

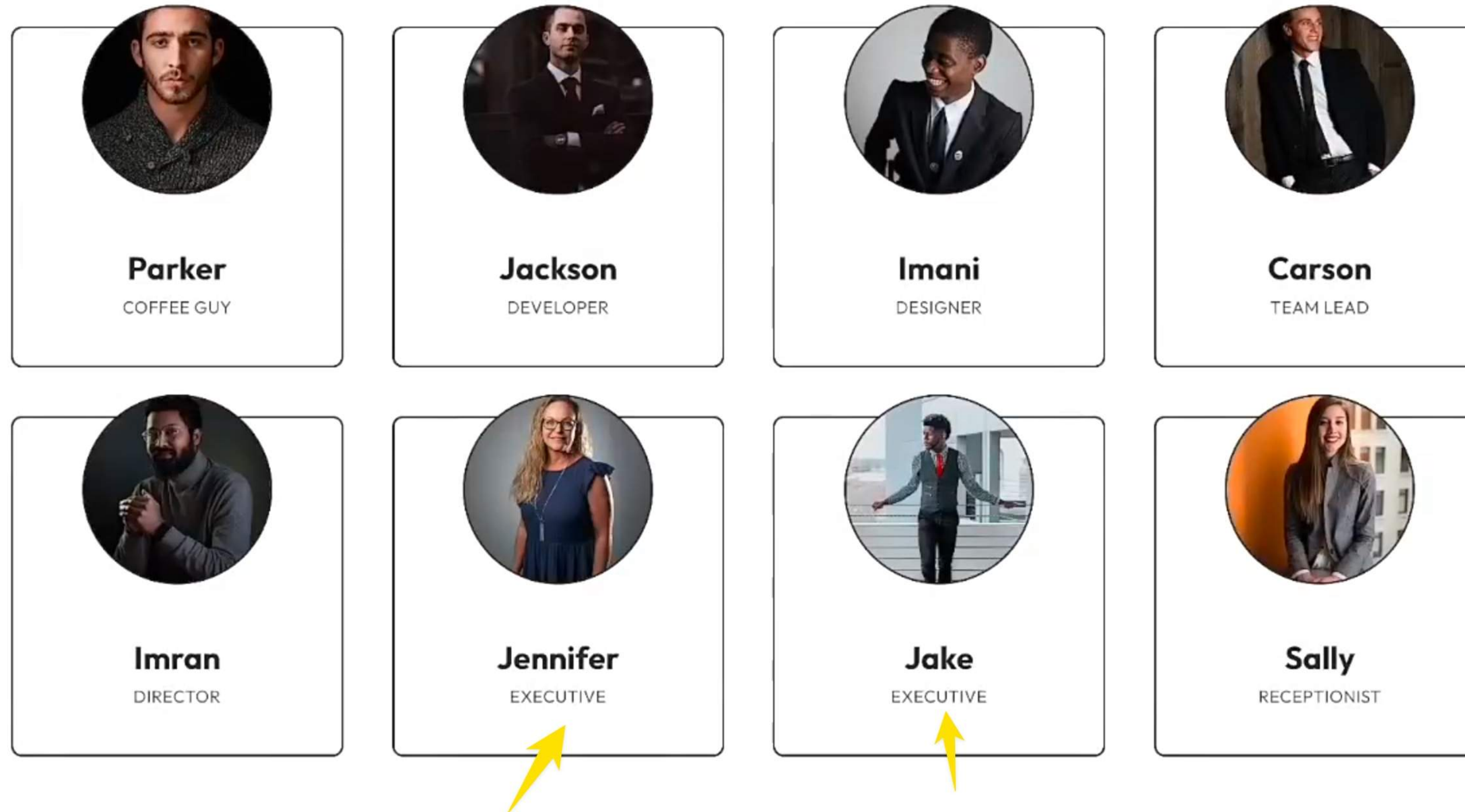
Video

Styling With Data Attributes

Data attributes can be targeted with CSS just like classes and IDs. In the grid below, let's order the grid by putting all the Executives first. Let's also make sure all executives are boxed cards with a dark background. Lastly, let's make sure Jake is always the first executive.

We want the Executives always to appear first

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.



Styling With Data Attributes

Data attributes can be targeted with CSS just like classes and IDs. In the grid below, let's order the grid by putting all the Executives first. Let's also make sure all executives are boxed cards with a dark background. Lastly, let's make sure Jake is always the first executive.

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.

 Parker COFFEE GUY	 Jackson DEVELOPER	 DESIGNER	 TEAM LEAD
 Imran DIRECTOR	 Jennifer EXECUTIVE	 Jake EXECUTIVE	 Sally RECEPTIONIST

The classes we have on the cards don't help us with this task. But attributes come to our rescue



[Home](#) [Units](#) [Practice](#) [Links](#) [Messagely](#)

What's a Data Attribute?

Classes and IDs are labels used for styling elements. Data attributes are another type of label, but the functionality goes beyond styling. Data attributes can be accessed and used by both CSS and JS in unique ways.

```
<div class="profile-card" data-profile-name="Kevin Geary">  
  <!-- Content -->  
</div>
```

In the above example, "Kevin Geary" is the value of the data attribute "data-profile-name." This data attribute identifies which card is Kevin's in the DOM and gives us an easy way to target Kevin's card specifically with CSS or JS.

What's an attribute selector?

IDs and classes are both "selectors" in CSS. But there's another kind of selector called an attribute selector. This allows you to select elements based on a specific attribute scenario.

```
[data-profile-name="Kevin Geary"] {  
  display: awesome;
```



specifically with CSS or JS.

What's an attribute selector?

IDs and classes are both “selectors” in CSS. But there’s another kind of selector called an attribute selector. This allows you to select elements based on a specific attribute scenario.

```
[data-profile-name="Kevin Geary"] {  
  display: awesome;  
}
```

Using the example above, we can target any element that has a data-profile-name attribute with a value equal to “Kevin Geary.”

The square brackets denote an attribute selector.

Styling With Data Attributes

Data attributes can be targeted with CSS just like classes and IDs. In the grid below, let’s order the grid by putting all the Executives first. Let’s also make sure all executives are boxed cards with a dark background. Lastly, let’s make sure Jake is always the first executive.

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.



Styling With Data Attributes

Data attributes can be targeted with CSS just like classes and IDs. In the grid below, let's order the grid by putting all the Executives first. Let's also make sure all executives are boxed cards with a dark background. Lastly, let's make sure Jake is always the first executive.

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.



Parker
COFFEE GUY



Jackson
DEVELOPER



Imani
DESIGNER



Carson
TEAM LEAD



Imran
DIRECTOR



Jennifer
EXECUTIVE



Jake
EXECUTIVE



Sally
RECEPTIONIST



Profile Card ...

#brxe-628b71 1

.fr-profile-card-delta

CONTE... STYLE

GRID ITEM

Grid column auto / auto

Grid row auto / auto

Use query loop ☒

Query ☐

HTML tag custom x

Custom tag li

Display flex

Flex wrap No wrap

Direction

Align self

Align main axis

Align cross axis

Column gap Normal

Row gap 20px

Flex grow 0

Flex shrink 1

Flex basis auto

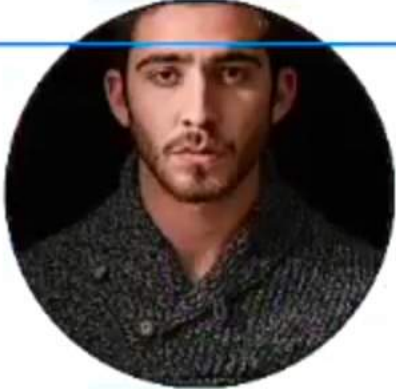
Order 0

Styling With Data Attributes

Data attributes can be targeted with CSS just like classes and IDs. In the grid below, let's order the grid by putting all the Executives first. Let's also make sure all executives are boxed cards with a dark background. Lastly, let's make sure Jake is always the first executive.

Only 1 card physically exists. All other cards are generated by the query loop

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.



Parker
COFFEE GUY



Jackson
DEVELOPER



Imani
DESIGNER



Carson
TEAM LEAD



Imran
DIRECTOR



Jennifer
EXECUTIVE



Jake
EXECUTIVE



Sally
RECEPTIONIST

Structure

Data Attr

Selector

ng

Container

Profile Grid Delta

Profile Card Delta

Displaying Content

JS Interactions

Superpowers

Section

08:49

Profile Card ...

#brxe-628b71 1

.fr-profile-card-delta

CONTE... STYLE

LAYOUT

TYPOGRAPHY

BACKGROUND

BORDER / BOX SHADOW

GRADIENT / OVERLAY

SHAPE DIVIDERS

TRANSFORM

CSS

ATTRIBUTES

data-profile-name

• Name

data-profile-name ⚡

• Value

{post_title}

+

ADD ATTRIBUTES

Custom attributes will be added to the most relevant HTML node.

Styling With Data Attributes

Data attributes can be targeted with CSS just like classes and IDs. In the grid below, let's order the grid by putting all the Executives first. Let's also make sure all executives are boxed cards with a dark background. Lastly, let's make sure Jake is always the first executive.

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.



