Object-Oriented Programming and User Interfaces

COSC346

Instructors

- Paul Crane (OOP)
  - Owheo 2.50
  - pcrane@cs.otago.ac.nz
- Hamza Bennani (UI)
  - Owheo 2.51
  - hamza@cs.otago.ac.nz

ousa
otago uni students' association

SEMESTER TWO 2018

Are you

- proactive, friendly and keen to contribute to your learning environment?
- a great communicator who can represent your peers?

What's in it for you?

- Kudos & Karma
- Great friendships
- Access to FREE professional training opportunities and support
- A feed (or three)
- A reference letter from OUSA for your CV
- Invitations to Class Rep social events throughout the year

Don't wait any longer... sign up now!

Talk to your lecturer or email:

classrep@ousa.org.nz

Schedule

- **Lectures:**
  - Tuesday 13:00–13:50, UOCE Tower Block Room G08, (TG08)
  - Thursday 13:00–13:50, UOCE Tower Block Room G08, (TG08)
- **Labs** (separate streams):
  - Wednesday 10:00 – 11:50, Owheo G.38 (Lab F)
  - Wednesday 12:00 – 13:50, Owheo G.38 (Lab F)
  - There **is a lab** in the first week
- **Tutorials** (separate streams):
  - Tuesday 10:00 – 10:50, Richardson, RGS2
  - Friday 10:00 – 10:50, Castle, Room C
  - **Run as requested**

Assessment

- Assignment 1: 20%, due Monday Sep 3rd
- Assignment 2: 20%, due Friday Oct 5th
- Final Exam: 60%

Course Overview: Lectures

	Date	Title	Reading	Example code
1	Tuesday Jul 10th	Course overview (PC/HB)		
2	Thursday Jul 12th	Introduction to Swift (PC)		
3	Tuesday Jul 17th	Classes and objects (PC)		
4	Thursday Jul 19th	Working with objects (PC)		
5	Tuesday Jul 24th	Inheritance I (PC)		
6	Thursday Jul 26th	Inheritance II (PC)		
7	Tuesday Jul 31st	Polymorphism (PC)		
8	Thursday Aug 2nd	Memory management (PC)		
9	Tuesday Aug 7th	Object interconnections (PC)		
10	Thursday Aug 9th	Swift Libraries (PC)		
11	Tuesday Aug 14th	Object oriented design (PC)		
12	Thursday Aug 16th	Object oriented design patterns (PC)		
13	Tuesday Aug 21st	OOP review (PC)		
14	Thursday Aug 23rd	Introduction to UI (PC)		
	Study break			
	Assignment 1 due, Monday, Sep 3rd			
15	Tuesday Sep 4th	Application programming on the Mac (HB)		
16	Thursday Sep 6th	Model View Controller (HB)		
17	Tuesday Sep 11th	Cocoa: Windows and Views (HB)		
18	Thursday Sep 13th	Cocoa: Multiple windows (HB)		
19	Tuesday Sep 18th	Cocoa: Mouse and Keyboard Events (HB)		
20	Thursday Sep 20th	Cocoa: Bindings (HB)		
21	Tuesday Sep 25th	Cocoa: Controllers and Undo (HB)		
22	Thursday Sep 27th	Cocoa: Preferences (HB)		
23	Tuesday Oct 2nd	UI design (HB)		
24	Thursday Oct 4th	Usability and visual design (HB)		
	Assignment 2 due, Friday, Oct 5th			
25	Tuesday Oct 9th	UI review (HB)		
26	Thursday Oct 11th	TBC		

- Object Oriented Programming
  - General concepts: abstraction, encapsulation, inheritance, polymorphism, coupling, cohesion
  - Swift language and Foundation Framework
  - Swift development tools - Xcode
  - Object oriented design principles
- User Interfaces
  - Cocoa Environment and Xcode
  - Interface design principles: usability, basics of graphic design

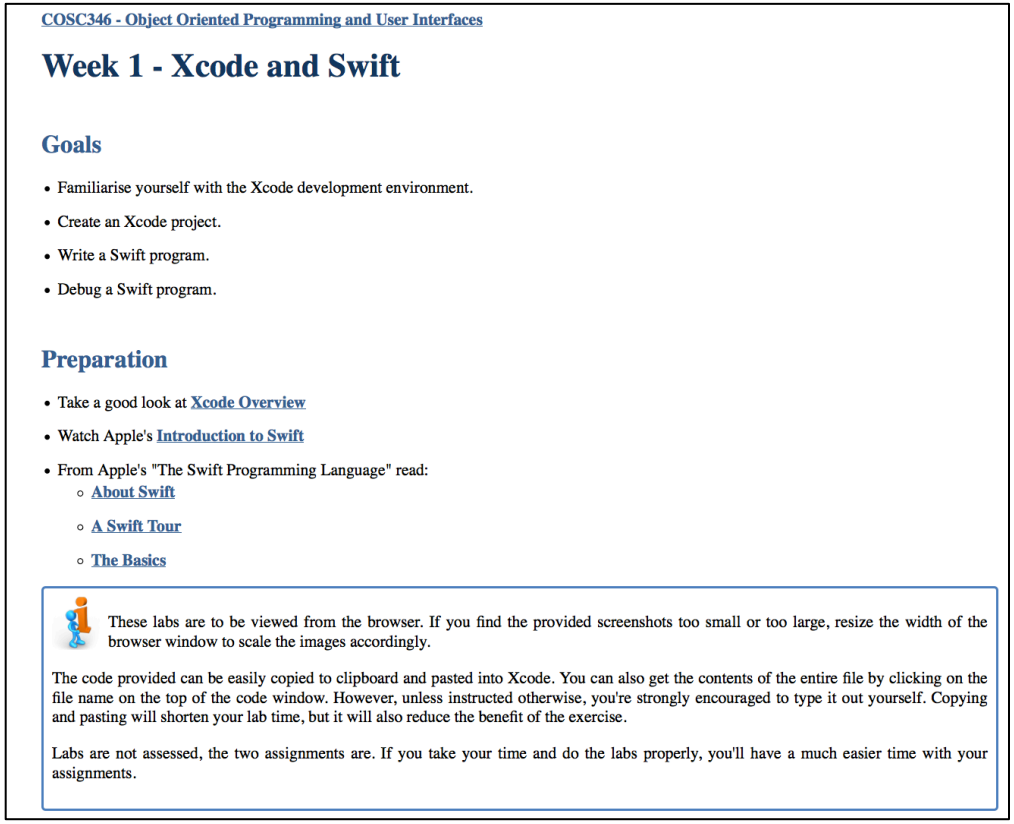
Everything you need will be in the lectures

