

# Inhoud

<b>1</b>	<b>Wat is Flutter?</b>	<b>1</b>
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
<b>2</b>	<b>Uw eerste Flutter-app</b>	<b>7</b>
2.1	Wat hebt u nodig?	8
2.2	De Flutter SDK installeren	9
2.3	<b>Beginnen met Android Studio</b>	<b>12</b>
	Android Studio installeren	12
	Het welkomscherm en de Flutter-plug-in	13
2.4	<b>Een demo-app maken</b>	<b>14</b>
2.5	<b>Android Studio gebruiken</b>	<b>15</b>
	De Project Explorer en de structuur van de app	16
	Vensters	17
	Flutter-vensters	17
	Menu- en knoppenbalken	18
2.6	<b>Testen op een virtueel Android-apparaat</b>	<b>19</b>
2.7	<b>Testen op een echt Android-apparaat</b>	<b>20</b>
2.8	<b>Testen op een virtueel iOS-apparaat</b>	<b>22</b>
	iOS-simulator op een Mac	22
	iOS-emulator op andere computers	23
	Testen met een iOS-simulator of -emulator	23
2.9	<b>Testen op een echt iOS-apparaat</b>	<b>23</b>
	Instellingen van het Developers-account	24
	De app signeren	24
	Een apparaat aansluiten en testen	26
	Service protocol	26
2.10	<b>Flutter geschikt maken voor webapps en desktopapps</b>	<b>27</b>
	De web- en desktopoptie activeren	27
	Web- en desktopapps testen	28

<b>3</b>	<b>Dart begrijpen</b>	<b>31</b>
<b>3.1</b>	<b>DartPad</b>	<b>32</b>
	Nieuwe pads	32
	Basisregels in Dart	33
<b>3.2</b>	<b>Functies en parameters</b>	<b>34</b>
	Parameters	36
	Waarden retourneren met return	38
	Fat arrows	39
<b>3.3</b>	<b>Variabelen</b>	<b>39</b>
	Sterke typering	39
	Variabelen zonder type	41
	Onveranderlijke variabelen	41
	Null	42
<b>3.4</b>	<b>Namen in Dart</b>	<b>43</b>
<b>3.5</b>	<b>Gegevenstypen</b>	<b>44</b>
	Integer (int)	44
	Getal met dubbele precisie (double)	44
	Boolean (bool)	45
	Tekst (String)	45
	Reeks (List)	46
	Map	48
	Meer gegevenstypen	48
<b>3.6</b>	<b>Typen omzetten</b>	<b>49</b>
	Van String naar getal	49
	Afronden op hele getallen	50
	Afronden op aantal decimalen	51
<b>3.7</b>	<b>Klassen, constructors en finals</b>	<b>51</b>
	Klassen	51
	Overerving	54
	Functies of variabelen overschrijven	55
	Standaardconstructors	56
	Initialisatielijsten	58
	Finals	59
<b>3.8</b>	<b>Methodes en eigenschappen bij typen en klassen</b>	<b>59</b>
	Constructors	61
	Properties	62
	Methods	63
	Static methods	63
<b>3.9</b>	<b>Operatoren</b>	<b>64</b>
	Rekenkundige operatoren	64
	Wortels en machten: importeren	65
	Toewijzingsoperatoren	65
	Vergelijkingsoperatoren	66

<b>3.10</b>	<b>Beslissingen nemen</b>	<b>68</b>
	Als...dan	68
	Korte notatie	70
	Switch	70
<b>3.11</b>	<b>Lussen</b>	<b>71</b>
	For-lus	71
	For...in-lus	73
	While	74
<b>3.12</b>	<b>Anonieme instanties en functies</b>	<b>75</b>
	Een timer met een callback	75
	Stop de tijd	77
	Anonieme functies	77
<b>3.13</b>	<b>Recursieve functies</b>	<b>78</b>
	Periodic-constructor	79
<b>3.14</b>	<b>Synchroon en asynchroon programmeren</b>	<b>80</b>
	Futures	82
	Foute futures	83
	Asynchrone functies	84
<b>3.15</b>	<b>Fouten maken</b>	<b>85</b>
<b>3.16</b>	<b>Meer Dart</b>	<b>87</b>
<b>4</b>	<b>Flutter-widgets</b>	<b>89</b>
<b>4.1</b>	<b>Wat zijn Flutter-widgets?</b>	<b>90</b>
	Alles in Dart	90
	Standaardwidgets	90
	Widgets, elementen en states	91
	Anonieme instanties	92
	Dart-bestanden	93
<b>4.2</b>	<b>Widgets in de demo-app</b>	<b>94</b>
	main() en MyApp	95
	MyHomePage()	96
<b>4.3</b>	<b>Stateless en stateful widgets</b>	<b>98</b>
	Stateless widgets	98
	Stateful widgets	99
	States gebruiken	101
	initState() en dispose()	103
<b>4.4</b>	<b>Basiswidgets</b>	<b>104</b>
	MaterialApp	104
	Scaffold	105
	Center-widget met de Flutter Outline	106
	Een basisapp maken en bewaren	107
<b>4.5</b>	<b>Rijen, kolommen en containers</b>	<b>109</b>
	Rijen of kolommen maken	110
	AxisAlignment	112

