@app.callback(  
 output=Output("metric-rows", "children"),  
 inputs=[Input("port-store", "value"),  
 ],  
)  
def update\_datarows(ports, data):  
 return [generate\_metric\_row\_helper(port) for port in ports]

1. 在port列表更改后，添加新交换机的port项

@app.callback(  
 output=Output('dummy-output', 'children'),  
 inputs=Input('interval-component', 'n\_intervals'),  
 state=State('port-store', 'data'),  
)  
def dynamic\_callback\_creation(interval, stored\_data):  
  
 for param in stored\_data:  
 update\_param\_row\_function = create\_callback(param)  
 app.callback(  
 output=[  
 Output(param + suffix\_count, "children"),  
 Output(param + suffix\_sparkline\_graph, "extendData"),  
 Output(param + suffix\_ooc\_n, "children"),  
 Output(param + suffix\_ooc\_g, "value"),  
 Output(param + suffix\_indicator, "color"),  
 ],  
 inputs=[Input("interval-component", "n\_intervals")],  
 state=[State("value-setter-store", "data")],  
 allow\_duplicate=True,  
 )(update\_param\_row\_function)  
  
 return "Dynamic callbacks created"

1. 时序更新，根据当前存储的数据，更新port项的内容，具体为：  
   1. suffix\_count：端口号  
   2. suffix\_sparkline\_graph：迷你图  
   3. suffix\_ooc\_n：

4. suffix\_ooc\_g:

5. suffix\_indicator:

# decorator for list of output输出更新  
def create\_callback(param):  
 def callback(interval, stored\_data):  
 count, ooc\_n, ooc\_g\_value, indicator = update\_count(  
 interval, port, stored\_data  
 )  
 spark\_line\_data = update\_sparkline(interval, param)  
 return count, spark\_line\_data, ooc\_n, ooc\_g\_value, indicator  
  
 return callback

1. 为2的回调内容，更新内容为：  
   1. suffix\_count： count  
   2. suffix\_sparkline\_graph：spark\_line\_data  
   3. suffix\_ooc\_n：ooc\_n

4. suffix\_ooc\_g: ooc\_g\_value

5. suffix\_indicator: indicator

其中spark\_line\_data = update\_sparkline(interval, param)

1. update\_count输出设定为：  
   # port, time, rate, occ, color