OOP

Swift

Course Overview: Labs

- On the course webpage
- Not assessed
- First lab tomorrow



COSC346 - Object Oriented Programming and User Interfaces

Week 1 - Xcode and Swift

Goals

- Familiarise yourself with the Xcode development environment.
- Create an Xcode project.
- Write a Swift program.
- Debug a Swift program.

Preparation

- Take a good look at Xcode Overview
- Watch Apple's Introduction to Swift
- From Apple's "The Swift Programming Language" read:
  - About Swift
  - A Swift Tour
  - The Basics

These labs are to be viewed from the browser. If you find the provided screenshots too small or too large, resize the width of the browser window to scale the images accordingly.

The code provided can be easily copied to clipboard and pasted into Xcode. You can also get the contents of the entire file by clicking on the file name on the top of the code window. However, unless instructed otherwise, you're strongly encouraged to type it out yourself. Copying and pasting will shorten your lab time, but it will also reduce the benefit of the exercise.

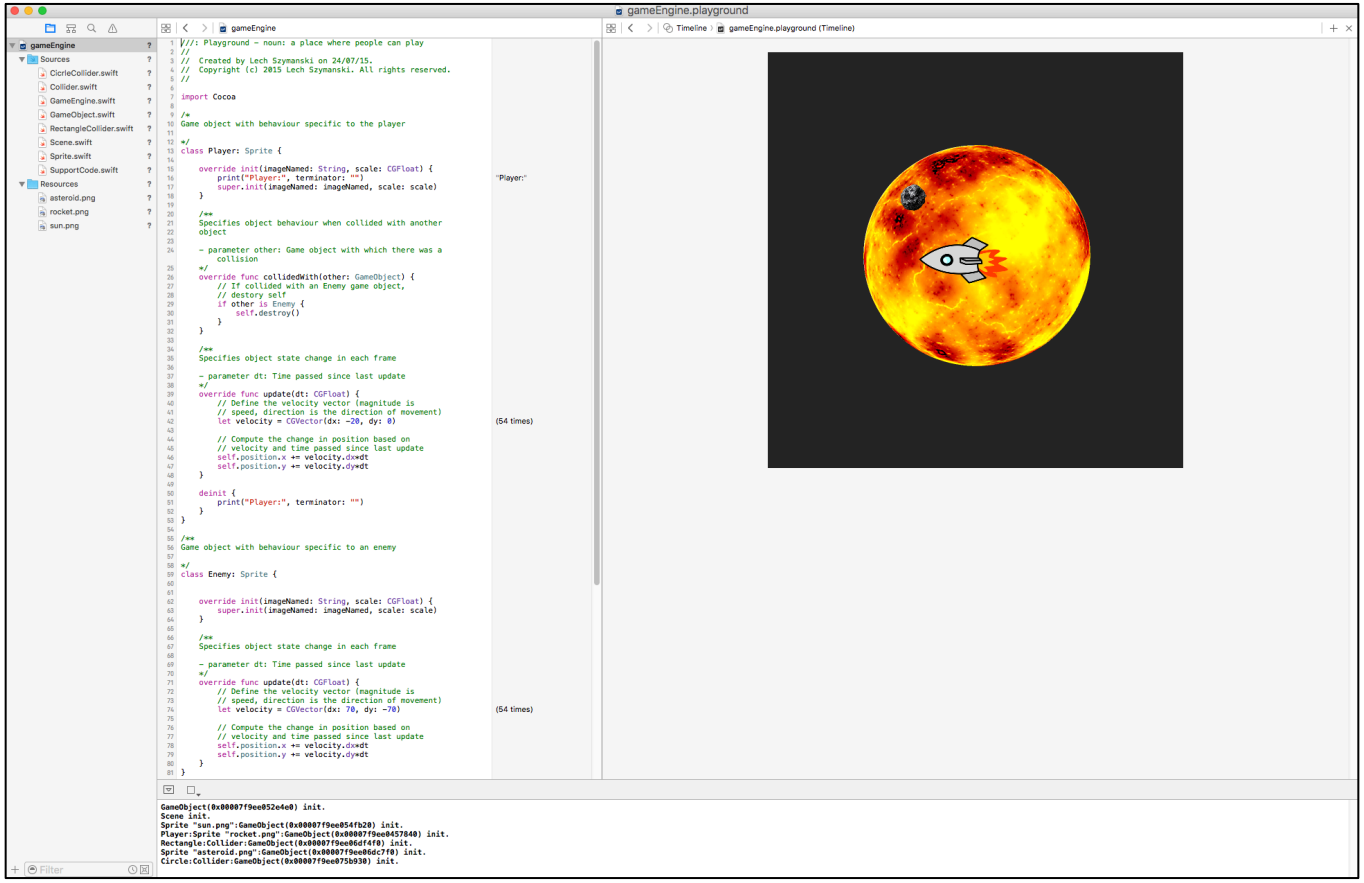
Labs are not assessed, the two assignments are. If you take your time and do the labs properly, you'll have a much easier time with your assignments.

