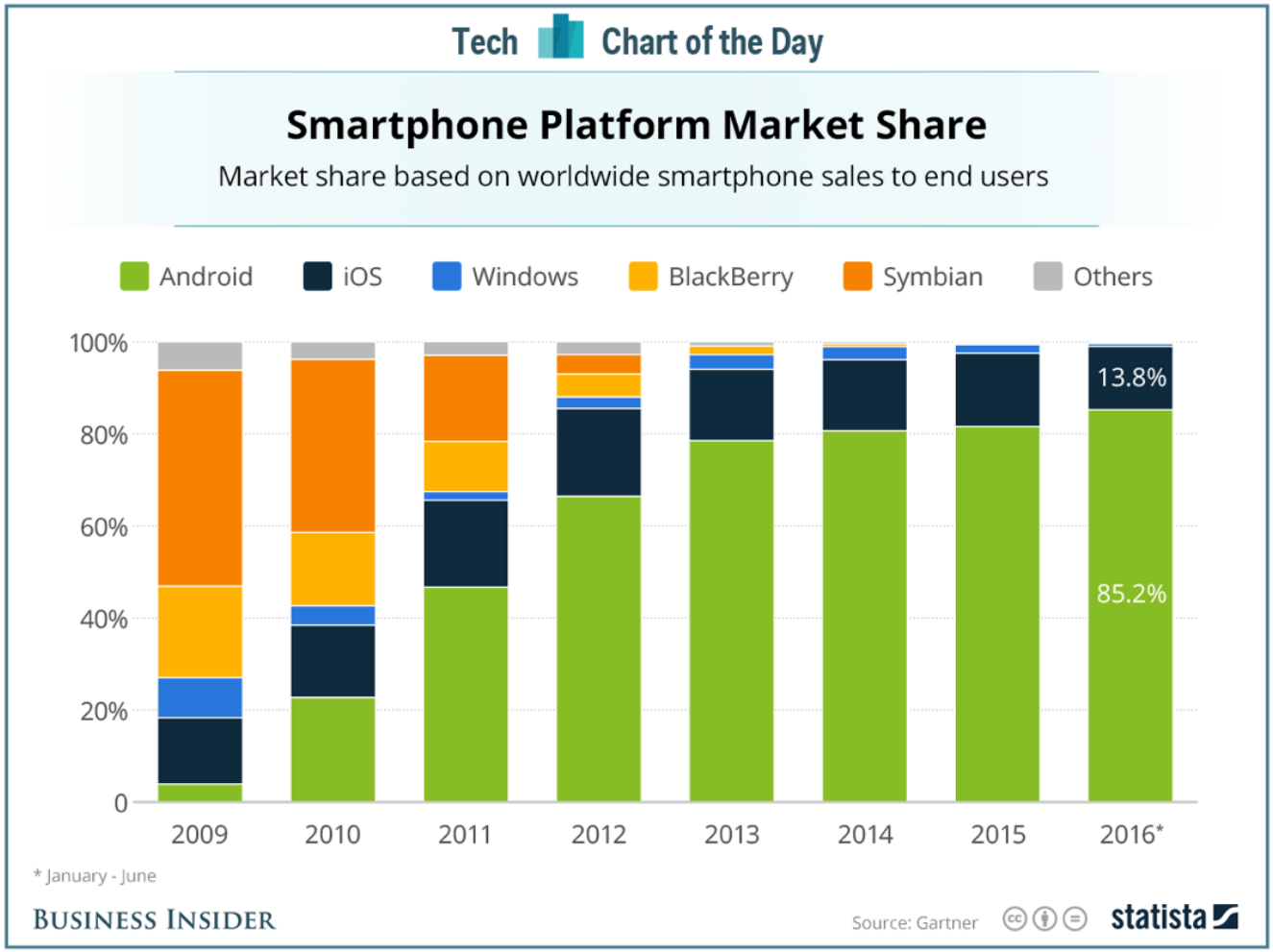
G53MDP
Mobile Device Programming

Lecture 20 – iOS, Cross-platform

Smartphone Platform Market Share



https://appsamurai.com/7-major-differences-between-ios-and-android-development/

Main Differences- Android vs. IOS

	Android	IOS
Market Share	82.8%	13.9%
Programming Language	Java Kotlin	Objective-C Swift (2014)
IDE	Eclipse (previously) Android Studio	XCode (now in version 7)
Operating Systems	Windows/Mac/Linux	Only on Mac!
Distribution Process	Release an App in hours	Release an App in weeks
Fees	$25 one-off	$100 per year

