



# GRAFISCH LYCEUM

# ROTTERDAM

# ANIMATE

## >ANDROID EN ANIMATE

MEDIADEVELOPMENT | CREBO: 95313 | LEERJAAR 1

GRAFISCH LYCEUM ROTTERDAM

SAMENSTELLER: BIJLSMA



## INHOUDSOPGAVE

<b>Inleiding .....</b>	<b>2</b>
<b>Werkwijze en beoordeling.....</b>	<b>3</b>
Beoordeling: .....	3
<b>Hoofdstuk 1: AIR for Android .....</b>	<b>4</b>
Create from Template.....	11
Mijn eerste AIR-applicatie voor Android .....	13
Air for Android Settings.....	15
De Application Descriptor .....	17
Opdracht 1 .....	17
<b>Hoofdstuk 2: Opslag van gegevens .....</b>	<b>18</b>
Shared Object.....	24
Data opslaan in een tekstbestand .....	25
Opdracht 2.....	27
<b>Hoofdstuk 3: Interactie.....</b>	<b>18</b>
Button .....	18
Touch.....	18
Multitouch .....	19
Zoom .....	20
Pan .....	21
Rotatie .....	21
Accelerometer .....	21
Opdracht 3.....	23
<b>Hoofdstuk 4: Plaatsbepaling .....</b>	<b>31</b>
Geolocation .....	31
Snelheid.....	31
Google Maps .....	32
Google API .....	33
Opdracht 4.....	34
<b>Hoofdstuk 5: Animate en PHP .....</b>	<b>28</b>
Data naar een php-pagina sturen.....	28
De php-pagina .....	29
Data uit een php-pagina lezen.....	29
Opdracht 5.....	30
<b>Hoofdstuk X: Lynda.com .....</b>	<b>27</b>

## INLEIDING

Deze periode gaan jullie met Animate mobiele apps voor Android ontwikkelen.

Er zijn zowel voor- als nadelen aan deze omgeving.

Een aantal nadelen:

- Om een Animate-applicatie op een juiste manier op een mobiel apparaat te laten draaien is een aparte omgeving nodig: AIR (Adobe Integrated Runtime) . Deze is via de Google Store te downloaden en te installeren op een Android smartphone, maar kan worden ge-embed in de applicatie; deze wordt daardoor wel groter.
- Door het gebruik van AIR zal de processor (en de batterij) van de smartphone zwaarder belast kunnen worden.

Voordelen:

- De omgeving van Animate is voor jullie bekend en werkt redelijk intuïtief.
- Je kan gewoon in Actionscript 3 programmeren, met alle mogelijkheden die dit biedt.
- Het testen via de emulator werkt een stuk sneller dan in bijvoorbeeld Android Studio. Er is trouwens een mogelijkheid om de AVD's te gebruiken waarmee ook in Android Studio wordt getest.
- Adobe heeft een volledige omgeving (framework) ontwikkeld waarbinnen een applicatie kan worden ontwikkeld: Flex. Een belangrijk onderdeel daarbinnen is Adobe Animate Builder. Voor meer informatie over Animate Bulder verwijst ik jullie naar <http://www.adobe.com/products/Animate-builder.html>
- Ook vanuit Animate is het mogelijk om de applicatie te publiceren naar een .apk-bestand.
- In Animate is ook mogelijk om applicaties te maken voor IOS.

## WERKWIJZE EN BEOORDELING

Lees de theorie goed door en maak de bijbehorende oefeningen.

Op internet zijn diverse tutorials over deze lesstof te vinden. Ga zelf op zoek naar goede tutorials en maak hier gebruik van.

Je kan o.a. gebruik maken van de videotutorials op <http://www.lynda.com>. Maar ook op youtube zijn diverse videotutorials over dit onderwerp te vinden.

Sla de opdrachten op in jouw netwerkmap. Maak in deze map een nieuwe map met de naam ANIMATE.

Als je een opdracht af hebt laat je deze in de les nakijken door de docent.

### BEOORDELING:

Opdr	Omschrijving	Pnt
1	Rekenmachine	10
2	Conversie	10
3	Accelerometer	5
	EXTRA ONDERDELEN	5
4	Interactie	7
	EXTRA	3
5	Boodschappenlijstje	7
	EXTRA: Boodschappenlijstje aanpassen	3
6	Animate en PHP	10
7	Coördinaten	7
	EXTRA: Google Maps API	3
X	Uitbreiding op opdrachten; beroepshouding	10
<b>TOTAAL</b>		<b>80</b>

De module is voldoende afgerond als er minimaal 50 punten zijn behaald

## HOOFDSTUK1: ANIMATE, DE BASIS

Animate is een onderdeel van het Adobe pakket.

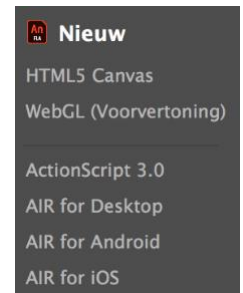
Dat betekent dat je gemakkelijk een bestand uit een ander programma (bijvoorbeeld Photoshop) kan importeren in Animate.

Om Animate op te starten klik je op het logo:

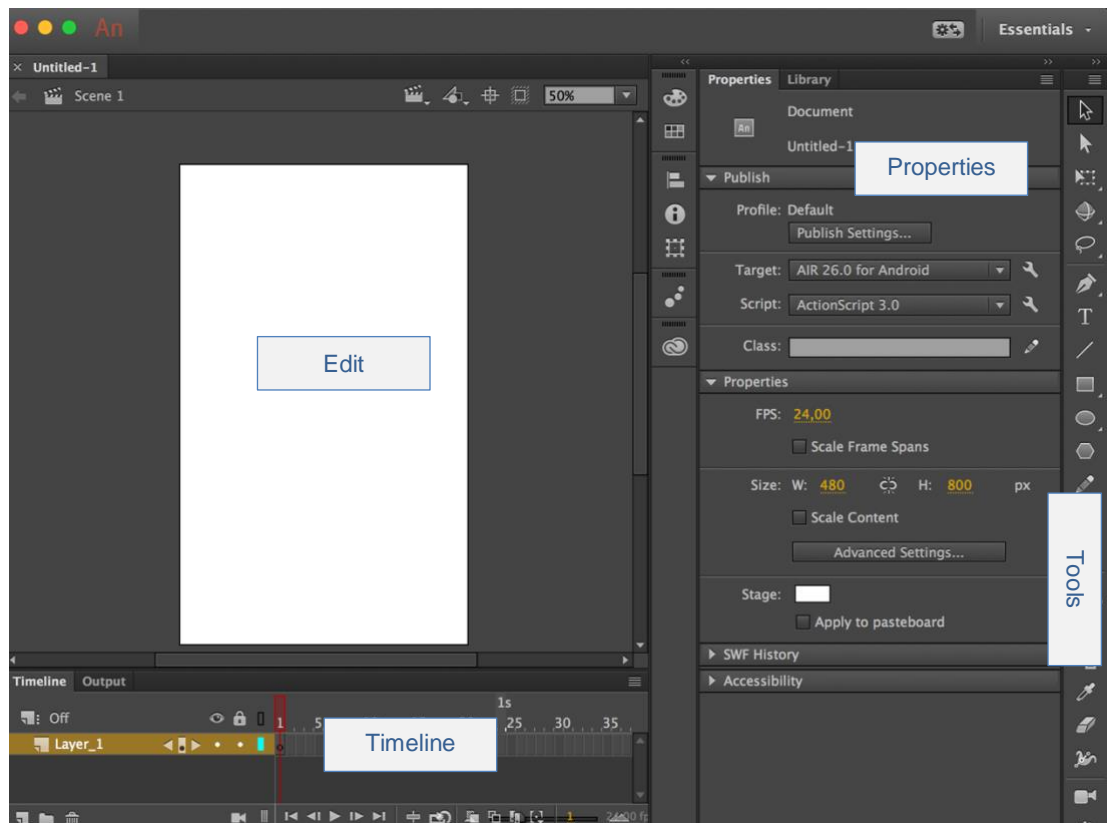
Als dit logo niet zichtbaar is dan kan je het programma zoeken via Launchpad, of je drukt de command-toets en spatiebalk tegelijk in en typt daarna "Animate".



Als je het startmenu ziet dan kies je voor "AIR for Android"  
Anders kies je "File – New – AIR for Android".



Nu wordt het werkvenster van Animate zichtbaar:



Het **Edit** venster gebruik je om je applicatie op te bouwen.

In het **Properties** venster staan de eigenschappen van je applicatie óf de eigenschappen van het element dat is geselecteerd.

Het **Tools** venster wordt gebruikt om elementen in jouw applicatie te tekenen, selecteren of aan te passen.

De **Timeline** gebruik je om eenvoudige animaties te maken.

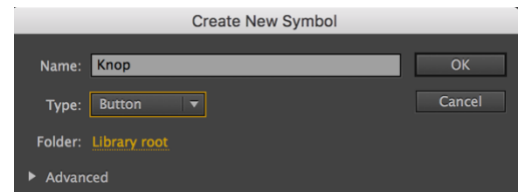
Extra uitleg over het werkvenster kan je o.a. op onderstaande webpagina vinden:

<https://helpx.adobe.com/animate/using/workflow-workspace.html>

## Oefening: button maken

Nu ga je jouw eerste applicatie maken met Adobe Animate.

- Maak een nieuwe AIR for Android applicatie:  
Kies "File – New... - AIR for Android"  
De applicatie is 480 pixels breed en 800 pixels hoog.  
Het aantal frames per seconde is 24 FPS  
Kies zelf een achtergrondkleur.
- Sla de applicatie op als "oefening\_knop fla"
- Maak een Button:
  - Kies "Insert – New Symbol..."
  - Kies als type: "Button"
  - Geef de knop een naam: "Knop"
  - Klik op "OK"

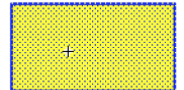


Er wordt een nieuw werkvenster (tabblad) geopend met de naam "Knop". Je ziet dat het tabblad ook een nader icoon heeft. In dit venster ontwerp je de knop:

- Kies in het Tools-venster de Rectangle-tool :
- In het Properties-venster kan je nu de eigenschappen van de knop instellen. Geef de knop een lijn-kleur en een vul-kleur. Pas eventueel de lijndikte en de afgeronde hoeken aan.
- Teken in het midden van het scherm, om het plusje ('aangrijpingspunt'), een rechthoek.



De juiste breedte en hoogte van de knop stel je in via het properties-venster. Ook de positie van de knop ten opzichte van het aangrijpingspunt:



- Kies in het Tools-venster de Selection-tool. (Zwarte pijltje)
- **Dubbeltklik** op jouw rechthoek.



LET OP: gebruik een dubbelklik om zowel de inhoud als de rand van de rechthoek te selecteren. Als je één keer klikt dan selecteer je alleen de inhoud. Je kan ook een rechthoek om jouw knop heen trekken om alles te selecteren.

- Pas nu de eigenschappen van de knop aan:
  - Breedte: 120 px
  - Hoogte: 60 px
 Positie ten opzichte van het aangrijpingspunt:
  - X: -60
  - Y: -30
 Het aangrijpingspunt zit nu precies in het midden van de knop.

Als laatste ga je een tekst op de knop plaatsen:

- Selecteer de Tekst-tool uit het tools-venster.
- Klik in de knop.



Er verschijnt nu een tekstvenster waarin je kan typen. Voordat je gaat typen pas je eerst de properties van de tekst aan.

- Kies een lettertype, grootte en kleur.
- Typ nu de tekst "KLIK" in de knop.

De tekst staat nog niet mooi in het midden

- Selecteer de tekst (met het selectiepijltje) en sleep de tekst in het midden van de knop.  
óf: Bepaal met de X- en Y-waarde de juiste positie van de tekst



De knop is klaar.

Om nu terug te gaan naar het hoofdscherm klik je op het bijbehorende tabblad: "Scene 1".

