.3%          ▼-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]  AppKit
216.0ms  10.3%            ▼__-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]_block_invoke_1  AppKit
214.0ms  10.2%              ▼CGContextDrawImage  CoreGraphics
214.0ms  10.2%                ▼ripc_DrawImage  libRIP.A.dylib
210.0ms  10.0%                  ▼ripc_RenderImage  libRIP.A.dylib
210.0ms  10.0%                    ▼ripl_BltImage  libRIP.A.dylib
197.0ms   9.4%                      ▼ripd_Mark  libRIP.A.dylib
197.0ms   9.4%                        ▼argb32_image  CoreGraphics
197.0ms   9.4%                            CGBlt_copyBytes  CoreGraphics
 13.0ms   0.6%                      ▶ripd_Lock  libRIP.A.dylib
  3.0ms   0.1%                  ▶ripc_AcquireImage  libRIP.A.dylib
  1.0ms   0.0%                    ripc_GetImageTransformation  libRIP.A.dylib
  1.0ms   0.0%              ▼CGContextGetClipBoundingBox  CoreGraphics
  1.0ms   0.0%                ▼get_device_clip_bounding_box  CoreGraphics
  1.0ms   0.0%                  ▼CGContextDelegateGetBounds  CoreGraphics
  1.0ms   0.0%                    ▶ripc_GetBounds  libRIP.A.dylib
  1.0ms   0.0%                CGRectApplyInverseAffineTransform  CoreGraphics
  1.0ms   0.0%            objc_msgSend  libobjc.A.dylib
  1.0ms   0.0%          objc_msgSend  libobjc.A.dylib
```

The Call Tree

The coolest call tree in the universe

- More room for symbol tree
  - “Sliding” symbol tree
  - “Sibling Banding”

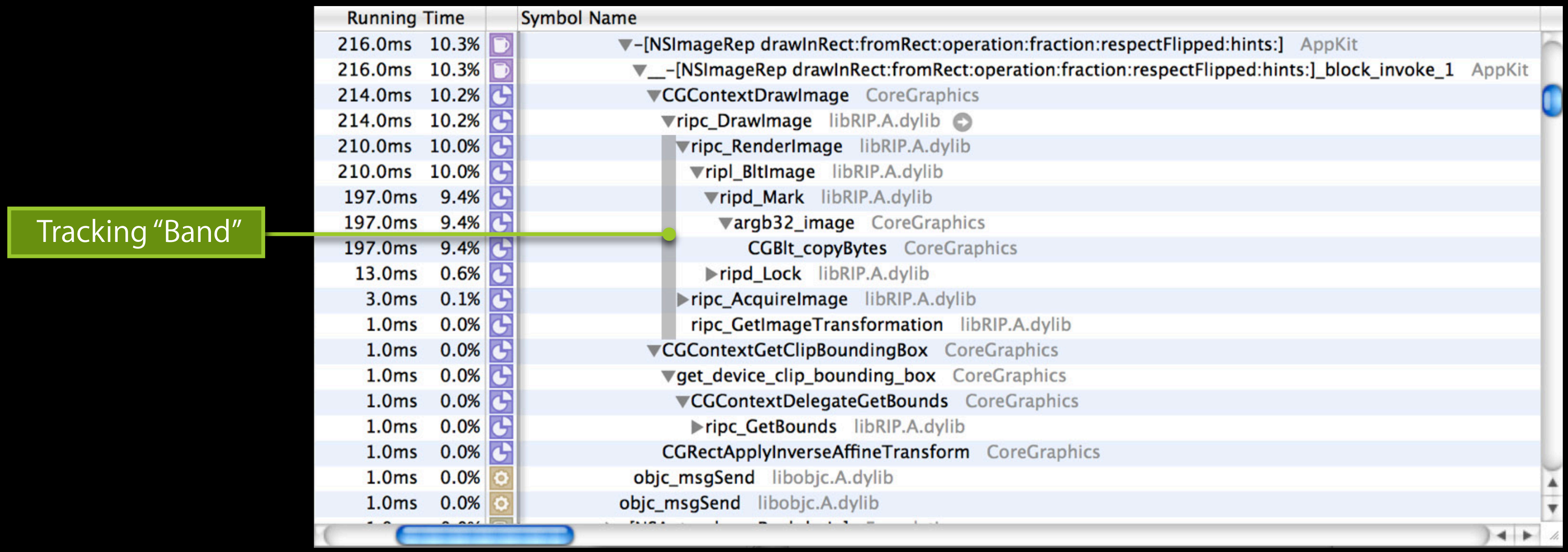


Running Time		Symbol Name
216.0ms	10.3%	▼-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:] AppKit
216.0ms	10.3%	▼__-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]_block_invoke_1 AppKit
214.0ms	10.2%	▼CGContextDrawImage CoreGraphics
214.0ms	10.2%	▼ripc_DrawImage libRIP.A.dylib
210.0ms	10.0%	▼ripc_RenderImage libRIP.A.dylib
210.0ms	10.0%	▼ripl_BltImage libRIP.A.dylib
197.0ms	9.4%	▼ripd_Mark libRIP.A.dylib
197.0ms	9.4%	▼argb32_image CoreGraphics
197.0ms	9.4%	CGBlt_copyBytes CoreGraphics
13.0ms	0.6%	▶ripd_Lock libRIP.A.dylib
3.0ms	0.1%	▶ripc_AcquireImage libRIP.A.dylib
1.0ms	0.0%	ripc_GetImageTransformation libRIP.A.dylib
1.0ms	0.0%	▼CGContextGetClipBoundingBox CoreGraphics
1.0ms	0.0%	▼get_device_clip_bounding_box CoreGraphics
1.0ms	0.0%	▼CGContextDelegateGetBounds CoreGraphics
1.0ms	0.0%	▶ripc_GetBounds libRIP.A.dylib
1.0ms	0.0%	CGRectApplyInverseAffineTransform CoreGraphics
1.0ms	0.0%	objc_msgSend libobjc.A.dylib
1.0ms	0.0%	objc_msgSend libobjc.A.dylib

The Call Tree

The coolest call tree in the universe

- More room for symbol tree
  - “Sliding” symbol tree
  - “Sibling Banding”

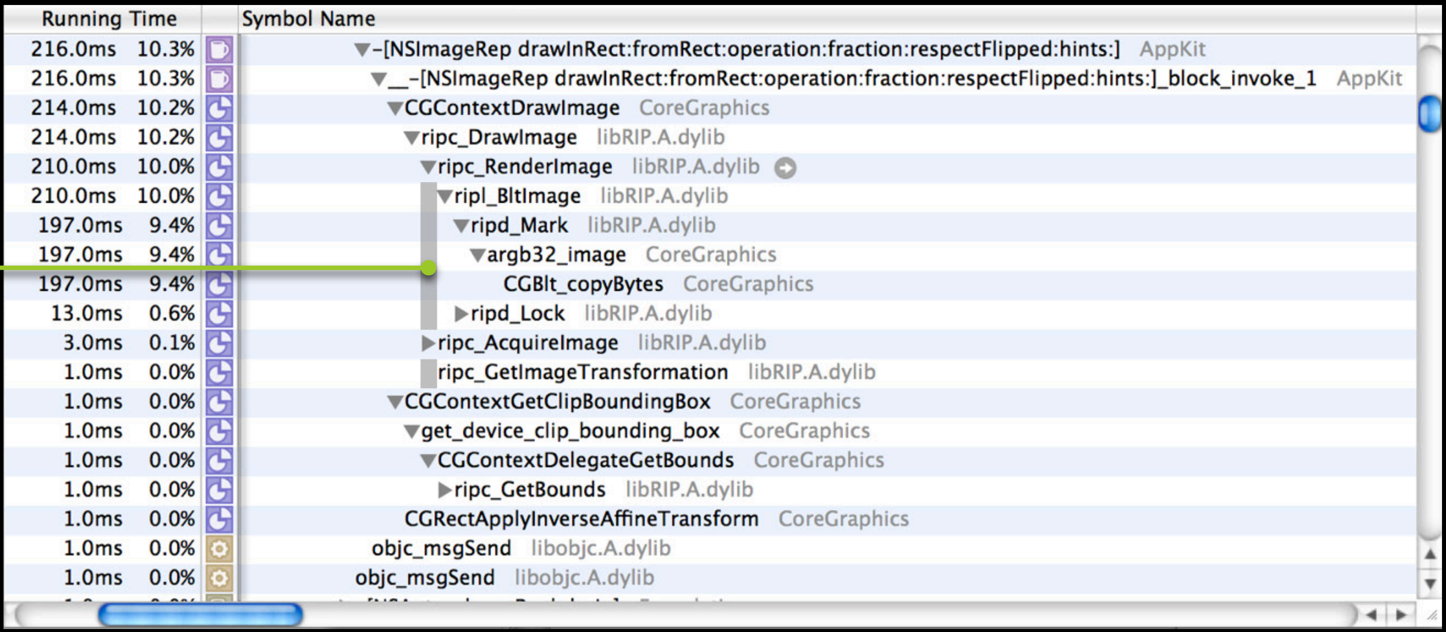


```
Running Time      Symbol Name
216.0ms  10.3%    ▼-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]  AppKit
216.0ms  10.3%     ▼__-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]_block_invoke_1  AppKit
214.0ms  10.2%      ▼CGContextDrawImage  CoreGraphics
214.0ms  10.2%       ▼ripc_DrawImage  libRIP.A.dylib
210.0ms  10.0%        ▼ripc_RenderImage  libRIP.A.dylib
210.0ms  10.0%         ▼ripl_BltImage  libRIP.A.dylib
197.0ms   9.4%          ▼ripd_Mark  libRIP.A.dylib
197.0ms   9.4%           ▼argb32_image  CoreGraphics
197.0ms   9.4%              CGBlt_copyBytes  CoreGraphics
 13.0ms   0.6%          ▶ripd_Lock  libRIP.A.dylib
  3.0ms   0.1%        ▶ripc_AcquireImage  libRIP.A.dylib
  1.0ms   0.0%         ripc_GetImageTransformation  libRIP.A.dylib
  1.0ms   0.0%      ▼CGContextGetClipBoundingBox  CoreGraphics
  1.0ms   0.0%       ▼get_device_clip_bounding_box  CoreGraphics
  1.0ms   0.0%        ▼CGContextDelegateGetBounds  CoreGraphics
  1.0ms   0.0%         ▶ripc_GetBounds  libRIP.A.dylib
  1.0ms   0.0%       CGRectApplyInverseAffineTransform  CoreGraphics
  1.0ms   0.0%     objc_msgSend  libobjc.A.dylib
  1.0ms   0.0%    objc_msgSend  libobjc.A.dylib
```

The Call Tree

The coolest call tree in the universe

- More room for symbol tree
  - “Sliding” symbol tree
  - “Sibling Banding”



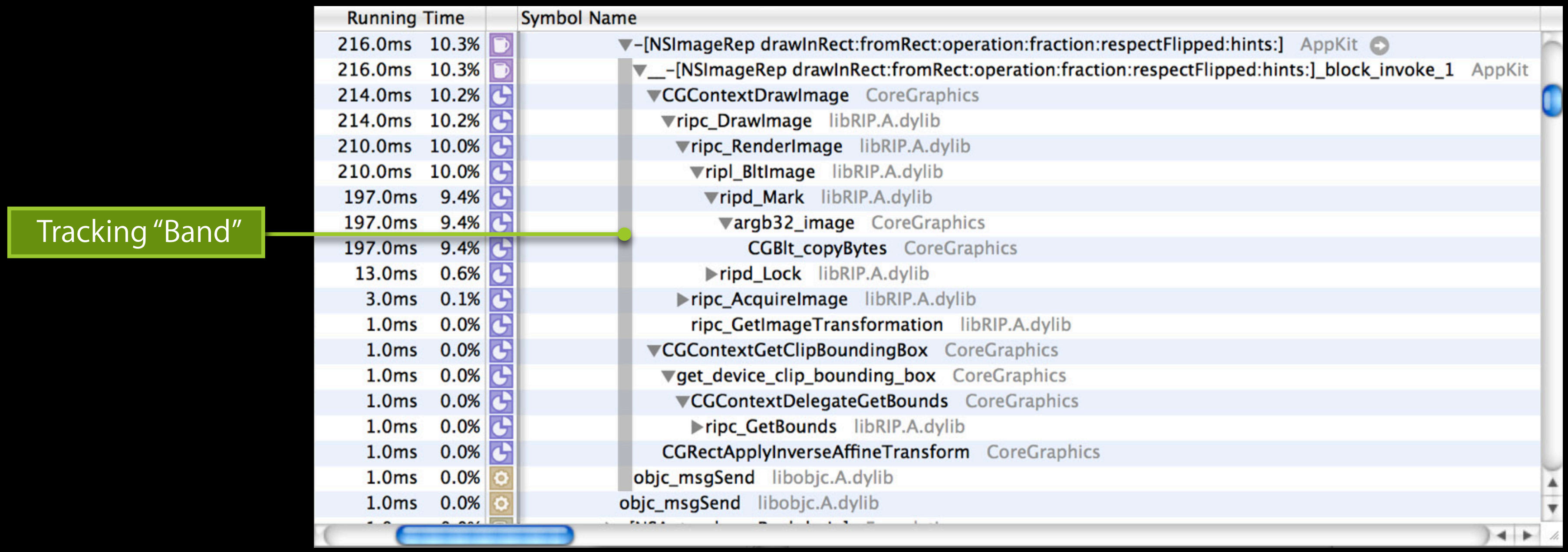
Tracking “Band”

```
Running Time      Symbol Name
216.0ms  10.3%          ▼-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]   AppKit
216.0ms  10.3%            ▼__-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]_block_invoke_1   AppKit
214.0ms  10.2%              ▼CGContextDrawImage   CoreGraphics
214.0ms  10.2%                ▼ripc_DrawImage   libRIP.A.dylib
210.0ms  10.0%                  ▼ripc_RenderImage   libRIP.A.dylib
210.0ms  10.0%                    ▼ripl_BltImage   libRIP.A.dylib
197.0ms   9.4%                      ▼ripd_Mark   libRIP.A.dylib
197.0ms   9.4%                        ▼argb32_image   CoreGraphics
197.0ms   9.4%                            CGBlt_copyBytes   CoreGraphics
 13.0ms   0.6%                      ▶ripd_Lock   libRIP.A.dylib
  3.0ms   0.1%                    ▶ripc_AcquireImage   libRIP.A.dylib
  1.0ms   0.0%                     ripc_GetImageTransformation   libRIP.A.dylib
  1.0ms   0.0%              ▼CGContextGetClipBoundingBox   CoreGraphics
  1.0ms   0.0%                ▼get_device_clip_bounding_box   CoreGraphics
  1.0ms   0.0%                  ▼CGContextDelegateGetBounds   CoreGraphics
  1.0ms   0.0%                    ▶ripc_GetBounds   libRIP.A.dylib
  1.0ms   0.0%               CGRectApplyInverseAffineTransform   CoreGraphics
  1.0ms   0.0%            objc_msgSend   libobjc.A.dylib
  1.0ms   0.0%          objc_msgSend   libobjc.A.dylib
```

The Call Tree

The coolest call tree in the universe

- More room for symbol tree
  - “Sliding” symbol tree
  - “Sibling Banding”

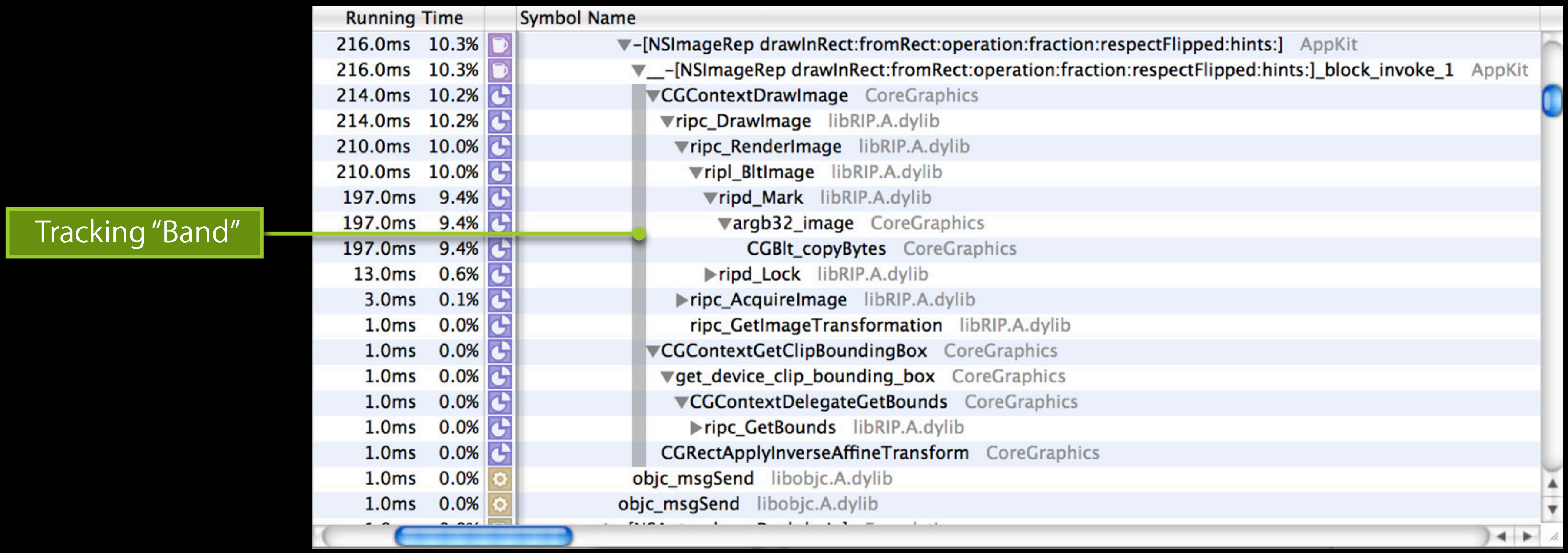


Running Time		Symbol Name
216.0ms	10.3%	▼-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:] AppKit
216.0ms	10.3%	▼__-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]_block_invoke_1 AppKit
214.0ms	10.2%	▼CGContextDrawImage CoreGraphics
214.0ms	10.2%	▼ripc_DrawImage libRIP.A.dylib
210.0ms	10.0%	▼ripc_RenderImage libRIP.A.dylib
210.0ms	10.0%	▼ripl_BltImage libRIP.A.dylib
197.0ms	9.4%	▼ripd_Mark libRIP.A.dylib
197.0ms	9.4%	▼argb32_image CoreGraphics
197.0ms	9.4%	CGBlt_copyBytes CoreGraphics
13.0ms	0.6%	▶ripd_Lock libRIP.A.dylib
3.0ms	0.1%	▶ripc_AcquireImage libRIP.A.dylib
1.0ms	0.0%	ripc_GetImageTransformation libRIP.A.dylib
1.0ms	0.0%	▼CGContextGetClipBoundingBox CoreGraphics
1.0ms	0.0%	▼get_device_clip_bounding_box CoreGraphics
1.0ms	0.0%	▼CGContextDelegateGetBounds CoreGraphics
1.0ms	0.0%	▶ripc_GetBounds libRIP.A.dylib
1.0ms	0.0%	CGRectApplyInverseAffineTransform CoreGraphics
1.0ms	0.0%	objc_msgSend libobjc.A.dylib
1.0ms	0.0%	objc_msgSend libobjc.A.dylib

The Call Tree

The coolest call tree in the universe

- More room for symbol tree
  - “Sliding” symbol tree
  - “Sibling Banding”

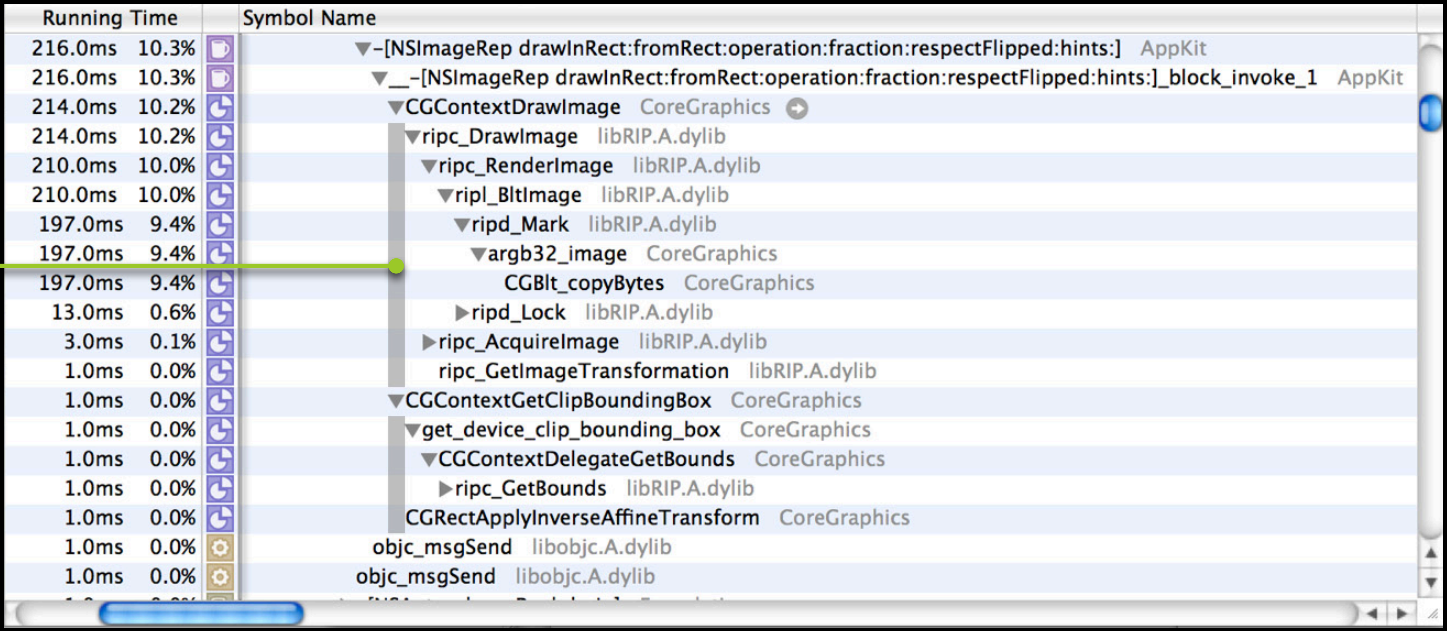


