whitehorses

member of the human network

WHITEBOOK

Hey Google!

Maart 2019

Auteur:
Laurens van der Starre
INTEGRATIESPECIALIST

Inleiding

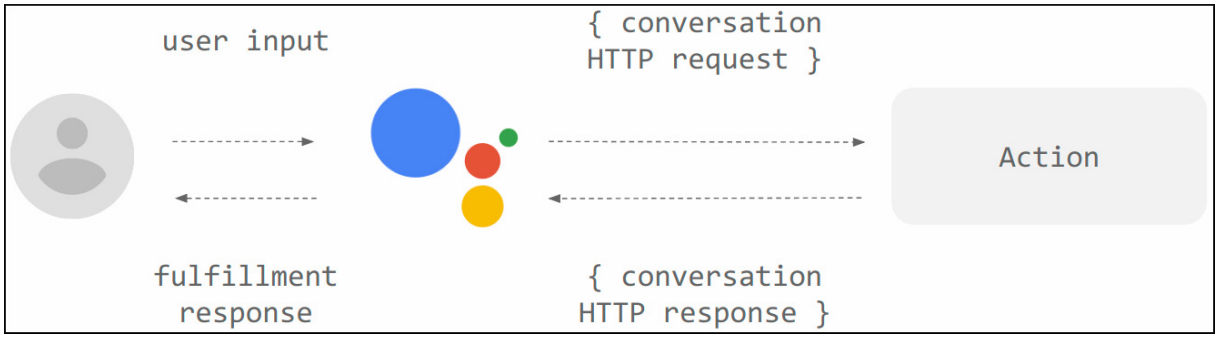
Onze huidige belevingswereld is vol met informatie dat onze aandacht probeert te vragen [1]. We zijn schermverslaafd, en alles heeft apps, valt ons lastig met notificaties en vereist interactie. Allerlei apps en services concurreren voor onze schermtijd en aandacht.
Golden Krishna schrijft terecht in *The best interface is no interface* [2] dat voor heel veel toepassingen een app niet de meest natuurlijke interface is. Immers, UX != UI.

Digitale assistenten

Een natuurlijke manier van interactie voor de mens is het gebruik van de gesproken taal. Digitale assistenten spelen daaropin. Nu is spraakbesturing niet nieuw. Maar eigenlijk was het altijd heel slecht. De gebruiker leerde al snel het apparaat te bedienen met zijn stem, in plaats van dat het apparaat de gebruiker echt begreep. Je was dus nog steeds bezig jezelf in alle bochten te wringen om gebruik te maken van deze interface.
De laatste jaren zijn de digitale assistenten van de grote technologiebedrijven in opkomst. Zo heeft Amazon Alexa, Apple Siri, Samsung Bixby, Huawei HiAssistant, Microsoft Cortana en Google natuurlijk de Google Assistant. Deze assistenten worden steeds beter in het begrijpen van de natuurlijke taal van de gebruiker waardoor de interactie op een natuurlijke manier plaatsvindt.
In dit Whitebook kijken we naar de Google Assistant, en hoe we deze assistent eenvoudig kunnen koppelen aan bestaande (backoffice) applicaties om zo services te ontsluiten via spraak in plaats van een gebruikelijke grafische userinterface in de vorm van een dedicated app of website.

Google Assistant

Uiteindelijk is de Google Assistant niets anders dan een interface waar de gebruiker in de vorm van een *conversatie* interactie mee heeft. Je wilt de Assistant een bepaalde actie (*Action*) laten uitvoeren. Dit kan volledig gesproken zijn, of in de vorm van geschreven tekst via de Google Home app, of diverse messaging apps (Facebook Messenger, Slack, Kik etc.). De Assistant begrijpt de gebruiker, vogelt uit wat het doel is (de zgn. *Intent*) en kan benodigde data (zgn. *Entities*) zelfstandig ontfutselen van deze gebruiker.
Op het moment dat de Intent helder is, en de benodigde Entities zijn bepaald, kan de verwerking beginnen en het antwoord aan de gebruiker worden teruggegeven. In dit Whitebook zullen we deze verwerking extern laten plaatsvinden, en zal de Assistant dus een call gaan doen naar een eigen externe API.



