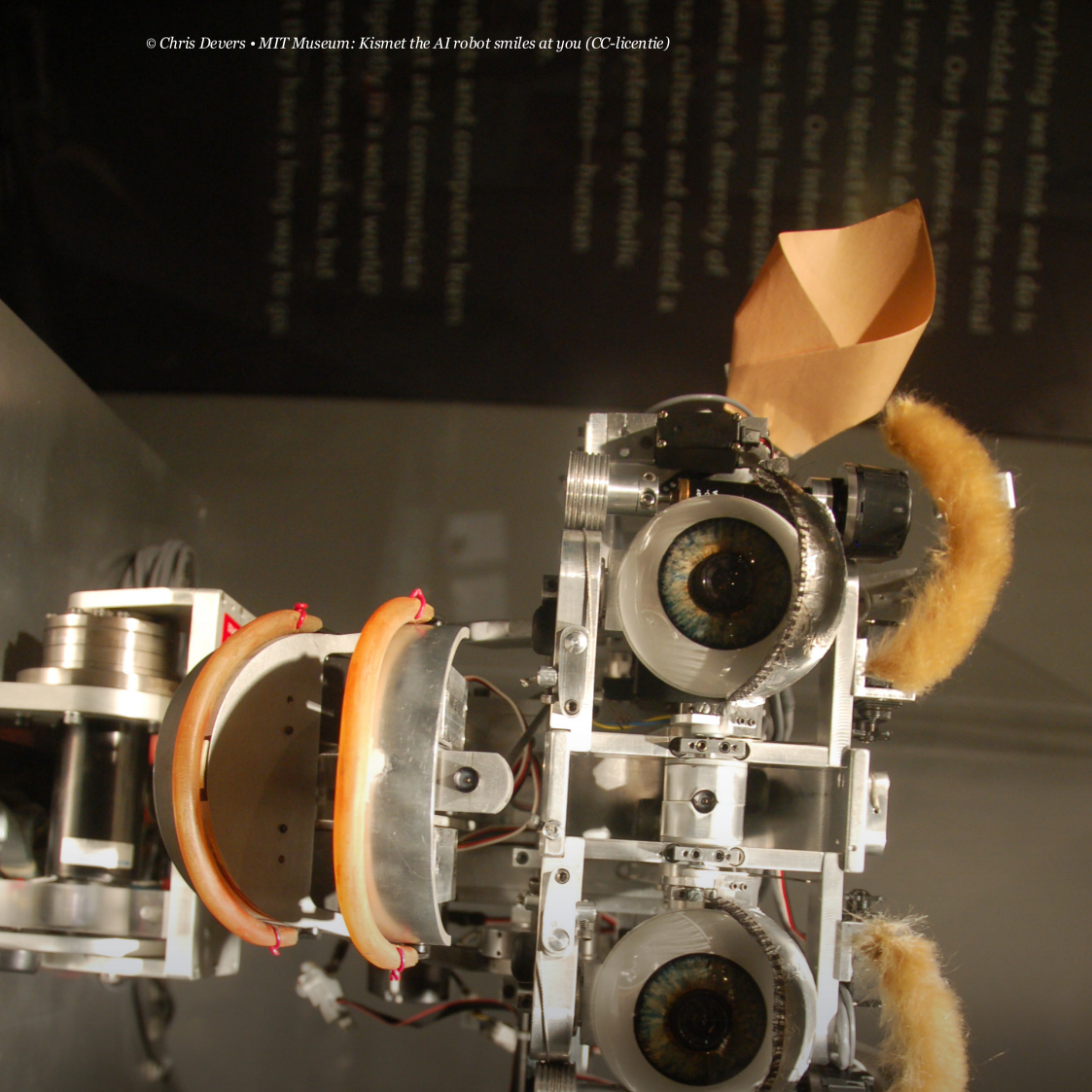
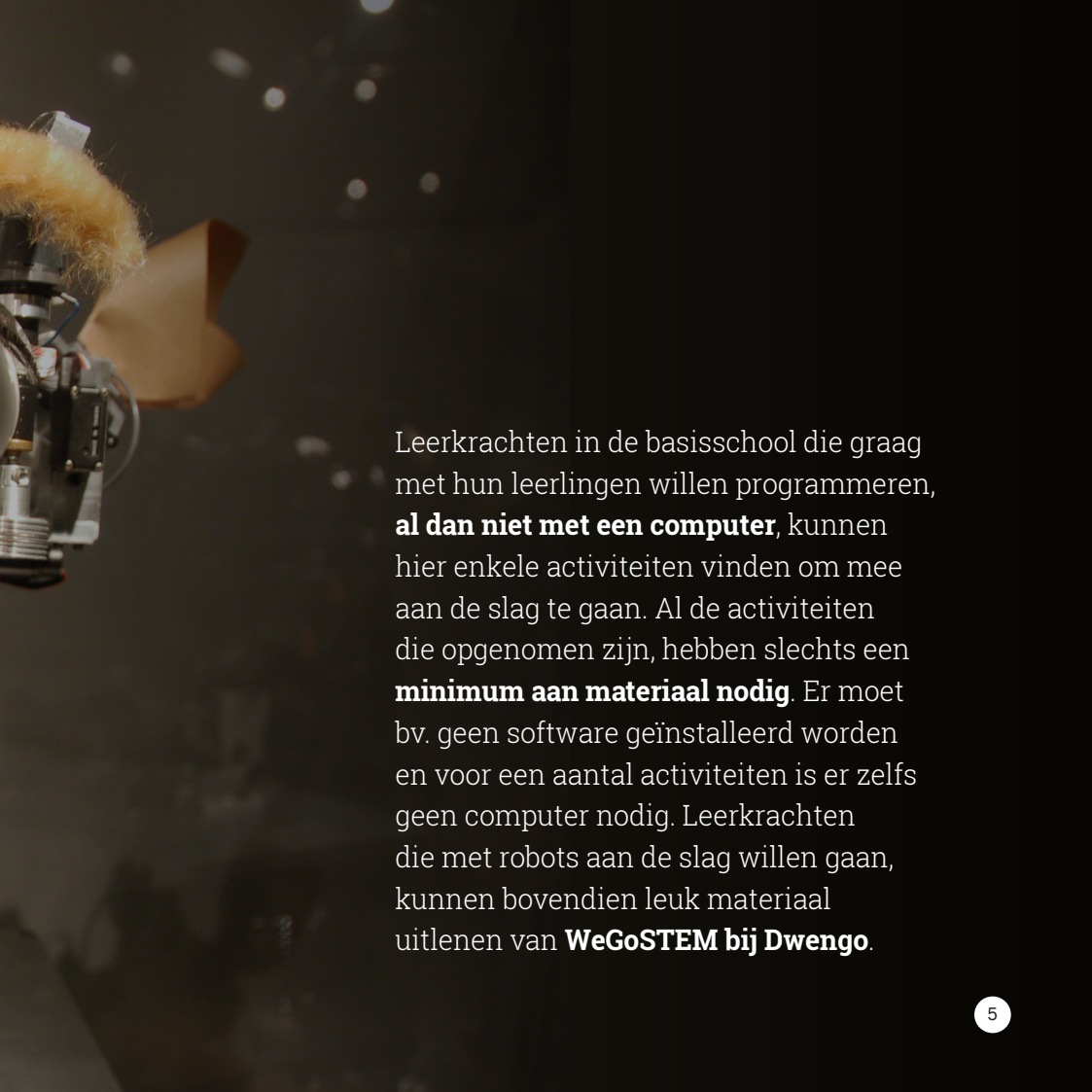
Ik kan al programmeren

Kant-en-klare projecten voor het basisonderwijs

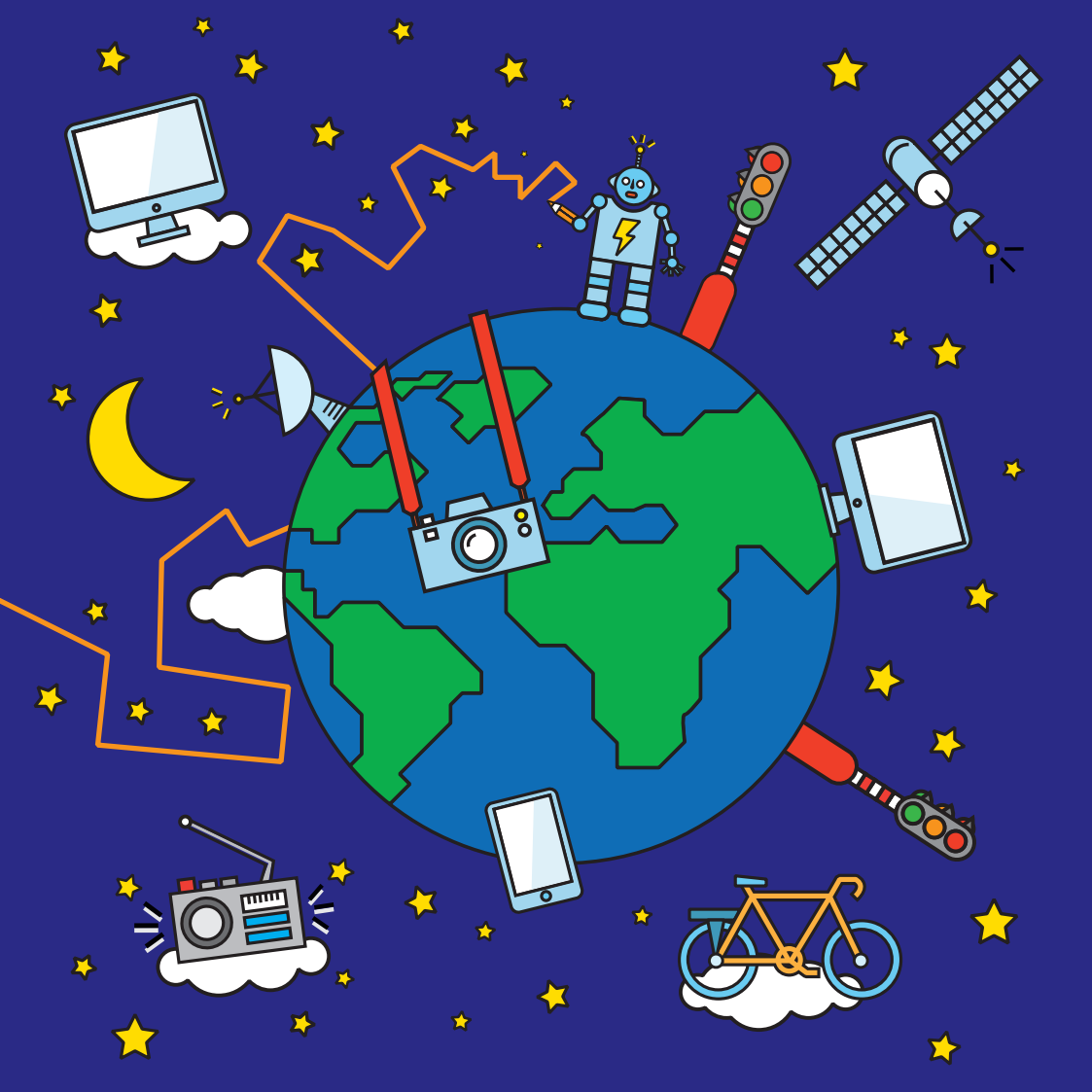
Dirk De Muynck, Natacha Gesquière en Francis wyffels

