

# Basis actionscript

## 1. Inleiding

De eerste lessen van deze cursus "basis programmeren/actionscripting" lijken soms niks te maken te hebben met Flash. Dat komt omdat je in feite eerst leert programmeren. We zullen ons uiterste best doen om ook in deze eerste lessen Flash in de stof te betrekken. Toch zal het soms misschien even doorbijten zijn, omdat met name de eerste oefeningen misschien weinig te maken lijken te hebben met de dingen die je van plan bent te gaan maken wanneer je actionscript eenmaal onder de knie hebt. Toch is deze basis onontbeerlijk.

Wanneer je al enige ervaring hebt met actionscript zal een oefening soms makkelijk lijken. Verderop in de cursus echter, veronderstellen we de basis als bekend en het kan nooit kwaad deze nog eens door te nemen. Voor degenen die voor het eerst beginnen met programmeren is het ten zeerste aan te raden alle oefeningen te doen. Voor veel oefeningen is het nodig iets in het Flash actionscript panel in te typen, en daarna de movie te testen. Zeker bij de eerste oefeningen kan de verleiding groot zijn deze stappen over te slaan en de oefening 'in je hoofd' te doen, en meteen verder te lezen. Ook dat is niet aan te raden, omdat het doel van deze oefeningen nu juist is dat je bepaalde dingen zelf leert ontdekken. Bovendien maken de eerste oefeningen je vertrouwd met het actionscript panel, en de foutmeldingen die flash je geeft wanneer je een movie test.

### 1.1 Wat is actionscript?

Actionscript is een taal, en net zoals bij gewone talen zoals Engels of Frans is het doel van deze taal communiceren. In het geval van actionscript, en in feite bij alle andere programmeertalen, is het belangrijk om precies te communiceren met een computer. Of eigenlijk, met een computerprogramma. Daarmee begint meteen een van de moeilijkheden, want in tegenstelling tot mensen, bij wie je nog aan zou kunnen komen met "geef zout" of misschien zelfs "zout" of zelfs "wat is het eten flauw" zijn computers niet zo flexibel en fantasievol. Bij programmeertalen zul je de taal tot op de letter nauwkeurig, exact volgens de syntax moeten gebruiken. Doe je dat niet, dan snapt de computer het niet, en krijg je een foutmelding.

Wanneer je gaat programmeren, en in ons geval dus actionscripten, zul je regelmatig merken dat Flash helemaal niet doet wat jij wilt. Flash doet letterlijk wat jij hebt gezegd, niets meer of minder. Computers voeren stap voor stap de opdrachten uit die jij ze geeft. Computers hebben geen inzicht in het uiteindelijke doel van die opdrachten, en houden zich niet bezig met boodschappen op betrekkingniveau of achterliggende motieven. Het doel van deze cursus is dat je leert uitzoeken hoe je Flash exact kunt vertellen wat jij wilt dat Flash moet doen.

## 1.2 Vooraf enkele belangrijke begrippen.

Wat is en wat doet een computerprogramma?

### 1. Statement

Een computerprogramma is een reeks van instructies of stappen dat de computer moet uitvoeren.

Iedere stap is bedoeld om een stukje informatie of data te manipuleren of te veranderen.

Elke individuele instructie noemt men een **statement**. Deze statements worden met een **;** afgesloten.

**vb. gotoAndPlay();**

### 2. Assignments (toewijzing)

Dit zijn allemaal voorbeelden van assignments:

```
a = 1
b = 10
a = b
naam = `bente`
naam = "irma"
datum = new Date()
dag = datum.getMonth()
```

En zo kunnen we nog wel een tijdje doorgaan. Als je wel eens in de code van een ander hebt gekeken zul je zien dat een assignment in programmeerland aan de orde van de dag is.

Met de assignment komen we meteen nog een belangrijk element in programmeertalen tegen: de variabele

**Een variabele is een naam waaraan je een waarde kunt toekennen.**

### 3. Variabelen

Aangezien programmeren (hier in dit geval actionscripting) erin bestaat stukjes data in het geheugen van de computer te veranderen moet er een manier bestaan om zo'n stukje data voor te stellen. Deze voorstelling representeert dan een stukje data in het geheugen van de computer. Deze voorstelling van een stukje data noemen we een variabele. Als we statements gaan schrijven die values (waarden) gaan veranderen moeten we aan die waarden een naam toekennen. De computer leest die naam en gebruikt direct de value of de waarde ervan.

De variable heeft 3 delen:

- naam van de variabele
- datatype dat opgeslagen is in de variabele
- de value of waarde

We maken een variable door het **var** statement te gebruiken.

**vb. var value1:Number = 17;**

## 4. Datatypes

Je kunt in actionscript veel datatypes gebruiken om variables te maken.

**Fundamental datatypes** (eenvoudige, bevatten maar één waarde)

### 1. String

tekstwaarde, zoals een woord, naam, titel enz.

```
var tekst:String = "titel";
```

```
vb var naam:String = "webklus";  
var mijn_tekst:String = "hallo";
```

let er op : de waarden van een string staan tussen aanhalingstekens

### 2. Number

gelijk welk numerieke waarde

```
var cijfer:Number = 5;
```

**int.** Integer: een "rond" cijfer.

**uint.** unsigned integer: een "rond" cijfer maar niet negatief

```
vb var mijnhuisnummer = 12;  
var leeftijd = 32;
```

**3. Boolean.** Een true of false value

```
var switch:Boolean = false;
```

### Complex Datatypes

**MovieClip, TextField, SimpleButton, Date**

Deze zijn complex omdat ze bestaan uit verschillende waarden die gegroepeerd zijn om deze datatypes te beko-  
men. vb. Een datum bestaat uit een dag maand jaar .

Een movieclip heeft een breedte en een hoogte, kleur ed. (properties)

## Werken met assignments, datatypes en variabelen

```
var leeftijd = 20;
var schoenmaat = 38;
var iq = leeftijd + schoenmaat;
trace(iq)
```

In de variabele iq staat nu 58. Getallen bij elkaar optellen is gewoon rekenen. Met trace(iq) zie je deze waarde in het outputvenster.

```
var voornaam = "Jodie ";
var achternaam = "Hertenjong";
var hele_naam = voornaam + achternaam;
```

In de variabele hele\_naam staat nu "Jodie Hertenjong".

Strings bij elkaar optellen betekent namelijk dat je ze aan elkaar plakt. Concateneren wordt dat ook wel genoemd.

### *waarde met 1 vermeerderen*

```
var teller:Number = 0;
teller=teller+1
teller=teller+1
```

```
//korter
teller++;
trace(teller) bekijk resultaat in outputwindow.
```

### *//waarde met 1 verminderen*

```
teller--;
```

### *//waarde vermeerderen in stappen*

```
var teller2 = teller2 + 10;
of korter
teller2+= 10;
trace(teller2);
```

```
met 10 verminderen          teller2 -= 10;
met 10 vermenigvuldigen    teller2 *= 10;
door 10 delen               teller2 /= 10;
```

## Operatoren

We werken niet alleen variabelen, assignments en datatypes, maar ook met operatoren.

**+ \* ==**

**De + operator** telt twee waarden bij elkaar en worden gebruikt om berekeningen te maken,

```
var sum:Number = 32 + 23;
```

**De \* operator** vermenigvuldigt waarden met elkaar met als resultaat één nummer.

```
var energy:Number = mass * speedOfLight * speedOfLight;
```

**De == operator** vergelijkt twee waarden om te zien of ze gelijk zijn, dat resulteert in een true of false boolean waarde

```
if (dayOfWeek == "Woensdag")
{
gotoAndPlay(trashday);
}
```

Het wordt zo langzamerhand de hoogste tijd voor wat oefeningen in Flash.

Voordat we zo ver zijn moeten we de Flash ontwikkel omgeving een beetje aanpassen.

- nieuwe layout kiezen met venster-werkruimte-ontwerper
- handelingenvenster aanpassen naar expertmode door de scriptassistentie uit te zetten. venster-handelingen-scriptassistentie uitklikken en bij de opties regelnummering aanvinken.

## Oefening 1

Open de Flash ontwikkel omgeving en maak eventueel een nieuwe movie aan.

**In frame 1 van layer 1 zet je de volgende code:**

```
// eerste oefening
var bericht:String = "hallo wereld";
trace( bericht );
```

Gelukt? Test je movie (ctrl-enter op de pc, cmd-enter op de mac) en bekijk het resultaat!

Kreeg je tijdens het testen het flash output window te zien met de tekst hallo wereld ?  
Gefeliciteerd, je hebt een klassieke opdracht volbracht!

Wat we in deze eerste oefening hebben gedaan is een variabele aangemaakt, met de naam bericht. We hebben er de waarde hallo wereld in gestopt, en vervolgens aan Flash gevraagd de inhoud van de variabele bericht naar het output window te sturen.

Er zijn nog een paar kleine dingen meer te vertellen over deze drie regels actionscript.

Misschien is je opgevallen dat we iedere regel hebben afgesloten met een ; . Dit is in actionscript niet strikt noodzakelijk, maar wel een goede gewoonte. Vrijwel alle andere talen verwachten namelijk wel een ; aan het einde van een opdrachtregel, en mocht je nog eens incidenteel een andere taal gaan gebruiken dan hoef je je in ieder geval niet meer aan te wennen om iedere regel netjes af te sluiten.

De eerste regel van het script begint met // . Door deze twee schuine strepen voor een regel te zetten wordt de regel commentaar. Flash slaat die regels over. Maar wij kunnen ze wel lezen, en dat is een goede zaak. Met commentaar kun je je code van verklarende tekst voorzien, om de leesbaarheid te vergroten.

De laatste regel bestaat uit het commando trace() .Je hebt al gezien dat trace() de waarde die je tussen haakjes meestuurt laat zien in het flash output window.

## Oefening 2: Werken met strings

Open de Flash ontwikkel omgeving en maak eventueel een nieuwe movie aan. In frame 1 van layer 1 zet je de volgende code:

```
var naam:String = "zomaar een naam";
var bericht :String = "Hallo" + naam + " Welkom";
trace(bericht)
```

1.test het script

2.vul je naam in bij 'naam'

3.test het script weer

4.probeer de tekst te veranderen in :hallo [naam] hoe gaat het met je? en welkom bij de cursus [naam]

## Oefening 3: Werken met strings – vervolg

Open de Flash ontwikkel omgeving en maak eventueel een nieuwe movie aan. In frame 1 van layer 1 zet je de volgende code:

```
var voornaam:String = "";  
var achternaam:String = "";  
var bericht = "Hallo " + voornaam + " Welkom bij de cursus";
```

**trace(bericht)**

- 1 .test het script
2. vul je voornaam in bij 'voornaam'
3. vul je achternaam in bij 'achternaam'
4. test het script weer

5.probeer deze tekst te veranderen in : hallo [voornaam] [achternaam], hoe gaat het met je?

---

## Oefening 4: Rekenen met getallen, appels en peren.

Open de Flash ontwikkel omgeving en maak eventueel een nieuwe movie aan. In frame 1 van layer 1 zet je de volgende code:

```
var appels:Number = 0;  
var peren:Number = 0;  
var bericht = appels + peren;  
// output  
trace(bericht)
```

- 1 .test het script
2. vul een getal in bij appels
- 3.vul een getal in bij peren
- test het script weer
- 4.zet nu de getallen bij appels en peren tussen aanhalingstekens.
- 6.Test het script nogmaals
- 7.verander het plusteken tussen appels en peren eens in een minteken, vermenigvuldigingsteken (\*) of deeltteken (/)\*

Het is je misschien opgevallen dat steeds wanneer een variabele voor de eerste keer gebruikt wordt, het woord var ervoor wordt gezet. Met var vertel je aan Flash dat je zojuist een nieuwe variabele bedacht hebt, die je vanaf dat moment wilt gaan gebruiken.

