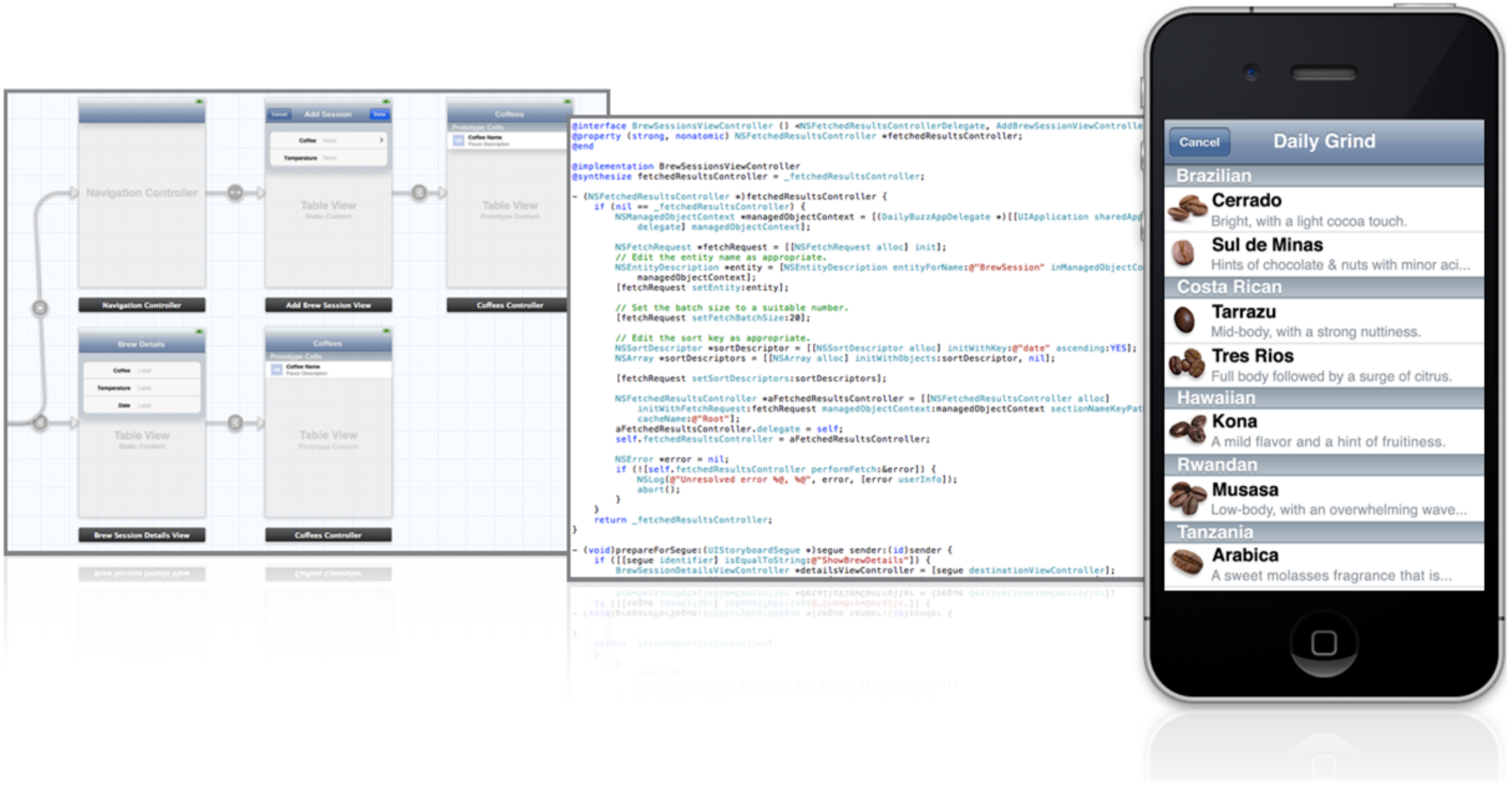


Mobile App Development 1
App Design

Robert O'Connor
roconnor@wit.ie
@roconnorwit

Outline

- App Design
- Know The Core Objects of Your App
- Defining a Document-based App
- The User Interface

