InterviewBit

iOS Interview Questions

To view the live version of the page, [click here.](#)

Contents

iOS Interview Questions for Experienced (.....Continued)

19. Explain Swift in iOS?
20. What are the important features of Swift?
21. Explain NSError in Swift.
22. What is Enum or Enumerations in Swift?
23. What do you mean by lazy property in iOS?
24. What are generics in swift and write its usage?
25. Explain dictionary in Swift.
26. Why design patterns are important? Name some of the popular design patterns used in iOS?
27. What is the JSON framework supported by iOS?
28. Explain iBeacons in iOS.
29. State the difference between KVC and KVO in Swift.
30. Explain TDD (Test-Driven Development).
31. Explain the function of the completion handler.
32. State the difference between strong, weak, read only and copy.
33. What do you mean by dynamic dispatch?
34. Explain the @dynamic and @synthesize commands in Objective-C.
35. How can you implement storage and persistence in iOS?
36. What are synchronous and asynchronous tasks in iOS? In what way can you execute asynchronous tasks in iOS?

Let's get Started

What is iOS

No doubt, we have seen the abbreviation iOS hundreds of times. iOS stands for "iPhone Operating System." It is the operating system for Apple devices, and it is considered the second most popular mobile operating system globally after Android. This operating system powers many of Apple's products including the iPhone, iPad, and iPod. iOS is widely praised for its intuitive and user-friendly interface. Learn More.

Features of the iOS Platform:

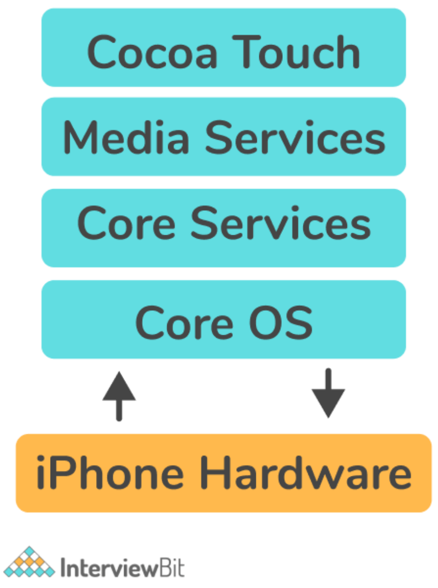
Because of its prominent features, iOS has become popular. Here are a few of its most notable features:

- The iPhone offers multitasking capabilities. On an iOS device, you can easily switch between apps using the multitasking feature or a multi-finger gesture.
- iOS helps you easily integrate social network interactions with your app by displaying an activity stream and sharing content.
- Apple's iCloud service allows users to store data on the Internet. It offers a high level of encryption and a backup option to ensure the user does not lose data.
- Apple's in-app purchases are available on all platforms, offering users additional services and materials including digital items (iOS, iPad, macOS), subscriptions, and premium content.
- You can see all of your app alerts in the Notification Center in iOS. However, the notification settings can be modified.
- iOS is a closed system. The source code of Apple's apps isn't available for developers, and iPhone and iPad owners can't modify the code on their devices. This makes iOS-powered devices harder to hack.

iOS Interview Questions for Freshers

1. Explain the Architecture of iOS.

iOS operates in a Layered structure. iOS Architecture is comprised of four layers, each of which offers a programming framework for creating applications that operate on top of the hardware. Communication will be enhanced by the layers between the Application Layer and the Hardware Layer. A lower-level layer provides the services that all applications require, while an upper-level layer (or high-level layer) provides graphics and interface-related services.



- **Core OS (or Application) Layer:** Core OS Layer sits directly on top of the device hardware and is the bottom layer of the iPhone OS stack. In addition to basic operating system services, such as memory management, handling of file systems, and threads, this layer provides low-level networking, access to external accessories, etc.
- **Service Layer:** Its purpose is to design the services that upper layers or users demand. Among its other essential features are block objects, Grand Central Dispatch, in-app purchases, and iCloud storage. The service layer has been strengthened by the addition of ARC Automatic Reference Counting.
- **Media Layer:** It handles media like video, audio, graphics, etc. The media layer will allow us to use all graphics, video, and audio technology of the system.
- **Cocoa Touch Layer:** It is also known as the application layer. This is the place where frameworks are created when applications are built. In addition, it functions as the interface for iOS users to work with the operating system. This includes touch and motion capabilities.

2. What do you mean by property in iOS?

Properties are basically values that are associated with a class, struct, or enum. They can be thought of as "sub-variables," i.e., parts of another object.

Example:

```
struct Icecream
{
    var flavor: String = ""
}
var choco = Icecream()
choco.flavor = "Chocolate Icecream"
```

In the above code, we created a structure called Icecream. One of its properties is called flavor, a String whose initial value is empty.

Classification of Properties

- **Stored Properties:** This type of property can be used to store constant or variable values as instances and is usually provided by classes and structures.

Example:

```
class Programmer {
    var progName: String
    let progLanguage: String
    var totalExperience = 0
    var secondLanguage: String?
}
```

Above, the Programmer class defines four stored properties: progName, progLanguage, totalExperience, and secondLanguage. These are all stored properties since they can contain values that are part of the instance of the class. The above example shows properties with and without default values, as well as an optional one.

- **Computed properties:** These properties can be used to calculate values instead of storing them and are usually provided by classes, enumerations, and structures.

Example:

```
struct Angle {
    var degrees: Double = 0

    var rads: Double {
        get {
            return degrees * .pi / 180
        }
        set(newRads) {
            degrees = newRads * 180 / .pi
        }
    }
}
```

As mentioned above, the angle structure has a stored property called degrees for storing angles in degrees. Moreover, angles can also be expressed in radians, so the Angle structure contains a Rads property, which is a computed property.

