Interactief Discord Klankwereld

Project Makerskills 2C

Mick Broer & Jimmy van der Heijden

Voorwoord

In dit project zullen wij een poging doen tot het ontwerpen van een interactieve bot voor het instant messaging platform Discord. Voorheen was het plan om een bot te maken die in staat was om de stem van de gebruiker te kopiëren, maar na verdere tests zijn wij er achter gekomen dat dit toch niet haalbaar is voor ons (op dit moment).
Het programmeerwerk is gedaan in Python waarbij de bot wordt aangeroepen in Discord.
Je kan het eindresultaat hier bekijken: https://youtu.be/KSu6qIBJ8QQ

Discord functionaliteit

Hieronder zullen wij het creeëren en functioneren van de bot binnen Discord uitleggen.
Daar komt bij kijken:

- Registreren
- Deelname aan een server
- Verbinding maken met Discord

Wat je nodig zult hebben:

- Python
- Text editor
- Discord account

Discord package installeren

Voordat we beginnen moeten de Discord package voor python installeren met het command:

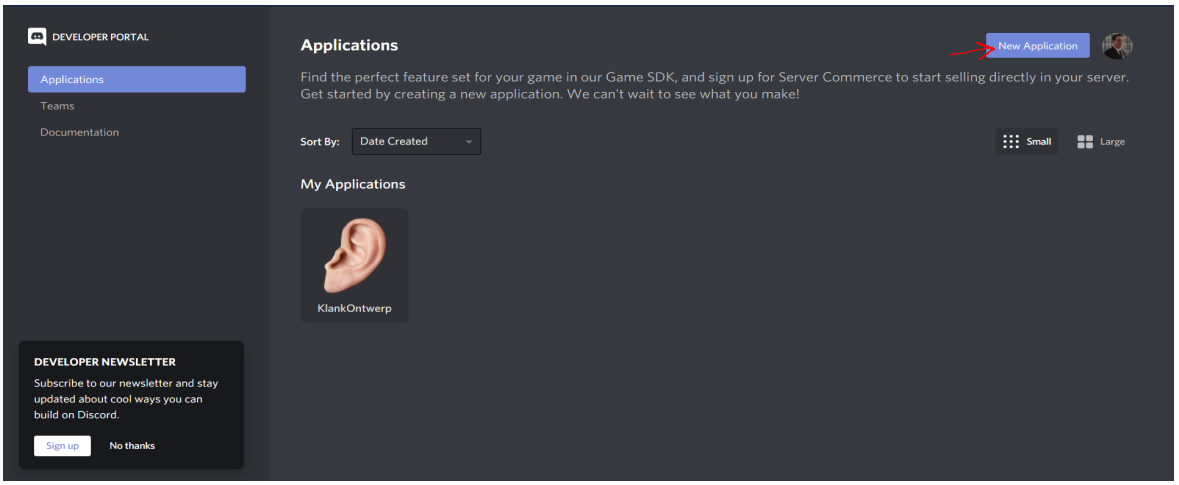
```
# Linux/macOS
python3 -m pip install -U discord.py

# Windows
py -3 -m pip install -U discord.py
```

Registreren

Om te beginnen moeten we naar het Discord's Developer Portal. Voor toegang heb je een al bestaand Discord account nodig of moet je er één aanmaken.
https://discord.com/developers/docs/intro

