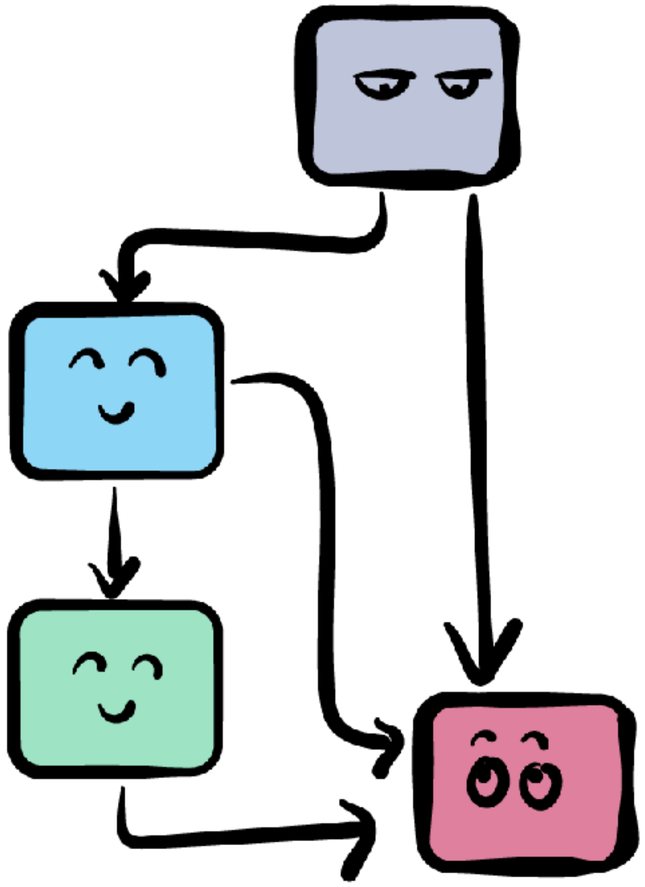
iOS Concurrency with GCD & Operations



Part 9: Cancelling Tasks

Cancelling an Operation

`cancel()`

Operation

```
isCancelled = false
isFinished  = false
```

Operation

```
isCancelled = false
isFinished  = false
```

Operation

```
isCancelled = true
isFinished  = false
```

Operation

```
open class Operation : NSObject {
  open var isCancelled: Bool { get }
  open func cancel()
  ...
}
```

```
extension AsyncOperation {
  ...
  override func cancel() {
    state = .Finished
  }
  override var isExecuting: Bool {
    return state == .Executing
  }
  override var isFinished: Bool {
    return state == .Finished
  }
}
```

OperationQueue

```
open class OperationQueue : NSObject {

  open func cancelAllOperations()
  ...
}
```

Cancelling DispatchWorkItem

```
public class DispatchWorkItem {
  public init(qos: DispatchQoS = default, flags: DispatchWorkItemFlags = default,
    block: @escaping @convention(block) () -> ())
  public func perform()
  public func wait()
  public func wait(timeout: DispatchTime) -> DispatchTimeoutResult
  public func wait(wallTimeout: DispatchWallTime) -> DispatchTimeoutResult
  public func notify(qos: DispatchQoS = default,
    flags: DispatchWorkItemFlags = default, queue: DispatchQueue,
    execute: @escaping @convention(block) () -> Swift.Void)
  public func notify(queue: DispatchQueue, execute: DispatchWorkItem)
  public func cancel()
  public var isCancelled: Bool { get }
}
```

Challenge Time!