Visual Debugging with Xcode

Session 410

Chris Miles  Xcode Debugger UI
Tyler Casella  Game Technologies
Daniel Delwood  Software Radiologist
Not too long ago…

Debugging
Debugging
Later...
Debugging

Now…
Overview

Runtime issues
View debugging
State machine Quick Looks
SpriteKit/SceneKit FPS gauge
Memory graph debugging
Runtime Issues
Runtime Issues
Runtime Issues

```swift
func resetUpdateState() {
    updateFinished = true
}

override func viewWillAppear(_ animated: Bool) {
    super.viewWillAppear(animated)

    // Scroll to middle image
    scrollView.scrollView.scrollToVisible(CGRect(x: scrollView.bounds.size.width, y: 0, width: scrollView.bounds.size.width, height: 20), animated: true)
}

override func willTransition(to newCollection: UITraitCollection, with coordinator: UIViewControllerTransitionCoordinator) {
    updateUIForTraitCollection(newCollection)
}
```
Runtime Issues

```swift
fillDescriptionTextView()
updateUIForTraitCollection(self.navigationController!.traitCollection)
loadInitialData()

routeEstimator?.requestUpdate(completion: { success in
    self.updateDidFinish(withSuccess: success)
})

func resetUpdateState() {
    updateFinished = true
}
```
Runtime Issues

```swift
func resetUpdateState() {
    updateFinished = true
}
```
Runtime Issues

- Threading Issues
  - Use of an uninitialized or destroyed mutex in CommitChange
- Data race in Trailblazer.TrailDetailsViewController.resetUpdateState () -> () at updateFinished
  - 'updateFinished' is a global variable (0x1031d8b68)
- Write of size 1 by thread 15
  - 0 TrailDetailsViewController.resetUpdateState()
  - 1 TrailDetailsViewController.notify() -> ()
  - 2 TrailDetailsViewController.updateDidFinish()
  - 3 TrailDetailsViewController.viewDidLoad()
Runtime Issues

- Threading Issues
  - Use of an uninitialized or destroyed mutex in CommitChange
  - Data race in Trailblazer.TrailDetailsViewController .resetUpdateState () -> () at updateFinished
    - 'updateFinished' is a global variable (0x1031d8b68)
    - Write of size 1 by thread 15
      - 0 TrailDetailsViewController.resetUpdateState()
      - 1 TrailDetailsViewController.notify() -> ()
      - 2 TrailDetailsViewController.updateDidFinishState(
        - 3 TrailDetailsViewController.viewDidLoad()

override func viewDidLoad()
    super.viewDidLoad()
    // Scroll to middle
    scrollView.scrollTo

