

---

# pyvirtualdisplay Documentation

*Release 0.0.4*

**ponty**

July 08, 2011

# CONTENTS

|          |                           |           |
|----------|---------------------------|-----------|
| <b>1</b> | <b>Basic usages</b>       | <b>2</b>  |
| <b>2</b> | <b>Installation</b>       | <b>3</b>  |
| 2.1      | General . . . . .         | 3         |
| 2.2      | Ubuntu . . . . .          | 3         |
| 2.3      | Uninstall . . . . .       | 3         |
| <b>3</b> | <b>Usage</b>              | <b>4</b>  |
| 3.1      | GUI Test . . . . .        | 4         |
| 3.2      | Screenshot . . . . .      | 4         |
| <b>4</b> | <b>API</b>                | <b>6</b>  |
| <b>5</b> | <b>Hierarchy</b>          | <b>8</b>  |
| <b>6</b> | <b>Indices and tables</b> | <b>9</b>  |
|          | <b>Index</b>              | <b>10</b> |



## pyvirtualdisplay

**Date** July 08, 2011

**PDF** [pyvirtualdisplay.pdf](#)

Contents:

pyvirtualdisplay is a python wrapper for [Xvfb](#) and [Xephyr](#)

home: <https://github.com/ponty/PyVirtualDisplay>

documentation: <http://ponty.github.com/PyVirtualDisplay>

### Possible applications:

- GUI testing
- automatic GUI screenshot

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

# INSTALLATION

## 2.1 General

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

if you have `setuptools` installed:

```
# as root
easy_install pyvirtualdisplay
```

if you have `pip` installed:

```
# as root
pip install pyvirtualdisplay
```

## 2.2 Ubuntu

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

## 2.3 Uninstall

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

# USAGE

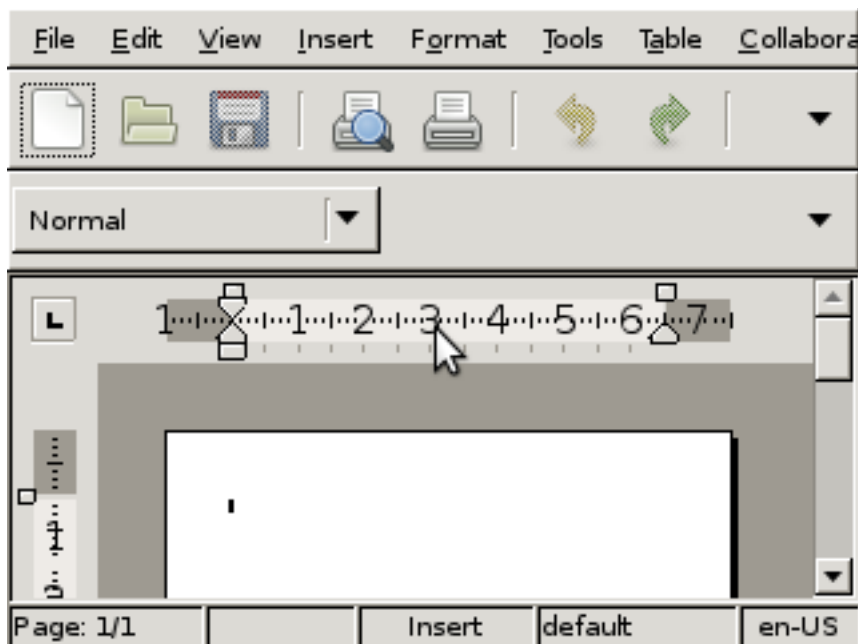
## 3.1 GUI Test

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:

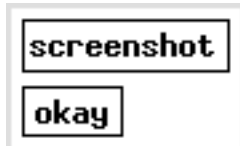
```

from easyprocess import EasyProcess
from pyvirtualdisplay.smartdisplay import SmartDisplay

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

```

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



The same with wrap() function:

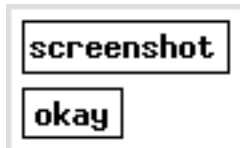
```

from easyprocess import EasyProcess
from pyvirtualdisplay.smartdisplay import SmartDisplay

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

```

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



The same using with statement:

```

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

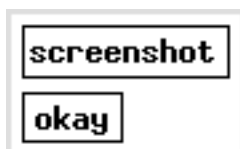
from easyprocess import EasyProcess
from pyvirtualdisplay.smartdisplay import SmartDisplay

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

```

```
img.show()
```

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





# API

**class** pyvirtualdisplay.**Display** (*visible=False, size=(1024, 768), color\_depth=24, bgcolor='black'*)  
Common class for XvfbDisplay and XephyrDisplay

**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** (*visible=False, size=(1024, 768), color\_depth=24, bgcolor='black'*)

**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*)

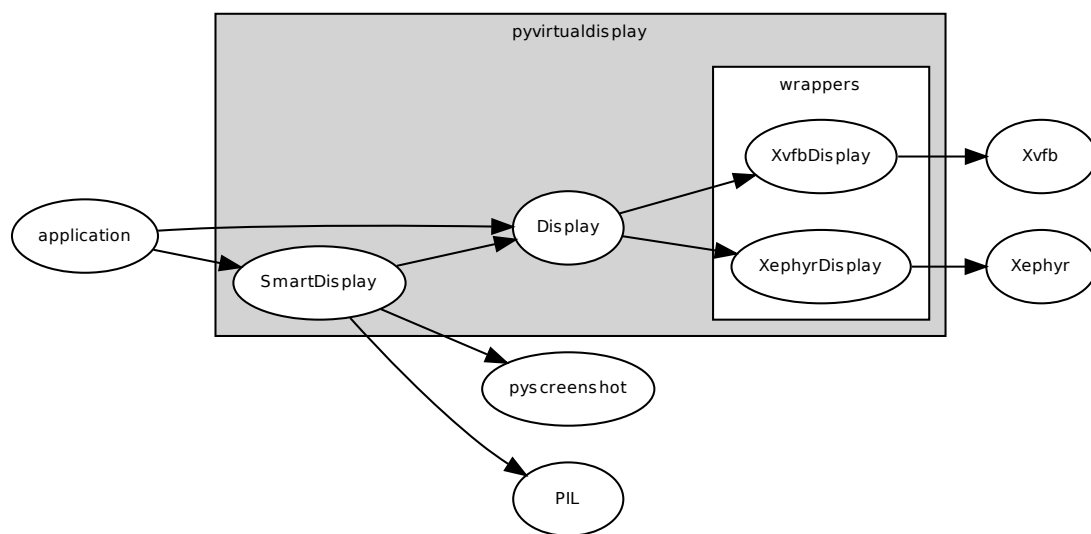
**waitgrab** (*timeout=10, autocrop=True, cb\_imgcheck=None*)  
start process and create screenshot. Repeat screenshot until it is not empty.

**Parameters**

- **autocrop** – True -> crop screenshot
- **timeout** – int

- **cb\_imgcheck** – callback for testing img, True=accept img, False = reject img

# HIERARCHY



# INDICES AND TABLES

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

# INDEX

## A

`autocrop()` (`pyvirtualdisplay.smartdisplay.SmartDisplay` method), [6](#)

## D

`Display` (class in `pyvirtualdisplay`), [6](#)

## G

`grab()` (`pyvirtualdisplay.smartdisplay.SmartDisplay` method), [6](#)

## S

`SmartDisplay` (class in `pyvirtualdisplay.smartdisplay`), [6](#)

`start()` (`pyvirtualdisplay.Display` method), [6](#)

`stop()` (`pyvirtualdisplay.Display` method), [6](#)

## W

`waitgrab()` (`pyvirtualdisplay.smartdisplay.SmartDisplay` method), [6](#)

`wrap()` (`pyvirtualdisplay.Display` method), [6](#)