Traducido del inglés al español - www.onlinedoctranslator.com

Documentación de PyAutoGUI

Al Sweigart

14 de septiembre de 2021

Contenido

PyAutoGUI permite que sus scripts de Python controlen el mouse y el teclado para automatizar las interacciones con otras aplicaciones. La API está diseñada para ser simple. PyAutoGUI funciona en Windows, macOS y Linux, y se ejecuta en Python 2 y 3.

Para instalar con pip, ejecutepip instala pyautogui.Ver el*Instalación*Página para más detalles. El código

fuente está disponible en:https://github.com/asweigart/pyautogui

PyAutoGUI tiene varias características:

- Mover el ratón y hacer clic en las ventanas de otras aplicaciones.
- Envío de pulsaciones de teclas a aplicaciones (por ejemplo, para rellenar formularios).
- Tomar capturas de pantalla y, dada una imagen (por ejemplo, de un botón o casilla de verificación), buscarla en la pantalla.
- Localizar la ventana de una aplicación y moverla, redimensionarla, maximizarla, minimizarla o cerrarla (actualmente, solo en Windows).
- Mostrar cuadros de alerta y mensajes.

Aquí estáUn video de YouTube de un bot que juega automáticamente al juego Sushi Go RoundEl robot observa la ventana de la aplicación del juego y busca imágenes de pedidos de sushi. Cuando encuentra una, hace clic en los botones de ingredientes para preparar el sushi. También hace clic en el teléfono dentro del juego para pedir más ingredientes según sea necesario. El robot es completamente autónomo y puede terminar los siete días del juego. Este es el tipo de automatización que PyAutoGUI es capaz de hacer.

CAPÍTULO 1

Ejemplos

```
>>> importarguia automática de py

>>> ancho de pantalla, alto de pantalla=guia automática de py.tamaño()#Obtener el tamaño del primario
→monitor.
>>> ancho de pantalla, alto de
pantalla (2560, 1440)

>>> actualRatónX, actualRatónY=guia automática de py.posición()#Obtenga la posición XY del
→ratón.
>>> actualRatónX, actualRatónY (1314,
    345)

>>> guia automática de py.moverA(100,150)#Mueva el ratón a las coordenadas XY.

>>> guia automática de py.hacer clic()                  # Haga clic con el ratón.
>>> guia automática de py.hacer clic(100,200)           # Mueva el mouse a las coordenadas XY y haga clic en él.
>>> guia automática de py.hacer clic('botón.png')#Busque dónde aparece button.png en la pantalla y
→Haz clic en él.

>>> guia automática de py.mover(400,0)                  # Mueva el ratón 400 píxeles a la derecha de su
→Posición actual.
>>> guia automática de py.doble clic()                  # Haga doble clic con el ratón.
>>> guia automática de py.moverA(500,500, duración=2, entre=guia automática de py.Facilidad de entrada y salida en cuadrantes          # Usar
→Función de interpolación/suavizado para mover el mouse durante 2 segundos.

>>> guia automática de py.escribir('¡Hola Mundo!', intervalo=0,25)#Escriba con pausa de un cuarto de segundo
→entre cada tecla
>>> guia automática de py.prensa('Esc')                 # Presione la tecla Esc. Todos los nombres de las teclas están en pyautogui.
→NOMBRES DE CLAVE

>>> conguia automática de py.sostener('cambio'):#Mantenga presionada la tecla Shift. pyautogui.press(['left',
        'left', 'left', 'left']) # Presione la tecla de flecha izquierda
→4 veces.
>>> #  La tecla Shift se suelta automáticamente.
```

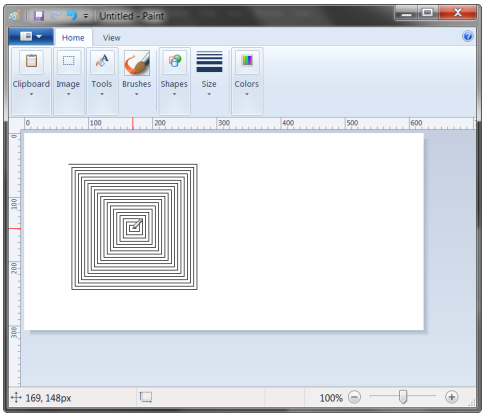
(continúa en la página siguiente)

(continúa de la página anterior)

```
>>> guia automática de py.tecla de acceso rápido('control','do')#Presione la combinación de teclas de acceso rápido Ctrl-C.

>>> guia automática de py.alerta('Éste es el mensaje a mostrar.')#Hacer que aparezca un cuadro de alerta y
...pausar el programa hasta que se haga clic en Aceptar.
```

Este ejemplo arrastra el mouse en forma de espiral cuadrada en MS Paint (o cualquier programa de dibujo gráfico):

```
>>> distancia=200
>>> mientrasdistancia>0: pyautogui.drag(distancia, 0,
        duración=0.5) distancia -= 5                    # moverse a la derecha

        pyautogui.drag(0, distancia, duración=0.5)      # moverse hacia abajo
        pyautogui.drag(-distancia, 0, duración=0.5) distancia -= 5    # moverse a la izquierda

        pyautogui.drag(0, -distancia, duración=0.5)     # subir
```



La ventaja de utilizar PyAutoGUI, a diferencia de un script que genera directamente el archivo de imagen, es que puedes utilizar las herramientas de pincel que proporciona MS Paint.

CAPÍTULO 2

FAQ: Preguntas frecuentes

Envíe preguntas aal@inventwithpython.com

P: ¿Puede PyAutoGUI funcionar en aplicaciones de Android, iOS o tabletas/teléfonos inteligentes?

R: Lamentablemente no. PyAutoGUI solo funciona en Windows, macOS y Linux.

P: ¿PyAutoGUI funciona en configuraciones de varios monitores?

R: No, en este momento PyAutoGUI solo maneja el monitor

principal. P: ¿PyAutoGUI realiza OCR?

R: No, pero es una característica que está en la hoja de ruta.

P: ¿Puede PyAutoGUI registrar teclas o detectar si una tecla está presionada en ese momento?

R: No, PyAutoGUI no puede hacer esto actualmente.

CAPÍTULO3

Al igual que las escobas encantadas del Aprendiz de brujo programadas para llenar (y luego sobrellenar) la bañera con agua, un error en el programa podría hacer que se descontrole. Es difícil usar el ratón para cerrar un programa si el cursor del ratón se mueve solo.

Como medida de seguridad, se habilita una función de seguridad por defecto. Cuando se llama a una función de PyAutoGUI, si el mouse está en cualquiera de las cuatro esquinas del monitor principal, se generará unpyautogui.FailSafeException.Hay un retraso de una décima de segundo después de llamar a cada función de PyAutoGUI para darle tiempo al usuario de golpear el mouse contra una esquina para activar el mecanismo de seguridad.