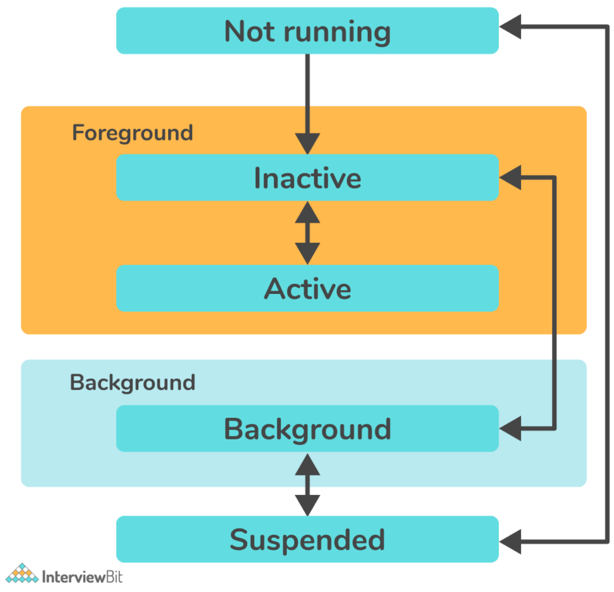
3. Can you explain the difference between atomic and nonatomic properties? What is the default for synthesized properties?

Atomic Property: It is the default property and ensures a valid value will be returned from the getter or set by the setter. This ensures that only one thread can access the getter/setter of a given property at a time and that all other threads must wait until the first thread releases the getter/setter. Despite being thread-safe, it is not fast, since it ensures that the process is completely completed.

Non-Atomic Property: With non-atomic properties, multiple threads can access the getter/setter method of a given property at the same time, so the potential for inconsistency between values exists. They come with enhanced access, but no guarantee of the return value.

4. What are different types of iOS Application States?

During the course of its execution, an iOS application goes through a series of states. Each of these states is referred to as an application's lifecycle state. Below are the five possible states for an iOS app:



- **Not running:** In the Not Running state, an application has either not been launched or has been closed/shut down by the system.
- **Inactive:** A brief state of inactivity occurs while the app is leaving or entering its active state. Despite running in the foreground, it isn't yet ready to accept user input or events. This means that the application remains inactive at this time.
- **Active:** The Active state indicates that the app is running in the foreground and receiving events. This is usually the normal mode for foreground apps and the User Interface is accessible.
- **Background:** During this state, the application's user interface is hidden, but it continues to run in the background of the iOS system. Applications usually pass through this state prior to being suspended.
- **Suspended:** In this case, the application is in the background but is not running code. However, it stays in my memory. Under low memory conditions, the system can delete apps in the suspended state without warning.

5. What is an iOS developer and what are his responsibilities?

An iOS developer is a programmer or software engineer who designs and develops applications that run Apple's iOS on iOS devices. Ideally, the iOS developer should be skilled in two programming languages i.e., Objective-C and Swift.

Main Responsibilities of an iOS Developer

- Clean, efficient coding for iOS applications.
- Ensure clean and secure codes by performing troubleshooting and bug fixes for applications.
- The development and deployment of advanced app features as well as the maintenance and improvement of existing features.
- Develop innovative solutions to meet the business needs of customers.
- Assisting with all aspects of application development, including design, testing, release, and support.
- Exploring, evaluating, and implementing new technologies continuously to maximize development efficiency.

6. State the difference between Android and iOS.

Android: It is the mobile operating system for Android devices offered by Google LLC (limited liability company) and is focused on touchscreen mobile devices like smartphones and tablets. Several programming languages were used in its development, including C, Java, C++, and others.

iOS: It is the operating system for Apple devices offered by Apple incorporation and it is considered the second most popular mobile operating system globally after Android. It is primarily designed for Apple mobile devices like the iPhone, iPod Touch, etc. Several programming languages were used in its development, including Objective-C, Swift, C++, and others.

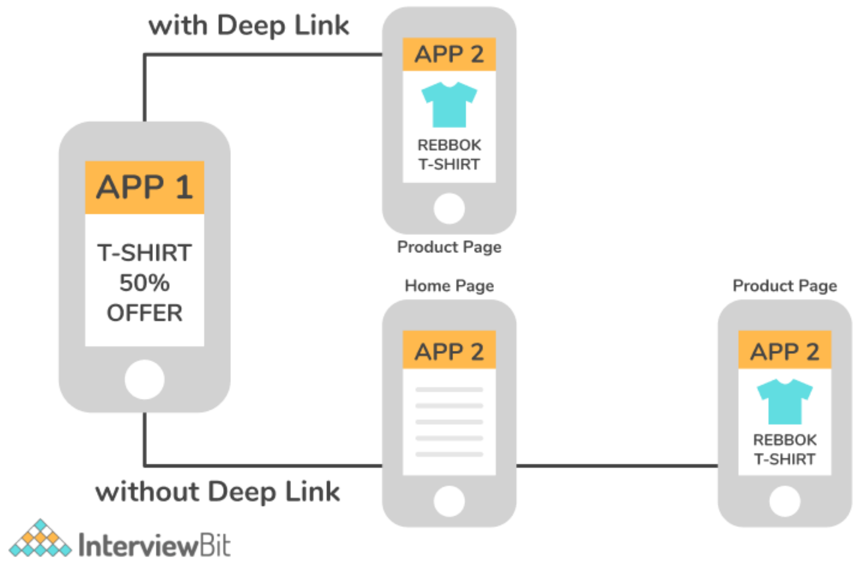
Difference between iOS and Android -

Android	iOS
It is the mobile operating system for Android devices offered by Google LLC (limited liability company).	It is the operating system for Apple devices offered by Apple incorporation.
It is specially designed for smartphones, tablets.	It is specially designed for Apple iPhones, iPods, and iPads.
It is mainly written in C, C++, Java and other languages.	It is mainly written in C, C++, Objective-C, Swift, and other languages.
On Android devices, Google Chrome is the default internet browser. However, any other browser can be installed.	Safari is the default Internet browser on iOS devices. However, any other browser can be installed.
The Android software is available for many manufacturers including Samsung, LG, etc., which could cause some quality issues in cheaper phones.	Apple has strict quality control over iOS and there are no quality problems.
Contrary to this, the performance of Android devices may deteriorate over time.	iOS devices run at a consistent speed over time.