App Design

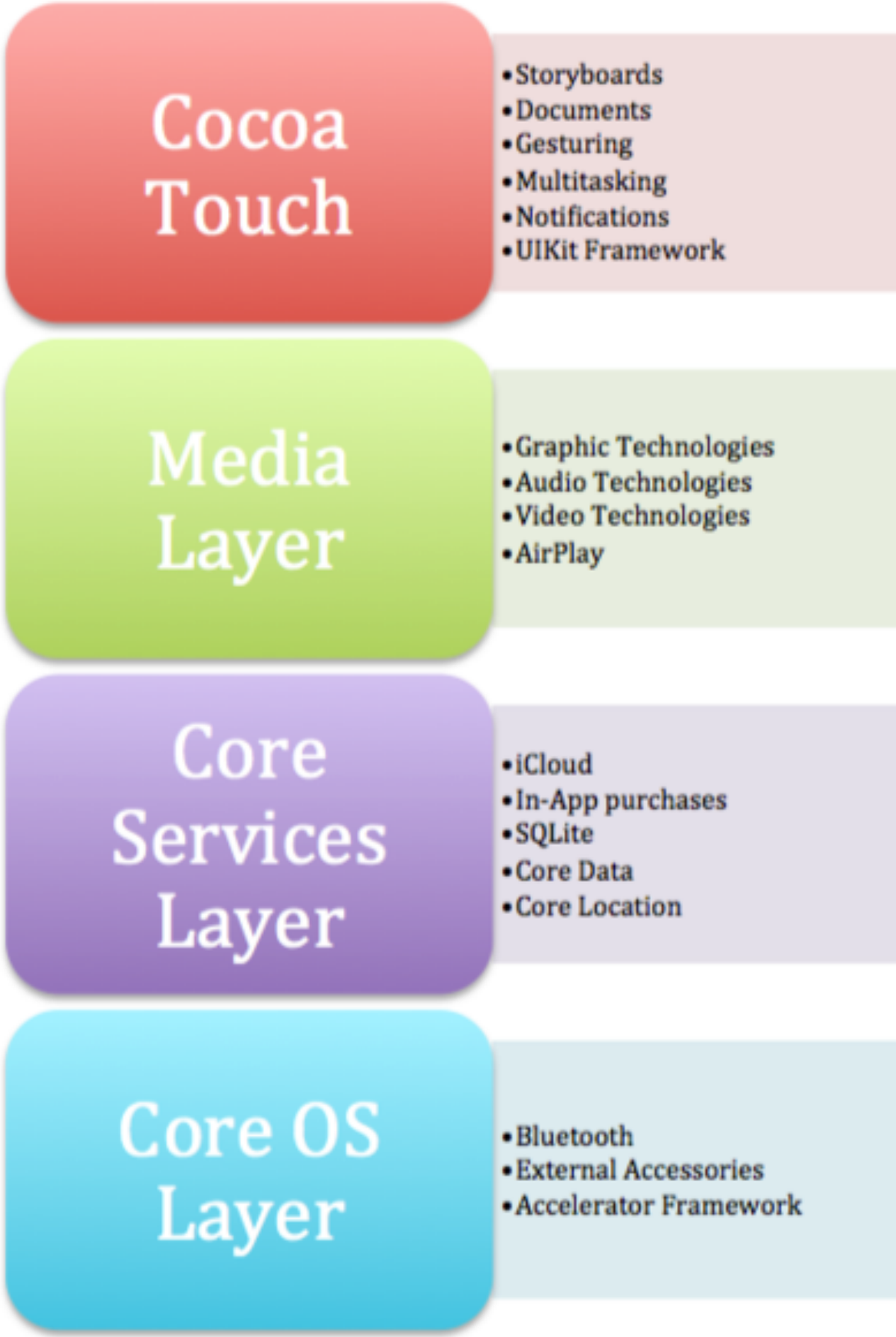
- Before writing any code, you must ask yourself some fundamental questions:
- What is the App's purpose?
- What are the App's features?
- What kind of data model will the App use?
- Is it a universal App?

App Design

- You also need to become familiar with the Cocoa Touch/ Android SDK frameworks.

- Your app depends on the objects you can create with these frameworks.

