Bachelorproef draft (14/05) (Officieel 28/05)

Inhoudsopgave

Samenvatting

Inleiding

De impact van mobiele applicaties (vanaf nu apps) op ons alledaags leven is niet te onderschatten. Ze zijn de dag van vandaag niet meer weg te denken. Dit komt door de grootte toename in de verkoop van smartphones. Ongeveer 40% van de wereldbevolking beschikt over een smartphone en in 2020 gingen er 1.38 miljard nieuwe toestellen de deuren uit. De toename in verkochte smartphones leidt tot een grotere markt voor apps. Dit hebben softwarebedrijven niet over het hoofd gezien. Het aantal applicaties aanwezig op de iOS App Store en de Google Play Store, is in de voorbije tien jaar respectievelijk vertwintig- en verdertigvoudigd. Elke smartphone maakt gebruik van een besturingssysteem, dit is de software die de hardware aanstuurt. Tegenwoordig is er een oligopolie van twee spelers op de markt van smartphone besturingssystemen (vanaf nu OS voor Operating System). Apple ontwikkelde hun eigen OS genaamd iOS terwijl de meeste andere merken zoals Samsung en OnePlus gebruik maken van het Android OS. Elk OS heeft een verschillende manier van applicatieontwikkeling. In de meeste gevallen zullen app eigenaars met hun applicatie een zo breed mogelijk publiek willen bereiken. Hierdoor zullen ze apps moeten ontwikkelen voor zowel iOS en Android.

Wat is Native Development?

Als het ontwikkelingsteam kiest om een applicatie native te gaan ontwikkelen dan wil dit zeggen dat een aparte codebase (verklaren) moet ontwikkeld worden voor elk mobiel besturingssysteem waar de app moet op kunnen draaien. Verschillende codebases wil zeggen meer ontwikkelings- en onderhoudswerk maar geeft anderzijds wel een optie tot een native- design en gedrag per codebase.

Wat is Cross-platform Development?

Een andere aanpak is het ontwikkelen van een cross-platform app. Dit is het schrijven van één codebase die kan gecompileerd (vertaald) worden naar verschillende besturingssystemen. Het is een optie die de laatste jaren meer en meer gekozen wordt. De voor- en nadelen die hiervoor zijn aangehaald, draaien zich om. Een enkele codebase is goedkoper om te bouwen en te onderhouden maar dit moet dan ook afgewogen worden tegen het laten van native- designs en gedrag. Er zijn verschillende ontwikkeltools gemaakt voor het maken van cross-platform apps. Een paar van de grootste zijn React Native, Xamarin, Flutter en Ionic.

Wat is Flutter

Flutter is een Google UI toolkit voor het bouwen van mooie, bijna natively gecompileerde apps voor mobiel, web, en desktop en dit alles van één enkele codebase. Flutter kondigde op 3 maart 2021 hun nieuwe update aan die hen weer een stap in de goede richting duwde. Dit is onder andere te danken aan de grootte steun en de snelle tractie van het platform in de voorbije jaren. Flutter is opensource wat wil zeggen dat ze hun broncode openstellen voor het publiek, waardoor iedereen het kan bekijken. Zo kunnen verbeteringen worden gedaan door eindgebruikers die het beste voor hebben met het platform. Als Flutter deze verbeteringen goedkeurt worden deze opgenomen in de broncode en beschikbaar gesteld voor iedereen.

Waarom Flutter versus native Android?

Flutter en Android zijn beiden ontwikkeld door Google. Alhoewel Flutter relatief jong is, kan dit niet gezegd worden van native Android. Het platform kende al veel updates en staat al lang niet meer in zijn jonge schoentjes. Vandaar dat het toetsen van deze twee een interessante kijk kan geven op het jonge flutter platform. De Flutter applicatie zal geschreven worden voor- en alleen gecompileerd worden naar Android. De sterkte van Cross-platform development is natuurlijk één codebase voor verschillende besturingssystemen, maar voor dit onderzoek is alleen Android van belang. Het onderzoek zal rekening houden met het geschreven aantal lijnen code per uitgewerkt hoofdstuk. Als het onderzoek rekening moet houden met iOS brengt dat het evenwicht uit balans.

