BEN BUCHANAN

MASTODON.SOCIAL/@2000K

REACT IN PYTHON

YOU ARE PROBABLY WONDERING

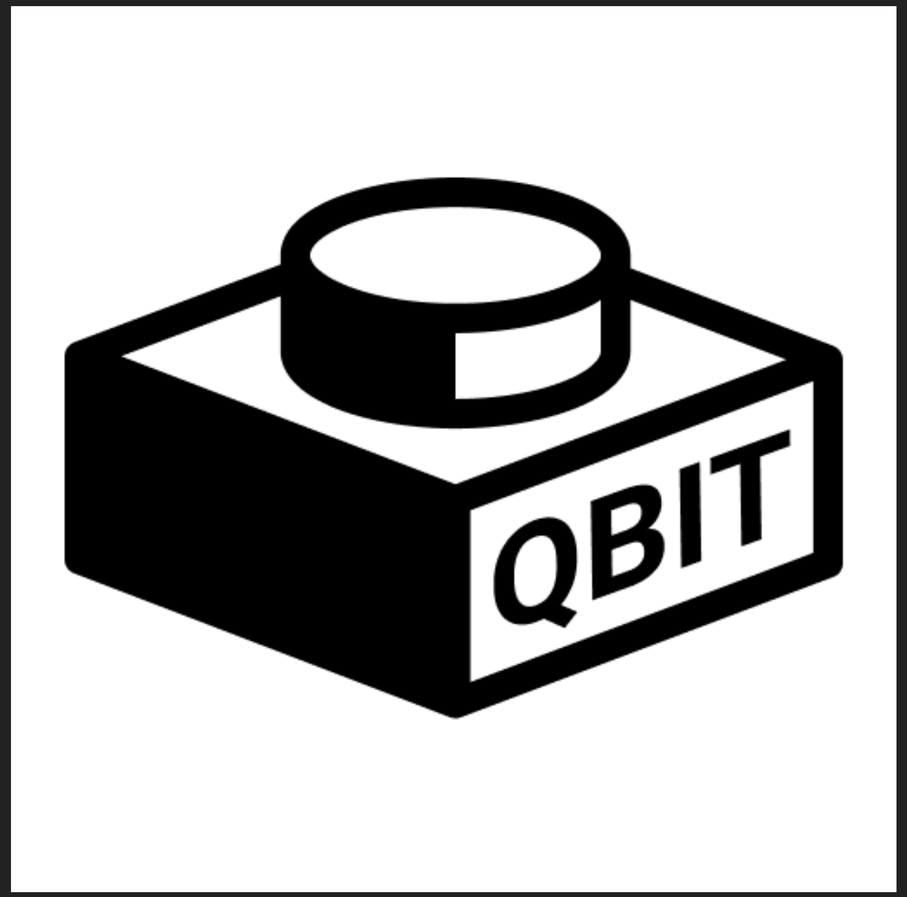
WHY?

WHY NOT? I'VE DONE UI WORK WITH LOTS OF STUFF...

- Perl (it was The Year 2000™)
- PHP (multiple)
- Lotus Notes (the actual worst)
- TCL (Vignette)
- Java with JSP
- Java with Soy
- Classic ASP/VBScript
- .NET
- Python (Dash)
- Vanilla JS
- jQuery
- Backbone (mercifully briefly)
- Pug (f.k.a. Jade)
- Angular
- React
- No JS (skkkrRRRR crowd gasps)
- Plus countless proprietary systems with custom templating solutions accepting raw HTML and CSS, and *sometimes* JS
- Plus more I forgot

BUT REALLY, HOW I ENDED UP WORKING WITH DASH

- Analysts like Python
- Analysts do not like JavaScript
- Analysts wanted Python dashboards styled like our React apps (which use our React UI library "Qbit")
- I had interns who needed a project



