

# Customization and branding

# Your turn

To keep everyone on the same page, we'll play with  
`quarto-websites-exercise-site_2025-10`

[andhs.co/quarto-websites-2025](https://andhs.co/quarto-websites-2025)

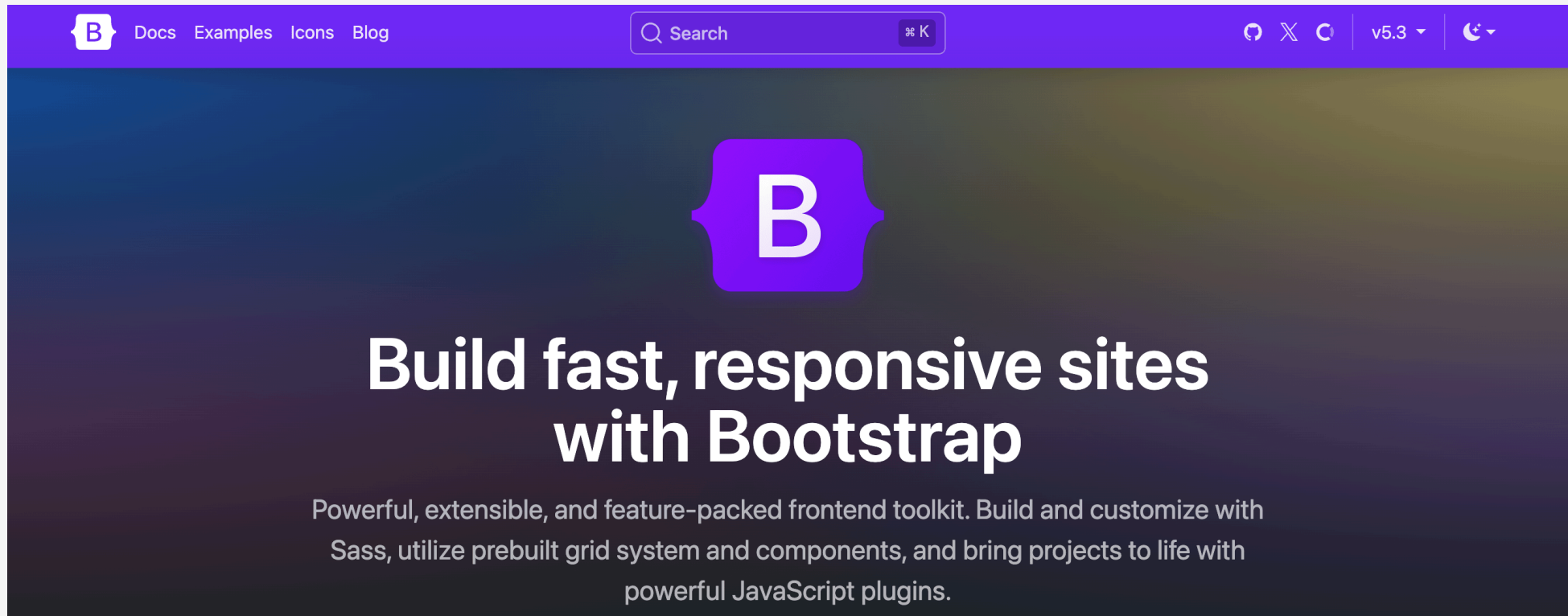
But keep your playground site from yesterday handy!

05:00

# Themes

# Bootstrap

Quarto uses the popular **Bootstrap** library for HTML structure and CSS styles.



# Bootstrap

You can use **Bootstrap components and classes** for special styling

```
[Here's a button](thing.pdf){.btn .btn-primary role="button"}
```

```
[Here's another button](thing.pdf){.btn .btn-warning role="button"}
```



Here's a button

Here's another button

# Bootswatch themes

Quarto includes 25 themes from **Bootswatch**:

- default
- cerulean
- cosmo
- cyborg
- darkly
- flatly
- journal
- litera
- lumen
- lux
- materia
- minty
- morph
- pulse
- quartz
- sandstone
- simplex
- sketchy
- slate
- solar
- spacelab
- superhero
- united
- vapor
- yeti
- zephyr

# Changing themes

Specify the custom theme under `theme` in the YAML settings:

```
_quarto.yml
1 format:
2   html:
3     theme:
4       - zephyr
```

# Your turn

1. Go to [bootswatch.com](https://bootswatch.com) and explore the different themes there (use the top navigation bar).
2. Preview your site, then try changing different Bootswatch themes in `_quarto.yml`.

