Programmeerproject: Eindverslag Fase 3

Alexandre Kahn
alexkahn@vub.ac.be
0500067
Faculteit Wetenschappen en Bio-ingenieurswetenschappen
Vrije Universiteit Brussel

15 mei 2016

Inhoudsopgave

1 Inleiding

Dit is het verslag van de laatste fase. We moesten er voor zorgen dat ons spel af was.
In de vorige versie moest het spel al extra spelelementen hebben, nu moest alles er zijn. De highscore wordt bijgehouden en opgeslagen. De score en de highscore moeten op het spelbord getekend worden. Er moeten power-ups zijn die verschillende zaken aanpassen in het spel, en na enkele seconden gedeactiveerd worden.

2 Abstracte Data Types

Het schip-ADT is exact hetzelfde gebleven, het kogel-adt is licht aangepast zodat er verschillende soorten kogels kunnen bestaan. In het alien-ADT waren er ook kleine veranderingen, zo wordt er aan het vloot-ADT meegegeven dat het zich moet opkuisen als er een alien geraakt is. De ADT's die met het menu te maken hebben zijn net als het spel-adt vooral opgekuist. Tot slot zijn er nieuwe ADT's voor de score en voor de power-ups.

2.1 Alien-ADT

De aliens worden allemaal apart aangemaakt en zijn objecten op hun eigen. Om er meerdere op te roepen is er het `Vloot-ADT`.
Elk alien-object zal zijn eigen positie hebben en bijhouden, alsook de kleur en hoeveel levens deze nog heeft. Het aantal levens is aan de kleur gelinkt.
Door elk hun eigen positie bij te houden kan het vloot-adt gemakkelijk de coördinaten te veranderen.

```
1    (maak-alien-adt <x-positie> <y-posititie> <kleur> <teken-adt>)
2
```

Message	Parameter	Resultaat
x	/	Geeft de positie van het opgevraagde x-coördinaat terug.
y	/	Geeft de positie van het opgevraagde y-coördinaat terug.
x!	nieuwe-x	Geeft het object een nieuwe x-positie.
y!	nieuwe-y	Geeft het object een nieuwe y-positie.
levens	/	Geeft het aantal levens terug.
levens!	getal	Past het aantal levens aan.
teken!	teken-adt	Geeft aan het teken-ADT mee dat het, het alien-object moet tekenen.
delete!	teken-adt	Geeft aan het teken-ADT mee dat het alien-object verwijderd moet worden.
geraakt!	kogel-adt teken-adt vloot-adt score-adt kogels-adt huidige-tijd spel	Deze message neemt een leven van de alien weg. Indien deze er geen meer heeft wordt de alien gedeletet en de vloot opgekuist.

2.2 Vloot-ADT

Bij aanmaak van de vloot moet er meegegeven worden hoeveel aliens men moet hebben. Deze worden dan allemaal in een lijst opgeslagen. Deze begint leeg en wordt dan aangevuld met alien-objecten. Als er meer dan een bepaald aantal zijn zullen deze paars zijn, die daarna zullen groen zijn en de laatste rij is altijd geel. Momenteel is dit zodat alles zo automatisch mogelijk gebeurt.
Er wordt dan aangevuld tot als het maximum voor een rij bereikt wordt. Hiervoor wordt dan de `afstand-tussen-2` berekend. Voor de volgende rijen gebeurt dan hetzelfde
Verder is er een functie om over de vloot te lopen, `loop-over-vloot`. Deze is zeer belangrijk voor de collision-detection, want er moet voor elke alien gekeken worden of deze geraakt wordt.
De beweging van de aliens wordt hier ook meegegeven. Er wordt altijd gecheckt of er een botsing gebeurt met een zijkant van het scherm en indien dit het geval is wordt de richting omgedraaid. Ook wordt er bijgehouden welk de laagste alien is en of deze de onderkant van het scherm raakt. In dat geval wordt het spel herstart. Er zijn ook functies om de vloot te versnellen en te vertragen.

```
1  (maak-vloot-adt <aantal> <teken-adt>)
2
```

Message	Parameter	Resultaat
maak-aliens!	Aantal pos-y	Vult de lege lijst aan met nieuwe aliens. En laat het teken-ADT nieuwe alien-tiles aanmaken.
teken!	teken-adt	Zorgt dat elk alien-object zichzelf tekent.
loop-over-vloot	functie	Zorgt dat een bepaalde functie gemapt wordt over alien-objecten.
beweeg!	/	Zorgt dat elk alien-object zich verplaatst.
vloot	/	Geeft lijst met al de aliens terug.
verwijder-vloot!	/	Zet de lijst met alle aliens terug op null.
stop!	/	Pauzeert de beweging van de aliens.
herstart!	/	herstart de beweging van de aliens.
delete-dode-aliens	/	Haalt de dode aliens uit de lijst.
maak-alien-tiles!	/	Maakt voor elke alien een tile aan in het teken-adt.

2.3 Schip-ADT

Het schip-adt maakt een schip aan. Deze zal over de onderrand van het scherm moeten bewegen en kogels kunnen afschieten.
De `y-positie` zal altijd 1 zijn, de onderrand van het scherm. In het begin zal de `x-positie` steeds 0.5, of het midden van het venster zijn. De `snelheid` of het aantal stappen waarmee het schip zich zal voortbewegen is 0.3, dit zorgt voor een mooie vloeiende beweging en een gemakkelijk bestuurbaar schip.
De `beweeg!`-functie zal afhankelijk van de meegegeven toetsenbord-input het

schip recht of links verplaatsten. Echter indien het zich al in de rand van het venster bevindt zal het niet verder bewegen, dit om te vermijden dat de raket van het spelbord zou verdwijnen.

```
(maak-schip-adt)
```

Message	Parameter	Resultaat
x!	nieuwe-x	Zorgt ervoor dat de x-positie van het schip aangepast wordt. Moet enkel voor x bestaan, aangezien de y-positie steeds 1 blijft.
x	/	Geeft de x-coördinaat van het schip terug.
beweeg!	richting	Zorgt afhankelijk van de meegegeven richting voor een verplaatsing naar die kant.
teken!	teken-adt	Geeft aan het teken-adt mee om het schip te tekenen.

2.4 Kogel-ADT

Het kogel-ADT wordt niet meer vanaf het begin van het spel aangemaakt, aangezien er nu meerdere van deze ADT's zullen zijn. Ze zullen beheerd worden door een adt dat op het vloot-adt lijkt. Elke kogel zal nog steeds zijn eigen type, staat en snelheid hebben en zal ook zelf zien of het een alien raakt. Ook de beweging zit in dit ADT en zal nagaan of de kogel zich tegen een rand bevindt.

```
(maak-kogel-adt posx type)
```

Message	Parameter	Resultaat
x!	nieuwe-x	Zorgt ervoor dat de x-positie van de kogel aangepast wordt.
y!	nieuwe-y	Zorgt ervoor dat de y-positie van de kogel aangepast wordt.
x	/	Geeft de x-coördinaat van de kogel terug.
y	/	Geeft de y-coördinaat van de kogel terug.
beweeg!	teken-adt	Zorgt voor de verplaatsing en kijkt of de kogel tegen de rand komt.
teken!	teken-adt	Geeft aan het teken-adt mee om het schip te tekenen.
staat?	/	Geeft de staat van de kogel terug.
schietlus	teken-adt vloot-adt	Gaat na of de kogel een alien raakt. en indien nodig delete deze de kogel en de alien
type	/	Geeft het type van de kogel terug
geraakt!	alien alien	Delete de alien, zet de kogel op inactief en geeft mee aan het teken-adt om deze te deleten

2.5 Kogels-ADT

Dit ADT zorgt om de meerdere kogels in goede banen te leiden en op een correcte manier aan te spreken. Het doel is om alle kogels in een lijst op te slaan. Dit

gebeurt door telkens de nieuwe kogel voor de al bestaande te consen. Het heeft functies om een nieuwe kogel te maken, een functie om functies op alle kogels uit te voeren, zorgt ervoor dat alles getekend wordt en dat alle kogels kunnen bewegen. In de `beweeg!` functie wordt er ook nagegaan of de kogel een alien raakt. Er is ook een functie die inactieve kogels verwijderd.

```
(maak-kogels-adt)

```

Message	Parameter	Resultaat
beweeg!	teken-adt vloot-adt	Zorgt dat elke kogel moet bewegen en nagaan of deze botst.
teken!	teken-adt	Geeft aan de kogels mee dat ze zichzelf moeten tekenen.
maak-kogel!	posx, teken-adt	Maakt een nieuw kogel-adt aan en plaatst deze in de lijst. Ook wordt er in het teken-adt een nieuwe kogel aangemaakt.
kogels	/	Geeft de lijst met kogels terug.
delete-inactieve-kogels	/	Haalt de inactieve kogels weg uit de lijst.
volgende-kogel!	type	Bepaalt het type van de volgende kogel.

2.6 Teken-ADT

Het `teken-ADT` zorgt ervoor dat alle objecten getekend worden. Het maakt hiervoor gebruik `Graphics.rkt`.
Bij het aanmaken van dit ADT, wordt er een venster met zwarte achtergrond getekend. Hierop komen dan verschillende lagen,`'make-layer`, voor de objecten. Deze worden op hun laag getekend als tegels, `'make-tile`. Voor het schip, het menu, de how-to en de rotator is dit simpel, er moet slechts een tegel gemaakt worden. De aliens en kogel zitten daarentegen in lijsten. Voor de beide is dit zodat er meerdere getekend kunnen worden.
Zowel de kogels als de aliens zijn te verwijderen, dit door in de lijst de correcte tegel weg te halen. Dat kan aan de hand van `neem-iets`. De functie zoekt het juiste object en geeft deze dan terug.
Het tekenen van de tiles gebeurt bijna altijd op dezelfde manier, zo wordt indien nodig de juiste tegel gezocht, dan worden de correcte coördinaten berekend, de x-waarde gecentraliseerd en de tegel op deze positie getekend.
Er zijn ook functies die alle sprites hertekenen en verwijderen indien er bijvoorbeeld een menu getekend moet worden.

```
(maak-adt-teken <titel> <hoogte> <breedte>)

