

Praktikum Entwicklung von Mediensystemen mit iOS

WS 2011

Prof. Dr. Michael Rohs
michael.rohs@ifi.lmu.de
MHCI Lab, LMU München

Today

- Schedule
- Organization
- Introduction to iOS
- Exercise 1

Schedule

- Phase 1 – Individual Phase
 - Introduction to basics about iOS
 - Exercises 1 to 3
 - Each student works on exercises himself/herself
 - Weekly meetings
- Phase 2 – Project Phase
 - Concept and implementation of an iOS application
 - Topic: explanation tools (← proposal only!)
 - Students work in teams
 - Regular milestone meetings
- Phase 3 – Evaluation
 - Evaluate your concept
- Phase 4 – Paper Phase
 - Write up results as a paper
 - (Submit it to a relevant conference)

Topic: Explanation Tools

- Mobile learning / teaching tool
- Mobile software that explains how something works
 - How to use the copier / scanner / fax machine
 - How to repair the tire on the bike
 - How to use a complicated kitchen appliance
 - How a plant grows / develops
- Functionality
 - Showing pictures of object / device
 - Selection of from different perspectives
 - Animations that show how to move / operate the object

Timeline

#	Date	Topic
1	19.10.2011	Introduction and overview of iOS
2	26.10.2011	App architecture, touch input, saving data
3	2.11.2011	Location, networking, sensors
4	16.11.2011	Interviews, Storyboarding; Brainstorming
5	30.11.2011	Paper prototyping test, start of software prototype
6	14.12.2011	Heuristic evaluation of software prototype
7	11.1.2012	Think-aloud user study
8	25.1.2012	Completion of software prototype
9	1.2.2012	Final presentation

Organization

- 4 SWS
- (Bi-)Weekly meetings
 - Thursday 16:00 s.t. – 18:00
 - Room 107, Amalienstraße 17
- Homepage:
 - <http://www.medien.ifi.lmu.de/lehre/ws1112/pem/>

Organization

- For team work
- SVN accounts for each team
 - `svn://tracsvn.medien.ifi.lmu.de/repos/pem_team[number]`
(e.g. `svn://tracsvn.medien.ifi.lmu.de/repos/pem_team1`)
- Students check in their exercises to their groups' SVN repository

Teams

- Team 5
- Team 6
- Team 7
- Team 8

Technology – SVN



Technology – SVN I



- SVN - General
 - Version control system
 - Enables collective editing of shared source code
 - Data stored in a „repository“ which is accessed over the network
 - Editing on local copies of the files
 - Old version available on the server
 - When possible, files will be merged automatically when edited by multiple users at the same time
 - Similar to CVS

Technology – SVN II



- SVN – First Steps (using Tortoise SVN)
 1. Download a SVN Client for Mac OS X
<http://gigaom.com/apple/12-subversion-apps-for-os-x/>
 2. SVN command line should be already installed on your Mac
Utilities → Terminal
 3. Checkout your team repository (creates a local copy of the repository)
Create an empty folder, open it, right-click and choose „Checkout“.
`svn://murx.medien.ifi.lmu.de/team1`

- SVN – First Steps (using Tortoise SVN)
 3. Each time you start working perform the “svn update” command
 4. Each time you are done working perform a “svn commit”
 5. Use “svn <command> help” to get help on a command
 6. Use “svn help” to discover new functionality...
 7. **Attention:** Do not use the OS-functionalities for “delete” or “rename”. Use svn commands for this, so that svn is not confused of missing or renamed files. Never ever touch the hidden .svn-Folders.

For further Information read the German SVN introduction by Richard Atterer, which can be found here:

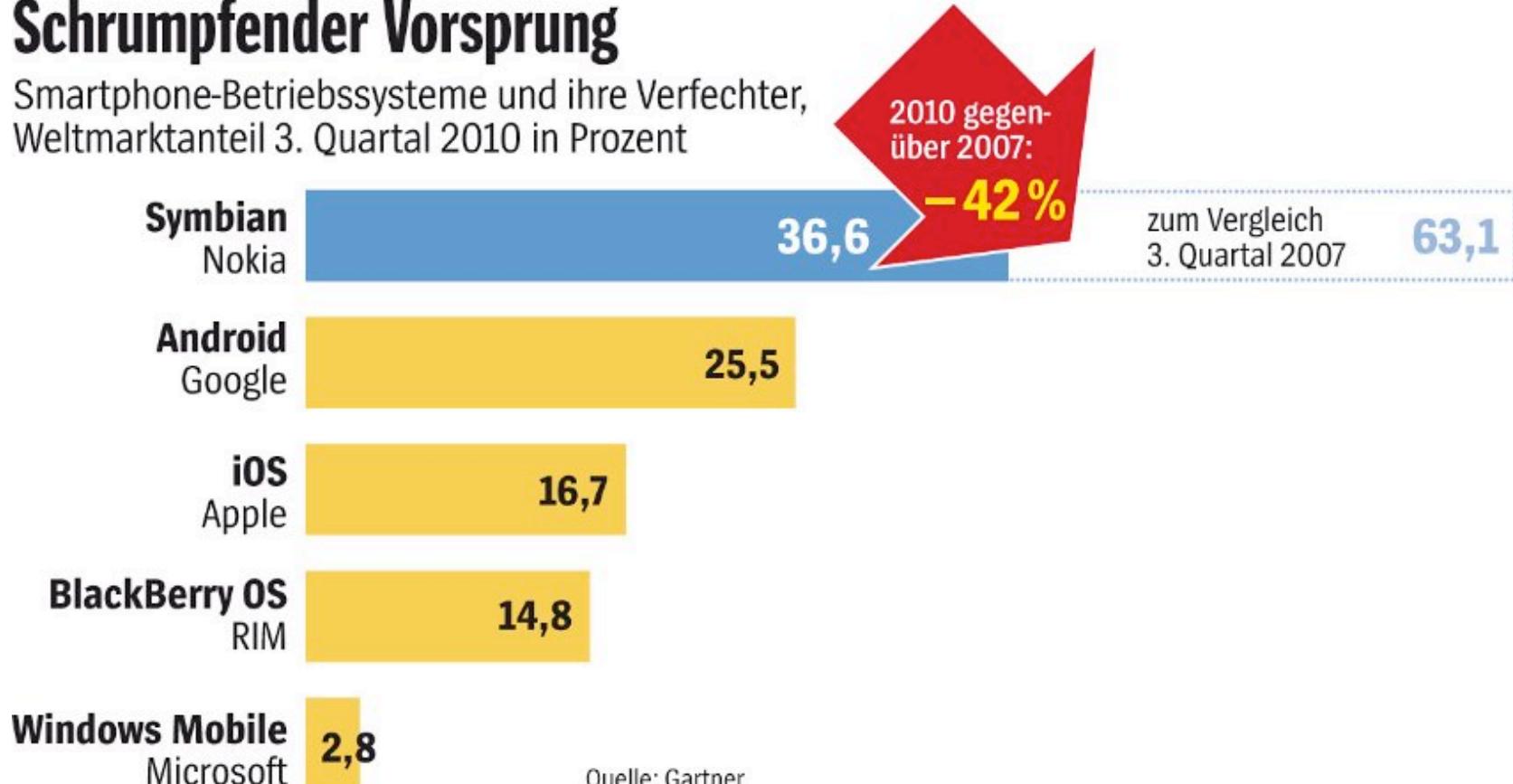
http://www.medien.ifi.lmu.de/fileadmin/mimuc/mmp_ss04/Projektaufgabe/mmp-subversion.pdf

Apple iOS Overview

Smartphone Operating Systems

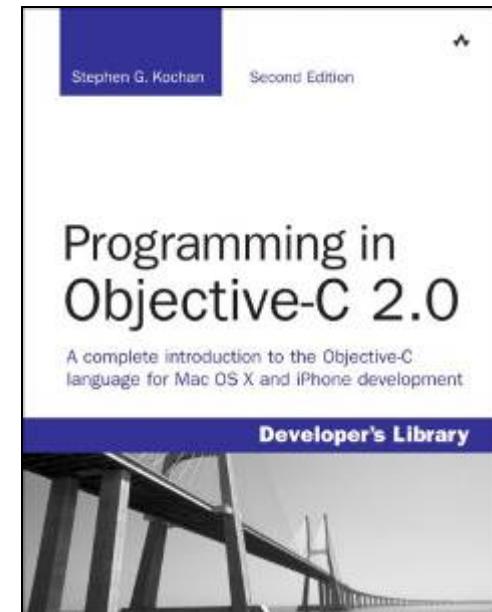
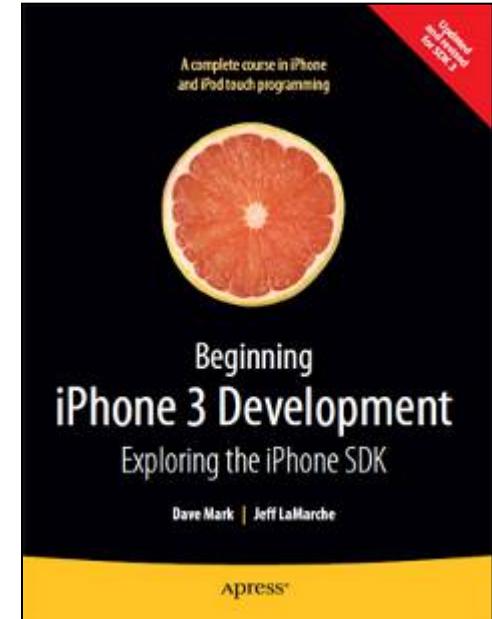
Schrumpfender Vorsprung

