Programmeren Bulletin

hcc! programmeren

Interessegroep

27ste jaargang voorjaar 2020 Nummer 1

Inhoud

Redactioneel

Beste Programmeerleden,

Helaas moet ik jullie meedelen dat ik als redacteur ermee stop. Ik ben begonnen in 2005 en ik heb met plezier de nieuwsbrieven geschreven. Ik wil iedereen bedanken voor de hulp en steun. Iedereen kan gewoon vragen en ideeën blijven sturen. Op mijn eigen website www.tronicasoftware.nl, komen onderwerpen over Unity.

Ik stop niet met de bijeenkomsten. Ik blijf gewoon komen naar De Bilt, Apeldoorn en waarschijnlijk ook nog Amersfoort, en natuurlijk de grote HCC dagen tussendoor.

Veel plezier met programmeren en zijn er vragen, kom dan eens langs of stuur een email naar m.a.kurvers@tronicasoftware.nl.

Marco Kurvers

Leren programmeren in Liberty BASIC

Liberty BASIC is een fijne Basic programmeertaal waarmee Windows programma's geschreven kunnen worden, maar ook grafische programma's en games.

De omgeving van Liberty BASIC is niet visueel. Windows componenten kunnen niet op de vensters worden geplaatst met de muis, maar moeten in de code geschreven worden.

De besturing met de componenten en vensters is niet te vergelijken met de oude C omgeving, toen nog zelfs de Windows main lus geschreven moest worden met alle events die dan nodig waren. Ook al moeten we dat nog steeds doen, toch werkt het in Liberty BASIC een stuk eenvoudiger dan in C.

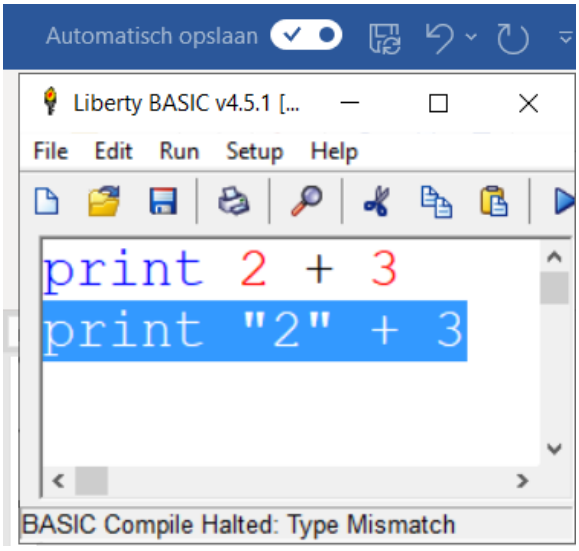
Het PRINT statement

In alle Basic versies en dialecten kennen we wel het PRINT statement. In Liberty BASIC kunnen we PRINT nog steeds zo gebruiken als het oude PRINT statement om tekst en expressies uit te voeren.

Start Liberty Basic en typ eens in: `PRINT "Hallo allemaal"`
Bovenaan zien we het Run menu. We kunnen het programma starten via het Run menu, maar het kan ook via het start icoon op de werkbalk.

Normaal gesproken zou de uitvoer onder de ingevoerde regel moeten verschijnen, zoals in sommige Basic dialecten gebeurt. Echter verschijnt er een apart venster met de uitvoer. Dat venster is het Debug venster. Dat het Debug venster verschijnt na het starten van de code, is niet altijd handig. Het Debug venster is nuttig tijdens het schrijven van een programma, maar niet als alles goed is en het programma correct werkt. Het Debug venster zit dan in de weg.

We kennen PRINT om expressies uit te voeren, zoals: `PRINT 2 + 3`
Proberen we cijfer 2 tussen aanhalingstekens uit te voeren, dan geeft de compiler direct een foutmelding en wordt de foute regel geselecteerd, zie de afbeelding.



Een oplossing is om de VAL() functie te gebruiken als er getallen als tekstwaarden worden gebruikt.

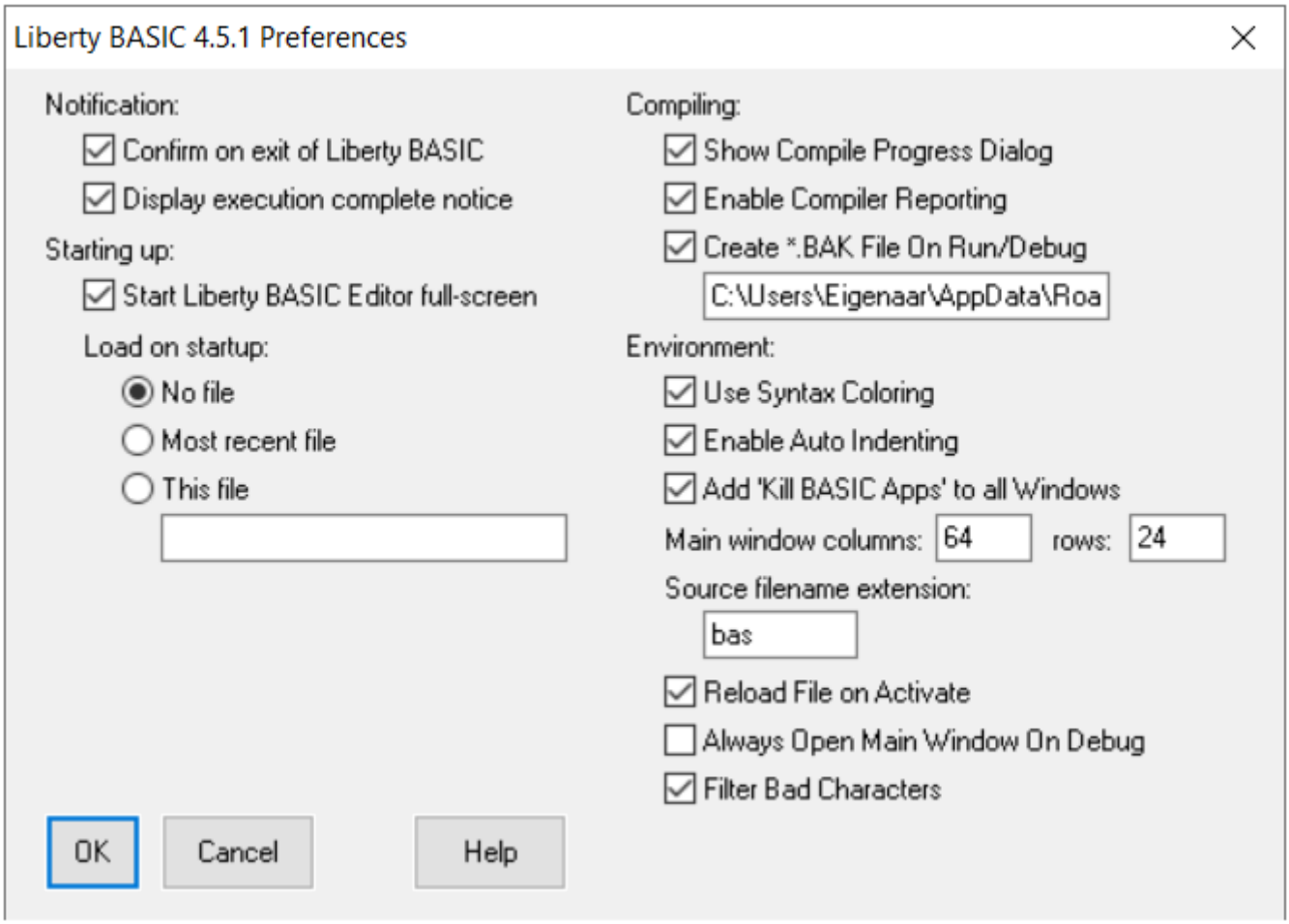
Vervangen we de laatste regel met

```
print val("2") + 3
```

dan zal de expressie hetzelfde zijn als 2 + 3.

Werken in kleur

Zoals de afbeelding laat zien, kunnen we in kleur werken. De statements, functies en waarden, en nog meer elementen, hebben hun eigen kleur. We kunnen de kleuren niet wijzigen, maar we kunnen de kleuren wel uitschakelen. De afbeelding hier laat het Preferences venster zien.



Er zijn veel opties, zoals je zelf het prettigste vindt, om in te stellen. Bij Environment kan de Use Syntax Coloring aan of uit worden gezet. Je kunt er ook voor kiezen om wel of niet met een full-screen te starten en of je direct een programma wilt openen als je Liberty BASIC start.

Variabelen

In Liberty BASIC kunnen de variabelen eenvoudig worden gebruikt. We hoeven ze niet te declareren en er zijn ook geen speciale types voor integers en floats. Het enige symbooltype dat we nodig hebben is het dollarteken **$**, om tekst aan variabelen toe te kennen.

We kunnen dus ook de VAL() functie als volgt gebruiken:

```
A$ = "2"
PRINT VAL(A$) + 5
```

De VAL() functie is ook handig voor het controleren van gebruikersinvoer. Is er een getal ingevoerd of niet? Wanneer niet, dan geeft de functie een nul waarde terug. Dat kan lastig zijn als er een 0 als getal ingevoerd is. Met een IF statement kunnen we eerst controleren of er een 0 ingevoerd is. Zo ja, dan is de invoer geldig en kunnen we het aan een numerieke variabele toekennen.