```

Message	Parameters	Resultaat
set-toets-functie!	fun	Zorgt dat de toetsenbordinput ingelezen wordt.
set-spel-lus-functie!	fun	Zorgt ervoor dat het speelbord steeds geupdatet wordt.
teken-schip!	schip-adt	Zorgt ervoor dat het schip getekend wordt.
teken-alien!	alien-adt	Zorgt ervoor dat de alien getekend wordt.
teken-kogel!	kogel-adt	Zorgt ervoor dat de kogel getekend wordt.
teken-menu!	menu-adt	Zorgt ervoor dat het menu getekend wordt.
teken-rotator!	rotator-adt	Zorgt ervoor dat de rotator getekend wordt.
teken-how-to!	how-to-adt	Zorgt ervoor dat de how-to getekend wordt.
teken-score!	score-adt	Zorgt ervoor dat de score en high-score getekend worden.
teken-power-up!	power-up-adt	Zorgt ervoor dat de power-up getekend wordt.
herteken-spelelementen	/	Zorgt dat alle spelementen hertekend worden.
nieuwe-alien!	alien-adt	Maakt een nieuwe alien-tile en voegt deze toe aan de lijst.
nieuwe-kogel!	kogel-adt	Maakt een nieuwe kogel-tile en voegt deze toe aan de lijst.
maak-rotator!	rotator-adt	Voegt de rotator-tile toe aan de laag.
maak-menu!	menu-adt	Voegt de menu-tile toe aan de laag.
maak-how-to	how-to-adt	Voegt de how-to-tile toe aan de laag.
delete-alien!	alien-adt	Zoekt de correcte alien-tile en verwijderd deze.
delete-kogel!	kogel-adt	Zoekt de correcte kogel-tile en verwijderd deze.
delete-schip!	/	Delete de schip-tile van het scherm.
delete-menu!	/	Delete de menu-tile van het scherm.
delete-rotator!	/	Delete de rotator-tile van het scherm.
delete-how-to!	/	Delete de how-to-tile van het scherm.
delete-power-up!	/	Delete de power-up-tile van het scherm.
verwijder-spelelementen	/	Delete het schip, alle aliens en alle kogels.
reset-vloot!	/	Haalt alle aliens weg uit de vlootlijst, haalt ook de tiles weg
voeg-alien-tile-toe!	alien-adt	Voegt de tile, die bij het correcte ADT hoort toe.

2.7 Spel-ADT

Dit ADT bevat de eigenlijke spellus. Er is de functie, `start` om het spel te starten. Verder zijn er `pauze`, om het spel te pauzeren en het menu te tekenen, `play`, om de gameloop te starten, `howto`, om de tutorial te tonen en `nieuw-level` om een nieuw level te starten. Er is een `spel-lus-functie-play` voor de spellus en ook een `spel-lus-functie-pauze` die niets doet, voor in het menu. Verder zijn er `toets-naar-spel`-functies voor zowel de spellus, het menu, als de how-to pagina.

```
1  (maak-adt-spel)
2
```

Message	Parameter	Resultaat
start	/	Dit is de entry point die ervoor zorgt dat de spellus en hierbij het hele spel start.
pauze!	/	Zet het spel op pauze.
restart!	/	Pauzeert het spel, delete de aliens en maakt er nieuwe aan.
nieuw-level!	/	Verhoogt het level en maakt meer en nieuwe aliens aan.

2.8 Menu-ADT

Dit ADT dient om het menu weer te geven en is zeer simpel. Het heeft een vaste positie en moet instaat zijn het menu te installeren, dit is alle spelelementen weghalen en het menu te tekenen, als het omgekeerde bij het `uninstall` commando. Ook houdt het de staat van het spel bij.

```
1 (maak-menu-adt)
2
```

Message	Parameter	Resultaat
x	/	Geeft de x-coördinaat van het menu terug.
y	/	Geeft de y-coördinaat van het menu terug.
teken!	teken-adt	Laat het teken-adt weten dat het menu getekend moet worden.
staat	/	Geeft de staat van het spel terug.
staat!	nieuwe-staat	Verandert de staat naar de nieuwe staat.
delete!	teken-adt	Verwijdert het menu uit het teken-adt.
install!	teken-adt	Haalt de spelelementen weg en maakt het menu aan.
uninstall	teken-adt	Delete het menu en hertekend de spelelementen.

2.9 Rotator-ADT

Dit ADT dient voor de selector in het menu. Het moet ook niet veel kunnen, behalve op de correcte posities verschijnen. Hiervoor heeft het ook een staat zodat het spel gemakkelijk kan weten welke keuze er gemaakt wordt.

```
1 (maak-menu-adt)
2
```

Message	Parameter	Resultaat
x	/	Geeft de x-coördinaat van de rotator terug.
y	/	Geeft de y-coördinaat van de rotator terug.
teken!	teken-adt	Laat het teken-adt weten dat de rotator getekend moet worden.
staat	/	Geeft de staat van de rotator terug.
up!	/	Doet de rotator stijgen, indien het kan en verandert de staat.
down!	/	Doet de rotator dalen, indien het kan en verandert de staat.

2.10 How-To-ADT

Dit ADT dient om de controls te tonen. Moet ook heel weinig kunnen. Hier zijn ook een `install!` en `uninstall!` functie gemaakt om de correcte dingen van het scherm te halen of juist te tonen.

```
(maak-menu-adt)

```

Message	Parameter	Resultaat
x	/	Geeft de x-coördinaat van de How-To terug.
y	/	Geeft de y-coördinaat van de How-To terug.
teken!	teken-adt	Laat het teken-adt weten dat de How-To getekend moet worden.
install!	teken-adt	Delete het menu en tekent de how-to.
uninstall!	teken-adt	Delete de how-to en tekent het menu.

2.11 Power-Up-ADT

Dit ADT dient om Power-ups aan te maken. Ze worden op random plaatsen getekend en maken gebruik van timers om te weten wanneer en hoe lang ze mogen verschijnen.

```
(maak-power-up-adt positie-x positie-y teken-adt)

```

Message	Parameter	Resultaat
x	/	Geeft de x-coördinaat terug.
y	/	Geeft de y-coördinaat terug.
kleur	/	Geeft dekleur terug.
levens	/	Geeft het aantal levens terug.
staat	/	Geeft de staat terug.
timer	/	Geeft de timer levens terug.
reset!	/	Reset het object.
activeer!	/	Activeert het object en geeft het de correcte kleur.
deactiveer!	/	Deactiveert het object.
random-nummer	/	Geeft het gegeneerde random nummer terug.
x!	pos-x	Verandert de x-coördinaat.
y !	pos-y	Verandert de y-coördinaat.
teken!	teken-adt	Laat het teken-adt weten dat de power-up getekend moet worden.
geraakt!	kogel-adt teken-adt vloot-adt score-adt kogels-adt huidige-tijd spel	Zorgt dat het correcte effect gebeurt. Start een timer en zet de staat op bezig.
zet-terug!	vloot-adt kogels-adt	Maakt het effect ongedaan.

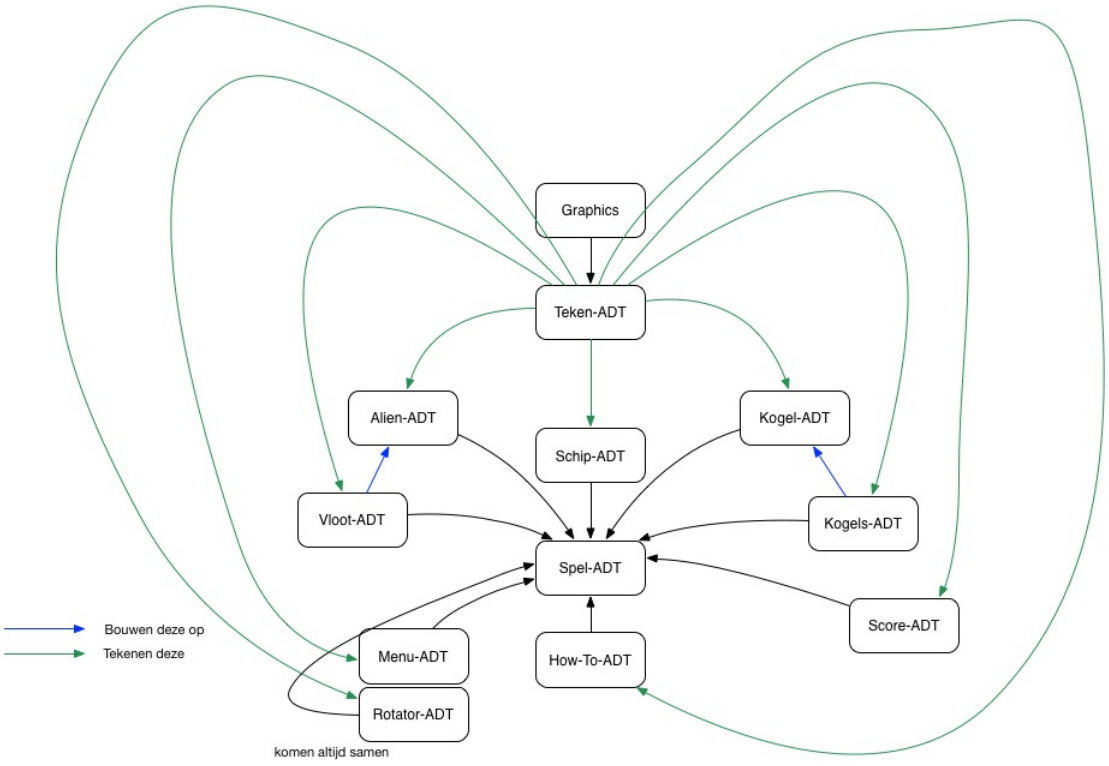
2.12 Score-ADT

Dit ADT dient om de score te tonen. Ook wordt de high-score bijgehouden en getoond.

```
1  (maak-menu-adt)
2
```

Message	Parameter	Resultaat
x	/	Geeft de x-coördinaat terug.
y	/	Geeft de y-coördinaat terug.
score	/	Geeft de score terug.
high-score	/	Geeft de high-score terug.
verhoog	getal	Maakt de nieuwe score aan en slaagt deze op indien nodig
teken!	teken-adt	Laat het teken-adt weten dat het object getekend moet worden.

3 Afhankelijkheidsdiagram



4 Verloop

Een aantal zaken in deze fase verliepen zeer vlot. Zo waren de power-ups snel en gemakkelijk te implementeren, ook met de scores had ik geen problemen. Waar ik heel lang op heb zitten zoeken was hoe ik het spel moest laten herstarten, nadat de aliens de onderkant van het spelvlak raken. Aan de basis van dit probleem lag een mijn slordige code als ook een klein foutje in een for-each. Heb op aanraden van de assistenten dan mijn code wat proberen op te kuisen en zo de bug gevonden.

