final project

by Saksham R

Submission date: 25-Jun-2020 07:33PM (UTC+0530)

Submission ID: 1349533407

File name: 161364Finalreport.pdf (305.94K)

Word count: 4371

Character count: 22618

final project

ORIGINALITY REPORT

19%	18%	1%	11%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	251eac70a0b4ddddc93142c2ee8c9d1d08c26b8a.googlec Internet Source	6%
2	developer.apple.com Internet Source	4%
3	www.interviewsortout.com Internet Source	3%
4	docs.swift.org Internet Source	2%
5	reactnative.dev Internet Source	2%
6	mrmacintosh.com Internet Source	1%
7	shrikar.com Internet Source	<1%
8	www.code-brew.com Internet Source	<1%
9	stackoverflow.com Internet Source	<1%

10	Submitted to University of North Texas Student Paper	<1%
11	Submitted to Asia Pacific Instutute of Information Technology Student Paper	<1%
12	vishnukrish.blogspot.com Internet Source	<1%
13	Submitted to University of Newcastle upon Tyne Student Paper	<1%
14	Submitted to Kazakh-British Technical University Student Paper	<1%
15	smibusiness.com.au Internet Source	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		

iOS APP DEVELOPMENT

Industrial Training Report

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD OF THE DEGREE OF

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE ENGINEERING

By

Sakshum Kalsotra (161364)

UNDER THE GUIDANCE OF

Dr.Rajni Mohana

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

May 2020

DECLARATION

We hereby declare that the work reported in the B.Tech Project Report entitled IOS APP DEVELOPMENT submitted **at Jaypee University of Information Technology, Waknaghat** , India is an authentic record of our work carried out under the supervision of **Dr**.**Rajni Mohana** We have not submitted this work elsewhere for any other degree or diploma

Signature of the Student

Sakshum Kalsotra
161364

This is to certify that the above statement made by the candidates is correct to the best of my knowledge

Signature of the Supervisor

Dr.Rajni Mohana

Date:

Head of the Department /Project Coordinator

ACKNOWLEDGEMENT

It is my privilege to thank all the people who contributed to make this internship period a great and unforgettable experience of my life.

First of all I would like to thank **Jaypee University of Information Technology, Waknaghat** for providing me the golden opportunity of gaining practical industry knowledge through internship training at **CODEBREW LABS**. I take this opportunity to express my sincere gratitude and deep regards to my Mentor, **Mrs Gurleen Kaur** for her exemplary guidance, monitoring and constant encouragement throughout the course of this internship. I also thank all my team members for sharing the knowledge and experience of their respective fields which assisted me in the successfully blend in the ongoing live project.

Finally I thank each and every member of the **CodeBrew** family in technical, non-technical fields and support staff who have directly or indirectly helped me. This internship program provided me a lot of opportunities to work in different settings and exposed me to interaction with different sets of people globally. I was able to utilize and enhance different skills in me like decision making, communication, teamwork and coordination.

ABSTRACT

Code Brew Labs is specialized in Design, Development, Branding & Online Marketing. As a well established leading Web Development Company in India they have mastered the art of translating client ideas and expectations into high quality deliverable services. They believe in simplicity and prioritize every client's requirements with a great deal of commitment. Moreover, their services are backed up with sound knowledge and skills of technology, along with cost effective plans. Their global clientele is diverse and ranges from small startups to large enterprises.

Role: IOS APP DEVELOPER

iOS is Apple’s mobile OS that runs on an iPhone, iPad, iPod Touch hardware. Apple provides tools and resources for creating iOS apps and accessories for these devices. As an iOS developer, you can program in native languages such as Swift or Objective-C or build cross-platform native applications using React Native (JavaScript) or Xamarin (C# & F#).

Our team of iOS developers is responsible for developing applications for mobile devices powered by Apple's **IOS** operating system.

Responsibilities

- Design and build applications for the IOS platform
- Ensure the performance, quality, and responsiveness of applications
- Collaborate with a team to define, design, and ship new features
- Identify and correct bottlenecks and fix bugs
- Help maintain code quality, organization, and automatization.

LIST OF FIGURES

INDEX

COMPANY PROFILE

Code Brew Labs is specialized in Design, Development, Branding & Online Marketing. As a well established leading Mobile And Web Development Company in India they have mastered the art of translating client ideas and expectations into high quality deliverable services. They believe in simplicity and prioritize every client's requirements with a great deal of commitment. Moreover, their services are backed up with sound knowledge and skills of technology, along with cost effective plans. Their global clientele is diverse and ranges from small startups to large enterprises. **Code Brew Labs** is a firm specializing in technological solutions for businesses - the ones that exist, and the ones that are still in their nascent stages. We make technology an invaluable asset for your business - be it a startup, an SMB or an enterprise - all of these are included in their expanding repertoire. They have dabbled in Business Analytics, IoT based infrastructure, cloud-based computing and solutions, along with forays into the world of AR and VR.

They extend this expertise to many industry verticals, be it Fintech, EduTech, Healthcare, Social Networking, Chat Apps, Mobile Games, Marketplaces and Food Delivery - to name a few. Mobile App Development, iPhone App development, Android App development, 2D / 3D game development, Mobile Strategy, Minimum Viable Product for Startups, App Monetization and Engagement, App Store Optimization, React Native are some of the technologies they excel in.

SERVICES

Accelerated Mobile Page:- An open source project created to improve the performance of web pages for mobile devices. The technology behind AMP enables lightweight pages that load more quickly for Smartphones and tablets users. With the help of this technology, you will be able to publish content much faster. It does not matter where your business is situated as the project allows us to implement the system and adapt it to all kind of audiences.

Top Mobile App Development Services for Startups & Enterprises

Code Brew Labs is a top-notch Mobile App Development company in India that offers some of the best development services. Whether you are a startup or an enterprise company, you can avail our highly coveted iPhone and Android App development services which will be intuitive, powerful and engaging.

Digital Marketing

Digital Marketing has become the core of all the businesses, irrespective of their size. Code Brew Labs is proud to be one of the best Digital Marketing Company in India, where we practice all the latest white hat techniques in order to make a business visible on internet.

