

Writing Energy Efficient Code Part 1

Session 710 Anthony Chivetta OS X Performance & Power

© 2014 Apple Inc. All rights reserved. Redistribution or public display not permitted without written permission from Apple.









3:38 Remaining Power Source: Battery

Apps Using Significant Energy

Show Percentage
Open Energy Saver Preferences...

÷••••	9:41	АМ	100% 💼	
Battery Usage				
BATTE	RY USAGE			
Last 24 Hours		Last	Last 7 Days	
	Safari		24%	
f	Facebook		19%	
	Phone		14%	
	Mail Background Activi	ty	13%	
	Maps Location		11%	
Ρ	Pandora ^{Audio}		9%	
	Messages		6%	
('A')	No Cell Cove	rage	4%	

Shows proportion of battery used by each app when iPhone is not charging.





Power and energy concepts





Power and energy concepts Improving your energy use

• Do it never



- Do it never
- Do it at a better time



- Do it never
- Do it at a better time
- Do it more efficiently



- Do it never
- Do it at a better time
- Do it more efficiently
- Do it less



Power and energy concepts Improving your energy use

- Do it never
- Do it at a better time
- Do it more efficiently
- Do it less

Part 2



Power and energy concepts Improving your energy use

- Do it never
- Do it at a better time
- Do it more efficiently
- Do it less

Part 2

Networking



Power and energy concepts Improving your energy use

- Do it never
- Do it at a better time
- Do it more efficiently
- Do it less

Part 2

- Networking
- Location



Power and energy concepts Improving your energy use

- Do it never
- Do it at a better time
- Do it more efficiently
- Do it less

Part 2

- Networking
- Location
- Sleep/Wake





CPU



CPU



Storage



CPU



Storage



Networking



CPU



Storage





Networking

Graphics

Power















Power





Power





Power





Energy and Power



Energy and Power



Energy and Power





Trading Power for Energy





Trading Power for Energy





Trading Power for Energy




Power Fundamentals

Work has a fixed cost

- For small workloads, this can dominate
- For intensive workloads, dynamic cost will dominate

Better performance = less energy

Do It Never



Your App		
Categories Purchases	Updates)
Your Ad		



Your App		
Categories Purchases	Updates)
Your Ad		





Active App Transitions iOS

- (void)applicationDidResignActive: (UIApplication *)application { // Pause animations and UI updates }
- (void)applicationDidBecomeActive: (UIApplication *)application { Resume animations and UI updates }
- Or, listen for the UIApplicationWillResignActiveNotification notification.



Active App Transitions OS X

- (void)applicationDidResignActive:(NSNotification *)aNotification { // Pause animations and UI updates }
- (void)applicationDidBecomeActive:(NSApplication *)aNotification { Resume animations and UI updates }
- Or, listen for the NSApplicationDidResignActiveNotification notification.





Occlusion notifications indicate visibility of windows or applications



- Occlusion notifications indicate visibility of windows or applications For applications, implement delegate method
- (void)applicationDidChangeOcclusionState: (NSNotification *)notification
- Or check
- if ([NSApp occlusionState] & NSApplicationOcclusionStateVisible)





- Occlusion notifications indicate visibility of windows or applications
- For applications, implement delegate method
- (void)applicationDidChangeOcclusionState:(NSNotification *)notification
- Or check
- if ([NSApp occlusionState] & NSApplicationOcclusionStateVisible)
 For windows, implement delegate method
- (void)windowDidChangeOcclusionState:(NSNotification *)notification
- Or check
- if ([window occlusionState] & NSWindowOcclusionStateVisible)





App Nap reduces an inactive app's energy use



App Nap reduces an inactive app's energy use App Nap relies on heuristics



App Nap reduces an inactive app's energy useApp Nap relies on heuristicsYou are the authoritative source for what work is important



App Nap reduces an inactive app's energy useApp Nap relies on heuristicsYou are the authoritative source for what work is importantIn a well-behaved app, App Nap should never be in effect during work



App Nap reduces an inactive app's energy use App Nap relies on heuristics You are the authoritative source for what work is important In a well-behaved app, App Nap should never be in effect during work reason:(NSString *)reason usingBlock:(void (^)(void))block



-NSProcessInfo (void)performActivityWithOptions:(NSActivityOptions)options

Avoiding Unnecessary Work

Monitor app state to know when work isn't visible Avoid updating UI until the user can see the results Nap yourself when not in use, so App Nap doesn't have to











Battery Remaining

6AM





























6AM





New API in OS X Yosemite



New API in OS X Yosemite Allows scheduling arbitrary tasks for a good time in the future



New API in OS X Yosemite Allows scheduling arbitrary tasks for a good time in the future Supports repeating or non-repeating activities



New API in OS X Yosemite

Allows scheduling arbitrary tasks for a good time in the future

Supports repeating or non-repeating activities

Can be used to schedule

- Periodic content fetch
- Update install
- Garbage collection and data maintenance tasks
- Automatic saves or backups



Creating a Scheduler

activity = [[NSBackgroundActivityScheduler alloc] initWithIdentifier:@"com.apple.sample-app.MyActivity"];

Each activity must have an identifier Identifier should be in reverse-DNS style Name should be unique, but the same across app runs

"com.example.MyApp.updatecheck"





Specifying Scheduling Properties

// Activity will fire in the next 10 minutes activity.tolerance = 10×60 ;



Specifying Scheduling Properties

// Activity will fire in the next 10 minutes activity tolerance = 10 * 60;

// Activity will fire between 15 and 45 minutes from now activity.interval = 30 * 60; activity.tolerance = 15 * 60;



Specifying Scheduling Properties

// Activity will fire in the next 10 minutes activity tolerance = 10×60 ;

// Activity will fire between 15 and 45 minutes from now activity.interval = 30×60 ; activity tolerance = 15×60 ;

// Activity will fire once each hour activity.repeats = YES activity_interval = 60 * 60;


Scheduling Your Work



Scheduling Your Work

[activity

// do the work

completion(NSBackgroundActivityResultFinished); }];





scheduleWithBlock:^(NSBackgroundActivityCompletionHandler completion){

Scheduling Your Work Deferring in-progress work

```
[activity
   for ( /* each item of work */) {
      if (activity.shouldDefer){
          completion(NSBackgroundActivityResultDeferred);
          return;
      }
      // do item of work
   }
   completion(NSBackgroundActivityResultFinished);
}];
```







scheduleWithBlock:^(NSBackgroundActivityCompletionHandler completion) {

Scheduling Your Work Deferring in-progress work









scheduleWithBlock:^(NSBackgroundActivityCompletionHandler completion){

You specify scheduling requirements for work



You specify scheduling requirements for work System select the best time to perform that work



You specify scheduling requirements for work System select the best time to perform that work Support for repeating tasks without drift



