Adviesrapport KMM

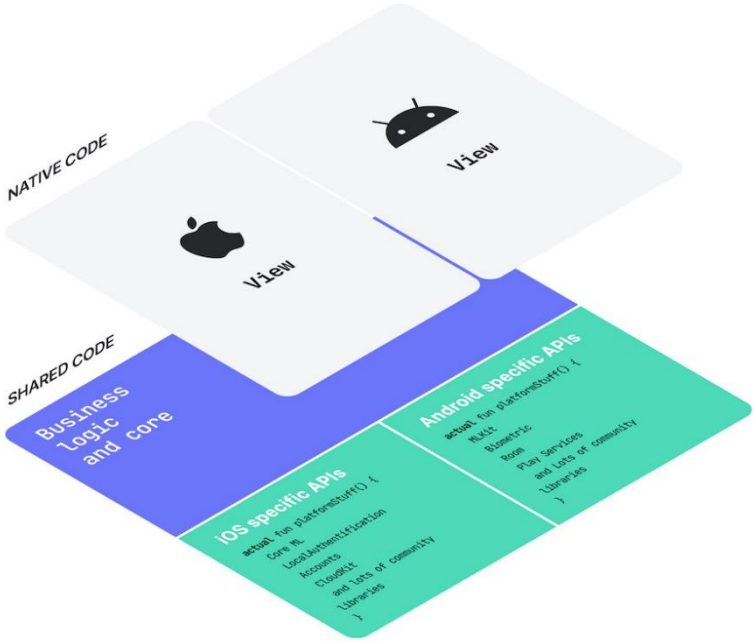
Door: Hicham el Marzgioui
klas: INF 3B
Jaar: 2020/2021

Inleiding

Dit adviesrapport is tot stand gekomen door het leerdoel adviseren 1: “Ik wil na afloop van mijn stage een aanbeveling kunnen doen van het gebruiken van Kotlin Multiplatform voor het ontwikkelen van een applicatie voor Android, iOS of het Web.” Ik heb gekozen voor dit leerdoel omdat het huidige bedrijf waar ik mijn stage loop gebruik maakt van Kotlin Multiplatform Mobile (KMM) voor het ontwikkelen van de Byewaste app.
In dit verslag zal er eerst worden uitgelegd wat KMM is. Vervolgens zal er worden gekeken naar de voor- en nadelen van het gebruiken van deze nieuwe technologie. Verder zal er een voorbeeld worden gegeven van Netflix die KMM gebruikt voor een applicatie. Ook wordt KMM vergeleken met Flutter, een andere cross-platform tool. Tenslotte is er een conclusie waar er een advies volgt voor het gebruiken van KMM en of dit toepasselijk is voor het bedrijf.

Wat is KMM?

KMM is een SDK die ontwikkelt is door JetBrains. De technologie Kotlin Multiplatform Mobile staat nog in haar kinderschoenen, de technologiekwam uit in 2018 met de release van Kotlin 1.2. KMM is nu nog in haar alfaversie en een stable-versie laat nog op zich wachten, tijdens het schrijven van dit adviesrapport.
KMM geeft de ontwikkelaar de mogelijkheid om de logica (business logic en core) eenmaal te schrijven en te gebruiken op Android, iOS en Web. Waar het nodig is aangezien KMM niet alles kan combineren, maar dit is makkelijk te verhelpen door het gebruiken van *actual* -en *extended fun*. Een voorbeeld waar ik bij Byewaste mee te maken heb gehad is het formateren van een string naar een datum, dit was niet mogelijk in de gedeelde logica (Main), dus is er een extended fun gecreëerd voor main en is er platformzijde aan de slag gegaan met de actual fun die terug communiceert met de extended fun. Verder is het handige aan KMM dat de ontwikkelaar de gebruikersinterface (native platformzijde kan maken en dit gemakkelijk kan laten verbinden met de logic.



Voor- en nadelen van KMM

De voordelen van het gebruiken van Kotlin Multiplatform Mobile:

- Snellere ontwikkeling, omdat de logica eenmaal geschreven hoeft te worden en op alle platformen te gebruiken is. Nu hoeft niet iedere platformzijde ontwikkelaar ook nog eens zijn eigen code te schrijven op het desbetreffende platform.
- Hogere code kwaliteit omdat de platformzijde ontwikkelaars van verschillende platformen samen aan een codebase kunnen werken. Dit zorgt ook voor meer creatieve oplossingen, omdat er nu verschillende ontwikkelaars met andere invalshoeken een oplossing kunnen bedenken.
- Het beste van twee werelden omdat KMM de ontwikkelaar instaat stelt de gebruikersinterface volledig native te laten doen.
- Makkelijker te onderhouden omdat de logic eenmaal geschreven is. Het is ook makkelijker te testen. Waar er eerst getest moest worden op verschillende platformen of de code wel het zelfde deed als op het andere platform.