INHOUD


© Chris Devers • MIT Museum: Kismet the AI robot smiles at you (CC-licentie)



Leerkrachten in de basisschool die graag met hun leerlingen willen programmeren, **al dan niet met een computer**, kunnen hier enkele activiteiten vinden om mee aan de slag te gaan. Al de activiteiten die opgenomen zijn, hebben slechts een **minimum aan materiaal nodig**. Er moet bv. geen software geïnstalleerd worden en voor een aantal activiteiten is er zelfs geen computer nodig. Leerkrachten die met robots aan de slag willen gaan, kunnen bovendien leuk materiaal uitlenen van **WeGoSTEM bij Dwengo**.



1 Een wereld vol technologie

In de maatschappij is technologie alom aanwezig: smartphones, robotgrasmaaiers, drones, intelligente zeepdispensers en binnenkort ook zelfrijdende auto's. Onze wereld is sterk gedigitaliseerd. Computers en andere intelligente apparaten zijn onmisbaar geworden, zowel op het werk als thuis.

Hoewel de technologie zeer snel wijzigt, blijven de **onderliggende principes** grotendeels dezelfde. Het is dus niet enkel van belang dat alle jongeren de **bestaande technologie leren gebruiken**, ze moeten ook **begrijpen hoe het eigenlijk allemaal in elkaar zit**. Dan pas zullen ze in staat zijn om de **komende evolutie van de technologie** te kunnen volgen.

Leerlingen staan er niet vaak bij stil dat **STEM** (Science, Technology, Engineering, Mathematics) zo'n **grote maatschappelijke impact** heeft. Dat ze hun laptop kunnen inpluggen, met de trein naar school kunnen of hun afval opgehaald wordt, vinden ze doodnormaal. Wanneer we hen attent maken op de vele toepassingen van wetenschap en technologie is hun **interesse snel gewekt.** Zeker als ze zelf de handen uit de mouwen mogen steken.

Door kinderen zelf te laten **experimenteren met een robot** of ze te **laten programmeren,** gaan ze de gedigitaliseerde wereld rondom hen beter begrijpen.

2 Basisprincipes programmeren

Programmeren en computationeel denken zijn termen die we steeds meer tegenkomen in kranten en op het nieuws. Maar wat betekenen ze en waarom zijn ze zo belangrijk?

Computationeel denken is een manier van denken waarbij je een **probleem zo aanpakt dat een computer kan helpen om het op te lossen**.

Daarvoor dien je een **opeenvolging van ondubbelzinnige opdrachten** te bedenken die leiden tot het gewenste resultaat. Dit stappenplan noemen we een **algoritme**. Je kan het vergelijken met een recept.

Een belangrijke techniek in het opstellen van een algoritme is het **opdelen van complexe problemen in enkele eenvoudige deelproblemen**. Dit kom je ook tegen in het dagelijkse leven. Wanneer je een lekke fietsband hebt, weet je soms niet hoe eraan te beginnen. Het wordt gemakkelijker als je het als volgt bekijkt: eerst jouw fiets omdraaien, vervolgens het wiel losmaken, daarna de buitenband van het wiel halen waarna je de binnenband kan plakken of vervangen. Tenslotte zet je er de buitenband terug op, pomp je de band op en plaats je het wiel terug op de fiets.

Een andere veelgebruikte techniek in het opstellen van algoritmes, is het **herkennen van patronen**. Als je weet hoe je opzoekt wanneer de volgende bus komt, dan zal je dat vast ook kunnen voor een trein. In beide gevallen zoek je in tijdstabellen van een publiek vervoermiddel. Zoals blijkt uit dit voorbeeld, wordt het gemakkelijker om een nieuw probleem op te lossen wanneer je aandacht hebt voor patronen en je je kunt baseren op een gelijkaardig probleem dat je eerder hebt opgelost.

Tot slot worden veel **problemen gemakkelijker wanneer je ze abstract kan voorstellen**. Zo kun je gemakkelijk jouw weg vinden in de stad aan de hand van een kaart waarin de verbindingen tussen de belangrijkste toeristische attracties en de belangrijkste wegen zijn vermeld. Op diezelfde kaart zijn details die niet helpen bij het vinden van de weg zoals bv. de huizen of de bomen weggelaten.

Eens je algoritme klaar is, moet je het nog **vertalen in opdrachten die de computer begrijpt**. Dit noemen we **programmeren**. Programmeren gebeurt niet in mensentaal, maar in een speciaal voor dat doel ontworpen **programmeertaal.**

3 Computeren zonder computer

Computers zijn **niet bijzonder slim**. Ze zijn gewoon heel goed in het snel uitvoeren van opdrachten. Omdat de computer de opdrachten zou verstaan moeten we die opdrachten formuleren in een voor hem **begrijpbare taal**: de **programmeertaal**. Een programma is een opeenvolging van opdrachten die de computer **letterlijk zal uitvoeren**, zelfs als dit een gek resultaat oplevert.

Leerlingen moeten op de **juiste manier met een computer leren communiceren**. Ze moeten **vertrouwd worden met de 'taal' die de computer spreekt**.

Om die taal te leren begrijpen hoeven leerlingen niet aan een computer te zitten. Door **zelf de plaats in te nemen van een computer** ondervinden ze zelf hoe het in zijn werk gaat.

Bij de volgende opdrachten zullen sommige leerlingen de rol van programmeur op zich nemen en anderen spelen de computer zelf. Bij sommige opdrachten oefenen ze eerder hun **rekenvaardigheden**, bij andere hun **communicatievaardigheden.** De leerlingen lossen op een creatieve manier problemen op en oefenen hun denkvaardigheden.

In de **eerste activiteit** leren de leerlingen dat een computer **precieze instructies nodig** heeft om een taak uit te voeren.

In de **daaropvolgende activiteit** maken ze kennis met **hoe overdracht van gegevens over het internet werkt**. Tenslotte ervaren de leerlingen dat er zich **bij de overdracht van gegevens fouten kunnen voordoen** én hoe ze die kunnen **herstellen**.

Deze opdrachten zijn gebaseerd op de activiteiten uit het **handboek CS Unplugged**. Elke activiteit in dat boek is zodanig opgevat dat leerlingen het probleem onderzoeken en zelf oplossingen bedenken. Na iedere activiteit is het aangewezen om de gevonden antwoorden in groep te bespreken.

Voor al deze activiteiten heb je geen computer nodig.

Programmeer eens een mens

Computers kunnen niet interpreteren en voeren letterlijk iedere instructie uit die je ze geeft. De uitdaging van de programmeur bestaat er in problemen op te lossen door ze op te delen in kleine stappen die uitvoerbaar zijn door de computer en de instructies op de juiste manier aan de computer te geven.

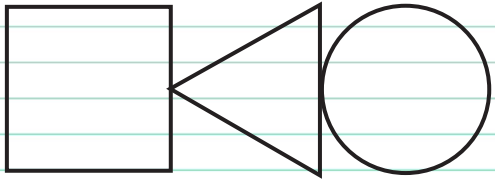
Doel

De leerlingen programmeren elkaar om een tekening te maken. Eén leerling is de **'programmeur'** en **geeft zorgvuldige instructies** aan de andere leerlingen om een bepaalde tekening te reproduceren. De andere leerlingen zijn **'de computer'**. Hoe **snel en nauwkeurig** wordt de tekening voltooid?

Vaardigheden

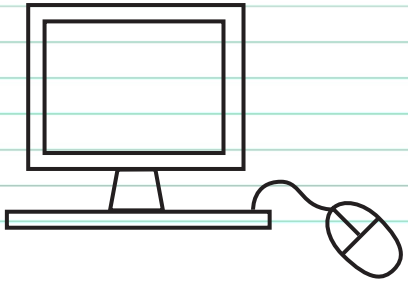
Leerlingen leren een relatief moeilijk probleem op te lossen door het op te splitsen in kleine stappen. Leerlingen moeten **heel precieze instructies geven én in de juiste volgorde**.

