Software ontwikkeling
en professionele vaardigheden

2de bachelorjaar informatica

Visual Programming IDE: Eindverslag

Groep 5:
Pim Goffinghs (1849269),
Wout Oben (1745605),
Joep Stevens (1848586)

Begeleider
Joris Herbots

Coördinerend verantwoordelijke
prof. dr. Wim Lamotte

Jaar 2019-2020

Inhoudsopgave

1 Executive summary

Voor het vak *Project: software ontwikkeling en professionele vaardigheden* werd de opdracht gegeven om een visuele programmeeromgeving te maken. De opdracht werd gegeven door Anaïs Ools, één van de CoderDojo Hasselt begeleiders. In naam van de CoderDojo, een non-profit beweging die evenementen organiseren waarbij kinderen aangezet worden om te leren programmeren, gaf zij de opdracht om een kindvriendelijke visuele programmeeromgeving te maken. In dit deel wordt een korte samenvatting gegeven over welke features in het project geïmplementeerd zijn.

1.1 De Blokkenlijst en het canvas

De blokkenlijst, te zien aan de linkerkant van de applicatie, bestaat uit verschillende categorieën. Een categorie kan geopend en gesloten worden door op de naam van de categorie te drukken. Onder een open categorie staan de blokken van de categorie. De blokken in een categorie hebben allemaal dezelfde kleur (met categorie 'basis' als uitzondering).

Vanuit de blokkenlijst kunnen blokken gesleept worden op het canvas, dat zich in het midden van de applicatie bevindt. Op het canvas kan een programma gemaakt worden door blokken met elkaar te verbinden. Blokken bevatten de vorm van een puzzelstuk, zo kunnen blokken met elkaar verbonden worden zoals puzzelstukken. Blokken zonder puzzelstuk vorm kunnen in de donkere of witte zones in een blok gezet worden. Dit zijn invoervelden. In een wit invoerveld kan de gebruiker ook zelf iets typen. Wanneer men een blok in een invoerveld probeert te stoppen, moet er wel op gelet worden dat het blok van hetzelfde type is als het invoerveld (een '+'-blok past bijvoorbeeld niet in het invoerveld van een 'als-dan'-blok).

Het canvas is groter dan men op het eerste zicht kan zien. Aan de onderkant en aan de rechterkant van het canvas bevinden zich scrollbars. Met deze scrollbars kan men over het canvas navigeren. Men kan ook over het canvas navigeren door het canvas te verslepen met de linkermuisknop. In de rechterbovenhoek van het canvas staat een overzicht. In het overzicht staan alle blokken, die zich momenteel op het canvas bevinden, in het klein afgebeeld, alsook een rechthoek dat aangeeft welk deel van het canvas er op dit moment in beeld is.

1.2 Actie knoppen

Rechts bovenaan de applicatie staan de actieknoppen. De eerste drie knoppen hebben een werking met betrekking tot het starten en stoppen van programma 's. De eerste knop, is de start-knop en zal de programma 's onder alle 'start'-blokken uitvoeren. De tweede knop, de stop-knop, stopt het uitvoeren van alle programma 's. De derde knop is de stap-knop, deze voert één stap van de programma 's onder de 'start'-blokken uit of stapt verder als het stappen al begonnen is. De vierde knop is de undo-knop, bij een druk op deze knop, zal er een actie ongedaan worden gemaakt. De ongedane actie kan hersteld worden met een druk op de vijfde knop, de redo-knop. Als laatste knop staat er een vertraging-schuifregelaar. Deze regelaar bepaalt de snelheid van het uitvoeren van de programma 's. De maximale vertraging is 2 seconden en de minimale vertraging 0 seconden, dit wilt zeggen dat het resultaat onmiddellijk zichtbaar zal zijn.

1.3 Output

Output is voorzien in 2 vormen: tekstuele output en output met behulp van afbeeldingen. Er is een tekst venster, waar output simpelweg naar geschreven kan worden met behulp van het 'toon tekst'-blok. Ook is er een afbeeldingen venster, waar 5 vooraf ingeladen afbeeldingen gebruikt kunnen worden. Met behulp van de blokken in de afbeeldingen-categorie kunnen een aantal bewerkingen uitgevoerd worden op deze afbeeldingen. Deze bewerkingen zijn: verplaatsen (naar onder, boven, links en rechts), draaien (met een bepaalde hoek uitgedrukt in graden), vergroten en verkleinen van een afbeelding. De afbeeldingen op het afbeeldingen venster kunnen zelf ook manueel verplaatst worden, door deze te verslepen.

1.4 Variabelen

Rechts onderaan de applicatie bevindt zich een venster waar men variabelen kan aanmaken. Dit doet men door een naam en een waarde te geven. In dit venster krijgt men ook een overzicht van alle bestaande variabelen en hun waardes te zien. 'Variabele'-blokken kunnen gebruikt worden in invoervelden. De waarde van een variabele kan tijdens het uitvoeren van een programma aangepast worden met behulp van het 'zet variabele gelijk aan'-blok.

Men kan een variabele verwijderen door op het kruis langs de variabele in het overzicht te drukken. Een variabele kan enkel verwijderd worden wanneer er geen 'variabele'-blokken van deze variabele meer op het canvas te vinden zijn. Wanneer men een variabele toch probeert te verwijderen, zullen de 'variabele'-blokken, die eerst verwijderd moeten worden, oplichten en zal er een waarschuwing gegeven worden.

1.5 Fout, waarschuwing en succes meldingen

Tijdens het uitvoeren van blokken, of tijdens het verbinden van blokken kan er een fout of waarschuwing melding gegeven worden. Blokken in de categorie 'API' kunnen ook een succes melding geven.

Deze meldingen zullen op een aantal manieren zichtbaar zijn. Er zal een pop-up verschijnen in de linkerhoek van het canvas, deze pop-up verdwijnt vanzelf na een paar seconden. De melding zal ook zichtbaar zijn op een blok. Er zal een icoon (afhankelijk van de melding) langs het blok verschijnen. Meer informatie over de melding kan gekregen worden door met de muis over het icoon te bewegen. Het icoon zal pas weg gaan wanneer het probleem, dat de melding veroorzaakte, opgelost is. Succes meldingen gaan automatisch na een paar seconden weg.

De fout en waarschuwing meldingen zullen ook zichtbaar zijn in het fout en waarschuwing overzicht. Dit overzicht kan men centraal bovenaan het canvas terugvinden. De iconen voor fout en waarschuwing meldingen zijn hier zichtbaar met een nummer dat aangeeft hoeveel fouten en waarschuwingen zich op het canvas bevinden. Als er op dit overzicht gedrukt wordt, zal er een lijst van alle fout en waarschuwing meldingen tevoorschijn komen.

1.6 Opslaan, inladen en delen van projecten met gebruiker accounts

In de header van de applicatie staan een aantal knoppen om projecten op te slaan, in te laden en te delen. Als men projecten wilt opslaan of eerder opgeslagen projecten wilt inladen, moet men ingelogd zijn. Dit kan simpel door rechts in de header op inloggen (of registreren) te duwen. Het aanmaken van een account vereist enkel een unieke gebruikersnaam en wachtwoord (dit wachtwoord kan in de vorm van een patroon of een standaard tekstueel wachtwoord gekozen worden). Het hebben van een account is niet verplicht om de applicatie te kunnen gebruiken. Eens ingelogd kan men projecten opslaan en inladen door op de bijhorende knop te duwen.

Het delen van projecten vereist geen account. Door op 'Genereer code' in de pop-up, voortgebracht wanneer men op de 'Delen'-knop drukt, te duwen, zal er een code gegenereerd worden. Bij het menu waar men projecten kan inladen, krijgt men de keuze om vooraf opgeslagen projecten in te laden (indien ingelogd) of om een project in te laden met behulp van een code. Wanneer men hier de gegenereerde code gebruikt, zal het gedeelde project ingeladen worden.

1.7 Lessen

Bovenaan in de header is er een knop 'Lessen' zichtbaar. Wanneer men hier op klikt, opent zich een menu met alle beschikbare lessen. Om een les te starten, moet men op de naam van de gewenste les klikken.

Een les werkt aan de hand van stappen die elk een actie vereisen. In een stap zal er een pagina element worden aangeduid, zodat de gebruiker weet waar er gekeken moet worden. Zo is het altijd duidelijk wat de gebruiker nu precies moeten doen. Ook is er bij iedere stap een tekstballon aanwezig waarin beschreven staat wat de gebruiker nu juist moet doen. Wanneer de laatste stap bereikt wordt, wordt de gebruiker gefeliciteerd en worden andere lessen aangeraden. Een les eerder afsluiten is altijd mogelijk door op 'stoppen' te duwen.

Lessen kunnen toegevoegd en verwijderd worden met een beheerder-account. Het beheerder-account krijgt in de header een extra knop te zien waar hij/zij zelf lessen kan aanmaken. Beheerders kunnen ook reeds bestaande lessen aanpassen. Wanneer een beheerder in de lessen lijst gaat kijken, staat er langs de naam van iedere les een wijzig knop, waarmee een les aangepast kan worden, en een 'X' voor een les te verwijderen.

2 Inleiding

De CoderDojo is een non-profit beweging die evenementen organiseren waarbij kinderen aangezet worden om te leren programmeren. Anaïs Ools, één van de Hasselt CoderDojo begeleiders, heeft ons de opdracht gegeven om een 'Visual Programming IDE' te creëren. Meer specifiek, een IDE waarbij kinderen van 8 tot 12 jaar op een interactieve manier de denkwijzen en de logica achter het programmeren kunnen aanleren.

Doordat er maar een gelimiteerd aantal laptops beschikbaar zijn op de evenementen van de CoderDojo, en omdat de kinderen niet altijd dezelfde laptop terug krijgen, moest het project web-based worden. Zo kunnen kinderen vanop elke laptop de applicatie gebruiken. Dit hele systeem zal draaien op een web-server die de CoderDojo zelf voorziet op hun evenement, hierdoor zal de applicatie enkel beschikbaar zijn op het evenement van de CoderDojo.

De logica achter het programmeren wordt aangeleerd door middel van blokken te slepen en te verbinden op een semi-oneindig canvas. Door de jonge leeftijd van het doelpubliek is een simpel, maar aantrekkelijke en uitnodigende lay-out dus zeker een must voor het programma. Dit proberen wij te creëren door de plaatsen waar blokken geconnecteerd kunnen woorden voor te stellen als puzzelstukken die in elkaar passen. Zo zal visueel ook duidelijk zijn welke blokken connecteerbaar zijn met elkaar en welke niet.

Ieder blok heeft zijn eigen betekenis in de programmeer-syntax. De volgende blokken werden expliciet gevraagd te implementeren:

- Lussen: for- en while-loops
- Conditionele operaties: if en else-blokken
- Logica operatoren: && (AND), || (OR), ! (NOT)
- Wiskundige operatoren: +, -, *, /, ==, ¡, ¿
- String operaties: Lengte van een string berekenen, strings samenvoegen...
- Variabelen: Een blok waaraan men een waarde van type string, boolean of integer aan kan koppelen.
- Functies: Een blok waarmee men stukken code kan groeperen, deze zal men dan meerdere malen kunnen oproepen in de code.

De variabelen en functies werden door de opdrachtgever als extra 's gezien en waren van minder belang.

In het user-interface zal zich een menu bevinden waar de gebruikers blokken uit een lijst kunnen kiezen. Hieruit kunnen de blokken vervolgens gesleept worden om deze op het canvas te plaatsen. Blokken moesten makkelijk terug te vinden zijn in de blokkenlijst, daarom zal de blokkenlijst opgedeeld worden in categorieën. Om deze reden bevat de eerste categorie ook veelgebruikte blokken, zo kan de gebruiker deze makkelijk terugvinden (denk aan de start-blok, een for-loop, een if-test, etc...). Een zelf gemaakt programma moest kunnen starten met een start-knop. Hiervoor zal een start-blok toegevoegd worden dat aangeeft waar een programma zal starten. Met een stop-knop kan het programma eerder stoppen. De output van een programma zal in een apart venster gebeuren. De opdrachtgever vond tekstuele output belangrijk, naar aanrading van het onderwijsteam is output via afbeeldingen ook toegevoegd.

Wanneer men een programma heeft gemaakt, kan het project opgeslagen en ingeladen worden met behulp van een account. Het aanmaken van een account zal niet verplicht zijn, enkel als men nieuwe projecten wilt opslaan of wanneer men wilt verder werken aan oudere projecten. Ook moesten projecten gemakkelijk gedeeld kunnen worden met anderen. Er zal de mogelijkheid zijn om een code van een project te creëren. Anderen kunnen deze code dan ingeven om een kopie van het project te openen. Voor het delen en inladen van gedeelde projecten hoeft men niet in te loggen.

Doordat het canvas groter moet zijn dan het gedeelte dat zichtbaar is, kunnen kinderen wel eens blokken verliezen. Om dit probleem op te lossen, is er een klein venster voorzien waar een overzicht van het canvas te zien is.

Er werd ook gevraagd naar API-ondersteuning. Er worden blokken aangemaakt voor een aantal HTTP request methods, denk aan GET, PUT, POST, etc... Deze blokken kunnen dan gemakkelijk gebruikt worden in samenwerking met een RESTful API die kan opgezet worden door de CoderDojo.

