CONTROLLING THE KEYBOARD & MOUSE WITH GUI AUTOMATION

Frederik Meutermans & Yannick Buelens

Intro

Intro

- Graphical User Interface automation is:
  - verzenden van Virtual keystrokes & mouse clicks
  - alsof de gebruiker achter zijn computer zit.

- PyAutoGUI

Required Modules

- Dependencies:
  - On Windows, there are no other modules to install.
  - On OS X, run `sudo pip3 install pyobjc-framework-Quartz`, `sudo pip3 install pyobjc-core`, and then `sudo pip3 install pyobjc`.
  - On Linux, run `sudo pip3 install python3-xlib`, `sudo apt-get install scrot`, `sudo apt-get install python3-tk`, and `sudo apt-get install python3-dev`. (Scrot is a screenshot program that PyAutoGUI uses.)
- Pip3 install pyautogui

How to escape/prevent problems 101

- Python kan de muis bewegen en typen aan een hoge snelheid.
- Soms zo snel dat andere programma's niet kunnen volgen.
- Programma kan daardoor out of control geraken.

= Geen controle meer over muis/toetsenbord

- Gelukkig kunnen we dit stoppen/voorkomen.

How to escape/prevent problems 101

- Oplossing 1:
  - Uitloggen
    - Windows: ctrl-alt-del
    - Linux: ctrl-alt-del
    - OS X: ⌘-shift-option-Q
  - Computer afzetten
  - Nadeel:
    - Unsaved work ben je kwijt.

How to escape/prevent problems 101

- Oplossing 2:
  - Pauses
    - Na elke function call een pauze inlassen.
    - pyautogui.PAUSE = (aantal seconden)
  - Fail-safes
    - Als je de muis naar de linkerbovenhoek beweegt, stopt het programma.
    - pyautogui.FAILSAFE = True

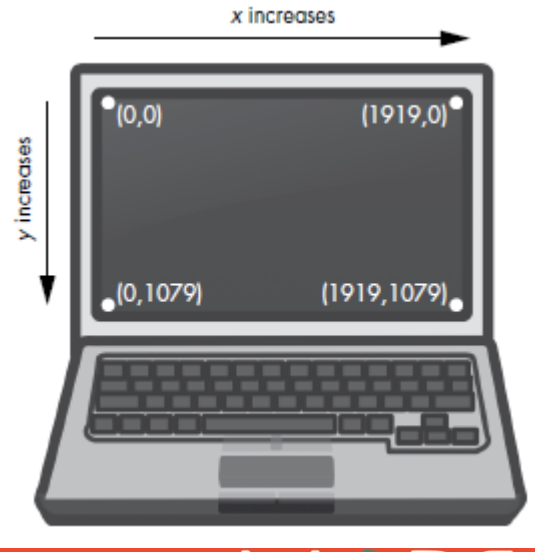
```
>>> import pyautogui
>>> pyautogui.PAUSE = 1
>>> pyautogui.FAILSAFE = True
```

Controlling Mouse Movement

How does it work?

- PyAutoGUI maakt gebruik van coördinaten.
  - (0,0) is linksboven.
  - (1919, 1079) is rechtsonder.
- pyautogui.size() returns een two-integer tuple met de resolutie van het scherm

```
>>> import pyautogui
>>> pyautogui.size()
(1920, 1080)
>>> width, height = pyautogui.size()
```



Moving the mouse

- pyautogui.moveTo(x, y, duration=s)
- X en y zijn de coördinaten naar waar de cursor moet bewegen
- Als 3de kun je de duration van de beweging meegeven (niet verplicht), default is dit 0 = instant.

```
>>> import pyautogui
>>> for i in range(10):
      pyautogui.moveTo(100, 100, duration=0.25)
      pyautogui.moveTo(200, 100, duration=0.25)
      pyautogui.moveTo(200, 200, duration=0.25)
      pyautogui.moveTo(100, 200, duration=0.25)
```

Moving the mouse

- pyautogui.moveRel(x, y, duration=s)
- Bij moveRel verplaatst de cursor zich relatief tegenover de huidige positie.

```
>>> import pyautogui
>>> for i in range(10):
        pyautogui.moveRel(100, 0, duration=0.25)
        pyautogui.moveRel(0, 100, duration=0.25)
```

