Fortune Info Solutions – iOS(iPhone) Interview Questions

1. What is cocoa?
Cocoa is an application environment for both the Mac OS X operating system and iOS, the operating system used on Multi-Touch devices such as iPhone, iPad, and iPod touch. It consists of a suite of object-oriented software libraries, a runtime system, and an integrated development environment.

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.

2. Development tools in cocoa?
Xcode and Interface Builder.

3.Latest versions of Xcode and iOS?
Xcode-4.5, iOS 6

4.What is Design pattern?
A design pattern is a template for a design that solves a general, recurring problem in a particular context.

5. MVC?
Object-oriented programs benefit in several ways by adapting the MVC design pattern for their designs.

Many objects in these programs tend to be more reusable and their interfaces tend to be better defined.

The programs overall are more adaptable to changing requirements—in other words, they are more easily extensible than programs that are not based on MVC.

The MVC design pattern considers there to be three types of objects: model objects, view objects, and controller objects.

*************Model Object*****************

Model Objects - Encapsulate Data and Basic Behaviors

Model objects represent special knowledge and expertise. They hold an application's data and define the logic that manipulates that data. A well-designed MVC application has all its important data encapsulated in model objects.

Any data that is part of the persistent state of the application (whether that persistent state is stored in files or databases) should reside in the model objects once the data is loaded into the application. Because they represent knowledge and expertise related to a specific problem domain, they tend to be reusable.

Ideally, a model object has no explicit connection to the user interface used to present and edit it. For example, if you have a model object that represents a person (say you are writing an address book), you might want to store a birthdate. That's a good thing to store in your Person model object.

*************View Object*******************

View Objects- Present Information to the User

A view object knows how to display, and might allow users to edit, the data from the application's model. The view should not be responsible for storing the data it is displaying. (This does not mean the view never actually stores data it's displaying, of course. A view can cache data or do similar tricks for performance reasons). A view object can be in charge of displaying just one part of a model object, or a whole model object, or even many different model objects.

View objects tend to be reusable and configurable, and they provide consistency between applications.

A view should ensure it is displaying the model correctly. Consequently, it usually needs to know about changes to the model. Because model objects should not be tied to specific view objects, they need a generic way of indicating that they have changed.

*************Controller Object*************

Controller Objects-Tie the Model to the View

A controller object acts as the intermediary between the application's view objects and its model objects. Controllers are often in charge of making sure the views have access to the model objects they need to display and act as the conduit through which views learn about changes to the model. Controller objects can also perform set-up and coordinating tasks for an application and manage the life cycles of other objects.

In a typical Cocoa MVC design, when users enter a value or indicate a choice through a view object, that value or choice is communicated to a controller object. The controller object might interpret the user input in some application-specific way and then either may tell a model object what to do with this input—for example, "add a new value" or "delete the current record"—or it may have the model object reflect a changed value in one of its properties. Based on this same user input, some controller objects might also tell a view object to change an aspect of its appearance or behavior, such as telling a button to disable itself. Conversely, when a model object changes—say, a new data source is accessed—the model object usually communicates that change to a controller object, which then requests one or more view objects to update themselves accordingly.

Controller objects can be either reusable or nonreusable, depending on their general type.

6. Difference between NSArray and NSMutableArray?
A NSArray's contents can not be modified once it's been created whereas a NSMutableArray can be modified as needed, i.e items can be added/removed from it.

7. Explain retain counts?
Retain counts are the way in which memory is managed in Objective-C. When you create an object, it has a retain count of 1. When you send an object a retain message, its retain count is incremented by 1.

When you send an object a release message, its retain count is decremented by 1. When you send an object a autorelease message, its retain count is decremented by 1 at some stage in the future. If an object's retain count is reduced to 0, it is deallocated.

8. Whats the difference between frame and bounds?
The frame of a view is the rectangle, expressed as a location (x,y) and size (width,height) relative to the superview it is contained within.

The bounds of a view is the rectangle, expressed as a location (x,y) and size (width,height) relative to its own coordinate system (0,0).

9. Can you explain what happens when you call autorelease on an object?
When you send an object a autorelease message, its retain count is decremented by 1 at some stage in the future.

The object is added to an autorelease pool on the current thread. The main thread loop creates an autorelease pool at the beginning of the function, and release it at the end. This establishes a pool for the lifetime of the task.

However, this also means that any autoreleased objects created during the lifetime of the task are not disposed of until the task completes. This may lead to the task's memory footprint increasing unnecessarily.

You can also consider creating pools with a narrower scope or use NSOperationQueue with it's own autorelease pool. (Also important – You only release or autorelease objects you own.)

