Spiffy Sounds: A Making Of

Abe Heemskerk, Robin Immel, Jeremy Guijt

25 juni 2015

1 Voorwoord

Dit verslag is opgesteld voor de eindbeoordeling van onze app, Spiffy Sounds, welke we gemaakt hebben voor het vak Research & Development 1. We gaan in dit document achtereenvolgens in op waarom we deze app gemaakt hebben, welke functionaliteit de app heeft, en op welke manier de app ontworpen is. Aan het eind van dit verslag vind u onze reflectie op het maken van de app, met dingen die goed en niet goed gingen en wat dingen zouden zijn die we in het vervolg anders zouden doen.

2 Beschrijving

2.1 Inleiding

Veel culturen, subculturen en vriendengroepen hebben hun 'eigen' humor. Ze zijn bekend met dezelfde dingen die populair zijn, filmpjes op internet, etc. Daarom kan het leuk zijn bekende geluiden af te spelen op een daarvoor gepast moment. Voor in radiostudio's bestaat er al tijden een apparaat dat hierin faciliteert: een soundboard. Met de opkomst van smartphones en apps is er echter een grotere markt voor soundboards ontstaan. Software-versies kunnen nu gemakkelijk vanuit de App Store of Play Store gedownload worden. Wij vonden echter dat deze apps erg veel op elkaar lijken. Ze mogen dan wel verschillende geluiden of een andere layout hebben, veelal is de functionaliteit hetzelfde. Er is een scherm waarop de gebruiker knoppen ziet, en bij het aanklikken van een knop wordt een geluidsfragment afgespeeld. Niets meer en niets minder. Hier is dus ruimte voor verbetering en wij wilden voor die verbetering gaan zorgen.

2.2 Productverantwoording

Huidige soundboard-apps doen wat ze moeten doen, en dat doen ze goed genoeg. Als je op een knop drukt wordt je geluidsfragment afgespeeld. Soms duurt het echter lang om je telefoon te pakken, de app op te starten en je gewilde geluidsfragment op te zoeken. Ook is de verzameling geluiden op de telefoon vaak erg statisch. Je download de app met alle geluiden die de app ondersteunt,

zonder de mogelijkheid geluiden toe te voegen. Tenslotte laat het design van apps nog weleens te wensen over. Niet alleen zijn de knoppen niet even mooi, maar soms heeft de gebruiker ook geen idee waarom een bepaalde tekst bij een geluidsfragment is gekozen. Dit doet afbreuk aan het makkelijk kunnen vinden van geluiden en aan de gebruikerservaring. Deze drie punten zijn dan ook de drie punten waarbij we het meeste ruimte zagen voor verbetering. Ze laten zich samenvatten in drie concepten: Snelheid van gebruik, dynamische content en customisatie van het design. Bij elk van deze verbeterpunten hadden we specifieke ideeën waarmee we onze app innovatiever konden maken dan de concurrentie.

- Snelheid van gebruik

  Het moment waarop bedacht wordt dat een geluidsfragment kan worden afgespeeld, totdat het geluidsfragment te horen is moet zo kort mogelijk zijn. Hierom willen we zorgen dat er zo min mogelijk handelingen nodig zijn voor het afspelen van een geluid en dat de geluiden in de app gemakkelijk vindbaar zijn.

- Dynamische content

  De gebruiker moet in staat zijn naar believen geluiden aan de app toe te voegen of van te verwijderen.

