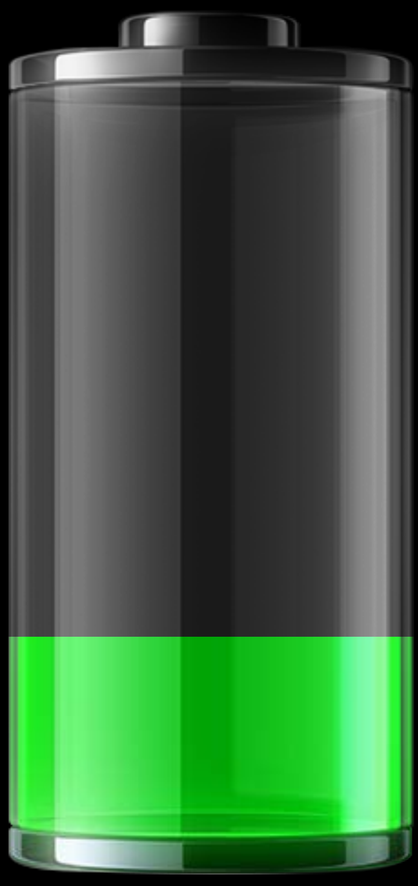
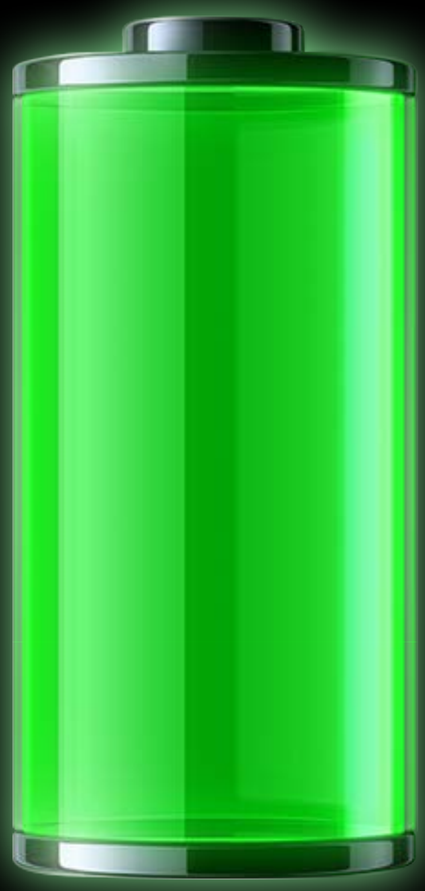
Energy Best Practices

Session 712

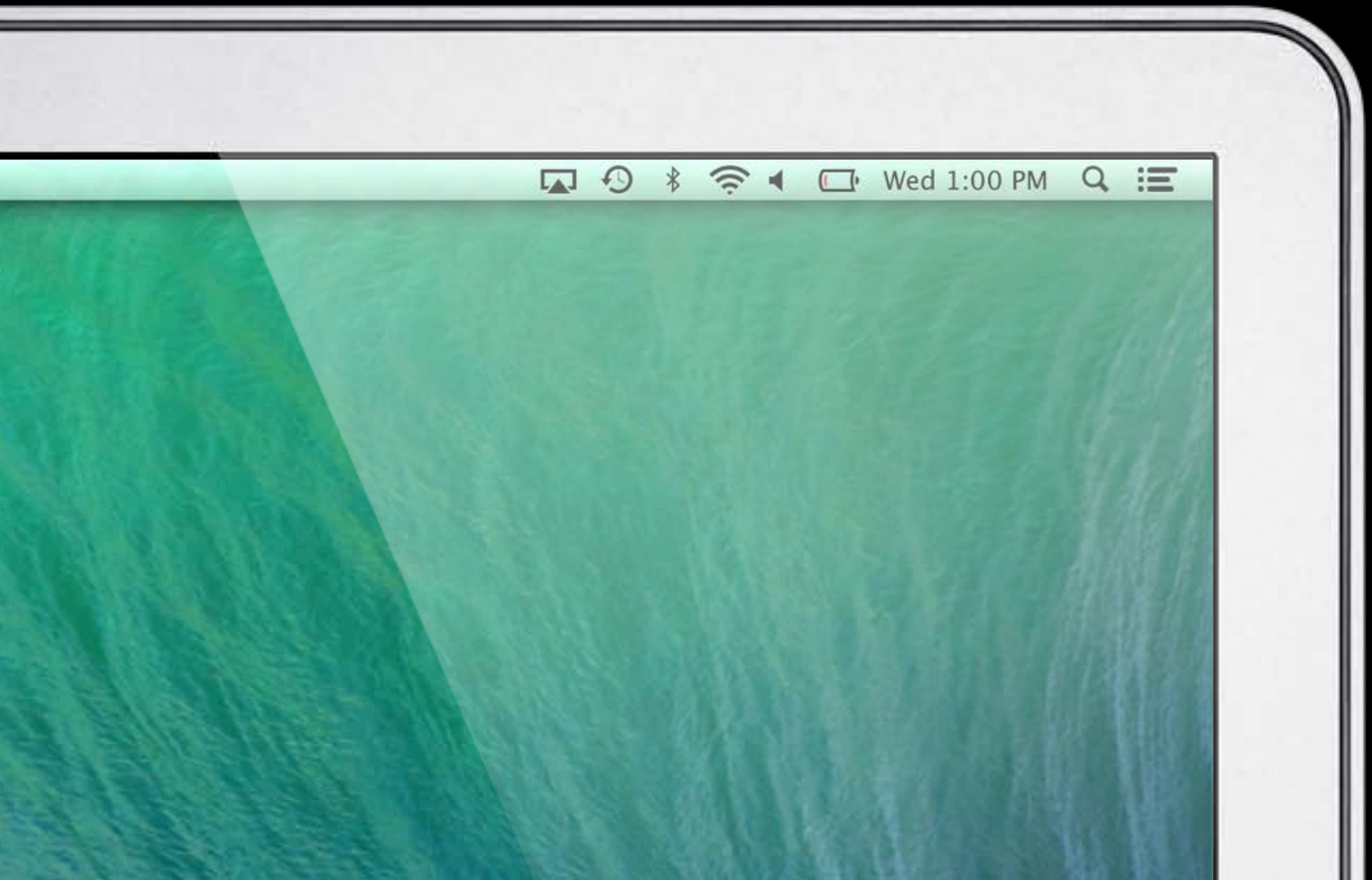
Matt Jacobson
OS X Performance Mercenary

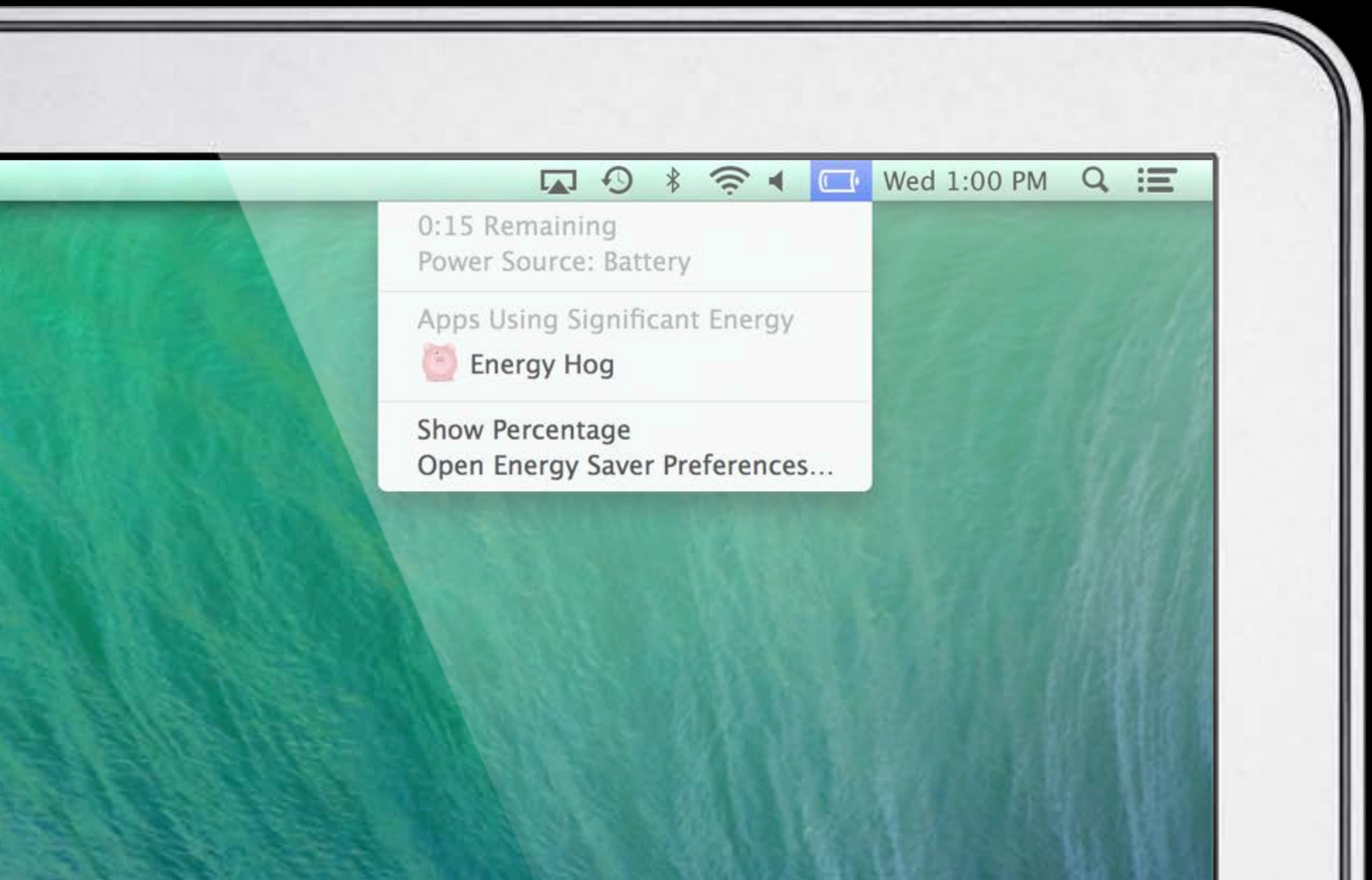
These are confidential sessions—please refrain from streaming, blogging, or taking pictures