- Get to know the framework objects that implement the basic structure of an app, that serve as the building blocks of your data model, and that compose the unique experience your app presents to users.



App Design

- An app that is well designed has features that users find **appealing**, **appropriate**, and **useful**.






Design Your App With Care

- App Design does not *just* mean user interface design

- App Design does not *just* mean code

- It begins with some high-level information and ideas

- You *then* think about how to best implement these ideas

Do Your Initial Design

- A good app starts with an idea that is expanded to more fully-featured product description

- Early in the design phase, it helps to understand just **WHAT** you want your app to do (not ***how*** you will do it)

- Write down the set of high-level features that would be required to implement your idea.

- Prioritise those features based on what you think your users will need.

- Do a little research into iOS itself so that you understand its capabilities and how you might be able to use them to achieve your goals.

Do You Initial Design

- Sketch out some rough interface designs on paper to visualise how your app might look
- A pen and paper may be most advanced tech you need at this point
- There are lots of app design tools out there to assist you. For example:
  - PSD image sets
  - Balsamiq Mockups http://www.balsamiq.com/products/mockups
  - Fluid UI https://www.fluidui.com/

Do You Initial Design

- The goal of your initial design is to answer some very important questions about your app.

- The set of features and the rough design of your interface help you think about what will be required later when you start writing code.

- At some point, you need to translate the information displayed by your app into a set of data objects.

- Similarly, the look of your app has an overwhelming influence on the choices you must make when implementing your user interface code.

- Doing your initial design on paper (rather than on the computer) gives you the freedom to come up with answers that are not limited by what is easy to do.

Create an Action Plan

- iOS assumes that all apps are built using the **Model-View-Controller** design pattern. Therefore, the first steps you can take toward achieving this goal are to choose approaches for the data and view portions of your app.

1. Choose a basic approach for your data model

2. Decide whether you need support for documents

3. Choose an approach for your user interface

Android calls this **Model-View-Presenter**

Create an Action Plan

1.Choose a basic approach for your data model:

- **Existing data model code**—If you already have data model code written in a C-based language, you can integrate that code directly into your iOS apps. C and C++ can be directly integrated into XCode/Android Studio projects

- **Custom objects data model**—A custom object typically combines some simple data (strings, numbers, dates, URLs, and so on) with the business logic needed to manage that data and ensure its consistency. Custom objects can store a combination of scalar values and pointers to other objects. For example, the Foundation framework defines classes for many simple data types and for storing collections of other objects. These classes make it much easier to define your own custom objects.

- **Structured data model**—If your data is highly structured—that is, it lends itself to storage in a database—use Core Data or SQLite to store the data. Core Data provides built-in support for some advanced features like undo and iCloud. (SQLite files cannot be used in conjunction with iCloud.) Firebase is a cloud-based data storage option.

Create an Action Plan

2.Decide whether you need support for documents

- Within an iOS context, documents are NOT word processing files

- The job of a document is to manage your app's in-memory data model objects and coordinate the storage of that data in a corresponding file (or set of files) on disk.

- Documents normally connote files that the user created but apps can use documents to manage files that are not user facing too.

- One big advantage of using documents is that the UIDocument class makes interacting with iCloud and the local file system much simpler. For apps that use Core Data to store their content, the UIManagedDocument class provides similar support

Create an Action Plan

3.Choose an approach for your user interface

- **Building block approach**—The easiest way to create your user interface is to assemble it using existing view objects. Views represent visual elements such as tables, buttons, text fields, and so on. You use many views as-is but you can also customise the appearance and behavior of standard views as needed to meet your needs. You can also implement new visual elements using custom views and mix those views freely with the standard views in your interface. The advantages of views are that they provide a consistent user experience and they allow you to define complex interfaces quickly and with relatively little code.

- **OpenGL ES–based approach**—If your app requires frequent screen updates or sophisticated rendering, you probably need to draw that content directly using OpenGL ES. The main use of OpenGL ES is for games and apps that rely heavily on sophisticated graphics and therefore need the best performance possible

Start the App Creation Process

- When starting to program in your IDE, you should have answers to the following questions in mind:

- **What is the basic interface style of your app?** Different types of app require different sets of initial views and view controllers. Knowing how you plan to organise your user interface lets you select an initial project template that is most suited to your needs. You can always change your user interface later, but choosing the most appropriate template first makes starting your project much easier.

