Mobile Development

FI MUNI - iOS Basics

Petr Dvořák
Partner & Mobile Strategy Consultant
@joshis_tweets

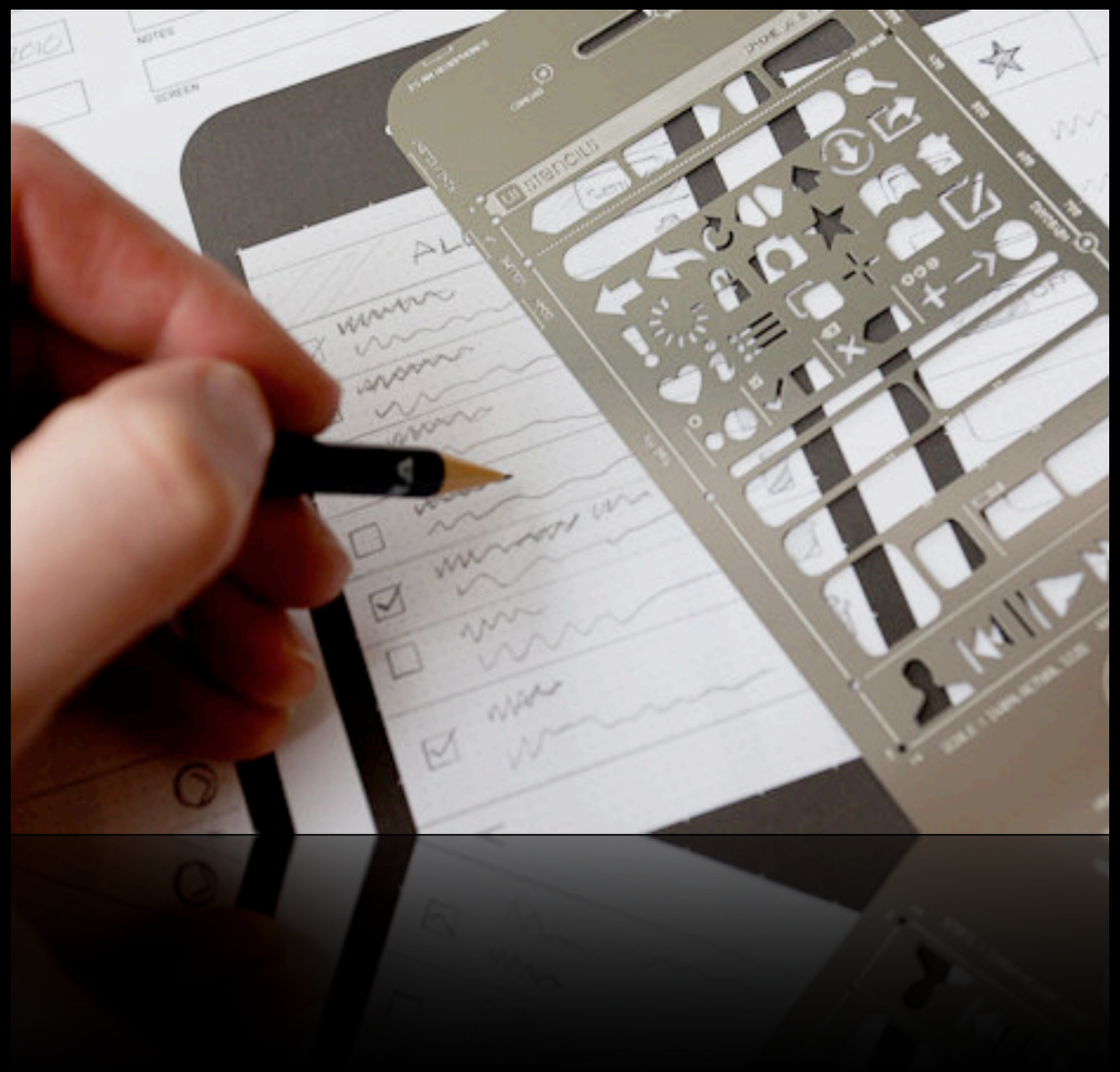
Mobile apps

2008 iPhone

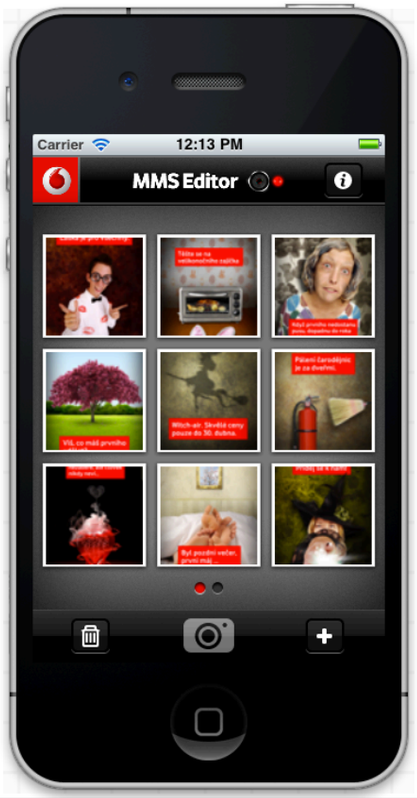
2010 iPad

Mobile apps

- Not just “smaller desktop” apps
- UI Optimization needed
- Create a “mission statement”
- Kill some feature bullet points
- Design guidelines available



Paper, Pencil & Stencils



1. Mockups **reproduces the experience of sketching** interfaces on a whiteboard, but using your computer, so they're easier to share, modify, and get honest feedback on.

2. Wireframes made with Mockups look like sketches, so **stakeholders won't get distracted by little details**, and can focus on what's really important instead.

Balsamiq Mockups

Design must be great!

... otherwise...