Figuur 1 Action on Google overview [3]

Intent

Zoals hierboven genoemd definieer je in de Assistant Actions. Een Action bestaat uit één of meerdere Intents. Een Intent is een doel of taak wat de gebruiker wil doen.
Verschillende Intents kunnen afhankelijk van elkaar zijn en elkaar opvolgen. Dit is dus een soort van conversatie die je kan hebben als gebruiker met de Assistant. De Assistant kan bijvoorbeeld vragen "*Kent u Whitehorses BV?*" en op basis van het antwoord kunnen verschillende Intents van toepassing zijn. Dit betekent dat Intents context nodig kunnen hebben, een soort van voorkennis (bepaalde data zoals bijvoorbeeld je naam, of locatie), maar ook context kunnen zetten. Als een context is vereist, dan kan de Intent niet geactiveerd worden voordat deze context is gezet. In het bovengenoemde voorbeeld zou een context kunnen zijn of de gebruiker Whitehorses BV kent, en op basis daarvan kunnen andere Intents deze context gebruiken.

Een Intent kan direct (hard coded) worden afgehandeld door de Assistant. Als een Intent bedoeld is om contactgegevens op te leveren voor bijvoorbeeld Whitehorses BV, dan kan dat direct worden teruggegeven als antwoord aan de gebruiker.
Helaas is het niet altijd zo simpel. In zulke gevallen kan de Assistant een *Fulfillment* starten. Deze Fulfillment is een API-call (HTTP REST) naar een externe API.

Fulfillments

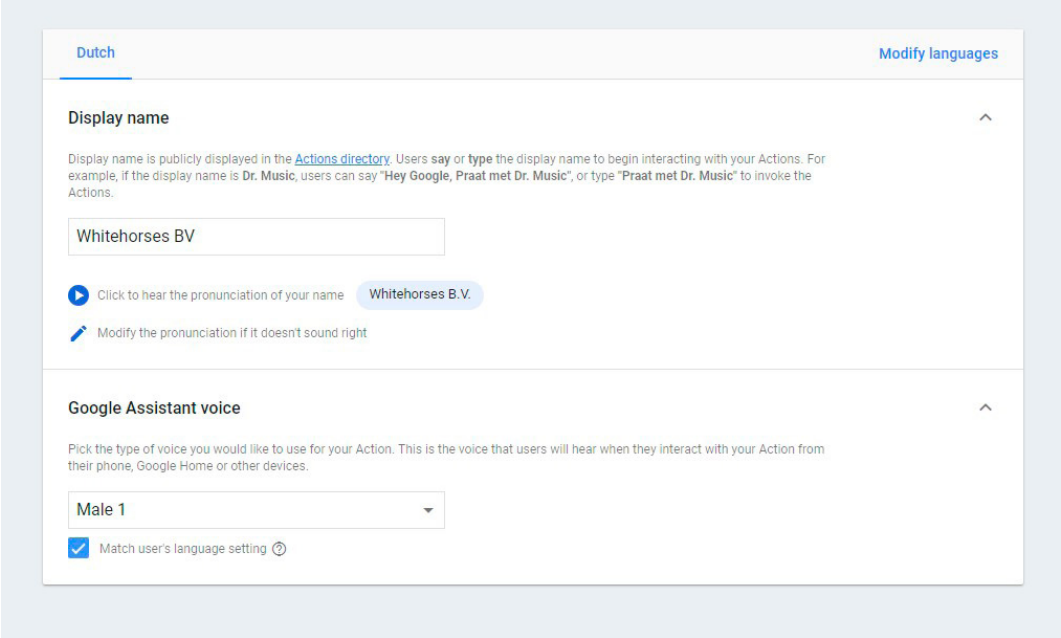
Zodra de Intent is geactiveerd, de benodigde Entities zijn bepaald, moet de Assistant een antwoord opleveren naar de gebruiker. Veelal moet er op basis van de Entities enige verwerking plaatsvinden. De Assistant kan dit niet. Dit gebeurt in de Fulfillment stap, waar de verwerking van de Intent extern gaat plaatsvinden (een externe API), en in het antwoord van deze API staat bepaald wat de Assistant dient te doen en dient te antwoorden naar de gebruiker. Dit antwoord bevat niet alleen de gesproken tekst (tekst-to-speech), maar kan ook bepaalde UI-elementen teruggeven. Dit laatste kan van toepassing zijn wanneer de gebruiker vanuit de Google Home app, of een andere client zoals Facebook Messenger de Assistant gebruikt. Ook kan de Assistant worden gestuurd om contexten te zetten of andere Intents te activeren. Google heeft hier de zgn. Diagflow Webhook Format voor opgesteld. Deze interface moet worden geïmplementeerd door de API.