Top Mobile Game Development Services

Code Brew Labs is one of the most promising and delivering Mobile App Game development company in India. We have years of experience in creating quality and augmented gaming applications.

Products:

A) GRINTAFY

An extraordinary application solely for football darlings in Middle East which permits clients to make and advance profiles, play with companions, and let different players participate with 100k+downloads taking fever of football to the following level.

Illustration 1: GRINTAFY

B) RED VAULT

A Rewarding App for each gamer In Middle East. Super-dazzling designs and intuitive UI let the players play and gain more. Red vault effectively handles 90 percent of gaming business sector of GCC with 100K+downloads.

C) GRADEUP :-An Exam Preparation App, Subsidiary of Times Internet.

One of the largest & most effective platforms for competitive exam preparation in India with 10M+ downloads. Learning made accessible and affordable!

Illustration 2 : GradeUp

D) ERP SOLUTION FOR SML ISUZU:-ERP Solution to the commercial vehicle manufacturer.SML ISUZU is a renowned commercial vehicle manufacturer, specializing in buses, ambulances and customized vehicles.SML Isuzu Limited is a commercial vehicle manufacturer established in 1983.SMLI produces and sells buses, ambulances and customized vehicles.

Illustration 3 : SML ISUZU

The ‘BREW’ Procedure Helps Exceed the Business Goals

Briefing Of The Business Idea

Examining the ideas & details, we propose the best possible solutions that will meet your brand requirements indeed.

- **Thorough research to keep you ahead of your competitors**
- **A framework for your customized app with specific features**

Researching A Perfect Architecture

Browsing our stack of latest technologies, our brewers create a fully-customized solution, made just for your business

- **An exceptional UI design for better customer experience**
- **Development of the core functionality with advanced features Evolving Bug-Free Business App.**

METHEDOLOGY

Methodologies for building iOS apps are systematic ways of developing software systems (the how) while considering all the other relevant aspects (the why, what, when, who, and where) of these systems.

Basic Design Stages for iOS Apps:- To To see how to configuration forms, you should comprehend configuration stages for iOS applications. Programming improvement steps are requested in time by stages, beginning from a phase of small understanding and moving to phases of logically expanded information and consistency, with the exercises beginning with the patrons or the visionaries and moving to the advancement group and afterward to the client. There are four phases:

- **Inception:** The inception stage is where things are being figured out — an approximate vision; a business case; scope; a high-level, potential architecture; and high-level estimates of efforts and cost.

- **Elaboration:** This is the stage where you refine the vision, validate the core architecture, and resolve risks. In fact, you might say that elaboration is all about the *resolution* of the risks. This is where most of the requirements identification is done and (with hope) realistic project management estimates are created.

- **Construction:** The stage where you iteratively implement any remaining features and prepare for deployment. The construction stage is where the tasks are steadily adding more and more features. Plenty of detailed OO design, implementation, and testing takes place here.

- **Transition:** This is the stage where you deploy a finished released. Here, the software is turned over to users. Incidentally, you might also beta test a system in this stage. Activities from the deployment, testing, and maintenance phases take place here.

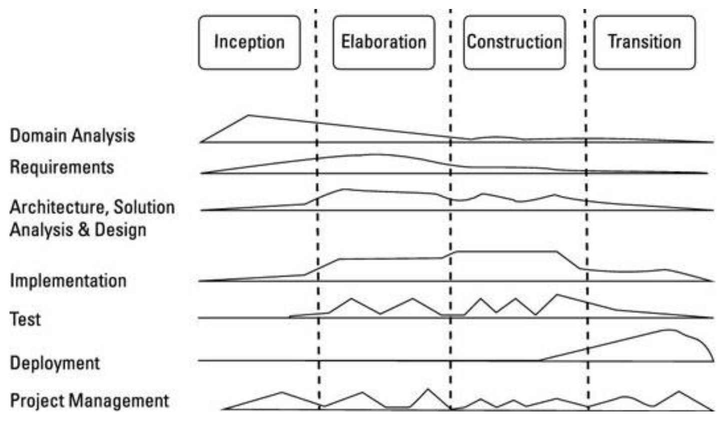


Illustration 5 :APPDESIGN

CHAPTER : 1

INTRODUCTION TO SWIFT

Swift is another programming language for iOS, macOS, watchOS, and tvOS application improvement. In any case, numerous pieces of Swift will be recognizable from your experience of creating in C and Objective-C.Swift gives its own renditions of all essential C and Objective-C types, including Int for whole numbers, Double and Float for coasting point esteems, Bool for Boolean qualities, and String for literary information. Quick likewise gives ground-breaking variants of the three essential assortment types, Array, Set, and Dictionary

Quick is a sort safe language, which implies the language causes you to be clear about the kinds of qualities your code can work with. On the off chance that piece of your code requires a String, type security keeps you from passing it an Int accidentally. In like manner, type security keeps you from incidentally passing a discretionary String to a bit of code that requires a non-discretionary String. Type security encourages you catch and fix blunders as ahead of schedule as conceivable in the improvement procedure.

Swift also introduces optional types, which handle the absence of a value. Optionals say either "there *is* a value, and it equals *x*" or "there *isn't* a value at all". Using optionals is similar to using nil with pointers in Objective-C, but they work for any type, not just classes. Not only are optionals safer and more expressive than nil pointers in Objective-C, they're at the heart of many of Swift's most powerful features.

Notwithstanding recognizable sorts, Swift presents propelled types not found in Objective-C, for example, tuples. Tuples empower you to make and go around groupings of qualities. You can utilize a tuple to restore different qualities from a capacity as a solitary compound worth.

1.1 Features of Swift

Swift Supports Dynamic Libraries

Dynamic libraries are executable pieces of code that can be connected to an application. This component permits current Swift applications to interface against fresher adaptations of the Swift language as it develops after some time. Dynamic libraries in Swift are legitimately transferred to the memory, eliminating the underlying size of the application and eventually expanding application execution.

