



# Python 键盘/鼠标监听及控制

dawsonenjoy 关注 0.685 2018.09.24 09:52:43 字数 424 阅读 57,111

## 键盘监听

需要用到 keyboard 模块 ( pip install keyboard )

### 按键字符

```
1 #字符
2 '1'
3 'a'
4 ...
5 #控制
6 'ctrl'
7 'alt'
8 'shift'
9 'enter'
10 'esc'
11 'f1'
12 ...
13 #方向键
14 'up'
15 'down'
16 'left'
17 'right'
18 #组合按键
19 'ctrl'+ 'alt'+ 'a'
20 ...
```

### 常用方法

#### wait()

监听按键，如果没设置按键，将会一直监听这句之前的按键；如果设置了按键，那么在按下该按键后就会停止监听，并执行后面的语句，举例：

```
1 import keyboard
2
3 print(0)
4 keyboard.wait('a')
5 #在按下a之前后面的语句都不会执行，下面同理
6 print(1)
7 keyboard.wait('b')
8 print(2)
9 keyboard.wait('c')
10 print(3)
11 keyboard.wait()
12
13 结果：
14 0
15 1
16 2
17 3
18 #继续监听
19 #只有按照顺序按下abc（中间过程随便按不干扰）才能输出0123，但因为最后一个没设置按键，所以会一直监听下去
```

#### add\_hotkey()

设置热键，一般和 wait() 组合使用，在wait监听下，当按下热键时会执行对应函数，举例：

```
1 import keyboard #监听键盘
2
3 def test_a():
4     print('aaa')
5
6 def test(x):
7     print(x)
8
9 if __name__ == '__main__':
10     keyboard.add_hotkey('f1', test_a)
11     #按f1输出aaa
12     keyboard.add_hotkey('ctrl+alt', test, args=('b',))
13     #按ctrl+alt输出b
14     keyboard.wait()
15     #wait里也可以设置按键，说明当按到该键时结束
```

#### record()

记录键盘事件，如果加上until参数，可以设置当按下某按键时结束监听，和wait方法有点像，举例：

```
1 import keyboard
2
3 keyboard.add_hotkey('ctrl', print, args=('aaa',))
4 keyboard.add_hotkey('alt', print, args=('bbb',))
5
6 recorded = keyboard.record(until='esc')
7 #当按下esc时结束按键监听，并输出所有按键事件
8 print(recorded)
9 结果为：
10 aaa
11 aaa
12 bbb
13 [KeyboardEvent(ctrl down), KeyboardEvent(ctrl up), KeyboardEvent(ctrl down),
14 KeyboardEvent(ctrl up), KeyboardEvent(alt down), KeyboardEvent(alt up),
15 KeyboardEvent(esc down)]
```

#### hook()

绑定所有按键事件，当只要有按键按下/松开时就会触发的回调函数，举例：

```
1 import keyboard
2
3 def abc(x):
4     print(x)
5     print("111")
6
7 keyboard.hook(abc)
8 #按下任何按键时，都会调用abc，其中一定会传一个值，就是键盘事件
9 keyboard.wait()
10
11 结果：
12 KeyboardEvent(w down)
13 111
14 KeyboardEvent(w up)
15 111
16 KeyboardEvent(space down)
17 111
18 KeyboardEvent(space up)
19 111
20 KeyboardEvent(tab down)
21 111
22 KeyboardEvent(tab up)
23 111
24 KeyboardEvent(ctrl down)
25 111
26 ...
```

#### KeyboardEvent()

一个按键事件，里面有3个常用参数： event\_type 、 scan\_code 、 name ，分别代表按键类型（ down / up ）、按键号（每个键都有对应的）和按键名，举例：

```
1 import keyboard
2
3 def abc(x):
4     a = keyboard.KeyboardEvent('down', 28, 'enter')
5     #按键事件a为按下enter键，第二个参数如果不知道每个按键的值就随便写，
6     #如果想知道按键的值可以用hook绑定所有事件后，输出x.scan_code即可
7     if x.event_type == 'down' and x.name == a.name:
8         print("你按下了enter键")
9     #当监听的事件为enter键，且是按下时
10
11 keyboard.hook(abc)
12 # keyboard.hook_key('enter', bcd)
13 # recorded = keyboard.record(until='esc')
14 keyboard.wait()
15
16 结果：
17 你按下了enter键
18 你按下了enter键
```

