Dash DAQ

Dash is a web application framework that provides pure Python abstraction around HTML, CSS, and JavaScript.

Dash DAQ comprises a robust set of controls that make it simpler to integrate data acquisition and controls into your Dash applications.

The source is on GitHub at **plotly/dash-daq**.

These docs are using version 0.6.0.

```
>>> import dash_daq as daq
>>> print(daq.__version__)
0.6.0
```

BooleanSwitch

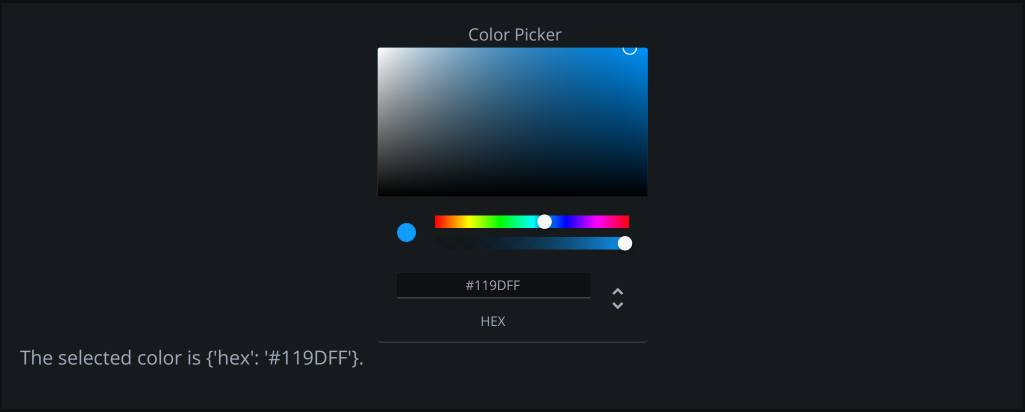
```
from dash import Dash, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.BooleanSwitch(id='our-boolean-switch', on=False),
    html.Div(id='boolean-switch-result')
])


@callback(
    Output('boolean-switch-result', 'children'),
    Input('our-boolean-switch', 'on')
)
def update_output(on):
    return f'The switch is {on}.'


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

The switch is False.

More BooleanSwitch Examples and Reference

ColorPicker

```python
from dash import Dash, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.ColorPicker(
        id='our-color-picker',
        label='Color Picker',
        value=dict(hex='#119DFF')
    ),
    html.Div(id='color-picker-result')
])


@callback(
    Output('color-picker-result', 'children'),
    Input('our-color-picker', 'value')
)
def update_output(value):
    return f'The selected color is {value}.'


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



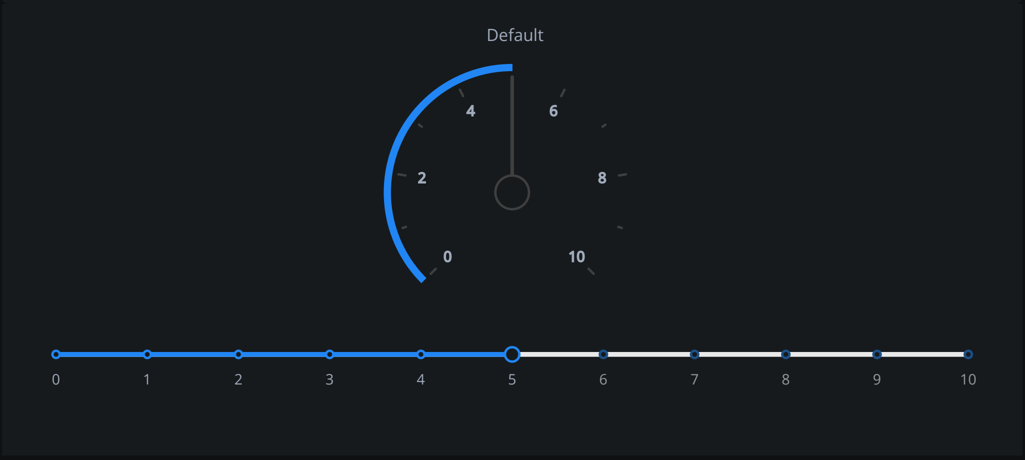
The selected color is {'hex': '#119DFF'}.

