

A_color_picker_in_python

When you want to let the user choose a color for a GUI element you need to display colors without annoying the user with technical details about RGB values! This example shows you a way to do it.

Remark : this code is using the easy keyboard handling method.

```
// Usual code as always
import e32
from appuifw import *
from key_codes import *

class Keyboard(object):
    def __init__(self):
        self.state = {} # is this key pressing ?
        self.buffer= {} # is it waiting to be processed ?
    def handle_event(self, event): # for event_callback
        code = event['scancode']
        if event['type'] == EEventKeyDown:
            self.buffer[code]= 1 # put into queue
            self.state[code] = 1
        elif event['type'] == EEventKeyUp:
            self.state[code] = 0
    def pressing(self, code): # just check
        return self.state.get(code,0)
    def pressed(self, code): # check and process the event
        if self.buffer.get(code,0):
            self.buffer[code] = 0 # take out of queue
            return 1
        return 0

key = Keyboard()
app.body = canvas = Canvas(event_callback=key.handle_event)

def quit():
    global running
    running = 0

app.exit_key_handler = quit
running = 1

ff00 = range(0xff, -1, -0x33)
pal = [(r,g,b) for r in ff00 for g in ff00 for b in ff00] # web-safe 216 colors
map_j = range(0,12,2)+range(11,0,-2) # make better grouping
for j in range(12):
    for i in range(18):
        k = 18*map_j[j] + i
        canvas.rectangle([(9*i+1, 9*j+1), (9*i+9, 9*j+9)], None, pal[k])

def clear_box(color=0xfffff):
    global x,y
    canvas.rectangle([(9*x, 9*y), (9*x+10, 9*y+10)], color) # cursor

x, y = 0, 0
black_white = 0
while running:
    if key.pressed(EScancodeLeftArrow):
        clear_box()
        if x > 0: x -= 1
    if key.pressed(EScancodeRightArrow):
        clear_box()
```

A_color_picker_in_python

```
    if x < 17: x += 1
if key.pressed(EScancodeUpArrow):
    clear_box()
    if y > 0: y -= 1
if key.pressed(EScancodeDownArrow):
    clear_box()
    if y < 11: y += 1
if key.pressed(EScancodeSelect):
    color = pal[18*map_j[y] + x]
    canvas.rectangle([(1,109), (17*9+9,130)], None, color) # show color
black_white ^= 0xffffffff # toggle
clear_box(black_white)
e32.ao_sleep(0.2)
```