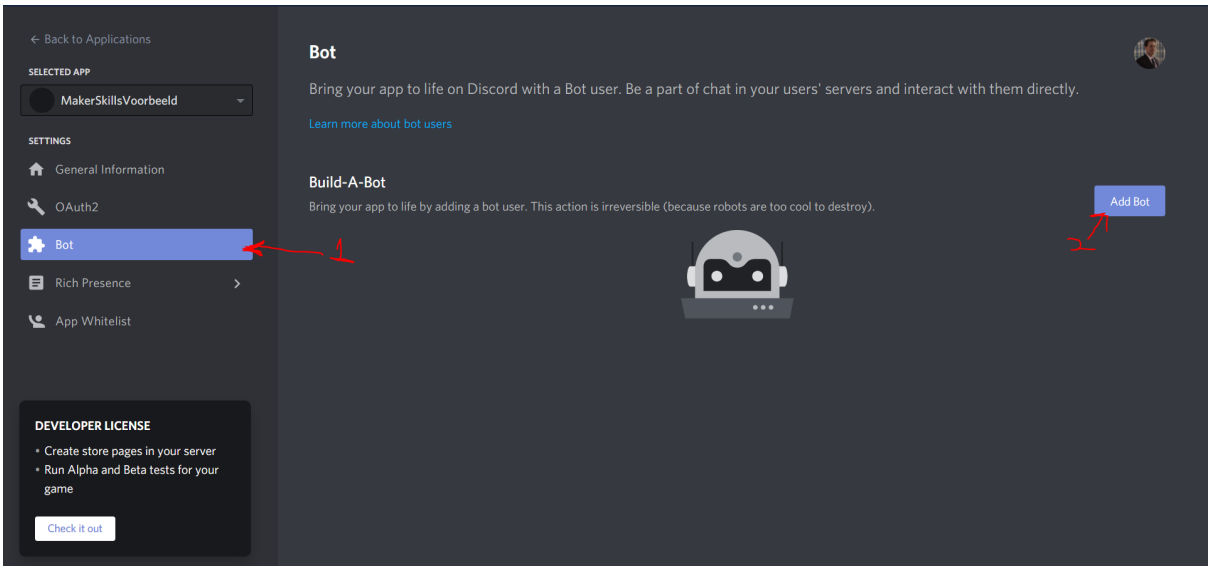
Ga naar de application tab en maak een nieuwe applicatie aan.



Voer een naam in en je zal gebracht worden naar de tab ‘General information’.

Top, we hebben nu een applicatie aangemaakt. Alleen wilt dat nog niet zeggen dat de bot ook al is aangemaakt.

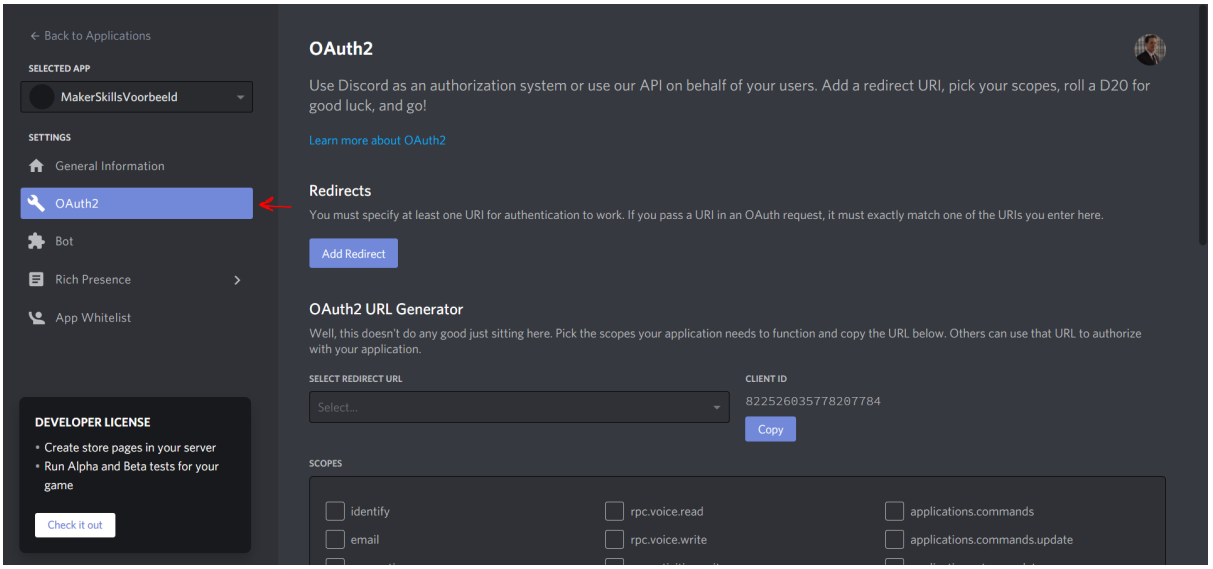
Ga na het tabje ‘Bot’ en klik op ‘Add Bot’.