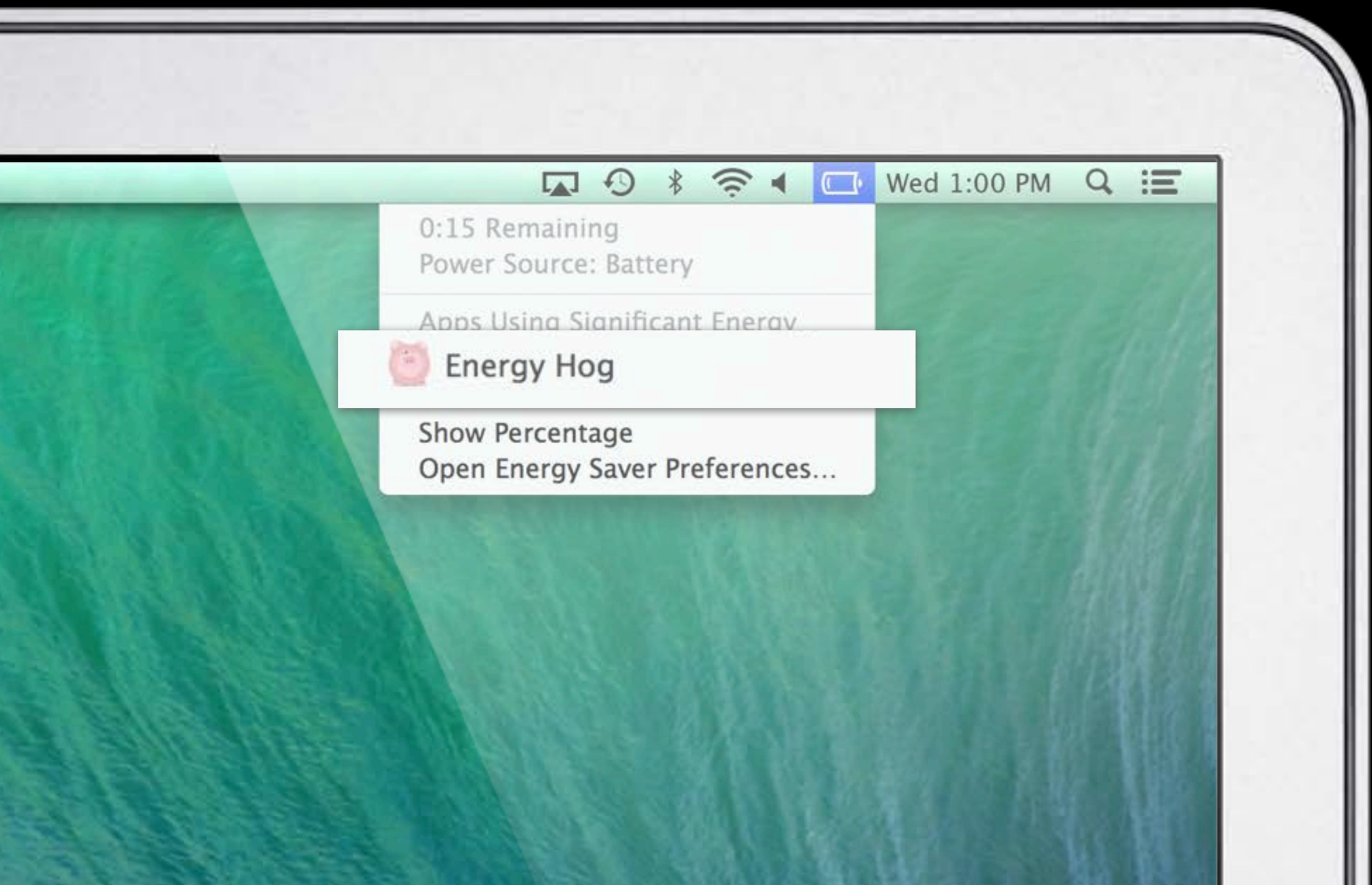


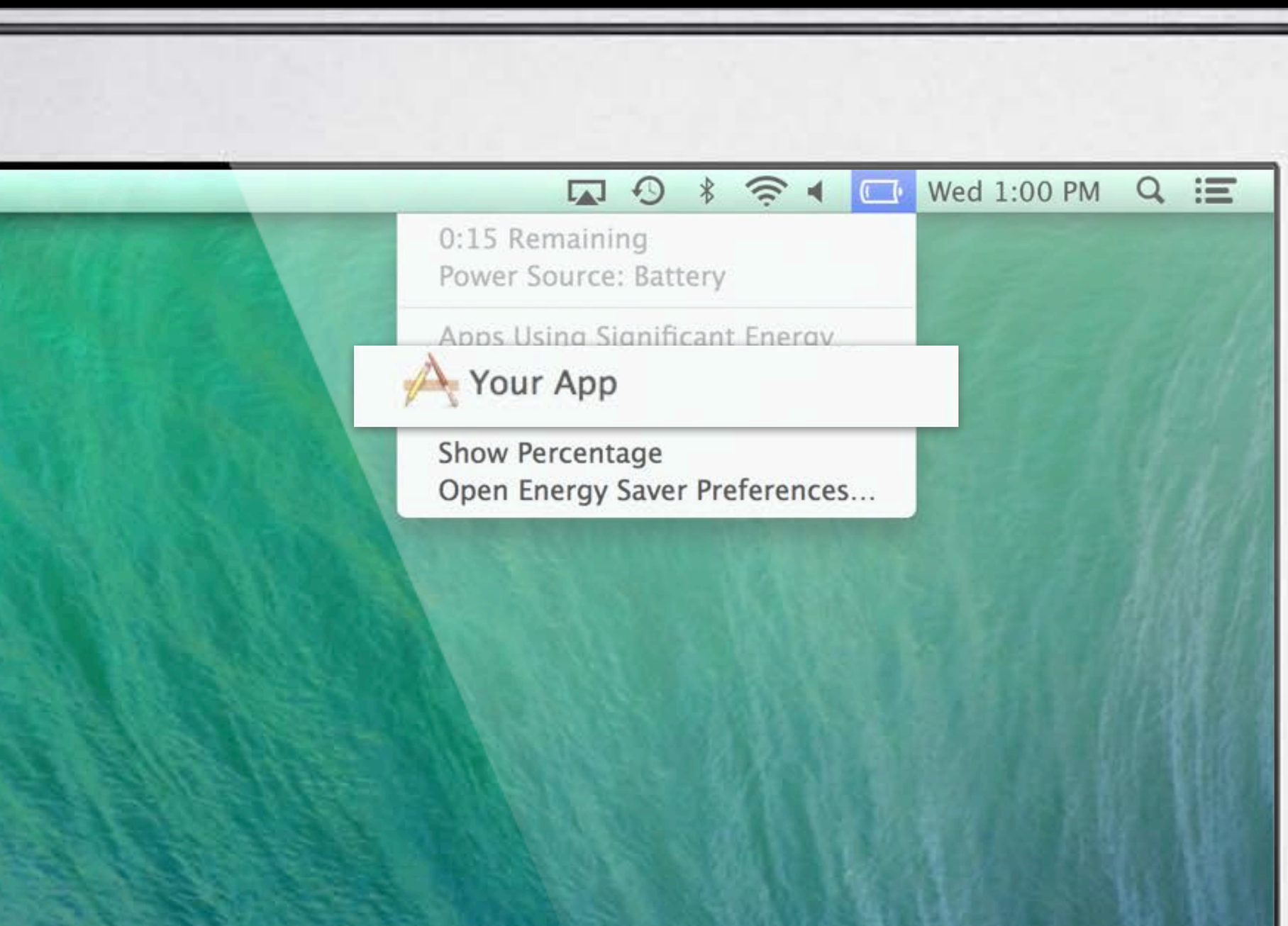



Finder File Edit View Go Window Help
Wed 1:00 PM
MacBook Air


Wed 1:00 PM


Wed 1:00 PM
0:15 Remaining
Power Source: Battery
Apps Using Significant Energy
Energy Hog
Show Percentage
Open Energy Saver Preferences...


Wed 1:00 PM
0:15 Remaining
Power Source: Battery
Apps Using Significant Energy
Energy Hog
Show Percentage
Open Energy Saver Preferences...


Wed 1:00 PM
0:15 Remaining
Power Source: Battery
Apps Using Significant Energy...
Your App
Show Percentage
Open Energy Saver Preferences...

Topics

Topics

- Recognizing energy issues

Topics

- Recognizing energy issues
- Diagnosing issues

Topics

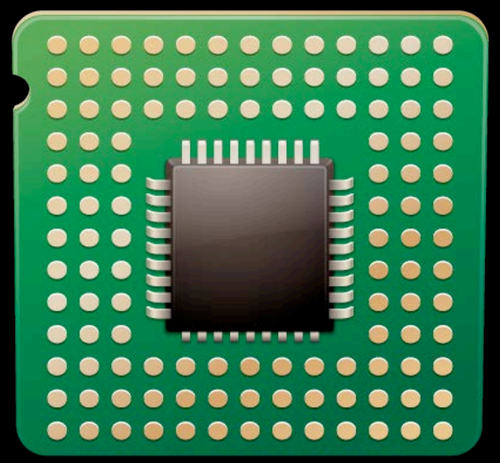
- Recognizing energy issues
- Diagnosing issues
- Avoiding common pitfalls

Topics

- Recognizing energy issues
- Diagnosing issues
- Avoiding common pitfalls
- Adopting best practices and new APIs

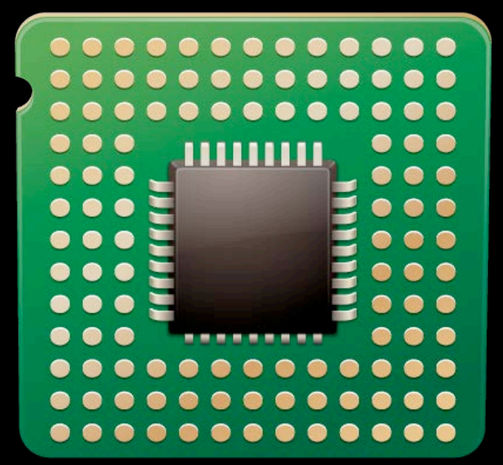
What Uses Energy?

What Uses Energy?



CPU

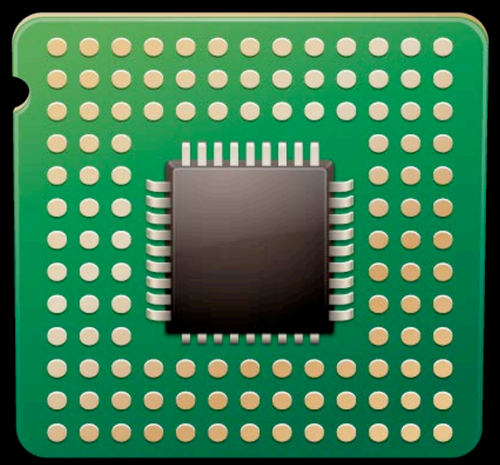
What Uses Energy?



CPU

I/O

What Uses Energy?

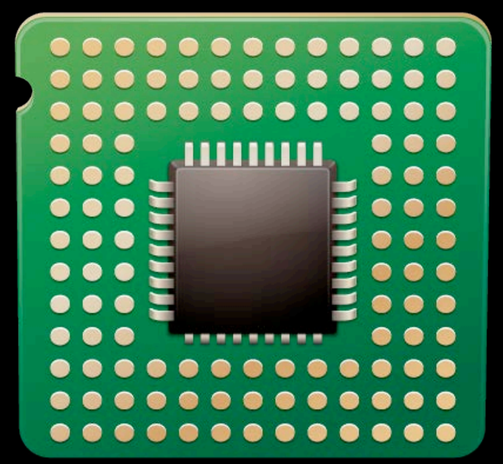


CPU

I/O

Graphics

What Uses Energy?



CPU

I/O

Graphics

CPU Usage

How to Use the CPU

How to Use the CPU

- Modern CPUs have many cores, fast clocks, etc.

How to Use the CPU

- Modern CPUs have many cores, fast clocks, etc.
- Still limited by the size of the battery

How to Use the CPU

- Modern CPUs have many cores, fast clocks, etc.
- Still limited by the size of the battery
- If you ask to run, you'll run

How to Use the CPU

- Modern CPUs have many cores, fast clocks, etc.
- Still limited by the size of the battery
- If you ask to run, you'll run
- Single app can impact battery life

CPU Power Draw

Vs. idle

CPU Power Draw

Vs. idle

1% CPU	10% higher power draw

CPU Power Draw

Vs. idle

1% CPU	10% higher power draw
10% CPU	2x power draw

CPU Power Draw

Vs. idle

1% CPU	10% higher power draw
10% CPU	2x power draw
100% CPU	10x power draw

How Much Is Too Much?

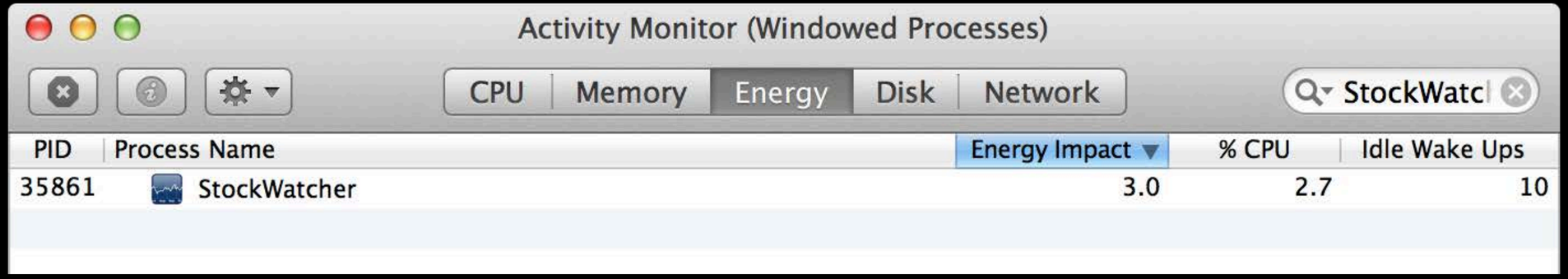
- Be absolutely idle when app is not in use
  - Eliminate nonuser-driven work
  - Wait quietly for user input

How Much Is Too Much?

- Be absolutely idle when app is not in use
  - Eliminate nonuser-driven work
  - Wait quietly for user input
- Be efficient when user requests action
  - Performance = power
  - Efficient, multithreaded algorithms win (use GCD)
  - Then race back to idle