Doordat kinderen van deze leeftijd vaak nog geen enkele ervaring hebben met programmeren, zijn er ook lessen geïmplementeerd. Deze leggen uit hoe onderdelen van de applicatie werken en zullen de gebruiker helpen programma 's te maken.

3 Functionaliteit

In deze sectie zullen alle onderdelen van de applicatie in detail worden uitgelegd. Er zal besproken worden hoe ieder onderdeel er uit zal zien voor de gebruiker, alsook de achterliggende werking van het onderdeel.

3.1 Blokken

Als eerste onderdeel zullen de blokken besproken worden. In deze sectie gaan de verschillende soorten blokken alsook de componenten waaruit deze bestaan, besproken worden.

3.1.1 Blok componenten

Een blok bestaat uit een aantal componenten. Bij het tekenen van een blok wordt er een basis rechthoek getekend waar verschillende componenten aan toegevoegd worden. Elk component heeft een uniek uiterlijk. Componenten kunnen in 3 klassen worden opgedeeld: top, base, en bottom componenten.

Top componenten Deze componenten worden bovenaan het basis rechthoek getekend. Blokken kunnen de volgende top componenten hebben:

Start component Dit component tekent een halve ovaal boven het blok en wordt gebruikt om de juiste blokken te vinden wanneer er een actie wordt uitgevoerd dat een programma kan starten.

Connect component Dit component geeft aan dat het blok aan een ander blok zijn attach of enclose component kan verbonden worden. Aan de bovenkant van het blok zal een halve cirkel ontbreken met dezelfde grootte als het uitsteeksel van de attach en enclose componenten.

API-connect component Een component met dezelfde werking als het connect component, maar werkt enkel voor API-attach en API-enclose componenten. Het heeft dan ook een ander uiterlijk dan het normale connect component. Bovenaan het blok zal er een driehoek ontbreken.

Base componenten Deze componenten worden van links naar rechts in de basis rechthoek getekend. De volgende componenten zijn base componenten:

Label component Dit component plaatst een gekozen tekst in het blok.

Input component Maakt een donker veld in het blok. In dit veld kunnen blokken met een return component geplaatst worden. Er is ook een optie om dit veld wit te maken. In het witte veld kan de gebruiker zelf een waarde typen. Het input component kan een bepaald type, bijvoorbeeld string, hebben en blokken weigeren wanneer hun return component het verkeerde type heeft.

Return component Dit component geeft aan dat het blok in een input component geplaatst kan worden. Dit component kan een bepaald type hebben, zodat het enkel in input componenten kan gezet worden met hetzelfde type. Dit component verandert niets aan het uiterlijk van het blok.

Comment component Een component dat een breed en lang tekstvak plaatst in het blok. Hier kunnen langere zinnen in getypt worden.

Choice component Plaatst een drop-down menu in het blok met een aantal opties. Dit component wordt gebruikt om afbeeldingen te kiezen voor de afbeeldingsblokken en om een toets te kiezen bij het 'start op druk van toets'-blok.

Break component Wanneer er een break component in een blok wordt geplaatst, worden alle volgende basis componenten op een nieuwe rij eronder gezet. Dit component dient enkel voor de opmaak van de base componenten aantrekkelijker te maken.

Button component Dit component plaatst een button in het blok waar de gebruiker op kan klikken. De actie die uitgevoerd wordt bij het drukken van de knop hangt af van blok tot blok.

Error component Alle blokken bevatten het error component. Dit is altijd het laatst component dat wordt toegevoegd aan het blok. Dit component is zichtbaar wanneer het blok een fout, waarschuwing of succes melding heeft en geeft het bijhorend icoon langs het blok. Als de gebruiker over dit icoon hovert, zal er meer info over de melding gegeven worden.

Bottom componenten Deze componenten worden onderaan het basis rechthoek getekend. Blokken hebben de volgende opties voor bottom componenten:

Attach component Voegt onderaan het blok een halve cirkel toe waaraan andere blokken met een connect component mee verbonden kunnen worden.

Enclose component Dit component voegt een plaats onderaan het blok toe waar andere blokken met een connect component in geplaatst kunnen worden. De blokken worden dan omsloten door dit component.

API-attach component Werkt hetzelfde als het attach component, alleen voor API-connect blokken en de halve cirkel krijgt de vorm van een driehoek.

API-enclose component Werkt hetzelfde als het enclose component, alleen voor API-connect blokken en de halve cirkel krijgt de vorm van een driehoek.

3.1.2 Blokken verslepen

Het canvas, de blokkenlijst en het variabelen venster hebben niet hetzelfde SVG-element, hierdoor kunnen blokken standaard niet versleept worden van een element naar een ander. Om toch blokken tussen deze elementen te kunnen slepen, wordt er gewerkt met een doorzichtig SVG-element dat de muis volgt. Wanneer er op een blok wordt gedrukt, wordt dit blok verplaatst naar dit doorzichtig SVG-element en zal het blok de muis volgen. Als de muis weer los wordt gelaten over het canvas zal dit blok uit het SVG-element worden gehaald en op het SVG-element van het canvas geplaatst worden. In de eerste versie van het eindproduct werkte het vastnemen van blokken niet optimaal. Wanneer men een component vastnam, verschoof de positie van de cursor naar de linkerbovenhoek van het blok. Bij het verslepen van blokken wordt er nu correct de offset van de muis berekend waardoor er geen 'glitchy' gedrag meer voorkomt.

3.1.3 Connecteren van blokken

In het originele eindproduct was er enkel de mogelijkheid om blokken onderaan of in andere blokken te verbinden (met behulp van het attach en enclose component). Nu is er ook de mogelijkheid om blokken langs de bovenkant te connecteren (met behulp van het connect component).

Het connecteren van blokken werd in het originele eindproduct gedaan met behulp van de positie van de muiscursor. Wanneer men een blok versleepte en de muis binnen het bereik van, bijvoorbeeld, een attach component kwam, lichtte het silhouet van het component op. Dit is aangepast zodat dit niet meer via de positie van de muis gebeurt, maar via de positie van de attach en connect componenten van het blok dat versleept wordt. Zo kan men blokken connecteren met behulp van de puzzelstuk vormen, dit voelt intuïtiever aan voor de gebruiker.

Ook lichtte het silhouet van de componenten origineel op zodra de muiscursor over het component bewoog. Dit werd aangepast zodat het silhouet enkel zichtbaar wordt wanneer men wel degelijk een blok versleept die men op die plaats kan verbinden.

3.1.4 Functie-blokken

Functie-blokken staan onder de categorie 'Functies'. Wanneer een functie blok voor de eerste keer op het canvas wordt geplaatst, wordt er aan de gebruiker gevraagd om de functie een naam te geven in een pop-up dat tevoorschijn komt. Deze pop-up sluit pas wanneer de functie een naam gekregen heeft. Op het functie-blok bevindt zich een knop met de tekst 'Verander naam'. Op deze knop kan gedrukt worden om de functie te hernoemen. De pop-up die dan verschijnt kan wel gesloten worden zonder de functie een nieuwe naam te geven. De oude naam zal in dit geval behouden worden.

Een functie-blok heeft een enclose component. Hier moeten de blokken in geplaatst worden die uitgevoerd zullen worden bij het aanroepen van de functie. De functie kan aangeroepen worden met het functie-aanroep-blok. Dit blok bevindt zich altijd aan het attach component van het functie-blok. Hier kan een aanroep-blok losgemaakt worden. Wanneer een dergelijk blok losgemaakt wordt, zal er direct een nieuw blok verschijnen. zo kunnen een onbeperkt aantal aanroep-blokken aangemaakt worden. Het aanroep-blok zal altijd verwijzen naar het functie-blok waarvan het losgemaakt is. De naam van de functie staat ook vermeld in het functie-aanroep-blok, zo is het altijd duidelijk tot welke functie het aanroep-blok behoort. Als de functie van naam verandert, verandert de naam van ieder aanroep-blok van deze functie ook automatisch. Om een functie op te roepen met behulp van een functie-aanroep-blok, kan men een dergelijk blok in een programma stoppen. Wanneer het blok wordt tegengekomen tijdens het uitvoeren, zal de bijhorende functie uitgevoerd worden. Nadat de functie is uitgevoerd, zal het uitvoeren van het programma verder gaan onder het functie-aanroep-blok dat de functie opgeroepen had.

3.1.5 'Variabele'-blokken

'Variabele'-blokken zijn terug te vinden in het variabelen venster (zie sectie 3.8). Deze blokken geven de waarde van een bepaald variabele terug bij uitvoering van het blok. De waarde van de variabele kan aangepast worden tijdens de uitvoering van een programma met behulp van een 'Zet

variabele gelijk aan'-blok. Dit blok is te vinden onder de categorie 'waardes' en verwacht als input het variabele blok en de nieuwe waarde die aan de variabele toegekend zal worden. Wanneer dit blok wordt uitgevoerd, zal de waarde van de variabele veranderen en zal het 'variabele'-blok bij volgende uitvoering de nieuwe waarde terug geven.

3.1.6 API-blokken

API-blokken zijn blokken waarmee men HTTP-request methods kan uitvoeren. Om de blokken iets duidelijker te maken voor kinderen, staat er in het nederlands vermeld wat het blok doet met de naam van de method vermeld. Bij het GET-block staat bijvoorbeeld 'Haal data op (GET)' vermeld. Er zijn blokken voorzien voor de GET, POST, PUT, PATCH en DELETE methods. Ieder blok vraagt een URL naar een webserver om een request op uit te voeren. Het 'GET'-blok geeft ook nog de optie om de naam van een veld mee te geven, zo kan men zorgen dat men 1 specifieke waarde verkrijgt. Wanneer men deze functionaliteit wilt gebruiken, verwacht het 'GET'-blok dat er maar 1 object teruggegeven wordt door de webserver.

Bij POST, PUT en PATCH krijgt men de mogelijkheid/moet men data meegeven in de vorm van een JSON-object. Dit doet men door JSON-parameter-blokken aan het blok toe te voegen. JSON-parameter-blokken zijn blokken die een veld en een waarde vragen. Alle JSON-parameter-blokken die toegevoegd zijn aan het API-enclosure component van het blok, worden tijdens de uitvoering van het blok omgezet tot 1 JSON-object en meegegeven met het HTTP-request. Bij een mislukt HTTP-request zal een error verschijnen in de vorm van een fout-melding, wanneer een HTTP-request gelukt is, zal een succes-melding zichtbaar worden (Voor meer info over fout- en succes-meldingen, zie sectie meldingen 3.4).

3.2 Blokkenlijst

Links in de applicatie staat de blokkenlijst. De blokkenlijst bestaat uit een aantal categorieën. Elke categorie kan geopend of gesloten worden door op de naam van de categorie te drukken. De eerste categorie (basis) staat bij het inladen van de pagina altijd open, de andere categorieën zijn oorspronkelijk gesloten. Onder de naam van een open categorie zullen de blokken van de categorie verschijnen. De blokken in een categorie (met de blokken in categorie 'basis' als uitzondering) hebben allemaal dezelfde kleur, namelijk de kleur van de categorie. Deze staat ook weergegeven langs de naam van de categorie.

De blokkenlijst is opgedeeld in de volgende categorieën:

Basis Deze categorie is een verzameling van blokken uit andere categorieën. Deze blokken zal de gebruiker vaker nodig hebben. Omdat deze categorie bovenaan de lijst staat, kan de gebruiker deze blokken gemakkelijk terugvinden. De kleur van deze categorie is wit. Omdat de blokken uit andere categorieën komen, zullen de blokken in de categorie echter niet wit zijn.

Lussen Deze categorie bevat blokken die andere blokken in hun enclosecomponent(s) meerdere keren kunnen uitvoeren. Het aantal keer dat de blokken worden uitgevoerd is afhankelijk van de input meegegeven aan het blok. De blokken in de categorie hebben een gele kleur.

Testen Deze donkerblauwe blokken laten programma 's toe om blokken uit te voeren afhankelijk van een bepaalde input. Een voorbeeld van een blok in de categorie is het 'als-dan'-blok, dat achterliggend als een if-test werkt.

Rekenen Een categorie voor blokken die bewerkingen uitvoeren met betrekking tot getallen, zoals optellen, aftrekken, vermenigvuldigen en delen. Deze blokken hebben een lichtblauwe kleur.

Tekst In deze categorie staan oranje blokken die bewerkingen uitvoeren op tekst. Ook staat het 'toon tekst'-blok, die tekst in het tekst venster laat zien, tussen deze categorie.

Waardes De blokken in deze donkergroene categorie zullen een vaste waarde teruggeven, zoals bijvoorbeeld 'waar' of 'onwaar'. Ook blokken die bewerkingen op variabelen doen, zoals het 'Zet variabele gelijk aan'-blok, staan in deze categorie. Ook zullen 'variabele'-blokken voor alle aangemaakte variabelen in deze categorie tevoorschijn komen.

Afbeeldingen Een categorie voor alle roze blokken die bewerkingen uitvoeren op de afbeeldingen uit het afbeeldingen venster.