07:00



# CSS and Sass

# Theme options

Sometimes we want to change theme settings:

- Fonts
- Colors
- Alignment
- Spacing

# Basic theme options

Quarto automatically supports **some common Bootstrap theme options**

```
format:  
  html:  
    theme: zephyr  
    fontsize: 1.2em  
    linestretch: 1.2  
    linkcolor: "#32a852"  
    backgroundcolor: "#f8f8c4"  
    mainfont: Comic Sans
```

# Total control with CSS

# Crash course in CSS

HTML elements can have IDs and classes:

```
<h2 id="my-section" class="special">A heading</h2>
```

- IDs are unique on the page
  - Shorthand: `#my-section`
- Classes aren't unique and can be repeated
  - Shorthand: `.special`

```
/* All H2s */
h2 {
  color: red;
}

/* All elements with id my-section */
/* This includes non H2s */
#my-section {
  color: red;
}

/* All H2s with id my-section */
h2#mysection {
  color: red;
}

/* All elements with class .special */
.special {
  font-style: italic;
}

/* All H2s with class .special */
h2.special {
  font-style: italic;
}
```

# Nesting

HTML elements can be nested inside each other

```
<div id="part1">
  <h2 id="my-section" class="special">A heading</h2>
</div>

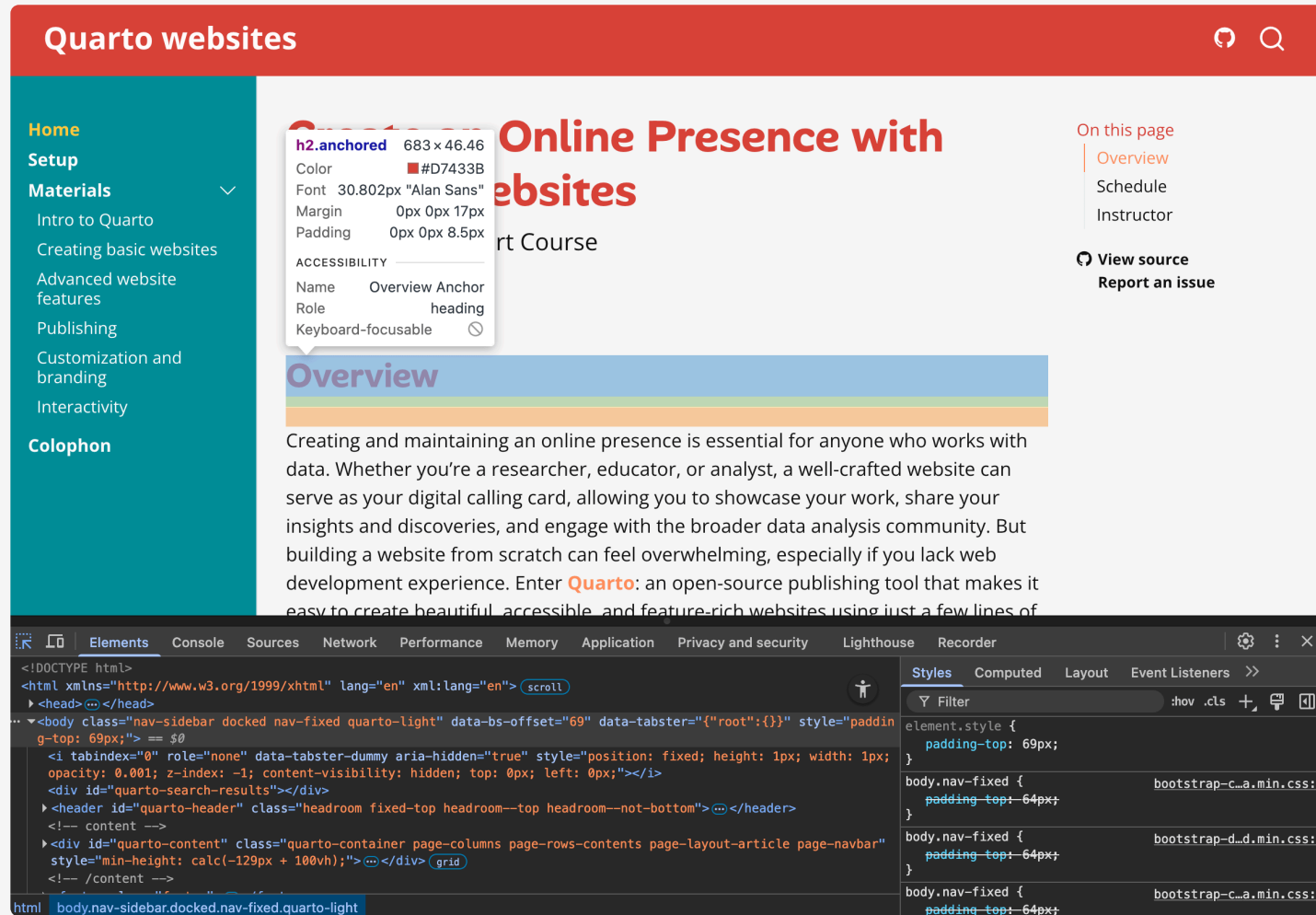
<div id="part2">
  <h2 id="blah" class="special">Another heading</h2>
</div>
```

```
/* All H2s in #part1 */
#part1 h2 {
  color: red;
}

/* All H2s in #part2 */
#part2 h2 {
  color: blue;
}
```