5 Broncodes

5.1 Abstracties.rkt

```
#lang racket

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;    Alexandre Kahn     ;;;;;;;;;;
;;;;;;;;;;       500067          ;;;;;;;;;;
;;;;;;;;;;      1e BA CW         ;;;;;;;;;;
;;;;;;;;;; alexkahn@vub.ac.be   ;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;Zo kunnen andere files deze bestanden gebruiken.
(provide venster-hoogte
         venster-breedte
         px-venster-hoogte
         px-venster-breedte
         px-alien-hoogte
         px-alien-breedte
         px-alien-reeel
         px-schip-hoogte
         px-schip-breedte
         px-kogel-hoogte
         px-kogel-breedte
         px-start-hoogte
         px-start-breedte
         px-rotator-hoogte
         px-rotator-breedte
         px-how-to-hoogte
         px-how-to-breedte
         helft
         increment
         decrement
         binnen-grenzen?
         check-geraakt?
         centraliseer
         timer5s)

;Speelvenster
(define venster-breedte 1)
(define venster-hoogte 1)

```

```
40 ;Speelvenster in aantal pixels
41 (define px-venster-hoogte 420)
42 (define px-venster-breedte 420)
43
44 ;Grootte van een alien in pixels
45 (define px-alien-hoogte 30)
46 (define px-alien-breedte 30)
47 ;Grootte van een alien op het venster
48 (define px-alien-reeel
49   (/ px-alien-hoogte px-venster-hoogte))
50 ;Grootte van het schip in pixels
51 (define px-schip-hoogte 20)
52 (define px-schip-breedte 30)
53 ;Grootte van de kogel in pixels
54 (define px-kogel-hoogte 8)
55 (define px-kogel-breedte 4)
56 ;Grootte van de start-knop in pixels
57 (define px-start-hoogte 120)
58 (define px-start-breedte 80)
59 ;Grootte van de rotator in pixels
60 (define px-rotator-hoogte 50)
61 (define px-rotator-breedte 50)
62 ;Grootte van de how-to in pixels
63 (define px-how-to-hoogte 300)
64 (define px-how-to-breedte 277)
65
66
67
68 ;Functie om de helft te berekenen.
69 (define (helft object)
70   (/ object 2))
71
72 ;Functie om een positie aan een bepaalde snelheid te doen stijgen
73 (define (increment positie snelheid)
74   (+ positie snelheid))
75 (define (decrement positie snelheid)
76   (- positie snelheid))
77
78 ;Functie om te zien of het object zich nog binnen de grenzen van
       het spel bevindt?
79 (define (binnen-grenzen? positie omvang)
80   (and (< 0 (- positie (helft omvang)))
81        (> px-venster-hoogte (- positie (helft omvang)))))
82
83 ;Functie om te zien of er een botsing tussen 2 objecten gebeurt.
84 (define (check-geraakt? object1 object2)
85   (and (> (object1 'levens) 0)
86        (and (and (<= (- (object1 'x) (helft px-alien-reeel)) (
       object2 'x))
87                  (>= (+ (object1 'x) (helft px-alien-reeel)) (
       object2 'x)))
88             (and (>= (+ (object1 'y) px-alien-reeel) (object2 'y))
89                  (<= (object1 'y) (object2 'y))))))
90
91
92 ;Functie om objecten te centraliseren. Anders worden ze teveel naar
       rechts getekend.
```

```
(define (centraliseer positie type)
  (cond ((eq? type 'alien) (- positie (helft px-alien-hoogte)))
        ((eq? type 'kogel) (- positie (helft px-kogel-hoogte)))
        ((eq? type 'start) (- positie (helft px-start-hoogte)))
        ((eq? type 'schip) (- positie (helft px-schip-breedte)))
        ((eq? type 'rotator) (- positie (helft px-rotator-breedte))
    )
        ((eq? type 'how-to) (- positie (helft px-how-to-breedte)))
        ))
(define timer5s 5000)
```

Abstracties.rkt

5.2 Alien.rkt

```
#lang racket


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;    Alexandre Kahn     ;;;;;;;;;;
;;;;;;;;;;       500067          ;;;;;;;;;;
;;;;;;;;;;      1e BA CW         ;;;;;;;;;;
;;;;;;;;;; alexkahn@vub.ac.be   ;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;
;;Alien ADT;;
;;;;;;;;;;;;;;
;Het Alien ADT zal gewoon alien's oproepen. Daarentegen zal het
    Vloot ADT ervoor zorgen dat er meerdere samen op het scherm
    verschijnen en deze in een latere fase zullen kunnen bewegen
    door de beweging door te geven.
(provide maak-alien-adt)
(require "teken-adt.rkt")
(require "Abstracties.rkt")

(define (maak-alien-adt positie-x positie-y kleur teken-adt)
  (define levens 0)
  (cond ((eq? kleur 'geel)
         (set! levens 1))
        ((eq? kleur 'groen)
         (set! levens 2))
        ((eq? kleur 'paars)
         (set! levens 3)))
  ;Hulpfuncties
  ;Om nieuwe posities mee te geven
  (define (set-positie-x! nieuwe-x)
    (set! positie-x nieuwe-x))
  (define (set-positie-y! nieuwe-y)
    (set! positie-y nieuwe-y))
  ;Om de objecten te tekenen of te verwijderen
  (define (teken! teken-adt)
    ((teken-adt 'teken-alien!) dispatch-alien))
  (define (delete! teken-adt)
    ((teken-adt 'delete-alien!) dispatch-alien))
```

```
;Loop van wat er moet gebeuren als een alien geraakt is.
;De kogel moet gedelete worden, zodat deze niet op het scherm blijft.
;De alien moet indien hij geen levens meer heeft ook gedeletet worden.
(define (geraakt! kogel teken-adt vloot-adt score-adt kogels-adt huidige-tijd spel)
  (teken-adt 'delete-kogel!)
  ((score-adt 'verhoog!) 10)
  (cond ((eq? 'normaal (kogel 'type))
         (levens! (- levens 1)))
        ((eq? 'speciaal (kogel 'type))
         (levens! (- levens 2))))
  (if (>= 0 levens)
      (begin
       (delete! teken-adt)
       ((vloot-adt 'delete-dode-aliens) spel))
      ((vloot-adt 'delete-dode-aliens) spel)))
(define (levens! getal)
  (set! levens getal))

;Dispatch er met dit object ge nterageerd kan worden.
(define (dispatch-alien msg)
  (cond
    ((eq? msg 'x!) set-positie-x!)
    ((eq? msg 'y!) set-positie-y!)
    ((eq? msg 'x) positie-x)
    ((eq? msg 'y) positie-y)
    ((eq? msg 'teken!) teken!)
    ((eq? msg 'delete!) delete!)
    ((eq? msg 'geraakt!) geraakt!)
    ((eq? msg 'levens) levens)
    ((eq? msg 'levens!) levens!)
    ((eq? msg 'kleur) kleur)
    ))

;Dispatch oproepen dit is noodzakelijk voor object-georienteerd programmeren in Scheme
dispatch-alien)
```

Alien.rkt

5.3 Howto.rkt

```
#lang racket



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;    Alexandre Kahn     ;;;;;;;;;
;;;;;;;;;       500067          ;;;;;;;;;
;;;;;;;;;       1e BA CW        ;;;;;;;;;
;;;;;;;;; alexkahn@vub.ac.be    ;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
14
15 (require "teken-adt.rkt")
16 (require "Abstracties.rkt")
17 (provide maak-how-to-adt)
18
19 (define (maak-how-to-adt)
20   (define positie-x 0.3)
21   (define positie-y 0.3)
22   (define (teken! teken-adt)
23     ((teken-adt 'teken-how-to!) dispatch-how-to))
24
25   (define (install! teken-adt)
26     ((teken-adt 'maak-how-to!))
27     ((teken-adt 'delete-menu!))
28     ((teken-adt 'delete-rotator!))
29     )
30   (define (uninstall! teken-adt)
31     ((teken-adt 'delete-how-to!))
32     ((teken-adt 'maak-menu!))
33     ((teken-adt 'maak-rotator!))
34     )
35
36   (define (dispatch-how-to msg)
37     (cond
38       ((eq? msg 'x) positie-x)
39       ((eq? msg 'y) positie-y)
40       ((eq? msg 'teken!) teken!)
41       ((eq? msg 'install!) install!)
42       ((eq? msg 'uninstall!) uninstall!)
43       ))
44   dispatch-how-to)
```