iOS

- OS of the iPhone/iPad/iPod Touch
  - Originally called iPhoneOS
  - Based (heavily) on MacOS X
- Closed Source
  - Tools, deployment, app ecosystem controlled by Apple
- Apps can only be installed from an App Store
  - Cryptographically signed
  - Apple runs iTunes App Store
  - Approves all apps available from it

iOS Apps

- Written in Objective-C (ObjC) / Swift
  - Extension of C to add support for OO
  - Using the Cocoa Touch UI framework
  - Can also use C/C++ libraries
  - Compiles to native code via LLVM (not interpreted/JIT as on Android)
  - Swift is getting popular

iOS Frameworks

App appearance, user interactions

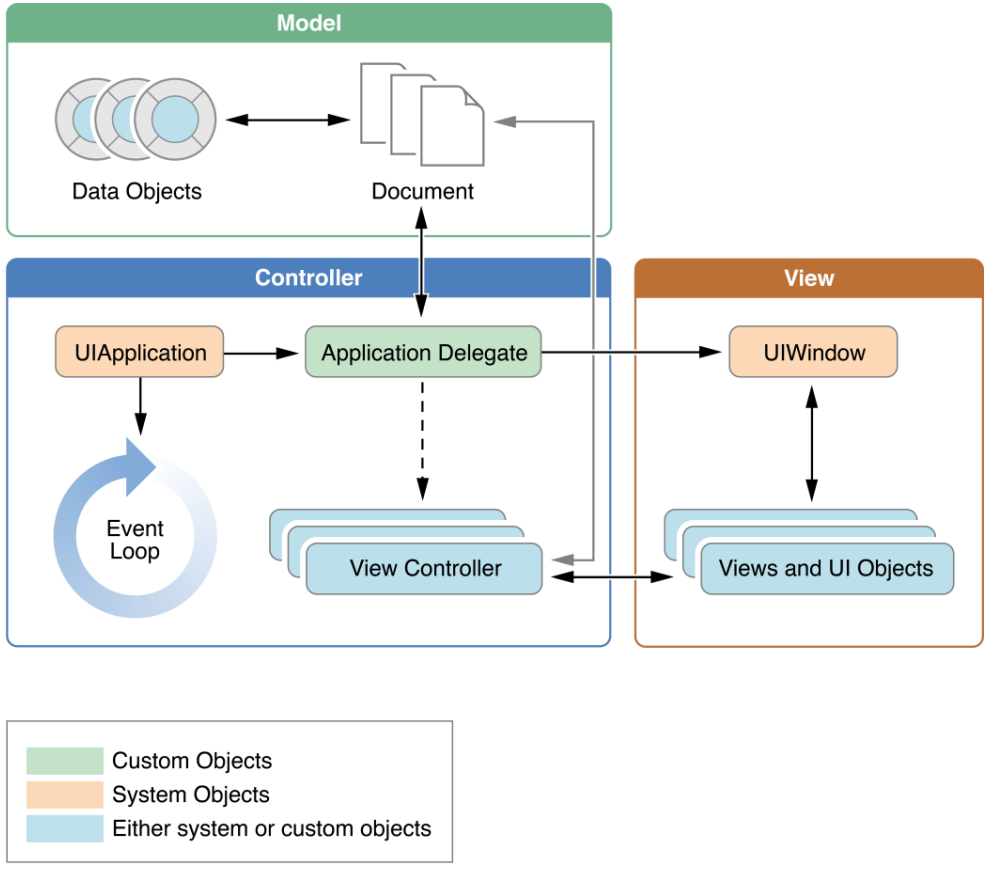
Media pipeline to handle audio/video codec, 3D graphics

Fundamental operating system services

Low-level services related to hardware & networks

Structure of an App
Model View Controller Design Pattern

- **Data Objects:** store data for the App (e.g. images, database)
- **Document:** manage data objects
- **UIApplication Object:** manages event loop & high level app behaviour
- **Delegate:** handle app initialisation, state transition
- **View Controller:** manages the presentation of app on the screen, load view, show view, rotate view, etc.
- **UIWindow:** manages how views are presented on screen (& external screen)
- **Views & UI objects:** draw content, buttons, text fields etc.