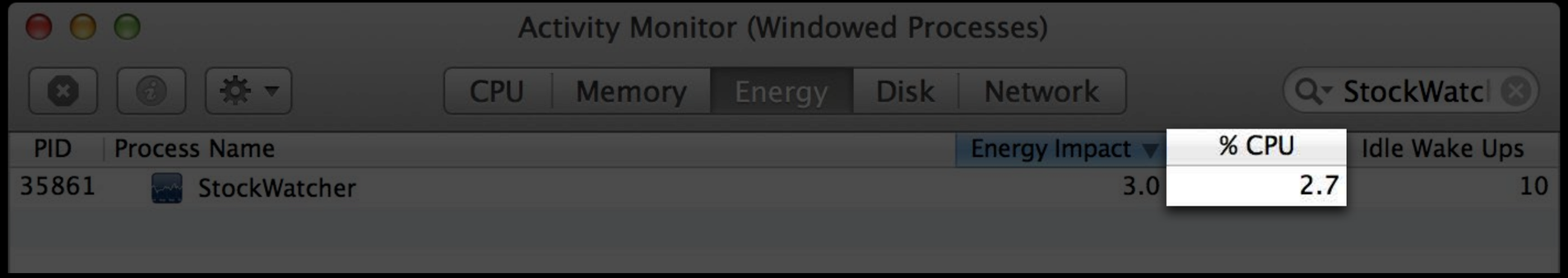
Recognizing High CPU Usage

Activity Monitor



Recognizing High CPU Usage

Activity Monitor



StockWatcher.xcodeproj

StockWatcher 〉 My Mac 64-bit

Running StockWatcher.app : StockWatcher — No Issues

CPU Report

StockWatcher.app
PID 2180, Running

CPU 3%

Memory 4.2 MB

Energy Impact High

CPU

Profile In Instruments

Percentage Utilized

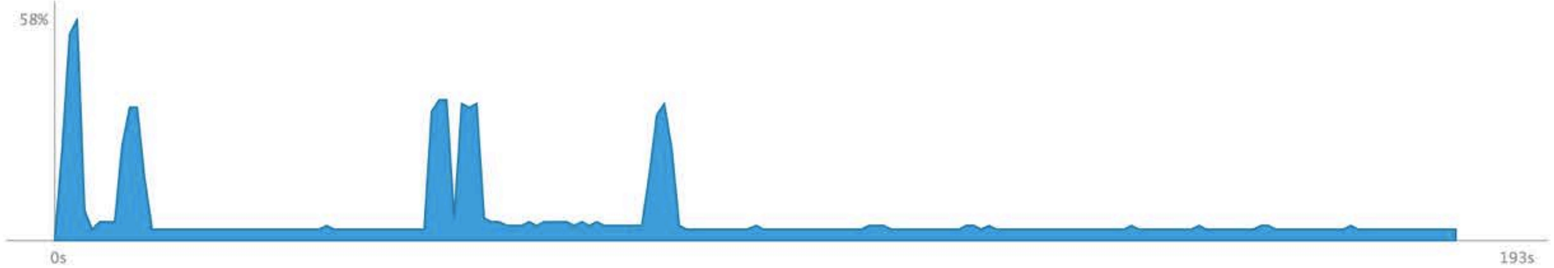
3%

Utilization Comparison

- **StockWatcher.app** 3%
- **Other Processes** 20%
- **Free** 377%

Utilization Over Time

Duration: 3 min 6 sec
High: 58%
Low: 3%



Threads

Thread 1
Unnamed

Thread 3
Unnamed

Thread 7
Unnamed

StockWatcher.app

StockWatcher.xcodeproj

StockWatcher 〉 My Mac 64-bit

Running StockWatcher.app : StockWatcher — No Issues

CPU Report

StockWatcher.app
PID 2180, Running

CPU	3%
Memory	4.2 MB
Energy Impact	High

CPU

Profile In Instruments

Percentage Utilized

3%

Utilization Comparison

- **StockWatcher.app** 3%
- **Other Processes** 20%
- **Free** 377%

Utilization Over Time

Duration: 3 min 6 sec
High: 58%
Low: 3%

Threads

Thread 1 Unnamed	
Thread 3 Unnamed	
Thread 7 Unnamed	

StockWatcher.app

StockWatcher.xcodeproj

StockWatcher 〉 My Mac 64-bit

Running StockWatcher.app : StockWatcher — No Issues

CPU Report

StockWatcher.app
PID 2180, Running

CPU 3%

Memory 4.2 MB

Energy Impact High

CPU

Profile In Instruments

Percentage Utilized

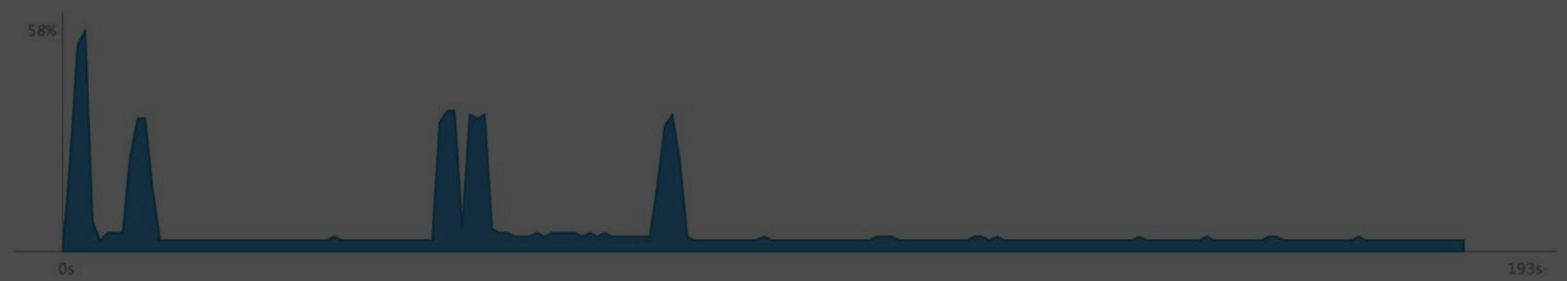
3%

Utilization Comparison

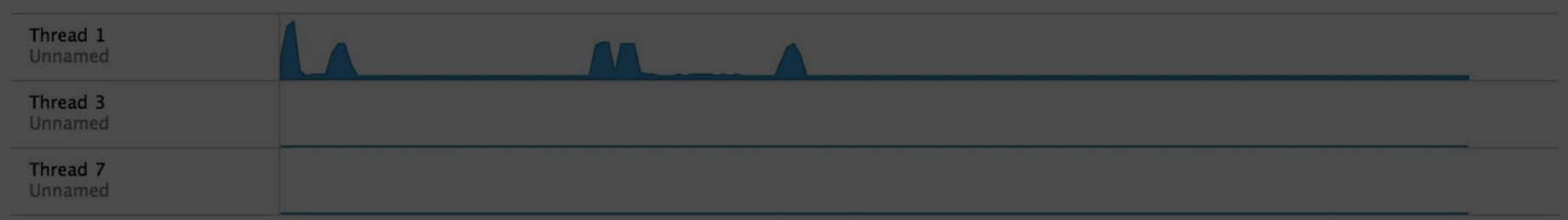
- **StockWatcher.app** 3%
- **Other Processes** 20%
- **Free** 377%

Utilization Over Time

Duration: 3 min 6 sec
High: 58%
Low: 3%



Threads



StockWatcher.app

StockWatcher.xcodeproj

StockWatcher 〉 My Mac 64-bit

Running StockWatcher.app : StockWatcher

No Issues

CPU Report

StockWatcher.app
PID 2180, Running

CPU 3%

Memory 4.2 MB

Energy Impact High

Profile In Instruments

CPU

Percentage Utilized

3%

Utilization Comparison

StockWatcher.app
3%

Other Processes
20%

Free
377%

Utilization Over Time

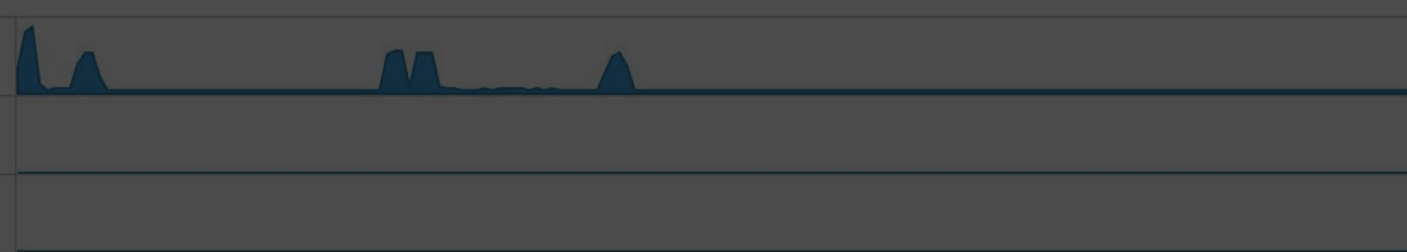
Duration: 3 min 6 sec
High: 58%
Low: 3%

Threads

Thread 1
Unnamed

Thread 3
Unnamed

Thread 7
Unnamed



StockWatcher.app

StockWatcher.xcodeproj

StockWatcher 〉 My Mac 64-bit

Running StockWatcher.app : StockWatcher — No Issues

CPU Report

StockWatcher.app
PID 2180, Running

CPU	3%
Memory	4.2 MB
Energy Impact	High

CPU

Profile In Instruments

Percentage Utilized

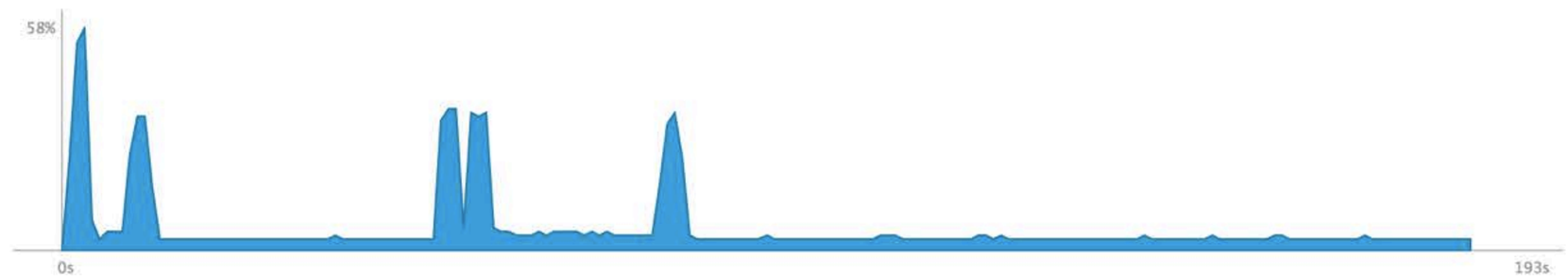
3%

Utilization Comparison

- **StockWatcher.app** 3%
- **Other Processes** 20%
- **Free** 377%

Utilization Over Time

Duration: 3 min 6 sec
High: 58%
Low: 3%



Threads

Thread 1 Unnamed	
Thread 3 Unnamed	
Thread 7 Unnamed	

StockWatcher.app

StockWatcher.xcodeproj

StockWatcher 〉 My Mac 64-bit

Running StockWatcher.app : StockWatcher — No Issues

CPU Report

StockWatcher.app
PID 2180, Running

CPU 3%

Memory 4.2 MB

Energy Impact High

CPU

Profile In Instruments

Percentage Utilized

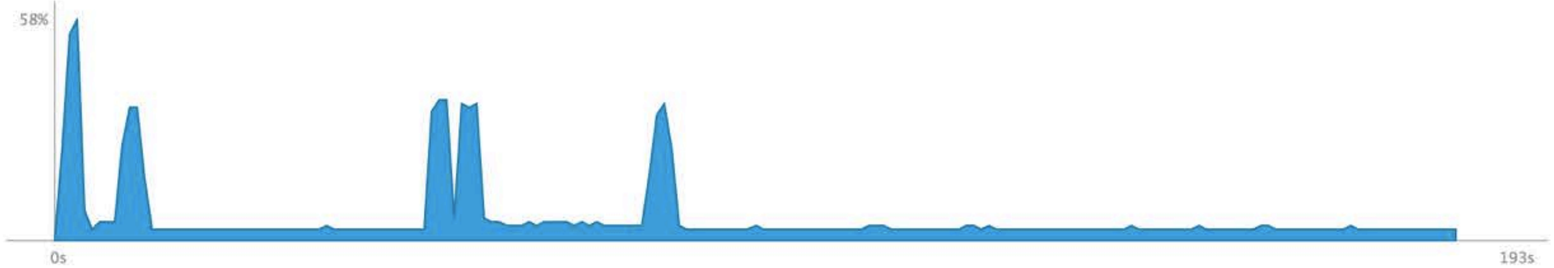
3%

Utilization Comparison

- **StockWatcher.app** 3%
- **Other Processes** 20%
- **Free** 377%

Utilization Over Time

Duration: 3 min 6 sec
High: 58%
Low: 3%



Threads

Thread 1
Unnamed

Thread 3
Unnamed

Thread 7
Unnamed

StockWatcher.app

Recognizing High CPU Usage

top(1)

```
$ top -pid <pid>
PID     COMMAND       %CPU   TIME      #TH   #WQ   #PORT #MREG MEM
23106   StockWatcher   2.2   00:13.55  7/1   4/1   132   168   6340K
```

Recognizing High CPU Usage

top(1)

```
$ top -pid <pid>
PID     COMMAND       %CPU   TIME      #TH   #WQ   #PORT #MREG MEM
23106   StockWatcher   2.2   00:13.55  7/1   4/1   132   168   6340K
```

Recognizing High CPU Usage

top(1)

```
$ top -pid <pid> -a
PID     COMMAND       %CPU   TIME      #TH   #WQ   #PORT #MREG MEM
23106   StockWatcher   2.2   00:00.20  7/1   4/1   132   168   6340K
```

Recognizing High CPU Usage

top(1)

```
$ top -pid <pid> -a
PID     COMMAND       %CPU   TIME       #TH   #WQ   #PORT #MREG MEM
23106   StockWatcher   2.2   00:00.20   7/1   4/1   132   168   6340K
```

Instruments

Record | Target: StockWatcher (1... | Inspection Range | 00:00:10 Run 1 of 1 | View | Library | Search: Involves Symbol

All Cores | All Processes / Threads

Instruments | 00:00 00:10 00:20 00:30 00:40 00:50 01:00 01:10

Time Profiler

Time Profiler | Call Tree | Call Tree

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
- Top Functions

Call Tree Constraints

Specific Data Mining

Running Time		Self	Symbol Name
191.0ms	100.0%	0.0	▼Main Thread 0x4456f
187.0ms	97.9%	0.0	▼start libdyld.dylib
187.0ms	97.9%	0.0	▼main StockWatcher
187.0ms	97.9%	0.0	▼NSApplicationMain AppKit
187.0ms	97.9%	0.0	▼-[NSApplication run] AppKit
187.0ms	97.9%	0.0	▼-[NSApplication nextEventMatchingMask:untilDate:inMode:dequeue:] AppKit
187.0ms	97.9%	0.0	▼_DPSNextEvent AppKit
187.0ms	97.9%	0.0	▼_BlockUntilNextEventMatchingListInModeWithFilter HIToolbox
187.0ms	97.9%	0.0	▼ReceiveNextEventCommon HIToolbox
187.0ms	97.9%	0.0	▼RunCurrentEventLoopInMode HIToolbox
187.0ms	97.9%	0.0	▼CFRunLoopRunSpecific CoreFoundation
187.0ms	97.9%	0.0	▼__CFRunLoopRun CoreFoundation
178.0ms	93.1%	0.0	▼__CFRunLoopDoObservers CoreFoundation
178.0ms	93.1%	0.0	▼__CFRUNLOOP_IS_CALLING_OUT_TO_AN_OBSERVER_CALLBACK_FUNCTION__ CoreFo
178.0ms	93.1%	0.0	▼__83-[NSWindow _postWindowNeedsDisplayOrLayoutOrUpdateConstraintsUnlessPos
178.0ms	93.1%	0.0	▼_handleWindowNeedsDisplayOrLayoutOrUpdateConstraints AppKit
178.0ms	93.1%	1.0	▼-[NSView displayIfNeeded] AppKit
177.0ms	92.6%	0.0	▼-[NSView _displayRectIgnoringOpacity:isVisibleRect:rectIsVisibleRectForView:]
167.0ms	87.4%	0.0	▼-[NSThemeFrame _recursiveDisplayRectIfNeededIgnoringOpacity:isVisibleRec
167.0ms	87.4%	0.0	▼-[NSView _recursiveDisplayRectIfNeededIgnoringOpacity:isVisibleRect:rectIs
163.0ms	85.3%	0.0	▼-[NSView _recursiveDisplayRectIfNeededIgnoringOpacity:isVisibleRect:rect

Instruments

Record | Target: StockWatcher (1... | Inspection Range | 00:00:10 Run 1 of 1 | View | Library | Search: Involves Symbol

All Cores | All Processes / Threads

Instruments

00:00 00:10 00:20 00:30 00:40 00:50 01:00 01:10

Time Profiler

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
  - Top Functions
- Call Tree Constraints
- Specific Data Mining