I GUESS I'M LEARNING PYTHON

¯_(ツ)_/¯

me, circa December 2019

TURNS OUT PYTHON IS NICE

- Easy to learn
- Extremely popular
- Massive ecosystem

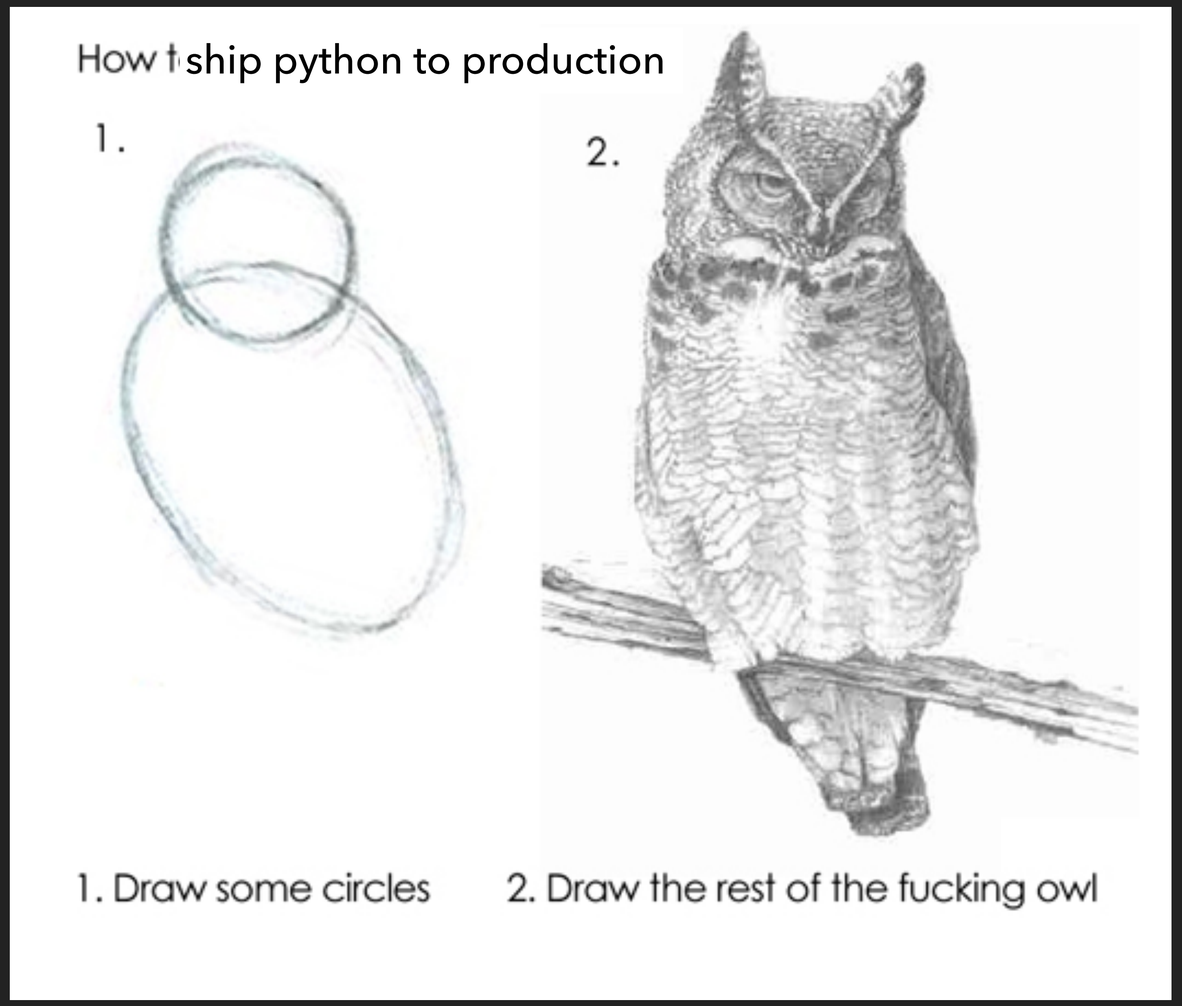
TIOBE (the software quality company) Schedule a demo

Dec 2023	Dec 2022	Change	Programming Language	Ratings	Change
1	1		Python	13.86%	-2.80%
2	2		C	11.44%	-5.12%
3	3		C++	10.01%	-1.92%
4	4		Java	7.99%	-3.83%
5	5		C#	7.30%	+2.38%
6	7	^	JavaScript	2.90%	-0.30%
7	10	^	PHP	2.01%	+0.39%
8	6	v	Visual Basic	1.82%	-2.12%
9	8	v	SQL	1.61%	-0.61%
10	9	v	Assembly language	1.11%	-0.76%

tiobe.com/tiobe-index

STEP THREE, PROFIT?