howto.rkt

5.4 Kogel.rkt

```
1 #lang racket
2 (require "teken-adt.rkt")
3 (require "Abstracties.rkt")
4 (provide maak-kogel-adt)
5
6
7 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
8 ;;;;;;;;;    Alexandre Kahn     ;;;;;;;;;
9 ;;;;;;;;;       500067          ;;;;;;;;;
10 ;;;;;;;;;      1e BA CW         ;;;;;;;;;
11 ;;;;;;;;; alexkahn@vub.ac.be  ;;;;;;;;;
12 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
13
14
15 ;;;;;;;;;;;;;;
16 ;;Kogel ADT;;
17 ;;;;;;;;;;;;;;
18 ;Dit ADT zal de kogel aanmaken.Het zal nadat het afgeschoten is
       moeten voortbewegen op het scherm.
19
20 (define (maak-kogel-adt positie-x type)
```

```
21  ;Functie om te schieten is weg. Handiger om elke kogel bij
      aanmaak de juiste x-waarde mee te geven.
22
23  ;Een aantal constantes
24  ;  (define positie-x 0.5)
25  (define positie-y 1)
26  (define snelheid 0.03)
27  (define staat 'actief)
28
29  ;Functies om de posities aan te passen
30  (define (set-positie-x! nieuwe-x)
31    (set! positie-x nieuwe-x))
32  (define (set-positie-y! nieuwe-y)
33    (set! positie-y nieuwe-y))
34
35
36  ;Deze functie zorgt voor de beweging van de kogel. Indien de
      kogel voorbij de rand van het scherm komt wordt hij als het
      ware herladen.
37  (define (beweeg! teken-adt kogels-adt)
38    (cond ((< 0 positie-y)
39           (set-positie-y! (decrement positie-y snelheid)))
40          (else (begin (set! staat 'inactief)
41                       ((teken-adt 'delete-kogel!) dispatch-kogel)
42                       ((kogels-adt 'delete-inactieve-kogels))
43                       ))))
44
45  ;Deze functie zorgt voor het verwijderen. teken-adt krijgt de
      boodschap om de kogel te deleten.
46  (define (delete! teken-adt)
47    ((teken-adt 'delete-kogel!) dispatch-kogel
48                                ))
49
50  ;Laat de kogel na een botsing zowel inactief worden als van het
      scherm te verdwijnen. In verdere fase kan dan naar het type
      kogel gekeken worden.
51  (define (geraakt! object teken-adt vloot-adt kogels-adt score-adt
       huidige-tijd spel)
52    (set! staat 'inactief)
53    (delete! teken-adt)
54    ((object 'geraakt!) dispatch-kogel teken-adt vloot-adt
      score-adt kogels-adt huidige-tijd spel)
55    ;((kogels-adt 'delete-inactieve-kogels))
56
57    )
58
59  ;Gaat na of er een botsing is.
60  (define (schietlus teken-adt vloot-adt kogels-adt score-adt
      power-up huidige-tijd spel) ;power-up
61    (define (schiet-op object)
62      (if (eq? staat 'actief)
63          (cond ((check-geraakt? object dispatch-kogel)
64                 (geraakt! object teken-adt vloot-adt kogels-adt
      score-adt huidige-tijd spel))
65                )
66          'ok))
67    ((vloot-adt 'loop-over-vloot) schiet-op)
```

```
68      (if (eq? (power-up 'staat) 'actief)
69          (cond ((check-geraakt? power-up dispatch-kogel)
70                 (geraakt! power-up teken-adt vloot-adt kogels-adt
      score-adt huidige-tijd spel)))
71          'ok)); Zorgt ervoor dat elke alien afgegaan wordt.
72
73    ;Zorgt ervoor dat het teken-adt het object tekent.
74    (define (teken! teken-adt)
75      ((teken-adt 'teken-kogel!) dispatch-kogel))
76
77    (define (dispatch-kogel msg)
78      (cond
79        ((eq? msg 'x!) set-positie-x!)
80        ((eq? msg 'y!) set-positie-y!)
81        ((eq? msg 'x) positie-x)
82        ((eq? msg 'y) positie-y)
83        ((eq? msg 'beweeg!) beweeg!)
84        ((eq? msg 'teken!) teken!)
85        ((eq? msg 'staat?) staat) ;Geeft de staat van de kogel terug
86        ((eq? msg 'geraakt!) geraakt!)
87        ((eq? msg 'type) type)
88        ((eq? msg 'schietlus) schietlus)
89        ))
90    dispatch-kogel)
```

Kogel.rkt

5.5 Kogels.rkt

```
1  #lang racket
2  (provide maak-kogels-adt)
3
4  (require "teken-adt.rkt")
5  (require "Abstracties.rkt")
6
7  (require "Kogel.rkt")
8
9
10
11 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
12 ;;;;;;;;;    Alexandre Kahn     ;;;;;;;;;
13 ;;;;;;;;;       500067          ;;;;;;;;;
14 ;;;;;;;;;      1e BA CW         ;;;;;;;;;
15 ;;;;;;;;; alexkahn@vub.ac.be   ;;;;;;;;;
16 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
17
18
19 ;;;;;;;;;;;;;;;
20 ;;Kogels ADT;;
21 ;;;;;;;;;;;;;;;
22
23 ;Dit ADT zal als doel hebben om verschillende kogels te maken en te
       zorgen dat deze elk de correcte handelingen doen.
24 (define (maak-kogels-adt)
25   (define kogels '())
26   (define volgende-kogel 'normaal)
27   (define (maak-kogel! posx teken-adt) ;wordt opgeroepen bij
       schieten
```

```
28      (let ((nieuwe-kogel (maak-kogel-adt posx volgende-kogel)))
29        (set! kogels (cons nieuwe-kogel kogels)) ;maakt dmv van cons
        cellen een lijst met kogels. Efficienter door van voor te
        consen.
30        ;((nieuwe-kogel 'schiet!) posx teken-adt) ;Zorgt dat de kogel
         afgeschoten wordt
31        ((teken-adt 'nieuwe-kogel!) nieuwe-kogel)
32        ((teken-adt 'teken-kogel!) nieuwe-kogel)
33
34        ))
35
36    (define (loop-over-kogels functie) ;Zorgt dat functies over de
        kogels gemapt kunnen worden.
37      (map functie kogels))
38
39    (define (teken! teken-adt)
40      (for-each (lambda (kogel-adt)
41                  ((kogel-adt 'teken!) teken-adt))
42                kogels))
43    (define (beweeg! teken-adt vloot-adt score-adt power-up
        huidige-tijd spel)
44      (for-each (lambda (kogel-adt)
45                  ((kogel-adt 'beweeg!) teken-adt dispatch)
46                  ((kogel-adt 'schietlus) teken-adt vloot-adt
        dispatch score-adt power-up huidige-tijd spel))
47                kogels))
48    (define (delete-inactieve-kogels)
49      (define (delete-loop lijst reslijst)
50        (if (not (null? lijst))
51            (if (eq? ((car lijst) 'staat?) 'inactief)
52                (delete-loop (cdr lijst) reslijst)
53                (delete-loop (cdr lijst) (cons (car lijst) reslijst))
        )
54            (set! kogels (reverse reslijst))))
55      (delete-loop kogels '()))
56    (define (volgende-kogel! type)
57      (set! volgende-kogel type))
58    (define (dispatch msg)
59      (cond ((eq? msg 'maak-kogel!) maak-kogel!)
60            ((eq? msg 'kogels) kogels)
61            ((eq? msg 'beweeg!) beweeg!)
62            ((eq? msg 'teken!) teken!)
63            ((eq? msg 'delete-inactieve-kogels)
        delete-inactieve-kogels)
64            ((eq? msg 'volgende-kogel!) volgende-kogel!)
65            ))
66    dispatch)
```

Kogels–adt.rkt

5.6 Menu.rkt

```
1 #lang racket
2 (provide maak-menu-adt)
3
4 (require "teken-adt.rkt")
5 (require "Abstracties.rkt")
```

```
6
7
8  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
9  ;;;;;;;;;    Alexandre Kahn     ;;;;;;;;;
10 ;;;;;;;;;       500067          ;;;;;;;;;
11 ;;;;;;;;;      1e BA CW         ;;;;;;;;;
12 ;;;;;;;;; alexkahn@vub.ac.be   ;;;;;;;;;
13 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
14
15
16 ;;;;;;;;;;;;;;;
17 ;; Menu ADT ;;
18 ;;;;;;;;;;;;;;;
19 (define (maak-menu-adt)
20   ;Een aantal constantes en berekende waarden
21   (define positie-x 0.60)
22   (define positie-y 0.4)
23   (define staat 'pauze)
24
25   ;geeft aan het teken-adt mee dat het menu getekend moet worden
26   (define (teken! teken-adt)
27     ((teken-adt 'teken-menu!) dispatch-menu))
28
29   ;geeft aan het teken-adt mee dat het menu gedeletet mag worden
30   (define (delete! teken-adt)
31     ((teken-adt 'delete-menu!)))
32
33   ;dient om de staat van het spel in op te slaan
34   (define (staat! nieuwe-staat)
35     (set! staat nieuwe-staat))
36
37   (define (install! teken-adt)
38     ((teken-adt 'maak-rotator!))
39     ((teken-adt 'maak-menu!))
40     ((teken-adt 'verwijder-spelelementen!))
41     )
42   (define (uninstall! teken-adt)
43     ((teken-adt 'delete-rotator!))
44     ((teken-adt 'delete-menu!))
45     ((teken-adt 'herteken-spelelementen!))
46     )
47
48   ;Dispatch functie voor interactie met het adt
49   (define (dispatch-menu msg)
50     (cond
51       ((eq? msg 'x) positie-x)
52       ((eq? msg 'y) positie-y)
53       ((eq? msg 'teken!) teken!)
54       ((eq? msg 'staat) staat)
55       ((eq? msg 'staat!) staat!)
56       ((eq? msg 'delete!) delete!)
57       ((eq? msg 'install!) install!)
58       ((eq? msg 'uninstall!) uninstall!)
59       (else (display "error, wrong msg " msg))
60       ))
61   dispatch-menu)
```

Menu.rkt

5.7 Rotator.rkt

```
#lang racket
(provide maak-rotator-adt)

(require "teken-adt.rkt")
(require "Abstracties.rkt")




;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;    Alexandre Kahn     ;;;;;;;;;;
;;;;;;;;;;       500067          ;;;;;;;;;;
;;;;;;;;;;      1e BA CW         ;;;;;;;;;;
;;;;;;;;;; alexkahn@vub.ac.be   ;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;Maakt het adt aan met de selector.
(define (maak-rotator-adt)
  (define staat 0)
  (define positie-x 0.45) ;posities zo gekozen dat het mooi uitkwam
  (define positie-y 0.4)

  (define (up! teken-adt) ;reageert als je met het pijltje naar
    boven gaat
    (if (> staat 0)
        (begin
          (set! staat (- staat 1))
          (set! positie-y (- positie-y 0.18))) ;zorgt dat het mooi
    uitkomt, is omdat het menu 1 foto is
        'ok)
    (teken! teken-adt))

  (define (down! teken-adt)
    ;reageert als je met het pijltje naar boven gaat
    (if (< staat 2)
        (begin
          (set! staat (+ staat 1))
          (set! positie-y (+ positie-y 0.18))) ;zorgt dat het mooi
    uitkomt, is omdat het menu 1 foto is
        'ok)
    (teken! teken-adt))

  ;Geeft mee aan het teken-adt om alles te tekenen
  (define (teken! teken-adt)
    ((teken-adt 'teken-rotator!) dispatch-rotator))
  (define (dispatch-rotator msg)
    (cond
      ((eq? msg 'x) positie-x)
      ((eq? msg 'y) positie-y)
      ((eq? msg 'teken!) teken!)
      ((eq? msg 'staat) staat)
      ((eq? msg 'up!) up!)
      ((eq? msg 'down!) down!)))
```

```
53   dispatch-rotator)
```

