iOS: XCode and Interface Builder

Tommy MacWilliam

`tmacwilliam@cs.harvard.edu`

March 7, 2012

Today

- ▶ Objective-C Review
- ▶ XCode
- ▶ Outlets and Actions
- ▶ Delegates and Protocols
- ▶ Tic-Tac-Toe

Classes

iOS: XCode
and Interface
Builder

Tommy
MacWilliam

Objective-C
Review

XCode

Interface
Builder

Outlets and
Actions

Delegates and
Protocols

Tic-Tac-Toe

- ▶ @interface: .h files
    - ▶ declare class methods and properties
    - ▶ @class Something to use other classes
- ▶ @implementation: .m files
    - ▶ define class methods and synthesize properties
    - ▶ #import "Something.h" to use other classes

Classes

iOS: XCode
and Interface
Builder

Tommy
MacWilliam

Objective-C
Review

XCode

Interface
Builder

Outlets and
Actions

Delegates and
Protocols

Tic-Tac-Toe

- −: instance methods, sent to objects (e.g. `init`)
- +: class methods, sent to class, not object (e.g. `alloc`)

Properties

▶ `@property (nonatomic, assign) int foo;`
  ▶ declare new property, specify how values are assigned
▶ `@synthesize foo = _foo;`
  ▶ generate getters/setters for property

Message Passing

- ‒ (id)init
- ‒ (id)initWithName:(NSString*)name
- ‒ (id)initWithName:(NSString*)name
  age:(int)age
- [object initWithName:@"name"];

GDB

- ► create breakpoint by clicking on line number
- ► `next`: continue line-by-line
- ► `step`: go into a called function
- ► `continue`: resume execution until next breakpoint
- ► `print` or mouse over variable to see value

Objective-C and GDB

iOS: XCode
and Interface
Builder

Tommy
MacWilliam

Objective-C
Review

XCode

Interface
Builder

Outlets and
Actions

Delegates and
Protocols

Tic-Tac-Toe

- GDB

- ▶ File Inspector: nib options
- ▶ Quick Help: view documentation for the selected `UIView`
- ▶ Identity Inspector: modify object properties
- ▶ Attributes Inspector: customize the selected `UIView`
- ▶ Size Inspector: change placement/size of `UIView`
- ▶ Connections Inspector: manage outlets/actions

Interface Builder

- ▶ File's Owner: object that loads nib
- ▶ First Responder: `UIView` currently in focus and responding to input
- ▶ App Delegate: responsible for loading, resuming, etc.
- ▶ Window: container for `UIView`

Outlets

iOS: XCode
and Interface
Builder

Tommy
MacWilliam

Objective-C
Review

XCode

Interface
Builder

Outlets and
Actions

Delegates and
Protocols

Tic-Tac-Toe

- ► `IBOutlet` tells XCode the object represents something in IB
  - ► NOT the type of the object, just a flag
  - ► `@property (nonatomic, strong) IBOutlet UIButton* button;`

- define `UIView` properties in `@interface`
- synthesize `UIView` properties in `@implementation`
- connect Objects in Interface Builder

Connecting Outlets

- select placeholder where property is defined (File's Owner or App Delegate)
- Ctrl-Click and drag to `UIView`
- select name of outlet
- ???
- profit!

Actions

► `IBAction` tells XCode the method is used by something in IB

  ► equivalent to the `void` type, so all methods are `void`

Connecting Actions

- select `UIView` to fire callback for, open Connections Inspector
- Ctrl-Click callback, drag to placeholder where method is defined (File's Owner or App Delegate)
- select name of method
- ???
- profit!

Outlets and Actions

- `TextFieldExample`

- define object to handle actions
- delegate object implements a protocol
  - guarantees definitions for methods exist

Protocol

```
@protocol SomeProtocol
– (void)something
– (int)calculateSomething:(int)
@end
```

UITextFieldDelegate

iOS: XCode
and Interface
Builder

Tommy
MacWilliam

Objective-C
Review

XCode

Interface
Builder

Outlets and
Actions

Delegates and
Protocols

Tic-Tac-Toe

- ▶ `textFieldShouldReturn`: "done" button pressed
- ▶ `textFieldShouldBeginEditing`: user about to edit text
- ▶ `textFieldShouldEndEditing`: text field about to lose focus

Using Protocols and Delegates

iOS: XCode
and Interface
Builder

Tommy
MacWilliam

Objective-C
Review

XCode

Interface
Builder

Outlets and
Actions

Delegates and
Protocols

Tic-Tac-Toe

▶ `BetterTextFieldExample`

▶ `UtilityApp`

Putting it All Together

- ► `TicTacToe`
  - ► our first of many award-winning apps
  - ► follow along!