Runtime Issues
Participating tools
Runtime Issues

Participating tools

Threads
Runtime Issues
Participating tools

UI
Threads
Runtime Issues
Participating tools

UI
Threads
Memory
Runtime Issues

Thread Sanitizer

Data races
Use of uninitialized mutexes
Unlock from wrong thread
Thread leaks
Unsafe calls in signal handlers
View Debugging
(lldb) po [self.view recursiveDescription]
<UIview: 0x7fcca9f613f68; frame = (0 0; 375 667); autoresize = RM+BM; layer = <CALayer: 0x7fcca9f614000>>
  <UIScrollView: 0x7fcca181e400; frame = (0 0; 375 667); clipsTo Bounds = YES; autoresize = RM+BM; gestureRecognizers = <NSArray 0x7fcca9f614270; frame = (0 0; 375 1673.5); autoresize = RM+BM; layer = <CALayer: 0x7fcca9f6141300>>
    <UILabel: 0x7fcca9f614400; frame = (166.5 24; 42 20.5); text = 'Label'; opaque = NO; autoresize = RM+BM; userInteractionEnabled = NO>
    <UIButton: 0x7fcca9f614a10; frame = (164.5 52.5; 46 30); opaque = NO; autoresize = RM+BM; layer = <CALayer: 0x7fcca9f614b70>; text = 'Button'; opaque = NO; userInteractionEnabled = NO>
    <UIButtonLabel: 0x7fcca13c9fd0; frame = (0 6; 46 18); text = 'Label'; opaque = NO; userInteractionEnabled = NO>
    <UISegmentedControl: 0x7fcca9f615080; frame = (127 138; 121 29); opaque = NO; autoresize = RM+BM; layer = <CALayer: 0x7fcca9f615d70>>
      <UISegment: 0x7fcca9f6158a0; frame = (61 0; 60 29); opaque = NO; layer = <CALayer: 0x7fcca9f615d70>>
        <UISegmentLabel: 0x7fcca9f6161d0; frame = (7 6.5; 46 16); text = 'First'; opaque = NO; userInteractionEnabled = NO>
        <UIImageView: 0x7fcca9f72a360; frame = (60 0; 1 29); alpha = 0; opaque = NO; autoresize = LM; userInteractionEnabled = NO>
      <UISegment: 0x7fcca1341e10; frame = (60 0; 60 29); opaque = NO; layer = <CALayer: 0x7fcca1310cb0>>
        <UISegmentLabel: 0x7fcca134bb0; frame = (6 0; 1 29); opaque = NO; autoresize = LM; userInteractionEnabled = NO>
        <UITextField: 0x7fcca9f70fd50; frame = (98.5 246; 178 30); text = ''; clipsToBounds = YES; opaque = NO; autoresize = NO; userInteractionEnabled = NO>
        <UITextField RoundedRect Background View Neue: 0x7fcca1267240; frame = (0 0; 178 30); opaque = NO; autoresize = NO; userInteractionEnabled = NO>
        <UITextField: 0x7fcca1265b70; frame = (7 8.5; 164 27.5); text = 'This is a placeholder'; opaque = NO; userInteractionEnabled = NO>
        <UITextField: 0x7fcca1115e30; frame = (142.5 285; 98.5 24.5); text = 'This is text'; clipsToBounds = YES; opaque = NO; autoresize = NO; userInteractionEnabled = NO>
        <UITextField: 0x7fcca11163d0; frame = (2 2; 87 20.5); text = 'This is text'; opaque = NO; userInteractionEnabled = NO>
      <UIButton: 0x7fcca1268fa0; frame = (226.5 56.5; 22 22); opaque = NO; autoresize = RM+BM; layer = <CALayer: 0x7fcca13c74c0>>
      <UIImageView: 0x7fcca13c74c0; frame = (0 0; 22 22); clipsToBounds = YES; opaque = NO; userInteractionEnabled = NO>
View Debugging
View Debugging
View Debugging
View Debugging
View Debugging
View Debugging
Better than ever
View Debugging
Better than ever

Up to 70% faster snapshots
View Debugging
Better than ever

Up to 70% faster snapshots
Layout and transform accuracy
View Debugging
Better than ever

Up to 70% faster snapshots
Layout and transform accuracy
Blur rendering
View Debugging
Better than ever

Up to 70% faster snapshots
Layout and transform accuracy
Blur rendering
Jump to class
View Debugging
Better than ever

Up to 70% faster snapshots
Layout and transform accuracy
Blur rendering
Jump to class
Navigator filtering
View Debugging
Better than ever

Up to 70% faster snapshots
Layout and transform accuracy
Blur rendering
Jump to class
Navigator filtering
Auto Layout debugging
Ambiguous layouts are reported as runtime issues

- Highlighted in the activity viewer
- Listed in the issue navigator
Ambiguous layout issues are badged in the view hierarchy outline.
Ambiguous layout issues are explained in the view’s size inspector.

<table>
<thead>
<tr>
<th>Constraints</th>
</tr>
</thead>
<tbody>
<tr>
<td>Width and horizontal position are ambiguous.</td>
</tr>
<tr>
<td>superview.trailing = self.trailing @ 1000</td>
</tr>
<tr>
<td>self.width = 79 @ 1000 (content size)</td>
</tr>
<tr>
<td>self.centerY = label.centerY @ 1000</td>
</tr>
<tr>
<td>self.height = 14 @ 1000 (content size)</td>
</tr>
<tr>
<td>self.leading = label.trailing + 4 @ 1000</td>
</tr>
<tr>
<td>label.trailing = self.trailing @ 1000</td>
</tr>
</tbody>
</table>

Content Hugging Priority
- Horizontal: 291
- Vertical: 261

Content Compression Resistance Priority
- Horizontal: 751
- Vertical: 751
Ambiguous layout issues are explained in the view’s size inspector.

