plotly

Star 23,447

Dash Python > ***Dash in Jupyter Environments***

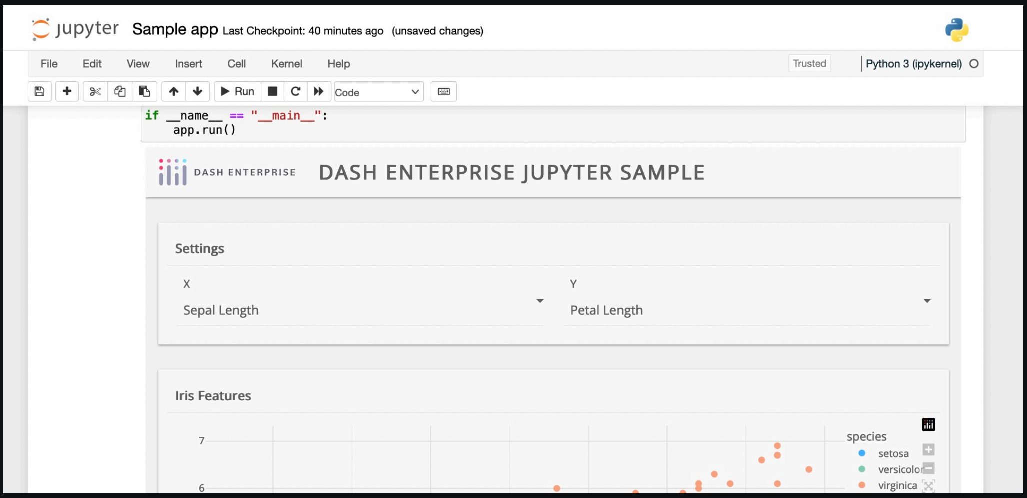
> Plotly Studio: Transform any dataset into an interactive data application in minutes with AI. **Sign up for early access now.**

Dash in Jupyter Environments

Dash 2.11 and later supports running Dash apps in classic Jupyter Notebooks and in JupyterLab without the need to update the code or use the additional `JupyterDash` library. If you are using an earlier version of Dash, you can run Dash apps in a notebook using **JupyterDash**. This page documents additional options available when running Dash apps in notebooks as well as troubleshooting information.

Display Modes

When you run an app in a cell, it displays inline by default:



```
if __name__ == "__main__":
    app.run()
```

DASH ENTERPRISE JUPYTER SAMPLE

Settings

X: Sepal Length

Y: Petal Length

Iris Features

External mode

You can also configure it to display a link to visit to view the app by using `external` mode:

```
app.run(jupyter_mode="external")
```

Tab mode

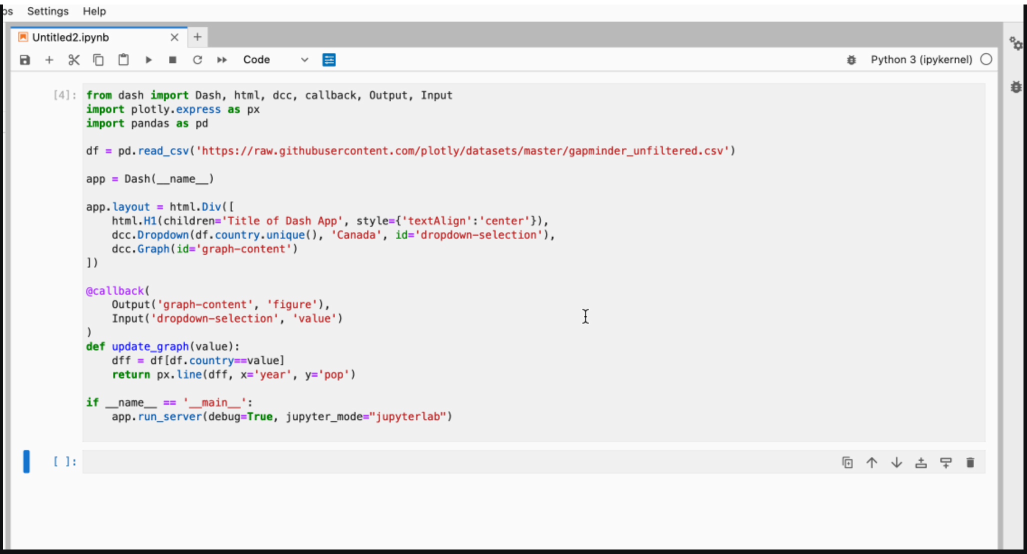
Use `tab` mode to automatically open the app in a new browser tab:

```
app.run(jupyter_mode="tab")
```