	Absolute grootte	113
	Relatieve grootte: Expanded en Flexible	114
	Nesting	115
	Stapelen	116
	Overzicht in de code	118
	Scrollen	118
	Informatie over widgets	120
<b>4.6</b>	<b>Menubalk</b>	<b>120</b>
	Routes maken	121
	De controller	121
	De stack	122
	De knoppenbalk	123
	Pictogrammen	124
<b>4.7</b>	<b>Dynamische navigatie</b>	<b>125</b>
	Twee routes maken	126
	De navigatieknoppen activeren	127
<b>4.8</b>	<b>Afbeeldingen</b>	<b>129</b>
	Een afbeelding gebruiken	130
<b>4.9</b>	<b>Geluid</b>	<b>134</b>
	Geluidsbestand toevoegen	135
	pubspec.yaml aanpassen	135
	Een player importeren	135
	De player gebruiken	137
<b>4.10</b>	<b>Video's</b>	<b>139</b>
	Plug-in installeren	139
	Video laden en starten	140
	De beeldverhouding vastzetten	141
	De video interactief maken	142
	Herhalen	143
	De interactiviteit uitbreiden	143
	Van stateless naar stateful	144
<b>4.11</b>	<b>Teksten en opmaak</b>	<b>147</b>
	Padding en marge	147
	Tekststijlen	149
	Tekstgrootte	150
	Meerdere stijlen in één tekst	151
	Standaardstijlen	151
	Cupertino	153
	Lettertypen toevoegen	153
<b>4.12</b>	<b>Interactie</b>	<b>156</b>
	GestureDetector	156
	Knoppen	161
<b>4.13</b>	<b>Gegevensinvoer</b>	<b>163</b>
	Element verbergen met Checkbox	164

Switch	166
Keuzerondjes maken met Radio	166
Schuifregelaars	167
Keuzelijst	168
Tekstinvoer	169
iOS-elementen	170
<b>4.14 Animatie</b>	<b>172</b>
Zelf een animatie definiëren	172
De status van de animatie	175
Animatie beëindigen	175
Gemakkelijker animeren	176
<b>4.15 Figuren tekenen</b>	<b>178</b>
CustomPaint maken op basis van schermgrootte	178
Painter maken	179
Tekenopdrachten geven	180
Tunnels en bogen	181
Canvasanimatie	182
<b>4.16 Lijsten, eigen widgets en keys</b>	<b>185</b>
Een lijst met kaarten	185
Herhaling voorkomen	186
Eigen widgets maken	187
Eigen widgets gebruiken	188
Dismissible en keys	188
Key	189
<b>4.17 Gegevens doorgeven</b>	<b>190</b>
Inherited widget	191
updateShouldNotify	191
Inherited widget maken	192
Inherited widget gebruiken	193
updateShouldNotify gebruiken	194
Een bibliotheek importeren	194
Reageren op veranderingen met streamcontrollers	196
Een streamcontroller maken	196
Inherited widget opruimen	197
Reageren op de streamcontroller	197
Flexibele controllers	198
<b>4.18 Gegevens bewaren en futures gebruiken</b>	<b>199</b>
shared_preferences importeren	199
Stateful widget met teller	200
Gegevens opslaan	201
Gegevens lezen	202
Gegevens uit een future halen	203
FutureBuilder	203
<b>4.19 Meer widgets</b>	<b>206</b>