Constraints

- Width and horizontal position are ambiguous.
- superview.trailing = self.trailing @ 1000
- self.width = 79 @ 1000 (content size)
- self.centerY = label.centerY @ 1000
- self.height = 14 @ 1000 (content size)
- self.leading ≥ label.trailing + 4 @ 1000
- label.trailing = self.trailing @ 1000
Demo
Xcode view debugging
Recap

Runtime issues

View debugging enhancements

• Ambiguous layout issue reporting

• macOS, iOS, tvOS
State Machine Quick Look

Tyler Casella Game Technologies
Quick Look
Xcode 7

Images
Geometry
Views
Colors

SpriteKit
SceneKit
Custom

...and more!
Quick Look
Xcode 8

State Machine
State Machine Quick Look

GKStateMachine

Available via GameplayKit
- macOS, iOS, tvOS
State Machine Quick Look

GKStateMachine

Available via GameplayKit

- macOS, iOS, tvOS

Directed graph defining complex behavior
State Machine Quick Look

GKStateMachine

Available via GameplayKit

• macOS, iOS, tvOS

Directed graph defining complex behavior

Provide discrete behavior per-state
State Machine Quick Look

GKStateMachine

Available via GameplayKit
- macOS, iOS, tvOS

Directed graph defining complex behavior
Provide discrete behavior per-state
Define transitions between states
State Machine Quick Look

GKStateMachine

Available via GameplayKit
• macOS, iOS, tvOS

Directed graph defining complex behavior
Provide discrete behavior per-state
Define transitions between states
Difficult to visualize from code
State Machine Quick Look

GKStateMachine

Available via GameplayKit
- macOS, iOS, tvOS

Directed graph defining complex behavior
Provide discrete behavior per-state
Define transitions between states
Difficult to visualize from code
State Machine Quick Look
Then and now

(lldb) po machine.currentState
  ▼ Optional<GKState>
    ▼ some : <DemoBots.BeamIdleState: 0x174026c00>

(lldb) po machine.canEnterState(BeamIdleState)
false

(lldb) po machine.canEnterState(BeamFiringState)
true

(lldb) po machine.canEnterState(BeamCoolingState)
false

(lldb) po machine.canEnterState(BeamDisabledState)
false

(lldb) po machine.canEnterState(BeamChargingState)
false

Xcode 7.3
State Machine Quick Look

Then and now

Xcode 7.3

Then:

```swift
(lldb) po machine.currentState
▿ Optional<GKState> some : <DemoBots.BeamIdleState: 0x174026c00>
```

Now:

```swift
(lldb) po machine.canEnterState(BeamIdleState) false
(lldb) po machine.canEnterState(BeamFiringState) true
(lldb) po machine.canEnterState(BeamCoolingState) false
(lldb) po machine.canEnterState(BeamDisabledState) false
(lldb) po machine.canEnterState(BeamChargingState) false
```

Xcode 8.0
State Machine Quick Look

Examples

![State Machine Diagram](image-url)
State Machine Quick Look

Examples

![State Machine Diagrams](image-url)
FPS Performance Gauge
FPS Performance Gauge

Real-time performance
FPS Performance Gauge

Real-time performance
FPS Performance Gauge
Real-time performance

Frame rate
FPS Performance Gauge
Real-time performance

Frame rate
FPS Performance Gauge
Real-time performance

Frame rate
GPU utilization
FPS Performance Gauge

Real-time performance

Frame rate

GPU utilization
FPS Performance Gauge
Real-time performance

Frame rate
GPU utilization
CPU / GPU frame time
FPS Performance Gauge

Real-time performance

Frame rate
GPU utilization
CPU / GPU frame time
FPS Performance Gauge
Real-time performance

