plotly

Star 23,447

Dash Python > ***Dash 3.0 Migration***

Dash 3.0 Migration

This page outlines changes in Dash version 3.0 and cases where you may need to update an existing app to work with version 3.0.

Removed

The following were deprecated in previous versions of Dash and have now been removed in Dash 3.0.

Long Callbacks

Replace long callbacks with **background callbacks**.

- Instead of `dash.long_callback.managers.CeleryLongCallbackManager`, use `dash.CeleryManager`.
- Instead of `dash.long_callback.managers.DiskcacheLongCallbackManager`, use `dash.DiskcacheManager`.
- Instead of the `long_callback_manager` argument on the `dash.Dash` constructor, use `background_callback_manager`.
- Also, update `@app.long_callback` to be a standard callback with `@app.callback` or `@dash.callback`, but which sets `background=True`.

  For example:

```
@app.long_callback(
    output=Output("paragraph_id", "children"),
    inputs=Input("button_id", "n_clicks"),
    manager=long_callback_manager,
)
```

  becomes

```
@callback(
    output=Output("paragraph_id", "children"),
    inputs=Input("button_id", "n_clicks"),
    background=True,
    manager=background_callback_manager,
)
```

  The functionality of parameters previously available to customize `app.long_callback` are available on `app.callback` and `dash.callback`: `interval`, `running`, `cancel`, `progress`, `progress_default`, `cache_args_to_ignore`, and `manager`.

Here's a complete app using long callbacks rewritten to use background callbacks:

Long Callback

Background Callback

```
import time

import dash
from dash import html, Input, Output
from dash.long_callback import DiskcacheLongCallbackManager

import diskcache

cache = diskcache.Cache("./cache")
long_callback_manager = DiskcacheLongCallbackManager(cache)

app = dash.Dash(__name__)

app.layout = html.Div(
    [
        html.Div([html.P(id="paragraph_id", children=["Button not clicked"])]),
        html.Button(id="button_id", children="Run Job!"),
    ]
)

@app.long_callback(
    output=Output("paragraph_id", "children"),
    inputs=Input("button_id", "n_clicks"),
    manager=long_callback_manager,
)
def callback(n_clicks):
    time.sleep(2.0)
    return [f"Clicked {n_clicks} times"]


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

`dcc.LogoutButton`

Replace `dcc.LogoutButton` with `html.Button` or use the **Dash Enterprise Auth** logout button functionality.

`Dash.run_server`

To run your app, replace `Dash.run_server` with `Dash.run`.

`dash_core_components`, `dash_html_components`, and `dash_table` Imports

Since Dash 2.0, `dash_core_components`, `dash_html_components`, and `dash_table` have been part of the `dash` package, instead of being separate packages. This changed how these packages are imported, though backwards compatibility was maintained in Dash 2.0 by including stub versions.

Dash 3.0 no longer includes the stub versions of `dash_core_components`, `dash_html_components`, and `dash_table`, meaning you may need to update some imports for your code to work with Dash 3.

If you have the following imports in your code, update them:

- Update `import dash_core_components` to `from dash import dcc`
- Update `import dash_html_components` to `from dash import html`
- Update `import dash_table` to `from dash import dash_table`

Default React Version

The default version of React in Dash 3.0 is React 18.3.1. In most cases, you won't need to make any updates to your apps and they will continue to work.

If you do need to use the previous default React version (16.14.0) in your app, you can add the following:

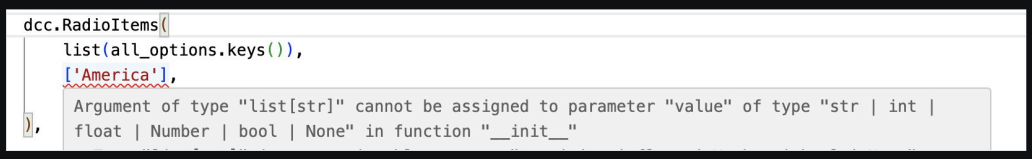
```
import dash
dash._dash_renderer._set_react_version("16.14.0")

app = dash.Dash()
...
```

`dash._dash_renderer._set_react_version("16.14.0")` sets the React version to 16.14.0 and must come before the app instance is created.

Typing Support

Components in `dash.html` and `dash.dcc` now support Python `typing` on the component `__init__`. This means that static type checkers can identify issues if you pass an incorrect type when creating a component, like in the following code.

```
dcc.RadioItems(
    options=list(all_options.keys()),
    value=['America'], # list is not a valid type for `value`
    id='countries-radio',
),
```



> Known issue: There is a known issue where Pyright displays errors when used with Dash 3.0. We are working on improving support for Pyright in future Dash releases. We also welcome pull requests that contribute to improving Pyright support. If you'd like to learn more about contributing to Dash, see the **Contributor Guide**.

Compatibility with Other Dash Libraries

If you are using any of the following libraries with Dash 3.0, you'll need to update to the minimum version specified:

Library	Minimum version for Dash 3.0
dash-ag-grid	31.3.1rc1
dash-bootstrap-components	2.0.0
dash-daq	1.6.0
dash-enterprise-auth	0.2.5
dash-design-kit	2.1.0
dash-mantine-components	1.0.0
dash-user-analytics	0.0.3

Additional Resources

- Dash 3 also introduces a new hooks feature for writing plugins. See the **Writing Dash Plugins using Dash Hooks** page for a guide to creating pip-installable plugins that work with Dash apps.

Dash Python **> *Dash 3.0 Migration***