Safer Platform

In the competitive mobile app marketplace, developing a secure app should be a priority. Swift's syntax and language constructions exclude the several types of mistakes possible in Objective-C. This stability means that there will be fewer crashes and cases of problematic behavior. It doesn't prevent programmers from writing bad code, but rather makes it less likely to make mistakes. This adds an extra layer of quality control during development.

Less Code & Less Legacy

With Objective-C, there are numerous issues that cause application crashes. Swift gives code that is less mistake inclined as a result of its inline support for controlling content strings and information. Moreover, classes aren't partitioned into two sections; the interface and the execution. This cuts the quantity of documents in the undertaking fifty-fifty, which makes it a lot simpler to deal with.

Automatic Reference Counting (ARC)

Swift uses *Automatic Reference Counting* (ARC) to track and manage your app’s memory usage. In most cases, this means that memory management “just works” in Swift, and you do not need to think about memory management yourself. ARC automatically frees up the memory used by class instances when those instances are no longer needed.

1.2 Swift Overview

Closures:-

Closures are self-contained blocks of functionality that can be passed around and used in your code. Closures in Swift are similar to blocks in C and Objective-C and to lambdas in other programming languages.Closures can capture and store references to any constants and variables from the context in which they are defined. This is known as *closing over* those constants and variables. Swift handles all of the memory management of capturing for you.

Enumerations:-

Enumerations in Swift are five star types in their own right. They receive numerous highlights generally bolstered uniquely by classes, for example, figured properties to give extra data about the identification's present worth, and case strategies to give usefulness identified with the qualities the specification speaks to. Lists can likewise characterize initializers to give an underlying case esteem; can be reached out to extend their usefulness past their unique execution; and can comply with conventions to give standard usefulness

Optional Chaining:-

Optional chaining is a process for querying and calling properties, methods, and subscripts on an optional that might currently be nil. If the optional contains a value, the property, method, or subscript call succeeds; if the optional is nil, the property, method, or subscript call returns nil. Multiple queries can be chained together, and the entire chain fails gracefully if any link in the chain is nil.

Protocols:- 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. Any type that satisfies the requirements of a protocol is said to *conform* to that protocol.

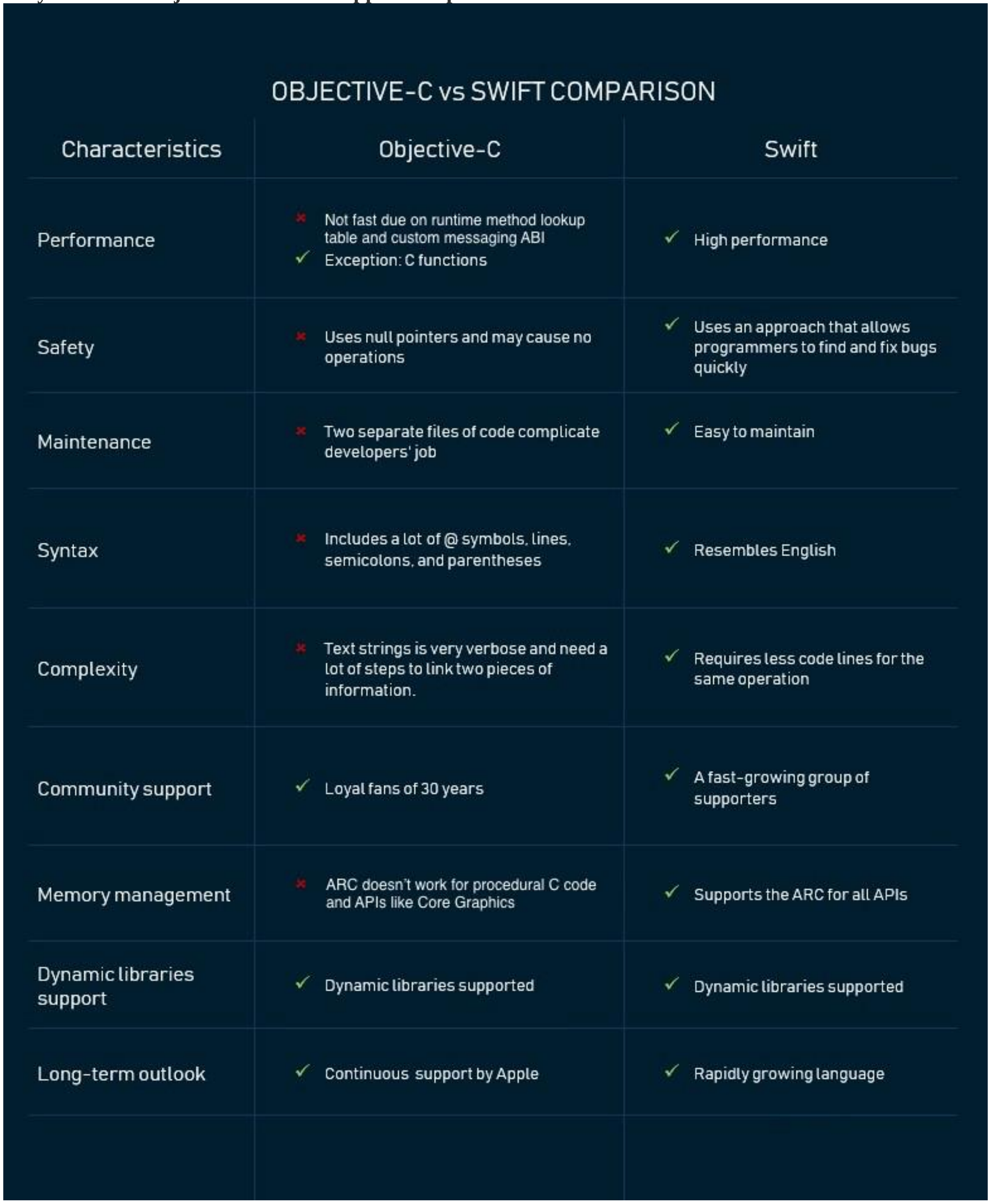
Generics:-Generics are one of the most powerful features of Swift, and much of the Swift

standard library is built with generic code. In fact, you’ve been using generics throughout the *Language Guide*, even if you didn’t realize it. For example,
Swift’s Array and Dictionary types are both generic collections. You can create an array that holds Int values, or an array that holds String values, or indeed an array for any other type that can be created in Swift.

