

Inhoud

Vooraf	xvii
Over dit boek	xvii
De allerlaatste versie	xvii
De opbouw van dit boek	xviii
Voor wie is dit boek?	xix
De code in dit boek	xix
1 Wat is Flutter?	1
1.1 Wat kunt u met Flutter?	2
1.2 Wat is Dart?	3
1.3 Flutter en andere systemen	4
1.4 De toekomst van Flutter	5
2 Uw eerste Flutter-app	7
2.1 Wat hebt u nodig?	8
2.2 De Flutter SDK installeren	10
2.3 Beginnen met Android Studio	13
Android Studio installeren	13
Het welkomscherm en de Flutter-plug-in	13
2.4 Een demoapp maken	15
2.5 Android Studio gebruiken	16
De Project Explorer en de structuur van de app	16
Vensters	18
Flutter-vensters	18
Menu- en knoppenbalken	19
Opslaan en meer	20
Hot reload, hot restart en volledige herstart	20
2.6 Bouwen en testen voor meer platformen	21
Platformen inschakelen	22
2.7 De demoapp testen in Chrome of als bureaubladapp	23
Experimenteren	24

2.8	Testen op een virtueel Android-apparaat	25
2.9	Testen op een echt Android-apparaat	26
2.10	Testen op een virtueel iOS-apparaat	27
	iOS-simulator op een Mac	27
	iOS-emulator op andere computers	28
	Testen met een iOS-simulator of -emulator	28
2.11	Testen op een echt iOS-apparaat	28
	Instellingen van het Developers-account	29
	De app signeren	29
	Een apparaat aansluiten en testen	31
2.12	Startproblemen oplossen	31
	1. Bibliotheken opnieuw ophalen	31
	2. Een nieuwe build	31
	3. Platform opnieuw genereren	32
	Flutter doctor	33
	Meer informatie	33
3	Dart begrijpen	35
3.1	DartPad	36
	Nieuwe pads	36
	Basisregels in Dart	37
3.2	Functies en parameters	38
	Parameters	40
	Waarden retourneren met return	42
	Fat arrows	43
3.3	Variabelen	44
	Sterke typering	44
	Variabelen zonder vast type	45
	Constanten	46
3.4	Null	46
	Null safety	46
	Null safety in Flutter aan- of uitzetten	48
3.5	Namen in Dart	49
3.6	Gegevenstypen	50
	Integer (int)	51
	Getal met dubbele precisie (double)	51
	Boolean (bool)	51
	Tekst (String)	52
	Reeks (List)	53
	Map	54
	Meer gegevenstypen	55

3.7 Typen omzetten	55
Van String naar getal	56
Afronden op hele getallen	57
Afronden op aantal decimalen	57
3.8 Klassen, constructors en finals	58
Klassen	58
Overerving	60
Functies of variabelen overschrijven	62
Standaardconstructors	63
Initialisatielijsten	65
Finals	66
3.9 Methodes en eigenschappen bij typen en klassen	66
Constructors	68
Properties	69
Methods	70
Static methods	70
3.10 Operatoren	71
Rekenkundige operatoren	71
Wortels en machten: importeren	72
Toewijzingsoperatoren	72
Vergelijkingsoperatoren	73
3.11 Beslissingen nemen	75
Als...dan	75
Korte notatie	77
Switch	77
3.12 Lussen	78
For-lus	78
For...in-lus	80
While	81
3.13 Anonieme instanties en functies	82
Een timer met een callback	82
Stop de tijd	84
Anonieme functies	84
3.14 Recursieve functies	85
Periodic-constructor	87
3.15 Synchron en asynchron programmeren	88
Futures	89
Foute futures	90
Asynchrone functies	92
3.16 Fouten maken	93
3.17 Meer Dart	94

4	Flutter-widgets	97
4.1	Wat zijn Flutter-widgets?	98
	Alles in Dart	98
	Standaardwidgets	98
	Widgets, elementen en states	99
	Anonieme instanties	100
	Dart-bestanden	101
4.2	Widgets in de demoapp	102
	main() en MyApp	103
	MyHomePage()	104
4.3	Stateless en stateful widgets	106
	Stateless widgets	106
	Stateful widgets	108
	States wijzigen	111
	initState() en dispose()	113
4.4	Basiswidgets	114
	MaterialApp	114
	Scaffold	115
	Center-widget en contextacties	116
	Verouderde code bijwerken	117
	Een basisapp maken en bewaren	118
4.5	Rijen, kolommen en containers	121
	Rijen of kolommen maken	122
	AxisAlignment	123
	Absolute grootte	125
	Relatieve grootte: Expanded en Flexible	125
	Nesting	127
	Stapelen	129
	Overzicht in de code	130
	Scrollen	130
	Informatie over widgets	132
4.6	Menubalk	133
	Routes maken	133
	De controller	134
	De stack	135
	De knoppenbalk	136
	Pictogrammen	136
4.7	Dynamische navigatie	138
	Twee routes maken	139
	De navigatieknoppen activeren	141
4.8	Afbeeldingen	142
	Een afbeelding gebruiken	143