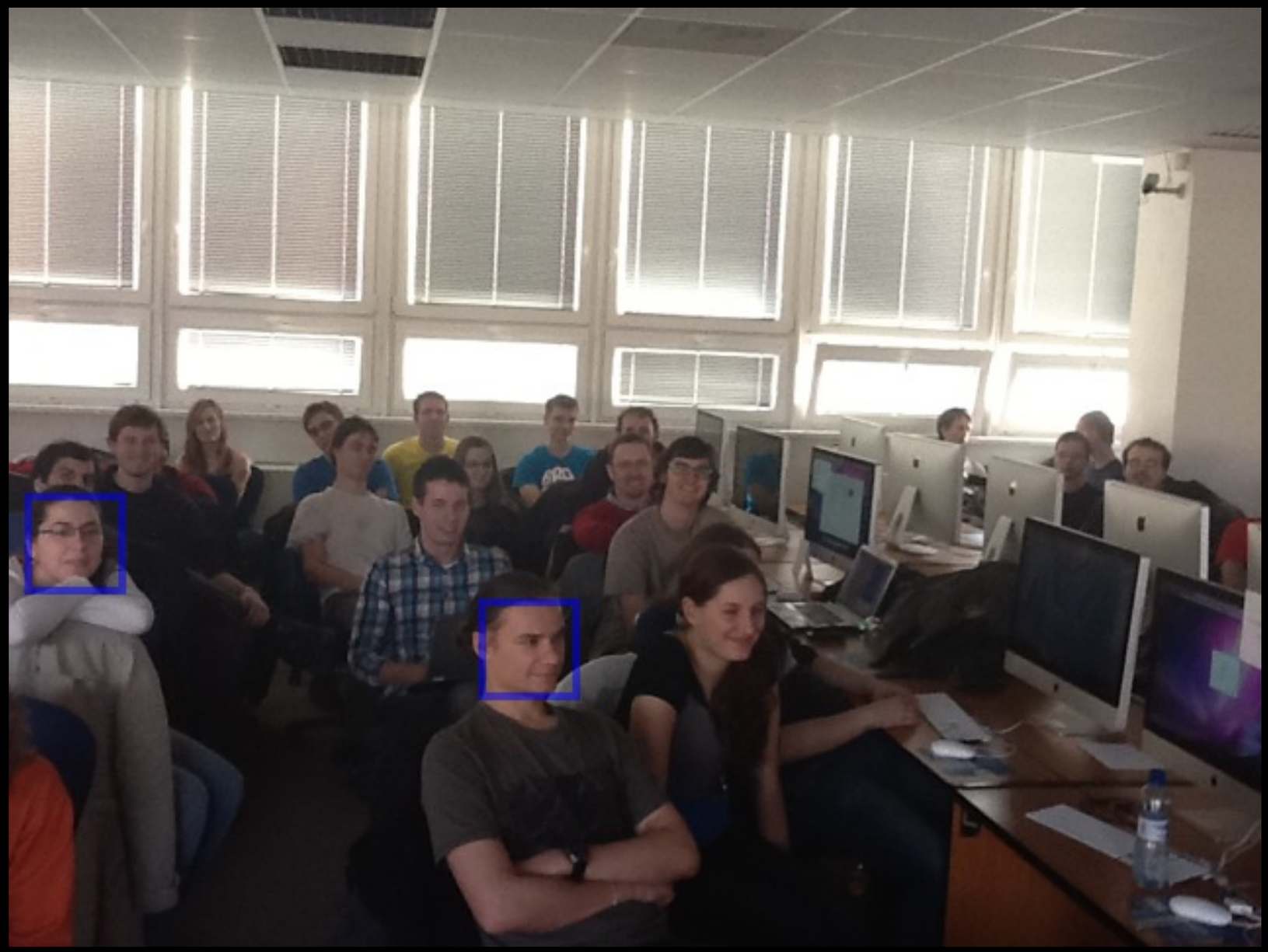
Mobile is different

Advanced stuff

- Augmented reality
  - Qualcom AR SDK for iOS
    - https://developer.qualcomm.com/develop/mobile-technologies/augmented-reality
- Face recognition
  - OpenCV library built for iOS
  - http://opencv.willowgarage.com

Augmented reality

Face Recognition



MUNI 2012

Success Stories

Slevolapka

Top Free iPhone Utilities Apps

Sort By: Bestsellers

1. **Find My iPhone**
Utilities
Updated 19 Sep…
FREE

2. **iPumpuj**
Utilities
Updated 06 Octob…
FREE

3. **QR Reader for iP…**
Utilities
Updated 24 May 2…
FREE

4. **Bosch Toolbox: s…**
Utilities
Updated 14 Septe…
FREE

5. **Chrome**
Utilities
Updated 24 September…
FREE

6. **Battery Doctor (B…**
Utilities
Updated 03 Octob…
FREE

7. **LED Light for iPh…**
Utilities
Updated 25 Februa…
FREE

8. **Speedtest.net Mo…**
Utilities
Updated 22 Novem…
FREE

9. **DSL.cz - měření r…**
Utilities
Updated 19 Septe…
FREE

10. **Emoji Free!**
Utilities
Updated 20 December …
FREE

11. **Scan**
Utilities
Updated 26 June 2…
FREE

12. **Seznam.cz QR čt…**
Utilities
Updated 23 August…
FREE

13. **Barcode Reader f…**
Utilities
Updated 03 Septe…
FREE

14. **iZip - Zip Unzip U…**
Utilities
Updated 18 Septe…
FREE

15. **Samsung Remote**
Utilities
Updated 20 November …
FREE

16. **Secret Key**
Utilities
Released 07 Augu…
FREE

17. **Core Monitor**
Utilities
Updated 07 Februa…
FREE

18. **Magnifier**
Utilities
Updated 02 Septe…
FREE

19. **Video Downloade…**
Utilities
Updated 20 Septe…
FREE

20. **Decibels**
Utilities
Released 29 July 2010
FREE

21. **AirPort Utility**
Utilities
Updated 19 Septe…
FREE

22. **Fing - Network Sc…**
Utilities
Updated 11 August…
FREE

23. **Emoji++**
Utilities
Updated 09 August…
FREE

24. **WinZip**
Utilities
Updated 04 May 2…
FREE

25. **Flashlight.**
Utilities
Updated 01 May 2012
FREE

26. **DS download**

27. **iHandy Level Free**

28. **Battery HD+**

29. **Best Flash Light!**

30. **Convert Units for Fre…**

iOS Ecosystem

Required hardware

- Any computer/laptop with Mac OS X
- Mac Mini - from 13 990 CZK
- MacBook Pro 13” - from 32 490 CZK

iOS Developer Account

- Bound to Apple ID
  - Registration is free
    - XCode/SDK download is free
      - but it offers development for iOS simulator only

iOS Developer Account

- iOS Developer Program - $99 / year
- Installation on devices
- App Store publishing
- Support

