TUIs com Textual

Live de python # 240

Roteiro

1. TUI

O que são Terminal User Interfaces

2. Textual

Uma introdução

3. Widgets

Uma visão geral

4. Debug

Quando as coisas não dão certo

picpay.me/dunossauro

apoia.se/livedepython

pix.dunossauro@gmail.com

Ajude o projeto <3

Ademar Peixoto, Adilson Herculano, Adriano Ferraz, Alemao, Alexandre Harano, Alexandre Lima, Alexandre Takahashi, Alexandre Villares, Alex Lima, Alynne Ferreira, Alysson Oliveira, Ana Carneiro, Andre Azevedo, Andre Mesquita, Aquiles Coutinho, Arnaldo Turque, Aslay Clevisson, Aurelio Costa, Bernardo At, Bernardo Fontes, Bruno Almeida, Bruno Barcellos, Bruno Barros, Bruno Freitas, Bruno Lopes, Bruno Ramos, Caio Nascimento, Christiano Morais, Damianth, Daniel Freitas, Daniel Wojcickoski, Danilo Boas, Danilo Segura, Danilo Silva, David Couto, David Kwast, Davi Goivinho, Davi Souza, Delton Porfiro, Denis Bernardo, Diego Farias, Diego Guimarães, Dilenon Delfino, Diogo Paschoal, Diogo Silva, Edgar, Eduardo Silveira, Eduardo Tolmasquim, Elias Silva, Emerson Rafael, Eneas Teles, Erick Andrade, Érico Andrei, Everton Silva, Fabiano Tomita, Fabio Barros, Fábio Barros, Fabio Castro, Fábio Thomaz, Fabricio Patrocinio, Felipe Rodrigues, Fernanda Prado, Fernando Celmer, Firehouse, Flávio Meira, Francisco Neto, Francisco Silvério, Gabriel Espindola, Gabriel Mizuno, Gabriel Paiva, Gabriel Simonetto, Geandreson Costa, Geizielder, Gilberto Abrao, Giovanna Teodoro, Giuliano Silva, Guilherme Felitti, Guilherme Gall, Guilherme Silva, Guionardo Furlan, Gustavo Pereira, Gustavo Suto, Harold Gautschi, Heitor Fernandes, Helvio Rezende, Hugo Cosme, Igor Riegel, Italo Silva, Janael Pinheiro, Jean Victor, Joelson Sartori, Jônatas Oliveira, Jônatas Silva, Jon Cardoso, Jorge Silva, José Gomes, Joseíto Júnior, Jose Mazolini, Juan Felipe, Juan Gutierrez, Juliana Machado, Julio Franco, Júlio Gazeta, Julio Silva, Kaio Peixoto, Kálita Lima, Kaneson Alves, Leandro Miranda, Leandro Silva, Leo Ivan, Leonardo Mello, Leonardo Nazareth, Leon Solon, Luancomputacao Roger, Lucas Adorno, Lucas Carderelli, Lucas Mendes, Lucas Nascimento, Lucas Schneider, Lucas Simon, Lucas Valino, Luciano Filho, Luciano Ratamero, Luciano Teixeira, Luis Alves, Luis Eduardo, Luiz Duarte, Luiz Lima, Luiz Paula, Luiz Perciliano, Mackilem Laan, Marcelo Campos, Marcio Moises, Marco Mello, Marcos Gomes, Maria Clara, Marina Passos, Mateus Lisboa, Mateus Ribeiro, Mateus Silva, Matheus Silva, Matheus Vian, Mauricio Nunes, Mírian Batista, Mlevi Lsantos, Murilo Viana, Nathan Branco, Nicolas Teodosio, Otávio Carneiro, Patricia Minamizawa, Patrick Felipe, Paulo Tadei, Pedro Henrique, Pedro Pereira, Peterson Santos, Priscila Santos, Pydocs Pro, Pytonyc, Rafael Lopes, Rafael Romão, Rafael Veloso, Raimundo Ramos, Ramayana Menezes, Regis Santos, Renato Oliveira, Rene Bastos, Ricardo Silva, Riverfount, Rjribeiro, Robson Maciel, Rodrigo Barretos, Rodrigo Freire, Rodrigo Oliveira, Rodrigo Quiles, Rodrigo Ribeiro, Rodrigo Vaccari, Rodrigo Vieira, Rogério Nogueira, Ronaldo Silveira, Rui Jr, Samanta Cicilia, Téo Calvo, Thaynara Pinto, Thiago Araujo, Thiago Borges, Thiago Curvelo, Thiago Souza, Tiago Minuzzi, Tony Dias, Tyrone Damasceno, Uadson Emile, Valcilon Silva, Valdir Tegon, Vcwild, Vicente Marcal, Vinicius Stein, Vladimir Lemos, Walter Reis, William Vitorino, Willian Lopes, Wilson Duarte, Wilson Neto, Zeca Figueiredo