### 更多参考

<https://pypi.org/project/keyboard/>

## 鼠标、键盘监听控制

这里介绍两个模块，一个是 pyautogui 模块（ pip install pyautogui ），另一个是 pynput（ pip install pynput ）

### pyautogui基本使用

#### 常用方法

##### position()

获取鼠标位置，举例：

```
1 import pyautogui as pag #监听鼠标
2
3 x1, y1 = pag.position()
4 print(x1, y1)
5 #输出鼠标当前位置
```

#### 实例-结合鼠标键盘截图

(按下两次alt+ctrl来确定图片左上角和右上角，然后截图保存)

```
1 import keyboard #监听键盘
2 import pyautogui as pag #监听鼠标
3 from PIL import ImageGrab #截图、读取图片、保存图片
4
5 if keyboard.wait(hotkey='ctrl+alt') == None:
6     x1, y1 = pag.position()
7     if keyboard.wait(hotkey='ctrl+alt') == None:
8         x2, y2 = pag.position()
9         image = ImageGrab.grab((x1, y1, x2, y2))
10         image.save("screen.png")
```

##### click()

鼠标点击控制，举例：

```
1 import pyautogui
2
3 pyautogui.click(button='right')
4 #点击鼠标右键
5 pyautogui.click(100, 100)
6 #要在指定位置点击左键
```

### 更多参考

pyautogui自动化控制鼠标键盘：

[https://blog.csdn.net/weixin\\_43430036/article/details/84650938](https://blog.csdn.net/weixin_43430036/article/details/84650938)

pyautogui控制键盘输入：[https://blog.csdn.net/qq\\_40379759/article/details/80427235](https://blog.csdn.net/qq_40379759/article/details/80427235)

### pynput基本使用

参考：<https://blog.csdn.net/longgb123/article/details/79090559>

14人点赞 > Python ...

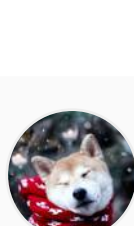
更多精彩内容，就在简书APP



"有人可以帮我试试赞赏的功能坏了吗2333"

赞赏支持

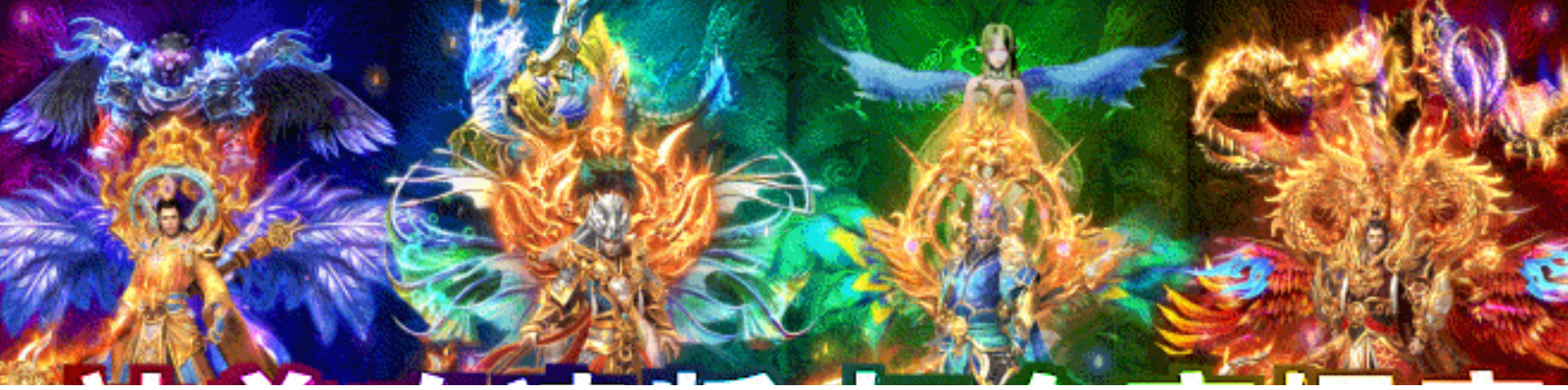
还没有人赞赏，支持一下



dawsonenjoy 总资产71 共写了32.1W字 获得276个赞 共156个粉丝

关注

## 开局四神兽，召唤妖化兽，一秒999，全服横着走！



妖化神兽版·效卓

广告

被以下专题收入，发现更多相似内容

python ...