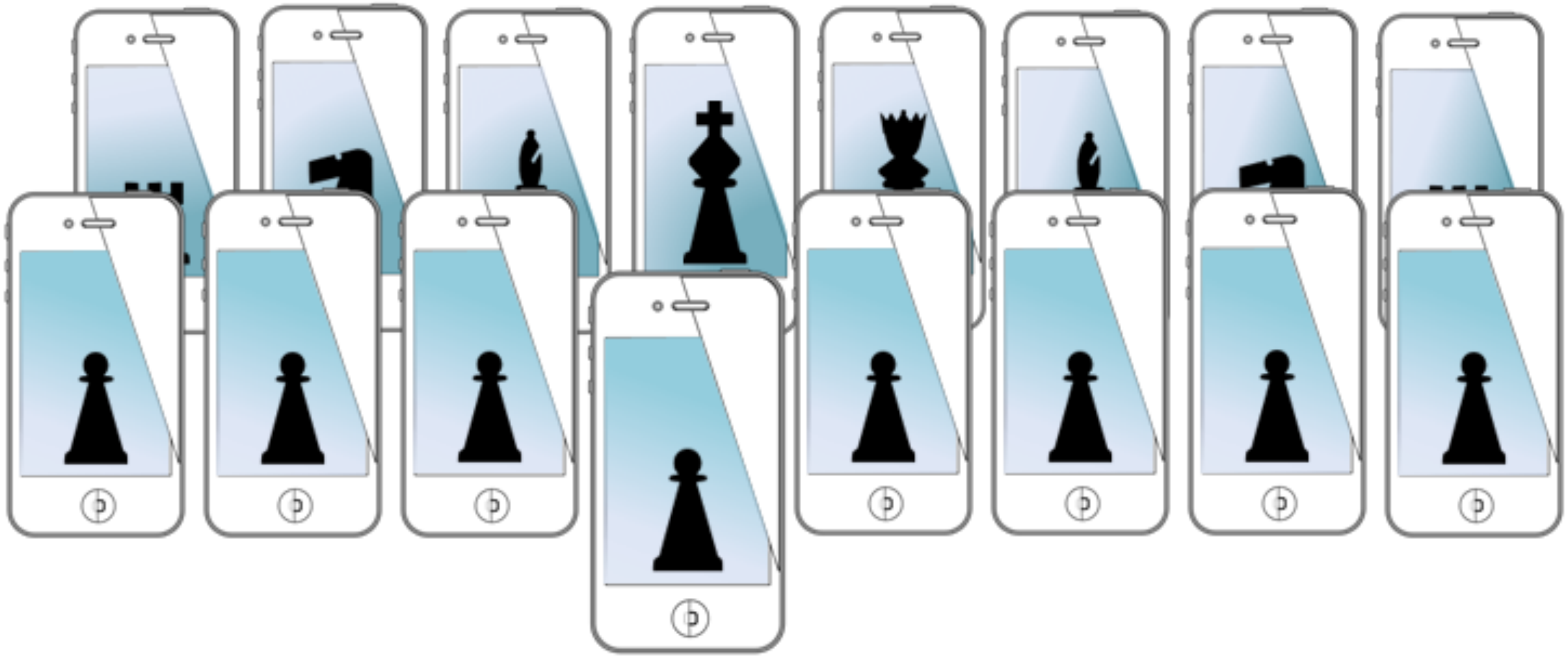
MOBILE SENSING LEARNING & CONTROL



CSE5323 & 7323

Mobile Sensing, Learning, and Control

lecture four: page controllers & core data

Eric C. Larson, Lyle School of Engineering,
Computer Science and Engineering, Southern Methodist University

course logistics

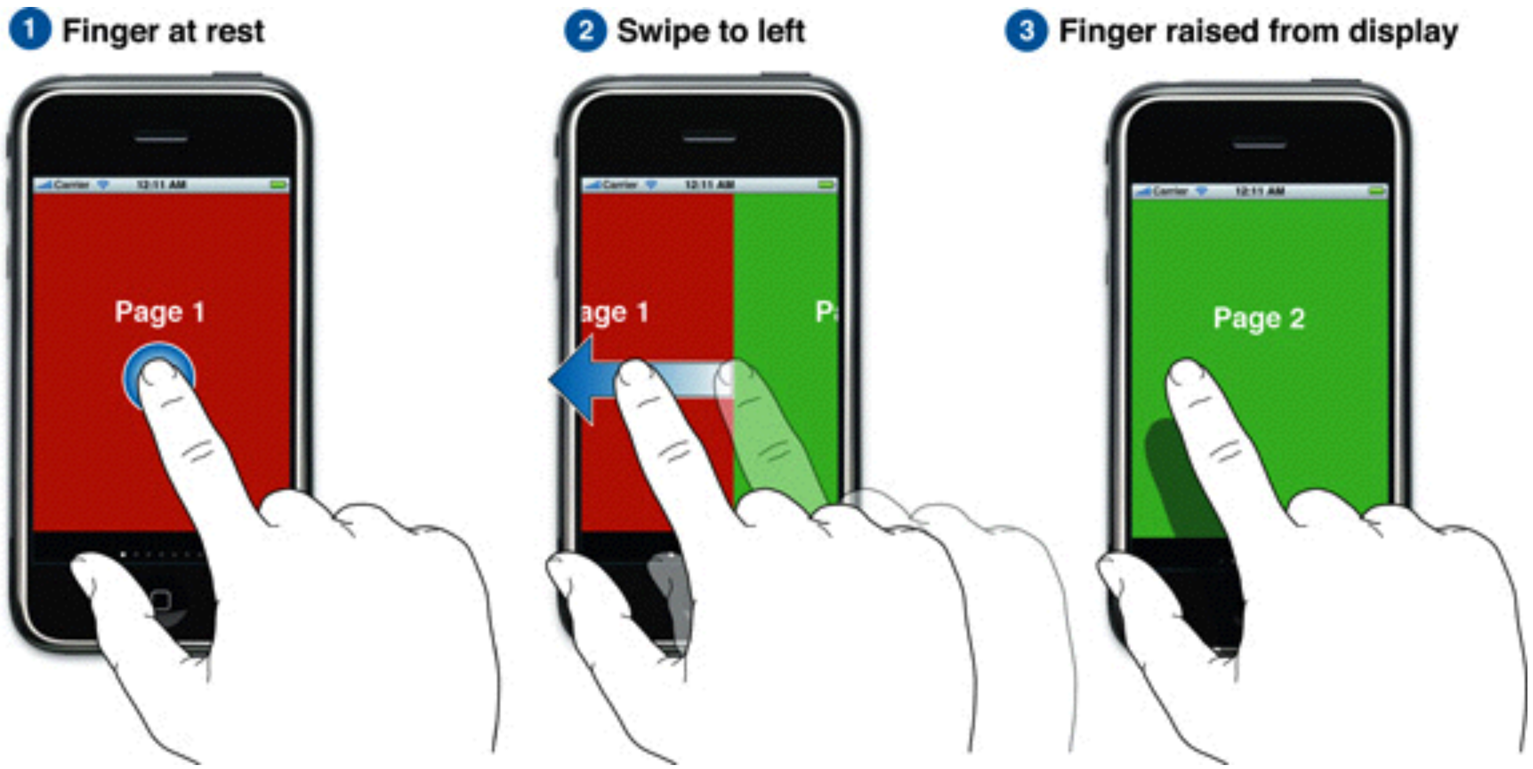
- A1 due this Friday
- I have 20 people on teams, and 2 unassigned
    - correct?