4.9	Geluid en packages	148
	Geluidsbestand toevoegen	148
	pubspec.yaml aanpassen	149
	Een package importeren	149
	De audioplayer gebruiken	151
4.10	Video's	154
	Package installeren	154
	Videobestand toevoegen	154
	Video laden en starten	155
	De beeldverhouding vastzetten	156
	De video interactief maken	157
	Herhalen	158
	De interactiviteit uitbreiden	158
	Van stateless naar stateful	159
4.11	Teksten en opmaak	162
	Padding en marge	162
	Tekststijlen	164
	Tekstgrootte	165
	Meerdere stijlen in één tekst	166
	Standaardstijlen in een thema	167
	Cupertino	168
	Lettertypen toevoegen	169
4.12	Interactie	171
	GestureDetector	172
	Knoppen	175
4.13	Gegevensinvoer	178
	Element verbergen met Checkbox	179
	Switch	180
	Keuzerondjes maken met Radio	181
	Schuifregelaars	182
	Keuzelijst	183
	Tekstinvoer	185
	iOS-elementen	186
	Besturingssysteem detecteren	187
4.14	Animatie	188
	Zelf een animatie definiëren	188
	De status van de animatie	191
	Animatie beëindigen	191
	Gemakkelijker animeren	192

4.15	Figuren tekenen	194
	CustomPaint maken op basis van schermgrootte	195
	Painter maken	195
	Tekenopdrachten geven	196
	Tunnels en bogen	197
	Canvasanimatie	199
4.16	Lijsten, eigen widgets en keys	202
	Een lijst met kaarten	202
	Herhaling voorkomen	203
	Eigen widgets maken	203
	Dismissible en keys	206
	Key	207
4.17	Gegevens doorgeven	208
	Publieke variabelen	208
	Inherited widget	209
	Inherited widget maken	209
	Inherited widget lezen	211
	Naar een inherited widget schrijven	212
	Bibliotheken	212
	Streams en streamcontrollers	214
4.18	Gegevens bewaren en futures gebruiken	217
	Stateful widget met teller	217
	shared_preferences importeren	218
	Gegevens opslaan	219
	Gegevens lezen	220
	FutureBuilder	221
4.19	Meer widgets	225
5	Een complete app	227
5.1	De app in dit hoofdstuk	228
	Functionele eisen	229
	Broncode	230
5.2	Fouten opsporen en analyseren	231
	Foutmeldingen	231
	Een app debuggen	232
5.3	De basisstructuur	234
	Een lege app	234
	Schermen	234
	Knoppenbalk	235
	Bibliotheek	237

5.4	Structuur van de quiz	238
5.5	De lay-out voor het vraagscherm	241
	Eerste rij: vraagnummer en score	242
	Tweede rij: de afbeelding	243
	Derde rij: de vraag	244
	Antwoordopties	245
	Antwoordknoppen	245
5.6	Vragen	248
	Afbeeldingen toevoegen	250
5.7	Inhoud in de quiz	250
	In quiz.dart	251
	In vraag.dart	251
	In antwoordKnop.dart	253
	Optie: flexibel aantal antwoorden met control-flow	255
5.8	Interactieve antwoordknoppen	257
	Interactiviteit detecteren	258
	Kleurovergang animeren	258
	Optie: meer animaties met dezelfde controller	261
	Optie: geluid	262
	Antwoord verwerken met een parameterfunctie	265
5.9	Het uitslagscherm	270
	De score doorgeven	270
	De uitslag	271
	Een herstartknop	272
	De quiz herstarten met een stream	272
	Optie: state van de quiz bewaren	276
5.10	Optie: uitslag versturen	277
	Mailserver	277
	De mailer importeren	278
	De mailknop	279
	Het dialoogvenster	280
	De mailfunctie	281
	Smtplib-server	282
	Een bericht maken	282
	Bericht versturen	283
	Variabelen in de mailfunctie	284
	De score	285
	Het opgegeven e-mailadres	285
	Datum en tijd	286
	Variabelen in het bericht	286
	Het invoerveld valideren	287
	De gebruiker informeren	290