```
Running Time      Symbol Name
216.0ms  10.3%    ▼-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]  AppKit
216.0ms  10.3%      ▼__-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]_block_invoke_1  AppKit
214.0ms  10.2%        ▼CGContextDrawImage  CoreGraphics
214.0ms  10.2%          ▼ripc_DrawImage  libRIP.A.dylib
210.0ms  10.0%            ▼ripc_RenderImage  libRIP.A.dylib
210.0ms  10.0%              ▼ripl_BltImage  libRIP.A.dylib
197.0ms   9.4%                ▼ripd_Mark  libRIP.A.dylib
197.0ms   9.4%                  ▼argb32_image  CoreGraphics
197.0ms   9.4%                      CGBlt_copyBytes  CoreGraphics
 13.0ms   0.6%                ▶ripd_Lock  libRIP.A.dylib
  3.0ms   0.1%            ▶ripc_AcquireImage  libRIP.A.dylib
  1.0ms   0.0%              ripc_GetImageTransformation  libRIP.A.dylib
  1.0ms   0.0%        ▼CGContextGetClipBoundingBox  CoreGraphics
  1.0ms   0.0%          ▼get_device_clip_bounding_box  CoreGraphics
  1.0ms   0.0%            ▼CGContextDelegateGetBounds  CoreGraphics
  1.0ms   0.0%              ▶ripc_GetBounds  libRIP.A.dylib
  1.0ms   0.0%          CGRectApplyInverseAffineTransform  CoreGraphics
  1.0ms   0.0%      objc_msgSend  libobjc.A.dylib
  1.0ms   0.0%    objc_msgSend  libobjc.A.dylib
```

The Call Tree

The coolest call tree in the universe

- More room for symbol tree
  - “Sliding” symbol tree
  - “Sibling Banding”



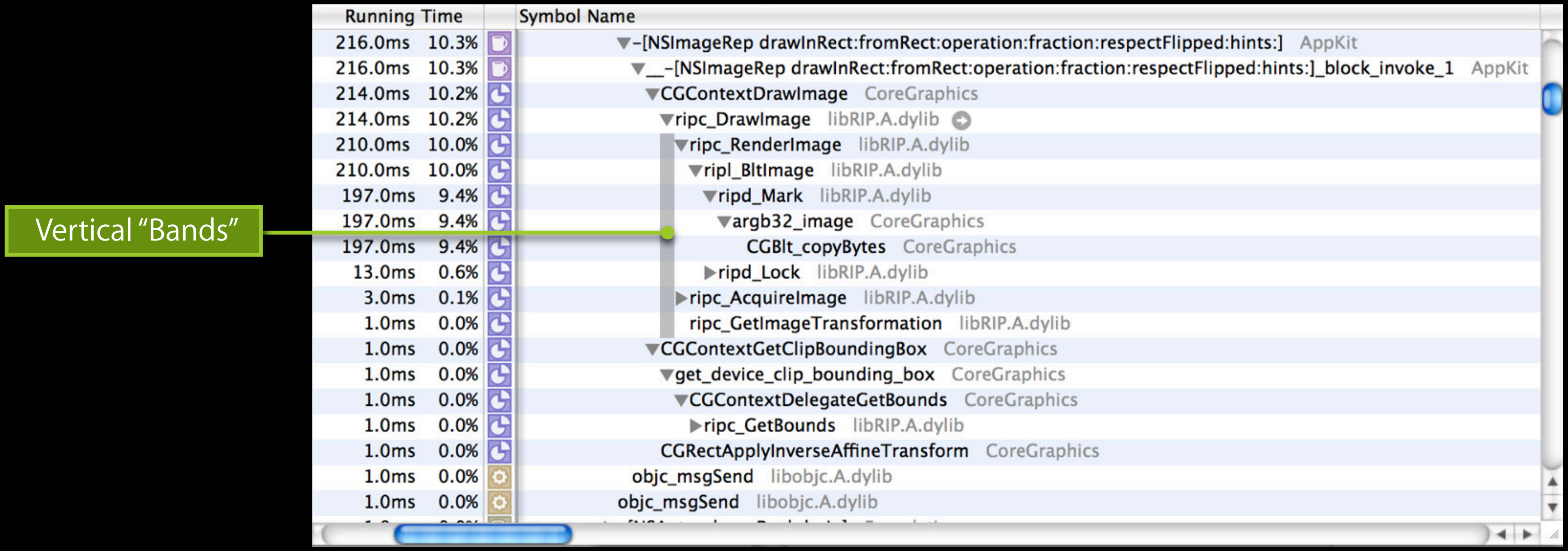
```
Running Time      Symbol Name
216.0ms  10.3%    ▼-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]  AppKit
216.0ms  10.3%     ▼__-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]_block_invoke_1  AppKit
214.0ms  10.2%      ▼CGContextDrawImage  CoreGraphics
214.0ms  10.2%       ▼ripc_DrawImage  libRIP.A.dylib
210.0ms  10.0%        ▼ripc_RenderImage  libRIP.A.dylib
210.0ms  10.0%         ▼ripl_BltImage  libRIP.A.dylib
197.0ms   9.4%          ▼ripd_Mark  libRIP.A.dylib
197.0ms   9.4%           ▼argb32_image  CoreGraphics
197.0ms   9.4%              CGBlt_copyBytes  CoreGraphics
 13.0ms   0.6%          ▶ripd_Lock  libRIP.A.dylib
  3.0ms   0.1%        ▶ripc_AcquireImage  libRIP.A.dylib
  1.0ms   0.0%         ripc_GetImageTransformation  libRIP.A.dylib
  1.0ms   0.0%      ▼CGContextGetClipBoundingBox  CoreGraphics
  1.0ms   0.0%       ▼get_device_clip_bounding_box  CoreGraphics
  1.0ms   0.0%        ▼CGContextDelegateGetBounds  CoreGraphics
  1.0ms   0.0%         ▶ripc_GetBounds  libRIP.A.dylib
  1.0ms   0.0%      CGRectApplyInverseAffineTransform  CoreGraphics
  1.0ms   0.0%     objc_msgSend  libobjc.A.dylib
  1.0ms   0.0%    objc_msgSend  libobjc.A.dylib
```

Vertical “Bands”

The Call Tree

The coolest call tree in the universe

- More room for symbol tree
  - “Sliding” symbol tree
  - “Sibling Banding”



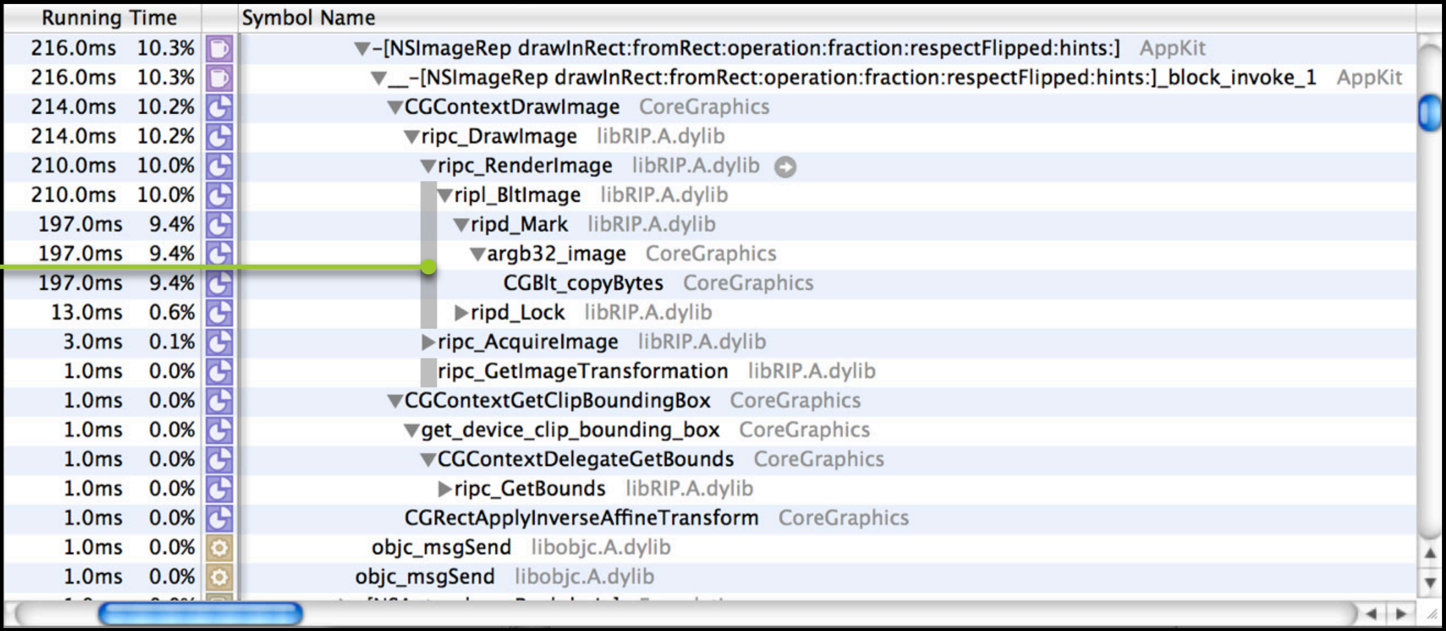
Vertical “Bands”

Running Time		Symbol Name	
216.0ms	10.3%	▼-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]	AppKit
216.0ms	10.3%	▼__-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]_block_invoke_1	AppKit
214.0ms	10.2%	▼CGContextDrawImage	CoreGraphics
214.0ms	10.2%	▼ripc_DrawImage	libRIP.A.dylib
210.0ms	10.0%	▼ripc_RenderImage	libRIP.A.dylib
210.0ms	10.0%	▼ripl_BltImage	libRIP.A.dylib
197.0ms	9.4%	▼ripd_Mark	libRIP.A.dylib
197.0ms	9.4%	▼argb32_image	CoreGraphics
197.0ms	9.4%	CGBlt_copyBytes	CoreGraphics
13.0ms	0.6%	▶ripd_Lock	libRIP.A.dylib
3.0ms	0.1%	▶ripc_AcquireImage	libRIP.A.dylib
1.0ms	0.0%	ripc_GetImageTransformation	libRIP.A.dylib
1.0ms	0.0%	▼CGContextGetClipBoundingBox	CoreGraphics
1.0ms	0.0%	▼get_device_clip_bounding_box	CoreGraphics
1.0ms	0.0%	▼CGContextDelegateGetBounds	CoreGraphics
1.0ms	0.0%	▶ripc_GetBounds	libRIP.A.dylib
1.0ms	0.0%	CGRectApplyInverseAffineTransform	CoreGraphics
1.0ms	0.0%	objc_msgSend	libobjc.A.dylib
1.0ms	0.0%	objc_msgSend	libobjc.A.dylib

The Call Tree

The coolest call tree in the universe

- More room for symbol tree
  - “Sliding” symbol tree
  - “Sibling Banding”

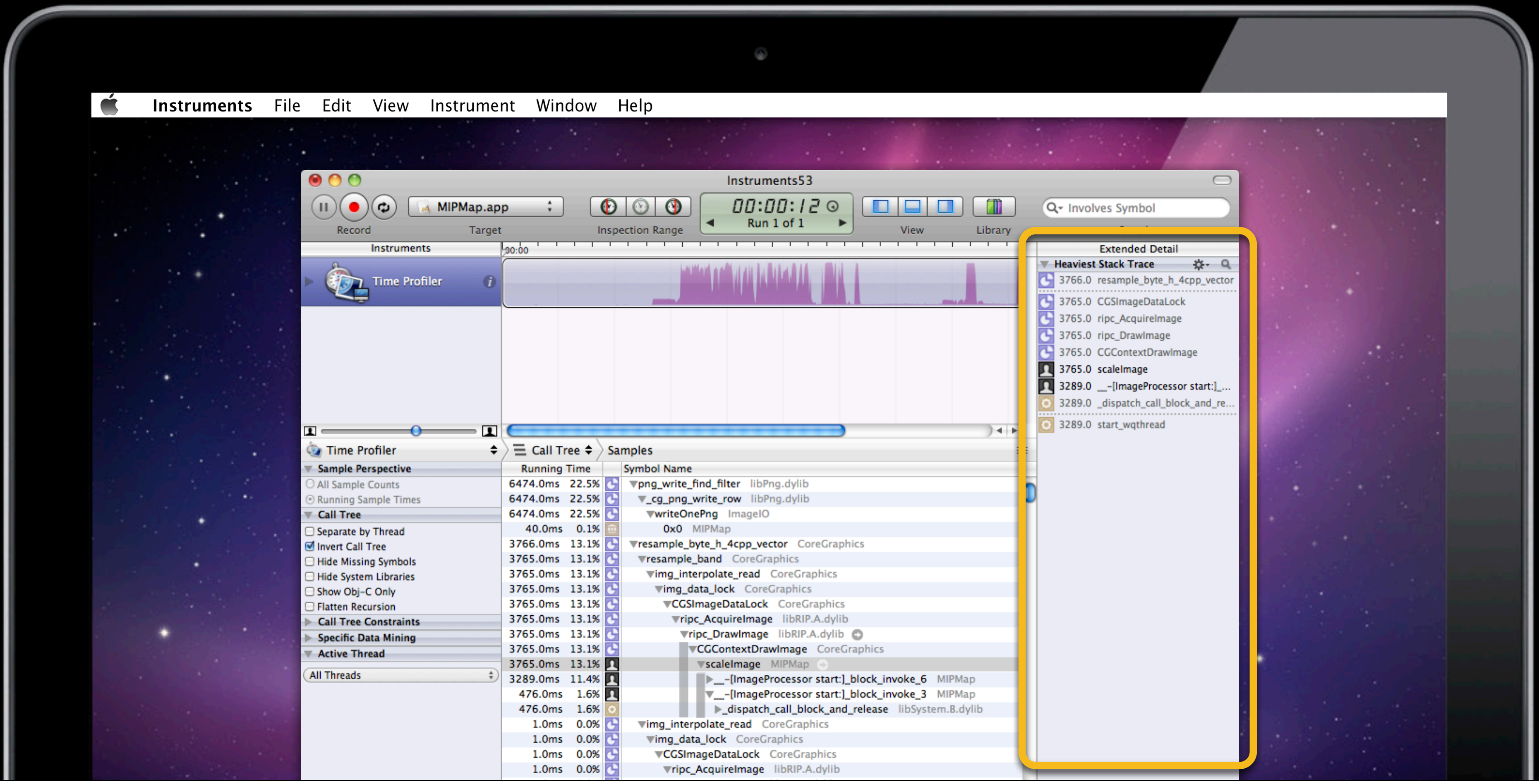


Tracking “Band”

Running Time		Symbol Name
216.0ms	10.3%	▼-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:] AppKit
216.0ms	10.3%	▼__-[NSImageRep drawInRect:fromRect:operation:fraction:respectFlipped:hints:]_block_invoke_1 AppKit
214.0ms	10.2%	▼CGContextDrawImage CoreGraphics
214.0ms	10.2%	▼ripc_DrawImage libRIP.A.dylib
210.0ms	10.0%	▼ripc_RenderImage libRIP.A.dylib
210.0ms	10.0%	▼ripl_BltImage libRIP.A.dylib
197.0ms	9.4%	▼ripd_Mark libRIP.A.dylib
197.0ms	9.4%	▼argb32_image CoreGraphics
197.0ms	9.4%	CGBlt_copyBytes CoreGraphics
13.0ms	0.6%	▶ripd_Lock libRIP.A.dylib
3.0ms	0.1%	▶ripc_AcquireImage libRIP.A.dylib
1.0ms	0.0%	ripc_GetImageTransformation libRIP.A.dylib
1.0ms	0.0%	▼CGContextGetClipBoundingBox CoreGraphics
1.0ms	0.0%	▼get_device_clip_bounding_box CoreGraphics
1.0ms	0.0%	▼CGContextDelegateGetBounds CoreGraphics
1.0ms	0.0%	▶ripc_GetBounds libRIP.A.dylib
1.0ms	0.0%	CGRectApplyInverseAffineTransform CoreGraphics
1.0ms	0.0%	objc_msgSend libobjc.A.dylib
1.0ms	0.0%	objc_msgSend libobjc.A.dylib

Backtrace Compression with Filtering

See only what matters



Sample Perspective
○ All Sample Counts
● Running Sample Times

Call Tree
□ Separate by Thread
☑ Invert Call Tree
□ Hide Missing Symbols
□ Hide System Libraries
□ Show Obj-C Only
□ Flatten Recursion

Call Tree Constraints

Specific Data Mining

Active Thread
All Threads

```
Running Time     Symbol Name
6474.0ms  22.5%  ▼png_write_find_filter  libPng.dylib
6474.0ms  22.5%   ▼_cg_png_write_row  libPng.dylib
6474.0ms  22.5%    ▼writeOnePng  ImageIO
  40.0ms   0.1%      0x0  MIPMap
3766.0ms  13.1%  ▼resample_byte_h_4cpp_vector  CoreGraphics
3765.0ms  13.1%   ▼resample_band  CoreGraphics
3765.0ms  13.1%    ▼img_interpolate_read  CoreGraphics
3765.0ms  13.1%     ▼img_data_lock  CoreGraphics
3765.0ms  13.1%      ▼CGSImageDataLock  CoreGraphics
3765.0ms  13.1%       ▼ripc_AcquireImage  libRIP.A.dylib
3765.0ms  13.1%        ▼ripc_DrawImage  libRIP.A.dylib
3765.0ms  13.1%         ▼CGContextDrawImage  CoreGraphics
3765.0ms  13.1%          ▼scaleImage  MIPMap
3289.0ms  11.4%           ▶__-[ImageProcessor start:]_block_invoke_6  MIPMap
 476.0ms   1.6%           ▼__-[ImageProcessor start:]_block_invoke_3  MIPMap
 476.0ms   1.6%            ▶_dispatch_call_block_and_release  libSystem.B.dylib
   1.0ms   0.0%   ▼img_interpolate_read  CoreGraphics
   1.0ms   0.0%    ▼img_data_lock  CoreGraphics
   1.0ms   0.0%     ▼CGSImageDataLock  CoreGraphics
   1.0ms   0.0%      ▼ripc_AcquireImage  libRIP.A.dylib
```

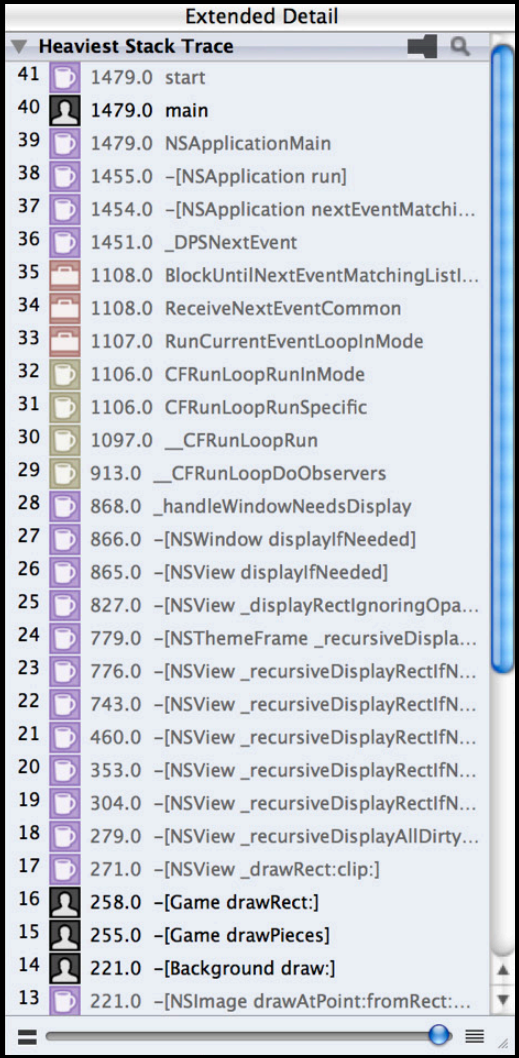
Extended Detail

```
▼ Heaviest Stack Trace
3766.0  resample_byte_h_4cpp_vector
...................................
3765.0  CGSImageDataLock
3765.0  ripc_AcquireImage
3765.0  ripc_DrawImage
3765.0  CGContextDrawImage
3765.0  scaleImage
3289.0  __-[ImageProcessor start:]_...
3289.0  _dispatch_call_block_and_re...
...................................
3289.0  start_wqthread
```

The Collapsable Backtrace

See only what matters

- Reduces down to your code
  - Collapses along boundaries
  - Use slider to adjust



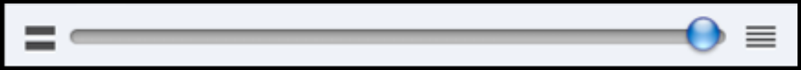
Heaviest Stack Trace

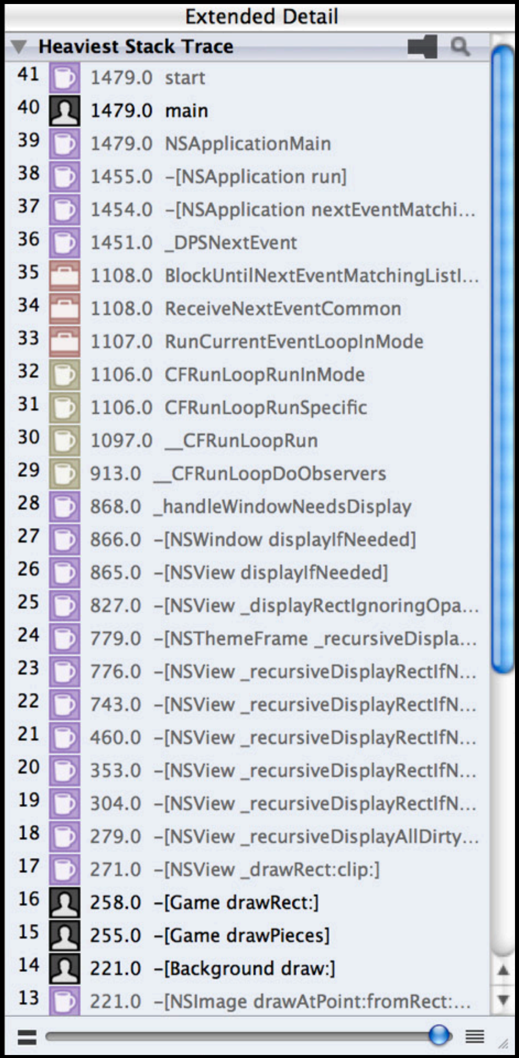
```
41  1479.0  start
40  1479.0  main
39  1479.0  NSApplicationMain
38  1455.0  -[NSApplication run]
37  1454.0  -[NSApplication nextEventMatchi...
36  1451.0  _DPSNextEvent
35  1108.0  BlockUntilNextEventMatchingListI...
34  1108.0  ReceiveNextEventCommon
33  1107.0  RunCurrentEventLoopInMode
32  1106.0  CFRunLoopRunInMode
31  1106.0  CFRunLoopRunSpecific
30  1097.0  __CFRunLoopRun
29  913.0   __CFRunLoopDoObservers
28  868.0   _handleWindowNeedsDisplay
27  866.0   -[NSWindow displayIfNeeded]
26  865.0   -[NSView displayIfNeeded]
25  827.0   -[NSView _displayRectIgnoringOpa...
24  779.0   -[NSThemeFrame _recursiveDispla...
23  776.0   -[NSView _recursiveDisplayRectIfN...
22  743.0   -[NSView _recursiveDisplayRectIfN...
21  460.0   -[NSView _recursiveDisplayRectIfN...
20  353.0   -[NSView _recursiveDisplayRectIfN...
19  304.0   -[NSView _recursiveDisplayRectIfN...
18  279.0   -[NSView _recursiveDisplayAllDirty...
17  271.0   -[NSView _drawRect:clip:]
16  258.0   -[Game drawRect:]
15  255.0   -[Game drawPieces]
14  221.0   -[Background draw:]
13  221.0   -[NSImage drawAtPoint:fromRect:...
```

The Collapsable Backtrace

See only what matters

- Reduces down to your code
  - Collapses along boundaries
  - Use slider to adjust





Extended Detail

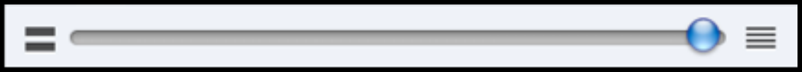
Heaviest Stack Trace

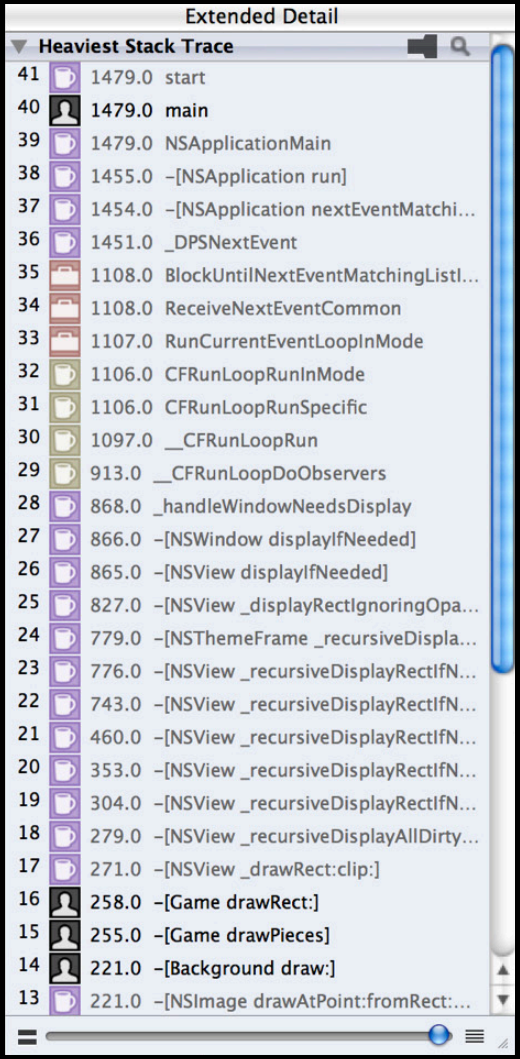
```
41  1479.0  start
40  1479.0  main
39  1479.0  NSApplicationMain
38  1455.0  -[NSApplication run]
37  1454.0  -[NSApplication nextEventMatchi...
36  1451.0  _DPSNextEvent
35  1108.0  BlockUntilNextEventMatchingListI...
34  1108.0  ReceiveNextEventCommon
33  1107.0  RunCurrentEventLoopInMode
32  1106.0  CFRunLoopRunInMode
31  1106.0  CFRunLoopRunSpecific
30  1097.0  __CFRunLoopRun
29   913.0  __CFRunLoopDoObservers
28   868.0  _handleWindowNeedsDisplay
27   866.0  -[NSWindow displayIfNeeded]
26   865.0  -[NSView displayIfNeeded]
25   827.0  -[NSView _displayRectIgnoringOpa...
24   779.0  -[NSThemeFrame _recursiveDispla...
23   776.0  -[NSView _recursiveDisplayRectIfN...
22   743.0  -[NSView _recursiveDisplayRectIfN...
21   460.0  -[NSView _recursiveDisplayRectIfN...
20   353.0  -[NSView _recursiveDisplayRectIfN...
19   304.0  -[NSView _recursiveDisplayRectIfN...
18   279.0  -[NSView _recursiveDisplayAllDirty...
17   271.0  -[NSView _drawRect:clip:]
16   258.0  -[Game drawRect:]
15   255.0  -[Game drawPieces]
14   221.0  -[Background draw:]
13   221.0  -[NSImage drawAtPoint:fromRect:...
```

The Collapsable Backtrace

See only what matters

- Reduces down to your code
  - Collapses along boundaries
  - Use slider to adjust



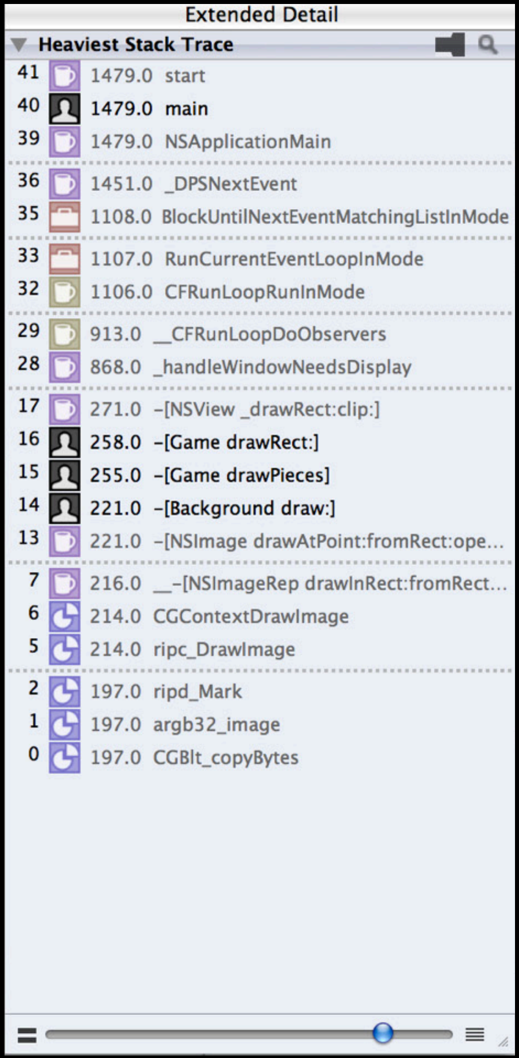