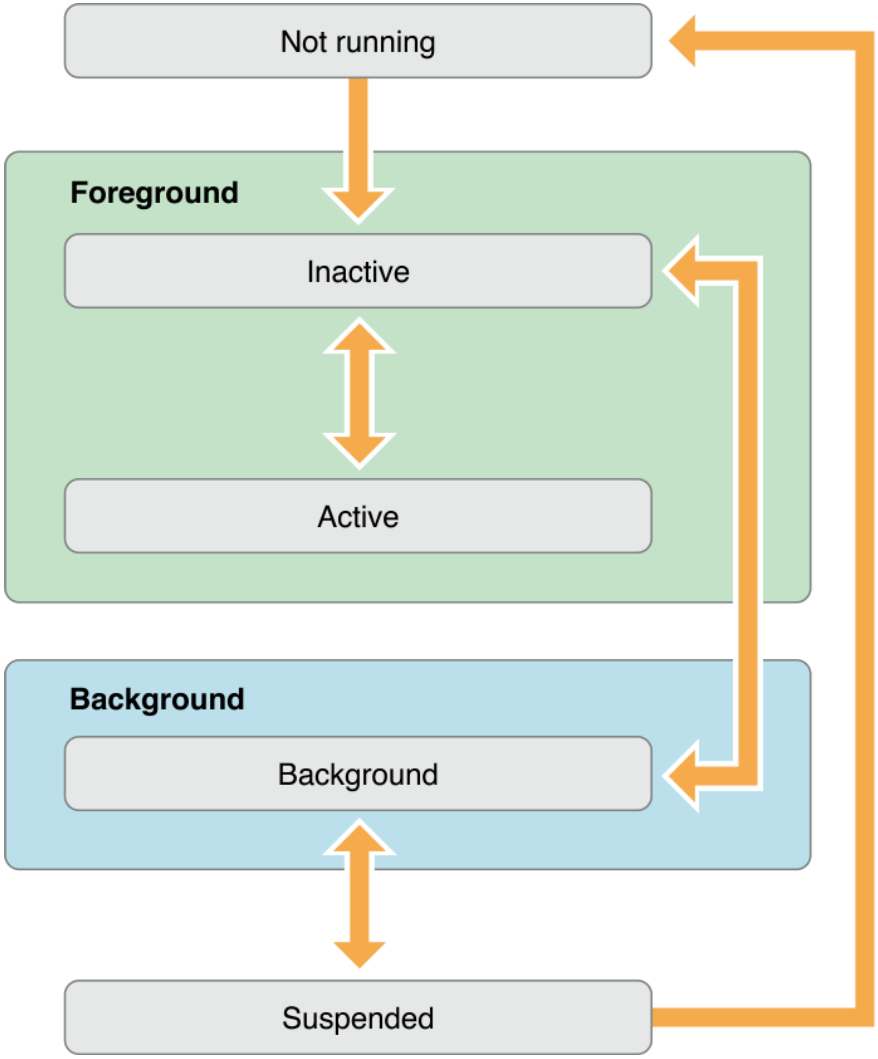


iOS Lifecycle

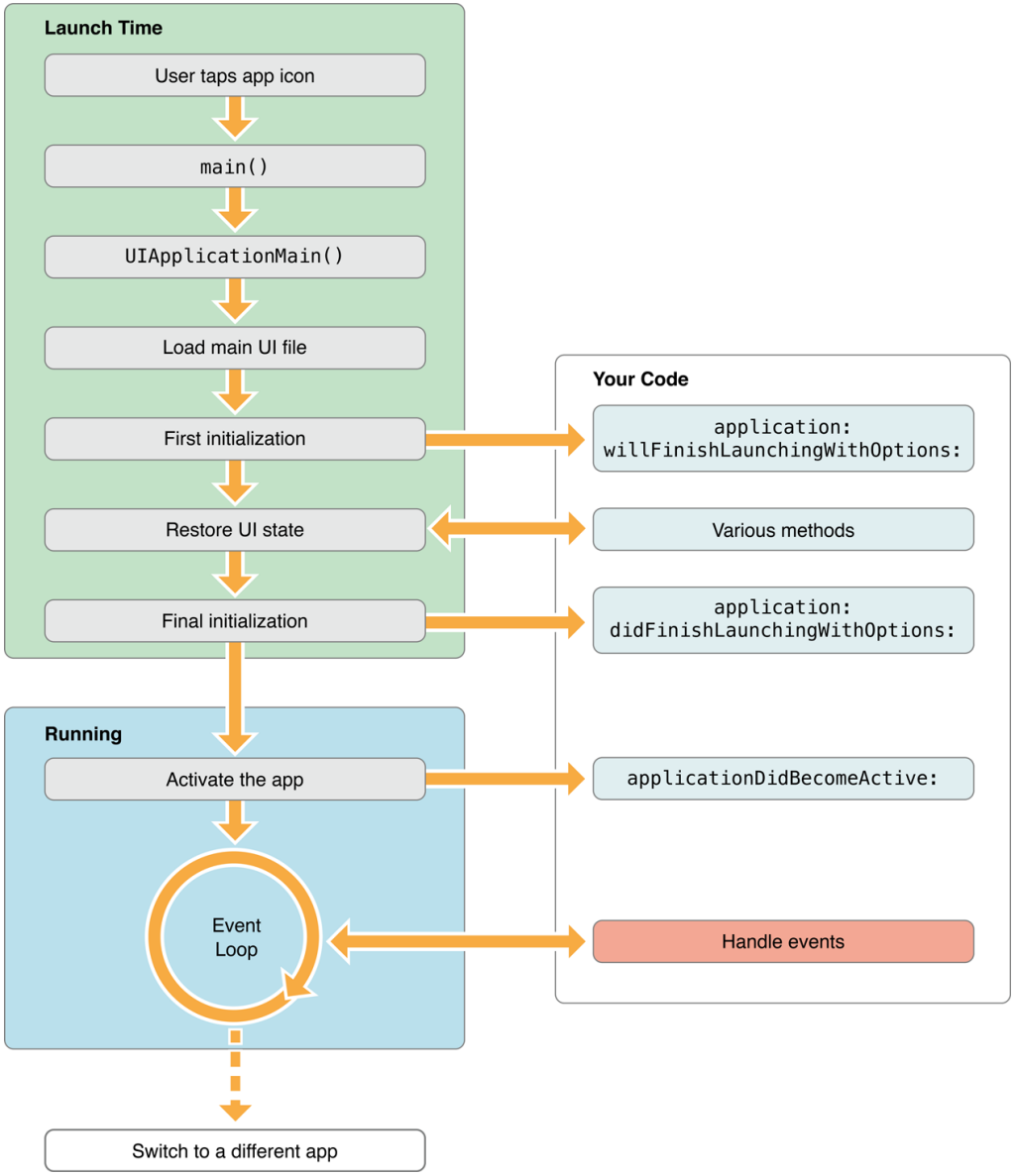
- Analogous to Android lifecycle
  - Apps are sandboxed, inter-app communication to pass data
  - Only one application in the foreground / visible at any one time
- A main loop processes events for the application
- An app can be a number of significant states
  - Active :foreground
  - Inactive: foreground but interrupted by a phone call, notification etc.
  - Background: can remain in this state to perform long running tasks (analogous to Services in Android)
  - Suspended: main loop no longer running, remained in memory but potentially killed by the operating system
- iOS 3.2 and earlier
  - No support for suspended / background states (no long running tasks)