Dit is een goede gewoonte. In de eerste plaats voor je zelf, omdat je later terug kunt zien waar je een variabele voor de eerste keer in je script introduceerde. Ook kun je dan zien hoe je van plan was de variabele precies te noemen, en dat is handig omdat iedere programmeur vroeg of laat een typefout maakt in een variabelenaam. Dit soort fouten vind je sneller terug wanneer je snel kunt zien waar je een variabele voor de eerste keer ging gebruiken. Je hoeft je variabele slechts een maal aan te kondigen met het woord var.

We hebben het gehad over **Strings en Numbers (getallen)** Dit zijn niet de enige twee **datatypen** die Flash kent. Later in de cursus zullen we nog kennis maken met **Booleans, Arrays en Objecten (Movieclips)**

## 1.2 Nog enkele belangrijke begrippen.

### 1. Class en Objects

Soms worden de woorden Class en Object als synoniem voor DataType gebruikt. Er is wel een fundamenteel verschil. Een Class is een definitie van een datatype. (vb. alle variabelen van het MovieClip datatype hebben de volgende karakteristieken, nl a,b,c). Een Object is een instantie van een datatype.

Hieronder lees je vier zinnen die steeds hetzelfde betekenen.

Het datatype van de variabele myVariable is Number.

De variabele myVariable is a Number instance.

De variabele myVariable is a Number object.

De variabele myVariable is an instance of the Number class.

### 2. Objects

Actionscript is een object-oriented programming language. Dit is een manier om code te organiseren aan de hand van objecten. De code is gegroepeerd in groepjes functionaliteit die daarna tot één geheel worden gebracht. Symbolen die vanuit de Library op de stage geplaatst worden, zijn objecten. Een movieclip die vanuit de Library op de stage wordt geplaatst is een instance of instantie van de MovieClip class. Op deze instantie kun je met Actionscript allerlei veranderingen aanbrengen.(positie, grootte, afspelen, verbergen ed)

**Karakteristieken van elke class.(vb movieclip): properties, methods, events.**

### 3. Properties .

Een property is één van de stukjes data die samen een object vormen.

Voor een MovieClip zou dat de rotatie, de plaats (.x,.y), de grootte en de alphawaarde kunnen zijn.

Het zijn child-variables in een object

```
vb square.x = 100; verplaatst de mc square naar het coördinaat x met 100 pixels
square.rotation = triangle.rotation;
square.scaleX = 1.5; vergroot de mc square met 1.5 keer zijn huidige grootte
```

Tussen de naam van het object en de property staat een **dot operator**. Deze dot operator geeft aan dat je één van de childvariables in een object aanspreekt.

### 4. Methods ()

Een method is een action dat kan uitgevoerd worden door een object.

Voor een MovieClip zou dat kunnen zijn, afspelen, stoppen, rewind, naar een bep. frame springen enz.

```
vb. shortFilm.stop();
shortFilm.gotoAndPlay(5);
```

**De haakjes** zijn een manier om een method aan te roepen. Instructie geven aan het object om deze method, deze actie uit te voeren. Soms komen er values tussen de haakjes om de actie correct te kunnen uitvoeren. gotoAndPlay(5), de 5 staat voor frame 5 op de timeline van het MovieClipobject.

## 1.2 Nog enkele belangrijke begrippen.

### Events

We haalden in het begin van de cursus aan dat een computerprogramma niets meer is dan een aantal instructies die stap voor stap door de computer worden uitgevoerd. Actionscriptprogramma's zijn echter ontworpen om te blijven "draaien" tot een gebruiker iets doet (klikken op een button) of te wachten tot er iets anders gebeurt. (de playhead passeert een frame. Dit zijn events (gebeurtenissen) waarop de flashplayer wacht en bepaalde acties doet indien ze zich voordoen.

### Basic Event Handling

De techniek om bepaalde acties te doen, op voorwaarde dat er bepaalde events zich voordoen noemen we event handling. In deze event handlers zitten drie belangrijke elementen:

- 1. Event Source**                      vb. buttonobject (ook eventTarget genoemd)
- 2. Event**                                wat gaat er gebeuren vb mouseEvent mouse\_over
- 3. The response**                      Welke acties moet flash ondernemen als bovenstaande events zich voordoen.  
(vb naar een bepaald frame springen)

```
function eventResponse (eventObject:EventType):void
{
    actions die moeten uitgevoerd worden komen hier
}

eventSource.addEventListener(EventType.EVENT_NAME, eventResponse);
```

Deze code bevat twee delen:

### Function

Een function is de manier om verschillende actions te groeperen onder één noemer nl. de naam van de function

Deze function moet je een naam geven (hier eventResponse) en een parameter aan toevoegen (hier eventObject) en een datatype voor de parameter aangeven (hier EventType)

```
{
    actions die moeten uitgevoerd worden
}
```

### addEventListener

Eens je de event-handling function hebt geschreven moet je de eventSource (vb button) laten weten de functie wil aanroepen als het event gebeurt. Twee parameters tussen haakjes plaatsen nl soort event en naam functie. vb. (**MouseEvent.CLICK**, eventResponse);

## 1. Nog enkele belangrijke begrippen.

### praktische voorbeelden:

#### 1. Naar een bepaald frame navigeren na klikken op een button.

```
function pagina(event:MouseEvent):void
{
gotoAndPlay (5);
}
button.addEventListener(MouseEvent.CLICK, pagina);
```

#### 1. Op een button klikken om een movieclip te laten afspelen.

In dit voorbeeld is playbutton het instance van de button.  
"this" is een speciale benaming voor "het huidige object" (de movieclip)

```
function playmovie(event:MouseEvent):void
{
this.Play ( );
}
playbutton.addEventListener(MouseEvent.CLICK, playmovie);
```

#### 2. Naar een bepaald frame navigeren na klikken op een button.

```
function pagina(event:MouseEvent):void
{
gotoAndPlay (5);
}
button.addEventListener(MouseEvent.CLICK, pagina);
```

#### 3. Op een button klikken om naar een URL te navigeren.

```
function gotoAdobeSite(event:MouseEvent):void
{
var adobeURL:URLRequest = new URLRequest("http://www.adobe.com/");
navigateToURL(adobeURL);
}
button.addEventListener(MouseEvent.CLICK, gotoAdobeSite);
```

# Oefeningen met Movieclips

## Oefening 5

### Situatie

Movieclip van pulserende zwarte bol en twee knoppen. Deze zijn ook mc's.

### Events

Groene knop laat animatie starten, rode knop laat animatie stoppen.

De beide knoppen zijn ook movieclips.

Instantienamen  
btnrood, btngroen,  
zwartebol.

```
//movieclips declareren variabelen aanmaken
var zwartebol:MovieClip;
var btngroen:MovieClip;
var btnrood:MovieClip;

//nu gaan we de animatie in de mczwartebol laten stoppen
zwartebol.stop();

// we maken de mc btngroen klaar om als button gebruikt te
knnen worden. We voegen een eventlistener toe die, na het
klikken(MouseEvent.CLICK) op de mc btngroen) de functie
playbol opstart.
btngroen.addEventListener(MouseEvent.CLICK, playbol);

//de functie playbol start de mczwartebol.
function playbol(event:MouseEvent):void
{
    zwartebol.play();
}
//zelfde werkwijze voor de rode btn.
btnrood.addEventListener(MouseEvent.CLICK, playstop);

function playstop(event:MouseEvent):void
{
    zwartebol.stop();
}
```

## Oefening 6

### Situatie

Drie bollen pulseren en worden transparant bij vergroting. Bij elk van de bollen hoort een button.

(ook mc)

### Events

Bij klikken op één van de knoppen animeert alleen de bol erboven, de andere bollen stoppen.

### instantienamen

zwartebol1 zwartebol2  
zwartebol3 btngroen,  
btngeel, btnrood

```
var zwartebol1:MovieClip;
var zwartebol2:MovieClip;
var zwartebol3:MovieClip;
var btngroen:MovieClip;
var btngeel:MovieClip;
var btnrood:MovieClip;

btngroen.addEventListener(MouseEvent.CLICK, speelGroen);
function speelGroen(e:MouseEvent):void {
    zwartebol2.stop();
    zwartebol3.stop();
    zwartebol1.play();
}

btngeel.addEventListener(MouseEvent.CLICK, speelGeel);
function speelGeel(e:MouseEvent):void {
    zwartebol2.play();
    zwartebol3.stop();
    zwartebol1.stop();
}

btnrood.addEventListener(MouseEvent.CLICK, speelRood);
function speelRood(e:MouseEvent):void {
    zwartebol2.stop();
    zwartebol3.play();
    zwartebol1.stop();
}
```

## Oefening 7

### Situatie

Drie pulserende bollen met elk een bijhorende button.(allen mc's) instantienamen knop1, knop2, knop3 bol1, bol2, bol3

### Events

Als je op een button klikt stopt de bijhorende bol en wel op zijn kleinste stand. Dit is hier opgelost door te verwijzen naar een label op de timeline van de mc. bvb "klein"  
De andere twee bollen blijven pulseren

```
var knop1:MovieClip;
var knop2:MovieClip;
var knop3:MovieClip;

knop1.addEventListener(MouseEvent.CLICK, stopbol1)
    function stopbol1(e:MouseEvent):void {

        bol1.gotoAndStop("klein");
        bol2.gotoAndPlay("start");
        bol3.gotoAndPlay("start");
    }
// hier worden de labels in de movieclips aangeroepen om
de
//bollen op een bepaalde plaats te laten starten bij klik-
ken op de //buttons

knop2.addEventListener(MouseEvent.CLICK, stopbol2)
    function stopbol2(e:MouseEvent):void {

        bol1.gotoAndPlay("start");
        bol2.gotoAndStop("klein");
        bol3.gotoAndPlay("start");
    }

knop3.addEventListener(MouseEvent.CLICK, stopbol3)
function stopbol3(e:MouseEvent):void {

    bol1.gotoAndPlay("start");
    bol2.gotoAndPlay("start");
    bol3.gotoAndStop("klein");
}
```

## Oefening 8

Lamp  
met aan- en uitknop.

```
var lamp:MovieClip;
var uitKnop:MovieClip;
var aanKnop:MovieClip;

// eerst de movieclip laten stoppen
lamp.stop();

// als op de aanknop wordt geklikt wordt de functie
aanZet uitgevoerd. (event)

function aanZet(event:MouseEvent):void
{
    lamp.gotoAndStop("aan");
}

// de aanknop clickable maken
aanKnop.addEventListener(MouseEvent.CLICK, aanZet);

// bovenstaande ook voor uitknop schrijven
function uitZet(event:MouseEvent):void
{
    lamp.gotoAndStop("uit");
}

uitKnop.addEventListener(MouseEvent.CLICK, uitZet);
```

### 3. Conditionals

Tot nu toe wordt iedere opdracht die we in het actionscript panel intypen door Flash uitgevoerd. Dat is niet altijd even wenselijk. **Soms wil je dat iets alleen gebeurt als er aan een bepaalde voorwaarde wordt voldaan.** Neem bijvoorbeeld een movieclip van een lamp. Als de lamp aan staat, kunnen we hem uitdoen. We kunnen de lamp dus alleen uitdoen als er aan een voorwaarde is voldaan, nl als de lamp aan is. We moeten de lamp niet uitdoen als hij al uit is. De voorwaarde is dat de lamp brandt. De syntax van een if statement is als volgt:

```
if (voorwaarde){  
    opdracht;  
    opdracht;  
    opdracht;  
}
```

**Bijvoorbeeld:**

```
if (x > 10){  
    trace("x is groter dan 10")  
}
```

Een if-statement begint met het woord if. Daarna volgt tussen haakjes een conditie of voorwaarde waaraan moet worden voldaan, gevolgd door accolades { }.Tussen deze accolades staan de opdrachten die moeten worden uitgevoerd wanneer aan de voorwaarde is voldaan.