Functies Dit is een categorie voor functie-blokken. Deze blokken hebben een paarse kleur.

API Alle API-blokken bevinden zicht in deze categorie, in een rode kleur.

Wanneer de blokkenlijst te groot wordt, doordat er meerdere categorieën openstaan, zal er een scrollbar verschijnen. De lijst kan verschoven worden door de scrollbar te bewegen of door met de muis te scrollen. Om aan te duiden hoeveel er ongeveer gescrold kan worden, zal de grootte van de scrollbar aangepast worden naar de grootte van de blokkenlijst.

3.3 Canvas

In het midden van de applicatie staat het canvas. Dit wordt gebruikt om programma's op te creëren door gebruik te maken van de verschillende blokken die beschikbaar zijn in de blokkenlijst. De gebruiker kan blokken op het canvas plaatsen door een blok te slepen vanuit de blokkenlijst tot op het canvas. Eens het blok op het canvas geplaatst is, kan de gebruiker het blok opnieuw van positie veranderen door het blok te verslepen.

Het canvas is groter dan men op het eerste zicht kan zien. Aan de onderkant en aan de rechterkant van het canvas, bevinden zich scrollbars. Met deze scrollbars kan men over het canvas navigeren. Men kan ook over het canvas navigeren door het canvas te verslepen met de linkermuisknop.

Er is ook ten allen tijde een overzicht aanwezig in de rechterbovenhoek van het canvas. Dit is een klein vierkant dat het hele canvas representeert. Op het overzicht zijn kleinere versies van alle blokken op het canvas zichtbaar. Deze staan gepositioneerd op het overzicht zoals ze op het werkelijke canvas gepositioneerd staan. Ook is er een kleiner vierkant zichtbaar op het overzicht, dit vierkant representeert het gedeelte van het canvas zichtbaar voor de gebruiker. Het overzicht zal onmiddellijk updaten bij een verandering van blokken op het canvas of bij een verschuiving van het zichtbare veld.

3.4 Fout, waarschuwing en succes meldingen

Wanneer er een onlogische of foute actie wordt uitgevoerd, worden er op een aantal plaatsen fout of waarschuwing meldingen getoond. Bij succes van het uitvoeren van een API-blok zal er ook een succes melding komen. In deze sectie worden de 3 plaatsen besproken waar de fout, waarschuwing en succes meldingen te zien zijn: het error component, de error pop-up en het fout en waarschuwing overzicht. Voor meer informatie over de mogelijke fout, waarschuwing en succes meldingen, raadpleeg de handleiding (zie sectie 8).

3.4.1 Error component

Tijdens het uitvoeren van een programma of bij een verkeerde actie op een blok kan er een fout of waarschuwing in het blok komen. Met behulp van het error component wordt deze fout melding duidelijk gemaakt aan de gebruiker. Dit component voegt bij een fout of waarschuwing een icoon toe aan het blok. Fouten en waarschuwingen hebben een verschillend icoon. Dit component heeft ook een verschillend icoon voor een succes melding, dat getoond zal worden bij een succesvol request van de API-blokken.

Wanneer de gebruiker zijn muis over het icoon beweegt, krijgt de gebruiker meer uitleg over de fout/waarschuwing/succes melding. De succes melding zal na een paar seconden automatisch

weggaan. De fout en waarschuwing melding gaat pas weg wanneer het probleem, dat de melding veroorzaakte, opgelost is.

3.4.2 Error pop-up

Voor elke fout, waarschuwing of succes melding zal er een pop-up tevoorschijn komen. Een rode pop-up voor fouten, een gele voor waarschuwingen en een groene voor succes meldingen. Deze pop-up zal een paar seconden in de linkerbovenhoek van het canvas zichtbaar blijven, voordat het weer verdwijnt.

3.4.3 Fout en waarschuwing overzicht

Centraal bovenaan het canvas staat het fout en het waarschuwing icoon met een nummer langs elk icoon. Dit is het fout en waarschuwing overzicht. De getallen naast de iconen geven aan hoeveel onopgeloste fouten en waarschuwingen zich op het canvas bevinden. De gebruiker kan ook op dit overzicht drukken om een lijst te verkrijgen van alle fouten en waarschuwingen. Deze lijst wordt weer gesloten wanneer men op het overzicht drukt.

3.5 Actieknoppen

Rechts bovenaan de applicatie staan een aantal knoppen. Een aantal knoppen hebben een werking met betrekking tot het uitvoeren van programma 's, anderen met betrekking tot het ongedaan maken van acties. In deze sectie zullen deze knoppen en hun achterliggende werking in detail besproken worden.

3.5.1 Uitvoeringsknoppen

De volgende knoppen hebben een werking met betrekking tot de uitvoering van een programma:

Start-knop De meest linkse knop is de start-knop. Wanneer er op de start-knop wordt gedrukt, zal iedere seconde (deze tijd is verstelbaar) de stap-knop achterliggend worden ingedrukt. Het uitvoeren zal stoppen wanneer alle programma 's afgelopen zijn. Dit kan ook eerder gestopt worden door op de stop-knop te drukken.

Stop-knop Rechts naast de start-knop staat een rode acht-hoek. Dit is de stop-knop. De stop-knop zal de uitvoering van alle programma 's stoppen. Alle actieve blokken zullen inactief worden gezet en niet meer oplichten.

Stap-knop De stap-knop bevindt zich rechts naast de stop-knop en wordt gerepresenteerd door een blauwe driehoek met een streep aan de linkerzijde. Wanneer er voor de eerste keer op de stap-knop wordt gedrukt, zullen alle start-blokken actief worden en oplichten. Bij een volgende druk zullen de blokken onder deze start-blokken uitgevoerd worden, vervolgens de blokken onder deze blokken... Dit zal zich blijven herhalen totdat er geen volgend blok meer is. Door op de stop-knop te drukken worden alle blokken weer inactief en zal de volgende stap weer alle start-blokken actief maken.

Vertraging-schuifregelaar Deze schuifregelaar staat rechts van de actieknoppen. Met deze regelaar kan de gebruiker de vertraging van de start-knop aanpassen. De vertraging zal standaard op 1 seconde staan en kan veranderd worden met stappen van 0.25 seconden. Naar links verschuiven vermindert de vertraging tot een minimum van 0 seconden. Naar rechts verschuiven, vermeerdert de vertraging tot een maximum van 2 seconden.

3.5.2 Werking van de stap-knop

Achterliggend werkt de stap-knop met behulp van een stack van blokken gekoppeld met een functie. Deze sectie zal de werking van de stap-functionaliteit uitleggen.

Eerste stap Wanneer er voor de eerste keer op de stap-knop wordt gedrukt en er dus nog geen stacks aangemaakt zijn, zal voor ieder start-blok op het canvas een stack aangemaakt worden. Ieder start-blok samen met een functie, dat het einde van een programma aangeeft, zal op zijn stack geplaatst worden. Alle start-blokken zullen vervolgens actief gemaakt worden. Een blok 'actief maken' wilt zeggen dat het blok zal oplichten, zodat de gebruiker weet welk blok uitgevoerd wordt.

Volgende stappen Wanneer men vervolgens op de stap-knop drukt, zal telkens de execute-functie van het blok bovenaan iedere stack uitgevoerd worden. De execute-functie van een blok kan een nieuw blok met een functie, die de resultaten van de uitvoering van het blok gebruikt, of een null-waarde met een lege functie toevoegen aan de stack. Het toegevoegde blok bovenaan de stack zal vervolgens actief gemaakt worden. In deze bijhorende functie van een blok kunnen ook opnieuw blokken toegevoegd worden aan de stack. Een 'als-dan'-blok zal bijvoorbeeld eerst het blok in zijn input component, samen met een functie toevoegen aan de stack. Deze functie zal het resultaat van de uitvoering van het blok gebruiken om te bepalen of dan het blok in zijn enlcose of zijn attach component op de stack wordt gezet.

Uitvoering van blok is klaar Wanneer er geen volgend blok opgehaald moet worden, zal er een null-waarde, samen met een lege functie, aan de stack toegevoegd worden. Wanneer men een volgend blok actief wilt maken en men komt de null-waarde tegen, zullen de null-waarde en de lege functie van de stack afgehaald worden. Vervolgens zal de bijhorende functie van het blok bovenaan de stack uitgevoerd worden. Na afloop van de uitvoering van deze functie, zullen de functie en het bijhorend blok op hun beurt van de stack afgehaald worden. Er zal dan onmiddellijk nog een keer gestapt worden.

Einde van de uitvoering Het programma zal bij iedere druk op de stap-knop iets toevoegen aan de stack, een functie uitvoeren, of iets verwijderen van de stack. Op het einde van een programma zal men in de stack terug bij de uitvoering van het start-blok uitkomen. De functie die bij het start-blok hoort, zal het einde van het programma aangeven.

Lege stack Wanneer een stack leeg is, wordt de stack weer verwijderd. Wanneer alle stacks dus verwijderd zijn, zal de applicatie weten dat er geen programma 's meer uitgevoerd worden en bij een volgende druk op de stap-knop zal dus ook terug de eerste stap uitgevoerd worden.

3.5.3 Undo- en redo-knop

De undo- en redo-knoppen zien eruit als een gedraaide pijl tegen de klok in en een gedraaide pijl dat met de klok mee draait. Wanneer er op de undo-knop wordt gedrukt, zal de laatste actie ongedaan worden gemaakt. De redo-knop zal ongedane acties weer herstellen. Een ongedane actie kan niet meer hersteld worden wanneer er reeds een nieuwe actie is gebeurd.

De volgende acties kunnen ongedaan gemaakt worden:

- Blok op het canvas plaatsen.
- Blok verwijderen van het canvas.
- Blok aan een ander blok connecteren of het losmaken van een blok.
- Blok in een input component plaatsen of deze eruit halen.
- Tekst schrijven in een input component.
- Variabelen aanmaken en verwijderen.

3.5.4 Werking van undo en redo

Undo en redo werkt met behulp van een stack. Wanneer één van de ondersteunde acties wordt uitgevoerd, zal de omgekeerde actie aan de undo-stack worden toegevoegd en wordt de redo-stack leeggemaakt. Wanneer er op één van de knoppen wordt gedrukt, zal de bovenste actie op zijn stack worden uitgevoerd. Het omgekeerde van deze actie wordt dan opnieuw op de stack van de andere knop geplaatst.

3.6 Afbeeldingen venster

Aan de rechterzijde van de applicatie staat het afbeeldingen venster. Hier zal bij het inladen van de applicatie 1 afbeelding in staan. Met behulp van selectie vakjes onderaan het afbeeldingen venster kunnen meerdere afbeeldingen aan dit venster toegevoegd of verwijderd worden.

De afbeeldingen kunnen manueel of met behulp van blokken gemanipuleerd worden. Met behulp van de muis kan men afbeeldingen manueel verplaatsen door deze te verslepen. Er zijn een aantal blokken die tijdens hun uitvoering deze afbeeldingen kunnen verplaatsen, vergroten, verkleinen of draaien. Al deze blokken staan onder de categorie 'Afbeeldingen' in de blokkenlijst. Ieder van deze blokken bezit een drop-down menu. Dit drop-down menu bevat de namen van alle afbeeldingen in het afbeeldingen venster, hiermee selecteert men de naam van de afbeelding die men wilt manipuleren. De namen in dit menu veranderen automatisch wanneer een afbeelding verwijderd of toegevoegd wordt van/aan het venster. Er is ook een blok dat een programma start wanneer er op een bepaalde afbeelding gedrukt wordt.

Linksboven op het afbeeldingen venster staat een knop 'herstel'. Deze knop zal alle afbeeldingen op hun originele plaats, grootte en rotatie zetten.

De voorziene afbeeldingen kunnen achteraf makkelijk gewijzigd worden, doordat deze in een images.json file geplaatst worden. In deze file staan links naar afbeeldingen gepaard met hun naam.

3.7 Tekst venster

Het tekst venster staat aan de rechterkant van de applicatie, tussen het afbeeldingen venster en het variabelen venster. Wanneer er tijdens het uitvoeren van een programma een 'toon tekst'-blok wordt tegengekomen, zal de input van het blok in dit venster als een bericht verschijnen. Alle berichten worden onder elkaar opgelijst, nieuwe berichten worden onderaan toegevoegd. Wanneer het bericht niet meer in het venster past, zal er er een scrollbar verschijnen. Bij elk nieuw bericht zal deze scrollbar automatisch naar beneden verschuiven om het nieuwe bericht in beeld te krijgen.

Het tekst venster heeft ook een 'Herstel'-knop. Deze knop zal het tekst venster weer leeg maken. Het tekst venster zal ook leeg gemaakt worden bij het inladen van een project of het herstellen van het canvas.

3.8 Variabelen venster

Variabelen kunnen aangemaakt worden in het variabelen venster dat zich rechts beneden de applicatie bevindt. Een variabele heeft een naam nodig. Er kan ook een waarde gegeven worden. Wanneer er dan op de 'voeg variabele toe' knop wordt gedrukt, verschijnt de variabele als blok in de lijst eronder. Als de variabele al bestond, wordt de waarde geüpdatet. Er zullen dus nooit 2 variabelen met dezelfde naam bestaan. Een 'variabele'-blok kan uit de variabelen lijst gehaald worden en op het canvas geplaatst worden. Deze variabele kan dan gebruikt worden als input voor andere blokken. Van iedere variabele kan men ook een 'variabele'-blok in de blokkenlijst terugvinden onder de categorie 'waardes'.

