Scratch (24)

René Suiker

En toen was het al weer nummer 24

Tijdens de HCC!kennisdag (zie elders in dit nummer) was ik aan het kijken wanneer de Scratch-reeks begon; in mijn hoofd dacht ik aan 2020, maar blijkbaar doe ik dit al wat langer. Het eerste artikel over Scratch verscheen in SoftwareBus nummer 5 in 2018. Dus, voor de mensen die nu inschakelen en terug willen lezen, dat kan nog steeds. Via https://www.compusers.nl/softwarebus/softwarebus-online kun je de oude exemplaren van de SoftwareBus nog teruglezen en dat gaat terug tot en met 2006, dus nog voordat ik als hoofdredacteur begon (2009 nummer 3).

Dit is, zoals ik elders in dit nummer al schreef, de laatste keer dat de SoftwareBus als zelfstandig blad uitkomt. Dat wil niet zeggen dat ik nu ophoud met schrijven, want in de PC-Active krijgen we een herkenbare ruimte. Wel stop ik als hoofdredacteur, maar niet persé als auteur. Want schrijven is toch een hobby en schrijven over je hobby is dus twee keer leuk.

Wat gaan we doen?

We gaan natuurlijk weer Scratchen. Tijdens de HCC!kennisdag was er belangstelling, dus ik vermoed dat die er hier ook nog steeds is. Zo niet, dan is er een manier om mij de mond te snoeren. Want ik vind schrijven leuk, maar ik vind het niet leuk om de hele SoftwareBus vol te schrijven. Bovendien, dat kan ik ook niet, zo veel vrije tijd heb ik ook weer niet. Het zou dus mooi zijn, als andere mensen ook wat meer gaan schrijven, dan kan ik het wat minder doen. Iedereen blij. Dus, wil je helpen, meld je aan, via het contactformulier op onze website.

Kennisdag

Elders in dit nummer lees je over de terugblik op de kennisdag. Ik ga dat niet allemaal herhalen hier, maar ik wil een paar dingen wel memoreren die daar ook aan de orde kwamen. Dus nog even samenvattend, met name voor de nieuwe lezers (m/v/etc.).

- Je kunt bij Scratch komen via https://scratch.mit.edu
- Dit draait dus bij MIT, de beste TU ter wereld, denk ik
- Je kunt daar gratis een account maken
- Gratis is dan ook echt gratis, geen spam etc.
- Scratch wordt binnen CompUsers ondersteund door het Platform WebOntwerp.
- Via de site van CompUsers (www.compusers.nl)
- Kan je via de menubalk naar ‘Platforms’ en daarbinnen
- Naar Platform WebOntwerp
- Daar zie je rechts in het menu “Scratch”
- Dan kom je op: https://www.compusers.nl/webontwerp/scratch
- Daar vind je een presentatie over Scratch
- Daar vind je een opname van een Webinar over Scratch

De presentatie gebruik ik nog steeds tijdens onze evenementen, want het geeft redelijk overzichtelijk weer wat Scratch is en waarom je het zou willen gebruiken. En het geeft wat richtlijnen om aan de slag te gaan, begint met eenvoudige opdrachten en gaat door tot het wat moeilijker wordt.

