

Solutions for Dash App Interactivity Assignment

Exercise A: Incorporate the [2011_us_ag_exports.csv](#) dataset into your app. And create the following layout in one app file:

1. A [Dropdown](#) that uses column `state` as the dropdown options. Then, assign “Alabama” as the initial value. The dropdown `id` should equal “state-dropdown”.

This is the result that you should see:

Alabama

Python

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

df =
pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/Dash-Course/US-Exports/2011_us_ag_exports.csv')

app = Dash(__name__)

app.layout = html.Div([
    dcc.Dropdown(id="state-dropdown", options=df.state.unique(), value="Alabama")
])

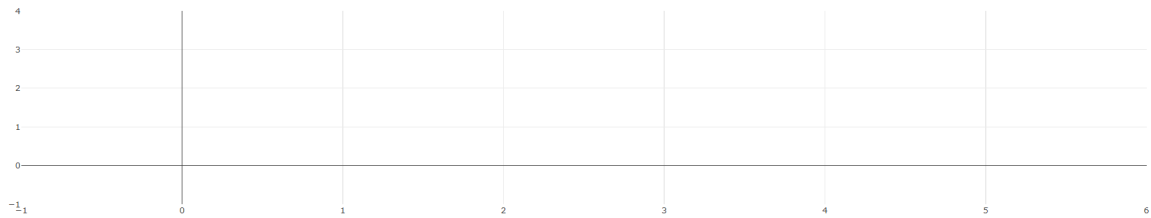
if __name__ == '__main__':
    app.run(debug=True)
```

2. Above the dropdown, add an `html.Div`, and assign to the `id` property the string “my-title”. Add your own title to the `children` property of the `html.Div`. Below the dropdown, add an empty `dcc.Graph`. The `id` of the graph component should be “graph1”.

This is the result that you should see:

Us Agricultural Exports in 2011

Alabama



Solution 1:

Python

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

df =
pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/Dash-Course/US-Exports/2011-us-ag-exports.csv')

app = Dash(__name__)

app.layout = html.Div([
    html.Div(id="my-title", children="Us Agricultural Exports in 2011"),
    dcc.Dropdown(id="state-dropdown", options=df.state.unique(), value="Alabama"),
    dcc.Graph(id="graph1")
])

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

Exercise B: Add a [callback](#) to your app.

1. Add a callback decorator that takes the `value` of the `dcc.Dropdown` as an Input argument and the `figure` of the `dcc.Graph` as an Output argument. Remember to import the Input and Output arguments at the very top.

Solution B1:

```

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

df =
pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/Dash-Course/US-Exports/2011_us_ag_exports.csv')

app = Dash(__name__)

app.layout = html.Div([
    html.Div(id="my-title", children="Us Agricultural Exports in 2011"),
    dcc.Dropdown(id="state-dropdown", options=df.state.unique(), value="Alabama"),
    dcc.Graph(id="graph1")
])

@app.callback(
    Output(component_id='graph1', component_property='figure'),
    Input(component_id='state-dropdown', component_property='value')
)