agenda

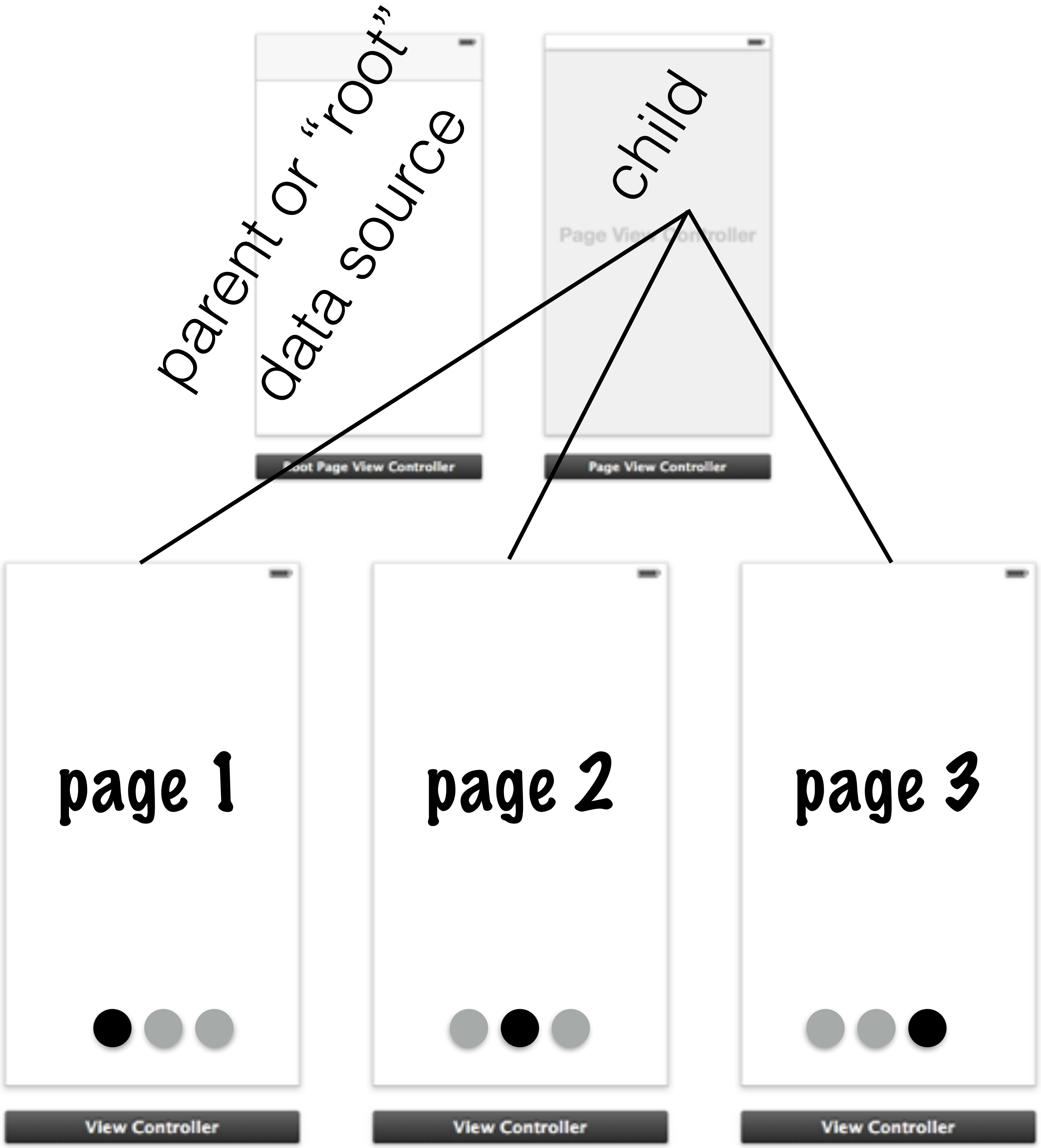
- page view controllers
- timers / segmented control
- persistent storage
    - core data for creating and using database schema
- blocks and multi-threading
- objective c++

page view controller

- place UIPageViewController in storyboard
- place a “root controller” for the page
    - adopt <UIPageViewControllerDataSource>
    - instantiate pageViewController from “root”
    - instantiate views to be paged in “root”



page view controller



parent or “root”

