Profiling in Depth
Do you know where your code is?
Session 412

Kris Markel Performance Tools Engineer
Chad Woolf Performance Tools Engineer
Time Profiler
Time Profiler

Hotspots

elementFound = [iterator getNextElement:&modelElement];
if (elementFound) {
    // Calculate the bar dimensions in view space.
    const CGFloat pathRectHeight = MIN(modelElement.value * _pointsPerValue,
                                         _viewRect.size.height);
    CGPathMoveToPoint(path, NULL, currentX, _viewRect.size.height);
    CGPathAddLineToPoint(path, NULL, currentX, _viewRect.size.height -
                         pathRectHeight);
    pathCount++;

    // Move to the next point that will hit the next nano.
elementFound = [iterator getNextElement:&modelElement];
if (elementFound) {
  // Calculate the bar dimensions in view space.
  const CGFloat pathRectHeight = MIN(modelElement.value * _pointsPerValue, _viewRect.size.height);
  CGPathMoveToPoint(path, NULL, _viewRect.size.height);
  CGPathAddLineToPoint(path, NULL, _viewRect.size.height - pathRectHeight);
  pathCount++;
  // Move to the next point that will hit the next nano.
Time Profiler

Call Trees
Time Profiler

Call Trees
Session Overview

Time profiling in depth
Session Overview

Time profiling in depth

Motivation
Session Overview

Time profiling in depth

Motivation

Demonstrations/Insights
Session Overview
Time profiling in depth

Motivation
Demonstrations/Insights
Final tips
Creating Instruments 7
Creating Instruments 7

Look and feel
Creating Instruments 7

Look and feel
New graphing styles
Creating Instruments 7

Look and feel
New graphing styles
Faster rendering
elementFound = [iterator getNextElement:&modelElement];
if (elementFound) {
    // Calculate the bar dimensions in view space.
    const CGFloat pathRectHeight = MIN(modelElement.value * _pointsPerValue, _viewRect.size.height);
    CGPathMoveToPoint(path, NULL, currentX, _viewRect.size.height);
    CGPathAddLineToPoint(path, NULL, currentX, _viewRect.size.height - pathRectHeight);
    pathCount++;
    // Move to the next point that will hit the next nano.
Creating Instruments 7

Track View

```c
elementFound = [iterator getNextElement:&modelElement];
if (elementFound) {
    // Calculate the bar dimensions in view space.
    const CGFloat pathRectHeight = MIN(modelElement.value * _pointsPerValue,
                                         _viewRect.size.height);
    CGPathMoveToPoint(path, NULL, currentX, _viewRect.size.height);
    CGPathAddLineToPoint(path, NULL, currentX, _viewRect.size.height - pathRectHeight);
    pathCount++;
    // Move to the next point that will hit the next nano.
```
Creating Instruments 7
Prototyping
Creating Instruments 7

Integration
Graphasaurus
All the other cool names were taken
Graphasaurus

Requirements

100,000 data points
60 fps
iPad Mini 1st Generation
Demo

Getting started
Time Profiler

Missing stack frames

```objc
-(void) dealloc {
    CFRelease(_path);
}

-(void) drawRect:(CGRect) rect {
    CGContextRef context = UIGraphicsGetCurrentContext();
    CGContextSetStrokeColorWithColor(context, [[UIColor purpleColor] CGColor]);
    CGContextAddPath(context, self.path);
    CGContextDrawPath(context, kCGPathStroke);
}

#pragma mark - Properties
```
Backtrace Example

Normal case

- (void) drawRect: (NSRect) rect {
    // draw stuff
    ...
    CGContextDrawPath(_path, kCGStroke);
}
Backtrace Example

Normal case

- (void) drawRect: (NSRect) rect {
   // draw stuff
   ...
   CGContextDrawPath(_path, kCGStroke);
}
Backtrace Example

Normal case

- (void) drawRect: (NSRect) rect {
  // draw stuff
  ...
  CGContextDrawPath(_path, kCGStroke);
}
Backtrace Example

Normal case

-(void) drawRect: (NSRect) rect {
    // draw stuff
    ...
    CGContextDrawPath(_path, kCGStroke);
}

- +0x00 push  {r4, r5, r6, fp, lr}
- +0x02 add   fp, sp, #12
Backtrace Example

Normal case

```c
- (void) drawRect: (NSRect) rect {
    // draw stuff
    ...
    CGContextDrawPath(_path, kCGStroke);
}
```

```
+0x00 push {r4, r5, r6, fp, lr}
+0x02 add fp, sp, #12
+0x04 sub sp, #180
```
Backtrace Example

