

# HTML in Dash

BUILDING DASHBOARDS WITH DASH AND PLOTLY



**Alex Scriven**  
Data Scientist

# HTML Revisited

- HTML = language for structuring web content
  - Uses 'tags' like Div, H1>H6
- Dash wraps HTML using dash html components ( `dash.html` )
  - H1 tag is `.H1()`
  - Div tag is `.Div()`
- Many more tags available!

```
<div>
  <div style=
    "background-color: red;
    width:250; height:250;">
  </div>
  <div style=
    "background-color: lightblue;
    width:250; height:250;">
    <h1>This box</h1>
  </div>
</div>
```

# Structuring tags

- Huge list of HTML tags available (See [documentation](#))
- Important structuring tags:
  - `.Br()` = New line break
  - `.Img()` = Insert an image

# Lists in HTML Dash

- `.Ul()` \ `.Ol()` & `.Li()` = Create lists
  - `.Ul()` for unordered list (bullet-points, like these!)
  - `.Ol()` for ordered list (numbered-points)
  - `.Li()` for each list element

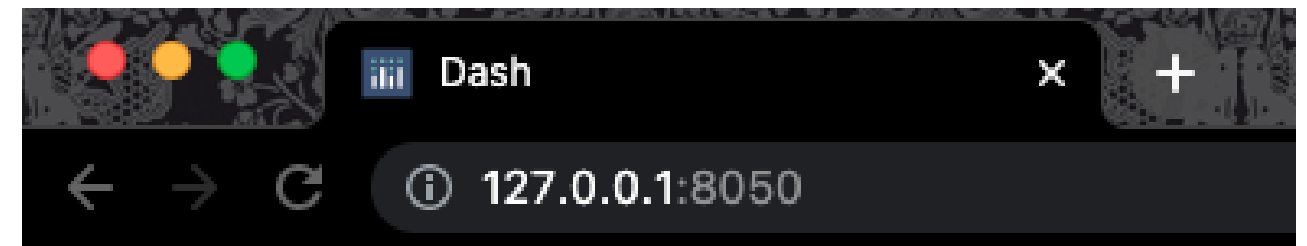
We will practice these!

# Inserting a company logo

Add company logo above the title;

```
app.layout = html.Div(children=[  
    html.Img(src='www.website.com/logo.png'),  
    html.H1("Our Sales Dashboard")  
])
```

Nice and professional!



**E-com global**



**Our Sales Dashboard**

# Text tags

Set and format text content

- `.P()` or `.Span()` = Insert plain text
  - Accept a children argument (list of text, `.P()` or `.Span()` )
- `.B()` = **Bold** some text
- `.I()` = *Italicize* some text

# HTML text tags in Dash

Some complex formatting:

```
app.layout = html.Div(children=[
    html.H1("Our Sales Dashboard"),
    html.Span(children=[
        f"Prepared: {datetime.now().date()}",
        " by ", html.B("Jessie Parker, "),
        html.I("Data Scientist")
    ])
])
```



## Our Sales Dashboard

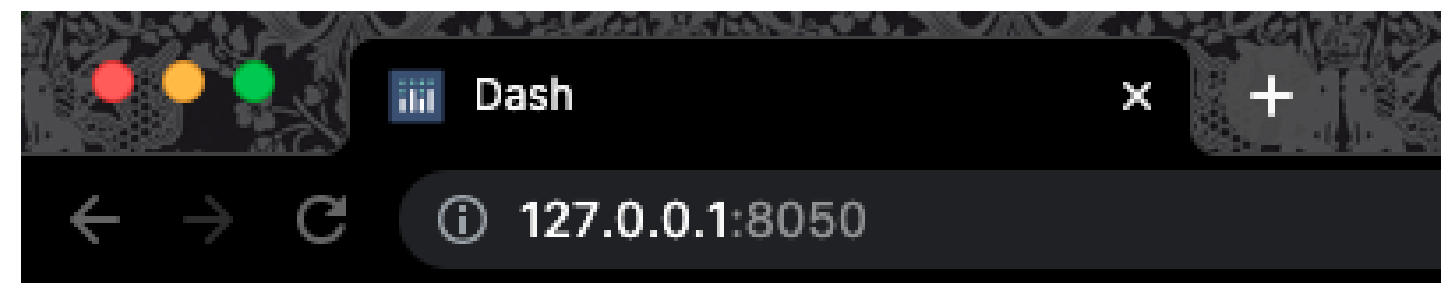
Prepared: 2021-06-27 by **Jessie Parker**, *Data Scientist*

# Breaking up the text

Adding line breaks:

```
app.layout = html.Div([
    html.H1("Our Sales Dashboard"),
    html.Span(children=[
        f"Prepared: {datetime.now().date()}",
        html.Br(),
        " by ", html.B("Jessie Parker, "),
        html.Br(),
        html.I("Data Scientist")]
    ])
])
```

With line breaks:



## Our Sales Dashboard

Prepared: 2021-06-27  
by **Jessie Parker**,  
*Data Scientist*

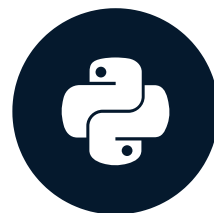


# Let's practice!

BUILDING DASHBOARDS WITH DASH AND PLOTLY

# CSS Basics in Dash

BUILDING DASHBOARDS WITH DASH AND PLOTLY



**Alex Scriven**  
Data Scientist

# What is CSS?

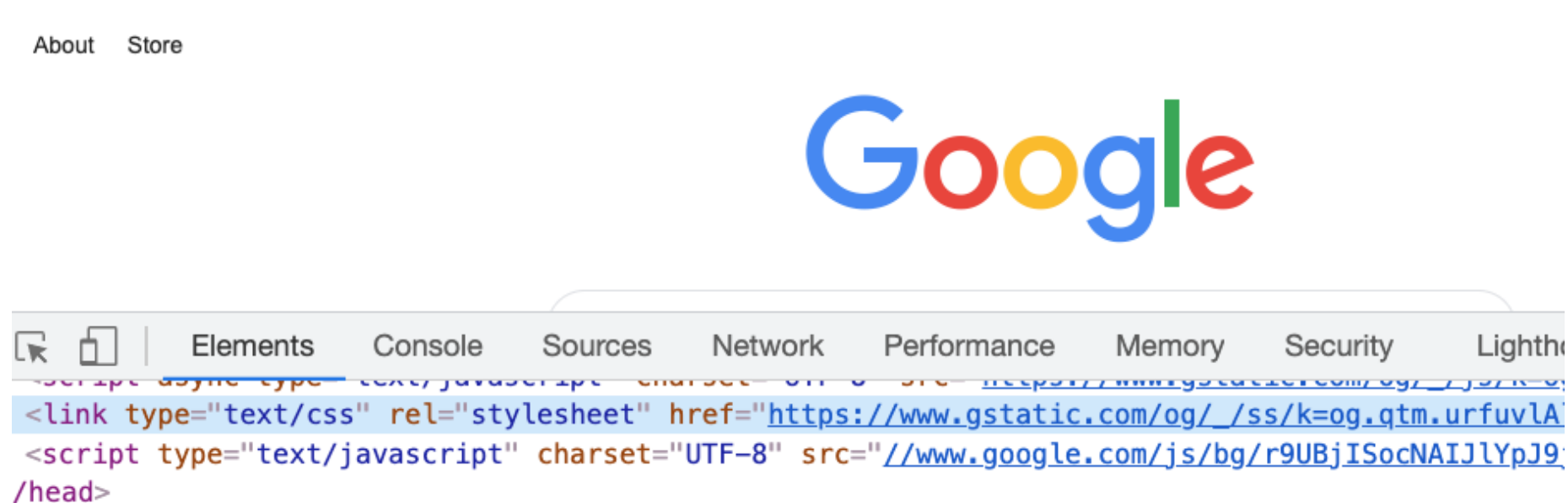
CSS stands for 'Cascading Style Sheets'

- A language to determine how a webpage is styled
- We can control:
  - Text/font properties
  - Size & shape of objects (HTML tags!)
  - Placement of objects

# CSS on the web

Most websites have CSS files;

- They are read in on page load
- One way to apply CSS to page elements
- Try opening up developer tools on Chrome to see (Google below!)
- This course: `style` property instead



# CSS on HTML elements

CSS can also be used on a HTML element

- Use the `style` property of a tag
- CSS statements in one big string separated by `;`
  - Statements are `property:value;property:value;`

```
<div>
  <div style="background-color: red;
            width:250; height:250;">

  </div>
  <div style="background-color: lightblue;
            width:250; height:250;">
    <h1>This box</h1>
    <h2>Another Title</h2>
  </div>
</div>
```

# Some CSS styling

```
<h1>Welcome to the website!</h1>  
<h2 style="font-size:50px;color:red">  
    Enjoy your stay!  
</h2>
```

We can see the styling on our second title:

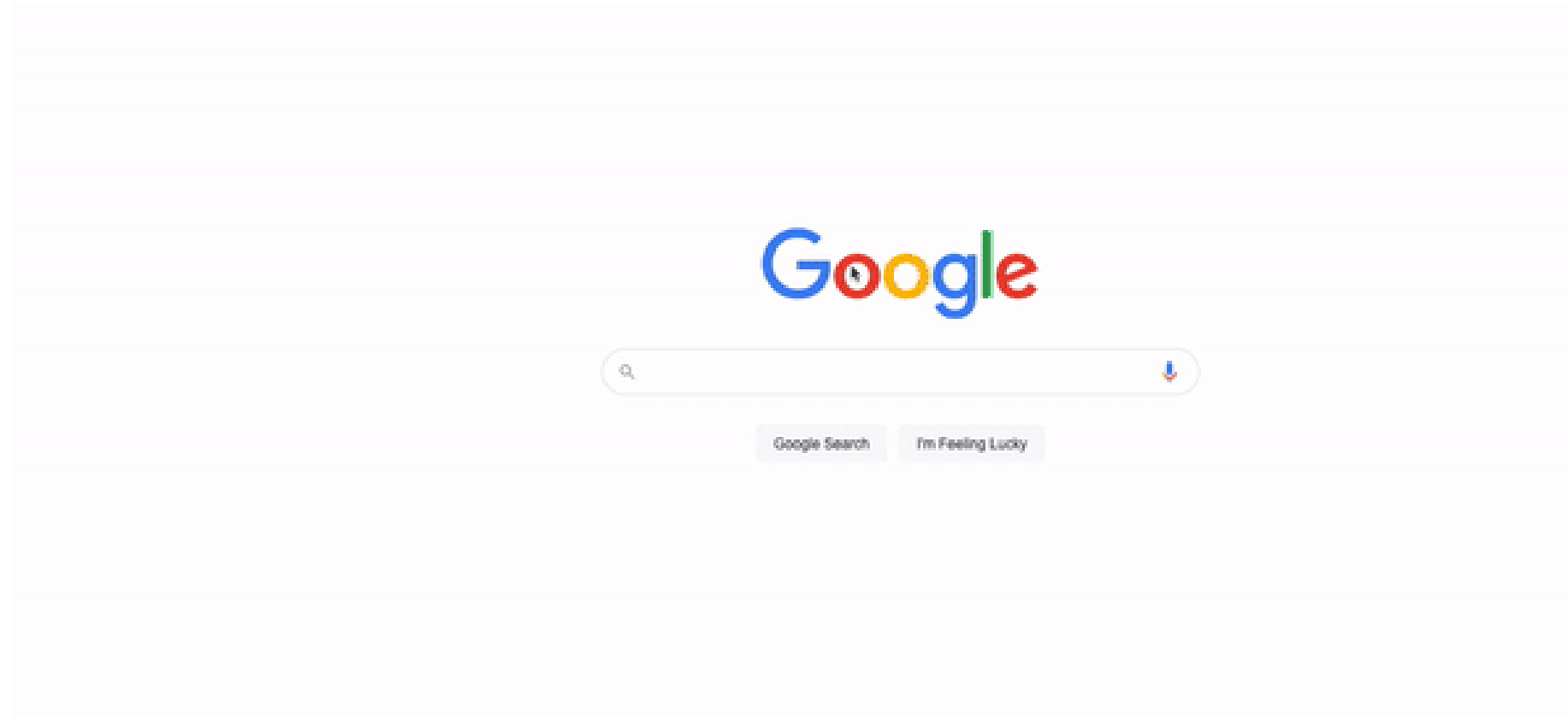
**Welcome to the website!**

**Enjoy your stay!**

# Editing live CSS

We can see how to edit some CSS on a live website

- Right click an element > select 'inspect' > edit a CSS property
- Note - only a local change (gone on refresh)



# CSS in Dash

- Dash components `style` argument
  - Accepts CSS as a dictionary
- Previous code in a Dash component:

```
app.layout = html.Div([
    html.H1('Welcome to the website!'),
    html.H2('Text',
    style={'font-size': '50px',
          'color': 'red'})
])
```



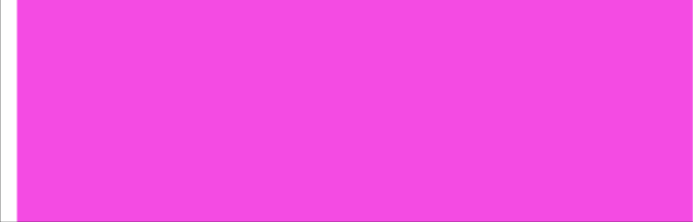



# CSS for color

CSS can be used to set the:

- Background color of an object ( `background-color` )
- Text color ( `color` )

Both methods accept strings (e.g., 'red') or RGB codes (e.g., `'rgb(0,0,255)'` is blue!)

Some color RGB codes:

Color	RGB Code
	<b>(245, 66, 230)</b>
	<b>(105, 245, 66)</b>
	<b>(245, 66, 87)</b>
	<b>(50, 47, 247)</b>

# CSS for Size

CSS can set the size via the `width` and `height` properties

```
app.layout = html.Div([
    # Add & resize the company logo
    html.Img(src=logo_link,
             style={'width': '250px', 'height': '250px'})
])
```

# CSS size as a percentage

```
app.layout = html.Div([  
    # Add & resize the company logo  
    html.Img(src=logo_link,  
             style={'width': '50%', 'height': '50%'})  
])
```

- 50% of parent element size

# Let's practice!

BUILDING DASHBOARDS WITH DASH AND PLOTLY

# Advanced CSS in Dash

BUILDING DASHBOARDS WITH DASH AND PLOTLY

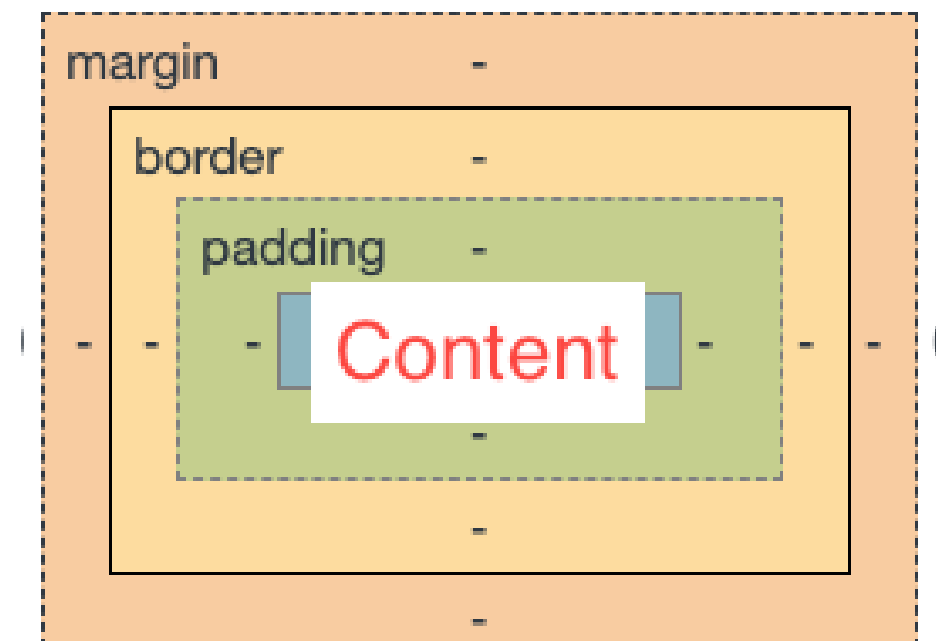


**Alex Scriven**  
Data Scientist

# CSS for spacing between objects

The 'Box Model' considers each HTML element as a box with these properties:

- Content of the object (`height` & `width` properties)
- Padding (outside the content)
- Border (between padding and margin)
- Margin (outside the border, separating one element from another)



# Adding a border

The `border` CSS argument has three elements:

- `'border': 'A B C'`
  - A = width in pixels
  - B = style (E.g., `solid` or `dotted`)
  - C = color (E.g., `red`)

# A border on our app

```
html.Div(dcc.Graph(figure=ecom_bar),  
         style={'width': '500px',  
               'height': '450px',  
               'border': '5px dotted red'})
```





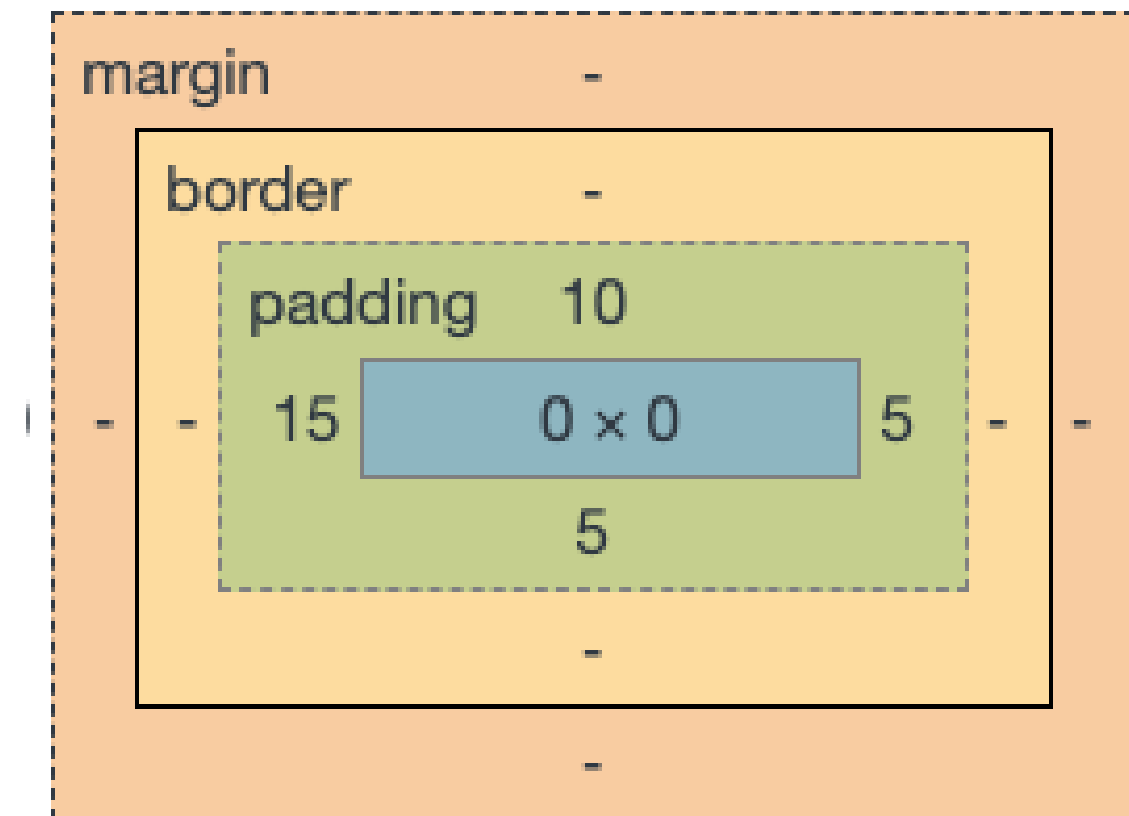
# CSS spacing

To set the spacing of an HTML element:

- Specify four numbers for each property (Padding & Margin)
  - Clockwise will be top, right, bottom, left
- Alternatively: one number (will be applied to all sides)
- Alternatively: two numbers for top-bottom and left-right