Call Tree > Call Tree

Running Time		Self	Symbol Name
191.0ms	100.0%	0.0	▼Main Thread 0x4456f
187.0ms	97.9%	0.0	▼start libdyld.dylib
187.0ms	97.9%	0.0	▼main StockWatcher
187.0ms	97.9%	0.0	▼NSApplicationMain AppKit
187.0ms	97.9%	0.0	▼-[NSApplication run] AppKit
187.0ms	97.9%	0.0	▼-[NSApplication nextEventMatchingMask:untilDate:inMode:dequeue:] AppKit
187.0ms	97.9%	0.0	▼_DPSNextEvent AppKit
187.0ms	97.9%	0.0	▼_BlockUntilNextEventMatchingListInModeWithFilter HIToolbox
187.0ms	97.9%	0.0	▼ReceiveNextEventCommon HIToolbox
187.0ms	97.9%	0.0	▼RunCurrentEventLoopInMode HIToolbox
187.0ms	97.9%	0.0	▼CFRunLoopRunSpecific CoreFoundation
187.0ms	97.9%	0.0	▼__CFRunLoopRun CoreFoundation
178.0ms	93.1%	0.0	▼__CFRunLoopDoObservers CoreFoundation
178.0ms	93.1%	0.0	▼__CFRUNLOOP_IS_CALLING_OUT_TO_AN_OBSERVER_CALLBACK_FUNCTION__ CoreFo
178.0ms	93.1%	0.0	▼__83-[NSWindow _postWindowNeedsDisplayOrLayoutOrUpdateConstraintsUnlessPos
178.0ms	93.1%	0.0	▼_handleWindowNeedsDisplayOrLayoutOrUpdateConstraints AppKit
178.0ms	93.1%	1.0	▼-[NSView displayIfNeeded] AppKit
177.0ms	92.6%	0.0	▼-[NSView _displayRectIgnoringOpacity:isVisibleRect:rectIsVisibleRectForView:]
167.0ms	87.4%	0.0	▼-[NSThemeFrame _recursiveDisplayRectIfNeededIgnoringOpacity:isVisibleRec
167.0ms	87.4%	0.0	▼-[NSView _recursiveDisplayRectIfNeededIgnoringOpacity:isVisibleRect:rectIs
163.0ms	85.3%	0.0	▼-[NSView _recursiveDisplayRectIfNeededIgnoringOpacity:isVisibleRect:rect

Instruments

Record | Target: StockWatcher (1... | Inspection Range | 00:00:10 Run 1 of 1 | View | Library | Search: Involves Symbol

All Cores | All Processes / Threads

Instruments | 00:00 00:10 00:20 00:30 00:40 00:50 01:00 01:10

Time Profiler

Time Profiler | Call Tree | Call Tree

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
- Top Functions

Call Tree Constraints

Specific Data Mining

Running Time		Self	Symbol Name
191.0ms	100.0%	0.0	▼Main Thread 0x4456f
187.0ms	97.9%	0.0	▼start libdyld.dylib
187.0ms	97.9%	0.0	▼main StockWatcher
187.0ms	97.9%	0.0	▼NSApplicationMain AppKit
187.0ms	97.9%	0.0	▼-[NSApplication run] AppKit
187.0ms	97.9%	0.0	▼-[NSApplication nextEventMatchingMask:untilDate:inMode:dequeue:] AppKit
187.0ms	97.9%	0.0	▼_DPSNextEvent AppKit
187.0ms	97.9%	0.0	▼_BlockUntilNextEventMatchingListInModeWithFilter HIToolbox
187.0ms	97.9%	0.0	▼ReceiveNextEventCommon HIToolbox
187.0ms	97.9%	0.0	▼RunCurrentEventLoopInMode HIToolbox
187.0ms	97.9%	0.0	▼CFRunLoopRunSpecific CoreFoundation
187.0ms	97.9%	0.0	▼__CFRunLoopRun CoreFoundation
178.0ms	93.1%	0.0	▼__CFRunLoopDoObservers CoreFoundation
178.0ms	93.1%	0.0	▼__CFRUNLOOP_IS_CALLING_OUT_TO_AN_OBSERVER_CALLBACK_FUNCTION__ CoreFo
178.0ms	93.1%	0.0	▼__83-[NSWindow _postWindowNeedsDisplayOrLayoutOrUpdateConstraintsUnlessPos
178.0ms	93.1%	0.0	▼_handleWindowNeedsDisplayOrLayoutOrUpdateConstraints AppKit
178.0ms	93.1%	1.0	▼-[NSView displayIfNeeded] AppKit
177.0ms	92.6%	0.0	▼-[NSView _displayRectIgnoringOpacity:isVisibleRect:rectIsVisibleRectForView:]
167.0ms	87.4%	0.0	▼-[NSThemeFrame _recursiveDisplayRectIfNeededIgnoringOpacity:isVisibleRec
167.0ms	87.4%	0.0	▼-[NSView _recursiveDisplayRectIfNeededIgnoringOpacity:isVisibleRect:rectIs
163.0ms	85.3%	0.0	▼-[NSView _recursiveDisplayRectIfNeededIgnoringOpacity:isVisibleRect:rect

Diagnosing High CPU Usage

Resource exceptions

Diagnosing High CPU Usage

Resource exceptions

```
kernel[0] <Debug>: process SomeApp[202] thread 1989 caught burning CPU! It
used more than 50% CPU (Actual recent usage: 97%) over 180 seconds ...
```

Diagnosing High CPU Usage

Resource exceptions

```
kernel[0] <Debug>: process SomeApp[202] thread 1989 caught burning CPU! It
used more than 50% CPU (Actual recent usage: 97%) over 180 seconds ...
```

```
/Library/Logs/DiagnosticReports/SomeApp_2013-06-13.cpu_resource.spin
```

Diagnosing High CPU Usage

Resource exceptions

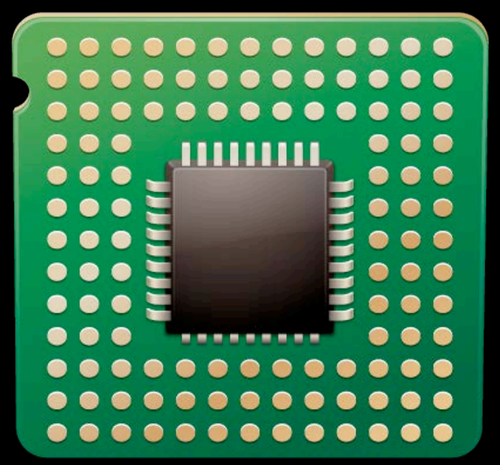
```
kernel[0] <Debug>: process SomeApp[202] thread 1989 caught burning CPU! It
used more than 50% CPU (Actual recent usage: 97%) over 180 seconds ...

/Library/Logs/DiagnosticReports/SomeApp_2013-06-13.cpu_resource.spin
Process:         SomeApp [202]
Architecture:    x86_64
Parent:          launchd [137]

Powerstats for:  SomeApp
Microstackshots: 64 samples
  64 thread_start + 13 (libsystem_pthread.dylib) [0x7fff922b2fe9]
    64 _pthread_start + 131 (libsystem_pthread.dylib) [0x7fff922ae7a0]
      64 _pthread_body + 138 (libsystem_pthread.dylib) [0x7fff922ae90f]
        64 run_thread + 17 (SomeApp) [0x10996bf11]
          64 do_stuff + 9 (SomeApp) [0x10996bef9]
```

Diagnosing High CPU Usage

Secondary indicators



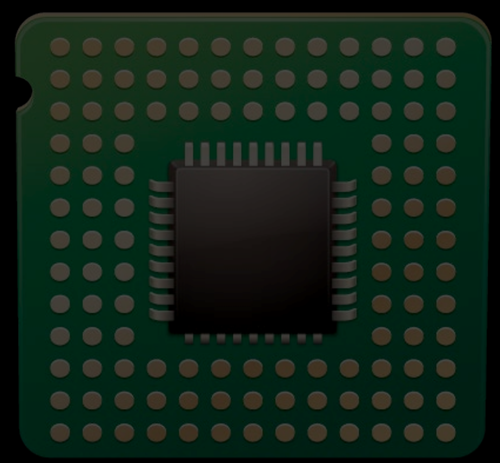
CPU

I/O

Graphics

Diagnosing High CPU Usage

Secondary indicators



CPU

I/O

Graphics

Diagnosing High CPU Usage

fs_usage

```
$ sudo fs_usage MyApp
```

Diagnosing High CPU Usage

fs_usage

```
$ sudo fs_usage MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:27  open            ../Downloads.plist        0.000086   MyApp
20:29:27  write                                     0.000017   MyApp
20:29:27  close                                     0.000048   MyApp
20:29:27  stat64          MyIcon.png                0.000002   MyApp
20:29:27  stat64          MyIcon.png                0.000002   MyApp
20:29:27  stat64          MyIcon.png                0.000002   MyApp
20:29:27  stat64          MyIcon.png                0.000002   MyApp
```

Diagnosing High CPU Usage

fs_usage

```
$ sudo fs_usage MyApp
20:29:26  stat64          MyIcon.png                  0.000002   MyApp
20:29:26  stat64          MyIcon.png                  0.000002   MyApp
20:29:26  stat64          MyIcon.png                  0.000002   MyApp
20:29:26  stat64          MyIcon.png                  0.000002   MyApp
20:29:26  stat64          MyIcon.png                  0.000002   MyApp
20:29:27  open            ../Downloads.plist          0.000086   MyApp
20:29:27  write                                       0.000017   MyApp
20:29:27  close                                       0.000048   MyApp
20:29:27  stat64          MyIcon.png                  0.000002   MyApp
20:29:27  stat64          MyIcon.png                  0.000002   MyApp
20:29:27  stat64          MyIcon.png                  0.000002   MyApp
20:29:27  stat64          MyIcon.png                  0.000002   MyApp
```

Diagnosing High CPU Usage

fs_usage

```
$ sudo fs_usage MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:27  open            ../Downloads.plist        0.000086   MyApp
20:29:27  write                                     0.000017   MyApp
20:29:27  close                                     0.000048   MyApp
20:29:27  stat64          MyIcon.png                0.000002   MyApp
20:29:27  stat64          MyIcon.png                0.000002   MyApp
20:29:27  stat64          MyIcon.png                0.000002   MyApp
20:29:27  stat64          MyIcon.png                0.000002   MyApp
```

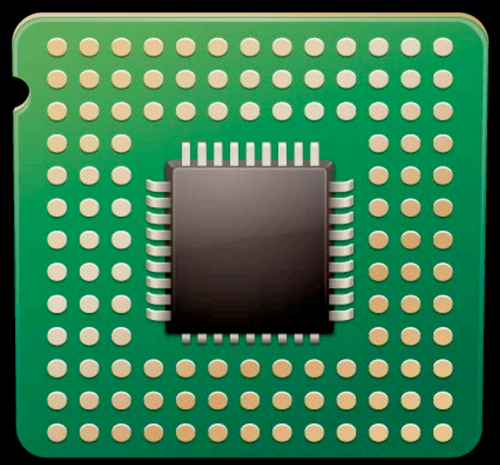
Diagnosing High CPU Usage

fs_usage

```
$ sudo fs_usage MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:26  stat64          MyIcon.png                0.000002   MyApp
20:29:27  open            ../Downloads.plist        0.000086   MyApp
20:29:27  write                                     0.000017   MyApp
20:29:27  close                                     0.000048   MyApp
20:29:27  stat64          MyIcon.png                0.000002   MyApp
20:29:27  stat64          MyIcon.png                0.000002   MyApp
20:29:27  stat64          MyIcon.png                0.000002   MyApp
20:29:27  stat64          MyIcon.png                0.000002   MyApp
```

