PyVirtualDisplay Documentation

Release 0.0.7

ponty

August 10, 2011

CONTENTS

pyvirtualdisplay

Date August 10, 2011

PDF pyvirtualdisplay.pdf

Contents:

pyvirtualdisplay is a python wrapper for Xvfb and Xephyr

Links:

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

Features:

- python wrapper
- backends: Xvfb, Xephyr, Xvnc

Known problems:

- Python 3 is not supported
- only a few backend options are supported

Possible applications:

- GUI testing
- automatic GUI screenshot

CHAPTER

ONE

BASIC USAGES

Start Xephyr:

```
from pyvirtualdisplay import Display
xephyr=Display(visible=1, size=(320, 240)).start()
```

Create screenshot of xmessage with Xvfb:

```
from easyprocess import EasyProcess
from pyvirtualdisplay.smartdisplay import SmartDisplay
with SmartDisplay(visible=0, bgcolor='black') as disp:
    with EasyProcess('xmessage hello'):
        img = disp.waitgrab()
img.show()
```

CHAPTER

TWO

INSTALLATION

2.1 General

- install Xvfb or Xephyr or Xvnc.
- install setuptools
- optional: pyscreenshot and PIL should be installed for `smartdisplay` submodule
- install the program:

  ```
  # as root
  easy_install pyvirtualdisplay
  ```

2.2 Ubuntu

```
sudo apt-get install python-setuptools
sudo apt-get install xvfb
sudo apt-get install xserver-xephyr
sudo apt-get install tightvncserver
sudo easy_install pyvirtualdisplay
# optional
sudo apt-get install python-imaging
sudo apt-get install scrot
sudo easy_install pyscreenshot
```

2.3 Uninstall

install pip:

```
# as root
pip uninstall pyvirtualdisplay
```

CHAPTER

THREE

USAGE

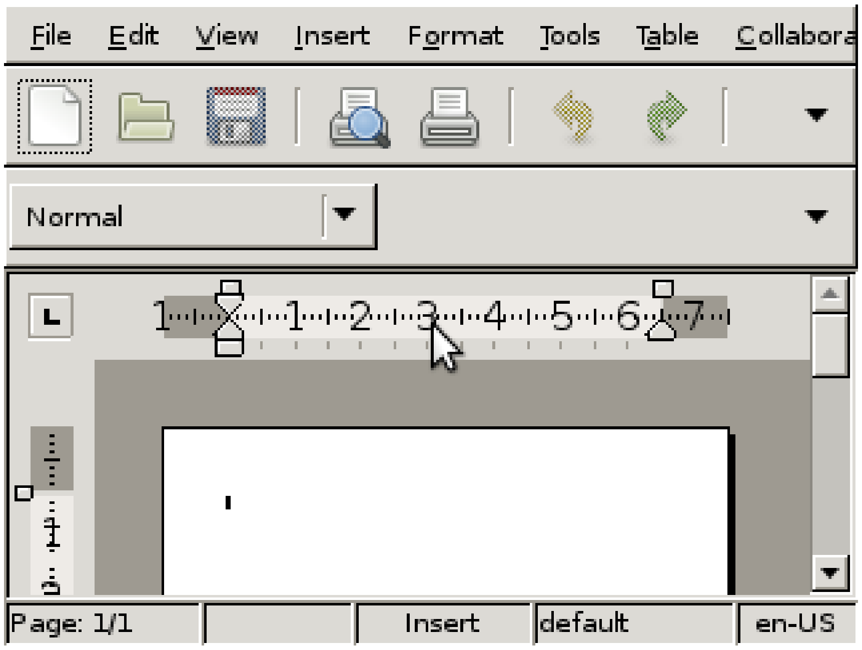
3.1 GUI Test

Testing `abiword` on low resolution:

```
from easyprocess import EasyProcess
from pyvirtualdisplay import Display

Display(visible=1, size=(320, 240)).start()
EasyProcess('abiword').start()
```

```
$ python -m pyvirtualdisplay.examples.lowres
```



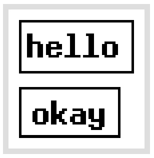
3.2 Screenshot

Create screenshot of `xmessage` in background:

```
from easyprocess import EasyProcess
from pyvirtualdisplay.smartdisplay import SmartDisplay

disp = SmartDisplay(visible=0, bgcolor='black').start()
xmessage = EasyProcess('xmessage hello').start()
img = disp.waitgrab()
xmessage.stop()
disp.stop()
img.show()
```

```
$ python -m pyvirtualdisplay.examples.screenshot1
```

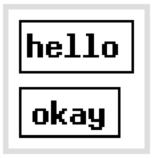


The same with wrap() function:

```
from easyprocess import EasyProcess
from pyvirtualdisplay.smartdisplay import SmartDisplay

disp = SmartDisplay(visible=0, bgcolor='black')
func = disp.wrap(EasyProcess('xmessage hello').wrap(disp.waitgrab))
img=func()
img.show()
```

```
$ python -m pyvirtualdisplay.examples.screenshot2
```