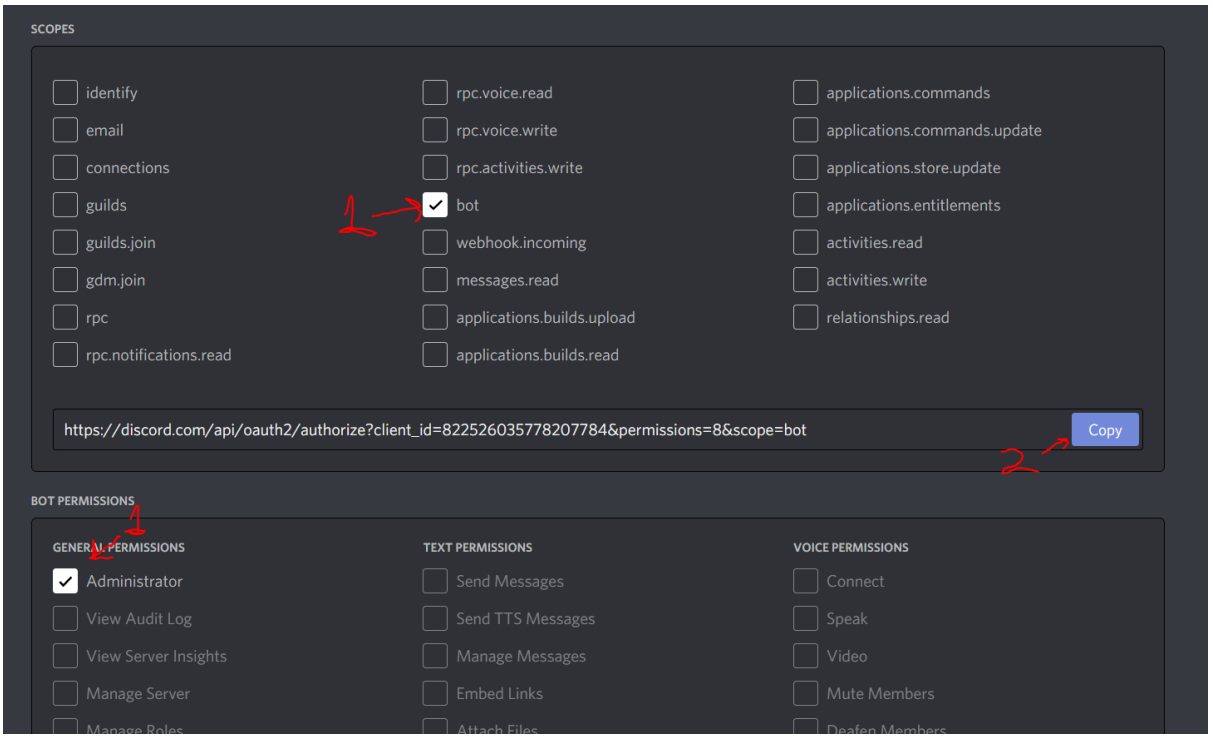
Zodra je hebt bevestigd dat je de bot aan je applicatie wilt toevoegen, zie je de nieuwe bot-gebruiker in het portaal.

De bot neemt vanzelf de naam aan die je aan de applicatie hebt gegeven. Dit kan je makkelijk veranderen door de username van de bot te veranderen.

Deelname aan server
Een bot kan geen uitnodigingen accepteren zoals een normale gebruiker dat wel kan. In plaats daarvan voeg je je bot toe met behulp van het OAuth2-protocol.



