Donate

Stay safe, friends. Learn to code from home. Use our free 2,000 hour curriculum.

16 JANUARY 2018 / #STYLE

CSS Naming Conventions that Will Save You Hours of Debugging

Emmanuel Ohans

```
<div class="container">
  <div class="row">
    <div class="col-md-6 col-lg-8"> <!-- ________ BEGIN NAVIGATION
      <nav id="nav" role="navigation">
        <ul>
          <li><a href="index.html">Home</a></li>
          <li><a href="home-events.html">Home Events</a></li>
          <li><a href="multi-col-menu.html">Multiple Column Men
          <li class="has-children"> <a href="#" class="current"
            <ul>
              <li><a href="tall-button-header.html">Tall But
              <li><a href="image-logo.html">Image Logo</a></
              <li class="active"><a href="tall-logo.html">Ta
            </ul>
          </li>
          <li class="has-children"> <a href="#">Carousels</a>
            <ul>
              <li><a href="variable-width-slider.html">Variab
```

I have heard lots of developers say they hate CSS. In my experience, this comes as a result of not taking the time

Donate

Korean ??

알림: 한국인 독자분들을 위해 본 기사는 한국어로 번역되었으며, 한국어 버전은 **여기**에서 보실 수 있습니다

CSS isn't the prettiest 'language,' but it has successfully powered the styling of the web for over 20 years now. Not doing too badly, huh?

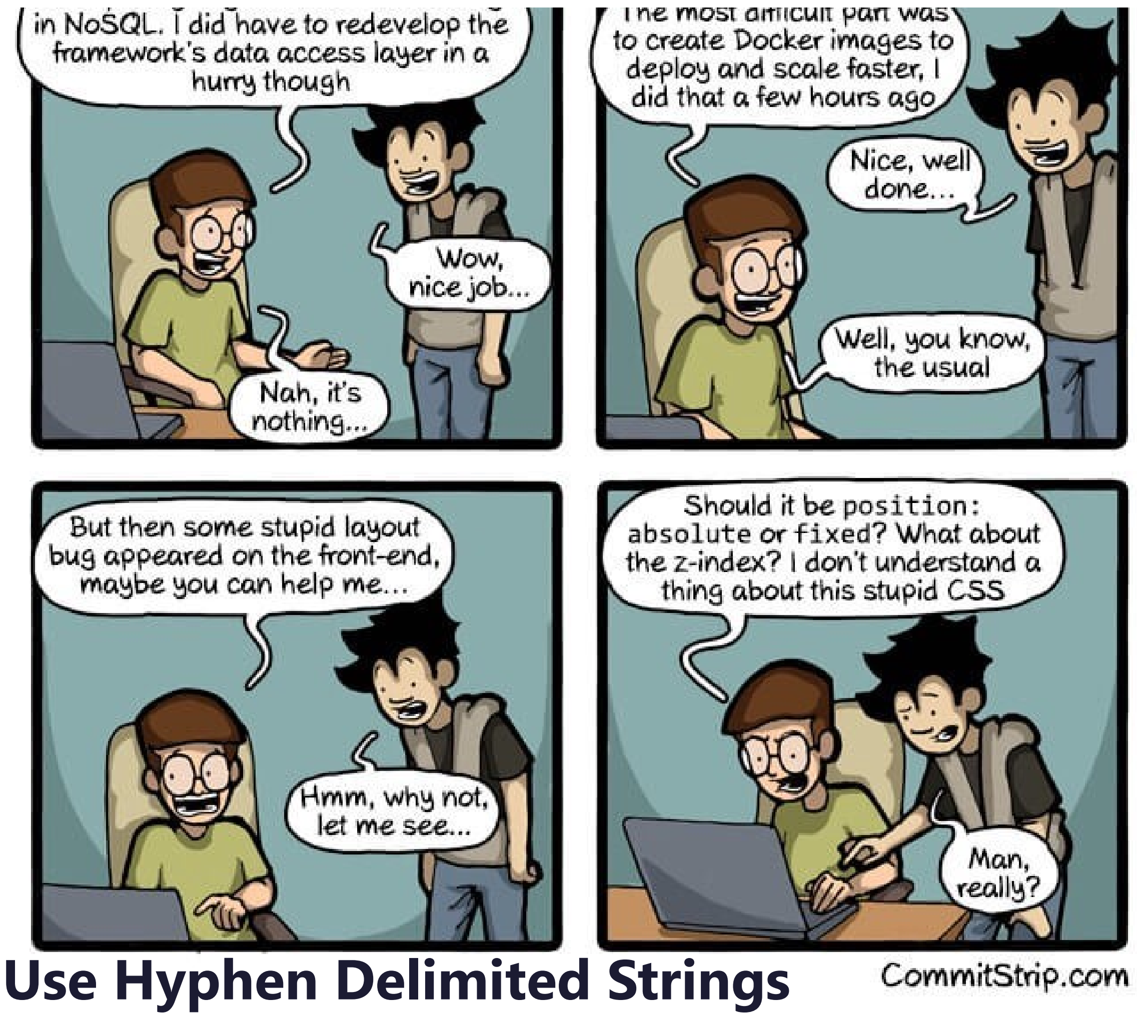
However, as you write more CSS, you quickly see one big downside.

