

Archived:Uma classe simples para implementar uma barra de progresso, em PySymbian

 Aquivado: Este artigo foi [arquivado](#), pois o conteúdo não é mais considerado relevante para se criar soluções comerciais atuais. Se você achar que este artigo ainda é importante, inclua o template {{ForArchiveReview|escreva a sua justificativa}}.

Acredita-se que este artigo ainda seja válido no contexto original (quando ele foi escrito)

Introdução

Aqui está uma implementação de uma barra de progresso "ProgressBar" utilizando topwindow em Python.

Uso

```
pb = ProgressBar()  
pb.set_text(u"Por favor aguarde...")  
i = 0  
for i in range(100):  
    pb.set_value(i)  
pb.close()  
del pb
```

Screenshots de exemplos



O código

Obs.: Como você pode notar, eu também estou usando a barra de ferramentas da tela feita por Marcelo Barros (Link: [Toolbar on canvas for touch and non touch S60 devices](#) (Inglês))

```

from graphics import *
from e32 import *
from TopWindow import *
from sysinfo import display_pixels

class ProgressBar(object):
    """Implemente uma Barra de progresso em TopWindow
    """
    def __init__(self, start=0, end=100, color=(0,0,77),
                 fill=(255,255,200), outline=(0,0,0)):
        screen_size = display_pixels()
        #tamanhos e posições
        self.height = 60
        self.width = int(screen_size[0] * 0.8)
        self.top = screen_size[1] - self.height - 5
        self.left = int((screen_size[0] - self.width) / 2)
        #Tamanho da barra de progresso
        self.progress_margin = 5
        self.progress_w = self.width - (2 * self.progress_margin)
        self.progress_h = 18 #altura da barra de progresso
        self.progress_l = self.progress_margin
        self.progress_t = self.height - self.progress_h - \
                          self.progress_margin
        #barra de progresso interna espera que o externo tenha 1px de fronteira
        self.internal_w_max = self.progress_w - 2
        self.internal_h = self.progress_h - 2
        self.internal_l = self.progress_l + 1
        self.internal_t = self.progress_t + 1
        self.internal_w = 0
        self.glow_h = int(self.internal_h / 2)
        #cores e valores
        self.start = start
        self.end = end
        self.value = start
        self.color = color
        self.glow_color = self.color_combine(color, (255,255,255), 0.5)
        self.outline = outline
        self.fill = fill
        #atributos de textos
        self.caption = u"""
        self.font = (u"dense", 12, FONT_ANTIALIAS)
        self.text_top = self.progress_t - \
                        self.progress_margin - \
                        self.font[1]
        #criando uma topwindow
        self.window = TopWindow()
        self.window.corner_type = 'square'
        self.window.position = (self.left, self.top)
        self.window.size = (self.width, self.height)
        self.canvas = Image.new(self.window.size)
        self.window.add_image(self.canvas,(0,0,self.width,self.height))
        #mostra a barra de progresso
        self.redraw()

```

```
self.window.show()

def close(self):
    #Fecha a janela e libera a imagem do buffers de memória
    del self.canvas
    self.window.hide()
    del self.window

def set_text(self, text):
    self.caption = text
    self.redraw()

def set_value(self, value):
    if value > self.end:
        value = self.end
    elif value < self.start:
        value = self.start
    self.value = value
    self.internal_w = int(((1.0 * self.value - self.start)/ \
                           (1.0 * self.end - self.start))  \
                           * self.internal_w_max)
    self.redraw()

def redraw(self):
    #Você não precisa chamar o método redesenhar. Basta usar set_value para
    redesenhar a barra de progresso
    #janela externa
    self.canvas.rectangle((0, 0, self.width, self.height),
                          outline=self.outline,
                          fill=self.fill)
    #fronteira externa da barra de progresso
    self.canvas.rectangle((self.progress_l,
                          self.progress_t,
                          self.progress_l + self.progress_w,
                          self.progress_t + self.progress_h),
                          outline=self.outline,
                          fill=self.fill)
    #conteúdo da barra de progresso com brilho
    self.canvas.rectangle((self.internal_l,
                          self.internal_t,
                          self.internal_l + self.internal_w,
                          self.internal_t + self.internal_h),
                          outline=None,
                          fill=self.color)
    self.canvas.rectangle((self.internal_l,
                          self.internal_t,
                          self.internal_l + self.internal_w,
                          self.internal_t + self.glow_h),
                          outline=None,
                          fill=self.glow_color)
    #legenda da janela
    self.canvas.text((self.progress_margin, self.text_top),
                    self.caption, fill = self.outline,
                    font = self.font)
    #troca de imagens
    self.window.remove_image(self.window.images[0][0])
    self.window.add_image(self.canvas, (0,0,self.width,self.height))
```

```
ao_sleep(0.001)

def color_combine(self, c1, c2, perc):
    c = map(lambda a,b: int(a*(1-perc)+0.5) + int(b*perc+0.5), c1, c2)
    return tuple(c)
```