Variabelen mogen gescheiden worden met punten. Een variabele mag bijvoorbeeld geschreven worden als: `c.d = 20`
De variabele c is nu niet gedefinieerd, alleen met de punt en de letter d erachter. Ook al is dit mogelijk, we kunnen het niet vergelijken met een record. Wel is het handig om variabelen te groeperen.

```
Fruitmand.Appels = 30
Fruitmand.Peren = 25
```

Een andere manier dat ook zo werkt is een structure, die als sleutelwoord STRUCT heet. We kunnen dus ook records maken, alleen is er wel een probleem. De structures kunnen we niet voor meerdere variabelen gebruiken, omdat een structure in Liberty BASIC geen type is. De naam van de STRUCT moeten we direct als een variabele gebruiken, op dezelfde manier als Fruitmand. Het enige verschil is dat we achter de velden het sleutelwoord STRUCT moeten gebruiken.

```
struct Fruitmand, _
    Appels as long, _
    Peren as Long

Fruitmand.Appels.struct = 30
Fruitmand.Peren.struct = 25

print "Totaal aantal fruit: "; Fruitmand.Appels.struct + _
Fruitmand.Peren.struct
```

De compiler wil dat er een puntkomma of een komma achter de tekst wordt gebruikt en geen plus teken. De compiler komt anders met een uitvoeringsfout, omdat we geen tekst met numerieke waarden op kunnen tellen.

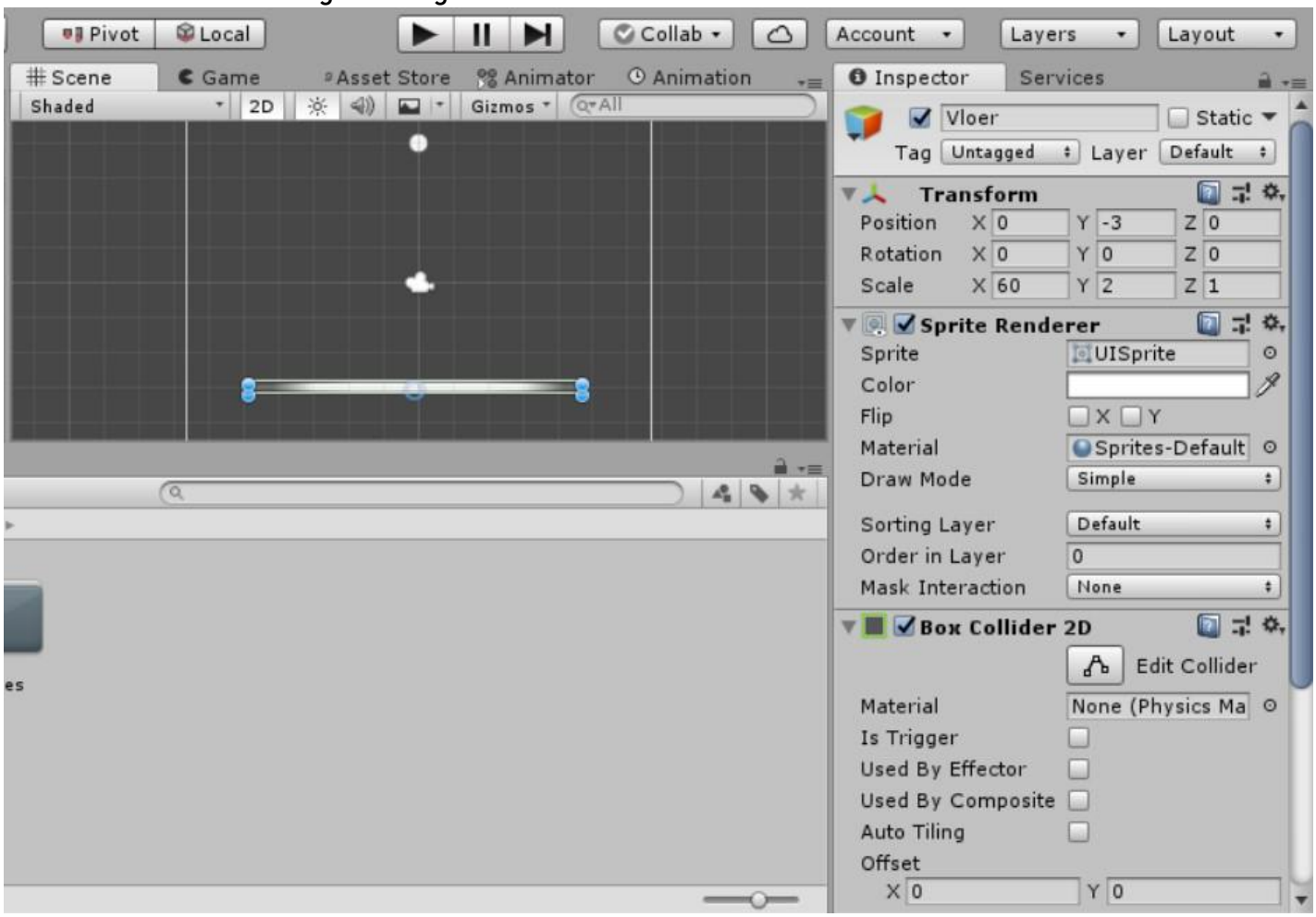
We hebben voor variabelen geen types, maar in structures wel. We mogen "as <type>" ook niet vergeten, want alleen de variabele als veldnaam is niet voldoende. In de structure kunnen allerlei types worden gebruikt, een long, een boolean en ook een double. Liberty BASIC kent meer types, maar die worden voor andere bewerkingen gebruikt, zoals een RECT type om de grootte van een venster door te geven aan Windows.

Unity – GameObjecten en scripts

De vorige keer heb ik laten zien hoe we in een 2D omgeving GameObjecten maken en componenten eraan toevoegen. Door een materiaal toe te voegen en wat in te stellen, hebben we een leuke vloer, een bal en stuitereffecten. Maar dat is niet genoeg om de vloer te laten bewegen en de bal weg te laten schieten vanaf de vloer. We hebben scripts nodig om de besturing te kunnen maken.

Heb je nog het project van de vorige keer, open dan deze. Heb je het niet, probeer dan met behulp van de vorige Bulletin 2019 nummer 2 het voorbeeld te maken.

Onderstaande afbeelding laat nog eens de vloer en de bal zien.



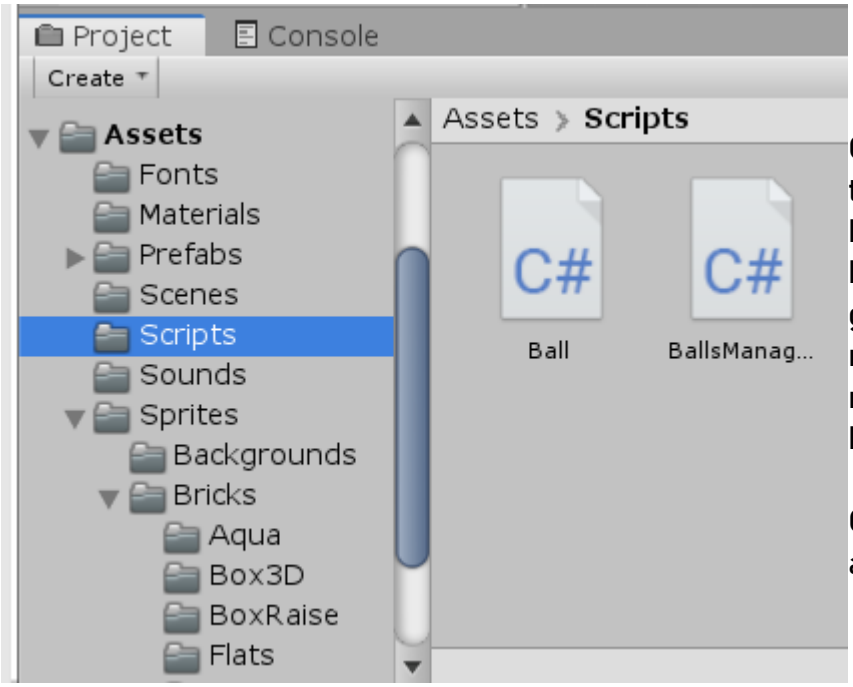
Wijzig de naam Vloer in Paddle bovenaan in de Inspector. Wijzig ook de Scale breedte 60 in een kleinere waarde, zodat de Paddle niet zo langgerekt is. Merk op dat hoe smaller de paddle wordt, hoe dikker het object lijkt. Zo kan de dikte ook wel in 1.5 worden gewijzigd in plaats van 2, wat je wilt.

Het Paddle script

In de vorige Bulletin konden we de bal laten stuiteren door het physics material in te stellen met een gravity (zwaartekracht), waardoor we de bal een karakter konden geven. Er is helaas geen instelmogelijkheid om de paddle heen en weer te laten bewegen. We hebben code nodig om dat te kunnen doen.

Voordat we een script gaan toevoegen, maken we eerst een map in de Assets aan, zodat we in die map de scripts kunnen plaatsen.

Klik met de rechter muisknop op Assets en kies Create → Folder. Er wordt een map aangemaakt. De naam kun je wijzigen door op de rechter muisknop op de map te klikken en Rename te kiezen. Wijzig de naam in Scripts. Hieronder een voorbeeld van een ander project. Je ziet hier dat een map ook weer mappen kan hebben, net zoals de Windows Verkenner.



Om een map in een geopende map aan te maken, kun je met de rechter muisknop op de map klikken en dezelfde handelingen doen zoals hierboven uitgelegd. Je kunt ook eerst de map openen en dan op het lege vlak aan de rechterkant met de rechter muisknop klikken en de map aanmaken.