Breakdown of update loop
• Render
• Client update
• Actions
• Physics

Easy to identify bottlenecks
Available on iOS and watchOS
FPS Performance Gauge

Real-time performance

Breakdown of update loop

• Render
• Client update
• Actions
• Physics

Easy to identify bottlenecks

Available on iOS and watchOS
FPS Performance Gauge
Real-time performance

Breakdown of update loop

• Render
• Client update
• Actions
• Physics

Easy to identify bottlenecks

Available on iOS and watchOS
FPS Performance Gauge
Real-time performance

Breakdown of update loop
• Render
• Client update
• Actions
• Physics

Easy to identify bottlenecks
Available on iOS and watchOS
FPS Performance Gauge
Real-time performance

Breakdown of update loop
- Render
- Client update
- Actions
- Physics

Easy to identify bottlenecks
Available on iOS and watchOS
FPS Performance Gauge

Real-time performance

Breakdown of update loop
- Render
- Client update
- Actions
- Physics

Easy to identify bottlenecks

Available on iOS and watchOS
FPS Performance Gauge
Real-time performance

Breakdown of update loop
• Render
• Client update
• Actions
• Physics

Easy to identify bottlenecks
Available on iOS and watchOS
Demo
State machine Quick Look and FPS performance gauge
Recap

State machine Quick Look
FPS performance gauge
Memory Graph Debugging

Daniel Delwood  Software Radiologist
"Why does this object exist?"
### Memory Graph Debugging

```bash
$ heap DemoBots [--addresses='.*Action']
```

### Zone DefaultMallocZone_0x121016000: 61176 nodes (68571200 bytes)

<table>
<thead>
<tr>
<th>COUNT</th>
<th>BYTES</th>
<th>AVG</th>
<th>CLASS_NAME</th>
<th>TYPE</th>
<th>BINARY</th>
</tr>
</thead>
<tbody>
<tr>
<td>22875</td>
<td>63015952</td>
<td>2754.8</td>
<td>non-object</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8898</td>
<td>429360</td>
<td>48.3</td>
<td>__NSCFString</td>
<td>ObjC</td>
<td>CoreFoundation</td>
</tr>
<tr>
<td>2402</td>
<td>576480</td>
<td>240.0</td>
<td>SKTexture</td>
<td>ObjC</td>
<td>SpriteKit</td>
</tr>
<tr>
<td>2183</td>
<td>209568</td>
<td>96.0</td>
<td>CUIRenditionKey</td>
<td>ObjC</td>
<td>CoreUI</td>
</tr>
<tr>
<td>1124</td>
<td>71936</td>
<td>64.0</td>
<td>NSConcreteData</td>
<td>ObjC</td>
<td>Foundation</td>
</tr>
<tr>
<td>1085</td>
<td>260400</td>
<td>240.0</td>
<td>CGImage</td>
<td>CFType</td>
<td>CoreGraphics</td>
</tr>
<tr>
<td>1080</td>
<td>171504</td>
<td>158.8</td>
<td>CGImageProvider</td>
<td>CFType</td>
<td>CoreGraphics</td>
</tr>
<tr>
<td>1075</td>
<td>172000</td>
<td>160.0</td>
<td>SKTextureCache</td>
<td>ObjC</td>
<td>SpriteKit</td>
</tr>
<tr>
<td>1070</td>
<td>256800</td>
<td>240.0</td>
<td>CGDataProvider</td>
<td>CFType</td>
<td>CoreGraphics</td>
</tr>
<tr>
<td>1065</td>
<td>187440</td>
<td>176.0</td>
<td>CUIRenditionMetrics</td>
<td>ObjC</td>
<td>CoreUI</td>
</tr>
<tr>
<td>1059</td>
<td>84720</td>
<td>80.0</td>
<td>CUIRenditionLayerReference</td>
<td>ObjC</td>
<td>CoreUI</td>
</tr>
<tr>
<td>1053</td>
<td>1027728</td>
<td>976.0</td>
<td>_CUIInternalLinkRendition</td>
<td>ObjC</td>
<td>CoreUI</td>
</tr>
<tr>
<td>1052</td>
<td>70160</td>
<td>66.7</td>
<td>NSPathStore2</td>
<td>ObjC</td>
<td>Foundation</td>
</tr>
<tr>
<td>814</td>
<td>39072</td>
<td>48.0</td>
<td>NSMutableArray</td>
<td>ObjC</td>
<td>CoreFoundation</td>
</tr>
<tr>
<td>582</td>
<td>64240</td>
<td>110.4</td>
<td>NSArray (Object Storage)</td>
<td>C</td>
<td>CoreFoundation</td>
</tr>
<tr>
<td>470</td>
<td>22560</td>
<td>48.0</td>
<td>NSmutableDictionary</td>
<td>ObjC</td>
<td>CoreFoundation</td>
</tr>
<tr>
<td>451</td>
<td>7216</td>
<td>16.0</td>
<td>NSArray</td>
<td>ObjC</td>
<td>CoreFoundation</td>
</tr>
<tr>
<td>444</td>
<td>21312</td>
<td>48.0</td>
<td>Swift._NSContiguousString</td>
<td>Swift</td>
<td>libswiftCore.dylib</td>
</tr>
<tr>
<td>411</td>
<td>22048</td>
<td>53.6</td>
<td><strong>NSMallocBlock</strong></td>
<td>ObjC</td>
<td>CoreFoundation</td>
</tr>
</tbody>
</table>
Memory Graph Debugging