The same using `with` statement:

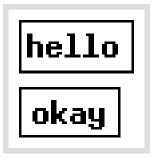
```
'''
using :keyword:`with` statement
'''

from easyprocess import EasyProcess
from pyvirtualdisplay.smartdisplay import SmartDisplay

with SmartDisplay(visible=0, bgcolor='black') as disp:
    with EasyProcess('xmessage hello'):
        img = disp.waitgrab()


img.show()
```

```
$ python -m pyvirtualdisplay.examples.screenshot3
```



3.3 vncserver

examples/vncserver.py

```
'''
Example for Xvnc backend
'''

from easyprocess import EasyProcess
from entrypoint2 import entrypoint
from pyvirtualdisplay.display import Display

@entrypoint
def main(rfbport=5904):
    with Display(backend='xvnc', rfbport=rfbport) as disp:
        with EasyProcess('xmessage hello') as proc:
            proc.wait()
```

CHAPTER

FOUR

API

class `pyvirtualdisplay.`**`Display`**(*backend=None*, *visible=False*, *size=(1024*, *768)*, *color_depth=24*, *bgcolor='black'*, ***kwargs*)

Common class

Parameters

- **color_depth** – [8, 16, 24, 32]
- **size** – screen size (width,height)
- **bgcolor** – background color ['black' or 'white']
- **visible** – True -> Xephyr, False -> Xvfb
- **backend** – 'xvfb', 'xvnc' or 'xephyr', ignores `visible`

`start`()

start display

Return type self

`stop`()

stop display

Return type self

`wrap`(*callable*, *delay=0*)

returns a function which:

1. start process
2. call callable, save result
3. stop process
4. returns result

similar to `with` statement

Return type

class `pyvirtualdisplay.smartdisplay.`**`SmartDisplay`**(*backend=None*, *visible=False*, *size=(1024*, *768)*, *color_depth=24*, *bgcolor='black'*, ***kwargs*)

`autocrop`(*im*)

Crop borders off an image.

@param im Source image. @param bgcolor Background color, using either a color tuple or a color name (1.1.4 only). @return An image without borders, or None if there's no actual content in the image.

`grab`(*autocrop=True*)

`waitgrab`(*timeout=10*, *autocrop=True*, *cb_imgcheck=None*)
start process and create screenshot. Repeat screenshot until it is not empty.

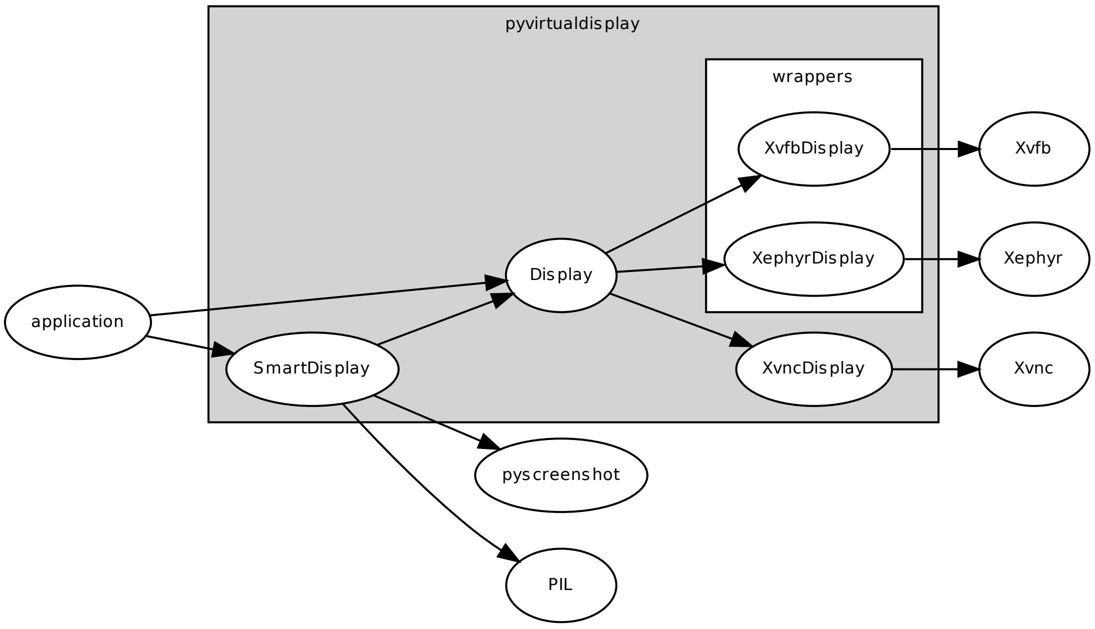
Parameters

- **autocrop** – True -> crop screenshot
- **timeout** – int
- **cb_imgcheck** – callback for testing img, True=accept img, False = reject img

CHAPTER

FIVE

HIERARCHY



CHAPTER

SIX

INDICES AND TABLES

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

INDEX