Getting the Mouse position

- pyautogui.position()
- returns two-integer tuple (x en y positie)

```
>>> pyautogui.position()
(311, 622)
>>> pyautogui.position()
(377, 481)
>>> pyautogui.position()
(1536, 637)
```

Controlling Mouse Interaction

Clicking the Mouse

- pyautogui.click()
  - Standaard wordt er met de linkermuisknop op de huidige positie geklikt.
- pyautogui.mouseDown() en pyautogui.mouseUp()
  - Met deze twee functies kan je ook een click simuleren.

```
>>> import pyautogui
>>> pyautogui.click(10, 5)
```

Clicking the Mouse

- pyautogui.doubleClick()
  - Dubbel klikt op het scherm met de linkermuisknop
- Bij alle bovenstaande click commando's kan je als argument een x en y positie meegeven en met welke knop er geklikt moet worden. (left, right, middle)

```
pyautogui.click(100, 150, button='left')
```

- pyautogui.rightClick()
- pyautogui.middleClick()

Dragging the Mouse

- Dragging = de muis bewegen terwijl er een muisknop ingeduwd is.
- dragTo() en dragRel()
  - x en y positie als argument
  - Duration kan meegegeven worden. (duration = seconden)

Scrolling the Mouse

- pyautogui.scroll()
  - Argument = hoeveel units je wilt scrollen (integer)
  - positieve integer = naar boven
  - negatieve integer = naar onder
- Je kan het programma laten ‘slapen’ om van venster te veranderen

```
>>> pyautogui.scroll(200)
```

```
>>> import time, pyautogui
>>> time.sleep(5); pyautogui.scroll(100)
```

Working with the Screen

Intro Working with the Screen

- GUI automation moet niet blindelings klikken en typen.
- Er zijn functie's om het scherm te analyseren vooraleer er wordt geklikt of getypt.
- Linux gebruikers moeten *scrot* installeren. (*sudo apt-get install scrot*)
- Je kan ook werken met pillow uit het vorige hoofdstuk.

Getting a Screenshot

- Om het scherm te analyseren, moeten we eerst er een screenshot van nemen.
- pyautogui.screenshot()
  - Returns een image object
  - Hier kunnen we methodes op uitvoeren, bv. getpixel() (returns RGB waarde van positie).

```
>>> import pyautogui
>>> im = pyautogui.screenshot()
```

```
>>> im.getpixel((0, 0))
(176, 176, 175)
>>> im.getpixel((50, 200))
(130, 135, 144)
```

Analyzing the Screenshot

- pyautogui.pixelMatchesColor(x, y (R, G, B))
  - Kijkt na of de kleur op het scherm gelijk is aan de meegegeven RGB waarde.
  - Returns True of False

```
>>> import pyautogui
>>> im = pyautogui.screenshot()
>>> im.getpixel((50, 200))
(130, 135, 144)
>>> pyautogui.pixelMatchesColor(50, 200, (130, 135, 144))
True
>>> pyautogui.pixelMatchesColor(50, 200, (255, 135, 144))
False
```

Image Recognition

Searching the button

- Wat als je niet op voorhand weet waar je moet klikken?

- locateOnScreen()
  - zoekt op scherm naar een gegeven afbeelding
  - geeft de coördinaten en grootte terug in tuple van de eerst gevonden match

```
>>> import pyautogui
>>> pyautogui.locateOnScreen('submit.png')
(643, 745, 70, 29)
```

x y b h

- locateAllOnScreen()
  - zoekt ook naar match maar geeft alle gevonden matchen in een lijst van tuples

```
>>> list(pyautogui.locateAllOnScreen('submit.png'))
[(643, 745, 70, 29), (1007, 801, 70, 29)]
```

- nu op deze knop klikken m.b.v center
  - center neemt de tuple en berekent het midden van deze waarden: x + de helft van de breedte, y + de helft van de hoogte

```
>>> pyautogui.locateOnScreen('submit.png')
(643, 745, 70, 29)
>>> pyautogui.center((643, 745, 70, 29))
(678, 759)
>>> pyautogui.click((678, 759))
```

Controlling the keyboard

Virtual keypresses

PyAutoGUI heeft ook functies om toetsaanslagen te automatiseren:

- pyautogui.typewrite(<string>)
- pyautogui.press(<character>)
- pyautogui.keyUp(<character>)
- pyautogui.keyDown(<character>)
- pyautogui.hotkey(<character>,<character>)