Puede desactivar esta función de seguridad configurandopyautogui.FAILSAFE = Falso.LE RECOMIENDO ENCARECIDAMENTE QUE NO DESACTIVE EL FAILSAFE.

El retraso de un décimo de segundo lo establece elpyautogui.PAUSAconfiguración, que es0,1por defecto. Puedes cambiar este valor. También hay unapyautogui.TIEMPO DE RECUPERACIÓN DE DARWINConfiguración que agrega un retraso adicional en macOS

después de los eventos del teclado y del mouse, ya que el sistema operativo parece necesitar un retraso después de que PyAutoGUI emita estos eventos. Está configurado para0,01De forma predeterminada, se agrega un retraso adicional de centésimas de segundo.

Contenido:

3.1 Instalación

Para instalar PyAutoGUI, instale elguia automática de pypaquete de PyPI ejecutandopip instala pyautogui (en Windows) opip3 instala pyautogui (en macOS y Linux). (En macOS y Linux,pepita(Se refiere a la herramienta pip de Python 2).

A continuación se encuentran las instrucciones específicas del sistema operativo.

3.1.1 Ventanas

En Windows, puedes utilizar elpy.exePrograma para ejecutar la última versión de Python:

```
py -m pip install pyautogui
```

Si tiene varias versiones de Python instaladas, puede seleccionar cuál con un argumento de línea de comando.Pi.Por ejemplo, para Python 3.8, ejecute:

```
py -3.8 -m pip instalar pyautogui (Esto es lo
```

mismo que correr(pip instala pyautogui.)

3.1.2 macOS

En macOS y Linux, debes ejecutarPython3:

```
python3 -m pip instalar pyautogui
```

Si está ejecutando El Capitan y tiene problemas al instalar pyobjc, intente:

```
MACOSX_DEPLOYMENT_TARGET=10.11 instalación de pip pyobjc
```

3.1.3 Linux

En macOS y Linux, debes ejecutarPython3:

```
python3 -m pip instalar pyautogui
```

En Linux, además es necesario instalar elescrotoaplicación, así como Tkinter:

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

PyAutoGUI instala los módulos de los que depende, incluidos PyTweening, PyScreeze, PyGetWindow, PymsgBox y MouseInfo.

3.2 Hoja de trucos

Esta es una referencia de inicio rápido para usar PyAutoGUI. PyAutoGUI es un módulo de automatización de GUI multiplataforma que funciona en Python 2 y 3. Puede controlar el mouse y el teclado, así como realizar un reconocimiento básico de imágenes para automatizar tareas en su computadora.

Todos los argumentos de palabras clave en los ejemplos de esta página son opcionales.

```
>>> importarguia automática de py
```

PyAutoGUI funciona en Windows/Mac/Linux y en Python 2 y 3. Instálelo desde PyPI conpip instala pyautogui.

3.2.1 Funciones generales

```
>>> guia automática de py.posición()#ratón actual x e y (968, 56)

>>> guia automática de py.tamaño()#Resolución de pantalla actual ancho y alto (1920,
    1080)
>>> guia automática de py.en pantalla(x, y)#Verdadero si x e y están dentro de la pantalla. Verdadero
```

3.2.2 Dispositivos de seguridad

Configure una pausa de 2,5 segundos después de cada llamada a PyAutoGUI:

```
>>> importarguia automática de py
>>> guia automática de py.PAUSA=2.5
```

Cuando el modo a prueba de fallos está activadoVerdadero,Al mover el ratón hacia la esquina superior izquierda, aparecerá unpyautogui.Excepción a prueba de fallos que puede abortar su programa:

```
>>> importarguia automática de py
>>> guia automática de py.A PRUEBA DE FALLOS=Verdadero
```

3.2.3 Funciones del ratón

Las coordenadas XY tienen origen 0, 0 en la esquina superior izquierda de la pantalla. X aumenta hacia la derecha, Y aumenta hacia abajo.

```
>>> guia automática de py.moverA(x, y, duración=num_segundos)#Mueva el ratón a las coordenadas XY
→num_second segundos
>>> guia automática de py.moveRel(xOffset, yOffset, duración=num_segundos)#mover el ratón relativamente
→a su posición actual
```

Siduraciónes 0 o no especificado, el movimiento es inmediato. Nota: arrastrar en Mac no puede ser inmediato.

```
>>> guia automática de py.dragTo(x, y, duración=num_segundos)#Arrastre el ratón hacia XY
>>> guia automática de py.dragRel(xOffset, yOffset, duración=num_segundos)#Arrastre el ratón relativamente
→a su posición actual
```

Vocaciónhacer clic()simplemente hace clic con el mouse una vez con el botón izquierdo en la ubicación actual del mouse, pero los argumentos de palabras clave pueden cambiar eso:

```
>>> guia automática de py.haga clic en (x)=moveToX, sí=moveToY, clics=num_of_clicks, intervalo=segundos_entre_
↪clics, botón='izquierda')
```

ElbotónEl argumento de palabra clave puede ser 'izquierda', 'medio',o 'bien'.

Todos los clics se pueden realizar conhacer clic(),pero estas funciones existen para facilitar la lectura. Los argumentos de las palabras clave son opcionales:

```
>>> guia automática de py.clic derecho(x)=moveToX, sí=moverAY)
>>> guia automática de py.clic medio(x)=moveToX, sí=moverAY)
>>> guia automática de py.doble clic(x=moveToX, sí=moverAY)
>>> guia automática de py.triple clic(x)=moveToX, sí=moverAY)
```

El desplazamiento positivo se desplazará hacia arriba, el desplazamiento negativo se desplazará hacia abajo:

```
>>> guia automática de py.scroll(cantidad_a_desplazarse, x=moveToX, sí=moverAY)
```

Los eventos de subir y bajar botones individuales se pueden llamar por separado:

```
>>> guia automática de py.ratónAbajo(x=moveToX, sí=moveToY, botón='izquierda')
>>> guia automática de py.ratónArriba(x=moveToX, sí=moveToY, botón='izquierda')
```

3.2.4 Funciones del teclado

Las pulsaciones de teclas van a cualquier lugar donde se encuentre el cursor del teclado en el momento de la llamada a la función.

```
>>> guia automática de py.escribir a máquina('¡Hola Mundo!\norte', intervalo=segundos_entre_teclas)#Útil para
↪Al ingresar texto, la nueva línea es Enter
```

También se puede pasar una lista de nombres de claves:

```
>>> guia automática de py.escribir a máquina(['a','b','do','izquierda','retroceso','ingresar','f1'],
↪intervalo=segundos_entre_teclas)
```

La lista completa de nombres clave se encuentra enpyautogui.TECLAS_DEL_TECLADO.