Member Center

- A dashboard website, a quick pointer
  - Dev Centers
  - Provisioning Portal
  - iTunes Connect
  - Resource Center, Developer Support, Forums

Developer Center

- Separate dev centers for iOS, Mac OS and Web development
- A place to find
  - Resources, videos, tutorials, docs, ...
  - Early access downloads

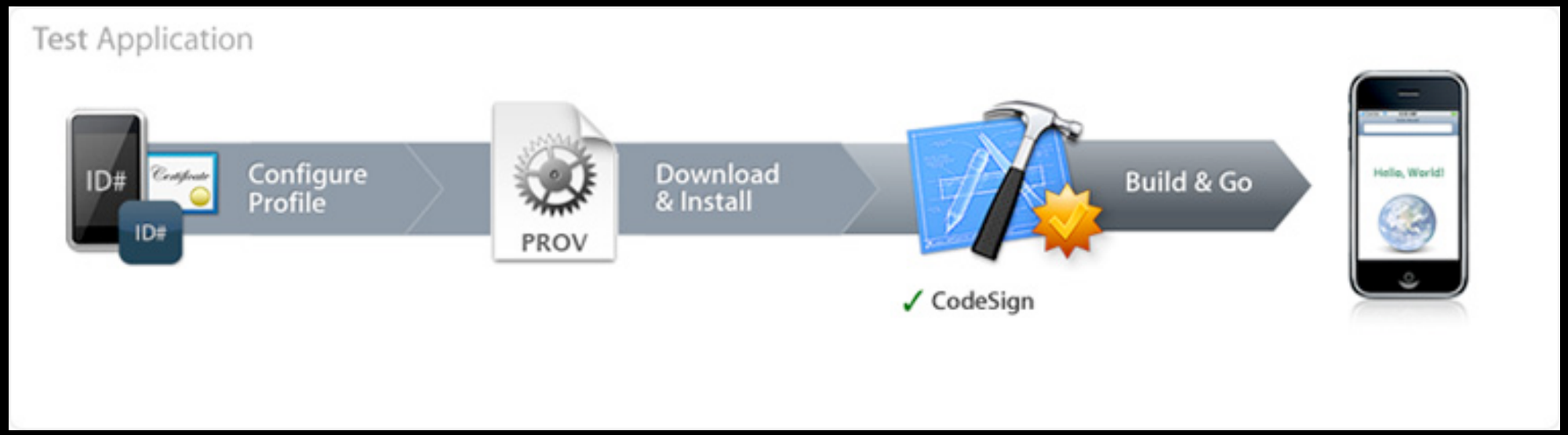
Provisioning portal

- Register your development devices
  - max. 100 iOS devices / year
  - one-time removal
- Manage certificates
- Register AppID
- Create a provisioning profile

Provisioning profile

- Development / Distribution profiles
- A composite entity that contains
  - Certificates
  - AppID
  - Devices

Provisioning Process



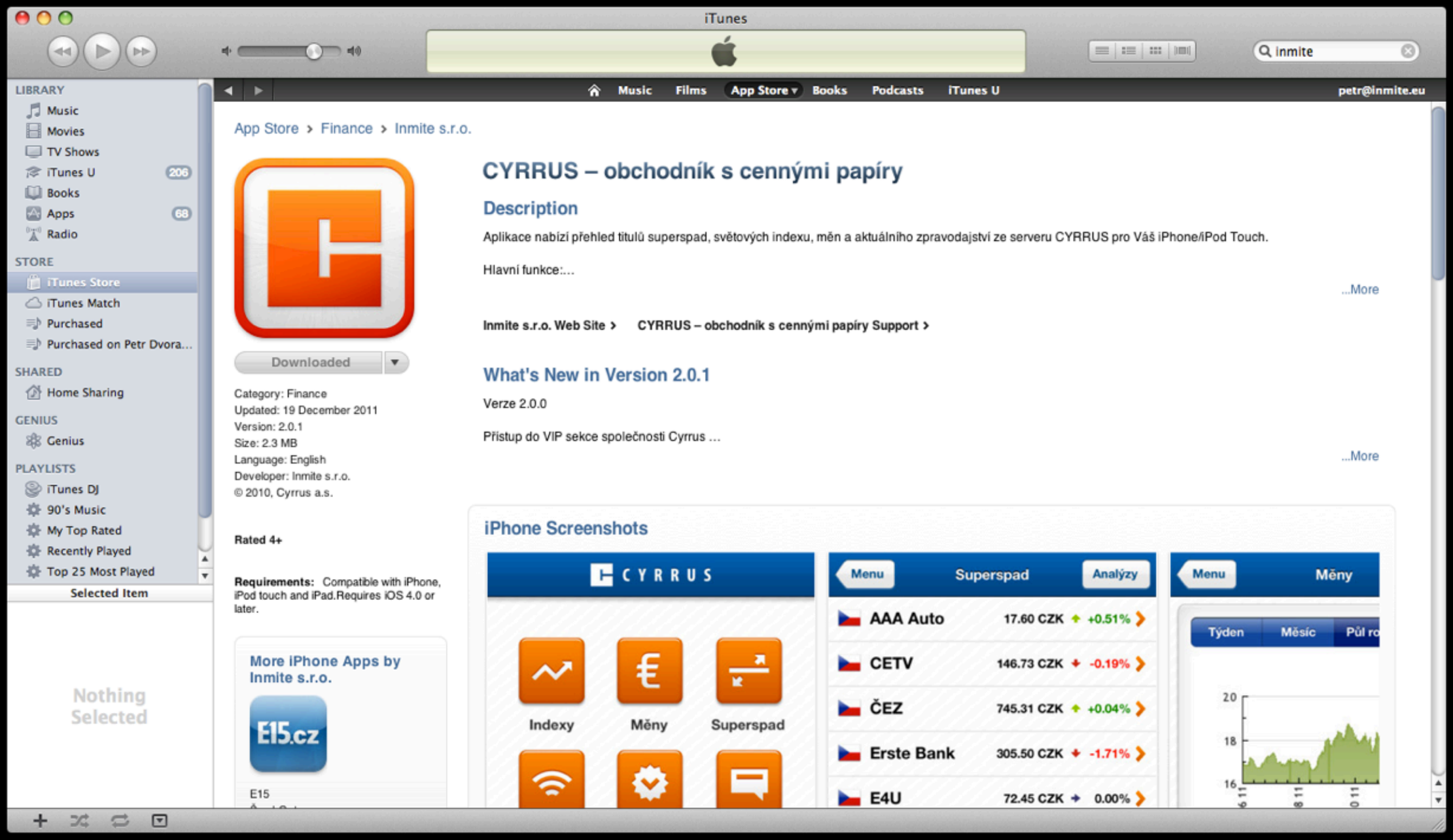
iTunes Connect

- Manage applications
- Manage legal contracts
- Sales statistics
- Financial Reports, User Management, Contact