De Google Assistant integreren

De mogelijkheid om Fulfillments in te zetten, en Google een Intent te laten verwerken in een backend systeem, maakt het een uitstekende kandidaat om opgenomen te worden in het (al bestaande) systeemlandschap. Google biedt een complete library aan voor NodeJS, maar je kan uiteraard ook je eigen technologie stack gebruiken. De aangeboden API die moet wel de Diagflow Webhook Format implementeren. Dit komt neer op een uitgebreid JSON inbound en outbound bericht.

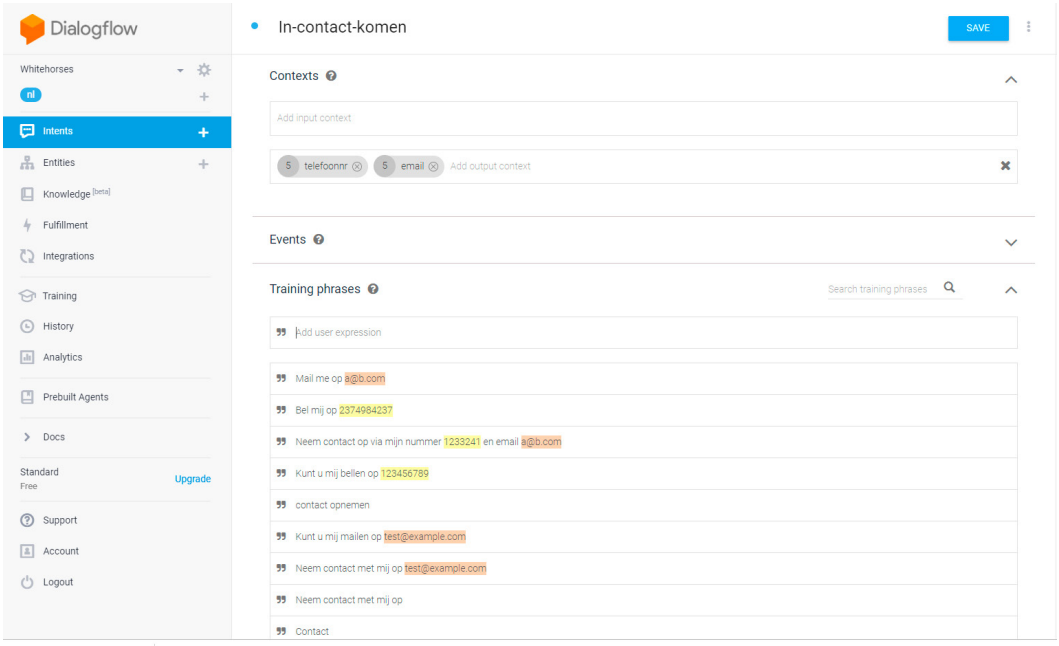
Praat met Whitehorses BV

Laten we kijken naar een simpel voorbeeld. Voor dit Whitebook heb ik bij wijze van voorbeeld de “Whitehorses BV” action geïmplementeerd. Een action maken voor de Google Assistant kan grotendeels allemaal in de web based console van Google.
Je maakt een zgn. Action on Google. Deze geef je een naam waarmee hij ook oproepbaar is via de Assistant.