- Python, the language:
  thousands of tutorials, books, etc
- Python, for programming beginners:
  thousands of tutorials, books, etc
- Python, for experienced devs:
  crickets, shrugs, etc



I WROTE DOWN WHAT WORKED FOR ME

- pyenv, pip, virtual environments, etc all map to NodeJS equivalents pretty easily
- You might save some time reading that
- Or read the Hitchhiker's Guide To Python

weblog.200ok.com.au

OK THAT'S PYTHON

WHAT'S DASH?

DASH IS THE ORIGINAL LOW-CODE FRAMEWORK FOR RAPIDLY BUILDING DATA APPS IN PYTHON.

Plotly (Dash creators)

DASH.PLOTLY.COM

- Dash is a Python web framework by Plotly
- You write Python; you get a Flask app with a React UI and a Python backend
- You can use components built in both Python and React

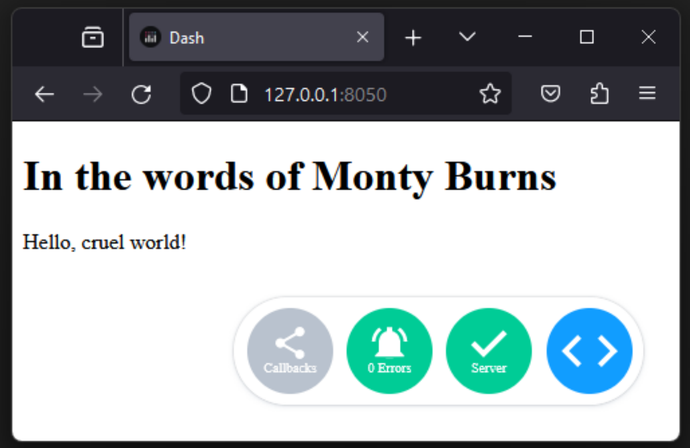
DASH SETUP IS AKSHULY EASY

- Dash In 20 Minutes really does take 20 minutes
- It's basically...

```
pip install dash
touch app.py
(minimal code - see screenshot)
python app.py
open http://127.0.0.1:8050/
```

app.py

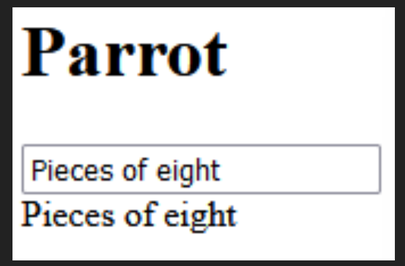
```python
from dash import Dash, html

app = Dash(__name__)

app.layout = html.Div([
    html.H1("In the words of Monty Burns"),
    html.P("Hello, cruel world!")
])

if __name__ == '__main__':
    app.run(debug=True)
```