Waarom Kotlin Android en geen Java Android?
Deze keuze is voor velen persoonlijk alhoewel hier ook onderzoek naar verricht is. In deze studie wordt Kotlin gebruikt omdat de syntax van de taal zal helpen bij het intomen van het aantal lijnen code. Hoe minder lijnen code, hoe eenvoudiger en overzichtelijker de app. Het is voor velen dan ook de voorkeurstaal om Android-applicaties in te ontwikkelen. Dit onderzoek zal geen stap voor stap uitleg inhouden van hoe de code in elkaar zit. Het doel is een vergelijking schetsen tussen Flutter en Android. De code die wordt geschreven zal dus alleen worden aangehaald indien dit relevant is voor de studie.

Probleemstelling
Het te onderzoeken domein kwam vanuit Next Apps. Een native app ontwikkelingsbedrijf uit Lokeren dat zich specialiseert in Android en iOS watch, phone en tablet apps. Zij merken dat niet alle klanten op de hoogte zijn van het hoge kostenplaatje van een native ontwikkelde app. Om in de toekomst een middenweg te vinden en de kost van een kleine app te dempen, denken zij erover om sommige apps cross-platform te gaan ontwikkelen. Bij het onderzoeken van de cross-platform markt kwamen ze Flutter tegen, het nieuwe platform met veel potentieel. Daarom werd de vraag gesteld om de stand van het Flutter platform te onderzoeken. Met deze scriptie hopen ze een duidelijk beeld te krijgen op de limieten en mogelijkheden van het framework.

Onderzoeksvraag
Hoofdonderzoeksvraag
Wat zijn de voor- en nadelen van app ontwikkeling in Flutter in vergelijking met native Android?
Het onderzoek zal zich vooral richten op het vinden van een antwoord op deze hoofdvraag. Aan de hand van deze kan een duidelijk beeld worden geschetst van de huidige stand van Flutter in vergelijking met Android. Hierbij wordt gelet op de ontwikkelingstijd van de features, de complexiteit van de code, het aantal lijnen geschreven lijnen code…
Hypothese: de voorkeur van ontwikkelingsplatform zal hoogstwaarschijnlijk afhangen van de soort app die ontwikkeld moet worden. Een grootere app met veel features zal eerder native ontwikkeld worden, terwijl kleinere apps die weinig native behaviour kunnen bevatten waarschijnlijk cross-platform ontwikkeld zullen worden.

Deelonderzoeksvragen
Is Flutter al matuur genoeg om te aanschouwen als volwaardig alternatief op native app ontwikkeling?
Deze vraag biedt een antwoord aan alle native app ontwikkelaars die denken om de overstap te maken naar cross-platform development. Het is belangrijk om een overzicht te krijgen in de limieten van een platform alvorens er tijd en geld in te investeren.
Hypothese: Flutter zal geen duidelijke standaard hebben. Best practices, goed ondersteunde libraries en coding principles zijn nog niet gedefinieerd door het platform waardoor het voor beginners onduidelijk zal lijken wat de beste manier van aanpak zal zijn. Ondanks de snelle groei en de grootte steun van het platform wordt wel verwacht dat hier snel verandering in gebracht wordt.

Is Flutter toegankelijk voor nieuwe ontwikkelaars?
Het proces van het ontwikkelen van de Flutter applicatie zal antwoord bieden op deze vraag. Voor velen is het moeilijk om te schakelen naar een nieuw platform met een nieuwe taal. Om deze overgang transparant te maken, wordt een beeld geschetst van de moeilijkheidsgraad van Flutter en Dart. Hierdoor kan de lezer zelf beslissen of hij deze keuze wil maken.
Hypothese: voor ontwikkelaars met een Java of C++ achtergrond zou de Dart taal geen probleem mogen vormen. Ze hebben een gelijkaardige syntax. Verwachtingen van het Flutter platform liggen hoog. De snelle groei van de gebruikersbasis doet vermoeden dat het een toegankelijk platform is.

