PROYECTO

Curso de Inteligencia Artificial: ChatGPT, DALL-E y Hugging Face

Por: César Vega L

Creación de Imágenes utilizando DALL-E

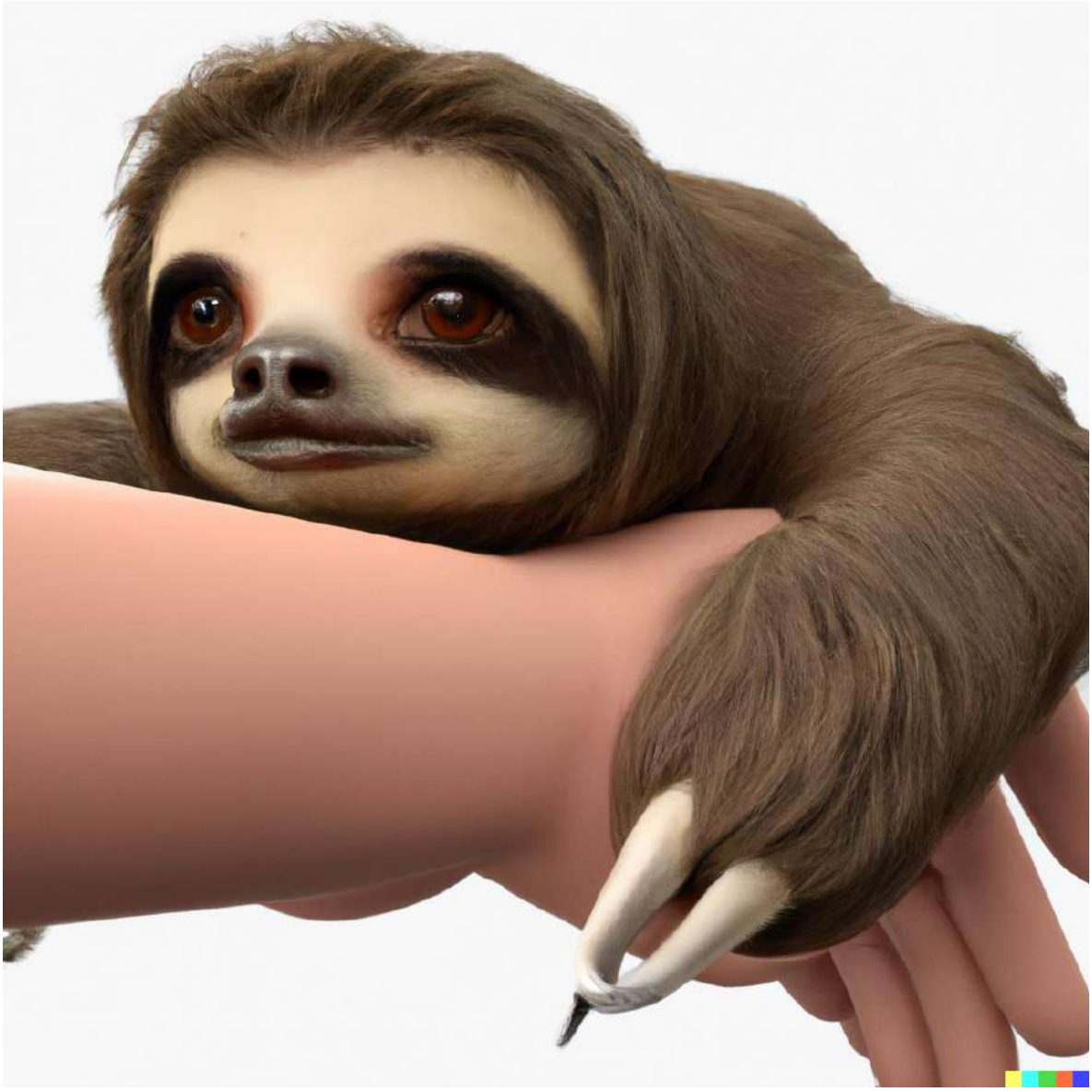
PROMPT: Young Korean couple, walking on the sidewalk together, wearing hanbok, sunny day, Expressionism style by Xooang Choi

PROMPT - Korean young couple, walking on a sidewalk together, wearing hanbok, sunny day, impressionist style

PROMPT - 3D render, Cutest Baby Hamster, runnig a horse, wearing an soldier uniform like second world war, ligth sword

PROMPT - 3D render, Cutest Baby Seal, playing pc games, wearing a cape like hacker

PROMPT - 3D render, Cute Baby Sloth climbing up a man's hand



PROMPT - 3D render, k9, Police dog, logo, multicam, anime character with magic, superpowers, wearing flight goggles and many details

Usando la API de DALL-E con python en Colab

Dalle_API (Proyecto Cesar Vega).pdf

Usando Leonardo.AI

create an avatar from me photo, 4K, portrait of a beautiful, masterpiece, realistic, anime, cloud, sunlight, cinematic light, hypernet:dalcefo_nocopy

ugly, duplicate, morbid, mutilated, out of frame, extra fingers, mustache, mutated hands, poorly drawn hands, poorly drawn face, mutation, deformed, ugly, blurry, bad anatomy, bad proportions, extra limbs, cloned face, disfigured, out of frame, ugly, extra limbs, bad anatomy, gross proportions, malformed limbs,