Course Overview: Git

- Labs and Assignments distributed via GitHub repositories
  - https://github.com (create an account here)
  - https://education.github.com/pack/offers
- Check email & website
  - email will be sent out before the start of the lab
- https://www.katacoda.com/courses/git
  - interactive git tutorial
- https://guides.github.com
  - various guides for git/github

Course Overview: Tutorials

- As needed



Course Overview: Plagiarism

- We place a high value on Academic Integrity and treat Academic Misconduct seriously
- The work you submit for assessment **must** be your own

In short: **Don't copy others' work!**

- Academic Integrity - A Brief Guide for Students

Readings

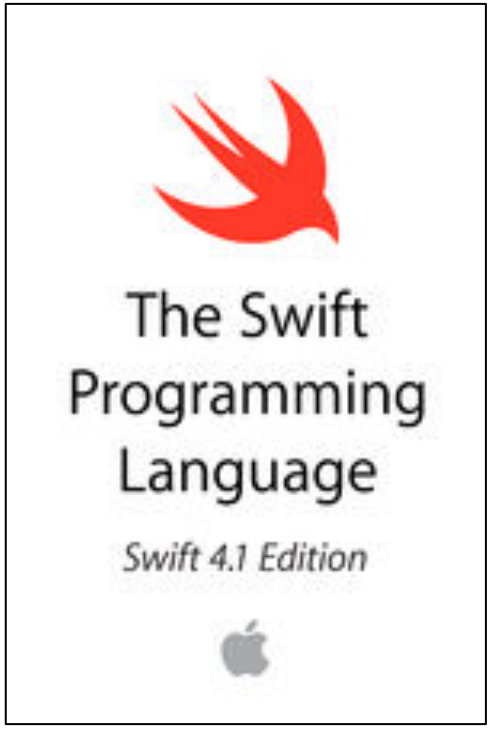
- Lots of material to use
- We expect you to do reading to figure out problems
  - lots of versions of swift that break old versions
  - take care to use the proper version
- We're not going to hand-hold

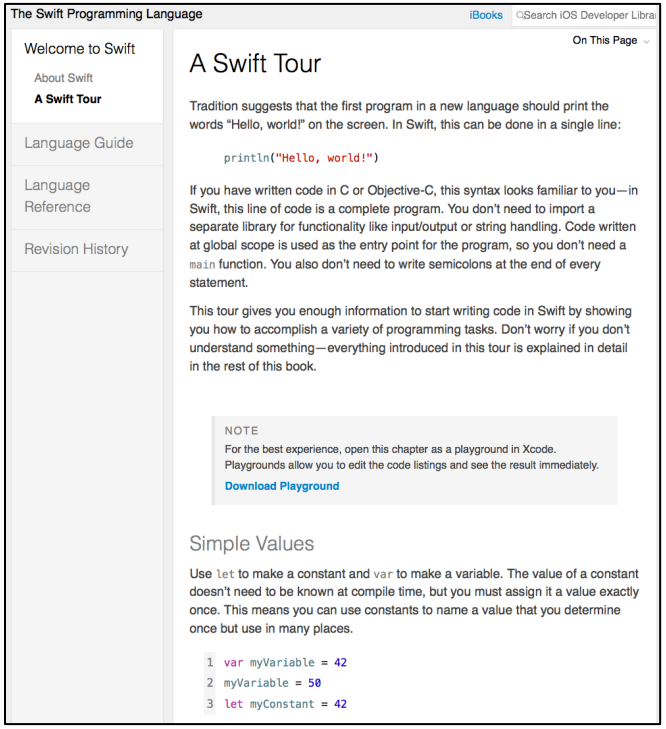
Reading

The Swift Programming Language (2017)

Apple Inc.