7. What do you mean by deep linking in iOS?

Deep links are links that send users directly to an app directly instead of a website or store using URI (Uniform resource locator) or universal links. The URL scheme is a well-known method of having deep links, but Universal Links are Apple's new approach to connecting your web page and your app under the same link. Deep linking involves not only creating a clickable link that opens up your app, but also a smart one that navigates to the resource you desire. Users are directed straight to in-app locations using these links, which saves them the time and effort of finding those pages themselves thus improving their user experience tremendously.

Explanation: If you use the URL fb://, you may open the Facebook app. However, if you use fb://profile/33138223345, you will open Wikipedia's profile in the Facebook app.



8. Explain what is GCD (Grand Central Dispatch) in iOS.

Grand Central Dispatch (GCD) is a low-level API that enables users to run concurrent tasks (occurring simultaneously) by managing threads in the background. Grand Central Dispatch is Apple's solution to build concurrency and parallelism into iOS applications, so multiple background tasks can be run concurrently in the background without affecting the main thread. It was introduced in iOS 4 to avoid the tedious process of serial execution of tasks.

9. What is ARC (Automatic Reference Counting)?

In the Swift programming language, automatic reference counting (ARC) is used to manage apps' memory usage. It initializes and deinitializes system resources, thereby releasing memory reserved by a class instance when it no longer needs it. ARC keeps track of how many properties, constants, and variables currently refer to each class instance. When there is at least one active reference to an instance, ARC will not deallocate that instance. The use of ARC concepts is an essential part of iOS development.

Functions of ARC -

- ARC creates a new class instance using init() and allocates a chunk of memory to store the information.
- Memory stores information about the instance type and its values.
- As soon as the class instance is no longer required, ARC automatically frees memory by calling deinit().
- By tracking current referring classes' properties, constants, and variables, ARC ensures that deinit() is only applied to those instances that are unused.

10. State the difference between Cocoa and Cocoa Touch.

Cocoa and Cocoa Touch are two of Apple's widely used application frameworks used for building applications. However, they differ in the following ways:

Cocoa	Cocoa Touch
It is an application framework for building applications that run on Mac OS X.	It is the application framework for building applications that run on devices like iPhones and iPad.
Frameworks such as Foundation and AppKit are incorporated into Cocoa.	Cocoa Touch is a combination of Frameworks such as Foundation and UIKit are incorporated into Cocoa Touch.
Cocoa classes use the NS (used for all classes and constants in the cocoa framework) prefix (like NSTextField, NSWindow).	Cocoa Touch classes, on the other hand, use the UI (used for all classes and constants in the cocoa touch framework) prefix (like UITextField and UIWindow).

11. Which programming languages are used for iOS development?

iOS development is done using the following programming languages:

- HTML5
- .NET
- C
- C++
- Swift
- Javascript
- Objective-C.

12. What is the framework that is utilized to build an application's interface for iOS?

Contrary to the Foundation framework that defines classes, protocols, and functions for both iOS and OS X development, UIKit is specifically designed for iOS development. In iOS, the user interface and graphical infrastructure of the application are developed using UIKit. It includes:

- Event handling (handle different gestures like input gestures, button-tap gestures, multi-touch gestures, etc.)
- App structure (Manages the interaction between the system and user)
- User Interface (Provides user interactions, the ability to share text and content, select images, edit videos, print files, etc.)
- Graphic, Drawing, and Printing

13. Write different ways to achieve concurrency in iOS?

Concurrency means "running multiple tasks simultaneously". Concurrency allows iOS devices to handle background tasks (such as downloading or processing data) while maintaining a responsive user interface. In iOS, you can manage concurrent tasks using Grand Central Dispatch (or GCD), and Operations (formally known as NSOperation). In order to achieve concurrency, iOS provides three ways as follows:

- **Dispatch queues:** They are used to manage tasks in first-in-first-out (FIFO) order and execute tasks sequentially or concurrently. This is an easy way to handle asynchronous (not occurring at the same time) and concurrent tasks in your application.
- **Threads:** An independent sequence of instructions can be executed separately from other code within a program. Through threads, one can execute multiple code paths simultaneously in a single application. Having a thread is especially useful when you need to perform a lengthy task without affecting the execution of the rest of the program.
- **Operation Queues:** Operation queue objects are invoked in accordance with their priority and readiness. Essentially, operation queues are high-level abstractions of queueing models, built on top of GCD (Grand Central Dispatch). It is possible, therefore, to execute tasks concurrently, just like GCD, but in an object-oriented manner.

14. State the difference between App ID and Bundle ID?

- **Bundle ID:** They are the unique identifiers of applications in Apple's ecosystem. In other words, no two applications can have the same identifier. The bundling ID is used for both OS X and iOS apps and can be used to recognize app updates.
    - **Example:**
      If our organization’s domain is scaler.com and we create an app named Edge, you could assign the string **com.scaler.edge** as our app’s bundle ID.
- **App ID:** This string uniquely identifies one or more apps from the same development team. There are two components to the string, the Team ID and the Bundle ID, separated by a period (.). Apple supplies a Team ID to identify a specific development team, whereas developers supply Bundle IDs to identify a single app or a collection of apps.
    - **Example:**
      ABCDE12345.com.scaler.edge
      In the above example, **ABCDE12345** is the Team ID and **com.scaler.edge** is the Bundle ID.

