Preserving UI State, like Zoom, in dcc.Graph with uirevision with Dash

announcements

chriddyp 1 November 19, 2018, 5:21pm

👋 Hello everyone, happy monday 🙂

I'd to share with you a really cool `dash-core-components` prerelease: the ability to preserve UI state in the `dcc.Graph` between callbacks.

Right now, if you're updating the `figure` property of a `dcc.Graph` component, the graph's "view" will get reset when the callback is fired. That is, if you zoom in your graph, click on the legend items, or twist a 3D surface plot, then those changes won't be preserved across callback updates.

For many callbacks, this is OK and actually desirable: if your graph is updating with completely new data, perhaps with completely different axes ranges, you'll want the graph to recompute its ranges.

However, for certain callbacks, especially those that have a similar set of axes ranges, you may want to preserve the UI state between callbacks: if your viewers painstakingly zoom into a certain region of a chart then they might not that graph to completely reset when a `dcc.Interval` fires or if they want to compare that region with another dataset.

```
pip install dash-core-components==0.39.0rc4
```

This version includes a new property in the `layout` property of the `figure` property of `dcc.Graph` : `uirevision` .

`uirevision` is where the magic happens. This key is tracked internally by `dcc.Graph` , when it changes from one update to the next, it resets all of the user-driven interactions (like zooming, panning, clicking on legend items). If it remains the same, then that user-driven UI state doesn't change.
It can be equal to anything, the important thing is to make sure that it changes when you want to reset the user state.

In the example below, we only reset the zoom when we change the dataset dropdown. If we change the color or if we add a "reference" trace, then we don't reset the zoom. Thus, we set the `uirevision` property to the `dataset` value. Read more in the comments embedded in the example.


```
import dash
from dash.dependencies import Input, Output
import dash_html_components as html
import dash_core_components as dcc
import pandas as pd

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

df['reference'] = df[df.columns[0]]

app = dash.Dash(__name__)
app.scripts.config.serve_locally = True
app.css.config.serve_locally = True

app.layout = html.Div([
    html.Label('Color'),
    dcc.Dropdown(
        id='color',
        options=[
            {'label': 'Navy', 'value': '#001f3f'},
            {'label': 'Blue', 'value': '#0074D9'},
            {'label': 'Aqua', 'value': '#7FDBFF'},
            {'label': 'TEAL', 'value': '#39CCCC'},
            {'label': 'OLIVE', 'value': '#3D9970'},
            {'label': 'GREEN', 'value': '#2ECC40'},
            {'label': 'LIME', 'value': '#01FF70'},
            {'label': 'YELLOW', 'value': '#FFDC00'},
            {'label': 'ORANGE', 'value': '#FF851B'},
```

Let us know what you think! We will leave this open for community feedback for a week or two before releasing it.

29 Likes

[Solved] Keep zoom level on interval'd update

Keep zoom, but change axes? uirevision

How to keep selected legend data persistent through callback

Preserve trace visibility in callback

Failing to maintain zoom and position with ScatterMapBox after hover

[New]Allow users to zoom into live graphs and maintain zoom for new data

Retain clicked data when the plot is updated

Save / capture interactive rotate/pan/zoom by user

Mapbox re-centering based on text input not working

How to refresh data on fly?

Updating subplots, without 'refreshing' the whole subplot

ScatterMapbox can't maintain the current zoom area with live update

Rangeselector button callback or persistence wanted

Update 3D surface without resetting translation (zoom, pan, rotation)

How to prevent traces from coming back after being disabled in a live graph

Preserving the zoom of choropleth in Dash

Realtime dashboard - working with maps

Custom hover_data in Choropleth_mapbox and keeping the current zoom after running update_layout

Zoom or pan breaks legendonly

Dash Club - Dispatch #5 - Fall Edition :maple_leaf:

---

chriddyp 2 November 19, 2018, 5:48pm

Also, for reference, here's where the real work happened in this feature 🔧: **https://github.com/plotly/plotly.js/pull/3236**

---

alexcjohnson 3 November 19, 2018, 6:37pm

The usage @chriddyp shows - where you simply specify `layout.uirevision` that changes only when you want to reset any user-driven changes to the plot - covers most cases. In fact, often you can just set it to a truthy constant (`True` , or `'the user is always right'`) and forget about it. That should work as long as you're not replacing the whole data set on the plot. If you *do* replace the whole data set it's important to change `uirevision` though, or the plot could be zoomed into a blank plot after the change, which could be confusing.

There are also cases where you want *more* control. Say the x and y data for a plot can be changed separately. Perhaps x is always time, but y can change between price and volume. You might want to preserve x zoom while resetting y zoom. There are lots of `uirevision` attributes, that normally all inherit from `layout.uirevision` but you can set them separately if you want. In this case set a constant `xaxis.uirevision = 'time'` but let `yaxis.revision` change between `'price'` and `'volume'` . Be sure to still set `layout.uirevision` to preserve other items like trace visibility and modebar buttons!

Trace visibility is a little different: traces are tracked by their `uid` (or by index in `data` if no `uid` is given) - So let's say your plot can have traces for France, Germany, and UK. Initially only Germany and UK are plotted and the user hides Germany (the first trace) using the legend. Then they add France, which becomes the new first trace. Using `uirevision` without `uid` , France is immediately hidden! But if you give each trace a `uid` (`'France'` , `'Germany'` , `'UK'` would do), the `visible: 'legendonly'` flag will follow Germany as it moves to the second trace.

The visibility state of *all* traces is preserved based on `layout.legend.uirevision` . So if you want to re-show *all* traces the user may have hidden by clicking the legend, change `legend.uirevision` . (We do have a `trace.uirevision` , but it only controls a few things like parcoords `constraintrange`)

4 Likes

Use uirevision with make_subplot

---

adi 5 November 22, 2018, 12:16am

@chriddyp This is awesome!
I use dash to make a live streaming app and it will be very handy to preserve the dcc.graph UI state.
Is there anyway to set uirevision to something that would make this constant? (Rather than setting it to something like dataset)

---

chriddyp 6 November 22, 2018, 5:11am

> adi:
>
> Is there anyway to set uirevision to something that would make this constant? (Rather than setting it to something like dataset)

Yeah, so uirevision can be *anything*, the important thing is that when you change it, we reset the graph. So if you never want to reset the graph, just set it to a constant string: *any* constant string, 'foo' or 'static' or 'dash' etc

---

adi 7 November 22, 2018, 9:19am

Awesome, thanks you for clarifying!