Uitslag ombouwen naar stateful	290
Variabele tekst in uitslagscherm	291
Functie in _UitslagState	292
Functie doorgeven aan MailDialog	293
Status aanpassen vanuit MailDialog	293
De mailfunctie verder verfijnen	294
5.11 Optie: vragen uit een online bron	295
Online bestanden	296
Toegangsproblemen	296
Vraag- en afbeeldingsbestanden	298
Vragen ophalen	299
Afbeeldingen ophalen	302
Start van de quiz	303
Antwoorden verbergen	304
5.12 Infoscherm en beginscherm	305
Lay-out van het infoscherm	305
Links toevoegen	307
Optie: meertaligheid	308
Welkomscherm	311
6 Een app afronden en publiceren	315
6.1 Pictogram en opstartscherm	316
Afbeeldingen voor pictogram en splashscreen	316
Pictogram voor iOS en Android instellen	317
Pictogrammen voor andere systemen	319
Splashscreens	320
6.2 Controles en instellingen	322
Dart Analysis	322
Pubspec.yaml	323
AndroidManifest.xml	323
build.gradle	324
iOS-instellingen in Xcode	325
Signing and capabilities	326
Testen	327
6.3 Een Android-app signeren en compileren	327
Keystore maken	327
Certificaat in pubspec.yaml opnemen	328
Een appbundle maken	329
6.4 Een app in Google Play Store plaatsen	330
Google Play Console en developersaccount	330
Nieuwe app maken	330
Winkelvermelding	331

App-releases	332
Contentclassificatie	334
App-content	334
Prijzen en distributie	334
App indienen	335
6.5 iOS-certificaten & -identifiers	336
Distributiecertificaat maken	337
Bundle-identificer maken	338
Provisioning profile maken	339
6.6 iOS-app aanmaken en uploaden	339
App aanmaken in App Store Connect	340
iOS-build maken vanuit Android Studio	341
Build valideren en uploaden vanuit Xcode	341
Handmatig signeren	342
6.7 Een iOS-app testen met TestFlight	343
6.8 De app weergeven in de Apple App Store	344
App Store information	345
iOS app	345
Screenshots	345
Beschrijvingen en URL's	346
Build	347
App Store Icon	347
Copyright	347
Age Rating	347
App Review Information	347
De app indienen	348
6.9 Webapps en desktopapps exporteren	349
App exporteren	349
App distribueren	350
Index	353

Vooraf

Over dit boek

Dit boek geeft u een vliegende start bij het maken van apps. U bouwt deze apps in Flutter: een systeem van Google dat het mogelijk maakt om echte *native* apps te bouwen voor Android, iOS, macOS, Windows, Linux en het web. Native wil zeggen dat uw apps de hardware direct aansturen, zonder JavaScript of andere tussenliggende lagen. De taal die u daarbij gebruikt heet Dart. Ook de beginselen van die taal komen in dit boek aan bod.

Flutter, Dart, Android Studio en de andere thema's in dit boek zijn omvangrijk genoeg voor afzonderlijke boeken. Dit boek is dan ook niet uitputtend: na het doornemen weet u niet alles van Flutter en Dart. Maar u weet wel genoeg om vanaf niets de programmeer- en testomgeving op te zetten, een app te bouwen, te testen en te publiceren in de stores. De nadruk in dit boek ligt op het begrijpen van het systeem en de taal. Als u doorgrondt hoe alles werkt, is het daarna gemakkelijk om meer informatie te vinden en verder te leren. Dit boek is daarvoor een goede springplank.

De allerlaatste versie

De ontwikkeling van Flutter stond bepaald niet stil sinds de eerste editie van dit boek in 2020. Er kwamen vooral veel mogelijkheden bij. Soms veranderden er ook zaken, waardoor de oorspronkelijke code niet meer werkte. Gelukkig geeft Flutter zelf dan duidelijk aan hoe u de code kunt aanpassen. Wij houden de ontwikkelingen ook nauwgezet in de gaten en zorgen ervoor dat u wijzigingen in tekst (errata) en de nieuwste broncode altijd gratis kunt downloaden. Dat kan vanaf **boek.flutter.nl** of vanaf de website van de uitgever, **www.vanduurenmedia.nl** (zoek op Flutter).