You specify scheduling requirements for work System select the best time to perform that work Support for repeating tasks without drift Available in OS X Yosemite or with the xpc_activity C API in 10.9



Your App



































Your App







+ backgroundSessionConfigurationWithIdentifier: "com.example.MySessionIdentifier"





configuration.discretionary = YES;



configuration.discretionary = YES; Discretionary sessions available in iOS 7.0 and OS X Yosemite



configuration.discretionary = YES; Discretionary sessions available in iOS 7.0 and OS X Yosemite Automatically picks the best time to do work



configuration.discretionary = YES; Discretionary sessions available in iOS 7.0 and OS X Yosemite Automatically picks the best time to do work Provides bandwidth monitoring and automatic retry



configuration.discretionary = YES; Discretionary sessions available in iOS 7.0 and OS X Yosemite Automatically picks the best time to do work Provides bandwidth monitoring and automatic retry Scheduling window can be adjusted with configuration.timeoutIntervalForResource = 24*60*60; // 1 day

- If this time elapses, an error will be thrown
- Should be >12 hours



Related Session

What's New in Foundation Networking

Nob Hill

Tuesday 3:15PM

Do It More Efficiently



Resource Management Properties

Responsiveness

Efficiency

Resource Management Properties



CPU Scheduler Priority

I/O Priority

Efficiency

Resource Management Properties



CPU Scheduler Priority

I/O Priority



Quality of Service Classes

UI	User Interactive	Main thre
ΙΝ	User Initiated	Immediat
UT	Utility	Long-runi
BG	Background	Not user v

ead, animations

te results

ning tasks

visible



User Interactive

Is this work actively involved in updating the UI? e.g., main thread, animations, input event processing

UI	User Interactive	Is this work a e.g., main thr
IN	User Initiated	Is this work r e.g., loading a

actively involved in updating the UI? read, animations, input event processing

required to continue user interaction? active content

UI	User Interactive	Is this work a e.g., main thr
IN	User Initiated	Is this work in the second sec
UT	Utility	Is the user a e.g., long-rur

actively involved in updating the UI? read, animations, input event processing

required to continue user interaction? active content

ware of the progress of this work? nning jobs with progress indicators

UI	User Interactive	Is this work a e.g., main thr
IN	User Initiated	Is this work in the second s
UT	Utility	Is the user a e.g., long-run
BG	Background	Can this wor e.g., if so, use

actively involved in updating the UI? read, animations, input event processing

required to continue user interaction? active content

ware of the progress of this work? nning jobs with progress indicators

rk be deferred to a better time?
NSBackgroundActivityScheduler





UI	User Interactive	Is it okay for
IN	User Initiated	Is it okay for
UT	Utility	Is it okay for
BG	Background	

User Interactive work to happen before my work?

this work to compete with other User Initiated work?

my work to take precedence over Utility work?


Background

User Initiated



Background





User Initiated





Background







PhotoMeister 3000





Search













Converting ... (10/100)



User Interactive

Main Thread (automatic)



PhotoMeister 3000





Search













Converting ... (10/100)



User Interactive

- Main Thread (automatic)
- User Initiated
- Thumbnail generation
- Image load (on click)



PhotoMeister 3000





Search













Converting ... (10/100)



User Interactive

- Main Thread (automatic)
- User Initiated
- Thumbnail generation
- Image load (on click)

Utility

Image import and conversion



PhotoMeister 3000





Search













Converting ... (10/100)



User Interactive

- Main Thread (automatic)
- User Initiated
- Thumbnail generation
- Image load (on click)

Utility

- Image import and conversion Background
- Search indexing



PhotoMeister 3000





Search













Converting ... (10/100)



Queue Structure



Queue Structure





Queue Structure







NSOperation and NSOperationQueue now have a qualityOfService property: operation.qualityOfService = NSQualityOfServiceUtility;

NSOperation and NSOperationQueue now have a qualityOfService property: operation.qualityOfService = NSQualityOfServiceUtility; If set on both the operation and queue, the higher will be used

NSOperation and NSOperationQueue now have a qualityOfService property: operation.qualityOfService = NSQualityOfServiceUtility; If set on both the operation and queue, the higher will be used If not set, NSOperations will infer a QoS from the environment when possible

QoS Application







QoS Application



Main Thread (User Interactive)



Thumbnail Generation NSOperationQueue – User Initiated



QoS Application



Main Thread (User Interactive)



Thumbnail Generation NSOperationQueue – User Initiated



QoS Is Not Static

The logical QoS of an operation may change over time

• e.g., when user requests the result of work that's already happening a lower QoS

QoS Is Not Static

The logical QoS of an operation may change over time

- e.g., when user requests the result of work that's already happening a lower QoS With NSOperation, the QoS of an operation can be promoted by:
- Enqueueing a higher QoS operation on the same queue
- Using addDependency: with a higher QoS operation
- waitUntilFinished: or waitUntilAllOperationsAreFinished: from a higher QoS thread



Main Thread





Image Click Event



Main Thread



Image Click Event // Find operation for image





Main Thread

Image Click Event // Find operation for image operation.queuePriority = NSOperationQueuePriorityVeryHigh;





Image Click Event
// Find operation for image
operation.queuePriority =
 NSOperationQueuePriorityVeryHigh;
operation.qualityOfService =
 NSQualityOfServiceUserInitiated;

Main Thread





Image Click Event
// Find operation for image
operation.queuePriority =
 NSOperationQueuePriorityVeryHigh;
operation.qualityOfService =
 NSQualityOfServiceUserInitiated;

Main Thread



Feed Reader 900	00	Search
News	Lorem ipsum dolo consectetur adipi	or sit amet, scing elit. Ut
Neat Things	volutpat accumsan tellus, eu imperdiet est elementum sed.	
Cats	Sed tincidunt, hib consectetur posu	ere, est metus.
Kittens	Mauris in elementum orci,	
26 ways	varius est. Sed fringilla velit a vestibulum sagi.	
		Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Loading content

Feed Reader 9000

News

Neat Things

Cats

Kittens

26 ways...

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut volutpat accumsan tellus, eu imperdiet est elementum sed. Sed tincidunt, nibh ut consectetur posuere, est metus.

Mauris in elementum orci, varius est. Sed fringilla velit a vestibulum sagi.





Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Updating Feeds...

Search

Loading content

User Initiated

Feed Reader 9000

News

Neat Things

Cats

Kittens

26 ways...

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut volutpat accumsan tellus, eu imperdiet est elementum sed. Sed tincidunt, nibh ut consectetur posuere, est metus.

Mauris in elementum orci, varius est. Sed fringilla velit a vestibulum sagi.





Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Updating Feeds...