Diagnosing High CPU Usage

Secondary indicators



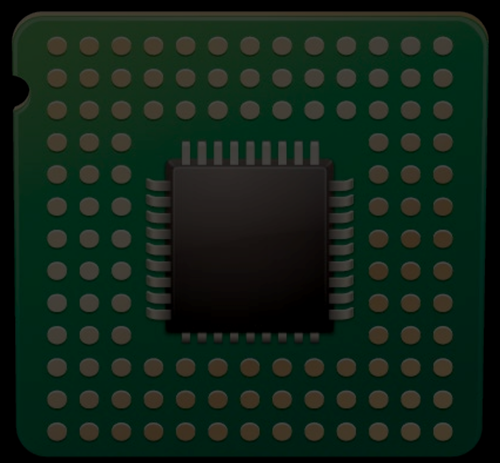
CPU

I/O

Graphics

Diagnosing High CPU Usage

Secondary indicators



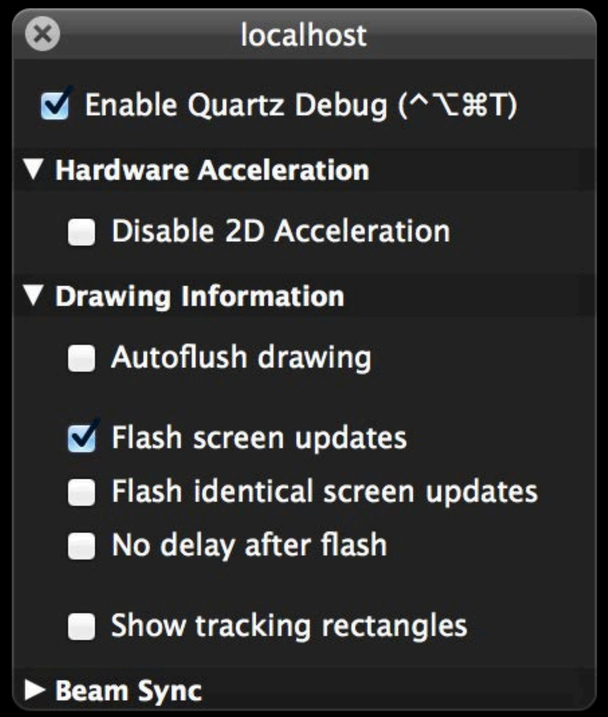
CPU

I/O

Graphics

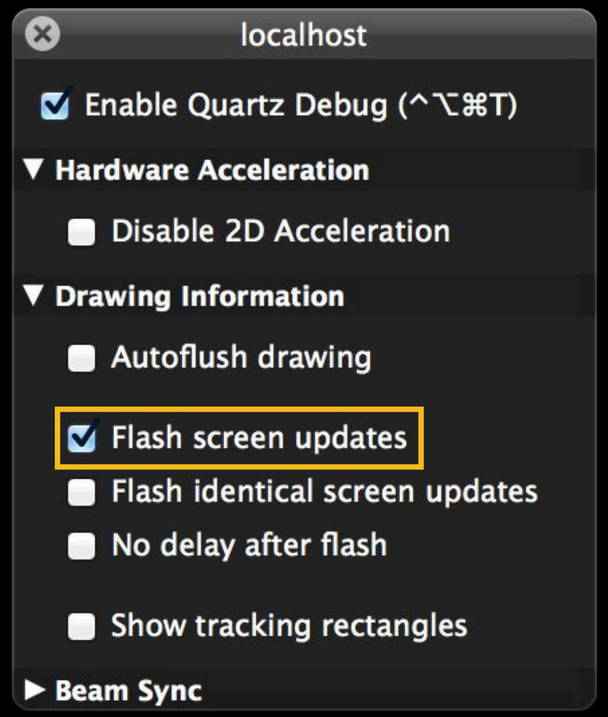
Diagnosing High CPU Usage

Quartz Debug



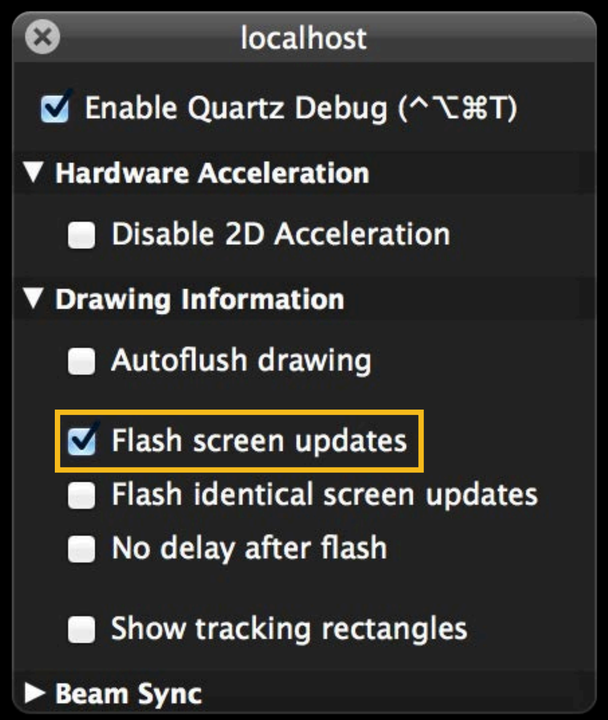
Diagnosing High CPU Usage

Quartz Debug



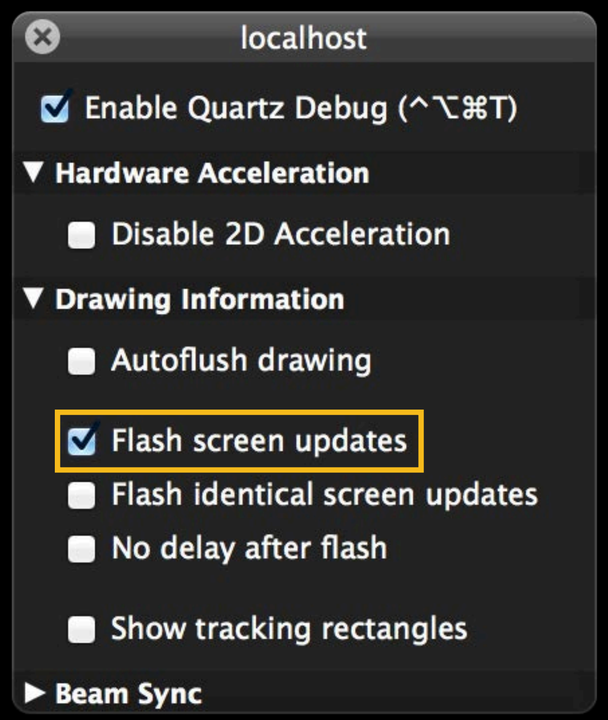
Diagnosing High CPU Usage

Quartz Debug



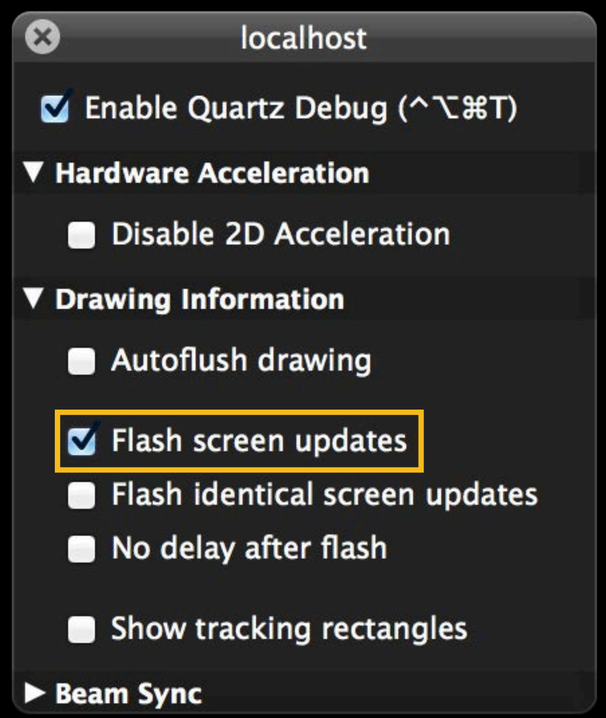
Diagnosing High CPU Usage

Quartz Debug



Diagnosing High CPU Usage

Quartz Debug

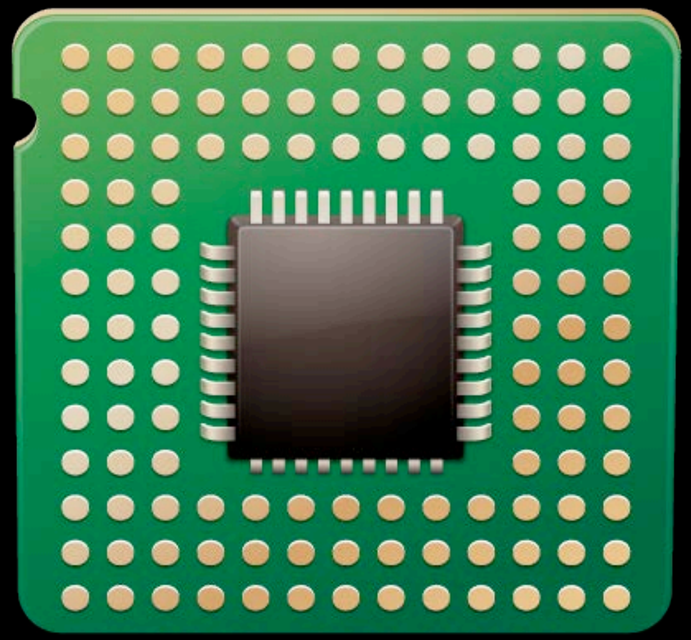


Graphics Tools for Xcode

http://developer.apple.com

CPU Usage

- A little is a lot
- Ensure absolute idle
- Watch with Activity Monitor
  - Especially when idle in the background
- Diagnose with Time Profiler



Timers

Timers

Timers

- Used to
  - Schedule future/periodic events
  - Drive animations

Timers

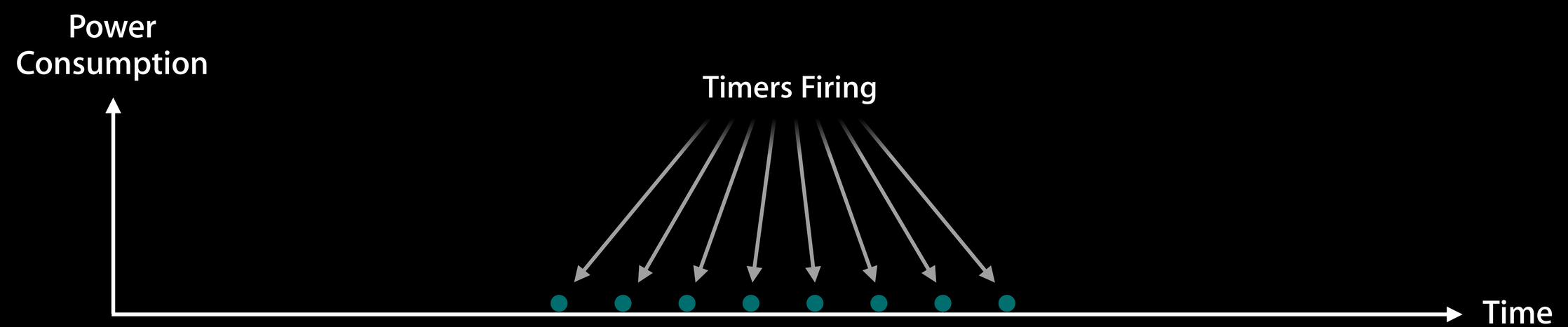
- Used to
  - Schedule future/periodic events
  - Drive animations
- Keep apps out of idle

Timers

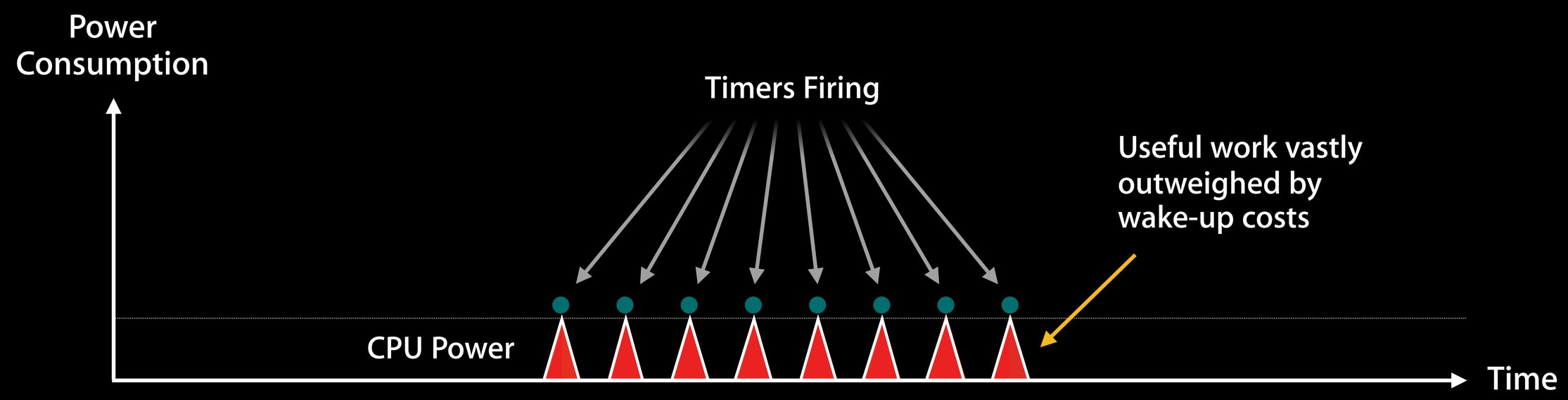
- Used to
  - Schedule future/periodic events
  - Drive animations
- Keep apps out of idle
- Can have outsize energy effect

Timers

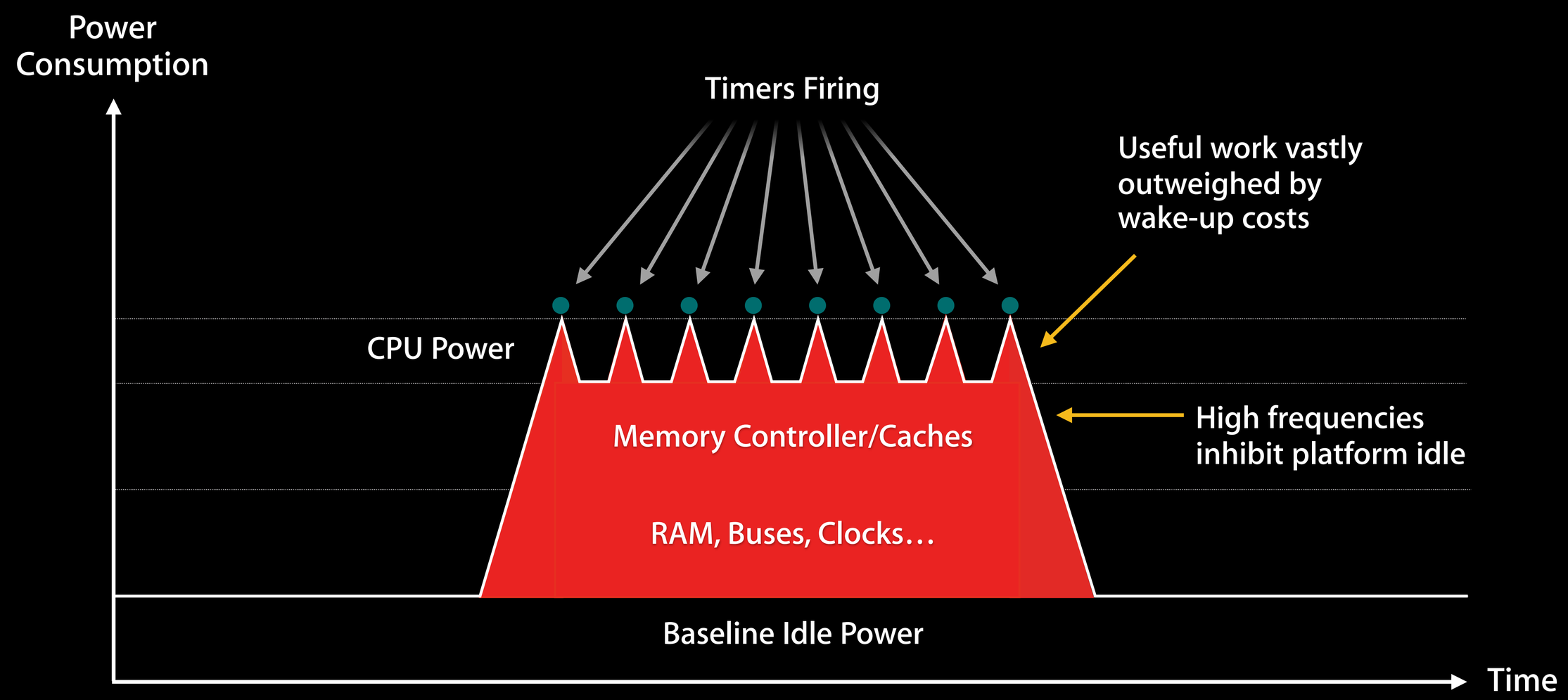
Timers



Timers


Power Consumption
Timers Firing
Useful work vastly outweighed by wake-up costs
CPU Power
Time

Timers


Power Consumption
Timers Firing
Useful work vastly outweighed by wake-up costs
CPU Power
Memory Controller/Caches
High frequencies inhibit platform idle
RAM, Buses, Clocks…
Baseline Idle Power
Time

What Is a Timer?