```
41  1479.0  start
40  1479.0  main
39  1479.0  NSApplicationMain
38  1455.0  -[NSApplication run]
37  1454.0  -[NSApplication nextEventMatchi...
36  1451.0  _DPSNextEvent
35  1108.0  BlockUntilNextEventMatchingListI...
34  1108.0  ReceiveNextEventCommon
33  1107.0  RunCurrentEventLoopInMode
32  1106.0  CFRunLoopRunInMode
31  1106.0  CFRunLoopRunSpecific
30  1097.0  __CFRunLoopRun
29  913.0   __CFRunLoopDoObservers
28  868.0   _handleWindowNeedsDisplay
27  866.0   -[NSWindow displayIfNeeded]
26  865.0   -[NSView displayIfNeeded]
25  827.0   -[NSView _displayRectIgnoringOpa...
24  779.0   -[NSThemeFrame _recursiveDispla...
23  776.0   -[NSView _recursiveDisplayRectIfN...
22  743.0   -[NSView _recursiveDisplayRectIfN...
21  460.0   -[NSView _recursiveDisplayRectIfN...
20  353.0   -[NSView _recursiveDisplayRectIfN...
19  304.0   -[NSView _recursiveDisplayRectIfN...
18  279.0   -[NSView _recursiveDisplayAllDirty...
17  271.0   -[NSView _drawRect:clip:]
16  258.0   -[Game drawRect:]
15  255.0   -[Game drawPieces]
14  221.0   -[Background draw:]
13  221.0   -[NSImage drawAtPoint:fromRect:...
```

The Collapsable Backtrace

See only what matters

- Reduces down to your code
  - Collapses along boundaries
  - Use slider to adjust



Extended Detail

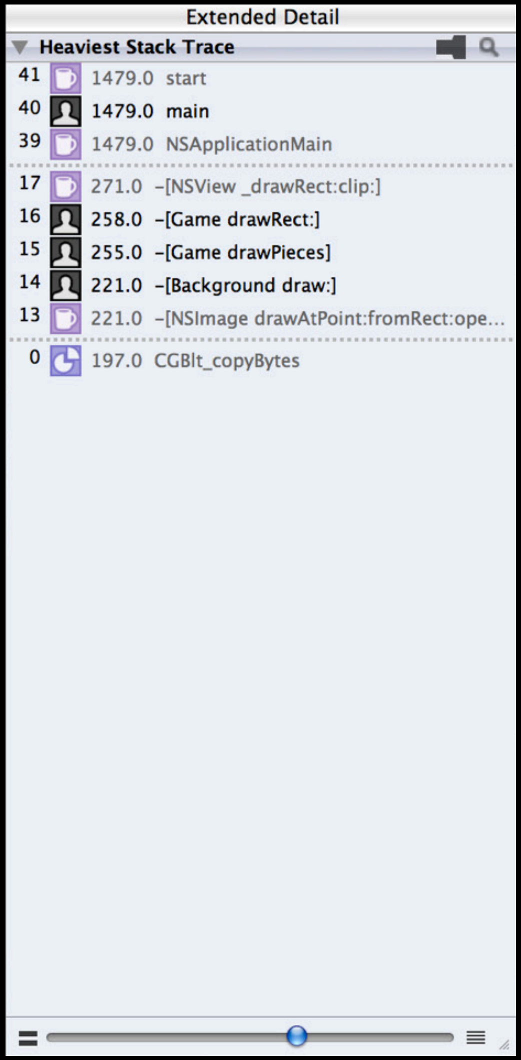
Heaviest Stack Trace

```
41  1479.0  start
40  1479.0  main
39  1479.0  NSApplicationMain
36  1451.0  _DPSNextEvent
35  1108.0  BlockUntilNextEventMatchingListInMode
33  1107.0  RunCurrentEventLoopInMode
32  1106.0  CFRunLoopRunInMode
29   913.0  __CFRunLoopDoObservers
28   868.0  _handleWindowNeedsDisplay
17   271.0  -[NSView _drawRect:clip:]
16   258.0  -[Game drawRect:]
15   255.0  -[Game drawPieces]
14   221.0  -[Background draw:]
13   221.0  -[NSImage drawAtPoint:fromRect:ope...
 7   216.0  __-[NSImageRep drawInRect:fromRect...
 6   214.0  CGContextDrawImage
 5   214.0  ripc_DrawImage
 2   197.0  ripd_Mark
 1   197.0  argb32_image
 0   197.0  CGBlt_copyBytes
```

The Collapsable Backtrace

See only what matters

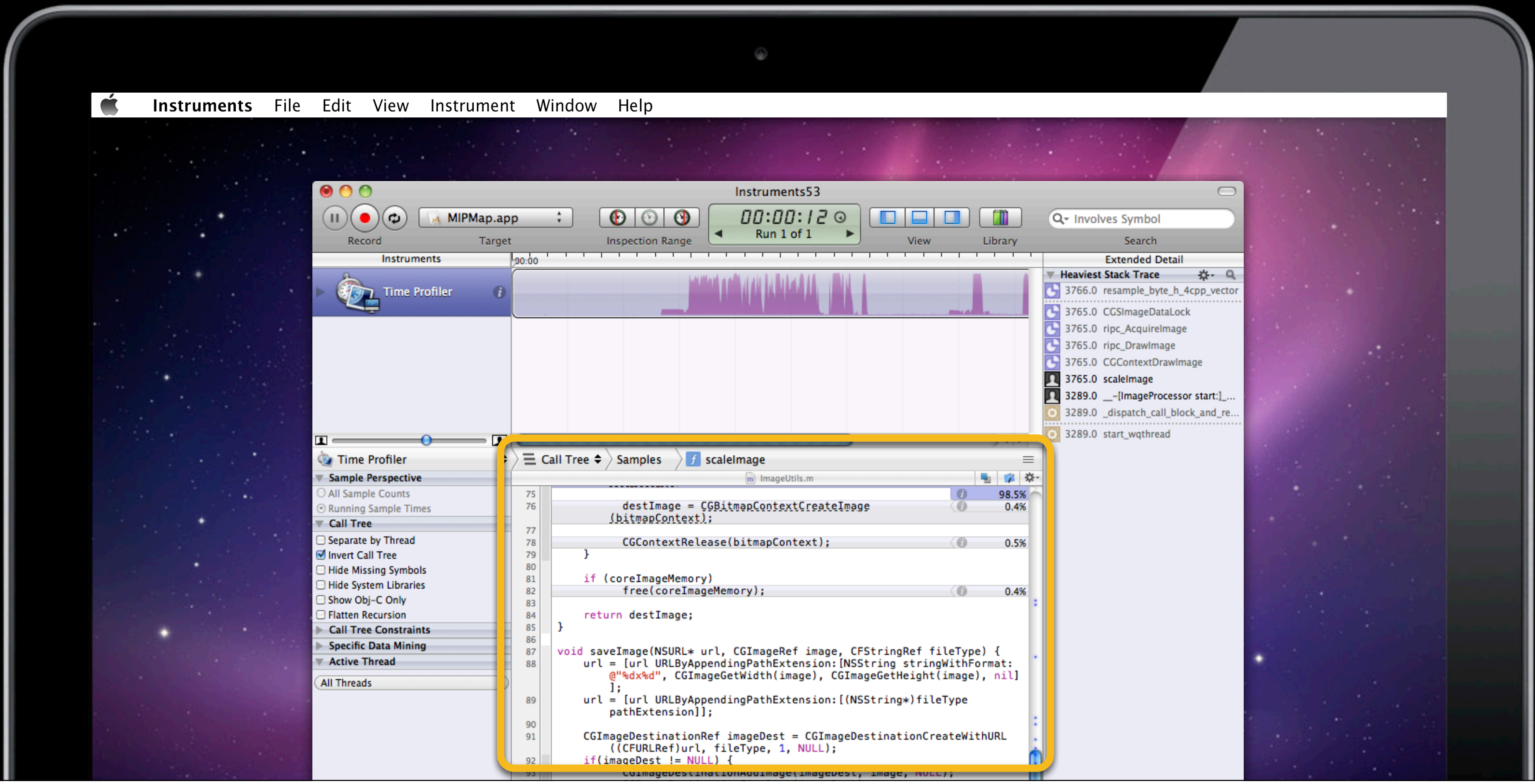
- Reduces down to your code
  - Collapses along boundaries
  - Use slider to adjust



```
41  1479.0  start
40  1479.0  main
39  1479.0  NSApplicationMain
17  271.0  -[NSView _drawRect:clip:]
16  258.0  -[Game drawRect:]
15  255.0  -[Game drawPieces]
14  221.0  -[Background draw:]
13  221.0  -[NSImage drawAtPoint:fromRect:ope...
 0  197.0  CGBlt_copyBytes
```

The Source View

Performance data inline with your source



Time Profiler

Sample Perspective
○ All Sample Counts
● Running Sample Times

Call Tree
☐ Separate by Thread
☑ Invert Call Tree
☐ Hide Missing Symbols
☐ Hide System Libraries
☐ Show Obj-C Only
☐ Flatten Recursion

Call Tree Constraints

Specific Data Mining

Active Thread

All Threads

```
                                                              98.5%
        destImage = CGBitmapContextCreateImage                 0.4%
    (bitmapContext);

        CGContextRelease(bitmapContext);                       0.5%
    }

    if (coreImageMemory)
        free(coreImageMemory);                                 0.4%

    return destImage;
}

void saveImage(NSURL* url, CGImageRef image, CFStringRef fileType) {
    url = [url URLByAppendingPathExtension:[NSString stringWithFormat:
        @"%dx%d", CGImageGetWidth(image), CGImageGetHeight(image), nil]
        ];
    url = [url URLByAppendingPathExtension:[(NSString*)fileType
        pathExtension]];

    CGImageDestinationRef imageDest = CGImageDestinationCreateWithURL
        ((CFURLRef)url, fileType, 1, NULL);
    if(imageDest != NULL) {
```

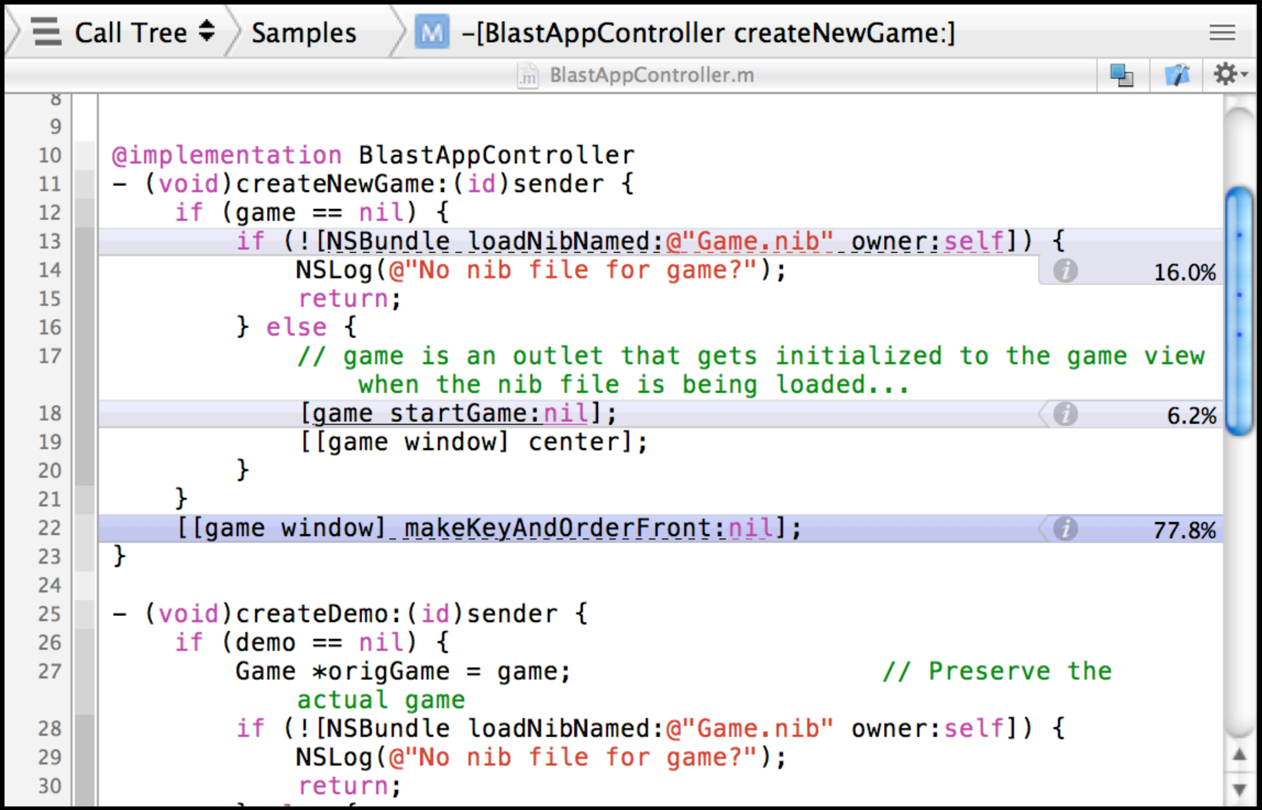
Heaviest Stack Trace

```
3766.0  resample_byte_h_4cpp_vector
3765.0  CGSImageDataLock
3765.0  ripc_AcquireImage
3765.0  ripc_DrawImage
3765.0  CGContextDrawImage
3765.0  scaleImage
3289.0  __-[ImageProcessor start:]_...
3289.0  _dispatch_call_block_and_re...
3289.0  start_wqthread
```

The Source View

Performance data inline with your source

- Directly accessed from data views



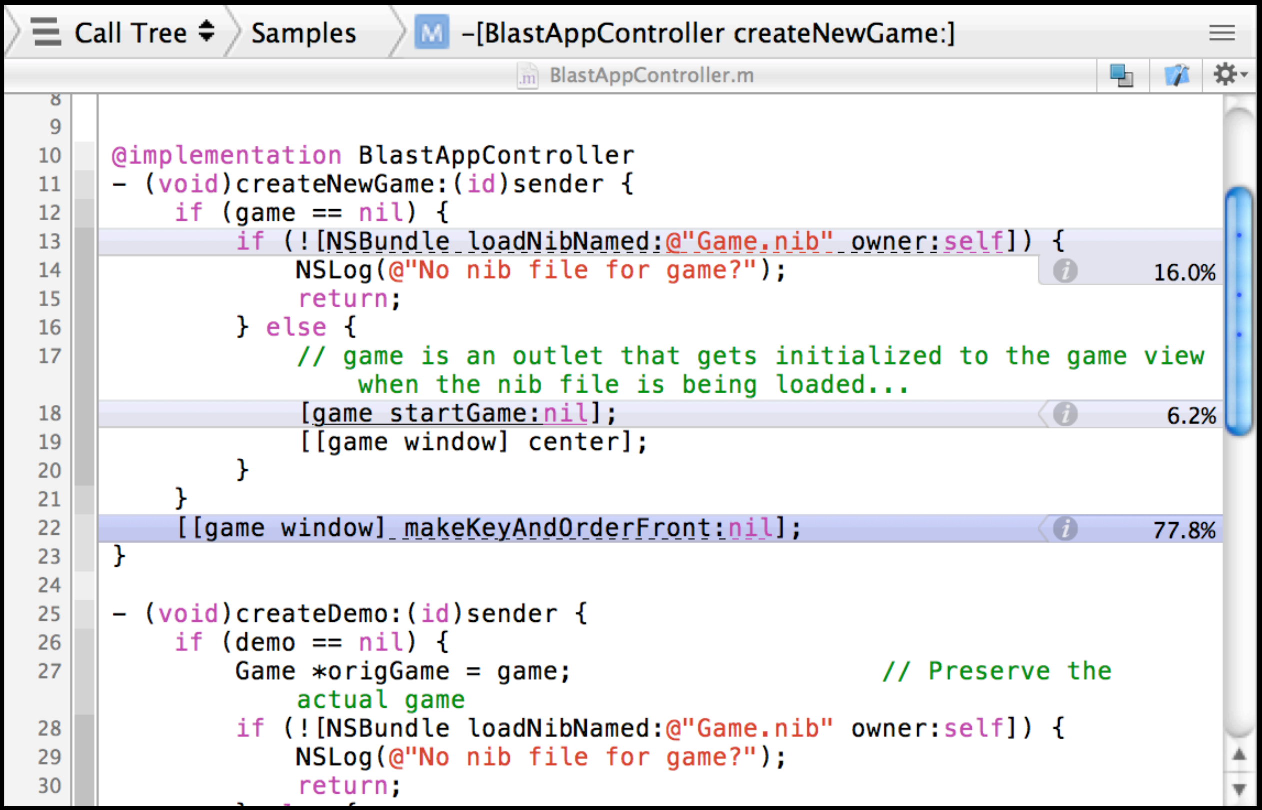
```
@implementation BlastAppController
- (void)createNewGame:(id)sender {
    if (game == nil) {
        if (![NSBundle loadNibNamed:@"Game.nib" owner:self]) {
            NSLog(@"No nib file for game?");                   16.0%
            return;
        } else {
            // game is an outlet that gets initialized to the game view
               when the nib file is being loaded...
            [game startGame:nil];                                6.2%
            [[game window] center];
        }
    }
    [[game window] makeKeyAndOrderFront:nil];                   77.8%
}

- (void)createDemo:(id)sender {
    if (demo == nil) {
        Game *origGame = game;                   // Preserve the
            actual game
        if (![NSBundle loadNibNamed:@"Game.nib" owner:self]) {
            NSLog(@"No nib file for game?");
            return;
```

The Source View

Performance data inline with your source

- Directly accessed from data views
- Colored by severity



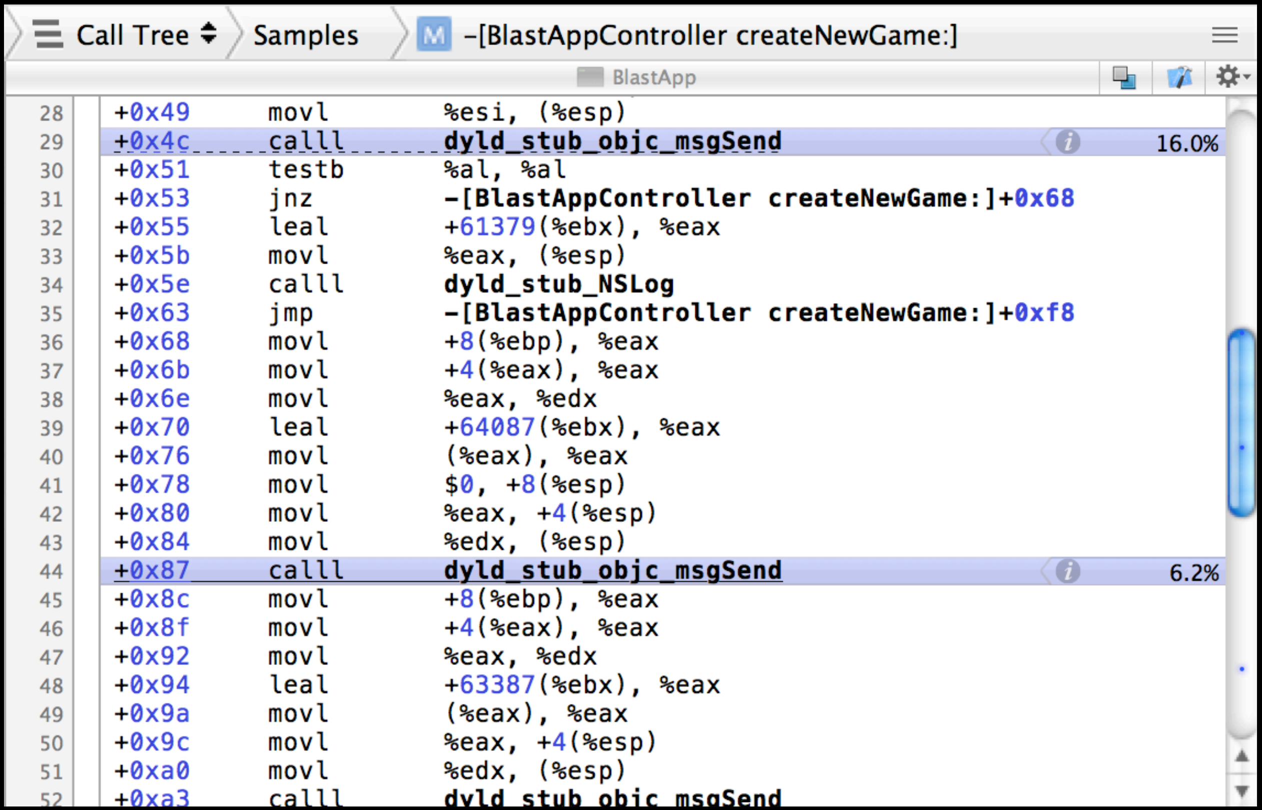
```
@implementation BlastAppController
- (void)createNewGame:(id)sender {
    if (game == nil) {
        if (![NSBundle loadNibNamed:@"Game.nib" owner:self]) {
            NSLog(@"No nib file for game?");                        16.0%
            return;
        } else {
            // game is an outlet that gets initialized to the game view
               when the nib file is being loaded...
            [game startGame:nil];                                    6.2%
            [[game window] center];
        }
    }
    [[game window] makeKeyAndOrderFront:nil];                       77.8%
}