Parker
COFFEE GUY



Jackson
DEVELOPER



Imani
DESIGNER



Carson
TEAM LEAD



Imran
DIRECTOR



Jennifer
EXECUTIVE



Jake
EXECUTIVE



Sally
RECEPTIONIST

Section

Structure

Data Attr

Selector

Styling

Container

Profile Grid Delta

Profile Card Delta

Displaying Content

JS Interactions

Superpowers



Post title

Elements

Console

1

Settings

Close

```
<section class="brxe-section">
  <div class="brxe-container">
    <ul class="brxe-container fr-profile-grid-uelta list--none">
      <li class="brxe-628b71 brxe-block fr-profile-card-delta" data-profile-name="Parker">
        <div class="brxe-80e4c5 brxe-block fr-profile-card-delta__body">
          <div class="brxe-b6a4a6 brxe-block fr-profile-card-delta__media-wrapper">
```

Styles

Computed

Layout

Event Listeners

Filter

:hov .cls +

```
element.style {
}

.fr-profile-card-delta {
  row-gap: var(--fr-card-gap);
  display: flex;
  flex-direction: column;
  position: relative;
  padding-top: 0;
  padding-right: var(--space-l);
  padding-bottom: var(--space-l);
  padding-left: var(--space-l);
  border: 2px solid var(--black);
  border-radius: var(--radius-m);
}

[class*=brxe-] {
  max-width: 100%;
}

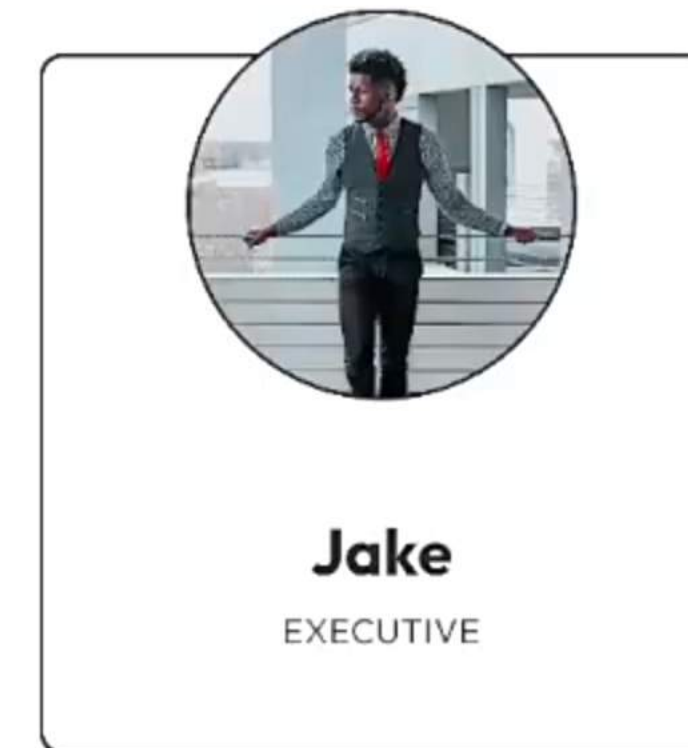
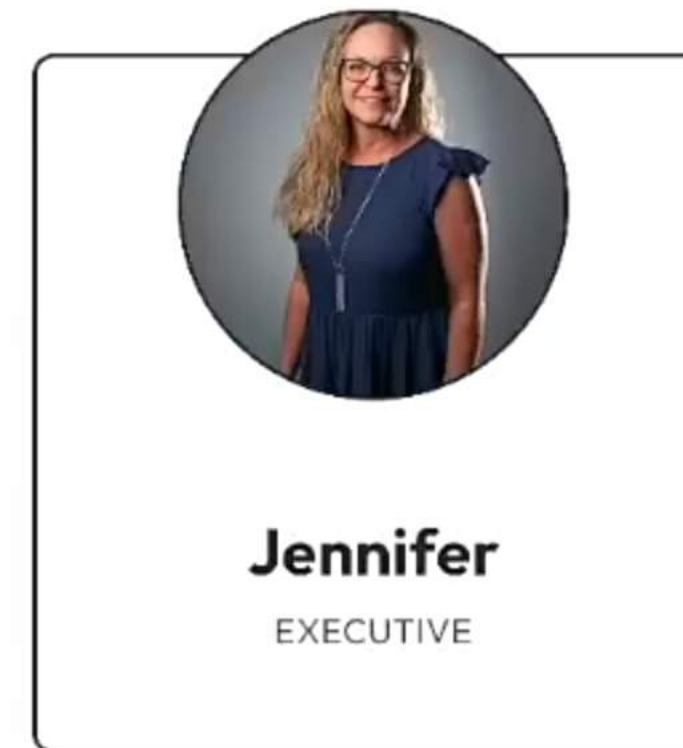
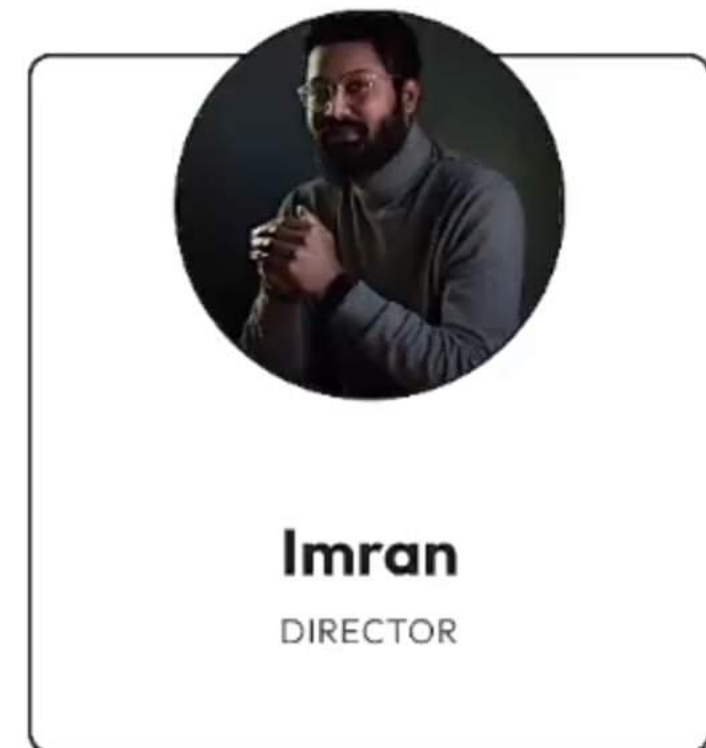
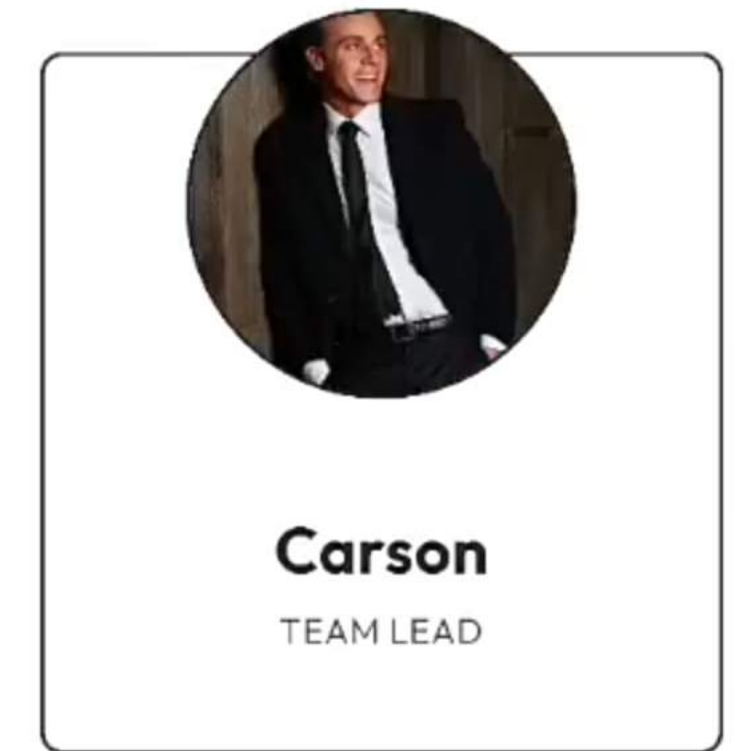
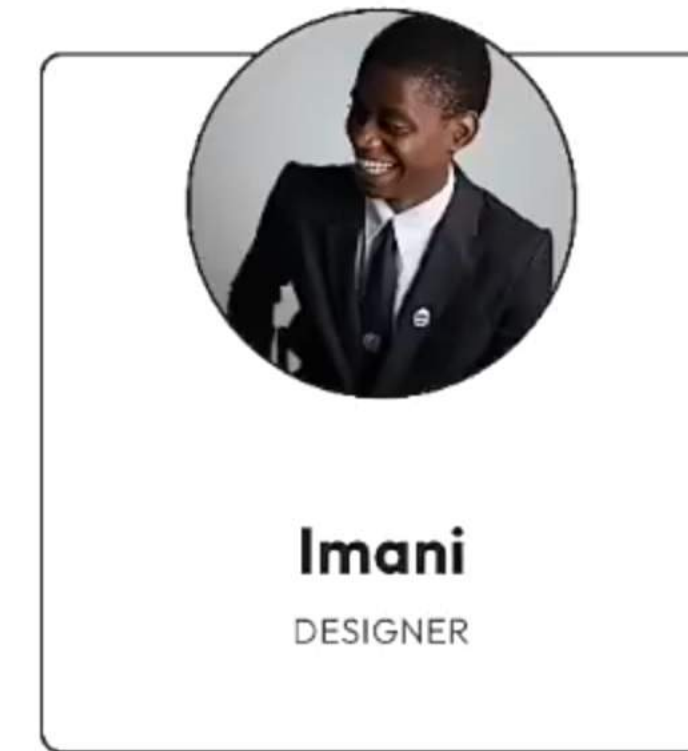
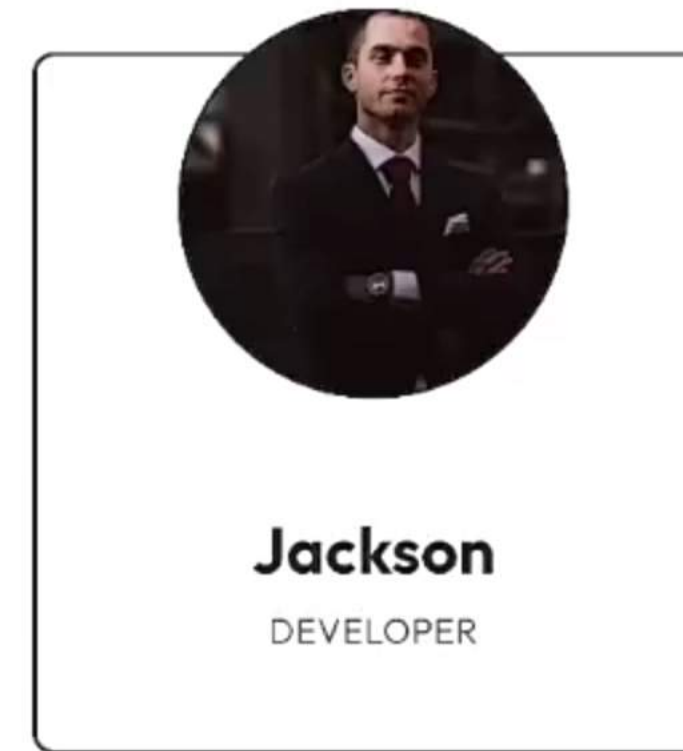
.brxe-block {
  align-items: flex-start;
  display: flex;
  flex-direction: column;
```

