Automazione delle Interazioni Grafiche

Cosa imparerai

- Introduzione a PyAutoGUI e le sue caratteristiche.
- Controllo del mouse: movimento, click, scroll.
- Controllo della tastiera: input di testo, tasti speciali.
- Acquisizione di screenshot e riconoscimento di immagini.
- Best practices per la scrittura di script PyAutoGUI.

Automazione delle Interazioni Grafiche

Introduzione a PyAutoGUI

PyAutoGUI è una libreria Python che permette di controllare il mouse e la tastiera per automatizzare le interazioni con l'interfaccia grafica del sistema operativo.

È utile per:

- Automatizzare compiti ripetitivi.
- Creare bot per giochi o applicazioni.
- Eseguire test automatici dell'interfaccia utente.
- Semplificare operazioni complesse.

PyAutoGUI funziona su Windows, macOS e Linux.

Automazione delle Interazioni Grafiche

Caratteristiche Principali

- **Controllo del Mouse:** Muovi il mouse, clicca, trascina, fai scroll.
- **Controllo della Tastiera:** Invia testo, premi tasti speciali (Ctrl, Alt, Shift, ecc.).
- **Screenshot:** Cattura schermate complete o parziali.
- **Image Recognition:** Individua immagini sullo schermo.
- **Pixel Access:** Leggi il colore di un pixel specifico.
- **Fail-safe:** Meccanismo di emergenza per interrompere lo script.

Automazione delle Interazioni Grafiche

Installazione e Configurazione

Per installare PyAutoGUI, utilizza pip:

```
pip install pyautogui
```

Importante:

Su Linux, potrebbe essere necessario installare `python3-tk` e `python3-dev`:

```
sudo apt-get update
sudo apt-get install python3-tk python3-dev
```

Automazione delle Interazioni Grafiche

Controllo del Mouse

Attraverso le funzioni di *pyautogui* è possibile simulare diverse operazioni con il mouse:

- leggere la posizione del mouse
- spostare il mouse
- effettuare click
- scrollare
- altro...

Automazione delle Interazioni Grafiche

Controllo del Mouse

Recuperare la posizione

Di seguito un esempio di come recuperare la posizione attuale del mouse.

```
import pyautogui

# Recupera la posizione attuale del mouse
x, y = pyautogui.position()

# Stampa la posizione
print(f"Posizione del mouse: x={x}, y={y}")
```

Automazione delle Interazioni Grafiche

Controllo del Mouse

Muovere il mouse

Di seguito un esempio di come spostare il puntatore del mouse.

```
# Muove il mouse a (100, 200) in 1 secondo
pyautogui.moveTo(100, 200, duration=1)

# Muove il mouse di 50 pixel verso il basso in 0.5 secondi
pyautogui.moveRel(0, 50, duration=0.5)
```

> ***NOTA:*** `moveRel` considera la posizione attuale del mouse.

Automazione delle Interazioni Grafiche

Controllo del Mouse

Effettuare click

Di seguito alcuni esempi di come effettuare i vari click:

```
pyautogui.click()           # Click sinistro nella posizione attuale
pyautogui.click(100, 200)   # Click sinistro a (100, 200)
pyautogui.rightClick()      # Click destro
pyautogui.doubleClick()     # Doppio click
pyautogui.mouseDown()       # Premi il pulsante del mouse
pyautogui.mouseUp()         # Rilascia il pulsante del mouse
```

Automazione delle Interazioni Grafiche

Controllo del Mouse

Scrollare

Se volessimo invece simulare lo scrolling, possiamo usare:

```
pyautogui.scroll(100)      # Scrolla verso l'alto di 100 "unità"
pyautogui.scroll(-100)     # Scrolla verso il basso di 100 "unità"
```

Automazione delle Interazioni Grafiche

Controllo della Tastiera

Anche per l'iterazione con la tastiera sono previste diverse funzioni:

- Scrittura del testo
- Pressione di singoli tasti
- Utilizzo di hotkey

> Fare attenzione all'encodig per certi caratteri come ad esempio `@` per i caratteri speciali è possibile gestire la copia dei caratteri tramite la libreria `pyperclip` e poi simulare la loro pressione.