- (void)createDemo:(id)sender {
    if (demo == nil) {
        Game *origGame = game;                    // Preserve the
            actual game
        if (![NSBundle loadNibNamed:@"Game.nib" owner:self]) {
            NSLog(@"No nib file for game?");
            return;
```

The Source View

Performance data inline with your code

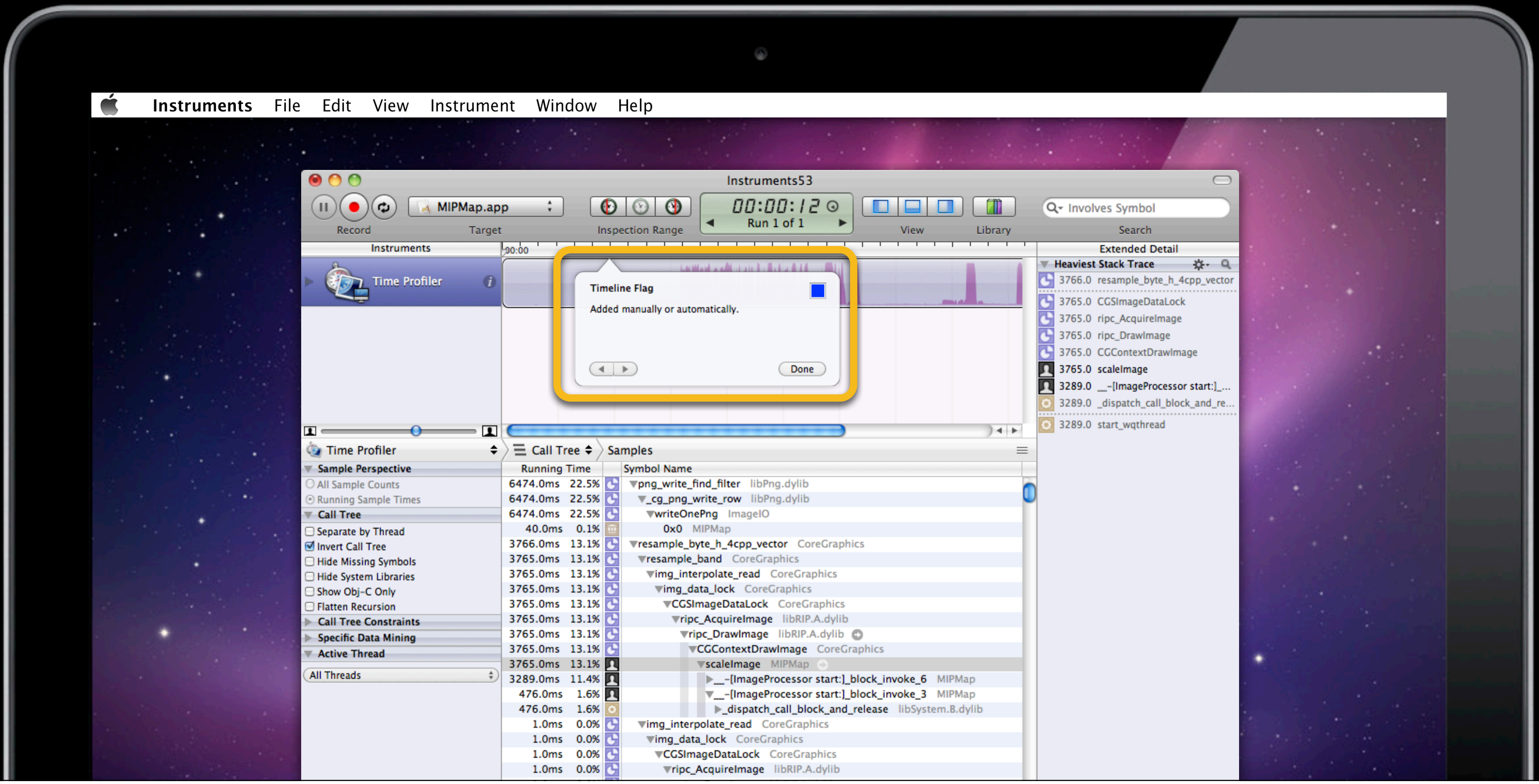
- Directly accessed from data views
- Colored by severity
- Disassembly
  - By function
  - By source line



```
+0x49    movl     %esi, (%esp)
+0x4c    calll    dyld_stub_objc_msgSend                             16.0%
+0x51    testb    %al, %al
+0x53    jnz      -[BlastAppController createNewGame:]+0x68
+0x55    leal     +61379(%ebx), %eax
+0x5b    movl     %eax, (%esp)
+0x5e    calll    dyld_stub_NSLog
+0x63    jmp      -[BlastAppController createNewGame:]+0xf8
+0x68    movl     +8(%ebp), %eax
+0x6b    movl     +4(%eax), %eax
+0x6e    movl     %eax, %edx
+0x70    leal     +64087(%ebx), %eax
+0x76    movl     (%eax), %eax
+0x78    movl     $0, +8(%esp)
+0x80    movl     %eax, +4(%esp)
+0x84    movl     %edx, (%esp)
+0x87    calll    dyld_stub_objc_msgSend                              6.2%
+0x8c    movl     +8(%ebp), %eax
+0x8f    movl     +4(%eax), %eax
+0x92    movl     %eax, %edx
+0x94    leal     +63387(%ebx), %eax
+0x9a    movl     (%eax), %eax
+0x9c    movl     %eax, +4(%esp)
+0xa0    movl     %edx, (%esp)
+0xa3    calll    dyld_stub_objc_msgSend
```

Timeline Flags

Note and remember important events



Timeline Flag

Added manually or automatically.

Time Profiler

Sample Perspective

- ○ All Sample Counts
- ● Running Sample Times

Call Tree

- □ Separate by Thread
- ☑ Invert Call Tree
- □ Hide Missing Symbols
- □ Hide System Libraries
- □ Show Obj-C Only
- □ Flatten Recursion

Call Tree Constraints

Specific Data Mining

Active Thread

All Threads

Call Tree › Samples

```
Running Time      Symbol Name
6474.0ms  22.5%   ▼png_write_find_filter  libPng.dylib
6474.0ms  22.5%     ▼_cg_png_write_row  libPng.dylib
6474.0ms  22.5%       ▼writeOnePng  ImageIO
  40.0ms   0.1%          0x0  MIPMap
3766.0ms  13.1%   ▼resample_byte_h_4cpp_vector  CoreGraphics
3765.0ms  13.1%     ▼resample_band  CoreGraphics
3765.0ms  13.1%       ▼img_interpolate_read  CoreGraphics
3765.0ms  13.1%         ▼img_data_lock  CoreGraphics
3765.0ms  13.1%           ▼CGSImageDataLock  CoreGraphics
3765.0ms  13.1%             ▼ripc_AcquireImage  libRIP.A.dylib
3765.0ms  13.1%               ▼ripc_DrawImage  libRIP.A.dylib
3765.0ms  13.1%                 ▼CGContextDrawImage  CoreGraphics
3765.0ms  13.1%                   ▼scaleImage  MIPMap
3289.0ms  11.4%                     ▶__-[ImageProcessor start:]_block_invoke_6  MIPMap
 476.0ms   1.6%                     ▼__-[ImageProcessor start:]_block_invoke_3  MIPMap
 476.0ms   1.6%                       ▶_dispatch_call_block_and_release  libSystem.B.dylib
   1.0ms   0.0%     ▼img_interpolate_read  CoreGraphics
   1.0ms   0.0%       ▼img_data_lock  CoreGraphics
   1.0ms   0.0%         ▼CGSImageDataLock  CoreGraphics
   1.0ms   0.0%           ▼ripc_AcquireImage  libRIP.A.dylib
```

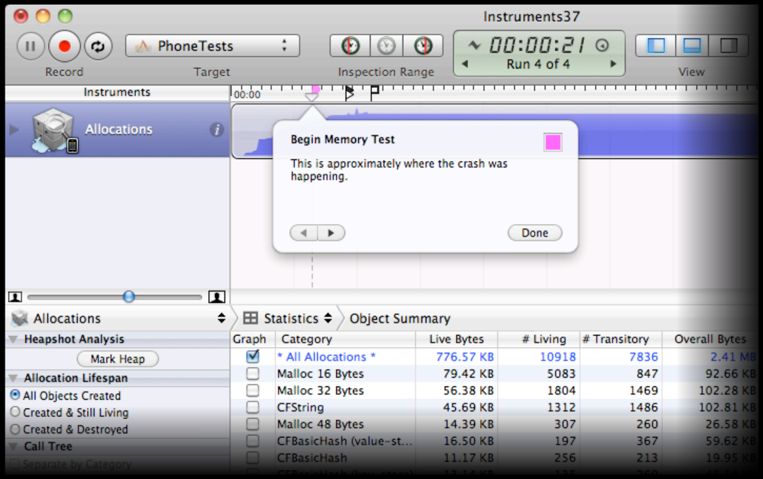
Heaviest Stack Trace

