VOORWOORD

Drie jaar lang studeerde ik Multimedia- en Communicatietechnologie aan de Erasmushogeschool Brussel. De opleiding zorgde ervoor dat mijn opgedane kennis uit het secundair (Multimedia) werd uitgebreid, en leerde me deze kennis om te zetten in praktische en gebruiksvriendelijke installaties en applicaties.

Ik draag dit werk (en voorafgaande jaren) op aan mijn ouders, die me hielpen en bijstonden waar ze konden. Verder richt ik dit dankwoord aan mijn mentor (zowel in de opleiding als daarbuiten) Wouter Van den Broeck. Ook de leden van Beyond.io, mijn broer (Koen Everaert) en zus (Leen Everaert) bedank ik graag voor hun praktische bijstand en goede input.

INLEIDING

De inspiratie voor dit onderzoek komt oorspronkelijk van één van de installaties op het Brusselse evenement “Nuit Blanche” (een kunstenfestival in Brussel http://nuitblanchebrussels.be/nl/). Eén van de docenten liet een video zien waarin een connectie werd gemaakt tussen fysieke componenten en een projectie. In het gesprek dat volgde werd hier een educatief doel aan gekoppeld. We spraken over fysieke componenten en een narratief dat het geheel kon ondersteunen. Het project werd ontwikkeld om kinderen zin geven in programmeren en hen de eerste beginselen en denkwijzen ervan bijbrengen.

Kinderen krijgen in dit project een verhaal aangeboden, in de vorm van een boek. Bepaalde pagina’s zetten kinderen aan tot interactie. Dit gebeurt door het leggen van afgedrukbare kaarten, die elk een klein stukje code symboliseren.

Doorheen de maanden is dit project sterk geëvolueerd. Zo heeft vooral het projectieoppervlak vele vormen gekend, van een tafeloppervlak naar een fysiek boek. Ook de code-elementen die een stukje code symboliseren, zijn doorheen de afgelopen maanden vaak van vorm en functionaliteit veranderd.

In dit project ontwikkelde ik een applicatie om jonge kinderen de eerste inzichten in verband met programmeren bij te brengen. Dit gebeurt op basis van [metaforische code] die de kinderen in de juiste volgorde leggen om een vooropgesteld doel te bereiken. Ik gebruik hierbij een verhaal waarmee ik de motivatie wil verhogen door gebruik te maken van een empathisch element.

Ik zocht voor dit project hulp bij externen, zoals Zach Lieberman (grondlegger van Openframeworks, en open source c++ platform), Pieter Feys (student pedagogie aan Universiteit Gent, met een grondige kennis in interactieve verhalen), en Het Geluidshuis (productiehuis van hoorspelen).

Dit verslag begint met een uitgebreide projectomschrijving. Daarna ga ik dieper in op programmeren en de wijze waarop dit zich verhoudt tot achterliggende denkvaardigheden. In het derde deel bespreek ik de rol van een narratief in dit project. Het laatste deel beschrijft de opbouw van de applicatie en diens achterliggende code.v

ONDERZOEKSVRAAG

PROBLEEMSCHETS

Er zijn reeds een aantal applicaties die als doel hebben het doel kinderen te leren programmeren. Dit gebeurt vaak in de vorm van een spel of kleine opdrachten. Deze applicaties missen vaak een doel, en blijft motivationeel beperkt. Een narratief kan ervoor zorgen dat kinderen langer gemotiveerd zijn, en minder snel afhaken omdat opdrachten op elkaar beginnen lijken. Elke nieuwe opdracht in dit project is ook verschillend van de vorige. Het kind doorloopt dus een aantal opdrachten, en weet na de laatste opdracht elk onderdeel correct te gebruiken. Het doel van het narratief is deze onderdelen aan elkaar te binden, zodat de opdrachten minder als oefeningen of taken aanvoelen.

DOELGROEP

De doelgroep van dit project ligt vooral tussen acht en twaalf. Dit is een leeftijdscategorie waarin lezen vaak niet veel moeite meer vraagt, en de focus dus kan liggen op het schrijven van de code.

DOEL VAN HET ONDERZOEK

Het onderzoek dient duidelijk te maken of een narratief effectief kan bijdragen aan een leukere leerervaring. Die leukere ervaring kan bereikt worden door een hands-on ervaring, en een empathische band met het hoofdpersonage van het verhaal. Het resultaat van dit onderzoek omvat ook een proof of concept in de vorm van een applicatie die een extra dimensie geeft aan een verhaal, en het toestaat meerdere verhalen met hetzelfde programma te kunnen gebruiken.

CENTRALE VRAAG

Hoe kunnen we kinderen aanzetten tot coderen in een narratieve omgeving, en aan welke voorwaarden moet de digitale omgeving voldoen om succesvol te zijn? Succesvol betekent in deze context het motiveren van de leerlingen, en het aanbieden van een specifiek doel waarnaar gestreefd kan worden.

DEELVRAGEN

De onderzoeksvragen laten zich indelen in een aantal categorieën. De eerste categorie bestaat vooral uit vragen omtrent het leerproces van het kind. Verder wordt ook vermeld welke elementen bij kunnen dragen aan een betere, leukere leerervaring. Het gaat hier dan vooral over de volgende vragen:

- Heeft handelend leren een positieve invloed, en hoe kan dit geïntegreerd worden in dit project
- Heeft empathie effect op leren en hoe werkt dat
- Leren in klasverband; welke randvoorwaarden moeten voldaan zijn om efficiënt te kunnen leren

Verder wordt het aanleren van programmeren aan kinderen in vraag gesteld:

- waarom zou je kinderen leren programmeren
- welke invloed heeft programmeren op kinderen
- hoe kan dit technisch gerealiseerd worden

Welke eerste basiselementen van programmeren kunnen bij jonge kinderen aangeleerd worden.
Naarmate het onderzoek vorderde, kwamen een aantal andere onderzoeksvragen aan bod. Deze gaan vooral over de functionaliteit van het programma, maar eveneens over de interactie die het programma aanbiedt aan de gebruiker.

- welke software en hulpmiddelen zijn reeds beschikbaar
- hoe kan een kind gemotiveerd worden een doel te bereiken

Sommige van deze vragen werden echter nooit beantwoord, omdat de applicatie of het verhaal hen zinloos maakte. Zo werd leren in klasverband overbodig omdat de omgeving variabel werd. Het huidige programma zorgt er namelijk voor dat men met simpele tools thuis of in een coderdojo-omgeving aan de slag kan gaan (een coderdojo is een evenement waar kinderen leren programmeren, met behulp van programma's zoals Scratch).