Als de gebruiker de variabele niet meer nodig heeft, kan deze verwijderd worden uit het variabelen venster. Naast het blok in het venster staat een kruis. Met een druk op dit kruis, kan de variabele verwijderd worden. De variabele kan niet verwijderd worden wanneer er nog instanties van 'variabele'-blokken van deze variabele op het canvas staan. In dat geval zal de gebruiker een waarschuwing melding krijgen en zullen alle 'variabele'-blokken, die eerst verwijderd moeten worden, tijdelijk oplichten zodat de blokken makkelijker te vinden zijn.

3.9 Lessen

Met behulp van lessen kan de gebruiker uitleg krijgen over onderdelen van de applicatie en de werking van blokken en kan men programma 's leren maken. Wanneer de pagina voor de eerste keer wordt ingeladen, zal de optie gegeven worden om een korte rondleiding te krijgen. Deze rondleiding wordt uitgevoerd in de vorm van een les. Deze les kan men achteraf samen met alle andere lessen ook onder de knop 'lessen', in de header, terugvinden. Bij een druk op deze knop wordt een lijst van alle lessen weergegeven. Lessen kunnen gestart worden door op de naam te klikken. Een les zal altijd starten op een nieuw project met een leeg canvas.

Een les bestaat uit een aantal stappen. Iedere stap heeft een tekstballon waarin men meer uitleg over de stap terug kan vinden. In deze tekstballon staat ook altijd een 'stoppen'-knop, waarmee men de les de stoppen. Er kan ook een 'volgende'-knop in dit tekstballon staan wanneer er geen actie vereist is voor naar de volgende stap over te gaan. Wanneer er wel een actie vereist is, wordt er een onderdeel van de pagina aangeduid en zal de volgende stap pas verschijnen wanneer de verwachte actie uitgevoerd is. Vanaf de 2de stap, verschijnt er rechtsboven een knop 'Voltooide stappen'. Deze knop bevat een lijst met alle uitleg van de vorige stappen, zo kan de gebruiker altijd bekijken wat er in een vorige stap gebeurde. Wanneer de laatste stap van een les voltooid is, zal er een pop-up verschijnen die de gebruiker feliciteert en doorverwijst naar andere lessen.

3.9.1 Werking van lessen

Lessen worden achterliggend ingeladen en opgeslagen als een string. Wanneer een les gestart wordt, worden de stappen uit deze string gehaald. Iedere stap kan tekst voor de tekstballon, een aangeduid element, een actie die de stap voltooid en de optie om een 'volgende'-knop te laten zien in de hint bevatten. Een stap is enkel verplicht om tekst voor het tekstballon te bevatten.

3.9.2 Aanmaken van lessen

Wanneer een beheerder inlogt, zal er in de header een 'Maak les'-knop verschijnen. Deze knop geeft de beheerder een interface dat toelaat een les te maken.

Bovenaan het interface zal men de naam van de les moeten ingeven. Daar onder zullen de stappen gemaakt worden. Aan de linkerkant kan er een stap worden aangemaakt. Er wordt een bericht voor het tekstballon verwacht en het aan te duiden element, de actie voor het voltooien en of er een 'volgende'-knop zal zijn, moet worden aangeduid. De stap wordt dan aangemaakt door op de 'Voeg stap toe'-knop te drukken.

Aan de rechterkant van het interface zullen alle aangemaakte stappen in een lijst verschijnen. Stappen in de lijst kunnen ook van plaats verwisseld en verwijderd worden. Wanneer de les volledig is, moet er op de 'Les opslaan'-knop, onderaan het interface, gedrukt worden om de les op te slaan. De les zal dan voor alle gebruikers verschijnen in de lijst van lessen.
In de lessen lijst krijgt de beheerder ook de optie om lessen te verwijderen door op de 'X' te drukken of lessen aan te passen door op 'wijzig' te drukken. De 'wijzig'-knop opent het interface voor het aanmaken van lessen met alle stappen van die les reeds ingeladen. Er kunnen dan aanpassingen aan de les gemaakt worden en met de 'Les opslaan'-knop kan de les overschreven worden.

3.10 Registreren en inloggen

Rechts bovenaan de pagina, in de header, staan de 'registreer'- en 'inloggen'-knoppen. De 'registreer'-knop opent een pop-up waar een account aangemaakt kan worden. De gebruiker moet een gebruikersnaam ingeven en daarna een wachtwoord methode kiezen.

Er zijn twee wachtwoord methodes. De eerste optie, die aangeraden is voor jongere gebruikers, is het patroon-wachtwoord. De gebruiker kiest dan op een 3x3 veld een aantal vakjes. Er moet minstens 1 vakje aangeduid worden. Als 2de optie kan de gebruiker kiezen voor een standaard wachtwoord, bestaande uit tekst. Hierbij moet de gebruiker minstens 6 tekens ingeven. Dit wachtwoord is veiliger, maar minder kindvriendelijk dan het patroon wachtwoord. Het tekstueel wachtwoord is bedoeld voor belangrijkere accounts, zoals het account van de beheerder, beter te

beveiligen, maar kan ook gebruikt worden door gebruikers die liever een tekstueel wachtwoord onthouden. Voor beide wachtwoord methoden moet de gebruiker het wachtwoord bij het registreren twee maal invoeren om fouten te voorkomen. Zodra er dan op de 'Registreer'-knop wordt gedrukt, zal er geprobeerd worden een account aan te maken. Als het aanmaken van het account mislukt, zal er info over waarom het mislukte onder de knop verschijnen. Als het aanmaken van een account lukt, zal de gebruiker automatisch ingelogd worden, de pop-up zal gesloten worden en de registreer- en inlog-knoppen zullen vervangen worden door de gebruikersnaam en een 'uitloggen'-knop. De 'uitloggen'-knop logt de gebruiker uit.
Als men reeds een account heeft aangemaakt, kan de gebruiker inloggen met de pop-up die verschijnt als er op de 'inloggen'-knop gedrukt wordt. Voor in te loggen, is een gebruikersnaam en een wachtwoord nodig. Ook hier moet er gekozen worden tussen patroon en tekst. De optie gebruikt bij het registreren zal dan gekozen moeten worden.

3.11 Opslaan en laden van projecten

In de header staat er een 'Opslaan'-knop (als er een opgeslagen project is ingeladen), een 'Opslaan als'-knop (als men ingelogd is) en een 'Laden'-knop (die altijd zichtbaar zal zijn).

Wanneer er op de 'Opslaan als'-knop gedrukt wordt, zal er een pop-up verschijnen. In deze pop-up kan het huidige project worden opgeslagen. Om een project op te slaan wordt er enkel een naam gevraagd. Als de gebruiker in deze pop-up vervolgens op 'Opslaan' klikt zal het project in de database opgeslagen worden. Het slagen of falen van het opslaan kan de gebruiker zien aan het response bericht dat boven de 'Opslaan'-knop zal verschijnen.

Opgeslagen projecten kunnen bekeken worden in de pop-up die verschijnt als de 'Laden'-knop wordt ingedrukt. In deze pop-up is een lijst te zien van alle opgeslagen projecten van de ingelogde gebruiker. Deze lijst wordt vervangen door een melding en een 'Login'-knop wanneer men niet is ingelogd. Wanneer er iets mis gaat tijdens het ophalen van de projecten zal er een error verschijnen. Opgeslagen projecten kunnen worden ingeladen door op de knop naast de naam te drukken. Dit vervangt het huidige project, hierdoor wordt er eerst ook een waarschuwingspop-up gegeven die de gebruiker er op zal wijzen dat het huidige project verloren zal gaan.

Als de gebruiker een opgeslagen project heeft ingeladen zal de 'Opslaan'-knop in de header verschijnen. Deze knop zorgt er voor dat het opgeslagen project in de database wordt overschreven. Bij het klikken op de knop gaat er oftewel een vinkje verschijnen om het succesvol opslaan weer te geven oftewel een kruis om het niet succesvol opslaan weer te geven.

3.12 Delen van projecten

Projecten kunnen gedeeld worden door de 'Project delen'-knop in te drukken in de header. Dit laat opnieuw een pop-up verschijnen waar een 'Genereer code'-knop aanwezig is. Als de gebruiker op deze knop klikt, gaat er een kopie van het huidige project in de database opgeslagen worden en krijgt de gebruiker een code waarmee het kopie van dit project geopend kan worden. Als een gebruiker een project wilt inladen met een code, dan kan dit door op de 'Project laden'-knop te drukken in de header, vervolgens gaat de pop-up voor het laden van projecten verschijnen. Onderaan deze pop-up staat er 'Laden met code:' geschreven, hiernaast is een input veld voorzien waar de gebruiker de code van het project, dat hij/zij wilt inladen, kan typen. Als de gebruiker daarna op 'Laden' drukt zal er een request naar de database gestuurd worden voor het project met de ingegeven code op te halen. Als dat succesvol gebeurt, verschijnt er een melding die aan de gebruiker vraagt of hij/zij zeker is dat hij/zij een nieuw project wilt inladen en dat zijn/haar huidig project verloren gaat als dit niet opgeslagen is. Anderzijds als deze request faalt, gaat er een error verschijnen die de gebruiker meer informatie geeft over waarom de request gefaald is.

4 Problemen

In deze sectie wordt beschreven hoe feedback van zowel de opdrachtgever, de begeleider als de doelgroep verwerkt is in het project. Daarnaast worden verschillende problemen aangehaald die voorgevallen zijn tijdens het creëren van de applicatie. Deze problemen worden opgedeeld in 4

categorieën: verwerking van feedback, problemen door structuur, problemen in verband met layout en andere problemen.

4.1 Verwerking van feedback

Op verschillende momenten is er feedback ontvangen over de applicatie. De meeste feedback kwam tijdens presentaties en meetings met de begeleider. Van de opdrachtgever was het jammer genoeg niet mogelijk om feedback te verkrijgen. Alle feedback is verwerkt en de meeste nieuwe voorgestelde features zijn toegepast. In deze sectie zal de verwerking van deze feedback besproken worden.

4.1.1 Stap-knop

De stap functionaliteit is een feature die tot stand is gekomen doordat het onderwijsteam aanhaalde dat de applicatie gebruikt zou worden om kinderen iets nieuws aan te leren. Toen dit werd voorgesteld, gaven programma 's nog onmiddellijk resultaat en was het niet mogelijk de uitvoering van blokken te volgen. Het starten moest daarom vertraagd worden. Er werd een tijdsinterval van een seconde gezet tussen de uitvoering van blokken. Ook werd de stap-knop geïmplementeerd zodat de gebruiker zelf op eigen tempo een programma kon uitvoeren. Het blok dat uitgevoerd wordt, kreeg ook een gele rand om dit visueel te verduidelijken naar de gebruiker. De implementatie van dit tijdsinterval en de stap-knop vereiste een heel nieuwe structuur van uitvoering waar eerder niet over was nagedacht.

Bij de herkansing van het project werd er ook een vertraging-schuifregelaar toegevoegd. Zo kan de gebruiker het tijdsinterval bij het uitvoeren met behulp van de start-knop zelf beslissen.

4.1.2 Undo- en Redo-knop

De undo- en redo-knop waren beide voorgesteld door het onderwijsteam. Deze knoppen zorgen ervoor dat de fouten die de gebruiker maakt minder erg zijn. Deze zijn dus geïmplementeerd om deze reden.

4.1.3 Wachtwoorden

Het gebruik van wachtwoorden werd afgeraden door het onderwijsteam, doordat de doelgroep kinderen van 8 tot 12 jaar zijn. Er werd lang nagedacht over alternatieven en uiteindelijk zijn we gegaan voor een patroon wachtwoord. Doordat deze nieuwe methode niet zo veilig is, kan de gebruiker kiezen tussen een tekstueel wachtwoord en een patroon wachtwoord. Op die manier kunnen belangrijke accounts toch beter beschermd worden met een veiliger wachtwoord.

4.1.4 Verwijderen van variabelen en functies

In een eerdere versie zouden variabelen en functies verwijderd kunnen worden terwijl er nog verwijzingen op het canvas staan. Het gebruik van deze verwijzingen zou dan een fout melding geven. Het werd aangeraden door het onderwijsteam om het verwijderen te blokkeren wanneer er nog verwijzingen op het canvas staan. Op die manier moet de gebruiker eerst zoeken waar de blokken staan voordat het verwijderd kan worden, dan weet de gebruiker precies wat er niet meer zal werken.

Bij een poging om een variabele of functie te verwijderen wanneer er nog verwijzingen op het canvas staan, zal er nu een waarschuwing melding gegeven worden links bovenaan het canvas en zullen alle verwijzingen tijdelijk oplichten, zodat deze makkelijker te vinden zijn.