Styling with Data Attributes

Data attributes can be targeted with CSS just like classes and IDs. In the grid below, let's order the grid by putting all the Executives first. Let's also make sure all executives are boxed cards with a dark background. Lastly, let's make sure Jake is always the first executive.

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.

li.brxe-628b71.brxe-block fr-profile-card-delta 307.75 x 315.78



Elements

Console

1

Settings

Close

<ul class="brxe-container fr-profile-grid-delta list--none">

grid

><li class="brxe-628b71 brxe-block fr-profile-card-delta" data-profile-name="Parker">

...

flex

== \$0

><li class="brxe-628b71 brxe-block fr-profile-card-delta" data-profile-name="Jackson">

...

flex

><li class="brxe-628b71 brxe-block fr-profile-card-delta" data-profile-name="Imani">

...

flex

a.list--none

li.brxe-628b71.brxe-block.fr-profile-card-delta

>

Styles

Computed

Layout

Event Listeners

>>

Filter

:hov .cls + -

element.style {

}

.fr-profile-card-delta {

?bricks_pre...97145276:73

row-gap: var(--fr-card-gap);

display: flex;

flex-direction: column;

position: relative;

padding-top: 0;

padding-right: var(--space-l);

padding-bottom: var(--space-l);

padding-left: var(--space-l);

border: 2px solid black;

border-radius: var(--radius-m);

}

[class*=brxe-] {

frontend-li...697063483:1

max-width: 100%;

}

.brxe-block {

frontend-li...697063483:1

align-items: flex-start;

display: flex;

flex-direction: column;

Styling with Data Attributes

Data attributes can be targeted with CSS just like classes and IDs. In the grid below, let's order the grid by putting all the Executives first. Let's also make sure all executives are boxed cards with a dark background. Lastly, let's make sure Jake is always the first executive.

All cards now have a data attribute with their name



Parker

COFFEE GUY



Jackson

DEVELOPER



Imani

DESIGNER



Carson

TEAM LEAD



Imran

DIRECTOR



Jennifer

EXECUTIVE



Jake

EXECUTIVE



Carson

TEAM LEAD



Imani

DESIGNER

10:14

Profile Card ...

#brxe-628b71 1

.fr-profile-card-delta

CONTE... STYLE

LAYOUT

TYPOGRAPHY

BACKGROUND

BORDER / BOX SHADOW

GRADIENT / OVERLAY

SHAPE DIVIDERS

TRANSFORM

CSS

ATTRIBUTES

data-profile-name

data-profile-title

Name

data-profile-title

Value

{acf_team_member_}

+

ADD ATTRIBUTES

Custom attributes will be added to the most relevant HTML node.

Styling With Data Attributes

Data attributes can be targeted with CSS just like classes and IDs. In the grid below, let's order the grid by putting all the Executives first. Let's also make sure all executives are boxed cards with a dark background. Lastly, let's make sure Jake is always the first executive.

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.



Parker
COFFEE GUY



Jackson
DEVELOPER



Imani
DESIGNER



Carson
TEAM LEAD



Imran
DIRECTOR



Jennifer
EXECUTIVE



Jake
EXECUTIVE



Sally
RECEPTIONIST

data-profile-title

CPT

Structure

Data Attr

Selector

Styling

Container

Profile Grid Delta

Profile Card Delta

Displaying Content

JS Interactions

Superpowers

11:17

Elements

Console

Styles

```
<section class="brxe-section">
  <div class="brxe-container">
    <ul class="brxe-container fr-profile-grid-uelta list--none">
      <li class="brxe-628b71 brxe-block fr-profile-card-delta" data-profile-name="Parker" data-profile-title="Coffee Guy">
        <div class="brxe-80e4c5 brxe-block fr-profile-card-delta__body">
          <div class="brxe-b6a4a6 brxe-block fr-profile-card-delta__media-wrapper">
```

Filter

:hov .cls

```
element.style {
}

.fr-profile-card-delta {
  row-gap: var(--fr-card-gap);
  display: flex;
  flex-direction: column;
  position: relative;
  padding-top: 0;
  padding-right: var(--space-l);
  padding-bottom: var(--space-l);
  padding-left: var(--space-l);
  border: 2px solid var(--black);
  border-radius: var(--radius-m);
}

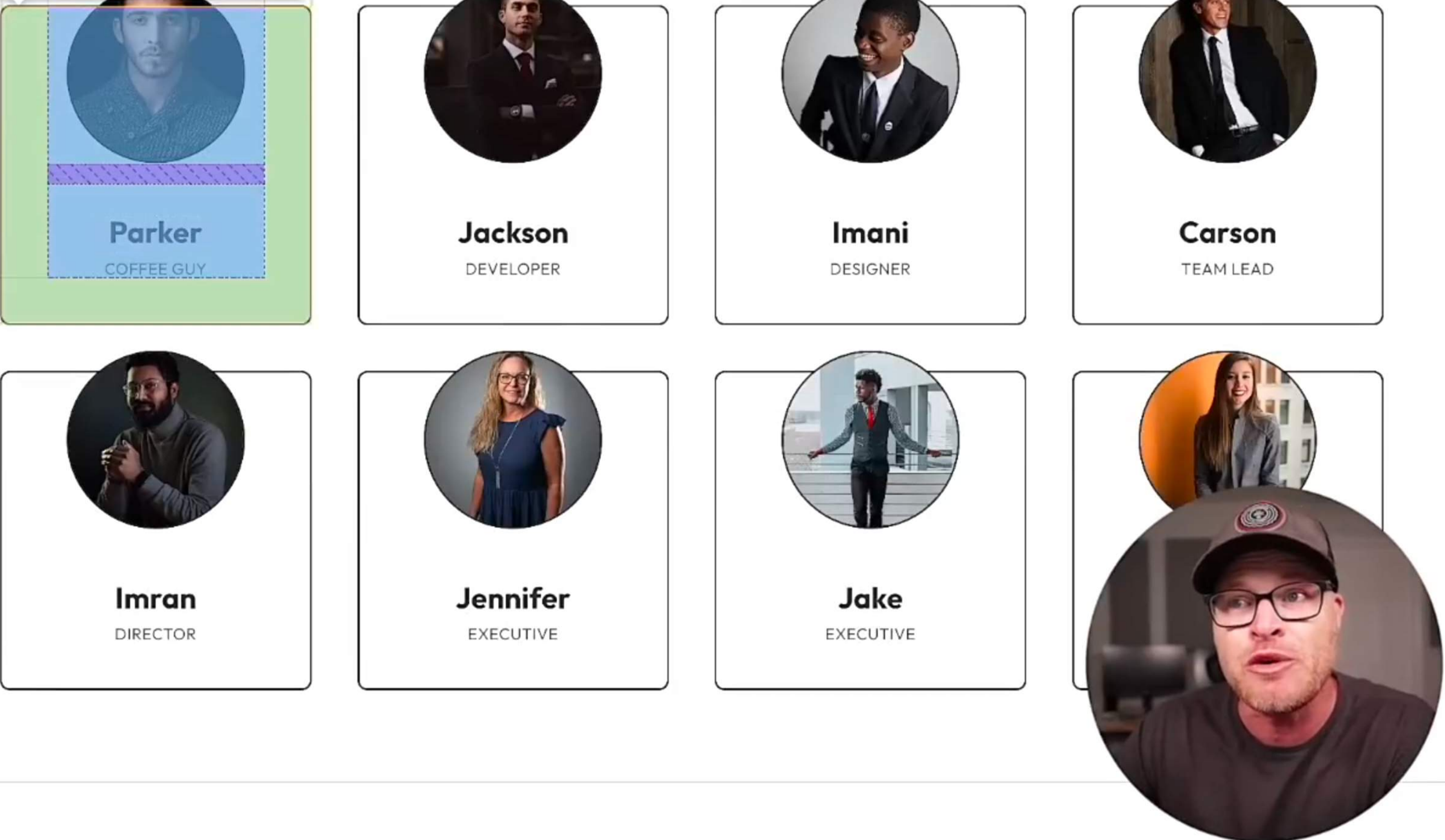
[class*=brxe-] {
  max-width: 100%;
}

.brxe-block {
  align-items: flex-start;
  display: flex;
  flex-direction: column;
```