$ leaks DemoBots --trace=0x7fc2e9e83c90
$ malloc_history DemoBots 0x7fc2e9e83c90

Memory Graph Debugging

ALLOC 0x7fc2e9e83c90-0x7fc2e9e83c90 [size=112]: thread_112f48000 | start | main | UIApplicationMain | GSEventRunModal |
CFRunLoopRunSpecific | __CFRunLoopRun | __CFRunLoopDoTimers | __CFRunLoopDoTimer |
__CFRUNLOOP_IS_CALLING_OUT_TO_A_TIMER_CALLBACK_FUNCTION__ | CA::Display::DisplayLink::dispatch_items(unsigned long long, unsigned long, unsigned long) | CA::Display::DisplayLink::dispatch_items(unsigned long long, unsigned long, unsigned long) |
CA::Display::DisplayLink::dispatch_items(unsigned long long, unsigned long, unsigned long) | __29-[SKView setUpRenderCallback]_block_invoke | -[SKView _vsyncRenderForTime:preRender:postRender:]_block_invoke.312 | -[SKView _vsyncRenderForTime:preRender:postRender:]_block_invoke.312 |
CA::Display::DisplayLink::dispatch_items(unsigned long long, unsigned long, unsigned long) | __29-[SKView setUpRenderCallback]_block_invoke | [-SKView _vsyncRenderForTime:preRender:postRender:]_block_invoke.312 | -[SKView _update:] | -[SKScene _update:] | @objc DemoBots.LevelScene.update (Swift.Double) -> () | DemoBots.LevelScene.update (Swift.Double) -> () | DemoBots.PlayerBot.update (withDeltaTime : Swift.Double) -> () | [SKNode runAction:] | [SKRepeat copyWithZone:] | [+SKRepeat repeatActionForever:] | -[SKSequence copyWithZone:] | +[SKSequence sequenceWithActions:] | -[SKAction copyWithZone:] | -[SKWait init] | -[SKAction init] | operator new(unsigned long) | malloc