Ga naar de OAuth2 pagina. Op deze pagina zie je de OAuth2 URL generator.
Deze generator maakt een autorisatie-link aan voor het activeren en deactiveren van verschillende bot-functies.
In dit geval wil je de botgebruiker toegang verlenen tot Discord API's met behulp van de OAuth2-credentials. Om dit te doen scroll je naar beneden en selecteer je ‘bot’ onder de ‘SCOPES’ opties en ‘Administrator’ onder de ‘BOT PERMISSIONS’ opties.



Nu heeft Discord de autorisatie-URL van je app gegenereerd met de geselecteerde machtigingen.

Selecteer ‘Copy’ naast de URL die voor je is gegenereerd, plak deze in je browser en selecteer de server/guild waaraan je bot moet deelnemen.

Open de Discord applicatie en je zult zien dat de bot heeft deelgenomen aan je server.

Verbinding maken met Discord
Er zal in dit verslag niet te diep ingegaan worden op de programmeertaal Python zelf meer eerder op de geschreven functies: wat ze doen en hoe ze te gebruiken etc.

Nu dat de bot zich bevindt in de gewenste server moeten we de bot ‘online’ krijgen. Met discord.py doe je dit door een instantie van Client in een text-editor te maken:

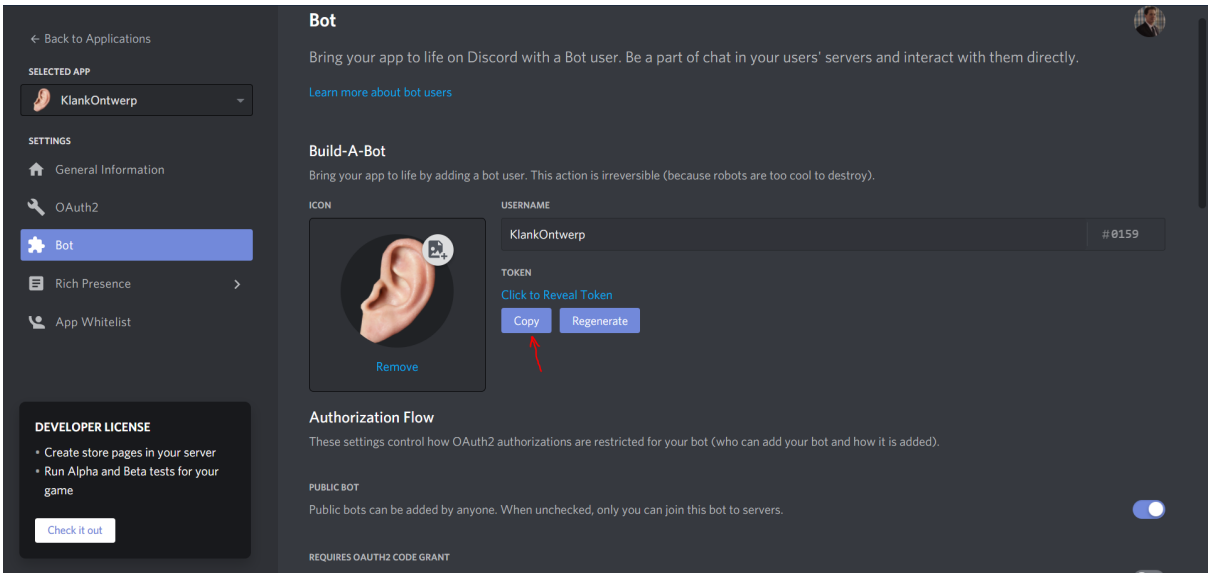
```
import discord
from discord.ext import commands

client = discord.Bot(command_prefix - ‘.’)

@client.event async def on_ready():
    print(‘Bot is ready.’')

client.run(TOKEN)
```

We hebben nog één ding nodig voordat we het bestand opslaan.
Je moet ‘TOKEN’ vervangen door jouw bot-token. Die kan je krijgen door terug te gaan naar de Bot-pagina op de Developer Portal en op Copy te klikken onder de TOKEN-sectie.
Houd je token privé. Wanneer anderen toegang tot de token krijgen zullen zij de mogelijkheid hebben om andere scripts te koppelen aan je bot. Mocht dit gebeuren kan je een nieuwe maken door op Regenerate te klikken. Dit zal de oude token ongeldig maken.