<b>5</b>	<b>Een complete app</b>	<b>209</b>
5.1	<b>De app in dit hoofdstuk</b>	<b>210</b>
	Functionele eisen	211
	Broncode	212
5.2	<b>Fouten opsporen en analyseren</b>	<b>213</b>
	Foutmeldingen	213
	Een app debuggen	214
5.3	<b>De basisstructuur</b>	<b>216</b>
	Een lege app	216
	Schermen	217
	Knoppenbalk	217
	Bibliotheek	219
5.4	<b>Structuur van de quiz</b>	<b>220</b>
5.5	<b>De lay-out voor het vraagscherm</b>	<b>223</b>
	Eerste rij: vraagnummer en score	224
	Tweede rij: de afbeelding	224
	Derde rij: de vraag	226
	Antwoordopties	227
	Antwoordknoppen	227
5.6	<b>Vragen</b>	<b>230</b>
	Afbeeldingen toevoegen	232
5.7	<b>Inhoud in de quiz</b>	<b>232</b>
	In quiz.dart	232
	In vraag.dart	233
	In antwoordKnop.dart	235
	Optie: flexibel aantal antwoorden met control-flow	237
5.8	<b>Interactieve antwoordknoppen</b>	<b>239</b>
	Interactiviteit detecteren	240
	Kleurovergang animeren	240
	Optie: meer animaties met dezelfde controller	243
	Optie: geluid	244
	Antwoord verwerken met een parameterfunctie	246
5.9	<b>Het uitslagscherm</b>	<b>250</b>
	De score doorgeven	250
	De uitslag	251
	Een herstartknop	252
	De quiz herstarten met een stream	252
	Optie: state van de quiz bewaren	256
5.10	<b>Optie: uitslag versturen</b>	<b>257</b>
	Mailserver	257
	De mailer importeren	258
	De mailknop	259
	Het dialoogvenster	260
	De mailfunctie	261

Smtp-server	262
Een bericht maken	262
Bericht versturen	263
Variabelen in de mailfunctie	264
De score	265
Het opgegeven e-mailadres	265
Datum en tijd	266
Variabelen in het bericht	266
Het invoerveld valideren	267
De gebruiker informeren	270
Uitslag ombouwen naar stateful	270
Variabele tekst in uitslagscherm	271
Functie in _UitslagState	272
Functie doorgeven aan MailDialog	273
Status aanpassen vanuit MailDialog	273
De mailfunctie verder verfijnen	274
<b>5.11 Optie: vragen uit een online bron</b>	<b>276</b>
Online bestanden	276
Vraag- en afbeeldingsbestanden	276
Vragen ophalen	278
Afbeeldingen ophalen	280
Start van de quiz	281
Antwoorden verbergen	283
<b>5.12 Infoscherm en beginscherm</b>	<b>284</b>
Lay-out van het infoscherm	284
Links toevoegen	286
Optie: meertaligheid	287
Welkomscherm	290
<b>6 Een app afronden en publiceren</b>	<b>293</b>
<b>6.1 Pictogram en opstartscherm</b>	<b>294</b>
Pictogram en splashscreen maken	294
Pictogram instellen	295
Splashscreens maken	297
Splashscreen voor Android	297
Android-splashscreens toevoegen	299
Launchscreen voor iOS	300
iOS-launchscreens toevoegen	300
<b>6.2 Controles en instellingen</b>	<b>302</b>
Dart Analysis	302
Pubspec.yaml	302
AndroidManifest.xml	303
build.gradle	304
iOS-instellingen in Xcode	305

	Signing and capabilities	306
	Testen	306
<b>6.3</b>	<b>Een Android-app signeren en compileren</b>	<b>307</b>
	Keystore maken	307
	Certificaat in pubspec.yaml opnemen	308
	Een appbundle maken	308
<b>6.4</b>	<b>Een app in Google Play Store plaatsen</b>	<b>310</b>
	Google Play Console en developersaccount	310
	Nieuwe app maken	310
	Winkelvermelding	310
	App-releases	312
	Contentclassificatie	314
	App-content	314
	Prijzen en distributie	314
	App indienen	315
<b>6.5</b>	<b>iOS-certificaten &amp; -identifiers</b>	<b>316</b>
	Distributiecertificaat maken	317
	Bundle-identificer maken	318
	Provisioning profile maken	319
<b>6.6</b>	<b>iOS-app aanmaken en uploaden</b>	<b>319</b>
	App aanmaken in App Store Connect	319
	iOS-build maken vanuit Android Studio	321
	Build valideren en uploaden vanuit Xcode	321
	Handmatig signeren	322
<b>6.7</b>	<b>Een iOS-app testen met TestFlight</b>	<b>323</b>
<b>6.8</b>	<b>De app weergeven in de Apple App Store</b>	<b>324</b>
	App Store information	325
	iOS app	325
	Screenshots	325
	Beschrijvingen en URL's	326
	Build	327
	App Store Icon	327
	Copyright	327
	Age Rating	327
	App Review Information	327
	De app indienen	328
<b>6.9</b>	<b>Webapps en desktopapps exporteren</b>	<b>329</b>
	App exporteren	329
	App distribueren	330
	Tot slot	331