QueRico 8 November 22, 2018, 5:19pm

Hello!
That doesn't seem to recognise it on my Dash:

> print(dcc.**version**)

gives me :

> 0.39.0rc4
>
> - Running on **http://127.0.0.1:8050/** (Press CTRL+C to quit)
> - Restarting with stat

So I should be fine. But the piece of code:

```
@app.callback(
Output('Charts','children'),
[Input('DropDown Live Charts', 'value'),
 Input('dataset_storage', 'children')]
)

def update_charts(factor_names,jon_dataset):
# ================================================================================
#     print('live chart data loading time')
#     t0 = time.time()
# ================================================================================

fact_mid_db = pd.read_csv(fact_mid_chart_path,index_col=0,parse_dates = True)
fact_mid_db = fact_mid_db[pricing.index.tolist()]
# ================================================================================
#     print(type(fact_mid_db.index[0]))
# ================================================================================
# ================================================================================
#     t1= time.time()
#     print (t1-t0)
#     t0 = time.time()
# ================================================================================

Charts = []
for factor in factor_names:
    Charts.append(html.Div([
        dcc.Graph(
            figure=go.Figure(
```

Doesn't let me run everything:

```
prop_descriptions=self._prop_descriptions))
ValueError: Invalid property specified for object of type plotly.graph_objs.Layout: 'uirevision'
```

Any idea?

Thank you

---

michaelbabyn 9 November 22, 2018, 6:30pm

@QueRico you are getting that `ValueError` from `plotly.py` because it's validation functions haven't been updated to include the new layout property `uirevision` yet.

For now, replacing `go.Figure` and `go.Layout` with `dict` will prevent this validation problem.

2 Likes

---

chriddyp Split this topic 10 November 26, 2018, 11:03pm

2 posts were split to a new topic: **Preserving state and selected inputs of a previous tab**

---

adi 11 November 28, 2018, 12:47am

Yup can confirm.
I had a bit of difficulty crafting the correct dictionary for Layout, but found it easy to use go.Layout and print that object to the terminal, copy and paste that dictionary and finally edit it.

---

itaidagan 12 December 3, 2018, 12:09pm

This is awesome, wish I had this when I started working on my app (ended up storing the zoom state in a cache between callbacks) but I'm very glad you implemented this. Thank you!

1 Like

---

brad 13 December 5, 2018, 5:20pm

Don't see any comments on the associated PRs. Is there a target release for adding the uirevision prop to dash-core-components?

---

chriddyp 14 December 6, 2018, 12:13am

> brad:
>
> Is there a target release for adding the uirevision prop to dash-core-components?

Good question. For now, use the pre-releases. For this to get part of an official release dash-core-components, we'll need to publish a new plotly.js release and then upgrade `dash-core-components` with the new release. I believe `uirevision` will be part of plotly.js 1.43.0 (**Releases · plotly/plotly.js · GitHub**), this week or next.

---

kimko 16 December 9, 2018, 1:26am

started playing with dash this weekend and loving it so far. Just came accross this problem and the new property works awesome! pretty excited to play with dash more!

1 Like

mifabpre 17 December 14, 2018, 11:07am

Thanks! I was doing sliders to work around the problem, but this is just amazing! It's working mega!

daragon 18 December 14, 2018, 3:56pm

I'm trying to add this uirevision to dcc.graph mapbox layout, but not having any luck. Is there an example of that type of implementation of uirevision? Is this possible yet? Would be a great addition and very important for mapping applications. Cheers

chriddyp 19 December 14, 2018, 4:23pm

> daragon:
>
> Is this possible yet?

This *should* be possible but I personally haven't tried this. Here are the keys that `uirevision` controls: `center`, `zoom`, `bearing`, `pitch` (source code: **plotly.js/src/plots/mapbox/layout_attributes.js at 4f8628f4c7b8f0d604f1f6e7585e962428316cff · plotly/plotly.js · GitHub**).

Could you create a small, reproducable example?

daragon 20 December 17, 2018, 10:59pm

Thank you for the direction, but I've made some additional attempts and am not getting the uirevision to work. Probably something simple I'm missing (hopefully). I've also tried emulating the functionality of the **dash-opioid example** which seems to use the State from dash.dependencies... no luck there either. Here's a simple example though that I'd like to get working. Any further thoughts on uirevision with mapbox is greatly appreciated. Cheers

```
import dash
import dash_html_components as html
import dash_core_components as dcc
from dash.dependencies import Input, Output


# Set mapbox public access token
mapbox_access_token = PROVIDE YOUR TOKEN HERE

app = dash.Dash(__name__)

'''
~~~~~~~~~~~~~~~~
~~ APP LAYOUT ~~
~~~~~~~~~~~~~~~~
'''

app.layout = html.Div([
html.Label('Color'),
dcc.Dropdown(
    id='selected_color',
    options=[
        {'label': 'Navy', 'value': '#001f3f'},
        {'label': 'Blue', 'value': '#0074D9'},
        {'label': 'Aqua', 'value': '#7FDBFF'},
        {'label': 'TEAL', 'value': '#39CCCC'},
    ],
    value='#001f3f'
```

levkach 21 December 27, 2018, 12:44pm

Can you please specify, what should I do, if the dataset is changed, but I want to preserve the zoom?
My example is - I have a 3D graph of frauds/non frauds and a threshold slider bar. I would like to keep the zoom, when i move the threshold, since I might find an interesting area to observe. But the zoom keeps resetting, after I move the slider, even when I've set the `uirevision` property to `True`, because the dataset is changed each time

alexcjohnson 22 December 28, 2018, 11:08pm

@levkach that looks like a bug - thanks! Filed at **https://github.com/plotly/plotly.js/issues/3378**, I suspect it'll be an easy fix.

1 Like

alexcjohnson 23 December 28, 2018, 11:32pm

@daragon looks like we have a bug with `uirevision` and `mapbox` (and `geo`) too - I've included those in the same plotly.js issue. Thanks for the report, we'll get all of these fixed soon!