4.1.5 Feedback van de doelgroep

Er werd gepland om de applicatie uit te testen bij de doelgroep. Zo kon er rechtstreeks feedback van de doelgroep toegepast worden op de applicatie. Wegens de lockdown door het coronavirus was dit niet meer mogelijk en is er niet veel feedback van de doelgroep toegepast kunnen worden.

Feedback die wel van de doelgroep kwam, was dat de start-, stop-, stap-knoppen niet draggable mochten zijn. <img>-tags zijn standaard draggable. Bij het proberen van het plaatsen van deze

knoppen op het canvas, was een lichtere versie van de knop zichtbaar die met de cursor mee bewoog, dit gaf het gevoel dat deze knoppen ook geplaatst konden worden op het canvas.

Er kwam ook positieve feedback van de doelgroep. De felle kleuren van de blokken en het nog lege canvas maakt het direct duidelijk aan de gebruiker dat er iets met de blokken uit de blokkenlijst moet gebeuren. Na het gebruik van een basis-les die de gebruiker een simpel programma liet bouwen, was de werking van de applicatie duidelijk en kon de gebruiker zelf verder de applicatie exploreren.

4.2 Problemen door structuur

Een aantal problemen zijn tot stand gekomen door een minder goed opgestelde structuur in de code. Sommige van deze problemen komen door slechte planning, anderen door het tekort aan ervaring met TypeScript en VueJS. In deze sectie staan een aantal van de grootste problemen vermeld.

4.2.1 'If-else'-blok

Origineel was de implementatie van een 'if-else'-blok gepland. Dit blok zou werken zoals een 'als-dan'-blok, maar met een extra enclose component eronder. Tussen de 2 enclose componenten zou er een label zitten dat aanduidde wanneer het 2de enclose component gebruikt zou worden. Dit label component kon niet op de gewenste plaats geplaatst worden, doordat het label component zich altijd in de basis van een blok bevindt. Zonder dit label was het 'if-else'-blok zeer onduidelijk. De oplossing voor dit probleem zou het toevoegen van een nieuw soort label component zijn dat onderaan het blok kan worden toegevoegd. Omdat dit blok makkelijk te maken is met behulp van 2 'als-dan'-blokken en 1 NIET-blok, is er voor gekozen om geen nieuw component te maken voor dit blok en dit blok simpelweg niet te implementeren.

4.2.2 Volgorde van componenten toevoegen

Als er in een blok 2 label componenten worden toegevoegd, zal het eerst toegevoegde label links van het 2de staan. Dit was de geplande manier voor het plaatsen van componenten in het blok.

Daarna werd er ontdekt dat componenten boven, onder of in het blok ook in een bepaalde volgorde moeten worden toegevoegd aan het blok. De verwachte volgorde waarin componenten moeten worden toegevoegd is top, bottom en base-componenten. Dit was eerst geen probleem, maar bij het implementeren van functie-blokken zorgde dit voor problemen. In het functie-blok werd een base component toegevoegd voor alle bottom componenten toegevoegd werden. Dit zorgde voor een bug in het enclose component. Het probleem met de volgorde van de componenten is niet opgelost doordat het een kleine bug was die maar betrekking had tot 1 blok en het aanpassen van de volgorde van de componenten in dat blok makkelijker was en minder tijd kostte dan het herschrijven van de componenten-structuur.

4.2.3 Type errors

Als er een blok met een return component in een blok met een input component geplaatst wordt, wordt er eerst gekeken of het type van de componenten gelijk zijn. Als dit niet het geval is, zal er een error gegeven worden. Return component en input component zijn beide klassen gemaakt met een template. Het vergelijken van types van een template klasse is echter onmogelijk in TypeScript. Dit probleem is opgelost door aan de constructor van ieder return en input component de naam van het type expliciet als string mee te geven.

4.2.4 Bewaren van blokken

De applicatie onthoudt enkel de bovenste blokken van een aaneenschakeling van blokken op het canvas. De blokken in een attach, enclose of input component worden nergens bijgehouden in een lijst. Op meerdere momenten heeft deze structuur ons gehinderd, bijvoorbeeld bij het verwijderen van een functie-blok of een variabele.

Wanneer de gebruiker een functie-blok of variabele verwijderd, moeten alle referenties ernaar gezocht worden. Hiervoor moet voor alle blokken op het canvas gekeken worden of ze een variabele-blok of functieaanroep-blok zijn. Daarna moet er bekeken of er nog referenties in het blok zijn attach, enclose of input component zitten.

Om al dit werk te vermijden worden variabele-blokken en functieaanroep-blokken, dat zich op het canvas bevinden, in een lijst bijgehouden. Bij het aanmaken van een dergelijk blok, zal het blok direct aan de lijst toegevoegd worden. Zo moet er bij het verwijderen van een functie of variabele enkel gekeken worden naar deze lijsten. Om deze lijsten up-to-date te houden, moet er echter iedere keer een blok verwijderd wordt, gekeken worden of er in de componenten van het blok geen variabele- of functieaanroep-blok zit. In het geval dit zo is, zullen de lijsten aangepast worden. Deze code zal jammer genoeg niet meer werken wanneer er een nieuw component wordt toegevoegd die deze blokken kan bevatten. De code zou doen aangepast moeten worden om ook op dit nieuwe component te testen.

Omdat alles rond deze structuur gebouwd is, is het niet mogelijk om alle blokken in een lijst te krijgen zonder grote aanpassingen te maken.

4.3 Problemen in verband met de lay-out

Tijdens de bespreking van de vorige versie van het project hebben we enkele opmerkingen gehad over de lay-out, deze opmerkingen gaan in deze sectie besproken worden.

4.3.1 Responsive design van de website

In de vorige versie werkte de applicatie enkel in een maximized venster en werden meerdere resoluties niet ondersteund (enkel 1080p). Dit is nu aangepast zodat de applicatie een responsive design heeft en werkt vanaf dat het scherm een 720p resolutie heeft, als het scherm toch een lagere resolutie heeft gaat de applicatie niet verder scalen maar gaan er scrollbars komen zodat de applicatie nog steeds bruikbaar is. Voor grotere resoluties gaat de applicatie automatisch zijn onderdelen aanpassen in de breedte en hoogte.

4.3.2 Consistentie van de lay-out

In deze versie van de applicatie is er ook voor gezorgd dat elk element van de UI zijn eigen header heeft met de naam van het element, zodat hier naar verwezen kan worden tijdens lessen. Verder is er ook voor gezorgd dat alle elementen in de UI consistent ons thema toepassen, zo is bijvoorbeeld de styling van de knoppen in de applicatie overal consistent. Als laatste zijn de pop-up meldingen die de gebruiker om bevestiging vragen niet meer de default browser pop-up maar een zelf gemaakte pop-up die binnen ons thema past.

4.3.3 Consistent gebruik van de Nederlandse taal

Er stonden nog enkele Engelse woorden in de applicatie zoals 'print', 'admin', 'warning', 'error', 'index', 'string' en 'substring'. Deze zijn nu vertaald naar het Nederlands.

4.3.4 Aanduiding in lessen

In een stap in een les kan een element van de applicatie aangeduid worden. De aanduiding van het element reageerde eerst niet op aanpassingen aan de applicatie. Wanneer het element bijvoorbeeld zou bewegen door een verandering van schermgrootte, bleef de aanduiding op de originele positie staan, terwijl het element zich daar niet meer bevond. Nu is de positie van de aanduiding gekoppeld aan de positie van het element, waardoor de aanduiding met het element zal meebewegen wanneer de positie van het element aangepast wordt.

4.4 Andere problemen

In deze sectie zullen alle opgeloste problemen, die niet in de vorige categorieën pasten, aangehaald worden.

4.4.1 Selecteren van blokken

Wanneer er op een blok geklikt wordt, doordat men het blok wilt verslepen, zal het blok relatief op dezelfde plek blijven ten opzichte van de muis tijdens het slepen. Dit gebeurde origineel niet in alle gevallen. Wanneer er op een component van het blok, en niet op de basis rechthoek van het blok, werd gedrukt, verplaatste het blok zich zodat de muis zich in de linkerhoek bevond. Dit kwam doordat er een fout zat in de berekening van de offset van de muis. De fout in de berekening is ondertussen opgelost.

4.4.2 Highlight van blokken

Actieve blokken, blokken die uitgevoerd worden, worden omlijnd met een gele rand. De gele kleur is niet even duidelijk op iedere blok, maar na het bekijken van alternatieven, werd er toch besloten om de kleur geel te behouden.

Een ander probleem met de omlijning is dat alle elementen van het blok omlijnt worden. Doordat het blok opgebouwd is uit verschillende componenten, worden al de componenten apart aangeduid. Dit ziet er niet altijd even goed uit, maar het oplossen van dit probleem zou te veel tijd innemen voor een relatief klein probleem. Hierdoor is het probleem niet opgelost.

4.4.3 Commentaar blok

Het commentaar blok is een blok met een tekstvak (HTML textarea tag). Het nut van dit blok is het geven van extra uitleg bij de gemaakte code. Het tekstvak wordt groter naarmate de hoeveelheid tekst er in staat. Wanneer men dit blok verschuift, verkleint het tekstvak en verschijnt er een scrollbar aan de zijkant. Eens men terug probeert te typen in het vak, zal het tekstvak terug vergroten. Het oplossen van deze bug is aan de kant geschoven doordat de verkleinde versie van het blok veel overzichtelijker was in gebruik wanneer men blokken probeert te verplaatsen. Wanneer men simpelweg tekst wilt laten zien is de grote versie van het blok echter handiger.

4.4.4 Browser ondersteuning

Gedeeltes van de code bevatten onderdelen die niet ondersteunt worden op iedere browser. Omdat we er voor willen zorgen dat elke gebruiker ons programma in de meest optimale manier kan gebruiken, geven we een foutmelding bij de browsers: Internet Explorer en Microsoft Edge.

5 Revisie van de analyse

Bij het creëren van het analyseverslag werden een aantal evaluatiecriteria opgesteld. In dit deel zullen aanpassingen in vergelijking met de analyse en deze criteria overlopen worden. Enkel de grootste aanpassingen uit de analyse zullen aangehaald worden, de evaluatiecriteria zullen allemaal overlopen en geëvalueerd worden.

5.1 Aanpassingen ten opzichte van de analyse

In deze sectie worden de grootste aanpassingen ten opzichte van de vooraf gemaakte analyse bekeken.

5.1.1 Algoritmen

In het analyseverslag staat beschreven dat de uitvoering van blokken in het blok zelf gebeurt. Ieder blok heeft zijn eigen execute-functie en een pointer naar het volgende blok. Als de execute-functie klaar is, zal het de execute-functie van het volgende blok oproepen. In het finale project wordt het volgende blok echter niet onmiddellijk aangeroepen, maar gebeurt dit met behulp van een flowcontroller-klasse. Deze flowcontroller-klasse zorgt ervoor dat het programma stap voor stap uitgevoerd kan worden. Dit was nodig voor de stap-knop en de vertraging van de start-knop te kunnen implementeren.

5.1.2 Datastructuren

In de sectie 'Datastructuren' van de analyse werden alle verschillende onderdelen, dat een blok kon hebben, aangehaald. Deze onderdelen worden 'componenten' genoemd. De blokken in de uiteindelijke applicatie bestaan uit deze componenten. Er zijn echter wel een paar componenten bijgekomen.

Het break component is één van deze componenten. Door een break component toe te voegen aan de basis van een blok kunnen andere componenten onder elkaar geplaatst worden in plaats van enkel naast elkaar. Buiten het verbeteren van de opmaak van het blok, heeft dit component geen functie.

Het choice component is een klein drop-down menu met een aantal opties. Dit component wordt gebruikt in afbeelding-blokken en het 'start op druk van toets'-blok om respectievelijk een afbeelding en een toets te kiezen.

Het button component is een basis component dat een knop toevoegt aan het blok. De tekst in de knop en de functie die uitgevoerd wordt bij het drukken op de knop, kan gekozen worden en hangt van blok tot blok af. Dit component wordt momenteel enkel gebruikt bij het functie blok om de naam van een functie te veranderen.

Het error component stond niet in de analyse en is bij elk blok toegevoegd. Dit komt doordat errors origineel los van het blok geïmplementeerd werden. Dit component laat toe om fout, waarschuwing en succes meldingen te tonen op het blok. Door dit component aan het blok te hangen, zal de melding altijd langs het blok staan.

De overige componenten die later zijn toegevoegd, zijn een kopie van andere componenten met een lichte aanpassing voor de API-blokken. Om kinderen geen JSON-objecten te laten typen, hebben we bij de API-blokken een blok toegevoegd dat een veld en een waarde opvraagt. API-blokken die een JSON-object nodig hebben voor hun request uit te voeren, hebben een 'API-enclosure component'. Dit is een enclosure component dat enkel de JSON-parameter-blokken aanvaardt. Om dit duidelijk te maken, gebruiken deze componenten driehoeken die in elkaar passen in plaats van cirkels als puzzelstuk vorm. De JSON-parameter-blokken hebben dus ook een API-connect component en een API-attach component.

