

O que é?

Dash

Dash



Dash é uma ferramenta para criação de gráficos interativos na web. Ela não requer conhecimento em WEB, HTML, CSS ou Javascript. A proposta é se apoiar totalmente em Python.

- MIT
- Flask, React.js, plotly.js
- Versão atual: 1.19.0
- Primeira versão: Set/2017

`pip install dash`



Como instalar?



1. Modo lento
2. Sem bases
3. Estamos pelos gráficos e uso da ferramenta



Disclaimers



Nosso primeiro código



```
1  from dash import Dash
2  from dash_html_components import H1
3
4  app = Dash(__name__)
5
6  app.layout = H1('Boas vindas')
7
8  app.run_server()
```

Nosso primeiro código



Flask APP

```
1 from dash import Dash
2 from dash_html_components import H1
3
4 app = Dash(__name__)
5
6 app.layout = H1('Boas vindas')
7
8 app.run_server()
```

Servidor na porta:
8050

Nosso primeiro código

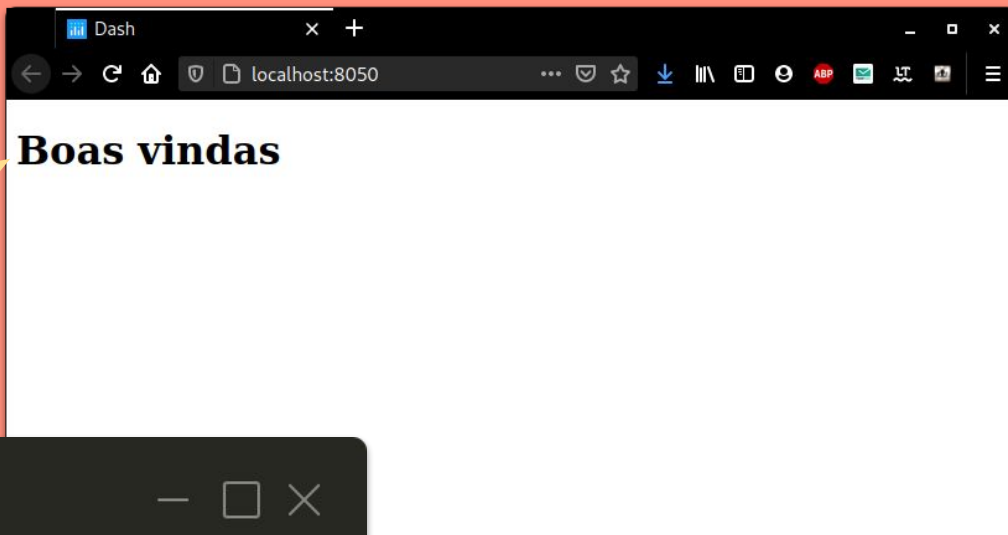


Layout do nosso
dashboard

```
from dash import Dash
from dash_html_components import H1
3
4 app = Dash(__name__)
5
6 app.layout = H1('Boas vindas')
7
8 app.run_server()
```



```
1 from dash import Dash
2 from dash_html_components import H1
3
4 app = Dash(__name__)
5
6 app.layout = H1('Boas vindas')
7
8 app.run_server()
```



```
1 python exemplo.py
```


ou layouts

DHC

Dash HTML components



Um dos grandes trunfos do Dash é não exigir que você use arquivos HTML. Todo o código é feito em python e com isso o dash trás a biblioteca de componentes HTML

- A
- Div
- p
- H1/H2/H3 ...
- Nav
- Meta

Como vimos no primeiro exemplo



```
1  from dash import Dash
2  from dash_html_components import H1
3
4  app = Dash(__name__)
5
6  app.layout = H1('Boas vindas')
7
8  app.run_server()
```

O layout tem que ser definido por um componente HTML

Mas como colocar mais de um componente?