A



B

C



Benodigdheden

Voor deze opdracht heb je **geen computer nodig**.
Papier, potloden en enkele tekeningen zijn alles wat je nodig hebt.

Opdracht

Geef een leerling een eenvoudige tekening (bv. de tekening **(A)** op de bladzijde hiernaast), die leerling is **'de programmeur'**. De andere leerlingen mogen die tekening niet zien, zij zijn **'de computers'**. De 'programmeur' geeft instructies aan de andere leerlingen hoe ze moeten tekenen, door zo nauwkeurig mogelijk de tekening te beschrijven. De andere leerlingen mogen geen vragen stellen ter verduidelijking van de instructies.

Nadien

Bespreek het resultaat van de opdracht. Waarom zijn er zoveel verschillende resultaten? Wat lukte goed? Wat ging er fout? Waren de instructies vatbaar voor interpretatie? Gaf de 'programmeur' de instructies stap voor stap? Hoe kan het beter?

Herhaal de opdracht met een andere tekening (bv. de smileys **(B)**) en doe het dan nog een keer met een moeilijke tekening (bv. de computer **(C)**). Maak na elke keer tijd om de resultaten te bespreken.

Afhankelijk van het leeftijdsniveau kan je leerlingen aansporen om ook zo goed mogelijk rekening te houden met de werkelijke verhoudingen van de tekening. Geef het verband tussen deze opdracht en het programmeren

Wil je er wat verder in gaan?

Geef elke leerling een tekening of laat hen zelf een tekening maken. Vraag om **gedetailleerde instructies** te bedenken. Laat ze deze instructies **uitschrijven** op papier. **Test** de geschreven instructies door deze aan een ander kind te geven en te laten uitvoeren.

Een menselijk computernetwerk

Je hebt zelf ook wel eens een foto verstuurd naar een vriend via het internet of via je smartphone. Computers, smartphones en andere toestellen zijn verbonden met het internet via een modem en kunnen zo **met elkaar communiceren**. Opdat ze elkaar zouden kunnen begrijpen dient die communicatie volgens bepaalde afspraken te verlopen. We noemen deze afspraken een **protocol**.

Doel

In deze activiteit zullen de leerlingen **zelf een protocol bedenken** om een tekening te versturen van de ene plaats naar een andere.

Vaardigheden

Leerlingen leren hoe een digitale afbeelding opgebouwd is uit pixels die je door getallen kan voorstellen. Leerlingen **leren doelgericht communiceren** over een probleem. Leerlingen kunnen **oplossingen** voor probleemstellingen **bedenken en toepassen**. Leerlingen leren **problemen in team** op te lossen. Leerlingen werken met natuurlijke getallen.

Benodigdheden

Elke groep van **5 tot 7 kinderen** heeft het volgende nodig:

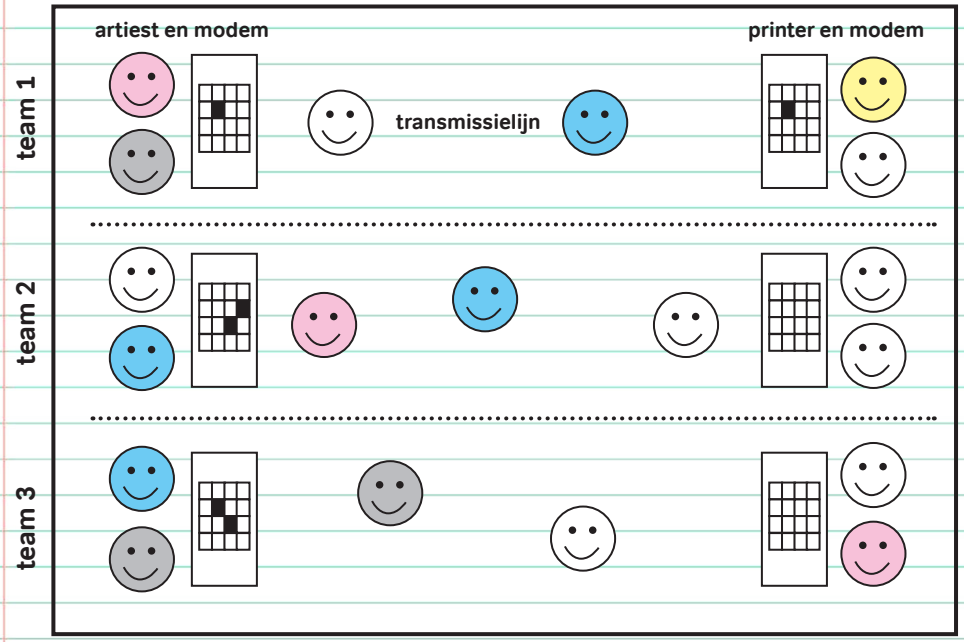
- Twee blanco rasters van 20 x 16 vakjes.
- Je kan ze vinden op: **http://www.dwengo.org/WeGoSTEM**
- Twee potloden.

Opdracht

Verdeel de leerlingen **in groepjes van 5 à 7**. Geef iedere groep twee blanco rasters en een potlood. Per groep is er een **rolverdeling**:

- Eén artiest die een tekening maakt op het blanco raster.
- Eén printer die later in het spel een tekening die via het 'netwerk' verstuurd werd, reconstrueert.
- Twee assistenten doen dienst als modem en helpen ofwel de artiest, ofwel de printer.
- De andere kinderen zullen boodschappen in het netwerk overbrengen.

Zet de kinderen per groep samen en laat de artiest een tekening maken. De tekening wordt gemaakt door de gewenste hokjes op het blanco raster in te kleuren. Zorg ervoor dat de andere groepen de tekening niet zien.


artiest en modem
printer en modem
team 1
transmissielijn
team 2
team 3

Wanneer de tekening klaar is, vraag je aan iedere groep om een manier te bedenken, **een protocol**, waarmee ze een getekende figuur kunnen voorstellen door middel van natuurlijke getallen en hoe ze deze getallen één per één zullen verzenden via het netwerk van artiest naar printer. Het verzenden gebeurt door kinderen die van de ene kant naar de andere kant lopen. **De bedoeling is dat de printer de tekening van de artiest kan printen zonder dat de printer de tekening gezien heeft.**

Het is belangrijk om goed te verduidelijken wat een natuurlijk getal is en dat er slechts één getal per keer mag doorgegeven worden. Met andere woorden, één getal komt overeen met één keer lopen, én de individuele cijfers van deze getallen kunnen niet afzonderlijk gespeld worden!

Haal de tekeningen nu op en stel de groepen als volgt op:

Zet de artiesten met hun modem aan één kant van de ruimte. Zet de printers samen met hun modem aan de andere kant van de ruimte. De andere kinderen zullen heen en weer lopen tussen artiest en printer.

De indeling van de ruimte kan je bekijken op de vorige pagina.

© Francis wyffels: Dwengo workshop in de Wittering (NL) (CC-licentie)

Verspreid nu de tekeningen willekeurig over de artiesten, maar geen enkele groep krijgt zijn eigen tekening.

Geef vervolgens het startsignaal waarop het versturen kan beginnen!

Wanneer een groep klaar is geven ze beide tekeningen af (het origineel en het geprinte).

Nadien

Bespreek de resultaten klassikaal.

- Laat achtereenvolgens per groep iemand naar voren komen.
- Komen de tekeningen overeen?
- Laat een leerling het gekozen protocol uitleggen.
- Enkele mogelijke vragen die je hierbij kan stellen :
  - Hoe zou je jouw protocol veranderen als de grootte van het raster onbekend is?
  - Hoeveel nummers zou je verzenden als je een wit beeld zou versturen of een volledig zwart?
  - Hoe zou je het aantal verzonden nummers kunnen verminderen?
  - Wat is het effect van fouten in de verzonden nummers op het uiteindelijke beeld?

Goocheltruc

In de vorige activiteit werkten de leerlingen een manier uit om **gegevens via een netwerk te verzenden**. In ieder netwerk kunnen er zich **fouten** voordoen waardoor de gegevens gewijzigd worden. Er bestaan verschillende manieren waarop je die fouten kunt **herstellen**.

Doel

In deze activiteit doe je een magische truc met de leerlingen om hen laten zien hoe je erachter kan komen dat gegevens fout zijn en hoe je die **fouten** terug kunt **herstellen**.

Vaardigheden

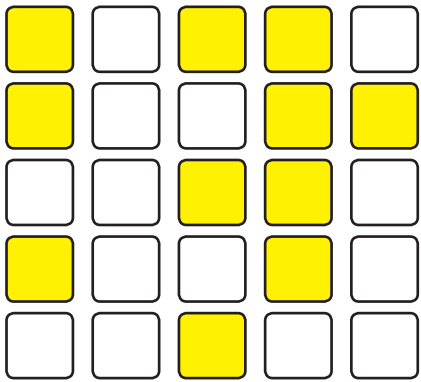
Leerlingen kunnen werken met even en oneven getallen. Leerlingen leren hoe via het toevoegen van extra gegevens aan foutherkenning en -herstel kan worden gedaan.

