Metal Game Performance Optimization

Session 612

Guillem Viñals Gangolells, GPU Software Performance
Ohad Frenkel, Game Technologies
Develop awesome games
Develop technically awesome games
The Talos Principle
Croteam / Devolver Digital

Features supported in iOS
Physically-based HDR rendering
Dynamic shadows
Real-time reflections
Multi-threaded rendering
Full-screen anti-aliasing
Post processing effects
Profiling Tools
Frame Pacing
Thread Priorities
Thermal States
Unnecessary GPU Work
Profiling Tools
Profile early and often
Know Your Tools

Instruments

Xcode Metal Frame Debugger
Game Performance Template

Game Performance
Game Performance Template
Game Performance Template

System Trace

NEW
Game Performance Template

System Trace

Time Profiler

NEW
Game Performance Template

System Trace

Time Profiler

Metal System Trace
Game Performance

Understand the key performance areas critical for game performance and smooth frame rates.
Choose a profiling template for: iPhone (12.0) Talos

- Blank
- Activity Monitor
- Core Animation
- Core Data
- Counters
- Energy Log
- File Activity
- Leaks
- Metal System Trace
- Network
- SceneKit
- System Trace
- System Usage
- Time Profiler
- Zombies

---

Game Performance

Understand the key performance areas critical for game performance and smooth frame rates.

Open an Existing File...
Recording in Windowed Mode
System Trace and Time Profiler
System Trace and Time Profiler

System Trace in Depth

Using Time Profiler in Instruments
Metal System Trace
Metal System Trace
Metal System Trace

Metal Performance Optimization Techniques  WWDC 2015

Metal 2 Optimization and Debugging  WWDC 2017
Thread States View

[Diagram showing thread states with various states such as blocked, running, etc.]
CPU
Frame Pacing
Micro Stuttering

Inconsistent frame rate
• Frame time higher than display refresh interval
• Game logic timing errors
Naive Approach

Present as fast as possible

// Render Scene
...

// Get drawable and present
if let drawable = view.currentDrawable {
    // Render Final Pass
    ...
    commandBuffer.present(drawable)
}
commandBuffer.commit()
Naive Approach

Present as fast as possible

```swift
// Render Scene
...

// Get drawable and present
if let drawable = view.currentDrawable {
    // Render Final Pass
    ...
    commandBuffer.present(drawable)
}
commandBuffer.commit()
```
Do not usleep() to pace frames!
Naive Approach

Present as fast as possible

<table>
<thead>
<tr>
<th>VSYNC</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Naive Approach

Present as fast as possible

<table>
<thead>
<tr>
<th>VSYNC</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display</td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Naive Approach

Present as fast as possible

<table>
<thead>
<tr>
<th>VSYNC</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>C</td>
<td></td>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td></td>
<td>B</td>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display</td>
<td></td>
<td></td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Naive Approach

Present as fast as possible

<table>
<thead>
<tr>
<th>VSYNC</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td></td>
<td>C</td>
<td></td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td></td>
<td>B</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display</td>
<td></td>
<td>A</td>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Naive Approach

Present as fast as possible

<table>
<thead>
<tr>
<th>VSYNC</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>C</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>Display</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td>B</td>
<td>C</td>
<td>A</td>
<td></td>
</tr>
</tbody>
</table>
Naive Approach

Present as fast as possible

<table>
<thead>
<tr>
<th>VSYNC</th>
<th>CPU</th>
<th>GPU</th>
<th>Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>C</td>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>1</td>
<td>A</td>
<td>C</td>
<td>B</td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>3</td>
<td>C</td>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>4</td>
<td>A</td>
<td>C</td>
<td>B</td>
</tr>
<tr>
<td>5</td>
<td>B</td>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>6</td>
<td>C</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>7</td>
<td>A</td>
<td>C</td>
<td>B</td>
</tr>
<tr>
<td>8</td>
<td>B</td>
<td>A</td>
<td>C</td>
</tr>
<tr>
<td>9</td>
<td>C</td>
<td>A</td>
<td>B</td>
</tr>
<tr>
<td>10</td>
<td>A</td>
<td>C</td>
<td>A</td>
</tr>
</tbody>
</table>
Best Practice