- Up to date
- **Free**
- HTML, iBook format





https://docs.swift.org/swift-book/GuidedTour/GuidedTour.html

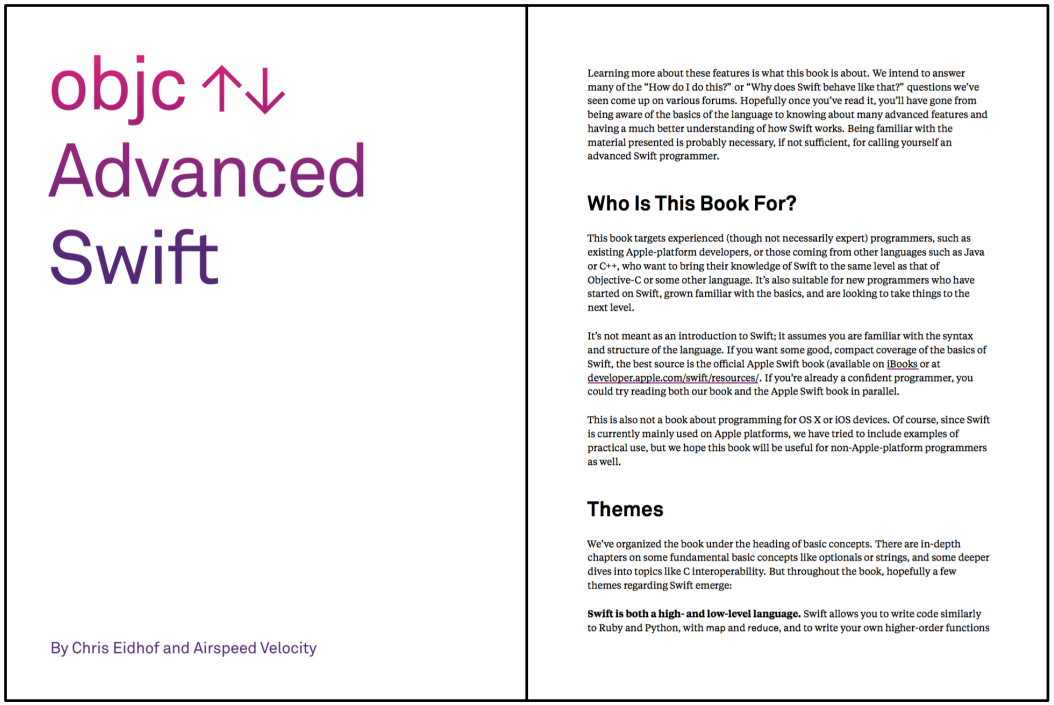
Reading

Advanced Swift (2016)

C. Eidhof, A. Velocity

Objc.io

- For programmers that come from other languages (such as Java)
- E-book formats: PDF, mobi



objc ↑↓
Advanced
Swift

By Chris Eidhof and Airspeed Velocity

Learning more about these features is what this book is about. We intend to answer many of the "How do I do this?" or "Why does Swift behave like that?" questions we've seen come up on various forums. Hopefully once you've read it, you'll have gone from being aware of the basics of the language to knowing about many advanced features and having a much better understanding of how Swift works. Being familiar with the material presented is probably necessary, if not sufficient, for calling yourself an advanced Swift programmer.

Who Is This Book For?

This book targets experienced (though not necessarily expert) programmers, such as existing Apple-platform developers, or those coming from other languages such as Java or C++, who want to bring their knowledge of Swift to the same level as that of Objective-C or some other language. It's also suitable for new programmers who have started on Swift, grown familiar with the basics, and are looking to take things to the next level.

It's not meant as an introduction to Swift; it assumes you are familiar with the syntax and structure of the language. If you want some good, compact coverage of the basics of Swift, the best source is the official Apple Swift book (available on iBooks or at developer.apple.com/swift/resources/. If you're already a confident programmer, you could try reading both our book and the Apple Swift book in parallel.

This is also not a book about programming for OS X or iOS devices. Of course, since Swift is currently mainly used on Apple platforms, we have tried to include examples of practical use, but we hope this book will be useful for non-Apple-platform programmers as well.

Themes

We've organized the book under the heading of basic concepts. There are in-depth chapters on some fundamental basic concepts like optionals or strings, and some deeper dives into topics like C interoperability. But throughout the book, hopefully a few themes regarding Swift emerge:

Swift is both a high- and low-level language. Swift allows you to write code similarly to Ruby and Python, with map and reduce, and to write your own higher-order functions

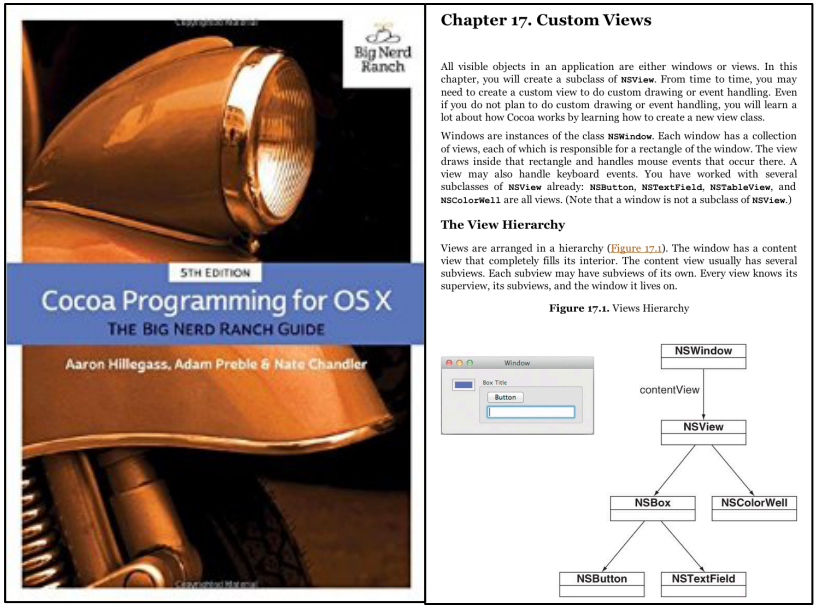
Reading

Cocoa Programming for Mac OS X, 5th ed. (2015)