Wanneer in het voorbeeld de variabele x een waarde zou hebben die groter is ( > ) dan 10 dan wordt de trace opdracht uitgevoerd. Heeft x een waarde die kleiner is dan 10 dan wordt de opdracht niet uitgevoerd.

De voorwaarde bestaat doorgaans uit een test om na te gaan of een variabele een bepaalde waarde bevat. Je kunt verschillende varianten testen. We zagen al dat je kunt nagaan of een waarde groter is dan een andere waarde met (x > 10).

Andere mogelijkheden zijn:

**(y >= 10)**

heeft variabele y een waarde die groter is dan of gelijk aan 10?

**(bananen < 7)**

heeft variabele bananen een waarde die kleiner is dan 7?

**(bananen <= 7)**

heeft variabele bananen een waarde die kleiner is dan of gelijk aan 7?

**(naam == "webklus")**

heeft variabele naam een waarde die gelijk is aan de tekst "webklus"?

**Let er op dat hier een dubbel is gelijk teken (==) wordt gebruikt!**

Met een enkel = teken zouden we namelijk de waarde "webklus" in naam stoppen (assignment), en dat is niet de bedoeling.

**(naam != "webklus")**

heeft variabele naam een waarde die ongelijk is aan de tekst "webklus"?

Het **uitroepteken** moet je bij programmeren lezen als "**niet**". Niet gelijk dus, in dit geval.

## 4. Wat als?

*Er zijn ook gevallen te bedenken waarin je een aantal regels code wilt uitvoeren als er aan een voorwaarde is voldaan (zoals in een if statement) en een aantal andere opdrachten wanneer juist niet aan de voorwaarde is voldaan. In dat geval kun je gebruik maken van een if-else statement:*

De syntax van een if-else statement is als volgt:

```
if (voorwaarde){  
  opdrachten die worden uitgevoerd als  
  aan de voorwaarde is voldaan  
}  
else{  
  opdrachten die worden uitgevoerd als juist niet  
  aan de voorwaarde is voldaan  
}
```

Bijvoorbeeld:

```
if (sexe == "vrouw"){  
  trace("hallo mevrouw");  
  trace("de lucht is blauw");  
}  
else{  
  trace("hallo meneer");  
  trace("tot ziens maar weer");  
}
```

De code tussen de accolades staat steeds een stukje naar rechts. Om precies te zijn een TAB.

Wederom een goede gewoonte die de leesbaarheid van code verhoogt. Deze if statements zijn nog overzichtelijk, maar wanneer er veel regels code in staan, of zelfs een tweede if, kan het wel eens zijn dat je een sluit-accolade vergeet. Dat soort fouten is een stuk sneller op te sporen wanneer de code binnen een if-statement als zodanig te herkennen is. Dat doe je dus door iedere regel binnen een if-statement (dus tussen de accolades) een TAB naar rechts te plaatsen.

Het is tijd voor wat oefeningen om vertrouwd te raken met if-else statements. Probeer bij deze oefeningen de code zo veel mogelijk zelf in te typen, in plaats van te copy-pasten. Het zijn niet echt moeilijke oefeningen, maar wie weet maak je hier en daar een typefout, of vergeet je een aanhalingsteken. Als dat het geval is zul je een foutmelding krijgen, en dat is ook een beetje de bedoeling. Een deel van de tijd die je aan het actionscripsten bent zul je te maken hebben met foutmeldingen en het leren interpreteren van deze meldingen is een onderdeel van het leren actionscripsten. Beter daar mee te beginnen nu de code nog relatief eenvoudig is.

Probeer bij fouten de foutmeldingen die Flash je geeft goed te lezen. Doorgaans staat er ook een regelnummer bij, waar de fout is opgetreden. Kijk dus altijd eerst op die regel, en vergelijk de code die je hebt geschreven met de code in de voorbeelden. Wanneer de code op die regel klopt kan de fout ook nog wel eens in de regels er voor zijn opgetreden. Maar nooit er na. Soms geeft Flash meer dan een foutmelding. De eerste fout heeft dan een aantal andere veroorzaakt. Los je de eerste op dan verdwijnen ook de andere foutmeldingen. Lees wel alle foutmeldingen, ze geven je een totaalbeeld van wat er mis kan zijn.

## Oefening9a: Het if statement

Open de Flash ontwikkel omgeving en maak eventueel een nieuwe movie aan. In frame 1 van layer 1 zet je de volgende code:

```
var favorieteKleur:String = "groen";  
var kleur:String = "rood";  
  
if ( favorieteKleur == kleur){  
    trace("wat een mooie kleur");  
}
```

1. probeer te voorspellen wat er gebeurt en test dan het script.
  2. pas het trace statement aan, zodat de tekst wordt "wat een mooie kleur: [ingevulde kleur]"
  3. test je script
  4. pas het if-statement aan, zodat het trace commando alleen wordt uitgevoerd als de kleur juist ongelijk is aan de favorieteKleur. Pas eventueel de tekst in het trace commando daar op aan.
  5. test je script. Pas eventueel de waarde van de variabele 'kleur' aan, zodat het trace statement wel degelijk wordt uitgevoerd.
- 

## Oefening9b: Het if-else statement

Open de Flash ontwikkel omgeving en maak eventueel een nieuwe movie aan. In frame 1 van layer 1 zet je de volgende code:

```
var favorieteKleur:String = "groen";  
var kleur:String = "blauw";  
  
if ( favorieteKleur == kleur){  
    trace("wat een mooie kleur");  
}
```

1. Probeer te voorspellen wat er gebeurt en test dan het script.
2. Pas de code aan: voeg een else toe aan deze if. Binnen de else zet je de code  
`trace(" nee, dat vind ik geen mooie kleur");`
3. test je script.

Je hebt nu een beetje ervaring opgedaan met assignments en if-else statements. Deze les eindigen we met een tweetal praktijk oefeningen, waarbij we het geleerde gaan toepassen in een wat minder abstracte manier: in een eerste oefening gaan we een lamp maken die je aan en uit kunt zetten met dezelfde knop. In een tweede oefening gaan we een timeline sturen.

---

## Oefening 9c Conditional if en else lamp met switchknop en boolean

### Situatie

Bij het klikken op een button springt een lamp aan. Bij nog eens klikken gaat de lamp uit.

### instantienamen

lamp voor de mc lamp  
schakelaar voor de mc button

De movieclip lamp heeft op de timeline twee labels staan.  
De label "uit" waar de lamp uit is.  
De label "aan" waar de lamp aan is.

Zet zowel de mc lamp als mc button op de stage en  
zet onderstaande code in de eerste frame van de timeline

```
var lamp:MovieClip;
var schakelaar:MovieClip;
var knipperlicht:Boolean = false;

lamp.stop();

function aanKlikken(event:MouseEvent):void
{
    if(knipperlicht)
    {
        knipperlicht = false;
        lamp.gotoAndStop("uit");
    }
    else
    {
        knipperlicht = true;
        lamp.gotoAndStop("aan");
    }
}
schakelaar.addEventListener(MouseEvent.CLICK, aanKlikken);
```

*Het gegevenstype Boolean bestaat uit twee waarden: true en false.  
Voor variabelen van het type Boolean zijn geen andere waarden geldig.  
De standaardwaarde van een Booleaanse variabele die is gedeclareerd maar  
niet is geïnitieerd, is false.*

## Oefening 10 Conditional if en else    Timeline sturen

timeline	1	2		30		50
actions	●	●		●		●
labels		loop ●	animatie	endloop ●		home
tekst						

Met actionscript kunnen we ook de timeline sturen en de playhead laten doen wat we zelf willen. We zetten actionscripts op keyframes in de timeline, en iedere keer dat de playhead zo'n keyframe passeert wordt de action uitgevoerd.

Deze keyframes hebben nummers. (vb `gotoAndPlay(5);`) Met deze code bezoekt de playhead frame 5. Dit heeft een nadeel. Als je constructie van je timeline verandert, en je past de framenummers niet aan in je actionscriptcode, dan heb je problemen. Om dit te vermijden kun je framelabels gebruiken. Bij verschuiven van de constructie van je timeline blijven de labelnamen hetzelfde, ze staan gewoon op een ander plaats. Dit gaat je as-code niet veranderen. Gebruik van labels is dus nuttig.

Hoe plaats je labels?

Je maakt een keyframe en in het propertyvenster kun je bij label een naam invoeren. Bij enter of verder werken zie je de labelnaam in je timeline verschijnen met een vlagje erbij.

### 1. Playhead laten lopen met actionscript.

Soms willen we een animatie herhalen, bvb tot een andere flashmovie geladen is. We kunnen dit doen via actionscript.

We gaan de animatie eerst laten lopen tussen de labels loop (begin animatie) en de label endloop (einde van de animatie).

We doen dit door in de labellaag het frame met framelabel endloop te selecteren en in het actionscriptpanel (F7)

de code **`gotoAndPlay("loop");`** in te tikken. (loop is een string en moet dus tussen aanhalingstekens).

als je nu de movie test zie je dat je flash de animatie herhaald.

### 2. We kunnen de playhead ook naar een label laten springen na een aantal loops.

Als we de playhead naar een andere label willen laten springen na een aantal loops (dat we zelf bepalen), moeten we eerst weten hoeveel keer de animatie al is afgespeeld.

Dit gaan we doen door de waarde van het aantal loops op te slaan in een variabele.

We selecteren keyframe 1 in de timeline en schrijven de volgende code.

**`var count:Number = 1;`**

In keyframe 30 schrijven we code die iedere keer 1 optelt bij onze "count" als de playhead frame 30 aandoet. Dus ieder keer als er een loop is gedaan.

**`count++;`** ( ++ wil zeggen één toevoegen aan huidige waarde.)

### 3. Het aantal loops weergeven in een tekstveld.

We hebben een animatie die loopt, we hebben een variabele die het aantal loops bijhoudt. We hebben echter geen zicht op het aantal keer er reeds geloopt is. We gaan het aantal loops vastleggen in een tekstveld.

#### Werkwijze

Maak met je teksttool in een aparte tekstlaag een tekstveld.  
Geef in je propertyvenster een instancename aan dit tekstveld (info\_txt.)  
Kies ook in datzelfde propertyvenster voor een dynamisch tekstveld.  
(dynamische tekstvelden zijn te manipuleren met actionscript)

Maak in je actionslayer een keyframe in frame 2 (F6) en voeg in je actionspanel de code toe.

```
info_txt.text=String(count);
```

Hiermee hebben we de textproperty van het tekstveld info\_txt gelijk gezet met de waarde van de variabele count. Maw de inhoud van het tekstveld zal worden gevuld met een cijfer die het aantal loops weergeeft.