It is darn difficult to maintain CSS.

Poorly written CSS will quickly turn into a nightmare.

Here are a few naming conventions that will save you a bit of stress and countless hours down the line.

Donate


in NoSQL. I did have to redevelop the framework's data access layer in a hurry though
Wow, nice job...
Nah, it's nothing...
The most difficult part was to create Docker images to deploy and scale faster, I did that a few hours ago
Nice, well done...
Well, you know, the usual
But then some stupid layout bug appeared on the front-end, maybe you can help me...
Hmm, why not, let me see...
Should it be position: absolute or fixed? What about the z-index? I don't understand a thing about this stupid CSS
Man, really?
CommitStrip.com

-you've been there before, haven't you?

Use Hyphen Delimited Strings

If you write a lot of JavaScript, then writing variables in camel case is common practice.

```
var redBox = document.getElementById('...')
```

Great, right?

The problem is that this form of naming isn't well-suited to CSS.

Do not do this:

Instead, do this:

```
.red-box {   border: 1px solid red;}
```

This is a pretty standard CSS naming convention. It is arguably more readable.

Also, it is consistent with the CSS property names.

```
// Correct

.some-class {   font-weight: 10em}

// Wrong

.some-class {   fontWeight: 10em}
```

The BEM Naming Convention

teams use hyphen delimiters, while others prefer to use the more structured naming convention called BEM.

Generally, there are 3 problems that CSS naming conventions try to solve:

1. To know what a selector does, just by looking at its name
2. To have an idea of where a selector can be used, just by looking at it
3. To know the relationships between class names, just by looking at them

Have you ever seen class names written like this:

```
.nav--secondary {  ...}
```

```
.nav__header {  ...}
```

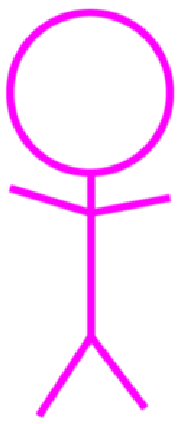
That is the BEM naming convention.

Explaining BEM to a 5 year Old

BEM attempts to divide the overall user interface into small reusable components.

Consider the image below:

Donate



It is an award winning image of a stick-man :)

No, it’s not award winning :(

The stick-man represents a component, such as a block of design.

You may have already guessed that the B in BEM stands for ‘Block’.

In the real world, this ‘block’ could represent a site navigation, header, footer, or any other block of design.