Onderzoeksdoelstelling

Zoals hiervoor vermeld zal dit onderzoek de voor- en nadelen van het Flutter framework proberen vinden. Om dit op een duidelijke manier te doen, zal het framework getoetst worden aan native Android. Door het jonge platform af te wegen tegen het ontwikkelde native tegengestelde, wordt gehoopt op een duidelijk beeld van Flutter. Het zal interessant zijn om te weten of het platform te kort doet aan native Android, want deze dure tegenhanger moet zo goed mogelijk zijn troeven uitspelen om niet ten onder te gaan aan het goedkope alternatief. Het onderzoek wordt gevoerd aan de hand van een vergelijkende studie.

Voor het onderzoek worden twee identieke applicaties ontwikkeld. Beide applicaties zullen ontwikkeld worden in Android studio maar de ene zal geschreven worden in Kotlin terwijl de andere geschreven wordt in Dart. Door het ontwikkelproces van de Flutter app af te toetsen tegen dat van de native Android app zal hopelijk een beeld worden geschetst van de stand van het Flutter framework. Met dit beeld zal dan een antwoord gevormd worden op de onderzoeksvraag.

Dit onderzoek gaat gepaard met een deadline, deze tijdsdruk zorgt ervoor dat een aantal criteria moeten gesteld worden alvorens te starten. Deze criteria zullen zorgen voor een kwaliteitsvol eindresultaat dat haalbaar is in het opgelegd tijdsbestek. Het eerste aantal criterium gaat over de performantie van de twee apps. Dit criterium wordt verder onderverdeeld in drie verschillende criteria. Allereerst wordt gekeken naar de opstartsnelheid van de apps. Vervolgens wordt de grootte van de uitvoeringsbestanden bekeken. Als laatste wordt gekeken naar het CPU gebruik van beide apps. De volgende criteria is creatie van views, hier wordt gekeken hoe een layout zich vertaalt naar een view. Hoe complexe views gemaakt worden en wat de mogelijkheden en grenzen zijn van cross-platform layouts.

Het beoogde resultaat van de studie is een antwoord bieden op alle onderzoeksvragen. Indien dit lukt, kan de studie als succesvol worden beschouwd, anders zal een suggestie worden gedaan voor toekomstig bijkomend onderzoek.

Opzet van deze bachelorproef

Deze scriptie kan opgedeeld worden in verschillende hoofdstukken, elk hoofdstuk kan bestaan uit verschillende secties of ondertitels. Deze sectie beschrijft de inhoud van elk van deze hoofdstukken.

Hoofdstuk 1 Inleiding: schetst een kort beeld van de inhoud van deze studie. Biedt een antwoord op sommige vragen rond deze paper.

Hoofdstuk 2 Stand van zaken: bevat een literatuurstudie. Een samenvatting van relevante onderzoeken die reeds voorafgingen aan dit onderzoek.

Hoofdstuk 3 Methodologie: omschrijft hoe het onderzoek in zijn werk zal gaan. De opzetting van het onderzoek en het onderzoek zelf zullen duidelijk toegelicht worden.

Hoofdstuk 4 Grootte van de uitvoeringsbestanden en opstartsnelheid van de app: kijkt naar twee performantie criteria van de app. Grootte van de uitvoeringsbestanden zal de omvang van twee identieke app APK's vergelijken.

Hoofdstuk 5 CPU gebruik van de app:

Hoofdstuk 6 Gebruik van online API's:

Hoofdstuk 7 Security:

Hoofdstuk 8 Creatie van views:

Hoofdstuk 9 Beschikbare libraries en code complexiteit:

Hoofdstuk 10 Asynchroon werken:

Hoofdstuk 11 Beschikbare tools:

Hoofdstuk 12 Appendix: zal verdere uitleg bieden rond bepaalde onderwerpen voor de geïnteresseerde lezer.

Hoofdstuk 13 Conclusie: dit is de algemene conclusie op deze scriptie. In deze conclusie zal een antwoordt gegeven worden op de reeds aangehaalde onderzoeksvragen. Daarbij wordt ook aangezet tot toekomstig onderzoek binnen dit vakgebied.