De knop staat in de Library. De Library is te vinden in het tabblad naast "Properties".

De knop staat nog niet in jouw applicatie.  
Daarom doe je het volgende:

- Sleep de knop vanuit het Library-venster naar jouw werkvenster.
- Bepaal via het Properties-venster de juiste positie.
- Geef de knop een naam. Bijvoorbeeld "knop1".

LET OP: Je kan een item uit de library vaker naar jouw werkvenster slepen. Zo kan je één knop vaker gebruiken. Een element op het werkvenster heet een **instantie** (instance). Elke instantie MOET een eigen naam krijgen zodat je ze via de code apart kan gebruiken. Het is een goede gewoonte om library-items met een hoofdletter te schrijven ("Knop") en instanties met kleine letters ("knop1"). De eigenschappen van een instantie zijn via het Properties-venster aan te passen.

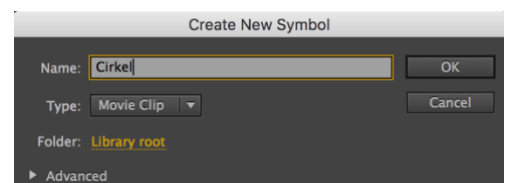
Voor meer opties met knoppen kan je kijken op de volgende pagina:

<https://helpx.adobe.com/nl/animate/using/creating-buttons.html>

## Oefening: Movieclip maken

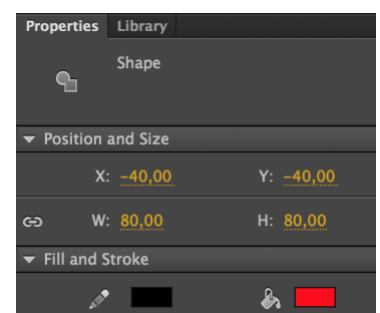
Een movieclip is een onderdeel binnen een applicatie die (meestal) wordt gebruikt in animaties. Het maken van een movieclip gaat op dezelfde manier als een button. Alleen zijn de eigenschappen en mogelijkheden iets anders.

- Kies "Insert – New Symbol..."  
Zet het type op "Movie Clip"  
Geef het symbool een naam: "Cirkel".  
(Let op de hoofdletter)  
Klik op "OK"



Er verschijnt een nieuw werkvenster (tabblad) waarin je de movieclip kan tekenen.

- Kies uit het Tools-venster de Oval tool.
- In het Properties venster kies je een lijn- en vulkleur.
- Teken nu een cirkel op het scherm.  
Om een perfecte cirkel te krijgen hou je de shift-toets ingedrukt tijdens het tekenen.
- Selecteer de hele(!) cirkel met de Selection-tool.  
Pas de grootte en positie t.o.v. het aangrijpingspunt aan:



Als je klaar bent ga je terug naar Scene 1.

In de Library staan nu het items Cirkel.

- Sleep een Cirkel naar jouw werkvenster.  
Geef deze instantie een naam: "rondje1".  
Pas de properties aan, zoals X- en Y-positie.

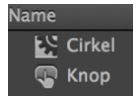
Voor meer uitleg over movieclips (en symbols), zie:

<https://helpx.adobe.com/nl/animate/using/symbols.html>



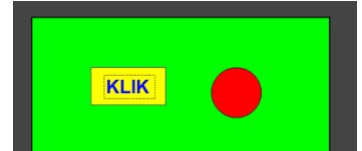
## Oefening: met een knop een movieclip bewegen

Om deze oefening te kunnen maken heb je een button (Knop) en movieclip (Cirkel) nodig. Zorg dat deze in jouw Library staan.



Het bewegen van de movieclip wordt gedaan via de code van Animate:  
**Actionscript.**

- Sleep de knop Knop vanuit de Library naar het werkvenster; bovenin het scherm. Noem de Knop "knop1".
- Sleep de movieclip Cirkel naar jouw werkvenster; bovenin, naast de knop. Noem deze "rondje1".

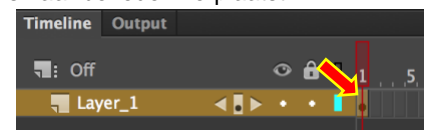


Nu moet er aan de knop een stuk code (Action script) gekoppeld worden; deze zorgt voor het volgende:

Als er op de knop wordt geklikt dan wordt het rondje 5 pixels naar beneden verplaatst.

De Actionscript plaatsen we in Frame 1 van de Timeline

- Klik met je rechter muisknop op in Frame 1 van de Timeline.
- In het popup-menu kies je "Actions".
- Typ de volgende code in het Actions-venster:



```

1 //importeer de muis-events
2 import flash.events.MouseEvent;
3
4 //Koppel de functie 'beweeg' aan de klik van knop1
5 knop1.addEventListener(MouseEvent.CLICK, beweeg);
6
7 //de functie 'beweeg'
8 function beweeg (event:MouseEvent):void
9 {
10     //beweeg het rondje 5 pixels naar beneden
11     rondje1.y = rondje1.y + 5;
12 }
  
```

Nu ga je applicatie **testen**.

Omdat je een Android-applicatie hebt gemaakt, test je jouw applicatie op een Android-device. Animate heeft een virtueel device waarmee je kan testen, dus je hoeft geen Android-telefoon aan te sluiten op jouw computer.

- Kies in het menu van Animate:  
"Debug – Debug Movie – In AIR Debug Launcher (Mobile)"  
of: "Control – Test Movie – In AIR Debug Launcher (Mobile)"

De applicatie wordt nu een virtueel device opgestart.

Daarnaast wordt ook een simulator opgestart. Daar komen we later op terug.

- Klik in de applicatie op de knop.  
Als het goed is verschuift het rondje 5 pixels naar beneden.

Probeer nu zelf de code aan te passen.

Bijvoorbeeld:

- Beweeg met de knop het rondje 10 pixels naar links.
- Beweeg met de knop het rondje 15 pixels omhoog én 5 pixels naar rechts.

Voeg zelf nog drie knoppen toe.

Zorg dat elke knop het rondje een andere richting op laat bewegen.

## Oefening: Tekstveld

Animate kent 3 soorten tekstvelden:

- **Static Text:** De tekst in dit tekstveld is niet aan te passen. Deze optie wordt vooral gebruikt voor labels en teksten op knoppen e.d.
- **Dynamic Text:** De tekst in dit tekstveld is aan te passen via de code (Actionscript).
- **Input Text:** De tekst in dit tekstveld is aan te passen door de gebruiker.

In dit voorbeeld ga je alle drie de tekstvelden gebruiken.

- Maak een nieuwe Android-applicatie: "File – New... - AIR for Android"  
Sla de applicatie op als "oefening\_tekst fla".
- Maak een knop: "Insert – New Symbol..."  
Geef de knop een naam: "Knop".  
Bepaal zelf het ontwerp van de knop.  
Ga daarna terug naar Scene 1. Als het goed is, staat Knop nu in de Library.

Een tekstveld is geen symbol. Je kan deze dus direct in jouw ontwerpvenster (Scene 1) maken.

- Selecteer de Text-tool uit het Tools-venster.
- Zet in het Properties het type op "Static Text".  
(Dit kan eventueel ook nog later).
- Kies een tekstkleur (en fonttype en grootte).
- Sleep een rechthoek linksboven in jouw applicatie.  
Zet hierin de tekst: "Voer een getal in:".  
Pas de grootte zo aan dat alle tekst op 1 regel staat.
- Klik buiten het tekstveld.  
De witte achtergrond verdwijnt nu.



- Selecteer opnieuw de Text-tool.
- Kies nu het type "Input Text".
- Sleep een rechthoek onder de vorige tekst.

Zorg dat deze geselecteerd blijft!

Je gaat deze nu aanpassen via het Properties-venster.

- Geef het veld een naam: "invoerVeld".
- Kies een tekstkleur, -type en -grootte
- Klik op het symbool "show border around tekst".  
Nu blijft het tekstveld zichtbaar als een witte rechthoek.



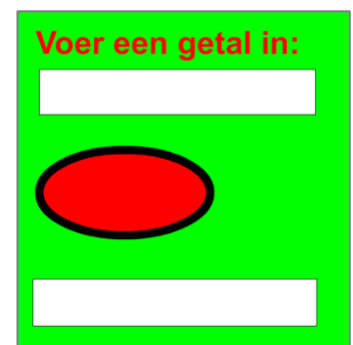
- Plaats nu de Knop op het scherm
- Geef deze een naam: "berekenKnop".
- Selecteer opnieuw de Text-tool.
- Kies nu het type "Dynamic Text".
- Sleep een rechthoek onder de vorige tekst.
- Geef het veld een naam: "uitvoerVeld".
- Zorg ervoor dat de achtergrond van het veld wit blijft.

De applicatie zou er nu als volgt kunnen uitzien:

De applicatie moet het volgende doen:

- De gebruiker voert in het eerste veld een getal in.
- Daarna druk hij op de knop.
- Via Actionscript wordt het eerste veld uitgelezen.
- Met dit getal wordt een berekening gedaan.
- Het antwoord komt in het tweede tekstveld te staan.

Je gaat nu de Actionscript voor deze handelingen schrijven.



Rechtsklik in Frame 1 van de Timeline. Klik op "Actions". Typ de volgende code over:

```

1 //importeer de muis-events
2 import flash.events.MouseEvent;
3
4 //Koppel de functie 'bereken' aan de klik van de berekenknop
5 berekenKnop.addEventListener(MouseEvent.CLICK, bereken);
6
7 //de functie 'bereken'
8 function bereken (event:MouseEvent):void
9 {
10     //lees het invoerVeld uit en plaats de waarde in de variabele 'tekst'
11     //LET OP: uit het invoerVeld komt tekst, dus de variabele is een String
12     var tekst:String = invoerVeld.text;
13     //Zet de tekst om in een integer; deze wordt bewaard in de variabele 'getal'
14     //getal is een integer (= een heel getal)
15     var getal:int = int(tekst);
16     //maak de berekening; zet het antwoord in de (int-)variabele 'antwoord'
17     var antwoord:int = getal * getal;
18     //Zet het getal om in tekst; bewaar deze in Stringvariabele 'antwoordTekst'
19     var antwoordTekst:String = antwoord.toString();
20     //Zet de antwoordTekst in het uitvoerVeld
21     uitvoerVeld.text = "Antwoord = " + antwoordTekst;
22 }

```

Je ziet dat Actionscript verschillende soorten variabelen ('datatypes') kent.

Van een goede programmeur wordt verwacht dat je bij een variabele altijd aangeeft wat voor type het is.

Actionscript kent o.a.:

- String: tekst
- Number: getal (met of zonder komma)
- int: heel getal (zonder komma)
- uint: heel getal, groter dan 0
- Boolean: kan 2 waarden hebben: 'true' of 'false'

Zoals je ziet, moet je een variabele (soms) eerst converteren voordat je deze kan gebruiken.

Voor berekeningen gebruik je int of Number, in een tekstveld plaats je een String.

Er zijn verschillende manieren om te converteren. Een voorbeeld vind je op:

<http://www.trainingtutorials101.com/2011/01/actionscript-3-convert-strings-to.html>

In het uitvoerVeld wordt de antwoordtekst gekoppeld aan een eigen stukje tekst met een plusteken. Je kan dat stukje tekst natuurlijk ook weglaten.

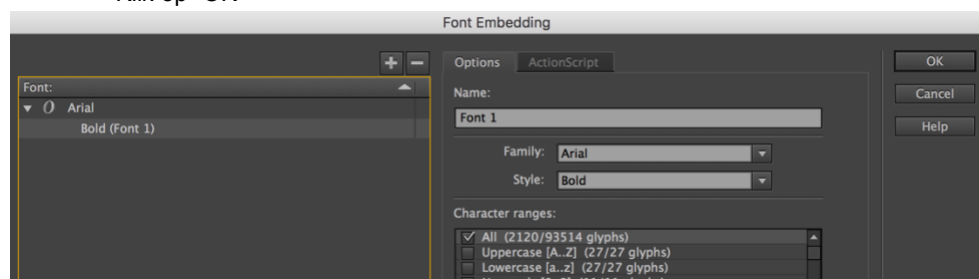
- Sla de applicatie op en test deze in de AIR Debug Launcher (Mobile)

**LET OP:** Sommige fonts werken niet goed tijdens het testen!

Sommige karakters kan je dan niet gebruiken of worden niet zichtbaar.