Benodigdheden

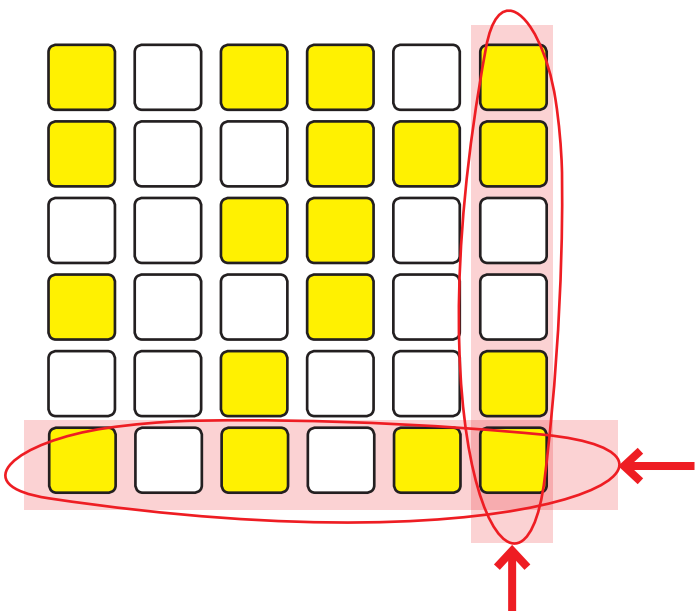
Een set van **36 dezelfde kaartjes**, met **voor- en achterkant verschillend van kleur.**
Het is nog handiger als je over een magneetbord beschikt en een set van minstens **36 dezelfde tweezijdige magneten**, met **voor- en achterkant verschillend van kleur**.

Opdracht

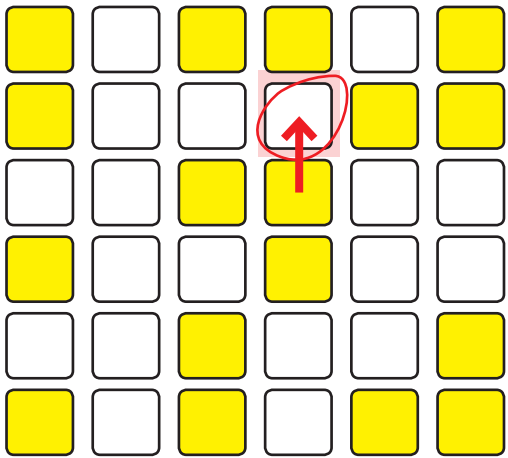
1. Vraag een leerling om de kaarten in een **raster van 5 bij 5** te plaatsen met een **willekeurige kant** boven. Zoals in de volgende afbeelding:



2. **Voeg vervolgens nog een rij en kolom toe** “om het iets moeilijker te maken”. Deze kaarten zijn de sleutel tot succes. Je moet de extra kaarten zo plaatsen dat je **in iedere rij en in iedere kolom een even aantal gekleurde kaarten** hebt.



3. Vraag de leerling om **één kaart om te draaien** terwijl jij niet kijkt. De rij en kolom die de omgedraaide kaart bevatten, zullen nu een oneven aantal gekleurde kaarten hebben en de omgedraaide kaart onthullen. Kunnen de leerlingen raden hoe de truc in elkaar steekt? Doe de truc meerdere keren totdat de meeste kinderen spontaan het antwoord vinden.



4. Moedig de kinderen aan om de truc aan hun familie of vrienden te tonen.

Nadien

Bespreek nu de truc met de kinderen en leg het **verband** tussen de goocheltruc en het versturen van gegevens over het internet.

Bij het versturen van gegevens kunnen die soms **ongewenst wijzigen**. Zulke fouten wil men opsporen én terug verbeteren. De kaarten stellen de **bits voor die 0 of 1 kunnen zijn**. Door er kaarten bij te leggen, verkrijgt men in elke rij en kolom steeds een even aantal gekleurde kaarten.

De extra kaarten (of bits) noemt men in de informatica ook wel **controle-bits**. Hoeveel fouten kun je opsporen en verbeteren met behulp van de extra rij en kolom?

Honger naar meer?

Materiaal bij de opdrachten

http://www.dwengo.org/WeGoSTEM

Nog enkele mogelijke opdrachten

Binaire getallen: http://www.csunplugged.nl/introductie/deel-1/01-binaire-getallen/
Programmeren op ruitjespapier: https://studio.code.org/s/coursed/stage/1/puzzle/1
Sorteernetwerken: http://www.csunplugged.nl/08-sorteernetwerken/

Links

www.csunplugged.org
https://code.org/curriculum/unplugged
www.dwengo.org
www.levendprogrammeren.nl
www.csunplugged.nl
www.program-uurtje.org
www.cs4fn.org/

4 Grafisch programmeren met gratis software

Programmeren is het schrijven van instructies in een bepaalde **volgorde**. Je schrijft deze instructies in een taal die de computer begrijpt: een **computertaal**. De computer zal deze instructies uitvoeren. Doet de computer wat je wilt dat hij doet? Dan heb je een **programma** geschreven.

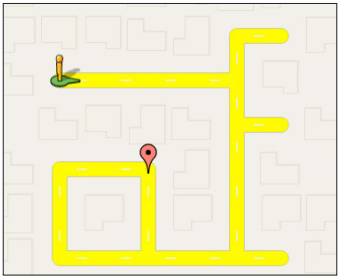
Programmeren te moeilijk? Niet met de gratis online tools Google Blockly en Scratch! Google Blockly en Scratch zijn **grafische programmeertalen.** Het enige dat je moet doen, is enkele **blokken verslepen en ze op de juiste manier combineren**, net zoals een puzzel. Voor de meer ervaren programmeurs bestaan er **tekstuele programmeertalen** zoals **Java, C++ en Python**. Deze tekstuele programmeertalen komen echter niet aan bod in dit boekje. In ieder geval moet je om een grafisch programma te maken op dezelfde manier denken als om een programma in een tekstuele computertaal te schrijven. Je moet dezelfde **structuren gebruiken en dezelfde instructies**. Het ziet er alleen een beetje anders uit. Net daarom gebruiken we een grafische programmeertaal als basis in dit boekje.

In de volgende opdrachten moeten de leerlingen **problemen oplossen door te programmeren met Blockly of Scratch.** De opdrachten met Blockly zijn zeer toegankelijk voor de leerlingen en ze bevatten verschillende niveaus die steeds moeilijker worden. De opdrachten met Scratch zijn iets uitdagender. De werkwijze is volledig uitgeschreven voor de leerkrachten.

Een doolhof met Blockly

Deze activiteit begint met eenvoudige opdrachten die de leerlingen het **belang van de volgorde van de instructies** doen inzien. In de hogere niveaus zullen de leerlingen merken dat dezelfde instructies **dikwijls herhaald** worden.

Om te vermijden dat ze deze steeds opnieuw moeten schrijven, zullen ze een **herhalingsstructuur** leren gebruiken. Tenslotte wordt het **gebruik van een keuzestructuur** ingeleid, waardoor een bepaalde actie enkel uitgevoerd wordt onder specifieke voorwaarden.



Doel

Een programma schrijven met Google Blockly zodat pegman (of de astronaut of pandabeer) het **rode icoontje bereikt**.

Er zijn 10 niveaus. Elk niveau is een beetje moeilijker dan het vorige.

Vaardigheden

De leerlingen leren een **probleem opdelen in verschillende instructies**. De leerlingen zien in dat de **volgorde van de instructies heel belangrijk** is. De leerlingen leren **patronen herkennen** en maken gebruik van **herhalingsstructuren om programma's efficiënter te schrijven**.

De leerlingen maken kennis met de **keuzestructuur** in programmeren.

Benodigdheden

Per groepje van 2 leerlingen is een **computer met een internetverbinding** en -browser nodig.

Opstarten van Blockly Games-Doolhof

1. Surf naar **https://blockly-games.appspot.com.**
2. Kies rechtsboven voor Nederlands, Vlaams!
3. Kies linksonder om de oefeningen te laten herbeginnen.
   Zo verdwijnen de oplossingen van de vorige klas.
4. Kies voor het **Doolhof**: 

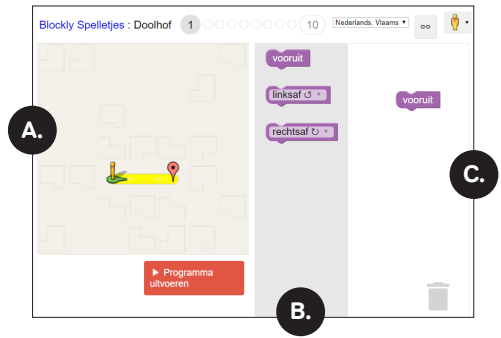
5. Duid rechtsboven het **gewenste figuurtje** aan: pegman, de astronaut of de pandabeer.

Je start met dit scherm:

A. Links staat het **doolhof**.

B. In het midden vind je de **beschikbare instructies**.

C. Rechts staat **jouw programma**.



Opdracht

De leerlingen maken een programma door de blokken in de juiste volgorde aan elkaar te klikken. Door op de knop **Programma uitvoeren** te klikken zal de computer het programma uitvoeren door de blokjes één voor één te overlopen. In het begin heb je niet veel blokjes nodig om pegman (of de astronaut of pandabeer) het rode icoontje te laten bereiken, maar het wordt snel moeilijker.

Probeer de leerlingen te laten nadenken over **hoe ze hun programma opbouwen**. Dit kan door ze **te laten voorspellen hoe** hun popje zal bewegen alvorens op Programma uitvoeren te klikken. Zo vermijd je dat ze willekeurig wat blokjes met elkaar gaan combineren.

Nadien

Vergeet niet om na het einde van een programmeer-sessie ook **terug te blikken**.

- Wat vonden de leerlingen moeilijk/gemakkelijk?
- Welke soorten blokjes konden ze herkennen? Merk op dat elke kleur zijn betekenis heeft. Groene blokjes dienen om te herhalen, blauwe om een voorwaarde te stellen en de paarse om actie te ondernemen.
- Zijn er verschillende oplossingen voor hetzelfde doolhof mogelijk?