Search

Loading content
User Initiated
Image pre-fetching

Feed Reader 900	00	Search
News	Lorem ipsum dolo consectetur adipi	or sit amet, scing elit. Ut
Neat Things	volutpat accumsan tellus, eu imperdiet est elementum sed.	
Cats	Sed tincidunt, hib consectetur posu	ere, est metus.
Kittens	Mauris in elementum orci,	
26 ways	varius est. Sed fringilla velit a vestibulum sagi.	
		Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Loading content

- User Initiated
 Image pre-fetching
- Background

Feed Reader 900	00	Search
News	Lorem ipsum dolo consectetur adipi	or sit amet, scing elit. Ut
Neat Things	volutpat accumsan tellus, eu imperdiet est elementum sed.	
Cats	Sed tincidunt, hib consectetur posu	ere, est metus.
Kittens	Mauris in elementum orci,	
26 ways	varius est. Sed fringilla velit a vestibulum sagi.	
		Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Loading content

- User Initiated
 Image pre-fetching
- Background
 Fetching new feeds

Feed Reader 900	00	Search
News Neat Things Cats	Lorem ipsum dole consectetur adipi volutpat accumsa imperdiet est eler Sed tincidunt, nib consectetur posu	or sit amet, scing elit. Ut an tellus, eu mentum sed. oh ut ere, est metus.
Kittens 26 ways	Mauris in elementum orci, varius est. Sed fringilla velit a vestibulum sagi.	
		Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Loading content

User Initiated

Image pre-fetching

Background

Fetching new feeds

Requested by user—User Initiated

Feed Reader 900	00	Search
News	Lorem ipsum dol consectetur adipi	or sit amet, scing elit. Ut
Neat Things	volutpat accumsan tellus, eu imperdiet est elementum sed.	
Cats	Sed tincidunt, nib consectetur posu	oh ut ere, est metus.
Kittens	Mauris in elementum orci,	
26 ways	varius est. Sed fringilla velit a vestibulum sagi.	
		Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Loading content

User Initiated

Image pre-fetching

Background

Fetching new feeds

- Requested by user—User Initiated
- Automatic—Utility

Feed Reader 900	00	Search
News	Lorem ipsum dol consectetur adipi	or sit amet, scing elit. Ut
Neat Things	volutpat accumsan tellus, eu imperdiet est elementum sed.	
Cats	Sed tincidunt, nib consectetur posu	oh ut ere, est metus.
Kittens	Mauris in elementum orci,	
26 ways	varius est. Sed fringilla velit a vestibulum sagi.	
		Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Loading content

User Initiated

Image pre-fetching

Background

Fetching new feeds

- Requested by user—User Initiated
- Automatic—Utility

Search indexing

Feed Reader 900	00	Search
News	Lorem ipsum dol consectetur adipi	or sit amet, scing elit. Ut
Neat Things	volutpat accumsan tellus, eu imperdiet est elementum sed.	
Cats	Sed tincidunt, hib consectetur posu	oh ut ere, est metus.
Kittens	Mauris in elementum orci,	
26 ways	varius est. Sed fringilla velit a vestibulum sagi.	
		Lorem ipsum dolor sit amet, consectetur adipiscing elit.

Loading content

User Initiated

Image pre-fetching

Background

Fetching new feeds

- Requested by user—User Initiated
- Automatic—Utility
- Search indexing
- Background

Feed Reader 900	00	Search
News	Lorem ipsum dol consectetur adipi	or sit amet, scing elit. Ut
Neat Things	volutpat accumsan tellus, eu imperdiet est elementum sed.	
Cats	Sed tincidunt, hib consectetur posu	oh ut ere, est metus.
Kittens	Mauris in elementum orci,	
26 ways	varius est. Sed fringilla velit a vestibulum sagi.	
		Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Set breakpoints to confirm requested QoS

Set breakpoints to confirm requested QoS Use powermetrics to confirm which QoS is in use

Set breakpoints to confirm requested QoS Use powermetrics to confirm which QoS is in use Use spindump to determine the QoS code is executing with

•	MyApplication >	My Mac 64-bit	Running MyApplication : MyApplication
		IIII < > E CPU Repor	đ
•	MyApplicationPID 7632, PausedCPU7%		Percentage Used
	Memory 12.7 MB Energy Impact Low Disk Zero KB/s Network Zero KB/s		20 4 1 400
•	 Thread 1 Queue: com.apple.main-thread (serial) 0 mach_msg_trap 11 NSApplicationMain 12 main 13 start Thread 2 Queue: com.apple.libdispatch-manager (serial) Thread 5 Queue: com.apple.root.utility-qos (concurrent) 0	Usage over Time Duration: 45 se High: 31% Low: 0%	r 31%
	 2 usleep 345-[AppDelegate applicationDidFini 4 _dispatch_call_block_and_release 9 start_wqthread 9 start_wqthread Enqueued from com.apple.main-thread (Thr 0 _dispatch_async_f_slow 1 -[AppDelegate applicationDidFinishLa 2CFNOTIFICATIONCENTER_IS_CALL 	Thread 1 User Interact Thread 2 User Interact Thread 3 QoS Unavaila	ive ive
A A	 18 NSApplicationMain 19 main 20 start Thread 7 Thread 9 	Thread 5 Utility Thread 7 User Interact	ive
•	ji Thread 10	Default	1 III / A MyApplication)
	I 🔘	Auto 🗘 💿 👩	

	< 🔺 >
d Usage Comparison MyApplication 7% Other Processes 70% Free 323%	

~			
		190s	
			¢.
			-
1	Thread 5 > 💽 0semwait_signa	al	
		(lldb)	
	•	All Output \$	Ū 🔲 🗖

•	••		МуАр	plication $ angle$ $igsquare$	My Mac 64-bit	Running MyApp	lication : MyApplication		A 1				
	in 77	Q A	♦ ≡ ■) (j	🛗 < 💛 🔳 CPU Repo	ort							< 🔺
•	MyAppli PID 7632	ication , Paused		0 E. 7%			Percentage Used			Usage Con	nparison		
	Memory Energy Disk	y Impact		12.7 MB Low Zero KB/s		4	20 89 400	7%			 MyApplication 7% Other Processes 70% Free 323% 	5	
C	Networ	k		Zero KB/s									
► ►	Threa Queue O n O n O n O n O n O n O n O n O n O n	ad 1 e: com.apple.m nach_msg_tra NSApplicatio main start ad 2 e: com.apple.lib ad 5 e: com.apple.ro	nain-thread (seri ap onMain bdispatch-mana oot.utility-qos (c	ial) ager (serial) concurrent)	Usage ove Time Duration: 45 s High: 31% Low: 0%	er ^{31%}	s						90s
	0 _ 2 u 2 u 2 3 _ 4 _ 0 9 s Enqueu 0 _ 1 -	_semwait_sig sleep _45-[AppDele dispatch_call tart_wqthread ued from com dispatch_asy [AppDelegate	gnal egate applicati I_block_and_re d n.apple.main-tl /nc_f_slow e applicationDi	onDidFini elease hread (Thr dFinishLa	Threads Thread 1 User Interac Thread 2 User Interac Thread 3	tive tive							
	2 _ 18 19 20	_CFNOTIFICA NSApplicatio main start	ATIONCENTEF onMain	?_IS_CALL	QoS Unavai Thread 5 Utility	able _							
►	👅 Threa	ad 7			User Interac	tive							
A A	i Threa	ad 9 ad 10			Thread 9 Default								
						1 🗉 🖌	A MyApplication >	Thread 5 > 🖸 0semwait_sig	Inal				
									(lldb)				
=					Auto 🗘 💿 👩			Θ	All Output \$				Ū 🗖 [