*In een tekstveld kan alleen literal tekst geplaatst worden. Aangezien we in onze variabele count met een datatype Number werken, moeten we die veranderen naar een String zodat dit nummer in een tekstveld kan weergegeven worden, vandaar String met waarde gelijk aan de waarde van count. Dus String(count).*

### 4. Conditional statement gebruiken om timeline te controleren.

Een conditional statement wordt gebruikt om Flash te laten reageren als aan een bepaalde voorwaarde is voldaan. In onze oefening gaan we de playhead naar het label home laten springen na de vierde loop. Dit is de voorwaarde, er moeten vier loops volbracht zijn vooraleer de playhead naar de label home mag springen. Of

**Als het aantal loops is bereikt spring dan naar label "home", anders, keer maar terug naar label "loop".**

#### Werkwijze

Selecteer frame 30 in de actionslayer en delete de aanwezige code. Voer de volgende code in.

```
if(count>3){  
  
    gotoAndStop("home");  
  
}else{  
  
    gotoAndPlay("loop");  
}
```

Een conditional statement kan voor veel situaties gebruikt worden:

- als een movie is geladen, toon iets..
- als een vraag goed is beantwoord, ga naar de volgende vraag..
- is een level gespeeld, update de score en ga naar het volgende level..
- is een product naar het karretje gesleept, bereken het totaal en verstuur het product.

## 5. Update de tekst op de homepage

selecteer frame 50 en voeg de code toe in het actionspanel

```
info_txt.text = "Welkom op de homepage";
```

Het dynamische tekstveld wordt nu gevuld met deze tekst ipv het aantal loops.

---

### Oefening 11 Buttons en navigatie, concateneren, taalwissel met buttons.

Zoals rees in de vorige lessen gezien kunnen we actionscript gebruiken zodat buttons reageren op events, gebeurtenissen in de Flashmovie (vgb mouseClick).

#### 1. Buttons maken om te navigeren.

Gebruik voor deze oefening je vorig gemaakte flashbestand op 10.

Voeg buttons toe op de volgende manier:

- Maak een nieuwe laag boven de tekstlaag. Geef naam buttons.
- Maak keyframe in deze buttonslaag op frame 2.
- Open het componentenvenster (window-components)
- Sleep hieruit twee instances van het buttonsymbool, en zet ze rechtsonderaan in je document (in buttonslaag op frame 2).
- Selecteer je eerste button en open het components inspectorvenster. Verander daarin het label(naam) van de button in Home.
- Doe hetzelfde voor de tweede button en verander de label in Flash Support.
- Met deze twee buttons gaan we respectievelijk navigeren naar de homepage en naar een webpagina met flashsupport.
- Selecteer de buttons op de stage in geef aan elk een instantienaam in het propertievenster. Deze instantienamen gebruikt AS3.

Homebtn = **home\_btn**

Flash Supportbtn = **help\_btn**.

Nu zijn onze buttons klaar. Ze staan op de stage, en hebben een instantienaam gekregen. Nu wordt het tijd om ze met AS3 te doen werken.

## 2. Functions toevoegen om te reageren op buttonkliks.

Nu gaan we de buttons doen werken. Selecteer daarvoor frame 2 in de actionslaag en voeg de volgende code toe in het actionspanel.

```
home_btn.addEventListener(MouseEvent.CLICK, goHome);  
help_btn.addEventListener(MouseEvent.CLICK, goHelp);
```

Nu heb je de buttons voorzien van een EventListener. De buttons "luisteren" nu naar een gebeurtenis (mouseEvent) om daarna te kunnen reageren.

- Als er op de home\_btn wordt geklikt moet een functie goHome in werking treden, die ons naar de homepage brengt.
- Als er op de help\_btn wordt geklikt moet een functie goHelp in werking treden, die ons naar de site van adobe flash support brengt.

De function ziet er als volgt uit voor de homebtn:

```
function goHome(e:MouseEvent):void{  
    gotoAndStop("home");  
}
```

De function ziet er als volgt uit voor de helpbtn:

```
function goHelp(e:MouseEvent):void{  
    navigateTOURL(newURLRequest(http://www.adobe.com/support/flash));  
}
```

## 3. We gaan ook een restart button toevoegen.

-We zetten een nieuwe keyframe op de buttonslaag op de hoogte van de homeframe. Hierdoor kunnen we een nieuwe instance maken van het buttonsymbool home. We kunnen daardoor dus met dezelfde button verschillende actions uitvoeren. We kunnen zelfs een andere naam voorzien (hier restart)

- Selecteer de homebtn op de stage
- Open componentsinspector en geef een nieuwe label(naam) nl. restart.
- Geef ook een nieuwe instantienaam in het propertievenster nl. restart\_btn.

voeg nu de code toe in het homeframe van de actionslaag.

```
restart_btn.addEventListener(MouseEvent.CLICK, goStart);  
  
function goStart(e:MouseEvent):void{  
    gotoAndPlay("loop");  
    count=1;  
}
```

De code voegt een listener toe dat luistert naar een mouseclick op de restartbtn. Flash reageert dan door de functie uit te voeren. De playhead springt terug naar het begin van de animatie nl looplevel. Omdat de animatie terug opnieuw moet beginnen moeten we de variable count een waarde 1 geven.

Bij testen van de movie zien we dat de home\_btn op de homepage in restart\_btn is aangepast. In het animatiegedeelte van de oefening heb je wel de home\_btn.

## 4. Functions toevoegen om te reageren op buttonkliks.

Nu gaan we de buttons doen werken. Selecteer daarvoor frame 2 in de actionslaag en voeg de volgende code toe in het actionspanel.

```
home_btn.addEventListener(MouseEvent.CLICK, goHome);  
help_btn.addEventListener(MouseEvent.CLICK, goHelp);
```

Nu heb je de buttons voorzien van een EventListener. De buttons "luisteren" nu naar een gebeurtenis (mouseEvent) om daarna te kunnen reageren.

- Als er op de home\_btn wordt geklikt moet een functie goHome in werking treden, die ons naar de homepage brengt.
- Als er op de help\_btn wordt geklikt moet een functie goHelp in werking treden, die ons naar de site van adobe flash support brengt.

De function ziet er als volgt uit voor de homebtn:

```
function goHome(e:MouseEvent):void{  
    gotoAndStop("home");  
}
```

De function ziet er als volgt uit voor de helpbtn:

```
function goHelp(e:MouseEvent):void{  
    navigateTOURL(newURLRequest(http://www.adobe.com/support/flash));  
}
```

## 3. We gaan ook een restart button toevoegen.

-We zetten een nieuwe keyframe op de buttonslaag op de hoogte van de homeframe. Hierdoor kunnen we een nieuwe instance maken van het buttonsymbool home. We kunnen daardoor dus met dezelfde button verschillende actions uitvoeren. We kunnen zelfs een andere naam voorzien (hier restart)

- Selecteer de homebtn op de stage
- Open componentsinspector en geef een nieuwe label(naam) nl. restart.
- Geef ook een nieuwe instantienaam in het propertievenster nl. restart\_btn.

voeg nu de code toe in het homeframe van de actionslaag.

```
restart_btn.addEventListener(MouseEvent.CLICK, goStart);  
  
function goStart(e:MouseEvent):void{  
    gotoAndPlay("loop");  
    count=1;  
}
```

De code voegt een listener toe dat luistert naar een mouseclick op de restartbtn. Flash reageert dan door de functie uit te voeren. De playhead springt terug naar het begin van de animatie nl looplevel. Omdat de animatie terug opnieuw moet beginnen moeten we de variable count een waarde 1 geven.

Bij testen van de movie zien we dat de home\_btn op de homepage in restart\_btn is aangepast. In het animatiegedeelte van de oefening heb je wel de home\_btn.

## 5. Inhoud tekstveld veranderen. Concateneren.

Het tekstveld toont nu alleen met een cijfer hoeveel loops er verlopen zijn. We gaan die informatie wat aangenamer maken door er een zin aan toe te voegen.

voor het ogenblik staat er op frame 2 de volgende code

```
info_text.text=String(count);
```

Verander de bovenstaande code in de onderste. (concateneren)

```
info_text.text=" The animation plays" + String(count) + " x."
```

## 6. Dmv. buttons taalkeuze voorzien in tekstveld.

We gaan de gebruiker laten kiezen welke taal hij wil zien in het tekstveld waar we het aantal loops in voorzien.

### Werkwijze

- Maak drie buttons aan voor Engels, Duits en Nederlands. (components)
- Selecteer frame 1 van actionslayer en voeg de volgende code onder de reeds bestaande.

```
var language:String="English";
```

*Deze variabele voorziet Engels als basistaal, en gaat ons later in staat stellen om die variabele te gebruiken om de taalkeuze te controleren.*

- in frame2 van actionslayer zet de volgende code:

```
function setLanguage():void{  
    if (language == "English"){  
        info_text.text=" The animation plays" + String(count) + " x."  
    }  
}
```

Bovenstaande functie wordt aangeroepen op frame 2 en zal controleren als "language" op English is "gezet". Als aan de voorwaarde is voldaan dan zal de engelse tekst in het dynamische tekstveld geplaatst worden.

We gaan dit ook doen voor de andere talen.

```

function setLanguage():void {
    if(language == "English") {
        info_txt.text = "The animation has played " + String(count) + " x.";
    }

    else if(language == "German") {
        info_txt.text = "Die Animation wurde " + String(count) + "x ab
gespielt.";
    }

    else if(language == "Dutch") {
        info_txt.text = "De animatie speelt" + String(count) + "x." ;
    }
}
setLanguage();

```

### **Opgepast**

language == "English" == vergelijkt twee waarden met elkaar

language = "English" = zet waarde 1 gelijk aan waarde 2

Nu gaan we de drie "taalbuttons" op de stage zetten (op frame 2)

Instantienamen english\_btn, german\_btn en dutch\_btn in het propertievenster toevoegen.

Voeg de onderstaande code toe in frame 2 actionslayer:

```

english_btn.addEventListener(MouseEvent.CLICK, setEnglish);
german_btn.addEventListener(MouseEvent.CLICK, setGerman);
dutch_btn.addEventListener(MouseEvent.CLICK, setDutch);