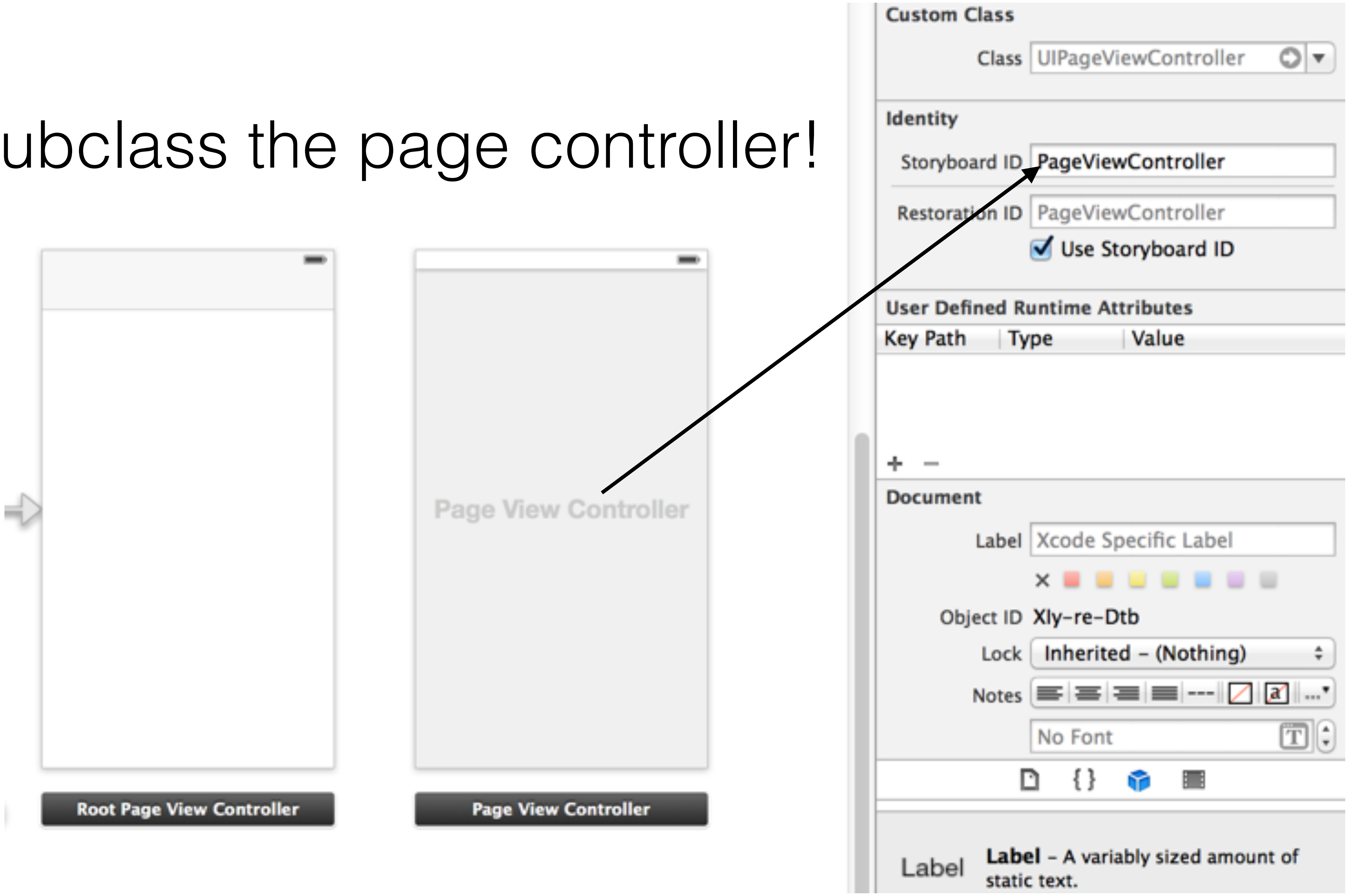
data source

child

different instantiations of view controller

page view controller

no need to subclass the page controller!



Custom Class

Class: UIPageViewController

Identity

Storyboard ID: PageViewController

Restoration ID: PageViewController

☑ Use Storyboard ID

User Defined Runtime Attributes

Key Path	Type	Value

Document

Label: Xcode Specific Label

Object ID: Xly-re-Dtb

Lock: Inherited - (Nothing)

Notes

No Font

Label **Label** - A variably sized amount of static text.

but root of the page controller must be the data source…

root page view controller

instantiation in root view controller

```
 @property (strong, nonatomic) UIPageViewController * pageViewController;
 @property (strong, nonatomic) NSArray *pageContent;

 _pageViewController = [self.storyboard instantiateViewControllerWithIdentifier:@"PageViewController"];
 _pageViewController.dataSource = self;
```

instantiate!

set first page

in viewDidLoad

```
[self.pageViewController setViewControllers:firstPageToDisplay // the page is a view controller!
                                  direction:UIPageViewControllerNavigationDirectionForward
                                   animated:NO
                                 completion:nil];

[self addChildViewController:_pageViewController];
[self.view addSubview:_pageViewController.view];
[self.pageViewController didMoveToParentViewController:self];
```

apple says do this, in order

some datasource protocol methods

```
  - (NSInteger)presentationCountForPageViewController:(UIPageViewController *)pageViewController
  {
      return [self.pageContent count];
  }

  - (NSInteger)presentationIndexForPageViewController:(UIPageViewController *)pageViewController
  {
      return 0;
  }
```

root page view controller

some datasource protocol methods (cont.)