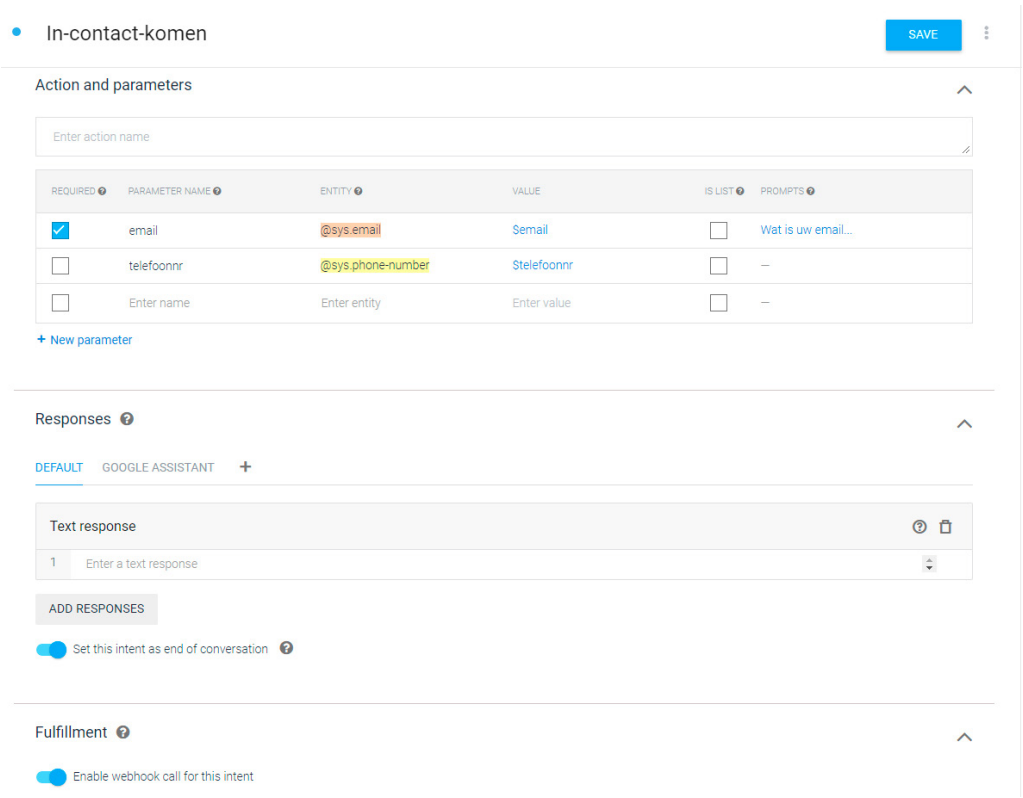
Figuur 2 Action on Google

De Action bevat Intents. Zoals eerdergenoemd wordt in de Intent bepaald wat de gebruiker wil. Er zijn verschillende dingen die geconfigureerd kunnen worden in de Intent.
Allereerst kan er worden aangegeven of de Intent nog afhankelijk is van enige context.
Een *input* context bevat data die gezet moet zijn *voordat* de Intent geactiveerd moet worden. Dit kan gezien worden als bijvoorbeeld een pre-conditie.
Een Intent kan ook contexten *zetten* (in de outbound context) om zo Entities te bewaren nadat ze zijn bepaald. Deze kunnen logischerwijs weer dienen als input context bij andere Intents.



