
**June 2012**

# Native iPad Apps? Why should I care?

What makes the iPad (and its sibling the iPhone) so special? Why do users seem to love working on the device? The overwhelming reaction to both of these questions is that users find it easier to use, highly intuitive and more productive overall. Even IT veterans are impressed by its capabilities and user acceptance – and attracted by its sexiness. The simple reason for this is the operating system that runs on the iPad – iOS.

The iOS architecture is very similar to the architecture found in Mac OS X - the reason a Mac is a Mac. Just this fact alone accounts for many of the similarities between an iPad and a Mac – and why users find the iPad easier to use than any other tablet, including those running Android.

The end point for the iPad is a consistently efficient end user experience. Know how to use one natively developed iPad application and you now know how to use all of them. Plus, all of the extra capabilities provided by Apple today as well as those in the future become available for developers to easily incorporate into their own applications.

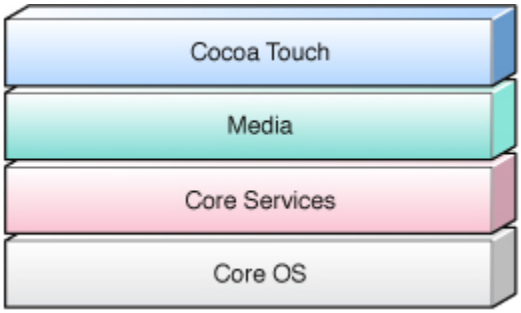
## The key is iOS itself!

At the broadest level, iOS acts as an intermediary between the underlying hardware and the applications that appear on your iPad screen. Applications rarely talk to the underlying hardware directly. Instead, they communicate through well-defined mature system interfaces that protect applications from hardware changes. This abstraction makes it easy to develop applications that work consistently on hardware devices with different capabilities.

Apple also provides an iOS SDK which contains the tools and interfaces needed to develop, install, and run native applications. Native applications for all iOS devices are built utilizing the iOS system frameworks, using Objective-C and running directly on iOS.

Unlike web applications, native applications are installed physically on the iPad and therefore are always available to the user and provide superior performance. They reside next to other system applications and both the application and any user data is synced to the user's computer through iTunes.

The implementation of iOS technologies is best viewed as a set of layers, which are depicted in the figure below from Apple. At the lower layers of the system are the fundamental services and technologies on which all applications rely. The higher-level layers contain more sophisticated services and technologies, all of which are available and easily usable by native application developers.




**iOS Layers**

The functions and features available in these layers and their seamless integration into native applications are what enables these applications to be special – faster, better integrated, able to be used offline and with a higher level of application logic and data security. All with the same consistent user interface now familiar to millions of highly satisfied users.

As of this writing, the new iOS 6 and its Software Development Kit, which provides even more extensive native development capabilities, will be available from Apple this fall.

Below are the highlights of functions available in each iOS layer and why natively developed applications can be superior to those hybrid or web applications for the iPad.

## Cocoa Touch

The Cocoa Touch layer contains the key frameworks for building iOS native applications, defining the basic application infrastructure and support for key technologies such as multitasking, touch-based input, push notifications, and many high-level system services.

High-Level Features available in the Cocoa Touch layer include:

> Storyboards - let you design your entire user interface in one place so you can see all of your views and view controllers and how they work together.
>
> Document Support - makes implementing document-based applications much easier, especially applications that store documents in iCloud and includes built-in support for asynchronous reading and writing of file data, safe saving of data, automatic saving of data, support for detecting iCloud conflicts, and support for flat file or package file representations.

Multitasking – enables applications to shift to a background execution context and helps applications smoothly transition to and from the background state.

Printing - allows applications to send content wirelessly to nearby printers and manages all of the printing interfaces, rendering the printable content, and handling the scheduling and execution of print jobs on the printer.

Data Protection - allows applications that work with sensitive user data to take advantage of built-in encryption enabling application specific files to be protected. When locked, these file are inaccessible to your application and to any outside access.

