

WildBytes – puntata 1. – Python

Ciao Youtubers e bentrovati su WildBytes! Oggi vedremo come accedere ai servizi di OpenAI, installare Python e tutto ciò che ci serve per iniziare.

Innanzitutto, dobbiamo configurare alcune cose. Se non avete ancora **Python** sul vostro computer, dovete scaricarlo. Andate sul sito ufficiale [Python.org](https://python.org) e scaricate l'ultima versione disponibile, che al momento è la 3.11.

Un piccolo avvertimento per chi usa Windows: durante l'installazione, assicuratevi di selezionare l'opzione per aggiungere Python alle variabili d'ambiente. Questo vi permetterà di eseguire Python direttamente dal terminale.

Una volta installato Python, potete aprire il terminale, digitare "Python" e premere invio. Vedrete apparire quello che viene chiamato REPL, un ambiente che vi permette di eseguire il codice Python riga per riga, proprio come si faceva con i computer negli anni '80.

Proviamo con un esempio:

```
a = 10
b = 1
print(a+b)
```

In Python ci sono vari modi per formattare il testo e uno dei modi che trovo più intuitivo è il metodo dell'“f-string” che consiste nel mettere una f seguita da due apici includendo il contenuto di una variabile fra parentesi graffe, ad esempio:

```
print(f"il contenuto di a è: {a}")
```

Per fare la parentesi graffa potete premere shift + alt grid + “[“ per l'apertura della parentesi graffa, e shift+alt grid + “]” per chiuderla.

Adesso possiamo importare le librerie di OpenAI. Prima però, è fondamentale discutere i costi associati all'utilizzo di queste API, per garantirvi una piena consapevolezza delle possibili spese che potreste affrontare.

Le API sono tecnicamente delle librerie che ci permettono di usare un determinato programma come ad esempio chatGpt, nel nostro programma. OpenAI ha fatto un gran lavoro per renderle semplici e comprensibili.

È cruciale comprendere i costi associati all'uso dell'API di OpenAI. Sebbene io abbia strutturato questo tutorial per farvi spendere meno di 5 centesimi a puntata, ma potete anche semplicemente osservare e prendere appunti per imparare mano a mano che procediamo, senza interagire direttamente con le API.

I costi variano a seconda del modello che si decide di utilizzare. Ad esempio, per il modello più avanzato, il prezzo è fissato a 0,06 dollari ogni 1000 tokens, che corrisponde approssimativamente a un testo di 450 parole. Questo significa che, con questo modello, potrete interagire con il bot fornendo un input lungo quanto 8 pagine di Word. E sottolineo, questo è uno dei modelli di punta nel campo dei chatbot.

Pensiamo per esempio a cosa potrebbe fare per noi chatGpt mentre siamo al volante, le condizioni climatiche possono cambiare rapidamente, soprattutto in un'epoca di cambiamenti climatici in cui siamo testimoni di fenomeni meteorologici sempre più estremi.

Prima di partire per un lungo viaggio o mentre ci stiamo dirigendo da qualche parte potrebbe servirci un qualcosa che analizzando i dati forniti dal meteo ci avvisi se nella direzione in cui stiamo andando, ci avverta in anticipo che potremmo trovare: grandine, oppure neve o nebbia o un downBurst che potrebbe farci avere un incidente.

- *"Attenzione, rilevata probabilità di grandine tra 20 km. Consiglio di cercare un rifugio per il tuo veicolo."*
- *"Nebbia fitta rilevata 10 km avanti. Riduci la velocità e accendi i fari antinebbia."*
- *"Downburst rilevato nella tua direzione. Per favore, fermati in un luogo sicuro."*

Tuttavia, se decidiamo di utilizzare il modello Chat Gpt-3.5 Turbo, i costi si riducono a 0.0015 dollari ogni 1000 tokens. Questo equivale a circa 0.00002 dollari per token.

Una risposta tipica di chatGpt potrebbe contenere circa 100 tokens, quindi 10 domande ogni 0.02 dollari sono 100 domande con 20 centesimi.

Durante questa puntata, vi mostrerò come gestire e limitare i costi per ottimizzare l'uso delle API. Se siete nuovi a tutto ciò, potreste iniziare con un budget di 5 dollari. Questo vi darà l'opportunità di familiarizzare con il sistema e comprendere meglio i costi effettivi.

Andiamo sul sito di OpenAI. Se non avete ancora un account, dovete registrarvi. La procedura è semplice e potete farlo utilizzando le vostre credenziali Google, Microsoft o Apple.