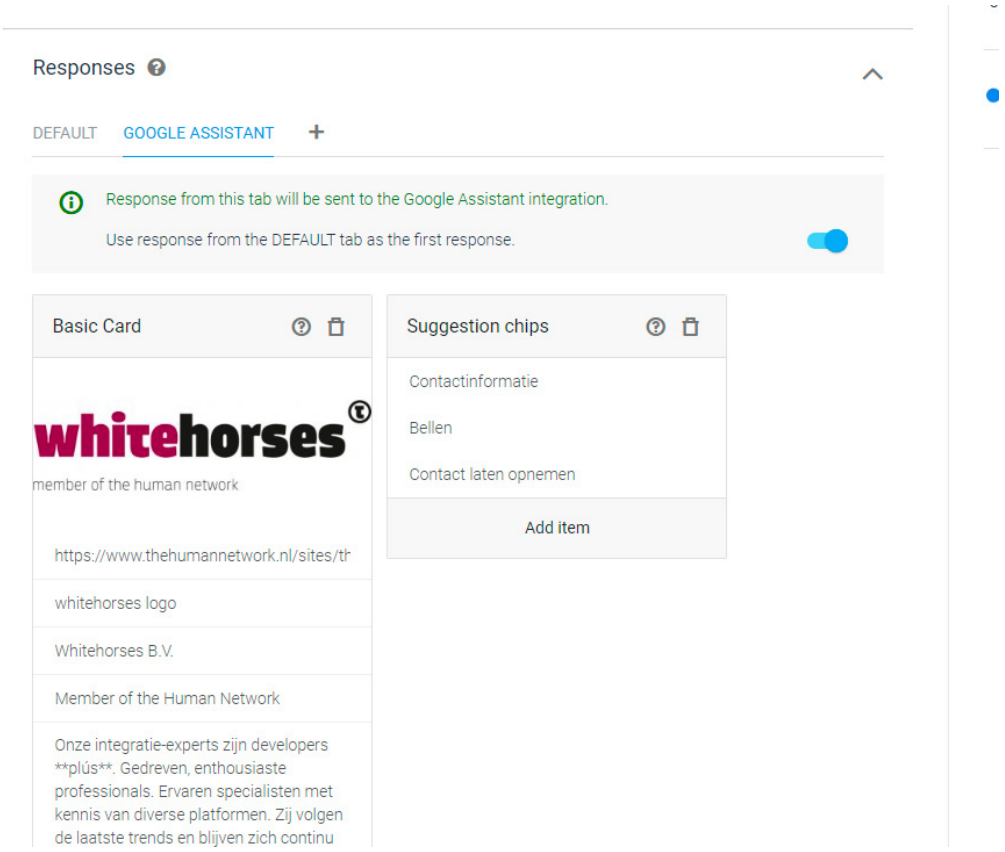
Figuur 3 Voorbeeld teksten

De Assistant moet kunnen bepalen wanneer de gebruiker de Intent uit. Hiervoor kan je de Assistant trainen. Trainen doe je door voorbeeldteksten op te geven die de gebruiker mogelijk gaat zeggen. De Assistant is slim genoeg om uit deze trainingsteksten de strekking van de Intent te halen. De gebruiker hoeft dus niet letterlijk deze teksten te zeggen (of te typen). Daarnaast kan de Assistant uit deze trainingsteksten vaak al bepalen welke Entities achterhaald moet worden. (Dit kan je overigens ook handmatig opgeven.)