Op dezelfde manier kun je ook scripts aanmaken, door C# Script te kiezen.

Geef het script de naam Paddle. In de Inspector verschijnt het geraamte van het script. We kunnen niet in de Inspector zelf de code gaan wijzigen. Er is een editor nodig waar we C# code in kunnen voeren en ook ingesteld is in Unity.

We kunnen Visual Studio gebruiken om Unity scripts te schrijven, maar Unity heeft ook een eigen editor genaamd MonoDevelop. Kies zelf wat je wilt om de scripts te schrijven. In Unity ga je naar Edit → Preferences → External Tools. Kun je bij External Script Editor geen editor kiezen, dan kun je Visual Studio Community gratis downloaden of installeer een nieuwe Unity update setup en vink in de lijst MonoDevelop aan. Op Youtube is er meer informatie over te vinden.

Dubbelklik op het Paddle script. Het script wordt geopend en onderstaande code verschijnt.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Paddle : MonoBehaviour
{

    // Start is called before the first frame update
    void Start()
    {

    }
```

```
    // Update is called once per frame
    void Update()
    {

    }
}
```

In Unity heeft een script een klasse. In oudere Unity versies was dat in C# ook al zo, maar in JavaScript werden geen klasses aangemaakt en kon er ook niet worden bepaald of de variabelen wel of niet in de Inspector mochten verschijnen, zoals we later zullen zien. In de Unity versies na versie 5 wordt JavaScript niet meer ondersteund en kan alleen C# worden gebruikt.

Elke klasse stamt af van MonoBehaviour. Dat is de basisklasse van Unity. De klasse Paddle erft alle events van de basisklasse, zoals Start() en Update(). Het event Start() voert code uit nadat het object is aangemaakt, maar voordat het eerste frame is geweest. Na Start() wordt na elke frame het Update() event aangeroepen. Het event Start() wordt dus aangeroepen nadat het object is aangemaakt, maar als er een object gemaakt wordt dan zal er een ander event eerst worden aangeroepen voor het Start() event. Dat is het Awake() event.

De Paddle besturen

Laten we in de Update() de Paddle besturen. We kunnen dit doen met de toetsen of de muis. Typ in het event onderstaande code.

```
float h = Input.GetAxis("Horizontal");
```

De GetAxis() functie haalt informatie op uit de gegeven axis. Unity heeft vooraf aangemaakte axis voor de besturing, zoals Vertical, Mouse X, Fire1 en Jump. Ga naar Edit → Project Settings… en klik op Input. Rechts verschijnen de axis besturingen die je kunt wijzigen of alleen even bekijken, bijvoorbeeld om te weten welke axis je in de GetAxis() functie nodig hebt. Voor de paddle hebben we Horizontal nodig. Open deze door op het driehoekje te klikken. Er zijn veel mogelijkheden, maar we hebben één nodig om in te stellen. Het Type bepaalt op welke manier je horizontaal kunt besturen: met de toetsen of met de muis. Bepaal zelf welk type je wilt.

Elke axis heeft dezelfde instellingen. Bovenaan zie je Size. Size toont het aantal besturingen die aangemaakt zijn. Je kunt ook zelf een axis besturing toevoegen.

Onderstaande code geeft de positie door aan het transform component van de paddle.

```
transform.Translate(Vector2.right * h * speed);
```

De Translate methode zorgt ervoor dat alleen door middel van Vector2.right een horizontale positie bepaalt wordt en de Y en Z posities ongewijzigd blijven. Onderstaand voorbeeld doet hetzelfde, maar de paddle zou nu met een te hoge snelheid omkantelen.

```
transform.position = new Vector2(Vector2.right * h * speed, transform.position.y);
```

Het is dus voor GameObjecten, die bestuurd moeten worden, beter om Translate te gebruiken.

Variabele of eigenschap

Zoals je in de code ziet, hebben we een variabele die nog niet bestaat: speed
We hebben een snelheid nodig om een goede beweging te kunnen maken. Zonder de snelheid zal de paddle heel traag bewegen. Vector2.right geeft een 1 en als we op toets D drukken of met de

muis naar rechts schuiven, dan zal de totale snelheid 1 * 1 zijn. Door een extra snelheid mee te geven, bepalen we hoe snel de paddle over het scherm moet bewegen. Hoe doen we dat?

Als eerste moeten we de variabele speed aanmaken. Dat doen we bovenaan in de klasse, zie hieronder.

```
public class Paddle : MonoBehaviour
{
    public float speed;
```

Variabelen worden altijd op die plaats aangemaakt. Maar speed heeft geen waarde. In de code is dat ook nog niet nodig. Omdat de variabele speed een public float is, kan het als eigenschap in de Inspector worden gebruikt.

In de Inspector is de variabele speed een eigenschap. Zouden we de variabele private declareren, dan is de variabele in de Inspector niet te zien. Dat is precies het probleem met JavaScript. Variabelen worden gedeclareerd zoals hieronder:

```
var speed : float;
```

Deze variabelen zijn altijd public en zijn dus altijd zichtbaar in de Inspector.

Private variabelen zijn dus niet te zien in de Inspector. Soms willen we dat ook, maar het nadeel is dat andere scriptklasses niet van de variabelen gebruik kunnen maken, alleen van de public variabelen.

Er is een manier om public variabelen niet weer te geven in de Inspector door gebruik te maken van eigenschappen.

```
public int eigenschap { get; set; }
```

Nu kunnen alle scripts gebruik maken van de eigenschap, maar deze is in de Inspector niet te zien. Waarschijnlijk heeft JavaScript ook die mogelijkheid om toch variabelen niet als Inspector eigenschappen weer te geven.

Sla het script op en ga terug naar Unity. Klik op het Paddle GameObject. Sleep het script naar de Inspector. Het script wordt als een component toegevoegd.

In de Inspector zie je ook de eigenschap Speed. Merk op dat we de variabele met een kleine letter hadden gedeclareerd. In de Inspector staan ze altijd met een hoofdletter. Hieronder een vergelijking tussen de code en de Inspector.

Code	Inspector
speed	Speed
clampLeft	Clamp Left
bool b	B (met een selecteervakje)
int[] a	Toont A met een waardenlijst. Eigenschap Size bepaalt het aantal waarden.

Als we het project starten, kunnen we de paddle heen en weer bewegen. De paddle komt ook buiten de camera, zodat we het object niet kunnen zien. We moeten dat voorkomen door colliders te maken aan de linkerkant en de rechterkant van de scene.

In het Paddle script zullen we twee variabelen gebruiken die de twee x posities bepalen. Eén daarvan staat in het lijstje hierboven: clampLeft

Stop de uitvoer en ga weer naar het Paddle script. Typ onderstaande code onder de variabele speed.

```
public float clampLeft;
public float clampRight;
```

Typ onder de Translate() regel onderstaande code.

```
if (transform.position.x < clampLeft)
{
	transform.position = new Vector2(clampLeft, transform.position.y);
}
if (transform.position.x > clampRight)
{
	transform.position = new Vector2(clampRight, transform.position.y);
}
```

Het liefst zouden we in willen typen:

```
transform.position.x = clampLeft; // Fout!
```

Maar dat is helaas niet toegestaan, omdat position een alleen-lezen object is. Alleen nieuwe vector objecten mogen aan position worden toegekend.

Sla het script op en ga terug naar Unity. In de Inspector van de Paddle zien we de twee eigenschappen verschijnen, precies zoals het lijstje aangaf: het begint met een hoofdletter en waar het met een hoofdletter begint, komt een spatie.

Verplaats de paddle naar de linkerkant van de scene om de Clamp Left te bepalen. Wanneer de Position de juiste waarde heeft, voer je dat in de Clamp Left eigenschap in. Doe dit net zo met de Clamp Right.

Start het project om het te testen. Als je de juiste linkerkant en rechterkant posities hebt gegeven, zal de paddle niet meer buiten de scene komen.

Zo heeft elk GameObject zijn eigen script component voor besturing. Er zijn ook GameObjecten die geen besturing nodig hebben. Deze zijn statisch. GameObjecten met besturing zijn dynamisch. In de vorige Bulletin werd het stuiteren en de Rigidbody van de bal uitgelegd. Nu moeten we de bal laten bewegen alleen binnen de scene. We moeten dus iets hebben dat er voor zorgt dat de bal niet buiten de scene komt, net als bij de paddle. Maar dezelfde manier gebruiken als bij de paddle, met een left en een right, zal bij de bal niet werken.

Op youtube.com zijn veel tutorials te vinden over de paddle en de ball.

Hieronder heb ik links geplaatst met tutorials die het beste te gebruiken zijn. Ze zijn Engelstalig, maar goed te volgen.
De links:

https://www.youtube.com/watch?v=NWG8v002oj4
Deze tutorial duurt vier en een half uur met uitgebreide uitleg van een paddle en een bal tot het maken van een menu en een highscore lijst en het spelen van muziek en geluid.

https://www.youtube.com/playlist?list=PLElhtYsGq0haMcAYcCPM-AGglUmKUkO5B
Deze tutorial is leuk om te volgen. Het is in delen, zodat je de volgende keer verder kunt met het deel waar je gebleven was. In deze tutorial leer je ook hoe je meerdere ballen kunt gebruiken, door gebruik te maken van een BallsManager.

In deel 2 is te zien dat hij GitHub gebruikt voor de opslag. Je kunt zelf bepalen in welke map je het wilt hebben. Een GitHub account is dan niet nodig. Hopelijk is hij ook goed te verstaan, want hij praat met een accent.

https://www.youtube.com/watch?v=t_ahbdB0AnI&t=2148s
Deze tutorial geeft minder details, maar is daardoor makkelijker te begrijpen. Het project is in 3D, maar de besturing is ook voor 2D projecten te gebruiken. Hij laat hier uitgebreid zien hoe je de canvas gebruikt om tekst weer te geven.

BASIC programma's voor verschillende computers

Vorig jaar zomer heeft mijn appartement een nieuw jasje gekregen. Nieuw laminaat, behang en meubels. Daarvoor heb ik veel opgeruimd. Wat een zooi!

Tijdens het opruimen vond ik een schrijfboekje, een kladblok, met allemaal BASIC programma's. Sommige programma's zijn heel groot, want die zijn heel apart. Ze waren in de tijd toen we nog radio Hilversum hadden. Teletekst was er met een extra pagina over Basicode. Enorme listings werden getoond die zo waren geschreven, dat ze in elk BASIC dialect overgenomen konden worden. Het belangrijkste deel, dat voor elke computer anders was, werd niet getoond.

De eerste regel is altijd de initialisatieregel. Variabele A bepaalt het aantal tekens in een string. Het commando GOTO20 initialiseert de variabelen en maakt alle variabelen leeg .

Elk BASIC dialect heeft zijn eigen commando om het scherm schoon te maken, zoals CLS of PRINT met een symbool. Dat doet regel 1010. In regel 100 is het schoonmaakcommando aanwezig.

In regel 110 wordt de cursor op de juiste plaats gezet. Dit wordt bepaald door de variabelen HO voor het aantal kolommen en VE voor het aantal rijen.

In regel 120 wordt de huidige cursorpositie gelezen. De variabelen HO en VE krijgen deze waarden.

In regel 200 wordt de invoer gecontroleerd. De ingedrukte toets wordt toegekend aan IN$. In regel 210 wordt er gewacht tot op een toets wordt gedrukt. Ook die toets wordt aan IN$ toegekend.

In regel 250 wordt een geluid gegeven. De pitch en volume worden ook in die regel bepaald.

In regel 260 wordt een waarde toegekend aan variabele RV. De waarde is gelijk of groter dan 0 en kleiner dan 1.

In regel 270 wordt een garbage-collection cycle gestart en aan variabele FR het aantal vrije geheugenbytes toegekend.

In regel 300 wordt de variabele SR$ ingesteld op een tekenreeksweergave van het aantal dat opgeslagen is in de variabele SR. De string bevat geen overvloedige spaties.

In regel 310 wordt ook SR$ ingesteld, maar nu altijd met een komma met een totale lengte gegeven in variabele CT voor het aantal karakters en variabele CN voor het aantal cijfers achter de komma, maar wanneer nodig wordt het afgerond. Als de weergave niet past, wordt een reeks met de lengte CN herhaaldelijk met het * teken teruggegeven.

In regel 350 wordt de tekst van variabele SR$ uitgeprint. Er wordt geen nieuwe regel meegegeven, tenzij deze al in de variabele aanwezig is.

In regel 360 wordt een nieuwe regel afgedrukt.

De variabelen hebben in Basicode een speciaal doel. De variabelen zijn:

A, CN, CT, FR, HO, IN$, RV, SR, SR$, VE

Schoonspringen

Eén van de listings was Schoonspringen. Wie het programma overnam, had het eerste gedeelte nodig, met de aangegeven regels bovenaan, om het in de juiste BASIC taal te kunnen laten werken.

```
1000 A=200:GOTO20:REM SCHOONSPRINGEN
1010 GOSUB100
1020 GOSUB2430:REM INIT + SCHERM OPMETEN
1030 GOSUB2560:REM TITELSCHERM UITLEG
1040 GOSUB1780:REM TEKEN SPEELBORD
1050 :
1060 HO=S+4:VE=20:GOSUB110:PRINTL$
1070 IFQ=1 THENGOSUB1920
1080 Q=0:IFW=0 THENB$="Start "
1090 IFW>0 THENB$="Sprong "
1100 B$=B$+"getal ? ":GOSUB1970
1110 Z$="":GOSUB1370:Z=VAL(Z$)
1120 :
1130 IF(Z$="S") OR(Z$="s")THEN1290
1140 IF(Z<1) OR(Z>64) THENQ=1:GOTO1060
1150 V=INT((Z-1)/8)+1:H=Z-(V-1)*8
1160 IFW=0 THEN1240
1170 VC=ABS(V1-V):CH=AB$(H1-H)
1180 IFR(Z)=1 THENQ=1
1190 IF(VC<>1) AND(VC<>2) THENQ=1
1200 IFVC=1 THENIFCH<>2 THENQ=1
1210 IFVC=2 THENIFCH<>1 THENQ=1
1220 IFQ=1 THEN1060
1230 GOSUB1510
1240 GOSUB1560:R(Z)=1
1250 W=W+1:GOSUB1610
1260 V1=V:H1=H
1270 IFW<>64 THEN1060
1280 :
1290 B$="":IFW=64 THENGOSUB1680
1300 HO=S+1:VE=20:GOSUB110:PRINTL$
1310 B$=B$+" Nog een keer ? J/N ":GOSUB1970
1320 GOSUB2100:IFT=1 THEN1040
1330 GOSUB2250
1340 END
1350 :
1360 REM INVOER
1370 GOSUB120
1380 GOSUB2050:IFTS=13 THEN1470
1390 IF(IN$="S") OR(IN$="s") OR(IN$="0") THEN1420
1400 IF(IN$="V") OR(IN$="v") THEN1440
1410 IFVAL(IN$)=0 THEN1380
1420 IFLEN(Z$)=2 THEN1380
1430 Z$=Z$+IN$:PRINTIN$;:HO=HO+1:GOTO1380
1440 IFZ$="" THEN1380
1450 HO=HO-1:GOSUB110:PRINT" ";:GOSUB110
1460 L=LEN(Z$):IFL>1 THENZ$=LEFT$(Z$,L-1):GOTO1380
1465 Z$=""
1470 IFZ$="" THEN1380
1480 RETURN
1490 :
```

```
1500 REM VELD LEEG
1510 HO=S+H1*2+H1-1:VE=V1*2
1520 GOSUB110:PRINT"  ";
1530 RETURN
1540 :
1550 REM PLAATS ##
1560 HO=S+H*2+H-1:VE=V*2
1570 GOSUB110:PRINT"##";
1580 RETURN
1590 :
1600 REM PRINT STAND
1610 SR=W:GOSUB310:HO=S+35:VE=6
1620 GOSUB110:PRINTSR$:VE=8
1630 SR=64-W:GOSUB310
1640 GOSUB110:PRINTSR$
1650 RETURN
1660 :
1670 REM BORD LEEG
1680 GOSUB1510:FORI=1 TO10:GOSUB1740
1690 GOSUB1570:GOSUB1740:GOSUB1520
1700 NEXTI:B$="Gefeliciteerd !"
1710 RETURN
1720 :
1730 REM KNIPPER VERTRAGING
1740 FORV=1 TO4:GOSUB2170:NEXTV
1750 RETURN
1760 :
1770 REM TEKEN SPEELBORD
1780 FORI=1 TO64:R(I)=0:NEXTI
1790 GOSUB100:PRINT:FORI=0 TO8
1800 PRINTS$;" +--+--+--+--+--+--+--+--+"
1810 IFI<8 THENPRINTS$;" :";
1820 FORJ=1 TO8:IFI=8 THEN1840
1830 SR=I*8+J:GOSUB310:PRINTSR$;":";
1840 NEXTJ:PRINT:NEXTI:W=0:HO=S+28
1850 VE=6:B$=" Weg    0":GOSUB1970
1860 VE=8:B$=" Over  64":GOSUB1970
1870 VE=14:B$=" V = Verb.":GOSUB1970
1880 VE=16:B$=" S = Stop":GOSUB1970
1890 RETURN
1900 :
1910 REM ONGELDIGE ZET
1920 GOSUB250:B$="Ongeldige zet !!"
1930 GOSUB1970:GOSUB2210:GOSUB110
1940 PRINTL$:RETURN
1950 :
1960 REM PLAATS TEKST
1970 GOSUB110
1980 FORI=1 TOLEN(B$)
1990 GOSUB2170
2000 PRINTMID$(B$,I,1);
2010 NEXTI
2020 RETURN
2030 :
2040 REM LEEG BUFFER HAAL TOETS
2050 GOSUB200:IFIN$<>"" THEN2050
2060 GOSUB210:TS=ASC(IN$)
2070 RETURN
2080 :
2090 REM JA / NEE
2100 T=0:GOSUB2050
```