Subscripts:- Classes, structures, and specifications can describe Subscripts, which are backup courses of action for getting to the part segments of a combination, overview, or progression. You use Subscripts to set and recuperate values by record without requiring separate strategies for setting and recuperation. For example, you get to parts in an Array case as someArray[index] and segments in a Dictionary event as someDictionary[key]

ARC:-Swift uses *Automatic Reference Counting* (ARC) to track and manage your app’s memory usage. In most cases, this means that memory management “just works” in Swift, and you do not need to think about memory management yourself. ARC automatically frees up the memory used by class instances when those instances are no longer needed.

1.3 Why swift over objective C for IOS App Development ?000JN /



OBJECTIVE-C vs SWIFT COMPARISON

Characteristics	Objective-C	Swift
Performance	✗ Not fast due on runtime method lookup table and custom messaging ABI ✓ Exception: C functions	✓ High performance
Safety	✗ Uses null pointers and may cause no operations	✓ Uses an approach that allows programmers to find and fix bugs quickly
Maintenance	✗ Two separate files of code complicate developers' job	✓ Easy to maintain
Syntax	✗ Includes a lot of @ symbols, lines, semicolons, and parentheses	✓ Resembles English
Complexity	✗ Text strings is very verbose and need a lot of steps to link two pieces of information.	✓ Requires less code lines for the same operation
Community support	✓ Loyal fans of 30 years	✓ A fast-growing group of supporters
Memory management	✗ ARC doesn't work for procedural C code and APIs like Core Graphics	✓ Supports the ARC for all APIs
Dynamic libraries support	✓ Dynamic libraries supported	✓ Dynamic libraries supported
Long-term outlook	✓ Continuous support by Apple	✓ Rapidly growing language

Illustration 6:SWIFT VS OBJECTIVEC

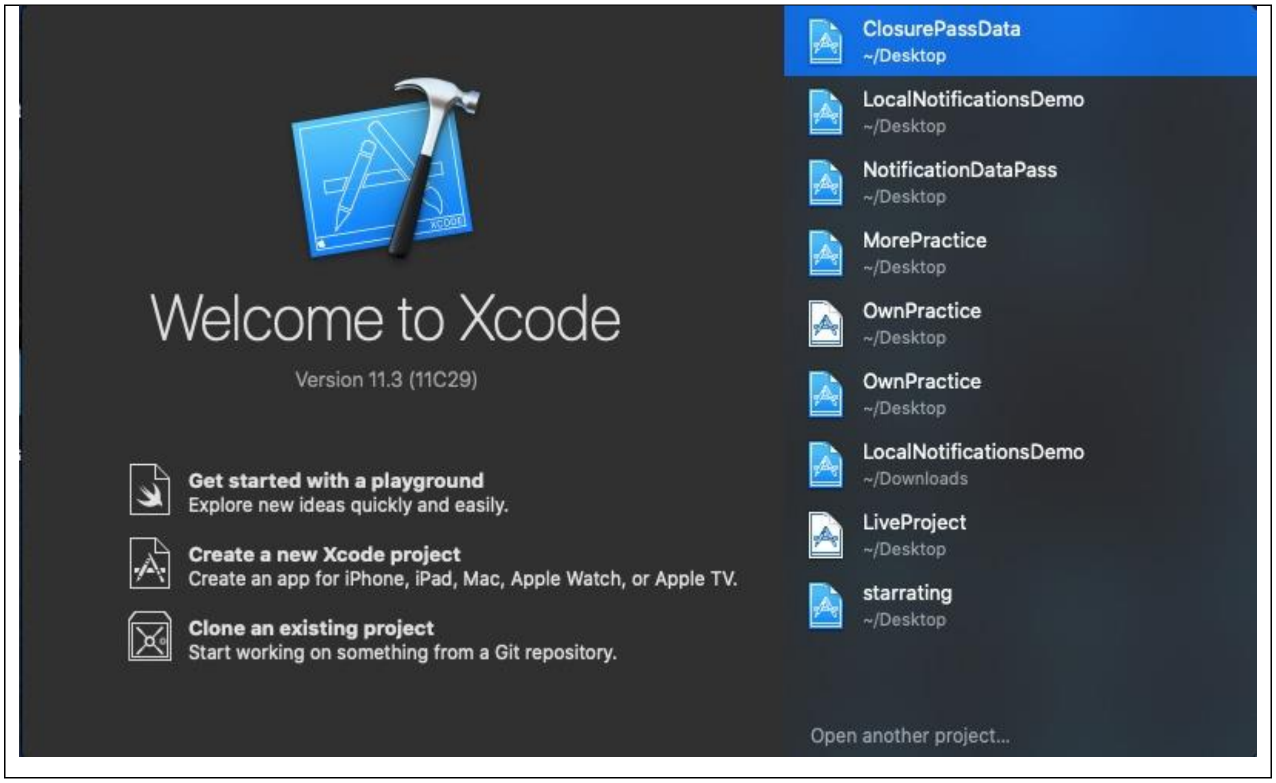
CHAPTER 2

INTRODUCTION TO XCODE

2.1 Introduction

XCode is an organized improvement condition made to manage Mac working structures. It contains a set-up of programming progression gadgets made by apple. XCode urges us to make programming for macOS, tvOS, iOS, and watchOS. The latest stable appearance of XCode is 11.3.2, which is available on the Mac App Store for all the customers of macOS Mojave. Xcode 11 is open in the Mac App Store and consolidates SDKs for iOS 13, macOS Catalina 10.15, watchOS 6, and tvOS 13. Xcode 11 sponsorships progression for contraptions running iOS 13.1. Xcode 11 sponsorships on-contraption researching for iOS 8 and later, tvOS 9 and later, and watchOS 2 and later

2.2 General Operations


Welcome to Xcode

Version 11.3 (11C29)