Smartphone-Betriebssysteme und ihre Verfechter,
Weltmarktanteil 3. Quartal 2010 in Prozent



Books

- iPhone development
 - Dave Mark, Jeff LaMarche: Beginning iPhone 3 Development: Exploring the iPhone SDK. Apress, 2009.
 - <http://www.amazon.com/BEGINNING-IPHONE-DEVELOPMENT-EXPLORING-SDK/dp/1430224592/>
- Objective C
 - Stephen G. Kochan: Programming in Objective-C 2.0. Addison-Wesley, 2nd edition, 2009.
 - <http://www.amazon.com/PROGRAMMING-OBJECTIVE-C-2-0-STEPHEN-KOCHAN/dp/0321566157/>



User Interface Guidelines

- Concrete guidelines for look-and-feel and behavior
 - Visual appearance, e.g., icon design
 - Purpose of user interface elements
 - Layout of user interface elements
 - Behavior, conventions of system features
- iOS Human Interface Guidelines
 - <http://developer.apple.com/library/ios/documentation/userexperience/conceptual/mobilehig/MobileHIG.pdf>
 - Aesthetic integrity, consistency, direct manipulation, feedback, metaphors, user control, ...

Apple iOS

- Optimized version of Mac OS X
 - New components for handling touch
 - Memory requirement < 0.5 GB
- Hardware
 - 620 MHz ARM 1176 – 1GHz Apple A5
 - 128-512 MB DRAM
 - 4/8/16/32 GB flash RAM
 - Graphics: PowerVR OpenGL ES chip
 - Camera: 2.0-8.0 megapixels
 - Screen: 320x480 pixels, 163 ppi – 640x960 pixels, 326 ppi
 - Connectivity: GSM/UMTS, Wi-Fi (802.11b/g/n), Bluetooth
- SDK available since spring 2008

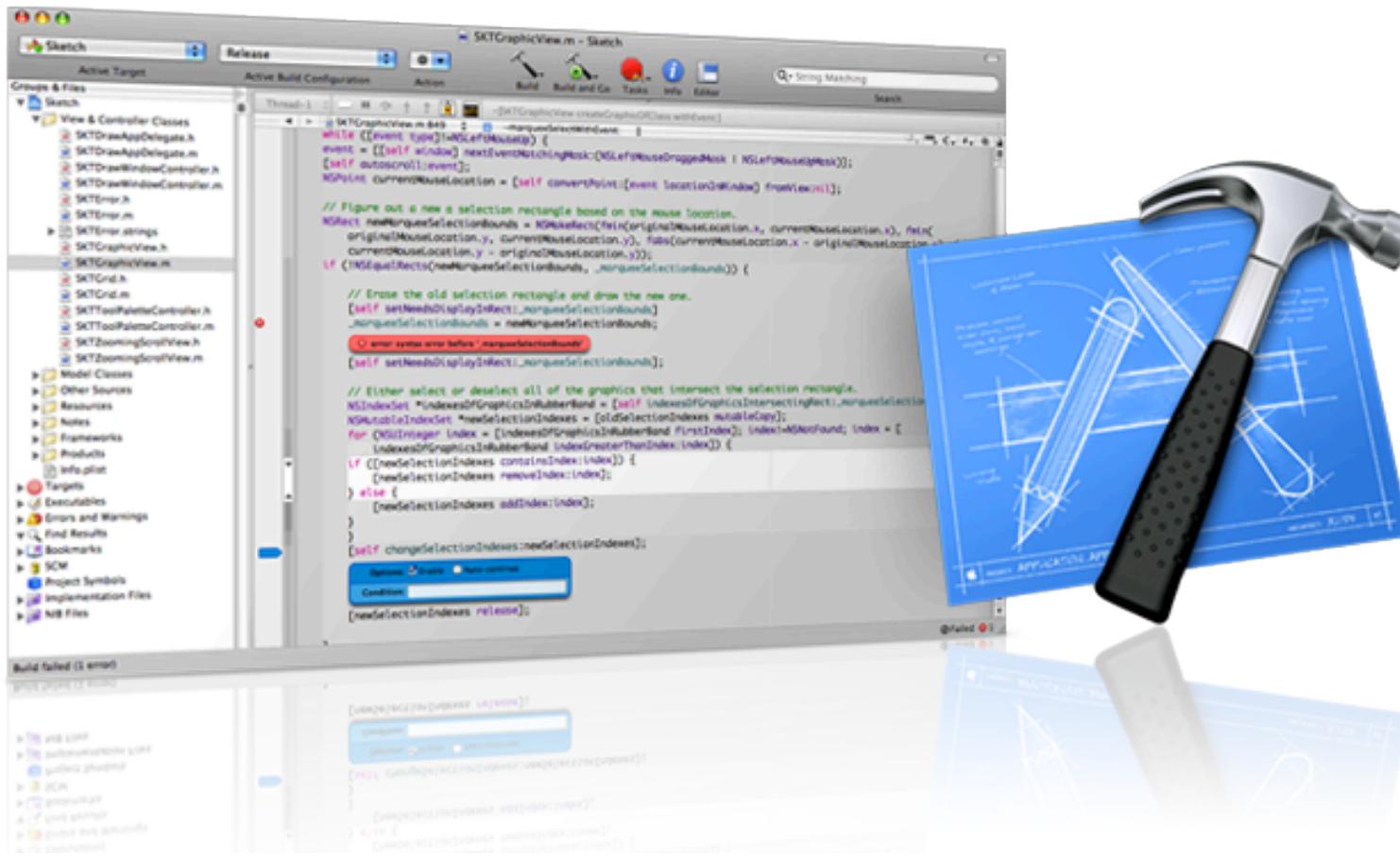


SDK Options

- Official iPhone SDK
 - Requires Mac to develop (IDE/compiler/debugger only for Mac)
 - Requires registration as developer (\$99 per year)
 - Official support
 - Possibility to release on Apple App Store
 - <http://developer.apple.com/devcenter/ios/>
- iPhone toolchain SDK
 - Unofficial SDK
 - Available for Mac, Linux, PC (with varying comfort)
 - Command line gcc compiler (on-device compiling also possible)
 - All features of the phone actually accessible (even closed ones)
 - Requires “jailbreaking” the phone
 - May be legally questionable
 - <http://code.google.com/p/iphone-dev/>