15. What do you mean by the SpriteKit and SceneKit framework in the context of game development?

SpriteKit: This framework is designed to make it easier and faster for game developers to create animated 2D assets/objects in casual games. With it, you can draw shapes, particles, text, images, and videos in two dimensions.

SceneKit: It is an iOS framework inherited from OS X, which helps to create 3D graphics. With SceneKit, you can build 3D animated scenes and effects for your iOS games and apps.

16. Write the difference between assign and retain keywords.

- **Assign:** With Assign, a reference is created from one object to another without increasing the source's retain count (a number that keeps track of how many objects are “holding onto" another object). It does not copy or retain the value but assigns it directly to the instance variable.

Example:

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

Here, Assign will generate a setter that assigns the value to the instance variable directly, rather than copying or retaining it.

- **Retain:** Using this method, you create a reference from one object to another and increase the retain count of the source object.

Example:

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

A retain message prevents an object from being deallocated until you are done using it.

iOS Interview Questions for Experienced

17. Explain Objective-C in OS.

Since the 1990s, Objective-C has been used by Apple as an object-oriented programming language. This language combines the advantages of two earlier languages - C and Smalltalk. As a superset of C, it provides object-oriented functionality and a dynamic runtime environment.

Features:

- In Objective C, meta classes are automatically created and managed during runtime.
- Dynamic typing and static typing are both supported.
- It is easy to understand.

18. Name the most important data types in Objective-C?

Following are data types that the developers mostly use in Objective – C:

- **BOOL:** Represents a Boolean value that is either true or false. Both the _Bool or BOOL keywords is valid.
    - ***Example:***
      _Bool flag = 0;
      BOOL secondflag = 1;
- **NSInteger:** Represents an Integer.
    - ***Example:***
      typedef long NSInteger;
      typedef int NSInteger;
- **NSUInteger:** Represents an unsigned integer.
    - ***Example:***
      typedef unsigned long NSUInteger;
      typedef unsigned int NSUInteger;
- **NSString:** Represents a string.
    - ***Example:***
      NSString *greeting = @"Hello";

19. Explain Swift in iOS?

Swift is the fastest-growing programming language today, created by Apple. With a significant advantage pool over the well-known Objective-C, Swift holds a leading position in iOS development. It's an entirely new language created specifically to develop software for Apple's operating systems. Since Swift implements all the features of other modern languages, you can find type inference, optional, generics, and such higher-order functions. It is compatible with macOS, iOS, watchOS, and tvOS.

20. What are the important features of Swift?

Swift programming language is being designed so that developers can write correct programs and maintain them easily. It offers the following features:

- **Safety:** Swift is an efficient way to write programs. Checking code before it is used in production is very important. Apple Swift removes any unsafe code before it is used in production.
- **Simple syntax:** Swift's syntax is simple and easy-to-use, just as developers would expect it. The syntax features of Swift enable you to write more expressive code.
- **Readability:** Swift has a simple syntax, which is easier to read and write. It is easier for developers to write Swift code since it is more similar to plain English, enabling them to spend less time looking for problematic code.
- **Multiplatform support**: Swift is fully compatible with iOS, macOS, tvOS, watchOS, Linux, and many other platforms. This means you are able to develop software that is compatible with all operating systems.
- **Open-source:** Swift is developed at swift.org, an open-source framework. For Swift to become a defining programming language, the technology had to be open to all. Swift supports all Apple platforms and makes programming easier, faster, and safer.
- **Compatible with Objective C:** Swift has full compatibility with Objective-C. Swift enables programmers to import frameworks from Objective-C using the Swift syntax. Programmers can utilize Objective-C libraries and classes inside Swift code.

21. Explain NSError in Swift.

Information about an error condition is encapsulated within an NSError object in an extendable and object-oriented manner. An NSError object is comprised of three basic attributes: a predefined error domain (represented as a string), a domain-specific error code, and a user info dictionary containing application-specific information.

Example: The examples below show how to create custom errors.

NSString *domain = @"com.MyCompany.MyApplication.ErrorDomain";

NSString *desc = NSLocalizedString(@"Unable to complete the process", @"");

NSDictionary *userInfo = @{ NSLocalizedDescriptionKey : desc };

NSError *error = [NSError errorWithDomain:domain code:-101 userInfo:userInfo];

22. What is Enum or Enumerations in Swift?

The term enumeration refers to a user-defined data type consisting of a set of related values that allows you to work with those values in your code in a type-safe manner. An enumerated data type is defined by the keyword enum.

Syntax:

```
enum enum_name
{
// enumeration values are described here
}
```

Example:

```
enum MonthsofaYear
{
case January
case Februrary
…
case December
}
```

Values defined in an enumeration MonthsofaYear such as January, February, and so on until December are its enumeration cases. New enumeration cases can be introduced using the case keyword. You can put multiple cases on a single line, separated by commas as follows:

```
enum MonthsofaYear
{
   case January, February,....,December
}
```

23. What do you mean by lazy property in iOS?

Lazy properties are properties whose initial value isn't computed until the first time they are used. Including the lazy modifier/keyword before the declaration of a stored property indicates it is lazy. This lets you delay the initialization of stored properties. This could be a great way to streamline your code and reduce unnecessary work. When a code is expensive and unlikely to be called consistently, a lazy variable can be a great solution.

Example:

```
class Person {
var name: String
lazy var personalizdgreeting : String = {
      return “HelloScala \(self.name)!”
}()
init(name: String) {
           self.name = name
            }
}
```

As shown above, we aren't sure what value personalizedgreeting should have. To know the correct greeting for this person, we need to wait until this object is initialized.