Las teclas de acceso rápido del teclado como Ctrl-S o Ctrl-Shift-1 se pueden usar pasando una lista de nombres de teclas atecla de acceso rápido():

```
>>> guia automática de py.tecla de acceso rápido('control','do')     # ctrl-c para copiar
>>> guia automática de py.tecla de acceso rápido('control','V')      # ctrl-v para pegar
```

Los eventos de subir y bajar botones individuales se pueden llamar por separado:

```
>>> guia automática de py.keyDown(nombre_clave)
>>> guia automática de py.keyUp(nombre_clave)
```

3.2.5 Funciones del cuadro de mensajes

Si necesita pausar el programa hasta que el usuario haga clic en Aceptar en algo, o desea mostrar alguna información al usuario, las funciones del cuadro de mensaje tienen nombres similares a los de JavaScript:

```
>>> guia automática de py.alerta('Esto muestra un texto con un botón Aceptar.')
>>> guia automática de py.confirmar('Esto muestra texto y tiene un botón Aceptar y Cancelar.') 'DE
    ACUERDO'
>>> guia automática de py.inmediato('Esto permite al usuario escribir una cadena y presionar
Aceptar.') "Esto es lo que escribí."
```

Elinmediato()La función retornaráNingunoSi el usuario hizo clic en Cancelar.

3.2.6 Funciones de captura de pantalla

PyAutoGUI utiliza Pillow/PIL para sus datos relacionados con imágenes.

En Linux, debes ejecutarsudo apt-get install scrotpara utilizar las funciones de captura de pantalla.

```
>>> guia automática de py.captura de pantalla()#devuelve un objeto de imagen de almohada/
PIL <PIL.Image.Image modo de imagen=RGB tamaño=1920x1080 en 0x24C3EF0>
>>> guia automática de py.captura de pantalla('foo.png')#devuelve un objeto de imagen Pillow/PIL y lo guarda
→a un archivo
<PIL.Image.Image modo de imagen=RGB tamaño=1920x1080 en 0x31AA198>
```

Si tiene un archivo de imagen de algo en lo que desea hacer clic, puede encontrarlo en la pantalla conlocalizarEnPantalla().

```
>>> guia automática de py.localizarEnPantalla('se parece a esto.png')#devuelve (izquierda, arriba, ancho,
→altura) del primer lugar se encuentra (863,
417, 70, 13)
```

EllocalizarTodoEnPantalla()La función devolverá un generador para todas las ubicaciones en las que se encuentre en la pantalla:

```
>>> paraienguia automática de py.localizarTodoEnPantalla('se parece a esto.png')
...
...
(863, 117, 70, 13) (623,
137, 70, 13) (853, 577,
70, 13) (883, 617, 70, 13)
(973, 657, 70, 13) (933,
877, 70, 13)
```

```
>>> lista(guia automática de py.localizarTodoEnPantalla('se parece a esto.png'))
[(863, 117, 70, 13), (623, 137, 70, 13), (853, 577, 70, 13), (883, 617, 70, 13), (973,
→ 657, 70, 13), (933, 877, 70, 13)]
```

EllocalizarCenterOnScreen()La función simplemente devuelve las coordenadas XY del punto medio donde se encuentra la imagen en la pantalla:

```
>>> guia automática de py.localizarCentroEnPantalla('se parece a esto.png')#devuelve el centro x e y (898,
    423)
```

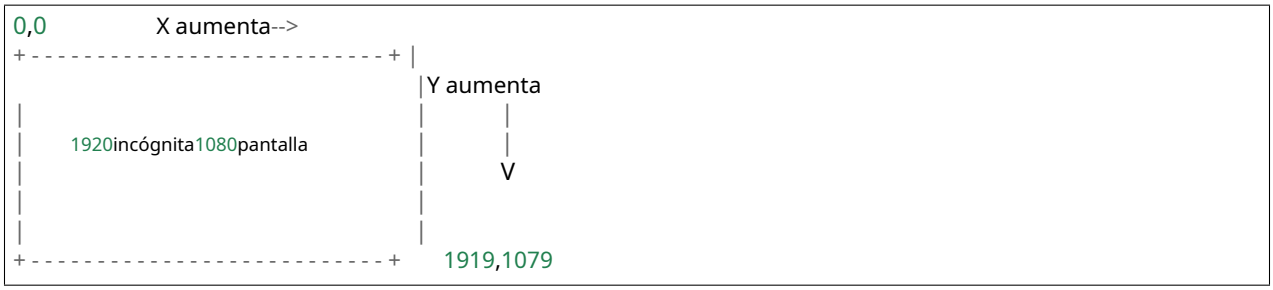
Estas funciones retornanNingunoSi no se puede encontrar la imagen en la pantalla.

Nota: Las funciones de localización son lentas y pueden tardar uno o dos segundos.

3.3 Funciones de control del ratón

3.3.1 La posición de la pantalla y del ratón

Las posiciones en la pantalla se indican mediante coordenadas cartesianas X e Y. La coordenada X comienza en 0 en el lado izquierdo y aumenta hacia la derecha. A diferencia de lo que ocurre en matemáticas, la coordenada Y comienza en 0 en la parte superior y aumenta hacia abajo.



El píxel en la esquina superior izquierda está en las coordenadas 0, 0. Si la resolución de su pantalla es 1920 x 1080, el píxel en la esquina inferior derecha será 1919, 1079 (ya que las coordenadas comienzan en 0, no en 1).

El tamaño de la resolución de la pantalla lo devuelve eltamaño()funciona como una tupla de dos números enteros. Las coordenadas X e Y actuales del cursor del ratón son devueltas por elposición()función.

Por ejemplo:

```
> > > guia automática de py.
tamaño() (1920, 1080)
> > > guia automática de py.
posición() (187, 567)
```

A continuación se muestra un breve programa de Python 3 que imprimirá constantemente la posición del cursor del mouse:

```
# !python3
importarguia automática de py,sistema imprimir(
'Presione Ctrl-C para salir.') intentar:


    Aunque es cierto:
        x, y=guia automática de py.posición()
        posiciónStr='X: '+cadena(incógnita).solo(4)+' Y: '+cadena(y).solo(4) imprimir(posiciónStr,
        fin='')
        imprimir('\b'*lente(posiciónStr), fin='', rubor=Verdadero) excepto
Interrupción del teclado:
    imprimir('\norte')
```

Aquí está la versión de Python 2:

```
# ! pitón
importarguia automática de py,sistema imprimir(
'Presione Ctrl-C para salir.') intentar:


    Aunque es cierto:
        x, y=guia automática de py.posición()
        posiciónStr='X: '+cadena(incógnita).solo(4)+' Y: '+cadena(y).solo(4) imprimirposiciónStr,

        imprimir'\b'*(lente(posiciónCadena)+2), sistema.
        salida estándar.enjuagar() excepto Interrupción del
teclado:
    imprimir'\norte'
```

Para verificar si las coordenadas XY están en la pantalla, páselas (ya sea como dos argumentos enteros o como una única tupla/lista de argumentos con dos enteros) a laen pantalla()función, que retornaráVerdaderosi están dentro de los límites de la pantalla yFALSO Si no, por ejemplo:

```
> > > guia automática de py.en pantalla(0,0)
    Verdadero
```