Op 3 maart 2021 lanceerde Google Flutter 2.0. De belangrijkste wijziging is de uitbreiding van het aantal systemen waarop de Flutter-apps werken: de eerste versie van Flutter was alleen gericht op mobiele apparaten met Android en iOS. En eerlijk is eerlijk: tijdens het programmeren zult u merken dat Flutter vooral voor deze systemen goed ontwikkeld is. Op andere systemen werken nog niet alle uitbreidingsbibliotheken, maar toch: u kunt met Flutter nu ook apps maken voor web, macOS, Windows en Linux. En dat allemaal op basis van één broncode.

De opbouw van dit boek

Dit boek bestaat uit zes hoofdstukken. Het eerste hoofdstuk beschrijft wat Flutter is en wat u ermee kunt. Als u twijfelt of u zich in Flutter wilt verdiepen, lees dan vooral dit hoofdstuk. In hoofdstuk 2 leest u wat u nodig hebt om op uw systeem Flutter-apps te ontwikkelen en te testen. Al deze software is overigens gratis. Aan het einde van dit hoofdstuk hebt u een demoapp die op meerdere platformen werkt.

Hoofdstuk 3 is een inleiding in de programmeertaal Dart. We illustreren de belangrijkste concepten met korte codevoorbeelden die u los van elkaar kunt gebruiken. Zo kunt u (later) gemakkelijk paragrafen los van elkaar nog eens doornemen. In hoofdstuk 4 behandelen we widgets: de bouwstenen van een Flutter-app. Ook hier: korte voorbeelden, zodat u elke paragraaf los kunt naslaan.

In hoofdstuk 5 passen we alles toe en ontwikkelen we een volledige app. In tegenstelling tot hoofdstuk 3 en 4 is de inhoud van dit hoofdstuk wel één volledige, doorlopende lijn. Hierin passen we concepten uit de vorige hoofdstukken toe en voegen daar nieuwe aan toe. Hoofdstuk 6 ten slotte laat u zien hoe u een app publiceert in de appstores. Daarbij besteden we vooral aandacht aan de publicatie van mobiele apps (dus voor Android en iOS), maar we geven ook handvatten voor andere platformen.

Voor wie is dit boek?

Met Flutter bouwt u apps met de programmeertaal Dart. We dui-ken dan ook de code in. Het helpt daarbij als u wat ervaring hebt met programmeren of webontwikkeling, maar ook als u die niet hebt, lukt het u om met dit boek uw eigen apps te maken. Dit boek is dus bedoeld voor nieuwe appmakers en voor mensen die andere systemen gewend zijn en snel op stoom willen komen met Flutter.

De code in dit boek

De code in dit boek is in het algemeen kort en bondig. We raden u aan om deze zo veel mogelijk zelf over te typen en te experimen-teren met wijzigingen. Dat is de beste manier om de taal en het systeem te leren kennen. Als u dat niet wilt, of als u een fout niet kunt ontdekken, dan is de code voor de hoofdstukken 4 en 5 beschikbaar als download op www.vanduurenmedia.nl (zoek op Flutter) en op boek.flutter.nl. U opent de voorbeeldcode als volgt:

- 1 Download het zip-bestand en pak het uit.
- 2 Klik in Android Studio op **File, Open** en blader naar de locatie waar u de het bestand hebt uitgepakt.
- 3 Kies een van de hoofdmappen, h4_widgets of h5_quiz, en klik op **Open**.
- 4 Open het bestand pubspec.yaml en klik op de knop **Pub get**.

Het bestand h4_widgets bevat de code van de losse widgets die u maakt in paragraaf 4.3 tot en met 4.19. Deze widgets zijn dus ver-zameld in één app. De app is voorzien van een overzichtspagina, waarmee u naar alle widgets kunt navigeren. Dit werkt binnen de app, maar u kunt de meeste widgets ook kopiëren en uitvoeren in een Flutter-DartPad (zie paragraaf 3.1). Uitzonderingen zijn de

widgets die gebruikmaken van aanvullende bestanden, zoals afbeeldingen, geluid of clips.

Het bestand h5_quiz bevat de volledige code van de app die u in hoofdstuk 5 maakt. In de app zijn ook bronbestanden per paragraaf opgenomen. Deze geven de situatie aan het eind van een paragraaf weer. Als u dus bij paragraaf 5.5 zou willen beginnen, dan gebruikt u de code die onder 5.4 staat.



Afbeelding 0.1 *De overzichtspagina van de app met voorbeeldwidgets van hoofdstuk 4.*

Wat is Flutter?