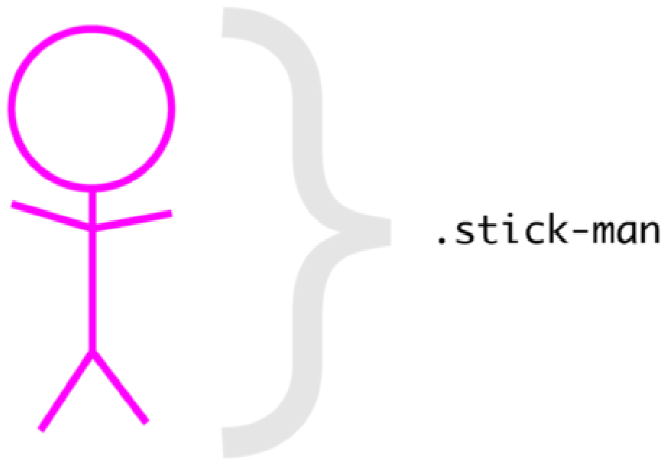
Following the practice explained above, an ideal class name for this component would be `stick-man` .

Donate

The component should be styled like so:

```
.stick-man {   }
```

We have used delimited strings here. Good!



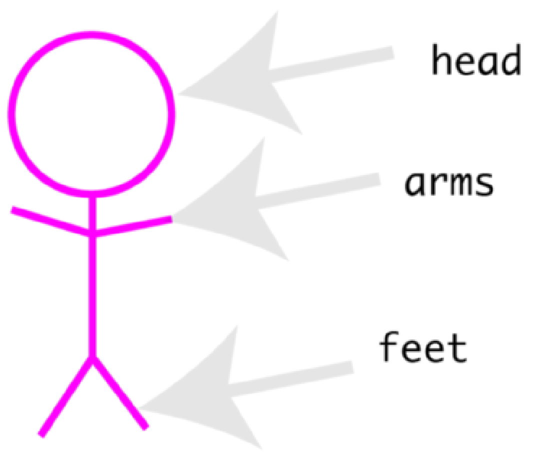
E for Elements

The E in 'BEM' stands for Elements.

Donate

Overall blocks of design rarely live in isolation.

For instance, the stick-man has a `head`, two gorgeous `arms`, and `feet`.



The `head`, `feet`, and `arms` are all elements within the component. They may be seen as child components, i.e. children of the overall parent component.

Using the BEM naming convention, element class names are derived

Donate

For example:

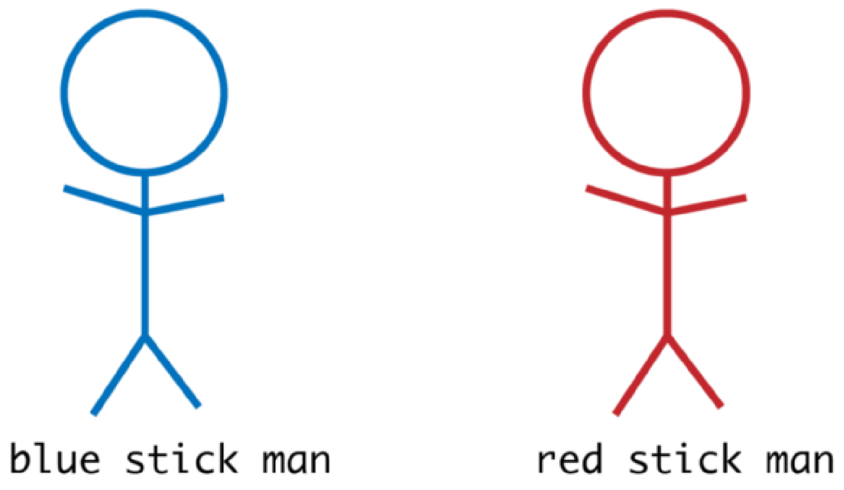
```
.stick-man__head {

}

.stick-man__arms {

}

.stick-man__feet {

}
```

M for Modifiers

The M in 'BEM' stands for Modifiers.

What if the stick-man was modified and we could have a `blue` or a `re`

Donate



In the real world, this could be a `red` button or `blue` button. These are modifications of the component in question.

Using BEM, modifier class names are derived by adding two **hyphens** followed by the element name.