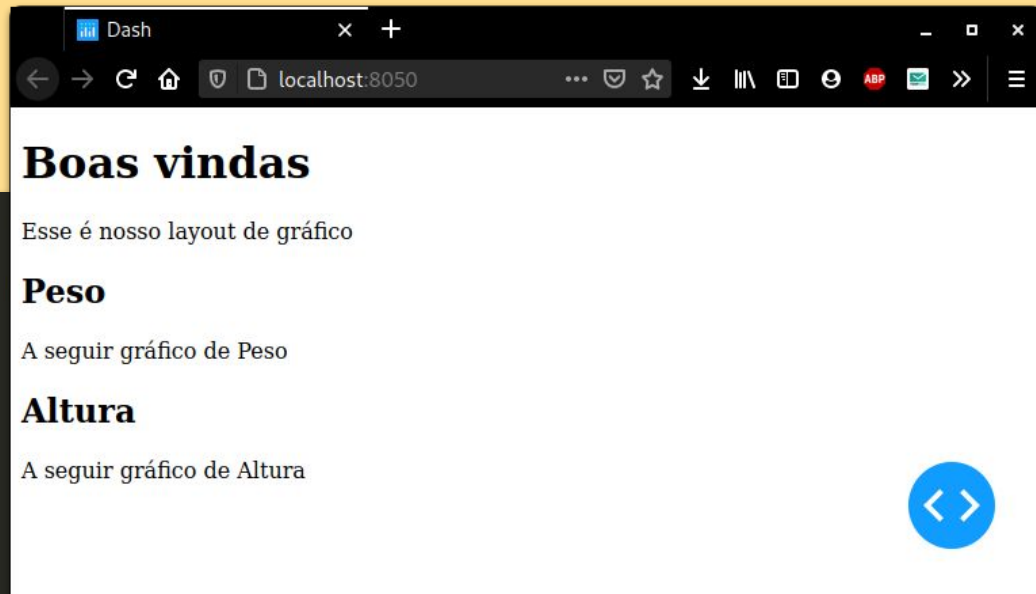


```
1  from dash_html_components import H1, Div, P, H2
2
3  app.layout = Div(
4      children=[
5          H1('Boas vindas'),
6          P('Esse é nosso layout de gráfico'),
7          H2('Peso'),
8          P('A seguir gráfico de Peso'),
9          H2('Altura'),
10         P('A seguir gráfico de Altura'),
11     ]
12 )
```

Como o layout todo é composto por um único componente, devemos usar Div e adicionar os outros componentes internamente a ela.

E com que cara isso fica?

```
1 from dash_html_components import H1, Div, P,  
2  
3 app.layout = Div(  
4     children=[  
5         H1('Boas vindas'),  
6         P('Esse é nosso layout de gráfico'),  
7         H2('Peso'),  
8         P('A seguir gráfico de Peso'),  
9         H2('Altura'),  
10        P('A seguir gráfico de Altura'),  
11    ]  
12 )
```



Dá pra dar um tapa nesse HTML?



Dentro de cada componente você pode usar a Tag style, como faria em HTML, ou você deve criar um arquivo CSS pra te ajudar nessa missão.

O Dash não abstrai CSS

0 CSS



Existem duas formas de usar CSS no Dash. Uma delas é usando um arquivo CSS local:

Você deve adicionar ele em **/assets/style.css**

Outra forma é usar links de estilo externos:

```
app = Dash(__name__, external_stylesheets=external_stylesheets)
```

Onde passamos uma lista de casos

Mostra pra galera como fica o estilo



Não esquecer



Componentes do
Core do Dash

DCC

Agora sim



Os core components são os componentes base para interação.

- Um campo de texto
- Um campo de seleção
- Um campo de escolha.

Em grande maioria os componentes do core são a abstração dos inputs do HTML.

Gráficos



Mas, contudo, porém, entretanto. Os gráficos básicos também estão no DCC(Dash Core Components).

A regra dos gráficos



Existem duas maneiras de gerar gráficos com Dash, uma é usando o plotly (python), outra usando o plotly.js.

Vamos usar somente a forma do plotly.js



```
1  from dash_core_components import Graph
2
3  Graph(
4      figure={
5          'data': [
6              {},
7          ],
8          'layout': {
9              }
10     }
11 )
```

Cada dicionário em
'data' é referente a
um valor no plot

```
from dash_core_components import Graph

Graph(
    figure={
        'data': [
            {},
        ],
        'layout': {
        }
    }
)
```

O layout é referente a tudo que não é dado, cores, título, e etc...

```
from dash_core_components import Graph  
graph(  
    figure={  
        'data': [  
            {},  
        ],  
        'layout': {  
        }  
    }  
)
```

```
1  from dash_core_components import Graph
2
3  Graph(
4      figure={
5          'data': [
6              {'x': [0, 25, 50, 75, 100], 'type': 'box'},
7          ],
8          'layout': {
9              }
10     }
11 )
```


Aqui temos um
boxplot na horizontal
(x)