Dan moet je eerst het font **embedden**:

- Klik op het tekstveld.
- Klik in het properties venster, onder 'character' op de knop "embed"
- Klik links op het type ("Bold") en selecteer welke karakters je wilt embedden ("All")
- Klik op "OK"



Oefen verder met deze applicatie:

- Wat gebeurt er als je een kommagetal invoert?
- Zorg dat de berekening ook werkt met kommagetallen.
- Maak een andere berekening.

Meer uitleg over een textbox in Animate vind je in de volgende video:

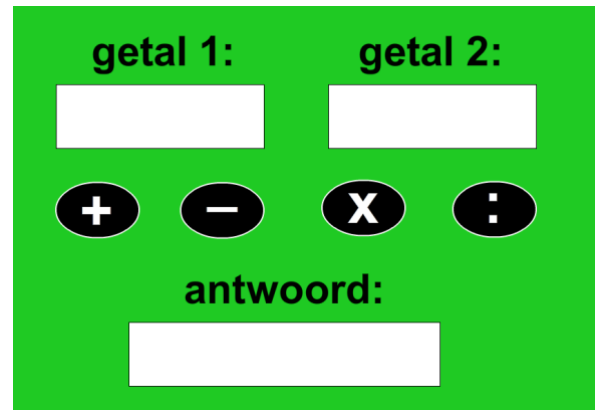
<https://www.youtube.com/watch?v=hx1VrJWPzOc>

## OPDRACHT 1: De rekenmachine

Maak een rekenmachine die de functies 'plus', 'min', 'keer' en 'gedeeld door' kan uitvoeren. De interface mag simpel zijn:

- Maak twee invoervelden voor getal1 en getal2.
- Maak 4 knoppen: 'plus', 'min', 'keer' en 'gedeeld door'
- Maak een antwoordveld waar de uitkomst wordt getoond.

Bijvoorbeeld:



Zorg dat aan elke knop een event wordt verbonden (MouseEvent.CLICK).

Als op de 'plus' wordt geklikt worden de twee velden 'getal1' en 'getal2' uitgelezen. De getallen worden opgeteld en het antwoord wordt getoond in het antwoordveld.

Voor de drie andere knoppen maak je een soortgelijke functie.

**LET OP:** de applicatie moet op een telefoon goed te gebruiken zijn. Maak daarom grote tekstvelden en knoppen!

## HOOFDSTUK 2: AIR FOR ANDROID

Animate heeft standaard het framework AIR (Adobe Integrated Runtime) tot zijn beschikking. Als je met Animate een applicatie voor een Android device wilt maken, kan je direct aan de slag. Zorg er wel voor dat je netjes en gestructureerd blijft werken. Het is bijvoorbeeld de gewoonte om alle Actionscript die geschreven wordt in een aparte, externe, class te bewaren.

Bij het creëren van een android-applicatie ontstaan er drie bestanden die van belang zijn:

- Het .fla bestand, dat automatisch door Animate wordt aangemaakt
- Het .as bestand, waarin alle actionscript wordt opgeslagen
- Het .xml bestand, waarin diverse instellingen van de app, waaronder het Android manifest, worden beschreven.

Hier komen we later op terug.

## CREATE FROM TEMPLATE

Als Animate is opgestart krijg je de keuze om een nieuw Animate document te starten. Dit kan een leeg document zijn, maar je kan ook beginnen met een template. Als introductie beginnen we met een template.

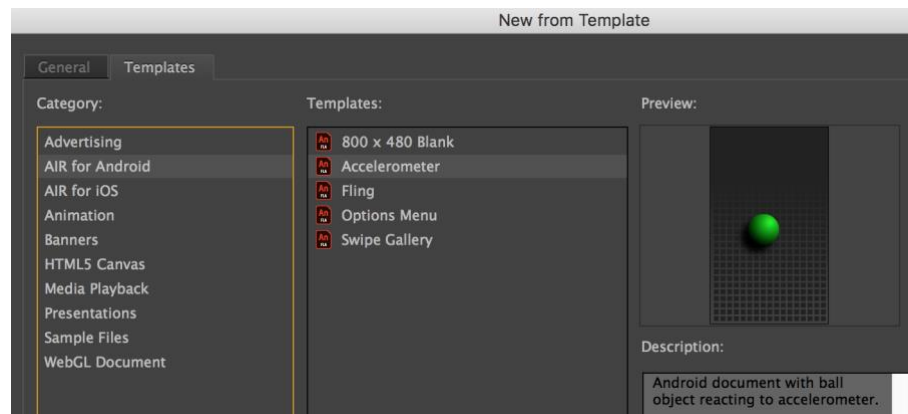
## Oefening

Kies in Animate *File – New....*

In het scherm dat verschijnt kies je het tabblad “Templates”.

Daarna kies je “AIR for Android”.

Je ziet nu een aantal mogelijke templates, met een voorbeeldafbeelding en een beschrijving:



Kies bijvoorbeeld *Accelerometer* en klik op *OK*.

De template wordt in Animate geladen.  
Sla het bestand op als (bijv.) "accel.flu".

In het Properties-venster zie je de instellingen :

De grootte van de stage is standaard ingesteld op 480 x 800; dit is een standaard verhouding die gebruikt kan worden voor de meeste telefoons. Het aantal frames per seconde staat op 60. Dit is redelijk hoog, maar in dit geval begrijpelijk.

De AIR-versie voor Android is 26.0.

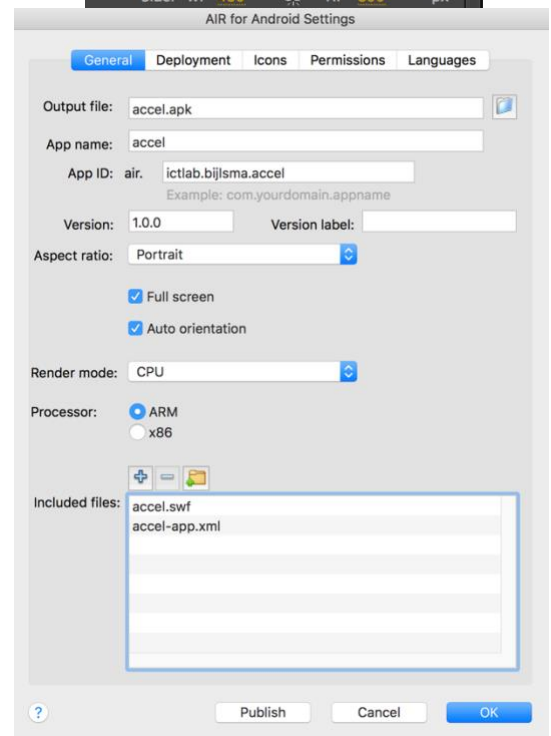
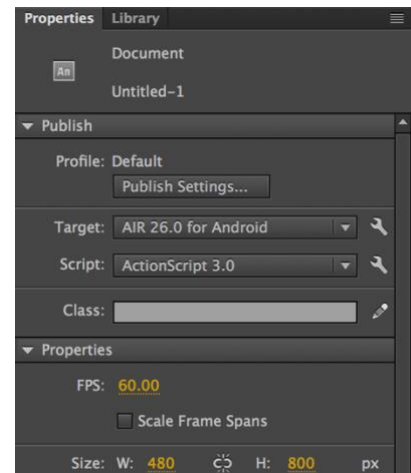
Klik op de 'steeksleutel' (*instellingen*) achter *Target*.

Hier kan je op diverse tabbladen diverse instellingen van de applicatie aanpassen voordat deze wordt gepubliceerd.

Geef het uitvoerbestand een duidelijk naam, eindigend op *.apk*. (android package)  
De *App ID* wordt gebruikt wanneer de app wordt gepubliceerd in de Android Market. Deze zal dan uniek moeten zijn.  
Ook het versienummer is voor de Market van belang. Als een verbeterde app geen hoger versienummer heeft, zal deze niet als een nieuwe versie worden herkend.

De overige instellingen van dit tabblad wijzen (hopelijk) voor zichzelf.

De andere tabbladen worden later behandeld.



We gaan de applicatie openen in de emulator.

Je kan hierbij kiezen voor *Test* of *Debug*.

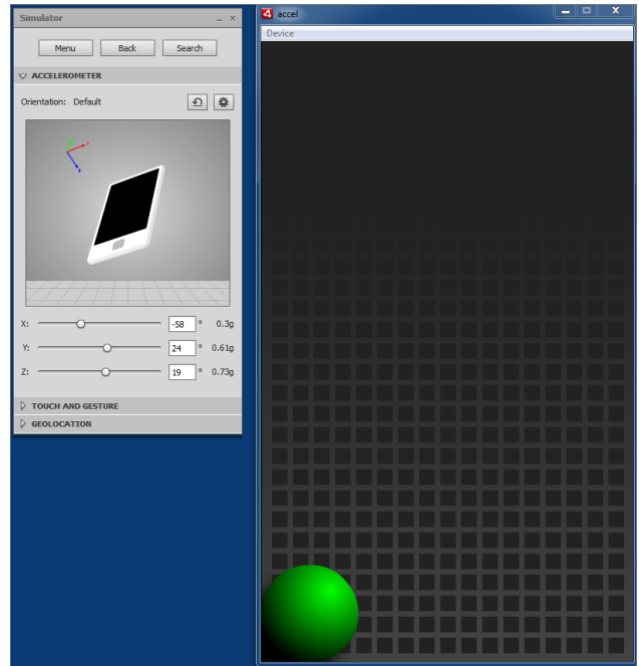
Kies :

- *Control – Test Movie – in Air Debug Launcher (Mobile)* of:
- *Debug – Debug Movie – in Air Debug Launcher (Mobile)*

De debug-versie heeft iets meer mogelijkheden, maar deze worden nu niet gebruikt.

Je kan nu onder het tabblad *Accelerometer* de emulator kantelen waardoor de bal op het scherm gaat bewegen.

De tabbladen *Touch and Gesture* en *Geolocation* hebben hier geen invloed. Ook die komen later aan bod.

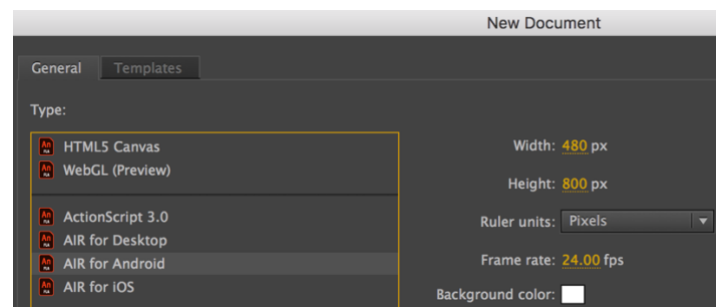


## MIJN EERSTE AIR-APPLICATIE VOOR ANDROID

Nu ga je zelf een eenvoudige app maken. Door eenvoudig gebruik te maken van de mogelijkheden in Animate heb je snel resultaat.

### Oefening

Kies *File – New .....* In het tabblad *Templates* kies je voor *AIR for Android*. Laat de instellingen voorlopig zo staan.



Sla het bestand op onder de naam *oefening1 fla*.

Plaats nu een button en textfield op de stage en geef deze een instantie-naam. Het textfield is dynamisch ('Dynamic Text').

Maak nu de actionscript-code die er voor zorgt dat, wanneer er op de knop wordt geklikt, de tekst "Hallo Wereld" in het tekstveld verschijnt.