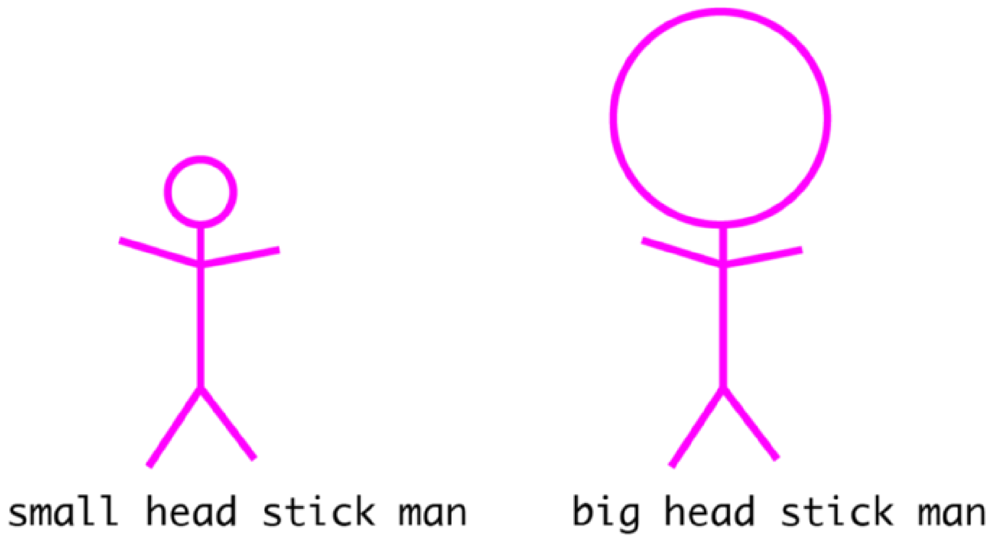
For example:

Donate

```
}

.stick-man--red {

}
```

The last example showed the parent component being modified. This is not always the case.

What if we had stick-men of different `head` sizes?



Donate

```
}
```

This time the element has been modified. Remember, the element is a child component within the overall containing block.

The `.stick-man` represents the `Block` , `.stick-man__head` the element.

As seen in the example above, double hyphens may also be used like so:

```
.stick-man__head--small {

}

.stick-man__head--big {

}
```

Donate

Again, note the use of the double **hyphens** in the example above. This is used to denote a modifier.

Now you've got it.

That's basically how the BEM naming convention works.

Personally, I tend to use only hyphen delimeter class names for simple projects, and BEM for more involved user interfaces.

You can read more about BEM.

BEM - Block Element Modifier
*BEM - Block Element Modifier is a methodology, that helps you to achieve reusable components and code sharing in the…*getbem.com

Why Use Naming Conventions?

> There are only two hard problems in Computer Science: cache invalidation and naming things — *Phil Karlton*

Naming things is hard. We're trying to make things easier, and save ourselves time in the future with more maintainable code.

Naming things correctly in CSS will make your code easier to read and maintain.

If you choose to use the BEM naming convention, it will become easier to see the relationship between your design components/blocks just by looking at the markup.

Feeling confident?

Donate

CSS Names with JavaScript Hooks

Today is John's first day at work.

He is handed over an `HTML` code that looks like this:

```
<div class="siteNavigation">



</div>
```

John has read this article and realizes this may not be the best way to name things in `CSS`. So he goes ahead and refactors the codebase like so:

```
<div class="site-navigation">



</div>
```

Looks good, huh?

Unknown to John, he had broken the codebase ???

How?

Donate

previous class name, `siteNavigation`:

```
//the Javasript code



const nav = document.querySelector('.siteNavigation')
```

So, with the change in the class name, the `nav` variable became `null`.

How sad.

To prevent cases like this, developers have come up with different strategies.

1. Use js- class names

One way to mitigate such bugs is to use a **`js-*`** class name to denote a relationship with the DOM element in question.

For example:

```
<div class="site-navigation js-site-navigation">



</div>
```

Donate

```
//the Javasript code
```

```
const nav = document.querySelector('.js-site-navigation')
```

As a convention, anyone who sees the `js-site-navigation` class name would understand that there is a relationship with that DOM element in the JavaScript code.