Usando HuggingFace

Usando la API de DALL-E con Python

Instalación de librería de OpenAI

```
!pip install openai
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting openai
  Downloading openai-0.26.5.tar.gz (55 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 55.5/55.5 KB 1.7 MB/s eta 0:00:00
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Installing backend dependencies ... done
  Preparing metadata (pyproject.toml) ... done
Requirement already satisfied: aiohttp in /usr/local/lib/python3.8/dist-packages (from openai) (3.8.3)
Requirement already satisfied: requests>=2.20 in /usr/local/lib/python3.8/dist-packages (from openai) (2.25.1)
Requirement already satisfied: tqdm in /usr/local/lib/python3.8/dist-packages (from openai) (4.64.1)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.8/dist-packages (from requests>=2.20->openai) (2022.12.7)
Requirement already satisfied: chardet<5,>=3.0.2 in /usr/local/lib/python3.8/dist-packages (from requests>=2.20->openai) (4.0.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.8/dist-packages (from requests>=2.20->openai) (1.24.3)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.8/dist-packages (from requests>=2.20->openai) (2.10)
Requirement already satisfied: async-timeout<5.0,>=4.0.0a3 in /usr/local/lib/python3.8/dist-packages (from aiohttp->openai) (4.0.2)
Requirement already satisfied: aiosignal>=1.1.2 in /usr/local/lib/python3.8/dist-packages (from aiohttp->openai) (1.3.1)
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.8/dist-packages (from aiohttp->openai) (1.3.3)
Requirement already satisfied: charset-normalizer<3.0,>=2.0 in /usr/local/lib/python3.8/dist-packages (from aiohttp->openai) (2.1.1)
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.8/dist-packages (from aiohttp->openai) (6.0.4)
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.8/dist-packages (from aiohttp->openai) (22.2.0)
Requirement already satisfied: yarl<2.0,>=1.0 in /usr/local/lib/python3.8/dist-packages (from aiohttp->openai) (1.8.2)
Building wheels for collected packages: openai
  Building wheel for openai (pyproject.toml) ... done
  Created wheel for openai: filename=openai-0.26.5-py3-none-any.whl size=67620 sha256=3c6599c65830c3acb24f9df85d82ad55aa5ce7b1717bde2f2b
  Stored in directory: /root/.cache/pip/wheels/a7/47/99/8273a59fbd59c303e8ff175416d5c1c9c03a2e83ebf7525a99
Successfully built openai
Installing collected packages: openai
Successfully installed openai-0.26.5
```

```
import openai

openai.api_key = "sk-vXeuGhX3VCov[redacted]"
```

Consumir API

```
response = openai.Image.create(
  prompt="3D render, A winged blue and purple anthropomorfic kitty wearing flight goggles, many details, nice relaxing feeling",
  n=1,
  size="1024x1024"
)
image_url = response['data'][0]['url']
```

```
response['data']
```

```
[<OpenAIObject at 0x7f8d9bb706d0> JSON: {
   "url": "https://oaidalleapiprodscus.blob.core.windows.net/private/org-mYgwSOueuOnRrpZguTlABckO/user-zeZuCFt5TDC1JvRxIfnA9nP5/img-lSqddcNpwDydwPhpjeP5Gbqs.png?st=2023-02-13T16%3A20%3A43Z&se=2023-02-13T18%3A20%3A43Z&sp=r&sv=2021-08-06&sr=b&rscd=inline&rsct=image/png&skoid=6aaadede-4fb3-4698-a8f6-684d7786b067&sktid=a48cca56-e6da-484e-a814-9c849652bcb3&skt=2023-02-13T15%3A20%3A32Z&ske=2023-02-14T15%3A20%3A32Z&sks=b&skv=2021-08-06&sig=YnGz3vTyIZVJ4LspcaxwwpeP/ICoeqU0EYBIlpvO9yk%3D"
 }]
```

```
from PIL import Image
import requests
from io import BytesIO
```

```
response_url = requests.get(response['data'][0]['url'])
```

```
response_url
```

```
<Response [200]>
```

```
1 img = Image.open(BytesIO(response_url.content))
```

```
1 img
```

Uso del API para variations

```
1 img.save('H:\Cursos\2023\IA\img_cat.png')
```

```
1 response = openai.Image.create_variation(
2   image=open("H:\Cursos\2023\IA\img_cat.png", "rb"),
3   n=1,
4   size="1024x1024"
5 )
6 image_url = response['data'][0]['url']
```

```
1
```

Haz doble clic (o pulsa Intro) para editar

```
1 response['data']
```

```
[<OpenAIObject at 0x7f8d9ab10a90> JSON: {
   "url": "https://oaidalleapiprodscus.blob.core.windows.net/private/org-mYgwSOueuOnRrpZguTlABckO/user-zeZuCFt5TDC1JvRxIfnA9nP5/img-skEbpmj0xhHsyUjEg1546lRA.png?st=2023-02-13T16%3A32%3A49Z&se=2023-02-13T18%3A32%3A49Z&sp=r&sv=2021-08-06&sr=b&rscd=inline&rsct=image/png&skoid=6aaadede-4fb3-4698-a8f6-684d7786b067&sktid=a48cca56-e6da-484e-a814-9c849652bcb3&skt=2023-02-13T06%3A47%3A16Z&ske=2023-02-14T06%3A47%3A16Z&sks=b&skv=2021-08-06&sig=DwQ%2BatM27ozJAKY%2BwQ8hV5/5cjk23nB%2ByYybOK%2BnLbw%3D"
 }]
```

```
1 response_url = requests.get(response['data'][0]['url'])
```

```
1 imgv = Image.open(BytesIO(response_url.content))
```

```
1 imgv
```

Imágenes en memoria

```
img_mem = BytesIO(response_url.content)
```

```
byte_stream: BytesIO = img_mem
byte_Array = byte_stream.getvalue()
response = openai.Image.create_variation(
  image=byte_Array,
  n=1,
  size="1024x1024"
)

```

```
response_url = requests.get(response['data'][0]['url'])
imgvar = Image.open(BytesIO(response_url.content))
imgvar
```