Stand van zaken

Inleiding

- Inhoud hoofdstuk
- Wat is een literatuurstudie
- Hoe bouwen we hierop voort
- Wat is de impact van de literatuurstudie op deze BAP

Zoek onderzoekartikels op.
Selecteer relevante onderzoeksartikels.
Breng de bevindingen uit de onderzoeksartikels in kaart.
Vat de bevindingen uit de onderzoeksartikels samen.

Dit hoofdstuk is equivalent aan een literatuurstudie. Eerst werd

Vervolgens wordt een opsomming gemaakt en een kort beeld geschetst van een aantal van deze studies en hun bijdrage in het vakgebied. In de volgende sectie zullen alleen de meest relevante studies, die dicht aansluiten bij dit onderzoek, worden aangehaald. De ondervindingen uit deze studies werden gebruikt als voorbereiding op het onderzoek en kunnen in volgende hoofdstukken gebruikt worden als steunpunten. Door de ondervindingen van verschillende studies te bundelen, kan gezocht worden naar een rode draad.

De te onderzoeken criteria van deze studie werden vastgelegd op basis van de reeds voorgaande onderzoeken. Het is de bedoeling dat deze studie bijdraagt aan het vakgebied. Hierdoor wordt weerhouden van reeds onderzochte criteria opnieuw te gaan onderzoeken. Toch zal blijken dat sommige reeds onderzochte criteria opnieuw worden bekeken. Deze herhaling komt voort uit de Flutter 2.0 release. Deze zorgde voor veranderingen in de prestaties van het framework, waardoor bevindingen van voorgaande studies verouderd zijn.

Voorgaande onderzoeken

Navaron Bracke – Android Native Development in Kotlin versus het Flutter Framework, een vergelijkende studie (2020)

Navaron Bracke, onderzocht in 2020 de verschillen en gelijkenissen tussen Android en Flutter. De scriptie vormde een beeld over de toenmalige stand van het Flutter framework door het te toetsen tegen het ontwikkelde Android. Het onderzoek was afgebakend door een vooraf vastgelegd aantal criterium. Zo kon de onderzoeker een zekerheid bieden over de grondigheid van de studie. Het onderzochte criterium waren: Internationalisering, navigatie, persisteren van gegevens, user Interfaces, asynchroon werk, permissies, software testing, opstarttijd van de applicatie, grootte van de applicatie. Uit deze studie bleek dat Flutter een gemakkelijkere implementatie biedt voor de meeste onderzochte criteria.

Austen Latture – Backdrop- an exploration of Flutter (2020)

Latture, een student aan Grand Valley State Universiteit, schreef in 2020 een studie over het ontwikkelen van een Flutter app. Hij stelde vast dat het een zeer toegankelijk framework is met een taal die voor iedereen met een achtergrond in C gemakkelijk op te pakken is. Hier waren geen vooropgestelde criteria aanwezig, deze studie omschreef de uitwerking van de Backdrop app. In zijn conclusie vergeleek hij Flutter met React Native en prees hierbij de werking van Flutter en het grote aantal voorziene functionaliteit. Echter concludeerde hij wel dat het kiezen van bepaalde libraries niet altijd gemakkelijk is. De jonge aard van het platform zorgt ervoor dat nog geen duidelijke standaarden zijn in termen van libraries.

Dang & Skelton – Teaching mobile app development: choosing the best development tools in practical labs (2019)

Daniel Dang en David Skelton van de Insitutue of Technology, schreven in 2019 een wetenschappelijk artikel over mobile app development in een educatieve context. Ze onderzochten verschillende native- en cross-platform opties om toe te voegen aan een curriculum. Onderzochte criteria bij dit onderzoek waren: Design App User Interface, User Event Handling, Services & Multiple threads used in Internet connection, Graphics and Animation, Device hardware and sensor, Wireless connectivity, Internal data storage, Real time database. Zij concludeerden dat Flutter en Native Android hiervoor de beste opties waren. Belangrijk hierbij te vermelden is dat de studie gebruik maakte van een verouderde Flutter release.