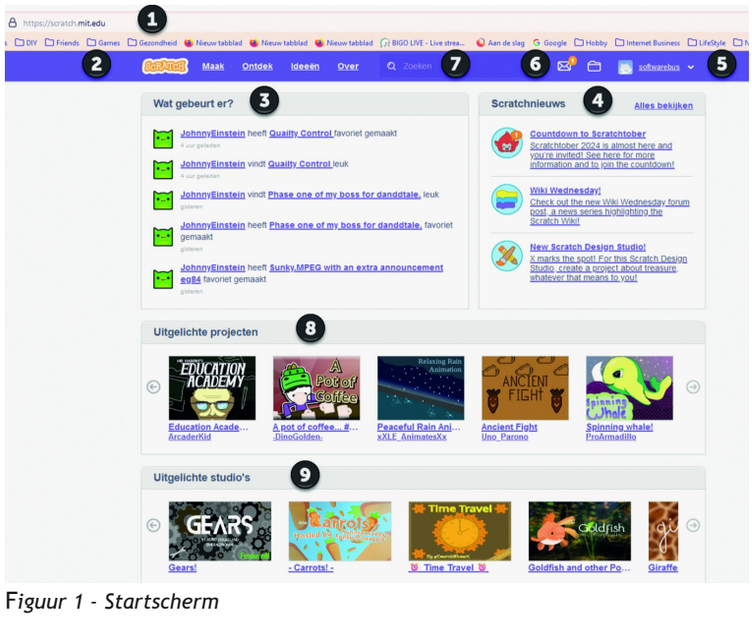
Ik kreeg dus wat ouders over de vloer met briljante kinderen, die Scratch al kenden en mijn vergelijking met Lego ook wel begrepen, al vond één van de kinderen Scratch toch beter te vergelijken met Duplo, omdat de mogelijkheden wel redelijk beperkt zijn. We waren het hier niet helemaal over eens, maar we konden wel overeenstemming bereiken over het feit dat je Scratch niet kon vergelijken met technisch lego.

Scratch leren

Je kunt dus Scratch leren, door een boek te kopen, door mijn artikelen te lezen of door een cursus aan te schaffen, maar je kunt ook gewoon op de site kijken wat er allemaal aangeboden wordt. En dan kan je in alle gevallen horen of lezen wat je kan en hoe dat moet, maar het beste leer je Scratch (en elke programmeertaal) het best door het gewoon te doen. En daarvoor zitten dus steeds opdrachten in mijn artikelen verweven, maar ik krijg de indruk dat die niet allemaal uitgevoerd worden. En dat is jammer, want juist door het maken van de opdrachten en kijken waar je vastloopt voordat ik de oplossing geef, daar kom je het verst mee. Natuurlijk mag je wel een keer vooruit kijken als je er echt niet uitkomt, maar juist door het te proberen zie je wat er wel en wat er niet werkt. Zelf ben ik ook geen programmeur. Hoewel ik het wel leuk vind, ben ik zelf meer van de hoofdlijnen, de oplossingsrichting, en niet zo zeer van de detailuitwerking. Daar heb ik eigenlijk het geduld niet voor.

Weer even: startscherm

Ik ga niet finaal opnieuw beginnen in deze reeks, maar ik blik toch wel even terug, want zo blijven zaken misschien toch iets beter hangen. Dus ik ga nog eens naar het startscherm, dus waar ik terecht kom als ik naar Scratch ga:



Figuur 1 - Startscherm

Bij (1) zie je dus de URL: https://scratch.mit.edu

Bij (2) zie je de menubalk en als je op het Scratch-symbool klikt kom je ook weer op deze pagina terecht. Bij (3) en (4) zie je dynamische inhoud. Bij (3) is dat ‘wat gebeurt er’ en bij mijn weten zijn dat de laatste publieke activiteiten binnen mijn netwerk. Als ik de pagina ververs kan hij er dus al

weer heel anders uitzien en als jullie naar deze pagina gaan zien jullie ook iets heel anders. Bij (4) zie je ‘Scratchnieuws’ en dat is dus nieuws van Scratch zelf. Ook dit is dynamisch, maar iets minder dynamisch dan de activiteiten, tenminste, meestal.

Bij (5) zie je dat ik al ingelogd ben en wel als ‘softwarebus’. Dat is één van mijn accounts en het account dat ik het meest gebruik als ik artikelen aan het schrijven ben. Als ik op ‘softwarebus’ klik krijg ik een profiel submenu, waarover later meer.

Bij (6) zie je dat er één bericht voor me klaar staat. Dit is in dit geval een bericht dat iemand mij volgt, maar het kunnen verschillende soorten berichten zijn.

Bij (7) kan je zoeken, zowel op een naam van een mede Scratcher als op een project als iets anders, zoals bijvoorbeeld een studio.