•	MyApplication >	My Mac 64-bit Running My	yApplication : MyApplication	A 1	
		IIII < > Marcelle CPU Report			< ▲
	MyApplication PID 7632, PausedImage: CPUCPU7%		Percentage Used	Usage Comparison	
	Memory 12.7 MB Energy Impact Low Disk Zero KB/s		20 89 7%	MyApp 7% Other 70% Free 323%	lication Processes
C	Network Zero KB/s				
•	 Thread 1 Queue: com.apple.main-thread (serial) 0 mach_msg_trap 11 NSApplicationMain 12 main 13 start 13 start Queue: com.apple.libdispatch-manager (serial) Thread 5 Queue: com.apple.root.utility-qos (concurrent)	Usage over Time Duration: 45 sec High: 31% Low: 0%	31% 0s		190s
	 0semwait_signal 2 usleep 345-[AppDelegate applicationDidFini 4 _dispatch_call_block_and_release 9 start_wqthread Enqueued from com.apple.main-thread (Thr 0 _dispatch_async_f_slow 1 -[AppDelegate applicationDidFinishLa 2CFNOTIFICATIONCENTER_IS_CALL 18 NSApplicationMain 19 main 20 start Thread 7 Thread 9 	Threads Thread 1 User Interactive Thread 2 User Interactive Thread 3 QoS Unavailable Thread 5 Utility Thread 7 User Interactive Thread 9			
	Thread 10	Default			
			MyApplication > <a>Thread 5 <a> <a> <a> <a> <a> <a> <a> <a> <a> <	nal (lldb)	
I		Auto 🗘 💿 🔞	Θ	All Output \$	Ū 🗖 [



	MyApplication >	My Mac 64-bit Running My	yApplication : MyApplication	▲ 1	
		🔛 < 💛 📠 CPU Report			I < 🔺 1
	MyApplication PID 7632, PausedImage: CPUCPU7%Memory12.7 MBEnergy ImpactLowDiskZero KB/s	4	Percentage Used	Usage C	Comparison MyApplication 7% Other Processes 70% Free 323%
	Network Zero KB/s Thread 1 Queue: com.apple.main-thread (serial) O mach_msg_trap 11 NSApplicationMain 12 main 13 start Thread 2 Queue: com.apple.libdispatch-manager (serial) Thread 5 Queue: com.apple.root.utility-qos (concurrent) O _semwait_signal 2 usleep 3 _45-[AppDelegate applicationDidFini 4 _dispatch_call_block_and_release 9 start_wqthread Enqueued from com.apple.main-thread (Thr 9 o_dispatch_async_f_slow 1 -[AppDelegate applicationDidFinishLa 2 _CFNOTIFICATIONCENTER_IS_CALL 1 a NSApplicationMain 2 o start Thread 7 Thread 10	Usage over Time Duration: 45 sec High: 31% Low: 0% Threads Thread 1 User Interactive Thread 2 User Interactive Thread 3 QoS Unavailable Thread 5 Utility Thread 7 User Interactive Thread 7 User Interactive Thread 9 Default			
			✓ A MyApplication > Thread 5 > 0 _	semwait_signal (11db)	
=		Auto 🗘 💿 🔞	Θ	All Output \$	Ū 🗖 🗖

•			lyApplication $ angle$ $igsqcelowbreak$	My Mac 6	4–bit		Running My	Application : MyApplic	atio
	i	A 🗢 📼	•	IIII <	> 📻 CF	PU Report			
•	MyApplication PID 7632, Pause	d	0 2.					Percentage L	Jsec
	CPU		7%						
	Memory		12.7 MB						
	Energy Impact		Low					20	
							4	69	
	Disk		Zero KB/s				1	400	
(Network		Zero KB/s						
1	Thread 1 Queue: com.a O mach_m 11 NSApp 12 main 13 start Thread 2 Queue: com.a Thread 5 Queue: com.a	pple.main-thread sg_trap licationMain pple.libdispatch-r pple.root.utility-q	(serial) nanager (serial) os (concurrent)		Usa Time Durat High: Low:	ge over e ion: 45 sec 31% 0%	31	%	
	 0semwa 2 usleep 345-[Ap 4 _dispate 9 start_wo 9 start_wo 0 _dispate 0 _dispate 1 -[AppDe 2CFNO 18 NSApp 19 main 	ait_signal ppDelegate appl h_call_block_ar thread n com.apple.ma h_async_f_slow legate applicatio TIFICATIONCEN licationMain	icationDidFini Id_release In-thread (Thr InDidFinishLa ITER_IS_CALL		Thre User Thre User Thre QoS Thre Utilit	eads ad 1 Interactiv ad 2 Interactiv ad 3 Unavailat ad 5	/e /e		
	 D start Thread 7 				Thre	ad 7			
	Thread 9				User	Interactiv	/e		
	🕨 🧃 Thread 10				Thre Defa	ad 9 ult			
						*	1 III -	MyApplication) 1
-				Auto					

on	A 1				
ed 7%		Usage Cor	MyApplication 7% Other Processes 70% Free 323%		
					905
				,	
Thread 5 > 🖸 0semwait_signa	I				
	(lldb)				

0

All Output 🗘

Ū | 🗖 🗖

% sudo powermetrics ---show-process-qos ---samplers tasks

% sudo powermetrics ---show-process-qos --samplers tasks
*** Sampled system activity (Wed May 28 00:03:12 2014 -0700) (5006.36ms
elapsed) ***

*** Running tasks ***

 Name
 ID
 CPU ms/s
 User%

 MyApplication
 8424
 88.89
 94.16

 Deadlines (<2 ms, 2-5 ms)</td>
 Wakeups (Intr, Pkg idle)

 227.89
 0.00
 228.89
 165.98

QOS (ms/s) Default Maint BG 0.00 0.00 0.04

 Util
 Lgcy
 U-Init
 U-Intr

 88.64
 0.03
 0.00
 0.17

% sudo powermetrics ---show-process-qos ---samplers tasks *** Sampled system activity (Wed May 28 00:03:12 2014 -0700) (5006.36ms elapsed) ***