	<b>Index</b>	<b>333</b>
--	--------------	------------



# Vooraf

## Over dit boek

Dit boek geeft u een vliegende start bij het maken van apps voor mobiele apparaten met iOS en Android en voor het web. U bouwt de apps in Flutter: een systeem van Google dat het sinds eind 2018 mogelijk maakt om echte *native* apps te bouwen voor beide besturingssystemen. 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 opbouw van dit boek

Het eerste hoofdstuk beschrijft wat Flutter is en wat u ermee kunt. Als u twijfelt of Flutter iets voor u is, lees dan vooral dat hoofdstuk. In hoofdstuk 2 leest u wat u nodig hebt om uw systeem geschikt te maken om Flutter-apps te ontwikkelen en te testen. Het hoofdstuk laat u stap voor stap zien welke programma's u nodig hebt en waar u deze kunt downloaden.

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.

## Voor wie is dit boek?

Met Flutter bouwt u native apps op basis van de programmeertaal Dart. We duiken 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 experimenteren met wijzigingen. Dat is de beste manier om de taal en het systeem te leren kennen. Als u dat niet wilt, is de code voor de hoofdstukken 4 en 5 beschikbaar als download. Deze vindt u op online op [boek.flutter.nl/codevoorbeelden.zip](http://boek.flutter.nl/codevoorbeelden.zip). U opent de voorbeeldcode als volgt:

- 1 Download het zip-bestand en pak het uit.
- 2 Kies in Android Studio in het menu **File, Open** en blader naar de locatie waar u de het bestand hebt uitgepakt.

- 3 Kies een van de hoofdmaps, `h4_widgets` of `h5_quiz`, en klik op **Open**.
- 4 Open het bestand `pubspec.yaml` en klik op de knop **Pub get**.

**h4\_widgets** bevat de code van de losse widgets die u maakt in paragraaf 4.3 tot en met 4.18. Deze widgets zijn dus verzameld 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.

**H5\_quiz** bevat de volledige code van de app die u in hoofdstuk 5 maakt.



**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 softwareontwikkelpлатform om *native* apps te bouwen voor iOS, Android en in de toekomst ook voor andere besturingssystemen. Maar wat betekent dat nu eigenlijk?

Flutter is dus een softwareontwikkelpлатform (*Software Development Kit* of SDK). Dat is een verzameling van allerlei hulpmiddelen om software te maken. Daarbij horen een programmeertaal, een omgeving om in te programmeren (de *Integrated Development Environment* of IDE), documentatie, middelen om apps te distribueren en meer. Flutter werkt met de programmeertaal Dart. Om Dart te programmeren, kunt u verschillende IDE's gebruiken, zoals Visual Studio en Android Studio. In dit boek gebruiken we de laatste, maar daarover leest u meer in het volgende hoofdstuk.

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<sup>1</sup>. 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. Er zijn ook al mogelijkheden om de eenmaal geschreven code te exporteren als webapp (PWA: *progressive web app*) of desktopapp, maar deze zijn nu (voorjaar 2020) nog experimenteel. In dit boek richten we ons dan ook vooral op mobiele apps voor Android en iOS. Om alvast een doorkijkje naar de toekomst te geven, laten we ook zien hoe u webapps en desktopapps exporteert. Daar kunt u dus mee experimenteren, maar verwacht daarbij nog niet de stabiele en snelle apps die u met Flutter voor iOS en Android maakt.

---

1 Opgvolgers van Objective-C en Java.

## 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 het is wel duidelijk dat Google serieus in Flutter investeert en grote plannen heeft. Het bedrijf werkt al jaren aan het nieuwe besturingssysteem Fuchsia, dat zowel Android als Chrome OS moet gaan vervangen. Google ontwikkelt belangrijke delen van Fuchsia in Flutter en Flutter-apps zullen hierop werken. Grote bedrijven omarmden Flutter als basis voor belangrijke apps. Naast Google zelf is de Chinese webshop Alibaba op dit moment de bekendste.

Flutter is in december 2018, na een lange bètatestfase, gestart als ontwikkelplatform voor apps voor iOS en Android (en daarmee ook voor Chrome OS). Voor deze mobiele apps is Flutter nu een



volwaardig en professioneel ontwikkelplatform. Dat geldt nog niet voor webapps en apps voor desktopcomputers. Het is al mogelijk om deze te genereren, maar de ondersteuning is nog niet optimaal. De verbetering hiervan is een belangrijke ontwikkeling in de nabije toekomst.