```

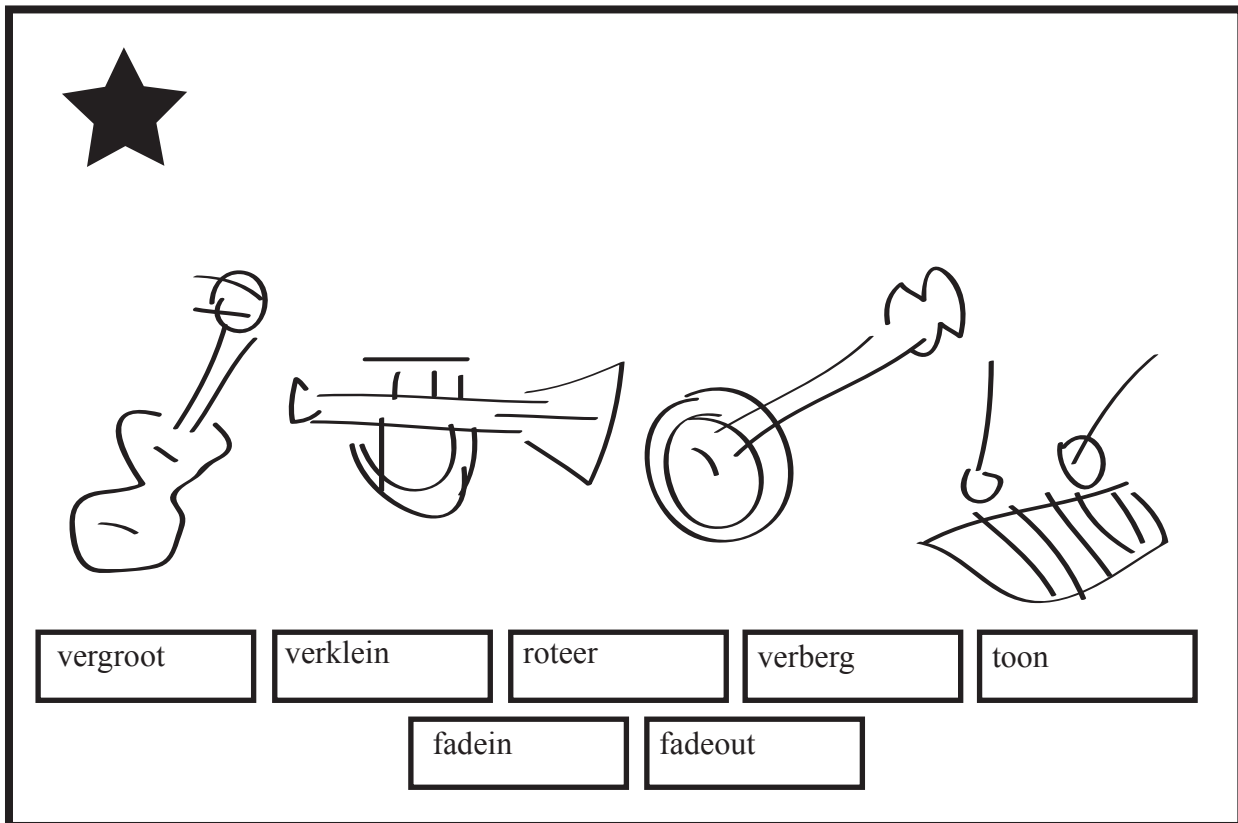
```

function setEnglish(e:MouseEvent):void {
    language = "English"
    setLanguage();
}
function setGerman(e:MouseEvent):void {
    language = "German"
    setLanguage();
}
function setDutch(e:MouseEvent):void {
    language = "Dutch"
    setLanguage();
}

```

## Oefening 12 Movieclip-properties aanpassen met buttons

beginsituatie



Begin een nieuw flashdocument en construeer de bovenstaande situatie. Geef aan elk element (movieclips) een gepaste instancename.

De gebruikte instancenamen zijn:

voor de ster	star_mc
voor de viool	viool_mc
voor de trompet	trompet_mc
voor de banjo	banjo_mc
voor klokspel	klokspel_mc

De instancenamen voor de buttons:

voor de vergrootbtn	groter
voor de verkleinbtn	verklein
voor de rotatiebtn	draai
voor de verbergbtn	weg
voor de toonbtn	toon
voor de fade-in	klaar
voor de fade-out	mist

### 1. Movieclip properties controleren met actionscript

basis syntax `instancename_vande_movieclip . propertyname = value.`

**vb. `star_mc . rotation = 90 ;`**

bekijk de volgend pagina voor de meest gebruikte properties

Propertes	waarden	beschrijving
x	- oneindig +oneindig	horizontale positie
y	-oneindig +oneindig	verticale positie
rotation	- 180° to 180 °	roteren
alpha	0(transparant) tot 1(opaque)	transparantie
scaleX	- oneindig +oneindig	Horizontale schaling
scaleY	- oneindig +oneindig	Verticale schaling
visible	true(zichtbaar)-false(onzichtbaar)	Zichtbaarheid (visibility)

## 2. Propertywaarde van movieclip veranderen.

Met de timeline en actionscriptpanel zichtbaar selecteer frame 1 in de actionslayer en typ de volgende code

```
star_mc.x = 275;
```

test de movie, je ziet dat je ster meer naar het midden van je stage is verplaatst.

typ nu de volgende code

```
star_mc.rotation = 90;
star_mc.alpha = .5;
```

test nu je movie, je ziet dat de ster gerooteerd en ook transparant geworden is.

## 3. Propertywaarden vermeerderen en verminderen.

We kunnen ook in plaats van een bepaalde waarde aan een property te geven, ook een bepaalde waarde toevoegen aan een reeds gestaaende waarde.

Verwijder alle code uit frame 1 en schrijf de volgende code.

```
star_mc.addEventListener(MouseEvent.CLICK, rotateStar);

function rotateStar(e:MouseEvent):void{
    star_mc.rotation += 5;
}
```

de += gebruiken is gewoon een shortcut om te zeggen  
star\_mc.rotation = star\_mc.rotation + 5;

**of neem de huidige waarde van het object links en voeg er de waarde rechts bij**

je kunt dit ook met -= doen natuurlijk

test nu je movie, iedere keer je op ster klikt, roteert hij 5 graden.

#### 4. Property van een movieclip animeren met het "Event-Frame" event.

We weten nu dat we de x-positie van een movieclip kunnen veranderen. Als we dit snel genoeg doen en achter elkaar herhalen, dan kunnen we een animatie maken.

De enter-frame event is daarvoor ideaal.

De enterframe event is actief zolang de movie speelt. Zelfs is er maar één frame. De werkwijze om zo'n enterframe event te maken is quasi dezelfde als met een MouseEvent. Dus eerst de listener toevoegen en dan de functie schrijven.

Aangezien we hier geen button nodig hebben om iets te doen, kunnen we de listener direct toevoegen aan de timeline. Voeg de onderstaande code toe bij frame 1. Zet die onder de reeds bestaande code.

```
addEventListener(Event.ENTER_FRAME, starMove);  
  
function starMove (e: Event):void{  
    star_mc += 2;  
}
```

test je movie, je ziet dat je ster over je stage beweegt van links naar rechts. De ster zal dan ook verdwijnen aan de rechterkant en in de "oneindigheid" verdwijnen.

We moeten dus nog code schrijven om de ster terug naar het beginpunt te krijgen eenmaal hij de rechterrاند van de stage heeft bereikt.

Hier gaan we een **conditional statement** voor gebruiken. Er moet een voorwaarde voldaan zijn om de ster terug naar het beginpunt van de animatie te zetten, hij moet namelijk de rechterkant bereikt hebben.

Verander de bovenstaande code

```
addEventListener(Event.ENTER_FRAME, starMove);  
  
if(star_mc.x < stage.stageWidth){  
    star_mc.x += 2;  
}else{  
    star_mc.x = 0;  
}  
}
```

De ster animeert tot aan de rechtse kant van de stage, als hij aan die voorwaarde heeft voldaan, keert hij terug naar het beginpunt van de animatie.

## 5. Selecteren van een movieclip, selectie weergeven in een tekstveld.

In de volgende fase van de oefening gaan we de properties van de movieclips veranderen aan de hand van buttons. Vb. de button "vergroot" zal bij klikken de geselecteerde movieclip met een aantal pixels vergroten.

Werkwijze

- Voeg de volgende variabelen toe aan frame 1 boven bestaande code.  
(instancename banjo\_mc)

```
var instrument:MovieClip = banjo_mc;
```

- Voorzie nu een dynamisch tekstveld (instancename tekstveld)

```
tekstveld.text = "banjo is geselecteerd";
```

- Nu gaan we de listeners en functions toevoegen aan de mc's zodat ze geselecteerd kunnen worden.

```
viool_mc.addEventListener(MouseEvent.CLICK, selectViool);  
banjo_mc.addEventListener(MouseEvent.CLICK, selectBanjo);  
trompet_mc.addEventListener(MouseEvent.CLICK, selectTrompet);  
klokspel_mc.addEventListener(MouseEvent.CLICK, selectKlokspel);
```

nu maken we de functies aan

```
function selectViool(e:MouseEvent):void{  
    instrument = viool_mc;  
    tekstveld.text = "viool is geselecteerd";  
}
```

```
function selectBanjo(e:MouseEvent):void{  
    instrument = banjo_mc;  
    tekstveld.text = "banjo is geselecteerd";  
}
```

```
function selectTrompet(e:MouseEvent):void{  
    instrument = trompet_mc;  
    tekstveld.text = "trompet is geselecteerd";  
}
```

```
function selectKlokspel(e:MouseEvent):void{  
    instrument = klokspel_mc;  
    tekstveld.text = "klokspel is geselecteerd";  
}
```

test nu je movie, ieder keer je op een instrument klikt verandert de tekst in het tekstveld.

## 6. Movieclip properties (eigenschappen) met buttons aanpassen

- eerst gaan we de listeners toevoegen aan de buttons

```
grow_btn.addEventListener(MouseEvent.CLICK, grow);
```

- nu de functie schrijven die de property gaat aanpassen.

```
function grow (e:MouseEvent):void{  
    instrument.scaleX += .5;  
    instrument.scaleY += .5;  
}
```

we doen dit voor alle buttons met hun specifieke propertyverandering

```
groter.addEventListener(MouseEvent.CLICK, groot);  
kleiner.addEventListener(MouseEvent.CLICK, klein);  
draai.addEventListener(MouseEvent.CLICK, rond);  
weg.addEventListener(MouseEvent.CLICK, geen);  
toon.addEventListener(MouseEvent.CLICK, wel);  
klaar.addEventListener(MouseEvent.CLICK, fel);  
mist.addEventListener(MouseEvent.CLICK, flou);
```

```
function groot(e:MouseEvent):void{  
    instrument.scaleX += .1  
    instrument.scaleY += .1;  
}
```

```
function klein(e:MouseEvent):void{  
    instrument.scaleX -= .1  
    instrument.scaleY -= .1;  
}
```

```
function rond(e:MouseEvent):void{  
    instrument.rotation += 5;  
}
```

```
function geen(e:MouseEvent):void{  
    instrument.visible = false;  
}
```

```
function wel(e:MouseEvent):void{  
    instrument.visible = true;  
}
```

```
function fel(e:MouseEvent):void{  
    instrument.alpha = 1;  
}
```

```
function flou(e:MouseEvent):void{  
    instrument.alpha = 0.6;  
}
```

TEST NU JE MOVIE

## Oefening 13 Tweening met actionscript.

Wat is een tweening in feite. Dit is niet meer dan de properties van een bepaald object veranderen over een bepaalde tijd, over een bepaald aantal frames.

We moeten dus een property selecteren dat we willen veranderen en daarna een begin- en eindsetting bepalen voor de animatie van die property.

vb. Een property van een object kan zijn x-positie zijn (horizontale beweging).

We kunnen die x-positie een beginsetting geven (vb 0 pixels)

en een eindsetting geven (vb 400 pixels).

We kunnen de animatie bepalen over een tijd van 5 seconden.

Het object gaat dan van 0 naar 400 pixels op de x-as binnen een tijd van 5 seconden bewegen.

De code daarvoor zou zijn:

```
var rocketTween:Tween=new Tween (rocket, "x", none.easeOut, 0 , 400, 5 , true);
```

We stockeren een nieuwe tweeninstance in een variabele. **var rocketTween**

het is een datatype Tween **:Tween**

het is een nieuwe tween **= new Tween**

deze tween heeft de volgende parameters

naam van het te tweenen object	<b>rocket</b>	
property dat zal geanimeerd worden is een horizontale beweging	<b>"x"</b>	is een string
easingtype (in dit geval geen easing)	<b>none.easeOut</b>	
startwaarde	<b>0</b>	pixels
eindwaarde	<b>400</b>	pixels
tijdspanne	<b>5</b>	
in seconden	<b>true</b>	

**! zonder true is de tijdspanne default in milliseconden !**

dus alles parameters die de tween controleren staan tussen haakjes

## 8. Importeren van de classes tweening en easing.

Alle actionscriptcode gebruikt in de vorige oefeningen maakten deel uit van de "built in" flash classes. Bvb de movieclip "classes" zijn automatisch beschikbaar in flash.