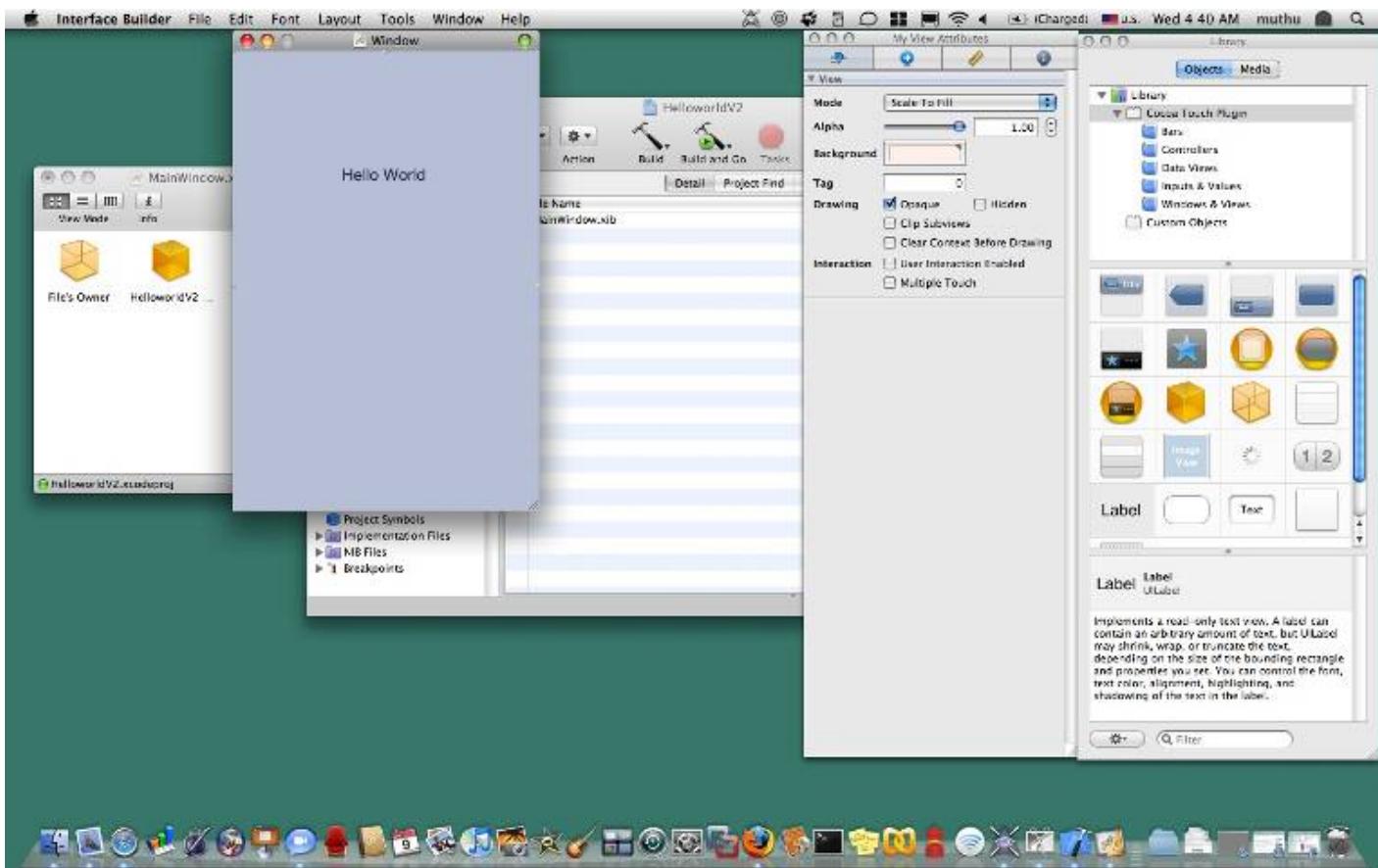
Development Environment

- Xcode: IDE + integrated compiler, run-time debugger



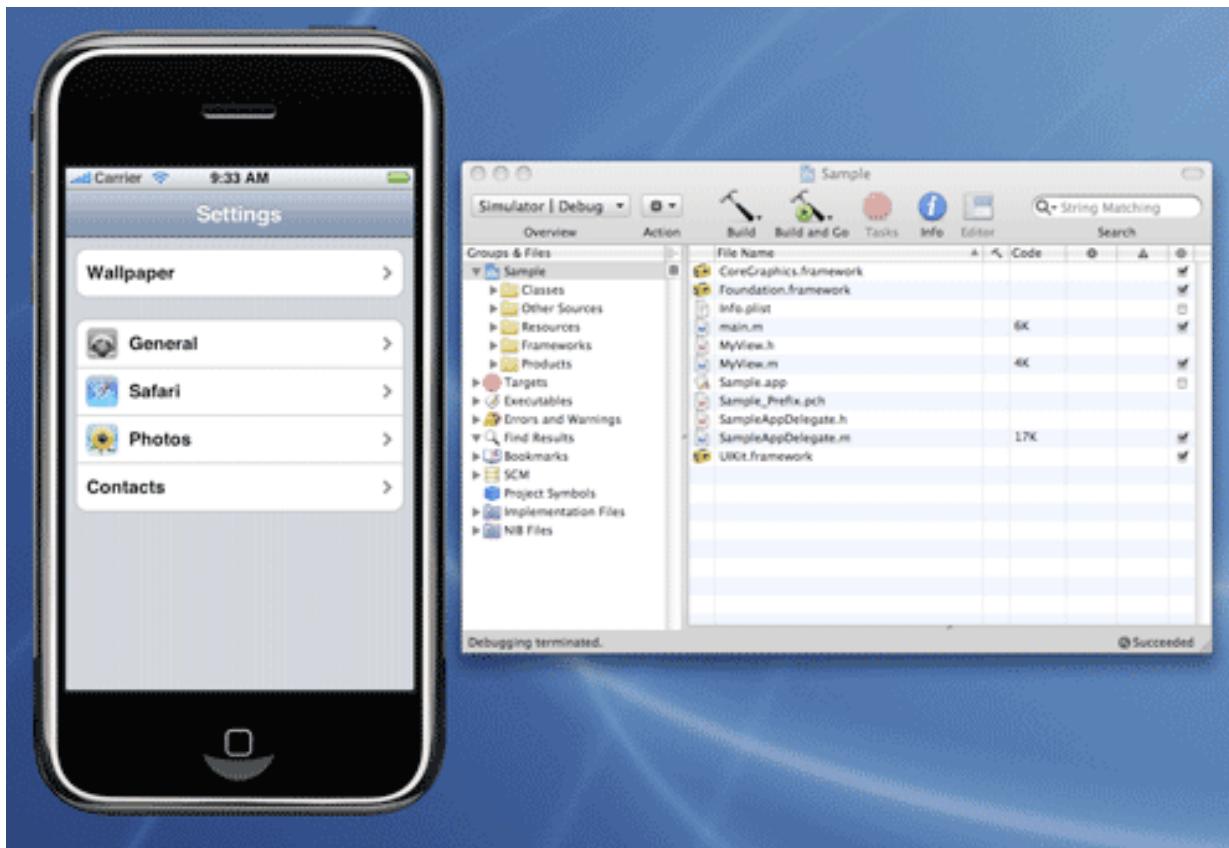
Development Environment

- Interface Builder: Graphical interface layouter



Development Environment

- iPhone Simulator: Mac simulator of iPhone
 - Most features except tilt, simulated multitouch



iOS Technical Background

Philosophy of the API

- Compatibility with Mac OS X
 - Foundation frameworks: shared, Cocoa Touch: iPhone-only
- Maintains general framework structure
- Benefit
 - Shared code development between iPhone and OS X
 - Rapid porting of applications
 - Developer familiarity (for previous Mac developers)
- Preferred language
 - Objective C (implementation language of the SDK)
 - C/C++ work
- Protective
 - Some APIs are privileged and cannot be accessed
 - Example: AudioCore, LayerKit (direct access to framebuffers)

Cocoa Touch Architecture

- Cocoa Touch
 - High level architecture for building iOS applications
- Cocoa Touch consists of:
- UIKit
 - User interface elements
 - Application runtime
 - Event handling
 - Hardware APIs
- Foundation
 - Utility classes
 - Collection classes
 - Object wrappers for system services
 - Subset of Foundation in Cocoa

Objective C

- Objective C is superset of C, with OO constructs
 - Unusual Syntax, rarely used outside Apple realm, inspired by SmallTalk
- General syntax for method calls (“messages”):
`object.method(parameter1, parameter2);` becomes:
`[object method:parameter1 parameterkey:parameter2];`
- Example
`employee.setSalary(100,20); // arguments base_salary, bonus`
`[employee setSalary:100 withBonus:20];`
- Learn more at
developer.apple.com/documentation/Cocoa/Conceptual/ObjectiveC

Objective C - Methods

- Method declaration syntax
 - ± (type) selector:(type)param paramkey:(type)param2;
 - Instance methods: - (void) myInstanceMethod;
 - Class methods: + (void) myClassMethod;
- Example
 - (void) setSalary:(int)income withBonus:(int)bonus;
- Basic classes, examples
 - NSObject is root class (basics of memory management)
 - NSString
 - Example: s = [NSString stringWithFormat: @"The answer is: %@", myObject];
 - Constant strings are @"this is a constant string"
 - NSLog(@"%@", object); (NSLog is your friend...)
 - NS... also offers collections (NSArray, NSDictionary etc) and other basic language service functionality
 - Prefix "NS" is derived from OS X predecessor, NextStep

Objective C – Features and Pitfalls

- Dynamically typed objects (or hard to find bugs)
 - `id someObject`
 - `id` is generic “pointer” without type (“`void*`”)
 - introspection allows finding out type at runtime
- Nil object pointers (or how to make really hard to find bugs)
`object = nil;`
`[object setProperty: nil];`
 - Will send message to nil, hard to find if objects didn’t get proper assignment
- `id`, `nil` and dynamic typing enable message-passing paradigm

Memory Management By Hand

- Don't create memory leaks!
- Object reference life cycle:

```
myobject = [[MyClass alloc] init];           // reference count = 1 after alloc  
[myobject retain];    // increment reference count (retainCount == 2)  
[myobject release];   // decrement reference count (retainCount == 1)  
[myobject release];   // decrement reference count (retainCount == 0)  
// at this point myobject is no longer valid, memory has been reclaimed  
[myobject someMethod]; // error: this will crash!
```

- Can inspect current reference count:
`NSLog(@"retainCount = %d", [textField retainCount]);`
- Can autorelease (system releases at some point in future)
`[myobject autorelease];`
Used when returning objects from methods.

Memory Management By Hand

- Memory rule: You are responsible for objects you allocate or copy (i.e. “allocate” or “copy” is some part of the name)!
- Not responsible:
`NSData *data = [NSData dataWithContentsOfFile:@"file.dat"];`
- Responsible:
`NSData *data = [[NSData alloc] initWithContentsOfFile:@"file.dat"];`
- Responsible:
`NSData *data2 = [data copy];`
- Never release objects you are not responsible for!