A. Hillegass, A. Preble, N. Chandler

Big Nerd Ranch Guides

- Written for Xcode 6 and Swift 2.x
- Excellent examples
- Still probably one of the best books on Cocoa development
- Hardcopy in the lab



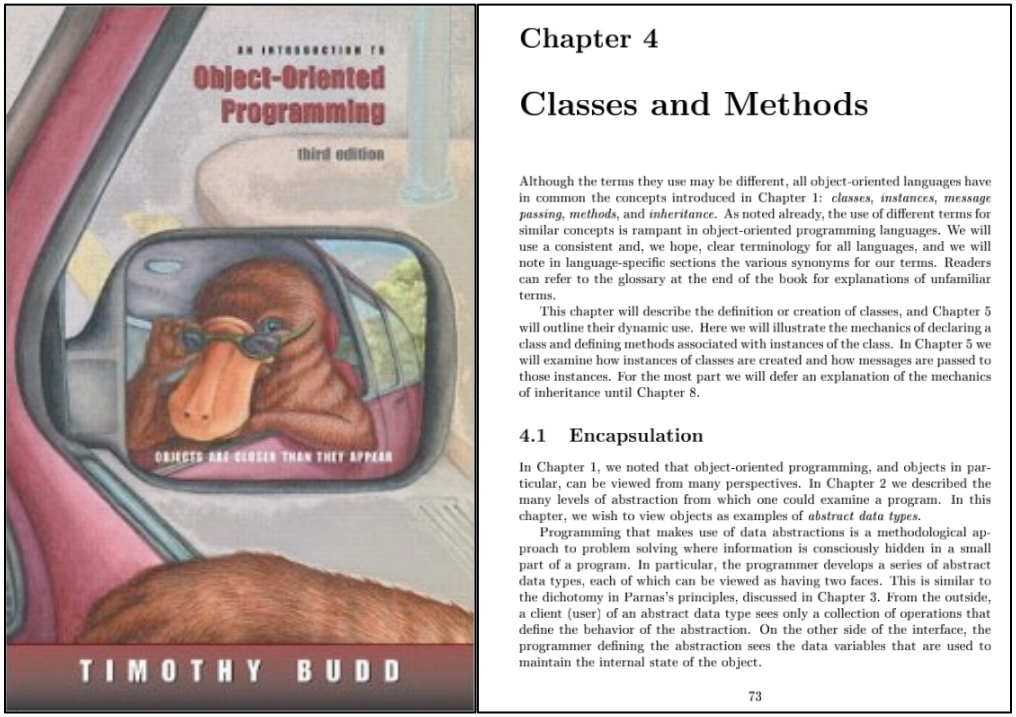
Reading

Object-Oriented Programming, 3rd ed. (2002)

Timothy Budd

Addison-Wesley

- General OOP principles
- Hardcopy in Science Library on reserve



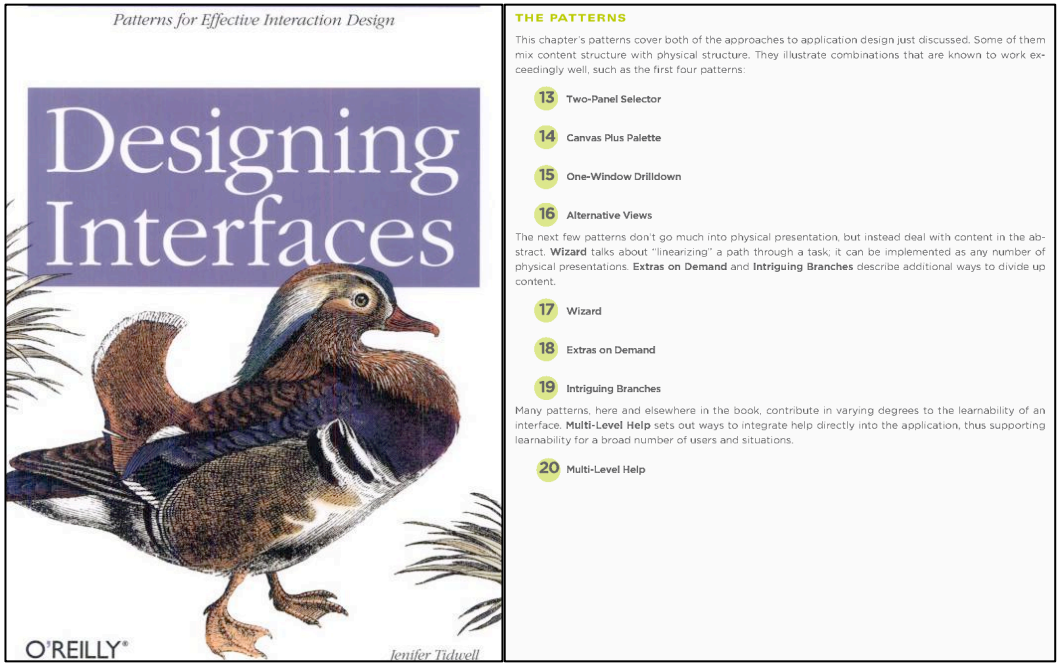
Reading

Designing Interfaces (2002)

Jenifer Tidwell

O'Reilly Media Inc.