*** Running tasks ***

Name	ID	CPU ms/s	User%	
MyApplication	8424	88.89	94.16	
Deadlines (<2 ms, 2–5 ms) 227.89 0.00	Wakeups (1 228.89 16	Intr, Pkg id 65.98	le)	
QOS (ms/s) Default Maint	BG U1	til Lgcy	U-Init	U-Intr
0.00 0.00	0.04 88	8.64 0.03	0.00	0.17

% sudo powermetrics ---show-process-qos ---samplers tasks *** Sampled system activity (Wed May 28 00:03:12 2014 -0700) (5006.36ms elapsed) ***

*** Running tasks ***

Name	
MyApplication	8

Deadline	es	(<2	ms,	2–5	ms)	Wakeup
227.89	0.	00				228.89

QOS	(ms/s)	Default	Maint	BG
		0.00	0.00	0.04

CPU ms/s User% 8424 88.89 94.16 os (Intr, Pkg idle)

165.98

Util Lgcy U-Init U-Intr 0.17 88.64 0.03 0.00

% sudo powermetrics ---show-process-qos ---samplers tasks *** Sampled system activity (Wed May 28 00:03:12 2014 -0700) (5006.36ms elapsed) ***

*** Running tasks ***

ID CPU ms/s Name User% MyApplication 8424 88.89 94.16 Deadlines (<2 ms, 2-5 ms) Wakeups (Intr, Pkg idle) 227.89 228.89 165.98 0.00 QOS (ms/s) Default Maint Util Lgcy **U**-Intr BG U-Init 0.00 0.00 0.04 88.64 0.03 0.00 0.17

% sudo spindump -timeline MyApplication

% sudo spindump -timeline MyApplication Thread 0x6bb7a DispatchQueue 6 1000 samples (1–1000) priority 16–20 cpu time 1.488s <thread QoS utility, priority 20> 1000 start_wqthread + 13 (libsystem_pthread.dylib + 6657) [0x7fff82e73a01] 1-1000 1000 _pthread_wqthread + 663 (libsystem_pthread.dylib + 15313) [0x7fff82e75bd1] 1-1000 1000 _dispatch_worker_thread3 + 79 (libdispatch.dylib + 72233) [0x10002fa29] 1-1000 1000 _dispatch_root_queue_drain + 1408 (libdispatch.dylib + 17968) [0x100022630] 1-1000

4 ____45-[AppDelegate applicationDidFinishLaunching:]_block_invoke + 181 (AppDelegate.m:25 in MyApplication + 4837) [0x1000012e5] 1-4<thread QoS background>

_45-[AppDelegate applicationDidFinishLaunching:]_block_invoke + 217 (AppDelegate.m:28 in MyApplication + 4837) [0x1000012e5] 1-4

% sudo spindump -timeline MyApplication Thread 0x6bb7a DispatchQueue 6 1000 samples (1–1000) priority 16–20 cpu time 1.488s <thread QoS utility, priority 20> start_wqthread + 13 (libsystem_pthread.dylib + 6657) [0x7fff82e73a01] 1-1000 10001000 _pthread_wqthread + 663 (libsystem_pthread.dylib + 15313) [0x7fff82e75bd1] 1-1000

4 ____45-[AppDelegate applicationDidFinishLaunching:]_block_invoke + 181 (AppDelegate.m:25 in MyApplication + 4837) [0x1000012e5] 1-4<thread QoS background>

_45-[AppDelegate applicationDidFinishLaunching:]_block_invoke + 217 (AppDelegate.m:28 in MyApplication + 4837) [0x1000012e5] 1-4

% sudo spindump -timeline MyApplication Thread 0x6bb7a DispatchQueue 6 1000 samples (1-1000) priority 16-20 cpu time 1.488s <thread QoS utility, priority 20> start_wqthread + 13 (libsystem_pthread.dylib + 6657) [0x7fff82e73a01] 1-1000 10001000 _pthread_wqthread + 663 (libsystem_pthread.dylib + 15313) [0x7fff82e75bd1] 1-1000 1000 _dispatch_worker_thread3 + 79 (libdispatch.dylib + 72233) [0x10002fa29] 1–1000 1000 _dispatch_root_queue_drain + 1408 (libdispatch.dylib + 17968) [0x100022630] 1-1000

4 ____45-[AppDelegate applicationDidFinishLaunching:]_block_invoke + 181 (AppDelegate.m:25 in MyApplication + 4837) [0x1000012e5] 1-4 <thread QoS background>

_45-[AppDelegate applicationDidFinishLaunching:]_block_invoke + 217 (AppDelegate.m:28 in MyApplication + 4837) [0x1000012e5] 1-4

Quality of Service

Specify the responsiveness and energy requirements of work Available in both Foundation and C APIs Classify long-running or resource-intensive operations in your existing code Aim for >90% of time at Utility or below when the user is inactive

Related Session

Power, Performance, and Diagnostics:
 What's New in GCD and XPC

Russian Hill Thursday 2:00PM

Do It Less





CPU



Graphics

Storage



	🔥 💻 м	y Mac 64–bit	Running M	yApplic	ation :	My	App	licat	ior
🖬 🎞 Q 🗛 🗢 🧮	▶ 🖗	=== (Energy	Report	81				
MyApplication PID 9197, Running	0 i.	Energy	-						
CPU	95%	5 <u>-</u>							
					I It	iliza	atio	n	
Memory	10.4 MB				0.				
Eperav Impact	High				1				
Disk	Zero KB/s							ŀ	
Network	Zero KB/s							E	ne
		Energy I	mpact						
		CPU High	Utilization						
		App Nap F	Prevention			4	<	<u> </u>	5
		Idle Wake	Prevention	(x x	xx	x	x	x	¢.
		CPU Wake	e Overhead						
				<u>1</u>	1		1	A	M







yApplication



	A> 🗖 M	y Mac 64-bit	Running I	MyAppl	icatio	n : My	App	lica	tior
	• • •	Ⅲ < →	Energy	y Repor	t				
MyApplication PID 9197, Running	0 E.	Energy	e -						
CPU	95%	-							
Memory	10.4 MB				ι	Jtiliz	atio	n	
Energy Impact	High				1				
🗾 Disk	Zero KB/s							ŀ	-1
Network	Zero KB/s							E	ne
		Energy I	mpact						
		CPU High	Utilization	ZZ	ZZ	ZZ	2	X	Z
		App Nap H	Prevention	~ ~ ~	1 21	~ ~			
		CPU Wake	e Overhead	~~ /		~ ~			
= •			- 2	± 1	1	1 -	◀	A	My