Member Center

- A dashboard that points to
  - Dev Centers ~ Learn & Do
  - Provisioning Portal ~ Test
  - iTunes Connect ~ Distribute
  - Resource Center, Developer Support, Forums

App Store


iTunes
inmite
Music
Films
App Store
Books
Podcasts
iTunes U
petr@inmite.eu
LIBRARY
Music
Movies
TV Shows
iTunes U 206
Books
Apps 68
Radio
STORE
iTunes Store
iTunes Match
Purchased
Purchased on Petr Dvora...
SHARED
Home Sharing
GENIUS
Genius
PLAYLISTS
iTunes DJ
90's Music
My Top Rated
Recently Played
Top 25 Most Played
Selected Item
Nothing Selected
App Store › Finance › Inmite s.r.o.
CYRRUS – obchodník s cennými papíry
Description
Aplikace nabízí přehled titulů superspad, světových indexu, měn a aktuálního zpravodajství ze serveru CYRRUS pro Váš iPhone/iPod Touch.
Hlavní funkce:...
...More
Inmite s.r.o. Web Site ›
CYRRUS – obchodník s cennými papíry Support ›
Downloaded
Category: Finance
Updated: 19 December 2011
Version: 2.0.1
Size: 2.3 MB
Language: English
Developer: Inmite s.r.o.
© 2010, Cyrrus a.s.
Rated 4+
Requirements: Compatible with iPhone, iPod touch and iPad.Requires iOS 4.0 or later.
More iPhone Apps by Inmite s.r.o.
E15.cz
E15
What's New in Version 2.0.1
Verze 2.0.0
Přístup do VIP sekce společnosti Cyrrus ...
...More
iPhone Screenshots
CYRRUS
Indexy
Měny
Superspad
Menu
Superspad
Analýzy
AAA Auto 17.60 CZK +0.51%
CETV 146.73 CZK -0.19%
ČEZ 745.31 CZK +0.04%
Erste Bank 305.50 CZK -1.71%
E4U 72.45 CZK 0.00%
Menu
Měny
Týden
Měsíc
20
18
16

iOS Overview

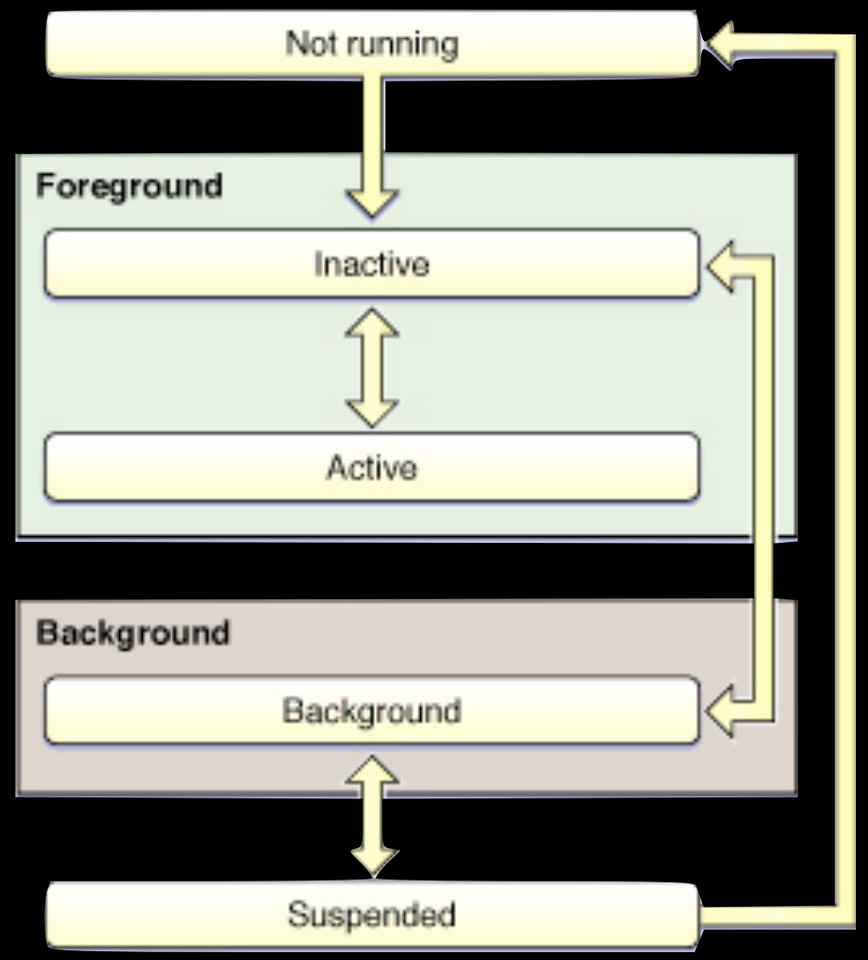
- Unix based, high performance
- Strict memory management
- Multi-tasking since iOS 4
- Apps are sandboxed
- Specific features accessed via public iOS SDK APIs

Memory management

- Application receives memory warnings from the OS
- Must react on it quickly
  - ... free up as much memory as possible as quickly as possible
- ... otherwise, it's killed

Multi-tasking

- Application transitions among states
  - Not Running
  - Inactive
  - Active
  - Background
  - Suspended



User's perspective

- Multitasking is transparent
- "List of last used apps"
- ~~"List of running apps"~~
- Default.png should resemble the first app screen

Multi-tasking

- Apps may remain alive
  - audio
  - voip
  - location
  - newsstand
  - external / bluetooth accessory