5.1.3 Klassendiagram en ontwerp

Tijdens het ontwerpen van het klassendiagram waren we 2 mogelijkheden tegengekomen om componenten toe te voegen aan een blok. Een array met elementen van klasse blockcomponent of het aanmaken van variabelen voor iedere component. Uiteindelijk verkozen we om beide methoden te gebruiken. Bij het aanmaken van een blok worden de componenten aangemaakt en opgeslagen in variabelen. Deze variabelen worden dan gebruikt in de execute-functie van het blok. Ieder blok heeft ook een array met componenten. Bij het aanmaken van een component, wordt het component niet enkel opgeslagen als variabele, maar ook toegevoegd aan deze array. Deze array wordt later vaak gebruikt om te kijken of een blok een bepaald component heeft.

Het finale klassendiagram is geen uitbreiding op het klassendiagram dat vermeld staat in de analyse. Deze is volledig opnieuw gemaakt. Dit komt doordat de leesbaarheid sterk verbetert bij een opgesplitst klassendiagram. In het finale klassendiagram zitten ook redelijk veel onderdelen die niet in het originele klassendiagram vermeld zijn. De structuur van de blokken en hun componenten is ongeveer hetzelfde gebleven, er zijn enkel een aantal componenten toegevoegd (zoals choice component en break component) waarvan de implementatie tijdens het programmeren pas nodig bleek.

5.1.4 Backend

In de analyse staat dat er geen framework gebruikt zou worden en dat de backend enkel zou bestaan uit PHP en AJAX. Tijdens het implementeren van de backend bleek dit moeilijker te zijn dan een database en web framework module te gebruiken.
Voor de database wordt er gebruik gemaakt van SQLite. Deze keuze werd gemaakt doordat SQLite de database inleest en opslaat in 1 file waardoor er makkelijk verschillende versies van de database opgeslagen kunnen worden.
Voor het web framework wordt er gebruik gemaakt van Express, dit werd gekozen doordat over dit framework het meeste informatie te vinden was.
Axios is de module die gebruikt wordt om XMLHttpRequests te sturen vanuit de browser. Deze module wordt gebruikt omdat axios het gebruik van Promises ondersteunt.
Voor gebruiker authenticatie wordt JsonWebToken gebruikt. In deze token worden de userID en of de user een admin is in opgeslagen. Deze token is telkens geldig voor 1 dag, als het token niet meer geldig is en de gebruiker een request naar de server probeert te sturen waar authenticatie voor nodig is, zal de gebruiker uitgelogd worden en zal de login pop-up verschijnen.

5.1.5 Data en databaseschema

De database en bestandformaten hebben in het eindproduct een aantal lichte aanpassingen gekregen. In deze sectie zullen de uiteindelijke databaseschemas en bestandsformaten geven worden en hun aanpassingen besproken worden.

User tabel In deze tabel worden de gebruiker-accounts opgeslagen. Het bevat de volgende velden:

- int id : uniek id per gebruiker (primary key)
- string username : gebruikersnaam (unique)
- string password : wachtwoord
- int admin : 1 als user admin is, anders 0

De gebruikersnaam moet uniek zijn zodat er mee ingelogd kan worden. Doordat er geen gebruik gemaakt wordt van PHP, kan er ook geen gebruik gemaakt worden van de PHP password_hash() functie. Er werden verschillende hashing algoritmes bekeken waaronder SHA256, bcrypt en Argon2. Bij SHA256 was het snel duidelijk dat deze niet gebruikt ging worden, doordat het aantal hashes per seconde die gegenereerd kunnen worden hier opmerkelijk hoger ligt dan bij de andere algoritmes. Argon2 is een redelijk nieuw hashing algoritme, dit algoritme is beter dan bcrypt omdat het meer parameters heeft die ingesteld kunnen worden, om zo het berekenen van hashes moeilijker te maken. Uiteindelijk werd er voor bcrypt gekozen omdat dit algoritme een goed ondersteunde module heeft en veilig genoeg is voor de doeleinden van deze applicatie. Dit is echter wel geen future proof oplossing, aangezien bcrypt in de toekomst niet meer heel veilig zal zijn.
Uiteindelijk zijn er twee wachtwoord systemen geïmplementeerd. Het standaard tekstueel wachtwoord en een wachtwoord met behulp van een patroon. Achterliggend blijft de werking echter hetzelfde. Het wachtwoord zal als string gebruikt worden. De string gebruikt bij het patroon wachtwoord bestaat uit een 9 karakter lange combinatie van enen en nullen. Een opgevuld vak in het patroon wordt gezien als een één, een onaangeduid vak als een 0. Deze nummers worden dan rij per rij van links naar rechts achter elkaar gezet om de string te vormen.

Projects tabel In deze tabel worden de projecten opgeslagen van de gebruikers. De tabel bevat de volgende velden:

- int fileid : uniek id per file (primary key)
- int userid : id van de bestandeigenaar (foreign key, references User tabel(id))
- string projectname : naam van het project, gekozen door de gebruiker

- json project : het project in JSON formaat

Voor het formaat van het project in JSON, zie 5.1.6.

Shared_projects tabel Deze tabel dient voor het delen van projecten. Bij het delen van projecten zal er een kopie gemaakt worden van het te delen project. Deze kopie wordt tijdelijk (tot middernacht) opgeslagen. Wanneer de server om middernacht niet opstaat, wordt de tabel niet gereset. Daarom is er een startup functie toegevoegd die er voor zorgt dat wanneer de server opstart, alle gedeelde projecten ouder dan een dag verwijdert worden. Er kan niet simpelweg een reset gedaan worden wanneer de server opstart, want als er zich dan een elektriciteitsstoring zou voordoen, zouden alle gedeelde projecten van die dag verloren gaan.
De gebruiker kan aanpassingen maken aan een ingeladen kopie zonder het originele projecten aan te passen. Als de gebruiker dan zijn aanpassing wilt opslaan, kan hij/zij dit doen onder zijn eigen account door het huidig project op te slaan. Bij het aanmaken van een kopie wordt ook een code aangemaakt. Deze code is uniek voor elk gedeeld project. De tabel heeft de volgende velden:

- int code : uniek per tijdelijke file, code waarmee de kopie geopend kan worden (key value)
- timestamp time : de tijd en datum wanneer het project werd opgeslagen in de database
- json project : het tijdelijke project in JSON formaat

Tutorials tabel Deze tabel dient voor het opslaan van lessen. De tabel heeft de volgende velden:

- string name : de naam van de les (primary key)
- string tutorialdata : de les in zijn string formaat

5.1.6 Bestandsformaat

In de uiteindelijke applicatie is het bestandsformaat voor het opslaan van projecten licht aangepast door inzichten tijdens het implementeren.
Het geüpdatet formaat ziet er als volgt uit:

- projectname
  Naam van het project
- blocks
  Een lijst van opgeslagen blokken
- projectvars
  Een lijst van variabelen namen gepaard met hun waardes
- projectimgs
  Alle aangevinkte afbeeldingen in het project worden opgeslagen samen met hun positie, grootte en rotatie.

Blokken worden opgeslagen op de volgende manier:

- x
  X-coördinaat van het blok
- y
  Y-coördinaat van het blok
- blockdata
  Het blockdata bestaat uit classname: de naam van de blok-klasse en classdata: de informatie over het blok. Classdata ziet er als volgt uit:
  - name
    Naam van het blok

- blockid
  Id van het blok
- color
  Kleur van het blok
- components
  Een lijst van opgeslagen componenten van het blok. Voor elk component wordt de naam, de kleur en unieke data voor het component opgeslagen. Unieke data voor een component kan, bijvoorbeeld, het blok of de tekst in een input component zijn.

Variabelen- en functie-blokken slaan buiten de standaard data ook de naam van de variabele of functie waartoe ze behoren op.

5.2 Evaluatie op basis van criteria: functionele vereisten

In deze sectie zullen de evaluatiecriteria overlopen en geëvalueerd worden. Eerst zullen de functionele vereisten overlopen worden aan de hand van de priority-list. Deze zag er als volgt uit:

High priority:

1. Oneindig canvas
2. Blokkenlijst
3. Basis blokken
4. Start- en stopknop
5. Output via console

Medium priority:

1. Opslaan en laden op webserver met accounts
2. API ondersteuning (GET)
3. Lessen
4. Output en input via afbeeldingen

Low priority:

1. Delen van projecten
2. String operaties
3. Variabelen
4. API ondersteuning (POST, PUT , DELETE)
5. Undo voor acties

5.2.1 Oneindig canvas

De basis-functionaliteit van het canvas was enkel een leeg canvas aanmaken bij de start van een nieuw programma, het plaatsen van blokken op het canvas, het verslepen van blokken op dit canvas en het verslepen van het canvas zelf. Deze functionaliteit is volledig geïmplementeerd in het project. De naam 'oneindig canvas' is wel een misleidende naam, het canvas is niet oneindig, maar simpelweg groter dan wat de gebruiker in 1 keer kan bekijken.

5.2.2 Blokkenlijst

De blokkenlijst is een lijst naast het canvas waarin alle blokken, die men kan gebruiken, zich bevinden. De blokken zijn opgedeeld per categorie. Deze categorieën kan men openen en sluiten om een duidelijker overzicht te hebben. Doordat de lijst langer is dan het scherm kan laten zien, kan men scrollen met behulp van de muis of door de scrollbar aan de zijkant van de lijst te verslepen. De blokkenlijst voldoet aan alle criteria die gezet waren. Een extra feature die de blocklist heeft, is dat naast de naam van de categorie de kleur van deze categorie afgebeeld wordt.

5.2.3 Basis blokken

Het criteria 'Basis blokken' stelde de implementatie van enkele blokken voor die in het begin van het project gebruikt zouden kunnen worden. Alle blokken die hier vermeld stonden zijn geïmplementeerd. Uiteraard zijn er nog vele andere blokken geïmplementeerd.

5.2.4 Start- en stopknop

Zowel de start- als de stop-knop zijn geïmplementeerd en werken zoals men zou verwachten. In de criteria staat vermeld dat de stopknop enkel ingedrukt zou kunnen worden wanneer een programma bezig is en de startknop enkel wanneer er nog geen programma bezig is. Dit klopt niet in de finale versie van de applicatie. De stopknop kan altijd ingedrukt worden, maar werkt enkel als er een programma bezig is. Dit komt doordat de stop-knop ook gebruikt kan worden wanneer men gebruik maakt van de stap-knop.

5.2.5 Output via console

Output die uitgeschreven wordt met behulp van het 'Toon tekst'-blok zal weergegeven worden in het tekst venster. Dit is een apart venster langs het canvas. De gebruiker kan alle tekst in het venster verwijderen door op de 'herstel'-knop te drukken. Deze vereiste is volledig voldaan.

5.2.6 Opslaan en laden op webserver met accounts

Voor projecten op te slaan en deze in te laden is het gebruik van accounts nodig. Accounts kunnen aangemaakt worden met een gebruikersnaam en wachtwoord. Wanneer de gebruiker ingelogd is, verschijnt de knop 'Opslaan als'. Hier kan men een project opslaan met een zelfgekozen naam. De knop 'Laden' is altijd zichtbaar, als men ingelogd is komen hier alle vooraf opgeslagen projecten van de gebruiker te staan. Vooraleer men een project inlaadt, wordt er een waarschuwing gegeven dat men het huidige project kwijt zal raken. Wanneer een project is ingeladen, verschijnt er ook een knop 'Opslaan'. Deze knop overschrijft het opgeslagen project met het huidige project.

5.2.7 API ondersteuning

Er zijn een aantal blokken geïmplementeerd die HTTP-request methods representeren. Zo is er een GET, POST, PUT, PATCH en DELETE-blok. De POST, PUT en PATCH-blokken verwachten data mee te krijgen in de vorm van een JSON-object. Om kinderen niet zelf JSON-code te laten schrijven, werd er een JSON-parameterblok toegevoegd. Deze blok verwacht een key en een value waarde. Door deze blokken in het API-enclosure component van de API-blokken te steken, kan men een JSON-object meegeven.

5.2.8 Lessen

De lessen in de finale versie van het project zien er anders uit dan beschreven staat bij de evaluatie-criteria. In de criteria werden lessen voorgesteld in een klein, apart venster. In dit venster zou stap per stap beschreven staan hoe de gebruiker een voorbeeld programma kon maken. Tegelijkertijd zou de gebruiker dan op zijn/haar eigen canvas kunnen experimenteren. Deze manier van lessen werd afgeraden door het onderwijsteam.
De lessen in het finale project werken aan de hand van acties. Zo wordt er telkens van de gebruiker

verwacht dat ze een bepaalde actie uitvoeren in de applicatie. De rest van de applicatie wordt tijdelijk verduisterd en men kan enkel deze actie uitvoeren, zo is het altijd duidelijk wat de gebruiker nu precies moet doen. Bijkomend is er ook een tekstballon waarin de uitleg van de stap wordt weergegeven. In dit tekstballon staat er ook altijd een knop 'stoppen' die de les op ieder moment kan laten stoppen. Met een beheerder-account kunnen er ook lessen toegevoegd worden. Er is een simpele user-interface voorzien om beheerders zelf lessen te laten creëren.

