pyscreenshot Documentation

Release 0.3.3

ponty

August 24, 2014

CONTENTS

pyscreenshot

Date August 24, 2014

PDF pyscreenshot.pdf

Contents:

The `pyscreenshot` module can be used to copy the contents of the screen to a PIL or Pillow image memory. Replacement for the ImageGrab Module, which works on Windows only. For handling image memory (e.g. saving to file, converting,..) please read PIL or Pillow documentation.

Links:

- home: https://github.com/ponty/pyscreenshot
- documentation: http://ponty.github.com/pyscreenshot

Goal: Pyscreenshot tries to allow to take screenshots without installing 3rd party libraries. It is cross-platform but useful for Linux based distributions. It is only a pure Python wrapper, a thin layer over existing back-ends. Its strategy should work on most Linux distributions: a lot of back-ends are wrapped, if at least one exists then it works, if not then one back-end should be installed. Performance and interactivity are not important for this library.

Features:

- Cross-platform wrapper
- Capturing the whole desktop
- Capturing an area
- saving to PIL or Pillow image memory
- some back-ends are based on this discussion: http://stackoverflow.com/questions/69645/take-a-screenshot-via-a-python-script-linux
- pure Python library
- supported python versions: 2.6, 2.7
- **Plugin based, it has wrappers for various back-ends:**
  - scrot
  - ImageMagick
  - PyGTK
  - PIL or Pillow (only on windows)
  - PyQt4
  - wxPython

Known problems:

- different back-ends generate slightly different images from the same desktop, this should be investigated
- ImageMagick creates blackbox on some systems
- PyGTK back-end does not check $DISPLAY -> not working with Xvfb
- slow: 0.2s - 0.7s

Similar projects:

- http://sourceforge.net/projects/gtkshots/
- http://pypi.python.org/pypi/autopy

CHAPTER

ONE

USAGE

Example:

```
import pyscreenshot as ImageGrab

# fullscreen
im=ImageGrab.grab()
im.show()

# part of the screen
im=ImageGrab.grab(bbox=(10,10,510,510)) # X1,Y1,X2,Y2
im.show()
```

CHAPTER

TWO

INSTALLATION

2.1 General

- install pip
- install PIL or Pillow
- install at least one back-end
- install the program:

```
# as root
pip install pyscreenshot
```

2.2 Ubuntu

```
sudo apt-get install python-pip
sudo pip install pyscreenshot
sudo apt-get install python-imaging
# optional back-ends
sudo apt-get install scrot
sudo apt-get install imagemagick
sudo apt-get install python-gtk2
sudo apt-get install python-qt4
# optional for examples
sudo pip install entrypoint2
```

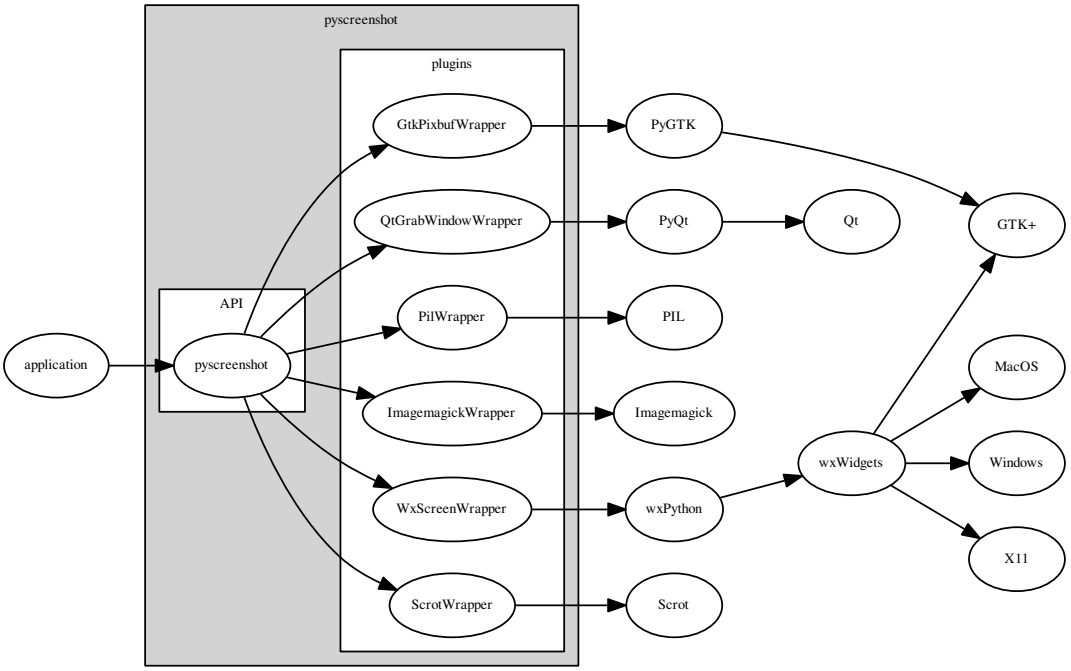
2.3 Uninstall

```
# as root
pip uninstall pyscreenshot
```