Bijvoorbeeld:

```

1 import flash.events.MouseEvent;
2
3 knop.addEventListener(MouseEvent.CLICK, onClick);
4
5 function onClick(mE:MouseEvent):void
6 {
7     tekstveld.text = "Hallo, Wereld";
8 }

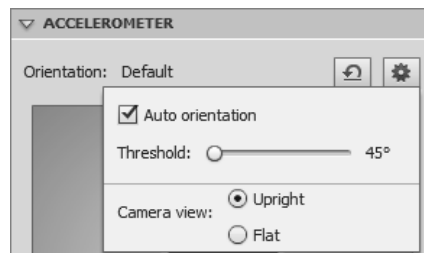
```

Sla het bestand op en test het.

Als het goed is verschijnt de tekst als je op de knop klikt.

Wanneer je de virtuele device (emulator) draait om de z-as, zodat deze in landscape positie staat, zie je dat de elementen wordt verplaatst en verkleind.

LET OP: Om de emulator mee te laten draaien met de Simulator, moet je eerst 'Auto Orientation' aanvinken bij de instellingen van de Accelerometer:



We gaan er voor zorgen dat de grootte en plaats, voor zover mogelijk, gelijk blijft.

Om de grootte van de beeldschermelementen niet te laten veranderen bij een andere schermgrootte of -indeling, gebruiken we:

```
stage.scaleMode = StageScaleMode.NO_SCALE;
```

Het is lastiger om de beeldschermelementen een vaste plaats op het scherm te geven als de oriëntatie verandert. Onderstaande code helpt hierbij:

```

stage.align = StageAlign.TOP_LEFT;
stage.addEventListener(Event.RESIZE, onChange);

function onChange(event:Event):void
{
    knop.x = Math.round((stage.stageWidth-knop.width)/2);
    knop.y = Math.round((stage.stageHeight-knop.height)/2);
    tekstveld.x = knop.x - (tekstveld.width/2);
    tekstveld.y = knop.y + 80;
}

```

Uitleg van de code:

- Allereerst worden alle elementen uitgelijnd vanaf linksboven.
- Daarna wordt de functie "onChange" aangeroepen als de grootte van het beeldscherm verandert.  
Je kan ook controleren of de oriëntatie verandert, maar deze handler houdt ook rekening met verschillende schermgroottes.
- In de bijbehorende functie wordt de knop in het midden van het scherm geplaatst. Controleer zelf de werking van deze code.
- Het tekstveld wordt 80 pixels onder de knop geplaatst.

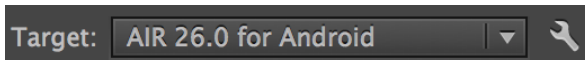
Voeg bovenstaande codes toe aan jouw actionscript en test de werking.

Pas eventueel de plaatsing van de knop en het tekstveld aan door de waarden aan te passen.



## AIR FOR ANDROID SETTINGS

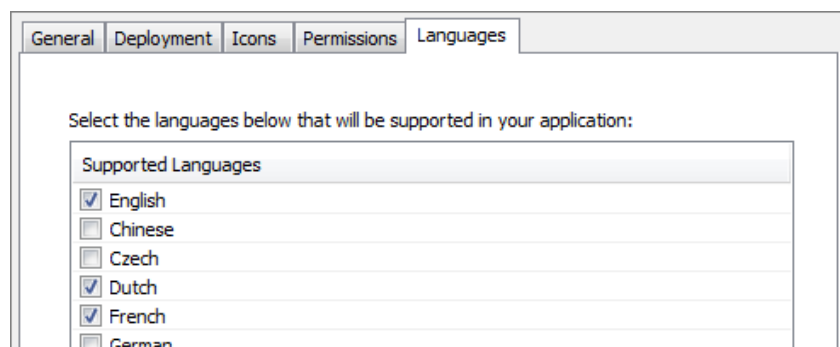
Als je in Animate naar de instellingen van AIR gaat, moet je op de 'steeksleutel' klikken.



Je ziet dan verschillende tabbladen. De eerste, *General*, hebben we al kort besproken. Ook de andere zullen we even doorlopen (van achteren naar voren):

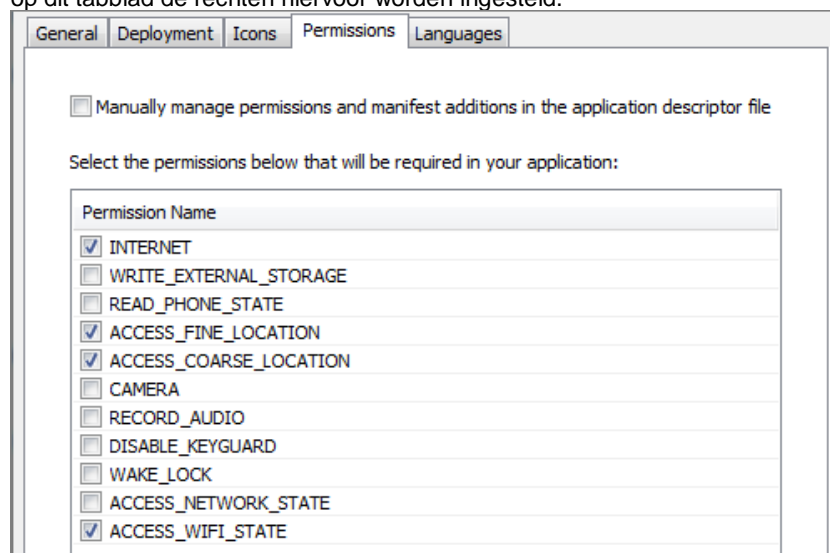
### Languages

In het tabblad *Languages* kan je aangeven welke talen ondersteund moeten worden.



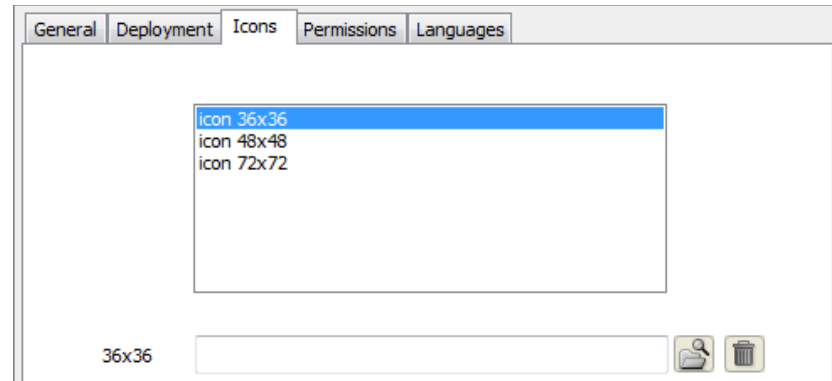
### Permissions

Als je wilt dat de applicatie toegang heeft tot bepaalde onderdelen van de mobiele device, kan op dit tabblad de rechten hiervoor worden ingesteld.



### Icons

Ja kan aan de applicatie een eigen icon toevoegen. Dit zal meestal een png zijn. Afhankelijk van de device kan de icon andere afmetingen hebben.



### Deployment

Dit tabblad is van belang als je jouw applicatie wilt publiceren.

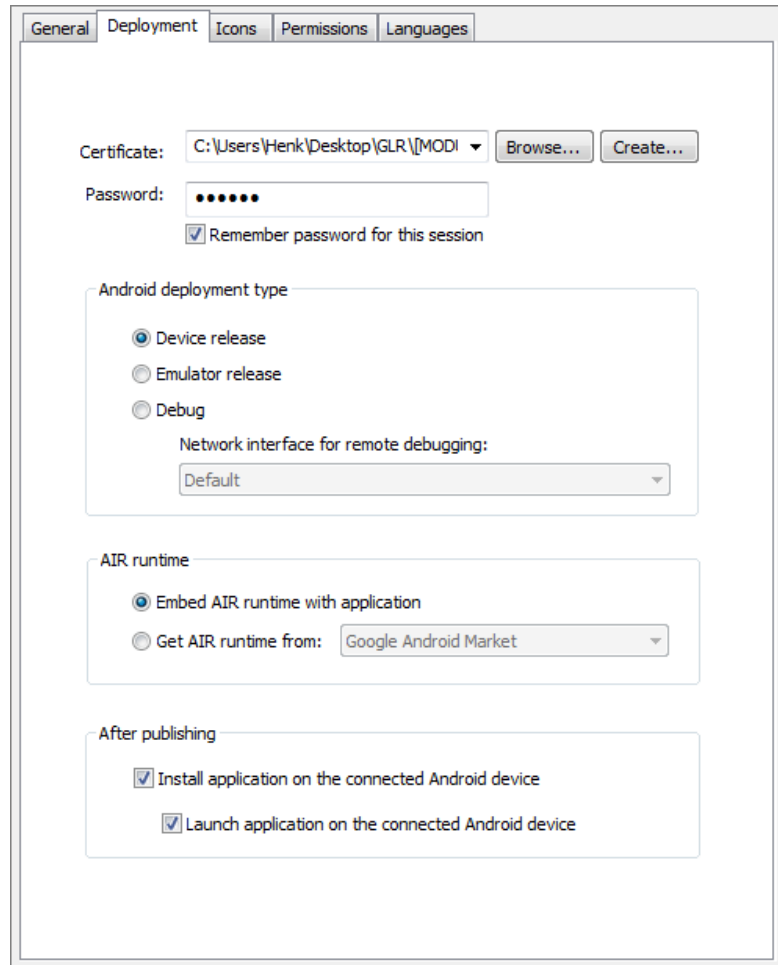
Allereerst zal je een certificaat moeten toevoegen aan jouw applicatie, voordat je deze aan de Android Market kan toevoegen. Je kan deze zelf maken als je op *Create* drukt.

Daarna geef je aan waarvoor je deze applicatie maakt. Dit zal meestal voor een device zijn...

Kies je voor *Emulator*, dan kan je de applicatie publiceren ('testen') op een AVD die ook wordt gebruikt bij Android Studio.

De applicaties die gemaakt worden in Animate draaien op AIR. Deze moet worden toegevoegd, of apart worden geïnstalleerd op de device. Als je AIR toevoegt aan de applicatie wordt deze aanzienlijk groter.

Als je een device via USB hebt aangesloten kan je de applicatie direct installeren op deze device.



## DE APPLICATION DESCRIPTOR

Alle instellingen die in de vorige paragraaf zijn doorgevoerd, via *AIR for Android* settings, worden opgeslagen in een *Application Descriptor File*. Dit is een xml-bestand en staat in dezelfde map als het fla-bestand.

Via *File – Open* kan je dit xml-bestand openen in Animate. Als je het doorleest zal je diverse instellingen kunnen terugvinden.

Een belangrijk gedeelte in dit bestand is het gedeelte onder `<android>`: de `<manifestAdditions>`. Bij het compileren naar een .apk-bestand worden de diverse instellingen binnen dit manifest meegecompileerd.

Je kan eventueel handmatig een aantal instellingen aan de Descriptor toevoegen of wijzigen.

## Opdracht 2

Maak een applicatie (“opdracht2”) voor Android die een conversie uitvoert.

Bijvoorbeeld van euro's naar dollars of van celsius naar fahrenheit.

Zorg dat je beide kanten op kan converteren.

Plaats hiervoor op de stage twee tekstvelden en een knop. Zorg voor een gebruikersvriendelijke applicatie. Test de applicatie. Als deze goed werkt, publiceer het geheel dan naar een apk-bestand.

Zorg dat de app zowel in portrait- als in landscape-stand goed te lezen en te gebruiken is.

## HOOFDSTUK 3: INTERACTIE

Op diverse manieren kan je het device laten reageren op de handelingen van de gebruiker. Aangezien we te maken hebben met een touchscreen is het logisch dat we naar de diverse mogelijkheden van het aanraken van het scherm zullen bekijken: touch, multitouch, zoom, rotate en swipe. Als laatste kijken we ook nog naar de accelerometer. De emulator van Animate heeft de mogelijkheden om bovenstaande handelingen te testen.