```
2110 IF(TS=13) OR(IN$="J") OR(IN$="j") THENT=1
2120 IF(IN$="N") OR(IN$="n") THENT=2
2130 IFT=0 THEN2100
2140 RETURN
2150 :
2160 REM PAUZE KORT
2170 FORJ=1 TOVT:NEXTJ
2180 RETURN
2190 :
2200 REM PAUZE LANG
2210 FORI=1 TO20*VT:NEXTI
2220 RETURN
2230 :
2240 REM AFSLUITING
2250 GOSUB100:IFW=64 THEN2380
2260 PRINT:PRINT
2270 PRINTS$;"     ...
... volgende regels hebben afsluitingstekst van de auteur waarvan de website niet meer bestaat ...
... ik heb besloten deze daarom niet te implementeren ...
2380 HO=S+2:VE=14:B$="Tot een volgende keer."
2390 GOSUB1970:PRINT:PRINT
2400 RETURN
2410 :
2420 REM INIT + SCHERM OPMETEN
2430 DIMR(64):R(0)=1:CT=2:CN=0:Q=0
2440 L$="":FORI=1 TO25:L$=L$+"":NEXTI
2450 VE=0:FORI=0 TO150
2460 PRINT"*";:GOSUB120
2470 PH=I:IFVE<>0 THENI=150
2480 NEXTI:GOSUB100
2490 S$=" ":IFPH<=39 THEN2520
2500 X=INT((PH-39)/2)
2510 FORI=2 TOX:S$=S$+" ":NEXTI
2520 S=LEN(S$):GOSUB3400
2530 RETURN
2540 :
2550 REM TITELSCHERM UITLEG
2560 K$="SCHOONSPRINGEN   "
2570 R$="":X=INT(PH/LEN(K$))+1
2580 FORI=1 TOX:R$=R$+K$:NEXTI
2590 FORI=1 TO22:HO=I+1:VE=I:GOSUB110
2600 PRINTLEFT$(R$,PH-I-1):NEXTI
2610 HO=0:VE=17:B$=" een":GOSUB1970
2620 VE=18:B$=" programma":GOSUB1970
2630 VE=19:B$=" van":GOSUB1970
2640 VE=20:B$=" L.D.R.P. Looyenga":GOSUB1970
2650 VE=22:B$=" Wil je uitleg ? J/N ":GOSUB1970
2660 GOSUB2100:IFT=2 THEN3180
2670 :
2680 GOSUB100:PRINT:PRINT
2690 PRINTS$;" De opzet van dit denkspel is – met"
2700 PRINTS$;" de paardensprong – alle 64 velden"
2710 PRINTS$;" van het speelbord schoon te maken."
2720 PRINT
2730 PRINTS$;" De velden zijn genummerd  1 t/m 64."
2740 PRINT
2750 PRINTS$;" Als het startveld is opgegeven, komt"
2760 PRINTS$;" daar ## te staan."
2770 PRINT
2780 PRINTS$;" Nu moet er een sprong getal gegeven"
2790 PRINTS$;" worden, waarna de ##  naar het veld"
```

```
2800 PRINTS$;" met dit getal springt. Een zet naar"
2810 PRINTS$;" een leeg veld is ongeldig."
2820 PRINT
2830 PRINTS$;" S  i.p.v. een getal is stoppen en"
2840 PRINTS$;" eventueel opnieuw beginnen."
2850 PRINT
2860 PRINTS$;" Met V wordt een tikfout verbeterd."
2870 PRINT:PRINT
2880 PRINTS$;"      Verder na een toets... ";
2890 GOSUB2050
2900 :
2910 GOSUB100:PRINT:PRINT
2920 PRINTS$;"      De toegestane zetten zijn:"
2930 PRINT:PRINT:PRINT:L1$=LEFT$(L$,10)
2940 PRINTS$;L1$;"   +--+ +--+"
2950 PRINTS$;L1$;"   :  :  :  :"
2960 PRINTS$;L1$;"+--+--+  +--+--+"
2970 PRINTS$;L1$;":  :        :  :"
2980 PRINTS$;L1$;"+--+  +--+  +--+"
2990 PRINTS$;L1$;"      :  :"
3000 PRINTS$;L1$;"+--+  +--+  +--+"
3010 PRINTS$;L1$;":  :        :  :"
3020 PRINTS$;L1$;"+--+--+  +--+--+"
3030 PRINTS$;L1$;"   :  :  :  :"
3040 PRINTS$;L1$;"   +--+  +--+"
3050 GOSUB3210
3060 PRINTS$;"Nog een keer de zetten ?   J/N ";
3070 GOSUB2100:IFT=2 THEN3180
3080 :
3090 GOSUB100:PRINT:PRINT
3100 PRINTS$;"  Nogmaals de toegestane zetten:"
3110 PRINT:PRINT
3120 U1$=S$+L1$+":  :  :  :  :  :"
3130 U2$=S$+"       --+--+--+--+--+--+--"
3140 FORI=1 TO6:PRINTU1$:PRINTU2$:NEXTI
3150 PRINTU1$:GOSUB3210
3160 PRINTS$;"Na een toets gaan we spelen... ";
3170 GOSUB2050
3180 VT=INT(VT/3):RETURN
3190 :
3200 REM ZETTEN
3210 HO=S+17:VE=11:GOSUB1570:GOSUB2210
3220 RESTORE:FORZ=1 TO8:READZ$
3230 FORI=1 TO4:IFI=3 THEN3270
3240 X=I:IFI=4 THENX=3
3250 HO=S+VAL(MID$(Z$,(X-1)*5+1,2))
3260 VE=VAL(MID$(Z$,(X-1)*5+3,2))
3270 IF(I=1) OR(I=3) THENGOSUB1520
3280 IF(I=2) OR(I=4) THENGOSUB1570
3290 GOSUB2170:NEXTI:GOSUB2210
3300 IFZ=8 THEN3320
3310 HO=S+17:VE=11:GOSUB1570:GOSUB2210
3320 NEXTZ:HO=2:VE=20:GOSUB110
3330 RETURN
3340 :
3400 VT=40:RETURN:REM VERTRAGING
3410 :
25000 DATA"1711 1411 1109","1711 1709 1407"
25010 DATA"1711 1709 2007","1711 2011 2309"
25020 DATA"1711 2011 2313","1711 1713 2015"
25030 DATA"1711 1713 1415","1711 1411 1113"
```

```
30000 REM U KUNT DE SNELHEID IN ENKELE
30010 REM ONDERDELEN VAN HET PROGRAMMA
30020 REM WIJZIGEN MET DE WAARDE VAN
30030 REM   VT IN REGEL 3400
30040 REM
30050 REM PROGRAMMA GESCHREVEN IN
30060 REM BASICODE 2  DOOR:
30070 REM
30080 REM L.D.R.P. LOOYENGA
30090 REM STAMPERWEG 1
30100 REM 3813 SZ AMERSFOORT
30110 REM
30120 REM OP EEN YUPPIE PC
30130 REM
30140 REM NOVEMBER 1990
30150 REM
30160 REM COPYRIGHT NOS HILVERSUM
```

Meer informatie over Basicode kun je vinden in Wikipedia:
https://nl.wikipedia.org/wiki/Basicode

Op een andere pagina is informatie te vinden over hoe een Basicode programma wordt opgebouwd met de uitleg van de regels die ik hierboven ook heb vermeldt:
https://github.com/robhagemans/basicode/blob/master/BASICODE.rst

Woord raden

Dit is een leuk ZX-Spectrum programma dat je ook zelf gemakkelijk uit kunt breiden. Het programma is ook gemakkelijk te vertalen naar een ander BASIC dialect. Raadpleeg de drie converteerlijsten op de HCC Programmeren website.

Raad een woord door de letters in te voeren. Je zult versteld staan om het aantal beurten dat nodig is.

```
100 DIM a$(49): DIM w$(7): DIM o(7)
105 RESTORE :  READ a$: RANDOMIZE
110 LET beu=0: LET lg=0
120 LET k=7*INT (RND*7)+1: LET o$=a$(k TO k+7)
130 FOR i=1 TO 3: LET h$=o$(8-i): LET o$(8-i)=o$(i): LET o$(i)=h$: NEXT i
140 LET w$="*******"
150 CLS : PRINT "******** WOORD RADEN ********"
160 PRINT
170 PRINT "Het te raden woord bestaat uit"
180 PRINT "zeven letters."
190 PRINT "Probeer dit in een zo min moge-"
200 PRINT "lijk aantal beurten te raden."
250 LET beu=beu+1
260 INPUT "Geef een letter: ";I$
270 FOR i=1 TO 7
280 IF o$(i)=I$ THEN  LET w$(i)=I$: LET o$(i)="*": LET lg=lg+1
290 NEXT i
295 PRINT AT 15,0;"Het woord na ";beu;" maal raden"
300 PRINT "is: ";w$
310 IF lg<7 THEN  GO TO 250
320 PRINT ; FLASH 1;"GEFELICITEERD"
330 INPUT "Wilt u nog eens?";j$
340 IF j$="ja" THEN  GO TO 110
500 DATA "gnineporetsmahleewulfneoziesnethcawgnilawdgitsgna"
```

1. Grafische programma's

```
10 REM SLANG
20 PI=3.14159
30 SCREEN 2
40 FOR R=1 TO 80
50 CIRCLE(128,96),80-R,15,SIN(R/80*PI)*2*PI,R/80*PI
60 NEXT R
70 GOTO 70
```

```
10 REM VOGEL
20 PI=3.14159
30 SCREEN 2
40 FOR R=1 TO 80
50 CIRCLE(128,96),R,15,SIN(R/80*PI)*2*PI,R/80*PI
60 NEXT R
70 GOTO 70
```

```
10 REM KLEUREN SINUS
20 SCREEN 3
30 PI=3.14159
40 FOR X=50 TO 200 STEP 4
50 FOR Y=45 TO 145 STEP 4
60 PSET(X,Y),((Y/6.25+SIN((X-50)/75*PI)*8+8)MOD 15+1)
70 NEXT Y
80 NEXT X
90 GOTO 90
```

2. Zeilen

Zeilen is een zeer boeiend programma. Na het intypen van het RUN commando zien we rechts op het scherm een pijl. Hiermee wordt de windrichting aangegeven. Links zien we onze boot en bovendien zien we nog de boei waar we naartoe moeten varen. Door op de ESC toets te drukken kunnen we achtereenvolgens de stand van het roer en van het zeil selecteren. Het werkelijk instellen van de stand gebeurt met de cursortoetsen (←: met de klok mee draaien en →: tegen de klok in draaien).
Goede vaart!

