


# iOS Interview questions and answers Part14

 Sandeep Reddy Challa · Follow  
6 min read · Apr 4

👤 3   🔍   📌   ⌚   📄

## 142.Mobile Device Management (MDM) in iOS?

The Mobile Device Management (MDM) protocol provides a way for system administrators to send device management commands to managed iOS devices running iOS 4 and later, macOS devices running macOS v10.7 and later, and Apple TV devices running iOS 7 (Apple TV software 6.0) and later. MDM allows for OS level control of multiple devices from a centralised location. A remote administrator can install/remove apps, install/revoke certificates, lock the device, change password requirements, etc.

## 143.Which is testing framework is best?

This really depends on which day you ask me that. I've been working with Expecta on XCTest recently, and I like that combination, but there are things that the other frameworks appear to do better. Here's a comparison of some key features:

## 144.How much test coverage is enough?

As much as you can. Programmer's often panic when asked to provide 100%, so instead I just say "test everything".

## 145.Should I unit test views and/or view controllers?

You should test every line of code that you write. If you are inheriting legacy code, then add tests for every line of code that you plan to modify.

## 146.Doesn't adding unit tests take more time?

That depends on which times you are comparing. If you compare the sum of the times spent writing the code, debugging the code, and later on extending the code, then unit tests should be the hands-down winner. If you are only comparing the time spent writing the code, then clearly it takes less time to write buggy, unmaintainable code that might not work correctly.

## 147.setUp() And tearDown() in XCTest?

- setUp() — This method is called before the invocation of each test method in the given class.
- tearDown() — This method is called after the invocation of each test method in given class.

You can customise setUp()and tearDown() in 5 different ways,

1. Override setUp() class method to setup initial state for all test methods.
2. Override setUp() instance method to reset initial state before each test method is executed.
3. You can have self-contained blocks of teardown code with the addTearDownBlock(...) method during a test method's execution.
4. Override tearDown() instance method to clean after each test method executes.
5. Override tearDown() class method to clean after all test methods execute.

## 148.Designated Initialisers and Convenience Initialisers?

Convenience initialisers are used when you have some class with a lot of properties that makes it kind of "Painful" to always initialise wit with all that variables, so what you do with convenience initialiser is that you just pass some of the variables to initialise the object, and assign the rest with a default value.Here is an example where you can see that instead of initialising my object with all those variables Im just giving it a title.

## 149.Local, Global and Static variables?

- **scope**, which determines where a name can be accessed — global and local
- **storage duration**, which determines when a variable is created and destroyed — static and auto

**Local:** The variables which are declared inside the function, compound statement (or block) are called Local variables.

**Global:** The variables declared outside any function are called global variables. They are not limited to any function. Any function can access and modify global variables. Global variables are automatically initialised to 0 at the time of declaration.

**Static:** A Static variable is able to retain its value between different function calls. The static variable is only initialised once, if it is not initialised, then it is automatically initialised to 0. Here is how to declare a static variable, Static variables have a lifetime that lasts until the end of the program. If they are local variables, then their value persists when execution leaves their scope.

**Auto:** Automatic variables are local variables whose lifetime ends when execution leaves their scope, and are recreated when the scope is reentered.

## 150.Which Inheritance will supports in Swift Single, Multiple or Multilevel?

Swift supports single inheritance, we can approach multiple inheritance using protocols and multilevel inheritance it won't support in Swift.

**Single:** a class is allowed to inherit from only one class. i.e. one sub class is inherited by one base class only.

**Multiple:** When a derived class is derived from more than one base class, it is known as multiple inheritance.

**Multilevel:** When one class inherits the properties of another class, which is further inherited by another class, it is known as multilevel inheritance.

## 151.Debugging skills in iOS

- Breakpoints
- LLDB(pop, p, v)
- Devices's Container
- Network Link Conditioner
- Identifying the issues which occur only on device(and not in debug mode)
- View Hierarchy

## 152.What difference between 'self' and 'Self'?

When you're writing protocols and protocol extensions, there's a difference between Self (capital S) and self (lowercase S). When used with a capital S, Self refers to the type that conform to the protocol, e.g. String or Int. When used with a lowercase S, self refers to the value inside that type, e.g. "hello Swift" or 786.

## 153.What's the difference between == and ===?

Swift gives us two equality operators, == and ===, that do slightly different things. You will almost certainly need to use both of them so it's worth taking the time to learn them.

First, == is the equality operator, which tests that two things are equal for whatever definition of "equal" those things use. For example, 5 == 5 is true because there == means an integer comparison, and the same is true for other built-in value types such as strings, booleans, and doubles.