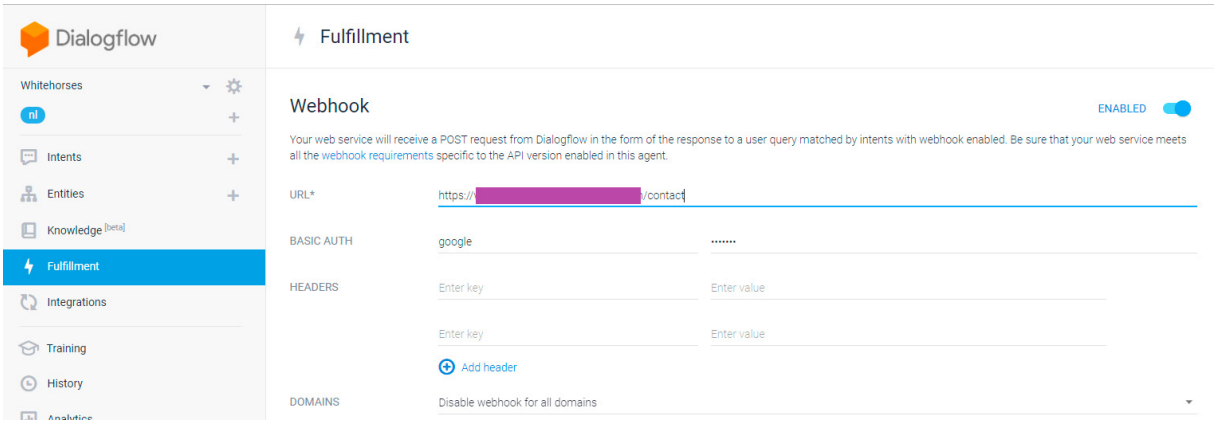
Figuur 4 Entities

In de Intent worden dus a.d.v. de trainingsteksten de Entities bepaald. Google heeft een hele serie vooraf gedefinieerde Entities die het kan bepalen. Zodra een Entity verplicht is gesteld, zal de Assistant deze altijd proberen te achterhalen, en zal doorvragen wanneer deze data niet in de initiële tekst is genoemd. Niet verplichte (optionele) Entities kunnen zijn bepaald door de input van de gebruiker, maar dit hoeft dus niet. Alleen als deze al in de initiële gebruikersinput zat, zal deze worden gematched en opgeslagen als Entity. Je kan er dus niet van uit gaan dat deze data aanwezig is als je de Entity gaat verwerken.
Zodra de Assistant heeft bepaald welke Intent de gebruiker heeft, en de verplichte Entities heeft verkregen, gaat de Assistant over op het antwoorden naar de gebruiker. Een (eenvoudig) “hardcoded” antwoord kan direct in de Intent worden geconfigureerd.



Figuur 5 Hardcoded response in de "Informatie" Intent

Wanneer er daadwerkelijk wat gebeuren moet met de verkregen Entities, of enige andere vorm van processing moet plaatsvinden zal deze verwerking extern plaats moeten vinden. Hiervoor kan de Fulfillment worden geconfigureerd, en een webhook call naar een backend plaatsvinden.



Figuur 6 Webhook call naar een externe API

De webhook call is een REST POST call, volgens het Webhook formaat. De inbound message van de Webhook bevat eigenlijk alle informatie over het Intent en de gebruiker. Zo krijg je binnen welke Intent het betreft, de Entities in de vorm van parameters, Contexten die waren gezet, maar ook wat de mogelijkheden zijn die de gebruiker tot zijn beschikking heeft. Dit laatste omvat bijvoorbeeld of de gebruiker een scherm tot zijn beschikking heeft, of alleen audio, de input heeft gegeven via spraak of keyboard en welke client is gebruikt (Google Assistant, Facebook Messenger e.d.).
Een snippet uit een request ziet er dan als volgt uit:

```
{
  “responseId”: “25af908d-7a28-44bc-af34-6680241a2afd”,
  “queryResult”: {
    “queryText”: “Neem contact op via mijn nummer 0612345678 en email
example@example.org”,
    “parameters”: {
      “email”: “example@example.org”,
      “telefoonnr”: “0612345678”
    }, ----etc
```

In het bovenstaande voorbeeld is duidelijk te zien dat de twee Entities gevuld zijn, en tevens is de tekst die de gebruiker heeft geuit tegen de Assistant ook meegegeven. Het antwoord dat de backend terug kan geven aan de Assistant geeft veel mogelijkheden. Aan de hand van de mogelijkheden die de gebruiker heeft, kan er voor gekozen worden het antwoord in een bepaalde grafische weergave weer te geven. Zo kan er bijvoorbeeld een “Basic card” worden geretourneerd die bestaat uit een afbeelding met begeleidende tekst. Het antwoord kan ook de Assistant sturen: je kan opgeven dat de gebruiker om bevestiging moet worden gevraagd, of dat bepaalde contexten moeten worden gezet bijvoorbeeld.