Objective C – Practical Aspects

- Based file extension .m
- Header file extension .h (expects Objective-C base file)
- Base file extension for Objective C++ is .mm (not .cpp)
- #import <...> (automatic redundancy check)

Objective C - Class

In .h file:

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

@interface Employee : NSObject
{ //Instance vars here
    NSString *name;
    int salary;
    int bonus;
}
// methods outside curly brackets
- (void)setSalary:(int)cash withBonus:(int)extra
@end
```

Objective C - Class

In .m file:

```
#import "Employee.h"

@implementation Employee
- (void)setSalary:(int)cash withBonus:(int)extra
{
    salary = cash;
    bonus = extra;
}
@end
```

Objective C - Protocols

- Used to simulate multiple inheritance and add functionality on top of existing objects (i.e. for delegates), similar to **interfaces** in Java:

```
@protocol Locking
```

```
- (void)lock;  
- (void)unlock;
```

```
@end
```

- Denotes that there is an abstract idea of „Locking“
- Classes can state that they implement „Locking“ by declaring the following:

```
@interface SomeClass : SomeSuperClass <Locking>
```

```
@end
```

Objective C Properties

- .h file:

```
@interface MyDetailViewController : UIViewController {  
    NSString *labelText;  
}  
  
@property (nonatomic, retain) NSString *labelText;  
@end
```

- .m file:

```
@synthesize labelText;  
  
-(void)someMethod {  
    self.labelText = @”hello”;  
}
```

creates accessor methods:
setLabelText (retains/releases)
and getLabelText.

dot-syntax means: use property's
setLabelText accessor method,
will retain the object

equivalent to
[self setLabelText:@”hello”];

Implicit Setter/Getter Accessor Methods

- .h file: `@property (nonatomic, retain) NSString *labelText;`
- .m file: `@synthesize labelText;`
- Automatic creation of accessor methods:

```
- (void) setLabelText:(NSString *)newLabelText {  
    [labelText release];  
    labelText = newLabelText;  
    [labelText retain];  
}  
- (NSString *) getLabelText {  
    return labelText;  
}
```

decrement reference counter on old object (if any)

increment reference counter on new object (if any)

- Properties are accessible from other classes, data members only if declared `@public`

Property Attributes

- Writability: `readwrite` (default), `readonly`
- Setter semantics: `assign`, `retain`, `copy`
- Atomicity: `atomic` (default), `nonatomic`
- “`readonly`” means only a getter, but no setter accessor method is generated by `@synthesize`

Selectors

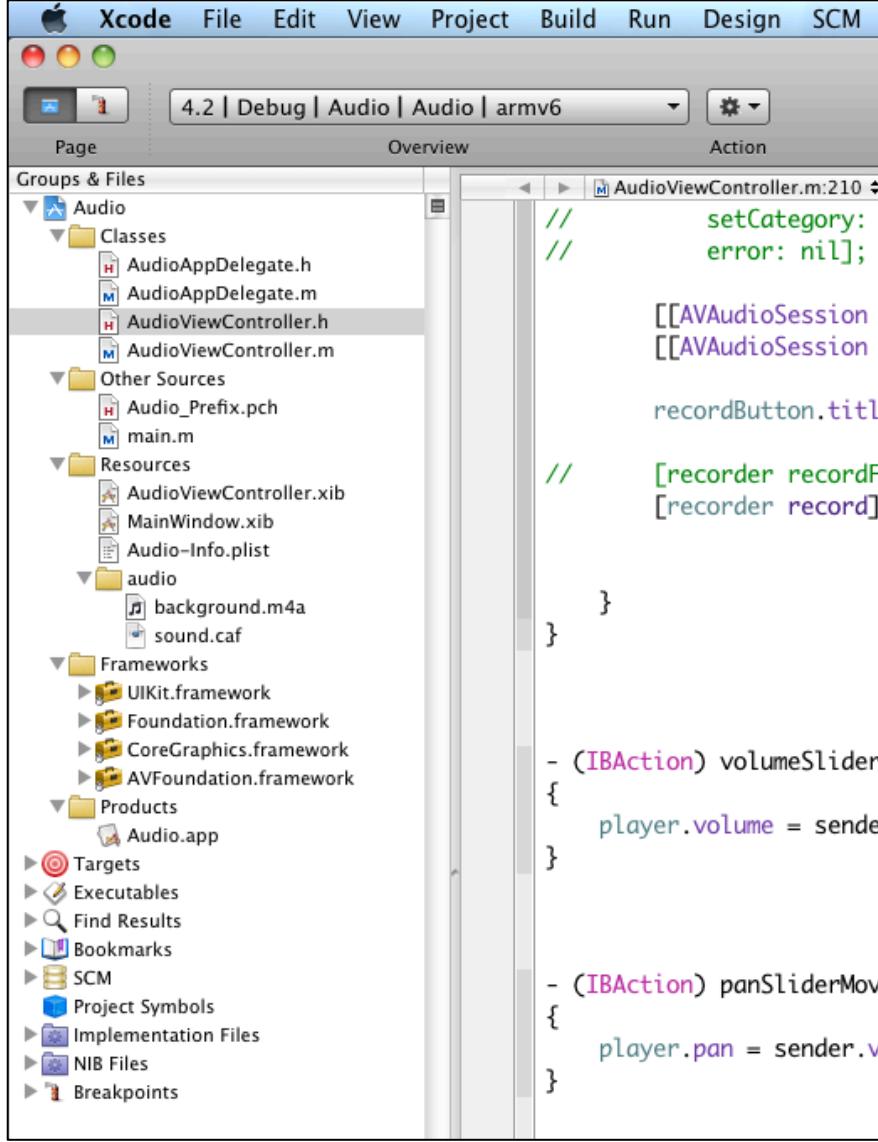
- Methods as arguments (useful for callback methods)
- Example: setting a button callback method
- .h file

```
@interface MyDetailViewController : UIViewController {  
    IBOutlet UIButton *newButton;  
}  
  
@property (nonatomic, retain) UIButton *newButton;  
- (void) newButtonPressed:(id)source;
```

- .m file
 - (void)someInitializationMethod {
 [newButton addTarget:self action:@selector(newButtonPressed:)
 forControlEvents:UIControlEventTouchUpInside];
}
 - (void) newButtonPressed:(id)source { NSLog(@"newButtonPressed"); }

Contents of an Xcode iPhone Project

- Source files
- Compiled Code
- Framework code
 - E.g. UIKit.framework
- Nib file (extension .xib)
 - Contains interface builder data
- Resources
 - Media (images, icons, sound)
- Info.plist file
 - Application configuration data



The screenshot shows the Xcode IDE interface. The top menu bar includes Xcode, File, Edit, View, Project, Build, Run, Design, and SCM. The toolbar below has buttons for Page, Overview, and Action. The main window has tabs for Groups & Files, Page, and Overview. The Groups & Files tab shows a project structure with folders for Audio, Classes, Other Sources, Resources, Frameworks, Products, Targets, Executables, Find Results, Bookmarks, SCM, Project Symbols, Implementation Files, NIB Files, and Breakpoints. Under the Audio folder, there are subfolders for Classes and Resources. The Classes folder contains files like AudioAppDelegate.h, AudioAppDelegate.m, AudioViewController.h, and AudioViewController.m. The Resources folder contains XIB files for AudioViewController and MainWindow, an info.plist file, and media files for audio (background.m4a, sound.caf). The Frameworks folder lists UIKit.framework, Foundation.framework, CoreGraphics.framework, and AVFoundation.framework. The Products folder contains the final app bundle named Audio.app. The right side of the interface is a code editor showing a portion of the AudioViewController.m file. The code includes imports for AVAudioSession, recorder, and player, and contains IBAction methods for volumeSlider and panSliderMove.

```
4.2 | Debug | Audio | Audio | armv6
Page Overview Action
Groups & Files
  Audio
    Classes
      AudioAppDelegate.h
      AudioAppDelegate.m
      AudioViewController.h
      AudioViewController.m
    Other Sources
      Audio_Prefix.pch
      main.m
  Resources
    AudioViewController.xib
    MainWindow.xib
    Audio-Info.plist
    audio
      background.m4a
      sound.caf
  Frameworks
    UIKit.framework
    Foundation.framework
    CoreGraphics.framework
    AVFoundation.framework
  Products
    Audio.app
Targets
Executables
Find Results
Bookmarks
SCM
Project Symbols
Implementation Files
NIB Files
Breakpoints

[[AVAudioSession
[[AVAudioSession
recordButton.title
[recorder record
[recorder record