- Customisatie van het design

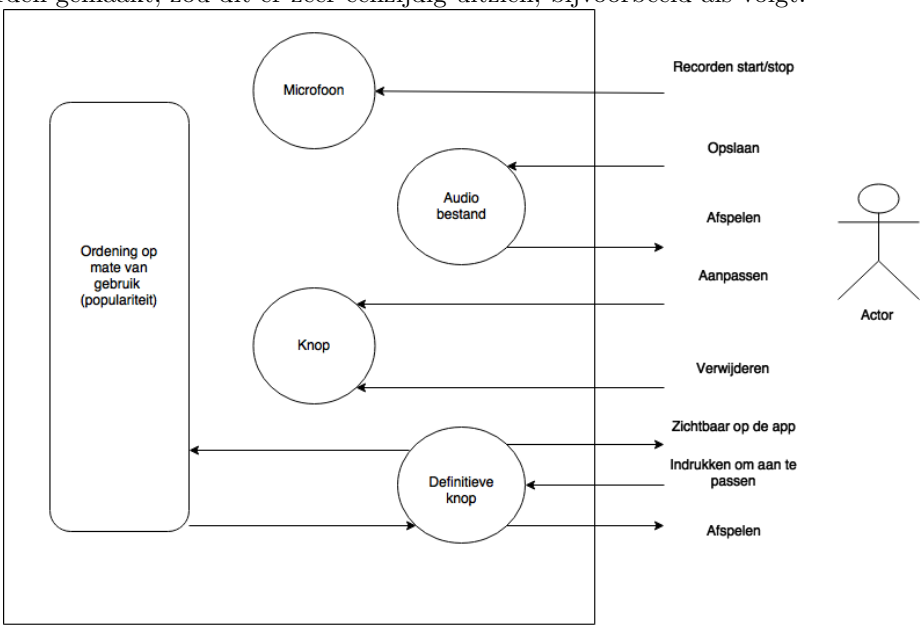
  De gebruiker moet in staat zijn de kleuren en tekst op knoppen aan te passen.

2.3 Specificaties

De volgende lijst geeft aan welke functionaliteit wij in de app wilden hebben. Helaas hebben we niet alle gewenste functionaliteit kunnen implementeren, dus dit is achter de betreffende functionaliteit aangegeven.

- Afspelen van geluiden
- Ordening van geluiden, op basis van populariteit
- Geluiden afspelen vanaf het lockscreen of het menu (niet geïmplementeerd)
- Geluiden opnemen met de microfoon, deze geluiden testen en opslaan
- Kleuren van knoppen aanpassen
- Tekst behorende bij knoppen aanpassen
- Knoppen verwijderen

Bij onze app is een use case model niet van toepassing. Dit komt omdat onze app geen interactie heeft tussen verschillende personen. Elke handeling die wordt uitgevoerd krijgt, bijna, altijd een directe respons naar buiten. Alles zou namelijk intern direct worden aangepast. Als er wel een use case model zou worden gemaakt, zou dit er zeer eenzijdig uitzien, bijvoorbeeld als volgt:



3 Ontwerp

3.1 Globaal ontwerp

De app is onderverdeeld in drie pagina's, ook wel *activities* genoemd. Met het eerste deel krijgt de gebruiker het meeste te maken. Deze 'main'-activity bied de gebruiker een overzicht van alle knoppen en speelt de geluiden af als erop geklikt wordt. Vanuit deze activity kan de gebruiker ook naar de andere activities gaan. Men gaat naar de 'record'-activity door in de *Action Bar* op record te klikken en men gaat naar de 'customize'-activitie door een knop lang ingedrukt te houden.

In de customize-activity kan met de kleur van de knop veranderen, door op een van de 8 preset kleuren te drukken. Deze kleuren zijn allemaal zo uitgekozen zodat ze bij de achtergrondkleur passen en dus nooit uit de toon vallen. Ook de naam van de knop kan hier veranderd worden, zodat hij beter bij het geluidsfragment past en voor de gebruiker beter terug te vinden is. Tenslotte kan de gebruiker in deze activity ook een knop verwijderen, door op de 'delete'-knop te drukken.

In de 'record'-activity kan iemand tenslotte zijn eigen geluiden opnemen. De 'record'-knop is zo gemaakt dat het voor de gebruiker duidelijk is wanneer hij opneemt en wanneer niet. Tevens is hier ook veel van de functionaliteit van de

customize pagina te vinden, zodat al voordat de knop in het menu verschijnt het design van de knop aanpasbaar is.