- User Interface principles and design patterns
- Electronic version from the Science Library



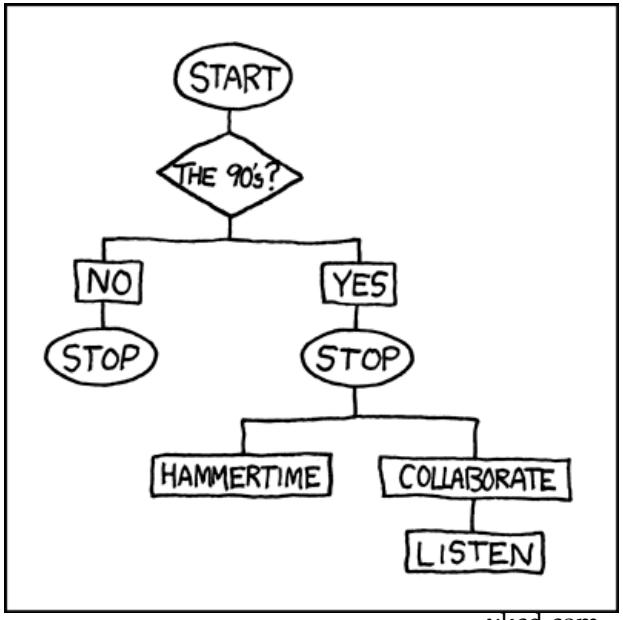
Reading

- The Swift Programming Language (2017), Apple Inc.
- C. Eidhof, A. Velocity (2016), Advanced Swift, Objc.io.
- A. Hillegass, A. Preble, N. Chandler (2015), *Cocoa Programming for Mac OS X (**5th ed**)*, Big Nerd Ranch Guides.
- Timothy Budd (2002), *Object-Oriented Programming (3rd ed)*, Addison-Wesley.
- Jenifer Tidwell (2006), Designing Interfaces, O'Reilly Media, Inc.

We'll give you the material you'll need for the course

What is OOP?

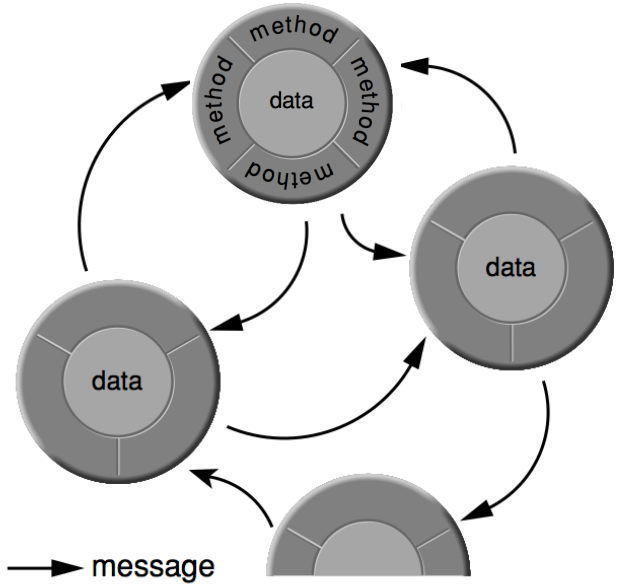
Procedural



xkcd.com

1. Functions act on data.
2. A program organises function calls to manipulate data.

Object-Oriented

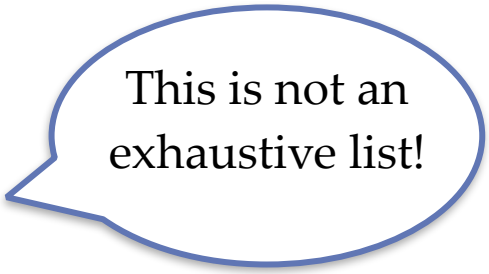


1. *Objects* contain *encapsulated* data and associated *methods*.
2. A program describes how objects interact via *messages*.

What is OOP?

Some other programming paradigms:

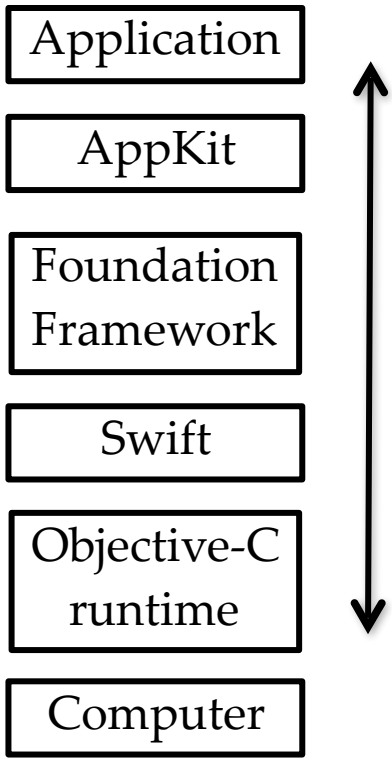
- Imperative
  - directly change computed state
- Declarative
  - defines logic but not control flow
- Functional
  - mathematical functions avoiding state and mutable data
- Event Driven
  - control flow is determined by events



What is OOP?