Application sandbox

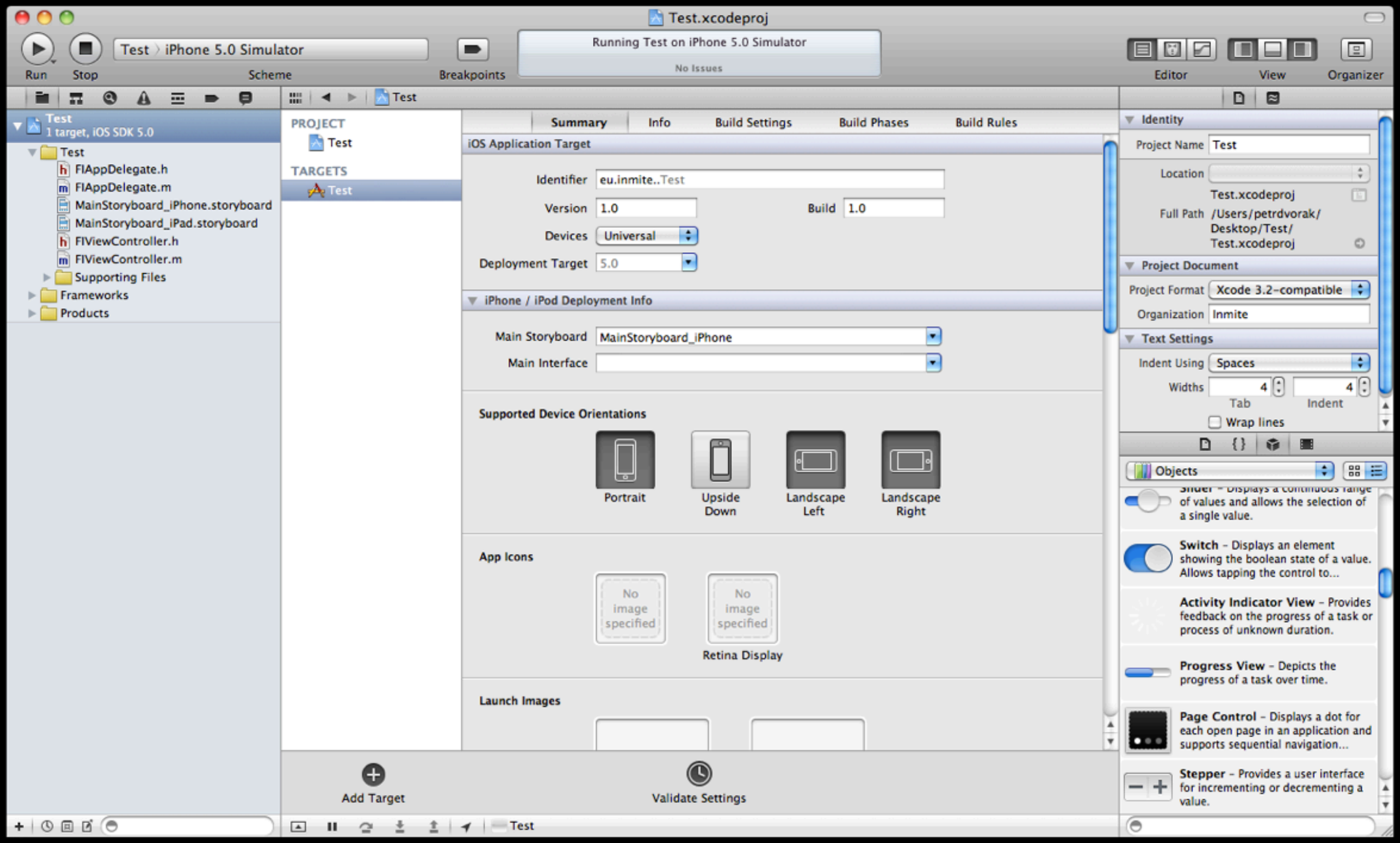
- Every application has a home folder
  - Documents folder - is backed up
  - Cache folder - isn’t backed up
  - tmp folder