# Browser inspector

Explore HTML and CSS right from your browser!



# Browser inspector

## Edit and tinker with styles

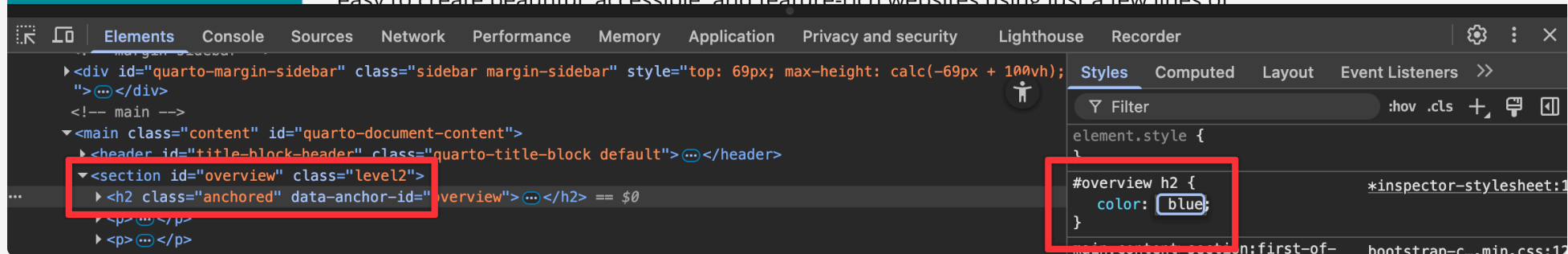
Customization and branding

Interactivity

Colophon

### Overview

Creating and maintaining an online presence is essential for anyone who works with data. Whether you're a researcher, educator, or analyst, a well-crafted website can serve as your digital calling card, allowing you to showcase your work, share your insights and discoveries, and engage with the broader data analysis community. But building a website from scratch can feel overwhelming, especially if you lack web development experience. Enter **Quarto**: an open-source publishing tool that makes it easy to create beautiful, accessible, and feature-rich websites using just a few lines of





# Browser inspector

See which rule applies

Customization and branding

Interactivity

Colophon

and create interactive elements like live code and data visualizations.