Niet alleen het aantal platformen groeit, maar ook de mogelijkheden van Flutter en het aantal Flutter-ontwikkelaars. Hoe snel dat gaat, weet niemand. Maar waarom wachten op de toekomst? Flutter is leuk, gratis<sup>2</sup> en u kunt er sneller en gemakkelijker een app mee bouwen dan met andere native systemen. De programmeertaal Dart lijkt sterk op andere AOT-talen en de overstap van of naar een andere taal is dan ook te overzien. Als u nu begint, hebt u aan het eind het volgende hoofdstuk een werkende standaardapp.

---

2 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 Flutter demo-app.*

**U leert in dit hoofdstuk:**

*Een systeem klaarmaken om Flutter-apps te ontwikkelen.*

*Een demo-app 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 demo-app 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, IntelliJ IDEA), dan kunt u ook daarmee werken.
- Een apparaat of simulator om te testen. De Android-simulator die bij Android Studio hoort is handig om te testen met verschillende typen apparaten. Maar het werkt fijn en snel als u in elk geval één echte Android-telefoon of -tablet hebt. Dat mag best een wat ouder, afgedankt toestel zijn, maar het is ook geen probleem als u het nog in gebruik hebt.

Hiermee kunt u apps bouwen, testen en publiceren voor Android en voor het web. De apps die u bouwt zijn ook geschikt voor iOS, maar publiceren 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 beter als u daarnaast een echte iPad of iPhone hebt.

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

## 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 Download het Flutter-pakket voor uw besturingssysteem (Windows, Mac, Linux of Chrome).
- 3 Unzip het gedownloadte pakket. Verplaats de uitgepakte map naar een locatie van waaruit u deze wilt gebruiken (bijvoorbeeld `/users/<uw naam>/flutter/` op een Mac of `C:\flutter` in Windows).
- 4 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



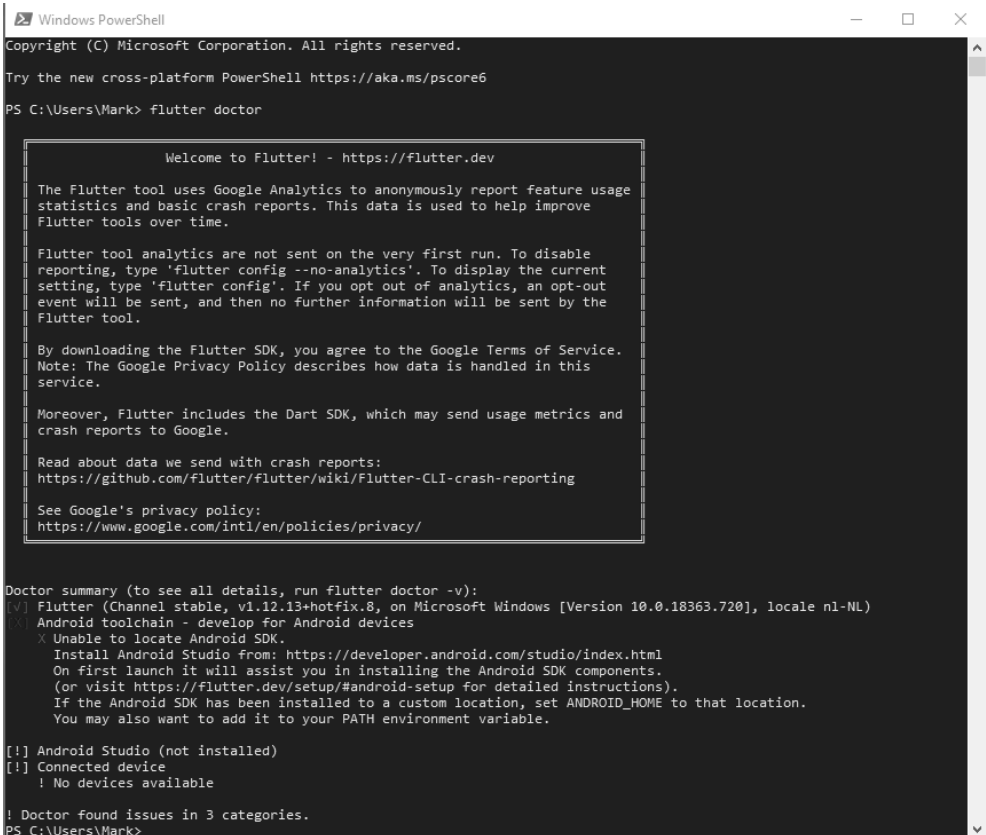
dige tekstverwerker (Nano) waarin u de PATH-instelling kunt bewerken.