State Transition

State transition handled by Delegate object

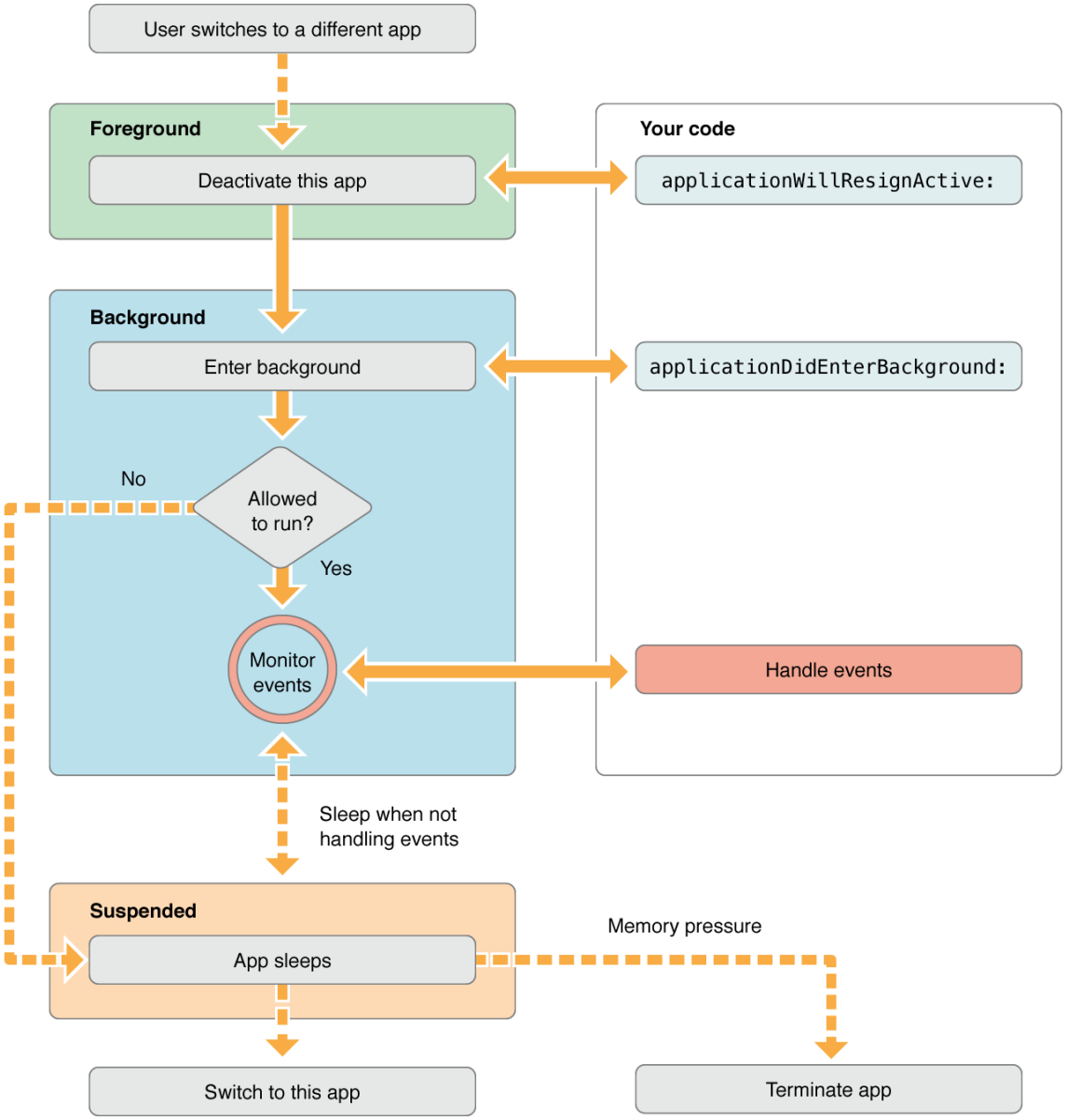


App Launch Cycle



Initialise app's critical data structures; prepare the app's window and views for display

Handling State Transitions



Save data, stop initiate new tasks, pause the process, etc.