3 Likes

daragon 24 January 7, 2019, 7:58pm

@alexcjohnson, thanks for the info! For anyone needing a solution in the interim here's a snippet of what worked for me (in the app callbacks):

```
# Update map figure
@app.callback(
        Output('risk-map', 'figure'),
```

```
        [Input('risk-checklist', 'values'),
        Input('structurebasedrisk_dropdown','value'),
        Input('confidence-slider', 'value'),
        Input('colorscale-picker', 'colorscale')],
        [State('risk-map', 'relayoutData')])
# def display_map(values, dropdownvalue, value, colorscale, figure):
def display_map(values, dropdownvalue, value, colorscale, relayoutData):
    cm = dict(zip(BINS, colorscale))

    # Control of zoom and center for mapbox map
    try: # hold existing map extent constant during user interaction
        latInitial = (relayoutData['mapbox.center']['lat'])
        lonInitial = (relayoutData['mapbox.center']['lon'])
        zoom = (relayoutData['mapbox.zoom'])
    except: # incase of using checklist before changing map extent
        latInitial=39.715
        lonInitial=-105.065
        zoom=12
```

As a note, seems that the mapbox object works with relayoutData[‘mapbox.center’][‘lon’] RATHER than [‘mapbox’][‘center’][‘lon’] as in some other examples put out there in the past.
Cheers

1 Like

---

christinab 26 February 8, 2019, 11:45am

Hi all,

I am trying to get some plots working and I also want the zoom and angle to stay fixed when updating the visible data since the user might have zoomed-in out or changed the angle in order to inspect and compare different datasets more closely. I am setting the following as a return from my callback function:

```
return {
    'data': traces,
    'layout': go.Layout(
        margin={'l': 40, 'b': 40, 't': 10, 'r': 10},
        uirevision='same',
        hovermode='closest',
        showlegend=False,
        scene=dict(camera=camera)
    )
}
```

You see how I have set the uirevision to a fixed string “same”. However, the plot is still reset to a fixed position every time I am changing my data. Any ideas what I could do to fix this?

Many thanks,
C.

---

chriddyp 27 February 8, 2019, 2:06pm

> christinab:
>
> Any ideas what I could do to fix this?

What version are you and what chart types do you have inside `traces`?

---

christinab 28 February 8, 2019, 2:23pm

Hi and thanks for the reply!

Dash version is 0.35.1

Traces is a list which has been filled in my callback function as follows:

```
for i in filtered_df:
    traces.append(go.Scatter3d(
        x=filtered_df['x'],
        y=filtered_df['y'],
        z=filtered_df['z'],
        text=filtered_df['Status'],
        mode='markers',
        opacity=0.7,
        name=i,
        marker={
            'size': marker_size,
            'color': filtered_df['color_val'],
            'colorscale': 'Viridis',
            'colorbar': {
                'title': 'Distance from mean',
                'tick0': df['color_val'].min(),
                'dtick': df['color_val'].max()
            }
        }
    ))
```

and filtered_df is a pandas DataFrame

Cheers,
C.

chriddyp 29 February 12, 2019, 8:24pm

> christinab:
>
> Dash version is 0.35.1

It looks like this was fixed in plotly.js 1.43.2 (**Release v1.43.2 · plotly/plotly.js · GitHub**). The latest version of `dash-core-components` bumped plotly.js to `1.44.3` (**dash-core-components/CHANGELOG.md at master · plotly/dash-core-components · GitHub**), so I recommend upgrading to the latest. Doing a `pip install dash --upgrade` should do the trick.

---

christinab 30 February 13, 2019, 12:43pm

Upgrading got uirevision working!

Many thanks:)
C.

---

chriddyp Split this topic 31 February 14, 2019, 5:19pm

A post was split to a new topic: **Retrieve state of graph after legend update**

---

lanster99 32 March 8, 2019, 3:11pm

Hi guys,

This is a really cool feature and very useful as I have made company dashboards that update every 30 seconds as they query a database to update.

For the life of me I cannot get uirevision working. I have the latest versions of Dash and Plotly and all the related packages.

I believe the issue is the fact that I cannot specify uid, so every 30 seconds it redraws the same traces with new uids, rendering uirevision useless. In the above examples no one specifies uid it just works, I cannot figure out what makes my case different.

Can post code if helpful.

How do I specify the uid of my traces?

Thanks!

Pommespapst 33 March 19, 2019, 1:49pm

I cannot get uirevision to work with 3D geo-type graphs. Both rotation and zoom are reset after callback execution although the revision property is never changed. I am using dash version 0.39 and plotly version 3.7.1

My code:

```
import dash
import plotly
import dash_html_components as html
import dash_core_components as dcc

# make app
app = dash.Dash(__name__)

# one marker
data = [plotly.graph_objs.Scattergeo(
    lat = [40.7127],
    lon = [-74.0059],
    hovertext="test"
)]

# layout = globe
layout = dict(
    geo=dict(
        projection = dict( type='orthographic' ),
        uirevision = False
    )
)

# graph = globe with one marker
graph = dcc.Graph(
    id='graph',
    figure=dict(data=data, layout=layout),
)
```

Pommespapst 34 March 20, 2019, 10:55am

Furthermore, workarounds using “relayout” data do not work. For some reason, the relayout state data contains no zoom information. It contains rotation data only.

---

mbkupfer 35 April 10, 2019, 4:41pm

I can’t quite understand how to get the legend selections to not reset after changing the trace.

Here is how I currently have it setup:

```
# for loop with logic to create each trace

    traces.append(go.Bar(
            x=x,
            y=y,
            uid=uid
            visible='legendonly'
        ))
    layout = deepcopy(BASE_LAYOUT)
    layout['legend']['uirevision'] = True
    return go.Figure(data=traces, layout=layout)
```

But this just results in all my traces becoming invisible at initial startup and resetting back to invisible for each new figure.

Also, setting trace uid does nothing for me, or at least it looks like it is just setting it to some random sequence (ex: `uid: "c5fff344-7522-4ece-b17d-9fd474c7f4a1"`) and not what I set it to.