Minecraft-avontuur met Blockly

In de vorige opdracht hebben de leerlingen geleerd dat de volgorde van de instructies zeer belangrijk is. Daarnaast leerden ze het gebruik en nut van een herhalingsstructuur. In deze opdracht wordt hier verder op geoefend. De focus van de hogere niveaus van deze opdracht is het gebruik van de keuzestructuur.

Doel

In deze oefening zullen de leerlingen een **eigen Minecraft-avontuur programmeren.** Minecraft-avonturier is één van de vele **Hour of Code-opdrachten**. Ze zijn speciaal ontworpen om op **één lesuur** leerlingen kennis te laten maken met de **beginselen van het programmeren.** Een leerling die deze activiteit tot een goed einde brengt, ontvangt na afloop een **certificaat.**

Vaardigheden

De leerlingen leren een **probleem opdelen** in verschillende instructies. De leerlingen zien in dat de **volgorde van de instructies** heel belangrijk is. Ze maken kennis met de **herhalings- en keuzestructuren** binnen programmeren.

Benodigdheden

Per groepje van 2 leerlingen is een **computer met een internetverbinding** en -browser nodig.

Opstarten van het Minecraft-avontuur

1. Surf naar **https://code.org/minecraft**.
2. **Kies Minecraft-avonturier** om het avontuur te starten.
3. Je kan de motiverende filmpjes (met Nederlandstalige ondertitels) bekijken of ze overslaan.

Je start met dit scherm:

A. Links staat **Minecraft**.

B. In het midden vind je de **beschikbare instructies**.

C. Rechts staat **het programma**.

Opdracht

In het **Minecraft-avonturierspel** zal de leerling door te programmeren Alex of Steve op vakantie laten gaan. Er zijn **14 niveaus** die telkens wat moeilijker worden.

Door op de knop **Start** te klikken, zal de computer het **programma uitvoeren** door de **blokjes één voor één te overlopen**.

Het aantal blokjes dat je gebruikt wordt geteld. Indien je meer blokjes gebruikt dan strikt noodzakelijk, krijg je de suggestie dat het ook korter kan. Zo worden leerlingen **gemotiveerd om efficiënter te programmeren** en zet je ze aan tot nadenken alvorens te beginnen met programmeren.

Nadien

Vergeet niet om na het einde van een programmeersessie ook **terug te blikken.**

- Wat vonden de leerlingen moeilijk/gemakkelijk?
- Welke soorten blokjes konden ze herkennen?
- Zijn er verschillende oplossingen voor dezelfde opdracht mogelijk?
- Hoe zochten ze efficiënt een oplossing voor eventuele fouten in hun programma's?

Frozen, ontdek met Anna en Elsa de magie van het ijs met Blockly

In de vorige activiteiten leerden de leerlingen herhalings- en keuzestructuren gebruiken. Nu leren we moeilijkere programma's maken door **herhalingsstructuren** te gebruiken binnen andere herhalingsstructuren. We noemen dit **geneste programmeerstructuren.**

Verder ontdekken de leerlingen het **gebruik van functies om nieuwe opdrachten aan de programmeertaal toe te voegen**. Dit is handig wanneer deze opdrachten een oplossing zijn voor een **duidelijk afgebakend deelprobleem** van het probleem dat we willen oplossen. Bovendien kunnen we wanneer het deelprobleem meermaals voorkomt de functie hergebruiken. Met andere woorden, de functie laat ons toe om die **volgorde van opdrachten slechts één maal neer te schrijven** en ze daarna steeds te **hergebruiken**.

Doel

Ontdek met Anna en Elsa de magie van het ijs is één van de vele **Hour of Code-opdrachten**. Ze zijn speciaal ontworpen om **op één lesuur** leerlingen kennis te laten maken met de beginselen van het programmeren. Een leerling die deze activiteit tot een goed einde brengt, ontvangt na afloop een **certificaat**.

Vaardigheden

De leerlingen gebruiken **herhalingsstructuren** en maken kennis met **geneste structuren en functies**.

Leerlingen werken met hoeken en vlakke figuren. De leerlingen tekenen meetkundige figuren door te programmeren.

Benodigdheden

Per groepje van 2 leerlingen is een **computer met een internetverbinding** en -browser nodig.

Opstarten van de Frozen-activiteit

Surf naar **https://code.org/frozen**

Je kunt de motiverende filmpjes (met Nederlandstalige ondertitels) bekijken of ze overslaan.

Je start met dit scherm:

A. Links staat **winterlandschap**.

B. In het midden vind je de **beschikbare instructies**.

C. Rechts staat **het programma**.

Opdracht

In de Frozen-activiteit zal de leerling door **grafisch te programmeren** Anna en Elsa helpen om sneeuwvlokken en patronen te tekenen. Er zijn **20 niveaus** die steeds wat moeilijker zijn.

Door op de knop **Start** te klikken zal de computer het **programma uitvoeren door de blokjes één voor één te overlopen**.

Het aantal blokjes dat je gebruikt wordt geteld. Indien je meer blokjes gebruikt dan strikt noodzakelijk, krijg je een hint dat het ook **korter kan**. Zo worden leerlingen gemotiveerd om na te denken voor ze iets programmeren en om **efficiëntere programma's** te maken.

Nadien

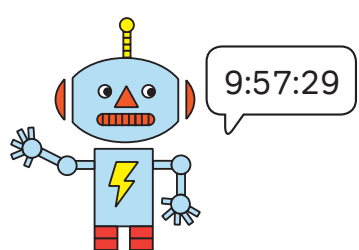
Vergeet niet om na het einde van een programmeersessie ook **terug te blikken**:

- Wat vonden de leerlingen moeilijk/gemakkelijk?
- Welke soorten blokjes konden ze herkennen?
- Zijn er verschillende oplossingen voor dezelfde opdracht mogelijk?
- Voor welke deelproblemen vonden ze nuttig om een functie aan te maken?

Digitale klok met Scratch

In deze activiteit werken de leerlingen met een **andere visuele programmeertaal** genaamd **Scratch**. Scratch is bijzonder **geschikt om animaties en eenvoudige spelletjes** mee te maken.

Deze activiteit is voor kinderen die **reeds ervaring** hebben **met programmeren**. Vooreerst is de programmeeromgeving van Scratch veel rijker dan de Blockly-omgevingen die we gebruikten voor de vorige activiteiten. Daarnaast worden technieken gebruikt om **tekst samen te voegen en introduceren we variabelen**.



Doel

In deze opdracht maak je een **digitale klok** met Scratch.

Vaardigheden

De leerlingen leren een **probleem opdelen in verschillende gemakkelijk op te lossen deelproblemen**.

De leerlingen zoeken met een zoekmachine op het internet en kunnen een figuur toevoegen binnen de gebruikte software. De leerlingen leren werken met tekst in een programma. Ze leren hoe ze tekst aan elkaar kunnen kleven om woorden en zinnen mee te maken.

Ze werken met een **herhalingsstructuur**.

In de uitbreidingsopdracht leren ze werken met variabelen, gebruiken ze **keuzestructuren** en de **wacht-functie.**

Benodigdheden

Per groepje van 2 leerlingen is een **computer met een internetverbinding** en -browser nodig.

Opstarten

Surf naar **https://scratch.mit.edu/projects/editor/?tip_bar=home.**

Opdracht

1. Verwijder  door rechts te klikken op 

en kies 

.

2. Je voegt een **nieuwe sprite** toe,
bv. een figuur van een robotje dat je vindt op het internet.
(Gebruik 'robot" om te zoeken op 'Afbeeldingen' met een zoekmachine bv. Google). **Sla de gekozen afbeelding op**.
Let op! Vergeet niet waar je haar bewaard hebt!

3. Ga naar 

en kies voor  .
Als je dat wilt, dan kun je de sprite wat groter maken met  of net kleiner met  .

4. Elk script begint met 

.
Je voegt de instructies **één voor één** toe in de **juiste volgorde** door ze hieraan **vast te klikken zoals een puzzel**.

Het robotje moet vertellen **hoe laat het is**. Hij zal zowel het **uur**, de **minuten** als de **seconden** geven. De robot iets laten zeggen, doe je met de functie 

De tijd verandert echter voortdurend.

Gelukkig kan je in **Scratch** met 

de werkelijke tijd opvragen, met het pijltje kan je 'minuut' wijzigen in 'uur' of in 'seconde'.

Om de werking hiervan te achterhalen probeer je best eens volgend programma uit:

. Klik op  om te starten.

Je wilt natuurlijk dat het robotje **ook het uur en de minuten** zegt. Hiervoor gebruik je de **samenvoeg-operator**: 

Je gebruikt die zodat uren, minuten en seconden **mooi naast elkaar verschijnen**:

Test uit.

5. Het resultaat is **nog niet optimaal**. Het ziet er nog niet uit als de weergave op een digitale klok. Tussen de uren, minuten en seconden moet er **telkens een dubbelpunt**! Dit kan je doen door het uur samen te voegen met een ":".

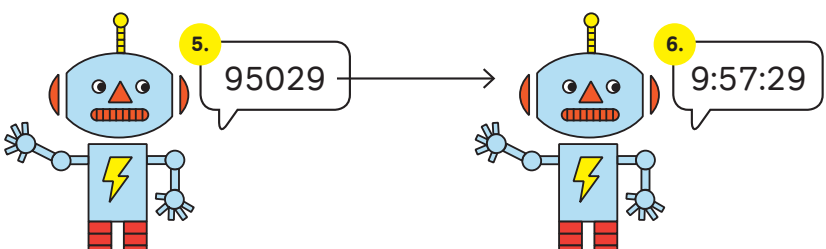
Vergeet dit ook niet te doen met de minuten en **test jouw programma** uit.

