Hands-On Micro Lab

Exercise 1. Use pygal to plot Fibonacci number as an interactive bar graph.

Exercise 2. Use plotly to show the same can be achieved using other libraries.

Exercise 3. Dash example to publish the graph on the Web. → WHY?

The Internet is the lowest cost system ever developed to communicate with a potential audience of hundreds of millions of people all over the world.

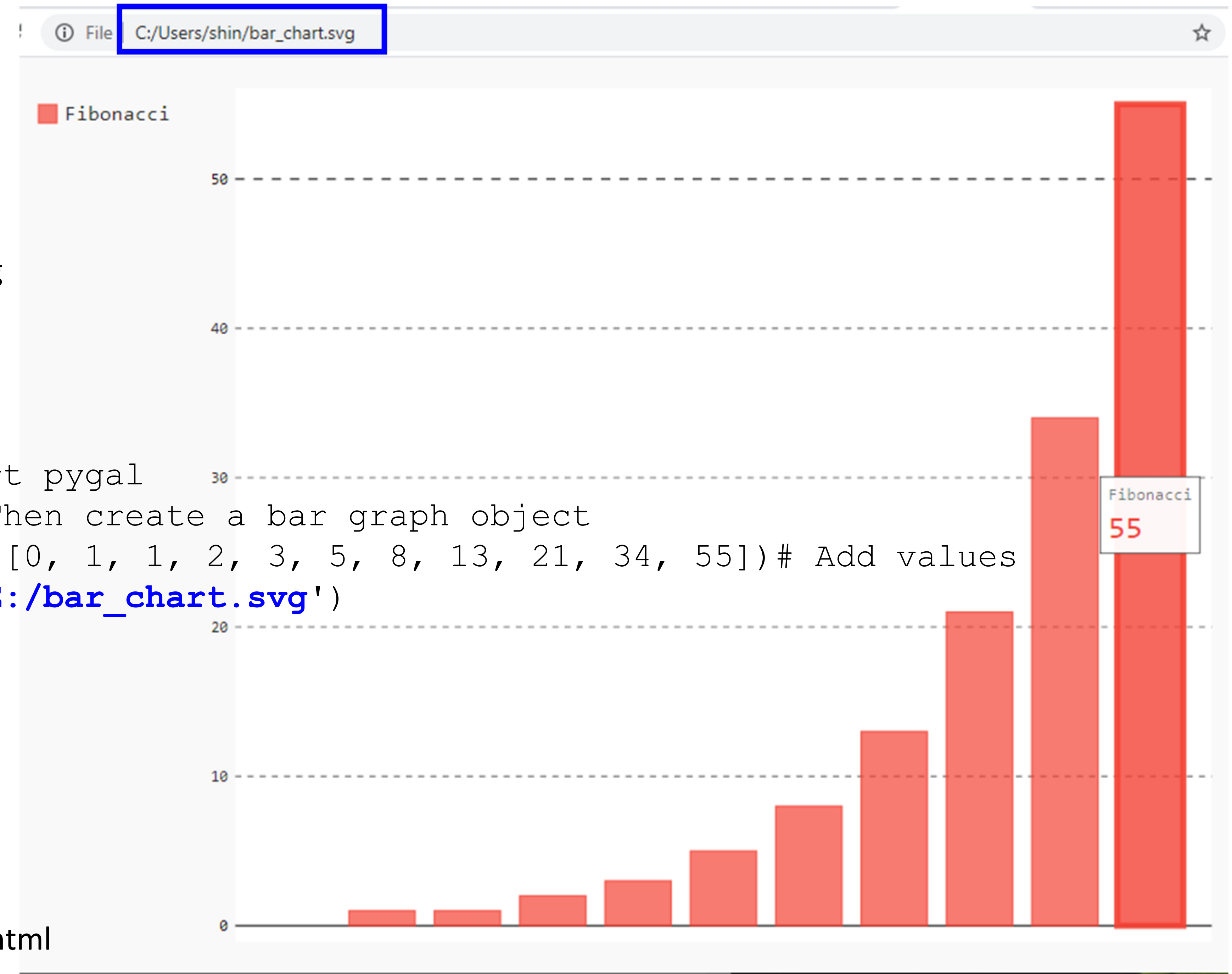
- "What Makes the Internet So Powerful?", InformIT 2002

http://www.pygal.org/en/latest/index.html

Charts display tooltips by default, but there's currently no way to zoom in and out or pan across plots.