ONDERZOEKSRESULTAAT

Het vooropgestelde resultaat zal een verhaal bevatten, dat op bepaalde momenten het kind zal aanzetten een stukje te programmeren alvorens verder te gaan met het verhaal. Dit proces wordt ondersteund door een geprojecteerde interface met de nodige tips. Het aantal stukjes code dat hiervoor gebruikt kunnen worden zal in dit onderzoek beperkt worden. Het programma zal gebruik maken van empathie via narratieve elementen en andere technieken om het kind te motiveren.

ONDERZOEK


story board
PLANNING
CODING
MAPPING
story integration
More plates
Animation
point & follow system
interface design
testing & perform.
debug
IMAGING
char.
Anim
Gif
text story
Story
code
UX

WAAROM LEREN WE KINDEREN PROGRAMMEREN?

In 2014 gaf de EU (www.eun.org) een rapport [1] uit onder de titel 'computing our future' dat het integreren van programmeren in het lager onderwijs bespreekt. Het gaat hier vooral over landen die dit reeds doen (zoals Estland) maar eveneens over landen die dit van plan zijn in de nabije toekomst (waaronder België). Ook het nut van programmeren in een schoolomgeving wordt besproken met het oog op de toekomst van zowel de economie (meer programmeurs), maar belangrijker nog, met het oog op de ontwikkeling van het kind.

Mitch Resnick (MIT, medeoprichter van Scratch) vergelijkt programmeren met schrijven;

"slechts weinig van de kinderen worden professioneel schrijver, maar kunnen schrijven is steeds handig.

"Hij vindt dat dit ook geldt voor programmeren. Programmeren stimuleert volgens Mitch het creatief denken, het systematisch redeneren en het samenwerken in teamverband. Verder kan het kind deze kennis gebruiken om zijn creativiteit te uiten (Mitch geeft hier het voorbeeld van kinderen die moederdagkaarten programmeerden).

Dit inzicht komt terug bij Derek Cabrera. Hij beschrijft in zijn TED-talk dat sommige kinderen het creatief denken verleerd hebben. Hij geeft hierbij het voorbeeld van Lego. Dit werd vroeger verkocht in dozen, zonder instructiehandleiding, terwijl deze momenteel vooral verkocht wordt in voorgemaakte elementen, waardoor men verplicht is de aanwijzingen te volgen. Dit omdat de blokjes slechts één enkel model toelaten. Volgens hem verleren kinderen hierdoor het creatief denkproces.

Tijdens dit onderzoek werd ik op het bestaan gewezen van 21st Century Skills, een organisatie die geregeld onderzoeken uitbrengt omtrent vernieuwing en integratie van deze skills in het onderwijs. 21st Century Skills stelt dat onze westerse maatschappij van een industriële samenleving naar een kennis-samenleving veranderd is. Hun doel is het onderwijs aan te passen om beter aan te sluiten bij dat beeld van de 21ste eeuw. De Europese Unie

is hier reeds mee bezig, samen met bijvoorbeeld Unesco en een aantal Amerikaanse organisaties. Deze discussie wordt helaas nog niet helemaal doorgetrokken naar het onderwijs. Skills zoals online communicatie die even belangrijk zouden moeten zijn als wiskunde of Nederlands. Joke Voogt (bijzonder hoogleraar ICT & Curriculum aan de Faculteit der Maatschappij- en Gedragswetenschappen van de Universiteit van Amsterdam) is voorstander van een publiek debat die deze discussie opentrekt waarin ook ouders en studenten hun mening kunnen uiten.

DIGITALE SAMENLEVING

De digitalisering van de samenleving is vooral merkbaar in de tewerkstelling in het Westen. Het aantal beroepen waarin productiewerk verricht wordt, daalt snel, maar de beroepen waarin men creatief moet nadenken en probleemoplossend moet werken, zitten in de lift. In die laatste categorie is een degelijke IT-kennis interessant om praktische redenen. Vooral voor communicatie, maar eveneens om de efficiëntie van het werk te verhogen. Daarom is het belangrijk dat we dit al van jongs af aan kunnen meegeven. Maar ook kan de logica van programmeren een oefenplaats zijn voor probleemoplossend denken.

21st CENTURY SKILLS

De term 21st Century Skills (niet te verwarren met de vereniging die deze term uitbrengt) heeft betrekking op een pakket van vaardigheden die nodig zijn om degelijk te functioneren in de huidige samenleving. Deze samenleving wordt onderverdeeld in een aantal groepen, “digital natives” en “digital immigrants”. De digital natives zijn in dit onderzoek de belangrijkste doelgroep. Zij zijn mee opgegroeid met de technologie, en ontdekken nieuwe ontwikkelingen en toepassingen, volledig onbevangen. Deze groep heeft weinig tot geen moeilijkheden met het gebruiken van nieuwe technologie, en zijn hier snel mee weg. Tot de groep van “digital immigrants” behoren de volwassenen die de technologie hebben zien groeien. Sommigen zijn snel weg met vernieuwing, anderen twijfelen eerder alvorens deze te omarmen. Ook deze groep gebruikers leert omgaan met de nieuwere technologieën, maar hebben soms meer moeite deze vlot te integreren. Het grote verschil tussen deze twee groepen is het gemak waarmee men overschakelt. Dit is vooral merkbaar in de manier van interactie. Terwijl de immigrants vooral vasthouden aan mail en formelere communicatie (ook face-to-face), maakt de groep van digital natives gebruik van sociale media zoals whatsapp en facebook, en schakelt vlot over wanneer een nieuwe vorm beschikbaar is (vb: snapchat). Beiden halen voordeel uit hun manier, maar in dit onderzoek leg ik de focus op de groep van natives.

21st Century Skills worden onderverdeeld in een aantal vaardigheden [2]. Deze worden vaak ondersteund door een degelijk gebruik van ICT. We spreken hier bijvoorbeeld over vaardigheden zoals:

- samenwerking
- kennisconstructie
- ICT gebruik voor leren
- probleemoplossend denken en creativiteit
- planmatig werken

De nadruk in dit onderzoek ligt op het probleemoplossend denken. Dit omdat deze vaardigheid gestimuleerd wordt door programmeeractiviteiten.

VOORWAARDEN VOOR EEN KRACHTIGE LEEROMGEVING

Met een krachtige leeromgeving bedoelen we een leeromgeving die participatie, betekenisgericht leren, levensechte contexten en zelfsturing mogelijk maken. Een dergelijke leeromgeving draagt bij tot de kwaliteitsverbetering van het onderwijsaanbod en tot de verhoging van het leerrendement bij alle leerlingen [3].

Om de activiteit om kinderen te leren programmeren effectief te maken, moeten ze dus zelf kunnen handelen (participatie). In het project realiseren we een betekenisvolle context via een narratief. Zelfsturing stimuleren we door de kinderen zelf oplossingen voor het gestelde probleem te laten zoeken. Het narratief zelf zorgt voor een ontspannen staat waarin de leerling snel en efficient kan leren.