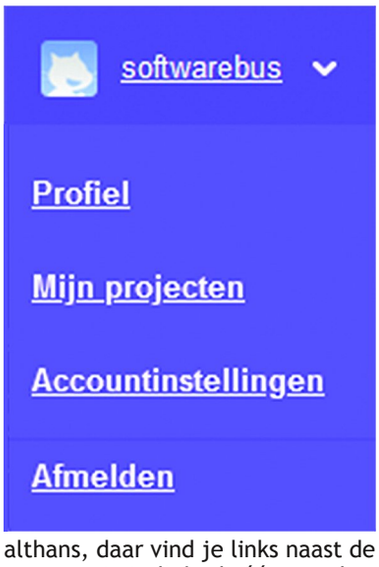
Bij (8) krijg je, ook dynamisch, een overzicht van een aantal ‘uitgelichte projecten’, waarbinnen je ook kunt scrollen. Dit is wel gebaseerd op je persoonlijke voorkeuren.

En bij (9) heb je, wederom dynamisch, een overzicht van uitgelichte studio’s.

Projecten zijn dus enkelvoudige Scratch programma’s; elk Scratch programma is in feite een project. Als je in de menubalk (2) op ‘Maak’ klikt wordt er een nieuw project aangemaakt. Intussen zijn er al meer dan 1 biljoen projecten binnen Scratch, dus ruim duizend miljard. Studio’s zijn in feite groepen van projecten. Je kunt als Scratcher meerdere studio’s aanmaken en beheren en je kunt ook soms gevraagd worden om studio’s van anderen te beheren. Het concept van studio’s kunnen we in een volgend nummer wel eens uitgebreid op in gaan. Voor vandaag is het genoeg, dat je weet dat je je projecten in studio’s kunt onderbrengen. Op die manier kun je je projecten thematisch organiseren.

Profiel submenu

Ik zou nog even terugkomen op het profiel submenu, dus als je in het voorgaande scherm op ‘softwarebus’ zou klikken. Je krijgt dan dit submenu te zien:

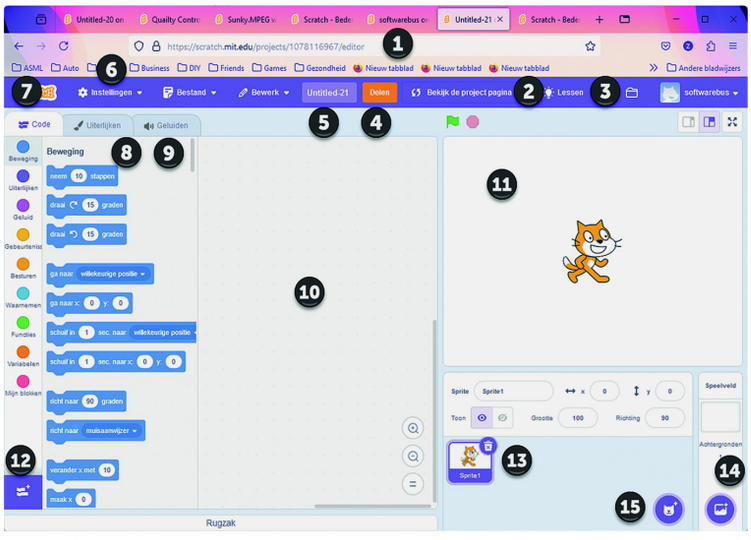


Figuur 2 - Profiel submenu

Hier kan je dus op een aantal opties klikken. Allemaal niet heel spannend, maar het geeft wel aan, dat je al je projecten weer terug kunt vinden, je kunt jezelf afmelden en je kunt wat gegevens over jezelf onderhouden. Via ‘mijn projecten’ kom je ook bij ‘mijn studio’s’, althans, daar vind je links naast de projecten nog een submenu met studio’s als één van de opties.

Project maken

Omdat herhaling af en toe wel goed is, druk ik nu weer een keer op ‘Maak’ en loop nog een keer langs alle opties:



Figuur 3 - Nieuw project

Hier is dus enorm veel te zien en het lijkt me nuttig om dit nog eens op te sommen. Misschien is het voor sommige lezers niet nodig, dus die slaan dan maar even een paar stukken over.