```
10 TIME=0
20 OPEN "GRP:" FOR OUTPUT AS #1
30 PI=3.14159265#
40 X=20:Y=160
50 B=1:H=PI
60 SCREEN 2:COLOR 15,4,4
70 PRESET(200,120):PRINT #1,"O"
80 DRAW"BM255,91L10E5G5F5"
90 Q$=INKEY$
100 IF Q$<>"" THEN AS=ASC(Q$)
110 D$=INKEY$:IF D$<>"" THEN 110
120 IF AS=27 THEN R=NOT R
130 PRESET(230,0)
140 IF R THEN LINE(220,0)-(255,8),4,BF:PRESET(220,0):PRINT #1,"Roer"
150 IF NOT R THEN LINE(220,0)-(255,8),4,BF:PRESET(220,0):PRINT #1,"Zeil"
160 IF AS=29 THEN IF NOT R THEN NZ=ZH+PI/16 ELSE NR=RH+PI/50
170 IF AS=28 THEN IF NOT R THEN NZ=ZH-PI/16 ELSE NR=RH-PI/50
180 AS=0
```

```
190 V=V+COS(NZ-PI/2+PI-H)*SIN(NZ-PI/2+PI-H)-ABS(V/10)^1.5*SGN(V)
200 NH=H-SIN(NR)*ABS(V)
210 NX=X-ABS(V)*SIN(PI-NH)
220 NY=Y-ABS(V)*COS(PI-NH)
230 C=4:GOSUB 300
240 GOSUB 340
250 X=NX:Y=NY
260 H=NH:RH=NR:ZH=NZ
270 C=15:GOSUB 300
280 IF X>196 AND X<204 AND Y>116 AND Y<124 THEN SCREEN 0:PRINT TIME/50;"SEC":END
290 GOTO 90
300 LINE(X,Y)-(X+6*COS(ZH+H),Y+6*SIN(ZH+H)),C
310 LINE(A,B)-(D,E),C
320 CIRCLE(X,Y),4,C
330 RETURN
340 A=NX+4*SIN(NH)
350 B=NY-4*COS(NH)
360 D=A+6*SIN(NH+NR)
370 E=B-6*COS(NH+NR)
380 RETURN
```

3. Integreren

Met het onderstaand programma kan men de integraal van een functie tussen twee grenzen berekenen.
Allereerst wordt gevraagd om het aantal stappen waarmee de integraal zal worden berekend. In het programma wordt gebruik gemaakt van de regel van Simpson. De functie is in een subroutine regel 220 ondergebracht. In dat geval gaat het als voorbeeld om de derde macht van X.

```
10 CLS
20 INPUT"AANTAL STAPPEN:";N
30 INPUT"ONDERGRENS    :";A
40 INPUT"BOVENGRENS    :";B
50 H=(B-A)/N
60 FOR K=1 TO N-1
70 X=A+K*H
80 C=Q*2+4
90 Q=NOT(Q)
100 GOSUB 220
110 SOM=SOM+C*Y
120 NEXT K
130 X=A
140 GOSUB 220
150 FA=Y
160 X=B
170 GOSUB 220
180 FB=Y
190 I=H/3*(FA+SOM+FB)
200 PRINT "INTEGRATIE LEVERT:";I
210 END
220 Y=X*X*X
230 RETURN
```

4. Regressie en correlatie

De regressie- en correlatieberekening houdt zich bezig met de vraag hoe men door een gegeven aantal punten zo goed mogelijk een rechte lijn kan trekken.

Deze rechte lijn wordt door twee grootheden getypeerd, namelijk de hellingshoek M en het punt waar de lijn de Y-as doorsnijdt (B). Ieder punt wordt getypeerd door een X- en Y coördinaat. Het programma vraagt allereerst hoeveel punten er worden ingevoerd.

Na het invoeren van de coördinaten worden dan de waarden M en B berekend. Ten slotte wordt de correlatiecoëfficiënt R getoond. Als alle punten precies op een rechte lijn liggen, zal de absolute waarde van R gelijk aan 1 zijn. Indien de punten niet precies op een rechte lijn liggen, zal de absolute waarde van R minder dan 1 zijn. Kennelijk geeft de waarde van R aan hoe goed de punten wel op de berekende rechte lijn liggen.

```
10 CLS
20 INPUT "Aantal punten:";N
30 FOR K=1 TO N
40 PRINT "X";K;
50 INPUT "=";X
60 PRINT "Y";K;
70 INPUT "=";Y
80 XS=XS+X
90 YS=YS+Y
100 X2=X2+X*X
110 Y2=Y2+Y*Y
120 XY=XY+X*Y
130 NEXT K
140 M=(XS*YS/N-XY)/(XS*XS/N-X2)
150 PRINT "M=";M
160 PRINT "B=";YS/N-M*XS/N
170 NM=SQR((N*X2-XS*XS)*(N*Y2-YS*YS))
180 T=N*XY-XS*YS
190 PRINT "R=";T/NM
```

Foutjes met variabelen en IF statements

Er bestaan verschillende fouten. Grote fouten die meteen te zien zijn. Uitvoerfouten die door de compiler ontdekt worden. Er bestaan ook fouten die zo klein zijn, dat ze moeilijk te vinden zijn en zelfs door de compiler niet ontdekt worden. Zulke kleine foutjes worden bugs genoemd.

Elke programmeertaal gaat anders om met variabelen. Onderstaande regel wordt niet door elke programmeertaal ondersteund als de variabele nog geen waarde heeft.

```
X = X + 1
```

Het probleem ligt aan de initialisatie van de variabele. Indien de variabele geen waarde heeft, is deze niet gedefinieerd en mag de variabele niet in een expressie worden gebruikt. Veel BASIC programmeertalen, die vooral geen compiler hebben, geven hier geen foutmelding. Dat komt doordat elke variabele standaard de waarde 0 heeft en niet geïnitialiseerd hoeft te worden. Andere Basic dialecten en versies zien de fout wel, maar ook weer op verschillende manieren: tijdens het typen van de code of tijdens de uitvoer.

Het gaat niet alleen om het toekennen, zoals bovenstaande regel, maar ook om het controleren. Onderstaande code veroorzaakt een fout als de variabele nog geen waarde heeft.

```
X = 0
IF X = A THEN ...
A = 10
```

Als variabele A nog geen waarde heeft, is het IF statement ongeldig.

Bugs die alleen foute uitvoer veroorzaken, maar geen foutmeldingen geven

Er zijn meerdere wegen naar Rome. Dat kennen we. Verkeerde afslagen kunnen zorgen voor vreemde, foute, uitvoer, dat alleen wij opmerken, maar de compiler niet. Dat overkwam mij ook toen ik een C# project schreef in MonoGame. Een foutje die ik maakte in een IF statement, liet het mannetje wel over het scherm bewegen, maar de animatie werkte niet. Onderstaand deel van het project laat de animatie zien.

```
protected virtual void SetAnimations()
{
	if (velocity.X > 0)
		_animationManager.Play(_animations["WalkRight"]);
	else if (velocity.X < 0)
		_animationManager.Play(_animations["WalkLeft"]);
	else if (velocity.Y > 0)
		_animationManager.Play(_animations["WalkDown"]);
	else if (velocity.Y < 0)
		_animationManager.Play(_animations["WalkUp"]);
}
public virtual void Update(GameTime gameTime, List<Sprite> sprites)
{
	Move();
	SetAnimations();
	if (!(velocity.X == 0 && velocity.Y == 0))
		_animationManager.Update(gameTime);
```

```
        position += velocity;
        velocity = Vector2.Zero;
}
```

Bovenstaand deel werkt goed, maar eerder had ik een foutje gemaakt in het IF statement in de Update event methode. Er staat nu:

```
if (!(velocity.x == 0 && velocity.y == 0))
```

Hier wordt gecontroleerd of er met de pijltjestoetsen met het mannetje wordt gelopen. Als x of y niet 0 zijn, dan wordt de animatiebeheerder bijgewerkt met behulp van de speeltijd (gameTime). Wordt er geen toets ingedrukt, dan staat het mannetje stil en wordt de animatiebeheerder niet bijgewerkt. Voordat ik de oplossing wist, had ik onderstaande bug gemaakt:

```
if (!velocity.x == 0 && !velocity.y == 0)
```

Je zou zeggen dat deze regel net zo werkt, maar schijn bedriegt. Stel dat we op het pijltje naar beneden drukken, dan zal x == 0 zijn, waardoor de conditie onwaar is en de animatie niet zal werken. Ik probeerde de oplossing te vinden, totdat ik in de gaten had dat zulke condities alleen goed werken als ze eerst tussen haakjes staan en dan pas met een NOT (!) worden gecontroleerd. Als x == 0, dan is het natuurlijk weer onwaar, maar deze keer zal wel de animatie worden bijgewerkt dankzij de NOT operator. Een waarde met een eigen NOT operator ANDen zal leiden tot uitvoerfouten die lastig op te lossen zijn. Zet ze dus altijd tussen haakjes.

Een andere oplossing is een OR operator in plaats van een AND, zodat onderstaande regel ook werkt:

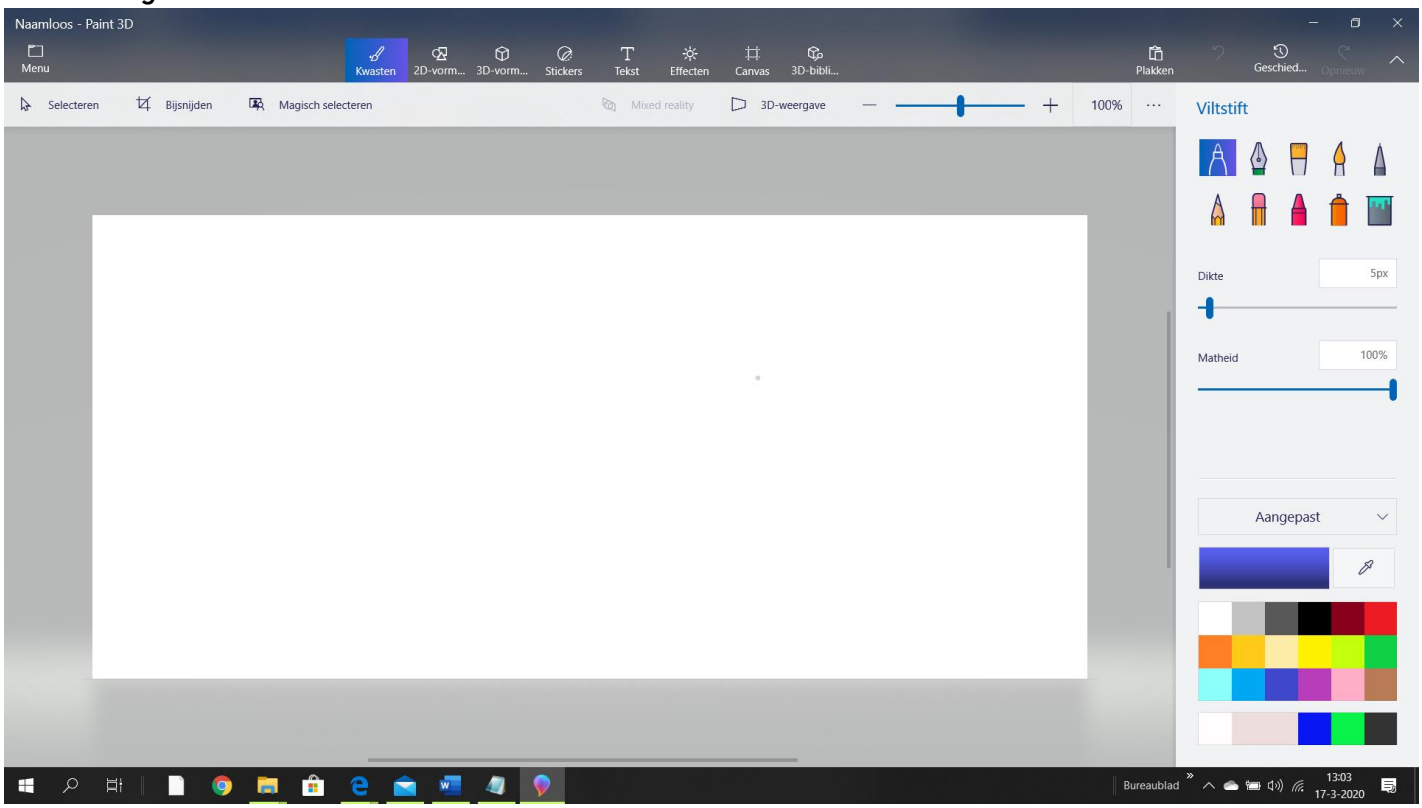
```
if (!velocity.x == 0 || !velocity.y == 0)
```

Is Paint3D beter dan de oude versie Paint?

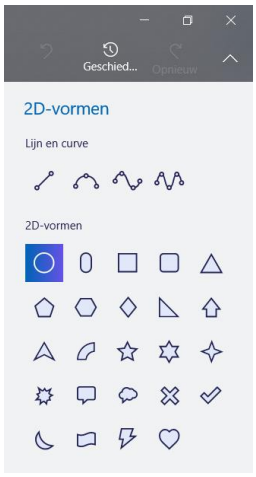
Soms hebben we Paint nodig om sprites, textures of achtergronden bij te werken. Paint is meer geschikt voor het bewerken van solid plaatjes. Een mogelijkheid om plaatjes te bewerken met een transparante kleur is niet mogelijk. Sprites maken, zoals Mario, werd dan ook in andere tekenprogramma's bewerkt die transparantie ondersteunen.

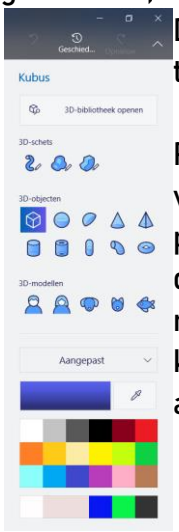
Paint3D is in vergelijking met Paint zeer uitgebreid en is in Windows 10 te vinden. Toch kan Paint af en toe nog wel nuttig zijn. In Paint3D ontbreekt bijvoorbeeld de grid, waardoor het moeilijker is om per vakje te bewerken.

Afbeelding: Paint3D



De naam Paint3D zegt het al: we kunnen ook 3D objecten tekenen. Paint3D heeft al objecten klaar, die op het tekenblad geplaatst kunnen worden. Net als in Paint kunnen we in Paint3D schilderen, zoals de afbeelding laat zien, maar niet met een grid.





Dat maakt het tekenen wat lastiger als we sprites en texturen willen tekenen.

Paint3D heeft veel 2D- en 3D-vormen om te tekenen. Het uitlijnen van de vormen is niet verbeterd. Men moet zelf uitzoeken of het plaatsen van de vormen goed gaat. Het snappen van de vormen, zodat ze netjes op het tekenblad staan, is ook niet aanwezig. Paint3D is nog steeds als Paint het meest geschikt om afbeeldingen bij te werken. Om echt goed te kunnen tekenen met belangrijke functies, zoals snappen, zal een geschikt tekenprogramma nodig zijn.

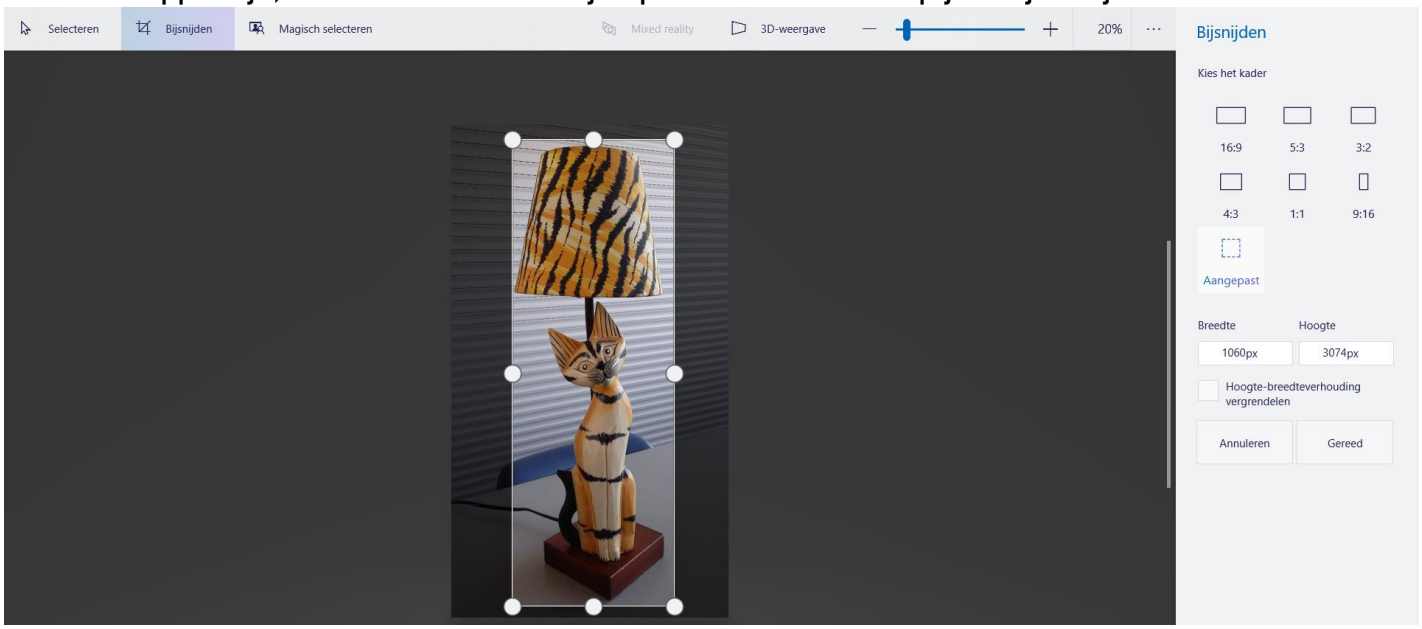
De canvas

Op de eerste afbeelding is het menu Canvas te vinden. De canvas is geschikt voor het maken van spritesheets en non-solid plaatjes. In de canvas kan de transparantie ingeschakeld worden, waardoor het tekenblad doorzichtig wordt. Als je terug gaat naar het tekenen van de vormen, blijft het tekenblad zo ingesteld.

Voor transparantie hoeft er geen kleur gekozen te worden. Bij de eerste afbeelding is rechtsboven op de tweede rij een gum te vinden. Als je met een bepaalde dikte gaat gummen, wordt het automatisch transparant. Staat de transparantie uit, dan zal het wit worden.

Bijsnijden

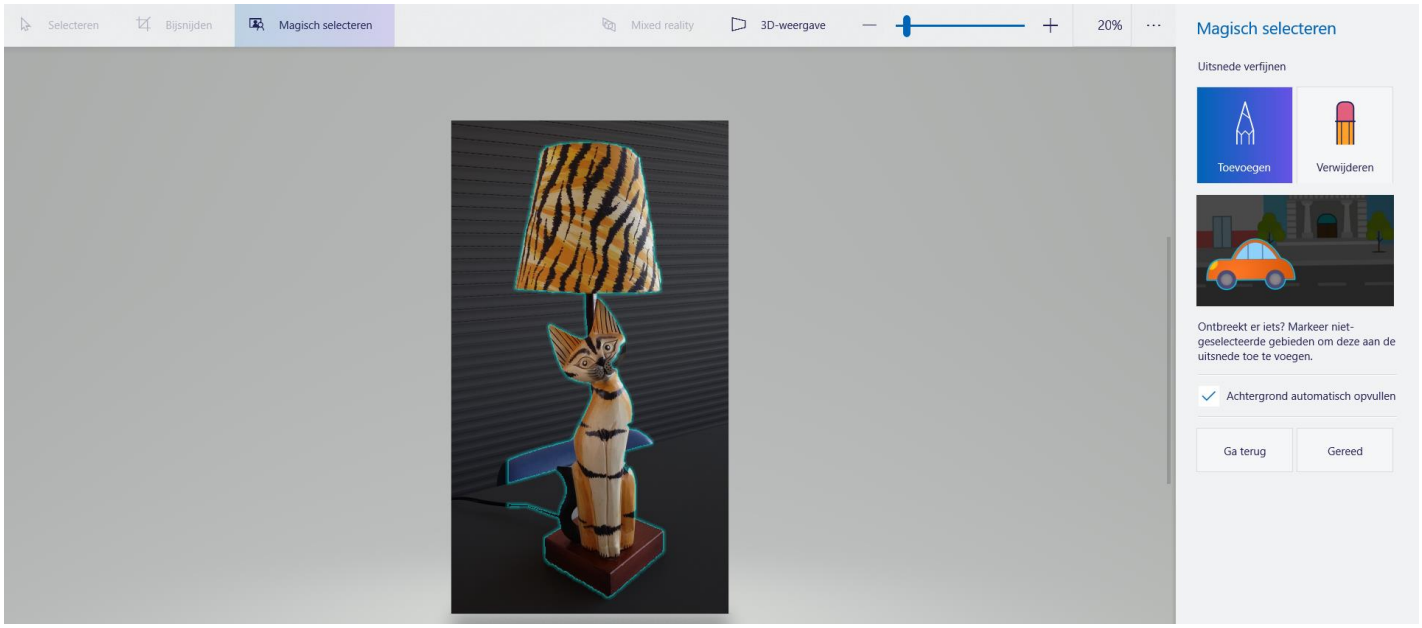