Obrigado você

Terminal User Interfaces

TUI

Terminal User Interfaces

GUI

CLI

Terminal User Interfaces

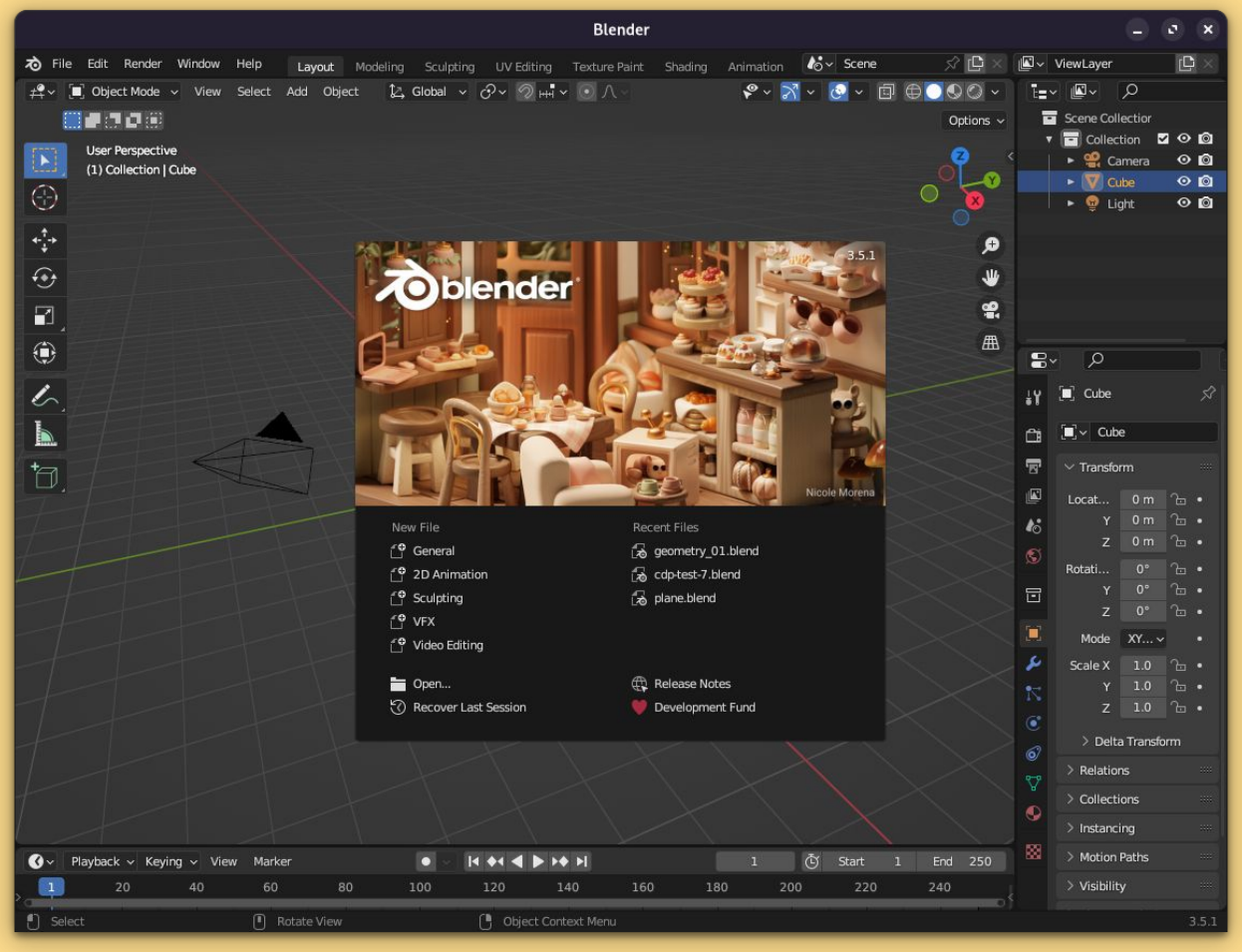
GUI

Aplicações baseadas em janelas. Como praticamente tudo que usamos.

CLI

Aplicações baseadas em linhas de texto. Programas que chamamos no terminal com flags e parâmetros.

GUIs – Graphical User Interface



CLI – Command Line Interface

```
dunossauro@babbage:~/lives-betas-2023/live_textual
py-3.12.0b1 babbage in ~/lives-betas-2023/live_textual
○ → poetry add textual
Using version ^0.27.0 for textual

Updating dependencies
Resolving dependencies... (0.4s)


Package operations: 11 installs, 0 updates, 0 removals

  • Installing mdurl (0.1.2)
  • Installing markdown-it-py (2.2.0)
  • Installing uc-micro-py (1.0.2)
  • Installing linkify-it-py (2.0.2)
  • Installing mdit-py-plugins (0.4.0)
  • Installing pygments (2.15.1)
  • Installing zipp (3.15.0)
  • Installing importlib-metadata (6.6.0)
  • Installing rich (13.4.2)
  • Installing typing-extensions (4.6.3)
  • Installing textual (0.27.0)

Writing lock file
```

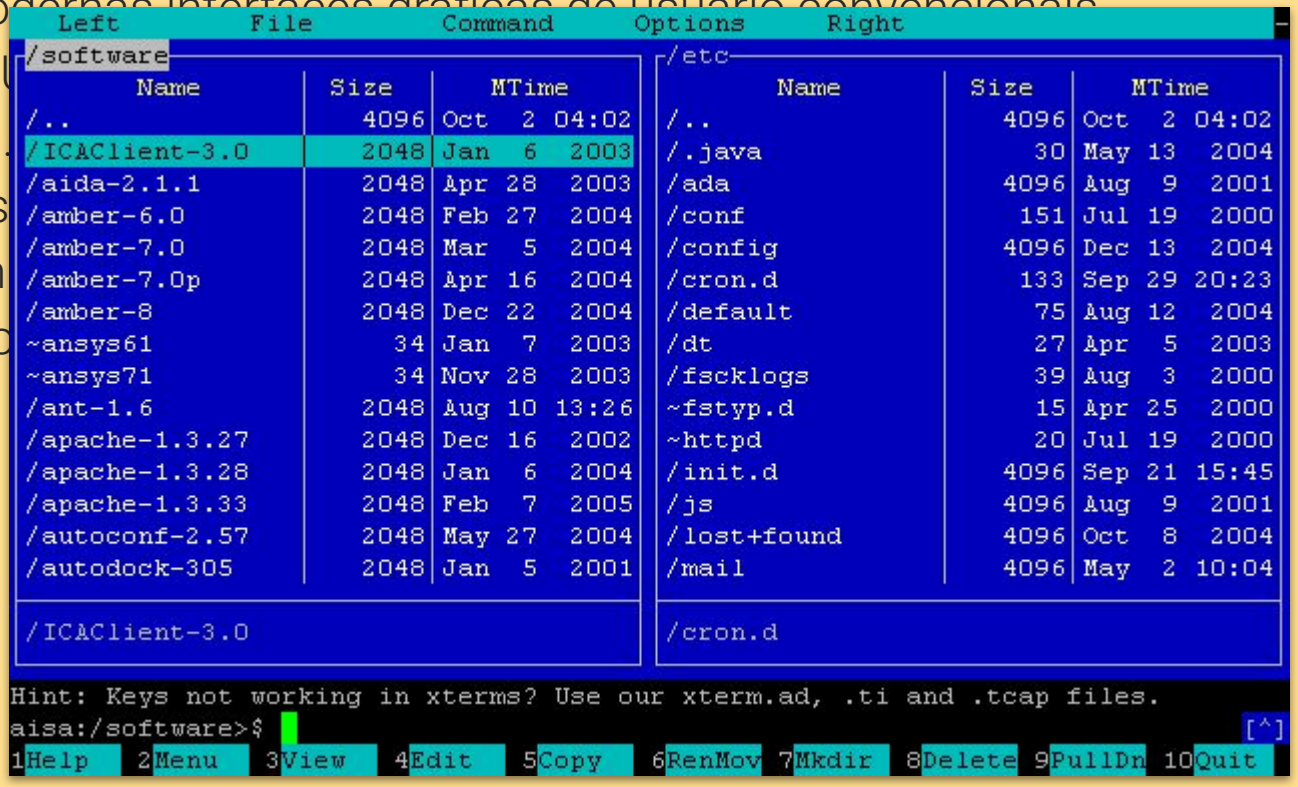
Interfaces de usuário baseadas em terminal