Normal case

- (void) drawRect: (NSRect) rect {
    // draw stuff
    ...
    CGContextDrawPath(_path, kCGStroke);
}
Backtrace Example

Normal case

- (void) drawRect: (NSRect) rect {
    // draw stuff
    ...
    CGContextDrawPath(_path, kCGStroke);
}

```c
... drawRect:
Return Address | Frame Ptr
---|---
drawLayer:inContext:
Return Address | Frame Ptr
drawInContext:
Return Address | Frame Ptr
CABackingStoreUpdate_
Return Address | Frame Ptr
... main
```

fp →
Backtrace Example

Normal case

-(void) drawRect: (NSRect) rect {
    // draw stuff
    ...
    CGContextDrawPath(_path, kCGStroke);
}
Backtrace Example

Normal case

- (void) drawRect: (NSRect) rect {
  // draw stuff
  ...
  CGContextDrawPath(_path, kCGStroke);
}
Normal case

- (void) drawRect: (NSRect) rect {
  // draw stuff
  ... CGContextDrawPath(_path, kCGStroke);
}

-fomit-frame-pointer
Backtrace Example

Normal case

- (void) drawRect: (NSRect) rect {
   // draw stuff
   ... 
   CGContextDrawPath(_path, kCGStroke);
}

-fomit-frame-pointer
Backtrace Example

Optimized case

- (void) drawRect: (NSRect) rect {
    // draw stuff
    ...
    CGContextDrawPath(_path, kCGStroke);
}

CABackingStoreUpdate_

drawRect:

drawLayer:inContext:

drawInContext:

main
Backtrace Example

Optimized case

- (void) drawRect: (NSRect) rect {
    // draw stuff
    ... 
    CGContextDrawPath(_path, kCGStroke);
    pop stack frame
    restore fp
    jump back to caller
}
- (void) drawRect: (NSRect) rect {
  // draw stuff
  ...
  pop stack frame
  restore fp
  CGContextDrawPath(_path, kCGStroke);
  jump back to caller
}
Backtrace Example

Optimized case

-(void)drawRect:(NSRect)rect {
    // draw stuff
    ...
    pop stack frame
    restore fp
    CGContextDrawPath(_path, kCGStroke);
}

drawRect:
<table>
<thead>
<tr>
<th>Return Address</th>
<th>Frame Ptr</th>
</tr>
</thead>
<tbody>
<tr>
<td>drawLayer:inContext:</td>
<td></td>
</tr>
<tr>
<td>Return Address</td>
<td>Frame Ptr</td>
</tr>
<tr>
<td>drawInContext:</td>
<td></td>
</tr>
<tr>
<td>Return Address</td>
<td>Frame Ptr</td>
</tr>
<tr>
<td>CABackingStoreUpdate_</td>
<td></td>
</tr>
<tr>
<td>Return Address</td>
<td>Frame Ptr</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>main</td>
<td></td>
</tr>
</tbody>
</table>
Backtrace Example

Optimized case

- (void) drawRect: (NSRect) rect {
    // draw stuff
    ...
    pop stack frame
    restore fp
    CGContextDrawPath(_path, kCGStroke);
}

+0x68  add    sp, #180
- (void) drawRect: (NSRect) rect {
    // draw stuff
    ...
    pop stack frame
    restore fp
    CGContextDrawPath(_path, kCGStroke);
}

drawLayer:inContext:
frame

drawInContext:
frame

CABackingStoreUpdate_
frame

main

+0x68 add sp, #180
+0x72 pop.w {r4, r5, r6, fp, lr}
Backtrace Example

Optimized case

- (void) drawRect: (NSRect) rect {
    // draw stuff
    ...
    pop stack frame
    restore fp
    CGContextDrawPath(_path, kCGStroke);
}

CGContextDrawPath

Return Address | Frame Ptr
-----------------|-----------------|
drawLayer:inContext: | |
drawInContext: | |
CABackingStoreUpdate_ | |
main | |

+0x68    add    sp, #180
+0x72    pop.w  {r4, r5, r6, fp, lr}
+0x76    b.w    "CGContextDrawPath$shim"
Backtrace Example

