

Installation & Usage Guide

GUIRobot

22/03/11

Martin Moore
martin.moore@mycit.ie

Table of Contents

1 Preface.....	2
2 Requirements.....	2
3 Installation.....	2
3.1 Easy Installation.....	2
3.2 Easy Update (Reinstallation).....	3
3.3 Git Installation.....	3
3.4 Git Update.....	3
4 Usage.....	4
4.1 Testing If GUIRobot Works.....	4
4.2 GUIRobot - Virtual Mouse.....	4
4.3 GUIRobot – Virtual Keyboard.....	5

1 Preface

This installation guide is intended for use with GUIRobot commit [7f90de013ede13ab9c90](https://github.com/Ph4g3/GUIRobot/commit/7f90de013ede13ab9c90). You can always find the latest version of the source on github: <https://github.com/Ph4g3/GUIRobot>. As this application is not in any way mature, a knowledge of the Python programming language will help.

2 Requirements

- Python v2.7.1

“Python is a remarkably powerful dynamic programming language that is used in a wide variety of application domains. Python is often compared to Tcl, Perl, Ruby, Scheme or Java.”

You can obtain a Python installer for Windows at the python homepage:

<http://www.python.org/download/>

For the purpose of this guide, I will assume you have installed Python in the default directory: C:\Python27\

- Windows OS

The GUIRobot is intended for use with Windows XP, Vista or 7. Should work for both 32- and 64-bit releases. It's possible that it might work with Windows 2000, but this claim is untested.

3 Installation

There are two methods of acquiring the GUIRobot application. If it is your intention to keep an up-to-date install of the application, you will want to look at “Git Installation”. Experience with git or a similar version control system is a plus.

3.1 Easy Installation

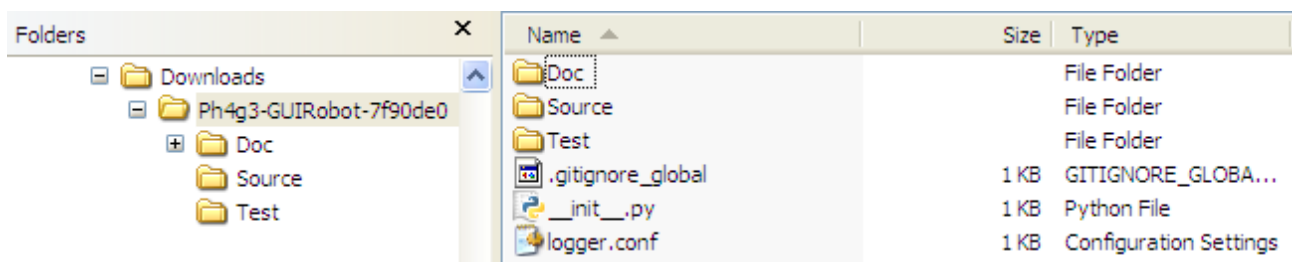
1. Go to the repository homepage: <https://github.com/Ph4g3/GUIRobot>

2. Click on the download icon:

 Downloads

3. Choose “Download .zip”. You can choose the .tar.gz file if you like, but remember that this is a Windows application.

4. Extract the files from the archive. Depending on the program used to extract the files, you may have an extra folder level. You want your folder structure to look like the following:



5. Move the Ph4g3-GUIRobot-<commit #> folder into your C:\Python27\ folder. Rename Ph4g3-GUIRobot-<commit #> to something a little more friendly like “GUIRobot”.

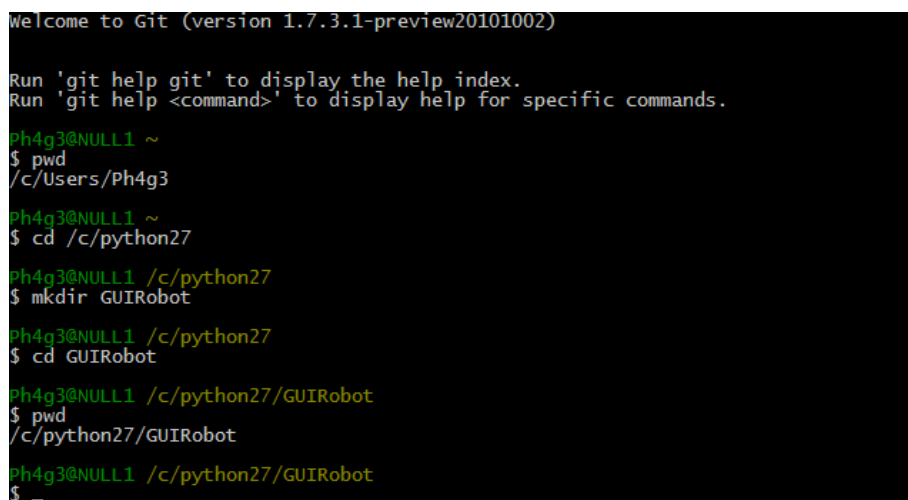
3.2 Easy Update (Reinstallation)

1. Go to C:\Python27\.
2. Delete GUIRobot folder.
3. Begin installation in steps in section 3.1.

Easy, but not exactly efficient.

