terminedia

Release 0.4.0dev0

João S. O. Bueno

Mar 21, 2023

CONTENTS:

CHAPTER

ONE

TERMINEDIA PACKAGE

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

Terminedia is meant to be a lightweight terminal library to provide color output, positioned text, and use block characters for displaying low-resolution graphics in the terminal itself.

Simple example:

```
from terminedia import Screen, pause

with Screen() as scr:
    scr.context.color = 1, 0, 0
    scr.high.draw.rect((5, 5), (30, 20))

    pause()
```

This will draw a rougly square rectangle (due to character block non square aspect-ratio) using 1/4 block unicode characters to draw lines. The output is:

(spacing between blocks will depend on the terminal program, font, and window size)

Besides allowing positioning text and graphics, there are resources that allow non blocking keyboard reading, and capability to read keypresses from the Arrow Keys, Escape and Function keys- which enables building rich, keyboard based, interfaces directly in the terminal, as well as simulating a vintage "8-bitish" personal computer era style games and apps.

The "examples" folder in the project source have other simple examples.

Bellow you will find all the documentation for the library, including some low-level and internal use modules. Unless you need to do something too specific, you can make use of the higher level `Screen` class.

It should be used as a context manager, and within the managed block, you can use the Screen's instance public methods, as well as its `Screen.context` attribute to select colors and text-direction, and it's `Screen.draw` attribute to access drawing primitives, which are the methods avaliable on the `Drawing` class. Last but not least, the attribute `Screen.high` on the Screen instance enables the use of 1/4 character "pixels" to draw in a seamless way. Just use the `.high.draw` namespace to access the drawing primitives the same way the `.draw` namespace allows drawing with full-block characters.

For input, the available functionalities are the `keyboard` context manager, which enables non-blocking use of the keyboard, and the `inkey` and `pause` functions which make use of this feature.

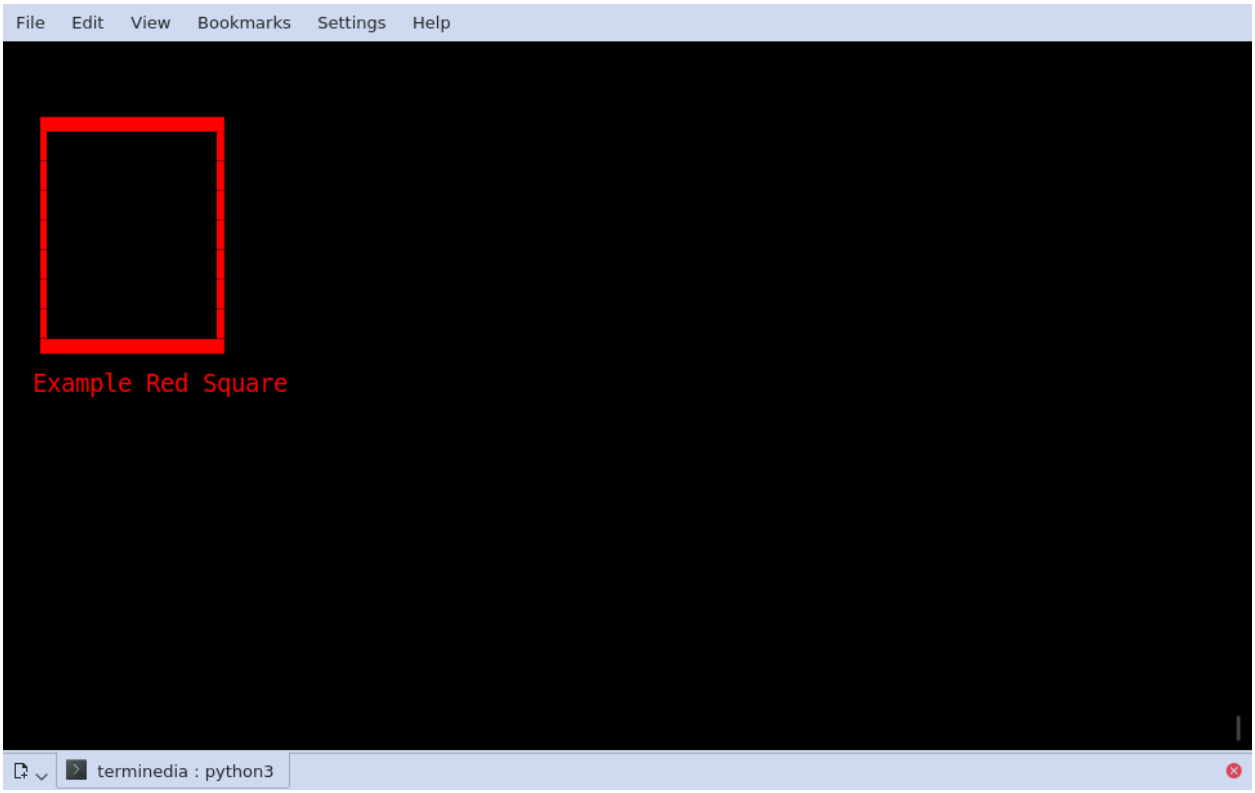

Fig. 1: Text-rectangle drawn on terminal by simple example.

CHAPTER

THREE

TERMINEDIA