finxter

Plotly Dash Cheat Sheet

Charming Data

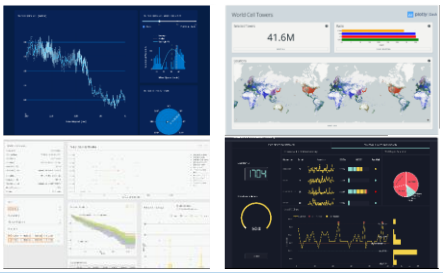
What is Dash?

Dash is one of the easiest Python frameworks to create beautiful dashboard applications. It is used in machine learning, data science, data modeling, and other data visualization applications. Setting up your first interactive dashboard app takes only a few minutes.

What Are the Best Dash resources?

- *Plotly Dash official site: https://dash.plotly.com*
- *Dash Gallery with many example dashboard apps: https://plot.ly/dash/gallery*
- *Dash YouTube Channel CharmingData: https://www.youtube.com/c/CharmingData*
- *Dash Python Book: https://blog.finxter.com/dash-book*

Dash Overview



How Does a Dash App Look Like?

- **Layout** describes how your app looks. It consists of a number of Dash components such as graphs, sliders, charts, and other inputs and outputs to visualize data.
- **Callbacks** define how the layout is connected and, thus, made interactive. For example, if a user changes an input such as a slider, your dashboard may be supposed to update a Dash component such as a graph. Callbacks make dashboards interactive.
- To develop a dashboard app with Python Dash, you simply define the layout—***how does it look?***—and the callbacks—***how does it behave?***
- *Users can then view your dashboard app in their browsers by typing in the URL on which you made it available.*

Get Started with Your First App

1. ***Install Dash***
   Type the following command in your terminal/shell:
   `pip install dash` → *Windows, macOS*
   `sudo pip install dash` → *Linux, Ubuntu*

2. ***Create minimal project***
   Copy&paste the code into a new file called "app.py" in a folder - with path /path/to/dash_app/app.py

3. ***Run Dash app***
   Open a terminal in the /path/to/dash_app/ and run python app.py

```
C:\Users\xcent\Desktop\Python\dash_example_project>python app.py
Dash is running on http://127.0.0.1:8050/

 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
```

4. ***Open Dash app in browser***
   Copy or click on the IP address 127.0.0.1:8050 and open it in your browser.

File: app.py

```
import dash
import dash_core_components as dcc
import dash_html_components as html
import plotly.graph_objects as go

es = ['https://codepen.io/chriddyp/pen/bWLwgP.css']
app = dash.Dash(__name__, external_stylesheets=es)

xs = list(range(30))
ys = [10000 * 1.07**i for i in xs]

fig = go.Figure(data=go.Scatter(x=xs, y=ys))
fig.update_layout(xaxis_title='Years', yaxis_title='$')

app.layout = html.Div(children=[
    html.H1(children='Assets'),
    dcc.Graph(figure=fig)])

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

Dash Components

The ***Dash Components*** build the user interface in your dashboard app. Describes the layout as a series of layout elements (components). Everything you can see and interact with in your app—except the concrete functionality that is implemented in the → ***Dash Callback***.

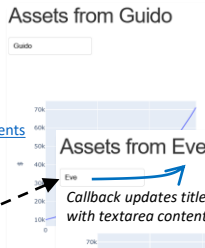
There are two types of components:

```
import dash_core_components as dcc
import dash_html_components as html
```

The Python Dash app defines one root component that contains all other components as a hierarchical tree of components. This is called the `app.layout`. You can put all types of components into it:

- The `dash_html_components` library provides the HTML tags. You instantiate them like normal Python classes using the constructor. You can pass all HTML attributes as keyword arguments such as `style`, `className`, and `id`.

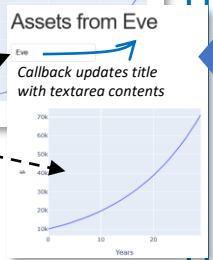
```
html.Div()
html.A()
html.Button()
html.H3(), html.H2(), html.H1()
html.Img()
```



→ More: https://dash.plotly.com/dash-html-components

- The `dash_core_components` library provides more advanced components such as graphs that are not generally part of the standard HTML elements.

```
dcc.Graph()
dcc.Input()
dcc.Dropdown()
dcc.Slider()
dcc.Textarea()
dcc.Checklist()
dcc.RadioItems()
dcc.DatePickerSingle()
```



→ More: https://dash.plotly.com/dash-core-components

Dash Callback

Dash Callback generates **user-interactivity** of your dashboard app. Connects the Dash Components to the Graphs. Defines functionality and how the → ***Dash Components*** interplay.

- A **callback** is a function that is called automatically if a defined event happens.
- A **Dash Callback** is a Python function that's automatically called whenever an input component's property changes.
- You can chain callbacks. For example, if one update in the user interface triggers a callback, this triggers another callback and so on. The purpose of doing this is to trigger several updates in the app.
- Examples callback trigger: click, select points on chart, zoom into chart, input text, choose radio button, slider input, hover.

File: app.py

```
import dash
...
from dash.dependencies import Input, Output
...

app.layout = html.Div([
    html.H1('Assets', id='my-title'),
    dcc.Input(id='my-name',
              value='Guido', type='text'),
    dcc.Graph(figure=fig)
    ])

@app.callback(
    Output(component_id='my-title',
           component_property='children'),
    [Input(component_id='my-name',
           component_property='value')]
)
def update_output_div(input_value):
    return 'Assets from {}'.format(input_value)
...
```

*Subscribe to the **11x FREE** Python Cheat Sheet Course:*
https://blog.finxter.com/python-cheat-sheets/