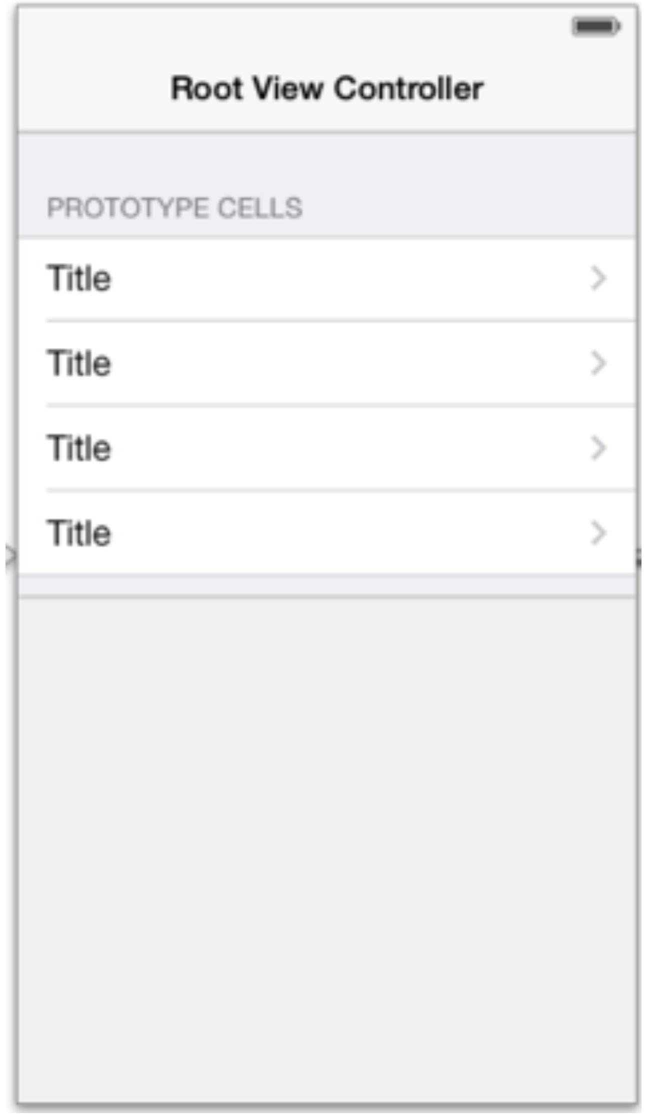
```
 - (NSInteger)presentationCountForPageViewController:(UIPageViewController *)pageViewController
 {
     return [self.pageContent count];
 }

 - (NSInteger)presentationIndexForPageViewController:(UIPageViewController *)pageViewController
 {
     return 0;
 }

-(UIViewController*)pageViewController:(UIPageViewController *)pageViewController
viewControllerBeforeViewController:(UIViewController *)viewController
{}

-(UIViewController*)pageViewController:(UIPageViewController *)pageViewController
viewControllerAfterViewController:(UIViewController *)viewController
{}
```

1. create pages (VCs)
2. set any information for loading
3. return the instantiated VC

page view demo

assignment one

- Automatic Layout (storyboard and programmatically)
- UIButtons (created in storyboard and programmatically)
- Sliders (created in storyboard and programmatically)
- Labels (created in storyboard and programmatically)
- Stepper
- Switch
- Picker (Date or otherwise)
- UINavigationController
- **UISegmentedControl**
- **NSTimer** (which should repeat and somehow update the UIView)
- UIScrollView (with scrollable, zoomable content)
- UIPageViewController
- UIImageView
- **(optional) Persistent storage via CoreData**

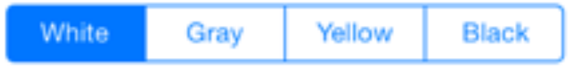


due Friday, Feb. 7

timers, segmented control

```
- (IBAction)updateFromSegmentedControl:(UISegmentedControl *)sender {

    NSString *selectedText = [sender titleForSegmentAtIndex: [sender selectedSegmentIndex]];

    YOUR_CODE
}
```

get title from control

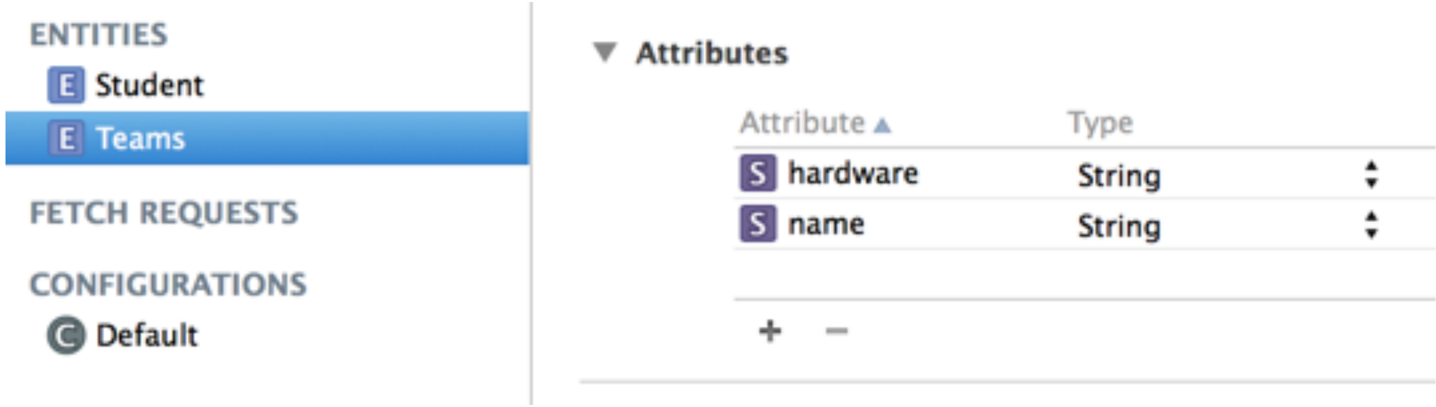


```
    NSTimer *timer = [NSTimer scheduledTimerWithTimeInterval:someIntervalInSeconds
                                                      target:self
                                                    selector:@selector(someFunction:)
                                                    userInfo:nil
                                                     repeats:YES];


    // don't get blocked by the main thread
    [[NSRunLoop mainRunLoop] addTimer:timer forMode:NSRunLoopCommonModes];
```

core data databases

- allows access to SQLite database
- integrated deeply into Xcode and into iOS
- highly optimized
- excellent for storing persistent table data
  - but usable for most anything

