plotly

Star 24,429

Dash Python **>** ***Part 3. Interactive Graphing and Crossfiltering***

> Introducing Plotly Cloud — The easiest way to publish your Dash apps. **Learn more.**

Interactive Visualizations

> This is the 3rd chapter of the Dash Fundamentals. The **previous chapter** covered basic callback usage. The **next chapter** describes how to share data between callbacks. Just getting started? Make sure to **install the necessary dependencies**.

The Dash Core Components (`dash.dcc`) module includes a Graph component called `dcc.Graph`.

`dcc.Graph` renders interactive data visualizations using the open source **plotly.js** JavaScript graphing library. Plotly.js supports over 35 chart types and renders charts in both vector-quality SVG and high-performance WebGL.

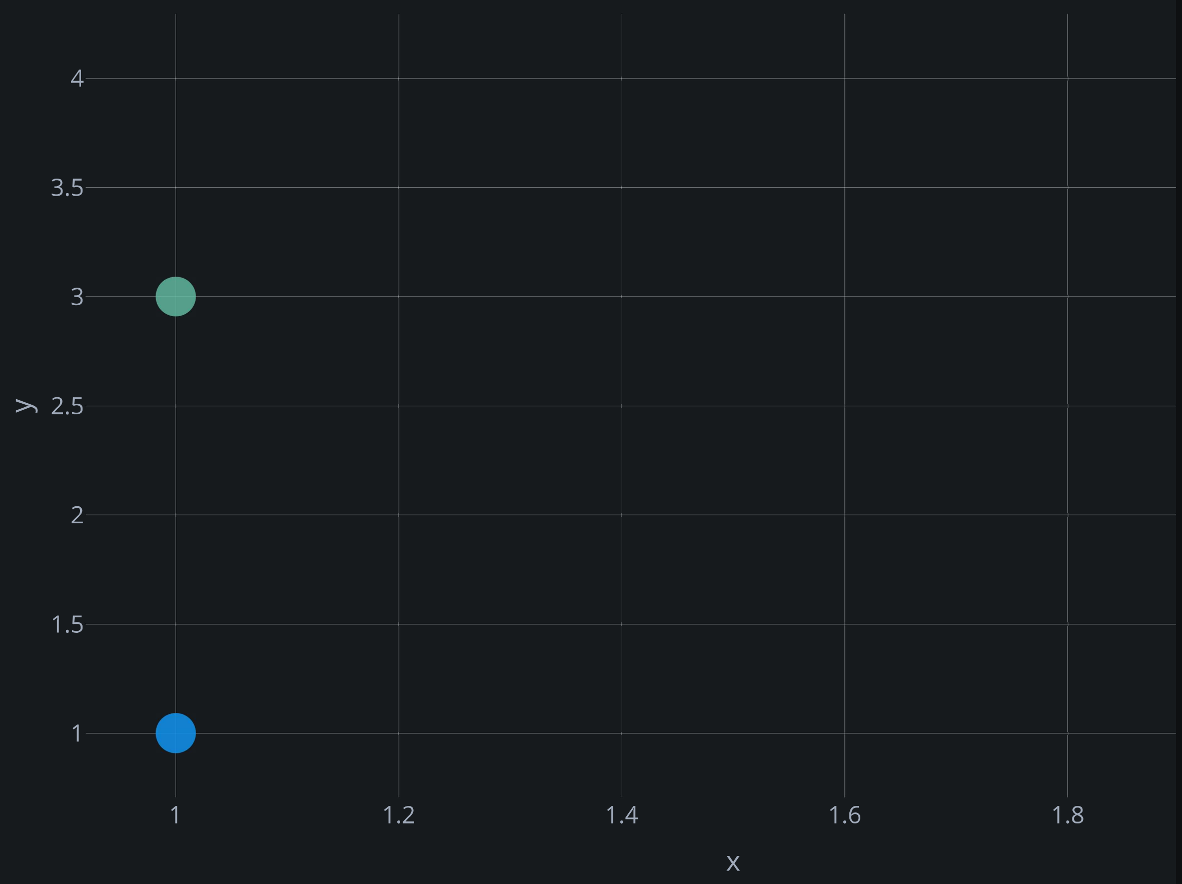
The `figure` argument in the `dcc.Graph` component is the same `figure` argument that is used by `plotly.py`. Check out the **plotly.py documentation and gallery** to learn more.

As we already saw, Dash components are described by a set of attributes. Any of these attributes can be updated by callback functions, but only a subset of these attributes are updated through user interaction, such as typing inside a `dcc.Input` component or clicking an option in a `dcc.Dropdown` component.

The `dcc.Graph` component has four attributes that can change through user-interaction: `hoverData`, `clickData`, `selectedData`, `relayoutData`. These properties update when you hover over points, click on points, or select regions of points in a graph.

Here's an example that prints these attributes to the screen.

▶ Show code



Hover Data

Mouse over values in the graph.

```
null
```

Click Data

Click on points in the graph.

```
null
```

Selection Data

Choose the lasso or rectangle tool in the graph's menu bar and then select points in the graph.