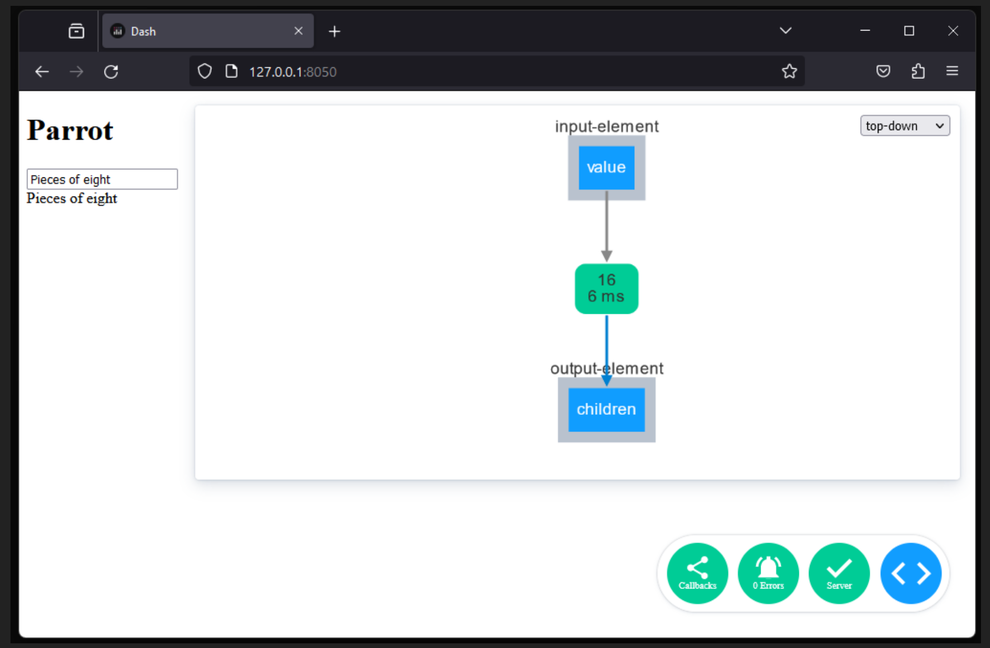


CALLBACKS

- Dash uses a curious mix of reactive state and DOM style callbacks
- But it's cool, you can still repeat a text input like all the SPA cool kids
  (to really crush it, do `"str"[::-1]`)

```python
@app.callback(
    Output(component_id='output-element', component_property='children'),
    [Input(component_id='input-element', component_property='value')]
)
def dostuff(str):
    return f"{format(str)}"

app.layout = html.Div([
    html.H1("Parrot"),
    dcc.Input(id="input-element", value="", type="text"),
    html.Div(id="output-element")
])
```