Things get more complicated when == is used with a struct you built, because by default they cannot be compared — you need to make them conform to the Equatable protocol.

In comparison, === is the identity operator, which checks whether two instances of a class point to the same memory. This is different from equality, because two objects that were created independently using the same values will be considered equal using == but not === because they are different objects.

The === operator is available only when using classes because structs are designed so they are always uniquely referenced.

## 154.Write a Palindrome program

## 155.What is QualityOfService in operation?

Used to indicate the nature and importance of work to the system. Work with higher quality of service classes receive more resources than work with lower quality of service classes whenever there is resource contention

- This property specifies the service level applied to operation objects added to the

queue • If the operation object has an explicit service level set, that value is used instead.

The default value of this property depends on how you created the queue. For queues you create yourself, the default value is NSOperationQualityOfServiceBackground. For the queue returned by the main method, the default value is NSOperationQualityOfServiceUserInteractive and cannot be changed.

<https://www.allaboutswift.com/dev/2016/5/21/gcd-with-qos-in-swift>

Types of QualityOfService • User Interactive All UI related tasks need to be assigned that QoS class. It basically tells iOS to run them on the main thread.

- User Initiated This is the class to choose if a task is triggered by the user but doesn't have to be run necessarily on the main thread.

- Default This class is only listed for informational purposes only. The system defaults to this class when not enough information is available to determine QoS.

- Utility This class needs to be chosen for tasks that have a dependency on other system resources like file I/O or network I/O. Those tasks need to wait for a certain amount of time before they are able to complete. They are always in one way or another initiated by the user who waits for their completion.

- Background This class is supposed to be used for maintenance tasks — Tasks that don't depend on their fast execution and that are oblivious to the user.

- Grand Central dispatch (Multi threading concept) GCD is a library that provides a low-level and object-based API to run tasks concurrently while managing threads behind the scenes. Terminology

- Dispatch Queues:- A dispatch queue is responsible for executing a task in the first-in, first-out order

- Serial Dispatch Queue :- A serial dispatch queue runs tasks one at a time.

- Concurrent Dispatch Queue:- A concurrent dispatch queue runs as many tasks as it can without waiting for the started tasks to finish.

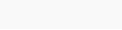
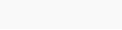
- Main Dispatch Queue:- A globally available serial queue that executes tasks on the application's main thread.

Objective C   Swift   iOS App Development   Xcode   Unitt Testing

👤 3   🔍   📌   ⌚   📄

 Written by Sandeep Reddy Challa  
171 Followers  
I have around 14 years of experience in iOS development and lead/senior developer

## More from Sandeep Reddy Challa

 Sandeep Reddy Challa    Sandeep Reddy Challa

### iOS Interview questions and answers Part25

6 min read · Nov 7

👤 1   🔍   📌   ⌚   📄

### iOS Interview questions and answers Part24

3 min read · Nov 7

👤 1   🔍   📌   ⌚   📄

### iOS Interview questions and answers Part26

274:iOS View Controller Life Cycle ?

5 min read · Nov 7

👤 6   🔍   📌   ⌚   📄

### iOS Interview questions and answers Part1

1-What is an iOS application and where does your code fit into it?

11 min read · Apr 28, 2021

👤 58   🔍   📌   ⌚   📄

See all from Sandeep Reddy Challa

## Recommended from Medium

 Sandeep Reddy Challa    Rohini vaidya

### iOS Interview questions and answers Part23

7 min read · Sep 4

👤 6   🔍   📌   ⌚   📄

### Basics on multi-threading in iOS using GCD

Multithreading is a concept of performing various tasks using different threads that th...

6 min read · Jul 25

👤 6   🔍   📌   ⌚   📄

## Lists

**Staff Picks**  
539 stories · 553 saves

 Nirajpaul Ios    Baljit Kaur in Swift Interview Preparations

### iOS Interview Question (Part 7)

1.What is type checking operator in Swift?

3 min read · Jul 16

👤 10   🔍   📌   ⌚   📄

### iOS Interview Questions-Part 1

Interview Questions about swift, iOS, Xcode

4 min read · Oct 22

👤 10   🔍   📌   ⌚   📄

 Nanesh Kukkala    Shafique Dasso

### Delegate vs Closure in iOS Development...

Hey folks! 🙌 Today, I want to shed some light on the differences between delegates and...

3 min read · Oct 6

👤 4   🔍   📌   ⌚   📄

### Enums in Swift: A Comprehensive Guide

By its dictionary definition, an enumeration (or 'enum' for short) is a comprehensive...

4 min read · Nov 14

👤 1   🔍   📌   ⌚   📄

See more recommendations

Help   Status   About   Careers   Blog   Privacy   Terms   Text to speech   Tears