## Schedule

All times are US Eastern (New York) time:

Day 1 (October 16)

10:30–11:30

Welcome + Intro to Quarto

Elements Console Sources Network Performance Memory Application Privacy and security Lighthouse Recorder

> <header id="title-block-header" class="quarto-title-block default"></header>

> <section id="overview" class="level2">

> <h2 class="anchored" data-anchor-id="overview"></h2>

> <p></p>

> <p></p>

</section>

> <section id="schedule" class="level2">

> <h2 class="anchored" data-anchor-id="schedule"></h2> == \$0

> <p>All times are US Eastern (New York) time:</p>

> <div class="cell"></div>

</section>

> <section id="instructor" class="level2"></section>

</main>

</main>

rows-contents.page-layout-article.page-navbar main#quarto-document-content.content section#schedule.level2 h2.anchored

Styles Computed Layout Event Listeners >>

> margin-block-end 17px

> margin-block-start 34px

> margin-bottom 17px

1rem h1, .h1, h2, .h2

.5rem h6, .h6, h5, .h5, h4, .h4, h3, .h3, h2, .h2, h1, .h1

1rem h1, .h1, h2, .h2

.5rem h6, .h6, h5, .h5, h4, .h4, h3, .h3, h2, .h2, h1, .h1

1rem h1, .h1, h2, .h2

.5rem h6, .h6, h5, .h5, h4, .h4, h3, .h3, h2, .h2, h1, .h1

1rem h1, .h1, h2, .h2

.5rem h6, .h6, h5, .h5, h4, .h4, h3, .h3, h2, .h2, h1, .h1

bootstrap-c3b88...c9a.min.css:12

bootstrap-c3b88...cc9a.min.css:5

bootstrap-dark...fad.min.css:12

bootstrap-dark...afad.min.css:5

bootstrap-c3b88...c9a.min.css:12

bootstrap-c3b88...cc9a.min.css:5

# Your turn

1. Preview your site
2. Use the browser inspector to select a heading
3. Create a CSS style that targets that heading and makes it
  - red (`color`) and
  - italic (`font-style`) and
  - 30px (`font-size`)
4. TODO

10:00

# Sass: CSS, but fancier

```
@import url('https://fonts.googleapis.com/css2?family=Roboto:ital,wght@0,100..900;1,100..900&display=swap');

/*-- scss:defaults --*/

/* Built-in Bootstrap variables */
$h2-font-size: 1.6rem !default;
$headings-font-weight: 500 !default;
$font-family-sans-serif: Roboto

/* Your own variables */
$my-neat-red: #e21818;

/*-- scss:rules --*/
#quarto-sidebar h2 {
  font-family: Roboto;
  color: $my-neat-red;

  /* Magically compiles to #quarto-sidebar h2 a:hover */
  a:hover {
    color: #d1c81d;
  }
}
```

# Bootstrap Sass variables

There are ***so many***

- TODO: Quarto list
- TODO: Bootstrap directory

# Using CSS/Sass with Quarto

Use vanilla CSS

```
format:  
  html:  
    css: custom.css
```

Use Sass

```
format:  
  html:  
    theme:  
      - zephyr  
      - custom.scss
```

TODO: Examples from other sites

# Your turn

TODO: Create a SCSS file and make a bunch of rules

# Branding

# CSS is a little inconvenient

- HTML, slides, and dashboards all use slightly different underlying HTML
- No easy way to reuse the colors and typography from your CSS customizations in R and Python plots or in PDF documents
- Hard to share consistent, reusable themes with others in your organization (or with the world)



# Style guides

U

CHART PARTS

COLOR

BEST PRACTICES

ACCESSIBILITY

EXAMPLES

MAIN GRAPHIC COLORS

Hex: #1696d2  
rgb(22,150,210)  
CMYK: 90,29,0,18

Hex: #d2d2d2  
rgb(210,210,210)  
CMYK: 0,0,0,18

Hex: #000000  
rgb(0,0,0)  
CMYK: 0,0,0,100

Hex: #fdbf11  
rgb(253,191,17)  
CMYK: 0,25,93,1

Hex: #ec008b  
rgb(236,0,139)  
CMYK: 0,100,41,7

Hex: #55b748  
rgb(85,183,72)  
CMYK: 54,0,61,28

Hex: #5c5859  
rgb(92,88,89)  
CMYK: 0,4,3,64

Hex: #db2b27  
rgb(219,43,39)  
CMYK: 0,80,82,14

TEXT AND CONTRAST

Urban Institute data visualizations should strive to meet [Web Content Accessibility Guidelines \(WCAG\)](#) to make web content most accessible to people with disabilities. (Read more about those international standards [here](#) and in the Urban Institute report [Do No Harm Guide: Centering Accessibility in Data Visualization](#)). Urban follows WCAG 2.0 Level AA guidance to ensure that background color and text pairings maximize contrast.