More ColorPicker Examples and Reference

Gauge

```python
from dash import Dash, dcc, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.Gauge(
        id='our-gauge',
        label="Default",
        value=6
    ),
    dcc.Slider(
        id='our-gauge-slider',
        min=0,
        max=10,
        step=1,
        value=5
    ),
])


@callback(Output('our-gauge', 'value'), Input('our-gauge-slider', 'value'))
def update_output(value):
    return value
```

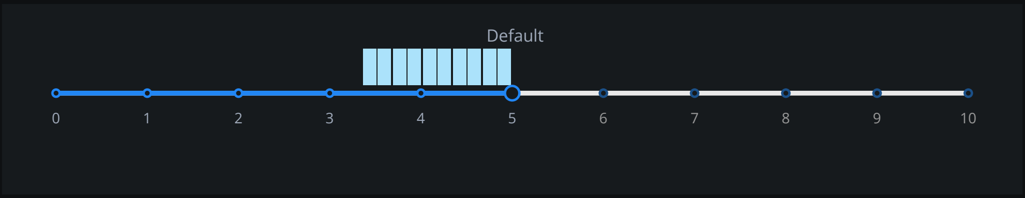
```
if __name__ == '__main__':
    app.run(debug=True)
```



More Gauge Examples and Reference

GraduatedBar

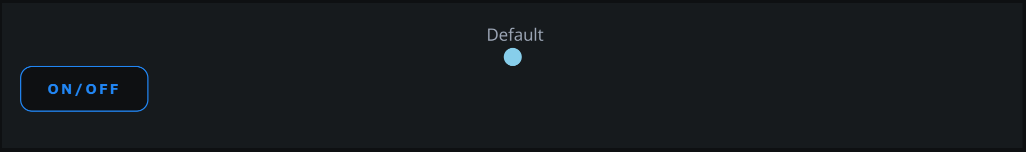
```
from dash import Dash, html, dcc, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.GraduatedBar(
        id='our-graduated-bar',
        label="Default",
        value=6
    ),
    dcc.Slider(
        id='our-graduated-bar-slider',
        min=0,
        max=10,
        step=1,
        value=5
    ),
])


@callback(
    Output('our-graduated-bar', 'value'),
    Input('our-graduated-bar-slider', 'value')
)
def update_output(value):
    return value


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



More GraduatedBar Examples and Reference

Indicator

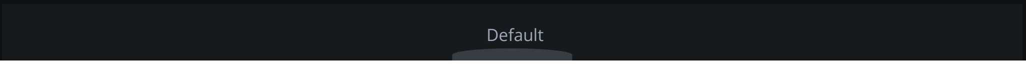
```
from dash import Dash, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.Indicator(
        id='our-indicator',
        label="Default",
    ),
    html.Button(
        'On/Off',
        id='our-indicator-button',
        n_clicks=0
    ),
])


@callback(
    Output('our-indicator', 'value'),
    Input('our-indicator-button', 'n_clicks')
)
def update_output(value):
    return True if value % 2 == 0 else False


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

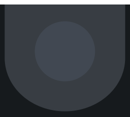


More Indicator Examples and Reference

Joystick

```
from dash import Dash, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.Joystick(
        id='our-joystick',
        label="Default",
        angle=0
    ),
    html.Div(id='joystick-result')
])


@callback(
    Output('joystick-result', 'children'),
    Input('our-joystick', 'angle'),
    Input('our-joystick', 'force')
)
def update_output(angle, force):
    return [f'Angle is {angle}', html.Br(), f'Force is {force}']


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



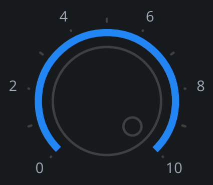


Angle is 0
Force is None

More Joystick Examples and Reference

Knob

```
from dash import Dash, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.Knob(id='our-knob'),
    html.Div(id='knob-result')
])


@callback(Output('knob-result', 'children'), Input('our-knob', 'value'))
def update_output(value):
    return f'The knob value is {value}.'


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



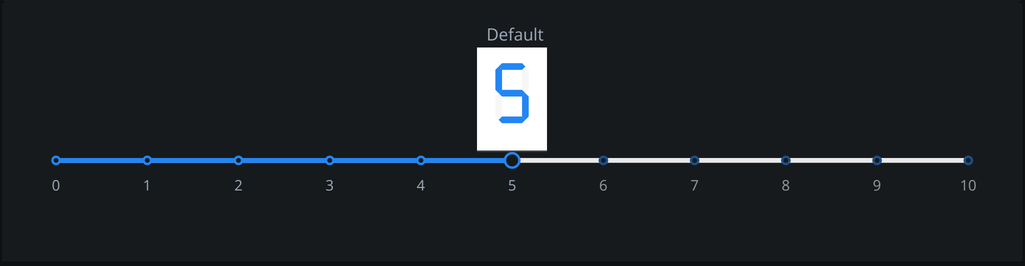
The knob value is None.

