iOS QUESTIONS

1- How could you setup Live Rendering ?

The attribute @IBDesignable lets Interface Builder perform live updates on a particular view.

2- What is the difference between Synchronous & Asynchronous task ?

Synchronous: waits until the task has completed **Asynchronous**: completes a task in background and can notify you when complete

3- What Are B-Trees?

B-trees are search trees that provide an ordered key-value store with excellent performance characteristics. In principle, each node maintains a sorted array of its own elements, and another array for its children.

4- What is made up of NSError object?

There are three parts of NSError object a domain, an error code, and a user info dictionary. The domain is a string that identifies what categories of errors this error is coming from.

5- What is @synthesize in Objective-C ?

synthesize generates getter and setter methods for your property.

6- What is @dynamic in Objective-C ?

We use dynamic for subclasses of NSManagedObject. **@dynamic** tells the compiler that getter and setters are implemented somewhere else.

7- What is the difference strong, weaks, read only and copy ?

strong, weak, assign property attributes define how memory for that property will be managed.

Strong means that the reference count will be increased and

the reference to it will be maintained through the life of the object

Weak, means that we are pointing to an object but not increasing its reference count. It's often used when creating a parent child relationship. The parent has a strong reference to the child but the child only has a weak reference to the parent.

Read only, we can set the property initially but then it can't be changed.

Copy, means that we're copying the value of the object when it's created. Also prevents its value from changing.

for more details check this out

8- What is Responder Chain ?

A ResponderChain is a hierarchy of objects that have the opportunity to respond to events received.

9- What is Decorator Design Pattern ?

In Objective-C there are two *very* common implementations of this pattern: **Category** and **Delegation**. In Swift there are also two *very* common implementations of this pattern: **Extensions** and **Delegation**.

10- Explain MVC

- **Models** — responsible for the domain data or a data access layer which manipulates the data, think of 'Person' or 'PersonDataProvider' classes.
- **Views** — responsible for the presentation layer (GUI), for iOS environment think of everything starting with 'UI' prefix.
- **Controller/Presenter/ViewModel** — the glue or the mediator between the Model and the View, in general responsible for altering the Model by reacting to the user's actions performed on the View and updating the View with changes from the Model.

11- Explain MVVM

UIKit independent representation of your View and its state. The View Model invokes changes in the Model and updates itself with the updated Model, and since we have a binding between the View and the View Model, the first is updated accordingly.

Your view model will actually take in your model, and it can format the information that's going to be displayed on your view.

There is a more known framework called RxSwift. It contains RxCocoa, which are reactive extensions for Cocoa and CocoaTouch.

12- What is Realm benefits ?

- An open-source database framework.
- Implemented from scratch.
- Zero copy object store.
- Fast.

13- What is the Swift main advantage ?

To mention some of the main advantages of Swift:

- Optional Types, which make applications crash-resistant
- Built-in error handling
- Closures
- Much faster compared to other languages
- Type-safe language
- Supports pattern matching

14- Explain lazy in Swift ?

An initial value of the lazy stored properties is calculated only when the property is called for the first time. There are situations when the lazyproperties come very handy to developers.

15- Explain what is defer ?

defer keyword which provides a block of code that will be executed in the case when execution is leaving the current scope.

16- How to pass a variable as a reference ?

We need to mention that there are two types of variables: reference and value types. The difference between these two types is that by passing value type, the variable will create a copy of its data, and the reference type variable will just point to the original data in the memory.

17- What is Concurrency ?

Concurrency is dividing up the execution paths of your program so that they are possibly running at the same time. The common terminology: process, thread, multithreading, and others. Terminology;

- **Process,** An instance of an executing app
- **Thread,** Path of execution for code
- **Multithreading,** Multiple threads or multiple paths of execution running at the same time.
- **Concurrency,** Execute multiple tasks at the same time in a scalable manner.
- **Queues,** Queues are lightweight data structures that manage objects in the order of First-in, First-out (FIFO).
- **Synchronous** vs **Asynchronous** tasks

18- Grand Central Dispatch (GCD)

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.

19- Readers-Writers

Multiple threads reading at the same time while there should be only one thread writing. The solution to the problem is a readers-writers lock which allows concurrent read-only access and an exclusive write access. Terminology;

- **Race Condition** A race condition occurs when two or more threads can access shared data and they try to change it at the same time.
- **Deadlock** A deadlock occurs when two or sometimes more tasks wait for the other to finish, and neither ever does.
- **Readers-Writers problem** Multiple threads reading at the same time while there should be only one thread writing.
- **Readers-writer lock** Such a lock allows concurrent read-only access to the shared resource while write operations require exclusive access.
- **Dispatch Barrier Block** Dispatch barrier blocks create a serial-style bottleneck when working with concurrent queues.

20- NSOperation — NSOperationQueue — NSBlockOperation

NSOperation adds a little extra overhead compared to GCD, but we can add dependency among various operations and re-use, cancel or suspend them.

NSOperationQueue, It allows a pool of threads to be created and used to execute NSOperations in parallel. Operation queues aren't part of GCD.

NSBlockOperation allows you to create an NSOperation from one or more closures. NSBlockOperations can have multiple blocks, that run concurrently.

21- KVC — KVO

KVC adds stands for Key-Value Coding. It's a mechanism by which an object's properties can be accessed using string's at runtime rather than having to statically know the property names at development time.

KVO stands for Key-Value Observing and allows a controller or class to observe changes to a property value. In KVO, an object can ask to be notified of any changes to a specific property, whenever that property changes value, the observer is automatically notified.

22- What is the difference Non-Escaping and Escaping Closures ?