The color palettes below contain the correct white or black text to pass the WCAG AA ratings for contrast at smaller text sizes (18px or less).

SHADES OF MAIN COLORS

Hex: #cfe8f3  
rgb(207,232,243)  
CMYK: 15,5,0,5

Hex: #a2d4ec  
rgb(162,212,236)  
CMYK: 31,10,0,7

Hex: #73bfe2  
rgb(115,191,226)  
CMYK: 49,15,0,11

Hex: #46abdb  
rgb(70,171,219)  
CMYK: 68,22,0,14


Hex: #1696d2  
rgb(22,150,210)  
CMYK: 90,29,0,18

Hex: #12719e  
rgb(18,113,158)  
CMYK: 89,28,0,38

Hex: #0a4c6a  
rgb(10,76,106)  
CMYK: 91,28,0,58

Hex: #062635  
rgb(6,38,53)  
CMYK: 89,28,0,79

# Style guides

 UNIVERSITY OF GEORGIA

Type to search...

**BRAND STYLE GUIDE**

Messaging | **Visual Identity** | Trademarks | Social Media | Web + Digital | Download Center

— VISUAL IDENTITY —

LOGOS >

**VISUAL STYLE** ▾

Color

**Typography**

Graphic Elements

Photography

Videography

USAGE GUIDELINES >

FAQS

CONTACT

## Typography

When it is used thoughtfully, typography becomes a powerful brand tool that can add visual meaning to what we communicate. Georgia's typography communicates clearly and cleanly, and is flexible for a wide range of situations.

### FONTS

Primary Sans-Serif

STYLES

**OSWALD**

OSWALD LIGHT  
**OSWALD MEDIUM**

Oswald is our primary sans-serif family and works best for headlines, subheads, and infographics. *The only approved weights are light and medium.*

Primary Serif

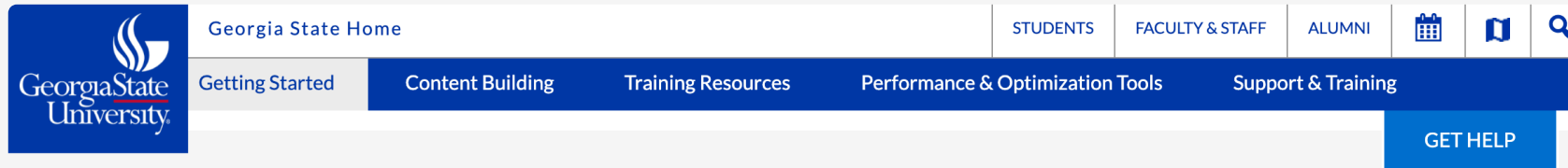
Styles

**Merriweather**

Merriweather Regular  
*Merriweather Italic*  
**Merriweather Bold**

For more sophisticated situations, Merriweather,

# Style guides



## Primary Colors

The university primary colors are blue and white. These two colors should be the strongest palette on layouts created for Georgia State.

**Georgia State  
Blue**

rgb **0 57 166**  
hex **0039A6**

**White**

rgb **255 255 255**  
hex **FFFFFF**

## Secondary Color Chart

**Red Accent**

rgb **198 12 48**  
hex **CC0000**

**Light Blue**

rgb **151 202 235**  
hex **97CAEB**

**Blue Steel**

rgb **55 64 87**  
hex **374057**

**Light Gray**

rgb **238 238 238**  
hex **EEEEEE**

**Cool Blue**

rgb **0 113 206**  
hex **0071CE**



**Medium Gray**

rgb **204 204 204**  
hex **CCCCCC**

# Matching style with CSS

You can recreate styles with CSS ([site](#); [custom.scss](#))

**2024 AYS DATA VISUALIZATION**SESSION ISESSION IISession III



## DATA VISUALIZATION FOR A PUBLIC AUDIENCE

Make data meaningful through storytelling. Design better charts, graphs, interactives, and dashboards by learning foundational graphic design principles and the universal grammar of data-based graphics.