Neurowetenschapper prof. C. Lafosse verwijst naar het belang van motorisch handelen in leersituaties. Via het manipuleren van concrete voorwerpen prikkelen we zenuwbanen die in onze hersenen verbindingen leggen waardoor we leren [4]. In dit project koos ik dan ook niet voor een virtuele oplossing (muis en toestenbord, touchscreen) maar voor het concreet plaatsen van objecten in een gekozen volgorde. Dit werkt anders dan games of applicaties via het scherm.

ALTERNATIEVEN

Natuurlijk zijn er reeds een aantal applicaties in omloop die zich focussen op het aanleren van code aan kinderen. Deze beperken zich vaak tot één enkele soort opdrachten, waardoor deze applicaties vaak repetitief aanvoelen, ondanks de achterliggende mogelijkheden.


PROGRAMMEREN

PROGRAMMEREN

Men kan het natuurlijk niet hebben over programmeren zonder de meest pure vorm te bespreken. Programmeren kent vele definities, Ik ben gegaan voor de definitie die de EU in zijn rapport [1] vermeld:

"Het kunnen realiseren van een algoritme in machinetaal, de stappen die genomen zijn kunnen interpreteren, [...] het kunnen compileren, runnen en debuggen van een programma, alsook het kunnen identificeren en hergebruiken van designpatterns"

Dit soort programmeren heeft een stijle leercurve. Het duurt even vooraleer men een eigenlijke output heeft, en het aanleren van de syntax neemt de nodige tijd in beslag. Kinderen kunnen hierdoor gedemotiveerd raken, en haken af als gevolg. Daarom wordt in dit project gebruik gemaakt van een abstractere vorm hiervan. Een vorm waarin de syntax niet zichtbaar is voor de gebruiker, en deze ook niet aan de nodige haakjes, komma's of punt-komma's moet denken.

Programmeren laat zich onderverdelen in een aantal categorieën.

ALGORITMISCH DENKEN

Algoritmisch denken wordt door de EU omschreven [1] als

"de mogelijkheid een probleem op te lossen via een stap-voor-stap oplossing (algoritme), waarin deze bestaat uit een eindige set duidelijk gedefinieerde stappen".

Dit soort denken wordt vaak gebruikt tijdens het programmeren. Het komt erop neer dat de programmeur (in dit geval ook de gebruiker) een probleem dient te verdelen in een set kleinere deelproblemen, om daarna deze deelproblemen te kunnen oplossen via een stappenplan. Men moet hierbij natuurlijk ook rekening houden met de beginsituatie en het einddoel.

De bijhorende programmeervorm omvat het programmeren zonder effectief code te schrijven. Enkel basisacties zijn hierin opgenomen. Variabelen en andere structuren komen hierin niet aan bod. Het oplossen van een probleem (meestal het bereiken van een bepaalde plek op het speelveld) kan hier opgelost worden door een opeenvolging van stappen zoals "stap naar rechts", "stap naar boven", en meer van dit soort blokken. Zoals eerder vermeld is deze vorm zeer geschikt voor jonge kinderen. Ze kunnen hier snel mee aan de slag en zien zeer snel resultaat. Dit is een ideale manier om snel en duidelijk een eerste probleem onder te verdelen in kleine stappen, en de drempel naar het volgende niveau is zeer laag. Voor ouderen is dit echter niet heel uitdagend, omdat de gelijkheid met het volgende level vaak te groot is en het als bandwerk aanvoelt.

Het aanleren van deze vaardigheid gebeurt reeds in bestaande programma's. Onder andere Scratch [https://scratch.mit.edu/] leert dit aan door gebruik te maken van een serie uitdagingen die het kind moet doorlopen. Tijdens dit proces leert hij het probleem op te delen in kleine acties. Nadat de gebruiker alle stappen heeft doorlopen, kan deze aan de slag met alle blokjes die Scratch bevat, om zelf creaties te maken.

Scratch kan een opstap zijn naar programma's of programmeertalen als Java, Visual Basic of Javascript, zonder de leerling

al direct te confronteren met de complexiteit van een volwaardige programmeertaal. Dit kan namelijk een grote drempel vormen, omdat een programmeur veel tijd doorbrengt een code te debuggen. Het werken met voorgemaakte blokjes neemt deze drempel weg.

METAFORISCH PROGRAMMEREN

In deze stap (niet noodzakelijk in deze volgorde) komen de eerste programmeerstructuren naar boven. Kinderen en volwassenen maken kennis met een “for-loop”, en een “if else” structuur. Ook variabelen komen hier aan bod.

De gemakkelijkste manier om deze vorm te gebruiken is door de code onder te verdelen in kant en klare blokjes. Deze kan de gebruiker dan aaneenschakelen en een groter geheel vormen. In veel applicaties kan deze vorm voorgesteld worden als een efficiënter alternatief voor algoritmisch denken.

Dit is tevens een vorm die stap voor stap aangebracht kan worden, waardoor de gebruiker het overzicht kan bewaren, en niet verloren loopt in de eindeloze mogelijkheden.

FLOW PROGRAMMEREN

Flow programming wordt voorgesteld in programma’s zoals pure data. De gebruiker verbindt blokjes die een actie of voorwaarde voorstellen met elkaar, om zo het programma uit te voeren. Met een vorm als deze is het echter moeilijk programmeerstructuren toe te passen.

KLASSIEK PROGRAMMEREN

Dit is de meest bekende vorm van programmeren. In deze vorm is de syntax van groot belang, en vele programmeertalen maken gebruik van deze vorm van programmeren. De nadruk ligt in deze vorm op het correct aanroepen van de juiste functies, en er moet vooraf heel hard nagedacht worden over de manier van implementatie. Deze vorm neemt ook de nodige tijd in beslag.

CONCLUSIE

Programmeren leert het kind op een andere manier nadenken over een probleem. Het bevordert het opdelen van een groot probleem in kleinere deelproblemen, en vlotter naar een oplossing te zoeken. Het algoritmisch nadenken staat het kind ook toe bepaalde acties die herhaald moeten worden misschien efficiënter uit te voeren.

Om dit aan te leren gebruikt men best een metaforische vorm van programmeren. Deze is duidelijk, en heeft een lage drempel. Het is overzichtelijk en kan stapsgewijs aangeleerd worden.


STORYTELLING

Om volledig te kunnen begrijpen wat een narratief kan betekenen voor een gebruiker, en hoe we dit kunnen gebruiken, moeten we natuurlijk eerst definiëren hoe een verhaal in elkaar zit.

EVOLUTIE VAN VERHALEN

We vertellen al verhalen sinds het begin van mensenheugenis. In het begin werden verhalen vooral verteld om geschiedenis door te geven, of fenomenen te verklaren. Met de komst van een geschreven taal zijn we deze verhalen ook beginnen neerschrijven. Het is echter pas sinds de uitvinding van de boekdrukkunst dat we deze in geschreven vorm ook beginnen verspreiden zijn. De tijd tussen deze eerste evoluties spreidt zich echter over duizenden jaren. Pas met de komst van de film is het echt snel beginnen gaan. Na film kwam radio en tv, waarna computers en het internet het rijtje vervoegden om verhalen door te geven.