**De tween en easing classes moeten eerst geïmporteerd worden.**

**Daarvoor gebruiken we een import statement.**

**Dus eerst importeren van tween en easing classes, dan pas tweening maken.**

1. We zetten onderstaande code bovenaan ons scriptvenster om de tweening en easingclasses in flash te importeren. Dit moet maar éénmaal gebeuren.(vb frame 1)

```
import fl.transitions.Tween;  
import fl.transitions.easing.*;
```

2. We maken de tween (keyframe maken en code toevoegen in scriptvenster)

```
var bewegingEen:Tween = new Tween(bol,"x",None.easeOut,25,525,1,true);  
"hier dus een horizontale beweging van instancenname bol van 25 pixels tot 525 pixels gedurende één seconde zonder easingeffect."
```

```
var bewegingEen:Tween = new Tween(bol,"x",None.easeOut,25,525,1,true);
```

```
var bewegingalfa:Tween = new Tween(bol,"alpha",None.easeOut,1,0,2,true);
```

```
var bewegingklein:Tween = new Tween(bol,"scaleX",None.easeOut,1,1,1,true);
```

```
var bewegingTwee:Tween = new Tween(bol,"y",None.easeOut,25,375,1,true);
```

```
var beweginggrote:Tween = new Tween(bol,"rotation",None.easeOut,0,360*3,1,true);
```

```
var bewegingkleintwee:Tween = new Tween(bol,"scaleX",None.easeOut,1,1,1,true);
```

```
var bewegingAlfaTwee:Tween = new Tween(bol,"alpha",None.easeOut,0,1,2,true);
```

```
var beweginggroot:Tween = new Tween(bol,"scaleX",None.easeOut,1,2,1,true);
```

### **praktische toepassing oefening 13 (stage 550 op 400 pixels fps 10)**

Tracht de bovenstaande codes te "ontcijferen" en probeer oefening 13 (bekijk het voorbeeld op oefeningenflash .swf op de site)

Bestudeer goed de oefening en schrijf op een blaadje papier wat er eigenlijk gebeurt en wanneer. Tracht je bevindingen om te zetten in een flash movie.

De bovenstaande codes kun je gebruiken als voorbeeld (eventueel aanpassen ?)

### **Tips**

Je ziet dat de bol eerst horizontaal , dan vertikaal, dan terug horizontaal en op het laatst terug vertikaal beweegt. Op de plaatsen waar van richting wordt veranderd, daar zet je op de timeline een keyframe. Op deze keyframes moet de code voor de tween geplaatst worden.

## Oefening 14 Werken met actionscript file. (.as) Custom actionscriptclass aanmaken.

Tot nu toe hebben we in onze oefeningen de actionscriptcodes binnen onze flashmovie geschreven. Je kan ook je actionscripts in een "externe file" schrijven en die dan "koppelen" aan je eigenlijke flashmovie. Dit wordt veel gedaan bij grote flashproducties waar zoveel code moet geschreven worden dat het op den duur onoverzichtelijk wordt. Dan worden delen van de code in aparte ".as" -files geschreven. Dan moet er in de eigenlijke flashmovie veel minder code worden gezet.

### 1. Hoe make we een actionscriptfile.

- in flash kies voor bestand-nieuw.
- in het "nieuw bestand dialoogkader" kies voor Actionscript file, en klik ok. (wat je nu ziet is een flashbestand zonder timeline en alleen een actionscriptpanel.)
- save dit bestand in een map "ASfile", en geef het de naam Ellipse.as.  
De eigenlijke flashmovie gaan we straks ook in dezelfde map saveen.

### 2. Basisstructuur van een Actionscript 3.0 class file.

De movieclip class is een "built in" actionscriptclass dat alles in zich heeft om te werken met movieclips.(class zie p.7)In flash zijn er verschillende classes text, soud, video enz.

Als we met externe actionscriptfiles werken moeten we die classes eerst importeren in de file.

Op de eerste lijn van ons Ellipse.as file schrijven we

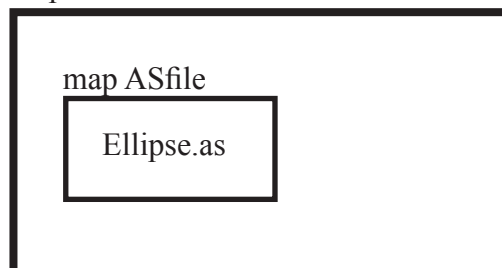
```
package{  
  
}
```

Dit moet je zien als een beschrijving van het "pad" naar je actionscriptfile (in dit geval Ellipse.as. Aangezien we de flashmovie die hoort bij het Ellipse.as file in dezelfde map gaan plaatsen (ASfile) kunnen we volstaan met package alleen.

*Moesten we de Ellipse.as file in een map "ASfile" plaatsen die op zijn beurt in de map "AS" zou zitten dan moeten we de onderstaande code schrijven*

```
package AS.AS file {  
  
}
```

map AS



Alle code voor het actionscriptfile komt tussen de accolades van package.

1. Eerst moeten we de classes importeren die gaan gebruikt worden in de movie. Als we code schrijven in een fl. -bestand zijn die classes reeds aanwezig. Als je externe as-files schrijft dan moet je die eerst importeren.

```
import flash.display.MovieClip;
```

## 2. Nu moeten we een nieuwe Actionscriptclass maken

```
public class Ellipse extends MovieClip {  
  
}
```

De code in de Ellips.as -file ziet er nu zo uit

```
package{  
  
import flash.display.MovieClip;  
  
public class Ellipse extends MovieClip{  
  
}  
  
}
```

**Alle code voor de nieuwe class komt binnen de binnenste accolades.**

### ***2.a. Nieuwe termen public en extends***

**Public:** de term public is een access modifier. Dit wil zeggen dat public aangeeft dat deze nieuwe class gebruikt kan worden in elke andere flashfile.

**Extends:** wil zeggen dat de huidige nieuwe class alle mogelijkheden van de parentclass (hier in dit geval MovieClipclass) heeft plus alle nieuwe mogelijkheden van de nieuwe class.

In deze class gaan we een simpele cirkel tekenen. Deze cirkel gaat dan ook alle mogelijkheden hebben van een movieclip nl. roteren, positie veranderen, alpha ed.

de functie ziet er zo uit:

```
public function Ellipse (w:Number=40,h:Number=40,color:Number=0xff0000){  
    graphics.beginFill(color);  
    graphics.drawEllipse(0,0,w,h);  
    graphics.endFill();  
}
```

De volledige code ziet er nu als volgt uit:

```
package{  
  
    import flash.display.MovieClip;  
  
    public class Ellipse extends MovieClip{  
  
public function Ellipse (w:Number=40,h:Number=40,color:Number=0xff0000){  
    graphics.beginFill(color);  
    graphics.drawEllipse(0,0,w,h);  
    graphics.endFill();  
    }  
    }  
}
```

---

### Verklaring:

#### 1. **public function Ellipse (w:Number=40,h:Number=40,color:Number=0xff0000){**

De waarden tussen haakjes zijn "default" waarden voor de cirkel.  
(rode cirkel van 40 op 40 pixels.)

#### 2. **graphics.beginFill(color); graphics.drawEllipse(0,0,w,h); graphics.endFill();**

Deze drie parameters zullen gebruikt worden om nieuwe instances van de defaultcirkel weer te geven. (vb. groene cirkel van 10 op 10 pixels)

#### 3. **public function Ellipse (w:Number=40,h:Number=40,color:Number=0xff0000){ graphics.beginFill(color); graphics.drawEllipse(0,0,w,h); graphics.endFill();**

### Deze functie is de "constructor" functie:

In as-files kunnen vele functies voorkomen maar altijd maar één constructorfunctie.

Deze heeft de naam van de class zelf... hier dus.... Ellipse.

Men roept de constructor functie aan als men een instance wil aanmaken.

Zien we verder in de oefening.

**We hebben nu een nieuwe class gemaakt in een extern as-file. we gaan die nu gebruiken in een flashmovie.**

### 3. Flashmovie aanmaken.

We beginnen een nieuwe flashdocument en gaan instances van het Ellipse.as file maken.

- Bestand - Nieuw
- kies voor Actionscript 3.0.
- Maak een actionslayer aan en selecteer frame 1 .

Hierin gaan we een instance maken van de Ellipse class.

- Schrijf de volgende code:

```
var ellipse:Ellipse = new Ellipse();
```

In flash maken we een nieuwe instance door **new** te gebruiken.

Om de instance toe te voegen aan de stage schrijven we de volgende code.

```
addChild(ellipse);
```

- Save je movie, nu zie een kleine rode cirkel op de stage.

#### 3.a Waarom addChild();

Alle objecten die op de stage staan staan zijn display objects en staan ook in de display list als ze zichtbaar zijn op de stage.

Als een object met actionscript gemaakt wordt, dan bestaat het wel maar dan staat het nog niet op de stage. Om een object in de display list te zetten, en daardoor ook onstage moet je de methode addChild() aanroepen.

### 4. Cirkel dupliceren met beweging cursor.

Nu hebben we een rode cirkel op de stage.(instance van ellipse class)

Voor het volgende deel van de oefening gaan we die cirkel gebruiken om te kunnen tekenen met de muis. (soort brush aanmaken).

We doen dit door eerst een eventListener toe te voegen voor de stage.

```
stage.addEventListener(MouseEvent.MOUSE_MOVE, makeShapes);
```

nu maken we de functie

```
function makeShapes(e:MouseEvent):void{  
    var ellipse:Ellipse = new Ellipse()  
    addChild(ellipse);  
}
```

Iedere keer we de muis bewegen dupliceert de cirkel maar op dezelfde plaats. We moeten nog aangeven dat de cirkels mee moeten bewegen met de muis. We doen dit op de volgende manier.

```
ellipse.x = mouseX
```

```
ellipse.y = mouseY
```

maw, de horizontale positie van de cirkel moet mee met de horizontale positie van de muis. En dit ook voor de verticale positie

de volledige code ziet er nu zo uit

```
stage.addEventListener(MouseEvent.CLICK, makeShapes);
```

```
function makeShapes(e:MouseEvent):void{
```

```
    var ellipse:Ellipse = new Ellipse()  
    addChild(ellipse);  
    ellipse.x = mouseX  
    ellipse.y = mouseY
```

```
}
```

Save en test je movie, dan zie je dat je kan tekenen met je cursor.

## **5. De parameters van een Ellipse instance veranderen ( de grootte en kleur van de rode cirkel veranderen)**

Nu kun je tekenen met rode cirkels. Dat komt door de parameters die in de constructor functie aanwezig waren in het as.file

```
public function Ellipse (w:Number=40,h:Number=40,color:Number=0xff0000){  
    graphics.beginFill(color);  
    graphics.drawEllipse(0,0,w,h);  
    graphics.endFill();
```

We kunnen deze waarden veranderen en dus ook de grootte en kleur van de cirkel. We doen dit op de volgende manier. We geven bij new Ellipse nieuwe waarden voor de cirkel in.

```
function makeShapes(e:MouseEvent):void{
```

```
    var ellipse:Ellipse = new Ellipse(10, 10, 0x00FF00)  
    addChild(ellipse);  
    ellipse.x = mouseX  
    ellipse.y = mouseY
```

```
}
```

Save en test je movie, je ziet dat er nu groene cirkels getekend worden.