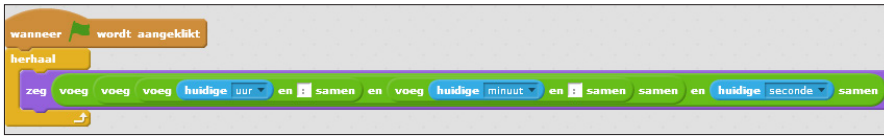
6. **Dat ziet er al veel beter uit.**

Het uur op een digitale klok verandert voortdurend. Bij jou nog niet.
Hoe kun je dit bekomen? Het programma zou voortdurend moeten gaan kijken wat de werkelijke tijd is.

Je gebruikt de **herhalingsstructuur:** 

.





Je digitale klok werkt!

7. In principe heb je nu al een goed werkende digitale klok, maar je kunt het **resultaat nog verfijnen**. Er is namelijk nog iets mis met de vormgeving. Bij een echte digitale klok wordt de tijd weergegeven als 09:57:30 en als 12:07:02. **Die nullen staan er bij jou nog niet.** Om die nullen toe te voegen zal je moeten gebruik maken van de **keuzestructuur én variabelen**. **Dit doen we in de uitbreiding**.

Uitbreiding

Een variabele is een plaats in het geheugen waaraan je een naam geeft en waaraan je een **waarde toekent**. Deze waarde kun je dan verder in het programma gebruiken door de variabele op te roepen. Je kunt de **waarde van de variabele zelfs wijzigen** in de loop van het programma. Vergelijk het met een lade met een label in een opbergkast waarvan je de inhoud regelmatig wijzigt. De lade is dan de variabele, de tekst op het label is dan

de naam van de variabele, de inhoud van de lade is de waarde van de variabele en de opbergkast is het geheugen. **In een programma slaan we echter geen voorwerpen op, maar wel getallen of tekst**. Merk op dat net zoals een opbergkast meerdere lades kan bevatten, je in jouw programma **meerdere variabelen** kunt aanmaken.

1. Je maakt **drie variabelen** aan:

- één voor het uur,
- één voor de minuten,
- één voor de seconden.

Eigenlijk volg je daarvoor **drie keer dezelfde werkwijze**. Ga daarvoor naar 

en kies 

, 

verschijnt.

Je maakt een **variabele aan met de naam uur**: 

.

Je doet hetzelfde voor de **variabele** met naam **minuut** en de **variabele** met naam **seconde**. Resultaat: 

.

2. Je moet de computer ook vertellen **welke waarde die variabelen moeten hebben**.

Aan de variabele uur geef je de waarde **'huidige uur'**. Je doet dat door in 

op de plaats van de 0 

in te vullen.

Dus: 

3. Je doet **hetzelfde voor de variabelen minuut en seconde**.

Voor de meeste waarden zal dat voldoende zijn, maar soms moet er dus ook een nul voor. Bv. 09:57:30, hier worden de waarde van de variabele minuut en de waarde van de variabele seconde weergegeven zoals ze zijn, maar de waarde van de variabele uur wordt voorafgegaan door een 0. **Die 0 moet dus toegevoegd worden.**

Wanneer moeten die **nullen erbij**? Als het aantal uur (of minuten of seconden) **kleiner is dan 10**. Je zal hiervoor gebruik maken van een **keuzestructuur**:

Bv. voor het uur:

ALS het aantal uur kleiner is dan 10
DAN is uur = een 0 met erna het huidige uur
ANDERS is uur = het huidige uur

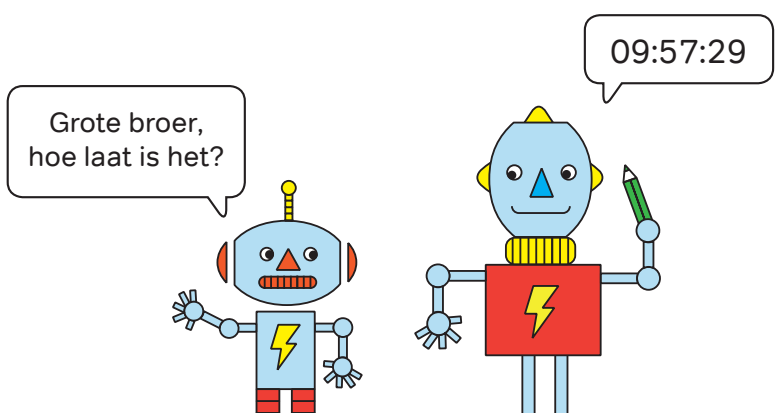
4. Doe vervolgens **hetzelfde voor de minuten en de seconden**.

5. Voeg tot slot **alles samen** en **test je programma**.

Wil je er wat verder in gaan?

Kun je dit programmeren? De kleine robot vraagt aan de grote robot hoe laat het is.
Na drie seconden verschijnt de digitale klok 'uit de mond' van de grote robot.

Om ervoor te zorgen dat de digitale klok **niet onmiddellijk verschijnt** bij de start van het programma, gebruik je de **wacht-functie**:



Analoge klok met Scratch

Doel

Een klok met **bewegende wijzers** maken met Scratch.

Vaardigheden

De leerlingen leren een **probleem opdelen in verschillende instructies**. Ze maken kennis met de **herhalingsstructuur** binnen programmeren. De leerlingen kunnen de **snelheid bepalen** waarmee de wijzers draaien. Ook leren de leerlingen een aantal wiskundige concepten uit de meetkunde zoals **coördinaten en rotaties**.

Benodigdheden

Per groepje van 2 leerlingen is een **computer met een internetverbinding** en -browser nodig.

Opstarten

Surf naar
https://scratch.mit.edu/projects/editor/?tip_bar=home.

Opdracht

1. Verwijder  door rechts te klikken op  en kies

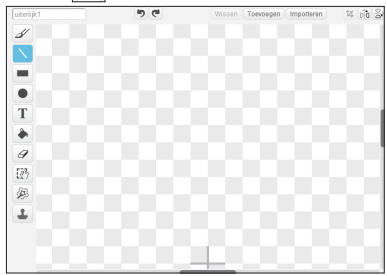
2. Je maakt **3 nieuwe sprites** aan:

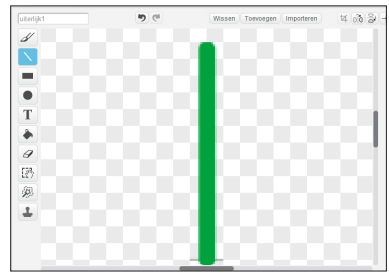
- één voor de secondewijzer,
- één voor de minutenwijzer,
- één voor de urenwijzer.

3. Ga naar 

en kies voor  .

4. Klik op  . Zoom het tekenscherm een beetje in:





5. Teken een **recht lijnstuk**, dit gaat het gemakkelijkst als je de shift-toets ondertussen indrukt.

Het is belangrijk dat je je **lijnstuk laat starten op het midden van het kruisje**:  . **Rond dit punt** zal de wijzer straks **draaien**. Goed, je **minutenwijzer is klaar**.

6. Op dezelfde manier teken je de **urenwijzer** en vervolgens je **secondenwijzer**. Je secondewijzer teken je het langst en het dunst.

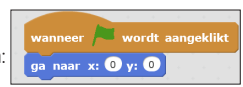
Let er in ieder geval op dat al je **wijzers mooi recht** staan.
Tip: experimenteer later met leukere en meer creatieve vormen als wijzer.

7. Voor elke wijzer zal je nu een script maken. Begin met de secondenwijzer.
Elk script begint met 
.

Hier klik je de **instructies één voor één aan in de juiste volgorde**.

8. Waar wil je dat je klok staat op je scherm? Juist, in het **midden**! Het midden van het scherm is in Scratch de plaats met **coördinaat x = 0 en y = 0**.

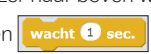
Gebruik de functie 

en vul voor **x** en **y** 0 in: 

9. Kijk wat er gebeurt als je nu op  klikt!

10. Is je secondewijzer naar het midden versprongen?


Zo bepaal je dat je wijzer naar boven wijst.
Nu zal je de functies 

en 

gebruiken.

11. Hoeveel graden moet een secondewijzer draaien gedurende 1 seconde? **Op 60 seconden moet hij de klok rondgaan**. Dat komt overeen met 360° draaien. Dus op 1 seconde moet de **secondewijzer 6° draaien (A)**. Je moet dit blijven herhalen, want je **klok mag niet stilvallen**. Gebruik hiervoor de **herhalingsstructuur (B)**. **Testen** maar! Dat ziet er goed uit **(C)**!

A.


B.


C.


12. **Doe nu hetzelfde voor de uren- en minutenwijzer**. Pas op! Die draaien niet zo snel!

De **minutenwijzer** draait **elke minuut 6°**, dat gaat geleidelijk aan (bv. per 20 seconden). Zie afbeelding **(D)**.

Als de **urenwijzer** de klok rond is gedraaid, dan zijn er 12 uur voorbij. Dus één **uur komt overeen met 30°**, nl. 360° gedeeld door 12. Je kan berekenen hoeveel seconden de urenwijzer doet over 1°. Zie afbeelding **(E)**.

D.

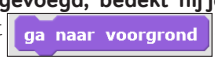
E.

13. Je zorgt ervoor dat ook deze wijzers naar boven wijzen:

Je hebt zo een klok gemaakt die **start op het tijdstip 0 uur, 0 minuten en 0 seconden**. **Je klok is klaar!**

Misschien wil je je klok wat opsmukken met een wijzerplaat.