Bij (1) zie je de URL, die is nu iets langer, want hier zie je ook de projectcode in de URL staan. Tevens zie je aan het eind ‘/editor’ om aan te geven dat je in de bewerkmode zit. Als je op (2) klikt, dan kom je op dezelfde URL minus het stukje ‘/editor’ en dan zie je beschrijvende pagina. Voor een nieuw project is dat beslist niet spannend, maar misschien dat ik ook daar nog een keer op terugkom.

Bij (3) zie je de link naar ‘lessen’ om Scratch te leren op de site van Scratch. Hier zijn zo’n 25 handleidingen over verschillende aspecten van Scratch te vinden. Kijk er gerust eens rond, het is de moeite waard.

Bij (4) vind je de optie om je project te delen. Zoals ik al eens eerder zei: dit wordt door de Scratch gemeenschap gewaardeerd. Het idee is, dat je samen meer kunt dan alleen. Als je echt nog helemaal in het begin van je project zit heeft delen nog niet heel veel zin, maar als je je project afgerond hebt en je wilt het anderen laten zien, dan moet je het delen, anders kan niemand anders er bij. En als je het gedeeld hebt, dan kunnen anderen ook naar je project kijken en ze kunnen er zelfs een kopie van maken, die ze dan verder kunnen bewerken. Op hun projectpagina zal dan wel staan, dat hun project gebaseerd is op jouw werk. Dus ze lenen je werk, maar je krijgt er wel de credits voor.

Bij (5) geef je je project een naam, want standaard heten ze allemaal ‘untitled-xx’ waarbij xx een volgnummer aangeeft. Dus als softwarebus heb ik blijkbaar al 20 projecten gemaakt. Voor Scratch maakt het overigens niet echt uit als je al je projecten hetzelfde noemt, want die onderscheidt de projecten op basis van het volgnummer in de URL. Het is alleen voor jezelf verwarrend, dus beter is om een logische naam te kiezen.

Bij (6) vind je de instellingen. Voorheen was dit de wereldbol, waar je de taal kon instellen. Dat kan nog steeds en er is nog steeds een overdaad aan talen beschikbaar, maar daarnaast kan je hier nog een kleurschema instellen.

Onder (7) zie je drie tabbladen en de linker is het blad ‘Code’. Hier zie je in verschillende kleuren de verschillende groepen van commando’s die je in je programma kunt gebruiken. Bij (8) vind je uiterlijken, dat zijn de vormen van de objecten (sprites) die je in je programma gebruikt. Bij (9) vind je de geluiden die door je sprites voortgebracht kunnen worden. Overigens kan je ook geluiden aan je achtergrond koppelen.

De commando's zijn dus onderverdeeld in een aantal groepen, welke door een kleurcode herkenbaar zijn. De gekleurde bolletjes laten je snel naar een categorie springen, want de lijst met commando's staat gewoon onder elkaar, maar wel gesorteerd per categorie. Met deze blokjes stel je je programma samen in het werkgebied (10).

Bij (11) vind je het speelveld. Hier vinden alle activiteiten plaats. Je kunt dit met het icoontje rechts boven in het speelveld ook maximaliseren. Als je programma af is, kan je het dus wat groter afspelen.

Bij (12) vind je een knop om nog wat extra modules in te schakelen. Toen we van versie 2 naar versie 3 gingen was ineens de pen uit de standaard verdwenen. Die werd blijkbaar niet heel veel gebruikt door de Scratchers, dus zijn de pencommando's naar een aparte module verhuisd en als je deze wilt gebruiken, zal je deze moeten toevoegen. Niet heel veel werk, je drukt op het icoontje bij (12) en je selecteert de pencommando's. Een kind kan de was doen.

Bij (13) vind je de sprites die onderdeel uitmaken van je project. Boven de geselecteerde sprite (in dit geval is er maar één) zie je wat parameters betreffende de sprite, zoals de naam, de positie en nog wat parameters.

Bij (14) kan je achtergronden tonen die meedoen. Met de knoppen bij (15) kan je sprites respectievelijk achtergronden toevoegen. Hierbij kan je een keuze maken uit beschikbare materialen, of zelf tekenen, of uploaden. Je kunt dus bijvoorbeeld in Blender een fantastische achtergrond ontwerpen en deze dan in Scratch importeren.