CHAPTER

THREE

HIERARCHY



CHAPTER

FOUR

EXAMPLES

pyscreenshot/examples/show.py:

```
from entrypoint2 import entrypoint
from pyscreenshot import grab


@entrypoint
def show(backend='auto'):
    if backend == 'auto':
        backend = None
    im = grab(bbox=(100, 200, 300, 400), backend=backend)
    im.show()
```

to start:

```
python -m pyscreenshot.examples.show
```

pyscreenshot/examples/showall.py:

```
from entrypoint2 import entrypoint
from pyscreenshot import backends
import time

import pyscreenshot as ImageGrab


@entrypoint
def show():
    im = []

    for x in backends():
        try:
            print 'grabbing by ' + x
            im.append(ImageGrab.grab(bbox=(500, 400, 800, 600), backend=x))
        except pyscreenshot.FailedBackendError as e:
            print e
    print im
    for x in im:
        x.show()
        time.sleep(1)
```

to start:

```
python -m pyscreenshot.examples.showall
```

CHAPTER

FIVE

BACK-END PERFORMANCE

```
$ python -m pyscreenshot.check.speedtest

n=10 , to_file: True , bounding box: None
-----------------------------------------------------
pil                  Forced backend not found, or cannot be loaded:pil
scrot                3    sec (  298 ms per call)
wx                   5.1  sec (  506 ms per call)
pygtk                4.1  sec (  410 ms per call)
pyqt                 2.6  sec (  260 ms per call)
imagemagick          17   sec ( 1663 ms per call)
mac_screencapture    Forced backend not found, or cannot be loaded:mac_screencapture
mac_quartz           Forced backend not found, or cannot be loaded:mac_quartz

n=10 , to_file: False , bounding box: None
-----------------------------------------------------
pil                  Forced backend not found, or cannot be loaded:pil
scrot                3    sec (  299 ms per call)
wx                   1.3  sec (  133 ms per call)
pygtk                4.1  sec (  409 ms per call)
pyqt                 2.4  sec (  243 ms per call)
imagemagick          17   sec ( 1663 ms per call)
mac_screencapture    Forced backend not found, or cannot be loaded:mac_screencapture
mac_quartz           Forced backend not found, or cannot be loaded:mac_quartz

n=10 , to_file: False , bounding box: (10, 10, 20, 20)
-----------------------------------------------------
pil                  Forced backend not found, or cannot be loaded:pil
scrot                3.7  sec (  368 ms per call)
wx                   1.4  sec (  139 ms per call)
pygtk                8.6  sec (  863 ms per call)
pyqt                 4.3  sec (  427 ms per call)
imagemagick          11   sec ( 1128 ms per call)
mac_screencapture    Forced backend not found, or cannot be loaded:mac_screencapture
mac_quartz           Forced backend not found, or cannot be loaded:mac_quartz
```

Test system versions:

```
$ python -m pyscreenshot.check.versions
pyscreenshot          0.3.3
pil                   missing
scrot                 0.8
wx                    2.8.12.1
pygtk                 2.28.6
pyqt                  not implemented
imagemagick           6.7.7
mac_screencapture     missing
mac_quartz            missing
```

CHAPTER

SIX

COMMAND LINE HELP

```
$ python -m pyscreenshot.examples.show --help
usage: show.py [-h] [-b BACKEND] [--debug]

optional arguments:
  -h, --help            show this help message and exit
  -b BACKEND, --backend BACKEND
  --debug               set logging level to DEBUG

$ python -m pyscreenshot.examples.showall --help
usage: showall.py [-h] [--debug]

optional arguments:
  -h, --help  show this help message and exit
  --debug     set logging level to DEBUG

$ python -m pyscreenshot.check.speedtest --help
usage: speedtest.py [-h] [--debug]

optional arguments:
  -h, --help  show this help message and exit
  --debug     set logging level to DEBUG

$ python -m pyscreenshot.check.versions --help
usage: versions.py [-h] [--debug]

optional arguments:
  -h, --help  show this help message and exit
  --debug     set logging level to DEBUG
```

CHAPTER

SEVEN

API

pyscreenshot.**backends**()
Back-end names as a list

pyscreenshot.**grab**(*bbox=None*, *childprocess=False*, *backend=None*)
Copy the contents of the screen to PIL image memory.

Parameters

- **bbox** – optional bounding box (x1,y1,x2,y2)
- **childprocess** – pyscreenshot can cause an error, if it is used on more different virtual displays and back-end is not in different process. Some back-ends are always different processes: scrot, imagemagick
- **backend** – back-end can be forced if set (examples:scrot, wx,..), otherwise back-end is automatic

pyscreenshot.**grab_to_file**(*filename*, *childprocess=False*, *backend=None*)
Copy the contents of the screen to a file.

Parameters

- **filename** – file for saving
- **childprocess** – see `grab()`
- **backend** – see `grab()`

CHAPTER

EIGHT

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

p

B

G

P