Carlos Chavez & Yoonsik Cheon – Creating Flutter Apps from Native Android Apps (2020)

Carlos Chavez, momenteel een student en Yoonsik Cheon, een professor aan de Universiteit van Texas in El Paso onderzochten in 2020 het herschrijven van Native Android Apps naar Flutter apps. Zij stelden de vraag of het best practice is om Native Android apps te herschrijven in Flutter. In de conclusie van de studie schreven ze dat rekening moet gehouden worden met de leeftijd van het platform maar ook de tools en gebruikersbasis. Zij concludeerden een tekort aan officiële richtlijnen, libraries en interfaces binnen het platform. Het grote aantal derde partij libraries zorgt ook voor onzekerheid bij het zoeken naar een goed ondersteunde library.

➔ Grote lijnen zijn niet duidelijk volgens studie

Matilda Olsson – A Comparison of Looks Between Flutter and Native Applications (2020)

Mathilda Olsson, een oud student aan de Blekinge hogeschool onderzocht in 2020 hoe flutter applicaties in vergelijking staan met native applicaties. In haar conclusie schreef ze: geen tot weinig verschil in CPU gebruik tussen native en flutter apps. Vervolgens haalde ze de codecomplexiteit en omvang aan. De Flutter app uit het onderzoek bleek 1/5 te zijn van de code nodig om 2 native apps te schrijven. Dit verschil is substantieel en duidt nogmaals op het grote voordeel van cross-platform development. Ze concludeerde dat Flutter veel potentieel heeft indien de steun van de community zo groot blijft.

Erik Blokland – An Empirical Evaluation of the User Interface Energy Consumption of React Native and Flutter (2019)

Erik Blokland, in 2019, een student aan de University of Technology in Delft, schreef een vergelijkende Thesis over de energieconsumptie van UI tussen React Native, Flutter en de Android. Hierbij werd Android als standaard genomen en werden de twee andere apps hiermee vergeleken. Drie vragen werden gesteld:

- Bestaat een verschil in energie verbruik tussen de drie apps
- Hoeveel energie verbruiken specifieke UI acties
- Is er een verschil tussen deze UI acties bij elke app

In conclusie bleken de verschillen tussen UI acties over de platformen heen zeer stabiel te zijn. De grote verschillen die gevonden werden waren in de verschillende uitgevoerde acties. Het openen van bijvoorbeeld een drawer menu was een veel energie zwaardere taak als het openen van een modal. Als laatste was het niet duidelijk uit onderzoek of er een verschil was tussen de verschillende apps in energieconsumptie. Deze testen waren te inconsistent om conclusies uit te trekken.

Jakhongir Fayzullaev – Native-like cross-platform mobile development (2018)

Jakhongir Fayzullaev, schreef in 2018 een Bachelor studie in naam van het Finse Xamk, Universiteit Toegepaste Wetenschappen. Het is een vergelijkende studie tussen drie verschillende cross-platform development frameworks. Kotlin Native, Multi-OS en Flutter werden vergeleken. Drie frameworks die meest bruikbaar zijn voor Android ontwikkelaars omdat ze dicht aansluiten bij de Java en Android ontwikkelingsmanieren. Dit waren in 2018, tijdens het onderzoek, nieuwe platformen met als gevolg geen tot weinig voorgaande studies waarop kon gesteund worden. Wat op zijn beurt leidde tot een oppervlakkige studie die de grote lijnen van elk platform vergeleek. Het Flutter deel van het onderzoek bekeek in grote lijnen: Widgets, het maken van layouts en de bouwstenen van flutter Hij concludeerde dat op vlak van performantie de drie platformen zo goed als identiek waren. Flutter was volgens hem een goed platform in 2018 maar door de jonge aard waren er niet veel bruikbare libraries en tools beschikbaar. Er was op dat moment niet veel built-in functionaliteit. Met als gevolg extra werk voor de programmeurs die meer code moesten schrijven ten opzichte van de andere meer ontwikkelde platformen. Fayzullaev omschrijft deze manier van coderen als ‘het opnieuw uitvinden van het wiel’.