É um **retrônimo** que descreve um tipo de interface de usuário (UI) comum como uma forma inicial da interação humano-computador, antes do advento das modernas interfaces gráficas de usuário convencionais (GUIs). Como GUIs, eles podem usar toda a área da tela e aceitar mouse e outras entradas. gráficos especiais, Eles também podem usar cores e muitas vezes estruturar a exibição usando caracteres como ╔ e ╣, referidos em Unicode como o conjunto de "desenho de caixa". O contexto moderno de uso é geralmente um emulador de terminal ."

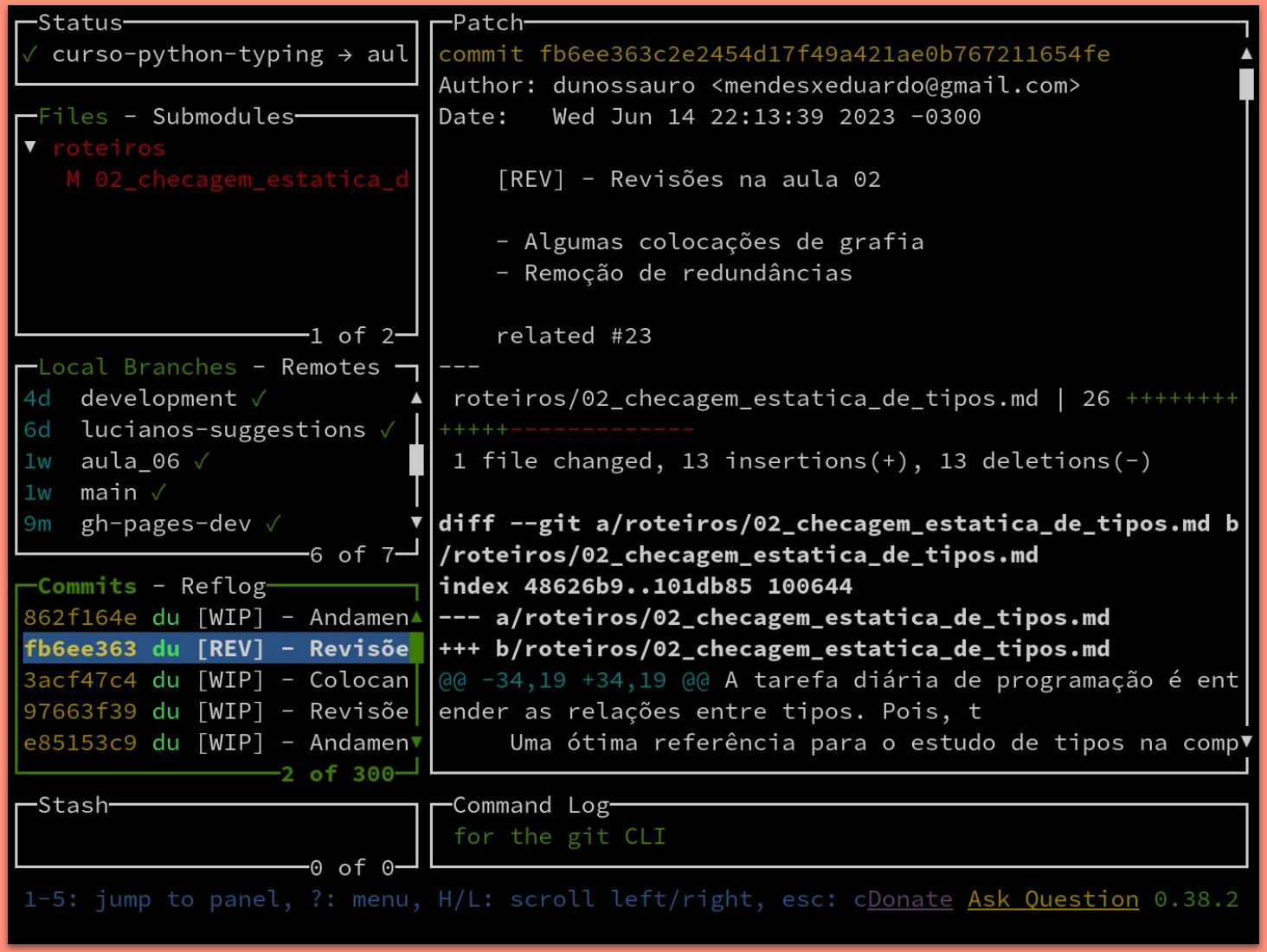
https://en.wikipedia.org/wiki/Text-based_user_interface