Target explicit frame rate

Use the following APIs (iOS 10.3+)

- `MTLDrawable addPresentedHandler`
- `MTLCommandBuffer presentDrawable afterMinimumDuration`
- `MTLCommandBuffer presentDrawable atTime`
Explicit Frame Pacing

Minimum frame duration of 33ms

<table>
<thead>
<tr>
<th>VSYNC</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Explicit Frame Pacing

Minimum frame duration of 33ms

VSYNC  0  1  2  3  4  5  6  7  8  9  10

CPU    C
GPU    B  C
Display A
Explicit Frame Pacing

Minimum frame duration of 33ms

VSYNC 0 1 2 3 4 5 6 7 8 9 10

CPU C A B C A
GPU B C A B C A
Display A B C A B
Explicit Frame Pacing

Minimum frame duration of 33ms
Explicit Frame Pacing

Minimum frame duration of 33ms

```swift
// Render Scene
...

// Get drawable and present at 30 FPS
if let drawable = view.currentDrawable {
    // Render Final Pass
    ...
    let duration = 33.0 / 1000.0  // Duration of 33 ms
    commandBuffer.present(drawable, afterMinimumDuration: duration)
}
commandBuffer.commit()
```
Explicit Frame Pacing

Minimum frame duration of 33ms

```swift
// Render Scene
...

// Get drawable and present at 30 FPS
if let drawable = view.currentDrawable {
    // Render Final Pass
    ...

    let duration = 33.0 / 1000.0 // Duration of 33 ms
    commandBuffer.present(drawable, afterMinimumDuration: duration)
}
commandBuffer.commit()
```
Thread Priorities
Thread Stalling

Render thread gets preempted due to low priority
- Priority decay
- Priority inversion
Naive Approach

Render thread gets preempted due to low priority

<table>
<thead>
<tr>
<th>VSINIC</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>GPU</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Display</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Naive Approach

Render thread gets preempted due to low priority

<table>
<thead>
<tr>
<th>VSYNC</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td>B</td>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display</td>
<td></td>
<td></td>
<td></td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Naive Approach

Render thread gets preempted due to low priority

<table>
<thead>
<tr>
<th>VSYNC</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>C</td>
<td></td>
<td>B</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td>B</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display</td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Naive Approach

Render thread gets preempted due to low priority
Naive Approach

Render thread gets preempted due to low priority
Naive Approach

Render thread gets preempted due to low priority

<table>
<thead>
<tr>
<th>VSYNC</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>C</td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td>B</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display</td>
<td>A</td>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td>A</td>
</tr>
</tbody>
</table>

CPU, GPU, Display
Demo
Best Practice

Configure the render thread

• Priority 45

• Opt out of Quality of Service
Correct Thread Priority

Priority set to 45 and no Quality of Service

<table>
<thead>
<tr>
<th>VSYNC</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPU</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Correct Thread Priority

Priority set to 45 and no Quality of Service
Correct Thread Priority

Priority set to 45 and no Quality of Service

VSYNC 0 1 2 3 4 5 6 7 8 9 10 11 12

CPU B C
GPU B C
Display A
Correct Thread Priority

Priority set to 45 and no Quality of Service
Correct Thread Priority

Priority set to 45 and no Quality of Service
Correct Thread Priority

Priority set to 45 and no Quality of Service

...  
r = pthread_attr_init(&attr);

r = pthread_attr_setschedpolicy(&attr, SCHED_RR);  // Opt out of Quality of Service
struct sched_param param = {.sched_priority = 45};  // Configure priority 45
r = pthread_attr_setschedparam(&attr, &param);  // Set priority

r = pthread_create(&posixThreadID, &attr, &PosixThreadMainRoutine, NULL);
r = pthread_attr_destroy(&attr);
...