2. Use the Rel attribute

I don't use this technique myself, but I have seen people do.

Do you recognize this?

```
<link rel="stylesheet" type="text/css" href="main.css">
```

Basically, the **rel attribute** defines the relationship that the linked resource has to the document from which it's referenced.

In the previous example with John, proponents of this technique would do this:

```
<div class="site-navigation" rel="js-site-navigation">
```

Donate

```
</div>
```

And in the JavaScript:

```
const nav = document.querySelector("[rel='js-site-navigation']")
```

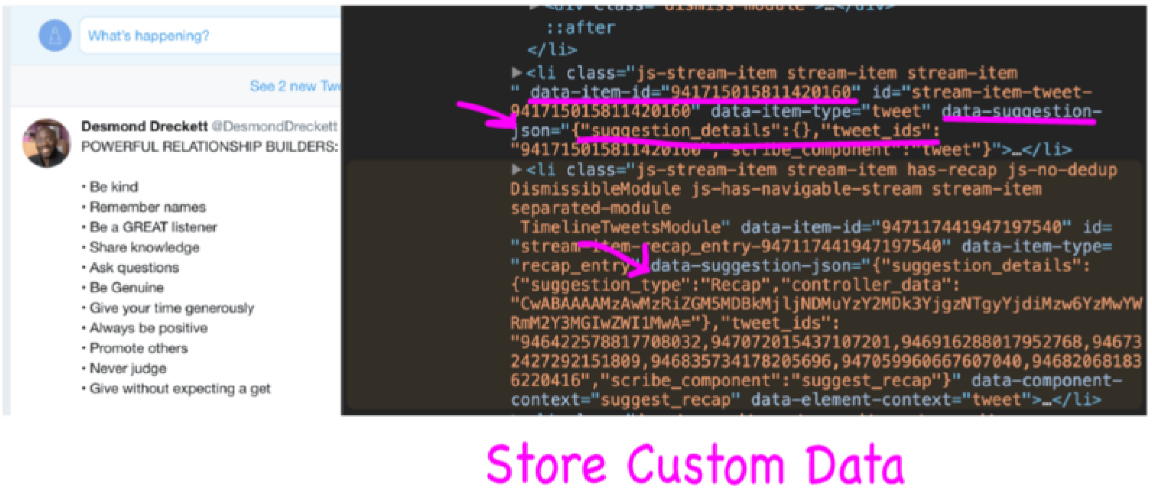
I have my doubts about this technique, but you're likely to come accross it in some codebases. The claim here is, *"well, there's a relationship with Javascript, so I use the rel attribute to denote that"*.

The web is a big place with lots of different "methods" for solving the same problem.

3. Don't use data attributes

Some developers use data attributes as JavaScript hooks. This isn't right. By definition, data attributes are used **to store custom data**.

Donate



What's happening?

See 2 new Tw

Desmond Dreckett @DesmondDreckett
POWERFUL RELATIONSHIP BUILDERS:

- Be kind
- Remember names
- Be a GREAT listener
- Share knowledge
- Ask questions
- Be Genuine
- Give your time generously
- Always be positive
- Promote others
- Never judge
- Give without expecting a get

```
  ::after
</li>
▶<li class="js-stream-item stream-item stream-item
" data-item-id="941715015811420160" id="stream-item-tweet-
941715015811420160" data-item-type="tweet" data-suggestion-
json="{"suggestion_details":{},"tweet_ids":
"941715015811420160","scribe_component":"tweet"}">…</li>
▶<li class="js-stream-item stream-item has-recap js-no-dedup
DismissibleModule js-has-navigable-stream stream-item
separated-module
 TimelineTweetsModule" data-item-id="947117441947197540" id=
"stream-item-recap_entry-947117441947197540" data-item-type=
"recap_entry" data-suggestion-json="{"suggestion_details":
{"suggestion_type":"Recap","controller_data":
"CwABAAAAMzAwMzRiZGM5MDBkMjljNDMuYzY2MDk3YjgzNTgyYjdiMzw6YzMwYW
RmM2Y3MGIwZWI1MwA="},"tweet_ids":
"946422578817708032,947072015437107201,946916288017952768,94673
2427292151809,946835734178205696,947059960667607040,94682068183
6220416","scribe_component":"suggest_recap"}" data-component-
context="suggest_recap" data-element-context="tweet">…</li>
```

Good use of data attributes. As seen on Twitter

Edit #1: As mentioned by some amazing people in the comment section, if people use the 'rel' attribute, then it's perhaps okay to use data attributes in certain cases. It's your call afterall.