10. Whats an NSOperationQueue and how/would you use it?
The NSOperationQueue class regulates the execution of a set of NSOperation objects. An operation queue is generally used to perform some asynchronous operations on a background thread so as not to block the main thread.

11. What is delegate?
Delegation is a simple and powerful pattern in which one object in a program act on behalf of or in co-ordination with other object.

In a clear way one object does the hands of work of other objects.

12. Does multiple Inheritance supports in objective-c?
Yes it supports, We can achieve it using Protocols concept.

13. What is a protocol?
Protocols declare methods that can be implemented by any class. Protocols are useful in at least three situations:

To declare methods that others are expected to implement

To declare the interface to an object while concealing its class

To capture similarities among classes that are not hierarchically related

A protocol is simply a list of method declarations, unattached to a class definition. For example, these methods that report user actions on the mouse could be gathered into a protocol:

```
- (void)mouseDown:(NSEvent *)theEvent;

- (void)mouseDragged:(NSEvent *)theEvent;

- (void)mouseUp:(NSEvent *)theEvent;
```

Any class that wanted to respond to mouse events could adopt the protocol and implement its methods.

14. What is an Abstract class?
Some classes are designed only or primarily so that other classes can inherit from them.

These abstract classes group methods and instance variables that can be used by a number of subclasses into a common definition.

The abstract class is typically incomplete by itself, but contains useful code that reduces the implementation burden of its subclasses. (Because abstract classes must have subclasses to be useful, they're sometimes also called abstract superclasses.)

15. What is a Category?
A category allows you to add methods to an existing class—even to one for which you do not have the source.

Categories are a powerful feature that allows you to extend the functionality of existing classes without subclassing.

Using categories, you can also distribute the implementation of your own classes among several files.

16. Difference between a protocol and Category?
A Protocol say's "Here are some methods I would like you to implement".

A Category say's "I am extending the functionlaity of this class with these additional methods"

17. Explain Singleton classes?
It's an extremely powerful way to share data between different parts of code without having to pass the data around manually.

18. When do we use @property and @synthesize?
1 down vote accepted

@property : you used it when you want to:

You can use some of the really useful generated code like nonatomic, atomic, retain without writing any lines of code.

You also have getter and setter methods. To use this, you have 2 other ways: @synthesize or @dynamic: @synthesize, compiler will generate the getter and setter automatically for you, @dynamic: you have to write them yourself.

@property is really good for memory management, for example: retain.

How can you do retain without @property?

```
 if (_variable != object) {
   [_variable release];
   _variable = nil;
   _variable = [object retain];
 }
```

How can you use it with @property?

```
self.variable = object;
```

When you are calling the above line, you actually call the setter like [self setVariable:object] and then the generated setter will do its job

19. assign, retain, copy?
assign: "Specifies that the setter uses simple assignment.This is the default.You typically use this attribute for scalar type", So you can imagine assigning a float(which isn't an object), So can't be retained, copied..etc.

retain: "Specifies that reatain should be involved on the object upon assignment... The previous value is sent a release message". So you can imagine an NSString instance(Which is an object and which you probably want to retain).

Copy: "Specifies the copy of object should be assigned and the perevious values is sent a release message.

20. Different types of data storages in iOS?
SQLite3, Property lists, Core data, files.

21. Different types of webservices?
1.NSXMLParser - Synchronous(Apple's default webservice provider)
2.LibXmlParser - Asynchronous
3.JSON - Its a light weight webservice provider and very easy to use.
And there are many more webservices like Tocuh XML, KissXML... etc.

22. iOS Architecture?
iOS Architecture mainly divided into 4 layers.

1.Cocoa Touch Layer
2.Media Layer
3.Core Services Layer
4.Core OS Layer

The API' Under Cocoa Touch Layer as follows:
-AddressBookUI
-EventKitUI
-GameKit
-iAd
-MapKit
-MessageUI
-Twitter
-UIKit

The API' Media Layer as follows:
-AssetsLibrary
-AudioToolbox
-AudioUnit
-AvFoundation
-CoreAudio
-CoreGraphics
-CoreImage
-CoreMIDI
-CoreText
-CoreVideo
-GLKit
-ImageIO
-MediaPlayer
-OpenAL
-OpenGLES
-QuartzCore

The API' Under Core Services Layer as follows:
-Accounts
-AddressBook
-CFNetwork
-CoreData
-CoreFoundation
-CoreLocation
-CoreMedia
-CoreMotion
-CoreTelephony
-EventKit
-Foundation
-MobileCoreServices
-NewsstandKit

-QuickLook
-StoreKit
-SystemConfiguration
-UIAutomation

The API' Under Core OS Layer as follows:
-Accelerate
-CoreBluetooth
-ExternalAccesory
-Security
-System

23. Debugging technics in iOS?
GDB, Breakpoints, NSZombie, Instruments.

24. Features in iOS 5?
iCloud Storage:
iCloud Storage APIs enable your apps to store user documents and key value data and wirelessly push any changes to all your user's computers and devices at the same time — automatically.