App Store

- “We will reject Apps for any content or behavior that we believe is over the line. What line, you ask? Well, as a Supreme Court Justice once said, "I'll know it when I see it". And we think that you will also know it when you cross it.”
- Pre-moderation
  - Apple checks all applications in advance manually
  - Vs Android – publish then revoke
- A long list of guidelines as to what is appropriate
  - Correct use of interface components
  - Substantial content

App Store Restrictions

- **2.5** Apps that use non-public APIs will be rejected
- **2.8** Apps that install or launch other executable code will be rejected
- **2.10** iPhone Apps must also run on iPad without modification, at iPhone resolution, and at 2X iPhone 3GS resolution
- **2.16** Multitasking Apps may only use background services for their intended purposes: VoIP, audio playback, location, task completion, local notifications, etc.
- **2.17** Apps that browse the web must use the iOS WebKit framework and WebKit Javascript
- **13.2** Apps that rapidly drain the device's battery or generate excessive heat will be rejected

Cross Platform?

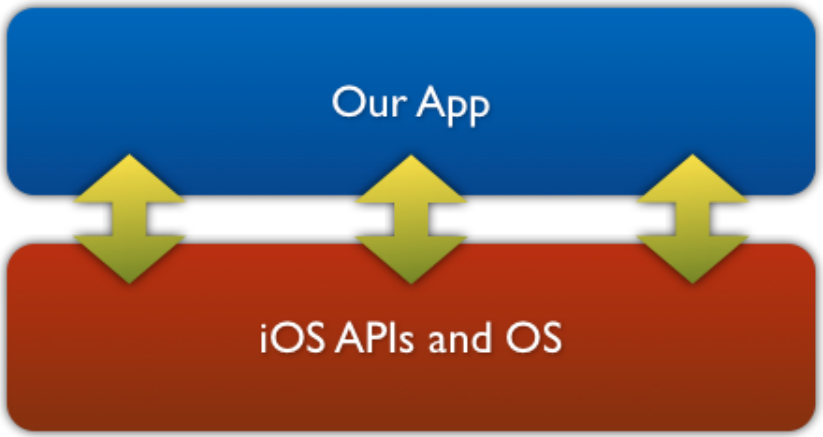
- Apps developed for one system won't work on another
- Would need to port it over
  - This can actually be desirable
    - Can tailor our app to the look and feel of the target device
    - Apple encourage the use of iOS "metaphors"
      - Sliding on/off switches, spinning picker wheels
  - Significant coding effort
- However, there are times when it is desirable to target multiple platforms
  - In-house apps
  - Games (Platform chrome usually irrelevant)
- What are the issues behind cross-platform support?