rotator.rkt

5.8 Schip.rkt

```
1  #lang racket
2  (require "teken-adt.rkt")
3  (require "Abstracties.rkt")
4  (provide maak-schip-adt)
5
6  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
7  ;;;;;;;;;    Alexandre Kahn     ;;;;;;;;;
8  ;;;;;;;;;       500067          ;;;;;;;;;
9  ;;;;;;;;;      1e BA CW         ;;;;;;;;;
10 ;;;;;;;;; alexkahn@vub.ac.be   ;;;;;;;;;
11 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
12
13
14
15 ;;;;;;;;;;;;;
16 ;;Schip ADT;;
17 ;;;;;;;;;;;;;
18 ;Het Schip-adt zal het schip aanmaken. Het zal ervoor kunnen zorgen
       dat het kan voortbewegen.
19
20 (define (maak-schip-adt)
21   ;Een aantal vaste beginwaarden
22   (define positie-x 0.5)
23   (define positie-y 1)
24   (define snelheid 0.06)
25
26   ;Zorgt dat we een nieuwe positie kunnen geven aan het schip
27   ;Geen set-positie-y! nodig aangezien het schip enkel op de x-as
       beweegt
28   (define (set-positie-x! nieuwe-x)
29     (set! positie-x nieuwe-x))
30
31   ;Het schip zal moeten bewegen op basis van invoer van het
       toetsenbord. Ook moet er gezien worden dat het schip niet uit
       het scherm kan komen
32   (define (beweeg! richting)
33     (cond
34       ((eq? richting 'links)
35        (if (< 0 positie-x)
36            (set-positie-x! (decrement positie-x snelheid))
37            positie-x))
38       ((eq? richting 'rechts)
39        (if (> 1 positie-x)
40            (set-positie-x! (increment positie-x snelheid))
41            positie-x))))
42
43   ;Zorgt ervoor dat het teken-adt het object tekent.
44   (define (teken! teken-adt)
45     ((teken-adt 'teken-schip!) dispatch-schip))
46
47   ;Dispatch er met dit object ge nterageerd kan worden.
```

```
48   (define (dispatch-schip msg)
49     (cond
50       ((eq? msg 'x!) set-positie-x!)
51       ((eq? msg 'x) positie-x)
52       ((eq? msg 'beweeg!) beweeg!)
53       ((eq? msg 'teken!) teken!)))
54
55   dispatch-schip)
```

Schip.rkt

5.9 Spel.rkt

```
1  #lang racket
2  (provide maak-adt-spel)
3  (require "teken-adt.rkt")
4  (require "Abstracties.rkt")
5  (require "Alien.rkt")
6  (require "Vloot.rkt")
7  (require "Schip.rkt")
8  (require "Kogel.rkt")
9  (require "Kogels-adt.rkt")
10 (require "Menu.rkt")
11 (require "rotator.rkt")
12 (require "howto.rkt")
13 (require "Score.rkt")
14 (require "Power-Ups.rkt")
15
16
17 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
18 ;;;;;;;;;    Alexandre Kahn     ;;;;;;;;;
19 ;;;;;;;;;       500067          ;;;;;;;;;
20 ;;;;;;;;;      1e BA CW         ;;;;;;;;;
21 ;;;;;;;;; alexkahn@vub.ac.be   ;;;;;;;;;
22 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
23
24
25
26 ;;;;;;;;;;;;;;;
27 ;; SPEL ADT ;;
28 ;;;;;;;;;;;;;;;
29 ;Dit adt zal de spellus bevatten. Ook zal het ervoor zorgen dat de
       input van het toetsenbord doorgegeven wordt aan de nodige
       spelelementen. Verder zal de collision detection ook in dit ADT
        gebeuren.
30
31 (define (maak-adt-spel)
32   ;Maakt alle nodige adt's aan.
33   (define teken-adt (maak-adt-teken "Space Invaders"
       px-venster-hoogte px-venster-breedte))
34   (define schip-adt (maak-schip-adt))
35   (define vloot-adt (maak-vloot-adt teken-adt))
36   (define kogels-adt (maak-kogels-adt))
37   (define menu-adt (maak-menu-adt))
38   (define rotator-adt (maak-rotator-adt))
39   (define how-to-adt (maak-how-to-adt))
40   (define score-adt (maak-score-adt))
```

```
41  (define Power-up-adt (maak-power-up-adt (/ (random 1 10) 10) (/ (
      random 5 10) 10) teken-adt))
42  (define level 1)
43  (define begin-aliens 10) ;aantal aliens die er in het 1e level
      moeten komen
44  (define spel-tijd 0)
45  (define start-positie-aliens 0.05)
46
47  (define (pauze!) ;alles wat moet gebeuren als het spel gepauzeerd
       wordt
48    ((menu-adt 'install!) teken-adt) ;verwijdert het schip, de
      kogels en de aliens en tekent het menu
49    ((teken-adt 'set-toets-functie!) toets-naar-spel-pauze) ;zet de
       correcte lussen
50    ((teken-adt 'set-spel-lus-functie!) spel-lus-functie-pauze)
51    ((rotator-adt 'teken!) teken-adt) ;tekent de rotator, een 1e
      keer
52    ((menu-adt 'teken!) teken-adt) ;tekent het menu, een 1e keer
53    )
54
55  (define (restart!) ; bereidt het spel voor om te herstarten
56    ((vloot-adt 'verwijder-vloot!))
57    (pauze!) ; zet automatisch alles op pauze
58    ((teken-adt 'reset-vloot!)) ; verwijdert de volledige vloot
59
60    ((vloot-adt 'maak-aliens!) begin-aliens start-positie-aliens) ;
       maakt een nieuwe vloot aan
61    (set! level 1) ; zet het level terug op 1
62    )
63
64  (define (play) ; start het spel op
65    ((menu-adt 'uninstall!) teken-adt) ;tekent de sprites terug
66    ;((teken-adt 'delete-menu!)) ; verwijdert het menu en de
      rotator
67    ((teken-adt 'set-toets-functie!) toets-naar-spel-play) ;zet de
      correcte lussen
68    ((teken-adt 'set-spel-lus-functie!) spel-lus-functie-play)
69    )
70
71  (define (howto) ; gaat naar het howto menu
72    ;((menu-adt 'uninstall!) teken-adt) ;vewijdert het menu
73    ((how-to-adt 'install!) teken-adt); maakt de correcte tiles aan
74    ((teken-adt 'set-toets-functie!) toets-naar-spel-howto) ;zet de
       correcte lus
75    ((teken-adt 'teken-how-to!) how-to-adt) ; tekent de juiste
      tiles
76    )
77  (define (nieuw-level!) ; maakt een nieuw level aan
78    ;((Power-up-adt 'delete!) teken-adt)
79    ;((Power-up-adt 'reset!))
80    (set! level (+ level 1)) ;verhoogt het level
81    ((vloot-adt 'maak-aliens!) (* level begin-aliens)
      start-positie-aliens) ; maakt nieuwe en meerdere aliens aan
82    ((vloot-adt 'maak-alien-tiles!)) ;tekent de tiles voor de
      nieuwe aliens
83    )
84
```

```
85  ;spellus
86  ;Leest de toetsenbordinput uit en stuurt dit door naar het spel.
87  ;Afhankelijk van de staat van het spel
88  (define (toets-naar-spel-play toets)
89    (cond
90      ((eq? toets 'right)
91       ((schip-adt 'beweeg!) 'rechts))
92      ((eq? toets 'left)
93       ((schip-adt 'beweeg!) 'links))
94      ((eq? toets #\space)
95       ((kogels-adt 'maak-kogel!) (schip-adt 'x) teken-adt))
96      ((eq? toets #\p)
97       (pauze!))))
98  (define (toets-naar-spel-pauze toets); pas menu-tekenen na move
99    (cond
100     ((eq? toets #\return)
101      (cond ((= 0 (rotator-adt 'staat))
102             (play))
103            ((= 1 (rotator-adt 'staat))
104             (howto))
105            ((= 2 (rotator-adt 'staat))
106             (exit))))
107     ((eq? toets 'up)
108      ((rotator-adt 'up!) teken-adt))
109     ((eq? toets 'down)
110      ((rotator-adt 'down!) teken-adt))
111     ))
112 (define (toets-naar-spel-howto toets)
113   (cond ((eq? toets #\return)
114          (begin
115            ;((menu-adt 'install!) teken-adt)
116            ((how-to-adt 'uninstall!) teken-adt)
117            ((teken-adt 'set-toets-functie!) toets-naar-spel-pauze
    )
118            ((teken-adt 'set-spel-lus-functie!)
    spel-lus-functie-pauze))
119          )
120         ))
121
122
123 ; Functies voor het starten van het spel
124 (define (start)
125   ((menu-adt 'install!) teken-adt)
126   ;Is de selector om in het menu te kiezen wat we willen openen
127   (set! spel-tijd 0)
128   ((vloot-adt 'maak-aliens!) begin-aliens start-positie-aliens)
129   ((menu-adt 'teken!) teken-adt)
130   ((rotator-adt 'teken!) teken-adt))
131
132 ;;Schietlus, moet enkel opgeroepen worden als er geschoten is
133 ;; Oproepen om het spel te laten lopen
134 (define (spel-lus-functie-play delta-tijd)
135   (set! spel-tijd (+ spel-tijd delta-tijd))
136   ((schip-adt 'teken!) teken-adt)
137   ((vloot-adt 'teken!) teken-adt)
138   ((vloot-adt 'beweeg!) dispatch-spel)
139   ;Indien de kogel afgeschoten is wordt de collision detection
```

```
      gedaan. Is performanter op deze manier, anders moet er elke
      beurt voor elke alien gecheckt worden.
140     ((kogels-adt 'beweeg!) teken-adt vloot-adt score-adt
      Power-up-adt spel-tijd dispatch-spel)
141     ((kogels-adt 'teken!) teken-adt)
142     ((score-adt 'teken!) teken-adt)
143     (if (eq? 'actief (Power-up-adt 'staat)) ;als het al actief is,
      moet het getekend worden, of na 5s van het scherm verdwijnen
144         (if (< spel-tijd (+ timer5s (Power-up-adt 'random-nummer)))
145             ((Power-up-adt 'teken!) teken-adt)
146             (begin ((Power-up-adt 'delete!) teken-adt)
147                    ((Power-up-adt 'reset!))))
148         (if (> spel-tijd (Power-up-adt 'random-nummer)) ;als het op
       inactief staat moet het op de goede moment getekend worden
149             ((Power-up-adt 'activeer!))
150             'ok)
151         )
152     (if (and (eq? 'bezig (Power-up-adt 'staat))
153              (> spel-tijd (+ timer5s (Power-up-adt 'timer))))
154         ((Power-up-adt 'zet-terug!) vloot-adt kogels-adt)
155         'ok); met de timer zien of het groter is
156     (if (null? (vloot-adt 'vloot))
157         (nieuw-level!)
158         'ok))
159   (define (spel-lus-functie-pauze delta-tijd)
160     ; Input voor in het menu
161     'ok
162     )
163
164   ;(nu worden de adt's zoals het moet in een lus hertekend)
165   ((teken-adt 'set-spel-lus-functie!) spel-lus-functie-pauze)
166   ((teken-adt 'set-toets-functie!) toets-naar-spel-pauze)
167
168   ;; Dispatch functie
169   (define (dispatch-spel msg)
170     (cond
171       ((eq? msg 'start) start)
172       ((eq? msg 'pauze!) pauze!)
173       ((eq? msg 'restart!) restart!)
174       ((eq? msg 'nieuw-level!) nieuw-level!)
175       ))
176   dispatch-spel)
177
178 ;Maakt het ADT van het spel, moet wel nog opgeroepen worden.
179 (define spel (maak-adt-spel))
180
181 ;Entry point om alles te laden
182 ((spel 'start))
183 ;https://docs.racket-lang.org/reference/strings.html
184 ;
```