Image: CPU Memory 10.4 MB Image: CPU Image: CPU	Image:			atior				
MyApplication PID 9197, Running CPU 95% Memory 10.4 MB Energy Impact High Disk Zero KB/s Network Zero KB/s CPU High Utilization App Nap Prevention Idle Wake Prevention Idle Wake Prevention Idle Wake Overhead		• 9	< >	Energy	Report			
 CPU 95% Memory 10.4 MB Energy Impact High Disk Zero KB/s Network Zero KB/s Energy Impact Energy Impact Energy Impact Energy Impact Energy Impact CPU High Utilization App Nap Prevention Ide Wake Prevention CPU Wake Overhead 	MyApplication PID 9197, Running	0 E.	Energy					
Memory 10.4 MB Energy Impact High Disk Zero KB/s Network Zero KB/s	CPU	95%	-					
Memory 10.4 MB Energy Impact High Network Zero KB/s CPU High Utilization App Nap Prevention Idle Wake Prevention Idle Wake Overhead						l Itiliza	ation	
Energy Impact High Disk Zero KB/s Network Zero KB/s	Memory	10.4 MB				Othize		
Disk Zero KB/s Network Zero KB/s Energy Impact CPU High Utilization App Nap Prevention Idle Wake Prevention Idle Wake Overhead	Energy Impact	High				6i		
Disk Zero KB/s Network Zero KB/s Energy Impact CPU High Utilization App Nap Prevention Idle Wake Prevention CPU Wake Overhead								11
Network Zero KB/s Energy Impact CPU High Utilization App Nap Prevention Idle Wake Prevention CPU Wake Overhead	Disk	Zero KB/s						Η
Energy Impact CPU High Utilization App Nap Prevention Idle Wake Prevention CPU Wake Overhead	Motwork	Zoro KP/o						Ene
Energy Impact	Metwork	Zero KD/S		E [[4			
Energy Impact								
Energy Impact								
CPU High Utilization			Energy II	mpact				
CPU High Utilization App Nap Prevention Idle Wake Prevention CPU Wake Overhead CPU Wake Overhead								_
CPU High Utilization App Nap Prevention Idle Wake Prevention CPU Wake Overhead CPU Wake Overhead								
App Nap Prevention Idle Wake Prevention CPU Wake Overhead CPU Wake Overhead			CPU High	Litilization				
Idle Wake Prevention CPU Wake Overhead			App Nap P	Prevention	~ ~ ~ ~ ~		ZZ	Z
CPU Wake Overhead			Idle Wake	Prevention	xxxx	xx	××	X
			CPU Wake	Overhead				
					L 1		1	M









1% CPU

10% CPU

100% CPU

· -	





10% CPU

100% CPU

· -	





100% CPU

· -	





•••	
	_
	E
	2
$\bullet \bullet \bullet$	
• • •	

10x power draw



•	● ● ▶ ■ AyApplication > ■	My Mac 64-bit	Running MyApplication : MyApplication
		📖 < 🛛 > 🔳 CPU Repor	rt
•	MyApplication I The State of S		Percentage Used
	Memory 12.7 MB		20
	Energy Impact Low		4 89
	Disk Zero KB/s		400
C	Network Zero KB/s	<u>-</u>	
• •	 Thread 1 Queue: com.apple.main-thread (serial) 0 mach_msg_trap 11 NSApplicationMain 12 main 13 start Thread 2 Queue: com.apple.libdispatch-manager (serial) Thread 5 Queue: com.apple.root.utility-qos (concurrent) 	Usage over Time Duration: 45 se High: 31% Low: 0%	r 31%
	 0semwait_signal 2 usleep 345-[AppDelegate applicationDidFini 4 _dispatch_call_block_and_release 	Threads	
	9 start_wqthread Enqueued from com.apple.main-thread (Thr	User Interact	ive
	0_dispatch_async_f_slow	User Interact	ive
	2CFNOTIFICATIONCENTER_IS_CALL	Thread 3 QoS Unavaila	able
	19 main	Thread 5 Utility	
►	Thread 7	Thread 7 User Interact	ive
•	Thread 9Thread 10	Thread 9	
			1 III / A MyApplication)
=		Auto 🗘 💿 🔞	