### SESSION I

**Truth and Beauty with Data in R: An Introduction to the Tools**

February 16, 2024

- Learn how to use R and the tidyverse to load, manipulate, and plot data
- Become familiar with the grammar of graphics—a new way to talk about data-based graphs
- Discover core graphic design principles that you can apply to any designed document
- *No prior coding experience is necessary*

### SESSION II

**Visualizing Data in R: Core Applications**

April 12, 2024

- Learn best practices for visualizing amounts, proportions, uncertainty, comparisons, and relationships.
- Discover best practices for styling and annotating data-based graphics
- *A basic understanding of R and {ggplot2} is required/highly recommended*

View materials

### SESSION III

**Visualizing Data in R: Extensions: Time, Space, and Text**

June 7, 2024

- Make interactive plots and dashboards
- Visualize trends in data over time
- Make maps and conduct basic geographic information systems (GIS) analysis
- Learn the best ways to visualize text and qualitative data
- *A basic understanding of R and {ggplot2} is required/highly recommended*



# BRAND YML

## Unified branding with a simple YAML file

Create reports, apps, dashboards, plots and more that match your organization's brand guidelines with a single `_brand.yml` file.

Learn more: <https://posit-dev.github.io/brand-yml/>

1. Define branding in a single `_brand.yml` file.

2. Apply that branding across almost all Quarto formats.

# `_brand.yml` elements

- `meta`: Identifying information, name of the company, URLs, etc.
- `logo`: Files or links to the brand's logos
- `color`: Colors in the brand's color palette
- `typography`: Fonts for different elements
- `defaults`: Additional context-specific settings

# brand.yml structure

\_brand.yml

```
1 meta:
2   name: Urban Institute
3   link:
4     home: https://urbaninstitute.github.io/graphics-styleguide/
5
6 logo:
7   images:
8     icon:
9       path: img/urban.png
10      alt: Urban Institute logo
11   small: img/urban-sm.png
12
13 color:
14   palette:
15     blue: "#1696d2"
16     gray-light: "#d2d2d2"
17     black: "#000000"
18     yellow: "#fdbf11"
```



# Enabling `_brand.yml` in Quarto

1. Define branding in `_brand.yml`.
2. Save in the root directory of your Quarto project.

Quarto will detect the presence of `_brand.yml` and automatically apply the brand to all documents of the supported formats in the project.

If your brand file has a different name or lives in a subdirectory, use the `brand` key.

```
my-document.qmd
```

```
1 ---
2 title: "My neat report"
3 format: html
4 brand: org_theme.yml
5 ---
```

# Disable `_brand.yml`

To turn off `_brand.yml` for a document, use `brand: false`.

my-document.qmd

```
1 ---
2 title: "My neat report"
3 format: html
4 brand: false
5 ---
```

# Your turn

1. There is a file named `urban_institute.yml`. Rename it to `_brand.yml` and rerender your site. What changes?

TODO

2. Modify some of the variables in the `_brand.yml` file and rerender to see how your site changes. Explore [the brand.yml documentation](#) and see what other settings you can adjust.

TODO You can reset the file by copying and pasting it from [here](#).

05:00

# Use `_brand.yml` in markdown

Access some `_brand.yml` values in Quarto documents with a shortcode

```
my_document.qmd
```

```
1 {{< brand color primary >}}
```

# Use `_brand.yml` in Sass

Access some `_brand.yml` values with the `$brand-*` prefix

```
custom-styles.scss
```

```
1  /* -- scss:rules -- */
2
3  h3 {
4    color: $brand-magenta;
5  }
```

# Use `_brand.yml` in R and Python

Access and apply specific brand elements

R

Python

```
1 library(brand.yml)
2
3 brand <- read_brand_yaml("_brand.yml")
4
5 brand$color$primary
6 #> [1] "#1696d2"
7
8 brand$color$palette$yellow
9 #> [1] "#fdbf11"
10
11 brand$typography$fonts[[1]]$family
12 #> [1] "Lato"
```

# Theme helpers

The {quarto} package contains theme helpers that apply branding to plots

R

Python

```
1 library(quarto)
2
3 my_theme <- theme_brand_ggplot2("_brand.yml")
```