Dit even als herhaling van het ontwikkelscherm. Weten we het weer? Zijn er nog vragen? Ik hoor niets, dus het zal wel duidelijk zijn.
Als je nog eens goed naar het werkgebied (10) kijkt, zie je in de rechterbovenhoek de afbeelding van de sprite. Als je meerdere sprites hebt, dan vind je in het werkgebied de afbeelding van de geselecteerde sprite. En in het werkgebied vind je de code die behoort bij die sprite. Elke sprite is een eigen object, die eigen code kan hebben. In plaats van het traditioneel sequentieel programmeren bekijk je nu per object wat die moet doen.

En op die manier bouw je object voor object je programma op. Op deze manier bouwen we ook redelijk compacte programma's. Als we nog eens terugkijken naar bijvoorbeeld 'Bricks' dat we in een aantal voorafgaande nummers bespraken, dan kan je je misschien wel voorstellen, dat je daar in bijvoorbeeld Basic ettelijke tientallen pagina's aan code voor nodig hebt. In Scratch heb je ongeveer één pagina per actieve sprite nodig en dan zijn de codeblokken die je gebruikt nog best groot.

Programmeren

Scratch is dus een programmeertaal, hiermee kan je dus je computer programmeren. Je programma wordt op de server van MIT geïnterpreteerd en regel voor regel uitgevoerd. Het resultaat zie je in de browser. We hebben het natuurlijk vaak over programmeren gehad, maar ook hier blik ik nog even terug.

Programmeren is dus de computer instructies geven wat te doen. En let op, de computer is in feite maar een dom apparaat, hij voert de instructies die hij krijgt letterlijk uit. Hij vraagt zich nooit af of het wel logisch is wat jij wilt en is niet in staat om gedachten te lezen.

Je kunt een programma opstellen door reeksen van instructies achter elkaar te zetten. Je moet ze in feite, net als Lego-stenen, in elkaar klikken. Als het niet 'klikt', dan is de combinatie van instructies in die volgorde niet werkend.

Je kunt in feite geen fouten in de syntax maken. In Basic, bijvoorbeeld, kan je een fout maken door twee lussen niet om elkaar heen te zetten (dat is goed) maar in feite door elkaar heen te laten lopen. Dat kan dus niet, dat geeft een fout, maar je kunt de code wel zo opschrijven:

```
100 FOR I = 1 to 10
110 FOR J = 1 to 10
120 PRINT I; " * "; J; " = "; I * J
130 NEXT I
140 NEXT J
```

In Scratch krijg je zo'n fout niet voor elkaar. Je kunt een lus in een lus zetten en daarbinnen nog een lus enz. Maar je kunt lussen niet in elkaar zetten zodat ze rare dingen doen. Het gaat gewoon niet passen.

Dat wil overigens niet zeggen dat je geen programmeerfouten zou kunnen maken. Je kunt nog steeds fouten maken, maar dan doet de computer nog steeds wat je zegt. En je kunt je programma ook altijd stoppen met de rode stip, ook als je in een eeuwige lus zit.

Het is dus zaak om goed na te blijven denken over wat je doet. En daarom zijn er dus ook de oefeningen, zodat je langzaamaan van absolute beginner doorgroeit naar redelijk gevorderd programmeur. En dus is het zaak om de oefeningen ook uit te voeren, als je zelf actief wilt zijn met Scratch en vooral als je het ook je (klein)kinderen wilt uitleggen.

Vroeger, toen ik nog leerde programmeren, werden we gestimuleerd om eerst bijvoorbeeld stroomdiagrammen te maken om aan te geven wat het programma ging doen. Zeker voor de wat grotere programma's is dat nog steeds een aanrader. Gewoon beginnen met coderen maakt het lastig om achteraf denkfouten terug te vinden. Maar het kan vaak nog wel, bijvoorbeeld door je variabelen zichtbaar te maken. In Scratch is dat een fluitje van een cent en hoef je niet vele regels code door om op die manier fouten te zoeken. Je kunt elke variabele op je scherm zetten en dat doe je bij de variabele zelf. Als je programma dus werkt, zet je deze weer uit en heb je er geen last meer van

Tot zover de terugblik

Nu dan weer verder waar we gebleven waren. We kijken even terug naar het huiswerk en zien wat er van is terecht gekomen.