Optimized case

- (void) drawRect: (NSRect) rect {
  // draw stuff
  ...
  pop stack frame
  restore fp
  CGContextDrawPath(_path, kCGStroke);
}

+0x68  add    sp, #180
+0x72  pop.w   {r4, r5, r6, fp, lr}
+0x76  b.w    "CGContextDrawPath$shim"
Tail Call Elimination

Benefits
Tail Call Elimination

Benefits

Saves stack memory
Tail Call Elimination

Benefits

Saves stack memory
Keeps caches hot
Tail Call Elimination

Benefits

Saves stack memory

Keeps caches hot

Best for recursive function tail calls
Tail Call Elimination

Disabling

CFLAGS="-fno-optimize-sibling-calls"
Call Semantics

ARM (32-bit)

- Branch and Link instruction (bl)
  - Sets “lr” to next address and jumps
- Tail Calls use simple branches (b)

Caller:
+0x174 blx "CGContextDrawPath $shim"

Caller:
+0x174 b.w "CGContextDrawPath $shim"
Demo

Going further
Objective-C Runtime
objc_msgSend
Objective-C Runtime

objc_msgSend

NSString *string = @"Hello";
NSUInteger len = [string length];  // or string.length
Objective-C Runtime

**objc_msgSend**

```objective-c
NSString *string = @"Hello";
NSUInteger len = [string length];  // or string.length
```

Looks up the method implementation for a selector
Objective-C Runtime
objc_msgSend

NSString *string = @"Hello";
NSUInteger len = [string length];  // or string.length

Looks up the method implementation for a selector
Invokes the method
Objective-C Runtime

objc_msgSend

NSString *string = @"Hello";
NSUInteger len = [string length];  // or string.length

Looks up the method implementation for a selector
Invokes the method
Fast and highly optimized
Objective-C Runtime
objc_msgSend

NSString *string = @"Hello";
NSUInteger len = [string length];  // or string.length

Looks up the method implementation for a selector
Invokes the method
Fast and highly optimized
Does not push a stack frame
Objective-C Runtime
objc_msgSend
Objective-C Runtime

`objc_msgSend`

Required when

- Invoking methods on classes and objects
- Accessing properties
Objective-C Runtime

**objc_msgSend**

Required when

- Invoking methods on classes and objects
- Accessing properties

Method caching not as fast as inlining
Objective-C Runtime
objc_msgSend

Required when
• Invoking methods on classes and objects
• Accessing properties

Method caching not as fast as inlining

Alternatives
• C functions
• Objective-C++ (including STL)
Swift

Performance
Swift

Performance

Only dynamic when it needs to be
Swift

Performance

Only dynamic when it needs to be
Classes can be internal
Swift

Performance

Only dynamic when it needs to be
Classes can be internal
Whole module optimization
Demo
Going further
Time Profiler

Going further
Time Profiler
Going further

More options to explore
• Record Waiting Threads
Time Profiler

Going further

More options to explore

• Record Waiting Threads
• Invert Call Tree
Time Profiler

Going further

More options to explore

- Record Waiting Threads
- Invert Call Tree
- Data Mining

Charge `-TGNGraphView drawRect:` to callers
Prune `-TGNGraphView drawRect:` and subtrees
Charge 'Graphasaurus' to callers
Flatten 'Graphasaurus' to boundary frames

Focus on subtree
Focus on calls made by `-TGNGraphView drawRect:`
Focus on callers of `-TGNGraphView drawRect:`
Focus on calls made by 'Graphasaurus'
Focus on callers of 'Graphasaurus'

Reveal in Xcode
Time Profiling in Depth

Lessons learned
Time Profiling in Depth

Lessons learned

Incorporate performance targets early
Time Profiling in Depth

Lessons learned

Incorporate performance targets early
Always measure
Time Profiling in Depth

Lessons learned

Incorporate performance targets early
Always measure
Keep digging
More Information

Swift Language Documentation
http://developer.apple.com/swift

Apple Developer Forums
http://developer.apple.com/forums

Stefan Lesser
Developer Tools Evangelist
slesser@apple.com
## Related Sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Date/Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debugging Energy Issues</td>
<td>Nob Hill</td>
<td>Wednesday 10:00AM</td>
</tr>
<tr>
<td>Performance on iOS and watchOS</td>
<td>Presidio</td>
<td>Friday 11:00AM</td>
</tr>
</tbody>
</table>
### Related Labs

| Instruments and Debugging Lab | Developer Tools Lab B | Friday 9:00AM |