FREE 0x7fc2e9e83c90-0x7fc2e9e83c90 [size=112]: thread_112f48000 | start | main | UIApplicationMain | GSEventRunModal |
CFRunLoopRunSpecific | __CFRunLoopRun | __CFRunLoopDoTimers | __CFRunLoopDoTimer |
__CFRUNLOOP_IS_CALLING_OUT_TO_A_TIMER_CALLBACK_FUNCTION__ | CA::Display::DisplayLink::dispatch_items(unsigned long long, unsigned long, unsigned long) | CA::Display::DisplayLink::dispatch_items(unsigned long long, unsigned long, unsigned long) |
CA::Display::DisplayLink::dispatch_items(unsigned long long, unsigned long, unsigned long) | __29-[SKView setUpRenderCallback]_block_invoke | -[SKView _update:] | -[SKScene _update:] | @objc DemoBots.LevelScene.update (Swift.Double) -> () | DemoBots.LevelScene.update (Swift.Double) -> () | DemoBots.PlayerBot.update (withDeltaTime : Swift.Double) -> () | [SKNode runAction:] | [SKRepeat copyWithZone:] | [+SKRepeat repeatActionForever:] | -[SKSequence copyWithZone:] | +[SKSequence sequenceWithActions:] | -[SKAction copyWithZone:] | -[SKWait init] | -[SKAction init] | operator delete(void*)

ALLOC 0x7fc2e9e83c90-0x7fc2e9e83c90 [size=112]: thread_112f48000 | start | main | UIApplicationMain | GSEventRunModal |
CFRunLoopRunSpecific | __CFRunLoopRun | __CFRunLoopDoTimers | __CFRunLoopDoTimer |
__CFRUNLOOP_IS_CALLING_OUT_TO_A_TIMER_CALLBACK_FUNCTION__ | CA::Display::DisplayLink::dispatch_items(unsigned long long, unsigned long, unsigned long) | CA::Display::DisplayLink::dispatch_items(unsigned long long, unsigned long, unsigned long) |
CA::Display::DisplayLink::dispatch_items(unsigned long long, unsigned long, unsigned long) | __29-[SKView setUpRenderCallback]_block_invoke | -[SKView _update:] | -[SKScene _update:] | @objc DemoBots.LevelScene.update (Swift.Double) -> () | DemoBots.LevelScene.update (Swift.Double) -> () | DemoBots.PlayerBot.update (withDeltaTime : Swift.Double) -> () | [SKNode runAction:] | [SKRepeat copyWithZone:] | [+SKRepeat repeatActionForever:] | -[SKSequence copyWithZone:] | +[SKSequence sequenceWithActions:] | -[SKAction copyWithZone:] | -[SKWait init] | -[SKAction init] | operator delete(void*)

malloc_history DemoBots 0x7fc2e9e83c90
Memory Graph Debugging
Memory Graph Debugging
Memory Graph Debugging
Memory Graph Debugging
Demo
Memory graph debugging
Memory Graph Debugging
Leaked and abandoned memory
Memory Graph Debugging

Leaked and abandoned memory

Debugger mode, pauses to inspect app

- Available on macOS, iOS 10, tvOS 10, watchOS 3
Memory Graph Debugging

Leaked and abandoned memory

Debugger mode, pauses to inspect app

- Available on macOS, iOS 10, tvOS 10, watchOS 3

Two graph styles:

- Root paths
  - Referenced memory
  - How is the memory held by globals/stacks?
Memory Graph Debugging

Leaked and abandoned memory

Debugger mode, pauses to inspect app

• Available on macOS, iOS 10, tvOS 10, watchOS 3

Two graph styles:

• Root paths
  - Referenced memory
  - How is the memory held by globals/stacks?

• Cycles
  - Leaked memory
  - How does the leak reference other leaks?
Memory Graph Debugging

Stack logging integration
Memory Graph Debugging

Stack logging integration

Opt-in via Diagnostics scheme tab

- All Allocations
  - MallocStackLogging=1
Memory Graph Debugging

Stack logging integration

Opt-in via Diagnostics scheme tab

- All Allocations
  - MallocStackLogging=1

- Live Allocations Only
  - Less memory/disk overhead
  - MallocStackLogging=lite
Memory Graph Debugging
Introducing .memgraph
Introducing .memgraph

Within Xcode:

• Save: File → “Export Memory Graph…”
• Load: double-click or drag to Xcode
  - No process in debugger — no backtraces, Quick Look, ‘po’
Memory Graph Debugging
Introducing .memgraph