Styling with Data Attributes

Data attributes can be targeted with CSS just like classes and IDs. In the grid below, let's order the grid by putting all the Executives first. Let's also make sure all executives are boxed cards with a dark background. Lastly, let's make sure Jake is always the first executive.

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.



11:26

Elements

Console

>>

Settings

Close

```
<li class="brxe-628b71 brxe-block fr-profile-card-delta" data-profile-name="Imran" data-profile-title="Director">...</li>
<li class="brxe-628b71 brxe-block fr-profile-card-delta" data-profile-name="Jennifer" data-profile-title="Executive">...</li>
<li class="brxe-628b71 brxe-block fr-profile-card-delta" data-profile-name="Jake" data-profile-title="Executive">...</li>
<li class="brxe-628b71 brxe-block fr-profile-card-delta" data-profile-name="Sally" data-profile-title="Executive">...</li>
```

Filter

:hov .cls + - [icon]

element.style {

}

.fr-profile-card-delta {

row-gap: var(--fr-card-gap);

display: flex;

flex-direction: column;

position: relative;

padding-top: 0;

padding-right: var(--space-l);

padding-bottom: var(--space-l);

padding-left: var(--space-l);

border: 2px solid var(--black);

border-radius: var(--radius-m);

}

[class*=brxe-] {

max-width: 100%;

}

.brxe-block {

align-items: flex-start;

display: flex;

flex-direction: column;

}

Styling with Data Attributes

Data attributes can be targeted with CSS just like classes and IDs. In the grid below, let's order the grid by putting all the Executives first. Let's also make sure all executives are boxed cards with a dark background. Lastly, let's make sure Jake is always the first executive.

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.



Parker
COFFEE GUY



Jackson
DEVELOPER



Imani
DESIGNER



Carson
TEAM LEAD



Imran
DIRECTOR



Jennifer
EXECUTIVE



Jake
EXECUTIVE



Sally
EXECUTIVE



Iman
EXECUTIVE

11:36

Profile Card Delta

cus

.fr-profile-card-delta

.fr-profile-card-delta

Custom CSS (css)

1 %root%[]

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

Custom tag

li

Styling With Data Attributes

Data attributes can be targeted with CSS just like classes and IDs. In the grid below, let's order the grid by putting all the Executives first. Let's also make sure all executives are boxed cards with a dark background. Lastly, let's make sure Jake is always the first executive.

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.



Parker
COFFEE GUY



Jackson
DEVELOPER



Imani
DESIGNER



Carson
TEAM LEAD



Imran
DIRECTOR



Jennifer
EXECUTIVE



Jake
EXECUTIVE



Sally
RECEPTIONIST

Structure

Data Attr

Selector

Styling

Container

Profile Grid Delta

Profile Card Delta

Displaying Content

JS Interactions

Superpowers

Important: We have to attach the attribute selector to the class (it's not a child, it's on the card)



Profile Card Delta

cus

.fr-profile-card-delta

.fr-profile-card-delta

Custom CSS (css)

```
1 %root%[data-profile-title="Executive"] {
2   border: 5px solid red;
3 }
```

Now we're selecting our executive cards already

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

Custom tag

li

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.



Parker

COFFEE GUY



Jackson

DEVELOPER



Imani

DESIGNER



Imran

DIRECTOR



Jennifer

EXECUTIVE



Jake

EXECUTIVE

Structure

Data Attr

Selector

Styling

Container

Profile Grid Delta

Profile Card Delta

Displaying Content

JS Interactions

Superpowers

Displaying Content With Data Attributes

13:53

Profile Card Delta

cus

.fr-profile-card-delta

.fr-profile-card-delta

Custom CSS (css)

```
1 %root%[data-profile-title="Executive"] {  
2   order: -1;  
3 }
```

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

Custom tag

li

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.

Jennifer
EXECUTIVE

Jake
EXECUTIVE

Parker
COFFEE GUY

Imani
DESIGNER

Carson
TEAM LEAD

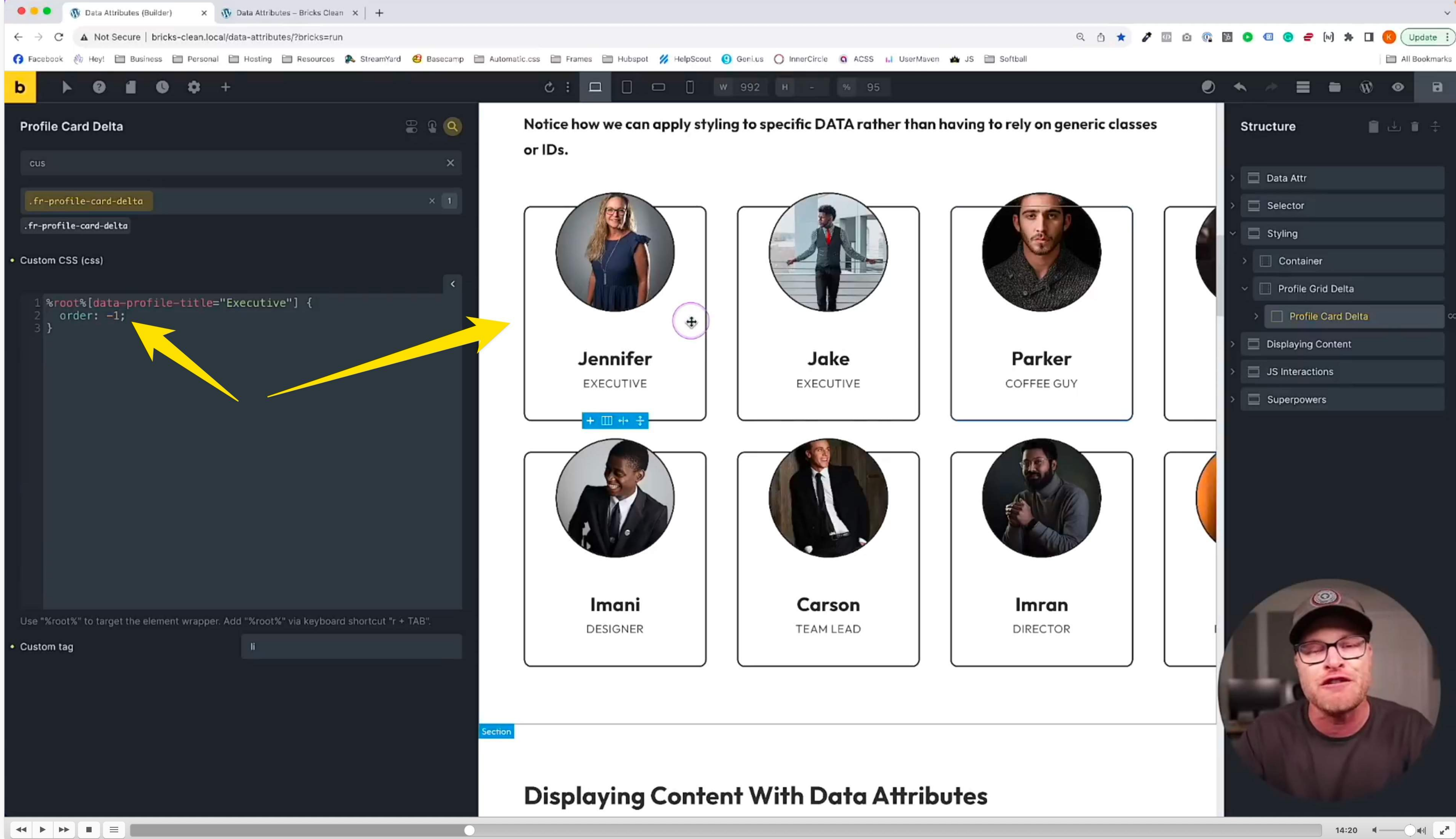
Imran
DIRECTOR

Structure

- Data Attr
- Selector
- Styling
 - Container
 - Profile Grid Delta
 - Profile Card Delta
- Displaying Content
- JS Interactions
- Superpowers

Section

Displaying Content With Data Attributes



Profile Card Delta

cus

.fr-profile-card-delta

.fr-profile-card-delta

Custom CSS (css)

```
1 %root%[data-profile-title="Executive"] {
2   order: -1;
3   background-color: var(--black);
4 }
5
6 %root%[data-profile-title="Executive"] :is(%root%__title,
7   %root%__name) {
8   color: var(--white);
9 }
```

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

Custom tag

li

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.