---

alexcjohnson 36 April 10, 2019, 5:49pm

@mbkupfer It does seem like something funny is happening with `uid` in `plotly.graph_objs` - your example works fine if I bypass it:

```
from random import random
import dash
import dash_core_components as dcc
import dash_html_components as html
from dash.dependencies import Input, Output
import plotly.graph_objs as go

app = dash.Dash(__name__)
app.layout = html.Div([
    html.Button(
        id='button',
        children='Update Data',
        n_clicks=0
    ),
    dcc.Graph(id='graph')
])


@app.callback(Output('graph', 'figure'), [Input('button', 'n_clicks')])
def update_graph(n_clicks):
    traces = [dict(
        type='bar',
        x=range(10),
        y=[random() for _ in range(10)],
        uid=str(i),
        visible='legendonly'
    ) for i in range(5)]
```

---

alexcjohnson 37 April 10, 2019, 5:58pm

@Pommespapst I missed your question when you asked it, so I hope you already solved this! But for anyone else looking at this now, it seems to work fine if I just set a truthy `uirevision`. `False` (or anything else falsy like missing/ `None`) will cause the user interaction state to be ignored when an updated figure is provided.

---

alexcjohnson 38 April 10, 2019, 9:01pm

Thanks @mbkupfer for bringing this up - we're working on a fix over at **https://github.com/plotly/plotly.py/issues/1512** but in the meantime you can construct your figure with a plain dict. I suspect only the top level matters `dict(data=traces, layout=layout)` instead of `go.Figure(data=traces, layout=layout)` but I haven't tested that.

1 Like

---

mbkupfer 39 April 10, 2019, 9:23pm

Using `dict(data=traces, layout=layout)` works! Thanks for the big help on this

By the way, not sure what you meant here

> But if you give each trace a `uid` (`'France'` , `'Germany'` , `'UK'` would do), the `visible: 'legendonly'` flag will follow Germany as it moves to the second trace.

Using `legendonly` resulted in my app starting with all traces being hidden. Removing this from my traces all together seemed to do what I was expecting. Am I misinterpreting something in your explanation?

---

mbkupfer 40 April 10, 2019, 9:34pm

@alexcjohnson, another issue I just noticed, which I'm not sure is related or not, is that changing `dcc.Tabs()` results in a the legend traces getting reset. I can create a reproduceable example, but not at the current moment. Either way, I thought I would at least put this out there and see if you had any hunches at to what is going on.

Pommespapst 41 April 11, 2019, 8:53am

> it seems to work fine if I just set a truthy `uirevision` . `False` (or anything else falsy like missing/ `None`) will cause the user interaction state to be ignored

This did the trick! Thank you so much

1 Like

alexcjohnson 42 April 11, 2019, 2:32pm

@mbkupfer here's what I meant with the countries:

```
from random import random
import dash
import dash_core_components as dcc
import dash_html_components as html
from dash.dependencies import Input, Output

app = dash.Dash(__name__)
app.layout = html.Div([
    dcc.Dropdown(
        id='countries',
        multi=True,
        options=[
            {'label': 'France', 'value': 'France'},
            {'label': 'Germany', 'value': 'Germany'},
            {'label': 'UK', 'value': 'UK'}
        ],
        value=['France', 'Germany', 'UK']
    ),
    dcc.Graph(id='graph')
])


@app.callback(Output('graph', 'figure'), [Input('countries', 'value')])
def update_graph(countries):
    traces = [dict(
        type='bar',
        x=range(10),
        y=[random() for _ in range(10)],
```

By using the country as `uid` , you can hide the 2nd or 3rd trace from the legend, then delete the first trace from the dropdown, and the same *country* will still be hidden in the legend. Without `uid` it would be the same *trace index* that was hidden. Note though that if you delete a hidden trace via dropdown then later bring it back, we'll lose memory that you had previously hidden it.

As to changing tabs resetting the UI - I'm afraid that's expected. plotly.js stores this UI state on the DOM element and when you switch to a different tab that DOM element is destroyed, only to be recreated when you come back. I suppose you could pre-render all the tabs and just hide the inactive ones, rather than removing them from the DOM; that might be bad for initial load time, though if done right (just using a multi-output on `style` props?) it could be good for switching performance. Other than that all I can think of is manually tracking these changes via `restyleData` etc... but the pain and complication associated with that is why we added `uirevision` in the first place!

2 Likes

mbkupfer 43 April 11, 2019, 3:07pm

> alexcjohnson:
>
> here's what I meant with the countries:

I see what you mean. My main confusion came from the `visible: legendonly` part as I thought we had to set it, but it appears from your example that this is not the case and that it is done implicitly.

As for the tabs part, thanks for clarifying what is going on with the DOM. Would it make a difference though if I'm **not** using callbacks to generate my tabs, rather they are all created decoratively in the children properties of each tab. Wouldn't they already be pre-rendered in this case, or is there more abstraction to how `dcc.Tabs` works?

alexcjohnson 44 April 11, 2019, 3:20pm

Ah I actually hadn't looked in detail at how we implemented `dcc.Tabs` but it turns out only the selected tab is rendered, so previously-visible tab contents are destroyed and removed from the DOM. Might be interesting to investigate adding a mode for this, something like `Tabs.inactive_mode: 'destroy' | 'hide'` , but it's not available now.

3 Likes

tusharsadana 45 June 4, 2019, 2:18pm

hey alexcjhonson, I was working with plotly in angularjs, can you help me with applying this uirevision in that?

alexcjohnson 46 June 4, 2019, 4:42pm

AFAIK there's nothing different about how this will work in Angular - provide an unchanging (but truthy) `uirevision` if you want user-initiated changes to persist across redraws, and a different value when you want to reset them. But I haven't used it in Angular myself. Is there a specific problem you're encountering?

---

tusharsadana 47 June 5, 2019, 9:46am

I am not able to find how to apply uirevision in angular, it is all in react. I am a beginner and having a hard time preserving the state of the graph

---

alexcjohnson 48 June 6, 2019, 1:44pm

Are you using **angular-plotly.js**? That wrapper uses the `Plotly.react` method to update a plot, so it should work fine with `uirevision`. The `react` method doesn't have anything to do with the React framework other than sharing a name.

For the most part, all you need to do is set the `uirevision` attribute of the plot's `layout`. Set it to something constant and truthy, and you should see user-initiated zoom/pan etc preserved when you update the plot. Once you have that working, if you need more fine-grained control you can explore changing `layout.uirevision` and/or using the derived versions like `layout.yaxis.uirevision` etc.