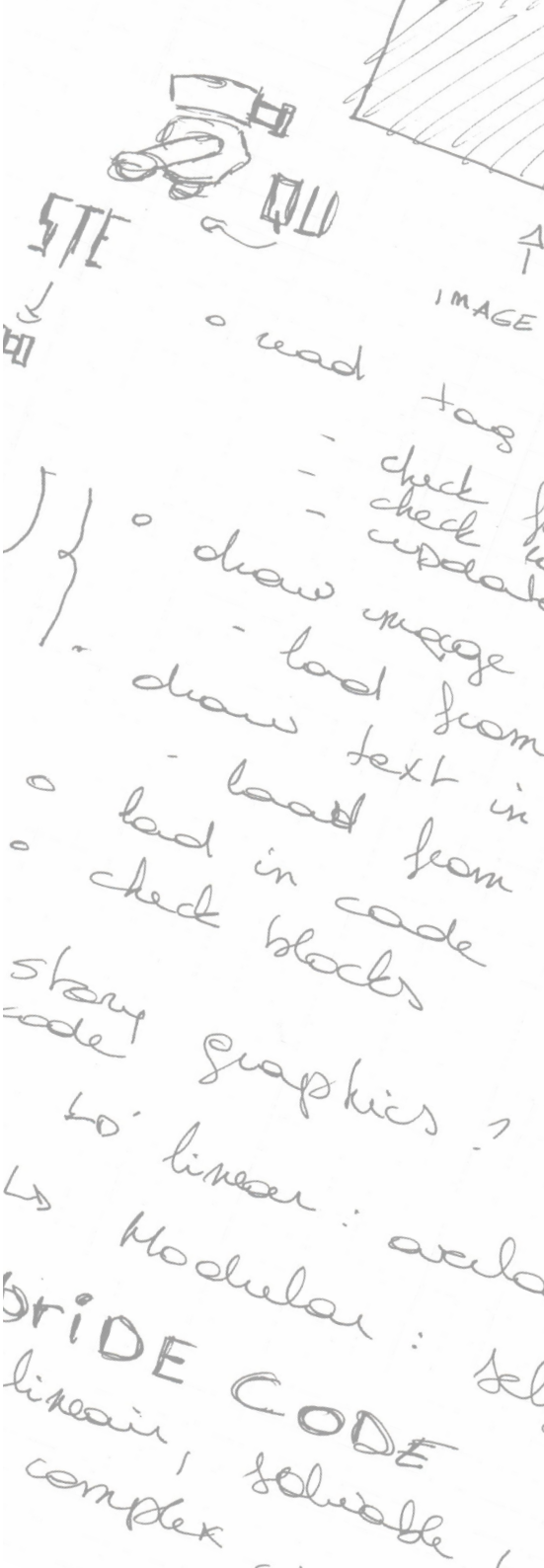
Elk van deze uitvindingen bracht met zich mee dat artiesten een nieuwe manier hadden om verhalen te vertellen, maar ook hun verhalen aan te passen naar het gebruikte medium. Denk maar aan de uitvinding van de stop-motion truck in de filmwereld, waarmee het plots mogelijk werd verhalen anders vorm te geven, en andere verhalen te vertellen. Acties moesten niet meer mogelijk zijn, men kon truckeren.

In 2005 kwam Youtube uit. Dit is een van de grotere mijlpalen in het vertellen van verhalen. Het werd mogelijk voor iedereen met een camera (elk soort camera) eigen verhalen en films te verspreiden. Dit in combinatie met het goedkoper worden van de filmcamera en smartphones zorgt voor een enorme boost in het aantal en de kwaliteit van verspreide films, en dus ook verhalen.

Nieuwe technologie gaat ook vaak gepaard met een nieuwe vorm van verhaalvertelling. Denk maar aan Vines, één van de recentere vormen hiervan. DIt is een online platform waarin de gebruiker een luttele 7 seconden kreeg om zijn punt, of verhaal, in duidelijk te maken.

Verhalen zijn doorheen de jaren ook niet alleen veranderd in vorm, maar ook in interactiviteit. In de middeleeuwen was het gebruikelijk dat bards (vertellers, meestal uit Wales en Schotland, opgelied in het vertellen van verhalen) hun verhalen aanpasten aan de plaats waar deze verteld werden. Inwoners werden verweven in het verhaal, en namen deel in de gebeurtenissen. Dit verdween met de uitvinding van de boekdrukkunst[7]. Het is pas sinds de uitvinding van narrative games dat verhalen weer interactiviteit hewinnen.

"we zijn op een punt in digital storytelling waar we niet goed weten wat te vertellen, maar enthousiast zijn over de mogelijkheden." dit was een situatie die voorkwam in het begin van film, waar mensen filmpjes animeerden waarin geen verhaal verteld werd, maar vooral de effecten getoond werden.



STORYTELLING

Het vertellen van verhalen heeft geen echte regels, eerder wegwijzers. Er zijn een aantal guidelines waar aan gedacht kan worden, maar die niet nodig zijn om het schrijven van verhaal tot een goed einde te brengen. Vaak zijn de verhalen die afwijken van deze regels eerder succesvol, omdat de lezer verrast wordt.

Kenmerken van een goed verhaal zijn (Tesselaar & Scheringa, 2009):

- het staat op zichzelf, heeft een begin, midden en eind
- het is authentiek (waar)
- het heeft een hoofdpersoon die iets meemaakt (bijvoorbeeld de verteller), plotwendingen en andere narratieve (verhalende) elementen
- het is persoonlijk
- het roept emotie op of wordt met emotie verteld.

FUNDAMENTEN VAN INTERACTIVE STORYTELLING

Als we spreken over digital storytelling, bedoelen we vooral een verhaal dat ondersteund wordt door een extra laag, die vaak computergestuurd is. Die laag kan aanvullend zijn, zoals het gebruik van augmented reality in een boek. Hij kan echter ook de enige laag zijn die we zien. Dit is vaak het geval in games of films. De gebruiker ziet het verhaal slechts afgespeeld op zijn scherm, en heeft hierop niet veel invloed.

De focus mag niet enkel liggen op het medium, er moet ook gedacht worden aan de manier van interactie, aan de hoeveelheid ervan, of het aantal variabelen waar de lezer mee te werk kan gaan. Men neemt hier vaak het voorbeeld van games. Deze werken eveneens met een narratief, dat vaak de input van users nodig heeft om te vorderen. Omdat dit proces zeer persoonlijk is, is het moeilijk een algemeen beeld op te stellen van hoe deze input verloopt. De ervaring van de interactiviteit varieert van speler tot speler.