Lukas Dagne - Flutter for cross-platform App and SDK development (2019)

Lukas Dagne, schreef in 2019 zijn Bachelorproef over Flutter in vergelijking met Native- en andere Cross-platform ontwikkelings platformen. De scriptie baseerde zich op een enkele grootte punten binnen App development: de interne specificaties van het framework en de mogelijke architectuur van een app. De conclusie stelde dat Flutter een goede keuze is voor cross-platform development. De weinige toegevingen die gedaan worden bij het ontwikkelen van een cross platform app in Flutter zorgt voor een duidelijk verschil met de andere frameworks. De grootte aanhang van het platform en de snelle ontwikkelingen duiden volgens Dagne alleen maar op een verbetering naar de toekomst toe. Alhoewel dat de SDK ontwikkeling van het framework nog niet optimaal is het geloof in verbetering groot.

Ly Hong Hoang - State management in Flutter (2019)

Ly Hong Hoang, in 2019, een student aan de Metropolia Universiteit Toegepaste Wetenschappen schreef een Bachelorproef studie over state management in Flutter. Hierbij werd gekeken welk state-management systeem het best past bij Flutter. De drie vergeleken systemen waren Business Logic Component (BloC), Redux en Scoped Model. Dit is geen wetenschappelijke studie, de resultaten van de paper zijn geen feiten maar eerder een discussiepunt met een conclusie getrokken op basis van een klein onderzoek. De conclusie stelde dat BloC het best was voor gebruik in Flutter. BloC is volledig op punt gesteld voor Dart, de taal die wordt geschreven in Flutter. Ergens is dit ook wel de logische keuze aangezien het ontwikkeld is door Google. Redux is hoog performant maar complex. Dit is bijvoorbeeld niet goed om te gebruiken in teamverband. Scoped Model is gemakkelijker om te begrijpen maar minder performant als Redux. Ook is het niet gemakkelijk schaalbaar. BloC Is schaalbaar, performant en simpel te verstaan. Het heeft dus alle goede kwaliteiten. Toen deze paper geschreven werd waren er nog niet veel best practices binnen Flutter. Daarom wou de schrijver met deze studie een soort van benchmark zetten voor het kiezen van een state managementsysteem binnen Flutter.

Ola Dahl – Exploring End User's Perception of Flutter Mobile Apps (2019)

Ola Dahl schreef in 2019 een master thesis in kader van de Zweedse Universiteit Malmö. De thesis ging gebruikers perceptie tussen twee identieke apps vergelijken. De eerste app werd geschreven in Native Android en de tweede in Flutter. Er werd dan een vergelijking opgesteld tussen de twee apps, gebruikers konden aan de hand van een aantal vragen aangeven welke app ze verkozen. Het onderzoek werd gevoerd op een kleine groep gebruikers dus conclusies zijn niet representatief. Deze studie concludeerde dat gebruikers de native applicatie verkozen boven de Flutter applicatie op vlak van snelheid. Echter op vlak van uitstraling en design ondervonden de gebruikers geen verschillen. De meerderheid verkoos de Android applicatie, slechts 10% verkoos de Flutter applicatie.

Sebastian Faust - Using Google's Flutter Framework for the development of a large-scale reference application (2020)

Sebastian Faust schreef in 2020 Bachelorproef Thesis in naam van de Universiteit Toegepaste Wetenschappen in Keulen. Het doel van dit onderzoek is andere applicatieontwikkelaars helpen wanneer zij voor dezelfde problemen komen te staan tijdens het schrijven van een Flutter app. Voor dit onderzoek schreef hij een app, tijdens dit proces documenteerde hij alle design keuzes en de problemen/obstakels die hieruit voortkwamen. Na het evalueren van elk obstakel kon hij altijd tot een optimale oplossing komen. Na de studie werden ondervindingen gebundeld en een set van richtlijen opgesteld voor het ontwikkelen van grootte Flutter applicaties. Richtlijnen en best practices waar developers zich best aan houden tijdens het ontwikkelingsproces. Deze werden gebundeld in een gids die later werd gepubliceerd en goed ontvangen werd door de Flutter community. Voor verdere ondersteuning is nog een interview afgenomen met een Flutter expert. Dit onderzoek werd toegevoegd aan de literatuurstudie om aan te tonen dat Flutter apps schaalbaar zijn.