Jennifer
EXECUTIVE



Jake
EXECUTIVE



Parker
COFFEE GUY



Imani
DESIGNER



Carson
TEAM LEAD



Imran
DIRECTOR

Section

Displaying Content With Data Attributes

Structure

Data Attr

Selector

Styling

Container

Profile Grid Delta

Profile Card Delta

Displaying Content

JS Interactions

Superpowers



16:15

Not Secure | bricks-clean.local/data-attributes/?bricks=run

Profile Card Delta

cus

.fr-profile-card-delta

Custom CSS (css)

```
1 %root%[data-profile-title="Executive"] {
2   order: -1;
3   background-color: var(--black);
4 }
5
6 %root%[data-profile-title="Executive"] :is(%root%__title,
7   %root%__name) {
8   color: var(--white);
9 }
10
11 %root%[data-profile-name="Jake"] {
12   order: -2;
13 }
```

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

Custom tag

li

Notice how we can apply styling to specific DATA rather than having to rely on generic classes or IDs.

Jake
EXECUTIVE

Jennifer
EXECUTIVE

Parker
COFFEE GUY

Imani
DESIGNER

Carson
TEAM LEAD

Imran
DIRECTOR

Structure

- Data Attr
- Selector
- Styling
 - Container
 - Profile Grid Delta
 - Profile Card Delta
- Displaying Content
- JS Interactions
- Superpowers

Displaying Content With Data Attributes

17:41

PB101- L19 - Dynamic Styling with Data Attributes & Attribute Selectors.mp4

Data Attributes (Builder)

Data Attributes – Bricks Clean

Data Attributes – Bricks Clean

+

Not Secure | bricks-clean.local/data-attributes/?bricks_preview=1697145276

Facebook

Hey!

Business

Personal

Hosting

Resources

StreamYard

Basecamp

Automatic.css

Frames

Hubspot

HelpScout

Genius

InnerCircle

ACSS

UserMaven

JS

Softball

All Bookmarks

Bricks Clean

Customize

0

New

Edit Page

Edit with Bricks

Render with Bricks

Automatic CSS

Howdy, admin

DESIGNER

TEAM LEAD

DIRECTOR

RECEPTIONIST

Displaying Content With Data Attributes

You can print the content of a data attribute using the attr() function in CSS. Let's use a data attribute to output the count in the notification bubble attached to the button below.

Inbox

8

JS "Interactions" With Data Attributes

Data attributes are often better and more efficient for JS interactions than classes, especially when you're dealing with a multiple possible values mapped to a single key. In the example below, we'll trigger a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaws in the class targeting method over the data attribute method**



18:59

Button

cus

#brxe-btxjff

.btn--black

• Custom CSS (css)

```
1 %root% {
2   position: relative;
3 }
4
5 %root%::before {
6   content: '8';
7   position: absolute;
8   background-color: var(--action);
9   color: var(--action-ultra-dark);
10  font-size: calc(var(--text-s) * 1.1);
11  font-weight: 700;
12  aspect-ratio: 1/1;
13  width: 3ch;
14  height: 3ch;
15  display: grid;
16  place-items: center;
17  border-radius: 50vw;
18  top: -1ch;
19  right: -1ch;
20 }
```



Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

Displaying Content With Data Attributes

You can print the content of a data attribute using the attr() function in CSS. Let's use a data attribute to output the count in the notification bubble attached to the button below.

Inbox

JS "Interactions" With Data Attributes

Data attributes are often better and more efficient for JS interactions than classes, especially when you're dealing with a multiple possible values mapped to a single key. In the example below, we'll trigger a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaws in the class targeting method over the data attribute method.**

Using Classes:

Success

Warning

Danger

Structure

- Data Attr
- Selector
- Styling
 - Container
 - Profile Grid Delta
 - Profile Card Delta
 - Displaying Content
 - Container
 - Heading
 - Rich Text
 - Container
 - Button
 - JS Interactions
 - Superpowers



19:37

#brxe-btxjff1

.btn--black

CONTE...STYLE

TYPOGRAPHY

BACKGROUND

BORDER / BOX SHADOW

GRADIENT / OVERLAY

TRANSFORM

CSS

ATTRIBUTES

data-notification-value

Name

data-notification-val⚡

ValueSelect dynamic data

9⚡

+ADD ATTRIBUTES

Custom attributes will be added to the most relevant HTML node.

Displaying Content With Data Attributes

You can print the content of a data attribute using the attr() function in CSS. Let's use a data attribute to output the count in the notification bubble attached to the button below.

Inbox8

JS "Interactions" With Data Attributes

Data attributes are often better and more efficient for JS interactions than classes, especially when you're dealing with a multiple possible values mapped to a single key. In the example below, we'll trigger a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaws in the class targeting method over the data attribute method.**

Using Classes:SuccessWarningDanger

Structure

Data Attr

Selector

Styling

Container

Profile Grid Delta

Profile Card Delta

Displaying Content

Container

Heading

Rich Text

Container

Button

JS Interactions

Superpowers

You can print the content of a data attribute using the `attr()` function in CSS. Let's use a data attribute to output the count in the notification bubble attached to the button below.

You can print the content of a data attribute using the `attr()` function in CSS. Let's use a data attribute to output the count in the notification bubble attached to the button below.



Data attributes are often better and more efficient for JS interactions than classes, especially when you're dealing with multiple possible values mapped to a single key. In the example below, we'll make a change in a status badge using classes first, and then again using data attributes. **Notice the flaws in the class targeting method over the data attribute method.**

Data attributes are often better and more efficient for JS interactions than classes, especially when you're dealing with multiple possible values mapped to a single key. In the example below, we'll make a change in a status badge using classes first, and then again using data attributes. **Notice the flaws in the class targeting method over the data attribute method.**



Button

cus

#brxe-btxjff

.btn--black

Custom CSS (css)

```
1 %root% {
2   position: relative;
3 }
4
5 %root%::before {
6   content: attr(data-notification-value);
7   position: attr(data-notification-value);
8   background-color: var(--action);
9   color: var(--action-ultra-dark);
10  font-size: calc(var(--text-s) * 1.1);
11  font-weight: 700;
12  aspect-ratio: 1/1;
13  width: 3ch;
14  height: 3ch;
15  display: grid;
16  place-items: center;
17  border-radius: 50vw;
18  top: -1ch;
19  right: -1ch;
20 }
```

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

You can print the con Width | data attribute using the attr() function in CSS. Let's use a data attribute to output the count in the notification bubble attached to the button below.

Inbox

Bricks is currently not showing the value in the Builder but it's there

JS "Interactions" With Data Attributes

Data attributes are often better and more efficient for JS interactions than classes, especially when you're dealing with a multiple possible values mapped to a single key. In the example below, we'll trigger a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaws in the class targeting method over the data attribute meth**

GOOD

Using Classes:

SuccessWarningDanger

Using a Data Attribute:

SuccessWarningDanger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Structure

Selector

Styling

Container

Profile Grid Delta

Profile Card Delta

Displaying Content

Container

Heading

Rich Text

Container

Button

JS Inter

S

22:19

FacebookHey!BusinessPersonalHostingResourcesStreamYardBasecampAutomatic.cssFramesHubspotHelpScoutGeni.usInnerCircleACSSUserMavenJSSoftballAll Bookmarks

ElementsConsole>>1

```
<section class="brxe-section">...</section>
<section class="brxe-section">flex
  <div class="brxe-container">...</div>flex
  <div class="brxe-container">flex
    <a id="brxe-btxjff" class="brxe-button btn--black bricks-button bricks-background-primary" data-notification-value="9" href="#">
      == $0
      ::beforegrid
      "Inbox"
    </a>
```

a#brxe-btxjff.brxe-button.btn--black.bricks-button.bricks-ba

StylesComputedLayoutEvent Listeners>>

Filter: :hov .cls +

```
element.style {
}

#brxe-btxjff {
  position: relative;
}

#brxe-btxjff {
  border-radius: var(--radius-m);
}

.btn--black {
  --btn-background: var(--black);
  --btn-background-hover: var(--neutral-dark);
  --btn-text-color: var(--white);
  --btn-text-color-hover: var(--white);
  --btn-border-color: var(--black);
  --btn-border-color-hover: var(--neutral-dark);
  --btn-outline-background-hover: var(--white);
  --btn-outline-border-hover: var(--white);
  --btn-outline-text-color: var(--shade-black);
  --btn-outline-text-color-hover: var(--action-ultra-light);
  --focus-color: var(--action);
}
```

Bricks CleanCustomize0+NewEdit PageEdit with BricksRender with BricksAutomatic CSSHowdy, admin

Displaying Content With Data Attributes

You can print the content of a data attribute using the attr() function in CSS. Let's use a data attribute to output the count in the notification bubble attached to the button below.