Correct Thread Priority

Priority set to 45 and no Quality of Service

... 

r = pthread_attr_init(&attr);

r = pthread_attr_setschedpolicy(&attr, SCHED_RR);  // Opt out of Quality of Service

struct sched_param param = {.sched_priority = 45}; // Configure priority 45

r = pthread_attr_setschedparam(&attr, &param);     // Set priority

r = pthread_create(&posixThreadID, &attr, &PosixThreadMainRoutine, NULL);

r = pthread_attr_destroy(&attr);

...
Design for sustained performance
Thermal Throttling

Impact on system performance

• High device temperature
• Low power mode enabled
Best Practice

Adjust the workload to the system state

Use the following APIs

- (iOS 11.0+) `NSProcessInfo thermalState`
- (iOS 9.0+) `NSProcessInfo lowPowerModeEnabled`
- (iOS 10.3+) `MTLCommandBuffer GPUStartTime/GPUEndTime`
// Determine thermal state

switch ProcessInfo.processInfo.thermalState {
    case .fair:
        // Thermals are fair
        // Consider taking proactive measures to prevent higher thermals
    case .serious:
        // Thermals are highly elevated
        // Help the system by taking corrective action
    case .critical:
        // Thermals are extremely elevated
        // Help the system by taking immediate corrective action
    default:
        // Thermals are okay
        // Go about your business
}
// Determine thermal state

switch ProcessInfo.processInfo.thermalState {
    case .fair:
        // Thermals are fair
        // Consider taking proactive measures to prevent higher thermals
    case .serious:
        // Thermals are highly elevated
        // Help the system by taking corrective action
    case .critical:
        // Thermals are extremely elevated
        // Help the system by taking immediate corrective action
    default:
        // Thermals are okay
        // Go about your business
}

// Determine thermal state

switch ProcessInfo.processInfo.thermalState {
    case .fair:
        // Thermals are fair
        // Consider taking proactive measures to prevent higher thermals
    case .serious:
        // Thermals are highly elevated
        // Help the system by taking corrective action
    case .critical:
        // Thermals are extremely elevated
        // Help the system by taking immediate corrective action
    default:
        // Thermals are okay
        // Go about your business
}
// Determine thermal state

switch ProcessInfo.processInfo.thermalState {
    case .fair:
        // Thermals are fair
        // Consider taking proactive measures to prevent higher thermals
    case .serious:
        // Thermals are highly elevated
        // Help the system by taking corrective action
    case .critical:
        // Thermals are extremely elevated
        // Help the system by taking immediate corrective action
    default:
        // Thermals are okay
        // Go about your business
}

// Determine thermal state

switch ProcessInfo.processInfo.thermalState {
    case .fair:
        // Thermals are fair
        // Consider taking proactive measures to prevent higher thermals
    case .serious:
        // Thermals are highly elevated
        // Help the system by taking corrective action
    case .critical:
        // Thermals are extremely elevated
        // Help the system by taking immediate corrective action
    default:
        // Thermals are okay
        // Go about your business
}

Adjust the Workload

- Target sustainable framerate
- Reduce the resolution
- Simplify the shadow maps
- Use smaller textures
- Decrease the level of detail (LOD) for geometry
- Simplify post-processing and effects
Unnecessary GPU Work

Ohad Frenkel, Game Technologies
Wasted GPU Time

Waste of power and GPU budget
• Large resources
• Unused GPU work
Best Practice

Profile the GPU

- Understand the cost of every rendering feature
- Remove excessive work
Metal System Trace

Accurate timing for Vertex, Fragment, and Compute work

Ideal to measure GPU budget
Metal System Trace

Accurate timing for Vertex, Fragment, and Compute work

Ideal to measure GPU budget
Dependency Viewer
Dependency Viewer

- **Main Pass**
  - **8.50 ms** GPU Timing
  - **37 Draw Calls**
  - **465525 Vertices**

- **Color0:** RGBA16Float
  - **800x600 Dimensions**
  - **16.81 MiB Allocation Size**
  - **Clear/M...Resolve Load/Store**

- **Color0:** RGBA16Float
  - **800x600 Dimensions**
  - **4.39 MiB Allocation Size**
  - **Clear/M...Resolve Load/Store**