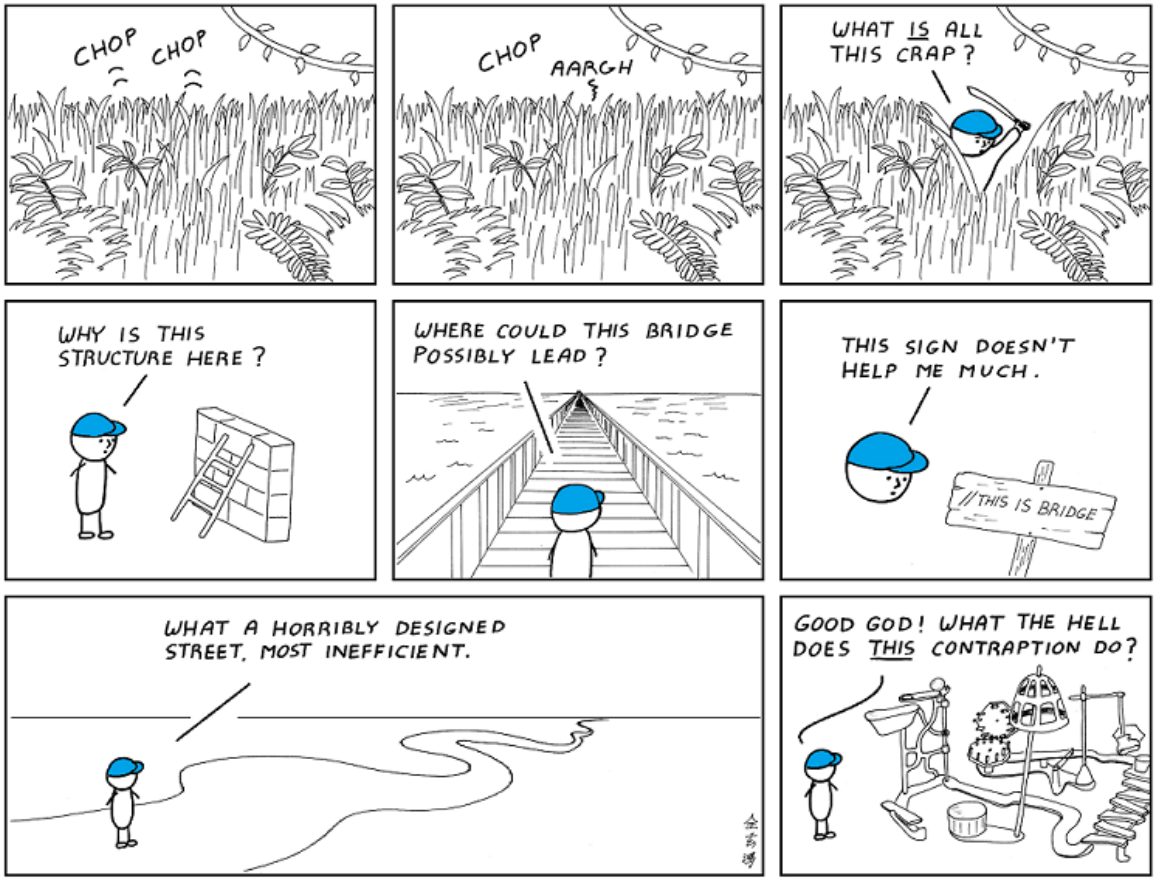
vs.

Why OOP?



Speed	Code	Development	Environment	User Interface
Slow	**Re-usable**	**Team, Fast**	**Runtime Decisions**	**Complex, Graphical**
Fast	**Specific**	**Individual, Slow**	**Compile-time Decisions**	**Simple, Text-based**

Reusability



I hate reading
other people's code.

abstrusegoose.com

Why Swift?

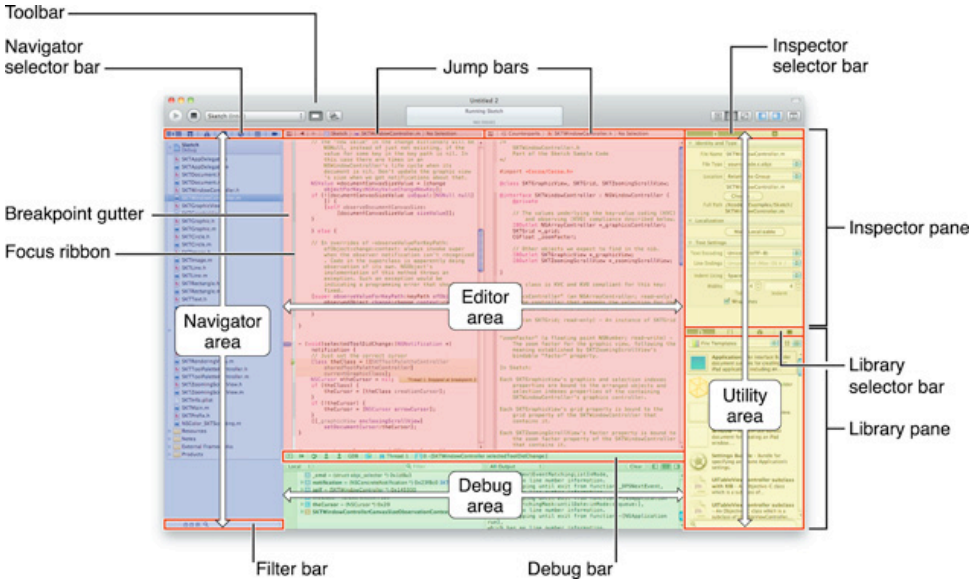
- Modern
  - Result of research on programming languages
  - Multi-paradigm – takes best features from many languages (in COSC346 we focus on the Object-Oriented aspect)
- Safe
  - Compiler forces you to do things right
  - Tries to detect errors at compile time, not run-time
- Concise
  - Easier and faster to develop software
  - Easier to create development tools
- Cocoa environment – good example of natural progression from OOP to User Interfaces

What is Cocoa?