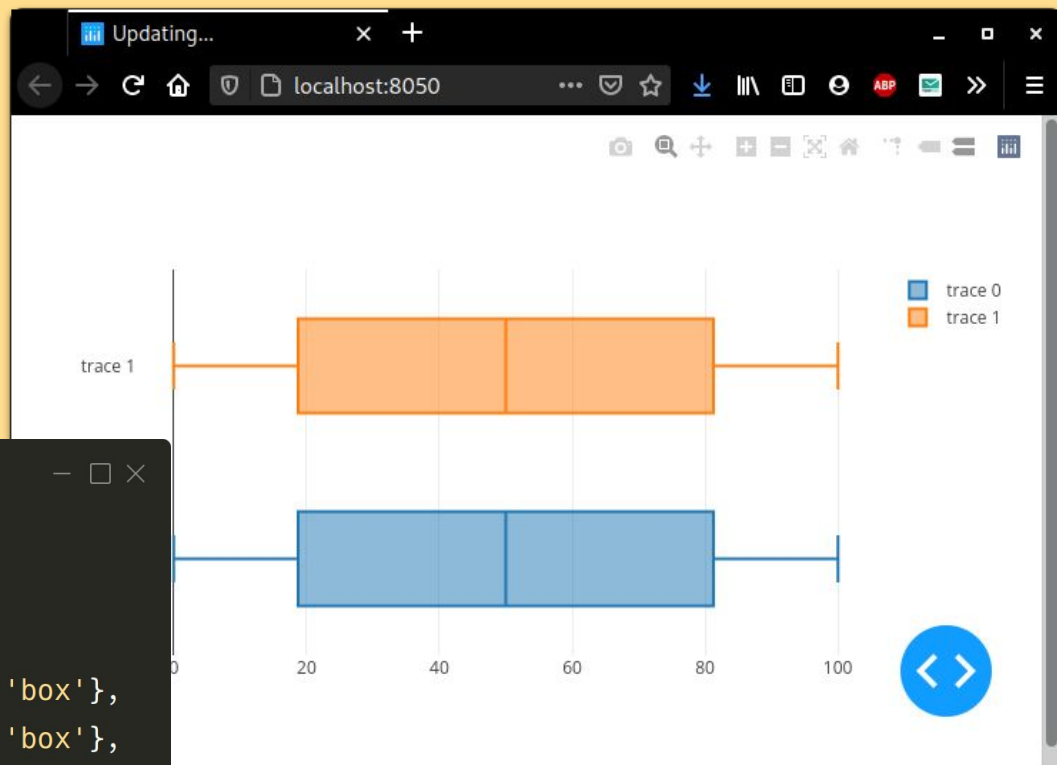
```
core_components import Graph

3  G
4  re={
5  'data': [
6      {'x': [0, 25, 50, 75, 100], 'type': 'box'},
7  ],
8  'layout': {
9  }
10 }
11 )
```

```

1  Graph(
2    figure={
3      'data': [
4        {'x': [0, 25, 50, 75, 100], 'type': 'box'},
5        {'x': [0, 25, 50, 75, 100], 'type': 'box'},
6      ],
7      'layout': {
8      }
9    }
10 )

```



Gráficos



Seguindo esse padrão simples, podemos gerar:

- histogramas
- linhas
- barras
- boxplot
- pizza
-

Vamos gerar algumas coisinhas aqui
<3



Let's Bora



DCC

Sobre os
componentes que
não falamos.



Dentro do DCC, temos todos os componentes de interatividade.

- Botões
- Dropdowns
- Slides
- etc...

Dropdown



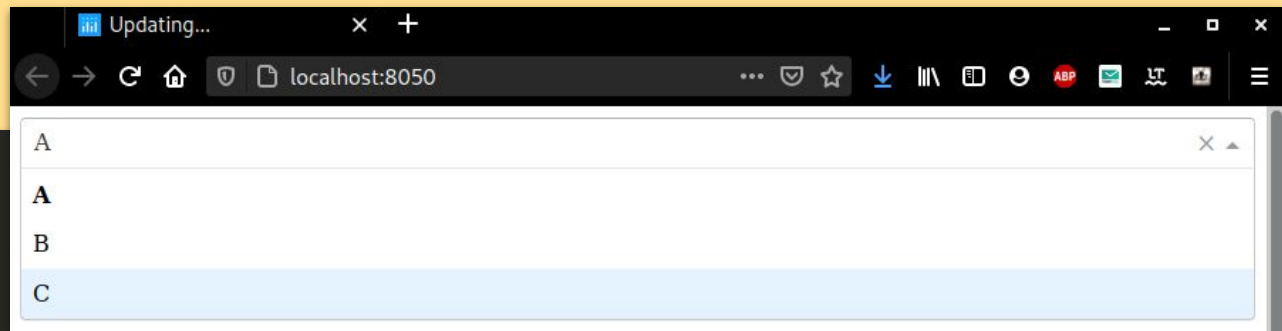
```
1 Dropdown(  
2     options=[  
3         {'label': 'A', 'value': 'a'},  
4         {'label': 'B', 'value': 'b'},  
5         {'label': 'C', 'value': 'c'},  
6     ],  
7     value='a'  
8 )
```

Abre um menu com as 3 opções

Dropdown



```
1 Dropdown(  
2     options=[  
3         {'label': 'A', 'value': 'a'},  
4         {'label': 'B', 'value': 'b'},  
5         {'label': 'C', 'value': 'c'},  
6     ],  
7     value='a'  
8 )
```



Bora?