Er is natuurlijk ook een deel van de app dat voor de gebruiker niet zichtbaar is. Het belangrijkste onderdeel waar de gebruiker niet direct mee te maken krijgt is de database. Hierin slaan we alle informatie op over de knoppen, zoals naam, kleur, locatie van het geluidsfragment en het aantal maal dat het geluid afgespeeld is.

3.2 Detailontwerp

Onze app bestaat uit de drie bovengenoemde activities en gebruikt een permission om audio op te kunnen nemen met de microfoon.

In de codebase van de app zijn 10 classes te vinden.

1. AppUtils
   Deze klasse heeft voornamelijk utiliteitsfuncties voor de rest van de app.

   copy(InputStream input, OutputStream output)
   Kopieert de data in de inputstream naar de data in de outputstream. Deze functie is belangrijk voor het kopieëren van raw resources naar het interne geheugen.

   copySoundsIfNecessary(Context c)
   Deze functie controleert van elk standaard geluidsfragment of het naar dit geluidsfragment interne geheugen gekopieerd moet worden. In principe wordt dit alleen gedaan bij het opstarten voor de eerste keer en wanneer de data van de app verwijderd wordt.

2. ButtonDatabase
   Deze klasse verzorgt al het contact tussen de app en de database. Het is een implementatie van de SQLiteHelper interface. Alle queries zijn String constanten waar de waarden voor de query (zoals bv een ID) later in worden geplaatst.

   onCreate(SQLiteDatabase db)
   Voert een query uit om de database te maken en creëert records voor alle standaardknoppen in de database.

   insertButton(String name, String filename, String color, SQLiteDatabase db)

   Voegt een knop toe a.d.h.v. de gegeven data.

   getButtons()
   Haalt alle knoppen op uit de database en zet ze in een ArrayList. Deze functie wordt gebruikt voor het tonen van de main-activity.

3. ColorButton
   Deze klasse zorgt er vooral voor dat de knoppen juist worden afgebeeld in het raster.

ColorButton(Context c, String btn_text, String btn_color)
Zorgt ervoor dat een drawable van een button wordt aangeroepen. Hierbij worden ook de tekst en buttonkleur bepaald.

setColor(String button_color)
Verandert de kleur van een knop door de achtergrondkleur aan te passen aan een van de vaste kleuren.

getColor()
Hierin zijn de vaste kleuren opgeslagen dmv rgb-waardes. We hebben hierbij gekozen voor 8 verschillende kleuren.

getContrastColor()
Door de contrastkleuren van de knoppen te onderscheiden in zwart en wit, zorgt deze functie ervoor dat de tekst op de knoppen leesbaar wordt afgebeeld.

4. CustomizeOnLongClick
Als je een button lang indrukt, dan zorgt deze onLongClickListener ervoor dat met de juiste buttonID het customisatiescherm wordt aangeroepen. Zo zal de gebruiker in staat worden gesteld om de knoppen aan te passen naar zijn/haar smaak.

5. PlaySoundOnClick
Deze klasse laat de geluiden afspelen door de app als er op een knop wordt gedrukt. Dit wordt gedaan door een onClickListener te implementeren, waardoor de een klik in de view wordt gelinkt aan een intent.

onClick(View v)
Door een (korte) klik van de gebruiker zal de mediaplayer worden aangeroepen. Dit is een ingebouwde functie van android waardoor het mogelijk wordt om geluiden af te spelen. Door het gebruik van deze mediaplayer zijn er een paar basisfuncties hiervan die het afspelen van geluiden makkelijker maakt. Zo kunnen we hier bepalen welk bestand moet worden afgespeeld en dat er voor onze app geen geluiden tegelijk afgespeeld mogen worden.