Development Tools

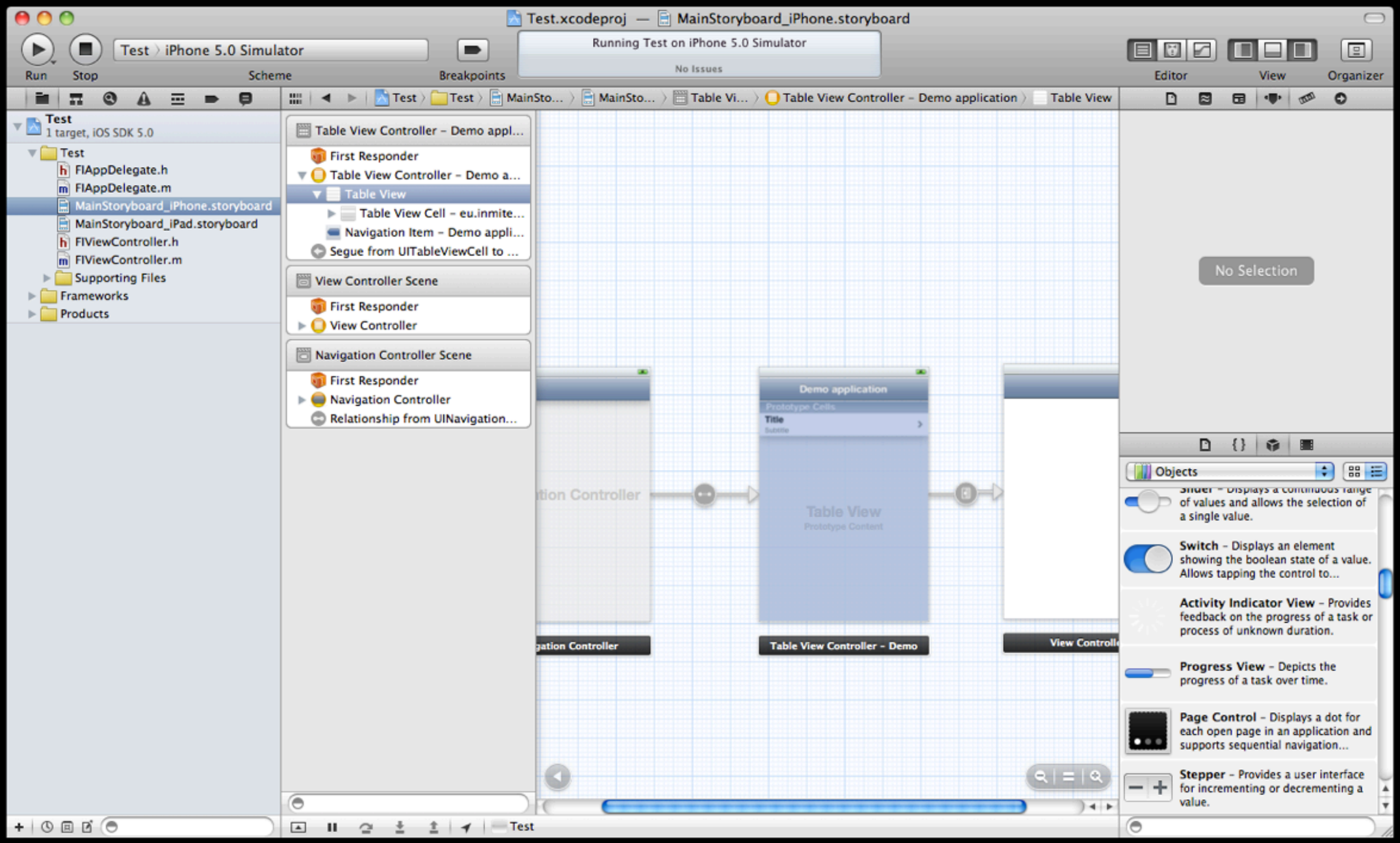
XCode

- IDE used for Mac / iOS development
- Tight git VCS integration
- LLVM / GCC compiler
- App Store submit, ad-hoc archives
- Distributed builds

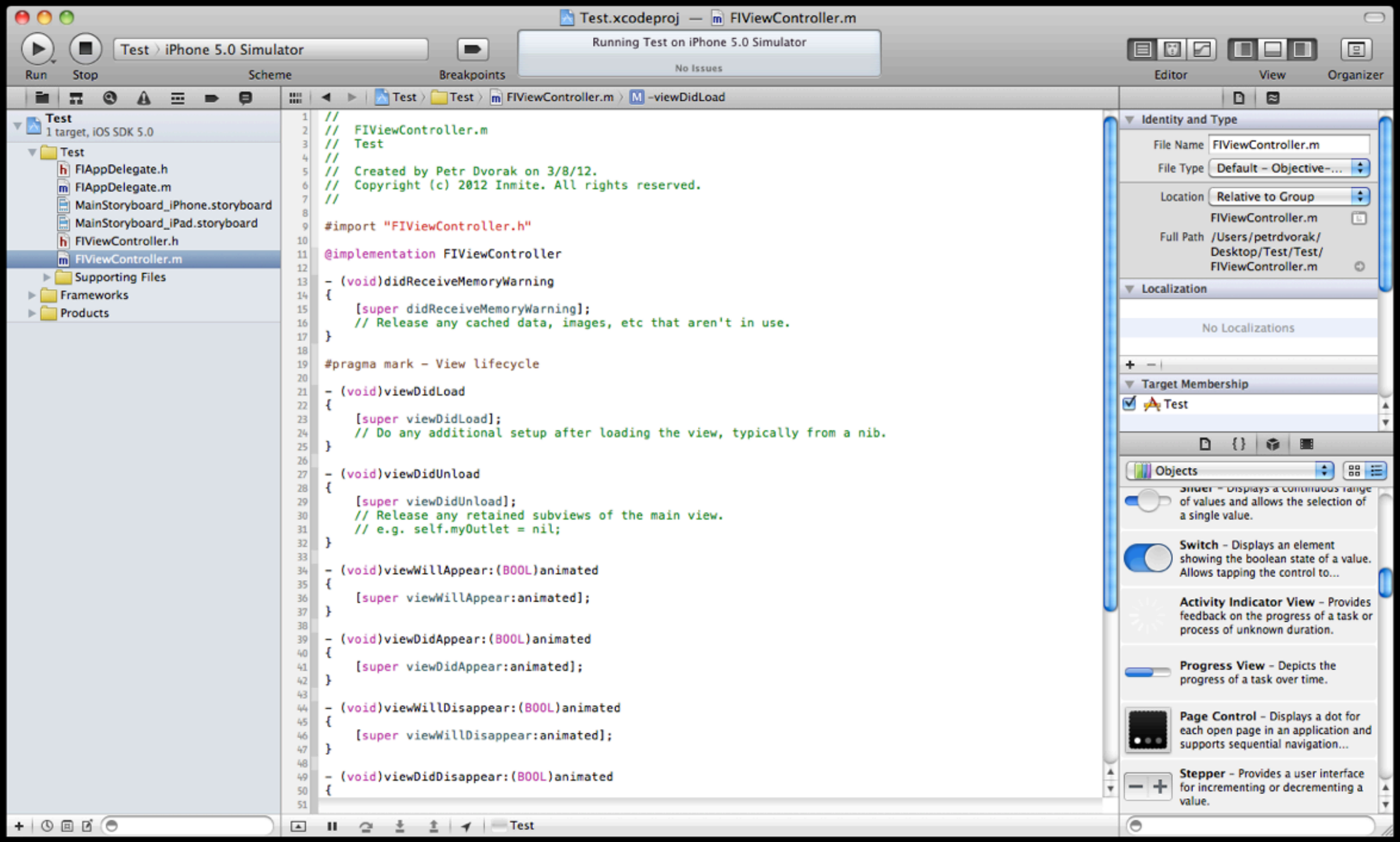
XCode - Project/Target



XCode - Storyboarding



XCode - Source Code



Instruments

- Set of visual debugging tools
  - Memory leaks / Zombie objects
  - CPU / Activity monitoring
  - Quartz performance
  - OpenGL ES performance

iPhone/iPad Simulator

- Almost like a real device
- Intel instruction set
- Inherits computer CPU and memory
- Limited set of device specific features
  - no push, no App Store, no phone calls, no accelerometer, ...