Jupyterlab mode

In `jupyterlab` mode, the notebook displays the app in a separate tab in JupyterLab.

```
app.run(jupyter_mode="jupyterlab")
```



```
[4]: from dash import Dash, html, dcc, callback, Output, Input
     import plotly.express as px
     import pandas as pd

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

     app = Dash(__name__)

     app.layout = html.Div([
         html.H1(children='Title of Dash App', style={'textAlign':'center'}),
         dcc.Dropdown(df.country.unique(), 'Canada', id='dropdown-selection'),
         dcc.Graph(id='graph-content')
     ])

     @callback(
         Output('graph-content', 'figure'),
         Input('dropdown-selection', 'value')
     )
     def update_graph(value):
         dff = df[df.country==value]
         return px.line(dff, x='year', y='pop')

     if __name__ == '__main__':
         app.run_server(debug=True, jupyter_mode="jupyterlab")

[ ]:
```

To use `jupyterlab` mode, you'll also need to build the `@plotly/dash-jupyterlab` extension. When you launch a new JupyterLab notebook after installing Dash, you'll be prompted to build `@plotly/dash-jupyterlab`.

> JupyterLab version 3 is required to run a Dash app in `jupyterlab` mode.

Set the mode at a notebook level

In the above examples, we configure the `jupyter_mode` on the Dash app. You can also apply this setting at a notebook level so that all Dash apps rendered in the notebook use it. Here, we set the default mode to "external":

```
from dash import jupyter_dash

jupyter_dash.default_mode="external"
```

Set the server URL for the app

By default, the app in external or tab mode is served at `http://127.0.0.1:8050/` (the localhost address on port 8050) if you don't set a different `host` or `port` on `app.run`. If you are using a hosted notebook, you will need to specify the URL used by that hosted environment. In **Dash Enterprise workspaces**, this is detected automatically and the external URL printed out in the notebook will be your Dash Enterprise app's full URL. In other environments, you can configure the URL by setting `jupyter_server_url` on `app.run`:

```
app.run(jupyter_server_url="<your-url>")
```

In hosted environments, you can also use `jupyter_dash.infer_jupyter_proxy_config()`, which will attempt to detect the URL and path that the hosted environment uses.

For more details, see the troubleshooting section below.

Notes on display modes

- Setting `jupyter_mode` on an individual app within the notebook overrides the default mode for that app.
- `jupyterlab` and `tab` modes are not supported in Google Colab.

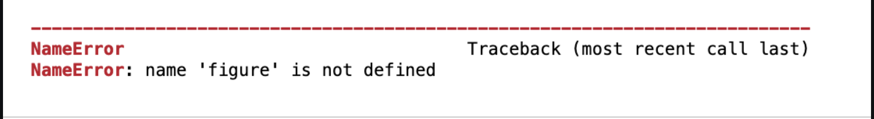
Height and Width

Dash apps run in a notebook at a default width of "100%" and height of "650". You can configure these with `jupyter_width` and `jupyter_height` on `app.run`:

```
app.run(jupyter_height=500, jupyter_width="70%")
```

Turn Off Inline Callback Exceptions

By default, any exceptions in your app's callbacks display inline.



```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
NameError: name 'figure' is not defined
```

You can disable inline callback errors by setting `jupyter_dash.inline_exceptions` to `False`:

```
from dash import jupyter_dash

jupyter_dash.inline_exceptions = False
```

Troubleshooting

- If you try to run a Dash app on a port that is already in use outside of the notebook (for example, if you have a Dash app running elsewhere on your computer), you'll receive an "Address already in use" message in the notebook. To resolve, either stop the other process that is using the port, or specify a different port in the Dash app in the notebook.

  Dash apps run on port 8050 by default. You can specify a different port in `app.run`:

  ```
  app.run(port=8060)
  ```

- If your notebook is hosted on a web server behind a proxy, Dash may not run correctly in your notebook. Try executing `jupyter_dash.infer_jupyter_proxy_config()` in a cell before your app and Dash will attempt to determine the proxy configuration.

  ```
  from dash import jupyter_dash

  jupyter_dash.infer_jupyter_proxy_config()
  ```

  Note: This may not work for all notebooks hosted on a web server behind a proxy. If you encounter an issue with a specific notebook environment, please open an issue in the **Dash GitHub repo**.

Limitations

- Multi-page apps using **Dash Pages** are not supported in notebooks.

Dash Python **>** ***Dash in Jupyter Environments***

Products

Dash

Consulting and Training

Pricing

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