Er zijn verschillende systemen om apps te bouwen. Is Flutter voor u de beste keuze of is het beter om tijd te investeren in een ander systeem? Het antwoord op die vraag hangt af van het soort apps dat u wilt maken en de besturingssystemen en apparaten waarvoor u dat wilt doen. Dit hoofdstuk laat zien wat Flutter is en hoe het zich verhoudt tot andere systemen. Op basis daarvan kunt u inschatten wat het u oplevert als u verder gaat met Flutter.

U leert in dit hoofdstuk:

Wat Flutter doet.

Verschillen en overeenkomsten met andere systemen om apps te ontwikkelen.

De toekomst van Flutter.



1.1 Wat kunt u met Flutter?

In één zin: Flutter is een softwareontwikkelplatform om *native* apps te bouwen voor iOS, Android en andere besturingssystemen. Maar wat betekent dat nu eigenlijk?

Flutter is dus een softwareontwikkelplatform (*software development kit* of SDK). Dat is een verzameling hulpmiddelen om software te maken. Daarbij horen een programmeertaal, een programmeeromgeving (de *integrated development environment* of IDE), documentatie en middelen om apps te distribueren. Flutter werkt met de programmeertaal Dart. Om Dart te programmeren kunt u verschillende IDE's gebruiken, bijvoorbeeld Visual Studio, IntelliJ Idea en Android Studio. In dit boek gebruiken we de laatste.

Een *native* app is een app die direct de hardware van een systeem aanstuurt. Zowel iOS als Android hebben eigen, native programmeertalen, respectievelijk Swift en Kotlin (de opvolgers van Objective-C en Java). Ook Flutter zet Dart-programma's om naar code die direct met de hardware communiceert.

Flutter-apps zijn geschikt voor iOS en Android: de besturingssystemen van het overgrote deel van tablets en smartphones. U kunt apps ook exporteren als webapp (PWA: *progressive web app*) of desktopapp.

Flutter is een volwassen en stabiel ontwikkelplatform voor Android en iOS. Bedrijven als Philips, Sonos, Alibaba, Google, Albert Heijn en Toyota maken er bijvoorbeeld gebruik van. De basis van Flutter werkt ook prima in web- en desktopapps, al kunt u hierin nog niet alle uitbreidingsbibliotheken (*packages*) gebruiken. Maar niets weerhoudt u ervan om uw apps ook hiervoor te exporteren en te testen. Vooral op dit gebied zal Flutter zich de komende periode ontwikkelen.

1.2 Wat is Dart?

Een computertaal, zoals Dart, is een verzameling gestandaardiseerde instructies die een computer vertellen wat deze moet doen. Computertalen zijn voor mensen nog redelijk te begrijpen, maar toch goed om te zetten naar de vrijwel onbegrijpelijke machinetaal waar een computer mee werkt. Dat omzetten, of compileren, kan op twee manieren:

- JIT (Just In Time)-talen compileren de code terwijl het programma uitgevoerd wordt. Het bekendste voorbeeld van een JIT-taal is JavaScript. Een JIT-taal is flexibel en het is gemakkelijker om fouten op te sporen: een programmeur ziet direct het resultaat van wijzigingen en kan de broncode van een programma dat in gebruik is, gewoon bekijken.
- AOT (Ahead of Time)-talen compileren de code vooraf. Als de programmeur klaar is, zet de IDE de code om naar machinetaal. Dat duurt soms minutenlang. Daar staat tegenover dat AOT-programma's sneller werken. De broncode van een eenmaal gecompileerd programma is in principe niet meer te lezen of te veranderen.

De meeste talen zijn of JIT of AOT, maar Dart kan het allebei. Tijdens het programmeren werkt u met de JIT-versie. Een simulator of een echt apparaat kan daardoor verandering razendsnel, vaak binnen een seconde, weergegeven. De app werkt dan wel langzamer. Als de app af is duurt het compileren enkele minuten, met een snelle AOT-versie als resultaat.

De taalregels (*syntax*) en structuur van Dart lijken sterk op AOT-talen zoals Swift en Kotlin. Zo zijn er strenge regels voor het gebruik van functies en gegevenstypen. Bij veel JIT-talen zijn die regels wat lossier. In hoofdstuk 3 komen we uitgebreid terug op de programmeertaal Dart.



1.3 Flutter en andere systemen

Swift, Kotlin en hun voorlopers Objective-C en Java zijn prachtige talen waarin u snelle, compacte apps kunt bouwen. Ze benutten de mogelijkheden van de smartphone en tablet optimaal, maar ze hebben één groot nadeel: ze werken maar op één besturings-systeem. Als u hiermee een app voor iOS en Android wilt maken, moet u twee complete apps in twee verschillende talen en omgevingen programmeren, testen en onderhouden.