```
3766.0  resample_byte_h_4cpp_vector
3765.0  CGSImageDataLock
3765.0  ripc_AcquireImage
3765.0  ripc_DrawImage
3765.0  CGContextDrawImage
3765.0  scaleImage
3289.0  __-[ImageProcessor start:]_...
3289.0  _dispatch_call_block_and_re...
3289.0  start_wqthread
```

Timeline Flags

Note and remember important events

- May be added manually
    - Use as “Bookmarks” within a trace



Begin Memory Test

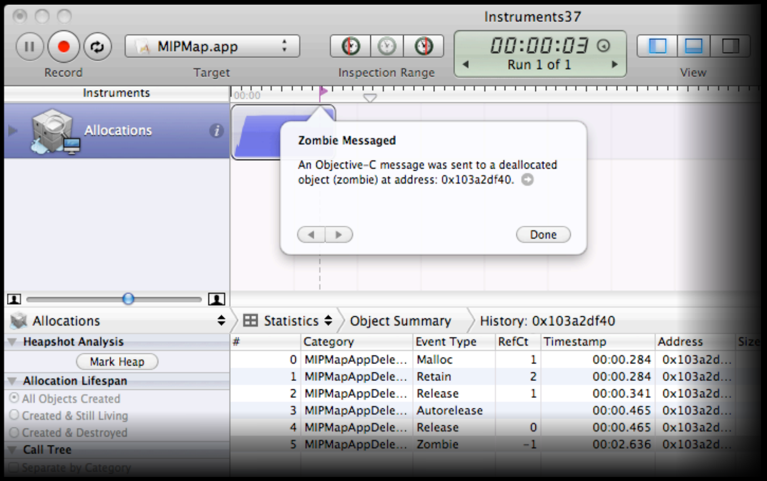
This is approximately where the crash was happening.

Graph	Category	Live Bytes	# Living	# Transitory	Overall Bytes
☑	* All Allocations *	776.57 KB	10918	7836	2.41 MB
□	Malloc 16 Bytes	79.42 KB	5083	847	92.66 KB
□	Malloc 32 Bytes	56.38 KB	1804	1469	102.28 KB
□	CFString	45.69 KB	1312	1486	102.81 KB
□	Malloc 48 Bytes	14.39 KB	307	260	26.58 KB
□	CFBasicHash (value-st...	16.50 KB	197	367	59.62 KB
□	CFBasicHash	11.17 KB	256	213	19.95 KB

Timeline Flags

Note and remember important events

- May be added manually
    - Use as "Bookmarks" with a trace
- Automatically added
    - Zombies Instrument



Zombie Messaged

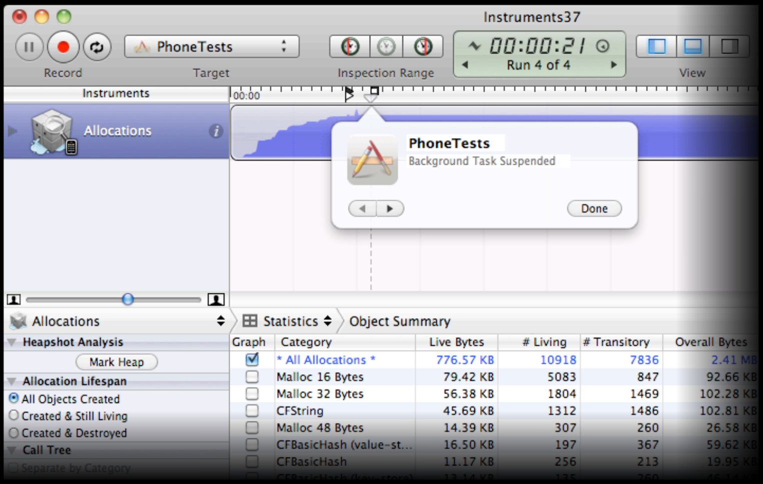
An Objective-C message was sent to a deallocated object (zombie) at address: 0x103a2df40.

#	Category	Event Type	RefCt	Timestamp	Address	Size
0	MIPMapAppDele…	Malloc	1	00:00.284	0x103a2d…	
1	MIPMapAppDele…	Retain	2	00:00.284	0x103a2d…	
2	MIPMapAppDele…	Release	1	00:00.341	0x103a2d…	
3	MIPMapAppDele…	Autorelease		00:00.465	0x103a2d…	
4	MIPMapAppDele…	Release	0	00:00.465	0x103a2d…	
5	MIPMapAppDele…	Zombie	-1	00:02.636	0x103a2d…	

Timeline Flags

Note and remember important events

- May be added manually
  - Use as “Bookmarks” with a trace
- Automatically added
  - Zombies Instrument
  - iOS 4 Multitasking State Transitions
    - For all processes or targeted process



PhoneTests
Background Task Suspended

Graph	Category	Live Bytes	# Living	# Transitory	Overall Bytes
☑	* All Allocations *	776.57 KB	10918	7836	2.41 MB
□	Malloc 16 Bytes	79.42 KB	5083	847	92.66 KB
□	Malloc 32 Bytes	56.38 KB	1804	1469	102.28 KB
□	CFString	45.69 KB	1312	1486	102.81 KB
□	Malloc 48 Bytes	14.39 KB	307	260	26.58 KB
□	CFBasicHash (value-st…	16.50 KB	197	367	59.62 KB
□	CFBasicHash	11.17 KB	256	213	19.95 KB

Demo

Using Instruments

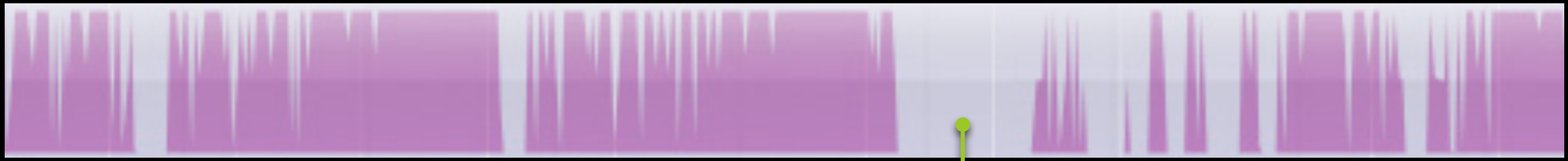
Recording in Instruments

Greater efficiency, greater latitude

Immediate Versus Deferred

Mac OS X and iOS 4

- **Immediate Mode**—"Classic Instruments Mode"
  - Processes and displays data immediately
    - Permits visual association between user actions and data spikes
    - Places heavier CPU load on Mac and iPhone

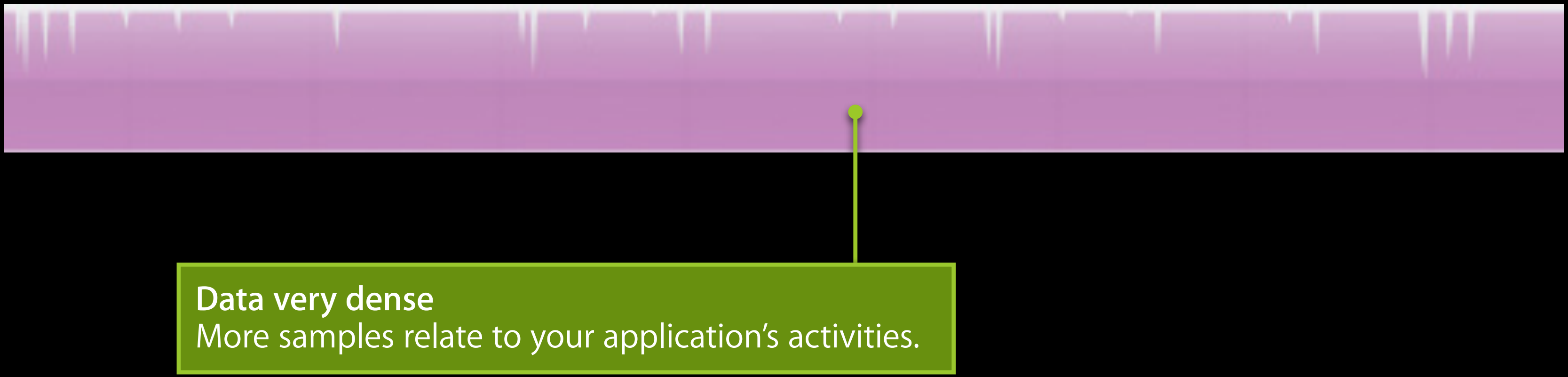


Data sometimes sparse
Valleys and gaps related to Instruments processing and displaying the data while your application is running.

Immediate Versus Deferred

Mac OS X and iOS 4

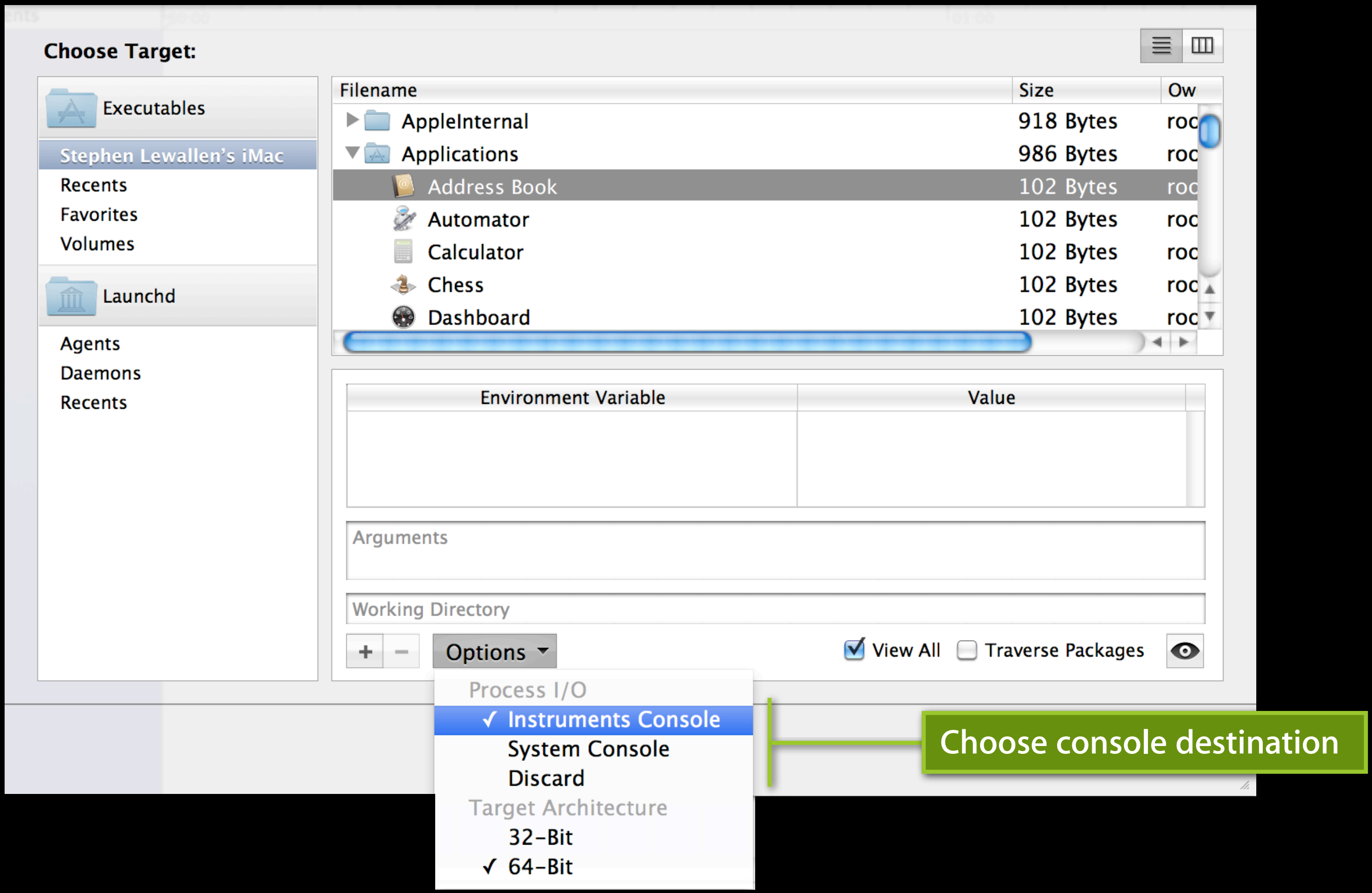
- **Deferred Mode**
  - Processes and displays data at end of recording
    - Vastly reduces “observer effect” by relinquishing CPU to target
    - Impairs ability to visually correlate user actions and data spikes



Data very dense
More samples relate to your application’s activities.

Launch Options

Mac OS X



Choose Target:

Filename	Size	Ow
AppleInternal	918 Bytes	roc
Applications	986 Bytes	roc
Address Book	102 Bytes	roc
Automator	102 Bytes	roc
Calculator	102 Bytes	roc
Chess	102 Bytes	roc
Dashboard	102 Bytes	roc

Environment Variable	Value

Arguments

Working Directory

☑ View All □ Traverse Packages

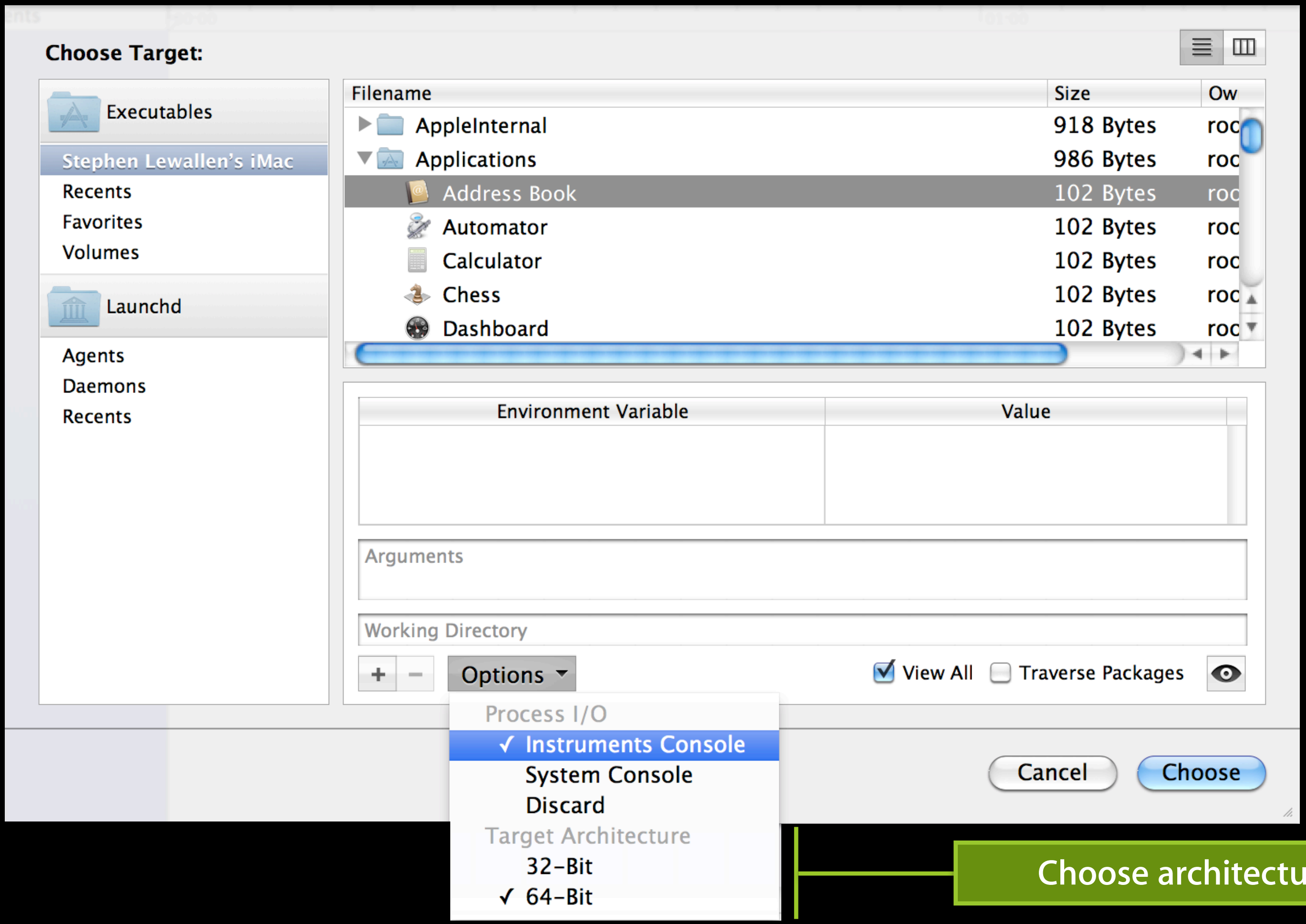
Process I/O: ✓ Instruments Console

Target Architecture: ✓ 64-Bit

Choose console destination

Launch Options

Mac OS X



Choose Target:

Executables

- Stephen Lewallen's iMac
- Recents
- Favorites
- Volumes

Launchd

- Agents
- Daemons
- Recents

Filename	Size	Ow
▶ AppleInternal	918 Bytes	roc
▼ Applications	986 Bytes	roc
Address Book	102 Bytes	roc
Automator	102 Bytes	roc
Calculator	102 Bytes	roc
Chess	102 Bytes	roc
Dashboard	102 Bytes	roc

Environment Variable	Value

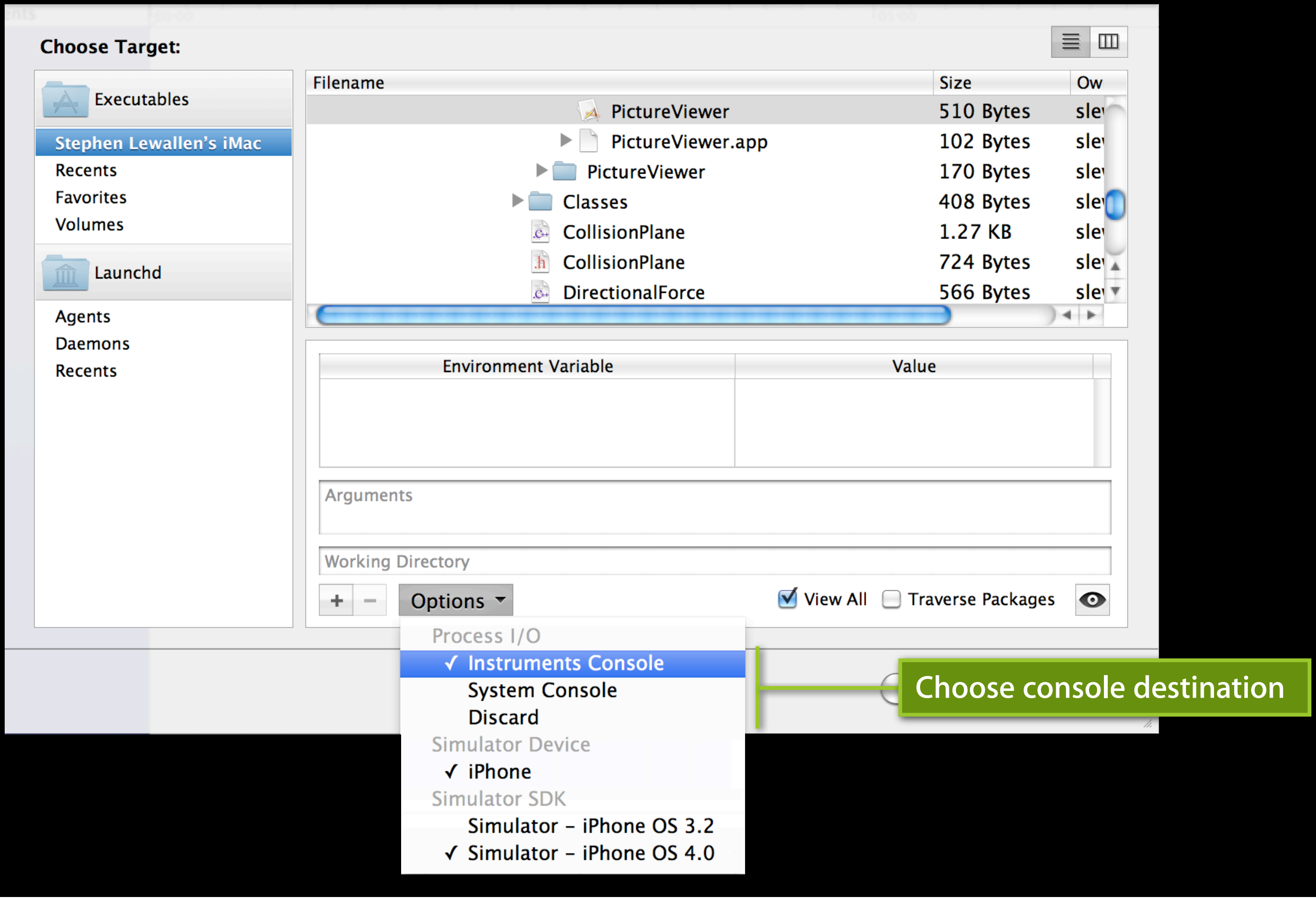
Arguments

Working Directory

Choose architecture

Launch Options

iPhone Simulator



Filename	Size	Ow
PictureViewer	510 Bytes	sle
PictureViewer.app	102 Bytes	sle
PictureViewer	170 Bytes	sle
Classes	408 Bytes	sle
CollisionPlane	1.27 KB	sle
CollisionPlane	724 Bytes	sle
DirectionalForce	566 Bytes	sle

☑ View All □ Traverse Packages

Process I/O
- ✓ Instruments Console
- System Console
- Discard

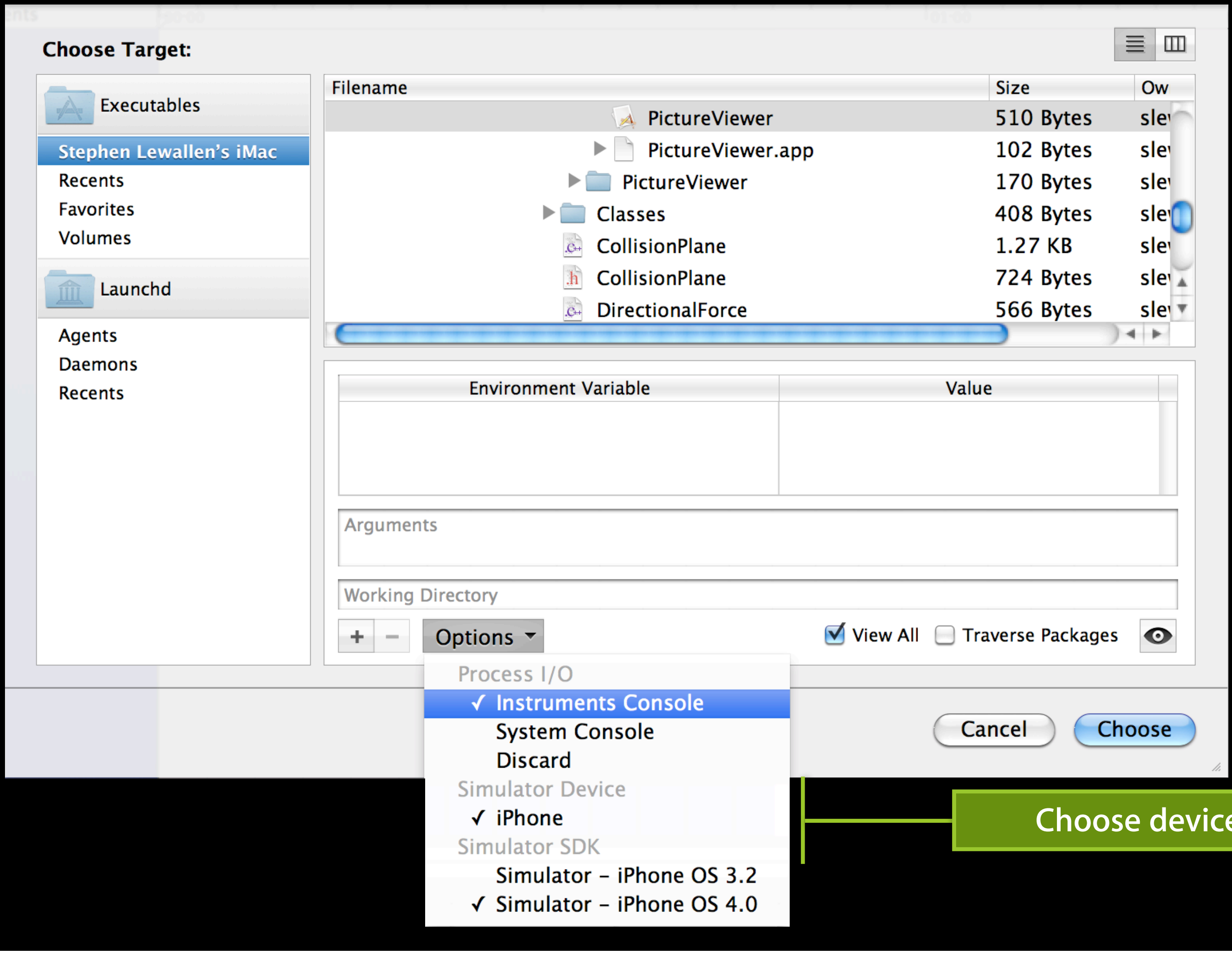
Simulator Device
- ✓ iPhone

Simulator SDK
- Simulator – iPhone OS 3.2
- ✓ Simulator – iPhone OS 4.0

Choose console destination

Launch Options

iPhone Simulator



Choose Target:

Filename	Size	Ow
PictureViewer	510 Bytes	sle
PictureViewer.app	102 Bytes	sle
PictureViewer	170 Bytes	sle
Classes	408 Bytes	sle
CollisionPlane	1.27 KB	sle
CollisionPlane	724 Bytes	sle
DirectionalForce	566 Bytes	sle

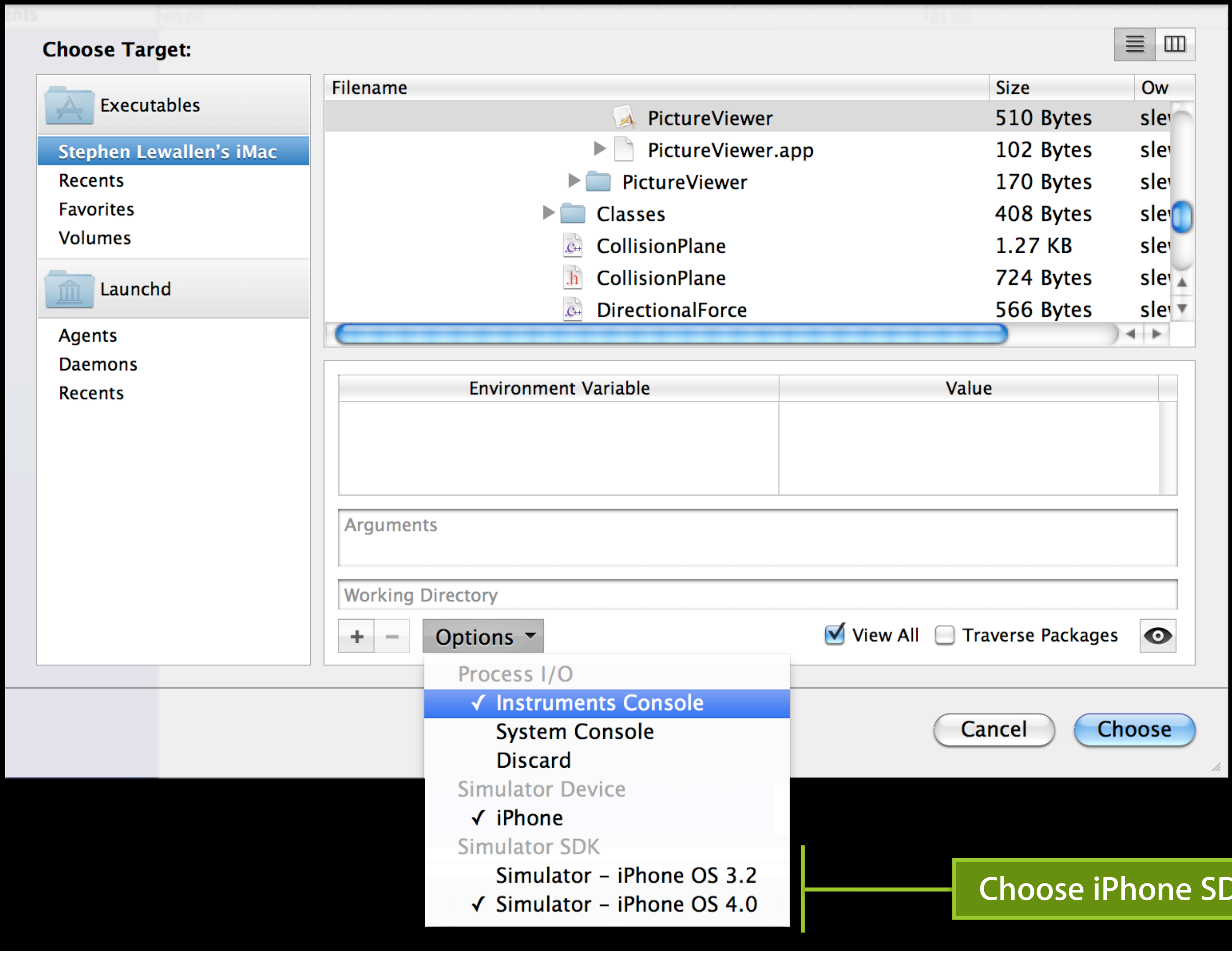
Environment Variable	Value

Process I/O
✓ Instruments Console
System Console
Discard
Simulator Device
✓ iPhone
Simulator SDK
Simulator – iPhone OS 3.2
✓ Simulator – iPhone OS 4.0

Choose device type

Launch Options

iPhone Simulator



Choose Target:

Filename	Size	Ow
PictureViewer	510 Bytes	sle
PictureViewer.app	102 Bytes	sle
PictureViewer	170 Bytes	sle
Classes	408 Bytes	sle
CollisionPlane	1.27 KB	sle
CollisionPlane	724 Bytes	sle
DirectionalForce	566 Bytes	sle

Environment Variable	Value

Arguments

Working Directory

Choose iPhone SDK version

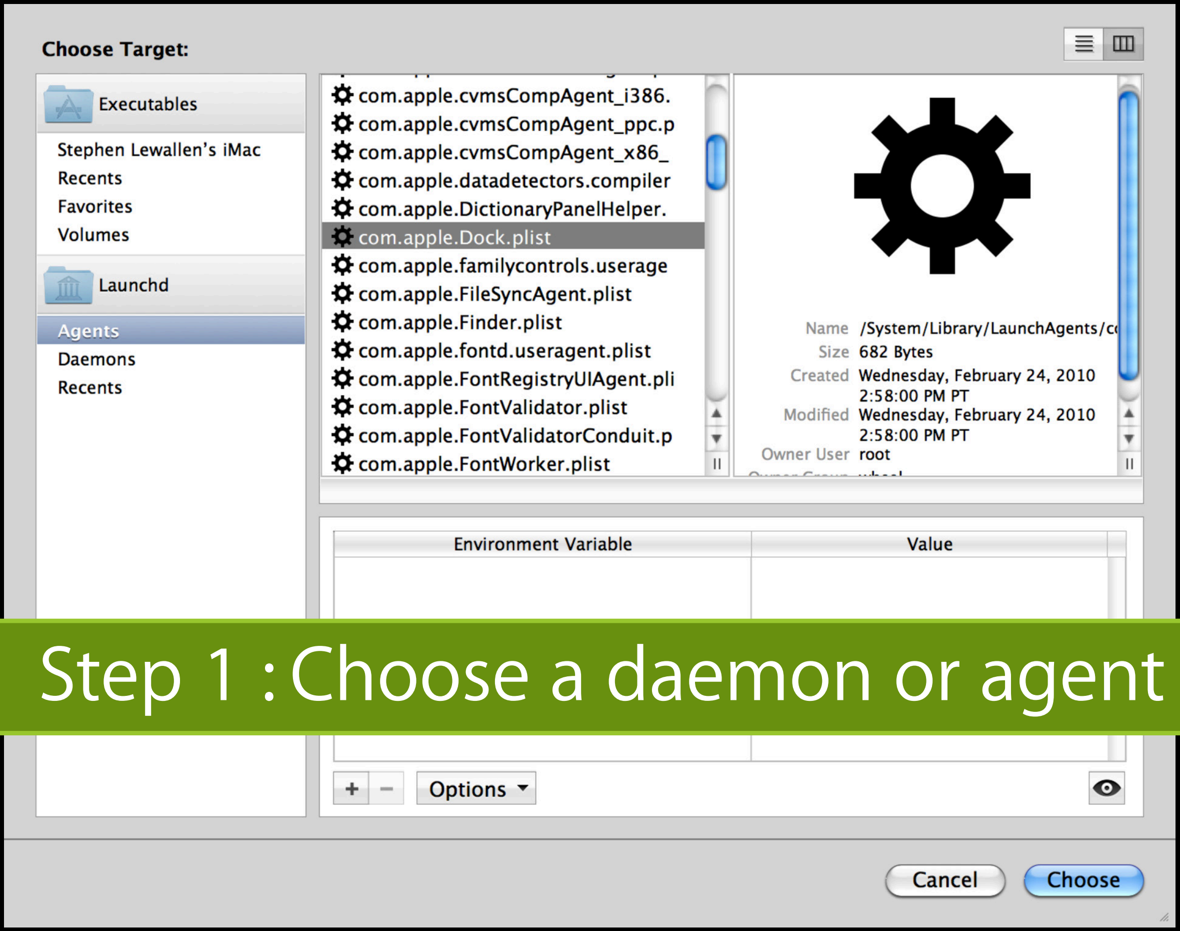
Launch Daemons and Agents

Mac OS X only

- Target launch daemons and agents
- Instruments coordinates with LaunchD
  - Begins analyzing when process is created by LaunchD

Launch Daemons and Agents

Mac OS X only



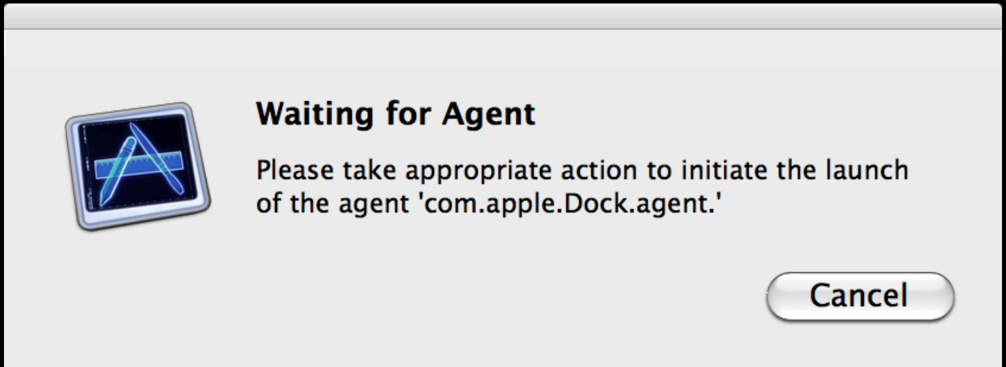
com.apple.cvmsCompAgent_i386.
com.apple.cvmsCompAgent_ppc.p
com.apple.cvmsCompAgent_x86_
com.apple.datadetectors.compiler
com.apple.DictionaryPanelHelper.
com.apple.Dock.plist
com.apple.familycontrols.userage
com.apple.FileSyncAgent.plist
com.apple.Finder.plist
com.apple.fontd.useragent.plist
com.apple.FontRegistryUIAgent.pli
com.apple.FontValidator.plist
com.apple.FontValidatorConduit.p
com.apple.FontWorker.plist

Name /System/Library/LaunchAgents/c
Size 682 Bytes
Created Wednesday, February 24, 2010 2:58:00 PM PT
Modified Wednesday, February 24, 2010 2:58:00 PM PT
Owner User root

Step 1 : Choose a daemon or agent

Launch Daemons and Agents

Mac OS X only



Waiting for Agent

Please take appropriate action to initiate the launch of the agent 'com.apple.Dock.agent.'

Step 2 : Start recording and trigger agent

Wireless

iPhone OS 3.1+

- Use Instruments over Wi-Fi rather than USB
    - Frees up USB port for accessory developers
    - Provides freedom of movement for accelerometer usage such as games
    - Requires Wi-Fi with Bonjour enabled (multicast)

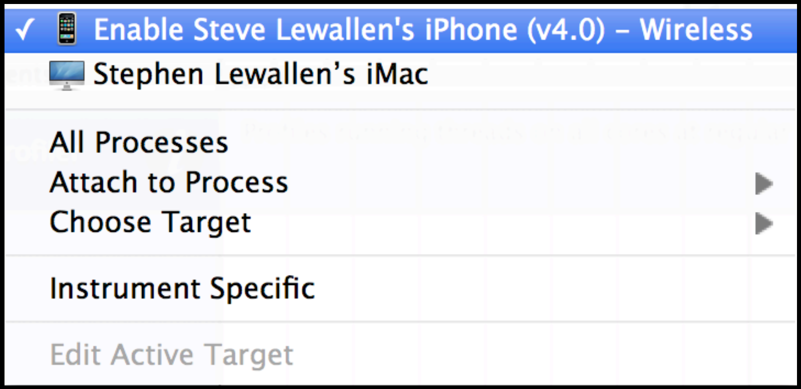
Enabling Wireless

iPhone OS 3.1+

Enabling Wireless

iPhone OS 3.1+

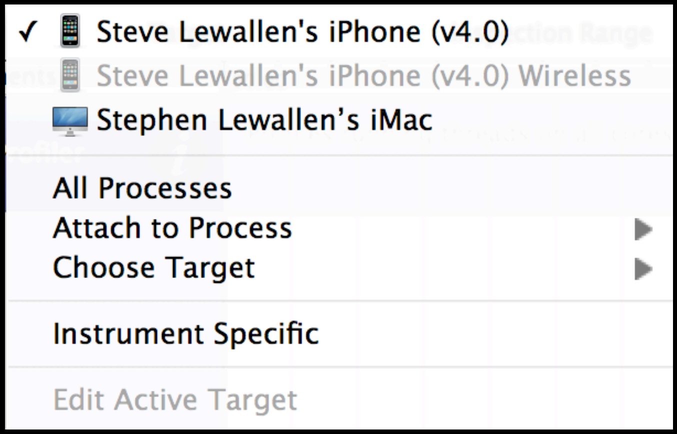
- Hold down option key while selecting target device to enable



Enabling Wireless

iPhone OS 3.1+

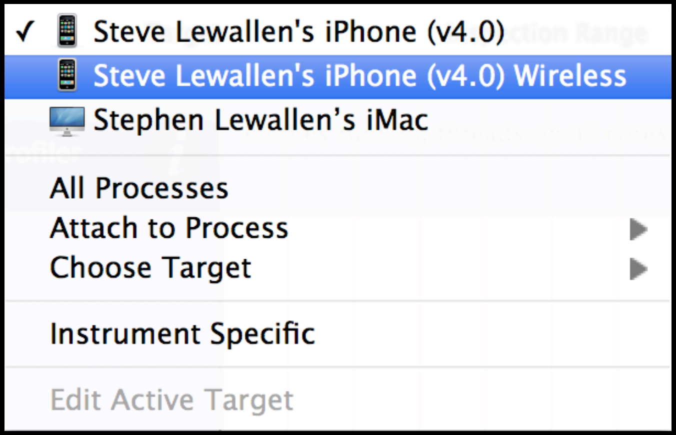
- Hold down option key while selecting target device to enable
- Device joins network



Enabling Wireless

iPhone OS 3.1+

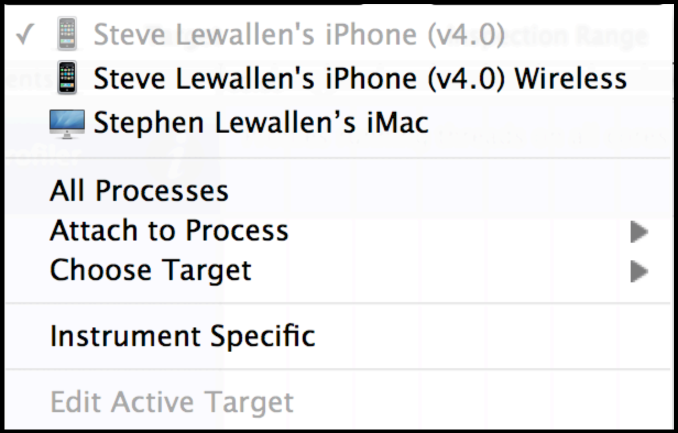
- Hold down option key while selecting target device to enable
- Device joins network
- Device is ready



Enabling Wireless

iPhone OS 3.1+

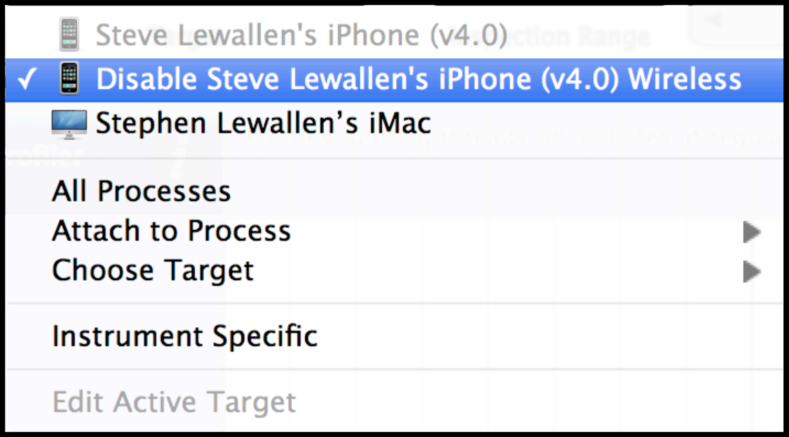
- Hold down option key while selecting target device to enable
- Device joins network
- Device is ready
- Unplug from USB



Disabling Wireless

iPhone OS 3.1+

- Hold down option key while selecting target device to enable



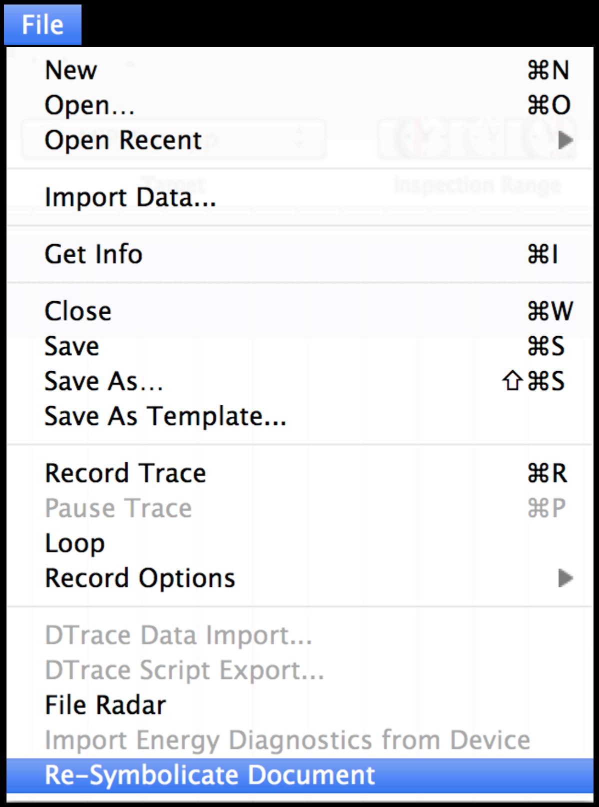
Symbolication

Mac OS X and iPhone

- Symbols normally found via SDK and Spotlight
  - Build “DWARF with dSym File”
  - Place where visible to Spotlight
- Re-symbolicate when necessary

Re-Symbolication

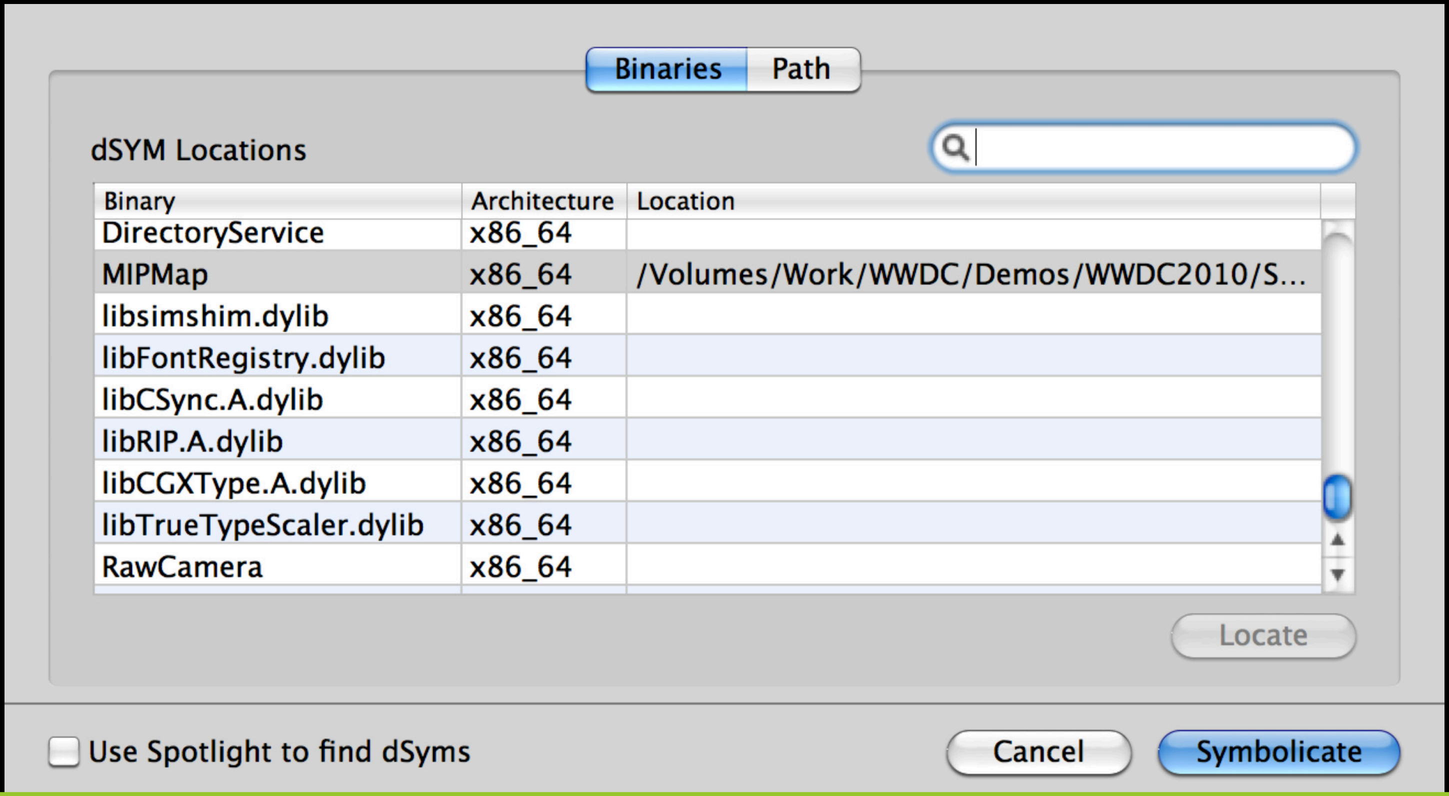
Mac OS X and iPhone OS



Step 1 : Choose Re-Symbolicate Document

Re-Symbolication

Mac OS X and iPhone OS



dSYM Locations

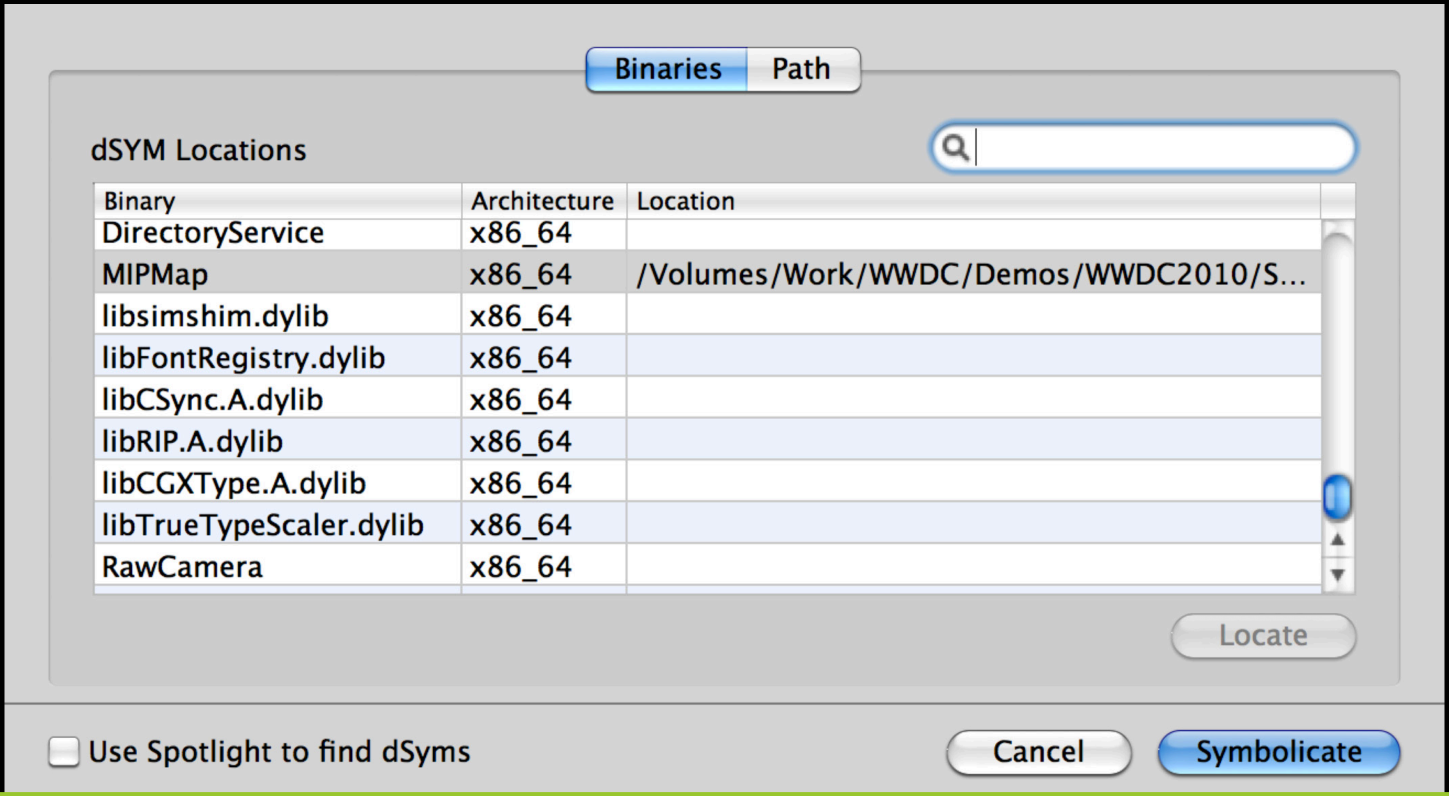
Binary	Architecture	Location
DirectoryService	x86_64	
MIPMap	x86_64	/Volumes/Work/WWDC/Demos/WWDC2010/S...
libsimshim.dylib	x86_64	
libFontRegistry.dylib	x86_64	
libCSync.A.dylib	x86_64	
libRIP.A.dylib	x86_64	
libCGXType.A.dylib	x86_64	
libTrueTypeScaler.dylib	x86_64	
RawCamera	x86_64	

□ Use Spotlight to find dSyms

Step 2 : Choose binaries and paths

Re-Symbolication

Mac OS X and iPhone OS



Binary	Architecture	Location
DirectoryService	x86_64	
MIPMap	x86_64	/Volumes/Work/WWDC/Demos/WWDC2010/S...
libsimshim.dylib	x86_64	
libFontRegistry.dylib	x86_64	
libCSync.A.dylib	x86_64	
libRIP.A.dylib	x86_64	
libCGXType.A.dylib	x86_64	
libTrueTypeScaler.dylib	x86_64	
RawCamera	x86_64	

□ Use Spotlight to find dSyms

Step 3 : Click 'Symbolicate'

iPhone SDK 4

Start/Stop Recording Trace

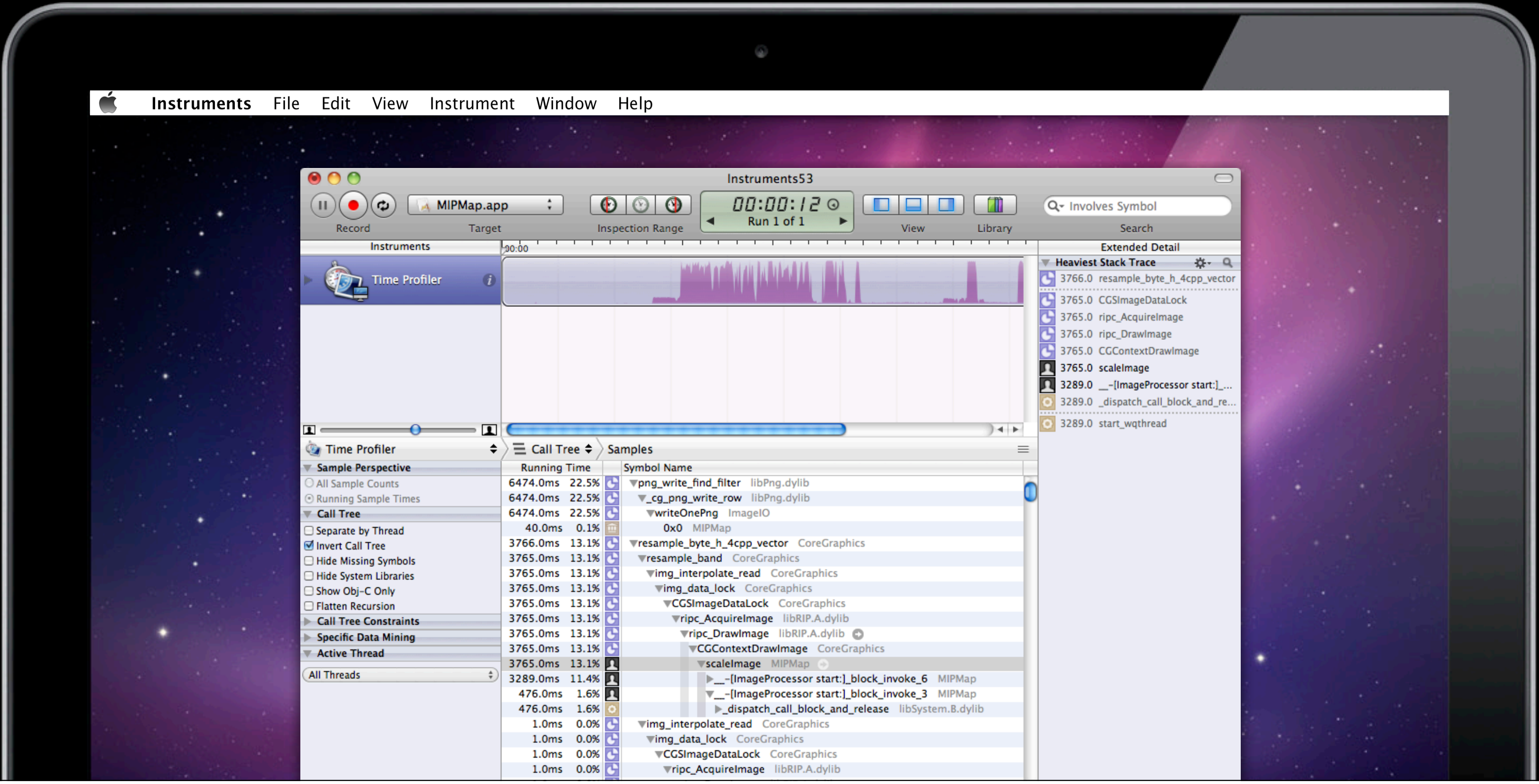
- Hitting the home button no longer terminates application
    - Applications move to the background first
    - To simply stop a trace, hit the stop button in Instruments
- Debugger must relinquish control before using Instruments
    - Always quit/kill the application being debugged before using Instruments

Improved Instrumentation

Major and minor improvements

Time Profiler

Mac OS X and iOS 4



Time Profiler

Sample Perspective
○ All Sample Counts
● Running Sample Times

Call Tree
□ Separate by Thread
☑ Invert Call Tree
□ Hide Missing Symbols
□ Hide System Libraries
□ Show Obj-C Only
□ Flatten Recursion

Call Tree Constraints

Specific Data Mining

Active Thread
All Threads

Call Tree › Samples