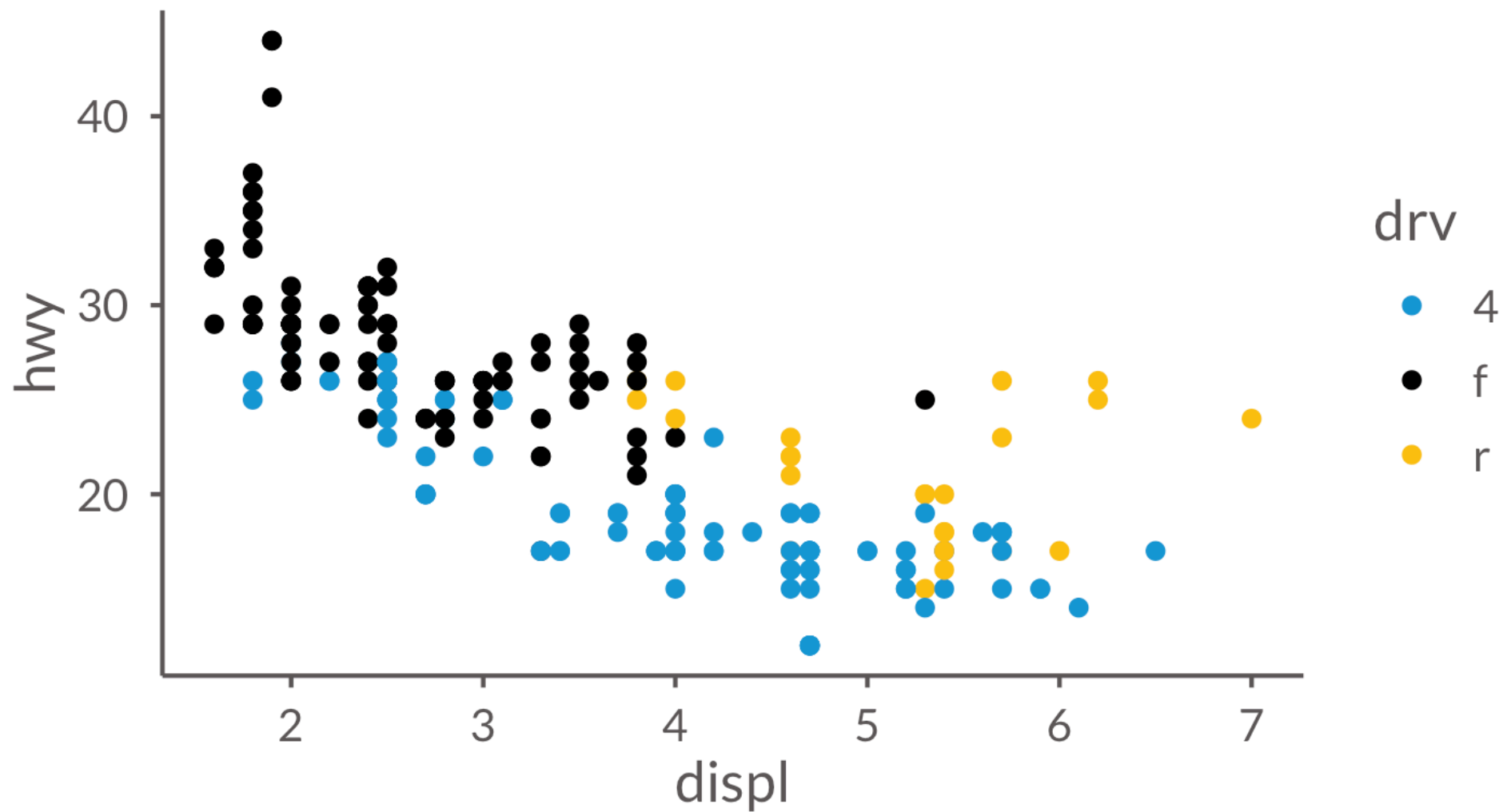
Doesn't do much (yet) beyond using the brand's [background](#) and [foreground](#) colors in the plot



# Example plot

```
1 library(tidyverse)
2 library(brand.yml)
3
4 brand <- read_brand_yaml("_brand.yml")
5
6 brand_theme <- quarto::theme_brand_ggplot2("_brand.yml") +
7   theme(
8     text = element_text(family = brand$typography$fonts[[1]]$family, size =
9   )
10
11 ggplot(mpg, aes(x = displ, y = hwy, color = drv)) +
12   geom_point() +
13   labs(title = "A plot") +
14   scale_color_manual(
15     values = c(
16       brand$color$palette$blue,
17       brand$color$palette$black,
18       brand$color$palette$yellow
```