Objective-C & iOS SDK

Objective-C

- Object oriented language
- Derived from C
- It’s not a C++, it’s not even similar
- More similar to SmallTalk
- Dynamic, improved in a rapid pace

Confusing parts

- Methods ~ Selectors
- YES/NO instead of TRUE/FALSE
- nil instead of null
- self instead of this
- *.m files are implementation files

Confusing parts

- Map is called NSDictionary
- NSString constant is written as @"aa"
- NSNumber constant is written as @1
- NS stands for NextStep
- "hello" is a C/C++ string

Calling a method

- Java

```
int a = inst.method(12.0);

MyClass.staticMethod(a, b);
```

- Objective-C

```
int a = [inst methodWithParam:12.0];

[MyClass staticMethodWithParam1:a
                         param2:b];
```

Objective-C methods

- Yes, it is split in multiple parts
  - Named parameters improve readability

```
self.label.textColor = [UIColor colorWithRed:1.0
                                       green:0.0
                                        blue:0.0
                                       alpha:1.0];

NSString *imageName = [NSString
                stringWithFormat:@”image_%d.png”, i];

[[UIImage imageNamed:imgName]
                stretchableImageWithLeftCapWidth:27
                topCapHeight:9];

string = [basePath stringByAppendingPathComponent:
                @"/some/file.txt"];
```

Declaring a method

- Java

```
public int method(double number);

private static void staticMethod(int a, bool b);
```

- Objective-C

```
- (int) methodWithParam:(double)number;

+ (void) staticMethodWithParam1:(int)a
                         param2:(BOOL)b;

// note: avoid “and”, “with”, ... in selector name
// WRONG=> initWithName:andSurname:
//    OK=> initWithName:surname:
```

Declaring a class

```
// Employee.h

#import <Foundation/Foundation.h>

@interface Employee: NSObject <EmployeeProtocol> {
    NSString _name; // often not necessary
}

@property (nonatomic, strong) NSString *name;
@property (nonatomic, strong) NSString *surname;

- (id) initWithName:(NSString*)name
            surname:(NSString*)surname;

@end
```

Defining a class

```
// Employee.m

#import “Employee.h”

@implementation Employee
@synthesize name = _name, surname;

- (id) initWithName:(NSString*)_name
            surname:(NSString*)_surname {
    self = [super init];
    if (self != nil) {
        self.name = _name;
        self.surname = _surname;
    }
    return self;
}

...
```

Defining a class

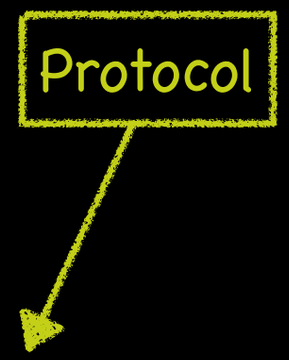
```
...

- (void) greet {
    NSLog(@"Hello, %@ %@!", self.name, self.surname);
}

@end
```

Declaring a protocol

- Protocol ~ Interface

```
// EmployeeProtocol.h
#import <Foundation/Foundation.h>

@protocol EmployeeProtocol <NSObject>

- (void) greet;

@optional

- (void) greetPolitely;

@end
```

Using a protocol




```
// Employee.h

#import <Foundation/Foundation.h>

@interface Employee: NSObject <EmployeeProtocol> {
    int someInstanceVariable;
}

@property (nonatomic, strong) NSString *name;
@property (nonatomic, strong) NSString *surname;

- (id) initWithName:(NSString*)name
            surname:(NSString*)surname;
```

Declaring a category

- Category = Extending class without subclassing

```
// NSString+Crypto.h

#import <Foundation/Foundation.h>

@interface NSString (Crypto)

- (NSString*) cryptedString;

@end
```

Declaring a category

- Category = Extending class without subclassing

```
// NSString+Crypto.m

#import "NSString+Crypto.h"

@implementation NSString (Crypto)

- (NSString*) cryptedString { ... }

@end
```

Class Extension

- "Category" with nothing in the brackets
- Usually implemented in the class implementation file
- Used to implement private methods, properties or to mask read-only modifier on a property

Blocks

- Block = piece of code
- Used throughough the SDK
- ~ Lambda, ~ anonymous
- Blocks are clojures
  - __block type specifier

Blocks - UI Animations

```
imageView.alpha = 0.0;
[UIView animateWithDuration:1.0 animations:^{
    imageView.alpha = 1.0;
} completion:^(BOOL finished) {
    if (finished) {
        // ...
    } else {
        // ...
    }
}];
```

Blocks - Set filtering

```
NSSet *iSet = [NSSet set];
// ... add objects to the set

[set objectsPassingTest:^(id obj, BOOL *stop) {
    return [self testValue:id]; // custom comparison
}];
```

Memory Management

Memory management

- Every NSObject keeps a reference count
- Object is created => references = 1
  - note: created ~ init, new, copy
- Object is retained => references++
- Object is released => references--
- (references == 0) => dealloc

Memory management

- Before iOS 5: manual memory management was needed
  - retain / release / autorelease
  - mechanical, tedious, boring

```
MyClass inst = [[MyClass alloc] init];
// ... do something
[inst release];
```

Since iOS 5 - ARC

- Automatic Reference Counting is available
- Occurs during compilation
- Still some libraries without ARC

```
MyClass inst = [[MyClass alloc] init];
// ... do something
// object will get released in due course
```

Autorelease pools

- Every thread has a stack of autorelease pools
  - Object can register in the pool
  - In due course, the pool sends release to the object
  - When drained, the pool sends release to the object
  - Useful when creating many objects

Autorelease pools

```
// HeavyMasacreComputator.m

- (void) doSomething {
    for (int i = 0; i < TRILLION_TRILLIONS; i++) {
        for (int j = 0; j < TRILLION; j++) {
            MyClass c = [[MyClass alloc] init];
            // do stuff with my class
        }
    }
}
```

Autorelease pools

```
// HeavyMasacreComputator.m

- (void) doSomething {
    for (int i = 0; i < TRILLION_TRILLIONS; i++) {
        @autoreleasepool {
            for (int j = 0; j < TRILLION; j++) {
                MyClass c = [[MyClass alloc] init];
                // do stuff with my class
            }
        }
    }
}
```

Autorelease pools

```
// HeavyMasacreComputator.m - pre iOS 5

- (void) doSomething {
    for (int i = 0; i < TRILLION_TRILLIONS; i++) {
        NSAutoreleasePool *pool = [[NSAutoreleasePool
                                    alloc] init];
        for (int j = 0; j < TRILLION; j++) {
            MyClass c = [[[MyClass alloc] init]
                                    autorelease];
            // do stuff with my class
        }
        [pool drain];
    }
}
```