```
Running Time      Symbol Name
6474.0ms 22.5%    ▼png_write_find_filter  libPng.dylib
6474.0ms 22.5%      ▼_cg_png_write_row  libPng.dylib
6474.0ms 22.5%        ▼writeOnePng  ImageIO
  40.0ms  0.1%           0x0  MIPMap
3766.0ms 13.1%    ▼resample_byte_h_4cpp_vector  CoreGraphics
3765.0ms 13.1%      ▼resample_band  CoreGraphics
3765.0ms 13.1%        ▼img_interpolate_read  CoreGraphics
3765.0ms 13.1%          ▼img_data_lock  CoreGraphics
3765.0ms 13.1%            ▼CGSImageDataLock  CoreGraphics
3765.0ms 13.1%              ▼ripc_AcquireImage  libRIP.A.dylib
3765.0ms 13.1%                ▼ripc_DrawImage  libRIP.A.dylib
3765.0ms 13.1%                  ▼CGContextDrawImage  CoreGraphics
3765.0ms 13.1%                    ▼scaleImage  MIPMap
3289.0ms 11.4%                      ▶__-[ImageProcessor start:]_block_invoke_6  MIPMap
 476.0ms  1.6%                      ▼__-[ImageProcessor start:]_block_invoke_3  MIPMap
 476.0ms  1.6%                        ▶_dispatch_call_block_and_release  libSystem.B.dylib
   1.0ms  0.0%      ▼img_interpolate_read  CoreGraphics
   1.0ms  0.0%        ▼img_data_lock  CoreGraphics
   1.0ms  0.0%          ▼CGSImageDataLock  CoreGraphics
   1.0ms  0.0%            ▼ripc_AcquireImage  libRIP.A.dylib
```

Heaviest Stack Trace