Flutter Engage - Flutter 2.0

Op woensdag 3 maart 2021 werd Flutter Engage georganiseerd, een live event gebracht door het Flutter team. Hier kwamen verschillende grootte namen binnen de Flutter community aan bod om de nieuwe releases aan de wereld te tonen. De developers gaven het de naam Flutter 2.0 door de talrijke upgrades die het platform is ondergaan. Niet alleen het Flutter platform kreeg een upgrade maar ook de taal Dart werd verbeterd voor een vlottere en vertrouwelijke programmeer ervaring.

Grootste Flutter upgrades

- Flutter is van een mobiel framework naar een portable framework gegaan. Oorspronkelijk werd Flutter alleen gecompileerd voor iOS en Android besturingssystemen maar door de upgrade zijn daar: Windows, macOS, Linux en Web bijgekomen. Dit wordt gezien als gratis upgrade aangezien het geen oude code breekt maar indien gewenst door een beperkt aantal lijnen code te herschrijven kan men naar 3 keer zoveel systemen compileren.
    - Canvas kit
    - SEO is nog niet goed ondersteund maar staat op het plan voor de toekomst
- Vouwbare smartphones werden ondersteund.
- Web is beschikbaar als stabiele release.
- Desktop ondersteuning (macOS, Windows, Linux) is beschikbaar als stabiele release (under early release flag)
    - De installatie van Ubuntu (Linux) wordt herschreven in Flutter
- Upgrade Firebase plugin -> Flutter Fire
- Google Mobile Ads SDK (open beta) (plugin)
- Nieuwe iOS features zijn aangekondigd
- Nieuwe widgets teogevoegd aan het framework.

Grootste Dart upgrades:

- Een van de grootste punten was dat sound null safety is toegevoegd, zonder oude code te breken. Dit is een grootte upgrade voor het schrijven van robuuste code. Dart zal zeggen waar er mogelijkse gevaren zitten op null excepties. Dit zorgt voor minder errors en dus betere code.
- Smarter flow analysis
- Late variables
- Required named parameters
- DevTools upgrade

Ook werden in de keynote verschillende grootte bedrijven zoals Ubuntu, Microsoft, Toyota… genoemd die in de toekomst nauw gaan samenwerken met Flutter. Deze samenwerkingen tonen de kracht, potentieel en groei van het platform.

Ook werd in de Keynote het aantal open issues op Github aangehaald. Doordat het een opensource framework is kan iedereen die de broncode wil zien, deze gaan opzoeken op GitHub. Moest een fout stuk code, een simpeler te schrijven stuk code of zelfs als een goede bijdrage aan het platform gevonden worden kan hier een issue van gemaakt worden. Als Flutter deze verbetering heeft nagekeken en goedgekeurd wordt deze toegevoegd aan de broncode. Op deze manier sparen zichzelf een hoop werk uit en groeit het platform snel en volgens de wensen van de gebruikers. Het grootte aantal issues die momenteel open staan zien zij dan ook als een teken van groei. De vele gebruikers die die Flutter willen beter maken is een teken dat er veel vertrouwen is in het platform.

Er zal wel rekening moeten gehouden worden met de upgrade bij het runnen van oudere applicaties, door de updates en aanpassingen zullen veel API’s verouderd zijn. Flutter heeft dit goed geanticipeerd door het Flutter fix commando toe te voegen. Deze zal de bestaande code doorlopen en tonen welke API’s verouderd zijn en in wat ze best vervangen worden.

Dart data classes staan op de toekomst plannen

Waarom deze studie?

- Zien of er nog problemen/moeilijkheden zijn bij het gebruik van Flutter
- Zien of en hoe je hier kan rondwerken

Cross-platform development heeft een snelgroeiende gebruikersbasis. De vele updates en verbeteringen aan de platformen maakt cross-platform development dan ook elke dag aantrekkelijker. Dit leidt tot meer onderzoek en studies binnen de sector. Tijdens de literatuurstudie bleek dat een groot aantal papers Flutter vergeleek met een ander cross-platform ontwikkel tool. Dit soort papers moet de lezer helpen bij het maken van een keuze tussen cross-platform tools. Door het verdelen van de aandacht over verschillende platformen, komt Flutter niet uitgebreid aanbod in de meeste van deze soort papers. Voor 2020 waren de meeste papers van deze aard. Flutter leek toen nog niet stabiel en groot genoeg om als afzonderlijke tool onderzocht te worden.