You can output charts as **SVGs** and add them to a web page with an embed tag or by inserting the code directly into the HTML. Like mpld3, pygal is suited for smaller datasets.

```
import pygal  # First import pygal
bar_chart = pygal.Bar() # Then create a bar graph object
bar_chart.add('Fibonacci', [0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55])# Add values
bar_chart.render_to_file(‘E:/bar_chart.svg')
```



SVG: “Scalable Vector Graphics”, a XML based two-dimensional graphic file format

http://www.pygal.org/en/latest/index.html

Getting Started with Plotly in Python

Overview

The plotly Python library is an interactive, open-source plotting library that supports over 40 unique chart types covering a wide range of statistical, financial, geographic, scientific, and 3-dimensional use-cases.

Built on top of the Plotly JavaScript library (plotly.js), plotly enables Python users to create beautiful interactive web-based visualizations that can be displayed in Jupyter notebooks, saved to standalone HTML files, or served as part of pure Python-built web applications using Dash. The plotly Python library is sometimes referred to as "plotly.py" to differentiate it from the JavaScript library.

Thanks to deep integration with our Kaleido image export utility, plotly also provides great support for non-web contexts including desktop editors (e.g. QtConsole, Spyder, PyCharm) and static document publishing (e.g. exporting notebooks to PDF with high-quality vector images).

This Getting Started guide explains how to install plotly and related optional pages. Once you've installed, you can use our documentation in three main ways:

https://plotly.com/python/getting-started/

Install Python : https://www.python.org/downloads/release/python-397/

Install pip : https://phoenixnap.com/kb/install-pip-windows

Install Jupyter lab : >> pip install jupyter-lab
Install Pygal : >> pip install pygal
Install Pandas : >> pip install pandas
Install plotly : >> pip install plotly

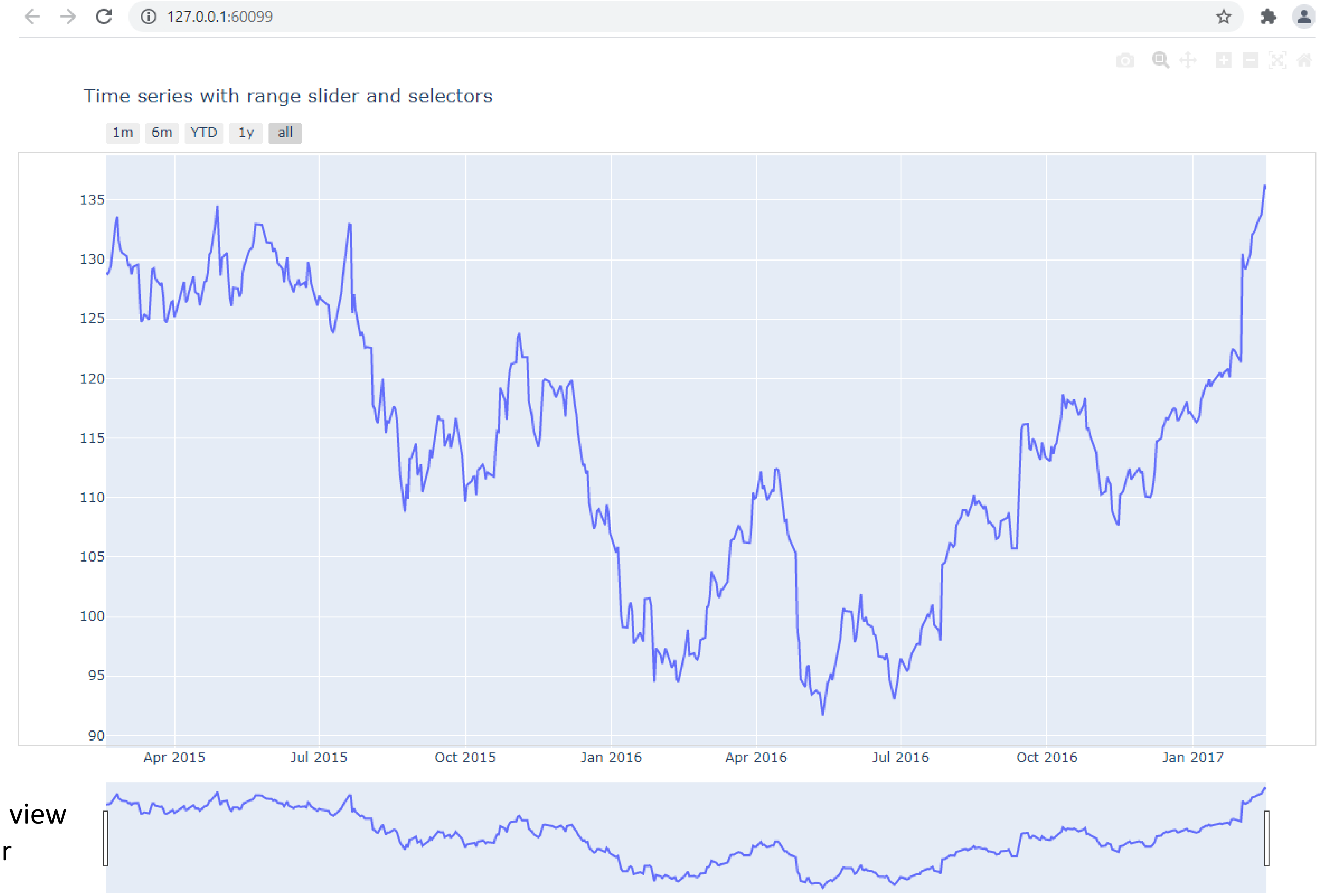
:

:

:

```
import plotly.graph_objects as go
import pandas as pd

# Load data
df = pd.read_csv("https://raw.githubusercontent.com/plotly/datasets/master/finance-charts-apple.csv")
df.columns = [col.replace("AAPL.", "") for col in df.columns]

# Create figure
fig = go.Figure()
fig.add_trace(go.Scatter(x=(df.Date), y=(df.High)))

# Set title
fig.update_layout(title_text="Time series with range slider and selectors")

# Add range slider
fig.update_layout(
    xaxis=dict(
        rangeselector=dict(
            buttons=list([
                dict(count=1, label="1m", step="month", stepmode="backward"),
                dict(count=6, label="6m", step="month", stepmode="backward"),
                dict(count=1, label="YTD", step="year", stepmode="todate"),
                dict(count=1, label="1y", step="year", stepmode="backward"),
                dict(step="all")
            ])
        ),
        rangeslider=dict(
            visible=True
        ),
        type="date"
    )
)
fig.show()
```

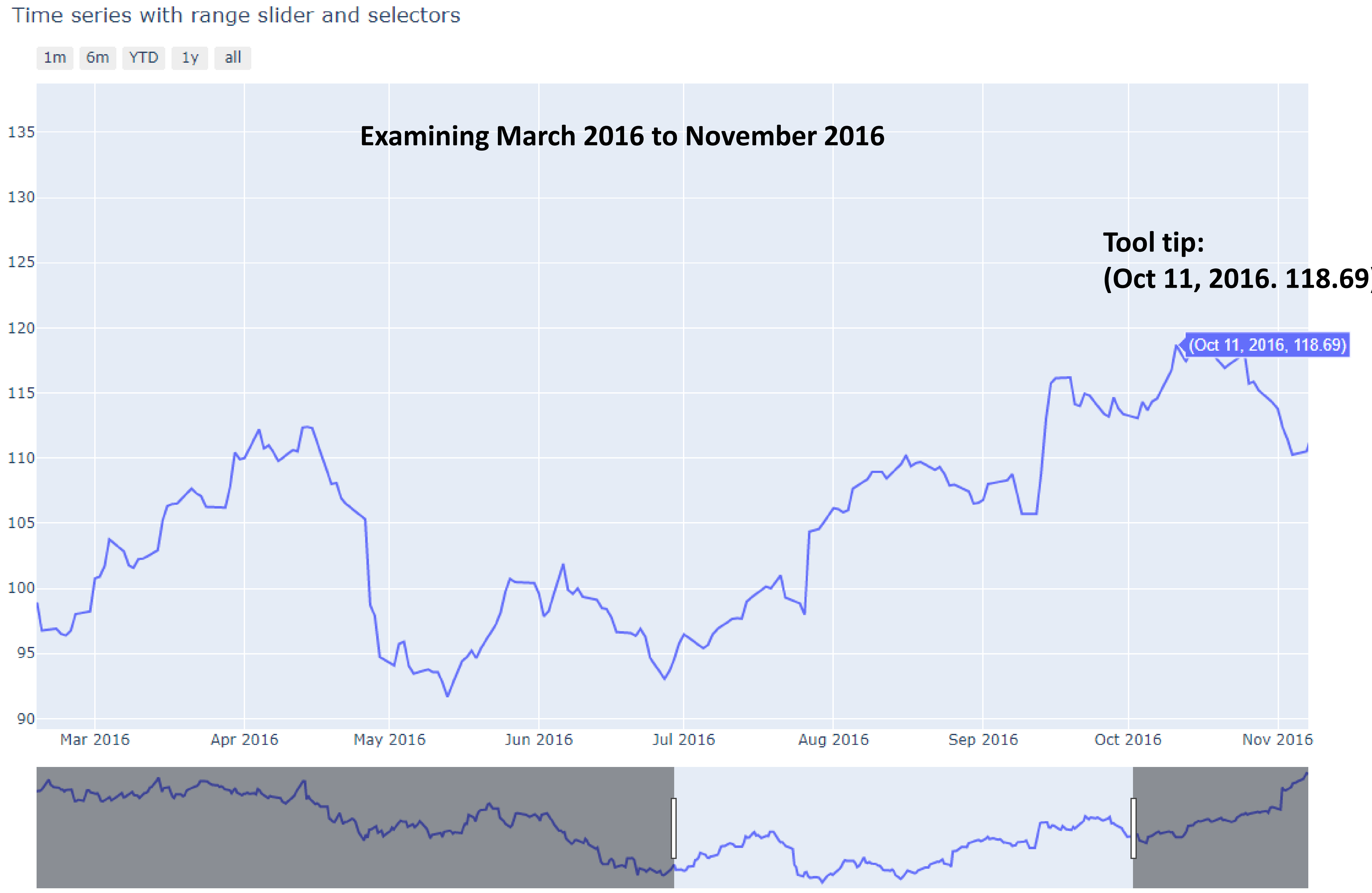
https://plotly.com/python/range-slider/



Squashed view with slider

Examining March 2016 to November 2016

Tool tip:
(Oct 11, 2016. 118.69)



What About Dash?

Dash is an open-source framework for building analytical applications, with no Javascript required, and it is tightly integrated with the Plotly graphing library.

Learn about how to install Dash at https://dash.plot.ly/installation.

Everywhere in this page that you see fig.show(), you can display the same figure in a Dash application by passing it to the figure argument of the Graph component from the built-in dash_core_components package like this:

Why Web Deployment?

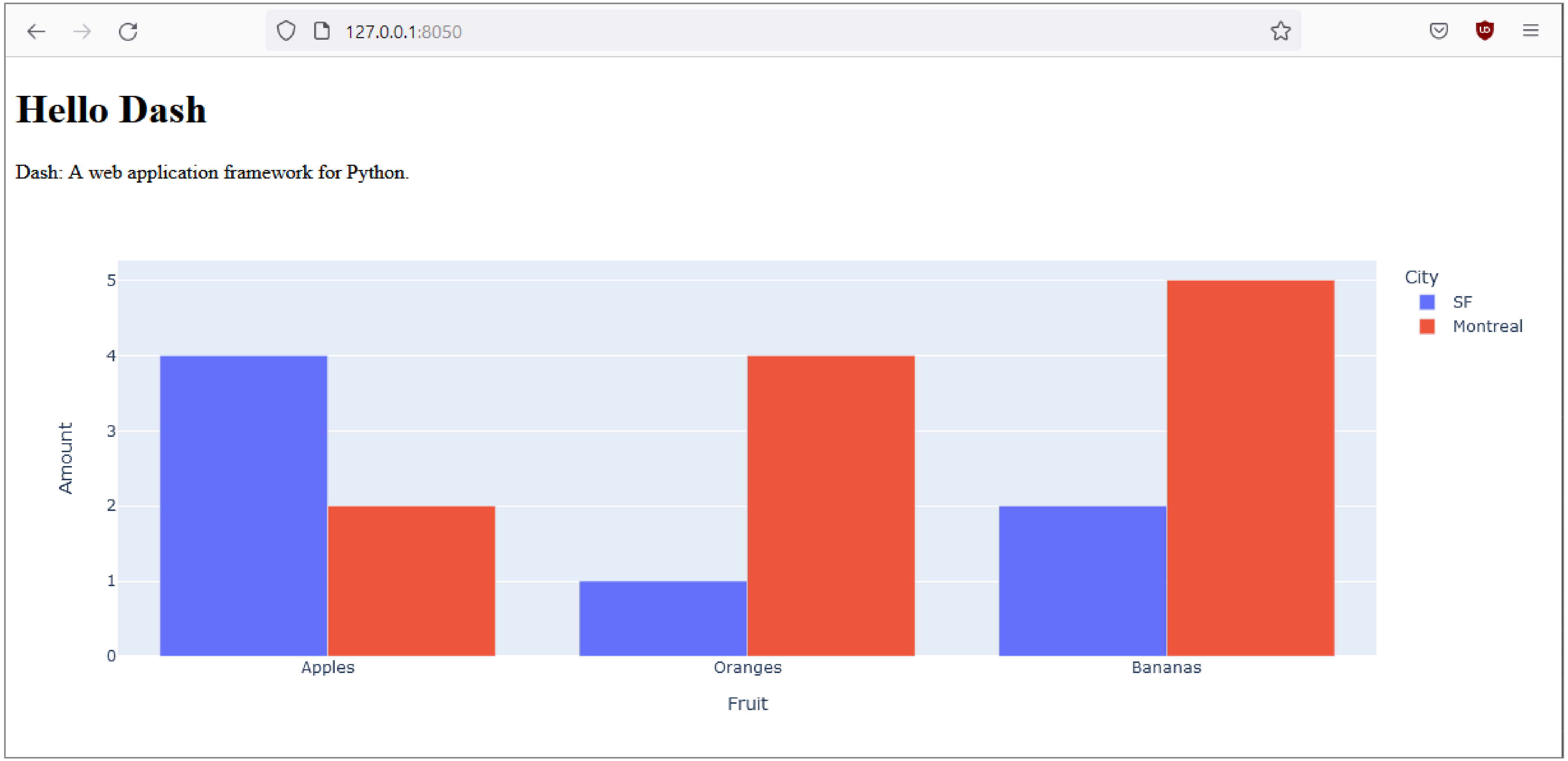
Web rules! Connectivity through the Internet.
Ease of application deployment; No software maintenance chore at the client end.