5.2.9 Output en input via afbeeldingen

In de criteria staat output via afbeeldingen beschreven als een afbeelding die uitgeschreven wordt naar de output console. In het werkelijk project is dit heel anders gedaan. Er is een afbeeldingen venster waarop een aantal afbeeldingen, die op voorhand ingeladen worden, zichtbaar zijn. Iedere afbeelding kan men zichtbaar of onzichtbaar maken (afhankelijk van welke afbeeldingen de gebruiker wilt gebruiken). Er zijn blokken voorzien om deze afbeeldingen te manipuleren. Zo kan men afbeeldingen verplaatsen, draaien, vergroten en verkleinen. Men kan ook manueel de afbeeldingen over het afbeeldingen venster verplaatsen door deze te verslepen. Input via afbeeldingen kan via het 'Start op klik van afbeelding'-blok. Dit blok start wanneer er op een gekozen afbeelding wordt gedrukt.

5.2.10 Delen van projecten

Er is geen account vereist om projecten te delen of gedeelde projecten in te laden. Bovenaan in de header bevindt zich de knop 'project delen'. Als men deze indrukt, kan een gebruiker een code genereren voor zijn/haar project. Een andere gebruiker kan dan via de knop 'Laden' het gedeelde project inladen met behulp van de code. (Hierdoor is de knop 'Laden' ook zichtbaar wanneer men niet ingelogd is.) Het delen van projecten werkt zoals beschreven staat in de criteria, op 1 punt na: de gegenereerde code is echter geen 5-cijferige code, maar een simpel getal.

5.2.11 String operaties

In het project zijn er verschillende blokken voorzien om string operaties voor te stellen. De string operaties die wij voorzien zijn: 'Toon tekst' (tekst in het tekst venster tonen), lengte (lengte van een string), samenvoegen (strings samenvoegen), letter op plaats (letter op index 'x', waar x meegegeven wordt), eerste plaats van letter (de eerste index van karakter 'x', waar x meegegeven wordt) en deel van tekst (substring beginnende op index 'x' met lengte 'y', waar x en y meegegeven worden aan het blok).

5.2.12 Variabelen

Variabelen worden aangemaakt in het variabelen venster langs het canvas. Om een variabele aan te maken, wordt er een naam en waarde gevraagd. Deze waarde is achterliggend een string. Als de variabele in een blok zit waar een integer verwacht wordt, wordt de waarde omgezet tot een integer met behulp van casting. Wanneer een variabele aangemaakt is, komt deze zowel in het variabele venster te staan, als in de blokkenlijst onder de categorie 'Waardes'. Het aanpassen van de waarde van een variabele gebeurt met behulp van het 'Zet variabele'-blok waarin men een variabele en een waarde moet stoppen. In het variabelen venster kunnen ook variabelen gewist worden, dit gaat enkel wanneer er geen 'variabele'-blokken van dat variabele meer op het canvas staan.

5.2.13 Undo voor acties

De undo-functionaliteit wordt ondersteund voor het plaatsen, verplaatsen, verwijderen, connecteren, deconnecteren van blokken, het invoegen van blokken in een inputcomponent, het schrijven van tekst in een inputcomponent en het aanmaken en verwijderen van variabelen. Alle acties verlopen via de blockcontroller. Deze onthoudt een omgekeerde operatie van alle acties op een stack. Zo moet de bovenste actie op de stack uitgevoerd worden wanneer men de undo-knop indrukt. Ook is er een redo-knop, deze werkt op dezelfde manier als de undo-knop. Bij redo worden echter gewoon omgekeerde operaties van alle acties die bij undo uitgevoerd worden, opgeslagen.

5.3 Evaluatie op basis van criteria: niet-functionele vereisten

De niet-functionele vereisten zijn de kwaliteitseisen waaraan het programma moet voldoen. Deze worden overlopen aan de hand van enkele subsecties, namelijk: lay-out, leesbaarheid, zoeken van blokken in de blokkenlijst, blokken terugvinden op het canvas en robuustheid.

5.3.1 Lay-out

De aandachtspunten voor de lay-out waren:

1. Makkelijk te begrijpen
   Het is direct zichtbaar dat men iets met moet doen met de fel gekleurde blokken die zich links in de blokkenlijst bevinden. Dit komt deels door de felle kleuren van de blokken en deels door het feit dat het canvas en de verschillende output-vensters rechts nog leeg zijn bij de start van een nieuw project.
   Ieder onderdeel van de applicatie bevat ook een eigen header met daarin de naam van het onderdeel. Zo kan er altijd naar een onderdeel verwezen worden, bijvoorbeeld in een les.

2. Aantrekkelijkheid
   De felle kleuren van de blokken nodigt kinderen uit om met de blokken te experimenteren.

3. Simpel en duidelijk
   Doordat veel onderdelen (zoals het canvas, het tekst venster, etc...) in het begin van het project nog leeg zijn, ziet een nieuw project er relatief simpel uit en wordt de gebruiker niet direct afgeschrikt door een te grote hoeveelheid informatie. Hierdoor is het ook duidelijk met welke onderdelen men eerst moet werken om met het programma te werken.

5.3.2 Leerbaarheid

Het canvas bevindt zich altijd centraal op het scherm. De plaats waar men nieuwe blokken kan nemen (de blokkenlijst) links van het canvas en alle output bevindt zich rechts van het canvas. Zo heeft alles een logische plaats en weet een gebruiker snel waar hij/zij iets kan terugvinden. Ook zorgen de gemaakte lessen ervoor dat de gebruiker niet alles zelf hoeft uit te zoeken. Wanneer de pagina voor het eerst wordt ingeladen, zal er een rondleiding gegeven worden.

5.3.3 Zoeken in de blokkenlijst

De blokkenlijst is opgedeeld in verschillende categorieën. Bovenaan bevindt zich de basis-categorie die nieuwe gebruikers snel op weg helpt. Gebruikers met meer ervaring kunnen tussen de verschillende categorieën gaan zoeken. De kleur van de categorie wordt weergegeven langs de naam van de categorie, zo kan een gebruiker ook snel een blok terugzoeken aan de hand van zijn kleur, zonder de categorie zelfs te openen.

5.3.4 Blokken vinden op het canvas

Rechts bovenaan het canvas bevindt zich een klein overzicht. Dit is een kleinere versie van het hele canvas. Kleinere versies van alle blokken op het canvas worden hier afgebeeld. Ook is er een vierkant te zien wat het deel van het canvas, dat zichtbaar is voor de gebruiker, voorstelt. Met dit overzicht kan de gebruiker blokken terugvinden over heel het canvas.

5.3.5 Robuustheid

In de evaluatiecriteria staat geschreven dat een silhouet van het blok zichtbaar zou zijn wanneer men een blok op een bepaalde plaats kan plaatsen. Dit klopt voor het verbinden van blokken aan een attach, enclose en connect component. Ook voor het invoegen van een blok in een input component zal er een silhouet verschijnen. Het verbinden van blokken gebeurt ook via de puzzelstukken op het blok. Zo moet de gebruiker de blokken verbinden op een logische manier.

5.4 Evaluatie op basis van criteria: extra 's

Eens alle vereisten gehaald zouden zijn, zou er aan de volgende extra's gewerkt worden:

1. Functies aanmaken
2. Output met behulp van geluid
3. Blokken voor het ondersteunen van lijsten en dictionaries
4. Ondersteuning van de Engelse taal

Uit deze lijst is enkel het aanmaken en gebruik van functies geïmplementeerd in de finale applicatie. Dit gebeurt aan de hand van een functie-blok.

6 Aanpassingen ten opzichte van het origineel eindproduct

Doordat de eerste versie van het project niet voldeed aan de verwachtingen van het onderwijsteam, werd het project herkanst. In deze sectie zullen de veranderingen besproken worden ten opzichte van het origineel eindproduct. Voor de herkansing zijn er een aantal oudere features aangepast en een aantal nieuwere toegevoegd.

6.1 Aangepaste features

In deze sectie worden de features besproken die aangepast zijn ten opzichte van het origineel eindproduct.

6.1.1 Responsive web-design

De originele applicatie werkte niet responsive. Wanneer het venster van de browser groter of kleiner gemaakt werd, paste de applicatie zich niet aan. Tijdens lessen paste de aanduiding van een element van de applicatie zich ook niet aan en nam de donkere achtergrond niet het hele venster in beslag voor kleinere vensters. Al deze problemen zijn opgelost. De applicatie past zich nu aan aan de grootte van het venster. Het canvas wordt groter bij grotere vensters en de applicatie zal mee scalen bij kleinere vensters.

6.1.2 Verbinden van blokken

In de originele applicatie moest men blokken verbinden door met de muiscursor over het gebied van het gewenste attach of enclose component te bewegen. De puzzelstuk vormen van het blok dienden dus enkel om weer te geven welke blokken met elkaar verbonden konden worden. Dit werkte visueel minder goed. Doordat de puzzelstuk vormen mooi in elkaar pasten, voelde het voor de gebruiker veel intuïtiever aan om de blokken te verbinden met behulp van de positie van de puzzelstuk vormen.
Dit is geïmplementeerd door alle mouse-events tijdens het verslepen van een blok op te vangen en afhankelijk van de componenten die het blok bezit een nieuw mouse-event te sturen met als locatie de plaats van dit component. Bij het loslaten van de linkermuisknop tijdens het verslepen van een blok, zal er bijvoorbeeld ter hoogte van ieder (API-)attach en (API-)connect component van het blok een mouserelease-event verstuurd worden, in plaats van aan de positie van de muiscursor.

In de vorige versie van de applicatie werd het silhouet van het attach of enclose component altijd zichtbaar wanneer men met de muis over het gebied bewoog. Dit werd aangepast, zodat het silhouet enkel zichtbaar wordt wanneer men een blok, dat op die plaats verbonden kan worden, over het gebied versleept. Dit principe is ook toegepast bij het silhouet van de input componenten.

Puzzelstukken kunnen in deze versie van de applicatie ook langs boven verbonden worden met behulp van het connect component, dit was eerder nog niet mogelijk. Door het verbinden van blokken aan zowel de bovenkant als de onderkant van een blok toe te laten, voelt het verbinden van blokken veel intuïtiever aan.

6.1.3 Veranderingen in de code

De code bezit deze keer over veel meer commentaar. Tijdens het programmeren werd aan nieuwe code onmiddellijk commentaar toegevoegd, maar ook oude code werd nagekeken. Aan oude code werd commentaar toegevoegd wanneer er grote onduidelijke stukken code waren. Ook is er op verschillende plaatsen debug/error logging toegevoegd. Dit was vooral handig in de projectcontroller, daar passeren namelijk de belangrijkste acties.
Er werd ook een folderstructuur toegepast om het terugvinden van bepaalde bestanden te vergemakkelijken.
Omdat sommige klassen groot en onoverzichtelijk waren, werden deze klassen opgesplitst in meerdere klassen. De volgende klassen zijn opgesplitst:

IdeApp In het Vue component IdeApp werden alle andere componenten apart toegevoegd. Er is nu een apart Vue component voor de header, één dat alle actieknoppen bevat en één waarin de output componenten aan toegevoegd worden.

Blocklist De blokkenlijst was origineel een lijst waar alle blokken aan toegevoegd werden. Nu is dit een lijst waar categoriën aan toegevoegd worden, die zelf op hun beurt alle blokken in de categorie toevoegen.

Tutorial Het tutorial Vue component is opgesplitst in 3 componenten: tutorialhighlight (dat het element aanduidt), tutorialhint (dat het tekstballon toont) en tutorialsteplist (dat de lijst van reeds voltooide stappen bijhoudt).
Ook het component voor het interface, waar men lessen kan aanmaken, is opgesplitst. Alle velden waar men input vraagt, zijn aparte componenten geworden.

Scrollbar De 2 scrollbars (scrollbarX en scrollbarY) zijn nu gecombineerd naar één klasse. Dit was mogelijk doordat beide scrollbars dezelfde logica bevatten, het enige verschil tussen de klassen was de richting van de scrollbar. Iedere scrollbar is nu een element van de klasse scrollbar, waar men de richting van de scrollbar meegeeft in de constructor van de klasse.

ProjectController De projectcontroller werd eerst gebruikt als de enige controller. Om het werk in de controller te verdelen, zijn er nu 5 nieuwe controllers toegevoegd. De projectcontroller houdt al deze controllers bij.
De actioncontroller houdt de undo- en redo-stack bij, de blockcontroller voert alle operaties met betrekking tot blokken uit, de dragcontroller houdt het blok bij dat de gebruiker versleept, de execcontroller voert het programma uit bij een druk op de start- of stap-knop en de errorcontroller houdt alle fouten en waarschuwingen bij.

6.1.4 Start- en stap-knop

In het originele project werkten de start- en stap-knop niet op dezelfde manier, alhoewel dit wel verwacht werd. Bij een groot aantal programma's, werd bij een druk op de stap-knop ieder programma gelijktijdig uitgevoerd. Bij een druk op de start-knop was er echter duidelijk te zien dat de programma's niet gelijktijdig uitgevoerd werden. Dit probleem werd opgelost door de werking van de start-knop hetzelfde algoritme te laten gebruiken als de werking van de stap-knop. De werking van de start-knop drukt nu na een, regelbaar, tijdsinterval de stap-knop in. Hierdoor werken beide knoppen, in deze versie van het project, op dezelfde manier.