6. RecordScreen
In deze activity is het mogelijk om geluiden op te nemen door een druk op de recordknop. De recording zal stoppen na een druk op dezelfde knop of na een tijdsbestek van tien seconden, dit zodat de grootte van de bestanden niet te groot wordt. Na het recorden zal de gebruiker in staat gesteld worden een kleur en naam toe te kennen aan het opgenomen geluid. Als alles naar wens is, zal de gebruiker door een klik op de 'save'-knop de knop kunnen toevoegen aan de database.

onCreate(Bundle savedInstanceState)
In de onCreate van deze klasse wordt gezorgd voor de koppeling tussen de XML van de activity en de functies die aanwezig zijn. Zo

worden de 'record'-button en de review d.m.v. van een drawable in de activity getekend in de juiste grootte en op de juiste plaats. Ook worden hierop onClickListeners gezet waardoor de record en playfuncties aan worden geroepen. Het (dynamisch) aanpassen van de tekst op de previewbutton wordt eveneens in de onCreate geregeld.

recordOnClick(View view)

Het aanpassen van de recordknop wordt in deze functie geregeld. Als er één keer op de knop wordt gedrukt zal de knop "recording" aangeven, zodat de gebruiker weet dat de actie gelukt is. Ook zal de knop in dit geval ervoor zorgen dat het recorden gestart wordt door recordStart aan te roepen. Als er vervolgens weer op de knop geduwd wordt, zal de recordknop weer naar de eerste staat verspringen. Het recorden wordt gestopt door recordStop aan te roepen.

recordStart()

Deze functie is een basisfunctie om de mediaplayer aan te roepen. De mediaplayer heeft meerdere states, waarbij functies gelimiteerd zijn aan bepaalde states. Hierdoor is de volgorde van het aanroepen hiervan erg belangrijk, doordat de volgorde van de states ook vaststaat. Als eerst zal de recorder een audiosource, een outputformaat en een encoder willen. Hierna kan er door een setMaxDuration-functie een maximale tijd worden ingesteld zonder dat de gebruiker hier invloed op heeft. Vervolgens moet er een outputfile worden aangegeven, waar wij voor een tijdelijke file hebben gekozen. Deze zal elke keer worden overschreven als de recordfunctie wordt aangeroepen waardoor er geen onnodig plaatsgebruik plaatsvindt.

7. SoundboardCustomize

   Deze activity lijkt erg veel op de 'record'-activity. Alle aanpasfuncties zijn hetzelfde, maar er is natuurlijk geen recordbutton aanwezig. Wel is er een deletebutton aanwezig die geluiden verwijdert en de buttons uit de database haalt.

   onCreate(Bundle savedInstanceState)

   Haalt de juiste gegevens op uit de database die horen bij de betreffende buttonid. Daarnaast zorgt de functie ervoor dat net als bij de recordactivity de UI wordt gekoppeld met de (aanpas)functies.

   deleteButton

   Verwijdert de button uit de database door de deleteButtonOnID aan te roepen.

8. SoundboardMain

   De 'main'-activity van onze app, waar alle geluidsfragmentknoppen te vinden zijn.

   onCreate(Bundle savedInstanceState)

   Deze functie zorgt voor het alles wat nodig is voor het opbouwen

van de activity. Er wordt een verbinding gelegd met de database, de mediaplayer wordt gemaakt en alle knoppen en hun listeners worden aan de activity toegevoegd.

9. SoundButton
   Een klasse die de buttondata uit de database voorstelt, zodat er op een makkelijke manier toegang tot deze data te krijgen is. Deze functie heeft naast de constructor alleen getters.

10. SoundButtonGridAdapter
    Deze klasse is de link tussen de lijst met alle knoppen en de XML van de 'main'-activity. De klasse is een implementatie van de BaseAdapter interface.

    getView(int position, View convertView, ViewGroup parent)
    Deze functie zet de data van een knop in de lijst om in een daadwerkelijke knop in de app. Ook wordt hij hier klikbaar gemaakt.

3.3 Ontwerpverantwoording

We hebben 2 designcases uitgelicht om te laten zien hoe we bij bepaalde designbeslissingen zijn gekomen.

