**Installation instructions**

Assuming you are using a Raspberry Pi (running Debian Jessie), follow the pre-requisites & instructions in the above repositories to wire up your display, then from a command-line:

$ sudo usermod -a -G i2c,spi,gpio pi

$ sudo apt install python-dev python-pip libfreetype6-dev libjpeg-dev

$ sudo apt install libsdl-dev libportmidi-dev libsdl-ttf2.0-dev libsdl-mixer1.2-dev libsdl-image1.2-dev

$ sudo -H pip install --upgrade pip setuptools

$ sudo apt purge python-pip

Log out and in again and clone this repository:

$ git clone https://github.com/rm-hull/luma.examples.git

$ cd luma.examples

Finally, install the luma libraries using:

$ sudo -H pip install -e .

**Running the examples**

After cloning the repository, enter the examples directory and try running the following examples:

| **Example** | **Description** |
| --- | --- |
| 3d\_box.py | Rotating 3D box wireframe & color dithering |
| bounce.py | Display a bouncing ball animation and frames per second |
| carousel.py | Showcase viewport and hotspot functionality |
| chroma.py | Trippy color rendering demo |
| clock.py | An analog clockface with date & time |
| colors.py | Color rendering demo |
| crawl.py | A vertical scrolling demo, which should be familiar |
| demo.py | Use misc draw commands to create a simple image |
| font\_awesome.py | A meander through some awesome fonts |
| game\_of\_life.py | Conway's game of life |
| grayscale.py | Greyscale rendering demo |
| invaders.py | Space Invaders demo |
| jetset\_willy.py | Sprite animation framework demo |
| maze.py | Maze generator |
| perfloop.py | Simple benchmarking utility to measure performance |
| picamera\_photo.py | Capture photo with picamera and display it on a screen |
| picamera\_video.py | Capture continuous video stream and display it on a screen |
| pi\_logo.py | Display the Raspberry Pi logo (loads image as .png) |
| runner.py | Sprite animation framework demo |
| savepoint.py | Example of savepoint/restore functionality |
| scrolling\_pixelart.py | Image dithering and viewport scrolling |
| sprite\_animation.py | Using sprite maps for animation effects |
| starfield.py | 3D starfield simulation |
| sys\_info.py | Display basic system information |
| terminal.py | Simple println capabilities |
| tv\_snow.py | Example image-blitting |
| tweet\_scroll.py | Using Twitter's Streaming API to display scrolling notifications |
| video.py | Display a video clip |
| weather.py | 3-day weather forecasts from the BBC |
| welcome.py | Unicode font rendering & scrolling |

By default, all the examples will asume I2C port 1, address 0x3C and the ssd1306 driver. If you need to use a different setting, these can be specified on the command line – each program can be invoked with a --help flag to show the options:

$ python examples/demo.py --help

usage: demo.py [-h] [--config CONFIG] [--display] [--width WIDTH]

[--height HEIGHT] [--rotate] [--interface]

[--i2c-port I2C\_PORT] [--i2c-address I2C\_ADDRESS]

[--spi-port SPI\_PORT] [--spi-device SPI\_DEVICE]

[--spi-bus-speed SPI\_BUS\_SPEED]

[--gpio-data-command GPIO\_DATA\_COMMAND]

[--gpio-reset GPIO\_RESET] [--gpio-backlight GPIO\_BACKLIGHT]

[--block-orientation] [--mode] [--framebuffer] [--bgr]

[--h-offset H\_OFFSET] [--v-offset V\_OFFSET]

[--backlight-active] [--transform] [--scale SCALE]

[--duration DURATION] [--loop LOOP] [--max-frames MAX\_FRAMES]

luma.examples arguments

optional arguments:

-h, --help show this help message and exit

General:

--config CONFIG, -f CONFIG

Load configuration settings from a file (default:

None)

--display , -d Display type, supports real devices or emulators.