Zoals uit voorgaande sectie blijkt, is Flutter meerdere malen onderzocht in 2020. In verschillende studies werd het tekort aan onderzoek voor 2020 aangehaald. Flutter is in het afgelopen jaar in een stroomversnelling geraakt op vlak van onderzoek. Dit is te danken aan de grootte stijging in het gebruikersaantal. Het platform is nog steeds relatief jong maar de aantrekkingskracht ligt hem in de gebruikersbasis en de grootte steun. Hoe meer gebruikers, hoe belangrijker het product in kwestie. Hoe belangrijker het product, hoe meer middelen hierin worden geïnvesteerd en dat lokt op zijn beurt weer meer gebruikers. Deze cirkel wordt alleen onderbroken als het platform in kwestie zijn aantrekkelijkheid verliest. Flutter lijkt hier geen last van te hebben en deze studie wil laten blijken waarom.

Dus kan geconcludeerd worden dat Flutter een interessant onderzoeksdomein is. Maar zoals bij elk onderzoek binnen de IT, zijn bevindingen al snel verouderd. De rappe evolutie binnen de sector doet elk product streven naar de beste versie van zichzelf. De competitie is hoog, dus moet performantie altijd voorop staan. Bij elke update of verbetering zijn voorgaande performantie studies verouderd en niet accuraat. Hierdoor zijn de studies, aangehaald in dit hoofdstuk, dan ook vaak zo recent mogelijk. Toch onderging Flutter net een grootte release. Vele zaken werden verbeterd, onder andere de performantie van het platform. De voorgaande studies zijn weer verouderd en hier ligt een kans om opnieuw onderzoek te doen en beeld te schetsen van de huidige performantie van Flutter. Ook lijkt het interessant om een beeld te schetsen van de performantie van Flutter over tijd.

Methodologie

Inleiding

Experiment

Onderzoekscriteria

Grootte van de uitvoeringsbestanden

Inleiding

- Is al vaak onderzocht -> Misschien niet te diep in detail
- Metric die vaak veranderd
- Door de tijd bekijken
- Net grote Flutter release

De prijzen van de high-end smartphones zijn

Opzet

Resultaten

Android

Flutter

Conclusie

Opstartsnelheid van de app

CPU gebruik van de app

Inleiding

- Metric die vaak veranderd
- Door de tijd bekijken

Opzet

Resultaten

Android

Flutter

Conclusie

Gebruik van online API’s

Inleiding

- Toegankelijkheid voor nieuwe developpers
- Beschikbare libraries
- Edge cases

Opzet

Android

- Retrofit
- Moshi
- Chuck
- Glide

Resultaten

Android

Flutter

Conclusie

Security

Inleiding

- Door gehele app
- Richtlijnen zoeken
- In app security?
- Permissions
- Hoe gemakkelijk is dit alles te implementeren in beide frameworks

Opzet

Resultaten

Android

Flutter

Conclusie

Beschikbare libraries en code complexiteit

//codecomplexiteit checken adhv lijnen code voor specifiek onderdeel

Inleiding

- Code complexiteit
    - Duidelijk schetsen wat verschillende manieren van dit afwegen zijn

Wat is een library?

Resultaten

Android

Retrofit
Moshi
Chuck
Glide
Timber
Room
RxJava
Android KTX
Dagger
Koin
Material

Flutter

Cupertino

Conclusie

Creatie van views

//onvolledig

Inleiding

Hergebruik van middelen

Scherm dimensies

Animaties

Conclusie

Asynchroon werken

Inleiding

Wat is asynchroon werken?

Opzet

Android

- Co-routines
- Java Thread
- // AsyncTasks
- Android services
- Callbacks
- Event bus
- Reactive programming (RxJava)
- Chanel

Flutter

- Future
- Steams

Resultaten

Android

Flutter

Conclusie

Beschikbare tools

Inleiding

Wat wordt bedoelt met ‘tool’

Resultaten

- Android
- Flutter

Appendix

Conclusie

Bibliografie