- Timer APIs
  - Grand Central Dispatch timers
  - CFRunLoopTimer
  - NSTimer
  - CFRunLoopRunInMode with timeout

What Is a Timer?

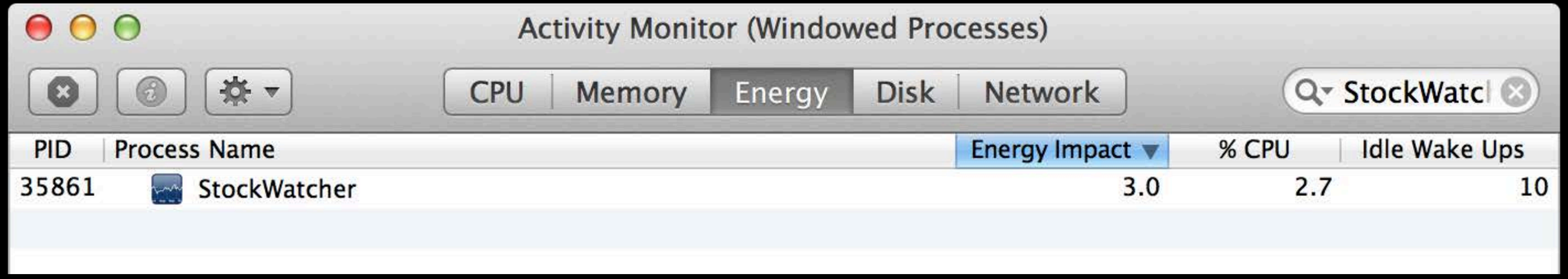
- Low-level APIs
  - sleep(), usleep(), nanosleep()
- APIs with timeouts
  - pthread_cond_timedwait()
  - dispatch_semaphore_wait()
  - select(), poll(), kevent(), mach_msg()

What Is a Timer?

- High-level APIs
  - dispatch_after()
  - -[NSObject performSelector:withObject:afterDelay:]
  - NSProgressIndicator
  - CVDisplayLink (OS X)/CADisplayLink (iOS)
  - And more…

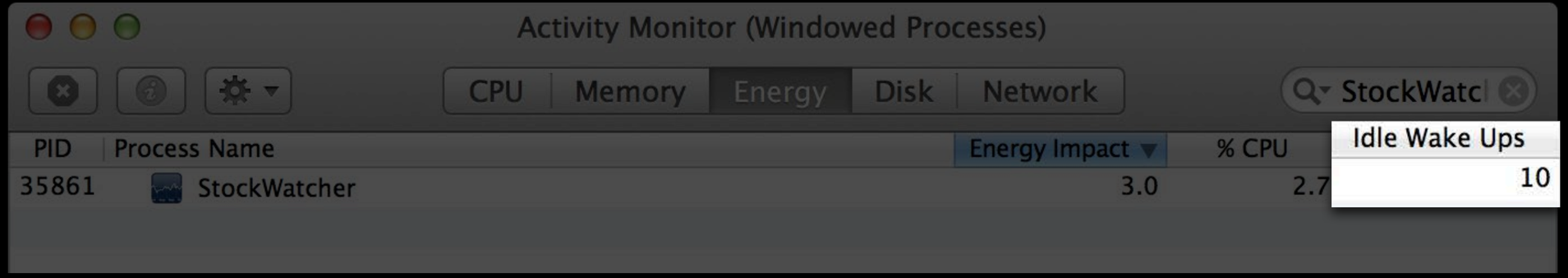
Recognizing Timer Issues

Activity Monitor



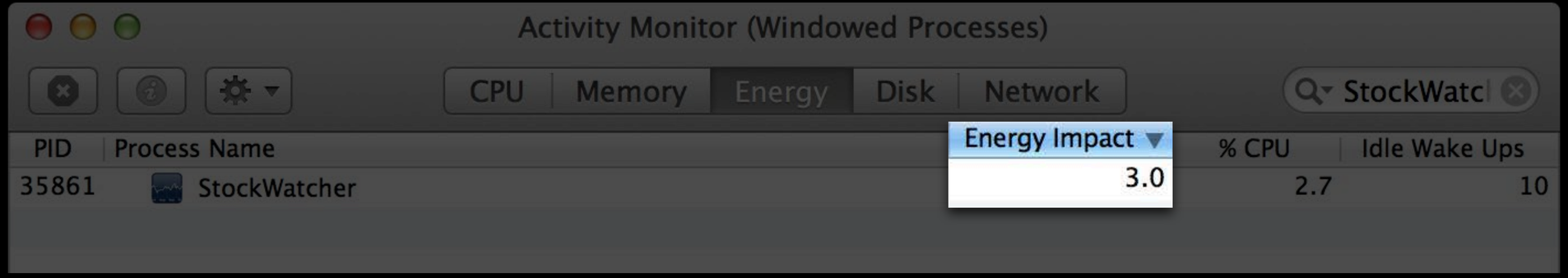
Recognizing Timer Issues

Activity Monitor



Recognizing Timer Issues

Activity Monitor



StockWatcher.xcodeproj

StockWatcher 〉 My Mac 64-bit

Running StockWatcher.app : StockWatcher — No Issues

Energy Report

StockWatcher.app
PID 2180, Running

CPU	3%
Memory	4.2 MB
Energy Impact	High

Energy

Utilization

High

App Nap

10 Wakes Last Second

20 Avg Per Second

Energy Impact

Very High

Utilization — CPU Wake Overhead

App Nap

To allow inactive applications to remain running while minimizing their impact on battery life, the system may lower execution priority and rate limit timers.

Track App Nap

Idle Prevention

Maximizing CPU idle time is an important part of energy efficiency. Timers can prevent CPU idle and should be avoided when other alternatives are available.

Find Timers

Find Polling

StockWatcher.app

StockWatcher.xcodeproj

StockWatcher 〉 My Mac 64-bit

Running StockWatcher.app : StockWatcher — No Issues

Energy Report

StockWatcher.app
PID 2180, Running

CPU	3%
Memory	4.2 MB
Energy Impact	High

Energy

Utilization

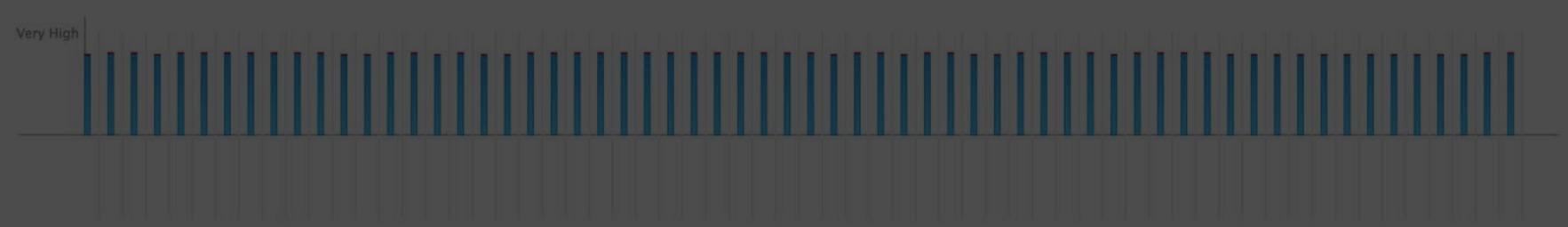
High

App Nap

10 Wakes Last Second

20 Avg Per Second

Energy Impact



App Nap

To allow inactive applications to remain running while minimizing their impact on battery life, the system may lower execution priority and rate limit timers.

Track App Nap

Idle Prevention

Maximizing CPU idle time is an important part of energy efficiency. Timers can prevent CPU idle and should be avoided when other alternatives are available.

Find Timers

Find Polling

StockWatcher.app

StockWatcher.xcodeproj

StockWatcher 〉 My Mac 64-bit

Running StockWatcher.app : StockWatcher — No Issues

Energy Report

StockWatcher.app
PID 2180, Running

CPU	3%
Memory	4.2 MB
Energy Impact	High

Energy

Utilization

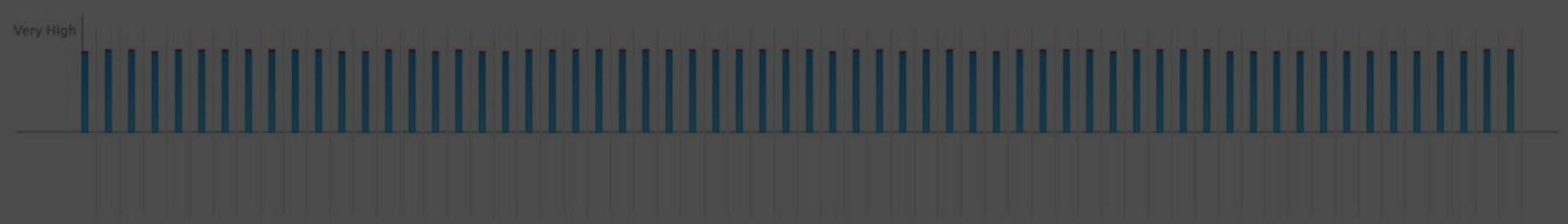
High

App Nap

10 Wakes Last Second

20 Avg Per Second

Energy Impact



App Nap

To allow inactive applications to remain running while minimizing their impact on battery life, the system may lower execution priority and rate limit timers.

Track App Nap

Idle Prevention

Maximizing CPU idle time is an important part of energy efficiency. Timers can prevent CPU idle and should be avoided when other alternatives are available.

Find Timers

Find Polling

StockWatcher.app

StockWatcher.xcodeproj

StockWatcher 〉 My Mac 64-bit | Running StockWatcher.app : StockWatcher | No Issues

Energy Report

StockWatcher.app
PID 2180, Running

CPU 3%

Memory 4.2 MB

Energy Impact High

Energy

Utilization

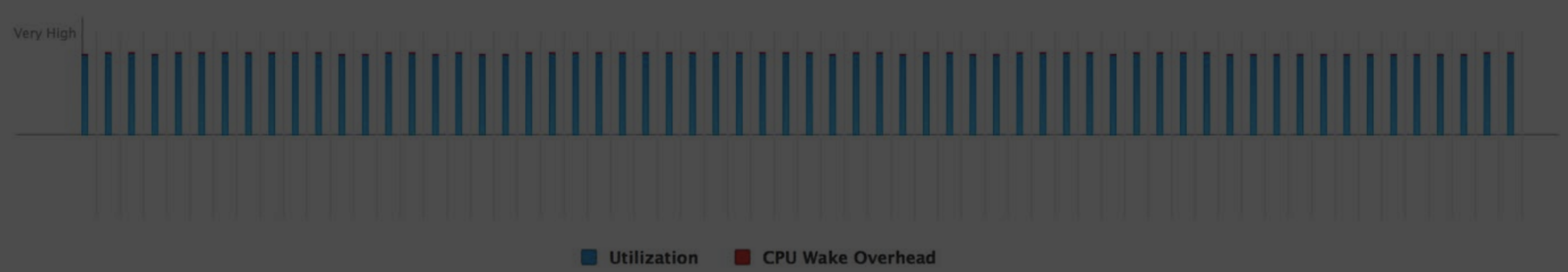
High

App Nap

10 Wakes Last Second

20 Avg Per Second

Energy Impact



App Nap

To allow inactive applications to remain running while minimizing their impact on battery life, the system may lower execution priority and rate limit timers.

Track App Nap

Idle Prevention

Maximizing CPU idle time is an important part of energy efficiency. Timers can prevent CPU idle and should be avoided when other alternatives are available.

Find Timers

Find Polling

StockWatcher.app

StockWatcher.xcodeproj

StockWatcher 〉 My Mac 64-bit

Running StockWatcher.app : StockWatcher — No Issues

Energy Report

StockWatcher.app
PID 2180, Running

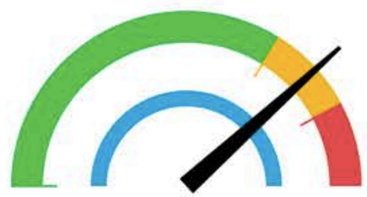
CPU 3%

Memory 4.2 MB

Energy Impact High

Energy

Utilization



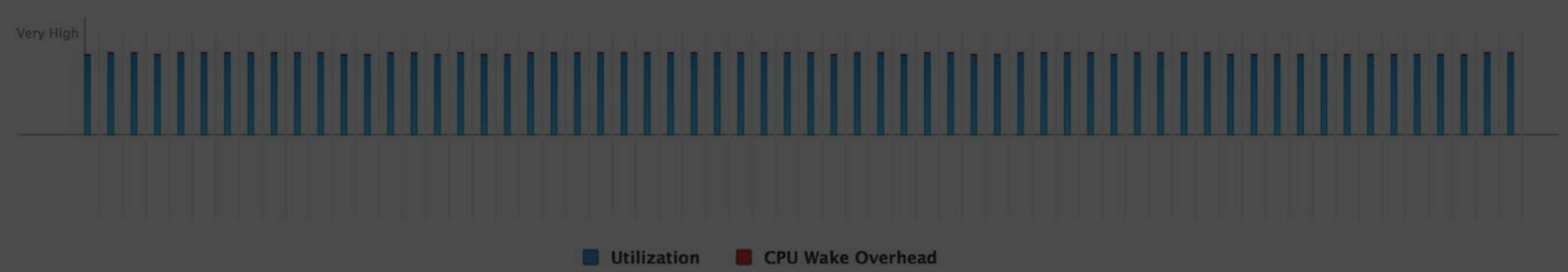
High

App Nap

10 Wakes Last Second

20 Avg Per Second

Energy Impact



App Nap

To allow inactive applications to remain running while minimizing their impact on battery life, the system may lower execution priority and rate limit timers.