- **Do you want to create a universal app or one targeted specifically for tablet or phone?** Creating a universal app requires specifying different sets of views and view controllers for iPad and iPhone and dynamically selecting the appropriate set at runtime. Universal apps are nice because they support more iOS devices but do require you to factor your code better for each platform. There is a similar setup in Android

Start the App Creation Process

- **Do you want your app to use storyboards?** Storyboards simplify the design process by showing both the views and view controllers of your user interface and the transitions between them. Storyboards are supported in iOS 5 and later and are enabled by default for new projects. If your app must run on earlier versions of iOS, though, you cannot use storyboards and should continue to use nib files.

- **What do you want to use for your data model?** Some types of apps lend themselves naturally to a structured data model, which makes them ideal candidates for using Realm, Core Data or SQLite.

Start the App Creation Process

- The following phases of app development are common:

1. Start writing your app's primary code
2. Add support for app state changes
3. Create the resources needed to support your app
4. As needed, implement any app-specific behaviours that are relevant for your app
5. Add the advanced features that make your app unique
6. Do some basic performance tuning for your app
7. Iterate

Start the App Creation Process

- **Start writing your app's primary code**. For new apps, you probably want to start creating the classes associated with your app's data model first. These classes usually have no dependencies on other parts of your app and should be something you can work on initially. You might also want to start playing around with designs for your user interface by adding views to your main storyboard or nib file. From these views, you can also start identifying the places in your code where you need to respond to interface-related changes. If your app supports iCloud, you should incorporate support for iCloud into your classes at an early stage.

- **Add support for app state changes**. The state of an app determines what it is allowed to do and when. App states are managed by high-level objects in your app but can affect many other objects as well, therefore, you need to consider how the current app state affects your data model and view code and update that code appropriately.

Start the App Creation Process

- **Create the resources needed to support your app.** Apps submitted to the App Store are expected to have specific resources such as icons and launch images to make the overall user experience better. Well-factored apps also make heavy use of resource files to keep their code separate from the data that code manipulates. This factoring makes it much easier to localise your app, tweak its appearance, and perform other tasks without rewriting any code.

- **As needed, implement any app-specific behaviors that are relevant for your app.** There are many ways to modify the way your app launches or interacts with the system. For example, you might want to implement local notifications for a certain feature.

Start the App Creation Process

- **Add the advanced features that make your app unique**. iOS and Android includes many other frameworks for managing multimedia, advanced rendering, game content, maps, contacts, location tracking, and many other advanced features. iOS Technology Overview gives an overview of the frameworks and features you can incorporate into your apps.

- **Do some basic performance tuning for your app**. All apps should be tuned for the best possible performance. Tuned apps run faster but also use system resources, such as memory and battery life, more efficiently.

- **Iterate!** App development is an iterative process. As you add new features, you might need to revisit some or all of the preceding steps to make adjustments to your existing code.

Summary

- App Design
- Know The Core Objects of Your App
- Defining a Document-based App
- The User Interface
- App Creation Process

More Information

- iOS Technology Overview describes the frameworks and other technologies that are available to your app in iOS.

- iOS Human Interface Guidelines teaches you how to make your app consistent with the user-interface conventions for iOS.

- Developing for the App Store walks you through the process of developing apps, provisioning devices for testing, and submitting apps to the App Store.

- Programming with Objective-C describes how to define classes, send messages, encapsulate data, and accomplish a variety of other tasks with the Objective-C programming language.

- iOS App Programming Guide explains the essential things you must know and do when developing an iOS app.

Bibliography

- Aaron Hillegass, ‘Cocoa Programming for Mac OS X’ (3rd Edition), Addison Wellsey, 2008

- Joe Conway & Aaron Hillegass, ‘iOS Programming’ (3rd Edition), Big Nerd Ranch. 2012

- Alan Cannistraro, ‘iPhone Application Development’, Stanford University, iTunes U, 2010

- Apple Developer Programme. http://developer.apple.com

- Anand Mehta, ‘A Crash Course in C’, Belmonte University. 1995

- Ray Wenderlich, Tutorials for iOS Developers. www.raywenderlich.com