core data schema



ENTITIES
- E Student
- E Teams

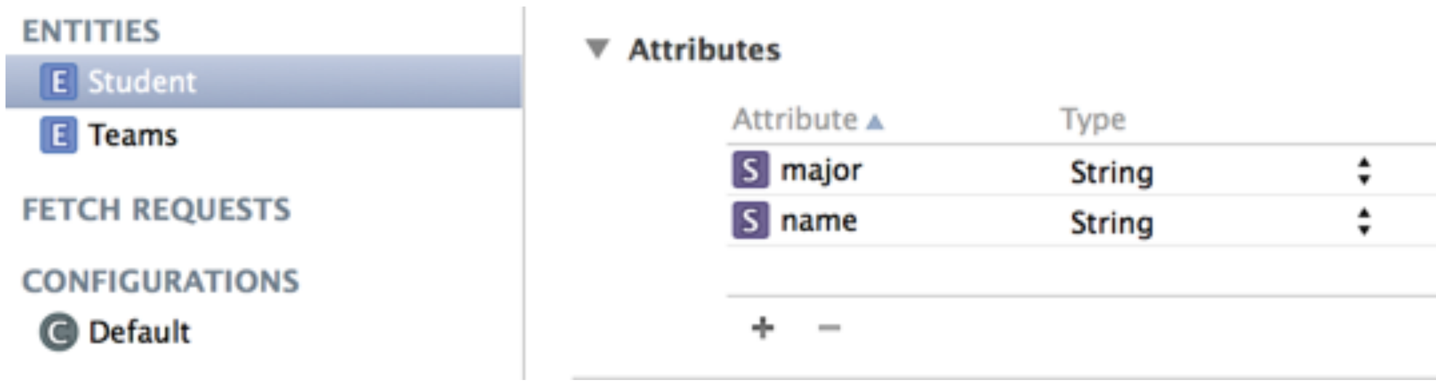
FETCH REQUESTS

CONFIGURATIONS
- C Default

▼ Attributes

Attribute ▲	Type
S hardware	String
S name	String

```
@interface Teams : NSManagedObject

@property (nonatomic, retain) NSString * name;
@property (nonatomic, retain) NSString * hardware;
@property (nonatomic, retain) NSSet *members;

@end
```



ENTITIES
- E Student
- E Teams

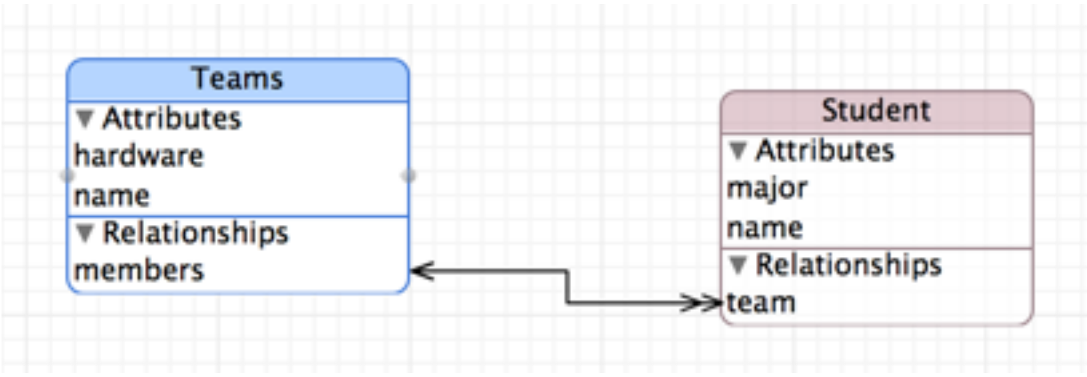
FETCH REQUESTS

CONFIGURATIONS
- C Default

▼ Attributes

Attribute ▲	Type
S major	String
S name	String

```
@interface Student : NSManagedObject

@property (nonatomic, retain) NSString * name;
@property (nonatomic, retain) NSString * major;
@property (nonatomic, retain) Teams *team;

@end
```



core data

- schema creation

  create SQLite Database on phone

  - automatic subclassing

    enable access through properties

- NSManagedObject

  bundle “data models”

- NSManagedObjectContext

  get “context” for using data model

- NSPersistentStore

  coordinate access to the data model

- NSFetchRequest

  create and execute queries

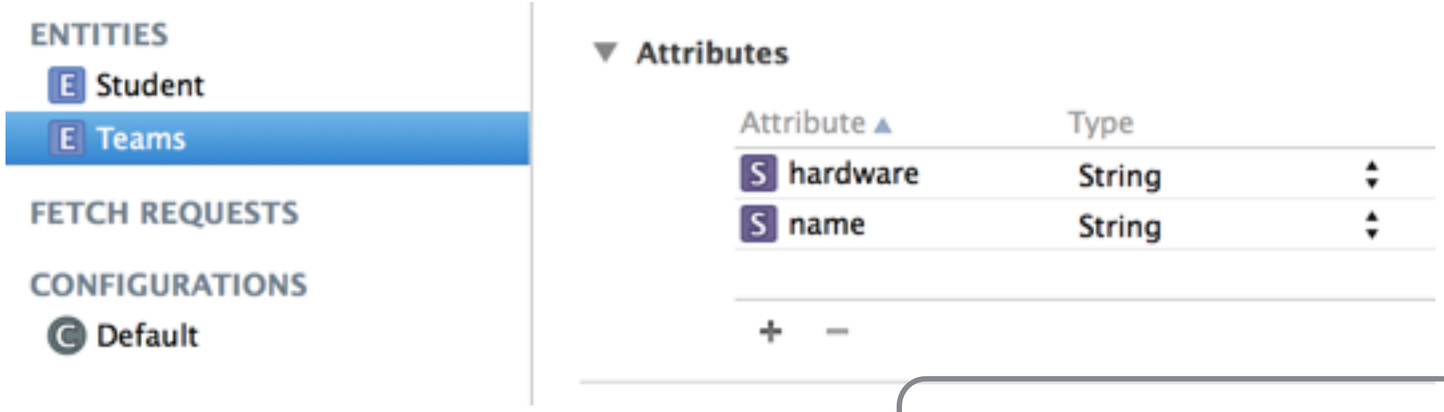
core data setup

```
// Getter for managed context
- (NSManagedObjectContext *) managedObjectContext {

    if(!_managedObjectContext){
        // create the storage coordinator
        NSPersistentStoreCoordinator *coordinator = [self persistentStoreCoordinator];
        if (coordinator != nil) {
            _managedObjectContext = [[NSManagedObjectContext alloc] init];
            [_managedObjectContext setPersistentStoreCoordinator: coordinator];
        }
    }

    return _managedObjectContext;
}

// getter for the storage coordinator
- (NSPersistentStoreCoordinator *)persistentStoreCoordinator {
    if (!_persistentStoreCoordinator) {

        // this points to our model
        NSURL *storeUrl = [NSURL fileURLWithPath: [[self applicationDocumentsDirectory]
                                                stringByAppendingPathComponent: @"ModelName.sqlite"]];
        NSError *error = nil;
        _persistentStoreCoordinator = [[NSPersistentStoreCoordinator alloc]
                                     initWithManagedObjectModel:[self managedObjectModel]];

        if(![_persistentStoreCoordinator addPersistentStoreWithType:NSSQLiteStoreType
                           configuration:nil URL:storeUrl options:nil error:&error]) {
            // exit gracefully if you need the database to function in the UI
        }
    }
    return _persistentStoreCoordinator;
}
```