Push Notification – enables alerts to provide users of new information, even when the specific application is not actively running. These alerts can be implemented locally on the IPad or using an outside server

Gestures – these recognizer functions are objects attached to views and used to detect common gestures – i.e. swipes and pinches. The recognizer tracks the touch events and applies the system-defined action for a given gesture.

File-Sharing - enables applications to make user data files available via iTunes.

Peer-to-Peer - provides peer-to-peer connectivity over Bluetooth to commence sessions with nearby devices.

System View Controllers – enable all native applications to present a consistent user interface.

## Media

The Media layer of iOS contains all of Apple's highly advanced technologies for graphics, audio, and video technologies, all designed by Apple to create the best multimedia experience available on any mobile device.

Graphics - high-quality graphics are a critical element of all iOS applications. Core graphic functions exist to handle all types of rendering, including taking maximum advantage of the new Retina displays, by automatically scaling as needed to support these high-resolution screens.

Audio Layer – supports all the major audio formats and includes the ability to play and record high-quality audio, and trigger the vibration feature on certain devices.

Video Layer - provides numerous technologies to capture and play your video-based content and utilize it seamlessly with your application.

AirPlay - enables your application to stream audio to Apple TV and to third-party AirPlay speakers and receivers.

## Core Services

This layer contains the fundamental system services that all native applications use. The following are some of the key technologies available in Core Services.

iCloud Storage - enables applications to write user documents and data to a central location and access those items from all of the user's computers and iOS devices. The user can view and edit those documents from any device without having to sync or transfer files. This also provides a layer of safety since the documents are only maintained in iCloud storage.

Automatic Referencing - Automatic Reference Counting (ARC) is a system-level feature that simplifies the process of managing instances of objects and managing the appropriate method calls at compile time.

The Core Services layer also enables native application developers the use of:

Block Objects – anonymous code attach
Grand Central Dispatch – for optimized threading
In-App Purchase – for vending
SQLite – lightweight version of SQL
XML Support

## Core OS

The Core OS Layer contains the low-level features that all of the above technologies are built upon and provide frameworks for security and communicating with external hardware. This layer includes:

Accelerate - enables applications to perform DSP, linear algebra, and image-processing calculations and optimized for all of the hardware configurations supported by iOS.

Bluetooth – Objective-C interface supports all Bluetooth Low-Energy ("LE") accessories.

External Accessory - provides support for communicating with hardware directly attached to an iOS-based device.

Security Services – this framework provides a standard set of security-related services to iOS applications.

Security Framework – guarantees the security of the data your application manages, with full support for certificates, public/private keys, trust policies and generation of cryptographically secure pseudorandom numbers. It also supports certificate and crypto-key storage in the keychain.

## A Highly Integrated Experience

All-in-all, the four layers of rich functionality provided in iOS represents a tightly integrated overall development environment enabling native iOS applications to have a look-and-feel all their own, while encompassing significantly enhanced features and techniques.

By utilizing the native functions and features of iOS, the end result is a consistent, high performance, secure application and one the user expects once installed on their device.

The higher-level frameworks in iOS are there to provide the native application developer with object-oriented abstractions for lower-level constructs. These abstractions make it much easier to quickly and efficiently write iPad native code and encapsulate complex features such as sockets and threads. The lower-level frameworks are also available for those developers who want to use aspects of the operating system not exposed by the higher layer frameworks.

Overall, the capabilities built into the operating system enable a native iOS developer to deliver whatever application they need, in fact whatever they can imagine, to the new breed of mobile end user. Developers of web applications trying to replicate the look and feel of iPad native applications can come close, but users in survey after survey say that they can tell the difference.

While the capabilities of pure web apps will continue to grow with HTML5 and garner developer support, and there will be those who prefer a hybrid approach coupling native and web functions together, there will always be those who believe that native applications with their additional features, functions and benefits will be worth the extra effort… …especially for an extraordinary device such as the iPad.