# Extra Oefening Xml Technologie Animation

Dit is een extra oefening op animation, bewegende xaml elementen. Het uiteindelijke resultaat vind je in de file *animation.avi*. Bekijk die avi eerst. Je moet deze oefening in Java programmeren.

Animation is een zeer complexe materie. Voor deze oefening houden we het redelijk simpel. We werken met een Canvas. We bouwen hier een sjabloon op in Java, dat je later kan aanpassen om complexere animations te programmeren.

Het eerste child van het Canvas ziet er zo uit:

<Canvas.Triggers>

<EventTrigger RoutedEvent="Canvas.Loaded">

<EventTrigger.Actions>

<BeginStoryboard>

<Storyboard BeginTime="0" Duration="0:0:12">

X

X

X

X

</Storyboard>

</BeginStoryboard>

</EventTrigger.Actions>

</EventTrigger>

</Canvas.Triggers>

* Eerst definiëren we de mogelijke triggers van een animation op het canvas.
* We werken met een event trigger, en de animation zal starten wanneer het canvas geladen is.
* Dan komen de actions die moeten worden uitgevoerd na het event.
* Die actions worden gedefinieerd in een BeginStoryBoard, dat kan je vergelijken met een scenario.
* Dat scenario bevat in dit geval 1 onderdeel, het StoryBoard, dat begint op tijdstip 0 en 12 seconden blijft draaien (“hh:mm:ss”).

Schrijf nu eerst een method in Java die dit element maakt en returnt. Geef de 3 attribuutwaarden mee als argument, het zijn allemaal Strings.

De animation zelf definieer je in een element DoubleAnimation. Double staat hier voor het datatype. We gaan een bepaalde Double laten variëren in de tijd. Dit element komt op de plaats van de X, als children van Storyboard.

De volgende attributen zijn nodig voor de animation :

Storyboard.TargetName De naam van het element dat geanimeerd moet worden.

Storyboard.TargetProperty Het attribuut dat geanimeerd moet worden.

From De beginwaarde van het attribuut.

To De eindwaarde van het attribuut.

AutoReverse Boolean die op het einde van de animation ze in de andere richting terug opstart.

Duration De duur van 1 animation in “hh:mm:ss” formaat

RepeatBehavior Hoe lang de animation moet herhaald worden.

Schrijf nu een method in Java die dit element maakt en returnt. Geef de waarden van de 7 attributen mee als String argument. We zullen deze method 4 keer aanroepen, omdat we 4 ellipsen hebben, elk met hun eigen animation. Hier zijn de Strings die je (letterlijk) moet meegeven voor de 4 elementen :

Storyboard.TargetName e1

Storyboard.TargetProperty (Canvas.Top)

From 0

To 400

AutoReverse true

Duration 0:0:1

RepeatBehavior Forever

Storyboard.TargetName e2

Storyboard.TargetProperty (Canvas.Top)

From 0

To 400

AutoReverse true

Duration 0:0:2

RepeatBehavior Forever

Storyboard.TargetName e3

Storyboard.TargetProperty (Canvas.Top)

From 0

To 400

AutoReverse true

Duration 0:0:3

RepeatBehavior Forever

Storyboard.TargetName e4

Storyboard.TargetProperty (Canvas.Left)

From 0

To 350

AutoReverse true

Duration 0:0:3

RepeatBehavior Forever

Schrijf nu een method die de 4 DoubleAnimation elementen maakt met de juiste argumenten, en ze als child aan het element Storyboard hangt, op de plaats van de X-en.

Tot nu toe hebben we de animation gedefinieerd. Tenslotte moeten we nog de ellipsen zelf op het Canvas plaatsen, en linken met de animations.

Schrijf hiervoor een method die een Ellipse element maakt en returnt. Geef de attribuut waarden mee als argument. Hier volgen de attributen van het element :

x:Name De naam van de animation (let op : extra namespace)

Opacity Dit is het tegenovergestelde van transparency (denk daar maar eens over na)

Fill De kleur

Height De hoogte

Width De breedte

Canvas.Left Het aantal pixels vanaf de linkerkant van het canvas

Canvas.Top Het aantal pixels vanaf de top van het canvas

Het attribuut Name dat in de X namespace is gedefinieerd zorgt voor de link tussen de animation en de ellipse.

Nu moet je de laatste method 4 maal aanroepen om de ellipsen te maken. Gebruik de volgende waarden voor de argumenten :

Ellipse 1

x:Name e1

Opacity 0.5

Fill Red

Height 100

Width 100

Canvas.Left 50

Canvas.Top 20

Ellipse 2

x:Name e2

Opacity 0.5

Fill DarkSlateGray

Height 100

Width 100

Canvas.Left 170

Canvas.Top 20

Ellipse 3

x:Name e3

Opacity 0.5

Fill Blue

Height 100

Width 100

Canvas.Left 290

Canvas.Top 20

Ellipse 4

x:Name e4

Opacity 0.5

Fill LightBlue

Height 50

Width 100

Canvas.Left 0

Canvas.Top 20

Tenslotte moet je deze 4 ellipsen aan het Canvas hangen. That’s it ! PROBEER JE RESULTAAT UIT IN KAXAML.