24. What are generics in swift and write its usage?

A major feature of Swift is generics, and much of the Swift standard library is written in generic code. Swift's 'Array' and 'Dictionary' types, for example, constitute generic collections. Generic code allows you to create flexible, reusable functions and types that work with any data type. You can create code that does not get too specific about underlying data types, resulting in cleaner code.

Example:

```
func Swapping(x: inout Int, y: inout Int)
{
    let temp = x
    x = y
    y = temp
}
var num1 = 10
var num2 = 50
print("Before Swapping: \(num1) and \(num2)")
Swapping(x: &num1, y: &num2)
print("After Swapping: \(num1) and \(num2)")
```

Output:

```
Before Swapping: 10 and 50
After Swapping: 50 and 10
```

In the above example, we have defined a function called Swapping() to swap integers. It takes two parameters x and y of int type. As seen in the output, the values of x and y are exchanged after the swapping.

25. Explain dictionary in Swift.

Swift programming defines a dictionary as an unordered collection of items. It stores items in key-value pairs. A dictionary uses a unique identifier called a key to store an associated value which can later be referenced and retrieved through the same key.

Syntax:

```
var Dict_name = [KeyType: ValueType]()
```

Example:

```
var examresult= ["Rahul": "79", "Aditya": "86", "Aditi": "67"]
print(examresult)
```

Output:

```
["Rahul": "79", "Aditya": "86", "Aditi": "67"]
```

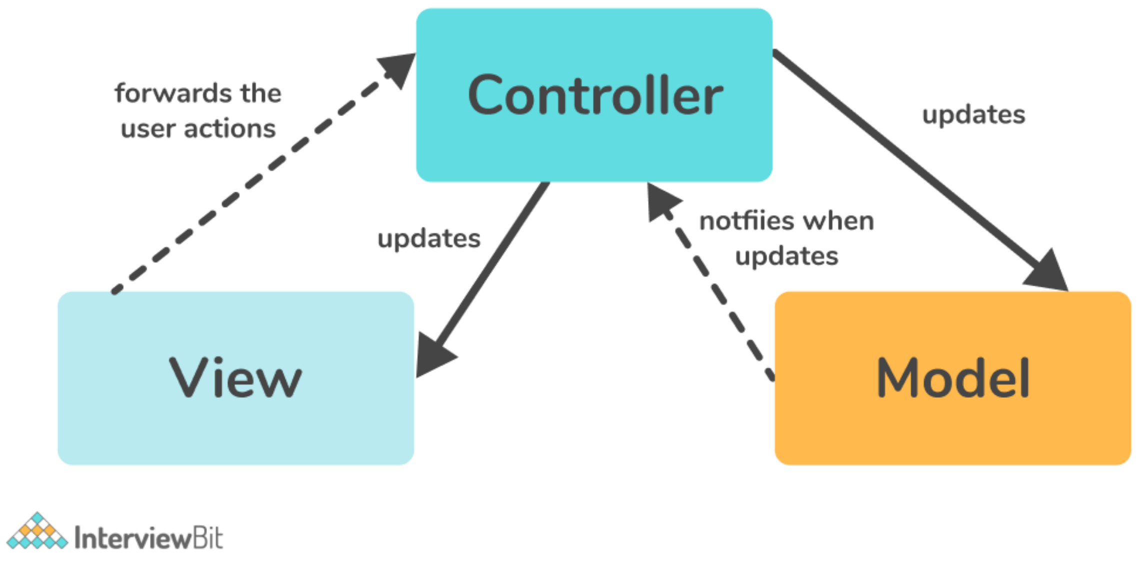
In the above example, we have created a dictionary named examresult. Here,

- **Keys** are "Rahul", "Aditya", "Aditi"
- **Values** are "79", "86", "67"

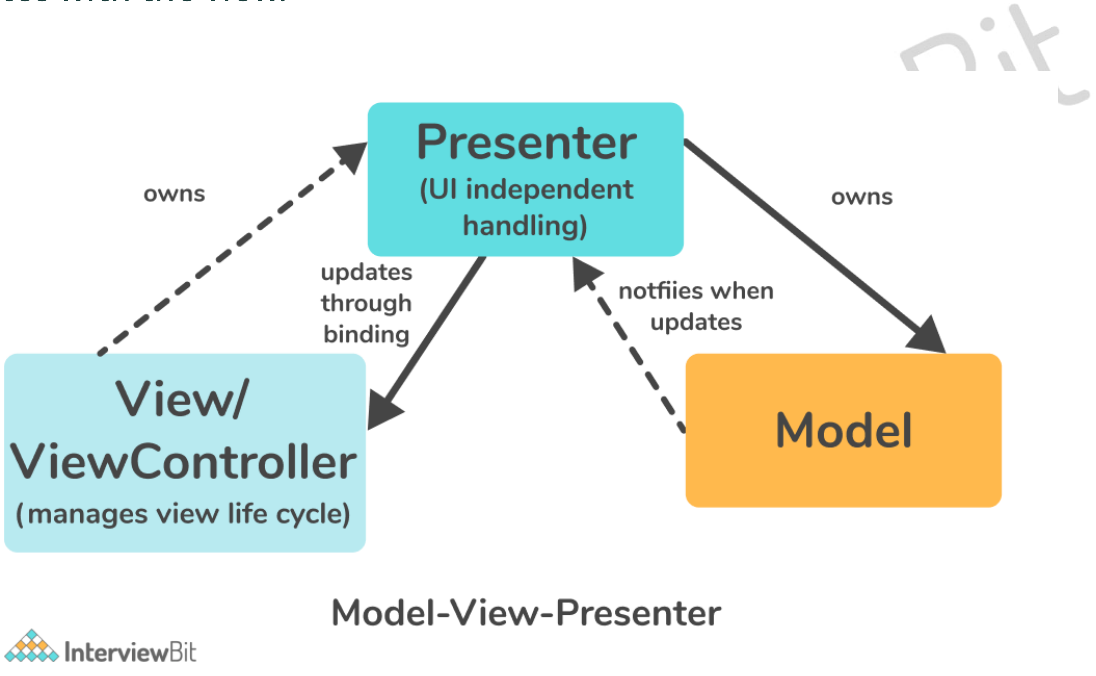
26. Why design patterns are important? Name some of the popular design patterns used in iOS?