Interfaces de usuário baseadas em terminal

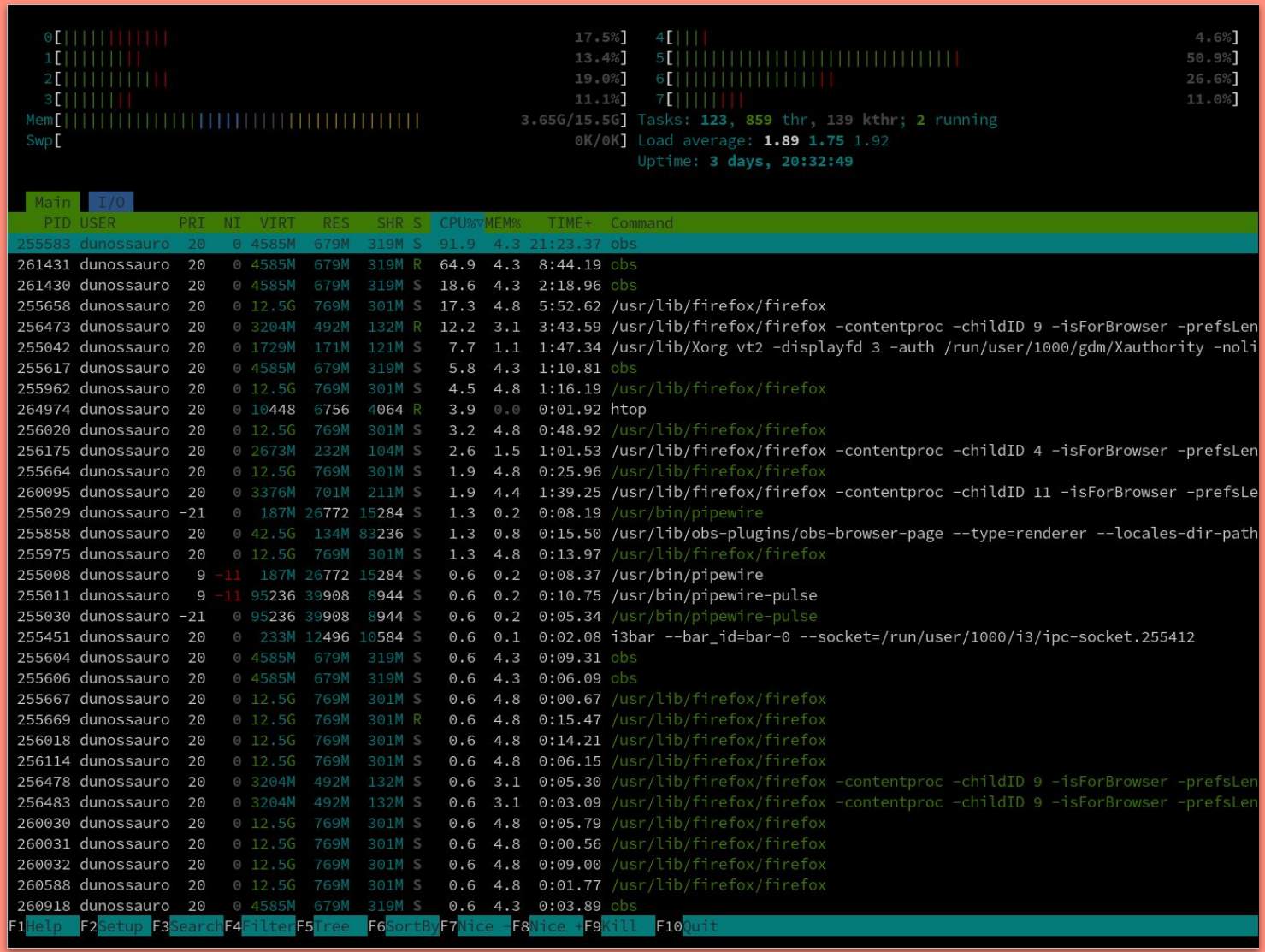
É um **retrônimo** que descreve um tipo de interface de usuário (UI) comum como uma forma inicial da interação humano-computador, antes do advento das mo
(GUIs). Como GU
outras entradas.
muitas vezes es
referidos em Un
moderno de uso



LazyGit


Status
✓ curso-python-typing → aul
Files - Submodules
▾ roteiros
M 02_checagem_estatica_d
1 of 2
Local Branches - Remotes
4d development ✓
6d lucianos-suggestions ✓
1w aula_06 ✓
1w main ✓
9m gh-pages-dev ✓
6 of 7
Commits - Reflog
862f164e du [WIP] - Andamen
fb6ee363 du [REV] - Revisõe
3acf47c4 du [WIP] - Colocan
97663f39 du [WIP] - Revisõe
e85153c9 du [WIP] - Andamen
2 of 300
Stash
0 of 0
Patch
commit fb6ee363c2e2454d17f49a421ae0b767211654fe
Author: dunossauro <mendesxeduardo@gmail.com>
Date: Wed Jun 14 22:13:39 2023 -0300
[REV] - Revisões na aula 02
- Algumas colocações de grafia
- Remoção de redundâncias
related #23
---
roteiros/02_checagem_estatica_de_tipos.md | 26 ++++++++
1 file changed, 13 insertions(+), 13 deletions(-)
diff --git a/roteiros/02_checagem_estatica_de_tipos.md b
/roteiros/02_checagem_estatica_de_tipos.md
index 48626b9..101db85 100644
--- a/roteiros/02_checagem_estatica_de_tipos.md
+++ b/roteiros/02_checagem_estatica_de_tipos.md
@@ -34,19 +34,19 @@ A tarefa diária de programação é ent
ender as relações entre tipos. Pois, t
Uma ótima referência para o estudo de tipos na comp
Command Log
for the git CLI
1-5: jump to panel, ?: menu, H/L: scroll left/right, esc: cDonate Ask Question 0.38.2