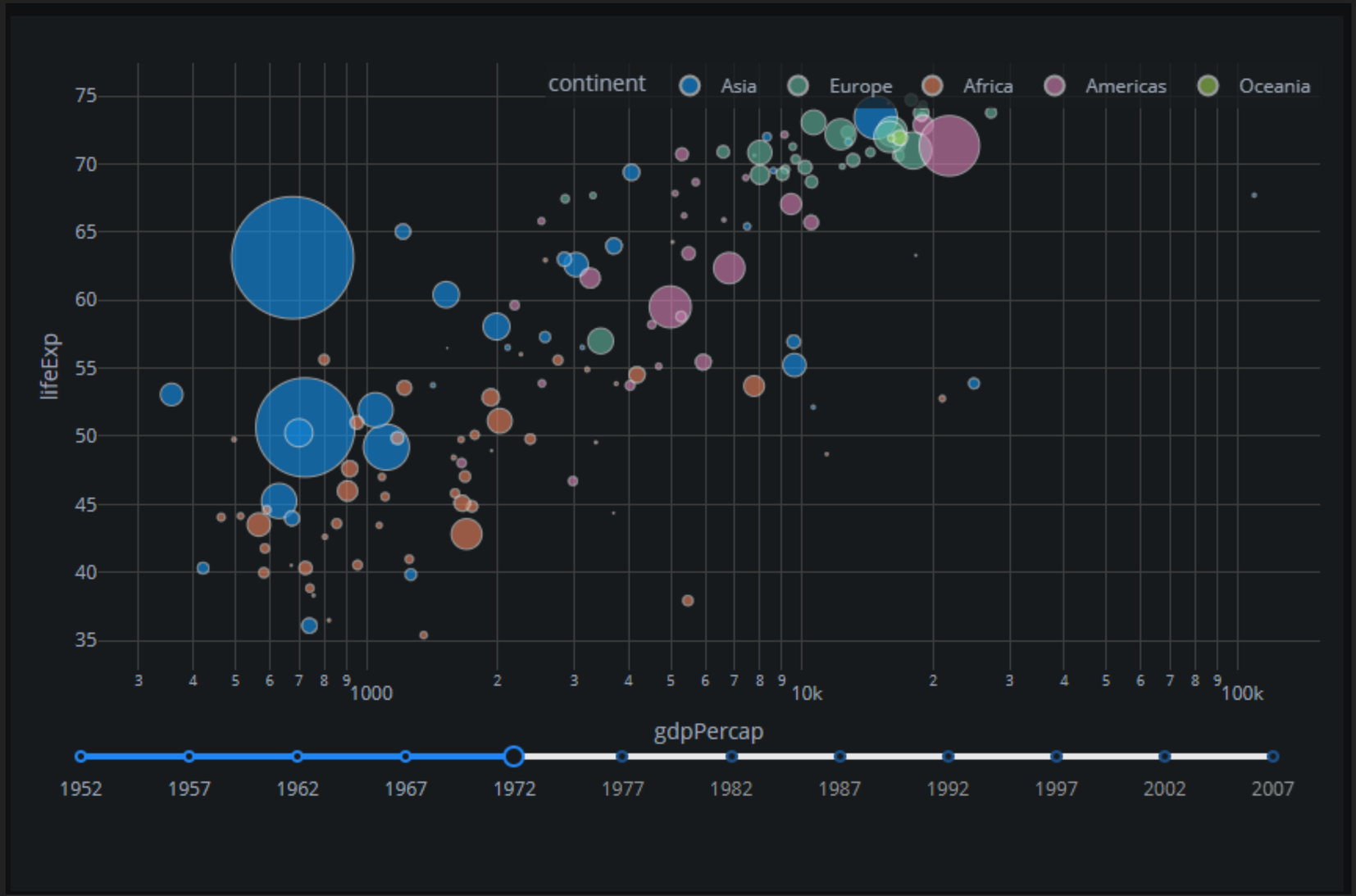
ALSO, PYTHON

- Plus whatever you want to do in Python
- Which is, after all, why we were here in the first place! ~~Pandas~~ Python.

```python
from dash import Dash, html, dcc, callback, Output, Input
import plotly.express as px
import pandas as pd

df = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/

app = Dash(__name__)

app.layout = html.Div([
    html.H1(children='Title of Dash App', style={'textAlign':'center'}),
    dcc.Dropdown(df.country.unique(), 'Canada', id='dropdown-selection'),
    dcc.Graph(id='graph-content')
])

@callback(
    Output('graph-content', 'figure'),
    Input('dropdown-selection', 'value')
)
def update_graph(value):
    dff = df[df.country==value]
    return px.line(dff, x='year', y='pop')

if __name__ == '__main__':
    app.run(debug=True)
```

ALSO, DATAVIZ

- Dash was basically built around Plotly, so it's good for dataviz.



EVERYONE GETS SOMETHING OUT OF IT

- Analysts build apps with minimal training
- Designers built a prototype in a brief workshop
- A sysadmin friend picked it up in a day
- Great way for frontenders to learn Python

BUT I WANNA USE REACT

REACT → DASH

PORTING YOUR REACT TO DASH

- Create a React wrapper to..
  1. Import your component
  2. Handle some Dash props
  3. Define PropTypes (required in Dash)
- Update manifest to locate the wrapper
- Package as PyPi
- Ready to import into Dash application

```
import { React, Component } from "react";
import { PropTypes } from "prop-types";
import { Card } from "@ns/your-lib";

export default class card extends Component {
  render() {
    const {
      children,
      setProps,
      ...other_props
    } = this.props;
    return (
      <Card
        data-theme={data_theme}
        {...other_props}
      >{children}</Card>
    );
  }
}

card.defaultProps = {};

card.propTypes = {
  variant: PropTypes.oneOf([
    "default",
    "primary",
    "secondary"
  ])
};
```

RELAX EVERYONE, ITS OUTPUT IS JUST AS UGLY AS REACT

REACT

```
1  import React from 'react';
2  import classNames from 'classnames';
3  import { Module } from '../module';
4  import '@quantium-enterprise/qbit-core/dist/components/card.css';
5  export var CardVariant;
6  (function (CardVariant) {
7      CardVariant["Default"] = "default";
8      CardVariant["Insight"] = "insight";
9      CardVariant["Primary"] = "primary";
10     CardVariant["Secondary"] = "secondary";
11 })(CardVariant || (CardVariant = {}));
12 export var CardSentiment;
13 (function (CardSentiment) {
14     CardSentiment["Bad"] = "bad";
15     CardSentiment["Good"] = "good";
16     CardSentiment["Neutral"] = "neutral";
17     CardSentiment["Warning"] = "warning";
18 })(CardSentiment || (CardSentiment = {}));
19 export var CardSpacing;
20 (function (CardSpacing) {
21     CardSpacing["Medium"] = "medium";
22     CardSpacing["Large"] = "large";
23 })(CardSpacing || (CardSpacing = {}));
24 export function Card({ className, header, children, footer, sentiment,
     spacing = CardSpacing.Large, variant, ...props }) {
25     const classes = classNames('q-card', sentiment && `q-card-sentiment-
         ${sentiment}`, spacing && `q-card-spacing-${spacing}`, variant &&
         `q-card-${variant}`, className);
26     return (React.createElement(Module, { className: classes, header:
         header, content: children, footer: footer, ...props }));
27 }
28 /* EXPORTS FOR THE EXPORT GOD, DEFAULTS FOR THE DEFAULT THRONE
29 ———————————————————————————————————————————————— */
30 export default Card;
31 //# sourceMappingURL=index.js.map
```

