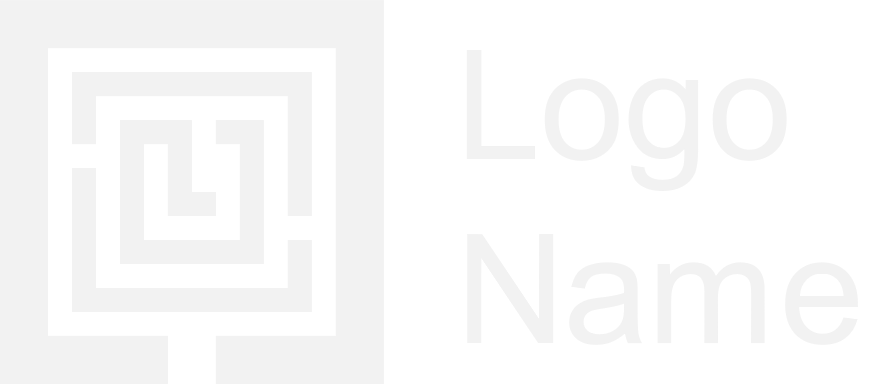


|  |  |
| --- | --- |
| KMM Transition Notes  2018 | |
|  | |
| February 20, 2021  Chorus Enterprise  Authored by: PAGE TYLER | |



# Overview

|  |
| --- |
| Making the Journey to the KMM environment When I made the decision to move my application to the Kotlin KMM environment I had know idea of what the iOS environment would bring. What I found in XCode and Swift was an environment was very similar to what I was use to in Android Studio. This document is presented as an overview of the process I used to make this transition.  The application that is referred to in this document is the KotlinDateTime example program. I used this small sample program to first address the number one issue that keep my program from being a multiple platform program and that was date processing. I also used it to introduce myself to the new environments of Swift and XCode. Please remember I am going into these new tools with know knowledge at all so if you have better suggestions on anything I present here then please add then to this post. I purposefully used the simplest Android Kotlin code in this example. I am trying to make it easily readable for someone coming the other way iOS to Android.  Here is what I have found |
|  |

# Preparing for KMM

## Before you get started here is a list of things to consider

1. Getting an iOS system,
   1. Purchasing a new or used iOS system:
      1. Based on everything I have read getting an old or used system can be very problematic. What I read was that a lot of issues can arise from old iOS and Apple hardware. These can cause development issues and slow processing.
      2. Buying a new iOS system is the best option if you can afford it. This was not an option for my budget. This is a very expensive option.
      3. Installing a VM version, I spent a week trying to get this to work. Then once getting it up and running I was faced with the inability to install anything on the system because it was not a valid installation of said system. Not being an iOS systems developer, I gave up at this point.
      4. Getting a cloud version of iOS (Recommended). This is the option that I found most helpful. It took the job of learning how to setup an iOS system out of my hands and put it in the hands of an expert. This left me with the task that I new how to do well.
   2. Getting a cloud version of iOS Options available,
      1. Flow ( [www.flow.swiss](https://flow.swiss/pricing#mac_bare_metal) ) This was way too expensive but if you can afford it this may be a faster solution for your development effort.
      2. Rent the hardware in the cloud at ( [www.macminivault.com](https://www.macminivault.com/mac-mini-mac-pro-rentals/?gclid=CjwKCAiAjp6BBhAIEiwAkO9WujSdWjC3Xu1qzxOuy1BenNeIRduiiBkMKyptGzWEApLFrI_yo7japxoCMFwQAvD_BwE) ) this is the middle of the road option. It cost more than I was willing to pay but it is a modestly priced option. It is more than likely faster and more available then the one I selected.
      3. Rent a Dedicated Mac Virtual Machine (Recommended) (<https://xcodeclub.com/> ). This is the option I selected. The price is very reasonable and the speed is acceptable. It is slow on start up but once it is up and you have your programs running the speed is quite acceptable.
         1. If you go this way and you plan to use Android Studio please ask to have your system enabled for VM.
         2. You will need to download and install Android Studio.
         3. If you are moving files in from another system you will need to place them in a subdirectory where you will need to perform a Linux [chmod -r] command on them via the terminal. You will do this remove them from read only mode and allow write access.
2. Learning how to use the iOS system
   1. Keyboard differences
      1. If you are connecting from a windows system here are a few tips;
         1. Copy and Paste are controlled by the windows key. This