More Knob Examples and Reference

LEDDisplay

```
from dash import Dash, dcc, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.LEDDisplay(
        id='our-LED-display',
        label="Default",
        value=6
    ),
    dcc.Slider(
        id='our-LED-display-slider',
        min=0,
        max=10,
        step=1,
        value=5
    ),
])
```

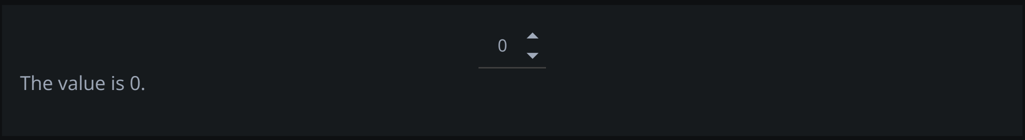
```
@callback(
    Output('our-LED-display', 'value'),
    Input('our-LED-display-slider', 'value')
)
def update_output(value):
    return str(value)


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



More LEDDisplay Examples and Reference

NumericInput

```
from dash import Dash, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.NumericInput(
        id='our-numeric-input',
        value=0
    ),
    html.Div(id='numeric-input-result')
])


@callback(
    Output('numeric-input-result', 'children'),
    Input('our-numeric-input', 'value')
)
def update_output(value):
    return f'The value is {value}.'


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



The value is 0.

More NumericInput Examples and Reference

PowerButton

```
from dash import Dash, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.PowerButton(
        id='our-power-button',
        on=False
    ),
    html.Div(id='power-button-result')
])


@callback(
    Output('power-button-result', 'children'),
    Input('our-power-button', 'on')
)
def update_output(on):
    return f'The button is {on}.'


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

The button is False.

More PowerButton Examples and Reference

PrecisionInput

```
from dash import Dash, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.PrecisionInput(
        id='our-precision',
        label='Default',
        precision=4,
        value=1234
    ),
    html.Div(id='precision-result')
])


@callback(
    Output('precision-result', 'children'),
    Input('our-precision', 'value')
)
def update_output(value):
    return f'The current value is {value}.'


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

Default

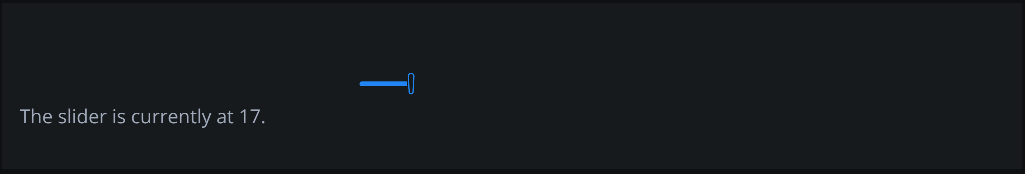
1 . 2 3 4 E 3

The current value is 1234.

More PrecisionInput Examples and Reference

Slider

```python
from dash import Dash, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.Slider(
        id='our-daq-slider-ex',
        value=17
    ),
    html.Div(id='slider-result')
])


@callback(
    Output('slider-result', 'children'),
    Input('our-daq-slider-ex', 'value')
)
def update_output(value):
    return f'The slider is currently at {value}.'


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

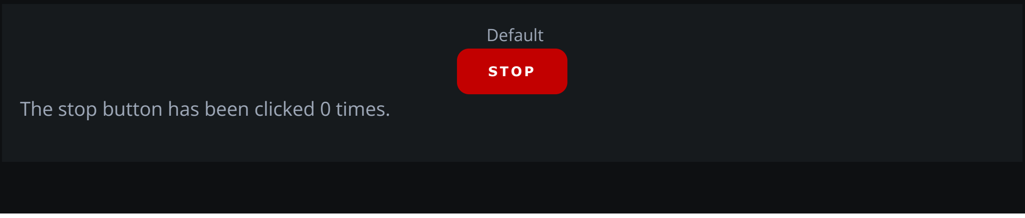


The slider is currently at 17.

More Slider Examples and Reference

StopButton

```python
from dash import Dash, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.StopButton(
        id='our-stop-button',
        label='Default',
        n_clicks=0
    ),
    html.Div(id='stop-button-result')
])


@callback(
    Output('stop-button-result', 'children'),
    Input('our-stop-button', 'n_clicks')
)
def update_output(n_clicks):
    return f'The stop button has been clicked {n_clicks} times.'