Track App Nap

Idle Prevention

Maximizing CPU idle time is an important part of energy efficiency. Timers can prevent CPU idle and should be avoided when other alternatives are available.

Find Timers

Find Polling

StockWatcher.app

StockWatcher.xcodeproj

StockWatcher 〉 My Mac 64-bit

Running StockWatcher.app : StockWatcher — No Issues

Energy Report

StockWatcher.app
PID 2180, Running

CPU	3%
Memory	4.2 MB
Energy Impact	High

Energy

Utilization

High

App Nap

10 Wakes Last Second

20 Avg Per Second

Energy Impact

Very High

Utilization — CPU Wake Overhead

App Nap

To allow inactive applications to remain running while minimizing their impact on battery life, the system may lower execution priority and rate limit timers.

Track App Nap

Idle Prevention

Maximizing CPU idle time is an important part of energy efficiency. Timers can prevent CPU idle and should be avoided when other alternatives are available.

Find Timers

Find Polling

StockWatcher.app

Diagnosing Timer Issues

timerfires

```
$ sudo timerfires -p <pid> -s
```

Diagnosing Timer Issues

timerfires

```
$ sudo timerfires -p <pid> -s
TIME(ms)   PID  PROCESS  TYPE      TIMER ROUTINE
     435  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
     933  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    1055  1603  MyApp    dispatch  MyApp`-[AppModel updateWidgets:]
    1435  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    1935  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    2055  1603  MyApp    dispatch  MyApp`-[AppModel updateWidgets:]
    2435  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    3004  1603  MyApp    sleep
              libsystem_kernel.dylib`__semwait_signal+0xa
              libsystem_c.dylib`usleep+0x36
              MyApp`-[AppModel pollForChange]+0x1a
              libdispatch.dylib`_dispatch_call_block_and_release+0xc
```

Diagnosing Timer Issues

timerfires

```
$ sudo timerfires -p <pid> -s
TIME(ms)   PID  PROCESS  TYPE      TIMER ROUTINE
     435  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
     933  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    1055  1603  MyApp    dispatch  MyApp`-[AppModel updateWidgets:]
    1435  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    1935  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    2055  1603  MyApp    dispatch  MyApp`-[AppModel updateWidgets:]
    2435  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    3004  1603  MyApp    sleep
              libsystem_kernel.dylib`__semwait_signal+0xa
              libsystem_c.dylib`usleep+0x36
              MyApp`-[AppModel pollForChange]+0x1a
              libdispatch.dylib`_dispatch_call_block_and_release+0xc
```

Diagnosing Timer Issues

timerfires

```
$ sudo timerfires -p <pid> -s
TIME(ms)   PID  PROCESS  TYPE      TIMER ROUTINE
     435  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
     933  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    1055  1603  MyApp    dispatch  MyApp`-[AppModel updateWidgets:]
    1435  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    1935  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    2055  1603  MyApp    dispatch  MyApp`-[AppModel updateWidgets:]
    2435  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    3004  1603  MyApp    sleep
              libsystem_kernel.dylib`__semwait_signal+0xa
              libsystem_c.dylib`usleep+0x36
              MyApp`-[AppModel pollForChange]+0x1a
              libdispatch.dylib`_dispatch_call_block_and_release+0xc
```

Diagnosing Timer Issues

timerfires

```
$ sudo timerfires -p <pid> -s
TIME(ms)   PID  PROCESS  TYPE      TIMER ROUTINE
     435  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
     933  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    1055  1603  MyApp    dispatch  MyApp`-[AppModel updateWidgets:]
    1435  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    1935  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    2055  1603  MyApp    dispatch  MyApp`-[AppModel updateWidgets:]
    2435  1603  MyApp    CF        MyApp`-[AppDelegate timerFired:]
    3004  1603  MyApp    sleep
              libsystem_kernel.dylib`__semwait_signal+0xa
              libsystem_c.dylib`usleep+0x36
              MyApp`-[AppModel pollForChange]+0x1a
              libdispatch.dylib`_dispatch_call_block_and_release+0xc
```

Don’t Use Timers for… Synchronization

Don’t Use Timers for… Synchronization

```
BOOL workIsDone = NO;

/* thread one */
void doWork(void) {
   /* wait for network ... */
   workIsDone = YES;
}
```

Don’t Use Timers for… Synchronization

```
BOOL workIsDone = NO;

/* thread one */
void doWork(void) {
   /* wait for network ... */
   workIsDone = YES;
}

/* thread two */
void waitForWorkToFinish(void) {
   while (!workIsDone) {
      usleep(100000); /* 100 ms */
   }
   [WorkController workDidFinish];
}
```

Use Built-in Synchronization Instead

- Serial dispatch queues
- Dispatch semaphores
- Pthread condition variables

Use Built-in Synchronization Instead

```
BOOL workIsDone = NO;

/* thread one */
void doWork(void) {
   /* wait for network ... */
   workIsDone = YES;
}

/* thread two */
void waitForWorkToFinish(void) {
   while (!workIsDone) {
      usleep(100000); /* 100 ms */
   }
   [WorkController workDidFinish];
}
```

Use Built-in Synchronization Instead

```
   my_queue = dispatch_queue_create("com.myapp.myq", DISPATCH_QUEUE_SERIAL);

/* thread one */
void doWork(void) {
   dispatch_sync(my_queue, ^{
      /* wait for network ... */
   });
}

/* thread two */
void waitForWorkToFinish(void) {
   dispatch_sync(my_queue, ^{
      [WorkController workDidFinish];
   });
}
```

Don’t Use Timers for… Polling

```
NSTimer *myTimer = [[NSTimer alloc] initWithFireDate:date
                                            interval:0.5
                                              target:self
                                            selector:@selector(checkForFile:)
                                            userInfo:nil
                                             repeats:YES];

[[NSRunLoop currentRunLoop] addTimer:myTimer forMode:NSDefaultRunLoopMode];
```

Use Event Streams Instead

- File system updates
  - DISPATCH_SOURCE_TYPE_VNODE for small-scale updates
  - FSEvents framework (OS X) for filesystem-wide updates
- Inter-process notifications
  - XPC
  - NSDistributedNotificationCenter
  - notify(3)

Use Event Streams Instead

- Disk Arbitration notifications
- I/O Kit matching notifications
- Network events
  - Apple Push Notification Service
  - Bonjour
  - Network reachability

Use Event Streams Instead

```
NSTimer *myTimer = [[NSTimer alloc] initWithFireDate:date
                                            interval:0.5
                                              target:self
                                            selector:@selector(checkForFile:)
                                            userInfo:nil
                                             repeats:YES];

[[NSRunLoop currentRunLoop] addTimer:myTimer forMode:NSDefaultRunLoopMode];
```

Use Event Streams Instead

```
const uint64_t flags = DISPATCH_VNODE_DELETE | DISPATCH_VNODE_WRITE;
dispatch_source_t my_source =
        dispatch_source_create(DISPATCH_SOURCE_TYPE_VNODE, fd, flags, queue);

dispatch_source_set_event_handler(my_source, ^{
    [self checkForFile];
});

dispatch_resume(my_source);
```

Specify Suitable Timeouts

Specify Suitable Timeouts

```
for (;;) {
      long rv;
      dispatch_time_t timeout =
                  dispatch_time(DISPATCH_TIME_NOW, 500 * NSEC_PER_MSEC);

      rv = dispatch_semaphore_wait(my_sema, timeout);

      if (havePendingWork) {
            [self doPendingWork];
      }
}
```

Specify Suitable Timeouts

```
for (;;) {
      long rv;
      dispatch_time_t timeout = DISPATCH_TIME_FOREVER;


      rv = dispatch_semaphore_wait(my_sema, timeout);

      if (havePendingWork) {
            [self doPendingWork];
      }
}
```

Manage Repeating Timers

```
NSTimer *myTimer = [[NSTimer alloc] initWithFireDate:date
                                            interval:1.0
                                              target:self
                                            selector:@selector(timerFired:)
                                            userInfo:nil
                                             repeats:YES];

/* ... */

/* myTimer not invalidated! */
```

Manage Repeating Timers

```
NSTimer *myTimer = [[NSTimer alloc] initWithFireDate:date
                                            interval:1.0
                                              target:self
                                            selector:@selector(timerFired:)
                                            userInfo:nil
                                             repeats:YES];

/* ... */

[myTimer invalidate];
```

Specify Timer Tolerance

- Hint to the system of timer flexibility
- Allows system to batch work

Specify Timer Tolerance

- For dispatch timers

```
dispatch_source_t my_timer =
            dispatch_source_create(DISPATCH_SOURCE_TYPE_TIMER, 0, 0, queue);
dispatch_source_set_timer(my_timer, DISPATCH_TIME_NOW,
                          1 * NSEC_PER_SEC, NSEC_PER_SEC / 10);

dispatch_source_set_event_handler(my_timer, ^{
    [self timerFired];
});
dispatch_resume(my_timer);
```

Specify Timer Tolerance

- For CFRunLoopTimers

```
CFRunLoopTimerRef myTimer = CFRunLoopTimerCreate(kCFAllocatorDefault,
                                                 fireDate,
                                                 1.0,
                                                 0,
                                                 0,
                                                 &timerFired,
                                                 NULL);
CFRunLoopTimerSetTolerance(myTimer, 0.1);

CFRunLoopAddTimer(CFRunLoopGetCurrent(), myTimer, kCFRunLoopDefaultMode);
```

Specify Timer Tolerance

- For NSTimers

```
NSTimer *myTimer = [[NSTimer alloc] initWithFireDate:date
                                                interval:1.0
                                                  target:self
                                                selector:@selector(timerFired:)
                                                userInfo:nil
                                                 repeats:YES];
[myTimer setTolerance:0.1];

[[NSRunLoop currentRunLoop] addTimer:myTimer forMode:NSDefaultRunLoopMode];
```

Demo

Putting it all together

Responsible Timer Use

- Be mindful of wakeup overhead
- Monitor for wakeups in Activity Monitor
- Debug with timerfires
- Specify timer tolerance

Smart Background Work

Smart Background Work

- Be idle when you can, eliminating…
  - extraneous CPU usage
  - unnecessary timers
- Be smart when you can't

Periodic Animations

- Stop driving UI the user isn't watching
- Apple apps work like this
  - Photo Booth
  - FaceTime
  - App Store

Frontmost App Notification

- App became frontmost/non-frontmost
  - Implement `-applicationDidResignActive:` and `-applicationDidBecomeActive:` in your app delegate
  - Or check

```
if ([NSApp isActive]) ...
```


StockWatcher
S&P 500 1630.74 23.67
APSC 3.16 3.15
DUFF 39.99 17.18
ZCFK 109.66 19.84
ACME 123.00 -1.00


Untitled
View Outline Pages Text Box Shapes Table Charts Comment Inspector Media Colors Fonts
S E Q U O I A C L U B
Redwoods
Aliquam sit amet tortor ac purus porttitor porta. Aenean sodales felis sed mauris. Sed ipsum erat, porttitor non, venenatis id, vestibulum non, sem.
A Quarterly Newsletter
Issue N° 17 — Fall 2009
Donec odio risus
Cras eget lectus. Quisque facilisis mattis eros. Vivamus felis augue, malesuada nec, congue nec, semper a, risus. Mauris quam. Morbi eget nunc nec eros bibendum.
Page 2
Maecenas faucibus
Sed ipsum erat, dign porttitor non, venenatis id, vestibulum. Amet eget massa vitae libero.
Page 2
Cras condimentum consectetuer orci
Sed ipsum erat, porttitor non, venenatis id, vestibulum. Phasellus eget massa vitae libero imperdiet lorem elementum.
Page 2
Nullam pellentesque
Suspendisse vestibulum placerat odio. Phasellus eget massa vitae lorem libero non.
Page 2
Nullam hendrerit
Nullam auctor enim quis nibh. Maecenas fermentum. Morbi placerat dign. Praesent fringilla sollicitudin neque. Fusce ipsum eleifend dolor.
Page 3
Morbi vitae ligula
Vivamus venenatisvelit sed nulla. Curabitur sodales ornare urna. Vivamus ultrices. Maecenas eget id a libero gravid.
Page 4
Pellentes que habitan
Lorem ipsum dolor sit amet, consectetuer adiPascing elit. Pellentesque ipsum tur Pas, ullamcorper at, molestie facilisis mattis eros. Vivamus felis augue, malesuada nec, congue nec, semper a, risus. Mauris quam.
Pellentesque nibh nisl, euismod eget, imperdiet ut, ultrices eu, ipsum. Pellentesque elementum dignissnon mauris. Maecenas ac enim. Sed purus eros, tempor eget, commodo vel, molestie sed, justo. Praesent fringilla sollicitudin neque. Fusce eleifend dolor ut odio. Sed faucibus, lectus a ultricies tempor, nulla metus porttitor nibh, at sodales nulla neque eu dolor. Vivamus venenatis velit sed nulla.
Nullam hendrerit consectetuer
Aliquam sit amet tortor ac purus porttitor porta. Aenean sodales felis sed mauris. Aliquam purus. Morbi interdum. Elis at. Vullan utet luptat vulput lan henis nulputpat. Tationsed duipit nim.
Nullam hendrerit consectetuer erat. Maec enas non justo a quam egestas euismod. Nam iaculis porttitor risus. Maecenas ac enim. Sed purus eros nullam gravida.
Im quis alit vel dolendreros eumsan henissim nostrud eu facilit ut nulput loborem quatin hendignim ssrit vullam euis at autat inciliquamet.
Aliquat lor ip er sent lortis ex elit dolore ver sit prating ea feuguer cillupt atueraesed dolesto dolore feui bla l ea corper aci tatue commy nulput nullam vulpute mod magnit iPasim vercil ute magna augue consecte doloreet ametuer cilisi. Vivamus ultrices.
Maecenas id massa a libero gravida suscipit. Mauris convallis. Aliquam risus sem, venenatis nec. Mauris vestibulum ipsum egestas tur Pas. Cras erat. Aliquam sit amet tortor ac purus porttitor porta. Aenean sodales felis sed mauris. Aliquam purus. Morbi interdum. Mauris sed justo. In iaculis felis non justo. Nullam pellentesque vehicula libero. Curabitur facilisis. Mauris lectus velit, vestibulum sed, laoreet at, porta vitae, nisl. Vestibulum dictum, sapien vitae gravida sollicitudin, lorem felis dapibus wisi, vel tincidunt erat wisi a lacus.
Vestibulum ante ipsum
Pellentesque nibh nisl, euismod eget, imperdiet ut, ultrices eu, ipsum. Pellentesque elementum dignissim felis. Proin neque elit, tempor vitae, malesuada vel, dictum et, purus. Maecenas ac enim. Sed purus eros, tempor eget, commodo vel, molestie.
Sed ipsum erat, porttitor non, venenatis id, vestibulum non, sem. Suspendisse vestibulum placerat odio. Phasellus eget massa vitae libero imperdiet elementum. Integer dapibus quam non mauris. Maecenas ac enim. Sed purus eros, tempor eget, commodo vel, molestie sed, justo.
— Turpis Egestas
Nullam arcu leo, facilisis ut
1
125% 528 Words Page 1 of 1

Untitled

View Outline Pages Text Box Shapes Table Charts Comment Inspector Media Colors Fonts

S E Q U O I A C L U B

Redwoods

Aliquam sit amet tortor ac purus porttitor porta. Aenean sodales felis sed mauris. Sed ipsum erat, porttitor non, venenatis id, vestibulum non, sem.