Inbox9

JS "Interactions" With Data Attributes

Data attributes are often better and more efficient for JS interactions than classes when you're dealing with a multiple possible values mapped to a single key. In the example below, we'll trigger a change in a status badge using classes first, and then again using data attributes.

fatal flaws in the class targeting method over the data attribute method.

bricks-clean.local/data-attributes/?bricks_preview=1697145276#

22:25

JS "Interactions" With Data Attributes

Data attributes are often better and more efficient for JS interactions than classes, especially when you're dealing with a multiple possible values mapped to a single key. In the example below, we'll trigger a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaws in the class targeting method over the data attribute method.**

Status

Using Classes:

Success

Warning

Danger

Using a Data Attribute:

Success

Warning

Danger



Button

#brxe-ghetas 1

.status

CONTE...STYLE

LAYOUT

TYPOGRAPHY

BACKGROUND

BORDER / BOX SHADOW

GRADIENT / OVERLAY

TRANSFORM

CSS

ATTRIBUTES

data-status

Name

data-status

Value

Status

ADD ATTRIBUTES

Custom attributes will be added to the most relevant HTML node.

a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaws in the class targeting method over the data attribute method.**

Using Classes:

Success

Warning

Danger

Using a Data Attribute:

Success

Warning

Danger

Structure

Data Attr

Selector

Styling

Container

Profile Grid Delta

Profile Card Delta

Displaying Content

Container

Heading

Rich Text

Container

Button

JS Interactions

Container

Container

Button

Container

Basic Text

Button

Button

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Target elements with a specific attribute

```
a[class] {
  border: 5px solid red;
}

a[title] {
  border: 5px solid red;
}
```



JS "Interactions" With Data Attributes

Data attributes are often better and more efficient for JS interactions than classes, especially when you're dealing with a multiple possible values mapped to a single key. In the example below, we'll trigger a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaws in the class targeting method over the data attribute method.**

Status

Our goal is to change the status by clicking one of the below buttons

Using Classes:

Success

Warning

Danger

Using a Data Attribute:

Success

Warning

Danger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Target elements with a specific attribute

```
a[class] {
```



ResourcesStreamYard

Not Secure | bricks-cl

FacebookHey!Business

bricks-cl

Update

FacebookHey!Business

bricks-cl

Update

Button

#brxe-ghetas

.btn--black

CONTENT

LAYOUT

TYPOGRAPHY

BACKGROUND

BORDER / BOX SHADOW

GRADIENT / OVERLAY

TRANSFORM

CSS

CSS Filters

Learn more about CSS filters

Transition

all 0.3s ease 0s

Learn more about CSS transitions

Custom CSS

1 %root% {
 color: firebrick;
}

Interactions

1

are often better and more efficient for JS interactions than classes, especially when

with a multiple possible values mapped to a single key. In the example below, we'll trigger

status badge using classes first, and then again using data attributes. **Notice the fatal**

class targeting method over the data attribute method.

Using Classes:

Success

Warning

Danger

Using a Data Attribute:

Success

Warning

Danger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Target elements with a specific attribute

```
a[class] {  
  border: 5px solid red;  
}  
  
a[title] {  
  border: 5px solid red;  
}
```

25:32

Button

INTERACTIONS

Click

Interaction ID: rpoxhu

Trigger: Click

Action: Set attribute

Key: class

Value: status--success

Target: CSS selector

CSS selector: .status

Run only once

INTERACTION CONDITIONS

Run this interaction if the following conditions are met.

ADD CONDITION

Relation: And

Data attributes are often better and more efficient for JS interactions than classes, especially when you're dealing with a multiple possible values mapped to a single key. In the example below, we'll trigger a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaws in the class targeting method over the data attribute method.**

Using Classes:

Success

Warning

Danger

Using a Data Attribute:

Success

Warning

Danger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Target elements with a specific attribute

```
a[class] {
  border: 5px solid red;
}

a[title] {
  border: 5px solid red;
}
```

Structure

Data Attr

Selector

Styling

Container

Profile Grid Delta

Profile Card Delta

Displaying Content

Container

Heading

Rich Text

Container

Button

JS Interactions

Container

Container

Button

Container

Basic Text

Button

Button

26:25

Not Secure | bricks-clean.local/data-attributes/?bricks_preview=1697145276

FacebookHey!BusinessPersonalHostingResourcesStreamYardBasecampAutomatic.cssFramesHubspotHelpScoutGeniusInnerCircleACSSUserMavenJSSoftballAll Bookmarks

ElementsConsole>>1

<section class="brxe-section">flex

><div class="brxe-container">...</div>fle

><div class="brxe-container">flex

...><span class="brxe-button status bricks-button bricks-background-primary status--success" data-status="Status" data-interactions="{"id":"vtlgjd","trigger":"click","action":"setAttribute","actionAttributeKey":"class","actionAttributeValue":"status--disabled",&

span.brxe-button.status.bricks-button.bricks-background-pi

StylesComputedLayoutEvent Listeners>>

Filter: :hov .cls +, ,

element.style { }

span[class*=brxe-]:not(.brxe-section):not(.brxe-container):not(.brxe-block):not(.brxe-div) { width: auto; }

.status { padding: 1em 1.5em; color: var(--text-color, var(--neutral-dark)); background-color: var(--background-color, var(--neutral-ultra-light)); }

.brxe-button { width: auto; }

.bricks-background-primary { background-color: var(--bricks-color primary); }

.bricks-button {

JS "Interactions" With Data Attributes

The class is added after clicking the button

Data attributes are often better and more efficient for JS interactions than classes, especially when you're dealing with a multiple possible values mapped to a single key. In the example below, we'll trigger a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaw in the class targeting method over the data attribute method.**

Using Classes:

SuccessWarningDanger

Using a Data Attribute:

SuccessWarningDanger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Target elements with a specific attribute

```
a[class] {
```

Howdy, admin

26:50

Button

#brxe-rqzvvv1

.status

CONTENTSTYLE

LAYOUT

TYPOGRAPHY

BACKGROUND

BORDER / BOX SHADOW

GRADIENT / OVERLAY

TRANSFORM

CSS

CSS Filters

Learn more about CSS filters

Transition

all 0s ease 0s

Learn more about CSS transitions

Custom CSS

1 %root%.status--success {
2 background-color: var(--success-light);
3 color: var(--success-dark);
4 }

Not Secure | bricks-clean.local/data-attributes/?bricks=run

FacebookHey!BusinessPersonalHostingResourcesStreamYardBasecampAutomatic.cssFramesHubspotHelpScoutGeni.usInnerCircleACSSUserMavenJSSoftballAll Bookmarks

W1404H100

Structure

Data Attr

Selector

Styling

Container

Profile Grid Delta

Profile Card Delta

Displaying Content

Container

Heading

Rich Text

Container

Button

JS Interactions

Container

Container

Button

Container

Basic Text

Button

Button

Using Classes:

SuccessWarningDanger

Using a Data Attribute:

SuccessWarningDanger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Target elements with a specific attribute

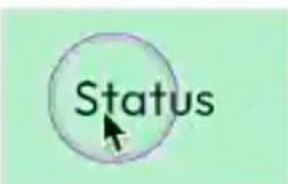
```
a[class] {  
  border: 5px solid red;  
}  
  
a[title] {  
  border: 5px solid red;  
}
```

27:19



JS "Interactions" With Data Attributes

Data attributes are often better and more efficient for JS interactions than classes, especially when you're dealing with a multiple possible values mapped to a single key. In the example below, we'll trigger a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaws in the class targeting method over the data attribute method.**



Using Classes:

Success

Warning

Danger

Using a Data Attribute:

Success

Warning

Danger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Target elements with a specific attribute

```
a[class] {
```



Not Secure | bricks-clean.local/data-attributes/?bricks_preview=1697145276

FacebookHey!BusinessPersonalHostingResourcesStreamYardBasecampAutomatic.cssFramesHubspotHelpScoutGeniusInnerCircleACSSUserMavenJSSoftballAll Bookmarks

ElementsConsole>>1⚙️⋮✕

```
<span id="brxe-ghetas" class="brxe-button status bricks-button bricks-background-primary status--warning status--danger status--success" data-status="Status" data-interactions="[{"id":"vtlgjd","trigger":"click","action":"setAttribute","actionAttributeKey":"class","actionAttributeValue":"status--disabled","target":"custom","targetSelector":":btn--status&quot;
```

span#brxe-ghetas.brxe-button.status.bricks-button.bricks-b

StylesComputedLayoutEvent Listeners>>

Filter: :hov .cls + 🗑️ 📏

```
element.style {
}
#brxe-ghetas.status--danger {
  background-color: var(--danger-light);
  color: var(--danger-dark);
}
#brxe-ghetas.status--warning {
  background-color: var(--warning-light);
  color: var(--warning-dark);
}
#brxe-ghetas.status--success {
  background-color: var(--success-light);
  color: var(--success-dark);
}
span[class*=brxe-]:not(.brxe-section):not(.brxe-container):not(.brxe-block):not(.brxe-div) {
  width: auto;
}
.status {
```

?bricks_pre...7145276:108

PB101- L19 - Dynamic Styling with Data Attributes & Attribute Selectors.mp4

Bricks CleanCustomize0+NewEdit PageEdit with BricksRender with BricksAutomatic CSSHowdy, admin

JS "Interactions" With Data Attributes

Data attributes are often better and more efficient for JS interactions than classes, especially when you're dealing with a multiple possible values mapped to a single key. In the example below, we'll trigger a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaws in the class targeting method over the data attribute method.**

Status

Everything works as long as we click again on Success after we clicked on Danger...