core data setup

```
// getter for the storage coordinator
- (NSPersistentStoreCoordinator *)persistentStoreCoordinator {
    if (!_persistentStoreCoordinator) {

        // this points to our model
        NSURL *storeUrl = [NSURL fileURLWithPath: [[self applicationDocumentsDirectory]
                                                   stringByAppendingPathComponent: @"ModelName.sqlite"]];
        NSError *error = nil;
        _persistentStoreCoordinator = [[NSPersistentStoreCoordinator alloc]
                                       initWithManagedObjectModel:[self managedObjectModel]];

        if(![_persistentStoreCoordinator addPersistentStoreWithType:NSSQLiteStoreType
                             configuration:nil URL:storeUrl options:nil error:&error]) {
            // exit gracefully if you need the database to function in the UI
        }
    }
    return _persistentStoreCoordinator;
}


// getter for the object model, create if needed
- (NSManagedObjectModel *)managedObjectModel {
    if (!_managedObjectModel) {
        _managedObjectModel = [NSManagedObjectModel mergedModelFromBundles:nil];
    }
    return _managedObjectModel;
}
```

entering data



ENTITIES
- Student
- Teams

FETCH REQUESTS

CONFIGURATIONS
- Default

Attributes

Attribute	Type
hardware	String
name	String

create a new entity from model

```
//  get a new entry
 team = [NSEntityDescription insertNewObjectForEntityForName:@"Teams"
                                         inManagedObjectContext:self.managedObjectContext];
 // save the attributes
 team.name = self.teamNameTextField.text;
 team.hardware = [self assignHardware];

 //  save into the database
 NSError *error;
 if (![self.managedObjectContext save:&error]) {
     NSLog(@"save database failed: %@", [error localizedDescription]);
 }
```

set attributes

not saved in database until here

queries in core data

```
-(NSArray*)getAllTeamsFromDatabase
{
    // initializing NSFetchRequest
    NSFetchRequest *fetchRequest = [[NSFetchRequest alloc] init];

    //Setting Entity to be Queried
    NSEntityDescription *entity = [NSEntityDescription entityForName:@"Teams"
                                              inManagedObjectContext:self.managedObjectContext];
    [fetchRequest setEntity:entity];
    NSError* error;

    // Query on managedObjectContext With Generated fetchRequest
    NSArray *fetchedRecords = [self.managedObjectContext executeFetchRequest:fetchRequest error:&error];

    // Returning Fetched Records
    return fetchedRecords;
}
```

request

entity to request from

fetch

array of results, even if size=0

```
-(NSArray*)getTeamFromDatabase:(NSString*)teamName
{
    // initializing NSFetchRequest
    …

   fetchRequest.predicate =
        [NSPredicate predicateWithFormat:@"name = %@",teamName];

    …

    // Returning Fetched Records
    return [self.managedObjectContext executeFetchRequest:fetchRequest error:&error];
}
```

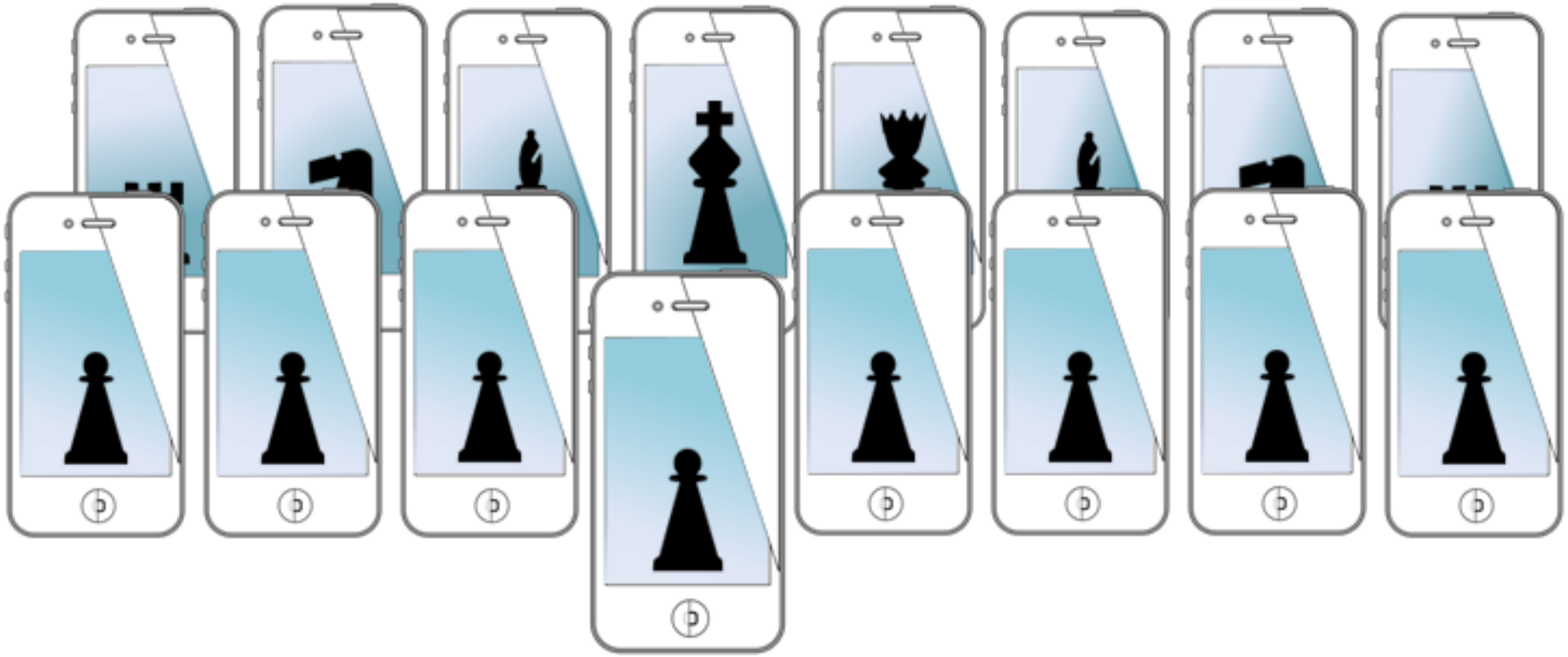
set predicate

```
@"name = %@“
@"name contains[c] %@“
@"value > 7“
@“team.name = %@“
@“any student.name contains %@“
```