Iedere keer we de muis bewegen dupliceert de cirkel maar op dezelfde plaats. We moeten nog aangeven dat de cirkels mee moeten bewegen met de muis. We doen dit op de volgende manier.

```
ellipse.x = mouseX
```

```
ellipse.y = mouseY
```

maw, de horizontale positie van de cirkel moet mee met de horizontale positie van de muis. En dit ook voor de verticale positie

de volledige code ziet er nu zo uit

```
stage.addEventListener(MouseEvent.MOUSE_MOVE, makeShapes);
```

```
function makeShapes(e:MouseEvent):void{
```

```
    var ellipse:Ellipse = new Ellipse()  
    addChild(ellipse);  
    ellipse.x = mouseX  
    ellipse.y = mouseY
```

```
}
```

Save en test je movie, dan zie je dat je kan tekenen met je cursor.

## **5. De parameters van een Ellipse instance veranderen ( de grootte en kleur van de rode cirkel veranderen)**

Nu kun je tekenen met rode cirkels. Dat komt door de parameters die in de constructor functie aanwezig waren in het as.file

```
public function Ellipse (w:Number=40,h:Number=40,color:Number=0xff0000){  
    graphics.beginFill(color);  
    graphics.drawEllipse(0,0,w,h);  
    graphics.endFill();
```

We kunnen deze waarden veranderen en dus ook de grootte en kleur van de cirkel. We doen dit op de volgende manier. We geven bij new Ellipse nieuwe waarden voor de cirkel in.

```
function makeShapes(e:MouseEvent):void{
```

```
    var ellipse:Ellipse = new Ellipse(10, 10, 0x00FF00)  
    addChild(ellipse);  
    ellipse.x = mouseX  
    ellipse.y = mouseY
```

```
}
```

Save en test je movie, je ziet dat er nu groene cirkels getekend worden.

## 6. Stoppen met tekenen (de makeShapes-functie aan en uit zetten)

We kunnen nu tekenen, maar we kunnen nog niet stoppen en weer beginnen met tekenen. Daarvoor moeten we de functie makeShapes kunnen "uitzetten" en weer "aanzetten".

Hoe doen we dit:

- In het actionscriptpanel zet je de cursor boven alle ander code en geef een aantal returns.
- We gaan nu twee nieuwe Listeners toevoegen die luisteren naar mouse down en mouse up events.

```
stage.addEventListener(MouseEvent.MOUSE_DOWN, startDrawing);  
stage.addEventListener(MouseEvent.MOUSE_UP, stopDrawing);
```

de mousedown event zal de functie startDrawing doen werken  
de mouseup event zal de functie stopDrawing doen werken

functie startDrawing

```
function startDrawing(e:MouseEvent):void{  
    stage.addEventListener(MouseEvent.MOUSE_MOVE, makeShapes);  
}
```

functie stopDrawing

```
function stopDrawing(e:MouseEvent):void{  
    stage.removeEventListener(MouseEvent.MOUSE_MOVE, makeShapes);  
}
```

Het resultaat van deze fucnties is dat de functie die de cirkels tekent bij beweging van de muis alleen zal werken bij klikken van de muis, en alleen zal stoppen bij het loslaten van die muis.

Save en test je movie, je ziet nu dat je kunt starten en stoppen met tekenen.

## 7. Flash willekeurig kleur laten kiezen.

Om een willekeurig nummer te kiezen in flash moet je de random methode kiezen van de Math class. De syntax is

```
Math.random ();
```

Deze code zal een willekeurig getal produceren tussen 0 en 1. Als je willekeurige getallen wilt produceren(genereren) bvb tussen 0 en 50 moet je het Math.random getal vermenigvuldigen met 50.

```
Math.random() * 50;
```

Math.random gebruiken om kleuren van de cirkels willekeurig te laten veranderen.

- Eerst een variabele maken om een kleurwaarde te kunnen stockeren.

```
var color:Number;
```

- zet bij de startDrawing functie de volgende code

```
color = math.random()*0xFFFFFFFF;
```

de functie ziet er dan zo uit

```
function startDrawing(e:MouseEvent):void{  
    stage.addEventListener(MouseEvent.MOUSE_MOVE, makeShapes);  
    color.math.random()*0xFFFFFFFF;  
}
```

Iedere keer er geklikt wordt zal er nu een andere willekeurig kleur "gekozen" worden.

## **8. Willekeurig kleur toevoegen aan de cirkels.**

```
var ellipse:Ellipse = new Ellipse(10, 10, color);
```

De volledige code ziet er als volgt uit:

```
var color:Number;  
  
stage.addEventListener(MouseEvent.MOUSE_DOWN, startDrawing);  
stage.addEventListener(MouseEvent.MOUSE_UP, stopDrawing);  
  
function makeShapes(e:MouseEvent):void{  
    var ellipse:Ellipse = new Ellipse(10,10,color);  
    addChild(ellipse);  
    ellipse.x = mouseX;  
    ellipse.y = mouseY;  
}  
  
function startDrawing(e:MouseEvent):void{  
    stage.addEventListener(MouseEvent.MOUSE_MOVE, makeShapes);  
    color=Math.random()* 0xFFFFFFFF;  
}  
  
function stopDrawing(e:MouseEvent):void{  
    stage.removeEventListener(MouseEvent.MOUSE_MOVE, makeShapes);  
}
```

## 9. Een custom cursor toevoegen.

Het zou leuk zijn om de cursor nu te veranderen in een penseel of potlood.

- Teken een penseel of potlood in Flash of Illustrator.
- Maak van deze figuur een movieclip (F8).
- Geef die movieclip een instancenaam. vb cursor.
- Voeg dan de code toe aan frame 1.  
Bvb onder of boven de reeds bestaande code.

```
Mouse.hide();  
stage.addEventListener(MouseEvent.MOUSE_MOVE, follow);  
function follow(evt:MouseEvent){  
    cursor.x = mouseX;  
    cursor.y = mouseY;  
}
```

```
Mouse.hide();  
// verbergt de cursor ( mouse.show() toont de cursor)  
  
stage.addEventListener(MouseEvent.MOUSE_MOVE, follow);  
// event listener luistert naar een bewegende cursor  
// dus als de cursor beweegt moet de functie follow in actie schieten.
```

```
function follow(evt:MouseEvent){  
    cursor.x = mouseX;  
    cursor.y = mouseY;  
  
// als de cursor beweegt (mouseEvent)wordt de horizontale positie "x"  
// van de movieclip gelijk gezet met de horizontale beweging van de muis  
//Hetzelfde geldt voor de verticale beweging "y".
```

## Oefening 15. Externe swf's laden met component List en UILoader Gallery maken

### Constructie

- maak een map aan genaamd oefvijftien.
- maak in die map twee andere mappen genaamd Afbeeldingen en Teksten.

De map Afbeeldingen bevat vier foto's. (hond en kat, zoek op internet)  
hond1.jpg , kat2.jpg, hondthumb.jpg, katthumb.jpg  
Maak van beide foto's een beeld op grootte 400 op 300 pixels en 150 op 150 pixels. (thumbnails en grote te laden foto)

De map Teksten bevat twee tekstbestanden. Elk met een stukje tekst over een hond en een kat. Respektievelijk hond1.txt en kat2.txt.(max 5 regels)

- in map oefvijftien is nog aanwezig:

basis.swf(800op 600px),

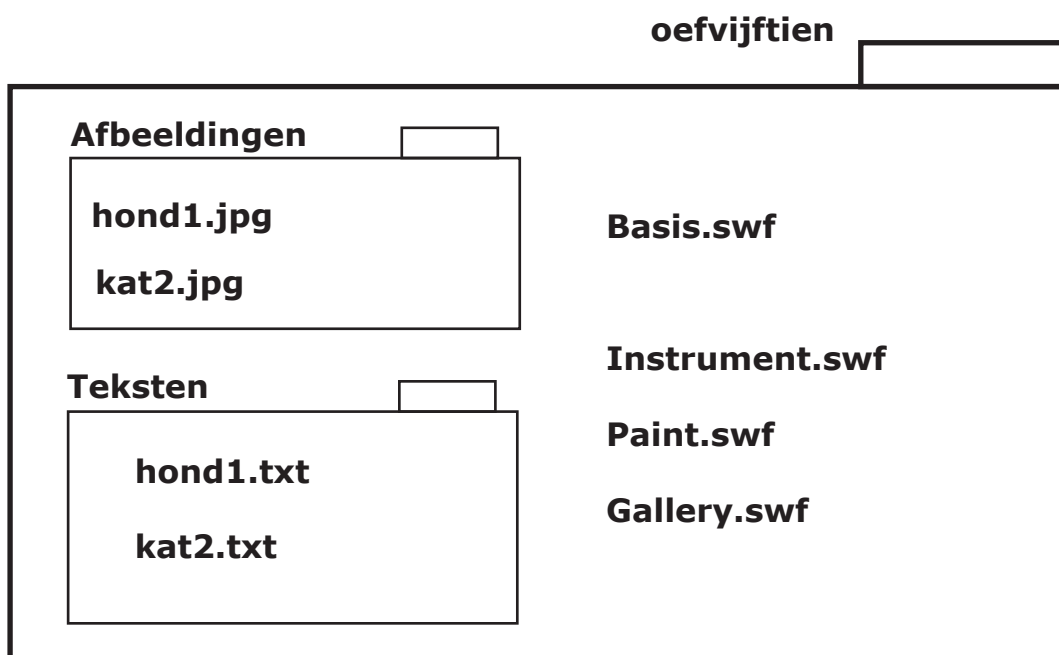
instrument.swf(550 op 400 pixels) ,

paint.swf (550 op 400 pixels)

gallery.swf.(550 op 400 pixels)

De instrument.swf is een flashmovie met kleine animatie van een instrument.  
Voor paint.swf met animatie met schilderij.

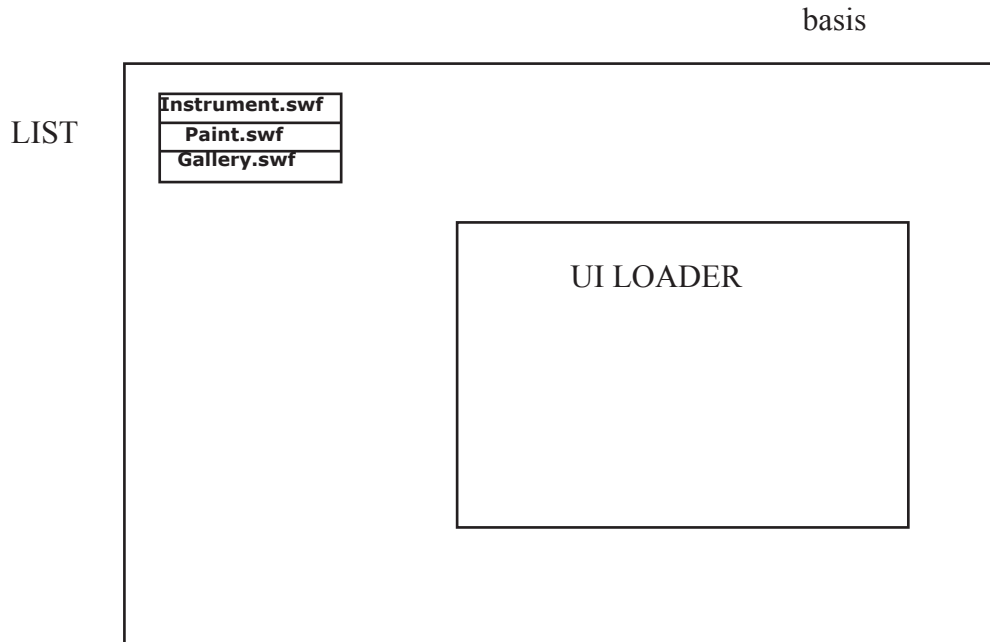
De gallery.swf is een flashmovie met thumbnails die bij klikken grotere foto's laden.  
Over de constructie van deze movie later meer informatie.