Exemplificar mais alguns



Tornando as
coisas interativas

Call
backs

Callbacks



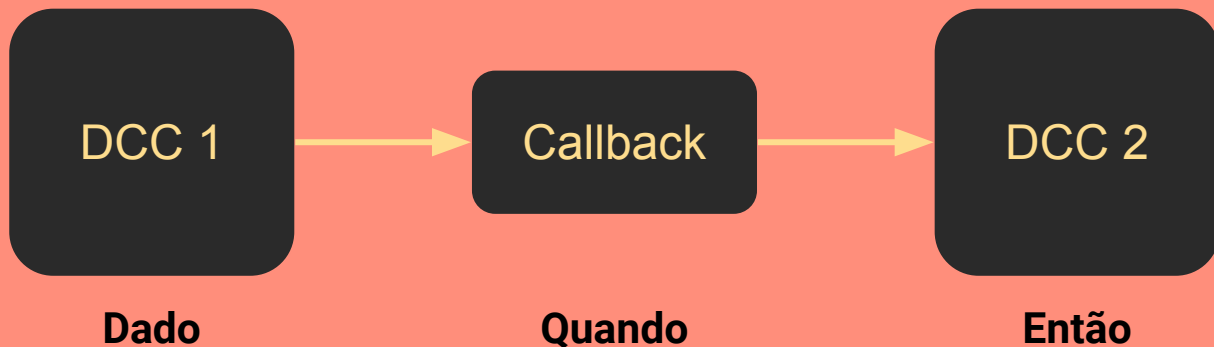
São formas de conectar 2 componentes de DCC de maneira reativa.



Callbacks



São formas de conectar 2 componentes de DCC de maneira reativa.



Callbacks



São formas de conectar 2 componentes de DCC de maneira reativa.



Callbacks




São formas de conectar 2 componentes de DCC de maneira reativa.





```
1 from dash_html_components import Div, P
2 from dash_core_components import Input as DCCInput
3 from dash.dependencies import Input, Output
4
5 app.layout = Div(
6     children=[
7         DCCInput(id='a_input'),
8         P(id='a_output'),
9     ]
10 )
11
12
13 @app.callback(
14     Output('a_output', 'children'),
15     Input('a_input', 'value')
16 )
17 def my_callback(input_value):
18     print(f'callback: {input_value=}')
19     return input_value
```

Basicamente,

 todas as vezes que o **valor** de **DCCInput** for alterado




```
1 from dash_html_components import Div, P
2 from dash_core_components import Input as DCCInput
3 from dash.dependencies import Input, Output
4
5 app.layout = Div(
6     children=[
7         DCCInput(id='a_input'),
8         P(id='a_output'),
9     ]
10 )
11
12
13 @app.callback(
14     Output('a_output', 'children'),
15     Input('a_input', 'value')
16 )
17 def my_callback(input_value):
18     print(f'callback: {input_value=}')
19     return input_value
```

Basicamente,

-  todas as vezes que o **valor** de **DCCInput** for alterado
-  O **children** de P, sofrerá a ação do callback


```
1 from dash_html_components import Div, P
2 from dash_core_components import Input as DCCInput
3 from dash.dependencies import Input, Output
4
5 app.layout = Div(
6     children=[
7         DCCInput(id='a_input'),
8         P(id='a_output'),
9     ]
10 )
11
12
13 @app.callback(
14     Output('a_output', 'children'),
15     Input('a_input', 'value')
16 )
17 def my_callback(input_value):
18     print(f'callback: {input_value}')
19     return input_value
```

Basicamente,

-  todas as vezes que o **valor** de **DDCInput** for alterado
-  O **children** de P, sofrerá a ação do callback
-  E receberá o valor de retorno da função **my_callback**

```

1 from dash_html_components import Div, P
2 from dash_core_components import Input as DCCInput
3 from dash.dependencies import Input, Output
4
5 app.layout = Div(
6     children=[
7         DCCInput(id='a_input'),
8         P(id='a_output'),
9     ]
10 )
11
12
13 @app.callback(
14     Output('a_output', 'children'),
15     Input('a_input', 'value')
16 )
17 def my_callback(input_value):
18     print(f'callback: {input_value=}')
19     return input_value

```

DCC 1

id='a_input'

DCC 2

id='a_output'

Callback

Callbacks



Tempo real



Pq sim



Do que não falamos?



- DAQ
 - Ferramentas específicas para dashboard / dinheiro etc..
 - Valeu uma outra live só para isso
- Bio
 - Plots para biologia
- Images
- Tables
- Bootstrap
 - <https://dash-bootstrap-components.opensource.faculty.ai/>