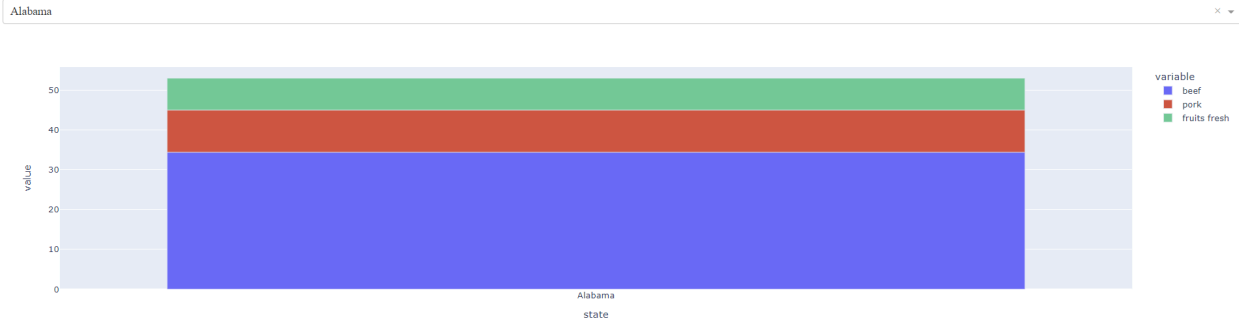
if __name__ == '__main__':
    app.run(debug=True)

```

2. Add a callback function directly underneath the decorator. The function should take an argument called “state_selected”. Use pandas to filter the original dataframe (df) so that only rows with the “state_selected” remain in the new dataframe. Name the new dataframe “df_country”.

Create a [bar chart](#) with three arguments: the “df_country” assigned to the `data_frame` attribute; “state” assigned to the x-axis; and a list of three column names assigned to the y-axis. `['beef', 'pork', 'fruits fresh']`

Return the bar chart at the end of the function. This is the app that you should see:



Solution B2:

Python

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

df =
pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/Dash-Course/US-Exports
/2011_us_ag_exports.csv')

app = Dash(__name__)

app.layout = html.Div([
    html.Div(id="my-title", children="Us Agricultural Exports in 2011"),
    dcc.Dropdown(id="state-dropdown", options=df.state.unique(), value="Alabama"),
    dcc.Graph(id="graph1"),
])

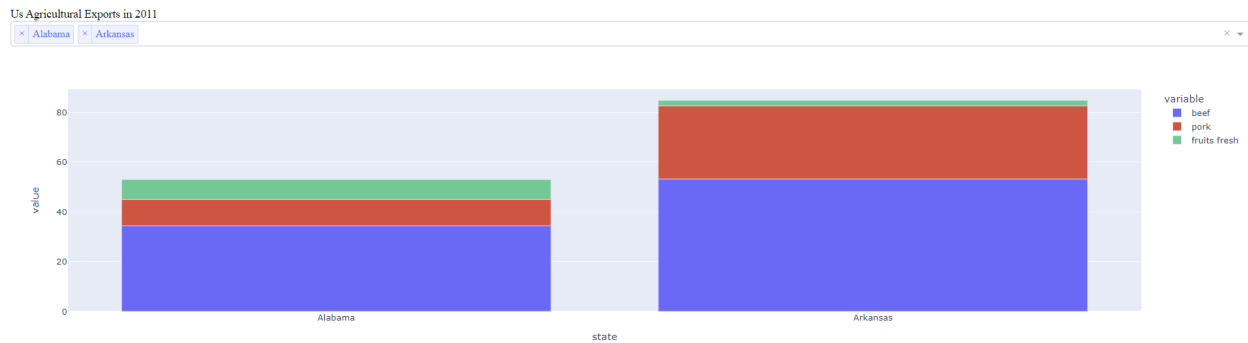
@app.callback(
    Output(component_id='graph1', component_property='figure'),
    Input(component_id='state-dropdown', component_property='value')
)
def update_graph(state_selected):
    df_country = df[df.state == state_selected]
    fig1 = px.bar(data_frame=df_country, x='state', y=['beef', 'pork', 'fruits fresh'])
    return fig1

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

3. Let's work with a [multi-value dropdown](#). Use the `multi` property to allow your dropdown to accept multiple values. Update the `value` property by assigning it `["Alabama", "Arkansas"]`

Now that the dropdown `value` property is of type list, update the callback function: use pandas' [isin method](#) on the `state` column to filter the dataframe according to the `states_selected`. The rest of the callback function can stay the same.

This is the app that you should see:



Solution B3:

Python

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

df =
pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/Dash-Course/US-Exports
/2011-us_ag_exports.csv')

app = Dash(__name__)

app.layout = html.Div([
    html.Div(id="my-title", children="Us Agricultural Exports in 2011"),
    dcc.Dropdown(id="state-dropdown", options=df.state.unique(),
value=["Alabama", "Arkansas"], multi=True),
    dcc.Graph(id="graph1"),
])

@app.callback(
    Output(component_id='graph1', component_property='figure'),
    Input(component_id='state-dropdown', component_property='value')
)
def update_graph(states_selected):
    df_country = df[df.state.isin(states_selected)]
    fig1 = px.bar(data_frame=df_country, x='state', y=['beef', 'pork', 'fruits fresh'])
```

```
return fig1

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

Exercise C: Choose your app project.

- We encourage you to think of an app that you would like to build by the end of this course. You can choose one of the two datasets ([agriculture](#), [makeup](#)) provided in the course or your own dataset. When choosing a project, think of the following:
 - What data would I like to analyze? Is the data simple and clean?
 - What is the purpose of building this app?
 - What are you trying to facilitate or demonstrate with the app?
 - Why would you or others use your app?
- Post your thoughts on the forum if you want community feedback
- At the end of the course, when you've finished your Dash app, make sure to post the code and app images on the forum, under the dash-course tag
- A couple of participants will be selected to present their apps to Plotly staff members at the last session