### BUTTON

Je kan via Animate de verschillende 'states' van een button instellen. Als je in Animate een button maakt en hierop dubbelklikt, kan je in de tijdlijn de verschillende states (*up*, *over*, *down* en *hit*) een andere opmaak geven. Je zal begrijpen dat het weinig zin heeft om de rollover een andere opmaak te geven. Je kan je het beste beperken tot *mouseUp* en *mouseDown*. In plaats van de opmaak op deze manier mee te geven, kan dit ook via Actionscript. Hieronder staat een voorbeeld waarbij de button een aparte opmaak voor *up* en *down* krijgt:

```
import flash.geom.ColorTransform;

knop_btn.addEventListener(MouseEvent.CLICK, mousedown);
knop_btn.addEventListener(MouseEvent.CLICK, mouseup);

function mousedown(evt:MouseEvent):void
{
    var nieuw:ColorTransform = new ColorTransform();
    nieuw.color = 0x00FF0099;
    knop_btn.transform.colorTransform = nieuw;
}

function mouseup(evt:MouseEvent):void
{
    var nieuw:ColorTransform = new ColorTransform();
    nieuw.color = 0x00FF0000;
    knop_btn.transform.colorTransform = nieuw;
}
```

### TOUCH

Speciaal voor de schermaanraking heeft Animate een speciale class ontwikkeld: *Animate.Events.TouchEvent*.

Afhankelijk van de bewegingen van de gebruiker kan je hiermee verschillende events aanroepen. Je kan bijvoorbeeld controleren of de gebruiker het scherm aanraakt (*TOUCH\_BEGIN*), over het scherm beweegt (*TOUCH\_MOVE*) of het scherm loslaat (*TOUCH\_END*).

Bovenstaande interactie wordt in onderstaand voorbeeld toegepast. In dit voorbeeld kan door de gebruiker een lijn worden getekend op het scherm.

Allereerst wordt de *inputmode* op *touch* gezet.

Daarna maken we voor de drie events (aanraken, bewegen, loslaten) een *eventlistener*, gekoppeld aan de stage:

```
Multitouch.inputMode = MultitouchInputMode.TOUCH_POINT;

this.stage.addEventListener(TouchEvent.TOUCH_BEGIN, begin);
this.stage.addEventListener(TouchEvent.TOUCH_MOVE, beweeg);
this.stage.addEventListener(TouchEvent.TOUCH_END, einde);
```

Daarna worden de verschillende functies geschreven:

```
function begin(event:TouchEvent):void
{
    this.graphics.lineStyle(5,0x0000FF);
    this.graphics.moveTo(event.stageX, event.stageY);
}

function beweeg(event:TouchEvent):void
{
    this.graphics.lineTo(event.stageX, event.stageY);
}

function einde(event:TouchEvent):void
{
    this.graphics.clear();
}
```

In de functie 'begin' wordt de lijndikte en kleur ingesteld, waarna de cursor op de juiste plek wordt gezet.

Elke keer als de functie 'beweeg' wordt aangeroepen wordt er een lijn getrokken naar het punt wat op dat moment wordt aangeraakt.

De functie 'einde' verwijdert de lijn van de stage. Als je deze tijdelijk uitzet kan je meerdere lijnen op de stage tekenen.

## MULTITOUCH

Het controleren op meerdere aanrakingen tegelijkertijd is niet veel moeilijker dan bij één aanraking.

Voor elke aanraking wordt er een nieuw ID aangemaakt. Bij meerder aanrakingen zullen er dus meerdere ID's worden aangemaakt. Let wel op dat, wanneer een vinger wordt opgetild en opnieuw op het scherm wordt gezet, een nieuw ID wordt aangemaakt. Om bij te houden welke ID's actief zijn wordt gebruik gemaakt van een array.

Hou wel rekening met het maximaal aantal aanraakpunten van het device. Je kan het maximaal aantal aanraakpunten instellen door gebruik te maken van de eigenschap *Multitouch.maxTouchPoints*.

Onderstaand voorbeeld is lastig te testen op de emulator, aangezien je hier niet vijf aanrakingspunten kan simuleren. Je kan daarom het beste dit voorbeeld publiceren en installeren op een echt device.

Met onderstaand voorbeeld is het mogelijk om met diverse vingers tegelijk te tekenen, in diverse kleuren. We gaan er vanuit dat er maximaal met vijf vingers tegelijk wordt getekend, dus we maken een array van vijf kleuren. Daarnaast houden we bij hoeveel vingers er tegelijkertijd het scherm aanraken. Het tekenen gebeurt via een *Sprite*. Er kunnen meerdere lijnen getekend worden, dus ook de Sprites worden opgeslagen in een array.

```
var kleuren:Array = [0xFF0000, 0x00FF00, 0x0000FF, 0xFFFF00, 0x00FFFF];
var sprites:Array = new Array();
var touchCount:int = 0;
```

Daarna worden er weer drie eventlisteners gemaakt om te controleren of het scherm wordt aangeraakt, of er wordt bewogen of losgelaten:

```
this.stage.addEventListener(TouchEvent.TOUCH_BEGIN, begin);
this.stage.addEventListener(TouchEvent.TOUCH_MOVE, beweeg);
this.stage.addEventListener(TouchEvent.TOUCH_END, einde);
```

Als het scherm wordt aangeraakt, wordt de functie 'begin' aangeroepen:

```
function begin(event:TouchEvent):void
{
    var mc:Sprite = new Sprite();
    addChild(mc);
    mc.graphics.lineStyle(5, kleuren[touchCount]);
    mc.graphics.moveTo(event.stageX, event.stageY);
    sprites[event.touchPointID] = mc;
    touchCount++;
}
```

Hierin gebeurt het volgende:

- Er wordt een nieuwe sprite aangemaakt, genaamd 'mc'.
- Deze wordt op de stage geplaatst.
- Via deze sprite wordt de dikte en kleur van de lijn ingesteld. De kleur wordt uit de array 'kleuren' gehaald en is afhankelijk van het aantal aanrakingen op dat moment.
- De sprite wordt naar het beginpunt (aanrakingspunt) verplaatst.
- De sprite wordt in de array 'sprites' geplaatst.
- Het aantal aanrakingen wordt verhoogd met 1.

In de functie 'beweeg' moet de juiste sprite bewegen. Door het juiste *touchPointID* te gebruiken, kunnen we de juiste sprite uit de array halen.

```
function beweeg(event:TouchEvent):void
{
    var mc:Sprite = sprites[event.touchPointID] as Sprite;
    mc.graphics.lineTo(event.stageX, event.stageY);
}
```

In de functie 'einde' gebeurt het volgende:

```
function einde(event:TouchEvent):void
{
    var mc:Sprite = sprites[event.touchPointID] as Sprite;
    this.removeChild(mc);
    delete sprites[event.touchPointID] ;
    touchCount--;
}
```

- De juiste sprite wordt uit de array gelezen.
- Deze wordt verwijderd van de stage.
- De sprite wordt verwijderd uit de array.
- Het aantal scherm-aanrakingen wordt met 1 verlaagd.

Natuurlijk kan je er ook voor kiezen om de sprites (lijnen) te laten staan.....

## ZOOM

Het in- en uitzoomen op een device gebeurt door twee vingers naar elkaar toe of van elkaar af te bewegen. De afstand tussen de vingers verandert; je zou kunnen zeggen dat deze afstand wordt 'geschaald'. Het 'schalen' van de vingers bepaalt het 'schalen' van het object.

Onderstaande code spreekt voor zichzelf. In dit voorbeeld wordt gebruik gemaakt van de movieclip 'figuur\_mc'.

Let op dat er gebruik wordt gemaakt van de inputmode *GESTURE*.

```
import flash.events.TransformGestureEvent;

Multitouch.inputMode = MultitouchInputMode.GESTURE;

figuur_mc.addEventListener(TransformGestureEvent.GESTURE_ZOOM, pasaan);

function pasaan(event:TransformGestureEvent):void
{
    figuur_mc.scaleX *= event.scaleX;
    figuur_mc.scaleY *= event.scaleY;
}
```

## PAN

Als na het zoomen de afbeelding buiten het scherm valt, wil je de gebruiker de mogelijkheid geven de afbeelding te verplaatsen.

LET OP: Je kan hierbij geen gebruik maken van de event *TOUCH\_MOVE*, omdat de applicatie niet tegelijkertijd naar een 'touch'- en een 'gesture'- event kan luisteren. Daarom maken we gebruik van de 'gesture'-event *GESTURE\_PAN*.

Dit event wordt geactiveerd als de gebruiker met twee vingers over het scherm beweegt. Onderstaande code kan je toevoegen aan bovenstaande code om deze mogelijkheid in te bouwen:

```
figuur_mc.addEventListener(TransformGestureEvent.GESTURE_PAN, verplaats);

function verplaats(event:TransformGestureEvent):void
{
    figuur_mc.x += event.offsetX;
    figuur_mc.y += event.offsetY;
}
```

## ROTATIE

Net als bij de zoom-functie kan je de beweging van de vingers koppelen aan de beweging van een movieclip. In dit geval gaat het om een rotatie:

```
import flash.events.TransformGestureEvent;

Multitouch.inputMode = MultitouchInputMode.GESTURE;

figuur_mc.addEventListener(TransformGestureEvent.GESTURE_ROTATE, draai);

function draai(event:TransformGestureEvent):void
{
    figuur_mc.rotation += event.rotation;
}
```

## ACCELEROMETER

Eén van de populairste interacties op een mobiel device is de accelerometer. Hiermee kan je door het bewegen van jouw device een object over het scherm laten bewegen.

Deze beweging vindt plaats over drie assen: de x-, y- en z-as.

Animate maakt hiervoor gebruik van de class met de logische naam *accelerometer*. Bij elke beweging wordt de *UPDATE*-event aangeroepen en wordt de versnelling in de drie richtingen teruggegeven via *accelerationX*, *accelerationY* en *accelerationZ*.

In onderstaand voorbeeld laten we een eenvoudig balletje over de stage bewegen. Hierbij hoeven alleen gebruik maken van de x- en y-beweging van het device.

Maak allereerst een movieclip, genaamd 'bal\_mc', en plaats deze op het midden van de stage.

Daarna typen we in het 'actions'-panel allereerst de volgende code:

```
var vX:Number = 0;
var vY:Number = 0;
```

De snelheid in x- en in y-richting is in eerste instantie nul.

Daarna maken we een *Accelerometer*-variabele. Deze koppelen we aan een functie die wordt aangeroepen zo gauw er een beweging wordt waargenomen:

```
var am:Accelerometer = new Accelerometer();
am.addEventListener(AccelerometerEvent.UPDATE, onUpdate);
```

In de bijbehorende functie passen we de snelheid van het balletje aan:

```
function onUpdate(event:AccelerometerEvent):void
{
    vX = vX + event.accelerationX * 20;
    vY = vY + event.accelerationY * 20;
}
```

De beweging van de bal zetten we in een nieuwe functie. Deze wordt elke keer aangeroepen als het frame wordt vernieuwd:

```
bal_mc.addEventListener(Event.ENTER_FRAME, onFrame);

function onFrame(ev:Event):void
{
    bal_mc.x += vX;
    bal_mc.y += vY;
}
```

Om er voor te zorgen dat het balletje niet buiten het scherm komt, kan je onderstaande code toevoegen aan de functie 'onFrame':

```
if(bal_mc.x > (stage.stageWidth-bal_mc.width/2)){
    bal_mc.x = stage.stageWidth-bal_mc.width/2;
    vX = 0;
}
if(bal_mc.x < (0+bal_mc.width/2)){
    bal_mc.x = 0+bal_mc.width/2;
    vX = 0;
}
if(bal_mc.y > (stage.stageHeight-bal_mc.width/2)){
    bal_mc.y = stage.stageHeight-bal_mc.width/2;
    vY = 0;
}
if(bal_mc.y < (0+bal_mc.width/2)){
    bal_mc.y = 0+bal_mc.width/2;
    vY = 0;
}
```



### Opdracht 3: Accelerometer

Maak een applicatie waarmee je een balletje laat bewegen over het scherm. De beweging van het balletje wordt bepaald door de beweging van het mobile apparaat. Hoe schuiner het mobiele apparaat, hoe sneller het balletje beweegt. Het balletje mag NIET buiten de stage komen en blijft dus altijd op het scherm.