Er bestaan wel alternatieven om in één keer een app voor iOS en Android te bouwen. Zo zijn er verschillende systemen die HTML, CSS en JavaScript, de bouwstenen van het web, in een app omzetten. Met deze ‘webhybriden’ kunt u dus dezelfde code voor een website én een app gebruiken. Cordova is daarbij het meest gebruikte platform.

Daarnaast bestaan er zogeheten *native hybriden*. Deze maken gebruik van onderdelen van de gebruikersinterface van het systeem zelf. Op een iOS-apparaat gebruikt de app de schuifjes, knoppen, tekstvakken en andere elementen van iOS. Op een Android-toestel benut de app Android-elementen. Dat maakt de apps sneller dan de webhybriden. Het bekendste voorbeeld van zo’n systeem is React Native van Facebook. JavaScript verbindt in dit systeem de onderdelen en zorgt voor de verwerking van gegevens.

Het gebruik van JavaScript is het belangrijkste voordeel én nadeel van dergelijke systemen. Veel mensen zijn thuis in deze taal, dus het maakt de ontwikkeling van de apps gemakkelijker. Maar de prestaties van de JIT-taal blijven achter bij native apps. Bij apps die veel standaardelementen uit de gebruikersinterface gebruiken, zoals communicatieapps, to-dolijstjes en dergelijke is het verschil niet of nauwelijks merkbaar. Bij grafische apps, spelletjes en

andere apps die weinig standaardelementen gebruiken is het verschil groter.

Flutter stuurt de hardware wel direct aan. Alle interactieve elementen liggen in de app zelf vast en zijn niet afhankelijk van het besturingssysteem. U kunt de app wel bij verschillende systemen aan laten sluiten door deze meerdere thema's mee te geven, bijvoorbeeld één voor Android en één voor iOS. Flutter-apps zijn compact en werken, mits goed geprogrammeerd, razendsnel. De belangrijkste drempel is dat u er een aparte taal en SDK voor moet leren. Daar kan dit boek bij helpen!

1.4 De toekomst van Flutter

Programmeertalen en -omgevingen gaan niet eeuwig mee. Na verloop van tijd verdwijnen ze om plaats te maken voor systemen die meer mogelijkheden bieden of gemakkelijker te programmeren zijn. De vraag is daarom niet of Flutter ooit verdwijnt (dat doet het ongetwijfeld), maar hoe lang het meegaat. Sterft Flutter na een paar jaar weer een stille dood of blijft het decennia een populair platform?

Dat is nooit helemaal met zekerheid te zeggen, maar de voor tekenen zijn goed. Flutter is het enige systeem waarmee u native apps voor zo veel verschillende platformen kunt maken. Het aantal programmeurs en bedrijven dat met Flutter werkt groeit dan ook snel. Voorbeelden van Flutter-apps van bekende bedrijven vindt u op www.flutter.dev/showcase. Ook het aantal vacatures neemt toe. Toen ik in 2019 begon met de eerste versie van dit boek, kwam ik nooit iemand tegen die Flutter kende. Nu (april 2021) is het wereldwijd het meest gebruikte systeem voor multiplatform-apps. Waarschijnlijk heeft dat ook met de kosten te maken: Google geeft de code van Flutter en alle onderdelen vrij. Flutter is dus gratis en open source.



Bijna helemaal gratis

Flutter en alle ontwikkelsoftware die u daarbij nodig hebt, is gratis. Publiceren in de webshops van Google en Apple kost wel geld. Voor Google is dat eenmalig ongeveer 25 euro, bij Apple ongeveer 100 euro per jaar.

Uw eerste Flutter-app

Flutter is zo gedownload en geïnstalleerd. Maar om Flutter-apps te maken hebt u ook software nodig om te programmeren en te testen. Dit hoofdstuk laat u stap voor stap zien hoe u deze downloadt, installeert en gebruikt. Aan het einde van het hoofdstuk hebt u een werkende demoapp in Flutter.

U leert in dit hoofdstuk:

Een systeem klaarmaken om Flutter-apps te ontwikkelen.

Een demoapp maken.

De belangrijkste Flutter-functies in Android Studio.

Een app testen in een simulator en op een tablet of telefoon.