Notification Center:
Notification Center provides an innovative way to easily display and manage your app notifications without interrupting your users. Notification Center in iOS 5 builds on the existing notification system, so your existing local and push notifications just work. Provisioning of push notifications is now built right into Xcode making it even easier to implement.

Newsstand:
Publish the latest issues of your magazines and newspapers directly to Newsstand, the new folder on the Home Screen. Newsstand Kit provides everything you need to update new issues in the background, so you can always present the most recent cover art. Apps built for Newsstand use In-App Purchase subscriptions, making it easy for users to manage their auto-renewable subscriptions. And it's now possible to provision your app for In-App Purchase within Xcode

Automatic Reference Counting:
Automatic Reference Counting (ARC) for Objective-C makes memory management the job of the compiler. By enabling ARC with the new Apple LLVM compiler, you will never need to type retain or release again, dramatically simplifying the development process, while reducing crashes and memory leaks. The compiler has a complete understanding of your objects, and releases each object the instant it is no longer used, so apps run as fast as ever, with predictable, smooth performance.

Twitter Integration:
Tweet directly from your apps using the new Tweet sheet. It provides all of the features available to built-in apps, including URL shortening, attaching current location, character count and hosting photos on Twitter. And if your app is a Twitter client, it's easy to tie into the single sign-on service using the Twitter APIs. It's even possible to migrate existing accounts to iOS.

Storyboards:
Layout the workflow of your app using the new Storyboards feature built into the design tools of Xcode. Created for apps that use navigation and tab bars to transition between views, Storyboards eases the development by managing the view controllers for you. You can specify the transitions and segues that are used when switching between views without having to code them by hand.

AirPlay:
Introduced in iOS 4.2, AirPlay streams video, audio and photos to Apple TV. With iOS 5, it's now possible to wirelessly mirror everything on your iPad 2 to an HDTV via Apple TV. Your apps are mirrored automatically. With additional APIs your app can display different content on each of the HDTV and the iPad 2 screens. In iOS 5, apps built with AV Foundation can now stream video and audio content through AirPlay, and AirPlay now supports encrypted streams delivered via HTTP Live Streaming

Core Image:
Create amazing effects in your camera and image editing apps with Core Image. Core Image is a hardware-accelerated framework that provides an easy way to enhance photos and videos. Core Image provides several built-in filters, such as color effects, distortions and transitions. It also includes advanced features such as auto enhance, red-eye reduction and facial recognition.

Game Center:
Game Center is taking multiplayer gaming on iOS one step further with the addition of turn-based game support. With turn-based games, players can play when they want and Game Center will manage each turn for them. Game Center will automatically send the next player a push notification via Notification Center and manage multiple game sessions. Other developer additions to Game Center include, adding players to existing multiplayer games, displaying achievement notification banners, and support for distinct icons for each leaderboard.

OpenGL ES:
It's now even easier to develop great looking games that take advantage of the latest iOS hardware. GLKit is a new high-level framework that combines the best practices of advanced rendering and texture techniques with the latest OpenGL ES 2.0 features. It's optimized to take advantage of hardware accelerated math operations, so you get the best performance without all the work. iOS 5 SDK also includes new Apple-developed OpenGL ES extensions designed specifically for advanced game developers. And the new OpenGL ES debugger in Xcode allows you to track down issues specific to OpenGL ES in your code.

iMessage:
iMessage is a new messaging service that works between all iOS 5 users over Wi-Fi and 3G. iMessages are automatically pushed to all iOS 5 devices, making it easy to maintain one conversation across iPhone, iPad and iPod touch. In iOS 5 SDK, the Message sheet now supports the iMessage service, so you can start individual or group text conversations from within your app.

New Instruments:
In addition to ARC, iOS 5 SDK includes several new instruments including time profiler with CPU strategy which gives you a new way to view time profiler data, as well as system trace, network activity and network connections instruments.

PC Free:
iOS 5 includes a host of features that give users the power, freedom, and flexibility to use their iOS devices without a Mac or PC. Expand the functionality of your apps and remove the need for users to access a PC. Take advantage of iCloud Storage to store documents and user data, so they are updated automatically and users can access them from all of their devices.

Location simulation:
Now you can test your location-based features in your app without leaving your desk. You can now select from preset locations and routes within the iOS Simulator and pick a custom latitude and longitude with accuracy while you're running your simulated app.