Allowed values are: ssd1306, ssd1322, ssd1325,

ssd1331, sh1106, pcd8544, st7735, max7219, ws2812,

neopixel, apa102, capture, gifanim, pygame (default:

ssd1306)

--width WIDTH Width of the device in pixels (default: 128)

--height HEIGHT Height of the device in pixels (default: 64)

--rotate , -r Rotation factor. Allowed values are: 0, 1, 2, 3

(default: 0)

--interface , -i Serial interface type. Allowed values are: i2c, spi,

bitbang (default: i2c)

I2C:

--i2c-port I2C\_PORT I2C bus number (default: 1)

--i2c-address I2C\_ADDRESS

I2C display address (default: 0x3C)

SPI:

--spi-port SPI\_PORT SPI port number (default: 0)

--spi-device SPI\_DEVICE

SPI device (default: 0)

--spi-bus-speed SPI\_BUS\_SPEED

SPI max bus speed (Hz) (default: 8000000)

GPIO:

--gpio-data-command GPIO\_DATA\_COMMAND

GPIO pin for D/C RESET (SPI devices only) (default:

24)

--gpio-reset GPIO\_RESET

GPIO pin for RESET (SPI devices only) (default: 25)

--gpio-backlight GPIO\_BACKLIGHT

GPIO pin for backlight (PCD8544, ST7735 devices only)

(default: 18)

Misc:

--block-orientation Fix 90° phase error (MAX7219 LED matrix only). Allowed

values are: 0, 90, -90 (default: 0)

--mode Colour mode (SSD1322, SSD1325 and emulator only).

Allowed values are: 1, RGB, RGBA (default: RGB)

--framebuffer Framebuffer implementation (SSD1331, SSD1322, ST7735

displays only). Allowed values are: diff\_to\_previous,

full\_frame (default: diff\_to\_previous)

--bgr Set if LCD pixels laid out in BGR (ST7735 displays

only). (default: False)

--h-offset H\_OFFSET Horizontal offset (in pixels) of screen to display

memory (ST7735 displays only) (default: 0)

--v-offset V\_OFFSET Vertical offset (in pixels) of screen to display

memory (ST7735 displays only) (default: 0)

--backlight-active Set to "low" if LCD backlight is active low (default),

else "high" otherwise (PCD8544, ST7735 displays only).

Allowed values are: low, high (default: low)

Emulator:

--transform Scaling transform to apply (emulator only). Allowed

values are: identity, led\_matrix, none, scale2x,

seven\_segment, smoothscale (default: scale2x)

--scale SCALE Scaling factor to apply (emulator only) (default: 2)

--duration DURATION Animation frame duration (gifanim emulator only)

(default: 0.01)

--loop LOOP Repeat loop, zero=forever (gifanim emulator only)

(default: 0)

--max-frames MAX\_FRAMES

Maximum frames to record (gifanim emulator only)

(default: None)

Note

1. Substitute python3 for python in the above examples if you are using python3.
2. python-dev (apt-get) and psutil (pip/pip3) are required to run the sys\_info.py example. See [install instructions](https://github.com/rm-hull/luma.examples/blob/master/examples/sys_info.py#L10-L13) for the exact commands to use.

**Emulators**

There are various display emulators available for running code against, for debugging and screen capture functionality:

* The luma.emulator.device.capture device will persist a numbered PNG file to disk every time its display method is called.
* The luma.emulator.device.gifanim device will record every image when its display method is called, and on program exit (or Ctrl-C), will assemble the images into an animated GIF.
* The luma.emulator.device.pygame device uses the pygame library to render the displayed image to a pygame display surface.

Invoke the demos with:

$ python examples/clock.py --display capture

or:

$ python examples/clock.py --display pygame

**Documentation**

Full documentation with installation instructions can be found in:

* [https://luma-oled.readthedocs.io](https://luma-oled.readthedocs.io/)
* [https://luma-lcd.readthedocs.io](https://luma-lcd.readthedocs.io/)
* [https://luma-led-matrix.readthedocs.io](https://luma-led-matrix.readthedocs.io/)
* [https://luma-core.readthedocs.io](https://luma-core.readthedocs.io/)
* [https://luma-emulator.readthedocs.io](https://luma-emulator.readthedocs.io/)