Wat ook verbeterd is tegenover Paint is het bijsnijden. Afbeeldingen kunnen preciezer worden bijgesneden door een rechthoek om een bepaald gebied te kiezen. In Paint selecteren we met behulp van een stippenlijn, maar in Paint3D klik je op de cirkels en sleep je tot je de juiste rechthoek hebt.



Heb je de juiste selectie, klik dan op Gereed, rechts van het tekenblad. De selectie wordt eruit geknipt, net zoals in Paint.

Magisch selecteren

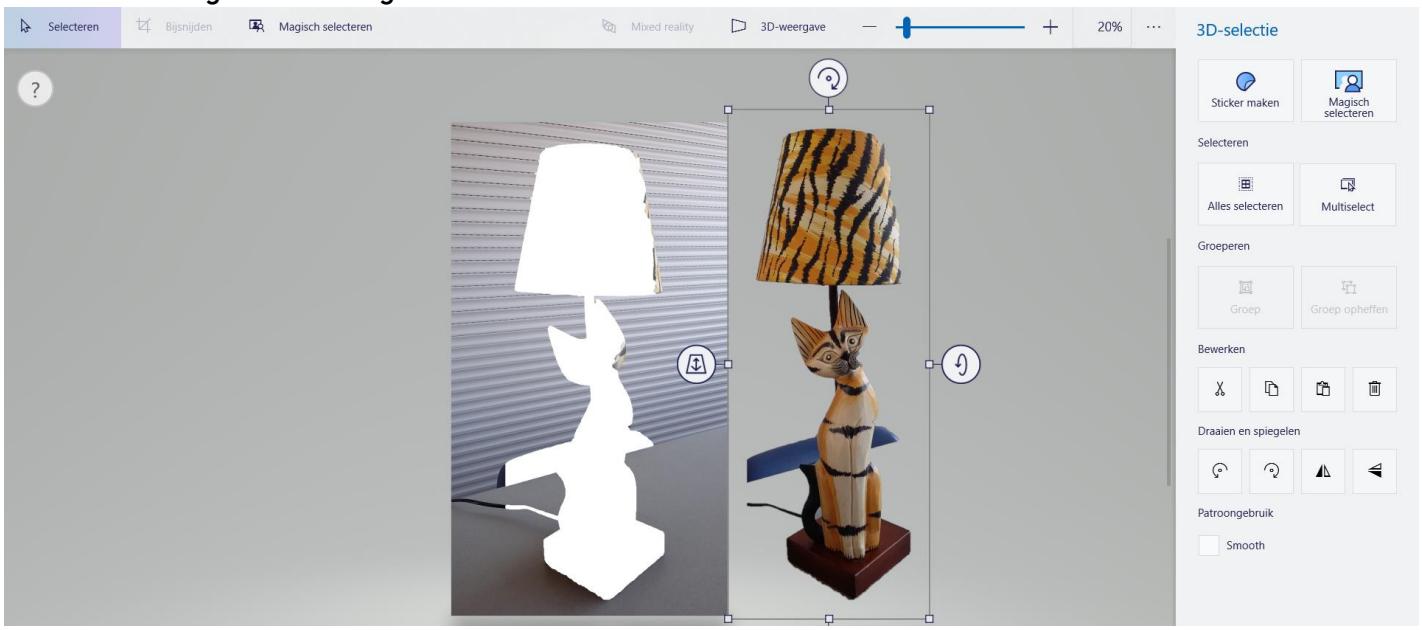
In Paint3D kun je, in plaats van een rechthoek maken, nu nog preciezer selecteren. We hoeven niet meer handmatig van een object een selectievorm te maken. Door Magisch selecteren te kiezen, doet Paint3D het nu voor je. Kies Magisch selecteren en snij eerst weer een rechthoek zoals bovenstaande afbeelding laat zien, maar nu met een andere rechterkant. Paint3D helpt je, met als voorbeeld een auto, met wat je moet doen.
Klik op Volgende om verder te gaan. Paint3D maakt een selectie om het object dat het beste uitkomt, zie onderstaande afbeelding.



Er is precies een selectie gemaakt om de lamp en de kat, maar ook om een deel van de stoel. Door op Verwijderen te klikken, kun je zelf een selectie op het deel van de stoel maken. Het deel zal automatisch uit de selectie verdwijnen.

Toch kan Magisch selecteren wel eens misgaan. Het zal niet altijd 100% goed gaan, en kleine randjes zullen niet mee geknipt worden. Op onderstaande afbeelding is dat goed te zien met de lamp.

Als je denkt dat je klaar bent, klik je op Gereed. Om het knipsel goed te kunnen zien, kun je deze naast de huidige afbeelding schuiven.



Het magisch selecteren is zeer handig voor het knippen van sprites uit spritesheets. Denk eraan dat je met magisch selecteren eerst de transparantie aanzet, anders komt het knipsel op een witte achtergrond.

Zie ook de video: Paint 3D: Magische selectie - Een item selecteren bewerken en verplaatsen op een foto
https://www.youtube.com/watch?v=L_zXtkT1PaU

Om een knipsel op te slaan, is een lastig werkje. Een normale selectie kan direct worden opgeslagen, maar een magische selectie niet. Ik heb een oplossing gevonden om het toch op te kunnen slaan.

Ga naar de Canvas en selecteer beide vinkjes uit, zodat alleen de canvas van grootte gewijzigd kan worden. Vergroot de canvas naar rechts zodat het knipsel in de canvas terecht komt. Sla het dan op en open daarna het bestand weer.

Kijk eerst bij Canvas of de transparantie aan staat. Kies dan Bijsnijden om het knipsel uit de afbeelding te halen. Klik op Gereed en alleen het knipsel zal op de canvas overblijven. Eigenlijk zou het fijn zijn als het knipsel direct opgeslagen kon worden zonder de rest. Echter blijft het natuurlijk Paint3D en geen groot tekenprogramma.

Voor kleine bewerkingen blijven Paint en Paint3D handig. Het is ook mogelijk om vanuit Paint naar Paint3D te gaan, door op het knopje Paint3D te klikken. Wil je door middel van het grid een sprite of een texture tekenen, maar deze wel op een transparante achtergrond hebben? Geen probleem. Sla de Paint tekening op en open deze in Paint3D. Zet in Canvas de transpantie aan en knip je tekening uit met behulp van Magisch selecteren of anders alleen Bijsnijden zoals bovenaan uitgelegd.