(continúa en la página siguiente)

(continúa de la página anterior)

```
>>> guia automática de py.en pantalla(0,-1)
    FALSO
>>> guia automática de py.en pantalla(0,99999999)
    FALSO
>>> guia automática de py.
tamaño() (1920, 1080)
>>> guia automática de py.en pantalla(1920,1080)
    FALSO
>>> guia automática de py.en pantalla(1919,1079)
    Verdadero
```

3.3.2 Movimiento del ratón

ElmoverA()La función moverá el cursor del mouse a las coordenadas enteras X e Y que le pase.NingunoSe puede pasar un valor a una coordenada para indicar "la posición actual del cursor del mouse". Por ejemplo:

```
>>> guia automática de py.moverA(100,200)       # mueve el mouse a X de 100, Y de 200.
>>> guia automática de py.moverA(Ninguno,500)   # mueve el mouse a X de 100, Y de 500.
>>> guia automática de py.moverA(600,Ninguno)   # mueve el mouse a X de 600, Y de 500.
```

Normalmente, el cursor del ratón se moverá instantáneamente a las nuevas coordenadas. Si desea que el ratón se mueva gradualmente a la nueva ubicación, pase un tercer argumento para la duración (en segundos) que debe tomar el movimiento. Por ejemplo:

```
>>> guia automática de py.moverA(100,200,2)     # mueve el mouse a X de 100, Y de 200 durante 2 segundos
```

(Si la duración es menor apyautogui.DURACIÓN MÍNIMAEl movimiento será instantáneo. Por defecto, pyautogui.DURACIÓN MÍNIMAes 0,1.)

Si desea mover el cursor del mouse sobre unos pocos píxeles*relativo*a su posición actual, utilice elmover()función. Esta función tiene parámetros similares amoverA().Por ejemplo:

```
>>> guia automática de py.moverA(100,200)       # mueve el mouse a X de 100, Y de 200.
>>> guia automática de py.mover(0,50)           # Mueva el ratón 50 píxeles hacia abajo.
>>> guia automática de py.mover(-30,0)          # Mueva el mouse 30 píxeles hacia la izquierda.
>>> guia automática de py.mover(-30,Ninguno)    # Mueva el mouse 30 píxeles hacia la izquierda.
```

3.3.3 Arrastre del ratón

PyAutoGUIarrastrarA()yarrastrar()Las funciones tienen parámetros similares a losmoverA()ymover()funciones. Además, tienen una botónPalabra clave que se puede configurar como 'izquierda', 'medio',y 'bien'para saber qué botón del ratón hay que mantener pulsado mientras se arrastra. Por ejemplo:

```
>>> guia automática de py.arrastrarA(100,200, botón='izquierda')      # Arrastre el mouse hacia X de 100, Y de 200
↪mientras mantiene presionado el botón izquierdo del mouse
>>> guia automática de py.arrastrarA(300,400,2, botón='izquierda')    # Arrastre el mouse hacia X de 300, Y de 400
↪más de 2 segundos mientras mantiene presionado el botón izquierdo del mouse
>>> guia automática de py.arrastrar(30,0,2, botón='bien')             # Arrastre el mouse hacia la izquierda 30 píxeles sobre 2
↪segundos mientras mantiene presionado el botón derecho del mouse
```

3.3.4 Funciones de interpolación y suavizado

La interpolación es una función adicional que permite hacer que los movimientos del ratón sean más elegantes. Probablemente puedas omitir esta sección si no te interesa.

Una función de interpolación o de aceleración dicta el progreso del mouse a medida que se mueve hacia su destino. Normalmente, cuando se mueve el mouse durante un período de tiempo, el mouse se mueve directamente hacia el destino en línea recta a una velocidad constante. Esto se conoce como*interpolación lineal*o*flexibilización lineal*función.

PyAutoGUI tiene otras funciones de interpolación disponibles en elguia automática de pymódulo. Elpyautogui.easeInQuad La función se puede pasar como cuarto argumento amoverA(), mover(), arrastrarA(),yarrastrar()funciones para que el cursor del ratón comience a moverse lentamente y luego acelere hacia el destino. La duración total sigue siendo la misma que el argumento pasado a la función.pyautogui.easeOutQuades lo contrario: el cursor del ratón comienza a moverse rápido pero disminuye su velocidad a medida que se acerca al destino.pyautogui.easeOutElasticsobrepasará el destino y se moverá "con banda elástica" de un lado a otro hasta que se estabilice en el destino.

Por ejemplo:

```
> > > guia automática de py.moverA(100,100,2, pyautogui.facilidadEnQuad)          # empieza lento, termina rápido
> > > guia automática de py.moverA(100,100,2, pyautogui.facilidadOutQuad)         # empieza rápido, termina lento
> > > guia automática de py.moverA(100,100,2, pyautogui.Facilidad de entrada y salida en cuadrantes  # empezar y terminar rápido,
→lento en el medio
> > > guia automática de py.moverA(100,100,2, pyautogui.facilidad de rebote)      # rebote al final
> > > guia automática de py.moverA(100,100,2, pyautogui.facilidadEnElasticidad)   # banda elástica al final
```

Estas funciones de interpolación se copian del módulo PyTweening de Al Sweigart:https://pypi.python.org/pypi/ Interpolación de Py https://github.com/asweigart/pytweeningNo es necesario instalar este módulo para utilizar las funciones de interpolación.

Si desea crear su propia función de interpolación, defina una función que tome un único argumento flotante entre0.0 (que representa el inicio del viaje del ratón) y1.0 (representa el final del recorrido del ratón) y devuelve un valor flotante entre0.0y1.0.

3.3.5 Clics del ratón

Elhacer clic()La función simula un solo clic con el botón izquierdo del mouse en la posición actual del mouse. Un "clic" se define como presionar el botón hacia abajo y luego soltarlo hacia arriba. Por ejemplo:

```
> > > guia automática de py.hacer clic()#Haga clic con el ratón
```

Para combinar unmoverA()Llamar antes del clic, pasar números enteros para elincógnitayyargumento de palabra clave:

```
> > > guia automática de py.haga clic en (x)=100, y=200)#Muévete a 100, 200 y luego haz clic con el botón izquierdo del mouse.
→botón.
```

Para especificar un botón diferente del mouse para hacer clic, pase 'izquierda', 'medio',o 'bien'Para elbotónargumento de palabra clave:

```
> > > guia automática de py.haga clic en el botón='bien')#Haga clic derecho con el ratón
```

Para hacer varios clics, pase un número entero alclicsargumento de palabra clave. Opcionalmente, puede pasar un valor flotante o entero al intervaloArgumento de palabra clave para especificar la cantidad de pausa entre los clics en segundos. Por ejemplo:

```
> > > guia automática de py.clic(clics)=2)#haga doble clic con el botón izquierdo del ratón
> > > guia automática de py.clic(clics)=2, intervalo=0,25)#haga doble clic con el botón izquierdo del ratón,
→pero con una pausa de un cuarto de segundo entre clics
> > > guia automática de py.haga clic en el botón='bien', clics=3, intervalo=0,25)##Haga triple clic en el
→Botón derecho del ratón con una pausa de un cuarto de segundo entre clics
```

(continúa en la página siguiente)

(continúa de la página anterior)

Como un atajo conveniente, eldoble clic()La función realizará un doble clic con el botón izquierdo del ratón. También tiene la opciónx, y, intervalo,ybotónargumentos de palabras clave. Por ejemplo:

```
>>> guia automática de py.doble clic()#Realizar doble clic con el botón izquierdo
```

También hay unatriple clic()función con argumentos de palabras clave opcionales similares.