HTOP


0[17.5%] 4[4.6%]
1[13.4%] 5[50.9%]
2[19.0%] 6[26.6%]
3[11.1%] 7[11.0%]
Mem[3.65G/15.5G] Tasks: 123, 859 thr, 139 kthr; 2 running
Swp[0K/0K] Load average: 1.89 1.75 1.92
Uptime: 3 days, 20:32:49
Main I/O
PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
255583 dunossauro 20 0 4585M 679M 319M S 91.9 4.3 21:23.37 obs
261431 dunossauro 20 0 4585M 679M 319M R 64.9 4.3 8:44.19 obs
261430 dunossauro 20 0 4585M 679M 319M S 18.6 4.3 2:18.96 obs
255658 dunossauro 20 0 12.5G 769M 301M S 17.3 4.8 5:52.62 /usr/lib/firefox/firefox
256473 dunossauro 20 0 3204M 492M 132M R 12.2 3.1 3:43.59 /usr/lib/firefox/firefox -contentproc -childID 9 -isForBrowser -prefsLen
255042 dunossauro 20 0 1729M 171M 121M S 7.7 1.1 1:47.34 /usr/lib/Xorg vt2 -displayfd 3 -auth /run/user/1000/gdm/Xauthority -noli
255617 dunossauro 20 0 4585M 679M 319M S 5.8 4.3 1:10.81 obs
255962 dunossauro 20 0 12.5G 769M 301M S 4.5 4.8 1:16.19 /usr/lib/firefox/firefox
264974 dunossauro 20 0 10448 6756 4064 R 3.9 0.0 0:01.92 htop
256020 dunossauro 20 0 12.5G 769M 301M S 3.2 4.8 0:48.92 /usr/lib/firefox/firefox
256175 dunossauro 20 0 2673M 232M 104M S 2.6 1.5 1:01.53 /usr/lib/firefox/firefox -contentproc -childID 4 -isForBrowser -prefsLen
255664 dunossauro 20 0 12.5G 769M 301M S 1.9 4.8 0:25.96 /usr/lib/firefox/firefox
260095 dunossauro 20 0 3376M 701M 211M S 1.9 4.4 1:39.25 /usr/lib/firefox/firefox -contentproc -childID 11 -isForBrowser -prefsLe
255029 dunossauro -21 0 187M 26772 15284 S 1.3 0.2 0:08.19 /usr/bin/pipewire
255858 dunossauro 20 0 42.5G 134M 83236 S 1.3 0.8 0:15.50 /usr/lib/obs-plugins/obs-browser-page --type=renderer --locales-dir-path
255975 dunossauro 20 0 12.5G 769M 301M S 1.3 4.8 0:13.97 /usr/lib/firefox/firefox
255008 dunossauro 9 -11 187M 26772 15284 S 0.6 0.2 0:08.37 /usr/bin/pipewire
255011 dunossauro 9 -11 95236 39908 8944 S 0.6 0.2 0:10.75 /usr/bin/pipewire-pulse
255030 dunossauro -21 0 95236 39908 8944 S 0.6 0.2 0:05.34 /usr/bin/pipewire-pulse
255451 dunossauro 20 0 233M 12496 10584 S 0.6 0.1 0:02.08 i3bar --bar_id=bar-0 --socket=/run/user/1000/i3/ipc-socket.255412
255604 dunossauro 20 0 4585M 679M 319M S 0.6 4.3 0:09.31 obs
255606 dunossauro 20 0 4585M 679M 319M S 0.6 4.3 0:06.09 obs
255667 dunossauro 20 0 12.5G 769M 301M S 0.6 4.8 0:00.67 /usr/lib/firefox/firefox
255669 dunossauro 20 0 12.5G 769M 301M R 0.6 4.8 0:15.47 /usr/lib/firefox/firefox
256018 dunossauro 20 0 12.5G 769M 301M S 0.6 4.8 0:14.21 /usr/lib/firefox/firefox
256114 dunossauro 20 0 12.5G 769M 301M S 0.6 4.8 0:06.15 /usr/lib/firefox/firefox
256478 dunossauro 20 0 3204M 492M 132M S 0.6 3.1 0:05.30 /usr/lib/firefox/firefox -contentproc -childID 9 -isForBrowser -prefsLen
256483 dunossauro 20 0 3204M 492M 132M S 0.6 3.1 0:03.09 /usr/lib/firefox/firefox -contentproc -childID 9 -isForBrowser -prefsLen
260030 dunossauro 20 0 12.5G 769M 301M S 0.6 4.8 0:05.79 /usr/lib/firefox/firefox
260031 dunossauro 20 0 12.5G 769M 301M S 0.6 4.8 0:00.56 /usr/lib/firefox/firefox
260032 dunossauro 20 0 12.5G 769M 301M S 0.6 4.8 0:09.00 /usr/lib/firefox/firefox
260588 dunossauro 20 0 12.5G 769M 301M S 0.6 4.8 0:01.77 /usr/lib/firefox/firefox
260918 dunossauro 20 0 4585M 679M 319M S 0.6 4.3 0:03.89 obs
F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice - F8Nice + F9Kill F10Quit

Uma introdução

Textual

Textual

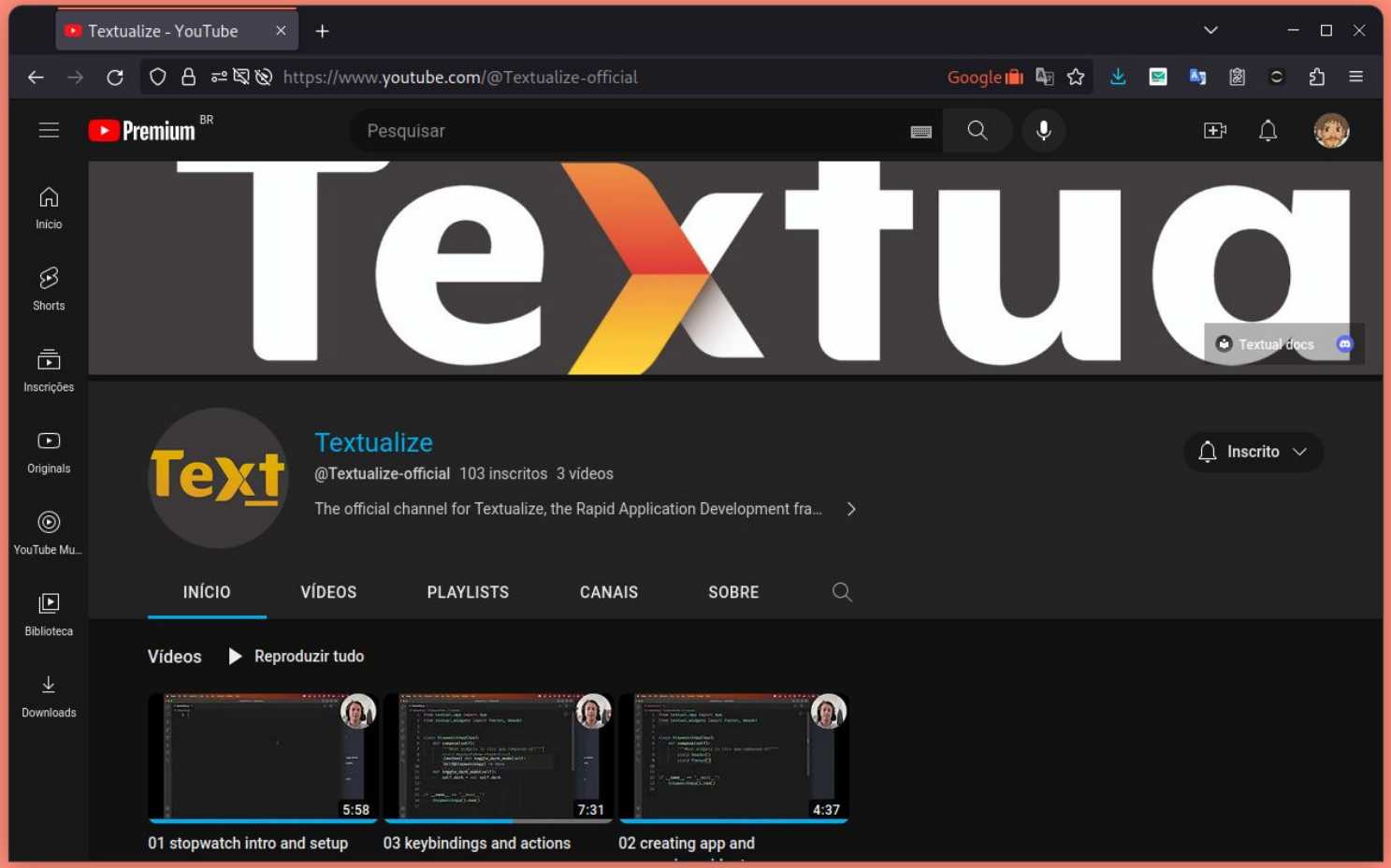
Textual é uma biblioteca para criar TUIs com python.