Menu-activity vs Knoppen-activity
De eerste ontwerpkeuze die we toelichten gaat over of we een menu-activity wilden implementeren. Een gebruiker zou in dat geval eerst in een menu de keuze maken uit de 'knoppen'-activity (wat nu dus de 'main'-activity is), de 'record'-activity en de 'customize'-activity. Het andere geval was dat er geen menu zou zijn en dat de gebruiker direct in de 'knoppen'-activity zou komen. Vanaf hier zou het dan mogelijk moeten zijn om naar de 'record'- en 'customize'-activities te gaan.
We hebben de voors en tegens op een rijtje gezet. Bij gebruik van een menu zou het voor een gebruiker erg overzichtelijk zijn wat de functionaliteit is van de app, en hoe hij bij deze functionaliteit terechtkomt. Als we geen menu zouden gebruiken dan is de gebruiker sneller bij het geluid dat hij wilt afspelen. Dit laatste woog voor ons het zwaarst, omdat een van de doelen van het maken van deze app was dat de tijdsduur vanaf het pakken van de telefoon tot het afspelen van een geluid zo klein mogelijk moest zijn. Hiernaast denken we dat de functionaliteit nog goed genoeg blijkt uit de layout van de app, zoals de goed zichtbare 'record'-knop in de Action Bar.

Stoppen en starten van het opnemen De tweede ontwerpkeuze die we toelichten gaat over het starten en stoppen van het opnemen op de 'record'-activity. Hier waren ook meerdere mogelijkheden om het te regelen. We hadden er bijvoorbeeld voor kunnen kiezen om twee knoppen te maken, één voor het starten van het opnemen en één voor het stoppen hiervan. We hebben

uiteindelijk gekozen voor het gebruik van één knop voor beide functies en deze knop bij het aanklikken van kleur en tekst te laten veranderen, zodat het voor de gebruiker meteen duidelijk is waar hij moet klikken om het opnemen te stoppen. Ook hebben we de standaardkleur voor opnemen (rood) overgenomen voor deze knop, zodat het er voor de gebruiker ook erg intuïtief uitziet.

4 Reflectie

Over het algemeen zijn wij zeer tevreden over het behaalde eindresultaat. Het proces, om bij de uiteindelijke applicatie te komen, verliep moeizaam. We hadden niet erg goed gepland en moesten daardoor in de laatste maand het meeste werk doen. Toen we er uiteindelijk aan begonnen gingen we er wel echt voor. Er werden veel uren gemaakt tot in de late nacht. Het idee wat wij hadden was redelijk simpel. Een Soundboard is iets wat al meerdere malen gemaakt is, daardoor wilden wij dat stapje verder gaan. Hier liepen we tijdens het programmeren al tegen het eerste grote probleem aan. We kregen maar geen toegang tot de microfoon. Dit maakte het verder werken met de "gemaakte" audiobestanden moeilijk. Uiteindelijk bleek dat we een permission niet goed hadden geplaatst. Nadat dit probleem was opgelost kon er met meerdere dingen verder worden gewerkt. Zo kon er verder gegaan worden met het opnemen van het audiobestand en de layout van deze activity, maar kon er ook worden gewerkt aan het maken van een database om al deze aangemaakte bestanden in op te slaan. De database was één van de grotere projecten in deze app. Het was even zoeken over hoe de bestanden opgeslagen konden worden in het interne geheugen. De wat kleinere problemen waren hoe de aangemaakte knoppen zich zouden aanpassen als ze tussen de andere knoppen werden geplaatst en hoe de standen van alle knoppen naar het interne geheugen konden worden geladen. Dit laatste was nodig om ervoor te zorgen dat als de applicatie zou worden afgesloten, dat bij het opnieuw opstarten van de applicatie de knoppen op dezelfde plek zouden zitten. Verder verliep het maken van deze app vrij soepel. Natuurlijk was het niet één al rozengeur en maneschijn maar tegen veel problemen niet aan. Al met al zijn wij zeer tevreden met wat wij gemaakt hebben en zijn we trots op het eindproduct.