

MAC X, iOS and Cocoa Touch Framework Overview

March 16, 2016 | Proprietary and Confidential | - 1 -



Lesson Objectives

➤ Learn

- What Is Cocoa?
- Cocoa History
- Cocoa class libraries
- How Cocoa Fits into Mac OS X
- Cocoa and Cocoa Touch
- How Cocoa Fits into iOS?
- Frameworks available in iOS-based devices



MAC X, iOS and Cocoa Touch Framework Overview

Topics

- What Is Cocoa?
- Cocoa History
- Cocoa class libraries
- How Cocoa Fits into Mac OS X
- Cocoa and Cocoa Touch
- How Cocoa Fits into iOS?
- Frameworks available in iOS-based devices

March 16, 2016 | Proprietary and Confidential | - 3 -



Add the notes here.

What Is Cocoa?

- Cocoa is an application environment for both the Mac OS X operating system and iOS.
- It consists of a suite of object-oriented software libraries, a runtime system, and an integrated development environment.
- Most of the applications you see in Mac OS X and iOS, including Mail and Safari, are Cocoa applications.
- An integrated development environment called Xcode supports application development for both platforms.

March 16, 2016

Proprietary and Confidential

- 4 -



Cocoa is a set of object-oriented frameworks that provides a runtime environment for applications running in Mac OS X and iOS. Cocoa is the preeminent application environment for Mac OS X and the only application environment for iOS. (Carbon is an alternative environment in Mac OS X, but it is a compatibility framework with procedural programmatic interfaces intended to support existing Mac OS X code bases.) Most of the applications you see in Mac OS X and iOS, including Mail and Safari, are Cocoa applications. An integrated development environment called Xcode, supports application development for both platforms. The combination of this development environment and Cocoa makes it easy to create a well-factored, full-featured application.

What Is Cocoa?

- You can use several programming languages when developing Cocoa software, but the essential, required language is Objective-C.
- Objective-C is a superset of ANSI to support object-oriented programming.
- The few added conventions are easy to learn and use.
- You can freely intermix straight C code with Objective-C code.
- Your code can call functions defined in non-Cocoa programmatic interfaces, such as the BSD library interfaces in `/usr/include`
- You can mix C++ code with your Cocoa code and link the compiled code into the same executable.

Cocoa History

- Apple acquired NeXT Software (as it was then called) in 1997, also bringing back Steve Jobs to the company he co-founded
- Much of the work that went into developing OpenStep was applied to the development of Mac OS X, and the application runtime environment was finally renamed to Cocoa.
- Because of its history all Cocoa classes begin with the acronym "NS" (NSObject, NSString, NSArray, etc.)

March 16, 2016 Proprietary and Confidential - 6 -



When Steve Jobs left Apple in 1985 he started a new company called NeXT Computer that developed and manufactured computer workstations intended for higher education and business markets. NeXT Computer developed and released version 1.0 of NeXTSTEP in 1989. In its early phase, NeXTSTEP was more than an application environment, the term referred to the entire operating system. Sales of NeXT computers were limited, but its innovative object-oriented NeXTSTEP operating system was very influential.

Around 1993 the OpenStep initiative took form. OpenStep was a collaboration between Sun and NeXT to port the higher levels of NeXTSTEP to Solaris. The official OpenStep API was published in September of 1994.

Apple acquired NeXT Software (as it was then called) in 1997, also bringing back Steve Jobs to the company he co-founded. Much of the work that went into developing OpenStep was applied to the development of Mac OS X, and the application runtime environment was finally renamed to Cocoa. Because of its history all Cocoa classes begin with the acronym "NS" (NSObject, NSString, NSArray, etc.)

Cocoa class libraries

- **The most important Cocoa class libraries come packaged in two core frameworks for each platform:**
 - Foundation and AppKit for Mac OS X, and
 - Foundation and UIKit for iOS.