The lifecycle of a non-escaping closure is simple:

1. Pass a closure into a function
2. The function runs the closure (or not)
3. The function returns

Escaping closure means, inside the function, you can still run the closure (or not); the extra bit of the closure is stored some place that will outlive the function. There are several ways to have a closure escape its containing function:

- **Asynchronous execution**: If you execute the closure asynchronously on a dispatch queue, the queue will hold onto the closure for you. You have no idea *when* the closure will be executed and there's no guarantee it will complete before the function returns.
- **Storage**: Storing the closure to a global variable, property, or any other bit of storage that lives on past the function call means the closure has also escaped.

for more details.

23- Explain [weak self] and [unowned self] ?

unowned does the same as weak with one exception: The variable will not become nil and therefore the variable must not be an optional.

But when you try to access the variable after its instance has been deallocated. That means, you should only use **unowned** when you are sure, that this variable will never be accessed after the corresponding instance has been deallocated.

However, if you don't want the variable to be weak AND you are sure that it can't be accessed after the corresponding instance has been deallocated, you can use unowned.

By declaring it **[weak self]** you get to handle the case that it might be nil inside the closure at some point and therefore the variable must be an optional. A case for using **[weak self]** in an asynchronous network request, is in a **view controller** where that request is used to populate the view.

24- What is ARC ?

ARC is a compile time feature that is Apple's version of automated memory management. It stands for *Automatic Reference Counting.* This means that it ***only*** frees up memory for objects when there are ***zero strong*** references/ to them.

25- What is iOS 11 SDK Features for Developers ?

- New MapKit Markers
- Configurable File Headers
- Block Based UITableView Updates
- MapKit Clustering
- Closure Based KVO
- Vector UIImage Support
- New MapKit Display Type

- Named colors in Asset Catalog
- Password Autofill
- Face landmarks, Barcode and Text Detection
- Multitasking using the new floating Dock, slide-over apps, pinned apps, and the new App Switcher
- Location Permission: A flashing blue status bar anytime an app is collecting your location data in the background. Updated locations permissions that always give the user the ability to choose only to share location while using the app.

26- What is NSFetchRequest ?

NSFetchRequest is the class responsible for fetching from Core Data. Fetch requests are both powerful and flexible. You can use fetch requests to fetch a set of objects meeting the provided criteria, individual values and more.

27- What is the difference open & public access level ?

open allows other modules to use the class and inherit the class; for members, it allows others modules to use the member and override it.

public only allows other modules to use the public classes and the public members. Public classes can no longer be subclassed, nor public members can be overridden.

28- What is the difference fileprivate, private and public private(set) access level ?

fileprivate is accessible within the current file, **private** is accessible within the current declaration.

public private(set) means getter is public, but the setter is private.

29- What is Internal access ?

Internal access enables entities to be used within any source file from their defining module, but not in any source file outside of the module.

Internal access is the default level of access. So even though we haven't been writing any access control specifiers in our code, our code has been at an internal level by default.

30- Where do we use Dependency Injection ?

We use a storyboard or xib in our iOS app, then we created IBOutlets. IBOutlet is a property related to a view. These are injected into the view controller when it is instantiated, which is essentially a form of Dependency Injection.

There are forms of dependency injection: constructor injection, property injection and method injection.

31- What is CoreData ?

Core data is an object graph manager which also has the ability to persist object graphs to the persistent store on a disk. An object graph is like a map of all the different model objects in a typical model view controller iOS application. CoreData has also integration with Core Spotlight.

But Core Data is not thread safe, meaning that, if you load a managed object on one thread, you can't pass it to another thread and use it safely. This becomes an issue when we want to start introducing threading for performance, so we have two choices.

The first is to keep everything on the main thread, which just means it's single threaded. Or the second, means making changes on background threads and passing managed object IDs and then loading those objects again on the main thread, but that would mean that you're on the main thread, which puts us right back where we started. Both of these kind of ruin the point of using threading within Core Data and they can add a lot of complexity to the data layer.

There’s also another option for that and it’s to convert the managed object to a plain old Swift object, or a POSO.

32- Explain the difference between atomic and nonatomic synthesized properties

atomic : It is the default behaviour. If an object is declared as atomic then it becomes thread-safe. Thread-safe means, at a time only one thread of a particular instance of that class can have the control over that object.

nonatomic: It is not thread-safe. We can use the nonatomic property attribute to specify that synthesized accessors simply set or return a value directly, with no guarantees about what happens if that same value is accessed simultaneously from different threads. For this reason, it’s faster to access a nonatomic property than an atomic one.

33- Explain differences between Foundation and CoreFoundation

The Foundation is a gathering of classes for running with numbers, strings, and collections. It also describes protocols, functions, data types, and constants. CoreFoundation is a C-based alternative to Foundation. Foundation essentially is a CoreFoundation. We have a free bridge with NS counterpart.

34- What is Protocol?

A protocol defines a blueprint of methods, properties and other requirements that suit a particular task or piece of functionality. The protocol can then be adopted by a class, structure, or enumeration to provide an actual implementation of those requirements.

35- What is CocoaTouch ?

CocoaTouch is a library used to build executable applications on iOS. CocoaTouch is an abstract layer on the iOS.

36- Explain Viper Architecture

Viper is an another design patters. It has five layers: **View, Interactor, Presenter, Entity, and Router**. It is based on Single Responsibility Principle.

Advantages of Viper architecture is communication from one entity to another is given through protocols. The idea is to isolate our app's dependencies, balancing the delegation of responsibilities among the entities.

37- What is LLDB?

It's the debugger of The **LLVM** Compiler Infrastructure.

38- What is the meaning of id ?

id is a pointer to any type, it always points to an Objective-C object. The AnyObject protocol is similar and it helps bridge the gap between Swift and Objective-C.