Elclic derecho()La función tiene opcionesincógnitayyargumentos de palabras clave.

3.3.6 Las funciones mouseDown() y mouseUp()

Los clics y arrastres del mouse se componen de presionar el botón del mouse hacia abajo y soltarlo hacia arriba. Si desea realizar estas acciones por separado, llame alratónAbajo()yratónArriba()funciones. Tienen las mismasx, y, y botón.Por ejemplo:

```
>>> guia automática de py.ratónAbajo(); pyautogui.ratónArriba()#hace lo mismo que un zurdo
→botón clic del ratón
>>> guia automática de py.botón mouseDown(='bien')#Presione el botón derecho hacia abajo
>>> guia automática de py.botón mouseUp(='bien', y=100, y=200)#Mueva el ratón a 100, 200,
→luego suelte el botón derecho hacia arriba.
```

3.3.7 Desplazamiento del ratón

La rueda de desplazamiento del ratón se puede simular llamando alvoluta()función y pasar un número entero de "clics" para desplazarse. La cantidad de desplazamiento en un "clic" varía entre plataformas. Opcionalmente, se pueden pasar números enteros para el incógnitayy Argumentos de palabras clave para mover el cursor del ratón antes de realizar el desplazamiento. Por ejemplo:

```
>>> guia automática de py.voluta(10)          # desplácese hacia arriba 10 "clics"
>>> guia automática de py.voluta(-10)         # desplácese hacia abajo 10 "clics"
>>> guia automática de py.voluta(10, y=100, y=100)       # Mueva el cursor del mouse a 100, 200 y luego desplácese
→hasta 10 "clics"
```

En las plataformas OS X y Linux, PyAutoGUI también puede realizar desplazamiento horizontal llamando a la función hscroll(). Por ejemplo:

```
>>> guia automática de py.desplazamiento horizontal(10)       # desplazarse hacia la derecha 10 "clics"
>>> guia automática de py.desplazamiento horizontal(-10)        # desplazarse hacia la izquierda 10 "clics"
```

Elvoluta()La función es un contenedor paradesplazamiento vertical(),que realiza desplazamiento vertical.

3.4 Funciones de control del teclado

3.4.1 La función write()

La función principal del teclado esescribir().Esta función escribirá los caracteres en la cadena que se pasa. Para agregar un intervalo de demora entre cada pulsación de tecla de carácter, pase un int o float para laintervaloargumento de palabra clave.

Por ejemplo:

```
>>> guia automática de py.escribir('¡Hola Mundo!')                    # imprime "¡Hola mundo!"
→ instantáneamente
>>> guia automática de py.escribir('¡Hola Mundo!', intervalo=0,25)    # imprime "¡Hola mundo!" con
→ Un cuarto de segundo de retraso después de cada carácter.
```

Solo puedes presionar teclas de un solo carácter conescribir(),Por ejemplo, no puedes presionar las teclas Shift o F1.

3.4.2 Las funciones press(), keyDown() y keyUp()

Para presionar estas teclas, llame alprensa()función y pasarle una cadena desde elpyautogui.TECLAS_DEL_TECLADOcomo entrar, esc, f1.Ver *TECLAS DEL TECLADO*.

Por ejemplo:

```
>>> guia automática de py.prensa('ingresar')     # presione la tecla Enter
>>> guia automática de py.prensa('f1')           # presione la tecla F1
>>> guia automática de py.prensa('izquierda')    # presione la tecla de flecha izquierda
```

Elprensa()La función es en realidad solo un contenedor para elteclaAbajo()yteclaArriba()Funciones que simulan presionar una tecla y luego soltarla. Estas funciones pueden llamarse por sí mismas. Por ejemplo, para presionar la tecla de flecha izquierda tres veces mientras se mantiene presionada la tecla Shift, llame a lo siguiente:

```
>>> guia automática de py.teclaAbajo('cambio')    # mantenga presionada la tecla shift
>>> guia automática de py.prensa('izquierda')     # presione la tecla de flecha izquierda
>>> guia automática de py.prensa('izquierda')     # presione la tecla de flecha izquierda
>>> guia automática de py.prensa('izquierda')     # presione la tecla de flecha izquierda
>>> guia automática de py.teclaArriba('cambio')   # suelte la tecla shift
```

Para presionar varias teclas de forma similar a lo queescribir()¿Pasa una lista de cadenas a?prensa().Por ejemplo:

```
>>> guia automática de py.prensa(['izquierda','izquierda','izquierda'])
```

O puedes configurar cuántas pulsacionesizquierda:

```
>>> guia automática de py.prensa('izquierda', prensas=3)
```

Para agregar un intervalo de retardo entre cada pulsación, pase un int o float para elintervaloargumento de palabra clave.

3.4.3 El administrador de contexto hold()

Para que sea más cómodo sostener una tecla, lasostener()La función se puede utilizar como un administrador de contexto y pasar una cadena desde elpyautogui.TECLAS_DEL_TECLADOcomomayúsculas, ctrl, alt,y esta clave se mantendrá durante la duración del conbloque de contexto. Ver *TECLAS DEL TECLADO*.

```
>>> conguia automática de py.sostener('cambio'):
        pyautogui.press(['izquierda', 'izquierda', 'izquierda'])
```

... es equivalente a este código:

```
>>> guia automática de py.teclaAbajo('cambio')    # mantenga presionada la tecla shift
>>> guia automática de py.prensa('izquierda')     # presione la tecla de flecha izquierda
>>> guia automática de py.prensa('izquierda')     # presione la tecla de flecha izquierda
>>> guia automática de py.prensa('izquierda')     # presione la tecla de flecha izquierda
>>> guia automática de py.teclaArriba('cambio')   # suelte la tecla shift
```

3.4.4 La función hotkey()

Para que sea más cómodo presionar teclas de acceso rápido o atajos del teclado,tecla de acceso rápido()Se pueden pasar varias cadenas de teclas que se presionarán en orden y luego se soltarán en orden inverso. Este código:

```
> > > guia automática de py.tecla de acceso rápido('control','cambio','Esc')
```

... es equivalente a este código:

```
> > > guia automática de py.teclaAbajo('control')
> > > guia automática de py.teclaAbajo('cambio')
> > > guia automática de py.teclaAbajo('Esc')
> > > guia automática de py.teclaArriba('Esc')
> > > guia automática de py.teclaArriba('cambio')
> > > guia automática de py.teclaArriba('control')
```

Para agregar un intervalo de retardo entre cada pulsación, pase un int o float para elintervaloargumento de palabra clave.