- Ga met de cursor naar de onderste regel en voeg de map uit stap 3 toe met daarachter `/bin`. Bijvoorbeeld `/users/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 systeminstellingen 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 10.0.18363.720], 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 met 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 4 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.

---

## 2.3 Beginnen met Android Studio

### Android Studio installeren

Of u nu op een Windows-computer of een Mac werkt, in beide gevallen is Android Studio een zeer complete omgeving om Flutter-apps te ontwikkelen.

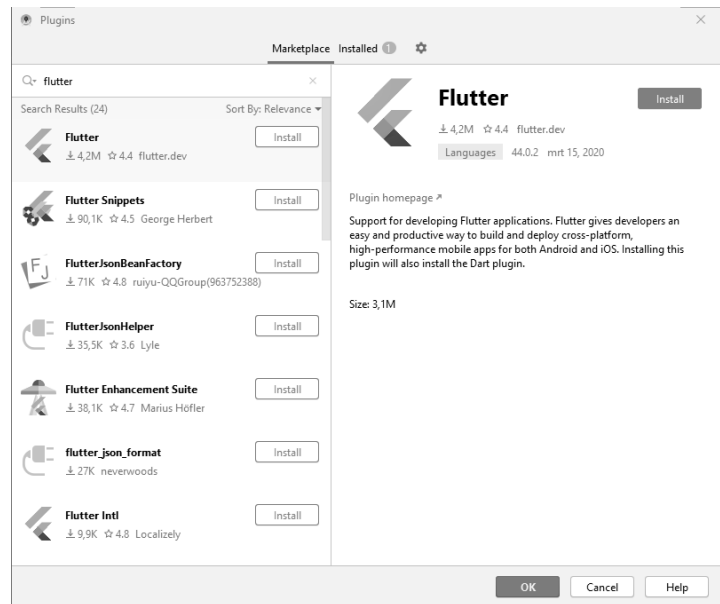
- 1 Download het programma van [developer.android.com/studio](https://developer.android.com/studio) en pak het zipbestand uit.
- 2 Dubbelklik op het .exe- (Windows) of .dmg-bestand (Mac) om het installatieproces te starten, volg de instructies op het scherm en kies de gewenste opties. Maak gebruik van de mogelijkheid om *Android Virtual Device* (een simulator-omgeving voor Android) te installeren.
- 3 Na de installatie kunt u het programma openen vanuit de programmamap op de computer (of via de automatisch gemaakte snelkoppeling).



## Het welkomstscherm en de Flutter-plug-in

Als u Android Studio opstart, verschijnt een welkomstscherm. Hier kunt u wel een nieuw Android-project starten, maar de opties voor Flutter ontbreken nog. Daarvoor moet u eerst de Flutter-plug-in installeren:

- 1 Klik in het welkomstscherm rechtsonder op **Configure** en kies de optie **Plugins**.
- 2 Zoek boven in het scherm op `flutter` en installeer de Flutter-plug-in. U krijgt de melding dat ook Dart geïnstalleerd zal worden.
- 3 Start Android Studio opnieuw op.



Afbeelding 2.2 De Flutter-plug-in installeren.

U ziet nu hetzelfde welkomstscherm met een extra optie: **Start new Flutter Project**. In dit boek werken we alleen met deze Flutter-projecten. Verder zijn er op het welkomstscherm opties om bestaande projecten te openen. Als u recente projecten hebt, ver-

schijnen deze links in een lijst en kunt u ze openen door erop te klikken. U kunt een bestaand project ook openen met de optie **Open an existing Android Studio project**. Klik hierop, blader naar de hoofdmap van het project en klik op **open**. Android Studio herkent automatisch dat het een Flutter-project is.