De opdrachten moesten je aan het denken zetten. Ik heb uitgelegd hoe je zo'n probleem aan kan pakken en dit toegepast op een aantal vraagstukken.

Voor het bekijken van het huiswerk moet je misschien nog wel even terugslaan naar de vorige SoftwareBus, want ik ga niet alle teksten herhalen. De projecten werden in nummer 23 uitvoerig beschreven, ik richt mij nu op het huiswerk.

Overigens, ik realiseerde me dat de opdrachten wel heel erg moeilijk waren, dus ik ga nu wat oplossingsrichtingen geven en laat jullie dan hiermee concreet aan de slag gaan. Hopelijk leidt dit wel tot wat inzendingen.

Elektronica bouwdoos

Opgave 23.1
Bedenk een manier waarop tijdens het spelen de gebruiker het object op kan pakken, kan draaien en neer kan leggen. Een 'auto-snap' om 'm automatisch met de contactpunten door te verbinden zou helemaal te gek zijn.
Dit is natuurlijk helemaal geen makkelijke opgave en ik ga 'm nu ook nog niet helemaal uitwerken. Maar alvast wel een tipje van een mogelijke sluier optillen. Er zijn in de wereld

van programmeren vaak meerdere oplossingen mogelijk. Wat is nu het best? Het best is de oplossing die:

1. Doet wat die moet doen
2. Niet doet wat die niet moet doen
3. Binnen een redelijke termijn

Ofwel, de eerste oplossing die werkt is de beste, totdat een betere oplossing langskomt. Een beetje net als de wetenschap, een stelling is waar, totdat het tegendeel bewezen is. Met dat in het achterhoofd stel ik het volgende voor:

We hebben dus het blok, dat ik de vorige keer beschreef, waarop de componenten geplaatst moeten worden. Daarnaast stellen we de componenten op. Dan heb je de mogelijkheid om gewoon zo veel componenten neer te leggen als er gebruikt mogen worden, maar het leuke van een systeem als Scratch is dat je er ook voor kunt kiezen om alle componenten die je hebt oneindig veel keren te gebruiken. Dit doe je dus door een component te klonen op het moment dat die geselecteerd wordt. Vervolgens ga je met de kloon aan de slag.

Aangezien je de beweging zelf kunt controleren kan je ‘m steeds een aantal stappen laten bewegen, zodat die altijd van het ene plaatsingspunt naar het andere beweegt en dus niet ertussen weggelegd kan worden. Als iemand dat toch lukt en hij ligt dus los op het bord, zou je er voor kunnen kiezen om de kloon te laten verdwijnen.

Dan hebben we nog niet het draaien opgelost, maar we doen ook niet alles tegelijk, want daar heb ik helemaal niet de ruimte voor. Brengt ons dus bij:

Opgave 24.1 (nieuwe opgave dus)
Bedenk hoe je een object wilt draaien.
Het draaien houdt in, dat in plaats van een horizontale aansluiting, je het object verticaal wilt plaatsen, hetzij naar beneden, hetzij naar boven, ten opzichte van één van de geselecteerde punten.

Nu heb ik eigenlijk al bijna alles verklapt dus verras mij met een concreet uitgewerkte oplossing.

Opgave 23.2
Bedenk alvast hoe je zou willen vaststellen of er een correct elektrisch circuit is gevormd.
Hiervoor moet je in feite beginnen bij de batterij. Kijk of er vanuit de pluspool van de batterij een verbinding loopt naar de minpool van de batterij. Je weet waar de pluspool is, want je weet de positie van de batterij. Je weet ook, waar de mogelijke verbindingen kunnen liggen. Als er meerdere liggen ga je beoordelen of deze uiteindelijk uitkomen bij de minpool. En dan ga je ook nog eens kijken of er componenten tussen zitten die de stroom beperken. Want we willen natuurlijk geen kortsluiting. Hiermee hebben we in feite het begin van de oplossing.