Dash apps are composed of two parts. The first part is the "layout" of the app and it describes what the application looks like. The second part describes the interactivity of the application.

Dash provides Python classes for all of the visual components of the application with the dash_core_components and the dash_html_components library. Custom JavaScript and React.js can be added.

Note: Python code examples are meant to be saved as files and executed using python app.py.

You can also use Jupyter with the JupyterDash library.



```
$ python app.py
...Running on http://127.0.0.1:8050/ (Press CTRL+C to quit)
```

Install Python : https://www.python.org/downloads/release/python-397/

Install pip : https://phoenixnap.com/kb/install-pip-windows

Install Dash	: >> pip install dash >> pip install jupyter-dash
Install dash_core_components	: >> pip install dash_core_components
Install dash_html_components	: >> pip install dash_html_components
Install dash_renderer	: >> pip install dash_renderer

:

:

:

```
import dash
import dash_core_components as dcc
import dash_html_components as html
import plotly.express as px
import pandas as pd
external_stylesheets = ['https://codepen.io/chriddyp/pen/bWLwgP.css']
app = dash.Dash(__name__, external_stylesheets=external_stylesheets)

df = pd.DataFrame({
    "Fruit": ["Apples", "Oranges", "Bananas", "Apples", "Oranges", "Bananas"],
    "Amount": [4, 1, 2, 2, 4, 5],
    "City": ["SF", "SF", "SF", "Montreal", "Montreal", "Montreal"]
})
fig = px.bar(df, x="Fruit", y="Amount", color="City", barmode="group")
app.layout = html.Div(children=[
    html.H1(children=‘Fruits'),
    html.Div(children=''' Fruits Amounts in San Francisco and Montreal. '''),
    dcc.Graph(
        id='example-graph',
        figure=fig
    )
])
if __name__ == '__main__':
    app.run_server(
        port=8050,
        host='127.0.0.1'
    )
```

1. The layout is composed of a tree of "components" like html.Div and dcc.Graph.
2. The dash_html_components library has a component for every HTML tag. The html.H1(children=‘Fruits') component generates a <h1>Fruits</h1> HTML element in your application.
3. Not all components are pure HTML. The dash_core_components describe higher-level components that are interactive and are generated with JavaScript, HTML, and CSS through the React.js library.
4. Each component is described entirely through keyword attributes. Dash is *declarative*: you will primarily describe your application through these attributes.
5. The children property is special. By convention, it's always the first attribute which means that you can omit it: html.H1(children='Fruits') is the same as html.H1(‘Fruits'). Also, it can contain a string, a number, a single component, or a list of components.
6. The fonts in your application will look a little bit different than what is displayed here. This application is using a custom CSS stylesheet to modify the default styles of the elements.