EXTRA ONDERDELEN:

- Maak een extra movieclip. Plaats deze op de stage en geef het een naam. Als het balletje deze movieclip raakt dan verdwijnt de movieclip.

Voor het botsen van twee movieclips gebruik je volgende code:

```
//Als de bal het ding raakt:
if(bal.hitTestObject(ding)){
    ding.x = -1000;
    ding.y = -1000;
}
```

De eenvoudigste (maar niet de mooiste) manier om een movieclip te 'verwijderen' is om deze buiten de stage te plaatsen.

- Zorg dat de movieclip op een willekeurige plaats op het scherm weer verschijnt.

Om een willekeurig getal te maken gebruik je de functie 'Math.random()'. Dit levert een getal tussen 0 en 1 op. Als je een getal tussen 0 en 100 wil dan vermenigvuldig je de functie met 100:

```
var myNum:Number = Math.random()*5;
```

Om de movieclip op een willekeurige plaats op de stage te plaatsen kan je onderstaande code gebruiken:

```
ding.x = Math.random()*stage.stageWidth;
ding.y = Math.random()*stage.stageHeight;
```

- Hou de score bij: elke keer als het balletje de movieclip raakt wordt de score met één verhoogt. De score wordt op het scherm getoond.
- Voeg nog twee extra movieclips toe; elk met een eigen kleur (en eigen score).

### Opdracht 4: Interactie

Plaats een (pas-)foto van jezelf op de stage. Zorg dat je er de volgende interacties op kan uitvoeren:

- Zoomen
- Roteren
- Verplaatsen

EXTRA ONDERDEEL:

Plaats ook een prullenbak op de stage. Als de foto op de prullenbak wordt geplaatst, wordt deze verwijderd. (gebruik hiervoor de methode 'hitTestObject').



## HOOFDSTUK 4: OPSLAG VAN GEGEVENS

Het bewaren van data op de device kan op verschillende manieren. We bespreken er twee: shared object en tekstbestand.

### SHARED OBJECT

Een Shared Object binnen Animate is als een cookie binnen PHP. Het is een gegevensbestand dat lokaal op het apparaat wordt opgeslagen. Binnen dit object kunnen meerdere gegevens, elk met hun eigen naam, worden opgeslagen. Hierbij kunnen zelfs hele arrays worden opgeslagen. Wil je dit op een telefoon gebruiken, dan moet je er wel voor zorgen dat de applicatie rechten heeft om te schrijven.

In deze paragraaf wordt in het kort beschreven hoe je een Shared Object maakt en hoe je hierin kan schrijven of lezen.

#### Het creëren

Maak in Animate een nieuwe “AIR for Android” – applicatie. Open “Actions” en voeg de volgende code toe:

```
import flash.net.SharedObject;

var so:SharedObject = SharedObject.getLocal("gegevens");
```

Als je de tweede regel intypt, zal automatisch de import worden toegevoegd. “so” is zelfgekozen; je kan hier zelf een andere naam voor gebruiken. Mocht het object “gegevens” niet bestaan, dan zal deze automatisch worden aangemaakt.

#### Het schrijven

Alle gegevens worden opgeslagen in de eigenschap “data” van het Shared Object. Binnen deze eigenschap kan elk item zijn of haar eigen naam krijgen. Als de data-items een waarde hebben gekregen, moeten deze nog worden toegevoegd aan het Shared Object. Dit gebeurt met de opdracht “flush”.

Met onderstaande code worden gegevens weggeschreven:

```
so.data.naam = "Kees";
so.data.leeftijd = 18;

var status:String = so.flush();
```

Met de tekstvariabele “status” kunnen we controleren of het wegschrijven gelukt is. Deze krijgt dan de waarde “flushed”.

#### Het lezen

Tijdens het lezen zetten we de diverse gegevens in een variabele voordat we deze tonen:

```
var nm:String = so.data.naam as String;
var lft:int = so.data.leeftijd as int;

trace (nm + " is " + lft + " jaar");
```

De gegevens in het data-object hebben in principe geen type (int, String, ..) meegekregen. Het is niet noodzakelijk, maar wel netjes om bij het uitlezen aan te geven om wat voor type variabele het gaat. Dit heet ‘typecasting’.

Als je niet weet welke eigenschappen in het data-object zijn weggeschreven, kan je ze allemaal op de volgende manier uitlezen:

```
for (var prop:String in so.data)
{
    trace (prop + " = " + so.data[prop]);
}
```

## Het verwijderen

Je kan een enkele eigenschap binnen een Shared Object verwijderen, of zelfs het volledige object:

```
var so:SharedObject = SharedObject.getLocal("gegevens");

delete so.data.leeftijd;

so.clear();
```

## DATA OPSLAAN IN EEN TEKSTBESTAND

In principe kan een Android applicatie alleen schrijven in de eigen map waarin de app is geïnstalleerd. Er is een aantal standaard mappen die worden gecreëerd bij installatie. Bij voorkeur staan deze op de sd-kaart:

*File.documentsDirectory*  
*File.applicationDirectory*  
*File.applicationStorageDirectory*

Deze mappen kunnen worden gebruikt om gegevens (SharedObjects, tekstbestanden, XML, SQLite databases) op te slaan.

Om er achter te komen wat de padnamen van deze mappen zijn kan je onderstaande code gebruiken:

```
trace (File.documentsDirectory.nativePath);
trace (File.applicationDirectory.nativePath);
trace (File.applicationStorageDirectory.nativePath);
```

Naast bovenstaande mappen zijn er ook nog algemene mappen waarin elke applicatie lees- en schrijfrechten heeft (als de permissie "WRITE\_EXTERNAL\_STORAGE" is aangevinkt):

*File.documentsDirectory*  
*File.userDirectory*  
*File.desktopDirectory*

In onderstaand voorbeeld gebruiken we de applicationStorageDirectory om een tekstbestand in te maken, waarin we kunnen lezen en schrijven.

LET OP: Zorg er wel voor dat in Animate de juiste bibliotheek wordt geïmporteerd, zodat je gebruik kan maken van het filesystem:

```
import flash.filesystem.FileMode;
```

## Schrijven naar bestand

Allereerst maken we een *File* aan in een bepaalde map ('applicationStorageDirectory') en met een bepaalde naam: 'gegevens.txt':

```
//MAAK EEN BESTAND IN DE JUISTE MAP:
var file:File = File.applicationStorageDirectory.resolvePath("gegevens.txt");
```

Daarna wordt een *FileStream* gecreërd, zodat er later geschreven kan worden naar het bestand. Mocht er geen *FileStream* gemaakt kunnen worden ('IOException'), moet er een foutmelding verschijnen.

```
//MAAK VERBINDING TUSSEN APPLICATIE EN BESTAND MOGELIJK
var stream:FileStream = new FileStream();
//FOUT BIJ VERBINDEN
stream.addEventListener(IOException.IO_ERROR, foutje);
function foutje(event:IOException):void
{
    trace(event);
}
```

Daarna wordt via deze stream het bestand geopend om te kunnen schrijven:

```
//OPEN HET BESTAND OM TE SCHRIJVEN
stream.openAsync(file, FileMode.WRITE);
//SCHRIJF DATA IN HET BESTAND
stream.writeUTFBytes("naam=Kees&plaats=Rotterdam");
//SLUIT DE VERBINDING
stream.close();
```

Het bestand wordt asynchroon geopend. Het voordeel hiervan is dat andere AIR-processen die lezen en of schrijven niet worden onderbroken.

Voor het schrijven naar het bestand zijn diverse mogelijkheden. Hier wordt gebruik gemaakt van 'writeUTFBytes'. Voor andere mogelijkheden verwijst ik je naar internet. Bijvoorbeeld:

[http://help.adobe.com/en\\_US/AnimatePlatform/reference/actionscript/3/Animate/filesystem/package-detail.html](http://help.adobe.com/en_US/AnimatePlatform/reference/actionscript/3/Animate/filesystem/package-detail.html)

## Een bestand uitlezen

Het begin van de AS3-code om een bestand uit te lezen is hetzelfde: het bestand moet gevonden worden en er moet een *FileStream* ('verbinding') gemaakt worden. Alleen wordt de verbinding nu gebruikt om het bestand te lezen i.p.v. te schrijven:

```
//MAAK EEN BESTAND IN DE JUISTE MAP:
var file:File = File.applicationStorageDirectory.resolvePath("gegevens.txt");
//MAAK VERBINDING TUSSEN APPLICATIE EN BESTAND MOGELIJK
var stream:FileStream = new FileStream();
//OPEN HET BESTAND OM TE LEZEN
stream.openAsync(file, FileMode.READ);
```

Pas als de verbinding is gemaakt kan het bestand worden uitgelezen. Vandaar dat er een *EventListener* wordt gemaakt. Deze roept een functie aan waarin de tekst uit het bestand wordt gelezen. Daarna wordt de gegevens in deze tekst ('naam' en 'woonplaats') uit elkaar gehaald en apart getoond:

```
stream.addListener(Event.COMPLETE, onComplete);

function onComplete(event:Event):void
{
    //LEES DE TEKST UIT HET BESTAND
    var str:String = stream.readUTFBytes(stream.bytesAvailable);
    //HAAL DE VARIABELEN UIT DE TEKST
    var gegevens:URLVariables = new URLVariables();
    gegevens.decode(str);
    //LAAT DE VARIABELEN ZIEN
    trace (gegevens.naam, gegevens.plaats);
    //SLUIT DE VERBINDING
    stream.close();
}
```

Omdat we tijdens het schrijven de gegevens hebben gescheiden met een '&'-teken kunnen we deze via 'gegevens.decode' weer afzonderlijk laten zien.

Als er in het bestand alleen maar tekst staat, kan deze eenvoudig worden uitgelzen op de volgende manier:

```
var str:String = stream.readUTFBytes(stream.bytesAvailable);
trace (str);
```

## Tekst toevoegen aan en bestand

Het is ook mogelijk om tekst toe te voegen aan een bestaand bestand:

```
//MAAK EEN BESTAND IN DE JUISTE MAP:
var file:File = File.applicationStorageDirectory.resolvePath("gegevens.txt");
//MAAK VERBINDING TUSSEN APPLICATIE EN BESTAND MOGELIJK
var stream:FileStream = new FileStream();
//OPEN HET BESTAND OM TOE TE VOEGEN
stream.openAsync(file, FileMode.APPEND);
//VOEG DATA TOE AAN HET BESTAND
stream.writeUTFBytes("$telefoon=010-1234567");
//SLUIT DE VERBINDING
stream.close();
```

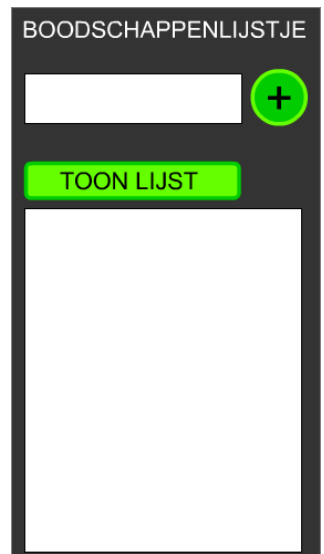
## Opdracht 5: Boodschappenlijstje

Maak met Animate een Android-applicatie waarmee je een boodschappenlijstje kan maken. Deze kan er zo uitzien:

Via een tekstveld kunnen er items in het lijstje worden bijgeschreven. De lijst kan worden getoond via als er op "toon lijst" wordt geklikt. Het boodschappenlijstje wordt in een bestand op de device opgeslagen.