## 2.4 Een demo-app maken

Kies op het welkomsscherm de optie **Start a new Flutter project**. U komt de volgende instellingen tegen:

- **Project name** De naam voor de app. Deze naam wordt ook gebruikt in de stores (tenzij u deze later aanpast). De naam mag alleen kleine letters en underscores bevatten. Kies een naam die u nog niet voor een andere app gebruikte.
- **Flutter SDK-path** De locatie van de Flutter SDK (uit paragraaf 2.2). Deze hoeft u alleen bij het eerste project in te vullen.
- **Project location** De plaats waar u het Flutter-project wilt opslaan. Zorg voor voldoende schijfruimte: Flutter-projecten gebruiken honderden MB's.
- **Description** Omschrijving van het project.
- **Create project offline** Zet dit bij voorkeur aan. Hiermee neemt het project meer schijfruimte in beslag, maar gaat het compileren sneller.
- **Package name** De systeemnaam voor uw app. Bij het uploaden naar de appstores, moet deze uniek zijn. Het is daarom gebruikelijk om hier de zogenoemde *reverse domain name notation* te gebruiken: als uw domein domeinnaam.nl is, en het project heet (bij 1) mijn\_app, dan wordt de volledige appnaam: nl.domeinnaam.mijn\_app. Als u geen domeinnaam hebt, kunt u in plaats van de domeinnaam ook uw eigen naam gebruiken.
- **Use Androidx.\* artefacts** Ondersteuning voor de naamgeving van nieuwe Android-bibliotheken. Waarschijnlijk hebt u

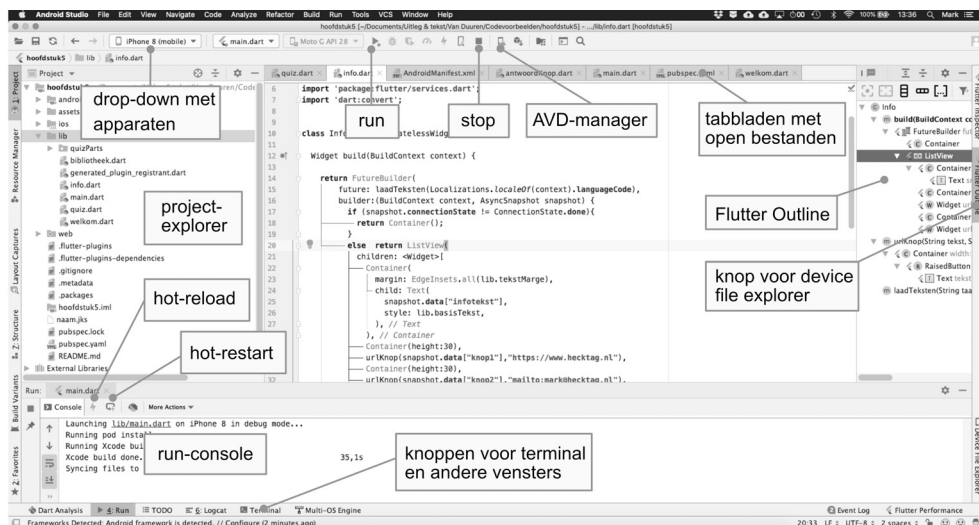
deze bibliotheken niet nodig, maar het kan geen kwaad om de optie aan te laten, voor het geval dat.

- **Include Kotlin en Swift** Zet deze opties aan als u de app wilt exporteren voor respectievelijk Android en iOS.

Dat was het. Het systeem is even bezig, maar dan hebt u ook wat: de eerste Flutter-app is aangemaakt! Alleen, we hebben nog niets om de app op uit te voeren en te kijken wat hij doet. Dat doen we verderop in dit hoofdstuk. Eerst kijken we naar Android Studio en de structuur van de app.

## 2.5 Android Studio gebruiken

Het valt buiten het bestek van dit boek om alle mogelijkheden van Android Studio te bespreken. We noemen hieronder daarom alleen de belangrijkste functies voor het ontwikkelen van Flutter-apps, die u ook in de afbeelding ziet. Enkele andere functies komen later in dit boek aan bod.



Afbeelding 2.3 Enkele belangrijke onderdelen van Android Studio.

## De Project Explorer en de structuur van de app

Aan de linkerkant van het scherm vindt u vier tabs. Verreweg de belangrijkste is de tab **1: Project**. Daarin ziet u uit welke bestanden en mappen de app bestaat. Deze projectbrowser werkt ongeveer zoals Verkenner in Windows of Finder op een Mac. De structuur die u hier ziet, is ook de werkelijke structuur op de schijf. U kunt bestanden vanaf andere locaties op de computer hier naartoe slepen of op een andere manier kopiëren. (Andersom werkt dat gek genoeg niet: u kunt geen bestanden vanuit de Project Explorer naar een andere map kopiëren. Daarvoor moet u de map waarin de Flutter-app staat openen in Finder of Verkenner.)