// 
}

- (IBAction) volumeSlider
{
    player.volume = sender.
}

- (IBAction) panSliderMove
{
    player.pan = sender.v
}

AudioViewController.m:210
setCategory:
error: nil];
//
```

HELLO WORLD

“Hello World” Steps

- Creating a project (“View-based Application”)
- Inspecting package contents
 - Navigator (left pane)
- Inspecting HelloWorldViewController.xib
 - Utilities (right pane)
 - Adding a label and a button
- Adding event handling method to HelloWorldViewController
 - (IBAction)** buttonPressed:(**id**)sender;
 - Log output: **NSLog(@"button pressed");**
- Linking button to event handler in xib file
- Linking button to event handler using @selector

“Hello World” Steps

- Set label text when button was pressed
 - Add label outlet and property in .h file
 - Synthesize label property and set label text in .m file
- Increment counter when button was pressed
 - Add variable in .h file
 - Use NSString stringWithFormat in .m file
- Access label view using a tag (no IBOutlet required)
 - Define tag for label in Interface Builder (e.g. Tag = 100)
 - UILabel *label = (UILabel*)[self.view viewWithTag:100];
- Explain what happens during instantiation
- Showing a UIAlertView
- Explain #pragma

“Hello World” Steps

- Text input
 - Add UITextField in Interface Builder
 - Add member variable and property to .h, synthesize in .m
 - Declare UITextFieldDelegate in .h
 - Implement delegate methods in .m, set label text on end editing
 - Set delegate in viewDidLoad method
- Action sheets
 - Implement UIActionSheetDelegate in .h file
 - Construct, showInView, release
 - Implement delegate method clickedButtonAtIndex

Choose a template for your new project:

The screenshot shows the Xcode interface for creating a new project. On the left, there's a sidebar with two sections: 'iOS' and 'Mac OS X'. Under 'iOS', the 'Application' option is selected, showing sub-options: 'Navigation-based Application', 'OpenGL ES Application', 'Split View-based Application', 'Tab Bar Application', 'Utility Application', and 'View-based Application'. Under 'Mac OS X', options include 'Application', 'Framework & Library', 'Application Plug-in', 'System Plug-in', and 'Other'. A large preview area in the center displays icons for each template: a blue arrow for Navigation-based, a purple cube for OpenGL ES, a grid with a plus sign for Split View-based, a star and dots for Tab Bar, a green double-headed arrow for Utility, and a blue square for View-based. Below the preview, a detailed description of the 'View-based Application' template is shown.

iOS

Application
Framework & Library
Other

Mac OS X

Application
Framework & Library
Application Plug-in
System Plug-in
Other

Navigation-based Application
OpenGL ES Application
Split View-based Application
Tab Bar Application
Utility Application
View-based Application

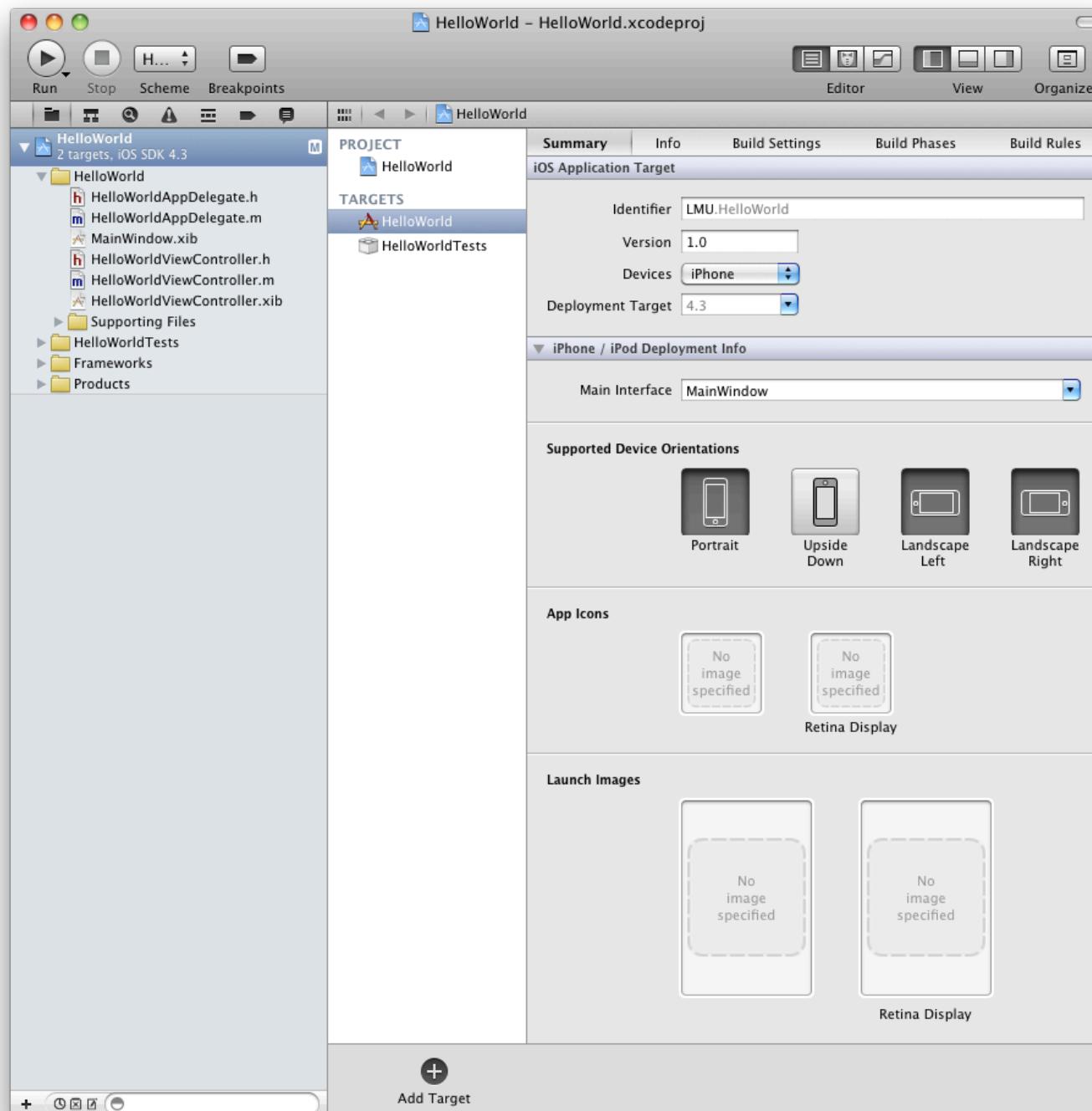
View-based Application

This template provides a starting point for an application that uses a single view. It provides a view controller to manage the view, and a nib file that contains the view.

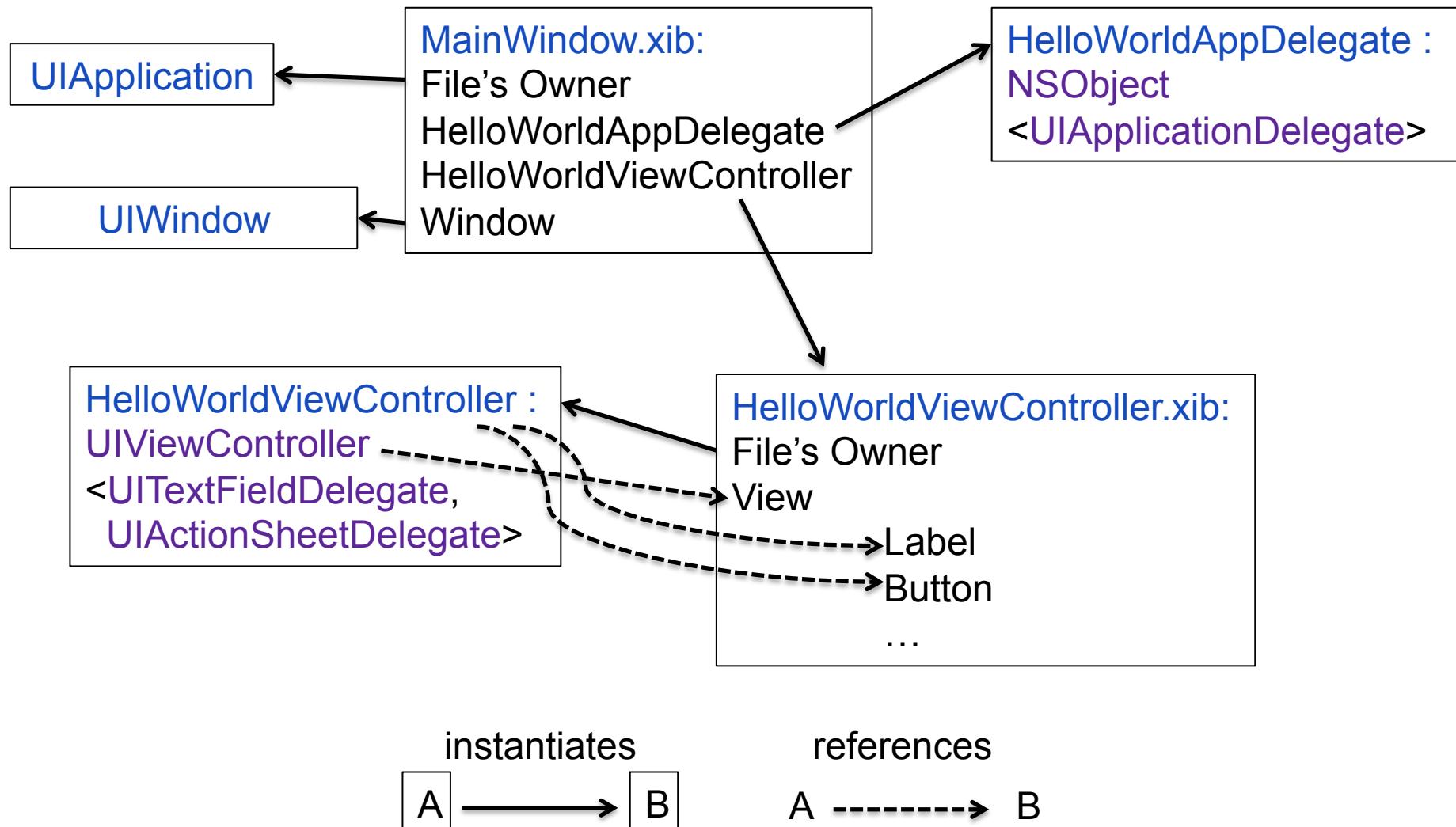
Cancel

Previous

Next



Hello World Application Architecture



instantiates
A → B

references
A -----> B

The screenshot shows the Xcode interface with the following details:

- Toolbar:** Run, Stop, Scheme, Breakpoints.
- File menu:** Editor, View, Organizer.
- Project Navigator:** Shows the project structure:
 - >HelloWorld (target)
 - ↳ HelloWorld (group)
 - ↳ HelloWorldAppDelegate.h
 - ↳ HelloWorldAppDelegate.m (selected)
 - ↳ MainWindow.xib
 - ↳ HelloWorldViewController.h
 - ↳ HelloWorldViewController.m
 - ↳ HelloWorldViewController.xib
 - ↳ Supporting Files
 - ↳ HelloWorld-Info.plist
 - ↳ InfoPlist.strings
 - ↳ HelloWorld-Prefix.pch
 - ↳ main.m
 - ↳ HelloWorldTests (group)
 - ↳ Frameworks
 - ↳ UIKit.framework
 - ↳ Foundation.framework
 - ↳ CoreGraphics.framework
 - ↳ Products
 - ↳ HelloWorld.app
 - ↳ HelloWorldTests.octest
- Editor:** Displays the code for `HelloWorldAppDelegate.m`. The code is as follows:

```
// HelloWorldAppDelegate.m
// HelloWorld
//
// Created by Michael Rohs on 3.5.2011.
// Copyright 2011 __MyCompanyName__. All rights reserved.

#import "HelloWorldAppDelegate.h"

#import "HelloWorldViewController.h"

@implementation HelloWorldAppDelegate

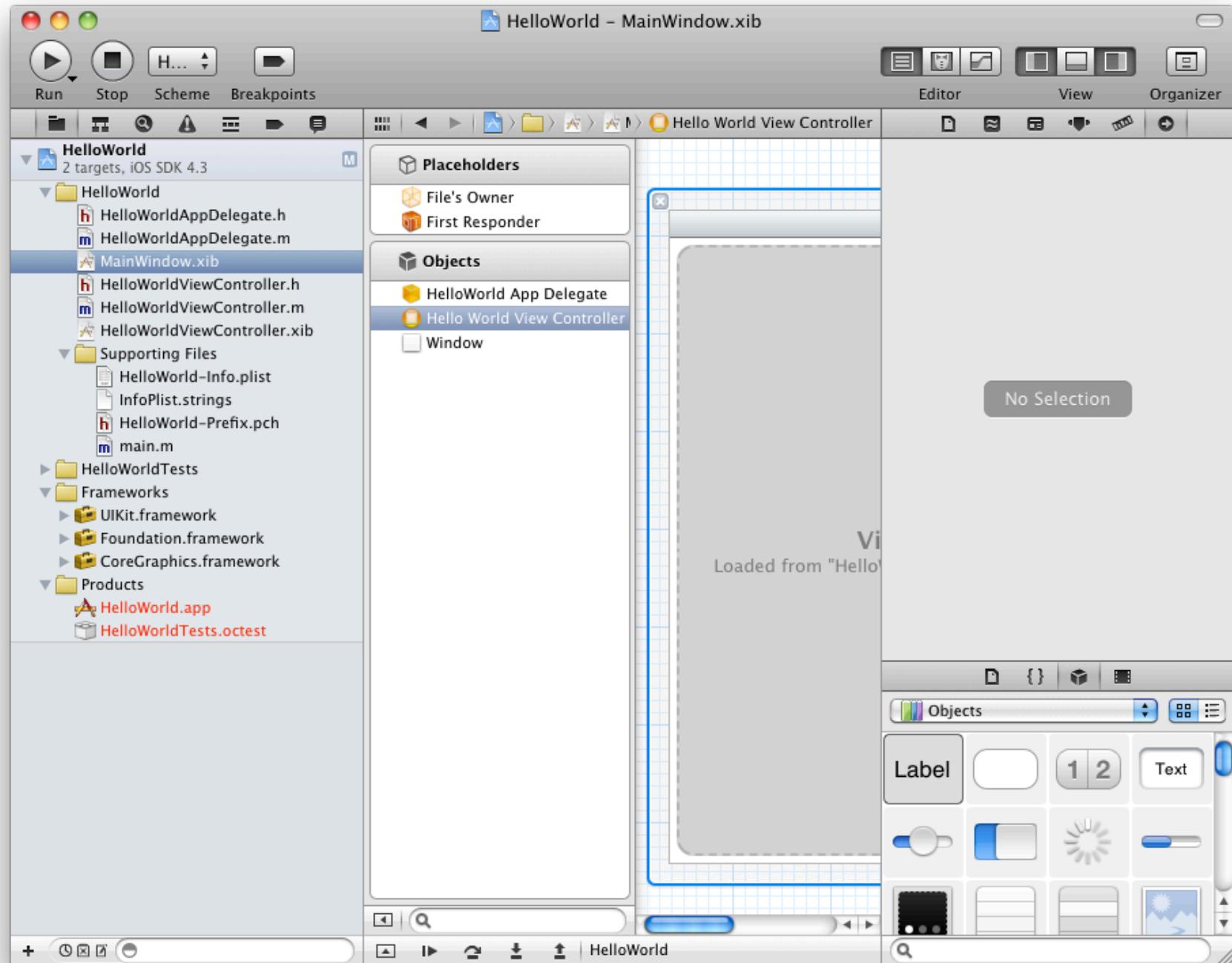
@synthesize window=_window;
@synthesize viewController=_viewController;

- (BOOL)application:(UIApplication *)application didFinishLaunchingWithOptions:
(NSDictionary *)launchOptions
{
    // Override point for customization after application launch.

    self.window.rootViewController = self.viewController;
    [self.window makeKeyAndVisible];
    return YES;
}

- (void)applicationWillResignActive:(UIApplication *)application
{
    /*
        Sent when the application is about to move from active to inactive state. This
        can occur for certain types of temporary interruptions (such as an
        incoming phone call or SMS message) or when the user quits the application
        and it begins the transition to the background state.
        Use this method to pause ongoing tasks, disable timers, and throttle down
        OpenGL ES frame rates. Games should use this method to pause the game.
    */
}

- (void)applicationDidEnterBackground:(UIApplication *)application
{
    /*
        Use this method to release shared resources, save user data, invalidate timers
        and store enough application state information to restore your
        application to its current state in case it is terminated later.
        If your application supports background execution, this method is called instead
        of applicationWillTerminate: for devices running OS X v10.5 or later.
    */
}
```



UIViewController subclasses

- View lifecycle
 - `(void)viewDidLoad`
 - `(void)viewDidUnload`
- View events
 - `(void) viewDidAppear:(BOOL)animated`
 - `(void) viewWillDisappear:(BOOL)animated`
 - `(void) viewDidDisappear:(BOOL)animated`
 - `(void) viewDidDisappear:(BOOL)animated`
- Rotation settings and events
 - `interfaceOrientation` property
 - `shouldAutorotateToInterfaceOrientation:`
- many more... → see documentation

HelloWorld – HelloWorldViewController.h

Run Stop Scheme Breakpoints Editor View Organizer

File Name HelloWorldViewController.h
File Type Default – C header
Location Relative to Group
Full Path /Users/michaelrohs/TLabs/localhome/Organization/LMU/11SS iPhone/TestProjects/HelloWorld/HelloWorld/HelloWorldViewController.h
Localization No Localizations
Target Membership HelloWorld HelloWorldTests
Objects Label 1 2 Text
Switch Button Progress Bar
Image View Image
Search

HelloWorld 2 targets, iOS SDK 4.3
 HelloWorld
 HelloWorldAppDelegate.h
 HelloWorldAppDelegate.m
 MainWindow.xib
 HelloWorldViewController.h
 HelloWorldViewController.m
 HelloWorldViewController.xib
 Supporting Files
 HelloWorld-Info.plist
 InfoPlist.strings
 HelloWorld-Prefix.pch
 main.m
 HelloWorldTests
 Frameworks
 UIKit.framework
 Foundation.framework
 CoreGraphics.framework
 Products
 HelloWorld.app
 HelloWorldTests.octest

```
// HelloWorldViewController.h
// HelloWorld
//
// Created by Michael Rohs on 3.5.2011.
// Copyright 2011 __MyCompanyName__. All rights reserved.