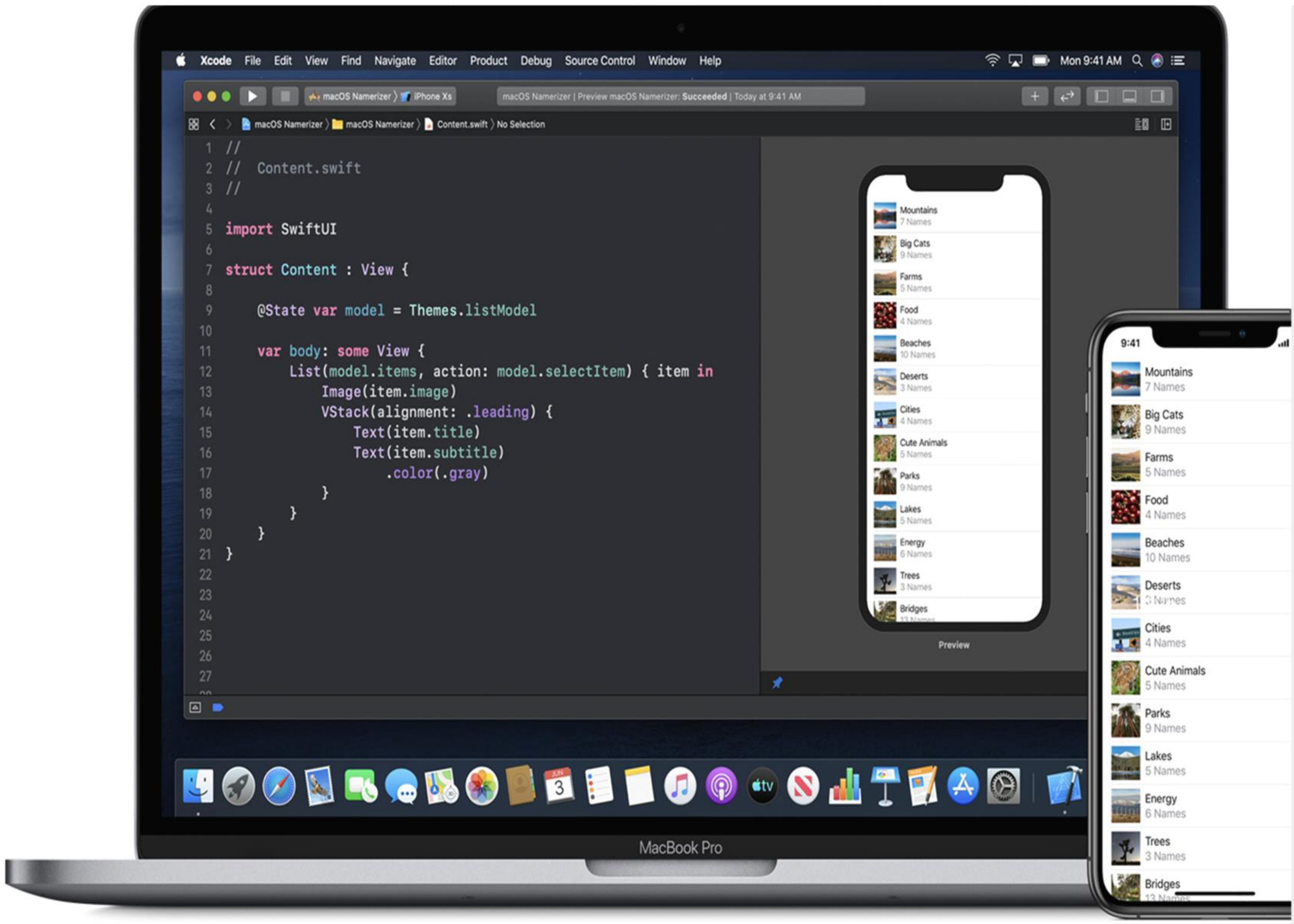
Get started with a playground
Explore new ideas quickly and easily.

Create a new Xcode project
Create an app for iPhone, iPad, Mac, Apple Watch, or Apple TV.

Clone an existing project
Start working on something from a Git repository.

ClosurePassData ~/Desktop
LocalNotificationsDemo ~/Desktop
NotificationDataPass ~/Desktop
MorePractice ~/Desktop
OwnPractice ~/Desktop
OwnPractice ~/Desktop
LocalNotificationsDemo ~/Downloads
LiveProject ~/Desktop
starrating ~/Desktop

Open another project...


Xcode File Edit View Find Navigate Editor Product Debug Source Control Window Help
Mon 9:41 AM
macOS Namerizer | Preview macOS Namerizer: Succeeded | Today at 9:41 AM
macOS Namerizer 〉 macOS Namerizer 〉 Content.swift 〉 No Selection
Mountains
Big Cats
Farms
Food
Beaches
Deserts
Cities
Cute Animals
Parks
Lakes
Energy
Trees
Bridges
Preview
MacBook Pro

```
//
//  Content.swift
//

import SwiftUI

struct Content : View {

    @State var model = Themes.listModel

    var body: some View {
        List(model.items, action: model.selectItem) { item in
            Image(item.image)
            VStack(alignment: .leading) {
                Text(item.title)
                Text(item.subtitle)
                    .color(.gray)
            }
        }
    }
}
```

Xcode 11 supports development with SwiftUI.

 Xcode 11 adds support for Mac Catalyst to bring iPad apps to the Mac.

 You can now change the appearance of Xcode independently of the system appearance setting.

 Xcode supports uploading apps from the Organizer window or from the command line with xcodebuild or xcrun altool. Application Loader is no longer included with Xcode.

 LaunchServices on macOS now regards the chose Xcode when propelling Instruments, Simulator, and other designer devices installed inside Xcode. For instance, when you double tap an Instruments follow in Finder, the rendition of Instruments for the chose Xcode dispatches. Change which Xcode is utilized with xcode-select from the order line.

 Editors can be added to any window without needing the Assistant Editor. Editors are added using the “Add Editor” button in the jump bar or the File > New > Editor command. Each editor can independently show its own assistant or canvas via commands in the Editor or Editor Options menus. Illustration 7 : XCODE

These two modes will automatically show relevant content when available. When using multiple editors, the View > Editor > Focus command can be used to temporarily expand the active editor to fill the entire window, hiding other editors. Additionally, “Show Editor Only” will reset a given editor, turning off any extra views that have been enabled in that editor. For source control support, the Code Review button in the Toolbar replaces the Comparison Editor. The “Show Authors” command is now available from the Source Editor’s Editor menu. The SCM Log is now in the Inspector Area.