Using Classes:

SuccessWarningDanger

Using a Data Attribute:

SuccessWarningDanger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Not Secure | bricks-clean.local/data-attributes/?bricks_preview=1697145276

FacebookHey!BusinessPersonalHostingResourcesStreamYardBasecampAutomatic.cssFramesHubspotHelpScoutGeniusInnerCircleACSSUserMavenJSSoftballAll Bookmarks

ElementsConsole>>1⚙️⋮✕

⋮

<span id="brxe-ghetas" class="brxe-button status bricks-button bricks-background-primary status--warning status--danger status--success" data-status="Status" data-interactions="[""id":"vtlgjd";"trigger":"click";"action":"setAttribute";"actionAttributeKey":"class";"actionAttributeValue":"status--disabled";"target":"custom";"targetSelector":"btn--status"]

span#brxe-ghetas.brxe-button.status.bricks-button.bricks-b

StylesComputedLayoutEvent Listeners>>

Filter: :hov .cls +🔍📄

element.style {

#brxe-ghetas.status--danger { post-1207.m...697147510:1
background-color: var(--danger-light);
color: var(--danger-dark);

#brxe-ghetas.status--warning { post-1207.m...697147510:1
background-color: var(--warning-light);
color: var(--warning-dark);

#brxe-ghetas.status--success { post-1207.m...697147510:1
background-color: var(--success-light);
color: var(--success-dark);

span[class*=brxe-]:not(.brxe-section):not(.brxe-container):not(.brxe-block):not(.brxe-div) { frontend-li...697063483:1
width: auto;

.status { ?bricks_pre...7145276:108

PB101- L19 - Dynamic Styling with Data Attributes & Attribute Selectors.mp4

Bricks CleanCustomize0+NewEdit PageEdit with BricksRender with BricksAutomatic CSSHowdy, admin

JS "Interactions" With Data Attributes

Data attributes are often better and more efficient for JS interactions than classes, especially when

The problem: Everytime we click on a button the class is added and not deleted after clicking the next button

flaws in the class targeting method over the data attribute method.

Status

Using Classes:

SuccessWarningDanger

Using a Data Attribute:

SuccessWarningDanger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

28:59

🔊

🔍

#brxe-rsgmrk1

.status

CONTENTSTYLE

Transition

all 0s ease 0s

[Learn more about CSS transitions](#)

Custom CSS

```
1 %root[data-status="success"] {
2   background-color: var(--success-light);
3   color: var(--success-dark);
4 }
5 %root[data-status="warning"] {
6   background-color: var(--warning-light);
7   color: var(--warning-dark);
8 }
9 %root[data-status="light"] {
10  background-color: var(--danger-light);
11  color: var(--danger-dark);
12 }
```

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

CSS classes

Using a Data Attribute:

SuccessWarningDanger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Target elements with a specific attribute

```
a[class] {
  border: 5px solid red;
}

a[title] {
  border: 5px solid red;
}
```

CHAMP

Link with a title

Link with no title

Target elements with a specific attribute with a specific value

```
a[class="btn"] {
  border: 5px solid red;
}

a[href="/"] {
  border: 5px solid red;
}
```

Structure

Data Attr

Selector

Styling

Container

Profile Grid Delta

Profile Card Delta

Displaying Content

Container

Heading

Rich Text

Container

Button

JS Interaction

Button

Button



30:54

Click

Interaction ID: sbkmce

Trigger: Click

Action: Set attribute

Key: data-status

Value: success

Target: CSS selector

CSS selector: .status

Run only once: ☐

INTERACTION CONDITIONS
Run this interaction if the following conditions are met.

Using Classes:

Success

Warning

Danger

Using a Data Attribute:

Success

Warning

Danger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Target elements with a specific attribute

```
a[class] {
  border: 5px solid red;
}
```

Structure

- Heading
- Rich Text
- Container
 - Button
- JS Interactions
 - Container
 - Container
 - Button
 - Container
 - Basic Text
 - Button
 - Button
 - Button
 - Container



Not Secure | bricks-clean.local/data-attributes/?bricks_preview=1697145276

FacebookHey!BusinessPersonalHostingResourcesStreamYardBasecampAutomatic.cssFramesHubspotHelpScoutGeniusInnerCircleACSSUserMavenJSSoftballAll Bookmarks

ElementsConsole1

```
<span id="brxe-ghetas" class="brxe-button status bricks-button bricks-background-primary" data-status="success" data-interactions="{"id":"vtlgjd","trigger":"click","action":"setAttribute","actionAttributeKey":"class","actionAttributeValue":"status--disabled","target":"custom","targetSelector":"btn--status"}" data-interaction-id="cceb3">flex == $0
```

span#brxe-ghetas.brxe-button.status.bricks-button.bricks-b

StylesComputedLayoutEvent Listeners>>

Filter: :hov .cls + -

element.style { }

#brxe-ghetas[data-status="success"] {
background-color: var(--success-light);
color: var(--success-dark);
}

span[class*=brxe-]:not(.brxe-section):not(.brxe-container):not(.brxe-block):not(.brxe-div) {
width: auto;
}

.status {
padding: 1em 1.5em;
color: var(--text-color, var(--neutral-dark));
background-color: var(--background-color, var(--neutral-ultra-light));
}

.brxe-button {
width: auto;
}

PB101- L19 - Dynamic Styling with Data Attributes & Attribute Selectors.mp4

Bricks CleanCustomize0+ NewEdit PageEdit with BricksRender with BricksAutomatic CSSHowdy, admin

success

Now we also get the value

Using Classes:

SuccessWarningDanger

Using a Data Attribute:

SuccessWarningDanger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common e

Target elements with a specific attribute

a[class] {

31:50

Console

Not Se

1697145276

Facebook

Hey

Basecamp

Automatic.css

Frames

Hubspot

HelpScout

Geni.us

InnerCircle

ACSS

UserMaven

JS

Softball

All Bookmarks

Update

Bricks Clean

Customize

0

New

Edit Page

Edit with Bricks

Render with Bricks

Automatic CSS

Howdy, admin

="brxe-ghetas" clas

cks-button bricks-back

atus="success" data-i

quot;:"vtlgjd&qu

quot;cli

attribute";"t

arg

" data

quot;class&qu

The attribute is changing and not adding up like classes

success

Using Classes:

Success

Warning

Danger

Using a Data Attribute:

Success

Warning

Danger

Styles

Computed

Layout

Event Listeners

>>

Filter

:hov .cls +

element.style {

}

#brxe-ghetas[data-status="success"] {

background-color: var(--success-light);

color: var(--success-dark);

}

span[class*=brxe-]:not(.brxe-section):not(.brxe-container):not(.brxe-block):not(.brxe-div) {

width: auto;

}

.status {

padding: 1em 1.5em;

color: var(--text-color, var(--neutral-dark));

background-color: var(--background-color, var(--neutral-ultra-light));

}

.brxe-button {

width: auto;

}

post-1207.m...697147761:1

frontend-li...697063483:1

?bricks_pre...7145276:108

post-1207.m...697147761:1

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common e

Target elements with a specific attribute

a[class] {

32:54

#brxe-ghetas1

.status

CONTENTSTYLE

[Learn more about CSS transitions](#)

Custom CSS

```
1 %root%[data-status] {
2   text-transform: capitalize;
3 }
4
5 %root%[data-status="success"] {
6   background-color: var(--success-light);
7   color: var(--success-dark);
8 }
9 %root%[data-status="warning"] {
10  background-color: var(--warning-light);
11  color: var(--warning-dark);
12 }
13 %root%[data-status="danger"] {
14  background-color: var(--danger-light);
15  color: var(--danger-dark);
16 }
```

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

CSS classes

Separated by space. Without class dot.

trigger a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaws in the class targeting method over the data attribute method.**

Using Classes:SuccessWarningDanger

Using a Data Attribute:SuccessWarningDanger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Target elements with a specific attribute

```
a[class] {
  border: 5px solid red;
}

a[title] {
  border: 5px solid red;
}
```

Link with a title

Link with no title

Structure

Container

Heading

Rich Text

Container

Button

JS Interactions

Container

Container

Button

Container

Basic Text

Button

Butt

Butt

Butt

33:48

Not Secure | bricks-clean.local/data-attributes/?bricks_preview=1697145276

FacebookHey!BusinessPersonalHostingResourcesStreamYardBasecampAutomatic.cssFramesHubspotHelpScoutGeniusInnerCircleACSSUserMavenJSSoftballAll Bookmarks

ElementsConsole>>⚙️⋮✕

▼flex == \$0

span#brxe-ghetas.brxe-button.status.bricks-button.bricks-b

StylesComputedLayoutEvent Listeners>>

Filter: :hov .cls +, ,

element.style {

#brxe-ghetas[data-status] {
text-transform: capitalize;

span[class*=brxe-]:not(.brxe-section):not(.brxe-container):not(.brxe-block):not(.brxe-div) {
width: auto;

.status {
padding: 1em 1.5em;
color: var(--text-color, var(--neutral-dark));
background-color: var(--background-color, var(--neutral-ultra-light));

.brxe-button {
width: auto;

.bricks-background-primary {

Bricks CleanCustomize0+NewEdit PageEdit with BricksRender with BricksAutomatic CSSHowdy, admin

Status

Using Classes:

SuccessWarningDanger

Using a Data Attribute:

SuccessWarningDanger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common ones.