#import <UIKit/UIKit.h>

@interface HelloWorldViewController : UIViewController {
}

-(IBAction) buttonPressed:(id)sender;

@end
```

The screenshot shows the Xcode interface for an iOS project named "HelloWorld".

Project Navigator:

- 2 targets, iOS SDK 4.3
- >HelloWorld (selected)
- ↳ HelloWorld
- ↳ HelloWorldAppDelegate.h
- ↳ HelloWorldAppDelegate.m
- ↳ MainWindow.xib
- ↳ HelloWorldViewController.h
- ↳ HelloWorldViewController.m (selected)
- ↳ HelloWorldViewController.xib
- ↳ Supporting Files
 - ↳ HelloWorld-Info.plist
 - ↳ InfoPlist.strings
 - ↳ HelloWorld-Prefix.pch
 - ↳ main.m
- ↳ HelloWorldTests
- ↳ Frameworks
 - ↳ UIKit.framework
 - ↳ Foundation.framework
 - ↳ CoreGraphics.framework
- ↳ Products
 - ↳ HelloWorld.app
 - ↳ HelloWorldTests.octest

Editor: HelloWorld – HelloWorldViewController.m

```
-(void)dealloc
{
    [super dealloc];
}

-(void)didReceiveMemoryWarning
{
    // Releases the view if it doesn't have a superview.
    [super didReceiveMemoryWarning];

    // Release any cached data, images, etc that aren't in use.
}

#pragma mark - View lifecycle

/*
// Implement viewDidLoad to do additional setup after loading the view,
// typically from a nib.
- (void)viewDidLoad
{
    [super viewDidLoad];
}

- (void)viewDidUnload
{
    [super viewDidUnload];
    // Release any retained subviews of the main view.
    // e.g. self.myOutlet = nil;
}

- (BOOL)shouldAutorotateToInterfaceOrientation:(UIInterfaceOrientation)interfaceOrientation
{
    // Return YES for supported orientations
    return (interfaceOrientation == UIInterfaceOrientationPortrait);
}