De meeste focus in dit stuk ligt op de interactiviteit tussen speler en narratief.

Digitale verhalen (zoals games en interactive stories) hebben het voordeel abstracte onderwerpen meer toegankelijk te maken voor verscheidene doelgroepen. Zo zijn er bijvoorbeeld veel Youtube kanalen die gebruik maken van een narratief om ingewikkelde wetenschappelijke onderwerpen op een klare manier uit te leggen. Ze kunnen ook helpen leerstof aantrekkelijker te maken, en zo minder gemotiveerde studenten toch aan te zetten tot leren. Het gebruik van de creativiteit van de gebruiker en zijn diepere gedachten creëren een sterker gevoel van doel in het leren (Hull and Katz, 2006).

Het beste leren we uit eigen ervaring. Verhalen zijn daarvoor ideaal, ze zorgen ervoor dat onze hersenen op zoek gaan naar connectie en betekenis. Ook worden bepaalde neuronen in onze hersenen geactiveerd, waardoor het lijkt alsof we het zelf meemaken. Een goed verteld verhaal kan aanvoelen alsof we het zelf meegemaakt hebben. Het is voor ons brein ook gemakkelijker verbindingen te vormen bij het aanhoren van een verhaal of anekdote [5].

Een narratieve context voegt dus betekenis toe, en legt verbindingen in je hersenen naar contexten die je herkent, waardoor je het beter opneemt en dus beter leert. Leren is verbindingen leggen in de hersenen, en een betekenisvolle context helpt daarbij.

Als we een verhaal aanhoren, zijn we zowel ontspannen als oplettend. Dit is een goede staat om nieuwe dingen te leren, en er een positieve ervaring aan te kopelen.

Ik ben lang uitgegaan van het idee het verhaal te laten variëren naarmate andere stukken code neergelegd werden, maar dit bracht met zich mee dat het fysieke boek meerdere delen moest bevatten. Naarmate de speler andere code neerlegde zou hem/haar meegedeeld worden door te gaan naar een bepaalde pagina, waar het verhaal dan een andere wending nam. Dit is mogelijk, maar is complex. Vooral omdat het verhaal dan via een boomstructuur verloopt, en verschillende zijverhalen moeten geschreven worden.

Verhalen kunnen dus een zeer belangrijke rol spelen in het leren van nieuwe informatie en skills. Zowel het verhaal als de staat waarin deze ons brengt dragen bij aan beter en efficienter leren. Natuurlijk zijn verhalen ook gewoon ontspannend en maken het leren veel aangenamer.

NARRATIVE DRIVEN PROGRAMMING

Na het inhoudelijke onderzoek ging ik van start met de concrete uitwerking. Dit bracht de nodige extra vragen met zich mee. Ik begon met de zoektocht naar een verhaal, waarna de code de aandacht overnam.

DOEL VAN HET PROJECT

Het project is ontworpen om kinderen op een leukere manier kennis te laten maken met algoritmes en de basis van programmeren. Dit op een narratieve manier, en een fysiek boek. Het gehele programma werkt als een platform waar schrijvers en leerkrachten zelf boeken kunnen invoeren. Het kind leest het boek, waardoor kinderen een empatische band crëren met het hoofdpersonage. Op een aantal momenten zal het boek dienen om een programmeermoment op te roepen. Dit gebeurt door het scannen van het boek. Daarna kan het kind met kaarten een stukje programmeren, en laten uitvoeren. Deze uitvoer wordt natuurlijk visueel weergegeven, waardoor de gebruiker fouten gemakkelijk kan opsporen.

ROL VAN HET VERHAAL

Eerder werd al vermeld dat het verhaal dynamisch ingevoerd kan worden. Dit omdat een vast verhaal de mogelijkheden te hard beperkt. In mijn zoektocht naar een degelijk verhaal viel me op dat kinderen soms reeds in klasverband gebruik maakten van een mascotte. Denk maar aan “Jules” [6]. Kinderen bouwen doorheen de lessenreeks een zekere empathie op met dit personage.
Het werken met een dynamisch verhaal laat toe dat dit personage ook gebruikt kan worden met de applicatie. De empathie is reeds opgebouwd, en de toepassing zal dus optimaal kunnen functioneren.

ZOEKTOCHT NAAR EEN NARRATIEF

In eerste instantie dacht ik het verhaal zelf te schrijven, om het zo volledig mogelijk te doen aansluiten bij de noden van het programma. Beyond.io raadde me aan het verhaal eerst te schrijven alvorens te beginnen programmeren. Ik zou op deze manier voorkomen dat ik anders functionaliteiten zou ontwikkelen die achteraf onnodig zouden zijn.

Omdat ik zelf niet de capaciteiten bezit een kinderverhaal te schrijven, deed ik beroep op andere schrijvers. Ik contacteerde onder andere Marc De Bel en Antony Horowitz. Marc De Bel antwoorde enthousiast, en stond toe dat ik een van zijn verhalen mocht gebruiken en zelfs aanpassen. Bij nader inzien leek dit echter geen goede optie, omdat specifiek zijn verhalen zowel te lang als te gevarieerd zijn om een degelijke programmeer oefening op te leveren. Verder kon hij mij ook niet helpen aan digitale versies van tekeningen, vaak gebruikt in zijn boek. Daarom contacteerde ik het Geluidshuis.

Het Geluidshuis is een productiehuis dat vooral luisterverhalen (de zogenaamde hoorspelen) uitbrengt. Op het moment dat ik hen contacteerde zaten de kinderen volgens het concept nog aan een grote tafel. Het gebruiken van een

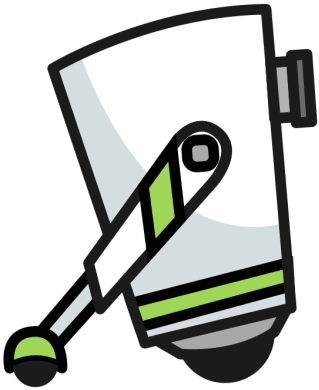
hoorspel loste het kijkhoekprobleem op. Dit wil zeggen dat de beeldende voorstelling van het verhaal niet naar een bepaalde richting kon wijzen, omdat kinderen errond zaten. Het luisteren lostte dit probleem op.