In dit specifieke geval geeft de backend alleen de ingevoerde Entities terug aan de gebruiker als een simpel antwoord, en zal verder op de achtergrond de contactgegevens doorsturen naar de daarvoor bestemde mailbox bij Whitehorses.

Het JSON-antwoord ziet er als volgt uit:

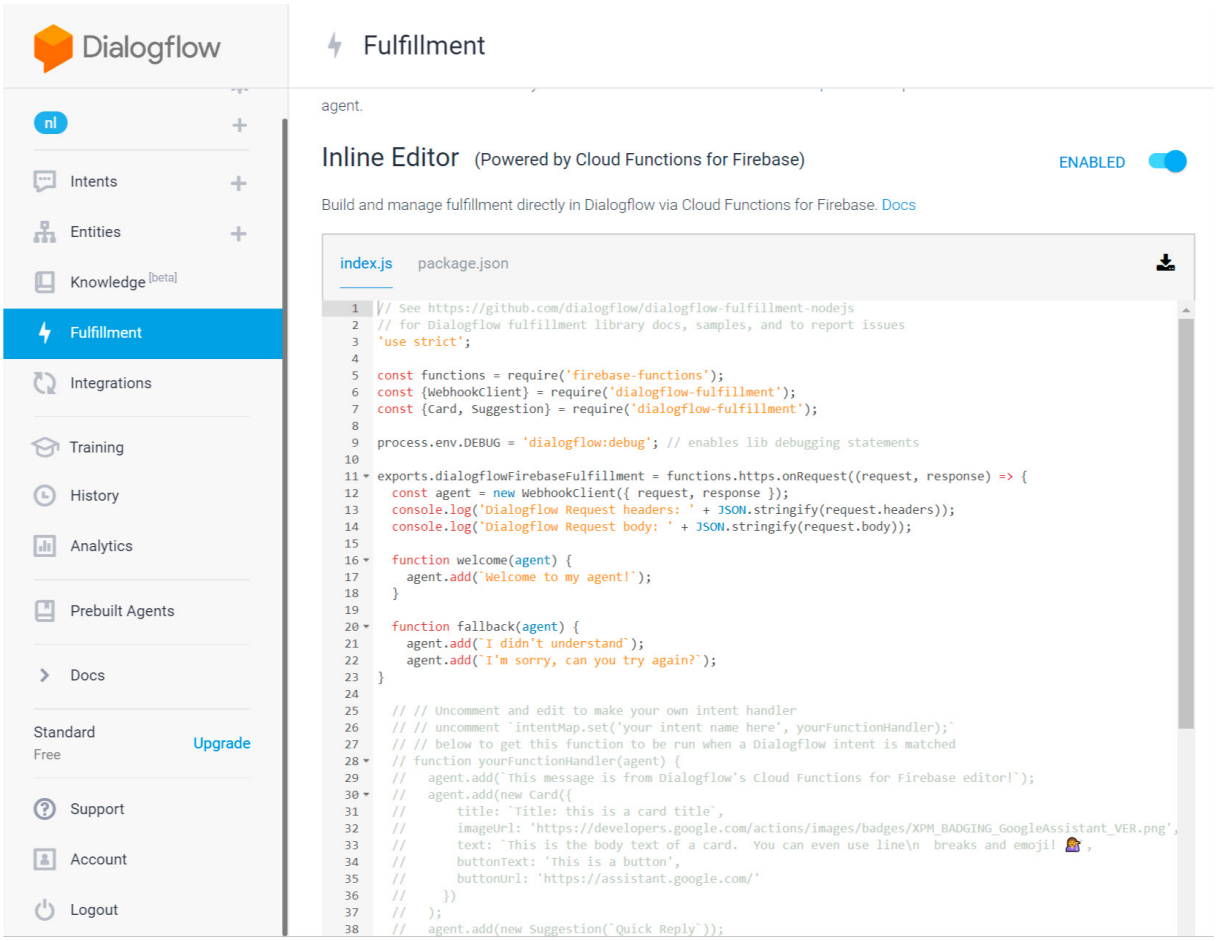
```
{
  “payload”: {
    “google”: {
      “expectUserResponse”: true,
      “richResponse”: {
        “items”: [
          {
            “simpleResponse”: {
              “textToSpeech”: “Dank voor
uw interesse in Whitehorses, we mailen u
op example@example.org, of bellen terug op
0612345678.”
            }
          }
        ]
      }
    }
  }
}
```