```
3766.0  resample_byte_h_4cpp_vector
3765.0  CGSImageDataLock
3765.0  ripc_AcquireImage
3765.0  ripc_DrawImage
3765.0  CGContextDrawImage
3765.0  scaleImage
3289.0  __-[ImageProcessor start:]_...
3289.0  _dispatch_call_block_and_re...
3289.0  start_wqthread
```

Time Profiler

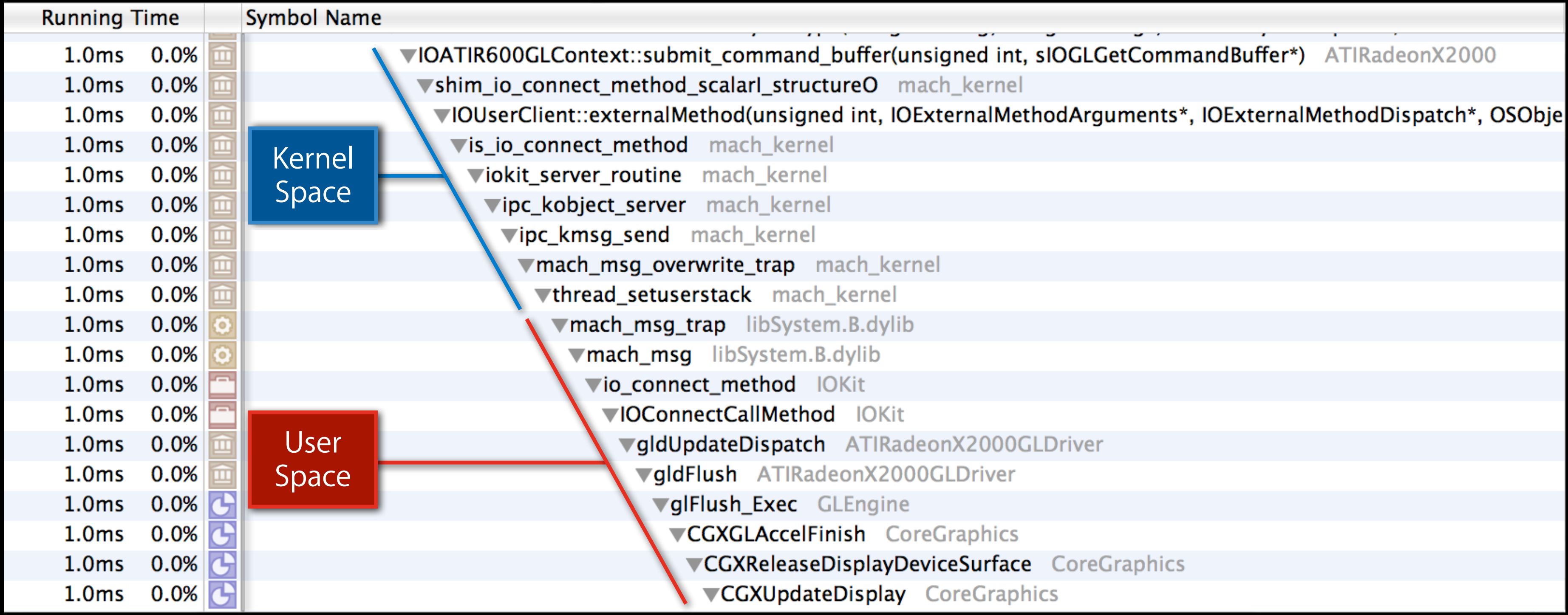
Mac OS X and iOS 4

- Most efficient time profiling mechanism
- Running threads
  - See where code is running
- All thread states
  - See where code is blocked as well as running
- Profile one or all processes

Time Profiler

Mac OS X and iOS 4

- Mac OS X
  - Now with Kernel symbols and unified backtraces



Running Time			Symbol Name	
1.0ms	0.0%		▼IOATIR600GLContext::submit_command_buffer(unsigned int, sIOGLGetCommandBuffer*)	ATIRadeonX2000
1.0ms	0.0%		▼shim_io_connect_method_scalarI_structureO	mach_kernel
1.0ms	0.0%		▼IOUserClient::externalMethod(unsigned int, IOExternalMethodArguments*, IOExternalMethodDispatch*, OSObje	
1.0ms	0.0%		▼is_io_connect_method	mach_kernel
1.0ms	0.0%		▼iokit_server_routine	mach_kernel
1.0ms	0.0%		▼ipc_kobject_server	mach_kernel
1.0ms	0.0%		▼ipc_kmsg_send	mach_kernel
1.0ms	0.0%		▼mach_msg_overwrite_trap	mach_kernel
1.0ms	0.0%		▼thread_setuserstack	mach_kernel
1.0ms	0.0%		▼mach_msg_trap	libSystem.B.dylib
1.0ms	0.0%		▼mach_msg	libSystem.B.dylib
1.0ms	0.0%		▼io_connect_method	IOKit
1.0ms	0.0%		▼IOConnectCallMethod	IOKit
1.0ms	0.0%		▼gldUpdateDispatch	ATIRadeonX2000GLDriver
1.0ms	0.0%		▼gldFlush	ATIRadeonX2000GLDriver
1.0ms	0.0%		▼glFlush_Exec	GLEngine
1.0ms	0.0%		▼CGXGLAccelFinish	CoreGraphics
1.0ms	0.0%		▼CGXReleaseDisplayDeviceSurface	CoreGraphics
1.0ms	0.0%		▼CGXUpdateDisplay	CoreGraphics

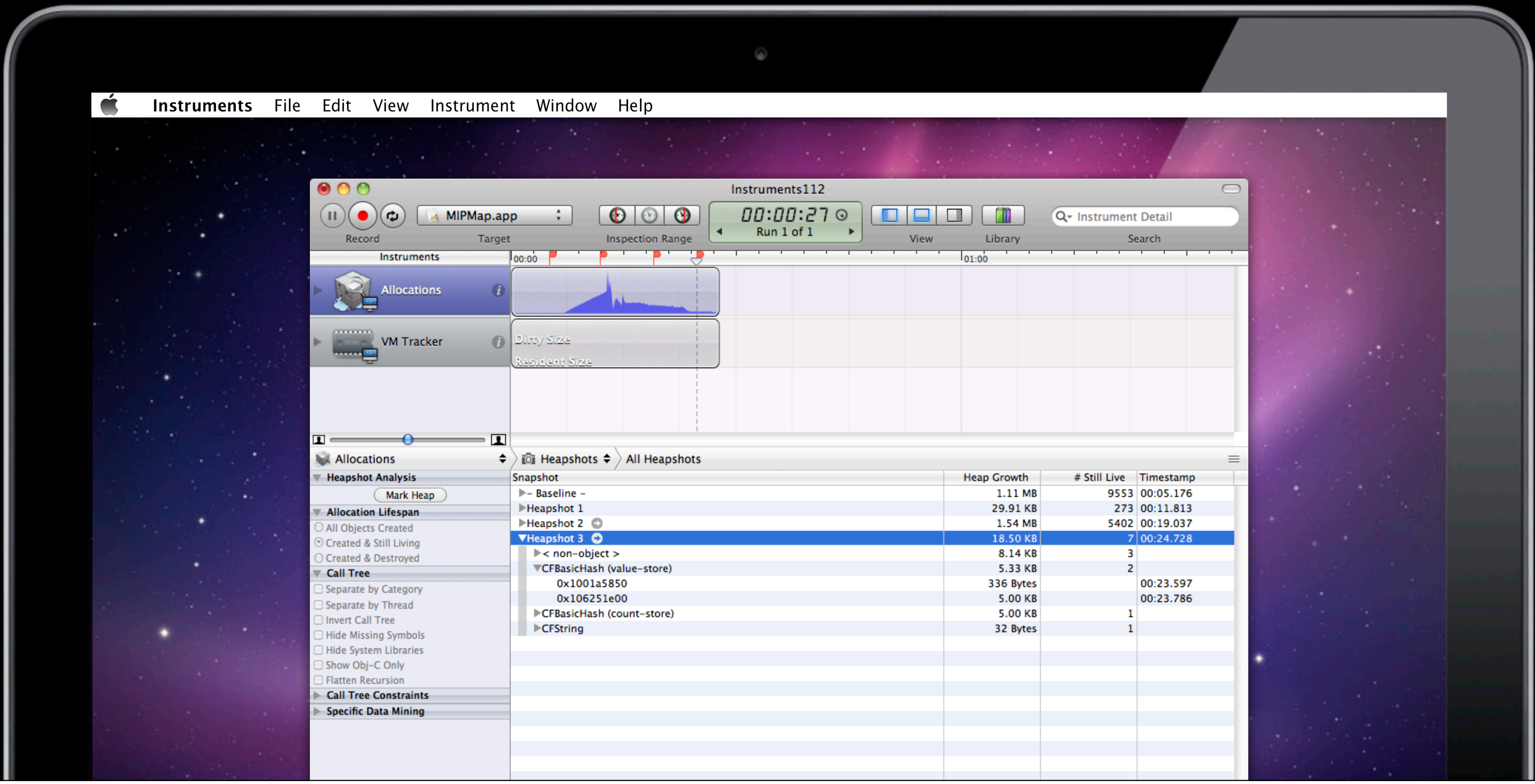
Time Profiler

Mac OS X and iOS 4

- iOS 4
  - New to SDK
  - Far more efficient than CPU Sampler
  - Deferred mode matters!

Heapshots

Mac OS X and iOS 4



Snapshot	Heap Growth	# Still Live	Timestamp
▶– Baseline –	1.11 MB	9553	00:05.176
▶Heapshot 1	29.91 KB	273	00:11.813
▶Heapshot 2	1.54 MB	5402	00:19.037
▼Heapshot 3	18.50 KB	7	00:24.728
▶< non-object >	8.14 KB	3	
▼CFBasicHash (value-store)	5.33 KB	2	
0x1001a5850	336 Bytes		00:23.597
0x106251e00	5.00 KB		00:23.786
▶CFBasicHash (count-store)	5.00 KB	1	
▶CFString	32 Bytes	1	

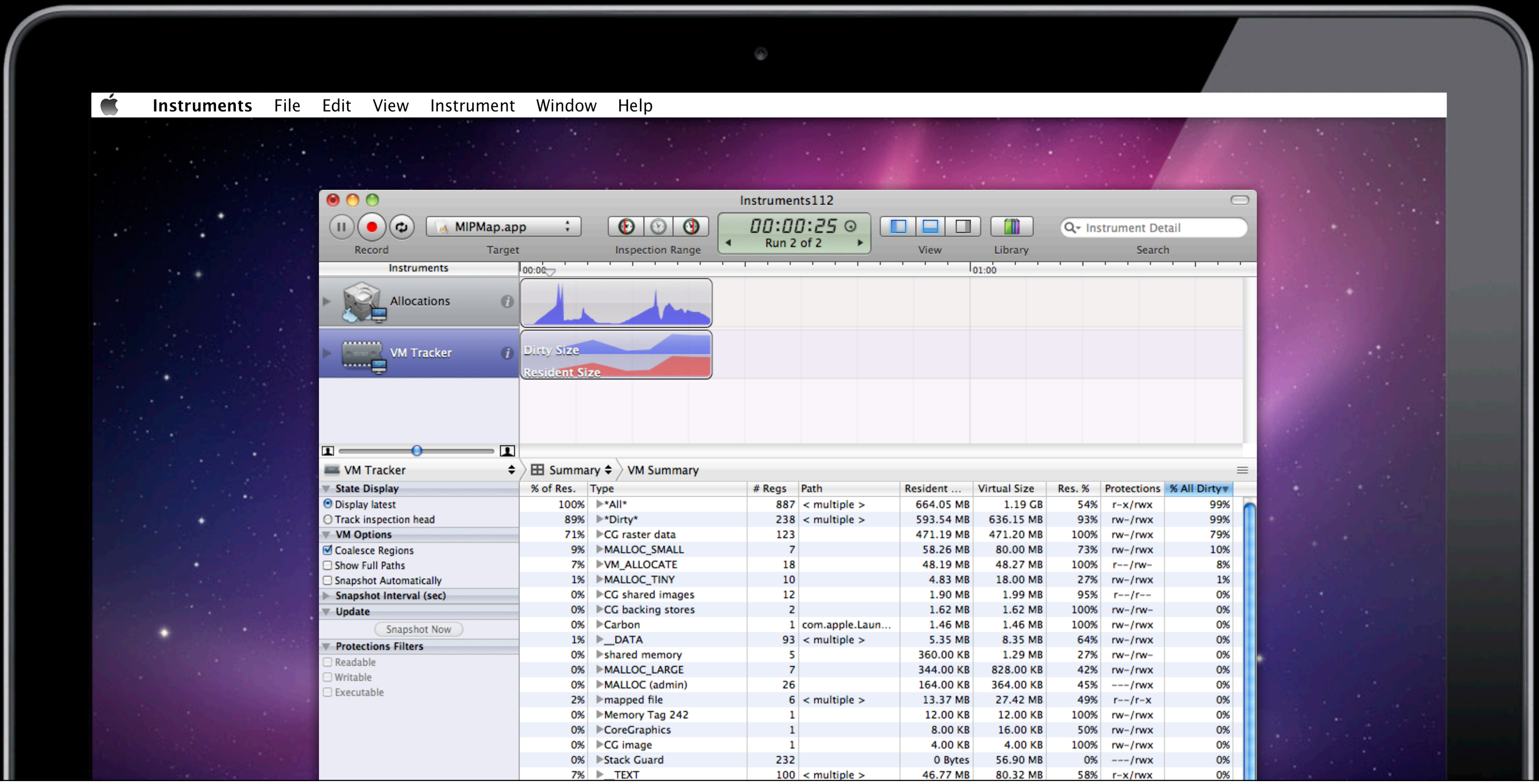
Heapshots

Mac OS X and iOS 4

- Part of Allocations template
  - Function of Allocations Instrument
- Assists with identifying “Abandoned Memory”
  - Memory which is referenced, possibly not needed, that builds up over time
  - Often more of a problem than leaked memory

VM Tracker

Mac OS X and iPhone OS 3.1



VM Tracker

State Display
- ● Display latest
- ○ Track inspection head

VM Options
- ☑ Coalesce Regions
- ☐ Show Full Paths
- ☐ Snapshot Automatically

Snapshot Interval (sec)

Update

Protections Filters
- ☐ Readable
- ☐ Writable
- ☐ Executable

% of Res.	Type	# Regs	Path	Resident ...	Virtual Size	Res. %	Protections	% All Dirty
100%	*All*	887	< multiple >	664.05 MB	1.19 GB	54%	r-x/rwx	99%
89%	*Dirty*	238	< multiple >	593.54 MB	636.15 MB	93%	rw-/rwx	99%
71%	CG raster data	123		471.19 MB	471.20 MB	100%	rw-/rwx	79%
9%	MALLOC_SMALL	7		58.26 MB	80.00 MB	73%	rw-/rwx	10%
7%	VM_ALLOCATE	18		48.19 MB	48.27 MB	100%	r--/rw-	8%
1%	MALLOC_TINY	10		4.83 MB	18.00 MB	27%	rw-/rwx	1%
0%	CG shared images	12		1.90 MB	1.99 MB	95%	r--/r--	0%
0%	CG backing stores	2		1.62 MB	1.62 MB	100%	rw-/rw-	0%
0%	Carbon	1	com.apple.Laun...	1.46 MB	1.46 MB	100%	rw-/rwx	0%
1%	__DATA	93	< multiple >	5.35 MB	8.35 MB	64%	rw-/rwx	0%
0%	shared memory	5		360.00 KB	1.29 MB	27%	rw-/rw-	0%
0%	MALLOC_LARGE	7		344.00 KB	828.00 KB	42%	rw-/rwx	0%
0%	MALLOC (admin)	26		164.00 KB	364.00 KB	45%	---/rwx	0%
2%	mapped file	6	< multiple >	13.37 MB	27.42 MB	49%	r--/r-x	0%
0%	Memory Tag 242	1		12.00 KB	12.00 KB	100%	rw-/rwx	0%
0%	CoreGraphics	1		8.00 KB	16.00 KB	50%	rw-/rwx	0%
0%	CG image	1		4.00 KB	4.00 KB	100%	rw-/rwx	0%
0%	Stack Guard	232		0 Bytes	56.90 MB	0%	---/rwx	0%
7%	__TEXT	100	< multiple >	46.77 MB	80.32 MB	58%	r-x/rwx	0%

VM Tracker

Mac OS X and iPhone OS 3.1

- Part of Allocations template
- Tracks the virtual memory of a process
- Identifies regions by VM tag and reports usage statistics
  - Resident Size
  - Virtual Size
- Use on Core Animation applications
- Associate with image files on disk

Demo

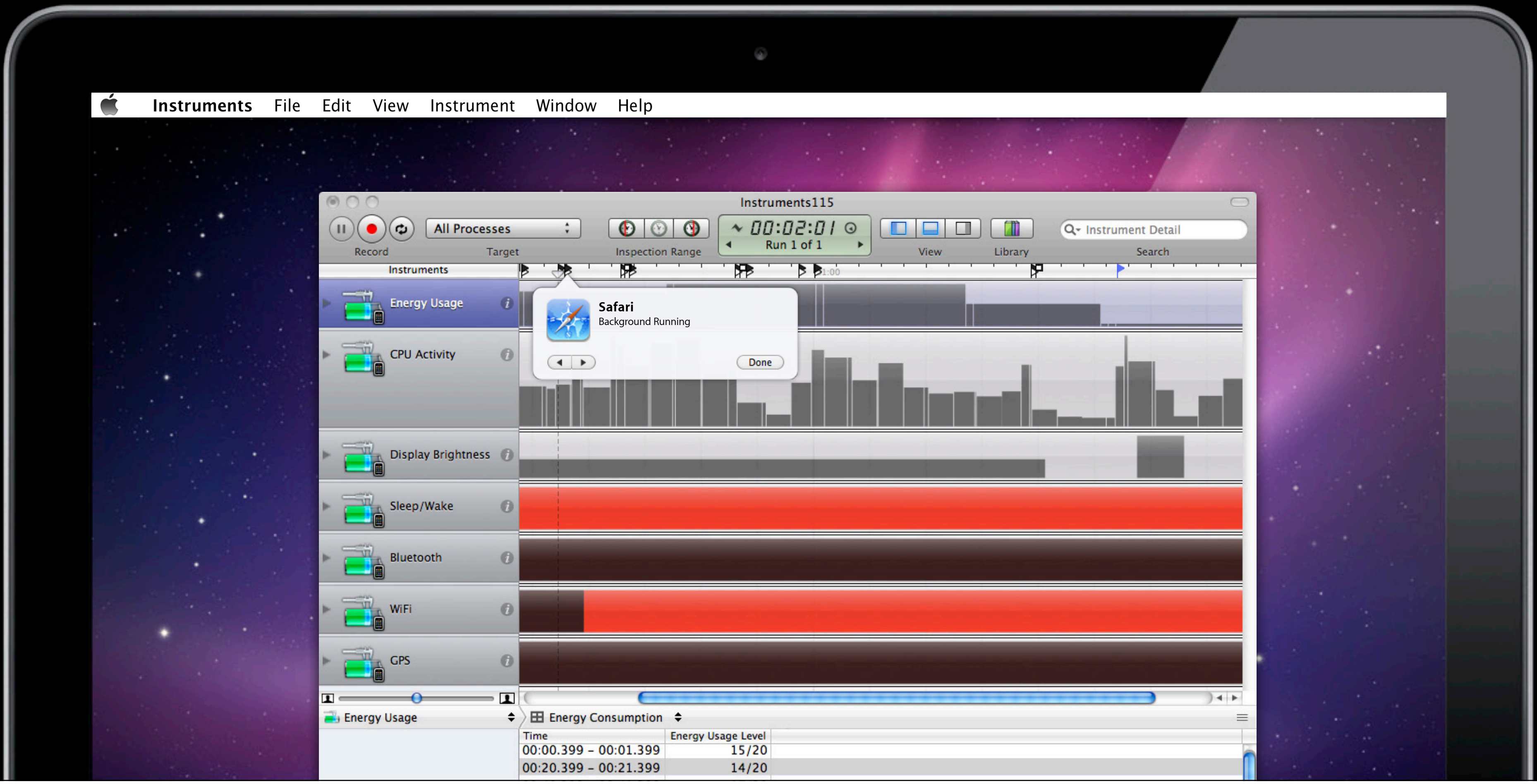
Advanced Recording Techniques

New Instrumentation

New

Energy Diagnostics

iPhone SDK 4 only

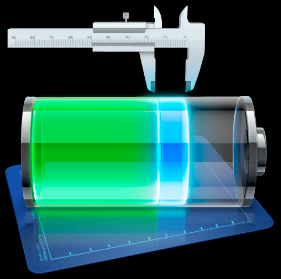


Safari
Background Running

Time	Energy Usage Level
00:00.399 – 00:01.399	15/20
00:20.399 – 00:21.399	14/20

Energy Diagnostics

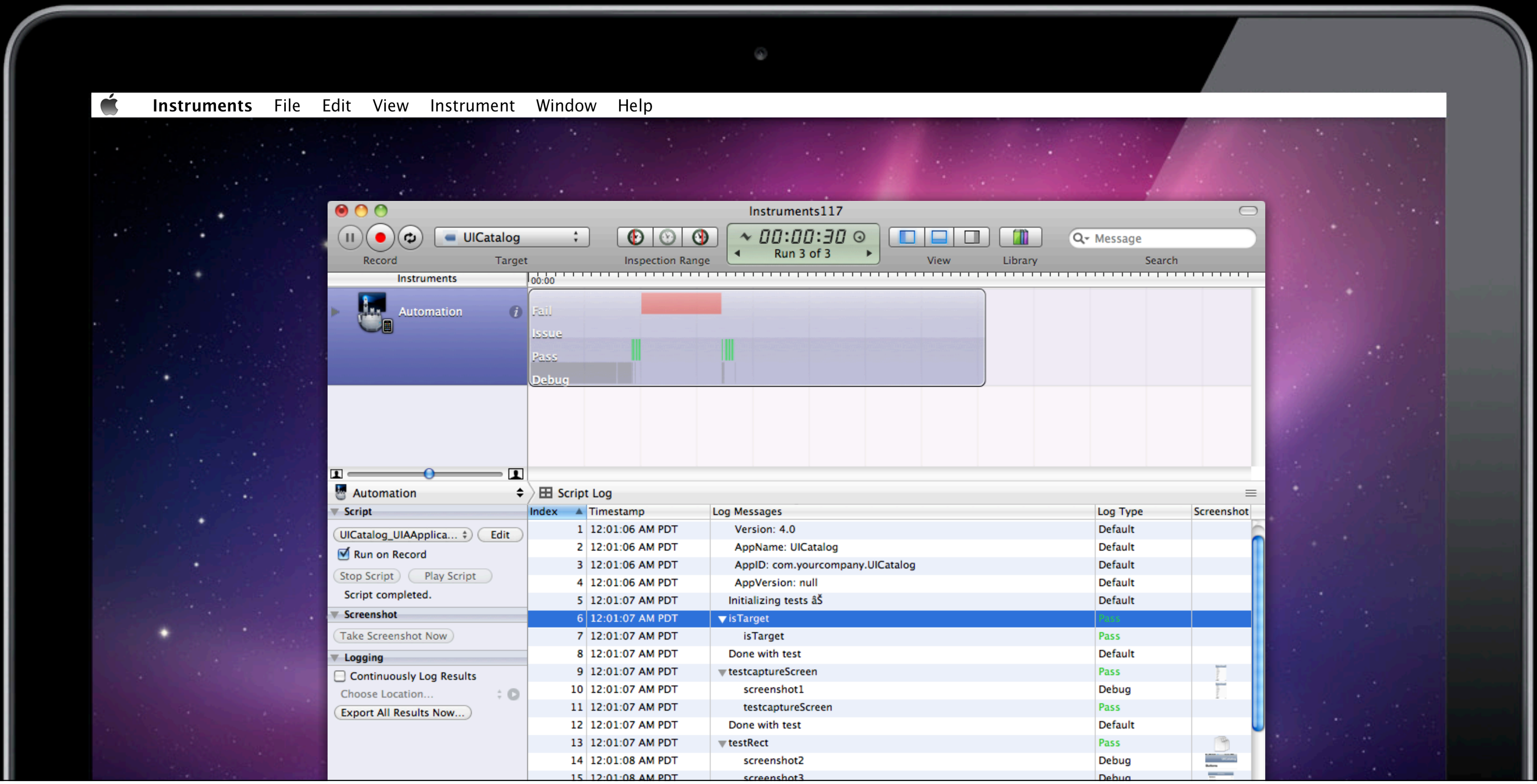
iPhone SDK 4 only



- Provides diagnostics regarding energy usage
- Records battery power and CPU usage
- Tracks on/off state of major device components

Automation

iPhone SDK 4



UICatalog_UIAApplica…

☑ Run on Record

Script completed.

□ Continuously Log Results

Index	Timestamp	Log Messages	Log Type	Screenshot
1	12:01:06 AM PDT	Version: 4.0	Default	
2	12:01:06 AM PDT	AppName: UICatalog	Default	
3	12:01:06 AM PDT	AppID: com.yourcompany.UICatalog	Default	
4	12:01:06 AM PDT	AppVersion: null	Default	
5	12:01:07 AM PDT	Initializing tests âŠ	Default	
6	12:01:07 AM PDT	▼isTarget	Pass	
7	12:01:07 AM PDT	isTarget	Pass	
8	12:01:07 AM PDT	Done with test	Default	
9	12:01:07 AM PDT	▼testcaptureScreen	Pass	
10	12:01:07 AM PDT	screenshot1	Debug	
11	12:01:07 AM PDT	testcaptureScreen	Pass	
12	12:01:07 AM PDT	Done with test	Default	
13	12:01:07 AM PDT	▼testRect	Pass	
14	12:01:08 AM PDT	screenshot2	Debug	
15	12:01:08 AM PDT	screenshot3	Debug	

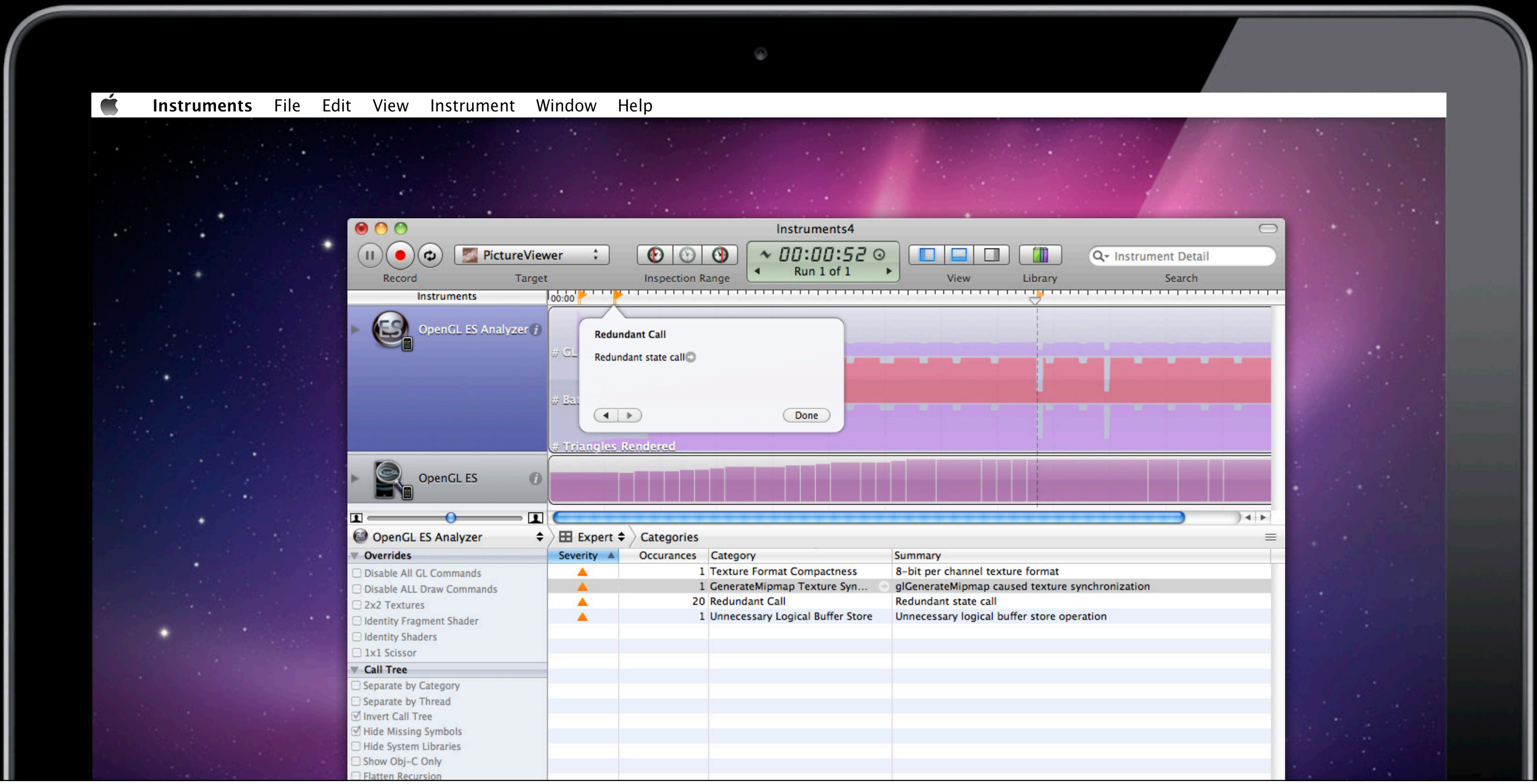
Automation

iPhone SDK 4 only

- Simulate UI interaction with an iPhone OS application
    - Automate QA workflow
    - Use with other instruments
- Leverages JavaScript to script iPhone UI components
    - Script may return results (pass, fail, comments, etc.)
- Results may be exported to other tools

OpenGL ES Analysis

iPhone SDK 4 only with Xcode 4



Redundant Call

Redundant state call

OpenGL ES Analyzer

Overrides

- □ Disable All GL Commands
- □ Disable ALL Draw Commands
- □ 2x2 Textures
- □ Identity Fragment Shader
- □ Identity Shaders
- □ 1x1 Scissor

Call Tree

- □ Separate by Category
- □ Separate by Thread
- ☑ Invert Call Tree
- ☑ Hide Missing Symbols
- □ Hide System Libraries
- □ Show Obj-C Only
- □ Flatten Recursion

Severity ▲	Occurances	Category	Summary
▲	1	Texture Format Compactness	8-bit per channel texture format
▲	1	GenerateMipmap Texture Syn...	glGenerateMipmap caused texture synchronization