- Criada por Will McGugan
  - Mesmo criador do Rich (que já vimos na live #224)
- Licença: **MIT**
- Primeira release: 0.1.0 em **06/2021**
- Release atual: **0.9.1**, lançada em 12/22

Tem suporte a [obs: não conseguiremos ver tudo]:

- widgets
- eventos
- interação com mouse e teclado
- asyncio
- CSS
- etc.

Canal oficial!



https://www.youtube.com/@Textualize-official

pip install textual

para o conjunto de ferramentas de desenvolvimento (debugging, logs, live editing)

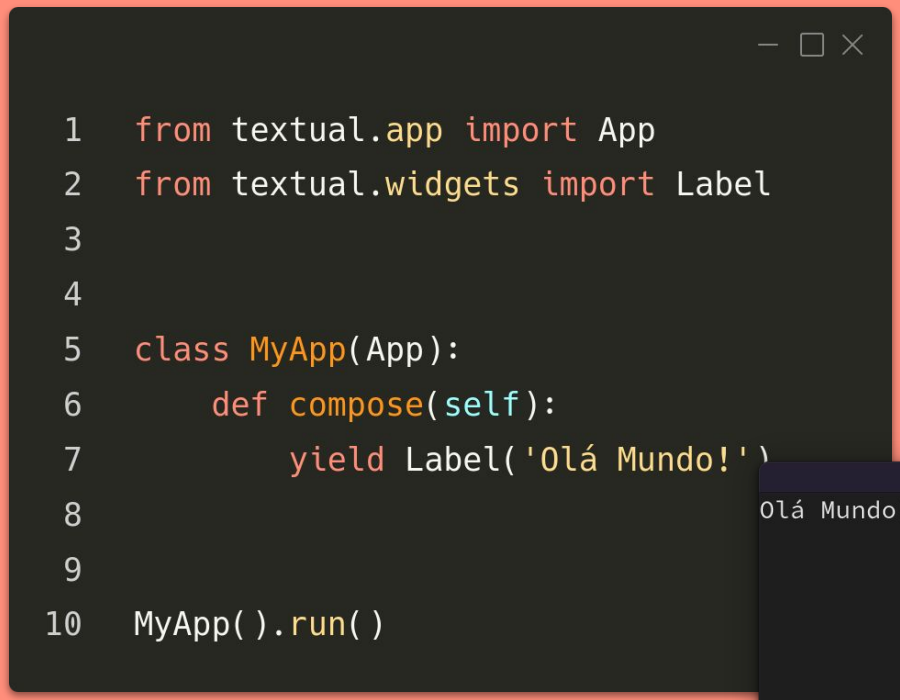
pip install textual[dev]

Para instalar

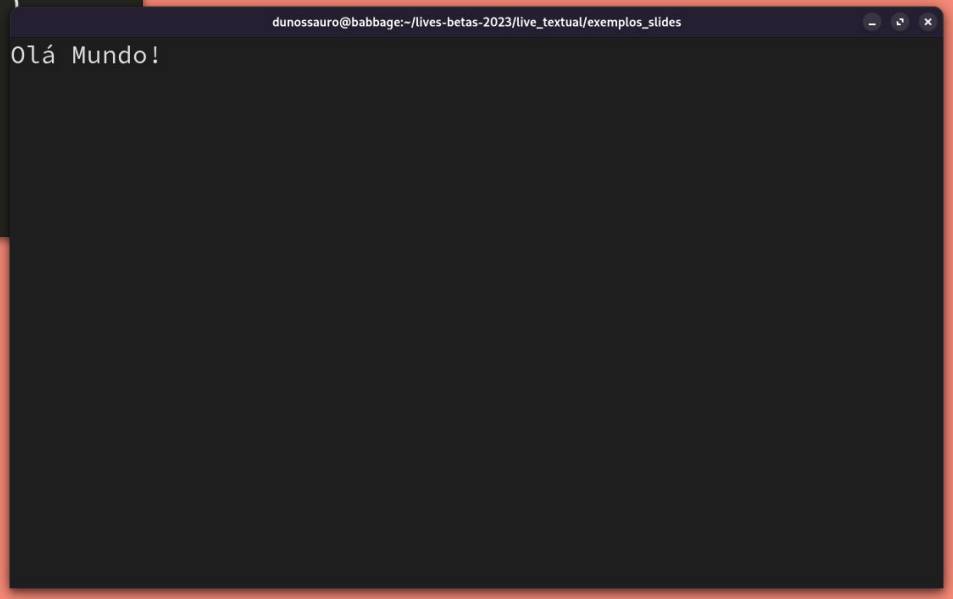
Um olá mundo [exemplo_00.py]

```python
from textual.app import App
from textual.widgets import Label


class MyApp(App):
    def compose(self):
        yield Label('Olá Mundo!')


MyApp().run()
```

Um olá mundo [exemplo_00.py]



```python
from textual.app import App
from textual.widgets import Label


class MyApp(App):
    def compose(self):
        yield Label('Olá Mundo!')


MyApp().run()
```



Anatomia do App [exemplo_00.py]

```python
from textual.app import App
from textual.widgets import Label


class MyApp(App):
    def compose(self):
        yield Label('Olá Mundo!')


MyApp().run()
```