- **Both platforms additionally support the Core Data framework, which is as important and useful as the core frameworks**

- **Mac OS X also ships with several other frameworks that publish Cocoa programmatic interfaces, such as the WebKit and Address Book frameworks**

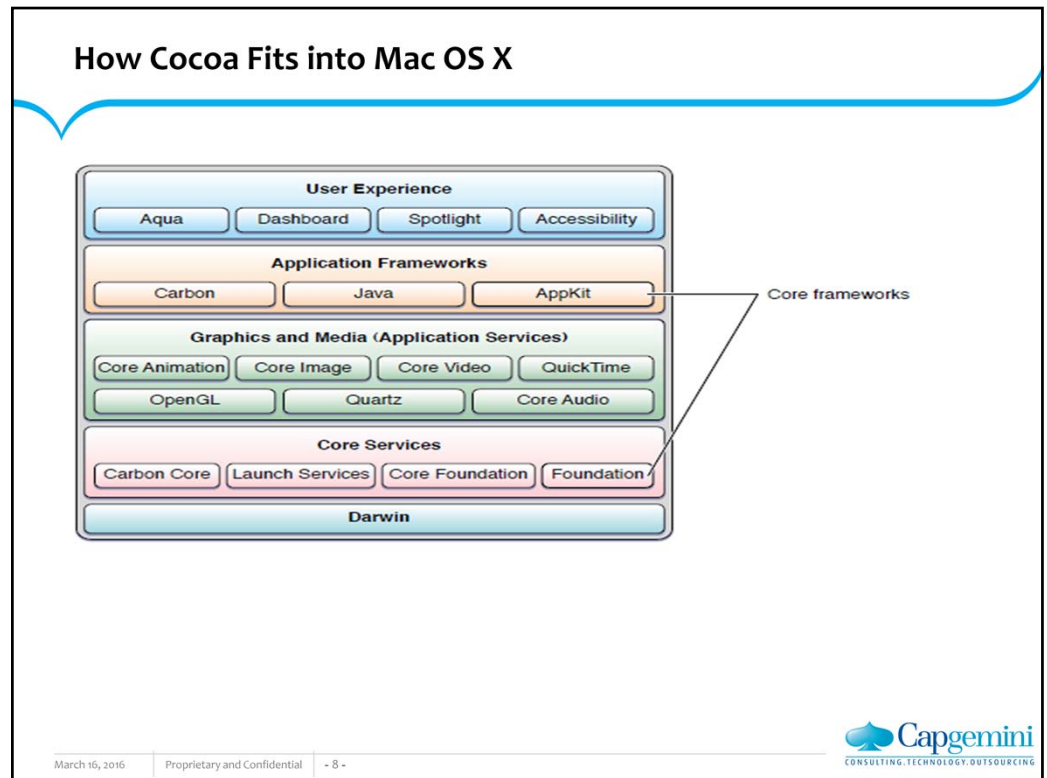
March 16, 2016 | Proprietary and Confidential | - 7 -



The most important Cocoa class libraries come packaged in two core frameworks for each platform: Foundation and AppKit for Mac OS X, and Foundation and UIKit for iOS. As with all frameworks, these contain not only a dynamically sharable library (or sometimes several versions of libraries required for backward compatibility), but header files, API documentation, and related resources. The pairing of Foundation with AppKit or UIKit reflects the division of the Cocoa programmatic interfaces into those classes that are not related to a graphical user interface and those that are. For each platform, its two core frameworks are essential to any Cocoa project whose end product is an application. Both platforms additionally support the Core Data framework, which is as important and useful as the core frameworks.

Mac OS X also ships with several other frameworks that publish Cocoa programmatic interfaces, such as the WebKit and Address Book frameworks; more Cocoa frameworks will be added to the operating system over time.

MAC X, iOS and Cocoa Touch Framework Overview



Architecturally, Mac OS X is a series of software layers going from the foundation of Darwin to the various application frameworks and the user experience they support. The intervening layers represent the system software largely (but not entirely) contained in the two major umbrella frameworks, Core Services and Application Services. A component at one layer generally has dependencies on the layer beneath it.

The Cocoa frameworks consist of libraries, APIs, and runtimes that form the development layer for all of Mac OS X. By developing with Cocoa, you will be creating applications the same way Mac OS X itself is created. Your application will automatically inherit the great behaviors and appearances of Mac OS X, with full access to the underlying power of the UNIX operating system. Using Cocoa with the Xcode IDE is simply the best way to create native Mac applications

How Cocoa Fits into Mac OS X

- **In Mac OS X, Cocoa has two core Objective-C frameworks :**
 - AppKit: It provides the objects an application displays in its user interface and defines the structure for application behavior, including event handling and drawing.
 - Foundation. This framework, in the Core Services layer, defines the basic behavior of objects, establishes mechanisms for their management, and provides objects for primitive data types, collections, and operating-system services.
- **Foundation is essentially an object-oriented version of the Core Foundation framework**

Cocoa and Cocoa Touch

- Cocoa is commonly referred to as the combination of the Foundation and AppKit frameworks, while Cocoa Touch is the combination of the Foundation and UIKit frameworks.
- Cocoa and Cocoa Touch sit on top of other collections of frameworks to create the API stacks. The other layers are Media, Core Services and Core OS.
- The main difference between Cocoa and Cocoa touch is that the UI classes and APIs aren't the same as Mac OS X, so instead of NSTextField, you have UITextField.
- Many of the classes share the same functionality and can be ported quite easily by simply changing the class name, though most will require some more changes, but usually nothing too heavy.