3.3 Git Installation

1. Obtain a copy of Git for Windows - msysgit: <http://code.google.com/p/msysgit/>.
2. Install msysgit with the default options.
3. Start Git Bash. “Bash” does indeed refer to the Bourne-again shell.
4. Navigate to the desired directory using Unix-style commands.



```
Welcome to Git (version 1.7.3.1-preview20101002)

Run 'git help git' to display the help index.
Run 'git help <command>' to display help for specific commands.

Ph4g3@NULL1 ~
$ pwd
/c/Users/Ph4g3

Ph4g3@NULL1 ~
$ cd /c/python27

Ph4g3@NULL1 /c/python27
$ mkdir GUIRobot

Ph4g3@NULL1 /c/python27
$ cd GUIRobot

Ph4g3@NULL1 /c/python27/GUIRobot
$ pwd
/c/python27/GUIRobot

Ph4g3@NULL1 /c/python27/GUIRobot
$
```

5. Initialize and clone the repository by issuing the following commands:

```
$ git init
$ git clone git://github.com/Ph4g3/GUIRobot.git
```

Git Bash will grab all the required files from the github repository. You now have an installation of GUIRobot in C:\Python27\GUIRobot.

3.4 Git Update

1. Start Git Bash.
2. Navigate to the repository directory:
`$ cd /c/python27/guirobot`
3. Issue the following command to update the repository:
`$ git pull git://github.com/Ph4g3/GUIRobot.git`

The “pull” command is simply the “fetch” and “merge” commands. It will check which files have changed and update them accordingly. It will also get the new files. This method is much more efficient than deleting the whole project, downloading it all again, unzipping it, renaming it and moving it back into the ...\\Python27\\ directory.

4 Usage

For the usage tutorial, I will be referring to IDLE, which is the Python interpreter for Windows. If you're familiar with Python, there's nothing stopping you from using the CLI or an IDE. IDLE is built into the base install, so you already have it:

Start > All Programs > Python 2.7 > IDLE (Python GUI)

4.1 Testing If GUIRobot Works

You'll be presented with the the Python interpreter. It will contain the following message:

```
Python 2.7.1 (r271:86832, Nov 27 2010, 18:30:46) [MSC v.1500 32 bit (Intel)] on win32
```

```
Type "copyright", "credits" or "license()" for more information.
```

```
>>>
```

First we want to test the application to see if it runs correctly. From the module “Test”, located in the package “GUIRobot”, we want to load the submodule “test_Virtual_HID”. To do that, issue the following command:

```
>>> from GUIRobot.Test import test_Virtual_HID
```

Now run the main function in the test submodule:

```
>>> test_Virtual_HID.main()
```

```
test_click (GUIRobot.Test.test_Virtual_HID.test_VMouse) ... ok
test_getCoord (GUIRobot.Test.test_Virtual_HID.test_VMouse) ... ok
test_setCoords (GUIRobot.Test.test_Virtual_HID.test_VMouse) ... ok
```

```
-----
Ran 3 tests in 1.845s
```

```
OK
```

If your mouse just jumped around a bit and did some clicking, then the application is running correctly. If not, feel free to send me an email.

4.2 GUIRobot - Virtual Mouse

Restart the interpreter to clean everything up. Do this by pressing <Ctrl + F6>, or going to Shell > Restart Shell. Lets import code for GUIRobot.

```
>>> from GUIRobot.Test import Virtual_HID
```

Now we have our virtual input devices, namely the VMouse and VKeyboard. To get more information about them, we can use the built in command “help()”:

```
>>> help(Virtual_HID.VMouse)
```

This will show us a bunch of methods we can use to control the mouse. First, we need to create an instance of the Vmouse class in order to use it. I'm going to call the variable “mouse”, but it can be anything:

```
>>> mouse = Virtual_HID.VMouse()
```

Lets grab the coordinates and store them in a variable called “pt”.

```
>>> pt = mouse.getCoords()
```

The “pt” variable now contains a POINT object, which has an 'x' and 'y' field. To see what these are, we can do the following:

```
>>> print(pt.x, pt.y)
(519, 829)
```

We can set the mouse back to these coordinates by doing the following:

```
>>> mouse.setCoords(pt.x, pt.y)
```

We can also use a similar method named `setPointCoords` and just pass the `POINT` object into it:

```
>>> mouse.setPointCoords(pt)
```

The rest of the methods in `VMouse` can be used in the same way. To see them, use the help function.

It is probably worth mentioning that you should be careful when using methods that cause an action to occur for an unspecified time. For example:

```
>>> mouse.holdLeft()
```

This will do exactly what you expect – it will keep the left mouse button held down. If you're still at the interpreter, you can negate it:

```
>>> mouse.releaseLeft()
```

If you just wanted a simple left mouse click, do the following:

```
>>> mouse.leftClick()
```

To hold the left mouse button for 1.5 seconds, you can do the following:

```
>>> import time
>>> mouse.holdLeft(); time.sleep(1.5); mouse.releaseLeft()
```

You can see this in action if you move your physical mouse around. The program is holding left click for you, while you move the mouse around.

4.3 GUIRobot – Virtual Keyboard