Toda aplicação estende a classe APP

.compose() deve representar um iterável de todos os widgets

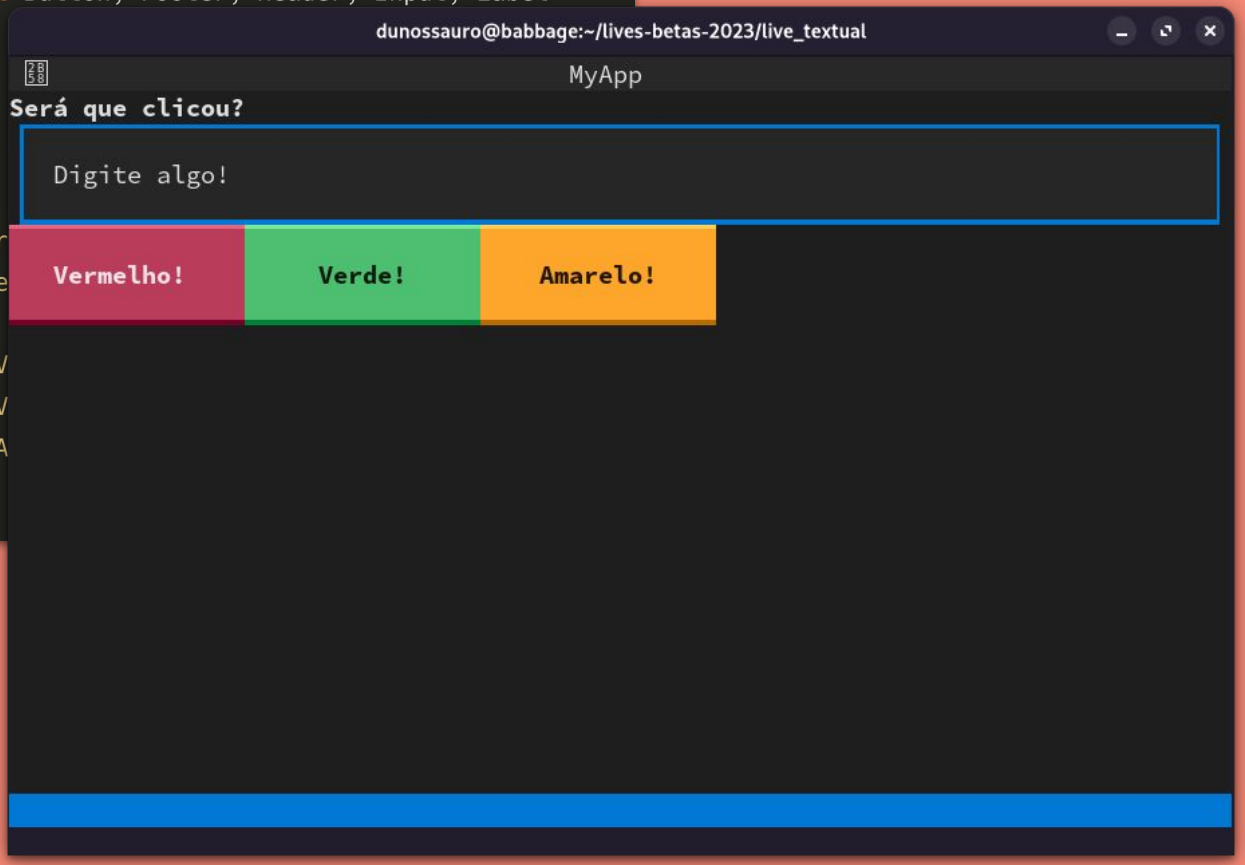
.run() executa o app

O método .compose() [exemplo_01.py]

```python
from textual.app import App
from textual.containers import Horizontal
from textual.widgets import Button, Footer, Header, Input, Label


class MyApp(App):
    def compose(self):
        yield Header()
        yield Label('[b]Será que clicou?[/]')
        yield Input('Digite algo!')
        with Horizontal():
            yield Button('Vermelho!', variant='error')
            yield Button('Verde!', variant='success')
            yield Button('Amarelo!', variant='warning')
        yield Footer()
```

O método .compose() [exemplo_01.py]

```python
from textual.app import App
from textual.containers import Horizontal
from textual.widgets import Button, Footer, Header, Input, Label


class MyApp(App):
    def compose(self):
        yield Header()
        yield Label('[b]Ser
        yield Input('Digite
        with Horizontal():
            yield Button('V
            yield Button('V
            yield Button('A
        yield Footer()
```



Eventos com @on [exemplo_02.py]

O textual provê um decorador chamado on, nele podemos esperar por algum evento. Como um botão ser pressionado ou uma alteração em um input de texto.

```
21      @on(Button.Pressed)
22      def button_event(self, event: Button.Pressed):
23          self.label.update(f'[b]Clicado no {event.button.label}[/]')
24
25      @on(Input.Changed)
26      def input_change(self, event: Input.Changed):
27          self.label.update(f'[b]Texto no Input {event.input.value}[/]')
28
29      @on(Input.Submitted)
30      def input_enter(self, event: Input.Submitted):
31          self.label.update(f'[b red]Texto no Input {event.input.value}[/]')
```

```python
from textual.app import App
from textual.widgets import Header, Footer, Button, Label, Input
from textual.containers import Horizontal
from textual import on


class MyApp(App):
    def compose(self):
        self.label = Label('[b]Será que clicou?[/]')
        yield Header()
        yield self.label
        yield Input('Digite algo!')

        with Horizontal():
            yield Button('Vermelho!', variant='error')
            yield Button('Verde!', variant='success')
            yield Button('Amarelo!', variant='warning')

        yield Footer()

    @on(Button.Pressed)
    def button_event(self, event: Button.Pressed):
        self.label.update(f'[b]Clicado no {event.button.label}[/]')

    @on(Input.Changed)
    def input_change(self, event: Input.Changed):
        self.label.update(f'[b]Texto no Input {event.input.value}[/]')

    @on(Input.Submitted)
    def input_enter(self, event: Input.Submitted):
        self.label.update(f'[b red]Texto no Input {event.input.value}[/]')
```

Mudam o estado de self.label

Eventos on_'event' [exemplo_03.py]

Se soubermos o nome do evento. Como **'Button.Pressed**'. Podemos criar um método no app associado a esse evento com o prefixo **'on_<evento>**'. Evitando o uso do decorador @on

```
20    def on_button_pressed(self, event: Button.Pressed):
21        self.label.update(f'[b]Clicado no {event.button.label}[/]')
22
23    def on_input_changed(self, event: Input.Changed):
24        self.label.update(f'[b]Texto no Input {event.input.value}[/]')
25
26    def on_input_submitted(self, event: Input.Submitted):
27        self.label.update(f'[b red]Texto no Input {event.input.value}[/]')
```