Bonus Tip: Write More CSS Comments

This has nothing to do with naming conventions, but it will save you some time too.

While a lot of web developers try to NOT write JavaScript comments or stick to a few, I think you should write more CSS comments.

Since CSS isn't the most elegant "language," well-structured comments can save time when you're trying to understand your code.

It doesn't hurt.

Take a look at how well commented the Bootstrap source code is.

Donate

color. But, if you're using a CSS trick that is less obvious, feel free to write a comment.

Ready to become Pro?

I have created a free CSS guide to get your CSS skills blazing, immediately. Get the free ebook.

Seven CSS Secrets you didn't know about

Donate

Emmanuel Ohans

Read more posts by this author.

If this article was helpful, tweet it.

Continue reading about

Style

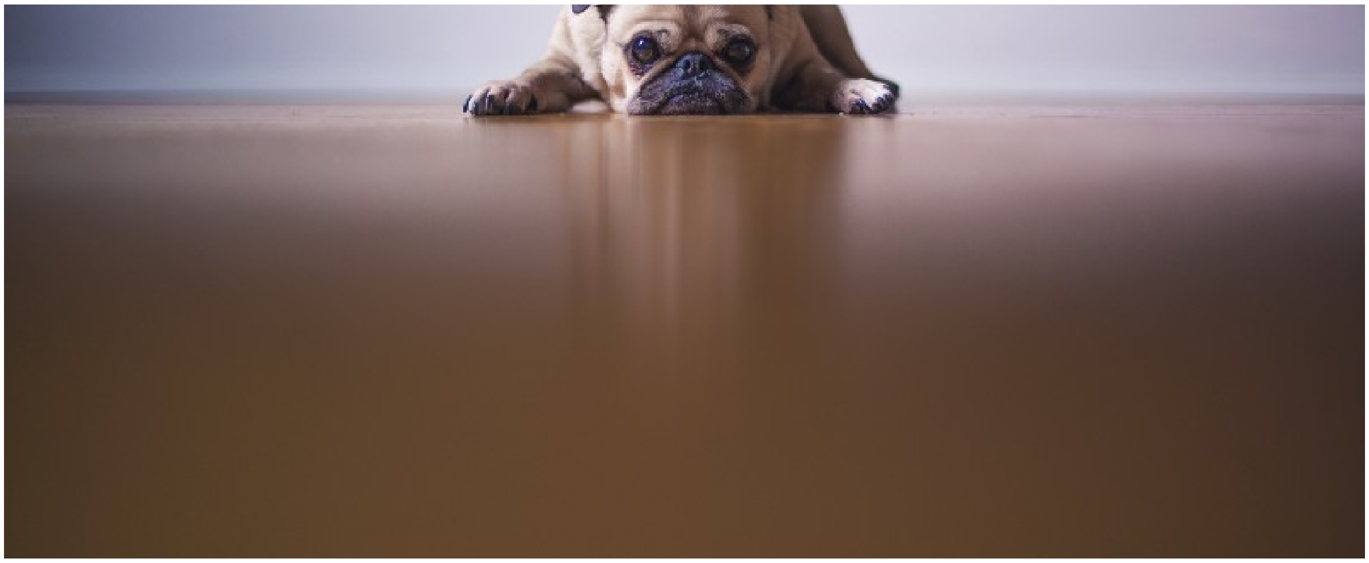
How to Style Links in CSS

CSS Comment Example – How To Comment Out CSS

Coding Commandments

See all 4 posts →

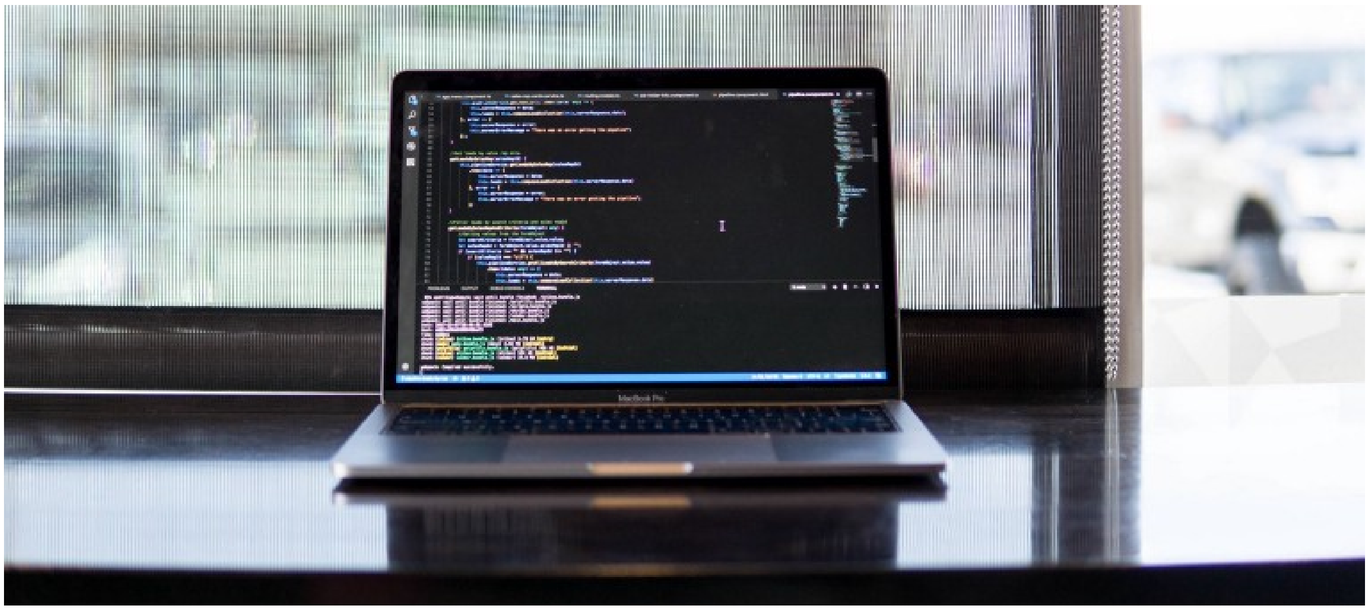
Donate



#SERVERLESS

I’m afraid you’re thinking about AWS Lambda cold starts all wrong

3 YEARS AGO



#WEBPACK

How to use Webpack with React: an in-depth tutorial

ESAU SILVA 3 YEARS AGO

Donate

Federal Tax Identification Number: 82-0779546)
Our mission: to help people learn to code for free. We accomplish this by creating thousands of videos, articles, and interactive coding lessons - all freely available to the public. We also have thousands of freeCodeCamp study groups around the world.

Donations to freeCodeCamp go toward our education initiatives, and help pay for servers, services, and staff.

You can make a tax-deductible donation here.

Trending Guides

Git Clone	UX
Agile Methods	Design Thinking
Python Main	Prime Numbers List
Callback	Product Design
Debounce	Digital Design
URL Encode	Coding Games
Blink HTML	SVM
Python Tuple	JavaScript forEach
JavaScript Push	Google BERT
Java List	Create Table SQL
Responsive Web Design	What Is TLS?
What Is an SVG File?	What Is a LAN?
PDF Password Remover	What is npm?
What Is a PDF?	RSync Examples
What Is Python?	Random Forest

Our Nonprofit

About Alumni Network Open Source Shop Support Sponsors Academic Honesty

Donate