3.4.5 TECLAS DEL TECLADO

Las siguientes son las cadenas válidas para pasar apresionar(), teclaAbajo(), teclaArriba(),ytecla de acceso rápido()Funciones:

```
['\t','\norte','\r',' ','!','"','#','$','%','&',"'",'(',
')','*','+',',','-','.','/','0','1','2','3','4','5','6','7', '8','9',':',';','<','=','>','?','@','[','\\',']','^','_','`', 'a','b','do','d','mi','F','gramo',
"ella",'i',"yo",'que',"yo",'metro','norte','Oh', 'pag','q','o','s','Eso','tú','V','que','incógnita','y','de la','{','|','}','~',
'aceptar','agregar','alt','alt-izquierda','muy bien','aplicaciones','retroceso', 'retroceso del navegador','favoritos
del navegador','Navegador hacia adelante','navegador de inicio', 'actualizar navegador','búsqueda del
navegador','detener el navegador','Bloq Mayús','claro', 'convertir','control','Ctrl izquierda','Ctrl+derecha',
'decimal','del','borrar', 'dividir','abajo','fin','ingresar','Esc','escapar','ejecutar','f1','f10', 'f11','f12','f13','f14','f15',
'f16','f17','f18','f19','f2','f20', 'f21','f22','f23','f24','f3','f4','f5','f6','f7','f8','f9', 'final','función','hanguel','Hangul',
'hanja','ayuda','hogar','insertar','Junja', 'cana','kanji','lanzamiento de aplicación1','lanzamiento de aplicación2',
'correo de lanzamiento', 'selección de medios de lanzamiento','izquierda','cambio de modo','multiplicar',
'próxima pista', 'no converso','num0','num1','num2','num3','num4','num5','num6', 'num7','num8','número 9',
'bloqueo numérico','avance de página','retroceder página','pausa','pgdn', 'regresar a página','reproducirpausa',
'pista anterior','imprimir','imprimir pantalla','imprimir pantalla', 'prtsc','impresora','devolver','bien','bloqueo de
desplazamiento','seleccionar','separador', 'cambio','desplazamiento a la izquierda','cambiar a la derecha',
'dormir','espacio','detener','sustraer','pestaña', 'arriba','bajar volumen','silenciar volumen','subir volumen',
'ganar','ganar a la izquierda','ganador','yen', 'dominio','opción','opciónizquierda','opciónderecha']
```

3.5 Funciones del cuadro de mensajes

PyAutoGUI utiliza las funciones de cuadro de mensajes de PyMsgBox para ofrecer una forma multiplataforma y puramente Python de mostrar cuadros de mensajes de estilo JavaScript. Se proporcionan cuatro funciones de cuadro de mensajes:

3.5.1 La función alert()

```
> > > alerta(texto='', título='', botón='DE ACUERDO')
```

Muestra un cuadro de mensaje simple con texto y un solo botón Aceptar. Devuelve el texto del botón en el que se hizo clic.

3.5.2 La función confirm()

```
> > > confirmar(texto='', título='', botones=['DE ACUERDO','Cancelar'])
```

Muestra un cuadro de mensaje con los botones Aceptar y Cancelar. Se puede personalizar la cantidad y el texto de los botones. Devuelve el texto del botón en el que se hizo clic.

3.5.3 La función prompt()

```
> > > mensaje de texto='', título='', por defecto='')
```

Muestra un cuadro de mensaje con entrada de texto y botones Aceptar y Cancelar. Devuelve el texto ingresado o Ninguno si se hizo clic en Cancelar.

3.5.4 La función password()

```
> > > contraseña(texto='', título='', por defecto='', mascarilla='*')
```

Muestra un cuadro de mensaje con entrada de texto y botones Aceptar y Cancelar. Los caracteres escritos aparecen como *. Devuelve el texto ingresado o Ninguno si se hizo clic en Cancelar.

3.6 Funciones de captura de pantalla

PyAutoGUI puede tomar capturas de pantalla, guardarlas en archivos y ubicar imágenes dentro de la pantalla. Esto resulta útil si tienes una imagen pequeña de, por ejemplo, un botón en el que se debe hacer clic y deseas ubicarlo en la pantalla. Estas funciones las proporciona el módulo PyScreeze, que se instala con PyAutoGUI.

La función de captura de pantalla requiere el módulo Pillow. OS X utiliza elcaptura de pantallacomando, que viene con el sistema operativo. Linux utiliza elescrotocomando, que se puede instalar ejecutandosudo apt-get install scrot.

3.6.1 La función screenshot()

Vocacióncaptura de pantalla()devolverá un objeto Image (consulte la documentación del módulo Pillow o PIL para obtener más detalles). Si pasa una cadena con el nombre de un archivo, se guardará la captura de pantalla en un archivo y se devolverá como un objeto Image.

```
> > > importarguia automática de py
> > > yo1=guia automática de py.captura de pantalla()
> > > yo2=guia automática de py.captura de pantalla('mi_captura_de_pantalla.png')
```

En una pantalla de 1920 x 1080, lacaptura de pantalla()La función tarda aproximadamente 100 milisegundos: no es rápida, pero tampoco es lenta.