-(IBAction)buttonPressed:(id)sender;
{
    NSLog(@"button pressed");
}

@end
```

Identity and Type:

- File Name: HelloWorldViewController.m
- File Type: Default – Objective-C
- Location: Relative to Group
- Full Path: /Users/michaelrohs/TLabs/localhome/Organization/LMU/11SS iPhone/TestProjects/HelloWorld>HelloWorld/HelloWorldViewController.m

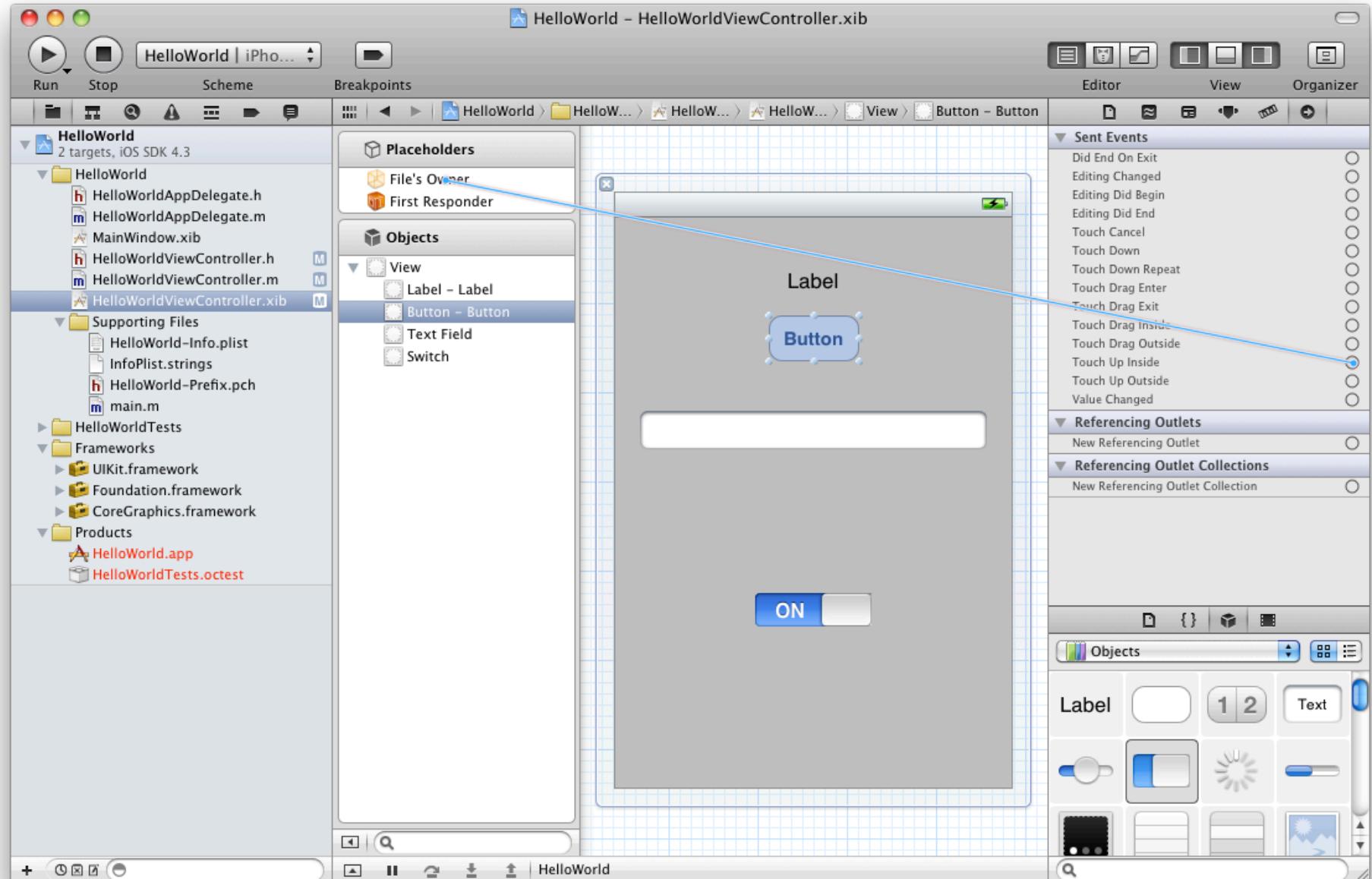
Localization: No Localizations

Target Membership:

- HelloWorld (checked)
- HelloWorldTests (unchecked)

Objects:

- Label
- 1 2
- Text
- Switch
- Stepper
- Image
- Image (with sun icon)
- Image (with landscape icon)



HelloWorld – HelloWorldViewController.h

Run Stop Scheme Breakpoints

Editor View Organizer

Quick Help No Quick Help

Objects

Label Text

1 2

3 4

5 6

7 8

9 10

11 12

13 14

15 16

17 18

19 20

21 22

23 24

25 26

27 28

29 30

31 32

33 34

35 36

37 38

39 40

41 42

43 44

45 46

47 48

49 50

51 52

53 54

55 56

57 58

59 60

61 62

63 64

65 66

67 68

69 70

71 72

73 74

75 76

77 78

79 80

81 82

83 84

85 86

87 88

89 90

91 92

93 94

95 96

97 98

99 100

101 102

103 104

105 106

107 108

109 110

111 112

113 114

115 116

117 118

119 120

121 122

123 124

125 126

127 128

129 130

131 132

133 134

135 136

137 138

139 140

141 142

143 144

145 146

147 148

149 150

151 152

153 154

155 156

157 158

159 160

161 162

163 164

165 166

167 168

169 170

171 172

173 174

175 176

177 178

179 180

181 182

183 184

185 186

187 188

189 190

191 192

193 194

195 196

197 198

199 200

201 202

203 204

205 206

207 208

209 210

211 212

213 214

215 216

217 218

219 220

221 222

223 224

225 226

227 228

229 230

231 232

233 234

235 236

237 238

239 240

241 242

243 244

245 246

247 248

249 250

251 252

253 254

255 256

257 258

259 260

261 262

263 264

265 266

267 268

269 270

271 272

273 274

275 276

277 278

279 280

281 282

283 284

285 286

287 288

289 290

291 292

293 294

295 296

297 298

299 300

301 302

303 304

305 306

307 308

309 310

311 312

313 314

315 316

317 318

319 320

321 322

323 324

325 326

327 328

329 330

331 332

333 334

335 336

337 338

339 340

341 342

343 344

345 346

347 348

349 350

351 352

353 354

355 356

357 358

359 360

361 362

363 364

365 366

367 368

369 370

371 372

373 374

375 376

377 378

379 380

381 382

383 384

385 386

387 388

389 390

391 392

393 394

395 396

397 398

399 400

401 402

403 404

405 406

407 408

409 409

410 411

411 412

412 413

413 414

414 415

415 416

416 417

417 418

418 419

419 420

420 421

421 422

422 423

423 424

424 425

425 426

426 427

427 428

428 429

429 430

430 431

431 432

432 433

433 434

434 435

435 436

436 437

437 438

438 439

439 440

440 441

441 442

442 443

443 444

444 445

445 446

446 447

447 448

448 449

449 450

450 451

451 452

452 453

453 454

454 455

455 456

456 457

457 458

458 459

459 460

460 461

461 462

462 463

463 464

464 465

465 466

466 467

467 468

468 469

469 470

470 471

471 472

472 473

473 474

474 475

475 476

476 477

477 478

478 479

479 480

480 481

481 482

482 483

483 484

484 485

485 486

486 487

487 488

488 489

489 490

490 491

491 492

492 493

493 494

494 495

495 496

496 497

497 498

498 499

499 500

500 501

501 502

502 503

503 504

504 505

505 506

506 507

507 508

508 509

509 510

510 511

511 512

512 513

513 514

514 515

515 516

516 517

517 518

518 519

519 520

520 521

521 522

522 523

523 524

524 525

525 526

526 527

527 528

528 529

529 530

530 531

531 532

532 533

533 534

534 535

535 536

536 537

537 538

538 539

539 540

540 541

541 542

542 543

543 544

544 545

545 546

546 547

547 548

548 549

549 550

550 551

551 552

552 553

553 554

554 555

555 556

556 557

557 558

558 559

559 560

560 561

561 562

562 563

563 564

564 565

565 566

566 567

567 568

568 569

569 570

570 571

571 572

572 573

573 574

574 575

575 576

576 577

577 578

578 579

579 580

580 581

581 582

582 583

583 584

584 585

585 586

586 587

587 588

588 589

589 590

590 591

591 592

592 593

593 594

594 595

595 596

596 597

597 598

598 599

599 600

600 601

601 602

602 603

603 604

604 605

605 606

606 607

607 608

608 609

609 610

610 611

611 612

612 613

613 614

614 615

615 616

616 617

617 618

618 619

619 620

620 621

621 622

622 623

623 624

624 625

625 626

626 627

627 628

628 629

629 630

630 631

631 632

632 633

633 634

634 635

635 636

636 637

637 638

638 639

639 640

640 641

641 642

642 643

643 644

644 645

645 646

646 647

647 648

648 649

649 650

650 651

651 652

652 653

653 654

654 655

655 656

656 657

657 658

658 659

659 660

660 661

661 662

662 663

663 664

664 665

665 666

666 667

667 668

668 669

669 670

670 671

671 672

672 673

673 674

674 675

675 676

676 677

677 678

678 679

679 680

680 681

681 682

682 683

683 684

684 685

685 686

686 687

687 688

688 689

689 690

690 691

691 692

692 693

693 694

694 695

695 696

696 697

697 698

698 699

699 700

700 701

701 702

702 703

703 704

704 705

705 706

706 707

707 708

708 709

709 710

710 711

711 712

712 713

713 714

714 715

715 716

716 717

717 718

718 719

719 720

720 721

721 722

722 723

723 724

724 725

725 726

726 727

727 728

728 729

729 730

730 731

731 732

732 733

733 734

734 735

735 736

736 737

737 738

738 739

739 740

740 741

741 742

742 743

743 744

744 745

745 746

746 747

747 748

748 749

749 750

750 751

751 752

752 753

753 754

754 755

755 756

756 757

757 758

758 759

759 760

760 761

761 762

762 763

763 764

764 765

765 766

766 767

767 768

768 769

769 770

770 771

771 772

772 773

773 774

774 775

775 776

776 777

777 778

778 779

779 780

780 781

781 782

782 783

783 784

784 785

785 786

786 787

787 788

788 789

789 790

790 791

791 792

792 793

793 794

794 795

795 796

796 797

797 798

798 799

799 800

800 801

801 802

802 803

803 804

804 805

805 806

806 807

807 808

808 809

809 810

810 811

811 812

812 813

813 814

814 815

815 816

816 817

817 818

818 819

819 820

820 821

821 822

822 823

823 824

824 825

825 826

826 827

827 828

828 829

829 830

830 831

831 832

832 833

833 834

834 835

835 836

836 837

837 838

838 839

839 840

840 841

841 842

842 843

843 844

844 845

845 846

846 847

847 848

848 849

849 850

850 851

851 852

852 853

853 854

854 855

855 856

856 857

857 858

858 859

859 860

860 861

861 862

862 863

863 864

864 865

865 866

866 867

867 868

868 869

869 870

870 871

871 872

872 873

873 874

874 875

875 876

876 877

877 878

878 879

879 880

880 881

881 882

882 883

883 884

884 885

885 886

886 887

887 888

888 889

889 890

890 891

891 892

892 893

893 894

894 895

895 896

896 897

897 898

898 899

899 900

900 901

901 902

902 903

903 904

904 905

905 906

906 907

907 908

908 909

909 910

910 911

911 912

912 913

913 914

914 915

915 916

916 917

917 918

918 919

919 920

920 921

921 922

922 923

923 924

924 925

925 926

926 927

927 928

928 929

929 930

930 931

931 932

932 933

933 934

934 935

935 936

936 937

937 938

938 939

939 940

940 941

941 942

942 943

943 944

944 945

945 946

946 947

947 948

948 949

949 950

950 951

951 952

952 953

953 954

954 955

955 956

956 957

957 958

958 959

959 960

960 961

961 962

962 963

963 964

964 965

965 966

966 967

967 968

968 969

969 970

970 971

971 972

972 973

973 974

974 975

975 976

976 977

977 978

978 979

979 980

980 981

981 982

982 983

983 984

984 985

985 986

986 987

987 988

988 989

989 990

990 991

991 992

992 993

993 994

994 995

995 996

996 997

997 998

998 999

999 1000

HelloWorld - HelloWorldViewController.m

Run Stop Scheme Breakpoints Editor View Organizer

HelloWorld 2 targets, iOS SDK 4.3

HelloWorld

- HelloWorldAppDelegate.h
- HelloWorldAppDelegate.m
- MainWindow.xib
- HelloWorldViewController.h
- HelloWorldViewController.m**
- HelloWorldViewController.xib
- Supporting Files
 - HelloWorld-Info.plist
 - InfoPlist.strings
 - HelloWorld-Prefix.pch
 - main.m

HelloWorldTests

Frameworks

- UIKit.framework
- Foundation.framework
- CoreGraphics.framework

Products

- HelloWorld.app
- HelloWorldTests.octest

// HelloWorldViewController.m
// HelloWorld
//
// Created by Michael Rohs on 3.5.2011.
// Copyright 2011 __MyCompanyName__. All rights reserved.

#import "HelloWorldViewController.h"

@implementation HelloWorldViewController

@synthesize label;

-(IBAction) buttonPressed:(id)sender;
{
 NSLog(@"button pressed");
 label.text = @"button pressed";
}

- (void)dealloc
{
 [label dealloc];
 [super dealloc];
}

- (void)didReceiveMemoryWarning
{
 // Releases the view if it doesn't have a superview.
 [super didReceiveMemoryWarning];

 // Release any cached data, images, etc that aren't in use.
}

#pragma mark - View lifecycle

/*
// Implement viewDidLoad to do additional setup after loading the view,
// typically from a nib.
- (void)viewDidLoad
{
 [super viewDidLoad];
}
*/

Quick Help

HelloWorldViewController :
UIViewController
Name: [UIViewController](#)
Availability: iOS (2.0 and later)
Abstract: The UIViewController class provides the fundamental view-management model for iPhone applications. The basic view controller class supports the presentation of an associated view, support for managing modal views, and support for rotating views in response to device orientation changes. Subclasses such as UINavigationController and UITabBarController provide additional behavior for managing complex hierarchies of view controllers and views.
Declared In: [UIViewController.h](#)
Reference: [UIViewController Class Reference](#)
Related Documents: [View Controller Programming Guide for iOS](#)
Sample Code: [CopyPasteTile](#), [ListAdder](#), [NavBar](#), [SimpleNetworkStreams](#), [iPhoneCoreDataRecipes](#)

Objects

Label	1	2	Text
Switch	On	Off	Toggle
Image	Image 1	Image 2	Image 3
Color	Color 1	Color 2	Color 3

Michael Rohs, LMU Praktikum Mediensysteme – iOS WS 2011 53

HELLO TABLEVIEW

UITableView Example

- Create new project (“View-based application”)
- Change controller base class to UITableViewController
- Declare UITableViewDataSource, UITableViewDelegate
- Add data array to header file, release data in dealloc
- Change view in nib file to UITableView, connect File’s Owner (view, data source, delegate)
- Create arrayWithObjects in viewDidLoad
- Implement table data source and delegate methods

HELLO MULTIVIEW

View Navigation Example

- Create a “Navigation-Based Application”
- Add NSArray *data to RootViewController
 - Add some data in viewDidLoad, retain!
- New File... → UIViewController subclass (with nib file) → “MyDetailViewController”
 - Add UILabel to nib file and to .h file (IBOutlet, @property) and to .m file (@synthesize)
- #import "MyDetailViewController.h"
- Implement didSelectRowAtIndexPath, set selected item
- Show that it does not work ☺ → Debugger
- Show that label is still nil → use member variable, set label in viewDidLoad

View Navigation Example

- Add back button: `self.navigationItem.title = @"List";`