1. Maak zelf een wijzerplaat, zoek er één op het internet (gebruik 'wijzerplaat zonder wijzers' om te zoeken in een zoekmachine, bv. Google) of download de wijzerplaat op **http://www.dwengo.org/WeGoSTEM.** De wijzerplaat is je **vierde sprite**.

2. Zorg ervoor dat ook hij mooi in het **midden** staat. Omdat je de wijzerplaat **later hebt toegevoegd, 'bedekt' hij je wijzers**. Breng de wijzers alle drie naar de voorgrond met 

.

3. De **grootte van je wijzerplaat** kun je ook aanpassen met 

.

4. Het is natuurlijk veel interessanter als de klok het **werkelijke uur weergeeft**. Daartoe moeten de wijzers van de klok bij de start van het programma wijzen naar het werkelijke uur van dat moment. Je wijzers wijzen bij de start naar boven, vanuit die positie laat je ze onmiddellijk verder draaien tot ze het werkelijke uur aanwijzen.

5. Bij de beschikbare blokken van Scratch vind je 

.
Met deze functie kun je de **werkelijke tijd** opvragen.

6. De **operatoren** 

zal je ook nodig hebben.

Je moet uitpluizen **over welke hoek de wijzers reeds gedraaid zijn**. Eerst de secondewijzer. Je weet dat de secondewijzer per seconde 6° draait, dus (F). En dan de minutenwijzer. Je weet dat de minutenwijzer per minuut 6° draait, dus (G).

F. 

G. 

7. Voor de **urenwijzer is het een beetje moeilijker**. De urenwijzer **schuift langzaam op**. Omdat de wijzerplaat verdeeld is in 12, deel je 360° door 12: de urenwijzer draait elk uur 30°. Maar hij doet dat niet ineens, in de loop van het uur draait de wijzer ook al, bv. na 30 minuten is hij al 15° gedraaid. Hoeveel graden is hij gedraaid na 2 minuten? En na 1 minuut?

Je bekomt: (H) voor de secondewijzer, (I) voor de minutenwijzer en tenslotte (J) voor de urenwijzer.

Merk op dat je dit programma korter kunt maken als je, net zoals bij de digitale klok, binnen de herhalingsstructuur de functie huidige minuut gebruikt om telkens de **werkelijke tijd op te vragen** en de wijzers rechtstreeks in te stellen. Maar het is veel **leerzamer om zelf de nodige berekeningen** te maken zoals we hier hebben voorgedaan.

Creatief met Scratch

Heb je de smaak te pakken? Na het maken van jouw eigen klok heb je voldoende kennis om leuke animaties te programmeren. **Maak een eigen animatie** door gebruik te maken van de blokken op deze pagina. Je mag ze één, twee of meerdere keren gebruiken. Maar je moet ze wel **allemaal minstens 1 keer gebruiken**.

Werkwijze

- Test ideeën door met ieder blok te **experimenteren**.
- **Combineer** de blokken op verschillende manieren.
- Experimenteer met verschillende achtergronden en andere of meerdere sprites.
- Werk rond een **bepaald thema of verhaal**: de natuur, de stad, ...
- Daag jezelf uit.

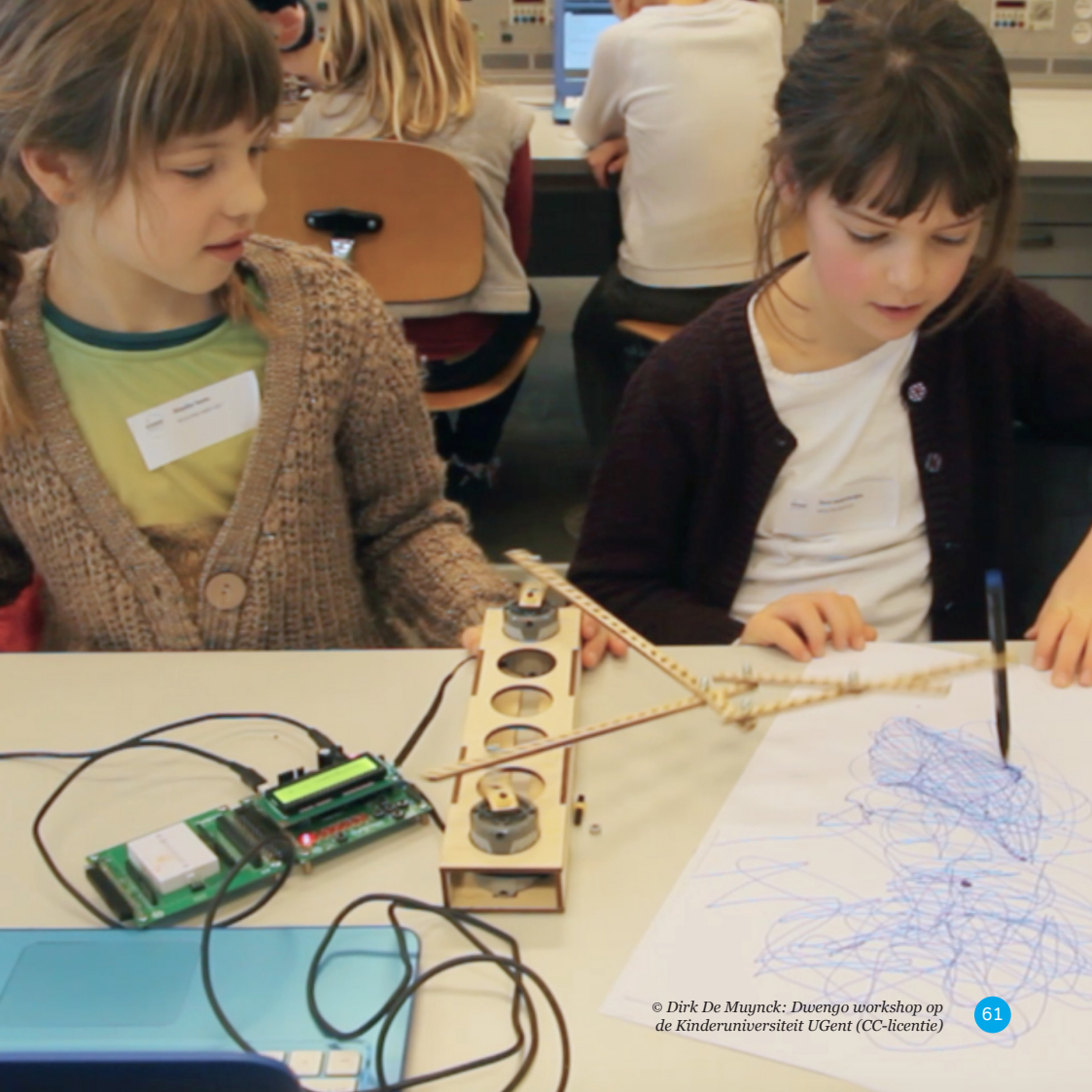
Honger naar meer?

Nog enkele mogelijke alternatieven

Code.org	
Star Wars (er is ook een offline versie)	https://code.org/starwars
Angry Birds	https://studio.code.org/hoc/1
De artiest	https://studio.code.org/s/20-hour/stage/7/puzzle/1
Lessenreeks	https://studio.code.org/s/20-hour/

Links

Google Blockly	https://blockly-games.appspot.com/
	http://www.dwengo.org/teach/blockly
Scratch	https://scratch.mit.edu/ http://scratchweb.nl/lesmateriaal



© Dirk De Muynck: Dwengo workshop op de Kinderuniversiteit UGent (CC-licentie)

5 WeGoSTEM uitleenkit

Wil je met je leerlingen de tekenrobot-activiteit van WeGoSTEM doen? Je kunt bij Dwengo het nodige materiaal ontlenen. In het pakket dat je zal ontvangen zit een handleiding met extra uitleg voor de leerkracht.

Surf daarvoor snel naar **http://www.dwengo.org/WeGoSTEM**

De leerlingen worden verdeeld in groepjes van twee. Elk groepje moet over **voldoende plaats beschikken:** plaats voor een **computer**, de **tekenrobot en een blad papier in A3-formaat**. Elke groepje moet ook beschikken over een stopcontact voor de adapter van de robot.

Elk groepje **bouwt een eigen tekenrobot**. Nadien gaan ze de robot zo programmeren dat hij kan tekenen. Eens ze daarin geslaagd zijn, krijgen ze nog voldoende tijd om te experimenteren met o.a. verschillende snelheden van de motoren.

De activiteit kan gedaan worden **in anderhalf uur**, maar als je ook de moeilijkere oefeningen wil maken dan moet je meerdere lessen organiseren. Zo kun je naast de tekenende robot **ook een rijdende robot maken** die zijn weg moeten vinden doorheen een doolhof met behulp van zijn sensoren.

6 Ga ervoor

Je merkt het. Als je met je leerlingen wil werken rond computationeel denken, dan zijn er heel wat mogelijkheden.