- **Depth:** Depth32Float
  - **800x600 Dimensions**
  - **8.80 MiB Allocation Size**
  - **Clear/M...Resolve Load/Store**

- **Depth:** Depth32Float
  - **800x600 Dimensions**
  - **2.23 MiB Allocation Size**
  - **Clear/M...Resolve Load/Store**
Dependency Viewer

- **MainPass**

- **Color0**: RGBA16Float
  - Format: RGBA16Float
  - Dimensions: 800x600
  - Allocation Size: 16.81 MiB
  - Clear/Resolve: LOAD/STORE

- **Color1**: RGBA16Float
  - Format: RGBA16Float
  - Dimensions: 800x600
  - Allocation Size: 4.39 MiB
  - Clear/Resolve: LOAD/STORE

- **Depth0**: Depth32Float
  - Format: Depth32Float
  - Dimensions: 800x600
  - Allocation Size: 8.80 MiB
  - Clear/Resolve: LOAD/STORE

- **Depth1**: Depth32Float
  - Format: Depth32Float
  - Dimensions: 800x600
  - Allocation Size: 2.23 MiB
  - Clear/Resolve: LOAD/STORE

- **GPU Timing**: 8.50 ms
- **Draw Calls**: 37
- **Vertices**: 465,525
Dependency Viewer

8.50 ms
GPU TIMING

37
DRAW CALLS

465525
VERTICES

Color0:...424a0
RGBA16Float
FORMAT
800x600
DIMENSIONS
16.81 MiB
ALLOCATION SIZE
Clear/M...Resolve
LOAD/STORE

Color0:...426b0
RGBA16Float
FORMAT
800x600
DIMENSIONS
4.39 MiB
ALLOCATION SIZE
Clear/M...Resolve
LOAD/STORE

Depth:...428c0
Depth32Float
FORMAT
800x600
DIMENSIONS
8.80 MiB
ALLOCATION SIZE
Clear/M...Resolve
LOAD/STORE

Depth:...42ad0
Depth32Float
FORMAT
800x600
DIMENSIONS
2.23 MiB
ALLOCATION SIZE
Clear/M...Resolve
LOAD/STORE
Dependency Viewer

- **8.50 ms**
  - GPU TIMING
- **37**
  - DRAW CALLS
- **465525**
  - VERTICES

- **RGBA16Float**
  - FORMAT
  - **800x600**
    - DIMENSIONS
  - **16.81 MiB**
    - ALLOCATION SIZE
  - Clear/M...Resolve
    - LOAD/STORE

- **RGBA16Float**
  - FORMAT
  - **800x600**
    - DIMENSIONS
  - **4.39 MiB**
    - ALLOCATION SIZE
  - Clear/M...Resolve
    - LOAD/STORE

- **Depth32Float**
  - FORMAT
  - **800x600**
    - DIMENSIONS
  - **8.80 MiB**
    - ALLOCATION SIZE
  - Clear/M...Resolve
    - LOAD/STORE

- **Depth32Float**
  - FORMAT
  - **800x600**
    - DIMENSIONS
  - **2.23 MiB**
    - ALLOCATION SIZE
  - Clear/M...Resolve
    - LOAD/STORE
Demo
Finding Hidden Complexity

Shadow map
Main pass
HUD
Composite pass
Cascaded shadow maps (3 passes)
SSAO (5 passes)
Main pass
HDR
Post-process (5 passes)
Composite pass
And more...
Profile!
**Take-Away**

Profile early and often

Target a consistent frame rate

Set the correct thread priorities

Adapt to system load and thermals

Don’t submit unnecessary work to the GPU
## More Information

https://developer.apple.com/wwdc18/612

<table>
<thead>
<tr>
<th>Event</th>
<th>Location</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metal Shader Debugging and Profiling</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Metal Debugging and Profiling Lab</td>
<td>Technology Lab 5</td>
<td>Fri 12:00 PM</td>
</tr>
<tr>
<td>Metal for Game Developers</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

WWDC 2018