EXTRA: voor onderstaande opties krijg je extra punten. Onderzoek de mogelijkheden en pas deze toe.

- Is het mogelijk om items te verwijderen uit de lijst?
- Kan je de lijst weggooien/leegmaken en opnieuw beginnen?



Zorg dat de applicatie een gebruikersvriendelijke interface heeft en zowel in portrait als in landscape werkt.

## HOOFDSTUK 5: ANIMATE EN PHP

In één van de vorige hoofdstukken heb je geleerd hoe je data lokaal kan opslaan. De enige die dan gebruik kan maken van deze data, is de gebruiker van de device waar deze data op staat. Om iets flexibeler met deze gegevens om te kunnen gaan is het beter om deze in een (MySQL) database op te slaan die voor meer mensen toegankelijk is.

Een MySQL-database is vanuit Animate te gebruiken. Als we tenminste via een PHP-pagina met deze database communiceren...

De gegevens moeten vanuit de Animate-applicatie naar een PHP-pagina worden gestuurd. Deze PHP-pagina stuurt daarna de gegevens naar de database. Het is gelukkig ook mogelijk om daarna diverse data vanuit de PHP-pagina terug te lezen in de Animate-applicatie. Hiervoor maken we gebruik van *URLVariables*. Ook die ben je al eerder tegengekomen.

Op internet zijn diverse tutorials te vinden die de samenwerking van AS3 en PHP uitleggen. Mocht onderstaande uitleg niet voldoende zijn, ga dan zelf op zoek naar antwoorden in één van die tutorials.

### DATA NAAR EEN PHP-PAGINA STUREN

In onderstaand voorbeeld is in Animate een pagina gemaakt met daarin twee tekstvelden. Daaronder staat een knop en een reactieveld. Als op de knop wordt geklikt worden de twee tekstvelden uitgelezen. De inhoud wordt naar een php-pagina gestuurd. De reactie ('gelukt' of 'niet gelukt') komt in het reactieveld.

Nu wordt er eerst in actionscript de mogelijkheid gecreëerd om met een php-pagina te communiceren:

```
//Maak een URLRequest naar de juiste URL
var url:String = "http://bijlsma.ict-lab.nl/flash/test1.php";
var reqURL:URLRequest = new URLRequest(url);
//gebruik de methode "POST"
reqURL.method = URLRequestMethod.POST;
//definieer een 'loader'
var loader:URLLoader = new ULLoader(reqURL);
//Zorg dat de loader diverse variabelen kan versturen
loader.dataFormat = ULLoaderDataFormat.VARIABLES;
```

In de *URLRequest* wordt aangegeven naar welke pagina de gegevens verstuurd moeten worden. Dit zal gebeuren met de *POST*-methode. Als laatste wordt er een *loader* gemaakt die verschillende variabelen naar de php-pagina kan versturen.

```
knop_btn.addEventListener(MouseEvent.CLICK, schrijven);

function schrijven (event:MouseEvent):void
{
    //Definieer de variabelen en geef ze een waarde
    var vars : URLVariables = new URLVariables();
    vars.tekst1 = veld1.text;
    vars.tekst2 = veld2.text;
    //Voeg de variabelen toe aan de URLRequest
    reqURL.data = vars;
    loader.load(reqURL);
    //Als de loader klaar is, wordt een functie aangeroepen
    loader.addEventListener(Event.COMPLETE, controle);
}
```

De variabelen worden uit de tekstvelden gelezen, in een *URL Variables*-variabele gestopt en verstuurd via de *loader*.  
 Als dit is gebeurd wordt een (controle-) functie aangeroepen.

Nu moet de php-pagina de data verder afhandelen.

## DE PHP-PAGINA

De PHP-pagina is redelijk eenvoudig: de gegevens worden via een *POST* uitgelezen en daarna aan de database toegevoegd:

```
<?php
require ('../connection.php');
$t1= $_POST['tekst1'];
$t2 = $_POST['tekst2'];

$opdracht = "INSERT INTO MEOI3_tabel(Veld1, Veld2) VALUES ('$t1', '$t2')";

if (mysql_query($opdracht))
{
    echo "opm=toegevoegd";
}
else
{
    echo "opm=fout".mysql_error();
}
?>
```

In de controle wordt een variabele 'opm' gemaakt. Deze wordt straks in Actionscript uitgelezen en in het reactieveld getoond.

Als je bovenstaande Animate-applicatie en de bijbehorende php-pagina hebt gemaakt kan je deze testen. Dan moet je wel even de laatste regel uit de functie 'schrijven' verwijderen.

## DATA UIT EEN PHP-PAGINA LEZEN

De *URLRequest* en de *loader* zijn al in de eerste paragraaf gemaakt. Het enige dat nu nog moet gebeuren, is de opmerking uitlezen die in de php-pagina is gemaakt:

```
function controle( event:Event):void
{
    reactieveld.text += loader.data.opm;
}
```

De *loader* kan dus gebruikt worden voor zowel schrijven als lezen. Natuurlijk mag het lezen pas starten als de *loader* compleet geladen is. Vandaar de eventlistener in de de functie 'schrijven'.

LET OP:

Het kan voorkomen dat de eerste variabele die in de php-pagina wordt ge-echo'd niet goed gelezen wordt in Animate. Je kan er dan voor kiezen om eerst een 'lege variabele' te sturen:

```
echo "eerste=leeg&opm=toegevoegd";
```

De verschillende variabelen worden gescheiden door een ampersand ('&').

## Opdracht 6

Maak een mobiele Animate-applicatie waarmee je de naam en telefoonnummer van een persoon kan wegschrijven naar een PHPMyAdmin-tabel.  
Door een druk op een knop worden de twee velden uitgelezen en de gegevens via een php-pagina weggeschreven naar de tabel.

Maak daarnaast een knop (en tekstveld) om alle gegevens uit deze tabel te lezen.  
Na een druk op de knop worden de gegevens overzichtelijk getoond in de applicatie.  
Laat ook hierbij de communicatie via een php-pagina lopen.

De Animate-applicatie wordt onder de naam 'opdracht6.flw' opgeslagen.  
De php-pagina('s) worden opgeslagen op de webserver van ict-lab.  
Maak hiervoor een speciale map 'Animate'.  
Ook voor de tabel ('Animate\_tabel') maak je gebruik van ict-lab (phpmyadmin.ict-lab.nl).

Zorg dat de volledige applicatie overzichtelijk en gebruikersvriendelijk is.



## HOOFDSTUK 6: PLAATSBEPALING

Mobiele apparaten kunnen jouw locatie bepalen via GPS, maar ook via wifi of het mobiele netwerk. Geolocatie via GPS is het meest nauwkeurig, maar kost ook de meeste batterijen. Animate maakt voor de plaatsbepaling gebruik van de klasse *Geolocation*. Tijdens elke *update* wordt o.a. een nieuwe lengte- en breedtegraad (*longitude* en *latitude*) doorgegeven. Je kan zelf instellen in welke interval de update uitgevoerd moet worden.

### GEOLOCATION

Laten we eerst maar eens gaan kijken hoe Animate de gegevens uitleest.

We beginnen met het maken van een instantie van de klasse *Geolocation*. Alleen als *Geolocation* wordt ondersteund, mag de code uitgevoerd worden. Anders komt er een melding op het scherm.

```
import flash.sensors.Geolocation;

if (Geolocation.isSupported)
{
    var geo:Geolocation = new Geolocation();
    geo.setRequestedUpdateInterval(100);
    geo.addEventListener(GeolocationEvent.UPDATE, geoUpdateHandler);
}
else
{
    uitvoer_txt.text = "Geen Geolocation ondersteuning!";
}
```

De interval wordt aangegeven in milliseconden. Als je deze functie weglaat wordt er ook een update-interval ingesteld, maar deze is afhankelijk van het apparaat waarop de applicatie draait. Bij elke update wordt de functie "geoUpdateHandler" aangeroepen.

```
function geoUpdateHandler(event:GeolocationEvent):void
{
    lengte_txt.text = event.longitude.toString()+"°";
    breedte_txt.text = event.latitude.toString()+"°";
}
```

Via de event wordt de *longitude* en *latitude* uitgelezen. Aangezien deze van het type *Number* zijn, worden ze omgezet naar het type *String* voordat ze in het tekstveld worden geplaatst. Als laatste wordt het graden-teken toegevoegd.

### SNELHEID

Je kan de *Geolocation*-klasse ook gebruiken om de snelheid te bepalen. Deze is natuurlijk afhankelijk van de nauwkeurigheid waarmee de gps-coördinaten kunnen worden bepaald. De snelheid wordt gemeten in meters per seconde. Plaats onderstaande code in de *updateHandler* om de snelheid (in km/uur) te berekenen:

```
var mps:Number = event.speed; //meter per seconde
var kph:Number = mps * 3.6; //km per uur

snelheid_txt.text = kph+" km/uur";
```

## GOOGLE MAPS

Je kan redelijk simpel de gevonden coördinaten tonen op de website van GoogleMaps. De coördinaten worden meegegeven als variabelen in een url. Het enige dat je nog moet doen, is zorgen dat de applicatie de webpagina in een browser opent. Zorg er daarbij wel voor dat de applicatie toestemming heeft om gebruik te maken van internet. In onderstaand voorbeeld doen we dat via een knopklik methode.

Maak een knop en koppel hier via een eventlistener een functie aan. In de functie wordt de url gecreëerd en naar deze url genavigeerd. Hier staat de volledige code:

```
import flash.events.MouseEvent;
import flash.net.URLRequest;

kaart_btn.addEventListener(MouseEvent.CLICK, onKlik);

function onKlik(event:MouseEvent):void
{
    var url:String = "http://maps.google.com/maps?ll=40.77,-73.97";
    var request:URLRequest = new URLRequest(url);
    navigateToURL(request);
}
```

Als je op deze manier een routeplanner wilt maken, dan is dat ook mogelijk. Je kan in de url het beginadres (*saddr*) en eindadres (*daddr*) aangeven:

```
function onKlik(event:MouseEvent):void
{
    var begin:String = encodeURI("Heer Bokelweg 255, Rotterdam");
    var eind:String = encodeURI("Cor Kieboomplein 501, Rotterdam");

    var url:String = "http://maps.google.nl/maps?saddr="+begin+"&daddr="+eind;
    var request:URLRequest = new URLRequest(url);
    navigateToURL(request);
}
```

Je kan de webpagina ook binnen de applicatie opvragen. Hierbij wordt gebruik gemaakt van de *StageWebView* klasse en een *ViewPort* gecreëerd. Dit is in zekere mate te vergelijken met een *iFrame* op een webpagina.

```
import flash.media.StageWebView;
import flash.geom.Rectangle;
import flash.events.LocationChangeEvent;

var web:StageWebView = new StageWebView();
web.viewPort = new Rectangle(0, 0, stage.stageWidth, stage.stageHeight);
web.stage = this.stage;

web.addEventListener(LocationChangeEvent.LOCATION_CHANGING, locChanging);

function locChanging(e:LocationChangeEvent):void
{
    e.preventDefault();
}

web.loadURL("http://maps.google.com/maps?ll=51.92762,4.47813");
```

Het zou kunnen gebeuren dat een mobiel apparaat standaard een webpagina in de browser wil openen. Om dit te voorkomen wordt er een eventlistener aangemaakt die dit voorkomt.

## GOOGLE API

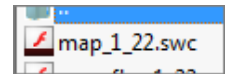
(Zie ook [https://developers.google.com/maps/documentation/javascript/tutorial#api\\_key](https://developers.google.com/maps/documentation/javascript/tutorial#api_key))

Het is natuurlijk nog mooier als je Google Maps in jouw eigen applicatie kan openen. Hiervoor moet je gebruik maken van een API van Google die geschikt is voor AIR for Android. Daarnaast zal je bij Google een key moeten aanvragen om gebruik te kunnen maken van deze API. De key wordt in eerste instantie gekoppeld aan een url. Aangezien je key voor een mobiele applicatie gaat gebruiken hoef je geen relevante url op te geven. Maar je moet hem wel onthouden!