Note that if `layout.clickmode = 'event+select'`, selection data also accumulates (or un-accumulates) selected data if you hold down the shift button while clicking.

```
null
```

Zoom and Relayout Data

Click and drag on the graph to zoom or click on the zoom buttons in the graph's menu bar.

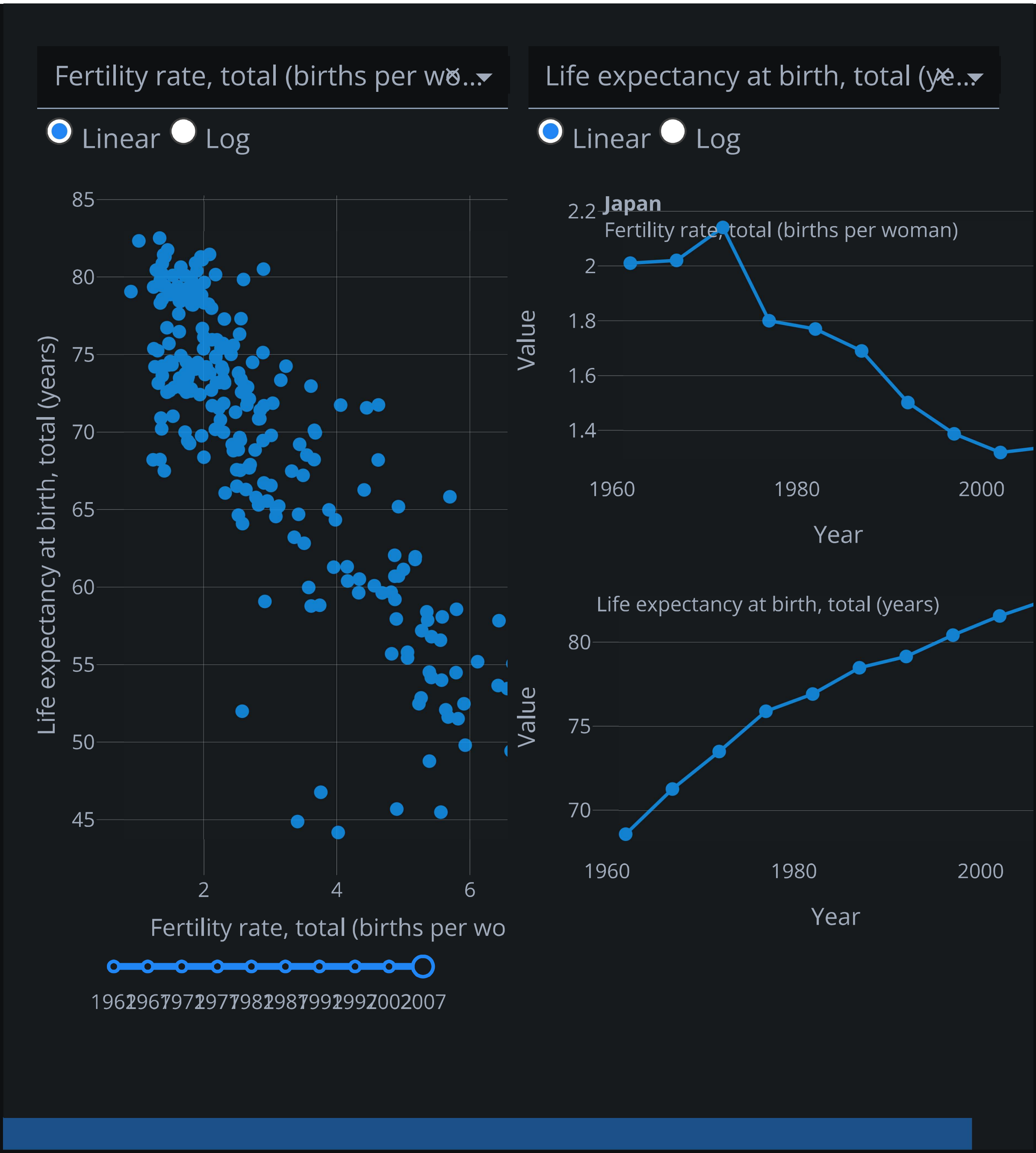
```
{
  "autosize": true
}
```

> For optimal user interaction and chart loading performance, Dash apps in production should consider the **Job Queue**, **HPC**, **Datashader**, and horizontal scaling capabilities of Dash Enterprise.

Update Graphs on Hover

Let's update our world indicators example from the previous chapter by updating the time series when we hover over points in our scatter plot.