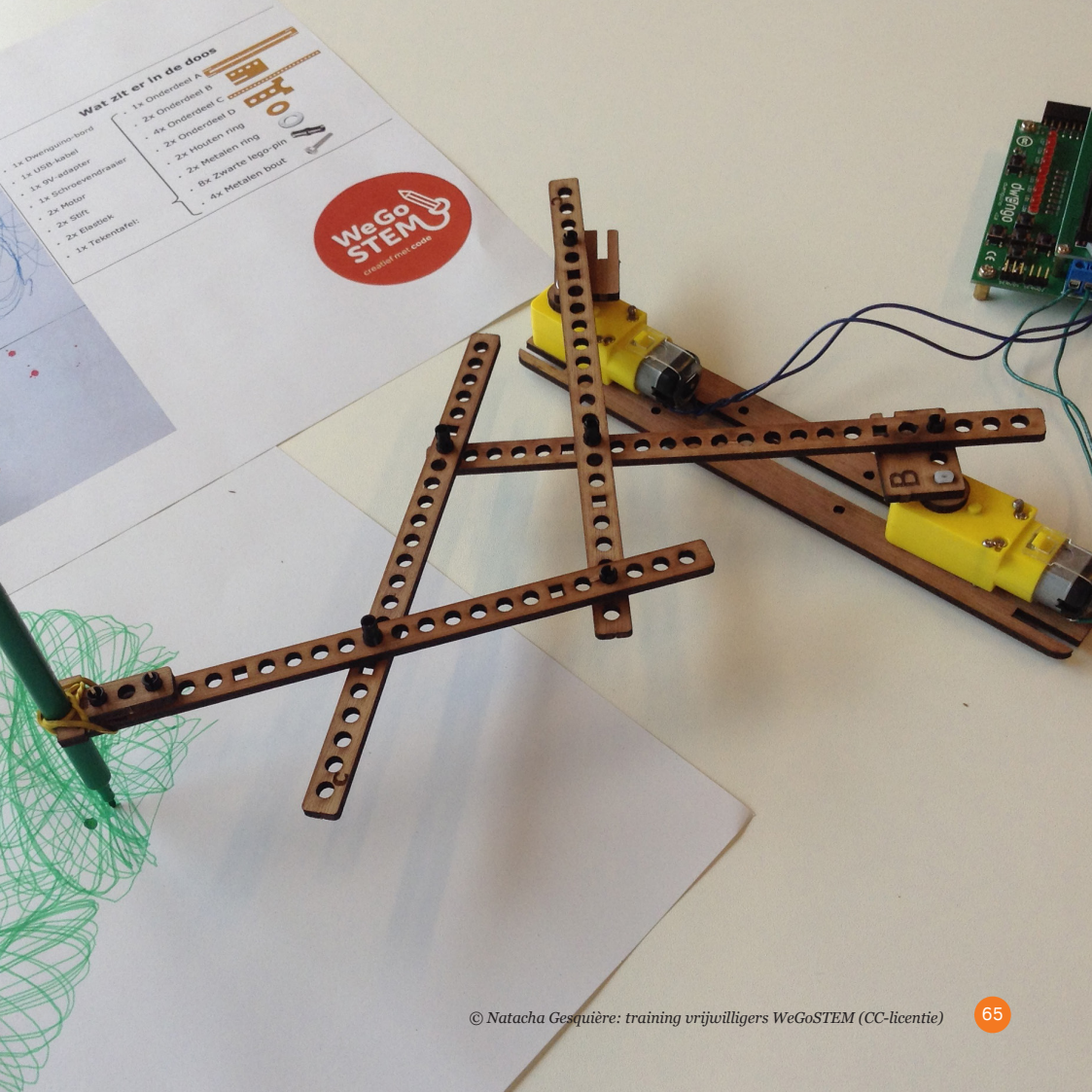
Er is er een groot aanbod in activiteiten. Sommige activiteiten doe je zonder computer, andere met. De ene activiteit doe je op de speelplaats, een andere in de klas. Wil je een wow-effect bereiken bij de leerlingen dan laat je hen experimenteren met een robot. Er is vast een activiteit bij die jou en jouw leerlingen aanspreekt.

Kies een speelse opdracht om de interesse van de leerlingen op te wekken. Kies een **moeilijkere opdracht om bepaalde leerlingen uit te dagen.** Kies een **wiskundige opdracht om de leerstof op een andere manier aan bod te laten komen**. Je kunt de leerlingen laten programmeren om hen op een originele manier iets te laten vertellen over hun hobby of favoriete boek.

Programmeren in de klas leent zich uitermate tot differentiëren. Je kunt snel nieuwe opdrachten verzinnen op het niveau van elk kind. De leerlingen kunnen alleen of per twee werken aan een opdracht op hun eigen tempo. Bovendien zijn er ook **veel**

opdrachten op internet te vinden die geleidelijk in **moeilijkheidsgraad** stijgen. Zo hebben alle leerlingen, elk volgens hun eigen kunnen, **succeservaringen**.

Wissel oefeningen met een vast stappenplan af met open opdrachten die ruimte laten voor het creatief inzetten van de verworven kennis. Daag bijvoorbeeld de kinderen uit met een open challenge die ze naar eigen inzicht mogen oplossen. Geef net genoeg informatie zodat ze verder kunnen met de opdracht. Licht af en toe een tipje van de sluier op, maar laat de leerlingen hun programma uitvoerig testen, het evalueren en het verbeteren. De blije gezichten spreken boekdelen.



Colofon

ISBN 978-90-819917-4-2
D/2017/Dwengo/1
NUR 980, 257

We willen graag zoveel mogelijk mensen laten kennismaken met programmeren en ze warm maken voor wetenschap en techniek. Daarom brengen we dit boek uit onder Creative Commons:

De gebruiker mag:

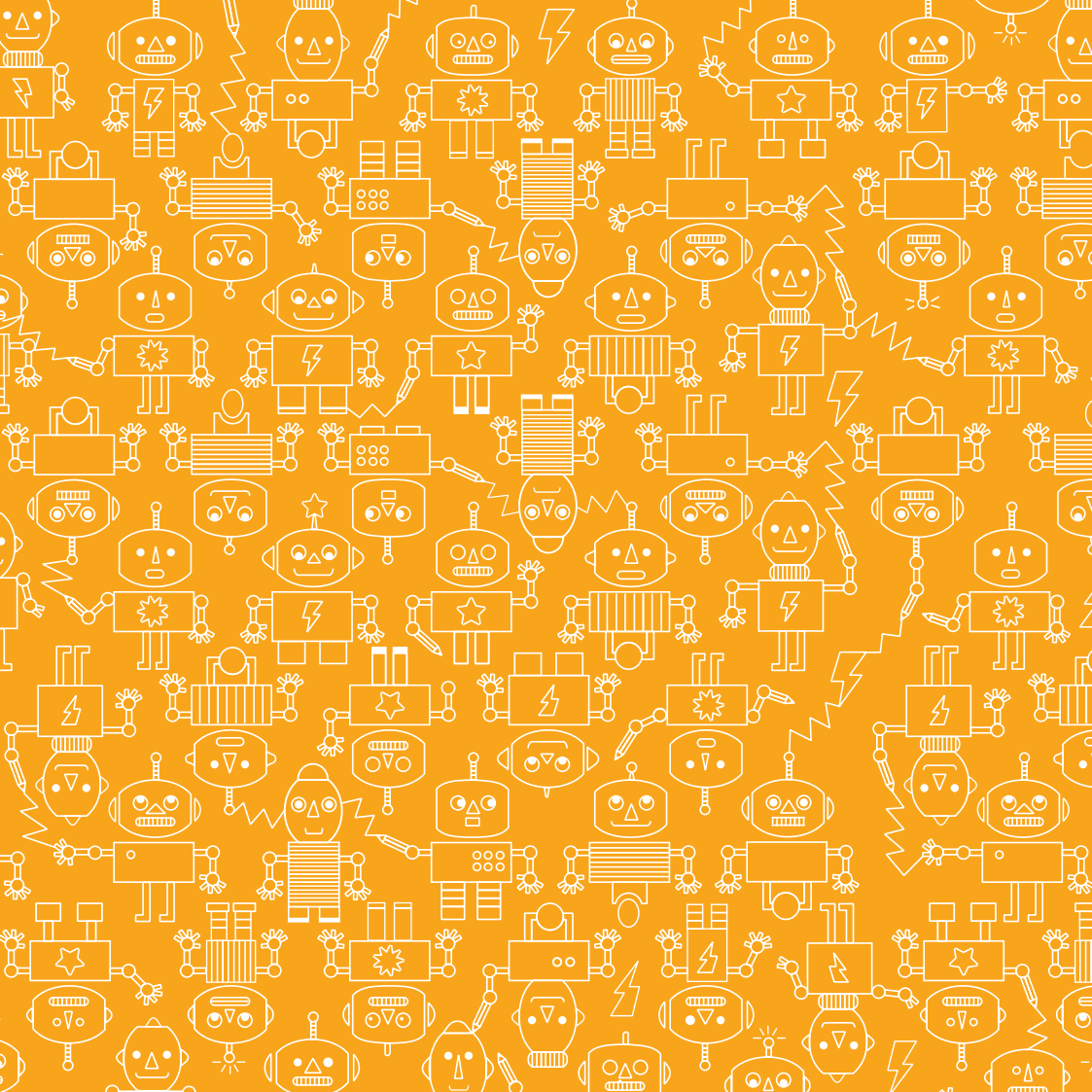
- het werk kopiëren, verspreiden en doorgeven
- remixen – afgeleide werken maken

Onder de volgende voorwaarden:

- **Naamsvermelding:** de gebruiker dient bij het werk steeds Dwengo vzw te vermelden (maar zonder de indruk te wekken dat Dwengo vzw instemt met je werk of jouw gebruik van het werk).
- **Niet-commercieel:** de gebruiker mag het werk niet voor commerciële doeleinden gebruiken.
- **Gelijk delen:** indien de gebruiker het werk bewerkt, kan het daaruit ontstane werk uitsluitend krachtens dezelfde licentie als de onderhavige licentie of een gelijksoortige licentie worden verspreid.

info@dwengo.org • www.dwengo.org

Design: Studio Blomme




WeGo
STEM
dwengo
9 789081 991742