A plot



# Your turn

1. In the first code chunk of `01-exercise.qmd`, replace `theme_minimal()` with `theme_brand_ggplot2()`. You will need to supply a brand file path ("`_brand.yml`").  
TODO
2. Re-render and see what changes.
3. Change the `foreground` and `background` colors in `_brand.yml` and rerun the code to create the plot.  
What changes?

05:00

# Your turn

1. In `01-exercise.qmd`, read in `_brand.yml` with `read_brand_yaml()` from the `{brand.yml}` package. Store it as a variable named `brand`. TODO
2. Change the color of the points to the brand's `blue` color and the line to the brand's `magenta` color. You'll use code that looks something like this:

```
geom_line(color = brand$color$palette$black)
```

3. **Bonus task!** Map a variable from the dataset to the color aesthetic and change the legend to use colors from the brand.

05:00

# Make your own `_brand.yml`

- In RStudio, go to File > New File > Text file
- In Positron or VS Code, use the Explorer panel to add a new file.

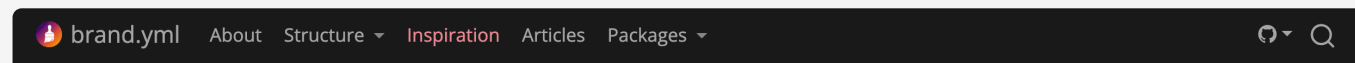
Save the file as `_brand.yml`

Or create a **Brand Extension** for sharing and distributing your brand

# Don't reinvent the wheel!

**DO NOT** try to write a `_brand.yml` from scratch!

Look at **the “Inspiration” page!**



## Inspiration

These brand guidelines and design systems served as an inspiration to the design of brand.yml. We've translated these guidelines into `_brand.yml` example files.

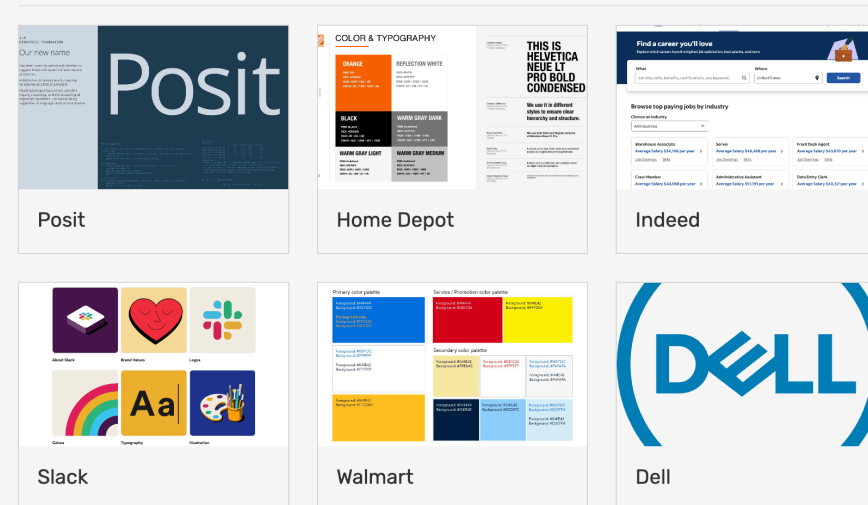
On this page

[Company Brand Guidelines](#)

[Report an issue](#)

[Edit this page](#)

## Company Brand Guidelines



Learn more: <https://posit-dev.github.io/brand.yml/inspiration/>

# Your turn

Create a brand file for yourself, your project, or your organization.

Don't try to do this from scratch! Use these resources:






- [The brand.yml documentation](#)
- [The brand.yml inspiration page](#)
- [This site's \\_brand.yml](#)

05:00





# Course outline

-  ~~Intro to Quarto~~
-  ~~Creating basic websites~~
-  ~~Advanced website features~~
-  ~~Publishing~~
-  ~~Customization and branding~~
- Interactivity



**Break!**