Plot CSV Data in Python

How to create charts from csv files with Plotly and Python

CSV or comma-delimited-values is a very popular format for storing structured data. In this tutorial, we will see how to plot beautiful graphs using csv data, and Pandas. We will learn how to import csv data from an external source (a url), and plot it using Plotly and pandas.

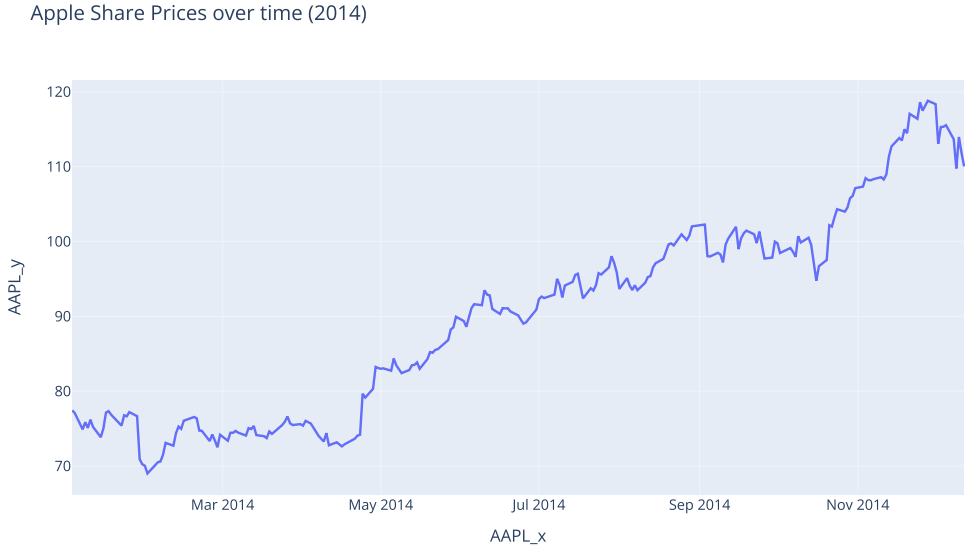
First we import the data and look at it.

```
import pandas as pd
df = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/2014_apple_stock.csv')
df.head()
```

	AAPL_x	AAPL_y
0	2014-01-02	77.445395
1	2014-01-03	77.045575
2	2014-01-06	74.896972
3	2014-01-07	75.856461
4	2014-01-08	75.091947

Plot from CSV with Plotly Express

```
import pandas as pd
import plotly.express as px

df = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/2014_apple_stock.csv')

fig = px.line(df, x = 'AAPL_x', y = 'AAPL_y', title='Apple Share Prices over time (2014)')
fig.show()
```



Plot from CSV in Dash

Dash (https://plotly.com/dash/) is the best way to build analytical apps in Python using Plotly figures. To run the app below, run `pip install dash`, click "Download" to get the code and run `python app.py`.

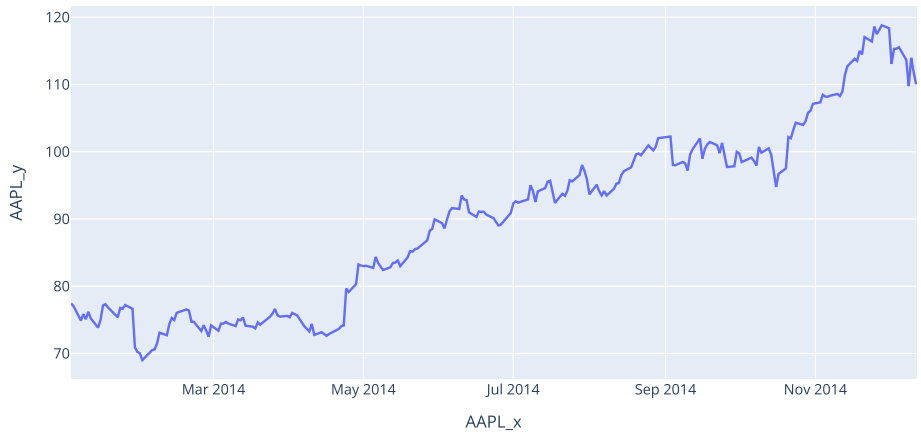
Get started with the official Dash docs (https://dash.plotly.com/installation) and **learn how to effortlessly** style (https://plotly.com/dash/design-kit/) **&** deploy (https://plotly.com/dash/app-manager/) **apps like this with** Dash Enterprise (https://plotly.com/dash/)**.**