DASH

```
28 - sentiment (a value equal to: "bad", "good", "neutral", "warning"; optional)
29
30 - spacing (a value equal to: "medium", "large"; optional)
31
32 - style (dict; optional)
33
34 - variant (a value equal to: "default", "insight", "primary", "secondary"; optional)"""
35     _children_props = ['header', 'footer']
36     _base_nodes = ['header', 'footer', 'children']
37     _namespace = 'qbit_dash'
38     _type = 'card'
39     @_explicitize_args
40     def __init__(self, children=None, sentiment=Component.UNDEFINED, variant=Component.
       UNDEFINED, spacing=Component.UNDEFINED, header=Component.UNDEFINED, footer=Component.
       UNDEFINED, id=Component.UNDEFINED, className=Component.UNDEFINED, data_qtheme=Component.
       UNDEFINED, ref=Component.UNDEFINED, role=Component.UNDEFINED, style=Component.UNDEFINED,
       **kwargs):
41         self._prop_names = ['children', 'id', 'className', 'data_qtheme', 'footer',
           'header', 'ref', 'role', 'sentiment', 'spacing', 'style', 'variant']
42         self._valid_wildcard_attributes =            []
43         self.available_properties = ['children', 'id', 'className', 'data_qtheme', 'footer',
           'header', 'ref', 'role', 'sentiment', 'spacing', 'style', 'variant']
44         self.available_wildcard_properties =            []
45         _explicit_args = kwargs.pop('_explicit_args')
46         _locals = locals()
47         _locals.update(kwargs)  # For wildcard attrs and excess named props
48         args = {k: _locals[k] for k in _explicit_args if k != 'children'}
49
50         super(card, self).__init__(children=children, **args)
51
```

NOW YOU CAN USE YOUR REACT IN DASH

- Dash props map to your React props
- Events map through as well
- You might miss JSX syntax, but people who have never used JSX don't

```python
import your_package as ns
from dash import html
from app import app

contents = [

    ns.card([
        html.P("Some content set with arg")
    ]),

    ns.card(children=[
        html.P("Some content set with kwarg")
    ]),

]

```

Some content set with arg

AUTOMATION

- We automated the conversion from Typescript to PropTypes, using ts-morph to walk the AST
- We also extract lists of components and props into JSON to generate tests and docs


```
9   html.Div([
10    ns.card(
11      variant=variant.lower(),
12      children=[
13        html.P(f"{variant} Content")
14      ]
15    )
16  ])
17  for variant in card_variants
```

SO...

WHAT DID WE LEARN?

OMG DOES THIS GUY REALISE HE'S SPEAKING AT SYDJS? YOU KNOW, SYDNEY JAVASCRIPT?

THE INCREDIBLY SUBTLE SUBTEXT OF THIS TALK

IF YOU HAVE NOT LEARNED A LANGUAGE OTHER THAN JAVASCRIPT, YOU SHOULD.