Eventos não relacionados a Widgets [exemplo_03.py]

```python
def on_key(self, event: Key):
    self.log(f'on_key {event.key} foi precionada', event=event)

def on_click(self, event: Click):
    self.log('on_click', event=event)

def on_mouse_scroll_up(self, event: MouseScrollUp):
    self.log('on_mouse_scroll_up', event=event)

def on_mouse_scroll_down(self, event: MouseScrollDown):
    self.log('on_mouse_scroll_down', event=event)
```

textual.textualize.io/events/

textual.textualize.io/events/

textual.textualize.io/widget_gallery/

Referências (abrir links)

Bidings e Actions [exemplo_04.py]

Um dos eventos interessantes são os associados a atalhos de teclado, chamados de Bidings. Nos ajudam a chamar métodos do nosso app!

```
class MyApp(App):
    BINDINGS = [
        ('t', 'change_theme()', 'Muda o tema!'),
        ('s', 'exit()', 'Sai da aplicação!'),
    ]

    def action_change_theme(self):
        self.dark = not self.dark

    def action_exit(self):
        self.exit()
```

CSS [exemplo_05.py]

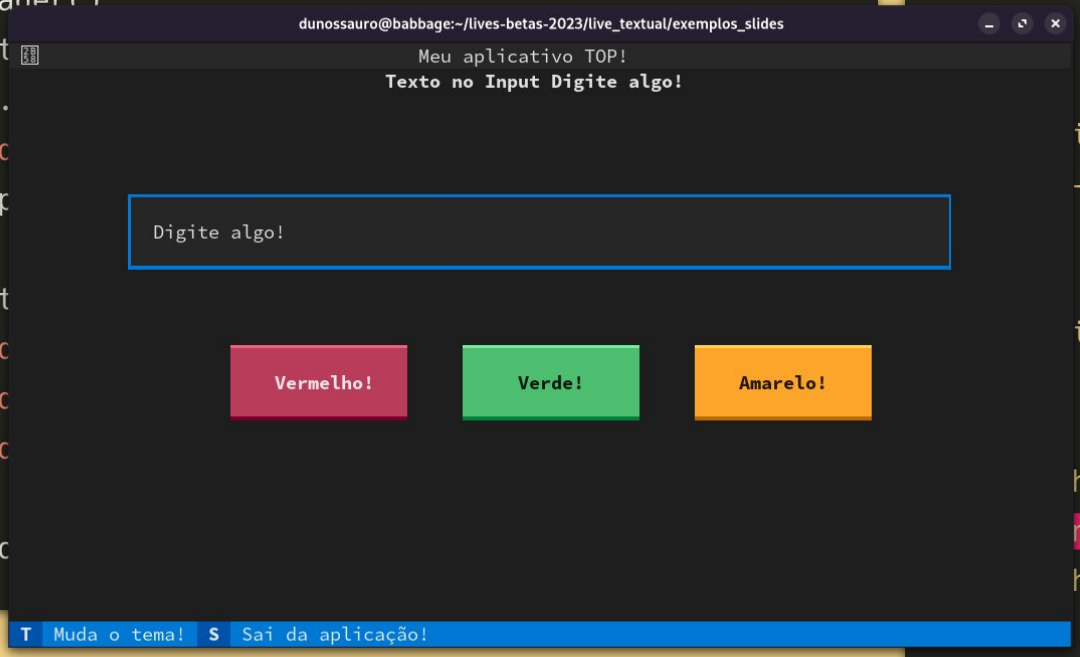
```python
class MyApp(App):
    TITLE = 'Meu aplicativo TOP!'
    CSS_PATH = 'teste.css'

    def compose(self):
        yield Header()
        with Container(classes='label'):
            self.label = Label('[b]Será que clicou?[/]')
            yield self.label
        yield Input('Digite algo!')

        with Container(classes='buttons'):
            yield Button('Vermelho!', variant='error')
            yield Button('Verde!', variant='success')
            yield Button('Amarelo!', variant='warning')

        yield Footer()
```

```css
Sceen {
    align: center middle;
}
.buttons {
    layout: horizontal;
    margin: 2 15;
    align: center middle;
    height: 30%;
    align: center middle;
    max-width: 80%;
}
Button {
    margin: 0 5;
    text-align: center;
}
Input {
    margin: 3 10 0 10;
}
.label {
    height: 10%;
    align: center middle;
    width: 100%;
}
```

CSS [exemplo_05.py]

```python
class MyApp(App):
    TITLE = 'Meu aplicativo TOP!'
    CSS_PATH = 'teste.css'

    def compose(self):
        yield Header()
        with Cont
            self.
            yield
        yield Inp

        with Cont
            yield
            yield
            yield

        yield Foc
```

```
Sceen {
    align: center middle;
}
.buttons {
    layout: horizontal;
    margin: 2 15;
    align: center middle;
    height: 30%;
    align: center middle;
    max-width: 80%;


in: 0 5;
-align: center;


in: 3 10 0 10;


ht: 10%;
n: center middle;
h: 100%;
```



Seletores [exemplo_06.py]

Como usamos CSS, podemos usar os seletores de #id, .classe e Elemento

```python
def compose(self):
    yield Header()
    with Container(classes='label'):
        yield Label('[b]Será que clicou?[/]', id='label')
    yield Input('Digite algo!')
    with Container(classes='buttons'):
        yield Button('Vermelho!', variant='error')
        yield Button('Verde!', variant='success')
        yield Button('Amarelo!', variant='warning')
    yield Footer()

def action_show_all_buttons(self):
    for element in self.query('Button'):
        self.log('Botão', element.label)

def on_button_pressed(self, event: Button.Pressed):
    self.query_one('#id').update(f'[b]Clicado no {event.button.label}[/]')
```

Debug

Para fazer o debug de uma aplicação textual você precisa de algumas coisas:

- textual[dev] instalado
- Dois terminais
- Chamar a aplicação usando o CLI do Textual

```
# Para rodar a aplicação em debug
textual run <arquivo>.py --dev
```

```
# Para subir o servidor de debug
textual console
```

Bora codar uma coisinha?

[caso não de tempo, o código que iriamos fazer é um leitor de rss.
Subirá como feed.py]

Se der tempo

picpay.me/dunossauro

apoia.se/livedepython

pix.dunossauro@gmail.com

Ajude o projeto <3