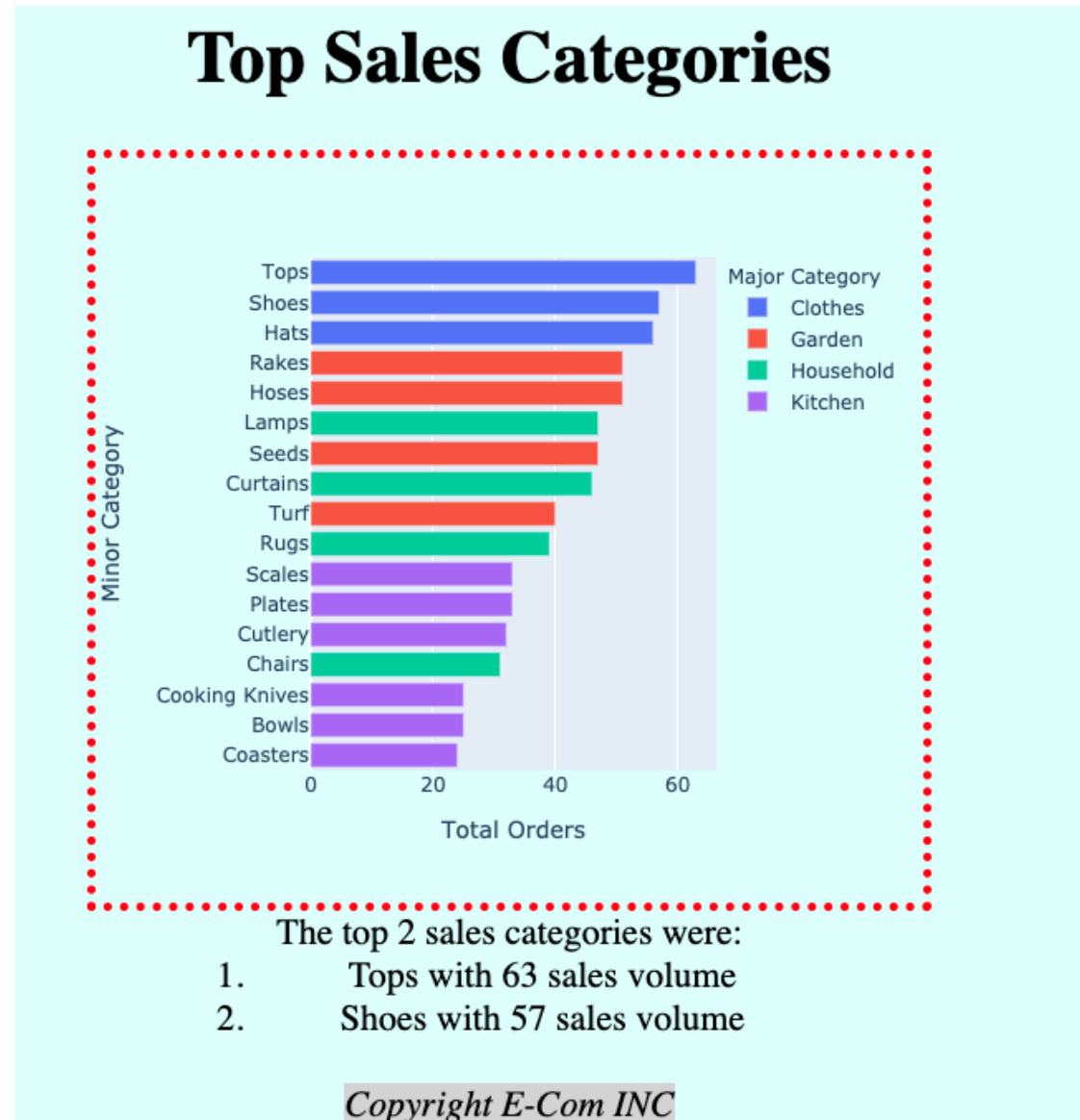
E.g., `'padding': '10px 5px 5px 15px'`

Produces this:

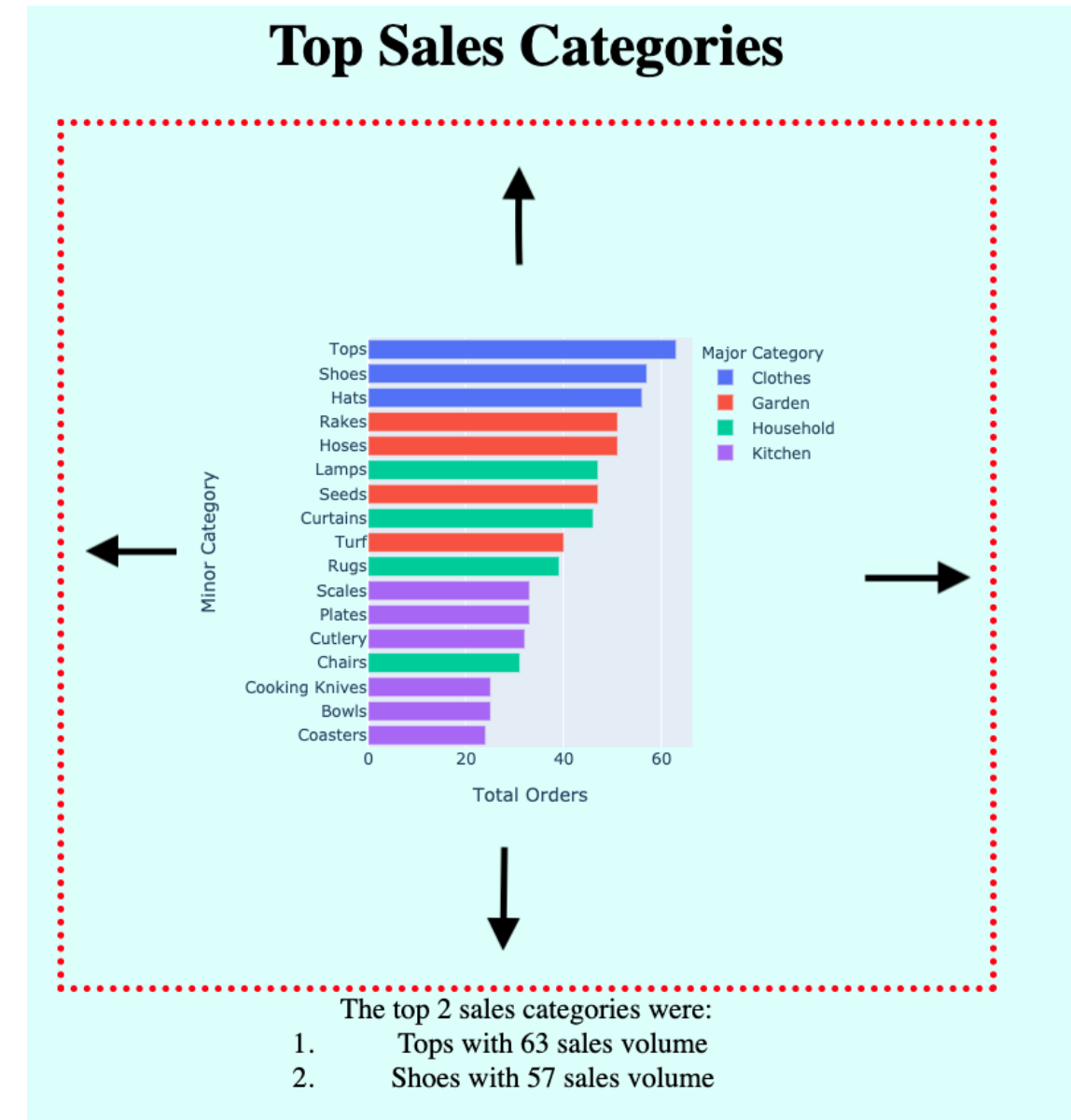


# Adding padding in Dash

No padding:



Adding `'padding': '100px'` :

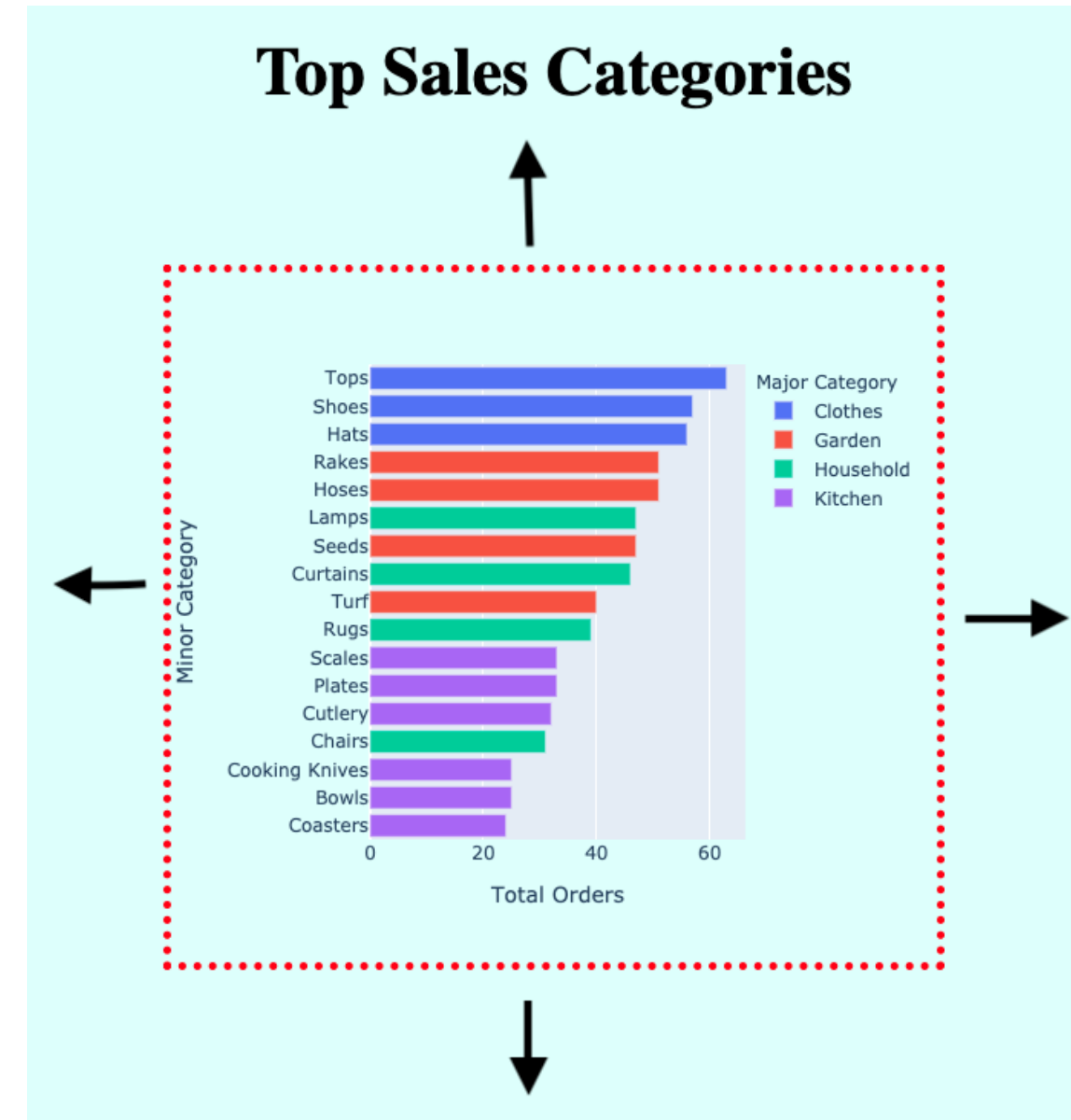


# Adding margin in Dash

No margin:

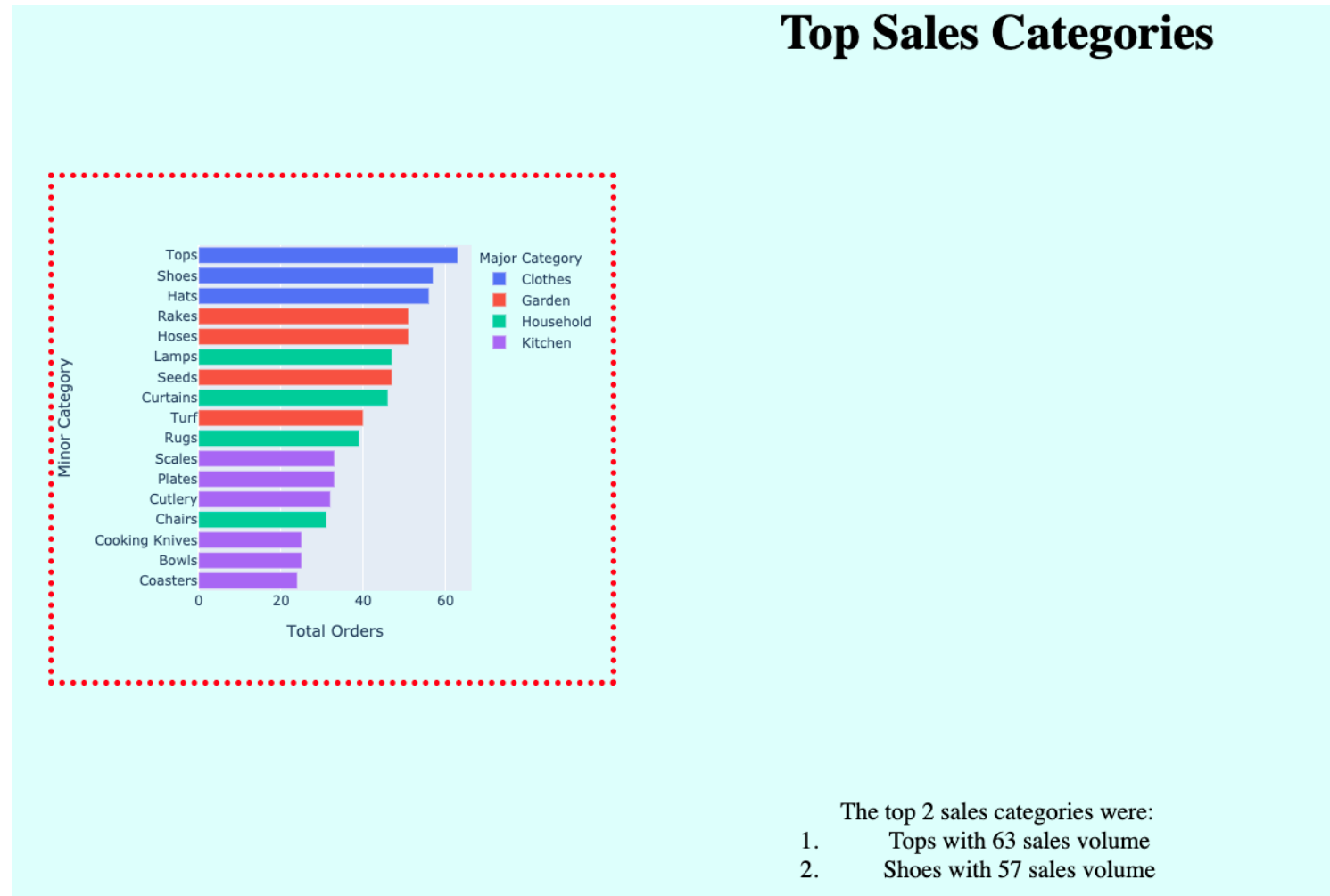


Adding `margin: 100px auto`:

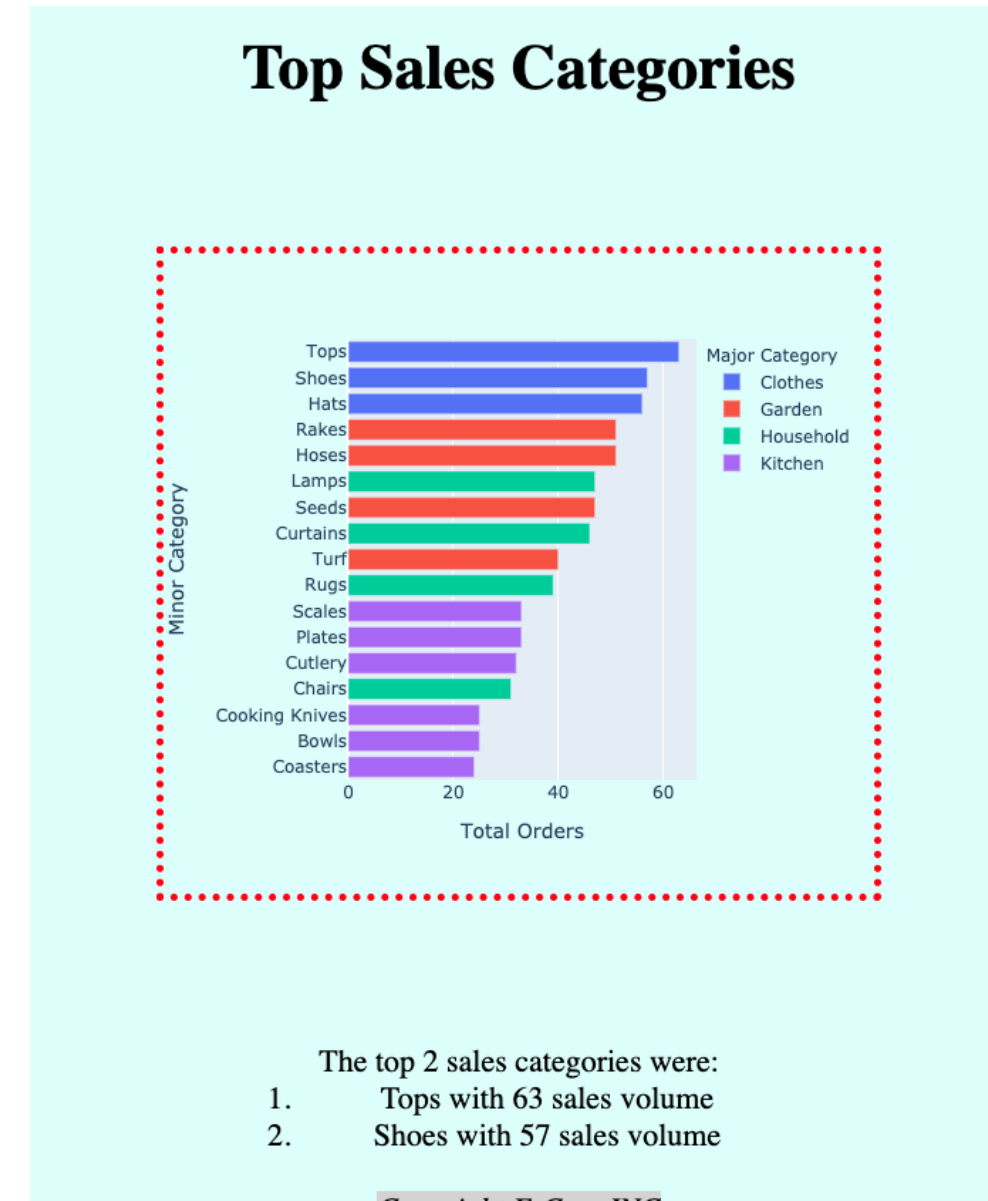


# Centering with auto margin

With: `'margin': '100px'`



With: `'margin': '100px auto'`



# CSS for layout

Elements not aligning? HTML elements can either be 'inline' or 'block' elements.

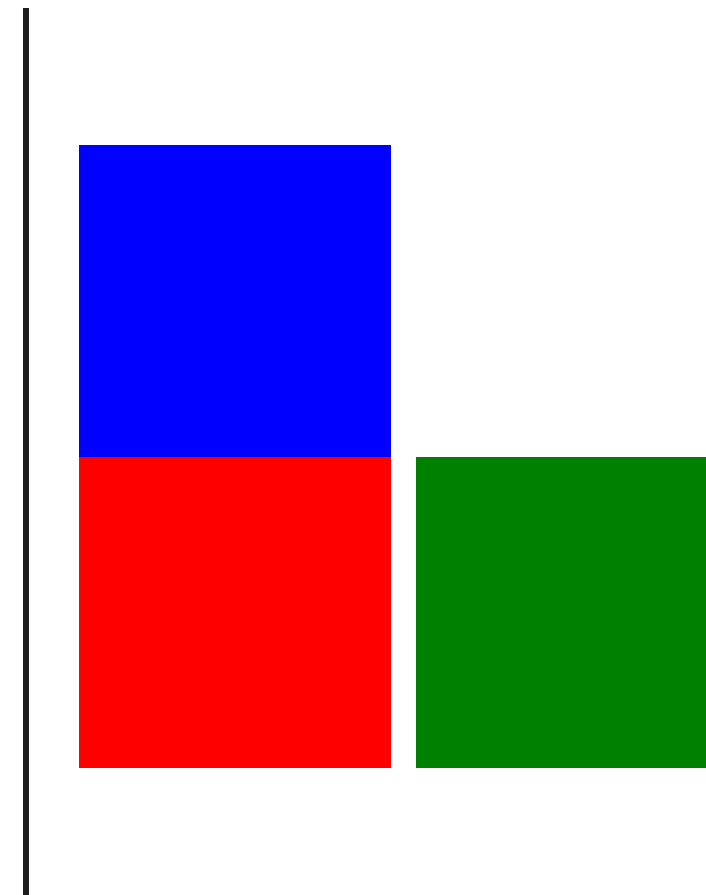
- Inline render on the same line
  - Have no height or width (or box) properties
  - Examples include `<strong>`, `<a>`, `<img>`
- Block stack on top of one another as they include a line break
  - Can't have more than one side-by-side
  - Examples include `<h1>`, `<div>`
- There is also inline-block
  - Can set height/width/box properties
  - Can be side-by-side

# Inline-block elements

We can create some divs that are block and inline-block;

```
<div style='width:50px;height:50px;
background-color:blue'></div>
<div style='width:50px;height:50px;
background-color:red
display:inline-block'></div>
<div style='width:50px;height:50px;
background-color:green;
display:inline-block'></div>
```

Notice the red beside green?



# Let's practice!

BUILDING DASHBOARDS WITH DASH AND PLOTLY