Nu heb je een Client aangemaakt en een on_ready () event geïmplementeerd. Dit event zal een melding in de console-log geven wanneer de Client een verbinding tot stand heeft gebracht met Discord en het klaar is met het voorbereiden van de gegevens die Discord heeft verzonden, zoals inlogstatus, gilde en kanaal data en meer.

Sla de het tekstbestand op als ‘bot.py’. Open het bestand met Python of een ander programma waarmee je Python taal kunt draaien.

Open Discord, je zult zien dat de bot nu online is!

Discord Bot interactie met externe programma's

Voor de interactie met andere programma's gebruiken we Open Sound Control. Om met OSC te werken in Python, moet je eerst via pip een library installeren:

```
$ pip install python-osc
```

Bovenaan in ons bot.py bestand importeren we deze library:

```
from pythonosc import udp_client
```

Vervolgens geven we deze client een aantal initialisatie waardes. Deze kunnen aangepast worden wanneer je bot.py aanroept in de terminal, door middel van '--port' of '--ip'.

```
parser = argparse.ArgumentParser()
parser.add_argument("--ip", default="127.0.0.1",
    help="The ip of the OSC server")
parser.add_argument("--port", type=int, default=5005,
    help="The port the OSC server is listening on")
args = parser.parse_args()
client2 = udp_client.SimpleUDPClient(args.ip, args.port)
```

Belangrijk is hier dat je de udp_client een andere naam geeft dan de Discord client. Wij hebben gekozen voor 'client2'.
Vervolgens moeten we een functie schrijven die commands uit Discord kan lezen en vervolgens een OSC message kan sturen naar de gewenste port:

```
@client.command()
async def rate(ctx, *, oscMessage):
            client2.send_message("/rate", oscMessage)
            await ctx.send(oscMessage)
```

Bij het bovenstaande voorbeeld zie je dat we eerst de Discord client laten weten dat we bezig zijn met een command. Vervolgens definiëren we de functie, we noemen hem in dit geval rate. We geven haar wat argumenten, en sturen het *'oscMessage'* argument direct via OSC door naar port 5005 en dan specifiek naar '/rate'.
'await' stuurt diezelfde *'oscMessage'* ook weer terug naar de gebruiker, zodat die enige feedback krijgt dat zijn opdracht aankomt. Vervolgens kun je de bot weer runnen:

```
python bot.py
```

Je kunt nu elk programma dat OSC kan ontvangen aansturen door middel van Discord commands. Wij hebben ervoor gekozen dit te doen in SuperCollider:

```
OSCdef(\rateOSC, {
       arg msg, time, addr, port;
       ~def.set(\rate, msg[1].asFloat);
       msg[1].postln;
}, '/rate', recvPort:5005);
```

Veel plezier met OSC en Discord ;)

Bronnen:
Lucas (2020.) *Python: Making a Discord Bot (Rewrite / v1.x)*
Geraadpleegd van:
https://youtube.com/playlist?list=PLW3GfRiBCHOhfVoiDZpSz8SM_HybXRPzZ

Alex Ronquillo *How to Make a Discord Bot in Python*
Geraadpleegd:
https://realpython.com/how-to-make-a-discord-bot-python/#creating-a-discord-account

Eli Fieldsteel *Week 10: OSC - MUS 499C Fall 2019 - Audio Coding with SuperCollider*
https://youtu.be/R0ulauoGCvI

python-osc 1.7.4
https://pypi.org/project/python-osc/