Creating NSOperation Subclasses

Brice Wilson
www.BriceWilson.net
@brice_wilson

pluralsight
hardcore dev and IT training

Introduction

Concurrent Operations

Non-Concurrent Operations

Defining Operations

Cancellation Support

Concurrent Versus Non-Concurrent Operations

- Inherit from NSOperation
- Non-Concurrent Operations Easier to Implement
- Non-Concurrent Operation + Queue == Concurrent Execution
- Define Non-Concurrent Operations to Implement Cancellation

Defining a Non-Concurrent Operation

```
@interface BuyTicketsOperation : NSOperation

@end

@implementation BuyTicketsOperation

-(void)main {

    // Do some work

    // Check to see if the operation was cancelled

    // Do some more work

    // Repeat as necessary

}

@end
```

Cancellation Support

- **Clients may cancel an operation at any time**
- **Use the isCancelled method to check for cancellation**
- **Check isCancelled at the following times**
    - Before performing any actual work
    - During each iteration of a loop
    - Any point in the code where it would be easy to cancel

Summary

- Concurrent Versus Non-Concurrent Operations
- Creating Custom NSOperation Subclasses
- Cancellation Support