Within Xcode:

• Save: File → “Export Memory Graph…”
• Load: double-click or drag to Xcode
  - No process in debugger — no backtraces, Quick Look, ‘po’

From command-line:

$ leaks --outputGraph=<path> <process>  # creates .memgraph file
$ {leaks|vmmmap|heap} <path/to/file.memgraph> [options]  # operates on .memgraph file
Memory Graph Debugging

Usage tips
Memory Graph Debugging

Usage tips

Graph is conservative
Memory Graph Debugging

Usage tips

Graph is conservative

- Avoids ‘leaks’ false-positives, but there may be extraneous references
Memory Graph Debugging

Usage tips

Graph is conservative

• Avoids ‘leaks’ false-positives, but there may be extraneous references
• Gray references are unknown, may be stale pointer or not strong
Memory Graph Debugging

Usage tips

Graph is conservative

- Avoids ‘leaks’ false-positives, but there may be extraneous references
- Gray references are unknown, may be stale pointer or not strong
  - Enabling Malloc Scribble may improve accuracy
Memory Graph Debugging

Usage tips

Graph is conservative

• Avoids ‘leaks’ false-positives, but there may be extraneous references
• Gray references are unknown, may be stale pointer or not strong
  - Enabling Malloc Scribble may improve accuracy
• Bold references are known to be strong
Memory Graph Debugging

Usage tips

Graph is conservative

- Avoids ‘leaks’ false-positives, but there may be extraneous references
- Gray references are unknown, may be stale pointer or not strong
  - Enabling Malloc Scribble may improve accuracy
- Bold references are known to be strong
  - Swift 3's reflection data more accurate
Memory Graph Debugging

Usage tips

Graph is conservative

- Avoids ‘leaks’ false-positives, but there may be extraneous references
- Gray references are unknown, may be stale pointer or not strong
  - Enabling Malloc Scribble may improve accuracy
- Bold references are known to be strong
  - Swift 3's reflection data more accurate

Requires turning off sanitizers
Memory Graph Debugging

Where to start
Memory Graph Debugging

Where to start

Validate your expectations

• Are there more objects of your types than you expect?
• Are objects deallocated when they’re no longer necessary?
Memory Graph Debugging
Where to start

Validate your expectations
• Are there more objects of your types than you expect?
• Are objects deallocated when they’re no longer necessary?

Find the path that shouldn’t be holding your object
• Strong captures from blocks and closures
• Back-references that should be weak/unowned
Summary

New and improved visual tools in Xcode 8
Built right into your debugging workflow
Try them out, improve your App today!
More Information

https://developer.apple.com/wwdc16/410
### Related Sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Date/Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Trace in Depth</td>
<td>Nob Hill</td>
<td>Thursday 9:00AM</td>
</tr>
<tr>
<td>Thread Sanitizer and Static Analysis</td>
<td>Nob Hill</td>
<td>Thursday 10:00AM</td>
</tr>
<tr>
<td>Debugging Tips and Tricks</td>
<td>Pacific Heights</td>
<td>Friday 1:40PM</td>
</tr>
<tr>
<td>Using Time Profiler in Instruments</td>
<td>Nob Hill</td>
<td>Friday 3:00PM</td>
</tr>
<tr>
<td>Labs</td>
<td>Location</td>
<td>Time</td>
</tr>
<tr>
<td>---------------------------</td>
<td>---------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>GameplayKit Lab</td>
<td>Graphic, Games and Media Lab B</td>
<td>Tuesday 10:10AM</td>
</tr>
<tr>
<td>Profiling and Debugging Lab</td>
<td>Tools Lab C</td>
<td>Thursday 3:00PM</td>
</tr>
<tr>
<td>SceneKit Lab</td>
<td>Graphic, Games and Media Lab A</td>
<td>Thursday 3:00PM</td>
</tr>
<tr>
<td>SpriteKit Lab</td>
<td>Graphic, Games and Media Lab B</td>
<td>Friday 12:00PM</td>
</tr>
</tbody>
</table>