Automazione delle Interazioni Grafiche

Controllo della Tastiera

Scrivere del testo

In questo esempio vediamo come scrivere del testo nella posizione attuale.

```
# Scrive il testo lentamente
pyautogui.typewrite("Hello, world!", interval=0.25)
```

> ***NOTA:*** Avolte è necessario simulare la scrittura umana per mantenere evitare di essere riconosciuti e dunque bloccati come *robot*, o anche per gestire i caricamenti dei campi *autocomplete* etc...

Automazione delle Interazioni Grafiche

Controllo della Tastiera

Pressione di Tasti

Avolte è necessario simulare la pressione dei tasti anziché l'inserimento di un testo. In questi casi:

```
# Premi il tasto "Invio"
pyautogui.press('enter')

# Premi "Ctrl" e "A" contemporaneamente (selezione completa)
pyautogui.press(['ctrl', 'a'])
```

Automazione delle Interazioni Grafiche

Controllo della Tastiera

Pressione di Tasti

Se è necessario controllare pressione e rilascio:

```
 # Premi il tasto "Shift"
pyautogui.keyDown('shift')
# Scrivi "HELLO" (maiuscolo)
pyautogui.typewrite('hello')
# Rilascia il tasto "Shift"
pyautogui.keyUp('shift')
```

Automazione delle Interazioni Grafiche

Controllo della Tastiera

Tasti Speciali

E' anche possibile utilizzare gli hotkey:

```
# Copia (Ctrl+C)
pyautogui.hotkey('ctrl', 'c')

# Incolla (Ctrl+V)
pyautogui.hotkey('ctrl', 'v')
```

Automazione delle Interazioni Grafiche

Acquisizione di Screenshot e Riconoscimento di Immagini

Screenshot

Molte volte può essere utile tracciare e salvare le videate. In questo caso si possono salvare gli screenshot:

```
# Cattura l'intero schermo
screenshot = pyautogui.screenshot()
screenshot.save("screenshot.png")
# Cattura una regione specifica
screenshot_area = pyautogui.screenshot(region=(0, 0, 300, 400))
screenshot_area.save("screenshot_area.png")
```

Automazione delle Interazioni Grafiche

Acquisizione di Screenshot e Riconoscimento di Immagini

Image Recognition

```
# Invia la mail tramite click di outlook_send.png
send_location = pyautogui.locateOnScreen('outlook_send.png', region=(0, 0,
640, 360))
if send_location is None:
    print("Pulsante 'Invia' non trovato.")
    return
# centro il mouse e faccio click
send_x, send_y = pyautogui.center(send_location)
pyautogui.click(send_x, send_y)
```

Automazione delle Interazioni Grafiche

Alcune raccomandazioni

Coordinate Assolute vs. Relative: Usa coordinate assolute (`moveTo(x, y)`) per posizioni fisse, coordinate relative (`moveRel(xOffset, yOffset)`) per spostamenti rispetto alla posizione corrente.

`time.sleep()`: Usa `time.sleep()` per dare tempo alle applicazioni di rispondere alle azioni automatizzate.

Error Handling: Implementa la gestione degli errori per intercettare eccezioni e comportamenti inattes

Logging: Registra le azioni eseguite dallo script per facilitare il debugging e il monitoraggio.

Riconoscimento di Immagini: Usa `confidence` per controllare la precisione della corrispondenza dell'immagine. Installa `opencv-python` per migliorare l'accuratezza del riconoscimento.

Commenti: Aggiungi commenti al codice per spiegare la logica e il funzionamento dello script.

Automazione delle Interazioni Grafiche

Domande

Automazione delle Interazioni Grafiche

Approfondimenti

Leggere la guida *quickstart* della Documentazione ufficiale

Automazione delle Interazioni Grafiche

ese01: Scrivi uno script che sposta il mouse al centro dello schermo e fa un click sinistro.

ese02: Crea uno script che apre un editor di testo, scrive "Ciao mondo!" e salva il file con il nome "ciao.txt" (assicurati che l'editor di testo sia già aperto e pronto).

ese03: Scrivi uno script che cattura un'area specifica dello schermo (es: un pulsante) e salva l'immagine in un file.

ese04: Scrivi uno script che apre outlook, crea un messaggio di posta verso l'indirizzo `l.salzone@alfasoft.it` e lo invia. Inserisci anche oggetto e corpo del messaggio.