core data demo

- Who Was In That!
- Class Teams! will make available on website

MOBILE SENSING LEARNING & CONTROL



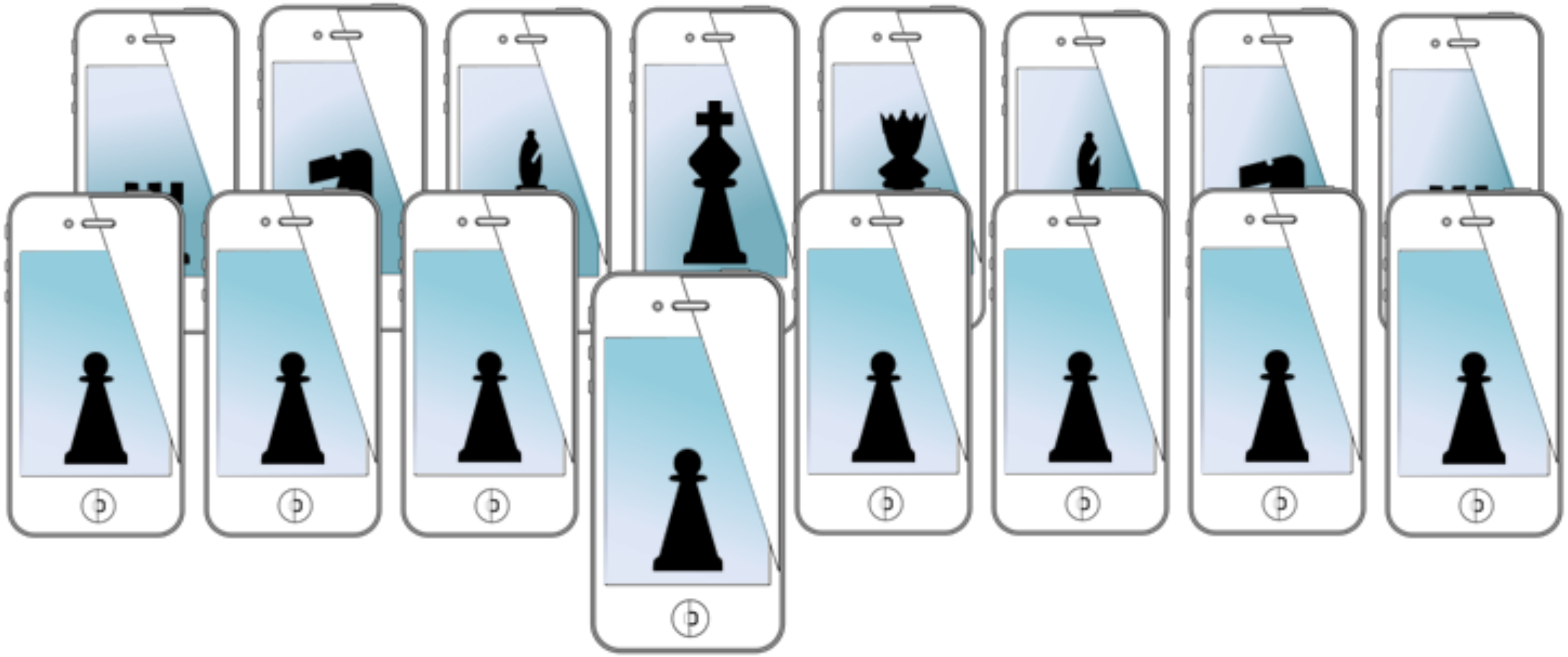
CSE5323 & 7323

Mobile Sensing, Learning, and Control

lecture four: page controllers & core data

Eric C. Larson, Lyle School of Engineering,
Computer Science and Engineering, Southern Methodist University

MOBILE SENSING LEARNING & CONTROL



CSE5323 & 7323

Mobile Sensing, Learning, and Control

lecture five: threads, blocks, c++, audio session

Eric C. Larson, Lyle School of Engineering,
Computer Science and Engineering, Southern Methodist University

blocks

- not callback functions (but similar)
  - created at runtime
  - can access data from scope when defined
  - syntax is ^(…)

block syntax

return type

block name

param types

```
 // create a block on the fly
  float (^onTheFlyBlockThatAddsTwoInts)(int,int); // declare the block, try not to make unclear
  // define the behavior of the block
  onTheFlyBlockThatAddsTwoInts =^(int a, int b){
      return (float)(a+b);
  };
  // use the block
  NSLog(@" On the fly value: %.4f",onTheFlyBlockThatAddsTwoInts(5,6));
```

define code that will execute

```
typedef float(^TypeDefinedBlock)(float,float);

TypeDefinedBlock blockAsObject = ^(float arg1, float arg2){
     return arg1 / arg2;
};
```

type define, more like callback

```
     //------------------------------
     //execute the block from typedef
     float value = blockAsObject(22.0,44.0);
     NSLog(@" Val = %.4f",value);
```