Dopo aver effettuato l'accesso, navigate alla sezione delle chiavi API di OpenAI per ottenere le credenziali necessarie all'utilizzo dei servizi.

A questo punto, possiamo iniziare a utilizzare Python. Ci sono diverse opzioni: possiamo scrivere il codice direttamente nel terminale, comando per comando; possiamo utilizzare l'IDE integrato di Python; oppure, per chi preferisce, è possibile scrivere il codice in Notepad, salvarlo con estensione ".py" e poi eseguirlo tramite terminale con il comando `Python nome_del_file.py`.

Installiamo la libreria di OpenAI digitando sul terminale "pip install openai".

Creiamo una cartella, ci servirà per tenere i file di wildBytes, io la creo sul desktop, vuoi potete crearla dove preferite e la chiamo "wildBytes", (la scrivo senza spazi perchè a volte gli spazi creano problemi) e dentro creiamo un nuovo file di testo e lo rinominiamo secretKeys.py con la "s" minuscola e la K maiuscola e dentro ci scriveremo qualcosa del tipo:

```
OpenAI = ""
```

Tra le virgolette, incolliamo la chiave API che abbiamo ottenuto dal nostro account OpenAI.

Windows di default nasconde le estensioni per i file conosciuti. Per visualizzare l'estensione del file e rinominarlo .py dobbiamo modificare le impostazioni nella finestra. Facendo doppio click sul file, ovviamente windows non saprà più come aprirlo, possiamo dirgli quindi di usare notepad come applicazione principale e in questo modo per modificarlo ci basterà fare doppio click.

Una volta incollata la chiave, salviamo e chiudiamo in file. Ne dobbiamo quindi creare un altro che conterrà il nostro programma e che chiameremo main.py.

Le chiavi per le API dovrebbero essere tenute private e non condivise o caricate pubblicamente (ad esempio su GitHub o you Tube). Questo per evitare utilizzi non autorizzati o costi inaspettati.

```
import OpenAI  
import secretKeys
```

```
OpenAI.api_key = secretKeys.OpenAI
```

```
print("hello world")
```

per vedere se funziona, possiamo salvarlo, aprire il terminale e digitare: Python main.py

e se tutto funziona come dovrebbe ci stampa la scritta hello world!

Ora che abbiamo impostato il tutto, possiamo davvero iniziare ad utilizzare ChatGPT. La prima cosa da fare è decidere quale modello di AI intendiamo utilizzare. OpenAI offre diversi modelli, alcuni più economici, altri più avanzati e costosi.

```
models = {"gpt-4", "gpt-3.5-turbo", "babbage-002", "davinci-002",  
          "text-davinci-003", "text-davinci-002", "davinci", "curie",  
          "babbage", "ada"}
```

Usiamo chat gpt-3.5-turbo e creiamo una variabile che conterrà appunto il modello che vogliamo usare e che chiameremo model, poi creiamo un prompt che conterrà la nostra domanda, che può essere al momento un semplice: "Ciao Chat! Questo è un test".

Quando interagiamo con ChatGPT attraverso l'API, dobbiamo strutturare i nostri messaggi in un formato specifico. Questo formato prevede una lista di messaggi, dove ogni messaggio ha un "role" (il ruolo, che può essere "user" o "assistant") e un "content" (il contenuto del messaggio).

Per avere una risposta, in una situazione tipo chat, possiamo creare una variabile "**response**" che sarà uguale ad **OpenAI**, la classe delle **API**, **ChatCompletion**, che è la classe che vogliamo usare, e della classe **chatCompletion** useremo **create**, la funzione che vogliamo utilizzare. Dentro le parentesi dobbiamo specificare almeno due parametri che saranno model = model e messages = messages. Quindi possiamo stampare la risposta.

```
import OpenAI  
import secretKeys  
  
OpenAI.api_key = secretKeys.OpenAI  
model = "gpt-3.5-turbo"  
prompt = "Ciao Chat! Questo è un test per vedere se le API funzionano  
correttamente. Come va?"
```

```

messages = [
    {"role": "user", "content": f"{prompt}"},
]
response = OpenAI.ChatCompletion.create(model=model, messages=messages)
print(response)

