Grand Central Dispatch

... getting started with concurrency on iOS

by Matthew Henderson
@ghostm
http://bitly.com/YgI3eS

I make apps

- I work for Empirical Development
- I run Haunted Robot in my spare time
- I've worked on a lot of apps
- GCD makes my life much nicer

*Colors taken from Moonrise Kingdom and idea stolen from @collindonnell

Concurrency?

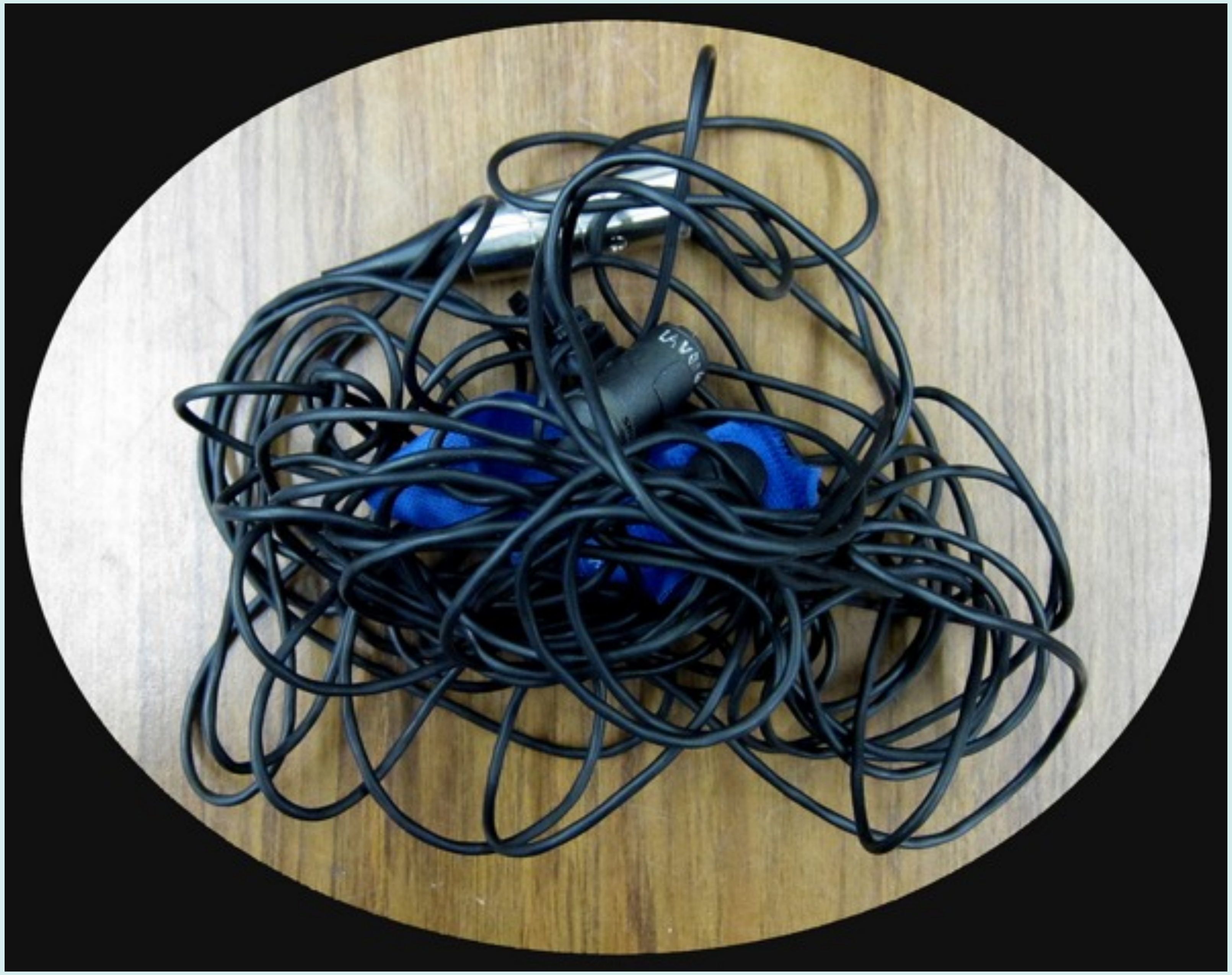
- Multiple things happening at the same time
- Networking, UI, core data, processing...

Does it matter?

- Can't have a UI that stutters
- Apps are doing more and more
- iOS devices now have multiple cores

Threads

Threads are hard



Grand Central Dispatch

GCD

- Apple calls it "A better way to do multicore"

GCD

- "The key innovation of GCD is shifting the responsibility for managing threads and their execution from applications to the operating system"

GCD helps you do this

GCD

- Extensions to the C language
- New API
- Runtime engine

GCD

- Blocks
- Dispatch queues
- Dispatch {sync/async/apply}

GCD

- Dispatch groups
- Dispatch semaphores
- Dispatch barriers

and more ...

- Dispatch sources
- Dispatch i/o

Beyond GCD

- NSOperation and NSOperationQueue

Blocks

- ^{...}
- Encapsulated unit of work
  - Several niceties such as allowing access to local variables

Blocks

- Get to know blocks, because they are very powerful and lead to better code.

GCD Blocks

- typedef void (^dispatch_block_t)(void);

Dispatch Queues

- All work in GCD is done using FIFO queues (system runs these using a pool of threads that the user can't control)

Blocks of work

Blocks of work with GCD

Dispatch Queues

- Atomic enqueue
- Automatic dequeue

Dispatch Queues

- Types of queues
  - Main
  - Concurrent
  - Serial

Dispatch Queues

- Types of queues
  - System provided queues
  - User created queues

Dispatch Queues

- Add work to queues using
    - dispatch_sync
    - dispatch_async
    - dispatch_apply
    - dispatch_barrier

dispatch_sync

dispatch_sync

- Useful for critical sections
- Can be used to synchronize a section of code
- Blocks execution until block finishes
- Can deadlock

dispatch_async

dispatch_async

- Deferred execution.
    - Returns immediately.
- Move work off the main (UI) thread
- Queue determines serial or concurrent execution
- Independent serial queues are processed concurrently

dispatch_apply

- Conceptually, dispatch_apply() is a convenient wrapper around dispatch_async() and a semaphore to wait for completion

dispatch_apply

- Avoid doing too little work compared to the overhead of queuing/ dispatching them
- Sometimes, when the block passed to dispatch_apply() is simple, the use of striding can tune performance

dispatch_apply

- Synchronous
  - wrap in dispatch_async

dispatch barrier

dispatch barrier

- Synchronization point in a concurrent queue
- Execution is delayed until previous blocks finish
- Executed by itself
- After completion queue resumes normal operation

dispatch group

- submit multiple blocks and track when they all complete
- works across multiple queues

dispatch semaphore

- Control/limit access to a resource
- A dispatch semaphore is an efficient implementation of a traditional counting semaphore

dispatch sources

- Async interfaces for
  - Timers
  - Unix signal
  - File/sockets
  - Custom

dispatch i/o

- Async read and write to file descriptors

NSOperation{Queue}

- Cocoa objects built on-top of GCD
- Extra overhead and extra features
- Operation dependencies
- Operation priority
- KVO on operations

Resources

- http://www.mikeash.com
- Concurrency Guide
- WWDC videos

Thanks

@ghostm

http://bitly.com/YgI3eS