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

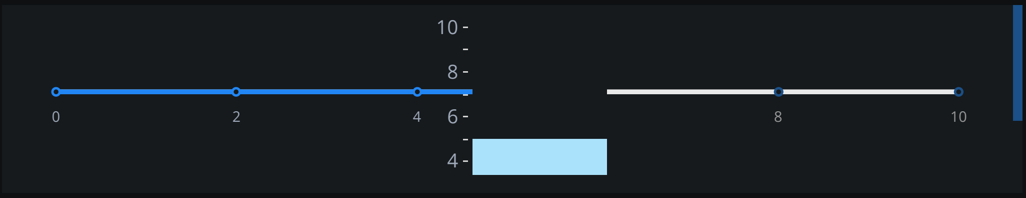


The stop button has been clicked 0 times.

More StopButton Examples and Reference

Tank

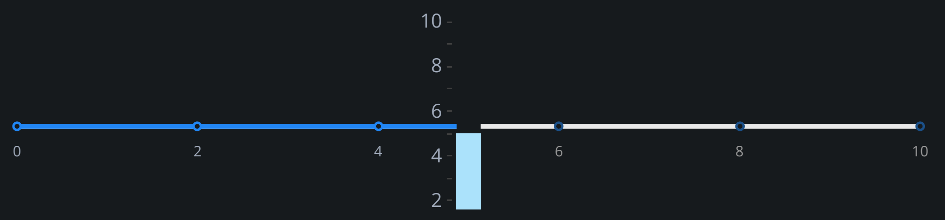
```
from dash import Dash, html, dcc, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.Tank(
        id='our-tank',
        value=5,
        min=0,
        max=10,
        style={'margin-left': '50px'}
    ),
    dcc.Slider(
        id='tank-slider',
        value=5,
        min=0,
        max=10,
    ),
])


@callback(Output('our-tank', 'value'), Input('tank-slider', 'value'))
def update_tank(value):
    return value


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



More Tank Examples and Reference

Thermometer

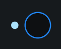
```
from dash import Dash, html, dcc, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.Thermometer(
        id='our-thermometer',
        value=5,
        min=0,
        max=10,
        style={
            'margin-bottom': '5%'
        }
    ),
    dcc.Slider(
        id='thermometer-slider',
        value=5,
        min=0,
        max=10,

    ),
])


@callback(
    Output('our-thermometer', 'value'),
    Input('thermometer-slider', 'value')
)
def update_thermometer(value):
    return value


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



More Thermometer Examples and Reference

ToggleSwitch

```
from dash import Dash, html, Input, Output, callback
import dash_daq as daq

app = Dash()

app.layout = html.Div([
    daq.ToggleSwitch(
        id='our-toggle-switch',
        value=False
    ),
    html.Div(id='toggle-switch-result')
])


@callback(
    Output('toggle-switch-result', 'children'),
    Input('our-toggle-switch', 'value')
)
def update_output(value):
    return f'The switch is {value}.'

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

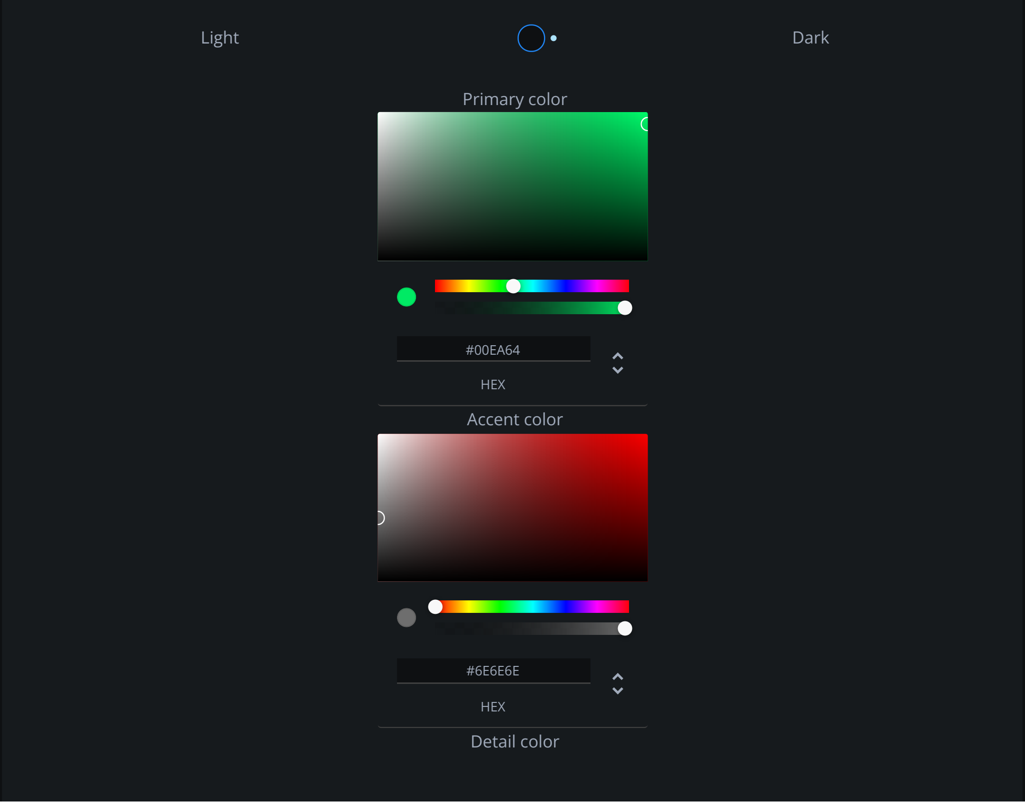


The switch is False.

More ToggleSwitch Examples and Reference

DarkThemeProvider