syntax to call block

```
     //------------------------------
     //enumerate an Array with a block
     NSArray *myArray = @[@34.5,@56.4567,@(M_PI)];

     // here the block is created on the fly for the enumeration
     [myArray enumerateObjectsUsingBlock:^(NSNumber *obj, NSUInteger idx, BOOL *stop) {
         // print the value of the NSNumber in a variety of ways
         NSLog(@"Float Value = %.2f, Int Value = %d",[obj floatValue],[obj integerValue]);
     }];
```

enumerate with block

concurrency in iOS

- grand central dispatch (GCD) handles all operations
    - GCD looks at “queues” of **blocks** that need to be run
    - GCD and the Xcode compiler work deep inside the OS, actually in the kernel — they are optimized
    - for a **serial queue** each block is run sequentially
    - for **concurrent queues** the first block is dequeued
        - if CPU is available, then the next block is also dequeued, but could finish any time
- the main queue handles all UI operations (and no other queue can generate UI changes)
    - so, no updating of the views, labels, buttons, (image views*) except from the main queue

queue syntax

```
// using c code:
 dispatch_queue_t someQueue = dispatch_queue_create("myCreatedQueue", NULL);
 dispatch_async(someQueue, ^{
     // your code to execute
     for(int i=0;i<3;i++)
         NSLog(@"I am being executed from a dispatched queue");

     // now I need to set something in the UI, but I am not in the main thread!
     // call from main thread
     dispatch_async(dispatch_get_main_queue(), ^{
         self.label.text = [NSString stringWithFormat:@"Finished running %d times, Safe",3];
     });


 }); // this operation adds the block to the queue in a single clock cycle, then returns
```

define block

create new queue

update UI, main thread

```
NSOperationQueue *newQueue = [[NSOperationQueue alloc] init];
    newQueue.name = @"ObjCQueue";
    [newQueue addOperationWithBlock:^{
        // your code to execute
        for(int i=0;i<3;i++)
            NSLog(@"I am being executed from a dispatched queue, from objective-c");

        // now I need to set something in the UI, but I am not in the main thread!
        // call from main thread
        [self performSelectorOnMainThread:@selector(setMyLabel)
                                   withObject:nil
                                waitUntilDone:NO];


    }];
```

create new queue

define block

update UI, main thread

objective c++

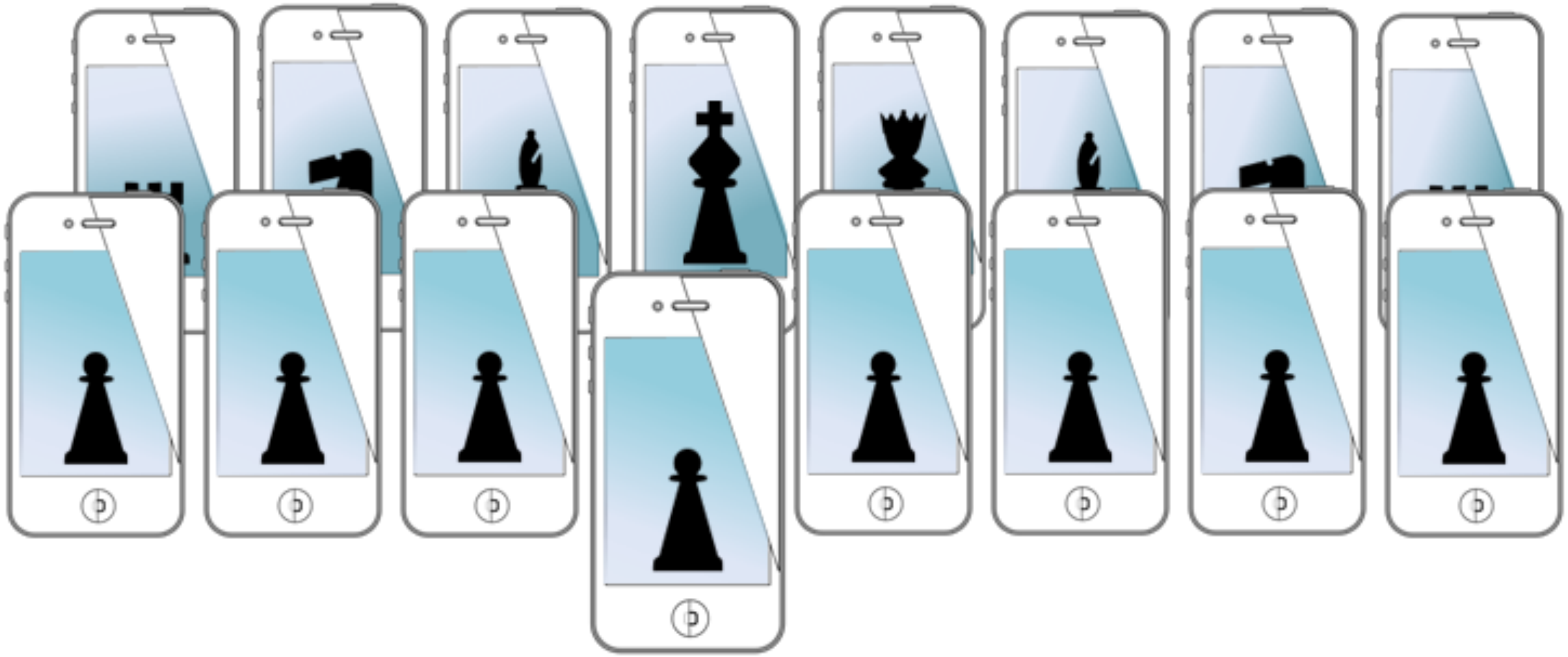
- actually, its just c++
- …but need to tell compiler we are using c++

- Add any #include statements
- Change extensions to .mm where you use c++ class(es)
- ARC won’t help you, so explicitly call dealloc and your class destructor

for next time…

- core audio
    - novocaine, audio session setup
    - playing songs
    - getting samples from microphone
    - showing samples with OpenGL

MOBILE SENSING LEARNING & CONTROL



CSE5323 & 7323

Mobile Sensing, Learning, and Control

lecture five: threads, blocks, c++, audio session

Eric C. Larson, Lyle School of Engineering,
Computer Science and Engineering, Southern Methodist University