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.

```
#### 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)
```