---

vararp 49 June 14, 2019, 6:34am

@alexcjohnson i am trying to plot a graph where the data points for the graph are fetched from database at a interval of 2 second and that are updated on graph, so that we can see that data point moving(data points are actually the lat,lon of cars) making it near to real time.
The problem is that everytime when the data points are updates the window is also refreshed, means the window is refreshed at every 2 second and hence we can't zoom, pan the window as it will go to the same state after 2 second.
Is there a way in which the UI state is preserved even when the data points are getting updated?
i tried uirevision, but it's not working.

---

alexcjohnson 50 June 24, 2019, 7:49pm

@vararp that's exactly the kind of situation `uirevision` is meant for - you should be able to pick some constant value and provide that same value as `layout.uirevision` on every update. If that's not working, can you post some code so we can get to the bottom of it?

---

vararp 51 June 27, 2019, 11:30am

@alexcjohnson here the data is getting updated every 2 second and i want the UI state to be preserved when data getting updates, can you help me regarding this. I am new to plotly actually, so don't have that much idea.

```
data = [
go.Scattermapbox(
lat = df.lat, lon = df.lon,
mode='markers',
marker=dict(
size=12,
color= df.status
),
showlegend = False
)
]
layout = go.Layout(
autosize=True, #width = 1400, height = 900,
margin = {'l': 0,'r': 0,'t': 0,'b': 0 },
hovermode='closest',
mapbox=dict(
accesstoken=mapbox_access_token,
bearing=0,
center=dict(
lat=17.9716,
lon=57.5946
),
pitch=0,
zoom=11
),
)
app.layout = html.Div(children=[
html.H1('xyz'),
# html.Div(style={ "height" : "100vh"}),
html.Div('"
updates after every 2 second
'"),
dcc.Graph(
id='bike-status-graph',
figure={
'data': data,
'layout': layout
},
animate = True,
style={'height': '85vh'}
),
dcc.Interval(
id='interval-component',
interval=2*1000, # in milliseconds
n_intervals=0
)
])
@app.callback(Output('bike-status-graph', 'figure'),
[Input('interval-component', 'n_intervals')])
def update_graph_live(n):
```

```
df = get_busy_bikes()
data = [
go.Scattermapbox(
lat = df.lat, lon = df.lon,
mode='markers',
marker=dict(
size=6,
color=df.status
),
)
]
fig={
'data':data,
'layout':layout
}
return fig
if name == 'main':
app.run_server(host='0.0.0.0', debug=True)
```

---

alexcjohnson 52 June 27, 2019, 12:00pm

Try:

```
layout = go.Layout(
    uirevision=True,
    autosize=True,
    ...
)
```

1 Like

---

vidapura 53 June 28, 2019, 11:46am

Hi,
I'm also new to Dash and also having trouble with losing the zoom when the data gets updated after an interval...

Can I please ask how I do this uirevision thing , just to stop losing the zoom ?

My code is like this:

```
app.layout = html.Div(children=[
    dcc.Graph(id='basic-interactions'),
    dcc.Interval(
        id='interval-component',
        interval=5 * 1000,  # in milliseconds
        n_intervals=0
    ),
```

So the post above is confusing as I don't understand what is go.Layout or what is the difference??

Also I'm not sure if I'm supposed to se uirevision=True in the layout or in the return from the callback ?

Thanks in advance.

---

alexcjohnson 54 June 28, 2019, 7:38pm

> vidapura:
>
> Also I'm not sure if I'm supposed to se uirevision=True in the layout or in the return from the callback ?

Whenever you construct a `figure` for a `dcc.Graph`, put `uirevision=True` in `figure['layout']`.

The way you have it in your post there's no `figure` in the graph in `app.layout`, so I guess it'll just be in the callback return.

---

vidapura 55 July 1, 2019, 9:09am

Hi Alex,
Thanks for taking the time to try to set me straight...

I'm very new to this so I'm cobbling together code from the examples as best I can...

My callback returns the following...

```
graphdata = {
        'data': traces,
        'layout': go.Layout(
            yaxis={'title': '# my title'},
            legend=dict(
                    x=1,
                    y=1,
                    traceorder='normal',
                    font=dict(
                        family='sans-serif',
                        size=12,
                        color='#000'
                    ),
                    bgcolor='#E2E2E2',
                    bordercolor='#FFFFFF',
                    borderwidth=2
```

```
                ),
            hovermode='closest',

        )
    }
```

I note now that there is no figure in it at all... so ... am I doing it wrong then?

Thanks

---

alexcjohnson 56 July 10, 2019, 9:18pm

That all looks reasonable - the `{data, layout}` dict is the `figure`. Then I'm assuming your callback is something like:

```
@app.callback(
    Output('basic-interactions', 'figure'),
    [Input('interval-component', 'n_intervals']
)
def make_figure(n):
    graphdata = ...
    # set graphdata['layout']['uirevision'] = True
    return graphdata
```

---

crosenth 57 July 12, 2019, 1:21pm

Great feature!

I noticed that in order to preserve the "view" between callbacks the user MUST update the figure or else the "view" will be reset no matter what uirevision is set to. Using chriddyp's example above, if I change the Reference from Display to Hide without zooming, etc, the "view" still resets. Is that by design or is there some way to prevent the "view" reset without necessarily updating the figure?