2.1 Wat hebt u nodig?

In dit hoofdstuk gaat u de ontwikkelomgeving opzetten en een eerste demoapp maken. Om Flutter-apps te bouwen hebt u in elk geval deze zaken nodig:

- Een computer met flink wat opslagruimte. Reken voor alle software en simulators op zo'n 25 GB. Voor de broncode van elke app hebt u zeker 1 GB nodig (terwijl de apps die u exporteert juist heel klein zijn). Als de vrije opslagruimte van uw computer te klein is, dan is een extern station een goede en betaalbare oplossing. Deze hebben doorgaans een lagere snelheid dan een ingebouwd station, maar bij het ontwikkelen van Flutter-apps is dat zelden een probleem.
- De Flutter SDK. Daarover gaat de volgende paragraaf.
- Een programmeeromgeving (IDE). In dit boek gebruiken we Android Studio, omdat het verschillende handige tools heeft voor het werken met Flutter. Als u al in een andere omgeving programmeert die ook Dart ondersteunt (bijvoorbeeld Visual Studio Code, IntelliJ Idea), dan kunt u ook daarmee werken.

Hiermee kunt u apps bouwen, testen en publiceren voor Android en voor het web. Als u op Android mikt, is het daarnaast handig als u in elk geval één echte Android-telefoon of -tablet hebt om op te testen. Dat mag best een wat ouder, afgedankt toestel zijn, maar het is ook geen probleem als u het nog in gebruik hebt.

Extra benodigheden voor iOS:

Publiceren en testen voor iOS lukt alleen vanaf een Apple-computer. Daarvoor hebt u nodig:

- Een MacBook of iMac.
- Xcode: de gratis programmeeromgeving van Apple. We gebruiken Xcode niet om te programmeren (dat doen we in

Android Studio) maar wel om apps te configureren, instellingen op te geven, te testen en te exporteren naar de App Store van Apple.

- Een iOS-apparaat of simulator om te testen. Met Xcode wordt standaard de iOS-simulator geïnstalleerd, maar ook hier is het fijn als u daarnaast een echte iPad of iPhone hebt.

In de volgende paragrafen leest u hoe u alle onderdelen downloadt en installeert.

Extra benodigdheden voor desktopapps

U kunt een app alleen exporteren voor en testen op het systeem dat u zelf gebruikt. U kunt dus alleen een Windows-app exporteren vanaf een Windows-computer, een macOS-app vanaf een Apple-computer en Linux-apps vanaf een Linux-computer. Daarvoor kunt u uiteraard wel dezelfde code gebruiken. U kunt dus een eenmaal gemaakte app van een Windows-computer naar een Apple-computer kopiëren en hem daar opnieuw testen en exporteren. Verder geldt voor desktopapps:

- Voor macOS hebt u dezelfde zaken nodig als voor iOS-apps. Zie hierboven.
- Voor Windows hebt u de gratis communityversie van Visual Studio (dus *niet* Visual Studio Code) nodig om de app te exporteren. Dit heeft dezelfde rol als Xcode op de Mac. Zoek dit programma op internet en installeer het vanaf de website of vanuit de Microsoft-store.
- Installeer Flutter voor Linux niet vanaf **flutter.dev** (zoals in de volgende stap wordt beschreven), maar vanuit de Snap Store. U installeert dan direct de meeste extra programma's. Om apps te exporteren en te distribueren moet u daarnaast snapcraft en multipass installeren, met de opdrachten `sudo snap install snapcraft --classic` en `sudo snap install multipass --classic`

2.2 De Flutter SDK installeren

De Flutter SDK installeert u zo:

1. Surf naar de website **flutter.dev** en klik rechtsboven op **Get started** of typ `install flutter` in uw zoekmachine. U komt nu op de Flutter-webpagina Install terecht.
2. Klik op uw besturingssysteem (Windows, Mac, Linux of Chrome OS).
3. Download het installatiebestand. Klik daarvoor op de blauwe knop onder *Get the Flutter SDK*.
4. Unzip het gedownloade pakket. Verplaats de uitgedeelte map naar een locatie van waaruit u deze wilt gebruiken (bijvoorbeeld `/users/<uw naam>/flutter/` op een Mac of `C:\flutter` in Windows).
5. Elke computer heeft een PATH-instelling. Die geeft aan in welke mappen het systeem zoekt als u een opdracht geeft. U moet de Flutter-map aan het PATH toevoegen.