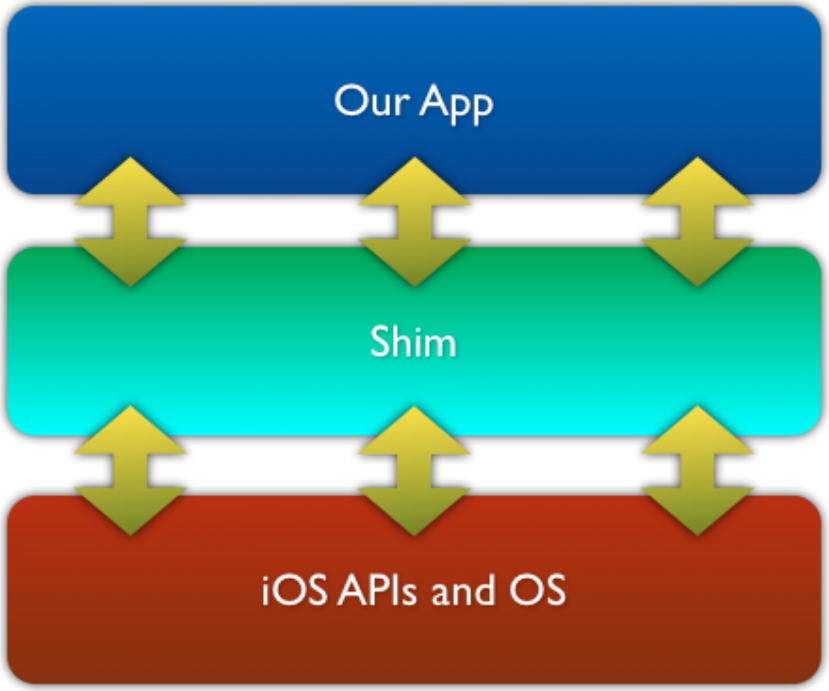
Language

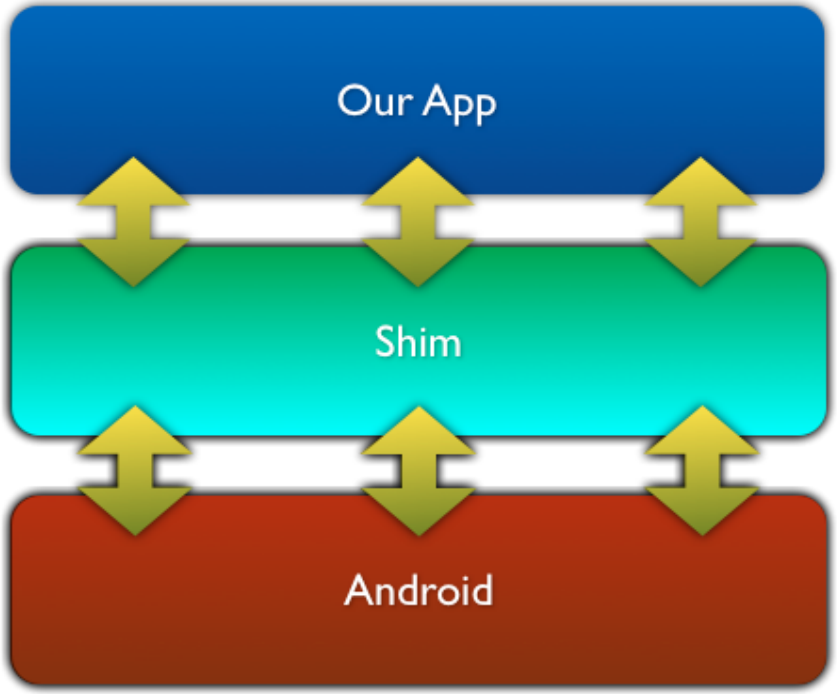
- Compile code for multiple platforms?
  - Can compile Java into native code (gcj)
    - Android supports native code libraries
  - C++ can be compiled to common language runtime (CLR)
- This would work at a technical level
- The code would execute
  - But the app wouldn't run
  - Each platform has different APIs…
    - Android — Activities, Intents, Services…
    - iOS — Views, ViewControllers,
- Could port the app logic relatively easily
  - But would still need to rewrite the UI
  - This maybe enough for games
    - OpenGL ES supported across several platforms
  - End up rewriting boilerplate UI setup stuff

Truly Cross Platform

- Assuming that we can compile code for each device
- To be completely cross platform we can insert a **shim** between our code and the APIs
  - Effectively abstracting our code from the original APIs
  - Our code calls our abstraction
- To port to another device, change the shim and recompile with the appropriate tool-chain
  - New shim provides the same interface to our app
  - But implements it using the native APIs of the new platform


Our App
iOS APIs and OS


Our App
Shim
iOS APIs and OS


Our App
Shim
Android

Current Solutions

- PhoneGap: https://phonegap.com/
  - Code is significantly reused
  - Doesn't look native
- Mono Wrapper: http://www.mono-project.com/; https://www.xamarin.com/
  - Written in C#
  - Access to almost all platform APIs
  - UI still written for each platform

References

- http://phonegap.com/
- http://unity3d.com/
- https://www.xamarin.com/
- https://developer.apple.com/library/content/documentation/iPhone/Conceptual/iPhoneOSProgrammingGuide/Introduction/Introduction.html#//apple_ref/doc/uid/TP40007072-CH1-SW1