Easy Interactive Data Applications with Dash
Matteo Guzzo
EuroPython 2018, Edinburgh
whoami

Matteo Guzzo, PhD

Data Science Consultant @ Berlin, Germany

Find me @:

- guzzo.matteo@gmail.com
- Twitter: @_teoguso
- GitHub/GitLab: teoguso
- LinkedIn: matteoguzzo
My first dataviz stack

(please don't judge)

1. Bash, awk, sed, grep, etc
2. Gnuplot
My current dataviz stack

1. Jupyter notebooks + pandas
2. Matplotlib (sometimes Seaborn)
Just another day in the life of a data scientist...
Just another day in the life of a data scientist...

- You have done the greatest work of your life and you can't wait to show it to the world
  (did anyone say reporting?)
Just another day in the life of a data scientist...

- You have done the greatest work of your life and you can't wait to show it to the world
  (did anyone say *reporting*?)

- **Problem 1**: the world (*aka* your boss/client) only uses windows and has no idea what python or jupyter are.
Just another day in the life of a data scientist...

- You have done the greatest work of your life and you can't wait to show it to the world
  (did anyone say *reporting*)

- **Problem 1**: the world (*aka* your boss/client) only uses windows and has no idea what python or jupyter are.

- **Problem 2** (*better?): they ask you to change small things all the time (e.g. axes limits)
Just another day in the life of a data scientist...

- You have done the greatest work of your life and you can't wait to show it to the world (did anyone say *reporting*?)

- **Problem 1**: the world (*aka* your boss/client) only uses windows and has no idea what python or jupyter are.

- **Problem 2** (*better?):* they ask you to change small things all the time (e.g. axes limits)

- **Problem 3** (*best?):* they want to play around with the visualization themselves *aka* "Could you do it in Excel?"
Or...
You just want to show off
You just want to show off

(that's fine too)
Enter Plotly.py
Enter Plotly.py

- Python API for plotly.js
- Open source
- Interactive!
- Works well with Jupyter notebooks
**Plotly.py**

`plotly.graph_objects` contains the main components of a plot:

- **Figure** contains all info for the visualization (data and layout)
  - Layout contains all info for styling
  - Scatter, Bar, Heatmap, etc, express different type of graphs.

**NOTE:** These objects can always be swapped with python dicts
Plotly.py

Minimal plotly example:

```python
import plotly.graph_objs as go
go.FigureWidget(data=[dict(x=[0,1,2], y=[3,4,2])])
```
Dash by Plotly
Dash by Plotly

Dash is a Python framework for building analytical web applications. No JavaScript required.

Built on top of Plotly.js, React, and Flask, Dash ties modern UI elements like dropdowns, sliders, and graphs to your analytical Python code.

Dash
Dash

- Frontend: JS (Plotly, React)
Dash

- Frontend: JS (Plotly, React)
- Backend: Flask
Dash

- Frontend: JS (Plotly, React)

- Backend: Flask

- You don't need to know any of that! (sort of...)
Minimal example

```python
import dash
import dash_html_components as html

app = dash.Dash()

app.layout = html.Div('Hello EuroPython!')

if __name__ == '__main__':
    app.run_server()
```
Dash - main components

- Layout (UI)
  - *html* components
  - *core* components
- Callbacks
Core Components

*aka* the moving, clickety stuff.

Example app with a lot of those (https://dash-oil-and-gas.plot.ly/)
Graphs

Core component that accepts plotly.py go.Figure object!
import dash_core_components as dcc
import plotly.graph_objs as go

app.layout = html.Div([
    html.H1('Hello EuroPython!'),
    dcc.Graph(
        id='my-first-graph',
        figure=dict(data=[dict(x=[0,1,2], y=[3,4,2])]),
    )
])
Callbacks

Where the magic happens!
Callbacks

from dash.dependencies import Input, Output

app.layout = html.Div([
    dcc.Input(id='my-id', value='initial value', type='text'),
    html.Div(id='my-div')
])

@app.callback(
    Output(component_id='my-div', component_property='children'),
    [Input(component_id='my-id', component_property='value')]
)
def update_output_div(input_value):
    return 'You\'ve entered "{}"'.format(input_value)
CSS

Let's make it prettier!
CSS

```javascript
app.css.append_css({'external_url': 'https://codepen.io/chriddyp/pen/bWLwgP.css'})
app.layout = html.Div([
    dcc.Input(id='my-id', value='initial value', type='text'),
    html.Div(id='my-div'),
],
    className='container',
)```
To summarize

- Html components (HTML tags)
- "Core" components (sliders, buttons, graphs)
- Graph objects use Plotly.py objects
- Callbacks connect the pieces
- CSS classes for pretty layout and styling
Deployment

Did I mention you don't need to know any Flask, JS, etc...?
Deployment

Did I mention you don't need to know any Flask, JS, etc...?

I lied.
Deployment
Deployment

You have choices:

1. Don't bother (1-person, local use only)
2. Know Flask
3. PaaS (e.g. Heroku, Digital Ocean)
4. Ask your engineer friend (aka Stack Overflow)
5. Ask Plotly (probably not for free)
Extra fancy stuff

(Non-exhaustive list)

• External JS
• Caching
• Optional WebGL graphs for billion-point visualization (actually >15K)
• Live updates
• Authentication
So, it’s cool and all, but...
So, it’s cool and all, but...

- You still need a web designer ¯\_(ツ)_/¯
- Understanding/debugging JS errors
- Offline mode not well supported yet/erratic
- Deployment at scale might not be trivial (but Heroku!)
I made something

twitch-viz.herokuapp.com (https://twitch-viz.herokuapp.com)
Questions?
Tweet at me @_teoguso

Further Reading/Help

- Community Forum: https://community.plot.ly/ (https://community.plot.ly/)