Spel.rkt

5.10 Teken-adt.rkt

```
1 #lang racket
2
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;    Alexandre Kahn     ;;;;;;;;;
;;;;;;;;;       500067          ;;;;;;;;;
;;;;;;;;;      1e BA CW         ;;;;;;;;;
;;;;;;;;; alexkahn@vub.ac.be   ;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(require "Graphics.rkt")
(require "helpers.rkt")
(require "Abstracties.rkt")
;Debug functies komen uit de helpers.rkt file die met het snakeproject meegegeven zijn. Deze werden tijdens het schrijven van de code veel gebruikt om zaken mee te testen.
;zodat ander files hieraan kunnen

(provide maak-adt-teken)
;Zorgt evoor dat het teken-adt vanuit de spelfile aangesproken kan worden.


;;;;;;;;;;;;;;;;;
;;Het teken-adt;;
;;;;;;;;;;;;;;;;;

;Maakt het teken-adt met een venster waarop getekend kan worden
(define (maak-adt-teken titel h-pixels v-pixels)
  (define venster (make-window v-pixels h-pixels titel))
  ((venster 'set-background!) "black")

  ;;;;;;;;;;
  ;;CONFIG;;
  ;;;;;;;;;;
  (define (redraw-all! lijst laag)
    (map (lambda (pair) ;neemt een assoclijst zoals de alien-tiles
           ((laag 'add-drawable) (cdr pair)))
         lijst))
  (define (verwijder-alle! lijst laag)
    (map (lambda (pair) ;neemt een assoclijst zoals de alien-tiles
           ((laag 'remove-drawable) (cdr pair)))
         lijst))
  ;;;;;;;;;;;;;;;;;;;;;;
  ;; Config voor alien ;;
  ;;;;;;;;;;;;;;;;;;;;;;

  ;;; Maakt een alien-laag aan. Hierop zullen de alien(-tiles) getekend worden.
  (define alien-laag (venster 'make-layer))

  ;;; Beginnen met een lege lijst aan tiles, zodat we er in de globale omgeving aankunnen
  (define vloot-tiles '())

  ;;; Functie om alien-tiles aan de vloot toe te voegen.
  (define (voeg-alien-toe! alien-adt)
    (define kleurfoto '())
    (cond ((eq? 'geel (alien-adt 'kleur))
           (set! kleurfoto (make-bitmap-tile "monster-geel.jpg")))
```

```
    ;moet met set! anders geen actie gedaan
55          ((eq? 'groen (alien-adt 'kleur))
56           (set! kleurfoto (make-bitmap-tile "monster-groen.jpg")))
57          ((eq? 'paars (alien-adt 'kleur))
58           (set! kleurfoto (make-bitmap-tile "monster-paars.jpg")))
    )
59    (set! vloot-tiles (cons (cons alien-adt kleurfoto) vloot-tiles)
    )
60    ;((alien-laag 'add-drawable) kleurfoto)
61    )
62  (define (voeg-alien-tile-toe! alien-adt)
63    (let* ((alien-tile (neem-iets alien-adt vloot-tiles)))
64      ((alien-laag 'add-drawable) alien-tile)))
65  ;;; Functie om de correcte alien of kogel uit de lijst te halen.
66  (define (neem-iets iets-adt uit-tiles)
67    (cdr (assoc iets-adt uit-tiles)))
68
69  ;;; Functie om een alien te verwijderen. Zowel uit de lijst van
      de vloot-tiles als effectief van het scherm (remove-drawable)
70  (define (verwijder-alien! alien-adt)
71    ;(debug "delete-alien")
72    (let* ((alien-tile (neem-iets alien-adt vloot-tiles)))
73      ((alien-laag 'remove-drawable) alien-tile))
74    (set! vloot-tiles (remove alien-adt vloot-tiles (lambda (r e) (
      eq? (car e) e)))))
75  (define (reset-vloot!)
76    (set! vloot-tiles '())
77    (verwijder-alle! vloot-tiles alien-laag)
78    )
79
80
81
82  ;;;;;;;;;;;;;;;;;;;;
83  ;;Config voor schip;;
84  ;;;;;;;;;;;;;;;;;;;;
85
86  ;;;De laag waarop het schip getekend zal worden
87  (define schip-laag (venster 'make-layer))
88
89  ;;; In dit geval is er slechts een tile dus die kan hardcoded
      staan
90  (define schip-tile (make-bitmap-tile "schip.jpg" "schip-mask.jpg"
      ))
91
92  ;;; Zorgt ervoor dat het schip getekend kan worden. Moet niet te
      deleten zijn.
93  (define (maak-schip!)
94    ((schip-laag 'add-drawable) schip-tile))
95
96  ;verwijderen van het schip
97  (define (verwijder-schip!)
98    ((schip-laag 'remove-drawable) schip-tile))
99
100
101 ;;;;;;;;;;;;;;;;;;;;
102 ;;Config voor kogel;;
103 ;;;;;;;;;;;;;;;;;;;;
```

```
104 ; Laag waarop de kogel getekend zal worden
105 (define kogel-laag (venster 'make-layer))
106
107 ; Een lege lijst voor de kogel, zo zal deze pas op het scherm
      komen nadat men hem nodig heeft.
108 (define kogel-tiles '())
109
110 ; Functie die de kogel aanmaakt en in assoc lijst steekt
111 (define (maak-kogel! kogel-adt)
112   (define kogel-obj '())
113   (if (eq? 'normaal (kogel-adt 'type))
114       (set! kogel-obj (make-bitmap-tile "kogel.jpg"))
115       (set! kogel-obj (make-bitmap-tile "kogel-groen.jpg")))
116   (set! kogel-tiles (cons (cons kogel-adt kogel-obj) kogel-tiles)
      )
117   ((kogel-laag 'add-drawable) kogel-obj)
118   )
119
120 ; Functie om een getekende kogel te verwijderen
121 (define (verwijder-kogel! kogel-adt)
122   (let ((kogel-obj (neem-iets kogel-adt kogel-tiles)))
123     ((kogel-laag 'remove-drawable) kogel-obj))
124   (set! kogel-tiles (remove kogel-adt kogel-tiles (lambda (r e) (
      eq? (car e) e)))))
125
126
127 ;;;;;;;;;;;;;;;;;;;;
128 ;;Config voor menu;;
129 ;;;;;;;;;;;;;;;;;;;;
130 ; Laag waarop het menu getekend zal worden
131 (define menu-laag (venster 'make-layer))
132 (define menu-tile (make-bitmap-tile "Menu.png"))
133
134 ;Functie om het menu te verwijderen
135 (define (verwijder-menu!)
136   ((menu-laag 'remove-drawable) menu-tile)
137   ;(display "verwijder menu")
138   )
139 ; tekent de menu-tile
140 (define (maak-menu!)
141   ((menu-laag 'add-drawable) menu-tile)
142   ;(display "Maak menu")
143   )
144
145 ;Functie om alles wat weggehaald wordt bij het aanmaken van een
      menu te hertekenen.
146 (define (herteken-spelelementen!)
147   (maak-schip!)
148   (redraw-all! vloot-tiles alien-laag)
149   ;((Power-Up-laag 'add-drawable) Power-Up-tile)
150   )
151
152 ; dient om alle spel-elementen te verwijderen
153 (define (verwijder-spelelementen!)
154   (verwijder-alle! vloot-tiles alien-laag)
155   (verwijder-alle! kogel-tiles kogel-laag)
156   (verwijder-schip!)
```

```
157     (delete-power-up!)
158     )
159
160   ;;;;;;;;;;;;;;;;;;;;;;;
161   ;;Config voor rotator;;
162   ;;;;;;;;;;;;;;;;;;;;;;;
163
164   ; Laag waarop de rotator getekend zal worden
165   (define rotator-laag (venster 'make-layer))
166   (define rotator-tile (make-bitmap-tile "rotator.jpg"))
167   (define (maak-rotator!)
168     ((rotator-laag 'add-drawable) rotator-tile))
169   (define (verwijder-rotator!)
170     ((rotator-laag 'remove-drawable) rotator-tile))
171
172   ;;;;;;;;;;;;;;;;;;;;;
173   ;;Config voor howto;;
174   ;;;;;;;;;;;;;;;;;;;;;
175
176   ; Laag waarop het how-to scherm getekend zal worden
177   (define how-to-laag (venster 'make-layer))
178   (define how-to-tile '())
179
180   (define (maak-how-to)
181     (let ((tile (make-bitmap-tile "HowTo.jpg")))
182       (set! how-to-tile tile)
183       ((how-to-laag 'add-drawable) tile)))
184   (define (verwijder-how-to!)