on	<u>∧</u> 1		<u>)</u>
		< 🛆	>
*d 7%	Usage C	Comparison MyApplication 7% Other Processes 70% Free 323%	

~~~			
		190s	
<b>Threa</b>	ad 5 > 🖸 0semwait_signa	al	
		(lldb)	
●		All Output 🗘	Ū   🗖 🗖

•	MyApplication >	My Mac 64-bit	Running MyApplication : MyApplication
		🗰   < 💛   📻 CPU Repor	't
• /	MyApplication PID 7632, Paused		Dercentere Llee
	7% CPU		Percentage Osec
	Memory 12.7 MB		
E	Energy Impact		4 89
	Disk Zero KB/s		400
C	Network Zero KB/s		
	<ul> <li>Thread 1 Queue: com.apple.main-thread (serial)</li> <li>0 mach_msg_trap</li> <li>11 NSApplicationMain</li> <li>12 main</li> <li>13 start</li> <li>Thread 2 Queue: com.apple.libdispatch-manager (serial)</li> <li>Thread 5 Queue: com.apple.root.utility-qos (concurrent)</li> </ul>	Usage over Time Duration: 45 se High: 31% Low: 0%	31% ec 0s
	<ul> <li>0semwait_signal</li> <li>2 usleep</li> <li>345-[AppDelegate applicationDidFini</li> <li>4 _dispatch_call_block_and_release</li> <li>9 start_wqthread</li> <li>Enqueued from com.apple.main-thread (Thr</li> </ul>	Threads Thread 1 User Interact Thread 2	ive
	<ul> <li>0 _dispatch_async_f_slow</li> <li>1 -[AppDelegate applicationDidFinishLa</li> <li>2CFNOTIFICATIONCENTER_IS_CALL</li> <li>18 NSApplicationMain</li> <li>19 main</li> </ul>	User Interacti Thread 3 QoS Unavaila Thread 5	ive able
Þ	20 start Thread 7	Thread 7	
•	Thread 9	Thread 9	
			1 III / I MyApplication >
-		Auto 🗘   💿 🔞	

on	<b>A</b> 1			
				< 🔺 >
ed	Usage Co	mparison		
7%		<ul> <li>MyApplication 7%</li> <li>Other Processes 70%</li> <li>Free 323%</li> </ul>		
			190s	
道 Thread 5 > 🖸 0semwait_signa	I			
	(lldb)			

0

All Output 🗘

Ū | 🗖 🗖

MyApplication >	My Mac 64-bit Running MyApplication : MyAp	oplication	<b>▲</b> 1	
	IIII   < >   CPU Report			I< 🔺 >
MyApplication PID 7632, Paused CPU Memory 12.7 MB Energy Impact Low Disk Zero KB/s	Percentag	ge Used	Usage Comp	<ul> <li>MyApplication 7%</li> <li>Other Processes 70%</li> <li>Free 323%</li> </ul>
<ul> <li>Thread 1 Queue: com.apple.main-thread (serial)</li> <li>0 mach_msg_trap</li> <li>11 NSApplicationMain</li> <li>12 main</li> <li>13 start</li> <li>Thread 2 Queue: com.apple.libdispatch-manager (serial)</li> <li>Thread 5 Queue: com.apple.root.utility-qos (concurrent)</li> <li>0semwait_signal</li> <li>2 usleep</li> <li>345-[AppDelegate applicationDidFini 4dispatch_call_block_and_release</li> <li>9 start_wqthread</li> <li>Enqueued from com.apple.main-thread (Thr</li> <li>0dispatch_async_f_slow</li> <li>1 -[AppDelegate applicationDidFinishLa</li> <li>2CFNOTIFICATIONCENTER_IS_CALL</li> <li>18 NSApplicationMain</li> <li>19 main</li> <li>20 start</li> <li>Thread 7</li> <li>Thread 10</li> </ul>	Usage over Time Duration: 45 sec High: 31% Low: 0% ThreadS ThreadS Thread 1 User Interactive Thread 2 User Interactive Thread 3 QoS Unavailable Thread 5 Utility Thread 7 User Interactive Thread 9 Default			190a
	Image: Second	ation 〉 🧊 Thread 5 〉 💽 0semwait_signa	ו (lldb)	
	Auto 🗘 💿 🔞		All Output \$	Ū   🗖 🗖

	-		
		MacBool	Run 1 of 1
All Cores	All	Processes	' Threads
<u>.</u>	Instru	ments	与o:do'''''''''''''''''''''''''''''''''''
	) Time	Profiler	And many and provide
	_	_	
ulu			
🚳 Time F	Profiler		E Call Tree Call Tree
Running Tir	me∽	Self	Symbol Name
1906.0ms	97.7%	0.0	Main Thread 0x6adef
1894.0ms	97.1%	0.0	▼start libdyld.dylib
1894.0ms	97.1%	0.0	NSApplicationMain AppKit
1827.0ms	93.6%	0.0	
1819.0ms	93.2%	0.0	Image: Sector
1816.0ms	93.1%	0.0	DPSNextEvent AppKit
1811.0ms	92.8%	0.0	BlockUntilNextEventMatchingListInModeW
1811.0ms	92.8%	1.0	ReceiveNextEventCommon HIToolbox
1793.0ms	91.9%	0.0	RunCurrentEventLoopInMode HIToolbo
1783.0ms	91.4%	0.0	CFRunLoopRunSpecific CoreFoundation
1782.0ms	91.3%	0.0	CFRunLoopRun CoreFoundation
1702.0ms	87.2%	0.0	CFRunLoopDoObservers CoreF
1699.0ms	87.1%	0.0	CFRUNLOOP_IS_CALLING_OU
1645.0ms	84.3%	0.0	83-[NSWindow _postWindow
1645.0ms	84.3%	0.0	_handleWindowNeedsDisplay
1622.0ms	83.1%	0.0	Instruction
1622.0ms	83.1%	0.0	[NSView displayIfNeeded]
1608.0ms	82.4%	0.0	Image: Section of the section of
14.0ms	0.7%	0.0	Image: SendViewWill
20.0ms	1.0%	0.0	CFAutoreleasePoolPop Co
2.0ms	0.1%	0.0	CA::Transaction::commit()
1.0ms	0.0%	1.0	-[NSWindow isVisible] App
50.0ms	2.5%	0.0	35-[NSWindow _postInvalidC
2.0ms	0.1%	0.0	38-[NSApplication setWindow
1.0ms	0.0%	0.0	[NSWindow(NSDrag) _register[

n	S	tr	u	m	e	nt	S	
8 A	~	MR.	-		~	1.1.1	<u> </u>	ι.

00:00:18

+ 🖸 🖃 🖼 🔲

1 1	1 <mark>0</mark> 0:30' ' '	ιιι ι <mark>ο</mark> ό:	40'''''	' ' ¹ 00:50'	 lo1:do' ' '	' ' ' '  01:1	0''''''	' loi:2

Q- Involves Symbol		$\odot$		
	Sample Perspe	ective		
		O All Sample	e Counts	
		Running S	ample Times	
	Call Tree			
Date:inMode:dequeue:] AppKit		Separate	by Thread	
		Invert Ca	ll Tree	
VithFilter HIToolbox		Hide Miss	sing Symbols	
		Hide Syst	tem Libraries	
OX		Flatten R	ecursion	
tion		Top Func	tions	
Foundation		trainte		
UT TO AN OBSERVER CALLBACK FUNCTION CoreF		anto (		
NeedsDisplayOrLayoutOrUpdateConstraintsUnlessPosting	Count	0	00	
OrLayoutOrUpdateConstraints AppKit	I Ime (ms)	-00	00	
ed] AppKit ] AppKit	Data Mining			
noringOpacity:isVisibleRect:rectIsVisibleRectForView:] Ap				_
DrawInRect:clipRootView:] AppKit				
oreFoundation				
QuartzCore				
Nit NurserRestal block inveks2020 AppKit	Symbol	Library	Rest	ore
wsNeedLindate:] block invoke2481 AppKit	Coymbol	Library	(Itest	
DragTypes:] AppKit				

### Performance Unit Tests

### Performance Unit Tests

Additions to XCTestCase API in Xcode 6

### Performance Unit Tests

Additions to XCTestCase API in Xcode 6 Helps you find performance regressions
## Performance Unit Tests

Additions to XCTestCase API in Xcode 6 Helps you find performance regressions