A design pattern is a solution to a specific problem you might encounter while designing an app's architecture. They are templates designed to make coding easier and more reusable for you. Following are some of the design patterns that can be used in iOS:

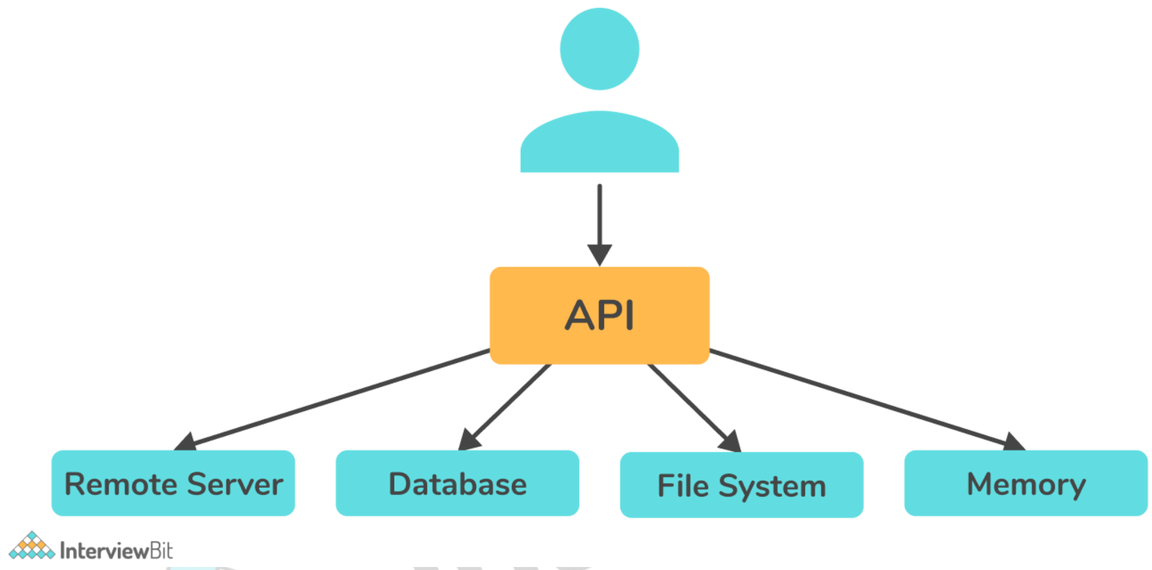
MVC (Model-View-Controller): MVC is a design pattern for developing web applications in which three components are connected i.e., view, controller, and model. Views can be conceived as the user interface shown to the end-user at a certain point in time. The model represents the data to be shown on the view. The controller acts as a link between view and model. There is always a relationship between these three components.



MVVM (Model-View-View Model): Unlike MVM, there is a special layer in MVVM called the View Model between the View and the Model. Using a view model, model information can be transformed into values that can be displayed on the view. There is a binding link between the view and View model that allows the view model to share updates with the view.



Facade: The facade design pattern provides a simplified (but limited) interface to a complex set of classes, libraries or frameworks. As opposed to providing the user with a set of classes and their APIs, you instead provide a simple, unified API. While Facade helps to reduce overall application complexity, it also helps to move unwanted dependencies to a single location.

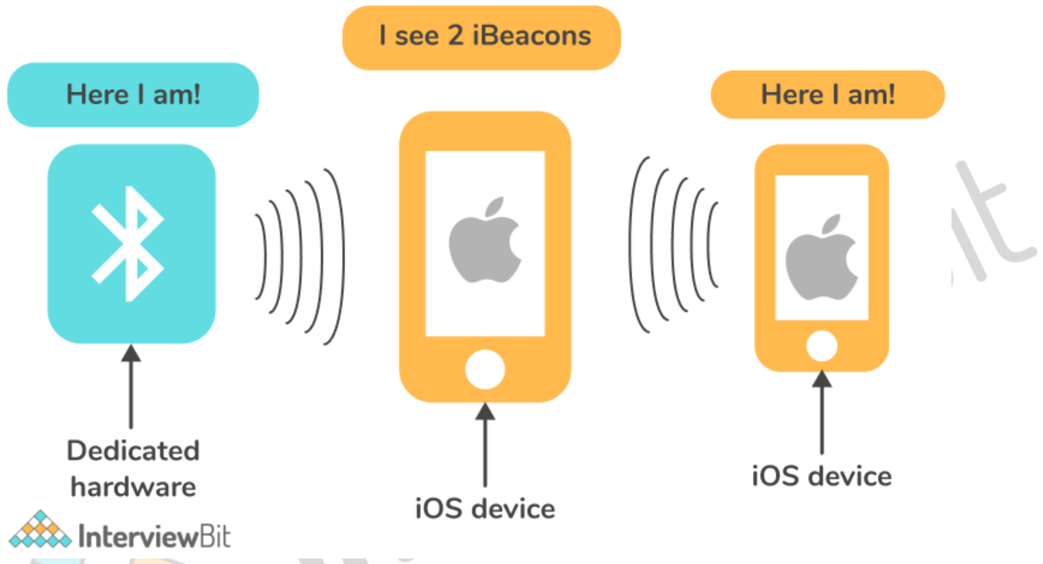


27. What is the JSON framework supported by iOS?

iOS supports the SBJson framework as the JSON framework. Humans and computers alike can easily read and write this lightweight data exchange formatter. JSON handling is simplified with SBJson's flexible APIs and additional control.