2.3 CocoaPods:-

Cocoapods is a reliance supervisor used to introduce conditions for quick and target C extends in XCode. It is like the expert or frying pan, which are utilized to introduce conditions in Java. Notwithstanding, a reliance supervisor is an apparatus that deals with the arrangement of structures to make life a couple of simpler for the engineers

In iOS, Cocoapods introduces outsider libraries like Firebase, Alamofire, and so on. In any case, in the event that we don't utilize Cocoapods to introduce an outsider library like Firebase, we have to introduce it physically, which is a dull procedure. Here, we have to introduce all the conditions of firebase alongside firebase, and in the event that Firebase makes changes to its SDK, at that point we have to redownload it into our task. CocoaPods is built with Ruby and it will be installable with the default Ruby available on macOS. You can use a Ruby Version manager, however we recommend that you use the standard Ruby available on macOS unless you know what you're doing.

Using the default Ruby install will require you to use `sudo` when installing gems. (This is only an issue for the duration of the gem installation, though.)

```
$ sudo gem install cocoapods
```

2.4 What is a Podfile?

The Podfile is a determination that depicts the conditions of the objectives of at least one Xcode ventures. The document ought to just be named Podfile. All the models in the aides depend on CocoaPods adaptation 1.0 and onwards.

How to use a PodFile?

To create a new project with CocoaPods, follow these simple steps:-

- Create a new project in Xcode as you would normally.
- Open a terminal window, and `$ cd` into your project directory.
- Create a Podfile. This can be done by running `$ pod init`.
- Open your Podfile. The first line should specify the platform and version supported.platform :ios, '9.0'
- In order to use CocoaPods you need to define the Xcode target to link them to. So for example if you are writing an iOS app, it would be the name of your app.Create a target section by writing `target '$TARGET_NAME' do` and an `end` a few lines after.
- Add a CocoaPod by specifying `pod '$PODNAME'` on a single line inside your target.

Question: What is `Podfile.lock`?

This record is produced after the main run of case introduce, and tracks the variant of each Pod that was introduced. For instance, envision the accompanying reliance determined in the Podfile.

Chapter 3

Application Life Cycles & Architecture

3.1 APPDELEGATE LIFE CYCLE:-

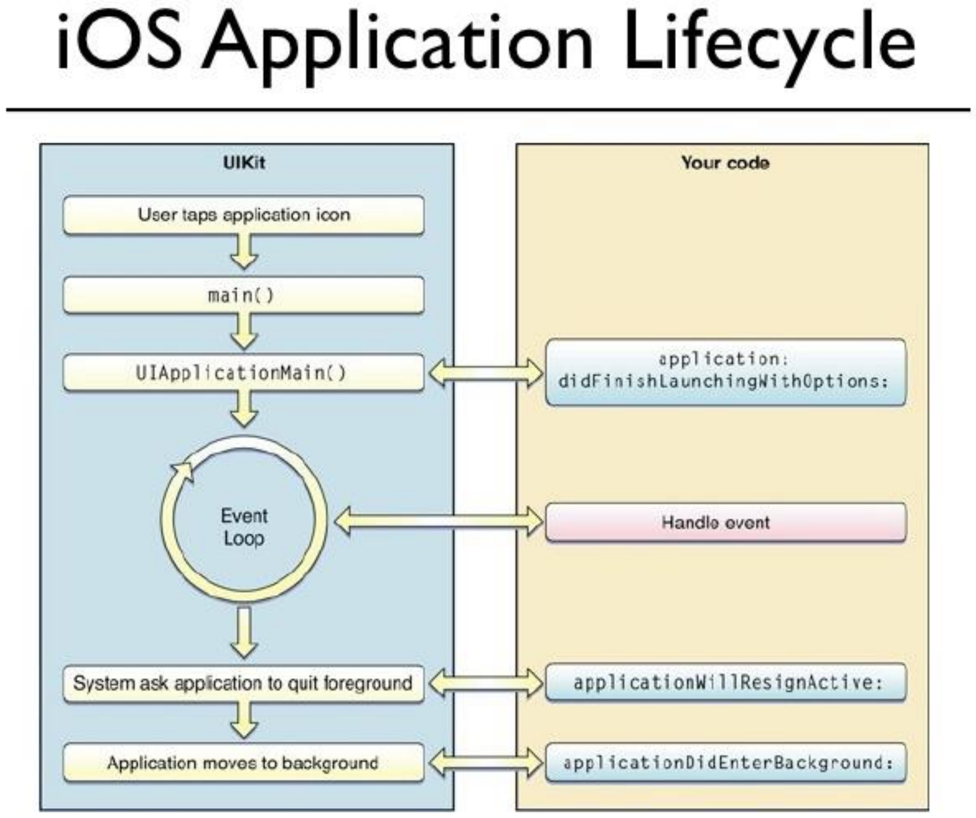

iOS Application Lifecycle
UIKit
Your code
User taps application icon
main()
UIApplicationMain()
application: didFinishLaunchingWithOptions:
Event Loop
Handle event
System ask application to quit foreground
applicationWillResignActive:
Application moves to background
applicationDidEnterBackground:

Illustration 8: APPDELEGATE LIFECYCLE

The primary concern of section into iOS applications is UIApplicationDelegate. UIApplicationDelegate is a convention that your application needs to actualize to get told about client occasions, for example, application dispatch, app goes into background or foreground, app is terminated, a push notification was opened, etc.

When an iOS app is launched the first thing called is **application: willFinishLaunchingWithOptions:-> Bool**. This method is intended for initial application setup. Storyboards have already been loaded at this point but state restoration hasn't occurred yet.