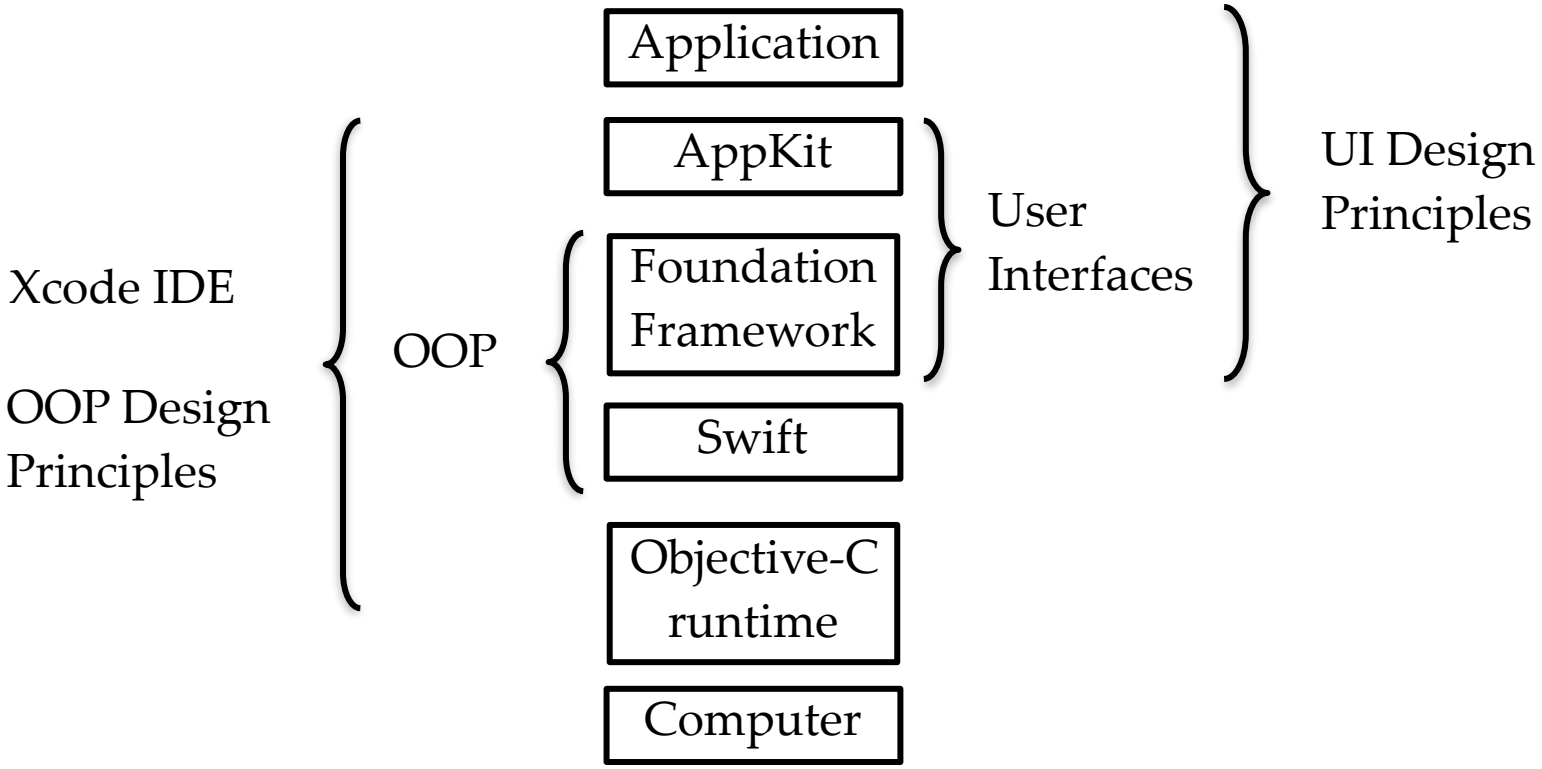
- Object-oriented framework for application development for macOS and iOS
  - In this course we will focus on macOS only
- "Its elegant and powerful design is ideally suited for the rapid development of software"
  – Cocoa Fundamentals Guide (2010, retired), Apple Inc.
- Huge number of classes and frameworks
  - Overwhelming for the first-time user
  - Powerful environment that abstracts away a lot of the details of application programming – you can concentrate on high level functionality

What is Xcode?

- Integrated Development Environment (IDE) for application development for macOS & iOS
  - It comes with iOS platform simulator
- Compiler and debugging tools
- Cocoa libraries and frameworks
  - Interface builder - GUI for building GUIs
- Editor and tools for analysis



Mac Platform



Goals

- Object-Oriented Programming:
  - (a) Learn Swift language
  - (b) Understand OOP design principles
- User Interfaces:
  - (a) Learn Application Kit Framework
  - (b) Understand UI design principles