Op een Mac kan dat onder meer als volgt:

- Open het programma Terminal.
- Typ de opdracht `sudo nano /etc/paths`, druk op Return en geef uw wachtwoord op. Hiermee opent u een eenvoudige tekstverwerker (Nano) waarin u de PATH-instelling kunt bewerken.
- Ga met de cursor naar de onderste regel en voeg de map uit stap 4 toe met daarachter `/bin`. Bijvoorbeeld `/users/<uw naam>/flutter/bin`.
- Druk op Ctrl-X om Nano af te sluiten. Druk daarna op Y om op te slaan.
- Meld u af en weer aan of herstart de computer.
- U kunt de instelling controleren met de opdracht `echo $PATH`. (U ziet dat er veel meer paden bestaan dan die in het bestandje zijn opgenomen).

Zo past u PATH in Windows aan:

- Typ het woord `systeminstellingen` in het zoekvak van het menu Start (cq. de Taakbalk) en kies **Geavanceerde systeeminstellingen weergeven**.
- Open het tabblad **Geavanceerd**. Klik hier in het onderste deel van het venster op **Omgevingsvariabelen**.
- In het bovenste deel van het venster staan instellingen voor de huidige gebruiker. Het onderste deel is voor alle gebruikers. Dubbelklik op de variabele **Path** in een van beide secties.
- Klik op **Nieuw** en voeg het pad naar de Flutter-map toe, met daarachter `bin\`. (Bijvoorbeeld: `C:\flutter\bin`). Bij oudere Windows-versies ontbreekt de knop **Nieuw** en ziet u het hele pad in een klein dialoogvenster. U kunt hier aan het eind, na een puntkomma, het pad toevoegen.
- Klik daarna op **OK** in elk venster.
- Meld u af en weer aan of herstart de computer.
- Open PowerShell (typ `shell` in het zoekvak van de Taakbalk). Geef de opdracht `$Env:Path` om te controleren of de Flutter-map goed aan het eind van de variabele is toegevoegd. (Gebruik `echo %Path%` bij oudere Windows-versies.)

Typ nu de opdracht `flutter doctor` in Terminal (Mac) of Windows PowerShell. Hiermee controleert u de installatie. Als het goed is krijgt u een vinkje bij de eerste test (Flutter). Daaronder volgen foutmeldingen over het ontbreken van Android Studio en andere onderdelen. Daar werken we in de volgende paragraaf aan.


```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Mark> flutter doctor

Welcome to Flutter! - https://flutter.dev

The Flutter tool uses Google Analytics to anonymously report feature usage
statistics and basic crash reports. This data is used to help improve
Flutter tools over time.

Flutter tool analytics are not sent on the very first run. To disable
reporting, type 'flutter config --no-analytics'. To display the current
setting, type 'flutter config'. If you opt out of analytics, an opt-out
event will be sent, and then no further information will be sent by the
Flutter tool.

By downloading the Flutter SDK, you agree to the Google Terms of Service.
Note: The Google Privacy Policy describes how data is handled in this
service.

Moreover, Flutter includes the Dart SDK, which may send usage metrics and
crash reports to Google.

Read about data we send with crash reports:
https://github.com/flutter/flutter/wiki/Flutter-CLI-crash-reporting

See Google's privacy policy:
https://www.google.com/intl/en/policies/privacy/

Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, v1.12.13-hotfix.8, on Microsoft Windows [Version 10.0.18363.728], locale nl-NL)
[!] Android toolchain - develop for Android devices
    Unable to locate Android SDK.
    Install Android Studio from: https://developer.android.com/studio/index.html
    On first launch it will assist you in installing the Android SDK components.
    (or visit https://flutter.dev/setup/android-setup for detailed instructions).
    If the Android SDK has been installed to a custom location, set ANDROID_HOME to that location.
    You may also want to add it to your PATH environment variable.
[!] Android Studio (not installed)
[!] Connected device
    ! No devices available

! Doctor found issues in 3 categories.
PS C:\Users\Mark>
```

Afbeelding 2.1 *Flutter doctor in Windows PowerShell: Flutter is goed geïnstalleerd, maar andere onderdelen ontbreken nog.*



Problemen bij het installeren van de SDK

Krijgt u bij de opdracht `flutter doctor` een melding als ‘Command not found’ of ‘The term ‘flutter’ is not recognized’? Ga dan naar de Flutter-map en probeer het daar nog een keer. Werkt het nu wel? Dan is het pad niet goed ingesteld. Zie stap 5 hierboven. Werkt het nog niet? Dan kan het zijn dat Flutter geïnstalleerd is op een plaats waarvoor u onvoldoende rechten hebt. Verplaats de Flutter-map, pas het pad aan en probeer het nog een keer. Krijgt u andere foutmeldingen? Lees deze goed. Soms ontbreken systeemonderdelen die nodig zijn voor een goede werking van Flutter. De melding geeft aan waar u die kunt downloaden.