LAUNCH

- **application: didFinishLaunchingWithOptions: -> Bool** is called next. This callback method is called when the application has finished launching and restored state and can do final initialization such as creating UI.
- **applicationWillEnterForeground:** is called after **application: didFinishLaunchingWithOptions:** or if your app becomes active again after receiving a phone call or other system interruption.
- **applicationDidBecomeActive:** is called after **applicationWillEnterForeground:** to finish up the transition to the foreground.

Termination

- **applicationWillResignActive:** is called when the app is about to become inactive (for example, when the phone receives a call or the user hits the Home button).
- **applicationDidEnterBackground:** is considered when your app enters a foundation state subsequent to getting latent. You have approximately five seconds to run any assignments you have to back things up on the off chance that the app gets ended later or just after that.
- **applicationWillTerminate:** is called when your app is about to be purged from memory. Call any final cleanups here.

Both **application: willFinishLaunchingWithOptions:** and **application: didFinishLaunchingWithOptions:** can possibly be impelled with decisions recognizing that the application was called to manage a spring up message or url or something else. You need to return substantial if your application can manage the given activity or url.

Knowing your app's lifecycle is important to properly initialize and set up your app and objects. You don't have to run everything in

application: didFinishLaunchingWithOptions, which often becomes a kitchen sink of setups and initializations of some sort.

3.2 VIEWDELEGATE LIFE CYCLE:-

iOS calls the `UIViewController` methods as follows:

- **`viewDidLoad()-`**: Considered when the view controller's substance see (the highest point of its view chain of command) is made and stacked from a storyboard. The view controller's outlets are ensured to have legitimate qualities when this technique is called. Utilize this technique to play out any extra arrangement required by your view controller.
- **`viewWillAppear()-`**: Called not long before the view controller's substance see is added to the application's view chain of importance. Utilize this strategy to trigger any activities that need to happen before the substance see is introduced onscreen. Notwithstanding the name, on the grounds that the framework calls this strategy, it doesn't ensure that the substance view will get obvious.
- **`viewDidAppear()-`:** Called soon after the view controller's substance see has been added to the application's view chain of importance. Utilize this strategy to trigger any activities that need to happen when the view is introduced onscreen, for example, getting information or appearing an animation.
- **`viewWillDisappear()-`**: called not long before the view controller's substance see is expelled from the application's view chain of importance. Utilize this technique to perform cleanup errands like submitting changes or leaving the person on call status.

- **`viewDidDisappear()`**-: Called soon after the view controller's substance see has been expelled from the application's view progression. Utilize this strategy to play out extra teardown exercises. In spite of the name, the framework doesn't call this strategy on the grounds that the substance see has gotten covered up or clouded. This strategy is possibly considered when the substance see has been expelled from the application's view chain of command.

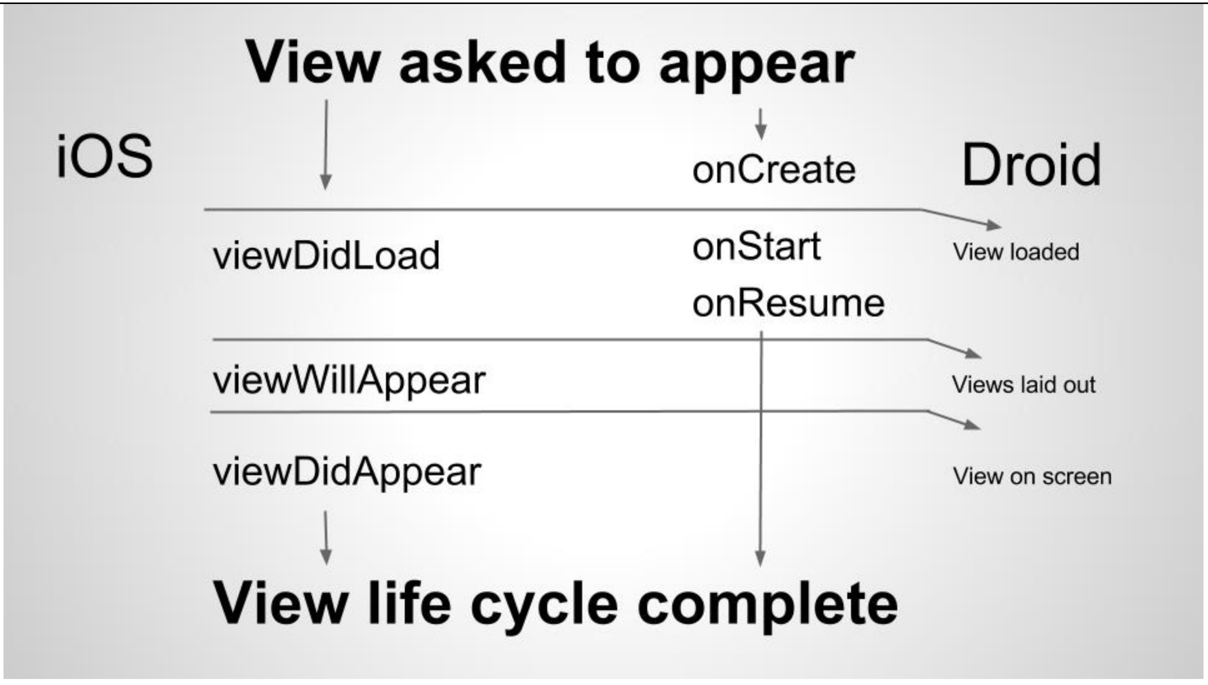


Illustration 9 : **VIEW LIFECYCLE**

3.3 SCENEDELEGATE LIFE CYCLE:-

A scene agent's most significant capacity is scene(_:willConnectTo:options:). As it were, it's generally like the job of the application(_:didFinishLaunchingWithOptions:) work on iOS 12. The capacity is considered when a scene is added to the application, so it's the ideal point to design that scene. In the above code, we're physically setting up the view controller stack, however progressively about that later.

The SceneDelegate also has these functions:

- sceneDidDisconnect(_:) is called when a scene has been disconnected from the app (Note that it can reconnect later on.)
- sceneDidBecomeActive(_:) is called when the user starts interacting with a scene, such as selecting it from the app switcher
- sceneWillResignActive(_:) is called when the user stops interacting with a scene, for example by switching to another scene
- sceneWillEnterForeground(_:) is called when a scene enters the foreground, i.e. starts or resumes from a background state
- sceneDidEnterBackground(_:) is called when a scene enters the background, i.e. the app is minimized but still present in the background

3.3 MVC ARCHITECTURE:-

The MVC — Model View Controller Architecture Design Pattern is the most well-known engineering found in iOS applications and is suggested by Apple.