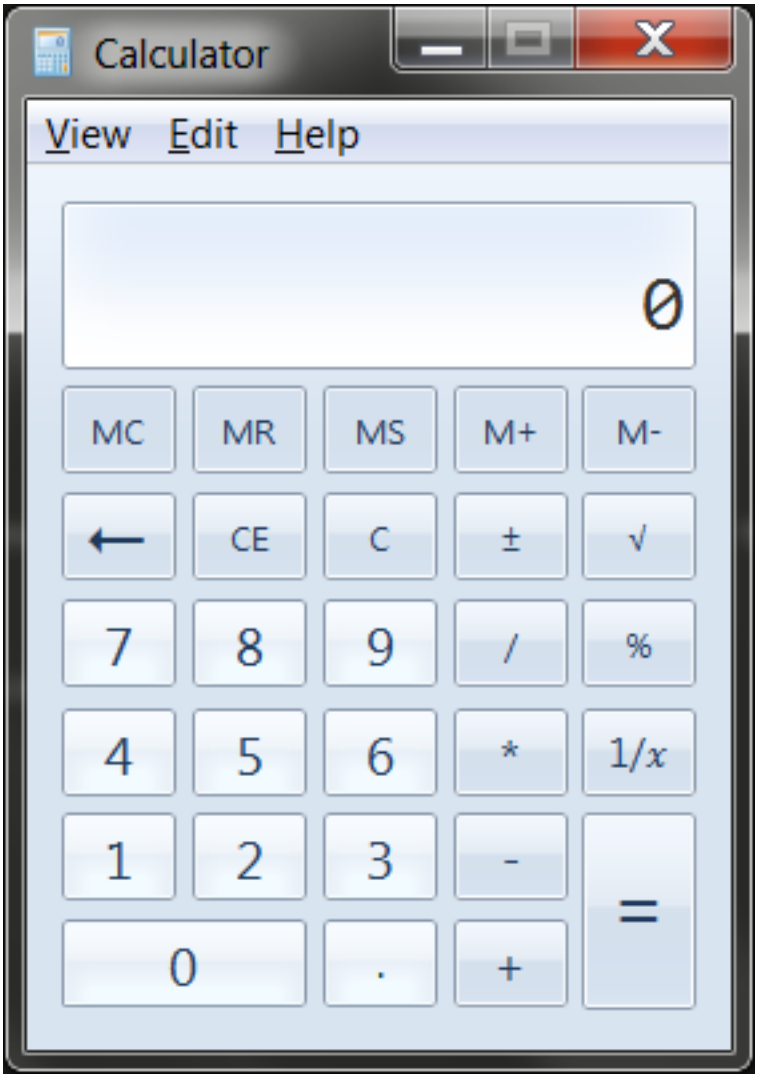
También hay una opciónregiónargumento de palabra clave, si no desea una captura de pantalla de toda la pantalla. Puede pasar una tupla de cuatro enteros de la izquierda, la parte superior, el ancho y la altura de la región a capturar:

```
>>> importar guia automática de py
>>> soy=guia automática de py.captura de pantalla(región=(0,0,300,400))
```

3.6.2 Las funciones de localización

NOTA: A partir de la versión 0.9.41, si las funciones de localización no pueden encontrar la imagen proporcionada, generarán un error. Excepción de imagen no encontradaEn lugar de regresarNinguno.

Puedes localizar visualmente algo en la pantalla si tienes un archivo de imagen del mismo. Por ejemplo, supongamos que la aplicación de calculadora se estaba ejecutando en tu computadora y se veía así:



No puedes llamar almoverA()yhacer clic()funciones si no conoce las coordenadas exactas de la pantalla donde se encuentran los botones de la calculadora. La calculadora puede aparecer en un lugar ligeramente diferente cada vez que se inicia, lo que hace que tenga que volver a encontrar las coordenadas cada vez. Sin embargo, si tiene una imagen del botón, como la imagen del botón 7:

...puedes llamar allocalizarEnPantalla('calc7key.png')Función para obtener las coordenadas de la pantalla. El valor de retorno es una tupla de 4 enteros: (izquierda, arriba, ancho, alto). Esta tupla se puede pasar acentro()para obtener las coordenadas X e Y en el centro de esta región. Si no se puede encontrar la imagen en la pantalla,localizarEnPantalla()aumenta Excepción de imagen no encontrada.

```
> > > importarguia automática de py
> > > Ubicación del botón 7=guia automática de py.localizarEnPantalla('calc7key.png')
> > > Ubicación del botón 7
Caja (izquierda=1416, superior=562, ancho=50, alto=41)
> > > botón7ubicación[0] 1416

> > > Ubicación del botón 7.izquierda
    1416
> > > botón7punto=guia automática de py.centro(ubicación del botón 7)
> > > botón7punto
Punto(x=1441, y=582)
> > > botón7punto[0]
    1441
> > > botón7punto.incógnita
    1441
> > > botón7x, botón7y=botón7punto
> > > guia automática de py.haga clic en (botón7x, botón7y)#hace clic en el centro de donde está el botón 7
→Fue encontrado
> > > guia automática de py.hacer clic('calc7key.png')#una versión abreviada para hacer clic en el centro de
→Dónde se encontró el botón 7
```

El opcionalconfianzaEl argumento de palabra clave especifica la precisión con la que la función debe ubicar la imagen en la pantalla. Esto resulta útil en caso de que la función no pueda ubicar una imagen debido a diferencias insignificantes en píxeles:

```
> > > importarguia automática de py
> > > Ubicación del botón 7=guia automática de py.localizarEnPantalla('calc7key.png', confianza=0.9)
> > > Ubicación del botón 7
Caja (izquierda=1416, superior=562, ancho=50, alto=41)
```

Nota:Necesitas tenerOpenCVinstalado para elconfianzaPalabra clave para trabajar.

EllocalizarCenterOnScreen()función combinalocalizarEnPantalla()ycentro():

```
> > > importarguia automática de py
> > > x, y=guia automática de py.localizarCentroEnPantalla('calc7key.png')
> > > guia automática de py.haga clic en (x, y)
```

En una pantalla de 1920 x 1080, las llamadas a la función de localización demoran aproximadamente 1 o 2 segundos. Esto puede ser demasiado lento para los videojuegos de acción, pero funciona para la mayoría de los propósitos y aplicaciones.

Hay varias funciones de "ubicación". Todas comienzan mirando la esquina superior izquierda de la pantalla (o imagen) y miran hacia la derecha y luego hacia abajo. Los argumentos pueden ser:

- locateOnScreen(imagen, escala de grises=Falso) -Devuelve las coordenadas (izquierda, superior, ancho, alto) de la primera instancia encontrada deimagenen la pantalla. LevantaExcepción de imagen no encontradaSi no se encuentra en la pantalla.
- locateCenterOnScreen(imagen, escala de grises=Falso) -Devuelve las coordenadas (x, y) del centro de la primera instancia encontrada de laimagenen la pantalla. LevantaExcepción de imagen no encontradaSi no se encuentra en la pantalla.
- locateAllOnScreen(imagen, escala de grises=Falso) -Devuelve un generador que produce tuplas (izquierda, superior, ancho, alto) para dónde se encuentra la imagen en la pantalla.
- locate(imagenaguja, imagenpajar, escala de grises=Falso) -Devuelve las coordenadas (izquierda, superior, ancho, alto) de la primera instancia encontrada deimagen de agujaenImagen de pajar. Aumenta

Excepción de imagen no encontradaSi no se encuentra en la pantalla.

- locateAll(imagenaguja, imagenpajar, escala de grises=Falso) -Devuelve un generador que produce tuplas (izquierda, superior, ancho, alto) para dondeimagen de agujase encuentra enImagen de pajar.

