

# Implementazione di un ChatBot per l'interazione con un NPC su Decentraland

## Installazione SpaCy RASA

Come modello di RASA ho deciso di utilizzare un modello pre-addestrato di RASA chiamato SpaCy.

Nel terminale digitare i seguenti comandi per l'installazione di SpaCy:

- `python3 -m pip install "rasa[spacy]"`
- `python3 -m spacy download it_core_news_md`

Creazione della cartella per il Progetto del Chatbot

- `mkdir mio_chatbot`
- `cd mio_chatbot`
- `rasa init --no-prompt` //crea il progetto senza chiedere nulla all'utente

Modifica del file config.yml del progetto RASA

- Language: IT
- Name: SpacyNLP
- Modello: it\_core\_news\_md

Modifica del file domain.yml del progetto RASA

- Responses: Sono state tradotte in italiano tutte le risposte

Modifica nlu.yml

- Modifica di tutte le intenzioni

Train del modello(dopo tutte le modifiche del progetto):

- `Rasa train`
- `Rasa train --debug` // per vedere eventuali warning o problemi

Interazione con il ChatBot

- `Rasa shall` //Per parlare con il chatbot

## Creazione scena di Decentraland

- `Mkdir NPC` // Creazione della cartella NPC
- `CD NPC` // Per entrare dentro la cartella
- Create new scene (estensione SDK7) // Su VSCode (estensione SDK7)
- Creazione del codice per l'NPC nel file index.ts

```
// We define the empty imports so the auto-complete feature works as expected.  
import { Vector3 } from '@dcl/sdk/math'  
import { Transform, engine } from '@dcl/sdk/ecs'  
import { AvatarShape } from "@dcl/sdk/ecs"  
  
import { changeColorSystem, circularSystem } from './systems'
```

```

import { setupUi } from './ui'

export function main() {
  // Defining behavior. See `src/systems.ts` file.
  engine.addSystem(circularSystem)
  engine.addSystem(changeColorSystem)

  //Add an entity
  const myAvatar = engine.addEntity()

  //Create an Avatar into entity
  AvatarShape.create(myAvatar)

  //Create a trasformation for the avatat position into the scene
  Transform.create(myAvatar, {
    position: Vector3.create(4, 0.25, 5) //Avatar position
  })

  // draw UI. Here is the logic to spawn cubes.
  setupUi()
}

```

- Documentazione presa da [decentraland](https://decentraland.org/docs/creating-an-npc)

## Comunicazione con NPC:

Dopo aver creato l’NPC nella scena di Decentraland, ho creato l’interazione tra l’NPC e il chatbot grazie al server di RASA:

- Avvio del server RASA:
  - `rasa run --enable-api --cors "*" --debug` //Avvio del server con debug
  - `rasa run --enable-api` // Avvio del server senza debug
- Interazione dell’NPC con il Player(in questo caso quando il player preme “E”, in automatico si invia al server un messaggio con scritto “Ciao”.

```

//Create an Avatar into entity
AvatarShape.create(myAvatar)

//Create a trasformation for the avatat position into the scene
Transform.create(myAvatar, {
  position: Vector3.create(4, 0, 5) //Avatar position
})

MeshCollider.setBox(myAvatar)

```

```

MeshRenderer.setBox(myAvatar)

// Interaction with Player
pointerEventsSystem.onPointerDown(
  {entity: myAvatar, opts: {button: InputAction.IA_PRIMARY, hoverText: "Parla con me"}},
  function(){
    executeTask(async()=>{
      const response = await askRasa("Ciao!") // Send a message to Rasa
      console.log("NPC Risponde:", response) // Show the responde of RasaBot
    })
  }
)

```

- Interazione con il server RASA  
Per far parlare l'NPC sfrutto il server rasa sulla porta localhost:5005

```

// Function for call the RASA API
async function askRasa(question: string): Promise<string>{
  try {
    const response = await fetch('http://localhost:5005/webhooks/rest/webhook', {
      method: 'POST',
      headers: { 'Content-Type': 'application/json' },
      body: JSON.stringify({ sender: "user", message: question })
    })

    const data = await response.json()
    if (data.length > 0) {
      return data[0].text // Ritorna la risposta di Rasa
    } else {
      return "Non ho capito, puoi ripetere?"
    }
  } catch (error) {
    console.error("Errore chiamando Rasa:", error)
    return "Errore nel contattare il server."
  }
}

```