## Na het opzetten van de nodige mappen en bestanden gaan we de eigenlijke basismovie aanmaken. (800 op 600 pixels)

Het is de bedoeling om de instrument.swf, paint.swf, gallery.swf in de basismovie te laden bij het klikken op een link aangemaakt met het component LIST.

Het "venster" waarin deze swf's geladen worden is het component UI LOADER



## Werkwijze

### 1. Instantie van component LIST op de stage zetten

- Maak nieuwe basismovie aan. (800 op 600 pixels)
- Maak in de basismovie twee layers aan nl. componenten en actions.
- Selecteer frame 1, maak keyframe F6 open het componentenvenster (window-component)
- Sleep de LIST-component naar de stage.(componentenlayer)  
Met dit component gaan we een lijst met links maken waarmee je de instrument.swf, de paint.swf en gallery.swf kunt laden in de UIloader.
- Selecteer de LIST op de stage en geef de instancename **loadList** in het propertyvenster.
- Nog in het propertyvenster geef de LIST de waarde x = 30, y = 150, w = 140 h= 60. Deze waarden bepalen de positie en de grootte van het component op de stage.
- Selecteer nu de LIST en **open het Componenten Inspector Panel.**(window-component inspector panel)
- Selecteer daarin de **dataprovieder parameter** en klik op het **vergrootglas**. Dit opent een venster waarin je de lijst kunt maken. Met labels en eraan gekoppelde bestanden.
- Klik daarvoor 3x op de plus, en geef bij label " instrument" en bij data "insturment.swf". Selecteer daarvoor label in het venster en typ het label in het veld voor label. Doe dit ook voor de andere te laden swf-files.
- klik ok.

Nu heb je de List volledig klaar met de namen van de links, en de gekoppelde swf-bestanden.

## 2. Instantie van component UI LOADER op de stage plaatsen.

We gaan de UI LOADER component gebruiken om de externe swf's te plaatsen op de stage. (Later gaan we nog een UI LOADER gebruiken in de gallery.swf ).

- in componentenlaag frame 1 selecteren.
- Sleep de UI LOADER uit het componentenvenster naar de stage.
- Geef een instantienaam **loadWindow** in het propertyvenster.
- Geef ook de volgende properties: x = 200, y = 135, w = 550, h = 400.

## 3. Een CHANGE eventListener toevoegen aan de LISTcomponent. (laden swf's).

Als een gebruiker op een label klikt in de List (instrument, paint of gallery) wordt een event CHANGE uitgevoerd. Om de bijpassende swf te laden moeten we nu nog een functie maken.

- Zet in frame 1 van de actionslaag en in het actionspanel de volgende code.

```
loadList.addEventListener(event.CHANGE, loadFile);  
  
function loadFile (e:Event):void{  
  
}
```

deze functie wordt aangeroepen iedere keer op een label in de list wordt geklikt.

### OPGEPAST

Als je gewoon een swf wil laden in de uiloader dan is de syntax

```
loadWindow.source = "instruments\swf"
```

instancename loader                      pad naar het bestand.

### MAAR

In deze oefening gaan we swf's laden via een LIST. Dan is de code enigzinds anders. We kunnen een code schrijven die direct werkt voor alle swf's die moeten geladen worden. (Denk aan de labels en de bijhorende swf's)

de functie is als volgt

```
loadList.addEventListener(event.CHANGE, loadFile);  
  
function loadFile (e:Event):void{  
  
loadWindow.source = e.target.selectedItem.data;  
}
```

**e.target.** is de lijst

**selectedItem.** is het label

**data;** is de swf

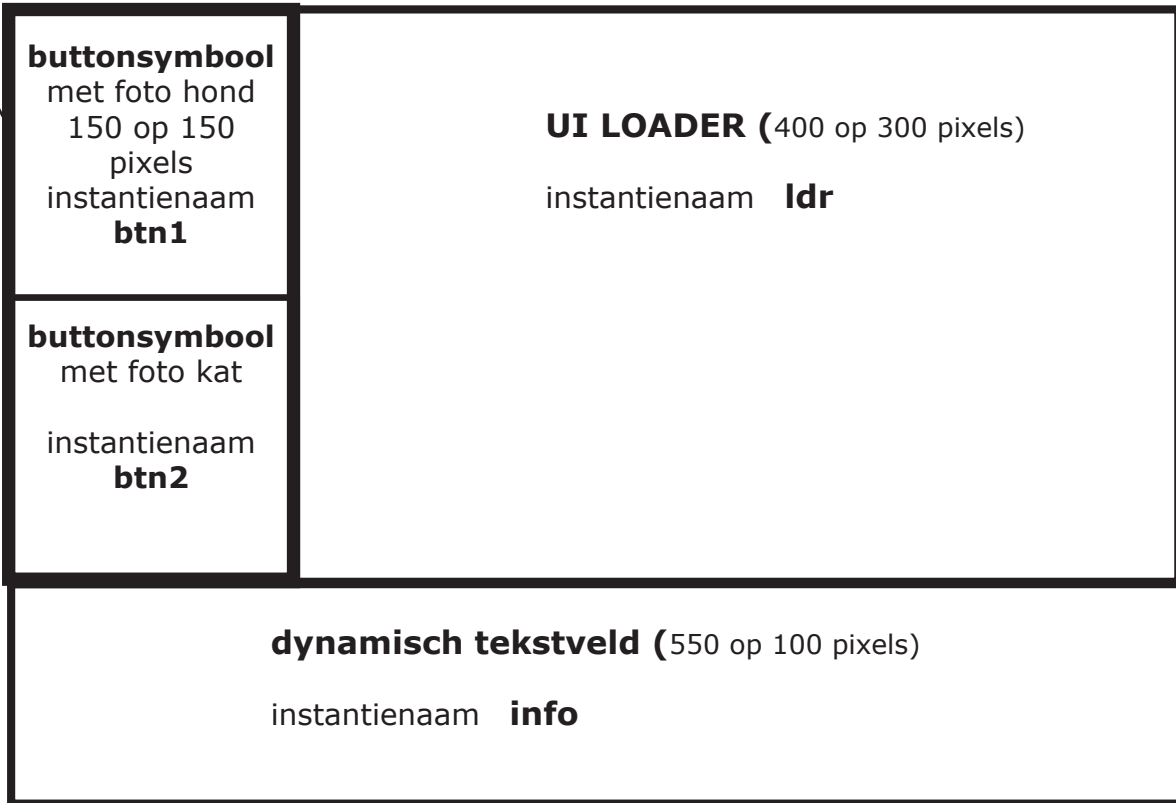
Save en test je movie. Als alles goed is laden de swf's in de loader (uitgezonderd gallery.swf)

#### 4. Aanmaken van de gallery.swf.

De gallery.fla heeft de volgende constructie.

**movieclip met thumbs 150 op 300 pixels**  
 instantienaam **thumbs\_mc**

**550 px**



#### gallery.fla lagen

<b>actions</b>	
<b>thumbs</b>	<b>movieclip met thumbs</b> instantienaam <b>thumbs_mc</b>
<b>loader</b>	<b>component UI loader</b> instantienaam <b>ldr</b>
<b>tekst</b>	<b>dynamisch tekstveld</b> instantienaam <b>info</b>

Als we in de gallery.swf op de btn1 en btn2 klikken wordt de grote illustratie van de hond (hond1.jpg)of de kat (kat2.jpg) in de UI Loader geladen (ldr). Ondertussen wordt ook het passende stukje tekst geladen in het dynamisch tekstveld(hond1.txt, kat2.txt),(info).

#### Werkwijze

1. We moet dus eerst zoals altijd de eventListeners op de buttons voorzien.

```
btn1.addEventListener(MouseEvent.CLICK, ldr1);
btn2.addEventListener(MouseEvent.CLICK, ldr2);
```

## 2. Afbeeldingen kat en hond te laden in de UI LOADER.

```
function ldr1(e:Event):void{  
    ldr.source = "../afbeeldingen/hond1.jpg";  
}
```

Met bovenstaande functie hebben we de code voor het laden van de afbeelding hond gemaakt

## 3. Tekstblokje laden in het dynamisch tekstveld.(info)

De teksten hond1.txt en kat2.txt moeten vanuit de map teksten geladen worden in het dynamisch tekstveld. We kunnen dit niet doen zoals we dat met de afbeeldingen hebben gedaan. Zo'n UI LOADER kan geen echte tekstelementen laden.

Daarvoor gebruikt Flash de URLLOADER class.

- Als we die url loader class willen gebruiken moeten we er een instance van maken. We doen dat met een variabele.

```
var loader:URLLoader = new URLLoader();
```

- nu schrijven we de functie

```
function textload (file:String, color:uint){  
    loader.load(new URLRequest(file));  
    info.backgroundColor = color;  
}
```

- bij de functie om te laden als we op btn 1 klikken voegen we nu toe

```
textload("../teksten/hond1.txt", 0xAAFFAA);
```

- de volledige functie die wordt aangeroepen als we op btn 1 klikken is dus

```
function ldr1(e:Event):void{  
    ldr.source = "../afbeeldingen/hond1.jpg";  
    textload("../teksten/hond1.txt", 0xAAFFAA);  
}
```

**Deze constructie verdient wat uitleg.**

**- De textload-functie doet twee zaken:**

**ten eerste**     **loader.load(new URLRequest(file));**  
                  hiermee zal de file (hond1.txt) geladen

**ten tweede**   **info.backgroundColor = color;**  
                  aangeven dat achtergrondkleur van het tekstveld in een kleur moet.

## - function ldr1

In deze functie wordt bij iedere klik op btn 1 de functie textload aangeroepen. Hier staat het pad naar het file hond1.txt in , en ook het gewenste hexidemale kleurnummer

```
textload("../teksten/hond1.txt", 0xAAFFAA);
```

## Tip

Ieder keer je AS gebruikt om externe bestanden te laden, is het goed om zeker te zijn dat de bestanden geladen zijn, voor je ze gebruikt of weergeeft.

## Werkwijze

```
loader.addEventListener(Event.COMPLETE, toonText);
```

```
function toonText (e:Event){
```

```
    info.text = (loader.data);  
    info.background = true;  
    info.border = true;  
    info.borderColor = 0x333333;
```

```
}
```

### volledige script

```
thumbs_mc.btn1.addEventListener(MouseEvent.CLICK, ldr1);  
thumbs_mc.btn2.addEventListener(MouseEvent.CLICK, ldr2);  
  
function ldr1(e:Event) {  
    ldr.source = "../oefvijftien/afbeeldingen/hond1.jpg";  
    textLoad("../oefvijftien/teksten/hond1.txt", 0xAAFFAA);  
}  
  
function ldr2(e:Event) {  
    ldr.source = "../oefvijftien/afbeeldingen/kat2.jpg";  
    textLoad("../oefvijftien/teksten/kat2.txt", 0xCCCCEE);  
}  
  
var loader:URLLoader = new URLLoader();  
  
    function textLoad(file:String, color:uint) {  
        loader.load(new URLRequest(file));  
        info.backgroundColor = color;  
    }  
  
loader.addEventListener(Event.COMPLETE, displayText);  
function displayText(e:Event) {  
    info.text = (loader.data);  
    info.background = true;  
    info.border = true;  
    info.borderColor = 0x333333;  
}
```