De Google Maps API voor Animate bevindt zich in een sdk die te vinden is op:

<http://maps.googleapis.com/maps/Animate/release/sdk.zip>

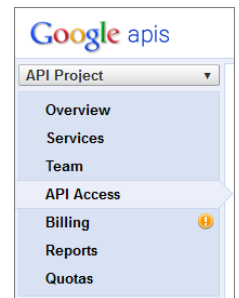
Als je het zipbestand uitpakt vind je in de 'lib'-map twee swc-bestanden. Het 'flex'-bestandje wordt gebruikt binnen Flex Builder. Voor Animate gebruiken we het andere bestand.



Om een key te krijgen, moet je inloggen op google met jouw google-account.

Als je die niet hebt, zal je die moeten aanmaken.

Als je bent ingelogd ga je naar de API Project console. Onder 'API Access' kan je een nieuwe key aanvragen. Klik op 'Create New Android key'.



De key die nu gecreëerd wordt zal je moeten gebruiken in Animate.

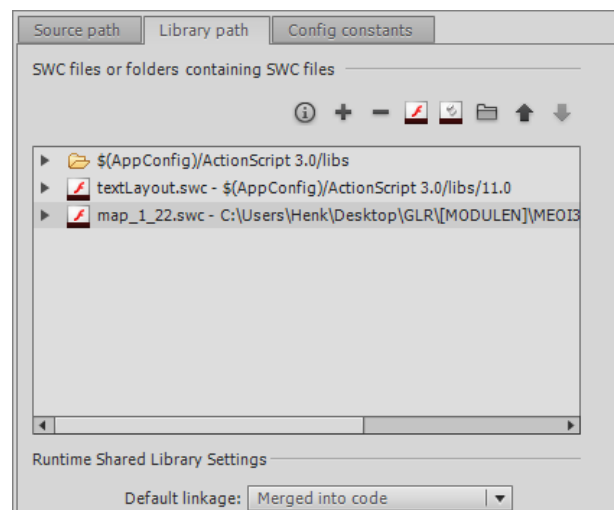
Als je de SDK (waar de API zich in bevindt) hebt gedownload, moet de API nog worden toegevoegd aan de applicatie.

Kies in Animate: "File – ActionScript Settings"

Selecteer het tabblad "Library path".

Klik op "Browse to SWC File". Blader naar het swc-bestand in de map "lib" van de sdk-map en selecteer deze.

Kies onderaan, bij "Default linkage", voor "Merged into code".



Klik op OK

De API is nu toegevoegd aan de applicatie.

We gaan nu de applicatie verder opbouwen.

In Actionscript gaan we allereerst de volledige googlemaps bibliotheek importeren, evenals het onderdeel uit de bibliotheek waarmee we een marker kunnen plaatsen. Daarna maken we een nieuwe Map-instantie waarbinnen de gehele map wordt gecreëerd. Vervolgens voegen we de API-key toe aan de map, evenals de url die bij deze key hoort.

```
import com.google.maps.*;
import com.google.maps.overlays.Marker;

var map:Map = new Map();
map.key = "AIzaSyB0g28";
map.url = "https://code.google.com/apis/console/#project:171674826463:access";
```

De map krijgt een grootte. Daarnaast wordt er een functie aangeroepen als de map is opgebouwd.

Dan wordt de map toegevoegd aan de stage.

```
map.setSize(new Point(480,800));

map.addEventListener(MapEvent.MAP_READY, onMapReady);

this.addChild(map);
```

Sla de applicatie op en test deze. Als het goed is krijg je nu de wereldkaart van Google Maps te zien.

Nu gaan we de coördinaten van onze huidige positie toevoegen.

We creëren een nieuwe Geolocation instantie. Maar we kunnen de bijbehorende update-handler pas uitvoeren als de map klaar is; vandaar dat we deze eventhandler in de functie "onMapReady" plaatsen. De handler roept een nieuwe functie op ("onGeoUpdate"). In deze functie gebeurt het volgende:

- De huidige coördinaten worden in één (*LatLng*-)variabele bewaard.
- De map wordt gecentreerd op de punt, de zoomgrootte en het maptype wordt bepaald.
- Er wordt een marker geplaatst op de huidige coördinaten.

```
var geo:Geolocation = new Geolocation();

function onMapReady(event:Event):void
{
    geo.addEventListener(GeolocationEvent.UPDATE, onGeoUpdate);
}

function onGeoUpdate(event:GeolocationEvent):void
{
    var latlong:LatLng = new LatLng(event.latitude, event.longitude);
    map.setCenter(latlong, 6, MapType.NORMAL_MAP_TYPE);

    var marker:Marker = new Marker(latlong);
    map.addOverlay(marker);
}
```

Sla de applicatie op en test deze door verschillende coördinaten in te vullen.

## Opdracht 7

Maak een applicatie die de coördinaten van de huidige locatie laat zien (lengte- en breedtegraad). Toon ook de snelheid.

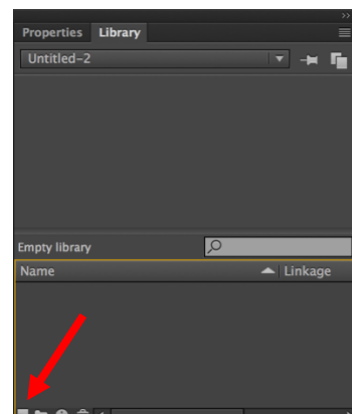
EXTRA:

- Na een druk op een knop verschijnt het punt in Google Maps.
- Gebruik de Google Maps API om de huidige locatie te tonen.

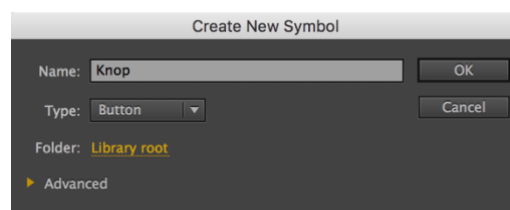
## BIJLAGEN

### EEN KNOP MAKEN

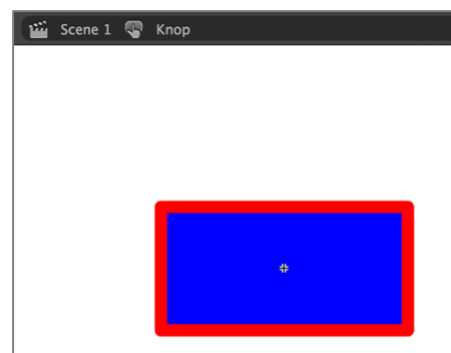
Klik in het tabblad "Library"  
onderin op het icon "New Symbol..."



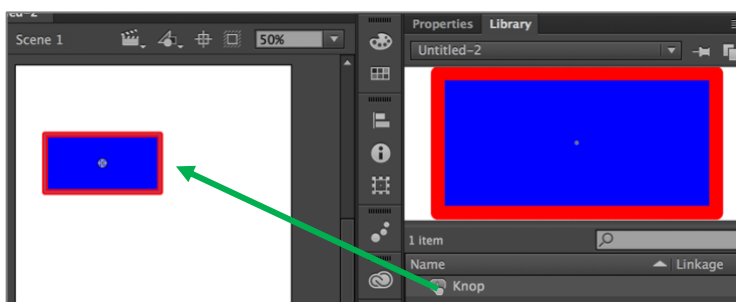
Kies bij "Type: Button".  
Geef de Button een naam (bijv. "Knop").



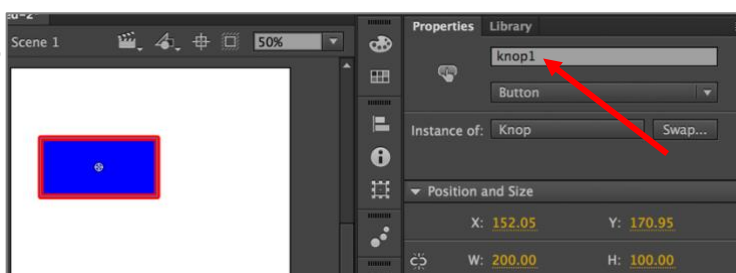
Teken in het nieuwe canvas een knop.  
Als je klaar bent, ga je terug naar scene 1.  
(Klik op "Scene 1").



In Scene 1 kan je nu vanuit  
je library de Knop in jouw  
applicatie slepen.



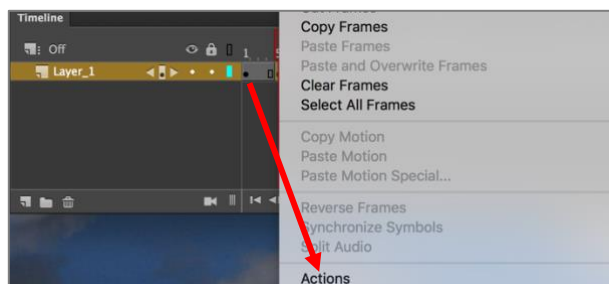
Vergeet niet deze "instantie"  
van de Knop een naam te  
geven via "Properties".



Je kan de knop uit de library vaker gebruiken. Geef elke instantie wel een andere naam.

## VOEG SCRIPT TOE AAN DE TIMELINE

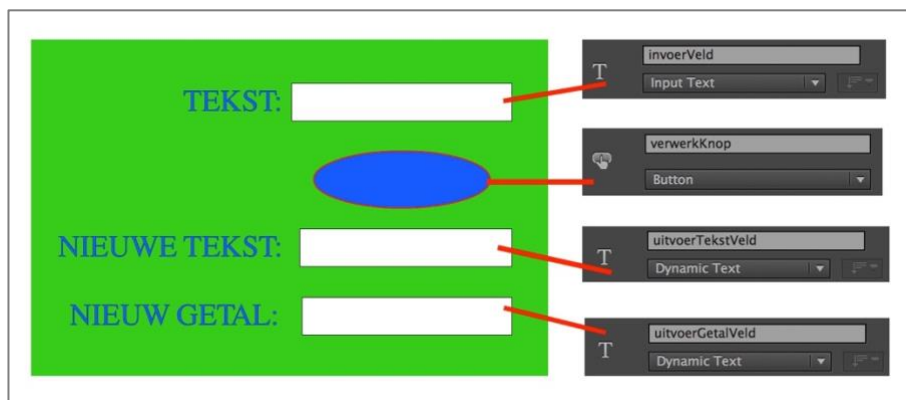
Zorg dat de timeline zichtbaar is in Animate.  
Is dit niet het geval, kies dan voor “Window – Timeline”.  
Klik met je rechter muisknop op een (key-)frame en kies voor “Actions”.



Typ je script in het Actions-venster.  
Sluit af als je klaar bent (klik op het kruisje).  
In het (key-)frame staat nu een  $\alpha$ -teken; dit geeft aan dat er in dit frame een script staat.

## TEKSTVELD UITLEZEN EN BEWERKEN VIA ACTIONSSCRIPT

Interface:



Rechtsklik in het juiste keyframe van de tijdlijn. Kies ‘Actions’.  
Voeg (bijvoorbeeld) de volgende code toe:

```
//functie maken bij knopklik
verwerkKnop.addEventListener(MouseEvent.CLICK, verwerk);

//de functie
function verwerk(mE: MouseEvent): void
{
    //tekstveld uitlezen:
    var tekst: String = invoerVeld.text;
    //tekst in tekstveld zetten
    uitvoerTekstVeld.text = "De tekst is: " + tekst;
    //tekst converteren naar integer
    var getal:int = int(tekst);
    //berekening:
    var nieuwGetal = 5 * getal;
    //omzetten naar tekst
    var nieuweTekst = nieuwGetal.toString();
    //nieuweTkest in het tekstveld zetten
    uitvoerGetalVeld.text = "nieuwe waarde: " + nieuweTekst;
}
```

**LET OP:**

**FONTS MOETEN (meestal) GE-EMBED WORDEN!!**