28. Explain iBeacons in iOS.

iBeacon, Apple's new low-energy Bluetooth wireless technology, allows iPhone and other iOS users to receive location-based information and services on smartphones. IBeacons are small, wireless transmitters that transmit signals to nearby smart devices via Bluetooth low energy technology.



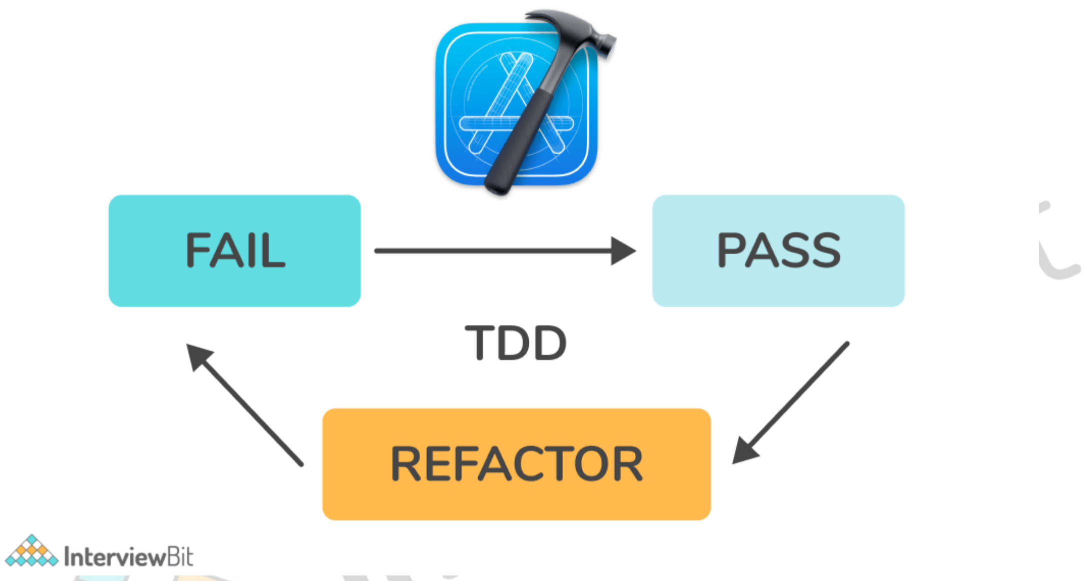
29. State the difference between KVC and KVO in Swift.

KVC (Key-Value Coding): It enables object properties to be accessed at runtime using strings rather than knowing the property names statically during development.

KVO (Key-Value Observing): In Objective-C and Swift, KVO is one of the methods for observing program state changes. If an object has instance variables, KVO enables other objects to observe the changes to those variables.

30. Explain TDD (Test-Driven Development).

Software developers can use testing-driven development (TDD) in the development of software. In TDD, developers plan the features of the software that they want to create and then write test cases for each feature before implementing it. Through test-driven development, we can gain insight into both the quality of the implementation (does it work) and the quality of the design (is it well structured).



At first, the test case will fail because the code is not yet implemented, and this is usually referred to as the red phase. After that, code is written to ensure that the test case passes and does not break any components or current test cases, which is called the green phase. The developer should then refactor the implementation of the code by cleaning and maintaining the codebase, as well as optimizing the efficiency. This process should then be repeated every time a new test case is added.

31. Explain the function of the completion handler.

Completion handlers are basically just functions passed as parameters to other functions. They are used to dealing with the response of asynchronous tasks since we do not know when they will end. Completion handlers inform an application when an operation, such as an API call, has been completed. The program is informed that the next step needs to be executed.

Example:

Let's create a simple class called CompletionHandler that has one method called count that counts from 0 to 50. Once it reaches 25 (a random value), it will make a network request to https://scaler.com. Once the request has been completed, we will print Received response.

```
class CompletionHandler {
    func count() {
        for i in 0...50 {
            if i == 25 {
                if let url = URL(string: "https://scaler.com") {
                    URLSession.shared.dataTask(with: url) { (data, response, error) in
                        print("Received response")
                    }.resume()
                }
            }
            print("I = ", i)
        }
    }
}
let newInstance = CompletionHandler()
newInstance.count()
```

You will see all the numbers printed out in the console as soon as the code runs, but the Received response will be printed only after all those other numbers have been printed.

32. State the difference between strong, weak, read only and copy.

The difference between strong, weak, read-only and copy is as follows:

- **Strong:** This property maintains a reference to property throughout the lifetime of an object. When you declare strong, you intend to "own" the object you are referencing. Data you assign to this property will not be destroyed as long as you or any other object references it strongly.
- **Weak:** It means that the object should be kept in memory as long as someone points to it strongly, and you don't need control over its lifetime.
- **Read-only:** An object's property can be defined initially, but it cannot be altered or modified.
- **Copy:** This attribute is an alternative to strong. In place of taking ownership of a current object, it creates a copy of whatever you assign to the property, then takes ownership of that copy.

33. What do you mean by dynamic dispatch?

In simple terms, dynamic dispatch means that the program decides at runtime which implementation of a particular method or function it needs to invoke. In the case where a subclass overrides a method of its superclass, dynamic dispatch determines whether to invoke the subclass implementation of the method or the parents.