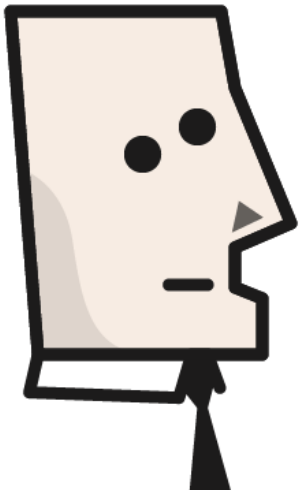
Eind januari maakte ik met het Geluidshuis een afspraak. Na een korte wachttijd kon ik met hen de volledige situatie bespreken. Al snel werd het duidelijk dat het moeilijk zou worden auteursrechtelijke toegang te krijgen tot de werken. Een nieuw werk maken ging zeker ook niet lukken, daarvoor waren de productiekosten veel te hoog. Er werd me wel hulp beloofd moest ik zelf een verhaal opzetten. Ik kon rekenen op intellectuele bijstand, maar infrastructuur was te duur om te mogen gebruiken.
Werken met een hoorspel bracht helaas ook nieuwe problemen met zich mee. Onder andere dat de verhalen lang waren, en kinderen op elkaar zouden moeten wachten alvorens verder te gaan met het verhaal. Daarom besloot ik niet verder te werken met Het Geluidshuis.

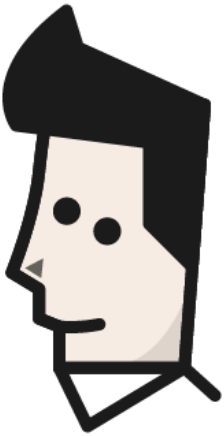
in plaats daarvan zou ik lesgevers, schrijvers en derden de mogelijkheid geven zelf een eigen verhaal in te voeren. Dit zou via een wegomgeving mogelijk zijn. De levels zouden ingevoerd kunnen worden via een webplatform, en zijn downloadbaar door het systeem.

SCHRIJVEN VAN EEN TESTVERHAAL

Ondanks het mogelijk maken van het invoeren van verhalen door anderen, was het natuurlijk nog steeds nodig een verhaal te schrijven om alle functionaliteiten uit te testen. In het verhaal beleeft Fitz (een robothersteller in de toekomst) een kort avontuur met zijn robotje QU. In dit verhaal moeten de kinderen kleine algoritmes doorgeven aan QU om Fitz te helpen in zijn avontuur. Om dit verhaal ten volle te kunnen testen was het ook nodig de nodige illustraties te maken. Fitz zelf nam niet veel tijd in beslag.







PROGRAMMASTRUCTUUR

Het programma werkt via een aantal fasen. Deze laten toe dat er toevoegingen kunnen gebeuren zonder te moeten zoeken in een ingewikkelde structuur van als-dan loops. Om de code duidelijk en gestructureerd te houden wordt er natuurlijk ook gewerkt met klassen. Niet alle klassen zullen in dit document uitgebreid besproken worden. Verder is de vooruitgang van de code ook deels uitgeschreven in de blog [http://www.crashlab.be/blog].

INTERFACEKLASSE

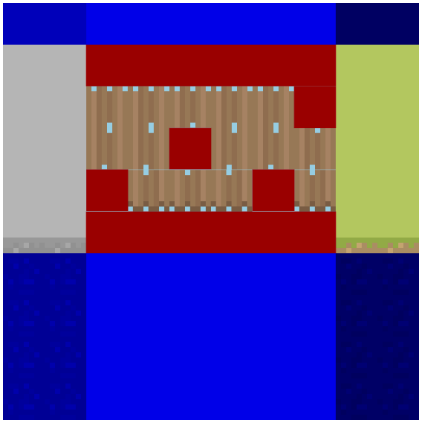
Deze klasse neemt de visuele output voor zijn rekening. Het zorgt ervoor dat de gebruiker weet waar het boek gelegd moet worden om gescand te worden, of welke codeblokjes met elkaar verbonden zijn. Om dit overzichtelijk te houden heeft elke status een eigen functie.
Deze klasse is pas vrij laat toegevoegd, omdat de nood aan een uitgebreide visuele output mij pas duidelijk werd na een gesprek met Beyond.io. Deze output bestond al reeds maar werd vooral door de hoofdklasse afgehandeld.

VERHAALKLASSE

Deze klasse wordt pas aangeroepen als een nieuw boek gescand wordt. Indien dit zich voordoet haalt deze klasse de nodige bestanden uit een online file-structuur. Deze zal eveneens een foutmelding weergeven als het gescand boek niet herkend wordt. Eens deze bestanden opgeslagen zijn zal deze klasse dienen als een centrale verzamelplaats waar andere klassen informatie uit kunnen halen, zoals bijvoorbeeld de compilatieklasse en de visuele weergave van het desbetreffende level.

COMPILERKLASSE

Deze klasse heeft een aantal zeer verschillende vormen gehad. Dit omdat de eigenlijke werking vrij ingewikkeld in elkaar zit. De klasse krijgt aanvankelijk een serie getallen door, die deze omzet in acties. Dit was natuurlijk een gemakkelijke functie, tot het gebruik van for-loops (herhaling van een aantal acties) in het blikveld terecht kwam. Hieronder wordt het gehele proces opgedeeld in een aantal stappen.



STAP 1

De eerste stap gebeurt reeds voor de gebruiker kaarten heeft neergelegd. In deze stap zal de compiler het speelveld opdelen in een aantal blokken. Sommige blokken zorgen ervoor dat de speler niet verder kan, anderen zijn zogenaamde valkuilen, waar de speler opnieuw moet beginnen indien hij deze raakt.

STAP 2

De speler heeft zijn kaarten gelegd, en heeft een groene kaart in de desbetreffende zone gelegd om duidelijk te maken dat hij zijn sequentie wil testen. De compiler zal hierop een eerste keer over de serie getallen lopen, om op zoek te gaan naar een for-loop. Indien deze tegengekomen wordt, zal hij de acties binnen deze loop herhalen. Het aantal herhalingen hangt af van wat de gebruiker ingegeven heeft.

Deze stappen zullen visueel voorgesteld worden tijdens de presentatie.

Het maken van het compile-systeem vergde veel tijd en inspanning. Dit proces is namelijk zeer abstract. In eerste instantie dacht ik aan een vastgelegd systeem, waar maar één oplossing mogelijk was. Omdat dit zeer beperkend is, werd dit idee al snel van de baan geveegd. Tijdens Resonate (multimedia conferentie) bedacht ik het systeem dat nu nog steeds gebruikt wordt.

STAP 3

Nadat alle speciale acties zijn omgezet in een lange serie afzonderlijke acties, zal deze klasse uitzoeken of de uitvoer mogelijk is. Omdat deze reeds weet waar de valkuilen en solide blokken zitten, kan deze stap voor stap nagaan of de speler een code heeft gelegd die lukt. Hij doet dit door actie voor actie uit te zoeken waar de speler staat, en of deze positie een gevolg heeft. Elke stap zal opgeslagen worden in een nieuwe Array (opeenvolging van acties).

