open a string in notepad at runtime in python

12 octobre 2010

<http://stackoverflow.com/questions/3914409/open-a-string-in-notepad-at-runtime-in-python>

There is an example [here](http://code.activestate.com/lists/python-win32/3009/).

#### Script to try to write something down in notepad

import win32api

import win32gui

import win32con

import time

import subprocess

#start notepad.exe asynchronously

subprocess.Popen('Notepad.exe')

# get the window handle of the blank, minimized notepad window

hwnd = win32gui.FindWindowEx(0, 0, 0, "Untitled - Notepad")

# print it just for kicks

print hwnd

win32gui.ShowWindow(hwnd, win32con.SW\_SHOWNORMAL)

#this restores the proper window, so we know we have correct handle

#just to give it a little pause

time.sleep(2)

print "trying to post message"

#try to send it a return key

win32api.SendMessage(hwnd, win32con.WM\_KEYDOWN, win32con.VK\_RETURN, 0)

win32api.SendMessage(hwnd, win32con.WM\_KEYUP, win32con.VK\_RETURN, 0)

#the above generates absolutely no effect on the notepad window.

#same effect no matter what vk code i use (e.g. 65 for A, VK\_SPACE for space, etc)

#### end of script

May I suggest you to use AutoIt3 facilities ([http://www.autoitscript.com/autoit3/docs/tutorials/notepad/notepad.htm "AutoIt Notepad Tutorial"](http://www.autoitscript.com/autoit3/docs/tutorials/notepad/notepad.htm))

AutoIt3 is a Windows scripting language to control quite anything in Windows. It provide a COM API so you can make integrate it in your Python script

from win32com.client import Dispatch

AutoIt = Dispatch("AutoItX3.Control")

AutoIt.Run('Notepad.exe')

AutoIt.WinWaitActive("Untitled - Notepad")

AutoIt.Send("This is some text.")

It may be also possible to use AutoHotKey (the fully GPL version of AutoIt)

|  |  |
| --- | --- |
|  | This code will send s into Notepad window from Python script.  class cls\_KeyBdInput(ct.Structure):  \_fields\_ = [  ("wVk", ct.c\_ushort),  ("wScan", ct.c\_ushort),  ("dwFlags", ct.c\_ulong),  ("time", ct.c\_ulong),  ("dwExtraInfo", ct.POINTER(ct.c\_ulong) )  ]  class cls\_HardwareInput(ct.Structure):  \_fields\_ = [  ("uMsg", ct.c\_ulong),  ("wParamL", ct.c\_short),  ("wParamH", ct.c\_ushort)  ]  class cls\_MouseInput(ct.Structure):  \_fields\_ = [  ("dx", ct.c\_long),  ("dy", ct.c\_long),  ("mouseData", ct.c\_ulong),  ("dwFlags", ct.c\_ulong),  ("time", ct.c\_ulong),  ("dwExtraInfo", ct.POINTER(ct.c\_ulong) )  ]  class cls\_Input\_I(ct.Union):  \_fields\_ = [  ("ki", cls\_KeyBdInput),  ("mi", cls\_MouseInput),  ("hi", cls\_HardwareInput)  ]  class cls\_Input(ct.Structure):  \_fields\_ = [  ("type", ct.c\_ulong),  ("ii", cls\_Input\_I)  ]  def make\_input\_objects( l\_keys ):  p\_ExtraInfo\_0 = ct.pointer(ct.c\_ulong(0))  l\_inputs = [ ]  for n\_key, n\_updown in l\_keys:  ki = cls\_KeyBdInput( n\_key, 0, n\_updown, 0, p\_ExtraInfo\_0 )  ii = cls\_Input\_I()  ii.ki = ki  l\_inputs.append( ii )  n\_inputs = len(l\_inputs)  l\_inputs\_2=[]  for ndx in range( 0, n\_inputs ):  s2 = "(1, l\_inputs[%s])" % ndx  l\_inputs\_2.append(s2)  s\_inputs = ', '.join(l\_inputs\_2)  cls\_input\_array = cls\_Input \* n\_inputs  o\_input\_array = eval( "cls\_input\_array( %s )" % s\_inputs )  p\_input\_array = ct.pointer( o\_input\_array )  n\_size\_0 = ct.sizeof( o\_input\_array[0] )  # these are the args for user32.SendInput()  return ( n\_inputs, p\_input\_array, n\_size\_0 )  def send\_s( window1 ):  t\_s = ( ( 0x53, 0 ), )  l\_keys = [ ]  l\_keys.extend( t\_s )  t\_inputs = make\_input\_objects( l\_s )  win32gui.ShowWindow(window1, win32con.SW\_SHOWNORMAL)  win32gui.SetForegroundWindow(window1)  rv = ct.windll.user32.SendInput( \*t\_inputs )  def find\_window( s\_app\_name ):  try:  window1 = FindWindow( None, s\_app\_name,)  return window1  except ui\_err:  pass  except:  raise  try:  window1 = FindWindow( s\_app\_name, None, )  return window1  except ui\_err:  return None  except:  raise  def search\_title(srch,ttls):  out=None  for i in range(len(ttls)):  #print i, ttls[i][1]  if srch in ttls[i][1]:  out= ttls[i][1]  return out  def get\_window\_titles():  titles = []  def foreach\_window(hwnd, lParam):  if IsWindowVisible(hwnd):  length = GetWindowTextLength(hwnd)  buff = ctypes.create\_unicode\_buffer(length + 1)  GetWindowText(hwnd, buff, length + 1)  ttl=buff.value  titles.append((hwnd, ttl))  return True  EnumWindows(EnumWindowsProc(foreach\_window), 0)  return titles  ttls=get\_window\_titles()  title=search\_title('Notepad',ttls)  window1 = find\_window( title )  send\_s( window1) |