```

lanciando il programma e se non abbiamo commesso errori, ci dovrebbe dare come risposta un qualcosa tipo questa:

```

{
  "id": "chatcmpl-7tyjNmgtIPKLbN2JnXbqAFj7inWYp",
  "object": "chat.completion",
  "created": 1693575845,
  "model": "gpt-3.5-turbo-0613",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "\"Ciao! Sono felice di sentire che stai testando le API. Sono un'intelligenza artificiale e sono qui per aiutarti. Come posso assisterti oggi?\""
      },
      "finish_reason": "stop"
    }
  ],
  "usage": {
    "prompt_tokens": 31,
    "completion_tokens": 43,
    "total_tokens": 74
  }
}

```

otteniamo una risposta in formato “**jSon**”, un formato particolare ci permette di avere una “**struttura dati**”, che possiamo usare per estrarre le informazioni che ci interessano. Oltre alla risposta, abbiamo il numero totale di tokens, e vediamo che ne abbiamo usati 31 nel nostro prompt e impareremo a fare meglio e ne abbiamo usati 43 per la risposta e anche qui impareremo decisamente a fare meglio.

Non vi preoccupate se ci sono dei termini che non conoscete, ricordate sempre che potete andare nella pagina ufficiale di chatGpt e chiedere: “Ciao Chat, che cos’è jSon”?

Per estrarre solo il contenuto del messaggio dobbiamo scrivere:

```
print(answer['choices'][0]['message']['content'])
```

Con questo metodo, possiamo anche estrarre informazioni dettagliate sui token utilizzati:

```
response = OpenAI.ChatCompletion.create(model=model, messages=messages)
answer = response['choices'][0]['message']['content']
promptToken = response['choices'][0]['prompt_tokens']
completionToken = response['choices'][0]['completion_tokens']
totalTokens = response['choices'][0]['total_tokens']
```

Questi dati sono essenziali per monitorare e gestire i costi associati all'utilizzo dell'API. Ad esempio, con queste informazioni, possiamo calcolare quanti token abbiamo a disposizione prima di raggiungere una spesa di 5 centesimi.

Quindi, considerando che il costo è di 0.0015 dollari ogni 1000 tokens posso creare un funzione che mi restituisce quanti altri tokens posso usare.

```
import OpenAI
import secretKeys

_promptTokens = 0 # tokens utilizzati per il prompt 0.0015 dollari
_completionTokens = 0 # tokens utilizzati per la completion 0.002 dollari
_totalTokens = 0
_budget = 0.05 # 5 centesimi di dollaro

def remainingTokens(balance, totalTokens):
    # Costo per 1000 tokens
    cost_per_1000_tokens = 0.002
    cost_per_token = cost_per_1000_tokens / 1000

    balance = balance - (cost_per_token * totalTokens)
    remaining_tokens = float(balance / cost_per_token)

    return balance, f"Ti rimangono ${balance:.4f}. Puoi ancora utilizzare {int(remaining_tokens)} tokens."

OpenAI.api_key = secretKeys.OpenAI
model = "gpt-3.5-turbo"
# Cambia questa variabile per fare una domanda diversa
```

```

prompt = "Ciao Chat! Questo è un test per vedere se le API funzionano
correttamente. Come va?"
messages = [
    {"role": "user", "content": f"{prompt}"},
]

response = OpenAI.ChatCompletion.create(model=model, messages=messages)
print(response)
answer = response['choices'][0]['message']['content']
_promptTokens += int(response['usage']['prompt_tokens'])
_completionTokens += int(response['usage']['completion_tokens'])
_totalTokens += int(response['usage']['total_tokens'])

print(answer)
budget, remaining = remainingTokens(_budget, _totalTokens)
print(remaining)

```

per cambiare domanda ci basterà cambiare il contenuto della nostra variabile prompt e ad esempio possiamo chiedergli qual'è la capitale della Francia? e lui ci risponderà che la capitale della Francia è Parigi.

Ovviamente dobbiamo aggiornare le variabili in modo che mi dica quanti centesimi mi sono rimasti. Possiamo mettere il loop il programma?

Sì, ma siamo giunti al termine di questa puntata, e oltre a ricordarvi è completamente gratuito, o come si dice, Open Source con licenza MIT vi ricordo che nella descrizione di ogni puntata, potete trovare il link alla mia pagina gitHub dove trovare il codice che utilizzeremo durante il video, e il pdf con il riassunto dei punti chiave e i link per gli eventuali approfondimenti. Ma prima di salutarvi vi ricordo di mettere un like e iscrivermi al canale se non l'avete ancora fatto e come sempre fate domande, tante domande... ma soprattutto sperimentate gente, sperimentate...

GoodByte Alla Prossima!