Opgave 24.2
Werk dit uit in een concrete code. Waar zou je die code aan op willen hangen? Hint: je kunt je afvragen of je misschien met een vaste batterij wilt werken, of dat je ook batterijen onbeperkt mag toevoegen.

Mens erger je niet

Opgave 23.3:
Laat pionnen over het bord schuiven. Welke problemen ervaar je?
Een deel van de problematiek had je natuurlijk ook al bij het plaatsen van de elektronische componenten. Daarnaast moet je ook rekening houden met het feit, dat we tweedimensionaal werken. Je kunt dus niet echt pionnen laten zien, op dit speelbord kijk je in feite vanaf de bovenkant. Je moet dus pionnen een afwijkende vorm geven, zodat je ze wel herkent op het speelbord. Verder moet je denk ik alle posities nummeren.

Tijdens het lopen moet je dus stapje voor stapje gaan overeenkomstig het aantal gegooide ogen. Je moet dus de hoeken in de gaten houden. Verder moet je ook in de gaten houden of je al bij je eigen kleur bent, zodat je naar binnen moet. Al met al moet je dus veel zaken in de gaten houden.

En dan speelt nog mee, de speler gooit en soms zijn er twee of meer opties die de speler kan uitvoeren. Op dat moment moet het programma de keuze ook aan de speler laten. Dus misschien moet er nog ergens een soort display komen waarin boodschappen van de computer aan de speler staan. Overigens, als de computer meerdere opties heeft, dan is de afweging als volgt:

1. Kan ik één van mijn pionnen in veiligheid brengen, dan doe ik dat; anders
2. Kan ik een pion van een tegenstander slaan, dan doe ik dat; anders
3. Kan ik een nieuwe pion in het veld brengen, dan doe ik dat; anders
4. Kan ik een pion dusdanig bewegen, dat er niet onmiddellijk een tegenspeler achter zit, dan doe ik dat; anders
5. Speel ik de pion die het dichts bij de thuishaven zit

Ik weet niet of dit de slimste keuze is, maar je mag gerust iets beters verzinnen. Heb je al een idee hoe je dit wilt implementeren?

Opgave 24.3
Werk concreet uit, dat een pion telkens het geworpen aantal ogen vooruit gaat over het bord. Je hoeft nog geen rekening te houden met je eindpunt, alleen per worp het juiste aantal stappen zetten.
Je moet hiervoor dus het spelbord gebruiken en gebruik maken van de waarde van ‘worp’ om te bepalen hoeveel stappen gezet moeten worden.

Waar gaan we verder Scratchen?

Op 2 november hebben we dus de ALV van CompUsers en wordt gekeken hoe we verder gaan als CompUsers. Er zijn verschillende opties, maar één ervan kan ook zijn, dat CompUsers wordt opgedoekt. Hoe gaan we dan verder met Scratch?
Er zijn een aantal opties, die ik even door wil nemen. Ik durf niet te beweren dat ik compleet ben, maar ik zie de volgende opties:

1. Ongeacht hoe, ik kan blijven schrijven voor de SoftwareBus, ook als dat een katern binnen PC-Active wordt
2. Binnen Hcc!AI wordt ook volop aandacht aan Scratch besteed, misschien een leuke optie voor mijn lezers
3. Binnen Senioren Academie wordt soms ook Scratch behandeld. Vaak ook met daarbij hardware aansturing. Daar hebben wij het nog niet echt over gehad
4. Je kunt je voorstellen dat Hcc!programmeren ook aandacht aan Scratch gaat besteden. Nu is dat nog niet echt het geval, hoewel op sommige vrijdag avond sessies Scratch wel eens aan de orde komt.

Al met al hoeven we Scratch zeker niet op te geven, zachtjes aan krijgt de jeugd belangstelling.

Opgave 24.4
Verzin eens wat ideeën wat we met Scratch zoal kunnen bouwen. Misschien wil je al eens een opzet maken en het hier verder laten groeien en bloeien. Misschien zijn er ook nog specifieke onderwerpen die hier besproken kunnen worden.