185     ((how-to-laag 'remove-drawable) how-to-tile)
186     (set! how-to-tile '()))
187   ;;;;;;;;;;;;;;;;;;;;;
188   ;;Config voor score;;
189   ;;;;;;;;;;;;;;;;;;;;;
190
191   ; Laag waarop het how-to scherm getekend zal worden
192   (define score-laag (venster 'make-layer))
193   (define score-tile (make-tile h-pixels v-pixels))
194   ((score-laag 'add-drawable) score-tile)
195
196   ; #TODO config voor Power-Ups
197   (define Power-Up-laag (venster 'make-layer))
198   (define Power-Up-tile (make-tile h-pixels v-pixels))
199   ;((Power-Up-laag 'add-drawable) Power-Up-tile)
200   (define (maak-power-up!)
201     ((Power-Up-laag 'add-drawable) Power-Up-tile))
202   (define (delete-power-up!)
203     ((Power-Up-laag 'remove-drawable) Power-Up-tile))
204   ;;;;;;;;;;;;;;;;;;
205   ;;TEKEN FUNCTIES;;
206   ;;;;;;;;;;;;;;;;;;
207
208   ;;;;;;;;
209   ;;Schip;;
210   ;;;;;;;;
211
212   (define (teken-schip! schip-adt)
213     (let* ((schip-x (* h-pixels (schip-adt 'x)))
```

```
214             (schip-y (- v-pixels px-schip-hoogte))
215             (absolute-x (centraliseer schip-x 'schip)))
216       ((schip-tile 'set-x!) absolute-x)
217       ((schip-tile 'set-y!) schip-y)
218       ;(debug "schip tekenen" schip-x " " schip-y)
219       ))
220
221   ;;;;;;;;;
222   ;;Alien;;
223   ;;;;;;;;;
224
225   (define (teken-alien! alien-adt)
226     (let* ((alien-x (* h-pixels (alien-adt 'x)))
227             (alien-y (* v-pixels   (alien-adt 'y)))
228             (alien-tile (neem-iets alien-adt vloot-tiles))
229             (absolute-x (centraliseer alien-x 'alien)))
230       ((alien-tile 'set-x!) absolute-x)
231       ((alien-tile 'set-y!) alien-y)
232       ;(debug "Alien tekenen" alien-x " " alien-y)
233       ))
234
235   ;;;;;;;;;
236   ;;Kogel;;
237   ;;;;;;;;;
238
239   (define (teken-kogel! kogel-adt)
240     (let* ((kogel-x (* h-pixels (kogel-adt 'x)))
241             (kogel-y (* v-pixels (kogel-adt 'y)))
242             (kogel-tile (neem-iets kogel-adt kogel-tiles))
243             (absolute-x (centraliseer kogel-x 'kogel)))
244       ((kogel-tile 'set-x!) absolute-x)
245       ((kogel-tile 'set-y!) kogel-y)
246       ;(debug "kogel tekenen" kogel-x " " kogel-y)
247       ))
248
249   ;;;;;;;;
250   ;;Menu;;
251   ;;;;;;;;
252
253   (define (teken-menu! menu-adt)
254     (let* ((start-x (* h-pixels (menu-adt 'x)))
255             (start-y (* v-pixels (menu-adt 'y)))
256             (absolute-x (centraliseer start-x 'start)))
257       ((menu-tile 'set-x!) absolute-x)
258       ((menu-tile 'set-y!)  start-y)
259       ))
260
261   ;;;;;;;;;;;
262   ;;Rotator;;
263   ;;;;;;;;;;;
264
265   (define (teken-rotator! rotator-adt)
266     (let* ((rotator-x (* h-pixels (rotator-adt 'x)))
267             (rotator-y (* v-pixels (rotator-adt 'y)))
268             (absolute-x (centraliseer rotator-x 'rotator)))
269       ((rotator-tile 'set-x!) absolute-x)
270       ((rotator-tile 'set-y!)  rotator-y)
```

```
271       ))
272
273   ;;;;;;;;;;
274   ;;How-To;;
275   ;;;;;;;;;;
276   (define (teken-how-to! how-to-adt)
277     (if (not (null? how-to-tile))
278         (begin
279           (let* ((how-to-x (* h-pixels (how-to-adt 'x)))
280                  (how-to-y (* v-pixels (how-to-adt 'y)))
281                  (absolute-x (centraliseer how-to-x 'how-to)))
282             ((how-to-tile 'set-x!) how-to-x)
283             ((how-to-tile 'set-y!)  how-to-y)
284             )
285           )
286         'ok))
287
288   ;;;;;;;;;
289   ;;Score;;
290   ;;;;;;;;;
291
292   (define (teken-score! score-adt)
293     (let* ((score-x (* h-pixels (score-adt 'x)))
294            (score-y (* v-pixels (score-adt 'y)))
295            (string (string-append "score: "(number->string (
      score-adt 'score)) "                    high-score: " (
      number->string (score-adt 'high-score))))
296
297            )
298       (score-tile 'clear)
299       ((score-tile 'draw-text) string
300                                20
301                                score-x
302                                score-y
303                                "green")))
304
305   ;TODO Tekenfunctie voor Power-Ups
306   (define (teken-Power-UP! Power-Up-adt kleur)
307     (let* ((Power-Up-x (* h-pixels (Power-Up-adt 'x)))
308            (Power-Up-y (* v-pixels (Power-Up-adt 'y)))
309            (absolute-x (centraliseer Power-Up-x 'alien))
310            )
311       (Power-Up-tile 'clear)
312       ((Power-Up-tile 'draw-ellipse) absolute-x Power-Up-y
      px-alien-hoogte px-alien-breedte kleur)
313       )
314     )
315
316
317   ;;;;;;;;;;;;;;;;;;;;;
318   ;; Spellus functies;;
319   ;;;;;;;;;;;;;;;;;;;;;
320
321   (define (set-spel-lus-functie! fun)
322     ((venster 'set-update-callback!) fun))
323
324   (define (set-toets-functie! fun)
```

```
325     ((venster 'set-key-callback!) fun))
326
327
328   ;;;;;;;;;;;;;;
329   ;; Dispatch ;;
330   ;;;;;;;;;;;;;;
331
332   (define (dispatch-teken-adt msg)
333     (cond ((eq? msg 'set-toets-functie!) set-toets-functie!)
334           ((eq? msg 'set-spel-lus-functie!) set-spel-lus-functie!)
335           ;; Teken functies.
336           ((eq? msg 'teken-schip!) teken-schip!)
337           ((eq? msg 'teken-alien!) teken-alien!)
338           ((eq? msg 'teken-kogel!) teken-kogel!)
339           ((eq? msg 'teken-menu!) teken-menu!)
340           ((eq? msg 'teken-rotator!) teken-rotator!)
341           ((eq? msg 'teken-how-to!) teken-how-to!)
342           ((eq? msg 'teken-score!) teken-score!)
343           ((eq? msg 'teken-power-up!) teken-Power-UP!)
344           ;; Delete en maak functies
345           ((eq? msg 'herteken-spelelementen!)
      herteken-spelelementen!)
346           ((eq? msg 'nieuwe-alien!) voeg-alien-toe!)
347           ((eq? msg 'nieuwe-kogel!) maak-kogel!)
348           ((eq? msg 'maak-rotator!) maak-rotator!)
349           ((eq? msg 'maak-menu!) maak-menu!)
350           ((eq? msg 'maak-how-to!) maak-how-to)
351           ((eq? msg 'maak-power-up!) maak-power-up!)
352           ((eq? msg 'delete-alien!) verwijder-alien!)
353           ((eq? msg 'delete-kogel!) verwijder-kogel!)
354           ((eq? msg 'delete-schip!) verwijder-schip!)
355           ((eq? msg 'delete-menu!) verwijder-menu!)
356           ((eq? msg 'delete-rotator!) verwijder-rotator!)
357           ((eq? msg 'delete-how-to!) verwijder-how-to!)
358           ((eq? msg 'delete-power-up!) delete-power-up!)
359           ((eq? msg 'verwijder-spelelementen!)
      verwijder-spelelementen!)
360           ((eq? msg 'reset-vloot!) reset-vloot!)
361           ((eq? msg 'voeg-alien-tile-toe!) voeg-alien-tile-toe!)
362           (else (display "error, message ") (display msg) (display
      "not understood"))))
363   dispatch-teken-adt)
```