De backend is in dit geval een eenvoudige Spring Boot applicatie.

```
@RestController
public class ContactOpnemenController {

    @RequestMapping(method = RequestMethod.POST, value = “/contact”)
    public Response contact(@RequestBody String inboundJSON) {

        String antwoord = “”;

        try {
            JsonNode json = new ObjectMapper().readTree(inboundJSON);

            String email = json.findPath(“queryResult”).
findPath(“parameters”).get(“email”).asText();
```

```
            String telefoon = json.findPath("queryResult").
findPath("parameters").get("telefoonnr").asText();
/* VERDERE PROCESSING */

            if (telefoon.isEmpty()) {
                antwoord = "Dank voor uw interesse in Whitehorses, we
mailen u op "+ email+".";
            } else {
                antwoord = "Dank voor uw interesse in Whitehorses, we
mailen u op "+ email+", of bellen terug op " + telefoon+".";
            }


        } catch (IOException e) {
            // TODO Auto-generated catch block
                        e.printStackTrace();
        }

        List itemList = new ArrayList();
        Item item = new Item(new SimpleResponse().
withTextToSpeech(antwoord));
        itemList.add(item);
// Jackson modelling for response JSON
        return new Response().withPayload(new Payload().
withGoogle(new Google().withRichResponse(new RichResponse(itemList)).
withExpectUserResponse(true)));

    }
```

Je kan overigens de Fulfillment direct in een inline NodeJS javascript laten verwerken. Google biedt een uitgebreide NodeJS library aan die je inline, of in een Firebase cloud oplossing van Google kan draaien.



Toekomst

Digitale assistenten zullen steeds op meer plekken ons leven in kruipen. Samsung praat al over het integreren van Bixby in hun koelkasten, Android TV's hebben de Google Assistant al, de Smart Hubs en smart speakers (Google Home, Sonos One e.v.a.) ontwikkelen zich sneller dan onkruid. Het wordt dus een serieuze interface die gebruikers verwachten te kunnen gebruiken, net zoals men nu verwacht alles te kunnen doen met een mobiele app of website. In dit Whitebook heb ik met een simpel voorbeeld laten zien dat het eenvoudig is een eigen (bestaande) API te koppelen met een spraak assistent, in dit geval de Google Assistant. De Assistant neemt alle taken op zich m.b.t. het helder krijgen van de Intent en het ontfutselen van de Entities. De Fulfillment en webhook call kan vervolgens alle processing van deze Intent in house laten plaatsvinden, met alle vrijheden die dit geeft.

Referenties

1. Calm Technology
   https://calmtech.com/

2. The Best Interface Is No Interface: The simple path to brilliant technology
   door Golden Krishna.
   ISBN-13: 978-0133890334

3. Figuur ©Google
   https://codelabs.developers.google.com