How Cocoa Fits into iOS?

➤ **The following summarizes some of the frameworks found at each layer of the iOS stack, starting from the foundation layer.**

- Core OS : This level contains the kernel, the file system, networking infrastructure, security, power management, and a number of device drivers.
- Core Services : Provide core services, such as string manipulation, collection management, networking, URL utilities, contact management, and preferences. They also provide services based on hardware features of a device, such as the GPS, compass, accelerometer, and gyroscope.
- Media: Provide graphical and multimedia services to the Cocoa Touch layer. They include Core Graphics, Core Text, OpenGL ES, Core Animation, AVFoundation, Core Audio, and video playback.
- Cocoa Touch. The frameworks in this layer directly support applications based in iOS. They include frameworks such as Game Kit, Map Kit, and iAd.

March 16, 2016

Proprietary and Confidential

- 11 -



The application-framework layer of iOS is called Cocoa Touch. Generally, the system libraries and frameworks of iOS that ultimately support UIKit are a subset of the libraries and frameworks in Mac OS X.

The following summarizes some of the frameworks found at each layer of the iOS stack, starting from the foundation layer.

- Core OS. This level contains the kernel, the file system, networking infrastructure, security, power management, and a number of device drivers.
- Core Services. The frameworks in this layer provide core services, such as string manipulation, collection management, networking, URL utilities, contact management, and preferences. They also provide services based on hardware features of a device, such as the GPS, compass, accelerometer, and gyroscope. Examples of frameworks in this layer are Core Location, Core Motion, and System Configuration.

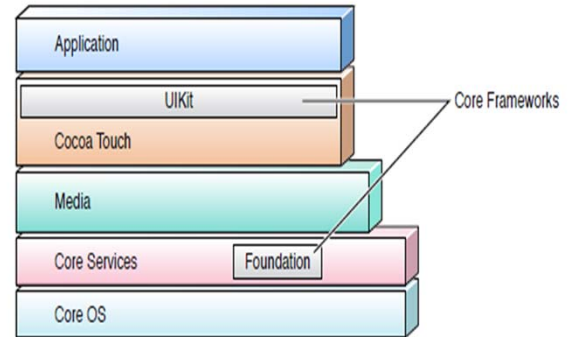
This layer includes both Foundation and Core Foundation, frameworks that provide abstractions for common data types such as strings and collections. The Core Frameworks layer also contains Core Data, a framework for object graph management and object persistence.

- Media. The frameworks and services in this layer depend on the Core Services layer and provide graphical and multimedia services to the Cocoa Touch layer. They include Core Graphics, Core Text, OpenGL ES, Core Animation, AVFoundation, Core Audio, and video playback.
- Cocoa Touch. The frameworks in this layer directly support applications based in iOS. They include frameworks such as Game Kit, Map Kit, and iAd.

How Cocoa Fits into iOS?

- Cocoa Touch is an API for building software programs to run on the iPhone, iPod Touch, and iPad.
- Core Cocoa frameworks in iOS:
 - UIKit
 - Foundation
- **UIKit: It provides the objects an application displays in its user interface and defines the structure for application behavior, including event handling and drawing.**

The architectural setting of iOS



Foundation. It defines the basic behavior of objects, establishes mechanisms for their management, and provides objects for primitive data types, collections, and operating-system services.

March 16, 2016 | Proprietary and Confidential | - 12 -



The application-framework layer of iOS is called Cocoa Touch. Generally, the system libraries and frameworks of iOS that ultimately support UIKit are a subset of the libraries and frameworks in Mac OS X.

The following summarizes some of the frameworks found at each layer of the iOS stack, starting from the foundation layer.

- **Core OS.** This level contains the kernel, the file system, networking infrastructure, security, power management, and a number of device drivers. It also has the libSystem library, which supports the POSIX/BSD 4.4/C99 API specifications and includes system-level APIs for many services.
- **Core Services.** The frameworks in this layer provide core services, such as string manipulation, collection management, networking, URL utilities, contact management, and preferences. They also provide services based on hardware features of a device, such as the GPS, compass, accelerometer, and gyroscope. Examples of frameworks in this layer are Core Location, Core Motion, and System Configuration.