DOWNLOAD

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

app = Dash(__name__)

app.layout = html.Div([
    html.H4('Simple stock plot with adjustable axis'),
    html.Button("Switch Axis", n_clicks=0,
                id='button'),
    dcc.Graph(id="graph"),
])


@app.callback(
    Output("graph", "figure"),
    Input("button", "n_clicks"))
def display_graph(n_clicks):
    df = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/2014_apple_stock.csv') # replace with your own data source

    if n_clicks % 2 == 0:
        x, y = 'AAPL_x', 'AAPL_y'
    else:
        x, y = 'AAPL_y', 'AAPL_x'
```

Simple stock plot with adjustable axis

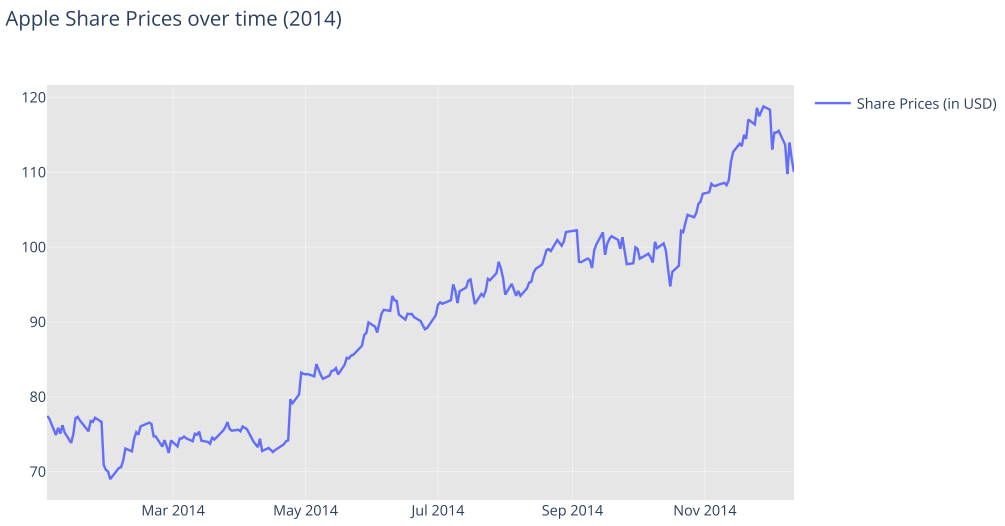
SWITCH AXIS



Sign up for Dash Club → Free cheat sheets plus updates from Chris Parmer and Adam Schroeder delivered to your inbox every two months. Includes tips and tricks, community apps, and deep dives into the Dash architecture. Join now (https://go.plotly.com/dash-club?utm_source=Dash+Club+2022&utm_medium=graphing_libraries&utm_content=inline).

Plot from CSV with graph_objects

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

df = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/2014_apple_stock.csv')

fig = go.Figure(go.Scatter(x = df['AAPL_x'], y = df['AAPL_y'],
                  name='Share Prices (in USD)'))

fig.update_layout(title=dict(text='Apple Share Prices over time (2014)'),
                   plot_bgcolor='rgb(230, 230,230)',
                   showlegend=True)

fig.show()
```



Reference

See https://plotly.com/python/getting-started (https://plotly.com/python/getting-started) for more information about Plotly's Python API!

What About Dash?

Dash (https://dash.plot.ly/) 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 (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 (https://dash.plot.ly/dash-core-components/graph) from the built-in dash_core_components package like this:

```
import plotly.graph_objects as go # or plotly.express as px
fig = go.Figure() # or any Plotly Express function e.g. px.bar(...)
# fig.add_trace( ... )
# fig.update_layout( ... )

from dash import Dash, dcc, html

app = Dash()
app.layout = html.Div([
    dcc.Graph(figure=fig)
])

app.run(debug=True, use_reloader=False)  # Turn off reloader if inside Jupyter
```