Hooman 58 October 23, 2019, 6:29am

Hello
I have got the code you can find it below, the data generate b y random function and i can see visually a live trend, the aim is that, user can zoom in and finnd out the details, i have used Uirevision, but it does not work. can you help me how to fix it.

app.layout = html.Div([
dcc.Graph(id='live-graph', animate=False,
figure=go.Figure(
data=[
go.Scatter(

x=data_DB['time'],
y=data_DB['value'],

```
                )
            ],
            layout=go.Layout(
                xaxis={#'range': ['13:00:00', '14:00:00'],
                       'autorange': plot_data['Auto_scale'],
                       'tickmode': 'auto',
                       'nticks': 4,
                       },
                yaxis={#'range': [0 , 1],
                       'autorange': plot_data['Auto_scale'],
                       'tickmode': 'auto',
                       'nticks': 3,
                       },
                       uirevision= 'refresh'
            )
        )

        ),
dcc.Interval(
    id='graph-update',
    interval=2 * 1000,
    n_intervals=0
),
html.Label('Dropdown'),
dcc.Dropdown(id='Dropdown-list',
             options=list(Drop_down_option),
             value='',
             multi=True
```

], style={'columnCount': 1})

@app.callback(dash.dependencies.Output('refresh', 'n_clicks'), # update data from OPC to Plot data
[dash.dependencies.Input('Dropdown-list', 'value'),
dash.dependencies.Input('graph-update', 'n_intervals')
],
[dash.dependencies.State('refresh', 'n_clicks')],
)
def data_update(names,n,state):
sim_add_DB()
if plot_data['names']==names:
if plot_data['Live_view']:
print('add data')
sim_add_live() # change to function that would add data from OPC ----------

return 1
else:
if len(names) > 0:
print('new data')
plot_data['names'] = names
if plot_data['Live_view']:
a=1
# later add function that would add data from data base that other tags have ----------
return 1
else:
sim_data_base() # change to function that would read data from data base ----------
return 1
else:
plot_data['names'] = []
a=2
return state

@app.callback(#redraw graph
Output('live-graph', 'figure'),
[Input('refresh', 'n_clicks')],
[State('live-graph', 'figure')]
)
def update_graph(n_clicks,fig,refresh):
figure = go.layout{
data=[
go.Scatter(
x=[0],
y=[0]
)
],
Layout_plot = #go.Layout(
xaxis={'autorange': plot_data['Auto_scale'],
'tickmode': 'auto',
'nticks': 9,
},
yaxis={'autorange': plot_data['Auto_scale'],
'tickmode': 'auto',
'nticks': 5,
},
uirevision= refresh
)}

```
if n_clicks != None:
    print('refreshed')
    i = 0
    if len(plot_data['names']) > 0:
        while i < len(plot_data['names']):

            data_name = 'Data' + str(i)
            if (i == 0):
                figure = go.Figure(
                    data=[
                        go.Scatter(
                            x=list(plot_data[data_name]['time']),
                            y=list(plot_data[data_name]['value']),
                            name = plot_data['names'][i]
                        )
                    ],)

            elif plot_data['names'][i] != []:
                figure.add_trace(go.Scatter(
                    x=list(plot_data[data_name]['time']),
                    y=list(plot_data[data_name]['value']),
                    name=plot_data['names'][i]
                ))
            i = i + 1
        a=1
```

return figure

```
return {
    'data' : figure['data'],
    'Layout' :Layout_plot
}
```

if **name** == '**main**':
#app.run_server(debug=True)
server.run()

---

dhhagan 59 November 9, 2019, 9:10pm

So...it's been a while since the original post. Is there a reason this isn't documented in the **primary docs**? I've updated to following versions:

```
dash==1.6.0
dash-core-components==1.5.0
dash-daq==0.2.2
dash-html-components==1.0.1
dash-renderer==1.2.0
dash-table==4.5.0
```

Still seems that the `uirevision` argument isn't recognized:

```
Failed component prop type: Invalid component prop `figure` key `uirevision` supplied to Graph.
```

Any thoughts on what could fix this?

---

chriddyp 60 November 10, 2019, 3:08am

uirevision should be in figure.layout, not figure

HansPeter123 61 November 15, 2019, 7:02pm

Hi there, I have only seen UI revision being implemented in Python. Is it possible to do the same in R? If that is the case, some suggestions would be greatly appreciated 🙂 Thanks.

Map Zoom's back, after a callback occurs

---

alexcjohnson 62 November 18, 2019, 4:03pm

Yep, `uirevision` is implemented entirely on the javascript side, so it'll work just as well from R as from Python. Just add `uirevision=TRUE` to your graph's `figure$layout`.

1 Like

HansPeter123 63 November 18, 2019, 6:03pm

Thank you 🙂

---

tks 64 November 26, 2019, 6:03pm

If I have a figure with a `go.Heatmap` trace plus some other overlayed traces, how can I make the heatmap always visible while still allowing the user to toggle the visibility of the other traces via the clickable legend? The unwanted scenario is when the heatmap gets hidden when another trace gets double-clicked to show that other trace in isolation. Thanks!

---

alexcjohnson 65 November 26, 2019, 6:57pm

Thanks for bringing this up! I'd call that behavior a bug -> **https://github.com/plotly/plotly.js/issues/4389**

---

chriddyp Split this topic 66 January 6, 2020, 9:26pm

A post was split to a new topic: **Uirevision and animation slider**

---

balagdivya 67 May 12, 2020, 5:21pm

Hi @alexcjohnson I'm facing the same issue. I'm using uirevision to preserve the UI state in graph. I am also using dcc.Tabs... When I change the tab and come back to the original tab, the graph is getting reset to the default state. Could you please help me in this?

1 Like

---

alexcjohnson 68 May 12, 2020, 7:08pm

@balagdivya I don't know that we have a good solution for this use case yet but it's come up before. There may be a way by not actually removing the graph when you switch tabs, just hiding it via CSS. But that could be really awkward depending on how many tabs you have and how complex they are. I've made an issue to look into a natural solution **https://github.com/plotly/dash-core-components/issues/806**

---

balagdivya 69 June 16, 2020, 10:23am

Hi @alexcjohnson , any update on the issue **https://github.com/plotly/dash-core-components/issues/806** ??

---

chriddyp Split this topic 70 June 16, 2020, 8:05pm

A post was split to a new topic: **Trigger callback when legend is clicked**

---

jvgomez 71 June 30, 2020, 11:19am

Hey @crosenth , what do you mean that "the user MUST update the figure"? I am having the issue where I am updating some traces of the figure, but the view still resets to where it was when I triggered the callback the first time the callback was called.

---

crosenth 72 June 30, 2020, 3:31pm

Hi @jvgomez , what I meant to say is the user must “interact” the figure in order to preserve . This feature is meant to preserve the user’s interactions. If you are adding traces to the plot it will reset unless you also interact (Zoom in, out, autoscale, etc).

There used to be a way to preserve a plots’ axes without interaction as described in this Issue (**https://github.com/plotly/dash-core-components/issues/321**) but I suspect it was accidentally squashed by a PR back in 2018.