34. Explain the @dynamic and @synthesize commands in Objective-C.

- **@synthesize:** This command generates getter and setter methods within the property and works in conjunction with the @dynamic command. By default, @synthesize creates a variable with the same name as the target of set/get as shown in the below example.
    - **Example1:**
      @property (nonatomic, retain) NSButton *someButton;
      ...
      @synthesize someButton;
      It is also possible to specify a member variable as its set / get target as shown below:
    - **Example2:**
      @property (nonatomic, retain) NSButton *someButton;
      ...
      @synthesize someButton= _homeButton;
      Here, the set / get method points to the member variable _homeButton.
- **@dynamic:** This tells the compiler that the getter and setter methods are not implemented within the class itself, but elsewhere (like the superclass or that they will be available at runtime).
    - **Example:**
      @property (nonatomic, retain) IBOutlet NSButton *someButton;
      ...
      @dynamic someButton;

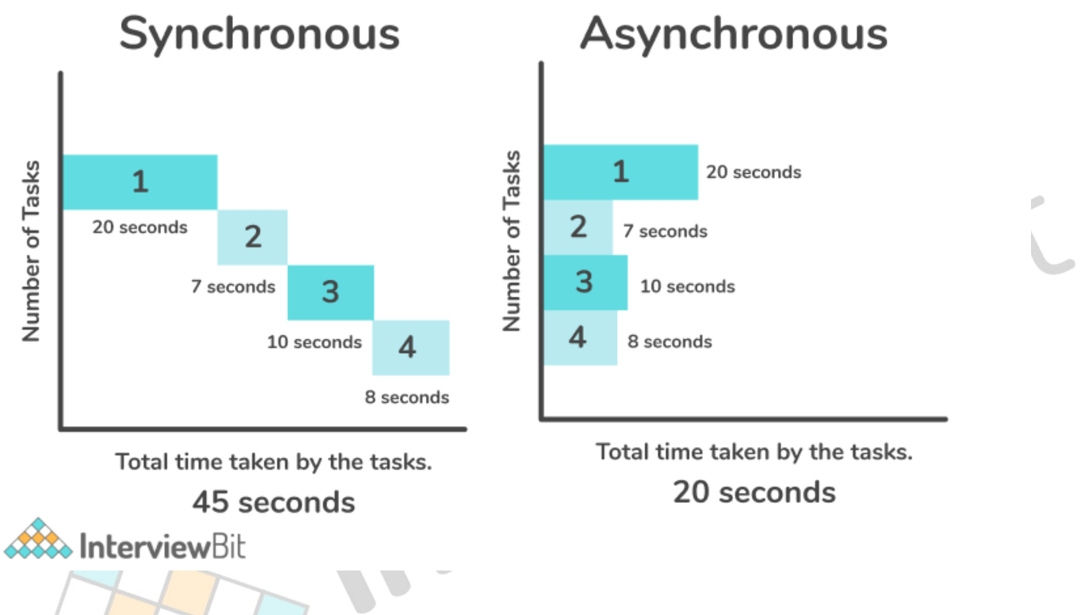
35. How can you implement storage and persistence in iOS?

Persistence means storing data on the disk so that it can be retrieved without being altered the next time the app is opened. From simple to complex, there are the following methods for storing data:

- Data structures such as arrays, dictionaries, sets, and other data structures are perfect for storing data intermediately.
- NSUserDefaults and Keychains are both simple key-value stores. NSUserDefaults is insecure, whereas Keychains is secure.
- A file or disk storage is a way to store data (serialized or not) to or from a disk using NSFileManager.
- Relational databases, such as SQLite, are good for implementing complex querying mechanisms.

36. What are synchronous and asynchronous tasks in iOS? In what way can you execute asynchronous tasks in iOS?

Apple's iOS supports both synchronous and asynchronous tasks. In synchronous operations, tasks are performed one at a time. Therefore, other tasks must wait until the previous task is completed before continuing. Asynchronous tasks run simultaneously in the background. If background tasks are completed, you will be notified. Asynchronous programming allows you to process multiple requests at the same time, so you can accomplish more tasks in a shorter amount of time.



The figure clearly indicates that synchronous tasks take more time to complete, i.e. 45 seconds since each task is performed one at a time. Asynchronous tasks, on the other hand, take less time to complete i.e. 20 seconds as they run simultaneously.

Async work can be handled using third-party libraries. A few notable examples are Promises (PromiseKit), RxSwift, and ReactiveCocoa.

Conclusion

Multibillions of Apple devices are in use worldwide. The number of iOS users worldwide has been rising steadily, which bodes well for iOS app developers. If you are seeking to make a career in iOS application development, then InterviewBit is the ideal platform to start with. In this article, we have put together 35+ of the most commonly asked iOS interview questions to help you succeed at your iOS job interview. iOS developers also need to stay apprised of changes in the iOS community. Make sure you read Apple developer news, listen to podcasts and read blogs as well. This is likely not to be a question in an interview, but it makes you stand out.

Hopefully, these answers will be helpful in better understanding iOS basics, Swift, and advanced topics. Anyone looking to succeed in an iOS interview would benefit from knowing the answers to these Swift and iOS developer interview questions. Good luck with your interview!

Useful Resources:

Android

Operating System

React Native

Links to More Interview Questions

C Interview Questions	Php Interview Questions	C Sharp Interview Questions
Web Api Interview Questions	Hibernate Interview Questions	Node Js Interview Questions
Cpp Interview Questions	Oops Interview Questions	Devops Interview Questions
Machine Learning Interview Questions	Docker Interview Questions	Mysql Interview Questions
Css Interview Questions	Laravel Interview Questions	Asp Net Interview Questions
Django Interview Questions	Dot Net Interview Questions	Kubernetes Interview Questions
Operating System Interview Questions	React Native Interview Questions	Aws Interview Questions
Git Interview Questions	Java 8 Interview Questions	Mongodb Interview Questions
Dbms Interview Questions	Spring Boot Interview Questions	Power Bi Interview Questions
Pl Sql Interview Questions	Tableau Interview Questions	Linux Interview Questions
Ansible Interview Questions	Java Interview Questions	Jenkins Interview Questions