```
- (void)testSomething
{
    [self measureBlock:^{
        // ...snip..
    }];
}
```

#### Related Sessions

#### • Testing in Xcode 6

• Continuous Integration with Xcode 6

Marina	Thursday 9:00AM
Marina	Thursday 2:00PM

CPU use has a huge dynamic range in power

CPU use has a huge dynamic range in power Monitor CPU use with Xcode debug gauge

CPU use has a huge dynamic range in power Monitor CPU use with Xcode debug gauge Profile with Instruments

CPU use has a huge dynamic range in power Monitor CPU use with Xcode debug gauge Profile with Instruments Prevent regressions with performance unit tests

#### Minimize Timers

#### Minimize Timers

-[NSObject performSelector:withObject:afterDelay:]

pthread_cond_timedwait()

NSTimer

sleep()

CVDisplayLink

dispatch_semaphore_wait()

#### CFRunLoopTimer

Grand Central Dispatch timers

#### select()









🛑 😑 🕨 🔳 🚕 🖾 My Mac 64–bit Running				МуА	pplic	atio	n : N	МуA	۱	icati	ior
	● 🖗	IIII   < →	Energ	y Re	port						
MyApplication PID 9197, Running	0 2.	Energy	é								
CPU	95%	1									
						1	14:11		tion		
Memory	10.4 MB					L	Jtil	IZa	tior	1	
Energy Impact	High										
										L	
Disk	Zero KB/s										
	7 1/0/									Er	าย
Metwork	Zero KB/s										
		Energy I	mpact			1					
		CPU High	Utilization	>	> >			>	>>	> >	Ŧ
		App Nap F	Prevention	~			9	~		4	1
		Idle Wake	Prevention	X	xx	×	X	X	x	K 3	e
		CPU Wake	e Overhead								
				L	4	1 0				A	N.A.
			2	Ě.	100			1		1	IVI







yApplication



My Mac 64-bit			Running M	lyApplicat	ion : My	Applic	atior
	<b>-</b>	<b>***</b>   < >	Energy	Report			
MyApplication PID 9197, Running	0 i.	Energy	ł j				
	95%						
Memory	10.4 MB				Utiliza	tion	
Energy Impact	High						
Disk	Zero KB/s						H
Network	Zero KB/s						=ne
		Energy I	mpact				
		CPU High	Utilization	222	ZZ	ZZ	Z
		App Nap F	Prevention				
		Idle Wake	Prevention	(XX)	< X X	××	X
				1 1		1   1	Ny My







Application



🔵 🌑 🕨 🔳 🔌 💻 My Mac 64-bit Runnin					plica	tion :	My	Арр	lica	atio	'n
	• 9	< >	Energy	y Rep	oort						
MyApplication PID 9197, Running	02.	Energy									
CPU	95%										
						L It	iliza	atio	n		
Memory	10.4 MB					U.	11120	1110			
Energy Impact	Hiah					/					
Disk	Zero KB/s								ľ		
Network	Zero KB/s								E	:ne	e
		Energy I	mpact					_			
		CPU High	Utilization	ZZ	2	ZZ	Z	2	X	Z	
		App Nap F	Prevention								
		CPU Wake	Prevention	XX	< X	XX	X	X	×	×	
				±	1	1	-	1	A	e M	IJ







Application



#### Diagnosing Timer Issues timerfires

\$ sudo timerfires -p MyApplication -s -g

#### Diagnosing Timer Issues timerfires

- \$ sudo timerfires -p MyApplication -s -g PROCESS TYPE TIMER ROUTINE COUNT PID MyApp dispatch MyApp`-[AppModel updateWidgets:] 1603 555 MyApp`-[AppDelegate timerFired:] MyApp CF 1603 735 1603 MyApp 1127 sleep libsystem_kernel.dylib`__semwait_signal+0xa
  - libsystem_c.dylib`usleep+0x36 MyApp`-[AppModel pollForChange]+0x1a
  - libdispatch.dylib`_dispatch_call_block_and_release+0xc

## Timer Coalescing



## Timer Coalescing



## Specify Timer Tolerance

# Specify Timer Tolerance

[myTimer setTolerance:60.0];

CFRunLoopTimerSetTolerance(myTimer, 60.0);

dispatch_source_set_timer(my_timer, DISPATCH_TIME_NOW,  $30 * NSEC_PER_SEC, 60 * NSEC_PER_SEC);$ 

## Minimize Timers

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



#### Related Session

Energy Best Practices

#### WWDC 2013



































# Limit Screen Updates

- Avoid extraneous screen updates
- Unnecessary drawing kicks graphics hardware out of low-power modes Drawing more content than needed causes extra power draw to update the screen • Use needsToDrawRect: or getRectsBeingDrawn:count: methods to fine-tune drawing









Graphics Tools for Xcode: http://developer.apple.com





#### Core Animation



#### Visual Effects

Translucent blurs have an energy cost Avoid placing over updating elements

#### Visual Effects

Translucent blurs have an energy cost Avoid placing over updating elements


### Visual Effects

Translucent blurs have an energy cost Avoid placing over updating elements





### Visual Effects

Translucent blurs have an energy cost Avoid placing over updating elements





# Efficient Graphics

Draw minimally and efficiently Monitor drawing with Quartz Debug or Instruments Avoid blurs on updating content

# Flash Power

Writes to Flash are much more energy hungry than reads

- Write the minimum content necessary
- Do writes in aggregate for better power efficiency



# Flash Power

Writes to Flash are much more energy hungry than reads

- Write the minimum content necessary
- Do writes in aggregate for better power efficiency Any I/O will pull device out of low-power states
- Use caching to your advantage



# Do It Less

Profile and monitor CPU Reduce timers Be efficient in the use of graphics Minimize I/O

Improving your app's energy consumption improves user experience

Improving your app's energy consumption improves user experience Continuously monitor your app's energy and resource consumption

- Improving your app's energy consumption improves user experience Continuously monitor your app's energy and resource consumption Look for ways to
- Do it never—Respond to changes in active state

Improving your app's energy consumption improves user experience Continuously monitor your app's energy and resource consumption Look for ways to

- Do it never—Respond to changes in active state
- Do it at a better time—Let the system schedule work

tive state chedule work

Improving your app's energy consumption improves user experience Continuously monitor your app's energy and resource consumption Look for ways to

- Do it never—Respond to changes in active state
- Do it at a better time—Let the system schedule work
- Do it more efficiently—Specify Quality of Service Classes on your work

Improving your app's energy consumption improves user experience Continuously monitor your app's energy and resource consumption Look for ways to

- Do it never—Respond to changes in active state
- Do it at a better time—Let the system schedule work
- Do it more efficiently—Specify Quality of Service Classes on your work
- Do it less—Optimize and improve your resource use

Improving your app's energy consumption improves user experience Continuously monitor your app's energy and resource consumption Look for ways to

- Do it never—Respond to changes in active state
- Do it at a better time—Let the system schedule work
- Do it more efficiently—Specify Quality of Service Classes on your work
- Do it less—Optimize and improve your resource use And stick around for Part 2

# More Information

Paul Danbold Core OS Evangelist danbold@apple.com

Energy Best Practices WWDC 2013

Building Resource Efficient Apps WWDC 2013

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

## Related Sessions

- What's New in Foundation Networkin
- Improving Your App with Instruments
- Writing Energy Efficient Code, Part 2
- Testing in Xcode 6
- Fix Bugs Faster Using Activity Tracing
- Continuous Integration with Xcode 6
- Power, Performance, and Diagnostics: What's New in GCD and XPC

g	Nob Hill	Tuesday 3:15PM
5	Marina	Tuesday 4:30PM
	Russian Hill	Wednesday 11:30AM
	Marina	Thursday 9:00AM
	Russian Hill	Thursday 11:30AM
	Marina	Thursday 2:00PM
	Russian Hill	Thursday 2:00PM



### Power and Performance Lab

Instruments Lab

### Power and Performance Lab

Core OS Lab B	Wednesday 2:00PM
Tools Lab B	Thursday 9:00AM
Core OS Lab A	Thursday 3:15PM