---

jvgomez 73 June 30, 2020, 3:47pm

@crosenth Not sure I follow. I have the case for example that I manually move the camera and click on the figure that triggers a callback that modifies one of the existing traces. So far so good.

Then I move the camera again, I click again on the figure to trigger the callback, and the camera resets to the prose it had on the previous click. So I understand user interactions are not preserved. Is this supposed to be fixed with `uirevision`? Because i try and didn’t work.

---

chriddyp 74 June 30, 2020, 6:16pm

@jvgomez - That should work, but maybe there is an issue with 3D interactions. Any chance you create a really simple, small, reproducible example with dummy data? That’ll help us create the bug report.

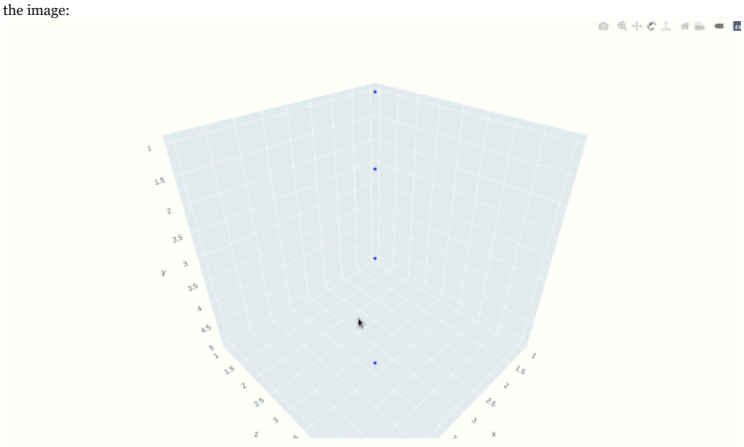
---

jvgomez 75 July 1, 2020, 11:42am

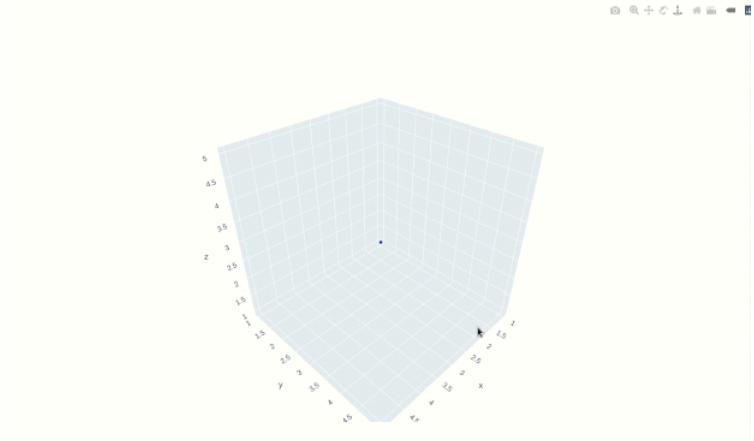
@crosenth I tried to reproduce the issue with the following code which represents my real application. Basically, a callback creates a figure and updates it. This callback is triggered when we click in a point on the figure and updates one of the traces to highlight the point clicked.

```
import json

import dash
import dash_core_components as dcc
import dash_html_components as html
from dash.dependencies import Input, Output, State

import plotly
import plotly.graph_objects as go


external_stylesheets = ['https://codepen.io/chriddyp/pen/bWLwgP.css']
app = dash.Dash(__name__, external_stylesheets=external_stylesheets)
data = [1,2,3,4,5]

# Layout
app.layout = html.Div([
    # Invisible div for data caching
    html.Div(id='point-data', style={'display': 'none'}),

    # 3D  graph div
    html.Div([
        dcc.Graph(id='3d-graph', style={'height': '60vh'})
    ]),
])

# Callbacks
@app.callback(Output('3d-graph', 'figure'),
```

So I realized that the default camera is causing the issue. The code above produces strange behavior of resetting camera to a previous position (I think it is the camera on the first callback), as shown in the image:



If you remove the custom layout when creating the figure, the behavior seems to be mostly OK. There are still small issues such as the first time the figure updates the camera returns to the initial position, or when zoom is changed, the camera readjusts it a bit when updated:



For my application the camera change is relatively important, any suggestion on how to deal with it?

Thanks!

1 Like

---

pherrman 76 July 8, 2020, 8:55pm

Hello everyone,

I am having an issue with the uirevision parameter in my live chart. I believe I am using it correctly however I cannot maintain zoom or panning. every other second the UI resets back to the default and I lose zoom and chart position. Any suggestions are much appreciated. My end goal is to have a live candlestick chart similar to trading view 🙂
thank you

```
import dash
from dash.dependencies import Output, Input
import dash_core_components as dcc
import dash_html_components as html
import plotly
import plotly.graph_objs as go
from collections import deque
from binance.client import Client
from binance.websockets import BinanceSocketManager
import time
from datetime import datetime
import pandas as pd


data = pd.DataFrame(columns=['Open', "High", "Low", "Close", 'Date'])
y_candle = []
index = 0

app = dash.Dash(__name__)
app.layout = html.Div([dcc.Graph(id='live-graph', animate=True, style={'height': '1080px'}),
                       dcc.Interval(id='graph-update', interval=1000, n_intervals=0)])


@app.callback(Output('live-graph', 'figure'),
              [Input('graph-update', 'n_intervals')])
```

---

nicolaskruchten 77 August 5, 2020, 7:48pm

I believe this is a bug in Plotly.js, or a combination of multiple bugs. I've logged them as **https://github.com/plotly/plotly.js/issues/5050** and **https://github.com/plotly/plotly.js/issues/5005** and **https://github.com/plotly/plotly.js/issues/5004**.

1 Like

---

pherrman 78 August 7, 2020, 4:00am

Ah ok I appreciate the feedback.

Thank you

---

chriddyp Split this topic 79 September 24, 2020, 7:56pm

A post was split to a new topic: **Vue & uirevision**

---

sislam1312 80 February 18, 2021, 5:35am

Hi Everyone,

I am building a streaming application. So, every 5 minutes my data is refreshed in the database and I am pulling the updated data from the database in the dash application. So, my customer wants to update the data tables but not the graphs because they will be seeing a trend if there are abnormal behaviour in the lines.