A Quarterly Newsletter — Issue N° 17 — Fall 2009

Donec odio risus
Cras eget lectus. Quisque facilisis mattis eros. Vivamus felis augue, malesuada nec, congue nec, semper a, risus. Mauris quam. Morbi eget nunc nec eros bibendum.
Page 2

Maecenas faucibus
Sed ipsum erat, dign porttitor non, venenatis id, vestibulum. Amet eget massa vitae libero.
Page 2

Cras condimentum consectetuer orci
Sed ipsum erat, porttitor non, venenatis id, vestibulum. Phasellus eget massa vitae libero imperdiet lorem elementum.
Page 2

Nullam pellentesque

Suspendisse vestibulum placerat odio. Phasellus eget massa vitae lorem libero non.
Page 2

Nullam hendrerit
Nullam auctor enim quis nibh. Maecenas fermentum. Morbi placerat dign. Praesent fringilla sollicitudin neque. Fusce ipsum eleifend dolor.
Page 3

Morbi vitae ligula
Vivamus venenatisvelit sed nulla. Curabitur sodales ornare urna. Vivamus ultrices. Maecenas eget id a libero gravid.
Page 4

Pellentes que habitan

Lorem ipsum dolor sit amet, consectetuer adiPascing elit. Pellentesque ipsum tur Pas, ullamcorper at, molestie facilisis mattis eros. Vivamus felis augue, malesuada nec, congue nec, semper a, risus. Mauris quam.

Pellentesque nibh nisl, euismod eget, imperdiet ut, ultrices eu, ipsum. Pellentesque elementum dignissnon mauris. Maecenas ac enim. Sed purus eros, tempor eget, commodo vel, molestie sed, justo. Praesent fringilla sollicitudin neque. Fusce eleifend dolor ut odio. Sed faucibus, lectus a ultricies tempor, nulla metus porttitor nibh, at sodales nulla neque eu dolor. Vivamus venenatis velit sed nulla.

Nullam hendrerit consectetuer

Aliquam sit amet tortor ac purus porttitor porta. Aenean sodales felis sed mauris. Aliquam purus. Morbi interdum. Elis at. Vullan utet luptat vulput lan henis nulputpat. Tationsed duipit nim.

Nullam hendrerit consectetuer erat. Maec enas non justo a quam egestas euismod. Nam iaculis porttitor risus. Maecenas ac enim. Sed purus eros nullam gravida.

Im quis alit vel dolendreros eumsan henissim nostrud eu facilit ut nulput loborem quatin hendignim ssrit vullam euis at autat inciliquamet.

Aliquat lor ip er sent lortis ex elit dolore ver sit prating ea feuguer cillupt atueraesed dolesto dolore feui bla l ea corper aci tatue commy nulput nullam vulpute mod magnit iPasim vercil ute magna augue consecte doloreet ametuer cilisi. Vivamus ultrices.

Maecenas id massa a libero gravida suscipit. Mauris convallis. Aliquam risus sem, venenatis nec. Mauris vestibulum ipsum egestas tur Pas. Cras erat. Aliquam sit amet tortor ac purus porttitor porta. Aenean sodales felis sed mauris. Aliquam purus. Morbi interdum. Mauris sed justo. In iaculis felis non justo. Nullam pellentesque vehicula libero. Curabitur facilisis. Mauris lectus velit, vestibulum sed, laoreet at, porta vitae, nisl. Vestibulum dictum, sapien vitae gravida sollicitudin, lorem felis dapibus wisi, vel tincidunt erat wisi a lacus.

Vestibulum ante ipsum

Pellentesque nibh nisl, euismod eget, imperdiet ut, ultrices eu, ipsum. Pellentesque elementum dignissim felis. Proin neque elit, tempor vitae, malesuada vel, dictum et, purus. Maecenas ac enim. Sed purus eros, tempor eget, commodo vel, molestie.

Sed ipsum erat, porttitor non, venenatis id, vestibulum non, sem. Suspendisse vestibulum placerat odio. Phasellus eget massa vitae libero imperdiet elementum. Integer dapibus quam non mauris. Maecenas ac enim. Sed purus eros, tempor eget, commodo vel, molestie sed, justo.

— Turpis Egestas

125% 528 Words Page 1 of 1

Occlusion Notifications

Occlusion Notifications

- App occluded
  - Implement `-applicationDidChangeOcclusionState:` in your app delegate
  - Or check

```
if ([NSApp occlusionState] & NSApplicationOcclusionStateVisible) ...
```

Occlusion Notifications

- App occluded
  - Implement `-applicationDidChangeOcclusionState:` in your app delegate
  - Or check

```
if ([NSApp occlusionState] & NSApplicationOcclusionStateVisible) ...
```

- Window occluded
  - Implement `-windowDidChangeOcclusionState:` in your window delegate
  - Or check

```
if ([window occlusionState] & NSWindowOcclusionStateVisible) ...
```

Occlusion Notifications

- App occluded
  - Implement `-applicationDidChangeOcclusionState:` in your app delegate
  - Or check

```
if ([NSApp occlusionState] & NSApplicationOcclusionStateVisible) ...
```

- Window occluded
  - Implement `-windowDidChangeOcclusionState:` in your window delegate
  - Or check

```
if ([window occlusionState] & NSWindowOcclusionStateVisible) ...
```

Batching Periodic Work

- Group maintenance operations together
- Do them later with user-requested activity
- Replaces timer-driven “cleanup” or “timeout” work
  - Causes wakeups
  - Hurts your race to idle

Batching Periodic Work

```
- (void)cacheImage:(NSImage *)anImage {
      /* ... */

      [self performSelector:@selector(removeImageIfStale:)
                  withObject:anImage
                  afterDelay:5.0];
}

- (void)removeImageIfStale:(NSImage *)anImage {
      if ([self imageIsStale:anImage])
            [self removeImage:anImage];
      else
            [self performSelector:@selector(removeImageIfStale:)
                        withObject:anImage
                        afterDelay:5.0];
}
```

Batching Periodic Work

```
- (void)cacheImage:(NSImage *)anImage {
      /* ... */

      [imageArray addObject:anImage];
}

- (void)handleUserEvent {
      dispatch_async(..., ^{
            for (NSImage *image in imageArray)
                  if ([self imageIsStale:anImage])
                        [self removeImage:anImage];
      });

      /* ... */
}
```

Centralized Task Scheduling

- Periodic work that can wait for wall power
  - Clean-up work
  - Syncing
  - Indexing
  - Downloading new content
- New XPC API

Centralized Task Scheduling

```
xpc_object_t criteria = xpc_dictionary_create(NULL, NULL, 0);
xpc_dictionary_set_bool(criteria, XPC_ACTIVITY_REPEATING, FALSE);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_DELAY, 3600);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_GRACE_PERIOD, 6 * 3600);
xpc_dictionary_set_string(criteria,
                  XPC_ACTIVITY_PRIORITY, XPC_ACTIVITY_PRIORITY_MAINTENANCE);

xpc_activity_register("com.myapp.MySimpleActivity", criteria,
      ^(xpc_activity_t activity) {
            /* do background work here */
      });
```

Centralized Task Scheduling

```
xpc_object_t criteria = xpc_dictionary_create(NULL, NULL, 0);
xpc_dictionary_set_bool(criteria, XPC_ACTIVITY_REPEATING, FALSE);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_DELAY, 3600);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_GRACE_PERIOD, 6 * 3600);
xpc_dictionary_set_string(criteria,
                    XPC_ACTIVITY_PRIORITY, XPC_ACTIVITY_PRIORITY_MAINTENANCE);

xpc_activity_register("com.myapp.MySimpleActivity", criteria,
      ^(xpc_activity_t activity) {
            /* do background work here */
      });
```

Centralized Task Scheduling

```
xpc_object_t criteria = xpc_dictionary_create(NULL, NULL, 0);
xpc_dictionary_set_bool(criteria, XPC_ACTIVITY_REPEATING, FALSE);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_DELAY, 3600);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_GRACE_PERIOD, 6 * 3600);
xpc_dictionary_set_string(criteria,
                    XPC_ACTIVITY_PRIORITY, XPC_ACTIVITY_PRIORITY_MAINTENANCE);

xpc_activity_register("com.myapp.MySimpleActivity", criteria,
      ^(xpc_activity_t activity) {
            /* do background work here */
      });
```

Centralized Task Scheduling

```
xpc_object_t criteria = xpc_dictionary_create(NULL, NULL, 0);
xpc_dictionary_set_bool(criteria, XPC_ACTIVITY_REPEATING, FALSE);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_DELAY, 3600);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_GRACE_PERIOD, 6 * 3600);
xpc_dictionary_set_string(criteria,
                    XPC_ACTIVITY_PRIORITY, XPC_ACTIVITY_PRIORITY_MAINTENANCE);

xpc_activity_register("com.myapp.MySimpleActivity", criteria,
      ^(xpc_activity_t activity) {
            /* do background work here */
      });
```

Centralized Task Scheduling

```
xpc_object_t criteria = xpc_dictionary_create(NULL, NULL, 0);
xpc_dictionary_set_bool(criteria, XPC_ACTIVITY_REPEATING, FALSE);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_DELAY, 3600);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_GRACE_PERIOD, 6 * 3600);
xpc_dictionary_set_string(criteria,
                    XPC_ACTIVITY_PRIORITY, XPC_ACTIVITY_PRIORITY_MAINTENANCE);

xpc_activity_register("com.myapp.MySimpleActivity", criteria,
      ^(xpc_activity_t activity) {
            /* do background work here */
      });
```

Centralized Task Scheduling

```
xpc_object_t criteria = xpc_dictionary_create(NULL, NULL, 0);
xpc_dictionary_set_bool(criteria, XPC_ACTIVITY_REPEATING, FALSE);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_DELAY, 3600);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_GRACE_PERIOD, 6 * 3600);
xpc_dictionary_set_string(criteria,
                  XPC_ACTIVITY_PRIORITY, XPC_ACTIVITY_PRIORITY_MAINTENANCE);

xpc_activity_register("com.myapp.MySimpleActivity", criteria,
      ^(xpc_activity_t activity) {
            /* do background work here */
      });
```

Centralized Task Scheduling

```
xpc_object_t criteria = xpc_dictionary_create(NULL, NULL, 0);
xpc_dictionary_set_bool(criteria, XPC_ACTIVITY_REPEATING, FALSE);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_DELAY, 3600);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_GRACE_PERIOD, 6 * 3600);
xpc_dictionary_set_string(criteria,
                  XPC_ACTIVITY_PRIORITY, XPC_ACTIVITY_PRIORITY_MAINTENANCE);

xpc_activity_register("com.myapp.MySimpleActivity", criteria,
      ^(xpc_activity_t activity) {
            /* do background work here */
      });
```

Centralized Task Scheduling

```
xpc_object_t criteria = xpc_dictionary_create(NULL, NULL, 0);
xpc_dictionary_set_bool(criteria, XPC_ACTIVITY_REPEATING, FALSE);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_DELAY, 3600);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_GRACE_PERIOD, 6 * 3600);
xpc_dictionary_set_string(criteria,
                  XPC_ACTIVITY_PRIORITY, XPC_ACTIVITY_PRIORITY_MAINTENANCE);

xpc_activity_register("com.myapp.MySimpleActivity", criteria,
      ^(xpc_activity_t activity) {
            /* do background work here */
      });
```

Centralized Task Scheduling

```
xpc_object_t criteria = xpc_dictionary_create(NULL, NULL, 0);
xpc_dictionary_set_bool(criteria, XPC_ACTIVITY_REPEATING, FALSE);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_DELAY, 3600);
xpc_dictionary_set_int64(criteria, XPC_ACTIVITY_GRACE_PERIOD, 6 * 3600);
xpc_dictionary_set_string(criteria,
                  XPC_ACTIVITY_PRIORITY, XPC_ACTIVITY_PRIORITY_MAINTENANCE);

xpc_activity_register("com.myapp.MySimpleActivity", criteria,
      ^(xpc_activity_t activity) {
            /* do background work here */
      });
```

Summary

Summary

- A little CPU costs a lot

Summary

- A little CPU costs a lot
- So achieve absolute idle

Summary

- A little CPU costs a lot
- So achieve absolute idle
- Test early, test often

More Information

Paul Danbold
Core OS Technologies Evangelist
danbold@apple.com

Dave DeLong
App Frameworks Evangelist
delong@apple.com

Documentation
Instruments User Guide
http://developer.apple.com

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

Related Sessions

Maximizing Battery Life on OS X	Mission Tuesday 11:30AM
Improving Power Efficiency with App Nap	Pacific Heights Wednesday 10:15AM
Power and Performance: Optimizing Your Website for Great Battery Life and Responsive Scrolling	Russian Hill Wednesday 9:00AM
Building Efficient OS X Apps	Nob Hill Tuesday 4:30PM
Efficient Design with XPC	Russian Hill Tuesday 2:00PM

Labs

Power and Performance for OS X Apps	Core OS Lab A Wednesday 10:15AM

WWDC2013