Las funciones "localizar todo" se pueden utilizar en bucles for o pasar alista():

```
> > > importarguia automática de py
> > > paraposiciónenguia automática de py.localizarTodoEnPantalla('algunBoton.png')
. . .     imprimir(posición)
. . .
(1101, 252, 50, 50) (59,
481, 50, 50) (1395, 640,
50, 50) (1838, 676, 50, 50)

> > > lista(guia automática de py.localizarTodoEnPantalla('algunBoton.png'))
[(1101, 252, 50, 50), (59, 481, 50, 50), (1395, 640, 50, 50), (1838, 676, 50, 50)]
```

Estas funciones de "localización" son bastante costosas; pueden tardar un segundo en ejecutarse. La mejor manera de acelerarlas es pasar unregión argumento (una tupla de 4 enteros de (izquierda, superior, ancho, alto)) para buscar solo en una región más pequeña de la pantalla en lugar de en la pantalla completa:

```
> > > importarguia automática de py
> > > guia automática de py.localizarEnPantalla('algunBoton.png', región=(0,0,300,400))
```

Coincidencia de escala de grises

Opcionalmente, puedes pasarescala de grises=Verdaderoa las funciones de localización para que aumenten ligeramente la velocidad (aproximadamente un 30 %). Esto desatura el color de las imágenes y capturas de pantalla, lo que acelera la localización pero puede provocar coincidencias con falsos positivos.

```
> > > importarguia automática de py
> > > Ubicación del botón 7=guia automática de py.localizarEnPantalla('calc7key.png', escala de grises=Verdadero)
> > > Ubicación del botón
7 (1416, 562, 50, 41)
```

Coincidencia de píxeles

Para obtener el color RGB de un píxel en una captura de pantalla, utilice el objeto Imagenobtenerpixel()método:

```
> > > importarguia automática de py
> > > soy=guia automática de py.captura de pantalla()
> > > soy.obtenerpixel((100,200))
(130, 135, 144)
```

O como una sola función, llame a lapíxel()Función PyAutoGUI, que es un contenedor para las llamadas anteriores:

```
> > > importarguia automática de py
> > > foto=guia automática de py.píxel(100,200)
> > > foto
RGB (rojo=130, verde=135, azul=144)
> > > pix[0]
130
> > > foto.rojo
130
```

Si solo necesita verificar que un solo píxel coincida con un píxel determinado, llame alpixelCoincide con el color()función, pasándole la coordenada X, la coordenada Y y la tupla RGB del color que representa:

```
>>> importarguia automática de py
>>> guia automática de py.pixelCoincide con el color(100,200, (130,135,144)) Verdadero

>>> guia automática de py.pixelCoincide con el color(100,200, (0,0,0))
    FALSO
```

El opcionaltoleranciaEl argumento de palabra clave especifica cuánto puede variar cada uno de los valores rojo, verde y azul sin dejar de coincidir:

```
>>> importarguia automática de py
>>> guia automática de py.pixelCoincide con el color(100,200, (130,135,144)) Verdadero

>>> guia automática de py.pixelCoincide con el color(100,200, (140,125,134))
    FALSO
>>> guia automática de py.pixelCoincide con el color(100,200, (140,125,134), tolerancia=10) Verdadero
```

3.7 Pruebas

Las pruebas unitarias de PyAutoGUI no son exhaustivas en la actualidad. Las pruebas (en basicTests.py) cubren lo siguiente:

- en pantalla()
- tamaño()
- posición()
- moverA()
- moverRel()
- escribir a máquina()
- PAUSA

3.7.1 Plataformas probadas

- Pitón 3.4, 3.3, 3.2, 3.1, 2.7, 2.6, 2.5
- Ventanas
- Sistema operativo X
- Frambuesa Pi

(Si ha ejecutado las pruebas unitarias con éxito en otras plataformas, infórmenosal@inventwithpython.com.)

PyAutoGUI no es compatible con Python 2.4 o anterior.

Las funciones del teclado no funcionan en Ubuntu cuando se ejecutan en VirtualBox en Windows.

3.8 Hoja de ruta

PyAutoGUI está planeado como un reemplazo para otros scripts de automatización de GUI de Python, como PyUserInput, PyKeyboard, PyMouse, pykey, etc. Con el tiempo, sería genial ofrecer el mismo tipo de funciones queSikuliofertas.

Por ahora, el objetivo principal de PyAutoGUI es el control del mouse y el teclado multiplataforma y una API simple.

Funciones futuras planificadas (versiones específicas aún no planificadas):

- Una herramienta para determinar por qué no se puede encontrar una imagen en una captura de pantalla en particular. (Esta es una fuente habitual de preguntas para los usuarios).
- Compatibilidad total con Raspberry Pis.
- Función “Wave”, que se utiliza simplemente para ver dónde se encuentra el ratón agitando un poco el cursor. Una pequeña función auxiliar.
- función locateNear(), que es como las otras funciones de lectura de pantalla relacionadas con la localización, excepto que encuentra la primera instancia cerca de un punto xy en la pantalla.
- Encuentra una lista de todas las ventanas y sus títulos.
- Haga clic en las coordenadas relativas a una ventana, en lugar de a toda la pantalla.
- Facilitar el trabajo en sistemas con múltiples monitores.
- Tipo de función GetKeyState()
- Capacidad de configurar teclas de acceso rápido globales en todas las plataformas para que pueda haber un "interruptor de apagado" fácil para los programas de automatización de GUI.
- Llamadas pyautogui sin bloqueo opcionales.
- modo “estricto” para el teclado: pasar una tecla del teclado no válida provoca una excepción en lugar de omitirla silenciosamente.
- cambiar el nombre de keyboardMapping a KEYBOARD_MAPPING
- Capacidad de convertir png y otros archivos de imagen en una cadena que se puede copiar y pegar directamente en el código fuente, de modo que no sea necesario compartirlos por separado con los scripts de pyautogui de las personas.
- Pruebe para asegurarse de que pyautogui funciona en máquinas virtuales Windows/Mac/Linux.
- Una forma de comparar dos imágenes y resaltar las diferencias entre ellas (útil para señalar cuando cambia una interfaz de usuario, etc.)

Características de manejo de ventanas:

- pyautogui.getWindows() # devuelve un diccionario de títulos de ventanas asignados a identificaciones de ventanas
- pyautogui.getWindow(str_title_or_int_id) # devuelve un objeto “Win”
- ganar.mover(x, y)
- win.resize(ancho, alto)
- ganar.maximizar()
- ganar.minimizar()
- ganar.restaurar()
- ganar.cerrar()
- win.position() # devuelve (x, y) de la esquina superior izquierda
- win.moveRel(x=0, y=0) # se mueve en relación con x, y de la esquina superior izquierda de la ventana

• win.clickRel(x=0, y=0, clicks=1, interval=0.0, button='left') # clic relativo a x, y de la esquina superior izquierda de la ventana

• Se agregaron funciones de captura de pantalla para que pueda capturar ventanas específicas en lugar de la pantalla completa.

3.9 pyautogui

3.9.1 paquete pyautogui

Submódulos

módulo pyautogui.keynames

Contenido del módulo

Esta documentación aún está en progreso.

CAPÍTULO 4

Índices y tablas

- índice genético
- índice de modificación
- buscar