6.1.5 Undo-functionaliteit bij het invoeren van tekst in een input component

In input componenten met een witte achtergrond, kan er tekst getypt worden. Wanneer men na het typen van een tekst op de undo-knop drukt, zal de inhoud van het input component terug veranderen naar de vorige inhoud van het component.
De tekst kan altijd terug gebracht worden door de redo-knop in te drukken.

6.1.6 Wachtwoord systeem

Om de applicatie kindvriendelijker te maken, werd er een nieuw wachtwoord systeem geïmplementeerd. Men kan nu ook inloggen met behulp van een patroon. Een dergelijk wachtwoord is minder veilig, maar veel gebruiksvriendelijker. Om andere belangrijkere accounts, zoals het account van een beheerder, veiliger te maken, is er nog steeds de optie om een standaard wachtwoord (dat gebruik maakt van tekst) te gebruiken.

6.1.7 Functies en variabelen

Functies en variabelen kunnen niet meer zo makkelijk verwijderd worden. Eerst moeten alle verwijzingen naar de functie of de variabele verwijderd worden van het canvas. Wanneer er nog 'variabele'-blokken of blokken waarmee men een functie kan aanroepen op het canvas staan, zal het verwijderen van de variabele of de functie tegengehouden worden. In dit geval zal er een waarschuwing gegeven worden en zullen de overige blokken tijdelijk oplichten, zodat men deze gemakkelijker kan terugvinden.
De implementatie van deze werking zorgde voor minder problemen tijdens het uitvoeren van programma's. Het is dus niet meer mogelijk om blokken op het canvas te gebruiken die verwijzen naar een niet-bestaande variabele of functie.

6.1.8 Lessen

Lessen (eerder tutorials genoemd) hebben, buiten de naamverandering, een aantal aanpassingen gekregen. Bij het starten van een les, wordt het canvas nu leeggemaakt. Zo kan de les niet gestoord worden door reeds bestaande blokken.
Het tekstballon zal nu niet meer over het aangeduide element staan, maar zal zich er altijd naast bevinden. De foute plaatsing van het tekstballon stoorde vaak in een les.
De achtergrond van de applicatie zal nu de hele les donker blijven. Wanneer deze achtergrond verdween, leek het alsof de les geëindigd was, terwijl dit niet altijd het geval was.
Tijdens lessen wordt nu getest of de correcte actie uitgevoerd wordt. Wanneer dit niet gebeurt, zal de incorrecte actie ongedaan gemaakt worden en zal de volgende stap nog niet starten.
Het is nu niet meer mogelijk om in de blokkenlijst te scrollen tijdens lessen, zo staat de aanduiding van een blok altijd correct.
Het typen van een tekst in een input component kan nu als actie aangeduid worden. Er wordt op deze input getest of de actie reeds uitgevoerd is. In de vorige versie van het project was dit niet mogelijk, waardoor de gebruiker de volgende stap kon starten wanneer men nog niets in het component getypt had.
Wanneer een gebruiker de applicatie voor de eerste keer opent, wordt er een rondleiding les gegeven. Deze les duidt alle onderdelen van de applicatie kort aan en geeft een korte uitleg over de onderdelen. Op deze manier geraakt de gebruiker sneller bekend met de applicatie.
Ten slotte krijgt de gebruiker aan het einde van een les een positievere feedback. In de vorige versie van het project eindigde een les door in de laatste stap op de knop 'stoppen' te drukken. Nu wordt de gebruiker gefeliciteerd met confetti en wordt deze doorverwezen naar andere lessen.

6.1.9 Kleine aanpassingen

Er zijn ook nog een aantal kleinere aanpassingen gemaakt. Deze zullen hier kort opgelijst worden:

- Alle knoppen hebben hetzelfde uiterlijk gekregen.
- Het overzicht van het canvas updatete pas wanneer men het canvas losliet na het verslepen van het canvas. Nu wordt dit onmiddellijk geupdate.
- Het toevoegen van afbeeldingen aan het afbeeldingen venster zorgde voor een opstapeling van alle afbeeldingen. Bij het toevoegen wordt er nu achterliggend op de herstel-knop gedrukt zodat de afbeeldingen automatisch naast elkaar komen te staan.
- De applicatie werkt volledig in het Nederlands. Alle Engelse woorden zijn uit de applicatie gehaald.

- JSONWebtoken gebruikt niet meer de default key.
- Het lettertype van de applicatie is veranderd en is dus niet meer Times New Roman.
- Het icoon van de pagina is veranderd naar de afbeelding van de dinosaurus uit het afbeeldingen venster.
- De tekst geschreven in blokken heeft een dunne zwarte rand gekregen, zodat de tekst beter leesbaar is.

6.2 Nieuwe features

De herkansing van het project bracht ook meteen een aantal nieuwe features met zich mee. De volgende nieuwe features zijn toegevoegd om de applicatie te verbeteren.

6.2.1 Nieuwe start-blokken

Er zijn twee nieuwe start-blokken toegevoegd. Een blok dat start wanneer een gekozen toets op het toetsenbord wordt ingedrukt en een blok dat start wanneer er op een gekozen afbeelding geklikt wordt. Deze nieuwe blokken laten toe om op hetzelfde canvas verschillende programma's te creëren, zonder dat deze altijd tegelijk uitgevoerd worden.

6.2.2 'Opslaan', 'opslaan als' en 'herstel canvas'

Origineel was er enkel een 'opslaan'-knop met de werking van een 'opslaan als'-knop geïmplementeerd. Er was geen manier om een project te overschrijven met een nieuwere versie.
Nu kan men met de 'opslaan als'-knop een nieuw project opslaan zoals verwacht. De nieuwe 'opslaan'-knop, die toegevoegd is in de header, zal enkel zichtbaar zijn wanneer men een reeds opgeslagen project inlaad en laat toe om ingeladen projecten te overschrijven.
Er is ook een knop 'herstel canvas' toegevoegd aan de header. Deze knop opent een nieuw leeg project en maakt het canvas dus terug leeg. In het originele project moest men de pagina 'refreshen' om een leeg canvas te krijgen zonder alle blokken manueel te verwijderen.

6.2.3 Fout, waarschuwing en succes meldingen

Fout, waarschuwing en succes meldingen werden eerst als kleine pop-ups langs het blok weergegeven. Het probleem met deze manier was dat de pop-ups makkelijk opstapelden, het niet duidelijk was bij welk blok het hoorde en ze één voor één gesloten moesten worden.
Om dit probleem op te lossen, is het error component aangemaakt. Het error component hangt de meldingen aan het blok, zo zal het altijd duidelijk zijn bij welke blok de melding hoort. Ook moeten de meldingen niet meer manueel gesloten worden. De melding is in eerste instantie enkel zichtbaar wanneer men met de muis over het component beweegt. De melding zal ook vanzelf verdwijnen uit het component wanneer de fout die de melding creëerde opgelost is.
Het blok dat de melding bevat zal niet altijd zichtbaar zijn op het canvas. Hierdoor zal er een pop-up verschijnen links boven het canvas wanneer een melding gegeven wordt.
Ook is er een fout en waarschuwing overzicht toegevoegd centraal bovenaan het canvas. Hier kan men altijd het aantal fouten en waarschuwingen op het canvas zien. Ten slotte kan men ook op dit overzicht drukken om alle informatie over de fouten en waarschuwingen op het canvas te bekijken.

7 Mogelijke uitbreidingen

Lijsten en dictionaries In het project zijn variabelen geïmplementeerd. Deze variabelen kunnen enkel gebruikt worden als string, integer of boolean waardes. Een mogelijke uitbreiding zou zijn om lijsten en dictionaries aan te kunnen maken.
Voor lijsten zou een nieuw soort blok aangemaakt moeten worden. Ook zouden er een aantal nieuwe blokken toegevoegd moeten worden die operaties kunnen uitvoeren op lijsten.

Ondersteuning van de Engelse taal Alle tekst die men kan lezen in de applicatie staat momenteel los in verschillende bestanden in de code geschreven. Hiervoor zou beter een JSON-bestand gemaakt zijn in het begin van het project. Zo zou er simpelweg een nieuwe JSON-bestand moeten toegevoegd worden om een andere taal te kunnen ondersteunen. Om dit te implementeren zou men dus simpelweg de tekst in alle bestanden in één JSON-bestand moeten verzamelen.

Output in de vorm van geluiden Een andere mogelijke uitbreiding zou output met behulp van geluid zijn. Hiervoor zou een nieuw blok aangemaakt moeten worden met een drop-down menu (choice component). In dit menu zouden een aantal standaard geluid opties kunnen zitten. Wanneer dit blok wordt uitgevoerd, zal het gekozen geluid afgespeeld worden. Het blok zou op dezelfde manier werken als het afbeeldingen venster en het tekst venster. Het blok zou een signaal sturen, waarin de naam van het geluid wordt meegegeven. Dit signaal wordt dan opgevangen door een nieuwe geluid-klasse. Deze klasse zal het geluid dan afspelen en zou ook ergens visuele feedback kunnen geven dat er een geluid wordt afgespeeld.

Verplaatsen van onderdelen De lay-out van de applicatie staat momenteel vast. Als mogelijke uitbreiding zou de gebruiker de onderdelen van de applicatie, zoals het canvas en het tekst venster, kunnen verplaatsen. Dit zou een redelijk moeilijke uitbreiding zijn. Hiervoor moeten een aantal posities vooraf gedefinieerd worden waar onderdelen geplaatst kunnen worden. Daarbovenop zouden andere onderdelen bij het vastnemen en het plaatsen van een onderdeel automatisch correct moeten verschuiven.

Modus voor een geavanceerde gebruiker Doordat een deel blokken een complexere werking hebben, zou het nuttig zijn om twee modi te voorzien in de applicatie. Eén voor de normale gebruiker en één voor een geavanceerde gebruiker. De normale gebruiker zou dan een minder uitgebreide blokkenlijst te zien krijgen zonder bijvoorbeeld functies en API-blokken. Ook het variabelen venster zou dan afgeschermd worden. De geavanceerde gebruiker zou dan wel alles te zien krijgen.

Ondersteuning voor mensen met een visuele beperking De applicatie bestaat op dit moment enkel uit één kleuren-thema. Het zou mogelijk zijn om de gebruiker de optie te geven om een thema zelf te kiezen. Zo zou er een donkerder thema geïmplementeerd kunnen worden, of een thema voor mensen met kleurenblindheid. Voor deze mensen zou dit een mooie uitbreiding zijn, want de blokken hebben momenteel veel verschillende kleuren. Op het moment is het moeilijk voor mensen met kleurenblindheid om de categorieën van blokken te onderscheiden. Dit kan het gebruik van het programma moeilijker maken voor deze groep mensen.
De kleuren worden in de applicatie opgeslagen in een JSON-bestand. Om deze thema's te realiseren, zou er een knop moeten toegevoegd worden die dit JSON-bestand omwisselt voor een nieuw bestand met nieuwe kleuren.

Afbeeldingen uploaden Het afbeeldingen venster bevat een aantal vooraf ingeladen afbeeldingen, die hetzelfde zijn voor iedereen. Als mogelijke uitbreiding zou de gebruiker zelf hier afbeeldingen aan kunnen toevoegen. Er zou dan een 'upload'-knop aan het venster worden toegevoegd. Deze knop zou dan een pop-up geven, waar men een afbeelding kan uploaden en waar een naam aan deze afbeelding gekoppeld kan worden. Deze naam zal dan tussen de selectie vakjes van het afbeeldingen venster verschijnen. Als laatste zouden deze afbeeldingen ook opgeslagen moeten worden bij het opslaan van een project, zodat bij het inladen van het project de afbeeldingen weer tussen de selectie vakjes in het afbeeldingen venster staan.

Uitleg over blokken De werking van een blok is niet altijd onmiddellijk duidelijk. Als uitbreiding zou er meer uitleg over een blok gegeven kunnen worden wanneer men met de rechtermuisknop op het blok klikt. Er zou dan een kleine pop-up tevoorschijn kunnen komen met een korte uitleg over het blok en al zijn onderdelen. Omdat deze uitleg voor ieder blok verschilt, zal elk blok een aparte uitleg moeten krijgen.

Een programma pauzeren Momenteel is er geen optie om een programma te pauzeren tijdens uitvoering. Dit zou geïmplementeerd kunnen worden door een pauze-functionaliteit aan de start-knop toe te voegen. Wanneer men dan op deze knop drukt, zou oftewel een programma beginnen met uitvoeren; een programma dat uitgevoerd wordt, pauzeren; of een gepauzeerd programma hervat worden.

'stap terug'-knop Met een 'stap terug'-knop zou een gebruiker een stap terug kunnen zetten, wanneer de gebruiker door een programma aan het stappen is. Zo zou men code live kunnen aanpassen wanneer men merkt dat het programma niet werkt zoals gewenst.

8 Handleiding

De handleiding bevindt zich in het bestand *psopv_2019-2020_handleiding_groep_5.pdf*.

9 Reflectie

De reflectie van het project bevindt zich in het bestand *psopv_2019-2020_reflectie_groep_5.pdf*.