cursebox Documentation

Release 1.0

Hamza Haiken

Sep 01, 2018

Contents

Curses made easy

Cursebox is a library based around the curses standard module. Its goal is to avoid the C-like ceremony of curses and provide a modern approach to terminal drawing:

```
>>> from cursebox import *
>>> with Cursebox() as cb:
...     width, height = cb.width, cb.height
...     greeting = "Hello, World!"
...     # Center text on the screen
...     cb.put(x=(width - len(greeting)) / 2,
...            y=height / 2, text=greeting,
...            fg=colors.black, bg=colors.white)
...     # Wait for any keypress
...     cb.poll_event()
```

It provides several useful features:

- No setup/teardown
- RGB conversion to terminal 256-colors palette
- Event management

CHAPTER 1

Installation

You can install cursebox directly from PyPI using PIP:

```
$ pip install cursebox
```

To test the library, you can run it as a module to see the demo animation:

```
$ python -m cursebox
```

CHAPTER 2

Basic Usage

CHAPTER 3

API Reference

3.1 Classes

class cursebox.**Cursebox**(*blocking_events=True*)

A wrapper for curses which provides simple API calls.

Instances should be created using the with statement, which will take care of initializing the curses environment and disposing of it when the context is lost:

```
>>> with Cursebox() as cb:
>>>     cb.put(42, 13, "Hello from curses!", colors.black, colors.white)
>>>     cb.refresh()
```

Cursebox also handles keyboard strokes and events:

```
>>> event = cb.poll_event()
>>> if event == EVENT_CTRL_C:
>>>     exit()
```

add_thread(*thread*)

clear()

Clears the terminal.

height

The height of the current terminal.

hide_cursor()

Hides the cursor.

poll_event()

Checks if an event happens and returns a string related to the event.

Returns -1 if nothing happened during self.screen.timeout milliseconds.

If the event is a normal (letter) key press, the letter is returned (case sensitive)

Returns Event type

put (*x*, *y*, *text*, *fg*, *bg*)
Puts a string at the desired coordinates using the provided colors.

Parameters

- **x** – X position
- **y** – Y position
- **text** – Text to write
- **fg** – Foreground color number
- **bg** – Background color number

put_arrow (*x*, *y*, *direction*, *fg*, *bg*)

refresh ()
Refreshes the screen.

set_cursor (*x*, *y*)
Sets the cursor to the desired position.

Parameters

- **x** – X position
- **y** – Y position

width
The width of the current terminal.

class cursebox.pairs.**Pairs**
Collection object that stores curses color pairs.

When a color pair is needed, it can be retrieved by calling __getitem__:

```
>>> pairs = Pairs()
>>> a_pair = pairs[0, 3]
```

If the pair does not exist, it is created and pushed in a FIFO dictionary of limited size (32767 entries). A very big terminal will never have more than 10000 characters displayed, so this enables to use virtually all color combinations at all times.

reset ()

Index