Open de hoofdmap (met de naam van de app) in de Project Explorer om de structuur te zien. Elke Flutter-app bestaat uit dezelfde hoofdonderdelen:

- De mappen iOS, Android en web: hier komen alle bestanden terecht die specifiek zijn voor het betreffende platform. Flutter genereert deze automatisch. Tijdens het ontwikkelen van de app hebt u deze mappen niet nodig. Zoals we in hoofdstuk 6 zullen zien, past u hier wel de laatste zaken voor publicatie aan.
- De map Lib: het hart van de app. Alle code die u schrijft en alle afbeeldingen, clips en andere bronnen komen hier terecht (al dan niet in submappen). Deze inhoud is de basis voor de versies die Flutter voor Android en iOS maakt en in de eerdergenoemde mappen plaatst. Het grootste deel van dit boek gaat over het werken met de bestanden in deze map.
- De map Test: hier kunt u zogenoemde unittests doen. Dat is vooral handig als u met meerdere mensen aan een app werkt. In dit boek zullen we tests gewoon in de app zelf uitvoeren. U kunt deze map veilig verwijderen (via het menu **Edit**, **Delete** of het contextmenu).

- Naast deze mappen ziet u enkele bestanden staan. Vooral `pubspec.yaml` is belangrijk. Hierin staan app-instellingen bewaard die zowel voor Android als iOS belangrijk zijn, zoals de naam en het versienummer. Ook gebruikt u dit bestand om Dart-bestanden, afbeeldingen en andere bronnen te importeren. In hoofdstuk 4 en 5 gaan we hiermee aan de slag.

## Vensters

Aan de onderkant van het scherm<sup>1</sup> kunt u verschillende vensters openen. U schakelt tussen de verschillende vensters met de knoppen helemaal onderaan in het scherm. De belangrijkste twee zijn:

- **Run-console** Als een app actief is op een simulator of apparaat, ziet u hier wat er gebeurt. U ziet hier bijvoorbeeld foutmeldingen of berichten die de code op het scherm plaatst. Dit venster bevat ook de knoppen **Hot reload** en **Hot restart**, waarmee u bliksemsnel, vaak binnen een seconde, wijzigingen doorvoert in een werkende app of de app opnieuw opstart.
- **Terminal** Dit scherm vervangt de Terminal of PowerShell van het besturingssysteem. U kunt hier nu bijvoorbeeld opnieuw de opdracht `flutter doctor` geven om de status van de installatie te controleren. In dit venster geeft u ook de opdrachten om de app te compileren voor Android en iOS of om een simulator te starten.

## Flutter-vensters

Aan de rechterkant van het scherm vindt u twee tabs die specifieke Flutter-hulpmiddelen openen.

---

1 Met de instellingenknop op de bovenrand van dit venster kunt u de positie wijzigen. Probeer bijvoorbeeld `move to, top left`.



- Met de Device File Explorer kunt u bestanden en mappen bekijken op het apparaat waarop de Flutter-app wordt uitgevoerd.
- De Flutter Outline is onmisbaar bij het ontwikkelen van Flutter-apps. Hier ziet u de structuur van de widgets in de app. Met de knoppen aan de bovenkant van dit venster kunt u gemakkelijk widgets uit elkaar halen of juist in elkaar plaatsen. Hoe belangrijk dat is, zal in hoofdstuk 4 blijken.

## Menu- en knoppenbalken

Net als bij andere programma's vindt u boven aan het scherm de menu- en knoppenbalken. De functies op de knoppenbalk staan ook in de menu's.

- In de menu's staan veel algemene taken, die u kent van andere programma's. Hiermee kunt u bestanden openen, opslaan, kopiëren, plakken enzovoort.
- Er zijn ook menu's voor specifieke programmeertaken. Zo is er een **Code**-menu om Dart-code te bewerken. Enkele functies daarvan komen in het volgende hoofdstuk terug.
- Belangrijk is het drop-downmenu met apparaten in de knoppenbalk. Hier kiest u op welk echt of virtueel apparaat de app start als u op **Run** klikt. Het kan zijn dat hier nu nog *<no devices>* staat. Daar gaan we verderop wat aan doen.
- Op de knoppenbalk staat ook een **Run**-knop (groen driehoekje) om een app te starten. Als de app eenmaal werkt, gebruikt u meestal de knoppen in het eerder beschreven run-venster om een app te verversen of te herstarten. Naast de **Run**-knop staat een **Stop**-knop (rood vierkantje) om de app te stoppen.
- De AVD (Android Virtual Device)-manager vindt u zowel in het menu (onder **Tools**) als op de knoppenbalk. Hiermee beheert u de virtuele Android-apparaten waarop u apps kunt testen. Dat is het onderwerp van de volgende paragraaf.