▶ Show code



Fertility rate, total (births per wo... ×

● Linear ○ Log

Life expectancy at birth, total (ye... ×

● Linear ○ Log

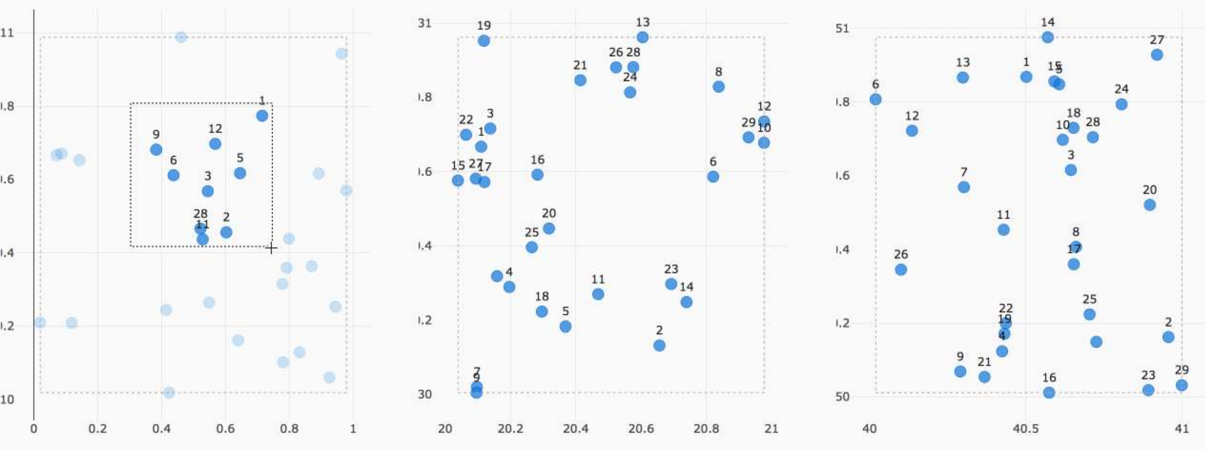
85
80
75
70
65
60
55
50
45
Life expectancy at birth, total (years)
2
4
6
Fertility rate, total (births per wo
1962
1967
1972
1977
1982
1987
1992
1997
2002
2007
Japan
Fertility rate, total (births per woman)
2.2
2
1.8
1.6
1.4
Value
1960
1980
2000
Year
Life expectancy at birth, total (years)
80
75
70
Value
1960
1980
2000
Year

Try moving the mouse over the points in the scatter plot on the left. Notice how the line graphs on the right update based on the point that you are hovering over.

Generic Crossfilter Recipe

Here's an example of crossfiltering across a six-column data set. Each scatter plot's selection filters the underlying dataset.

▶ Show code



On every selection, the three graph callbacks are fired with the latest selected regions of each plot. A pandas dataframe is filtered based on the selected points and the graphs are replotted with the selected points highlighted and the selected region drawn as a dashed rectangle.

> As an aside, if you find yourself filtering and visualizing highly-dimensional datasets, you should consider checking out the **parallel coordinates** chart type.

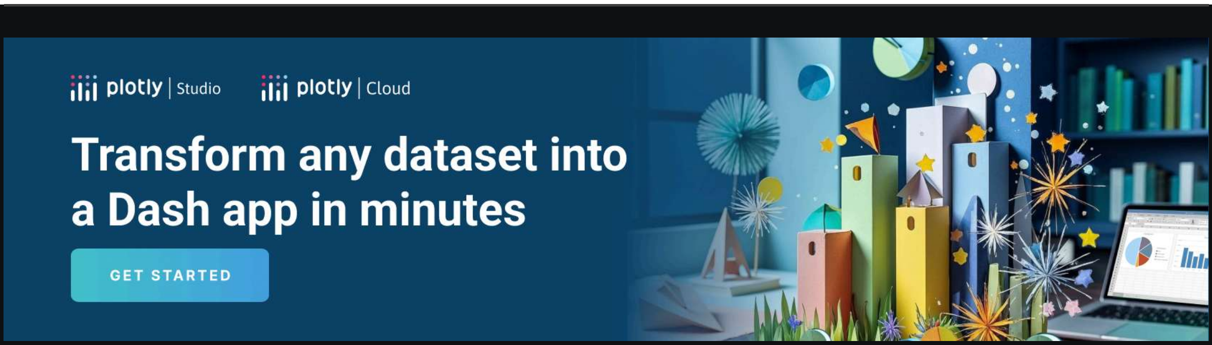
Current Limitations

There are a few limitations in graph interactions right now.

- It is not currently possible to customize the style of the hover interactions or the select box. This issue is being worked on in **https://github.com/plotly/plotly.js/issues/1847**.

There's a lot that you can do with these interactive plotting features. If you need help exploring your use case, open up a thread in the **Dash Community Forum**.

The next chapter of the Dash Fundamentals explains how to share data between callbacks. **Go to Dash Fundamentals Part 4: Sharing Data Between Callbacks**.



Dash Python > ***Part 3. Interactive Graphing and Crossfiltering***

Products

Plotly Studio

Plotly Cloud

Dash Enterprise

New Releases →

Pricing

Studio and Cloud Pricing

Dash Enterprise Pricing

About Us

Careers

Resources

Blog

Support

Community Support

Graphing Documentation

Join our mailing list

Sign up to stay in the loop with all things Plotly — from Dash Club to product updates, webinars, and more!

SUBSCRIBE

Terms of Service Privacy Policy