De nadelen van het gebruiken van Kotlin Multiplatform Mobile zijn:

- Heeft een cross-functioneel team nodig. Het is voor de Android developer makkelijker om de code te schrijven voor de logica dan de iOS developer. Het kan ook zo zijn dat als het iOS-team geen Kotlin wilt leren de volledige ontwikkeling van de logica op het Android-team valt. En dan zullen veel voordelen van KMM wegvallen.
- Het is moeilijker te debuggen voor de iOS. De error logs zijn vaag en niet specifiek.
- Kotlin Multiplatform Mobile is ook een nieuwe technologie. Het is momenteel nog in de alfaversie, ook komen de updates traag en de documentatie is aan te wensen over te laten. Omdat het een nieuwe technologie is het ook moeilijker om een developer te vinden die meteen aan de slag kan gaan met de code. Er is eerst een leercurve.

Netflix en KMM

Een groot en populair bedrijf waar iedereen wel van gehoord heeft is Netflix. Netflix is een van de bedrijven die ook het Kotlin Multiplatform gebruikt voor het ontwikkelen van haar applicaties. Een van die applicaties heet Prodicle. Prodicle is een applicatie die het managen makkelijker moet maken voor het productieteam. Netflix heeft gekozen voor deze ontwikkelen van deze applicatie met KMM omdat het een robuuste applicatie nodig heeft die ook offline goed te gebruiken is, omdat het productie team niet in alle gevallen een goede internet verbinding heeft. Ook heeft Netflix gekozen voor KMM omdat de wereld van de fysieke productie zeer snel, en ook met veel problemen gepaard gaat en dus moet de app met snelle oplossingen komen om dit tegen te gaan. Hier is KMM ook zeer handig voor aangezien de logica maar eenmaal geschreven hoeft te worden en meteen aan de platformzijde gebruikt kan worden.

KMM en Flutter

Flutter is net zoals KMM een multiplatform tool om apps te ontwikkelen. Wat KMM en Flutter van elkaar onderscheidt is het volgende: KMM gebruikt de programmeertaal Kotlin en Flutter gebruikt Dart. Het handige van Flutter is dat er alleen maar met Dart gewerkt hoeft te worden om zowel voor de UI en business logic te ontwikkelen. Voor KMM is er een cross-platform team nodig dat ook Objective-c/Swift kan gebruiken voor de iOS-zijde. Maar dit heeft ook een nadeel en dat is dat je met Flutter ook veel grotere app bestanden zult krijgen. KMM is dus veel efficiënter dan Flutter.
De programmeertaal dart is ook vrij nieuw. De programmeertaal is ook veel minder intuïtief dan Kotlin. Kotlin is een programmeertaal dat veel op Java lijkt. Dat maakt de stap van Java naar Kotlin ook makkelijker dan naar Dart. Om de UI te maken met Flutter moet er gebruikt worden gemaakt van de Flutter libraries. Deze libraries groeien erg snel en er komen telkens meer functionaliteiten bij maar toch is er nog veel aan te wensen over te laten. Veel functies van de native llibrary van Android of iOS zijn er simpelweg nog niet voor Flutter en als die er zijn ze vaak van mindere kwaliteit dan van de native. Dit probleem heeft Kotlin niet aangezien het alleen 1 aspect van dit proces wilt overnemen en dat is de business logic.

Conclusie

Kotlin Multiplatform Mobile heeft veel voordelen, zoals snelle ontwikkeling, verbeterde code kwaliteit en nog veel meer. Maar er zijn ook nadelen aan deze nieuwe technologie. Om gebruik te maken van de technologie is er een cross-platform team nodig, waar de iOS-developers open moet staan voor het leren van een nieuwe programmeertaal. Ook is het debuggen moeilijker voor de iOS-developers. Ik heb in dit adviesrapport ook een vergelijking getrokken met KMM en Flutter. Waar ik de verschillen van beide uitleg. Waar KMM alleen 1 specifiek gedeelte van het ontwikkelproces op zich neemt wilt Flutter dit als geheel doen. En dit heeft als gevolg dat de app overbodig groot en minder efficiënt is.
Mijn aanbeveling voor het gebruiken van KMM is een “ja”. Ik raad de technologie zeker aan om te gebruiken voor het creëren van een nieuwe mobiele applicatie. Waarom ik gekozen heb voor deze aanbeveling is omdat KMM, de ontwikkelaar het beste biedt van beide werelden. De logica wordt eenmaal geschreven en kan dan worden verbonden met de platformzijde code. Hierbij heeft de ontwikkelaar het gemak dat de logica eenmaal geschreven hoeft te worden en de logica te implementeren aan de native kant. Dit zal het ontwikkelproces drastisch inkorten. Verder raad ik het gebruik van KMM ook aan vanwege de verbeterde code kwaliteit, omdat er meerdere koppen aan een oplossing zullen denken voor een probleem en mekaars code kunnen bezichtigen voor verbeteringen. Ik snap ook wel dat veel van de nadelen bij het iOS-team liggen voor het werken met KMM. Maar met de toekomstige updates en de leercurve van Kotlin zullen deze problemen in het niets vallen met de waarde die KMM met zich meebrengt.

Bronnen

https://archer-soft.com/blog/how-kotlin-multiplatform-helps-reduce-app-development-time#challenges%20of%20multiple-team%20development

https://freshworks.io/kotlin-multiplatform-pros-and-cons-for-app-development/

https://netflixtechblog.com/netflix-android-and-ios-studio-apps-kotlin-multiplatform-d6d4d8d25d23

https://instabug.com/blog/flutter-vs-kotlin-mutliplatform-the-2021-guide/