This layer includes both Foundation and Core Foundation, frameworks that provide abstractions for common data types such as strings and collections. The Core Frameworks layer also contains Core Data, a framework for object graph management and object persistence.

- **Media.** The frameworks and services in this layer depend on the Core Services layer and provide graphical and multimedia services to the Cocoa Touch layer. They include Core Graphics, Core Text, OpenGL ES, Core Animation, AVFoundation, Core Audio, and video playback.
- **Cocoa Touch.** The frameworks in this layer directly support applications based in iOS. They include frameworks such as Game Kit, Map Kit, and iAd.

Frameworks available in iOS-based devices

- `AddressBook.framework 2.0 AB` : Contains functions for accessing the user's contacts database directly.
- `AddressBookUI.framework 2.0 AB` :Contains classes for displaying the system-defined people picker and editor interfaces.
- `AssetsLibrary.framework 4.0 AL`: Contains classes for accessing the user's photos and videos.
- `AudioToolbox.framework 2.0 AU,Audio` :Contains the interfaces for handling audio stream data and for playing and recording audio.
- `AVFoundation.framework 2.2 AV`:Contains Objective-C interfaces for playing and recording audio and video.
- `CFNetwork.framework 2.0 CF`:Contains interfaces for accessing the network via Wi-Fi and cellular radios.

March 16, 2016 | Proprietary and Confidential | - 13 -



The slide gives some of the frameworks and its use along with available from which version of iOS and prefix for the classes.

Frameworks available in iOS-based devices

- **CoreData.framework 3.0 NS:**Contains interfaces for managing your application's data model.
- **CoreFoundation.framework 2.0 CF:**Provides fundamental software services, including abstractions for common data types, string utilities, collection utilities, resource management, and preferences.
- **CoreGraphics.framework 2.0 CG:**Contains the interfaces for Quartz 2D.
- **CoreImage.framework 5.0 CI:**Contains interfaces for manipulating video and still images.
- **CoreLocation.framework 2.0 CL:**Contains the interfaces for determining the user's location.
- **CoreMedia.framework 4.0 CM:**Contains low-level routines for manipulating audio and video.
- **CoreMotion.framework 4.0 CM :**Contains interfaces for accessing accelerometer and gyro data.

March 16, 2016 | Proprietary and Confidential | - 14 -



The slide gives some of the frameworks and its use along with available from which version of iOS and prefix for the classes.

Frameworks available in iOS-based devices

- **CoreTelephony.framework 4.0 CT:** Contains routines for accessing telephony-related information.
- **CoreText.framework 3.2 CT :**Contains a text layout and rendering engine.
- **CoreVideo.framework 4.0 CV :**Contains low-level routines for manipulating audio and video. Do not use this framework directly.
- **EventKit.framework 4.0 EK:**Contains interfaces for accessing a user's calendar event data.
- **EventKitUI.framework 4.0 EK :** Contains classes for displaying the standard system calendar interfaces.
- **Foundation.framework 2.0 NS:**Contains interfaces for managing strings, collections, and other low-level data types.
- **GMapKit.framework 3.0 MK:** Contains classes for embedding a map interface into your application and for reverse-geocoding coordinates.

March 16, 2016 | Proprietary and Confidential | - 15 -



The slide gives some of the frameworks and its use along with available from which version of iOS and prefix for the classes.

Frameworks available in iOS-based devices

- **MediaPlayer.framework 2.0 MP:** Contains interfaces for playing full-screen video.
- **OpenGLES.framework 2.0 EAGL, GL :** Contains the interfaces for OpenGL ES, which is an embedded version of the OpenGL cross-platform 2D and 3D graphics rendering library.
- **QuartzCore.framework 2.0 CA:** Contains the Core Animation interfaces.
- **System Configuration.framework 2.0 SC:** Contains interfaces for determining the network configuration of a device.
- **UIKit.framework 2.0 UI:** Contains classes and methods for the iOS application user interface layer.

March 16, 2016 | Proprietary and Confidential | - 16 -



The slide gives some of the frameworks and its use along with available from which version of iOS and prefix for the classes.

How Cocoa Fits into iOS?

- Demo on developing simple helloworld application using cocoa touch



Add the notes here.

Summary

➤ The topics covered..

- What Is Cocoa?
- Cocoa History
- Cocoa class libraries
- How Cocoa Fits into Mac OS X
- Cocoa and Cocoa Touch
- How Cocoa Fits into iOS?
- Frameworks available in iOS-based devices



Add the notes here.