STAP 4

Hierna (onafhankelijk van het resultaat van voorafgaande functie) zal de laatstgenoemde array uitgevoerd worden. Dit gebeurt traag, en stap voor stap, zodat de gebruiker kan volgen via de kaarten die hij reeds heeft neergelegd. Op deze manier kan de gebruiker ook terugvinden waar hij een fout heeft gemaakt. Terwijl deze acties worden uitgevoerd, wordt de kaart die overeenkomt met de actie aangeduid via de projectie.

TAGDETECTIE KLASSE

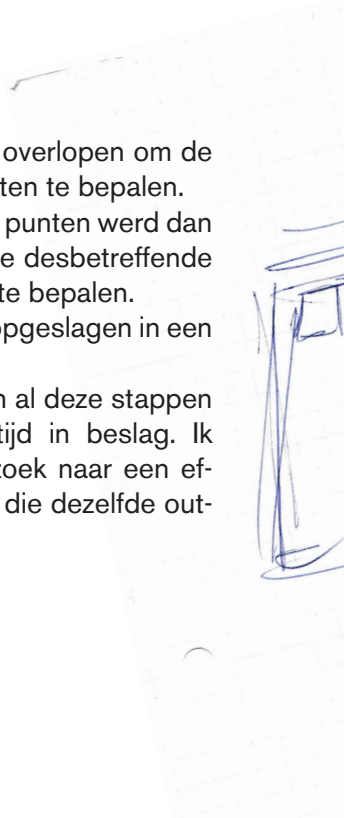
Tags zijn snel herkenbare afbeeldingen die eruit zien als een vereenvoudiging van een QR code. Ze worden algemeen gebruikt om informatie door te geven aan het programma. Onder andere het overgaan naar een codemoment, maar eveneens de positie van blokken code worden via deze tags geregeld. Die tags hebben een cruciale rol in het geheel. Daarom moest dit stukje software bijzonder efficiënt kunnen lopen, en de nodige ID's als output hebben.
Ik ging natuurlijk eerst op zoek naar een soortgelijke plugin, die me verder zou helpen. De gevonden systemen waren helaas slechts in staat met eigen tags te werken. Het aanpassen om bijvoorbeeld enkel bepaalde kleuren te tracken, het beeld negatief te maken, of de positie in een bepaalde vorm weer te geven zou teveel tijd hebben genomen. Verder was het ook een bijzonder zware plugin, omdat deze heel het cameraveld analyseert. Ik besloot zelf een eigen plugin te schrijven die dit waar kon maken.

ALPHA VERSIE

Eerlijk is eerlijk, mijn eerste versie liep zo mogelijk nog minder efficiënt dan de eerder vermeldde plugin. Ik maakte gebruik van een systeem dat de volgende stappen onderneemt:

De vorige positie van de tag bepaald het centrum van de huidige scan. 360 graden rond deze positie werd dan op verscheidene afstanden de kleur bepaald. Indien deze overeenkomt met de te scannen kleur, wordt deze toegevoegd aan een lijst.

Deze lijst wordt overlopen om de vier uiterste punten te bepalen.
tussen deze vier punten werd dan geïtereerd om de desbetreffende code en rotatie te bepalen.
deze info werd opgeslagen in een object.
Natuurlijk namen al deze stappen bijzonder veel tijd in beslag. Ik moest dus op zoek naar een efficiëntere versie die dezelfde output kon geven.

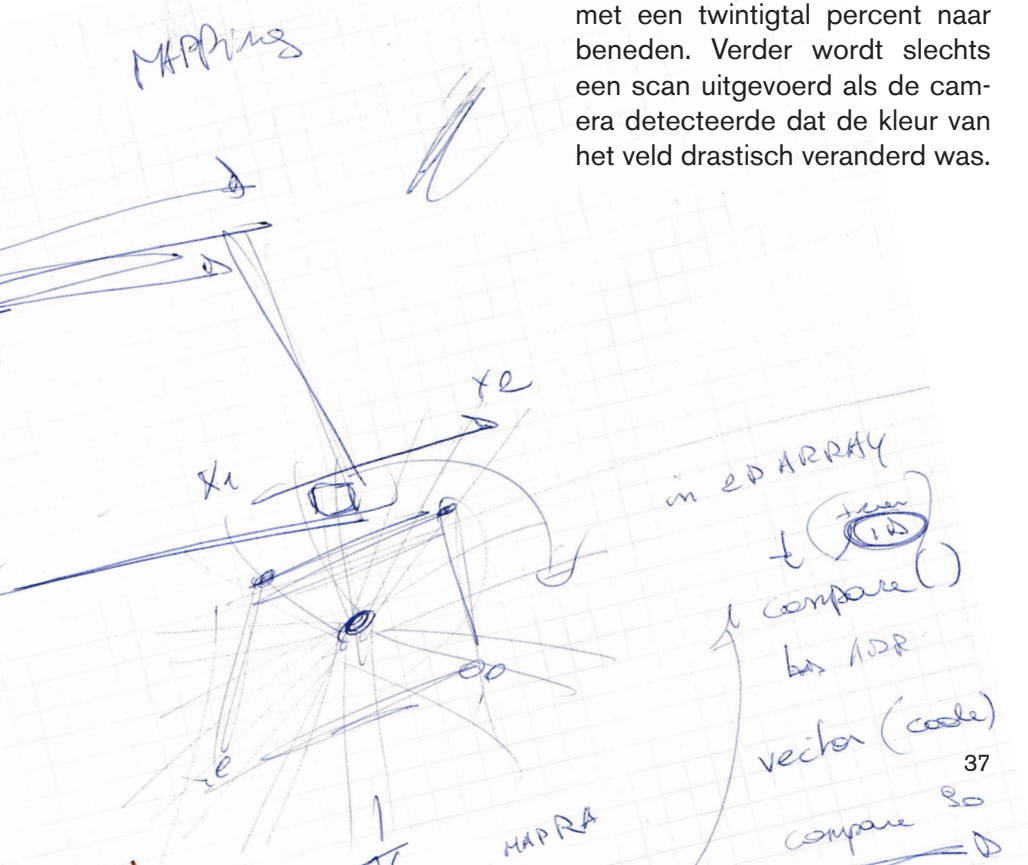


BETA VERSIE

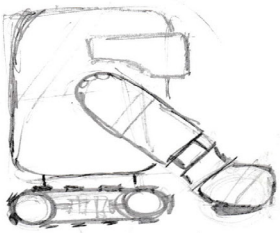
Het probleem met deze eerste versie is dat deze het gehele oppervlak moet afzoeken naar een tag. Dit kon efficiënter. Ik besloot het probleem op te lossen door gebruik te maken van bepaalde scanvelden. Dit zijn kleine velden die geprojecteerd worden waarin een kaart gescand kan worden. Het programma moest in plaats van 1400 op 900 pixels (standaard resolutie van een macbook pro scherm) dus nog maar 200 op 200 pixels overlopen.