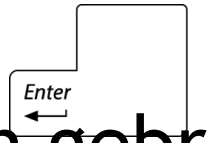
Sending a string

- pyautogui.typewrite(<string>, 0.25)
  - voor een hele zin
  - <string>: de tekst die moet worden getypt
  - 0.25: optioneel, tijdsduur voor elke letter of karakter uit de tekst.
  - simuleert ook SHIFT toets bij hoofdletters of tekens

```
>>> pyautogui.click(100, 100); pyautogui.typewrite('Hello world!')
```

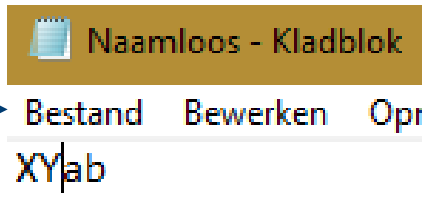
Key names

Niet elke toets is aan te duiden met 1 karakter, bv enter



daarom gebruik maken van “key names” zoals ‘esc’,‘enter’,‘left’,‘right’

```
>>> pyautogui.typewrite(['a', 'b', 'left', 'left', 'X', 'Y'])
```



THOMAS MORE

Key names

Lijst van alle keys die je kan gebruiken:

Keyboard key string	Meaning
'a', 'b', 'c', 'A', 'B', 'C', '1', '2', '3', '!', '@', '#', and so on	The keys for single characters
'enter' (or 'return' or '\n')	The ENTER key
'esc'	The ESC key
'shiftleft', 'shiftright'	The left and right SHIFT keys
'altleft', 'altright'	The left and right ALT keys
'ctrlleft', 'ctrlright'	The left and right CTRL keys
'tab' (or '\t')	The TAB key
'backspace', 'delete'	The BACKSPACE and DELETE keys
'pageup', 'pagedown'	The PAGE UP and PAGE DOWN keys
'home', 'end'	The HOME and END keys
'up', 'down', 'left', 'right'	The up, down, left, and right arrow keys
'f1', 'f2', 'f3', and so on	The F1 to F12 keys
'volumemute', 'volumedown', 'volumeup'	The mute, volume down, and volume up keys (some keyboards do not have these keys, but your operating system will still be able to understand these simulated keypresses)
'pause'	The PAUSE key
'capslock', 'numlock', 'scrolllock'	The CAPS LOCK, NUM LOCK, and SCROLL LOCK keys
'insert'	The INS or INSERT key
'printscreen'	The PRTSC or PRINT SCREEN key
'winleft', 'winright'	The left and right WIN keys (on Windows)
'command'	The Command (⌘) key (on OS X)
'option'	The OPTION key (on OS X)

Pressing and releasing the keyboard

- Net zoals mouseUp() en mouseDown() ook keyUp() en keyDown()

```
>>> pyautogui.keyDown('shift'); pyautogui.press('4'); pyautogui.keyUp('shift')
```

- pyautogui.press() combineert keyDown en keyUp in 1 functie
- functies aanvaarden enkel characters of key names

Hotkey combinations

- Een hotkey of sneltoets is een bepaalde functie in een programma (CTRL-C,CTRL-V, CTRL-X, CTRL-S, ...)
- met keyDown() en keyUp():

```
pyautogui.keyDown('ctrl')
pyautogui.keyDown('c')
pyautogui.keyUp('c')
pyautogui.keyUp('ctrl')
```

- met hotkey():

=

```
pyautogui.hotkey('ctrl', 'c')
```

- nuttig voor lange hotkeys (CTRL-ALT-SHIFT-S)

Quiz

1. Hoe trigger je de fail-safe om het programma te stoppen?
2. Welke functie geeft de huidige schermresolutie?
3. Welke functie toont de huidige positie van de muis?
4. Wat is het verschil tussen pyautogui.moveTo() en pyautogui.moveRel()?
5. Welke functies kan je gebruiken om de muis te slepen?
6. Welke functie typt de volgende karakters: "Hello world!"?
7. Hoe type je speciale toetsen zoals ‘pijl naar links’ ?
9. Welke code zorgt voor een pauze na elke PyAutoGUI functie?

Questions & Practice

- Vragen?

- Oefeningen staan op Trac