Ownership Qualifiers

- How objects are assigned
  - __strong - retained
  - __unsafe_unretained - not retained
  - __weak - dtto, set to nil on dealloc
  - __autoreleasing - retained, autoreleased

Properties

- Getter / Setter for the instance variables
- “Dot notation” for access
- Flags in the header file
- Synthesized in the implementation file
- Read-only / read-write properties

Properties

- Property vs. iVar
  - self.name ~ getter / setter used
    - [self name];
    - [self setName:@”Petr”];
  - name ~ direct access to iVar
    - name = @”hello”;

Properties

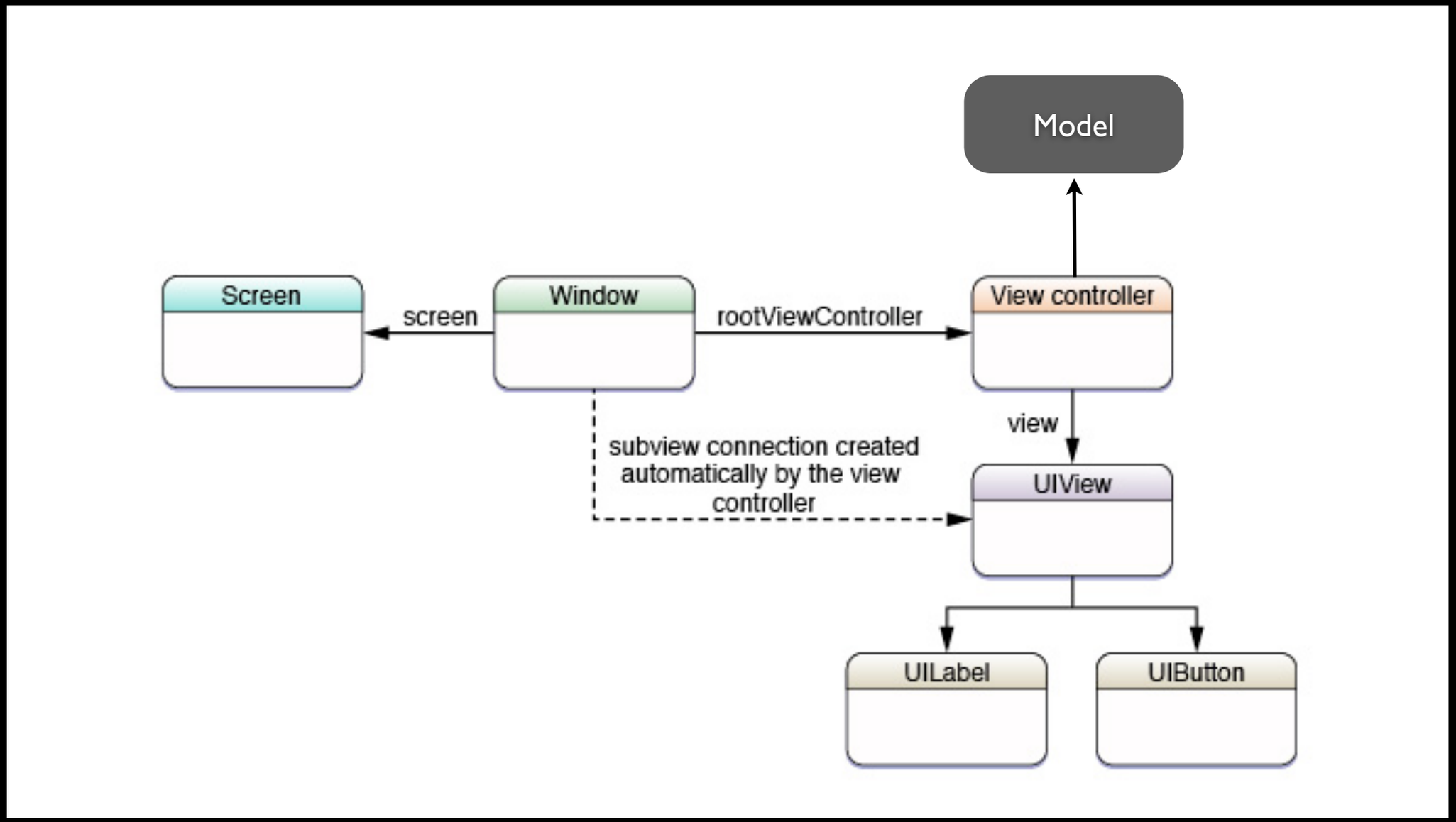
- Implied ownership qualifiers for iVar
  - strong, retain, copy
    => __strong
  - weak
    => __weak
  - assign, unsafe_unretained
    => __unsafe_unretained

UI Elements

UIKit - The Gist

- Framework for iOS UI
- UI Components, gesture recognizers
- MVC approach
  - UIViewController manages UIView
  - UIViewController references model

UIKit - MVC diagram



UIKit - Threading

- UIKit must be used on the main thread to be safe
- Slow tasks must be run on other thread
- UI would be blocked otherwise

UI Consistency

- Controls and views are customizable
- iOS is a very consistent platform, less is often more
- Follow iOS Human Interface Guidelines

iOS UI - Tools

- Storyboard / Interface Builder
- Hints in code for actions/outlets
  - IBAction - for UI actions
  - IBOutlet - for UI elements
- “Segue” since iOS 5
  - View transitions for free

Maps & Location

MapKit

- iOS 6: Apple Maps
- Pre-iOS 6: Google Maps
- High-level API
- Annotations, Routes, Overlays

MKMapView

- UIView subclass
- Based on adding “annotations”
  - = model classes
- Support for user’s location
- Customizable maps & annotations
- Delegate-based API

MKAnnotation

- Protocol that enables a model class for showing up on maps
  - coordinate, title, subtitle
- MKPlacemark
  - conforms to MKAnnotation
  - country, state, city, address

MKAnnotationView

- View related to a particular MKAnnotation instance
- Reused in the map view
- MKPinAnnotationView
  - The classic “iOS map pin”
  - Three colors

Thank you

http://www.inmite.eu/

Petr Dvořák
Partner & Mobile Strategy Consultant
@joshis_tweets