I have tried using uirevision but it not working. Can someone help in this? Thanks in advance

```
#call back for All Plating Lines

@app.callback(
    dash.dependencies.Output('table', 'children'),
    [dash.dependencies.Input('lineSelect', 'value'),
    dash.dependencies.Input('refresh-table', 'n_intervals'),])

def filter_table(input,num):
    if num == 99999999999:
        raise PreventUpdate
    else:
        print ("The code is inside the function")
        df2 = pd.read_sql_query(query_3, engine)
        df2['location'] = df2['location'].replace(regex='AmpsActual',value='')
        df2['line'] = np.where(df2.location.str[3] == '1' ,df2.line+' 1',df2.line)
        df2['line'] = np.where(df2.location.str[3] == '2' ,df2.line+' 2',df2.line)

        tableData = df2.pivot_table('alarmflag',['line','linespeed'],'location',aggfunc = 'first')
        tableData = pd.DataFrame(tableData.to_records())
        tableData.columns = [hdr.replace("('alarmflag',","").replace(")","") for hdr in tableData.columns]

        td1 = tableData
        td1['linespeed'] = td1['linespeed'].round(1)
        td1['linespeed'] = td1['linespeed'].astype(str)


        for j in td1.columns:
```

M4rk 82 July 4, 2022, 6:27pm

Hi all,
I am also facing a problem with uirevision. In my application, which is based on a 3D scatter plot, I set uirevision to preserve the UI state but I found a case where it does not work: when I click on a legend trace, the camera change to a prior position.

Any idea of I can fix that?

Thank you.

---

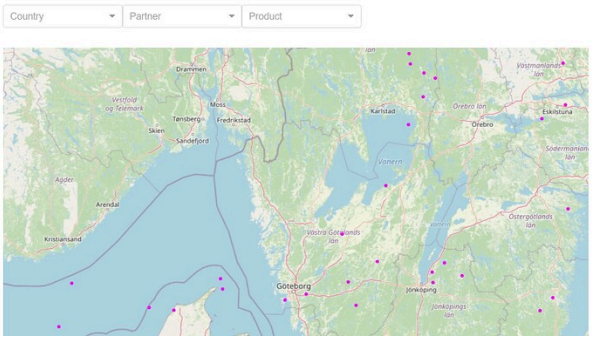
parvizalizada 83 February 10, 2023, 10:27pm

UPDATE: Solved this with different syntax for map figure.

hi @chriddyp
I was looking for this functionality but I'm not sure if I understand the syntax. I have a map with points on it. These points are filtered based on three dropdown filters. Each time I apply filtering, the map centers to initial position. I'm attaching part of the code that shows dropdown items, map, and callback. I'm wondering where I should have uirevision in the code.

```
html.Div([
            dcc.Dropdown(
                df['country'].unique().tolist(),
                placeholder='Country',
                id='country-filter',
                multi=True,
            ),
        ],
        style={'width': '15%','float': 'left', 'display': 'inline-block'}),

        html.Div([
            dcc.Dropdown(
                df['partner'].unique().tolist(),
                placeholder='Partner',
                id='partner-filter',
                multi=True,
            ),
        ], style={'width': '15%', 'float': 'left', 'display': 'inline-block'}),

        html.Div([
            dcc.Dropdown(
                df['product'].unique().tolist(),
                placeholder='Product',
                id='product-filter',
                multi=True,
            ),
        ], style={'width': '15%', 'float': 'left', 'display': 'inline-block'})
```

Dash app screenshot:



I have

dash=2.8.1
dash-core-components=2.0.0

---

matteo.bastico 84 December 4, 2023, 1:21pm

Hello, I have the same small issues using `uirevision` on `scatter3d` plots, such as the camera returning to the initial position after the first update (only the first time) and the zoom changing a bit when the camera readjusts. Any suggestion on how to fix them?

---

plotter2 85 December 19, 2023, 3:18pm

Hey @alexcjohnson i've been unable to make uirevision to work while using the orthographic projection on 3D scatter plot using graph object.
here's a simplified example :

```
import dash
from dash import dcc, html
from dash.dependencies import Input, Output
import plotly.graph_objs as go
import pandas as pd
from numpy.random import randn
# Sample data
data = {
    'sepal_width': [3.5, 3.0, 3.2, 3.1, 3.6],
    'sepal_length': [5.1, 4.9, 4.7, 4.6, 5.0],
    'petal_length': [1.4, 1.4, 1.3, 1.5, 1.4],
}
data2 = {
    'sepal_width': [3.5, 3.0, 3.2, 3.1, 3.6]*3,
    'sepal_length': [5.1, 4.9, 4.7, 4.6, 5.0]*2,
    'petal_length': [1.4, 1.4, 1.3, 1.5, 1.4]*1,
}
df = pd.DataFrame(data)

# Initial Figure
scatter_figure = go.Figure()
scatter_figure.layout.uirevision = True
scatter_figure.layout.scene.camera.projection.type = 'orthographic'
# Add 3D Scatter Plot Trace
scatter_figure.add_trace(
    go.Scatter3d(
        x=df['sepal_width'],
        y=df['sepal_length'],
```

every time im toggling the check box the view completely reversed to its original position, and only when im specifing projection to be orthographic , works perfectly on perspective, am i missing something?
thanks in advance

---

alexcjohnson 86 January 4, 2024, 6:04pm

> plotter2:
>
> and only when im specifing projection to be orthographic , works perfectly on perspective

(Sorry, lost track of this during the holidays - Happy New Year!)

That sure sounds like a bug to me, would you mind creating a **new plotly.js issue** and including the app you posted above? We'll try to take a look.

---

plotter2 87 January 11, 2024, 9:01am

hey , I'm not really familiar with plotly.js.
I will add the app and open an issue under the link you shared , Happy New Year !

Related topics

Topic	Replies	Activity
Uirevision between a callback and a clientside callback	0	May 20, 2020
UIRevision on a 3d dcc.Graph doesn't work the first time	0	June 3, 2020

Topic	Replies	Activity
Uirevision not working as intended	3	October 28, 2024
[New]Allow users to zoom into live graphs and maintain zoom for new data show-and-tell	2	March 4, 2021
Retain pan/zoom across live updates	2	December 14, 2019