NICE REASONS TO LEARN ANOTHER LANGUAGE

- Realise that you can do it
  I have known far too many frontenders who didn't believe they could.
- Learn another ecosystem
- Experience another community
- Collaborate with new people
- Add a big new tool to the toolkit

SPICY REASONS TO LEARN ANOTHER LANGUAGE

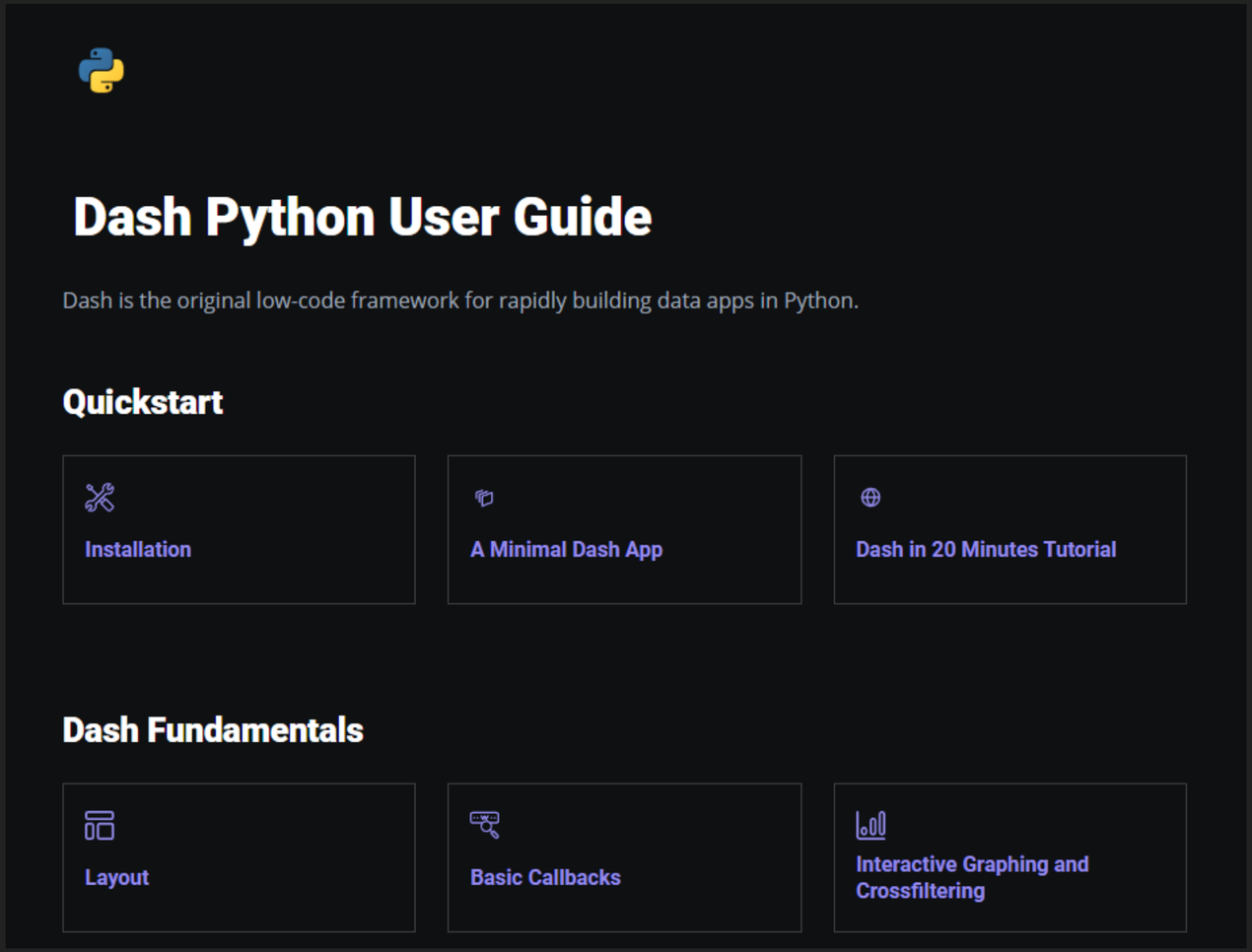
- Ha! Ha! I'm Full Stack now!
- Improved bullshit detector
- You might change how you work

TO BE REALLY CLEAR

I AM NOT SAYING "STOP USING JS". JUST TRY SOMETHING ELSE AS WELL!

FOR DASH SPECIFICALLY

- It's a really interesting way to extend the reach of your React, into Python
- There are also R and Julia versions
- Give it a try!



THANKS!

MASTODON.SOCIAL/@200OK
THAT'S TWO HUNDRED OK, LIKE THE HTTP STATUS

THAT WAS: REACT IN PYTHON