▲	20	Redundant Call	Redundant state call
▲	1	Unnecessary Logical Buffer Store	Unnecessary logical buffer store operation

OpenGL ES Analysis

iPhone SDK 4 only with Xcode 4

- Measures and analyzes OpenGL ES activity
    - Provides insight necessary to tune your graphics code
    - Isolates graphics pipeline bottlenecks
- Provides actionable advice to improve performance

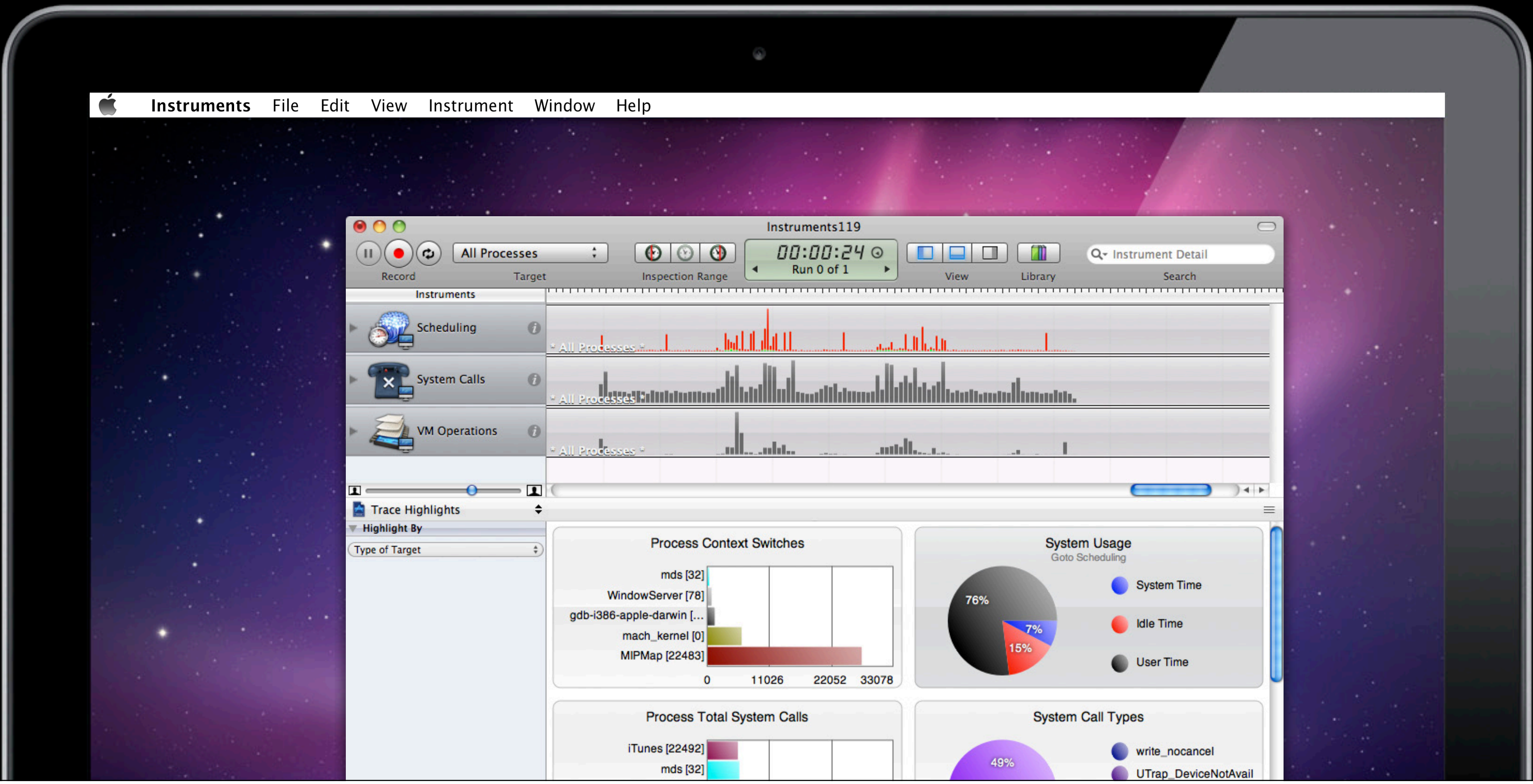
Really New Instrumentation

Features of Instruments in Xcode 4

New

System Trace

Mac OS X



Target: All Processes

Highlight By: Type of Target

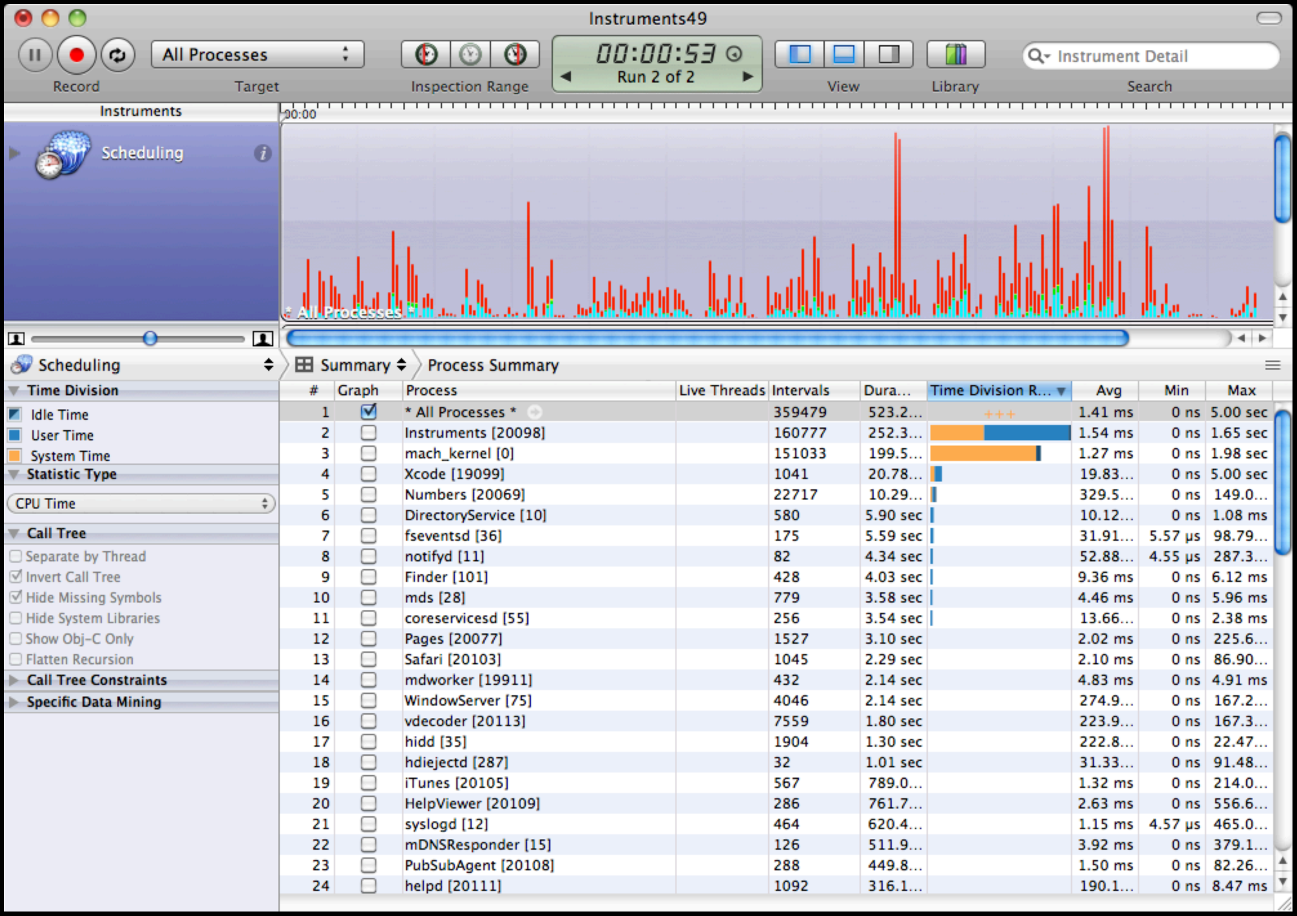
System Trace

Mac OS X

- Provides comprehensive information on system behavior
- Identifies when threads are scheduled and why
- Displays thread transitions from user space into system code
    - System calls
    - VM operations

Scheduling Instrument

Records thread context switches and tenures



Scheduling

Time Division

- ☑ Idle Time
- ☑ User Time
- ☑ System Time

Statistic Type

CPU Time

Call Tree

- □ Separate by Thread
- ☑ Invert Call Tree
- ☑ Hide Missing Symbols
- □ Hide System Libraries
- □ Show Obj-C Only
- □ Flatten Recursion

Call Tree Constraints

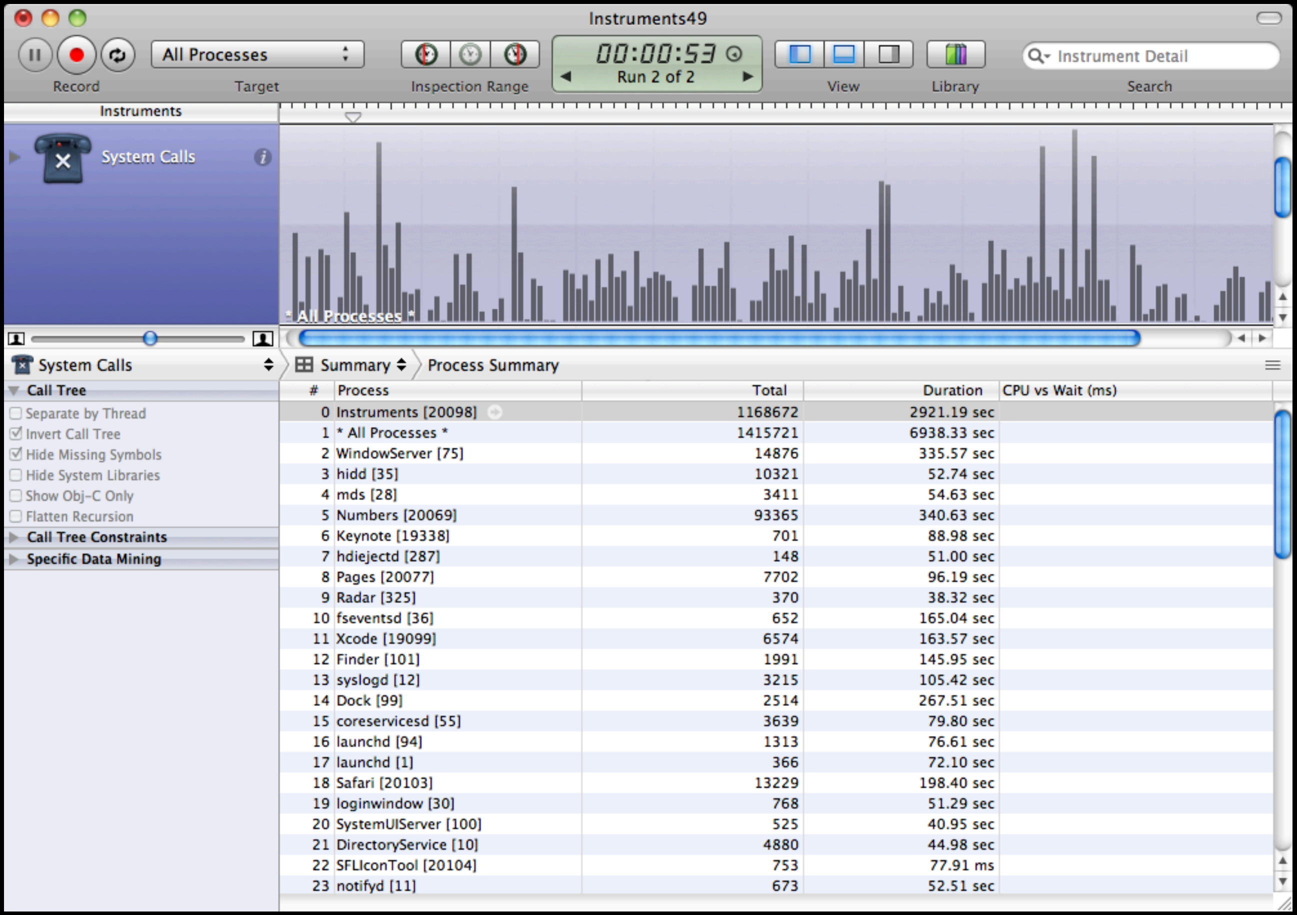
Specific Data Mining

Summary › Process Summary

#	Graph	Process	Live Threads	Intervals	Dura…	Time Division R…	Avg	Min	Max
1	☑	* All Processes *		359479	523.2…	+++	1.41 ms	0 ns	5.00 sec
2	□	Instruments [20098]		160777	252.3…		1.54 ms	0 ns	1.65 sec
3	□	mach_kernel [0]		151033	199.5…		1.27 ms	0 ns	1.98 sec
4	□	Xcode [19099]		1041	20.78…		19.83…	0 ns	5.00 sec
5	□	Numbers [20069]		22717	10.29…		329.5…	0 ns	149.0…
6	□	DirectoryService [10]		580	5.90 sec		10.12…	0 ns	1.08 ms
7	□	fseventsd [36]		175	5.59 sec		31.91…	5.57 μs	98.79…
8	□	notifyd [11]		82	4.34 sec		52.88…	4.55 μs	287.3…
9	□	Finder [101]		428	4.03 sec		9.36 ms	0 ns	6.12 ms
10	□	mds [28]		779	3.58 sec		4.46 ms	0 ns	5.96 ms
11	□	coreservicesd [55]		256	3.54 sec		13.66…	0 ns	2.38 ms
12	□	Pages [20077]		1527	3.10 sec		2.02 ms	0 ns	225.6…
13	□	Safari [20103]		1045	2.29 sec		2.10 ms	0 ns	86.90…
14	□	mdworker [19911]		432	2.14 sec		4.83 ms	0 ns	4.91 ms
15	□	WindowServer [75]		4046	2.14 sec		274.9…	0 ns	167.2…
16	□	vdecoder [20113]		7559	1.80 sec		223.9…	0 ns	167.3…
17	□	hidd [35]		1904	1.30 sec		222.8…	0 ns	22.47…
18	□	hdiejectd [287]		32	1.01 sec		31.33…	0 ns	91.48…
19	□	iTunes [20105]		567	789.0…		1.32 ms	0 ns	214.0…
20	□	HelpViewer [20109]		286	761.7…		2.63 ms	0 ns	556.6…
21	□	syslogd [12]		464	620.4…		1.15 ms	4.57 μs	465.0…
22	□	mDNSResponder [15]		126	511.9…		3.92 ms	0 ns	379.1…
23	□	PubSubAgent [20108]		288	449.8…		1.50 ms	0 ns	82.26…
24	□	helpd [20111]		1092	316.1…		190.1…	0 ns	8.47 ms

System Calls Instrument

Records system calls and their duration



System Calls

Call Tree

- □ Separate by Thread
- ☑ Invert Call Tree
- ☑ Hide Missing Symbols
- □ Hide System Libraries
- □ Show Obj-C Only
- □ Flatten Recursion

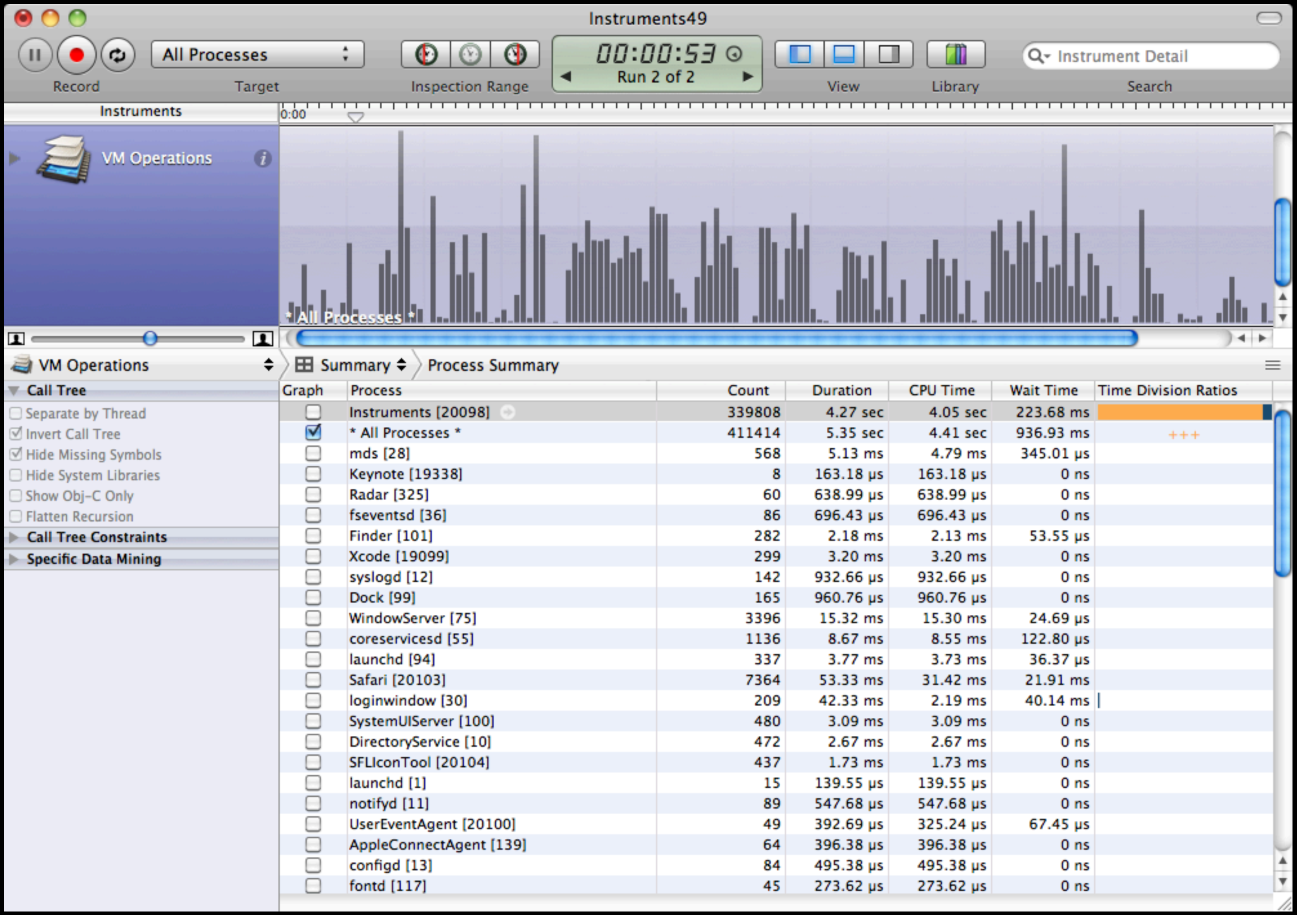
Call Tree Constraints

Specific Data Mining

#	Process	Total	Duration	CPU vs Wait (ms)
0	Instruments [20098]	1168672	2921.19 sec	
1	* All Processes *	1415721	6938.33 sec	
2	WindowServer [75]	14876	335.57 sec	
3	hidd [35]	10321	52.74 sec	
4	mds [28]	3411	54.63 sec	
5	Numbers [20069]	93365	340.63 sec	
6	Keynote [19338]	701	88.98 sec	
7	hdiejectd [287]	148	51.00 sec	
8	Pages [20077]	7702	96.19 sec	
9	Radar [325]	370	38.32 sec	
10	fseventsd [36]	652	165.04 sec	
11	Xcode [19099]	6574	163.57 sec	
12	Finder [101]	1991	145.95 sec	
13	syslogd [12]	3215	105.42 sec	
14	Dock [99]	2514	267.51 sec	
15	coreservicesd [55]	3639	79.80 sec	
16	launchd [94]	1313	76.61 sec	
17	launchd [1]	366	72.10 sec	
18	Safari [20103]	13229	198.40 sec	
19	loginwindow [30]	768	51.29 sec	
20	SystemUIServer [100]	525	40.95 sec	
21	DirectoryService [10]	4880	44.98 sec	
22	SFLIconTool [20104]	753	77.91 ms	
23	notifyd [11]	673	52.51 sec	

VM Operations Instrument

Records virtual memory events



Call Tree

- ☐ Separate by Thread
- ☑ Invert Call Tree
- ☑ Hide Missing Symbols
- ☐ Hide System Libraries
- ☐ Show Obj-C Only
- ☐ Flatten Recursion

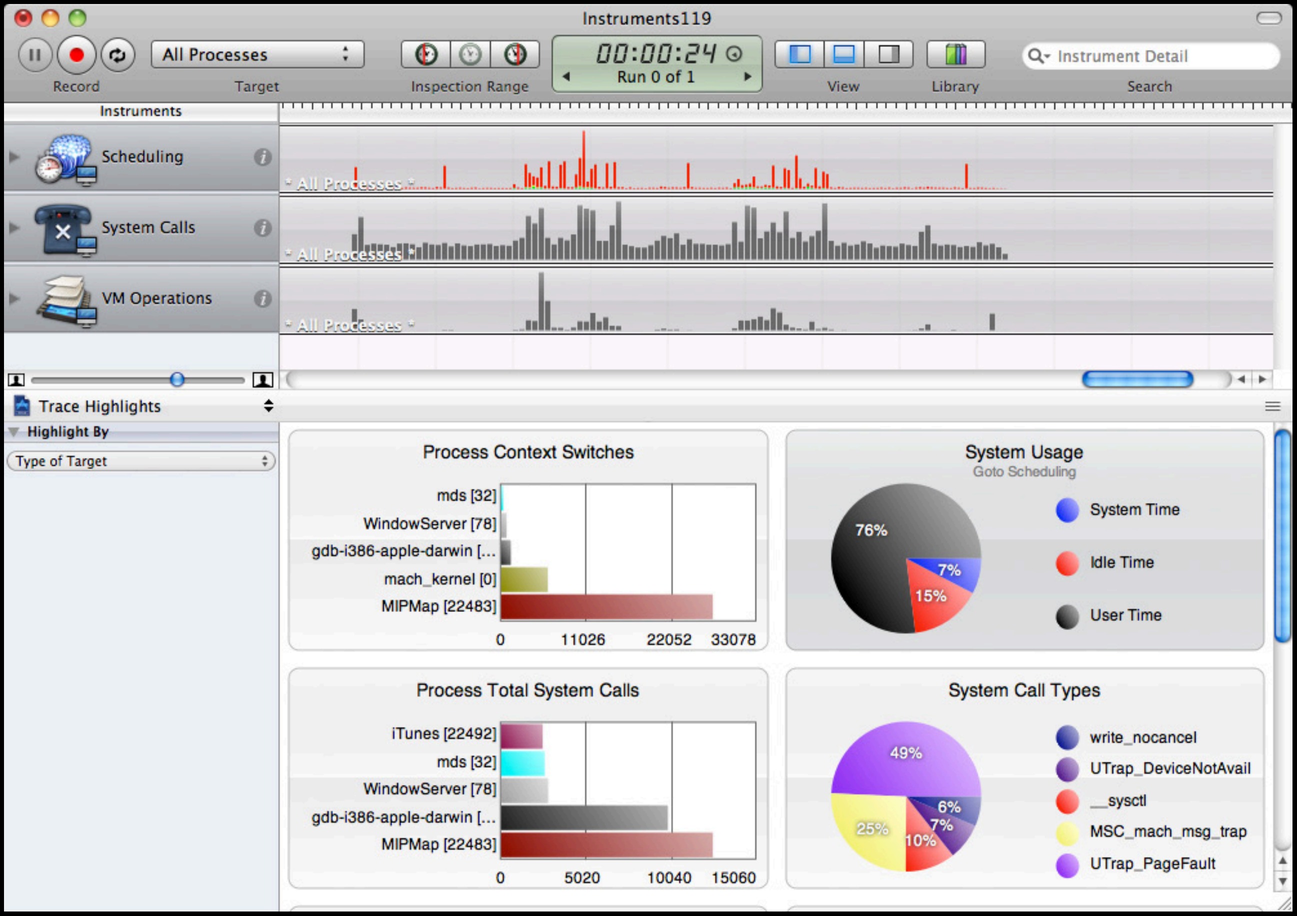
Call Tree Constraints

Specific Data Mining

Graph	Process	Count	Duration	CPU Time	Wait Time	Time Division Ratios
☐	Instruments [20098]	339808	4.27 sec	4.05 sec	223.68 ms	
☑	* All Processes *	411414	5.35 sec	4.41 sec	936.93 ms	+++
☐	mds [28]	568	5.13 ms	4.79 ms	345.01 µs	
☐	Keynote [19338]	8	163.18 µs	163.18 µs	0 ns	
☐	Radar [325]	60	638.99 µs	638.99 µs	0 ns	
☐	fseventsd [36]	86	696.43 µs	696.43 µs	0 ns	
☐	Finder [101]	282	2.18 ms	2.13 ms	53.55 µs	
☐	Xcode [19099]	299	3.20 ms	3.20 ms	0 ns	
☐	syslogd [12]	142	932.66 µs	932.66 µs	0 ns	
☐	Dock [99]	165	960.76 µs	960.76 µs	0 ns	
☐	WindowServer [75]	3396	15.32 ms	15.30 ms	24.69 µs	
☐	coreservicesd [55]	1136	8.67 ms	8.55 ms	122.80 µs	
☐	launchd [94]	337	3.77 ms	3.73 ms	36.37 µs	
☐	Safari [20103]	7364	53.33 ms	31.42 ms	21.91 ms	
☐	loginwindow [30]	209	42.33 ms	2.19 ms	40.14 ms	
☐	SystemUIServer [100]	480	3.09 ms	3.09 ms	0 ns	
☐	DirectoryService [10]	472	2.67 ms	2.67 ms	0 ns	
☐	SFLIconTool [20104]	437	1.73 ms	1.73 ms	0 ns	
☐	launchd [1]	15	139.55 µs	139.55 µs	0 ns	
☐	notifyd [11]	89	547.68 µs	547.68 µs	0 ns	
☐	UserEventAgent [20100]	49	392.69 µs	325.24 µs	67.45 µs	
☐	AppleConnectAgent [139]	64	396.38 µs	396.38 µs	0 ns	
☐	configd [13]	84	495.38 µs	495.38 µs	0 ns	
☐	fontd [117]	45	273.62 µs	273.62 µs	0 ns	

Highlights View

Summary graphs of key statistics



Demo

In Closing…

- Instruments is your go-to tool for performance analysis
- More capabilities than ever
- Easier to use
- Provides accurate insights from the kernel to the user interface

Related Sessions

Advanced Memory Analysis with Instruments	Presidio Thursday 11:30AM
Advanced Performance Analysis with Instruments	Mission Thursday 9:00AM
OpenGL ES Tuning and Optimization	Presidio Wednesday 4:30PM
Performance Optimization on iPhone OS	Presidio Thursday 2:00PM
Advanced Performance Optimization on iPhone OS, Part 1	Mission Thursday 3:15PM
Automated User Interface Testing with Instruments	Marina Wednesday 2:00PM

Labs

Xcode for iPhone Development Lab	Developer Tools Lab B Thursday 2:00PM
Xcode 4 Lab	Multiple Labs Throughout Week Check schedule
Performance Lab	Multiple Labs Throughout Week Check schedule
Automated User Interface Testing Lab	Developer Tools Lab A Wednesday 4:30PM

More Information

Michael Jurewitz
Developer Tools Evangelist
jurewitz@apple.com

Instruments Documentation
Instruments User Guide (Xcode documentation)

Apple Developer Forums
http://devforums.apple.com

WWDC10