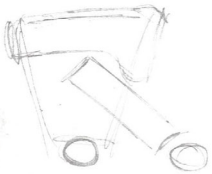
FINALE VERSIE

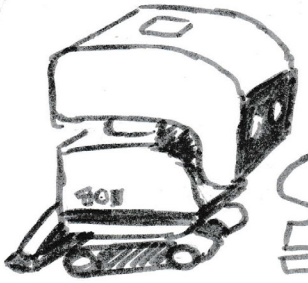
Omdat ik toch al werkte met een beperkt oppervlak, leek het me onnodig vanuit het midden te beginnen scannen. Bij gevolg verliep de scan nu in een veel efficiëntere vorm. Elk quadrant wordt apart behandeld. Er werd van buitenaf naar binnen gescand, tot er een eerste donkere punt werd gedetecteerd. op dit moment wordt dit opgeslagen als verste hoek, en werd dit quadrant verder genegeerd. Deze actie alleen haalde de nodige processor kracht al met een twintigtal percent naar beneden. Verder wordt slechts een scan uitgevoerd als de camera detecteerde dat de kleur van het veld drastisch veranderd was.



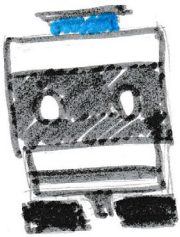
qu







Sr 2.1



- cleaning → FD robot
- finds entrance outside world in night
- cross bridge ?
- first nature
- fight b/g robot ? (safe b/g garbage robot
  - survive
- freedom ↔ open end

HARDWARE EN VORM

INLEIDING

De hardware en de manier van werken is doorheen het project sterk gewijzigd. In het volgende stuk worden de grootste veranderingen besproken. Deze beslissingen komen voort uit input van derden, zoals Zach Lieberman en Beyond.io.

TAFEL

Het aantal kinderen dat tegelijk kon deelnemen heeft sterk gevariëerd. Aan de start van het project was er sprake van een hele klas, of minstens 10 kinderen. Dit veranderde echter naar een tweetal kinderen. Dit was efficiënter en stelde me in staat te focussen op belangrijkere functionaliteiten zoals tags en compileer-mogelijkheden.

TAGS

De allereerste tags waren blokjes plexiglass, uitgesneden met de lasercutter. Om deze blokjes bijeen te houden om het scannen gemakkelijk en accuraat te houden waren magneetjes nodig.
Later zou een uitsparing gemaakt worden in het onderliggend projecteeroppervlakte, waardoor de magneet overbodig werd. Dit zou helaas de flexibiliteit van het aantal tags beperken.

Tijdens Resonate (een conferentie rond multimedia en installaties) in Belgrado), heb ik het genoegen gehad een gesprek te kunnen hebben met Zach Liebermann (grondlegger Openframeworks). Deze raadde me aan in plaats van blokjes over te schakelen naar een kaartspel. Dit heeft een aantal gevolgen:

1. de codes die op het kaartje moeten komen kunnen groter gemaakt worden.
2. de ruimte rond de tag wordt wit, waardoor dit vlotter uit te lezen valt.
3. naargelang de stijl van illustraties kan dit kaartje ook aangepast worden aan de gangbare stijl.
4. het gehele project wordt een stuk gemakkelijker zelf te maken. Mensen met kennis van programmeren kunnen deze tool gebruiken om hun of andere kinderen gemakkelijk en zonder extra middelen te leren programmeren.

WERKVELD

Aan de start van dit project had elk kind een eigen projectiebord. Dit zorgde ervoor dat kinderen zich konden verplaatsen zonder problemen. Dit was een bord ter grootte van een A3 papier. Aan de rechterkant was een uitsparing voor de toenmalig plexiglazen tags. Dit bord bleek echter niet geschikt (veel processorkracht nodig om het beeld te vervormen). Helaas was het moeilijk een statisch en stabiel beeld te genereren op basis van de positie van het projectieoppervlak. De overgang van een plank naar een boek gebeurde tijdens een bespreking met mijnheer Van den Broeck. Die wees mij buiten de functionele waarde op het gemak van lezen in een fysiek boek. Het boek zou groot genoeg moeten zijn om het speelveld (die de programmeeropdracht bevatte) te kunnen herbergen. Tijdens een vergadering met Beyond.io werd onder mijn aandacht gebracht dat dit niet het geval moest zijn. Het zou jammer geweest zijn om niet de volledige projectie te gebruiken en me te beperken tot een klein deel van een pagina. Dit had tevens ook de beeldkwaliteit aangetast. Ik besloot uiteindelijk het volledige speelveld weg te nemen uit het boek, en te werken met een afzonderlijk werkveld dat aangeroepen kan worden door het scannen van een code in het boek.

CONCLUSIE EN VERDERE PLANNEN

“work with technology that exists, not with the promises of tomorrow”

tawny schlieski

Verhalen hebben altijd een grote rol gespeeld in het aanleren van nieuwe technieken. Een narratief zorgt ervoor dat de gebruiker of lezer sneller verbindingen maakt, en dus handelingen en technieken beter onthoud. Dit project gebruikt deze leervorm om programmeren aan te leren aan kinderen, en hen hierbij visueel te ondersteunen. De applicatie leert en helpt het kind programmeerstructuren te begrijpen en toe te passen, op een luchtige en ontspannen manier.

In de toekomst kunnen vele verhalen samenwerken met dit programma, en kan het gebruikt worden in andere omgevingen (klas, leermomenten, externe vormingen). Om dit doel te bereiken is het nodig vele functionaliteiten te automatiseren, denk maar aan het mappen van de output (plaatsen en vervormen van het beeld om overeen te komen met het projectieoppervlak), of het invoeren van programmeeroefeningen. Ook kunnen de tags, die er nu strak, ma fantasieloos uitzien, verrijkt worden met afbeeldingen, om beter bij het verhaal te passen.

"we zijn op een punt in digital storytelling waar we niet goed weten wat te vertellen, maar enthousiast zijn over de mogelijkheden." dit was een situatie die voorkwam in het begin van film, waar mensen filmpjes animeerden waarin geen verhaal verteld wordt, maar vooral de effecten getoond werden."

REFERENTIES

1 http://www.eun.org/c/document_library/get_file?uuid=521cb928-6ec4-4a86-b522-9d8fd5cf60ce&groupId=43887

2 http://www.21stcenturyskills.nl/visie-en-missie-21st-century-skills/

3 http://www.steunpuntdiversiteitenleren.be/nl/themas/krachtige-leeromgeving

4 http://www.klasse.be/tvklasse/20627-Christophe-Lafosse-Het-GeTalenteerde-Brein#.VWoKSYZUeK0

5 Milton Erickson, of recenter: Brein@work, 2010; Reynolds, 2011 http://www.idealearninggroup.com/blog/the-role-of-stories-in-learning

6 http://www.dagjules.be/

7 https://www.youtube.com/watch?v=08rArPC48L4