

# Customization and branding

# 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 though

Many common basic options:

<https://quarto.org/docs/output-formats/html-themes.html#basic-options>

# Total control with CSS

# Crash course in CSS

HTML elements can have IDs and classes

HTML elements can be nested inside each other

You can target HTML elements with different degrees of specificity:

- All `<h3>` headings
- H3 headings with the class `neato`
- `<a>` links that are inside a div with the id `quarto-sidebar`

# Browser inspector

Explore and edit any HTML and CSS right from your browser.

# Sass: CSS, but fancier

variables

rules that use variables

# Combining CSS and themes

asdf

# Your turn

Create a SCSS file and make a bunch of rules

# Branding

# CSS is a little inconvenient

- Doesn't play well across HTML outputs - 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

Organizations and projects generally have style guides for consistency

Urban Institute

This course! See the colophon



# BRAND YML

Unified branding with a simple YAML file

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

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

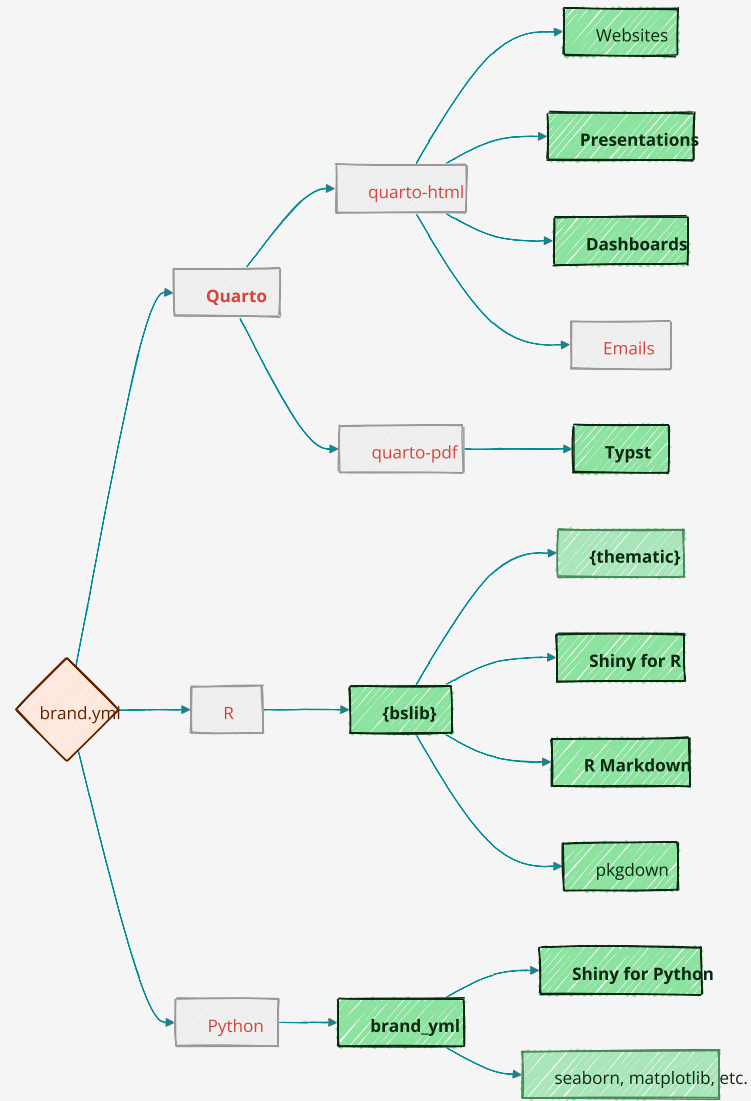
Branding can be frustrating

But brand.yml makes it easier

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

2. Apply that branding across almost all Quarto formats.

`brand.yml` also works with Shiny!



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

# 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: World Happiness Report
3   link:
4     home: https://www.worldhappiness.report/
5
6 logo:
7   images:
8     icon:
9       path: images/whr_logo.png
10      alt: World Happiness Report icon.
11   small: images/whr_logo.png
12
13 color:
14   palette:
15     dark-purple: "#93358c"
16     light-purple: "#f8f6fb"
17     teal: "#04afaf"
18     dark-blue: "#086992"
```

# How to apply brand.yml to 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: "World Happiness Report"
3 subtitle: "Happiness Trends and Contributing Factors (2011-2024)"
4 format: html
5 brand: org_theme.yml
6 ---
```

# Disable brand.yml

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

my-document.qmd

```
1 ---
2 title: "World Happiness Report"
3 subtitle: "Happiness Trends and Contributing Factors (2011-2024)"
4 format: html
5 brand: false
6 ---
```

# brand shortcode

Access some brand.yml values with a shortcode.

```
my_document.qmd
```

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

# Your turn

05:00

# What about plots?

# Theme helpers

The quarto packages contain theme helpers that apply branding to plots.

R

Python

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

R: <https://quarto-dev.github.io/quarto-r/articles/theme-helpers.html>

Python: <https://github.com/quarto-dev/quarto-python?tab=readme-overview#theme-helpers>

# Your turn

05:00

# brand.yml packages

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
```

# Templates



# Course outline

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



**Break!**