teken–adt.rkt

5.11 Vloot.rkt

```
1  #lang racket
2  (provide maak-vloot-adt)
3  (require "teken-adt.rkt")
4  (require "Abstracties.rkt")
5  (require "Alien.rkt")
6
7
8  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
9  ;;;;;;;;;    Alexandre Kahn     ;;;;;;;;;
10 ;;;;;;;;;       500067          ;;;;;;;;;
```

```scheme
;;;;;;;;;      1e BA CW        ;;;;;;;;;
;;;;;;;;; alexkahn@vub.ac.be  ;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;



;;;;;;;;;;;;;
;;Vloot ADT;;
;;;;;;;;;;;;;
;Het vloot-adt zal in een lijst alle aliens bijhouden. Ook zal dit ervoor zorgen dat bepaalde functies op alle aliens toegepast kunnen worden.

(define (maak-vloot-adt teken-adt)
  ;De (+ aantal 1) is om te zorgen dat ze mooi gecentreerd staan
  ;(define afstand-tussen-2 0)
  ;Dit maakt een lege vloot aan in de vorm van een lijst.
  (define vloot '())
  (define richting-vloot 'rechts)
  (define horizon-snelheid 0.002)
  (define vert-snelheid 0.02)
  (define afstand-tussen-2 0)
  (define aantal-per-rij 10)
  (define threshold 25)
  ;maak-aliens! maakt aliens aan, deze zullen dan een positie en dergelijke hebben, met maximaal 10 per rij. en worden in een lijst opgeslagen.
  ;het teken-adt wordt ook aangesproken om daar de nieuwe alien-tiles aan te maken
  (define (maak-aliens! aantal pos-y)
    (define (maak-rij-aliens! teller pos-x pos-y aantal-op-rij kleur)
      (if (< teller aantal-op-rij)
          (begin  (let* ((nieuwe-alien (maak-alien-adt pos-x pos-y kleur teken-adt)))
                    (set! vloot (cons nieuwe-alien vloot))
                    ((teken-adt 'nieuwe-alien!) nieuwe-alien)
                    )
                  (maak-rij-aliens! (+ teller 1) (- pos-x afstand-tussen-2) pos-y aantal-op-rij kleur))
          'ok))
    (if (> aantal threshold)
        (begin
          (set! afstand-tussen-2 (/ venster-breedte aantal-per-rij))
          (maak-rij-aliens! 0 (- venster-breedte afstand-tussen-2) pos-y aantal-per-rij 'paars)
          (maak-aliens! (- aantal aantal-per-rij) (+ pos-y 0.1)))
        (if (> aantal aantal-per-rij)
            (begin
              (set! afstand-tussen-2 (/ venster-breedte aantal-per-rij))
              (maak-rij-aliens! 0 (- venster-breedte afstand-tussen-2) pos-y aantal-per-rij 'groen)
              (maak-aliens! (- aantal aantal-per-rij) (+ pos-y 0.1)))
            (begin
```

```
55              (set! afstand-tussen-2 (/ venster-breedte aantal))
56              (maak-rij-aliens! 0 (- venster-breedte
    afstand-tussen-2) pos-y aantal 'geel))))
57    )
58  ;Zorgt ervoor dat voor elke alien oproept om zichzelf te tekenen
59  (define (teken! teken-adt)
60    (for-each (lambda (alien-adt)
61                ((alien-adt 'teken!) teken-adt))
62              vloot))
63
64  ;Zorgt ervoor dat men een functie kan uitvoeren op elke alien (
    nodig voor de collision detection).
65  (define (loop-over-vloot functie)
66    (map functie vloot))
67
68  (define (verwijder-alle-aliens)
69    (for-each (lambda (alien-adt)
70                ((alien-adt 'delete!) teken-adt))
71              vloot))
72
73  (define (maak-alien-tiles!)
74    (for-each (lambda (alien-adt)
75                ((teken-adt 'voeg-alien-tile-toe!) alien-adt))
76              vloot))
77  (define (verwijder-vloot!)
78    (set! vloot '()))
79
80  ;Zorgt voor de beweging van de aliens
81  ;Gaat na of de aliens met de rand van het scherm botsen en laat
    ze dan terug draaien.
82  (define (beweeg! spel)
83    (define laagste-alien (car vloot))
84    (define (daal!)
85      (for-each (lambda (alien-adt)
86                  (let ((y (alien-adt 'y)))
87                    ((alien-adt 'y!) (increment y vert-snelheid))))
88                vloot))
89    (for-each (lambda (alien-adt)
90                (let ((y (alien-adt 'y))
91                      (x (alien-adt 'x)))
92                  (when (> (alien-adt 'y) (laagste-alien 'y))
93                    (set! laagste-alien alien-adt)
94                    )
95                  (cond
96                    ((eq? richting-vloot 'links) (if  (< 0 x)
97                                                      ((alien-adt '
    x!) (decrement x horizon-snelheid))
98                                                      (begin
99                                                        (set!
    richting-vloot 'rechts)
100                                                       (daal!)
101                                                       ((alien-adt
     'x!) (increment x horizon-snelheid)))
102                                                     ))
103                   ((eq? richting-vloot 'rechts) (if (> 1 x)
104                                                     ((alien-adt '
    x!) (increment x horizon-snelheid))
```

```
105                                                          (begin
106                                                            (set!
       richting-vloot 'links)
107                                                            (daal!)
108                                                            ((alien-adt
        'x!) (increment x horizon-snelheid))))))
109                     )
110                   )
111                 vloot)
112     (if (> (laagste-alien 'y) 1) ;probleem was dat de restart in de
        for-each zat, bijgevolg werd het menu voor elke alien getekend
113         ((spel 'restart!))
114         'ok)
115     )
116
117   (define (delete-dode-aliens spel)
118     (define (delete-loop lijst reslijst)
119       (if (not (null? lijst))
120           (if (= ((car lijst) 'levens) 0)
121               (delete-loop (cdr lijst) reslijst)
122               (delete-loop (cdr lijst) (cons (car lijst) reslijst))
       )
123           (set! vloot (reverse reslijst))))
124     (delete-loop vloot '()))
125
126
127
128   (define (versnel! factor) ;versnelling kan ook negatief zijn
129     (set! vert-snelheid (* horizon-snelheid factor)))
130   ;Zo kan dezelfde functie gebruikt worden om zowel de witte
        power-up te starten als de gele te stoppen
131   (define (stop!)
132     (set! horizon-snelheid 0))
133   (define (herstart!)
134     (set! horizon-snelheid 0.002))
135
136   ;Dispatch er met dit object ge nterageerd kan worden.
137   (define (dispatch msg)
138     (cond ((eq? msg 'maak-aliens!) maak-aliens!)
139           ((eq? msg 'teken!) teken!)
140           ((eq? msg 'loop-over-vloot) loop-over-vloot)
141           ((eq? msg 'beweeg!) beweeg!)
142           ((eq? msg 'verwijder-vloot!) verwijder-vloot!)
143           ((eq? msg 'vloot) vloot)
144           ((eq? msg 'delete-dode-aliens) delete-dode-aliens)
145           ((eq? msg 'versnel!) versnel!)
146           ((eq? msg 'stop!) stop!)
147           ((eq? msg 'herstart!) herstart!)
148           ((eq? msg 'maak-alien-tiles!) maak-alien-tiles!)
149           ))
150   dispatch)
```

vloot.rkt

5.12 Power-Ups-adt

```
1 #lang racket
```

```
2
3
4  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
5  ;;;;;;;;;    Alexandre Kahn     ;;;;;;;;;
6  ;;;;;;;;;       500067          ;;;;;;;;;
7  ;;;;;;;;;      1e BA CW         ;;;;;;;;;
8  ;;;;;;;;; alexkahn@vub.ac.be   ;;;;;;;;;
9  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
10
11 ;; Random functie werkt pas vanaf Racket 6.4!!!
12
13 ;;;;;;;;;;;;;;;;;
14 ;;Power-Ups ADT;;
15 ;;;;;;;;;;;;;;;;;
16 (provide maak-power-up-adt)
17 (require "teken-adt.rkt")
18 (require "Abstracties.rkt")
19
20 (define (maak-power-up-adt positie-x positie-y teken-adt)
21   (define timer 0)
22   (define levens 1)
23   (define staat 'inactief)
24   (define random-nummer (random 1 10000))
25
26   ;Hulpfuncties
27   ;Om nieuwe posities mee te geven
28   (define (set-positie-x! nieuwe-x)
29     (set! positie-x nieuwe-x))
30   (define (set-positie-y! nieuwe-y)
31     (set! positie-y nieuwe-y))
32   ;Om de objecten te tekenen of te verwijderen
33   (define (teken! teken-adt)
34     ((teken-adt 'teken-power-up!) dispatch-power-up kleur)
35     )
36   (define (delete! teken-adt)
37     ((teken-adt 'delete-power-up!))
38     ;(deactiveer!)
39     (set! random-nummer (+ random-nummer (random 5000 20000))))
40   (define (reset!)
41     (deactiveer!)
42     (set! id (random 1 4)) ;kiest een getal tussen 1 en 3
43     (set-positie-x! (/ (random 1 10) 10))
44     (set-positie-y! (/ (random 5 10) 10)))
45
46   (define (geraakt! kogel-adt teken-adt vloot-adt score-adt
       kogels-adt huidige-tijd spel)
47     (cond
48       ((eq? kleur "yellow")
49        ((vloot-adt 'stop!)))
50       ((eq? kleur "red")
51        ((kogels-adt 'volgende-kogel!) 'speciaal))
52       ((eq? kleur "red")
53        ((vloot-adt 'versnel!) 2)))
54     (set! timer huidige-tijd)
55     (set! staat 'bezig)
56     (delete! teken-adt)
57     ;(reset!)
```

```
58      ((score-adt 'verhoog! ) 5))
59    (define id (random 1 4))
60    (define kleur '())
61    (define (zoek-kleur)
62      (cond ((= id 1) (set! kleur "yellow"))
63            ((= id 2) (set! kleur "white"))
64            ((= id 3) (set! kleur "red"))))
65
66    (define (activeer!)
67      (zoek-kleur)
68      ((teken-adt 'maak-power-up!))
69      (set! staat 'actief)
70      ;(set! random-nummer (+ random-nummer (random 1 10000)))
71
72      )
73    (define (deactiveer!)
74      (set! staat 'inactief))
75    (define (zet-terug! vloot-adt kogels-adt)
76      (cond
77        ((eq? kleur "yellow")
78         ((vloot-adt 'herstart!)))
79        ((eq? kleur "red")
80         ((kogels-adt 'volgende-kogel!) 'normaal))
81        ((eq? kleur "white")
82         (vloot-adt 'versnel!) 0.5))
83      (reset!))
84
85
86    (define (dispatch-power-up msg)
87      (cond
88        ((eq? msg 'x!) set-positie-x!)
89        ((eq? msg 'y!) set-positie-y!)
90        ((eq? msg 'x) positie-x)
91        ((eq? msg 'y) positie-y)
92        ((eq? msg 'teken!) teken!)
93        ((eq? msg 'delete!) delete!)
94        ((eq? msg 'kleur) kleur)
95        ((eq? msg 'levens) levens)
96        ((eq? msg 'geraakt!) geraakt!)
97        ((eq? msg 'activeer!) activeer!)
98        ((eq? msg 'deactiveer!) deactiveer!)
99        ((eq? msg 'staat) staat)
100       ((eq? msg 'random-nummer) random-nummer)
101       ((eq? msg 'timer) timer)
102       ((eq? msg 'reset!) reset!)
103       ((eq? msg 'zet-terug!) zet-terug!)
104       ))
105
106   dispatch-power-up)
```

Power–Ups.rkt

5.13 Score

```
1 #lang racket
2 (provide maak-score-adt)
3 (require "teken-adt.rkt")
```

```
(require "Abstracties.rkt")

(define (maak-score-adt)
  (define score 0)
  (define high-score 0)
  (define positie-x 0)
  (define positie-y 0)
  (define pad "score.txt")

  (define (lees-van-schijf!)
    (when (file-exists? pad) ;als het bestand niet bestaat wordt het ook niet opgeroepen
      (call-with-input-file pad
        (lambda (file)
          (set! high-score (read file))))))

  (define (opslaan!)
    (when (> score high-score) (begin
                                 (save-scores-to-file!)
                                 (set! high-score score))))

  (define (verhoog! aantal)
    (set! score (+ score aantal))
    (opslaan!))

  (define (teken! teken-adt)
    ((teken-adt 'teken-score!) dispatch-score))
  ;variabele bestand is eigenlijk een poort naar het bestand
  (define (save-scores-to-file!)
    (define bestand (open-output-file pad
                                      #:mode'binary
                                      #:exists'replace)) ;als het bestand als bestaat, wordt het gewoon overschreven
    (write high-score bestand)
    (close-output-port bestand))

  (define (dispatch-score msg)
    (cond ((eq? msg 'verhoog!) verhoog!)
          ((eq? msg 'score) score)
          ((eq? msg 'high-score) high-score)
          ((eq? msg 'x) positie-x)
          ((eq? msg 'y) positie-y)
          ((eq? msg 'teken!) teken!)
          ))
  (lees-van-schijf!)
  dispatch-score)
;http://docs.racket-lang.org/reference/file-ports.html?q=do
```

score.rkt

6 Bibliografie

Sussman G.J. Abelson, A. *Structure and Interpretation of Computer Programs*, The MIT Press, 1996.

`https://docs.racket-lang.org/reference/strings.html`

`http://docs.racket-lang.org/reference/file-ports.html?q=do`