MVC represents Model-View-Controller. It is an engineering or a product configuration design that makes making enormous applications simple. It doesn't have a place with explicit programming language or system, yet it is an idea that you can use in making any sort of utilization or programming in any programming language.

For example, if you are developing an application in PHP, you can use structures like Laravel or Codeigniter that uses MVC building to help you with making applications snappy and direct. It might be to some degree hard to get your head around the MVC structure from the beginning if you have made amazingly direct applications without using any kind of designing or framework. In any case, this guide is to reveal you how to work on MVC structure. At the point when you will perceive how it capacities, you will basically love managing MV

An app which adopts the MVC pattern typically has 3 main layers.

1. Model -: Models are portrayals of your application's information. For instance, objects like Users or Posts would be models of information. A User class or struct could hold data attached to a particular client like username, birthdate, and so on. Model works legitimately with the database. It doesn't need to manage UI or information handling. In certifiable situation, you will basically utilize model to get, supplement, refresh and erase information from your database.

2. View :- Views are objects that users of your app can see and interact with. View objects should be reusable and flexible. Things like UI Labels which can show any kind of text are views.

3. Controller :- Controllers mediate between the View and the Model. Controllers set up views with the data from models and update models when users interact with views.

Advantages of MVC architecture:

- Development of the application becomes fast.
- Easy for multiple developers to collaborate and work together.
- Easier to Update the application.
- Easier to Debug as we have multiple levels properly written in the application.

There is not much in the disadvantages part of the architecture. And the disadvantages are not so huge and are very easy to ignore in comparison with all the benefits we get.

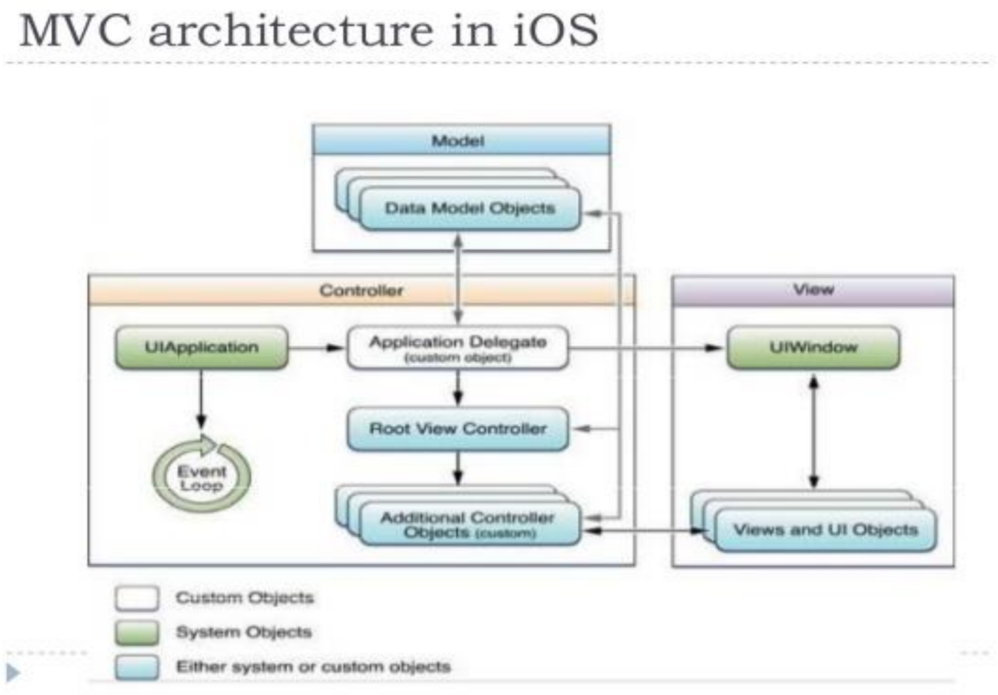


ILLUSTRATION 10: MVC ARCHITECTURE

CONCLUSION

- Learnt Swift which is a **general-purpose, multi-paradigm, compiled programming language** developed by Apple Inc. for iOS, iPadOS, macOS, watchOS, tvOS, Linux, and z/OS.
- Had an incredible involvement with finding out about the different advancements which can be utilized to make our iOS applications

considerably progressively powerful and client intuitive.

- Learnt how the backend is an essential support for an app and how various API models are formed and mapped in our apps

- Hands on understanding to investigate/troubleshoot the issues or heighten issues to the concerned groups.

- Experience to deal with industry related apparatuses and gradually picking up ability on incorporating them to tweak an inventive application.

- Hands on experience to analyze/debug the code in Xcode and VisualStudio
- Hands-on experience to analyze/debug the issues or escalate issues to concernedteams.

- Experience of working on a Real time App Suzo which has a market base and learnt about the apps's transition from initial to its deployment stage.

- Implemented the MVC Architecture for my project and designed the UI alongwith fetching data for the design using Moya.

- Hands on experience to analyze/debug the code in Xcode and VisualStudio
- Hands-on experience to analyze/debug the issues or escalate issues to concerned teams.

- Experience of taking a shot at activities, for example, the coures application and Kashkol and found out about the applications' change from starting to its sending stage.

- Learnt how to integrate facebook login and google loogin within an app as User friendly options

- Implemented the MVC Architecture for my project and designed the UI alongwith fetching data for the design using Moya.

REFERENCES

- https://www.code-brew.com/
- https://swift.org/documentation/
- https://developer.apple.com/xcode/
- https://cocoapods.org/
- https://medium.com/@hacknicity/how-to-switch-your-ios-app-and-scene-delegate
- https://developer.apple.com/documentation/objectivec
- https://developer.apple.com/library/archive/documentation/Cocoa/Conceptual/ProgrammingWithObjectiveC/Introduction/Introduction.html