```
from dash import Dash, html, Input, Output, State, callback
import dash_daq as daq

app = Dash()

theme = {
    'dark': True,
    'detail': '#007439',
    'primary': '#00EA64',
    'secondary': '#6E6E6E',
}

rootLayout = html.Div([
    daq.BooleanSwitch(
        on=True,
        className='dark-theme-control'
    ), html.Br(),
    daq.ToggleSwitch(
        className='dark-theme-control'
    ), html.Br(),
    daq.ColorPicker(
        value=dict(hex='#171717'),
        className='dark-theme-control'
    ), html.Br(),
    daq.Gauge(
        min=0,
        max=10,
        value=6,
        color=theme['primary'],
        className='dark-theme-control'
    ), html.Br(),
    daq.GraduatedBar(
        value=4,
        color=theme['primary'],
        className='dark-theme-control'
    ), html.Br(),
    daq.Indicator(
        value=True,
        color=theme['primary'],
        className='dark-theme-control'
    ), html.Br(),
    daq.Knob(
        min=0,
        max=10,
        value=6,
        className='dark-theme-control'
    ), html.Br(),
    daq.LEDDisplay(
        value="3.14159",
        color=theme['primary'],
        className='dark-theme-control'
    ), html.Br(),
```


Light

Dark

Primary color

#00EA64

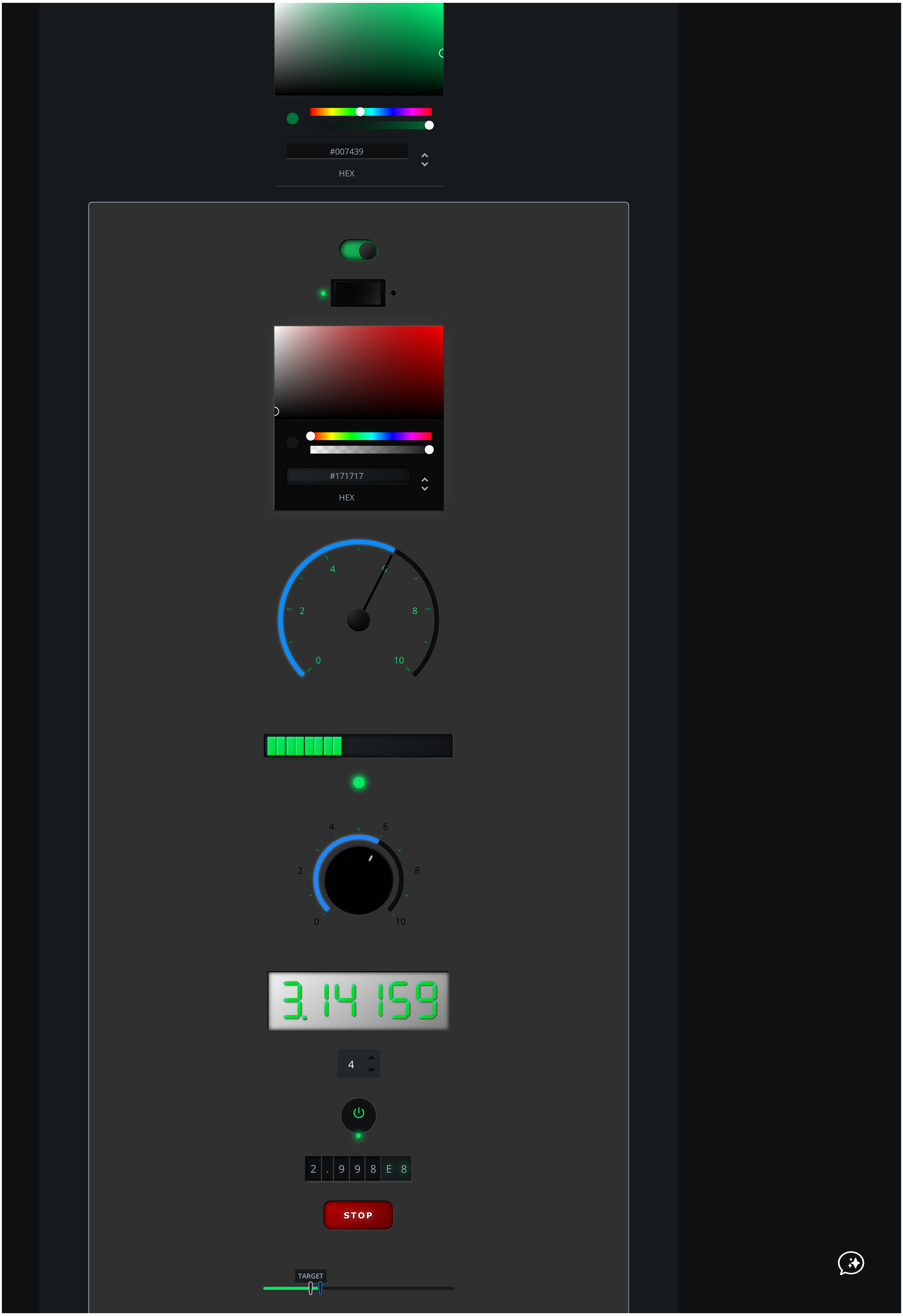
HEX

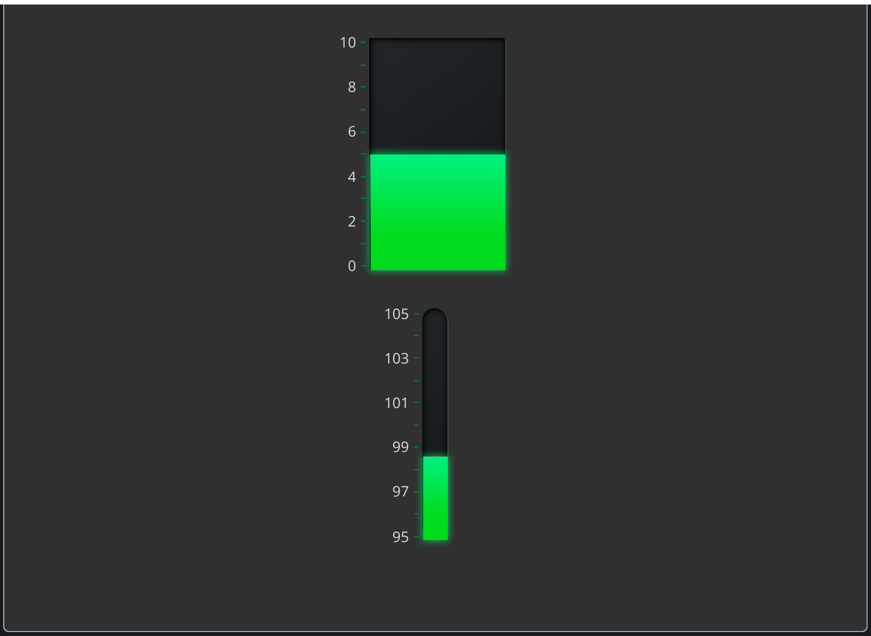
Accent color

#6E6E6E

HEX

Detail color


#007439
HEX
#171717
HEX
0
2
4
6
8
10
4
6
2
8
0
10
3.14159
4
2 . 9 9 8 E 8
STOP
TARGET



More DarkThemeProvider Examples and Reference

Dash Python ***> Dash DAQ***

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