Target elements with a specific attribute

a[class] {

33:51

Button

cus

#brxe-ghetas

.status

Custom CSS (css)

```
1 %root%[data-status="success" i] {
2   background-color: var(--success-light);
3   color: var(--success-dark);
4 }
5 %root%[data-status="warning" i] {
6   background-color: var(--warning-light);
7   color: var(--warning-dark);
8 }
9 %root%[data-status="danger" i] {
10  background-color: var(--danger-light);
11  color: var(--danger-dark);
12 }
```

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

trigger a change in a status badge using classes first, and then again using data attributes. **Notice the fatal flaws in the class targeting method over the data attribute method.**

Using Classes:

Success

Warning

Danger

Using a Data Attribute:

Success

Warning

Danger

Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

```
a[class] {
  border: 5px solid red;
}

a[title] {
  border: 5px solid red;
}
```

Link with a title

Link with no title

Structure

Container

Heading

Rich Text

Container

Button

JS Interactions

Container

Container

Button

Container

Basic Text

Button

Butt

Butt

Butt

The 'i' tells that the value is no longer case sensitive (in case we have put the data attribute in capital)



Attribute Selector Superpowers

There are TONS of ways to target elements by attribute. Here are a few common examples.

Target elements with a specific attribute

```
a[class] {  
  border: 5px solid red;  
}  
  
a[title] {  
  border: 5px solid red;  
}
```

Link with a title

Link with no title



Video player interface showing a tutorial on dynamic styling with data attributes and attribute selectors.

Browser Tab: Data Attributes (Builder)

Address Bar: Not Secure | bricks-clean.local/data-attributes/?bricks=run

Code Editor:

```
#brxe-gfnkiw
```

Structure Panel:

- Superpowers
 - Container
 - Container
 - Heading
 - Code
 - Text link
 - Text link
 - Container
 - Heading
 - Code
 - Text link
 - Text link
 - Text link

Code Editor (Highlighted):

```
border: 5px solid red;
}

a[title] {
  border: 5px solid red;
}

a[class="btn"] {
  border: 5px solid red;
}

a[href="/"] {
  border: 5px solid red;
}

[data-profile-name="Kevin Geary"] {
  border: 5px solid red;
}
```

Code Editor (Below Highlighted):

```
a[class*="btn--"] {
  border: 5px solid red;
}
```

Video Content:

Target elements with a specific attribute with a specific value

Link with a title

Link with no title

Link to home page

Plain Link

Link with .btn

Target elements with an attribute value that contains a specific string

38:33

Container

cus

#brxe-eohsxy

Custom CSS (css)

```
1 %root% a[href="/"] {  
2   border: 5px solid red;  
3 }  
4  
5 %root% a[class="btn"] {  
6   border: 5px solid red;  
7 }
```

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

Link to home page

Plain Link

Link with .btn

Target elements with an attribute value that contains a specific string

```
a[class*=".btn--"] {  
  border: 5px solid red;  
}
```

Link with class .btn

Link with class .btn--dark

Link with class .btn--light

Target elements with an attribute value that contains a specific word

```
[data-name*="Kevin"] {  
  border: 5px solid red;  
}
```

[data-name="Mr. Geary"]

[data-name="Mr. Kevin Geary"]

[data-name="KevinGeary"]

Target elements with an attribute value that starts with a specific string

Structure

- Superpowers
 - Container
 - Heading
 - Code
 - Text link
 - Text link
 - Container
 - Heading
 - Code
 - Text link
 - Text link
 - Text link

This is not working because the element has more classes than 'btn'

Container

cus

#brxe-wrynaa

• Custom CSS (css)

```
1 %root% a[class*="btn"] {  
2   border: 5px solid red;  
3 }
```

The star stands for contain

Link to home page

Plain Link

Link with .btn

Target elements with an attribute value that contains a specific string

```
a[class*="btn--"] {  
  border: 5px solid red;  
}
```

Link with class .btn

Link with class .btn--dark

Link with class .btn--light

Target elements with an attribute value that contains a specific word

```
[data-name*="Kevin"] {  
  border: 5px solid red;  
}
```

[data-name="Mr. Geary"]

[data-name="Mr. Kevin Geary"]

[data-name="KevinGeary"]

Target elements with an attribute value that starts with a specific string

Structure

- Text link
- Text link
- Container
 - Heading
 - Code
 - Text link
 - Text link
 - Text link
- Container
 - Heading
 - Code
 - Text link
 - Text link

41:12

Container

cus

#brxe-wrynaa

Custom CSS (css)

```
1 %root% [data-name~="Kevin"] {  
2   border: 5px solid red;  
3 }
```

Tilde stands for word

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

Link with class .btn

Link with class .btn--dark

Link with class .btn--light

Target elements with an attribute value that contains a specific word

```
[data-name~="Kevin"] {  
  border: 5px solid red;  
}
```

[data-name="Mr. Geary"]

[data-name="Mr. Kevin Geary"]

Text link

[data-name="KevinGeary"]

Target elements with an attribute value that starts with a specific string

```
a[href^="http://"] {  
  border: 5px solid red;  
}
```

Link with https://

Link with https://

Link with http://

Target elements with an attribute value that ends with a specific string

Structure

- Text link
- Text link
- Container
 - Heading
 - Code
 - Text link
 - Text link
 - Text link
- Container
 - Heading
 - Code
 - Text link
 - Text link
 - Text link



Not Secure | bricks-clean.local/data-attributes/?bricks=run

Facebook Hey! Business Personal Hosting Resources StreamYard Basecamp Automatic.css Frames Hubspot HelpScout Geni.us InnerCircle ACSS UserMaven JS Softball All Bookmarks

Container

cus

#brxe-esdbzl

• Custom CSS (css)

```
1 %root% [data-name*="kevin" i]:not([data-name~="Kevin"]){
2   border: 5px solid red;
3 }
```

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

Link with class .btn

Link with class .btn--dark

Link with class .btn--light

Target elements with an attribute value that contains a specific word

```
[data-name~="Kevin"] {
  border: 5px solid red;
}
```

[data-name="Mr. Geary"]

[data-name="Mr. Kevin Geary"]

[data-name="KevinGeary"]

Target elements with an attribute value that starts with a specific string

```
a[href^="http://"] {
  border: 5px solid red;
}
```

Link with https://

Link with https://

Link with http://

Target elements with an attribute value that ends with a specific string

Structure

- Text link
- Text link
- Text link
- Container
 - Heading
 - Code
 - Text link
 - Text link
 - Text link
- Container
 - Heading
 - Code
 - Text link
 - Text link

43:39

Container

cus

#brxe-esdbzl

Custom CSS (css)

1 %root% {
 color: firebrick;
}

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

[data-name="Mr. Kevin Geary"]

[data-name="KevinGeary"]

Target elements with an attribute value that starts with a specific string

```
a[href^="http://"] {  
  border: 5px solid red;  
}
```

Link with https://

Link with https://

Link with http://

Target elements with an attribute value that ends with a specific string

```
a[class$="dark"] {  
  border: 5px solid red;  
}
```

Link with class .btn

Link with class .btn--dark

Link with class .btn--light

Remove case-sensitivity from an attribute selector

Use "i" at the end of the attribute selector to remove case-sensitivity.

Structure

Text link

Text link

Text link

Container

Heading

Code

Text link

Text link

Text link

Container

Heading

Code

Text link

Text link

Text link

Text link

44:32

Not Secure | bricks-clean.local/data-attributes/?bricks=run

Facebook Hey! Business Personal Hosting Resources StreamYard Basecamp Automatic.css Frames Hubspot HelpScout Geni.us InnerCircle ACSS UserMaven JS Softball All Bookmarks

Container

cus

#brxe-esdbzl

• Custom CSS (css)

```
1 %root% a[href^="http://"] {
2   border: 5px solid red;
3 }
```

Use "%root%" to target the element wrapper. Add "%root%" via keyboard shortcut "r + TAB".

Structure

- Text link
- Text link
- Text link
- Container
 - Heading
 - Code
 - Text link
 - Text link
 - Text link
- Container
 - Heading
 - Code
 - Text link
 - Text link
 - Text link

border: 5px solid red;

Link with https://

Link with https://

Link with http://

Target elements with an attribute value that ends with a specific string

```
a[class$="dark"] {
  border: 5px solid red;
}
```

Link with class .btn

Link with class .btn--dark

Link with class .btn--light

Remove case-sensitivity from an attribute selector

Use "i" at the end of the attribute selector to remove case-sensitivity.

```
[data-profile-name="kevin geary" i] {
  style: here;
}
```

45:48