



DrupalCon
SEATTLE 2019
APRIL 8-12

CSS-in-JS and Drupal sitting in a tree...

John Albin Wilkins
@JohnAlbin

CSS-in-JS and Drupal sitting in a tree...

TALKING POINTS

- 1 How we picked a CSS-in-JS project
- 2 Why we picked CSS Modules
- 3 Cool features of CSS Modules
- 4 A quick configuration



John Albin Wilkins

Senior Front-end Developer



2017 Amazee Labs

2004 Drupal

1997 CSS

1993 Web





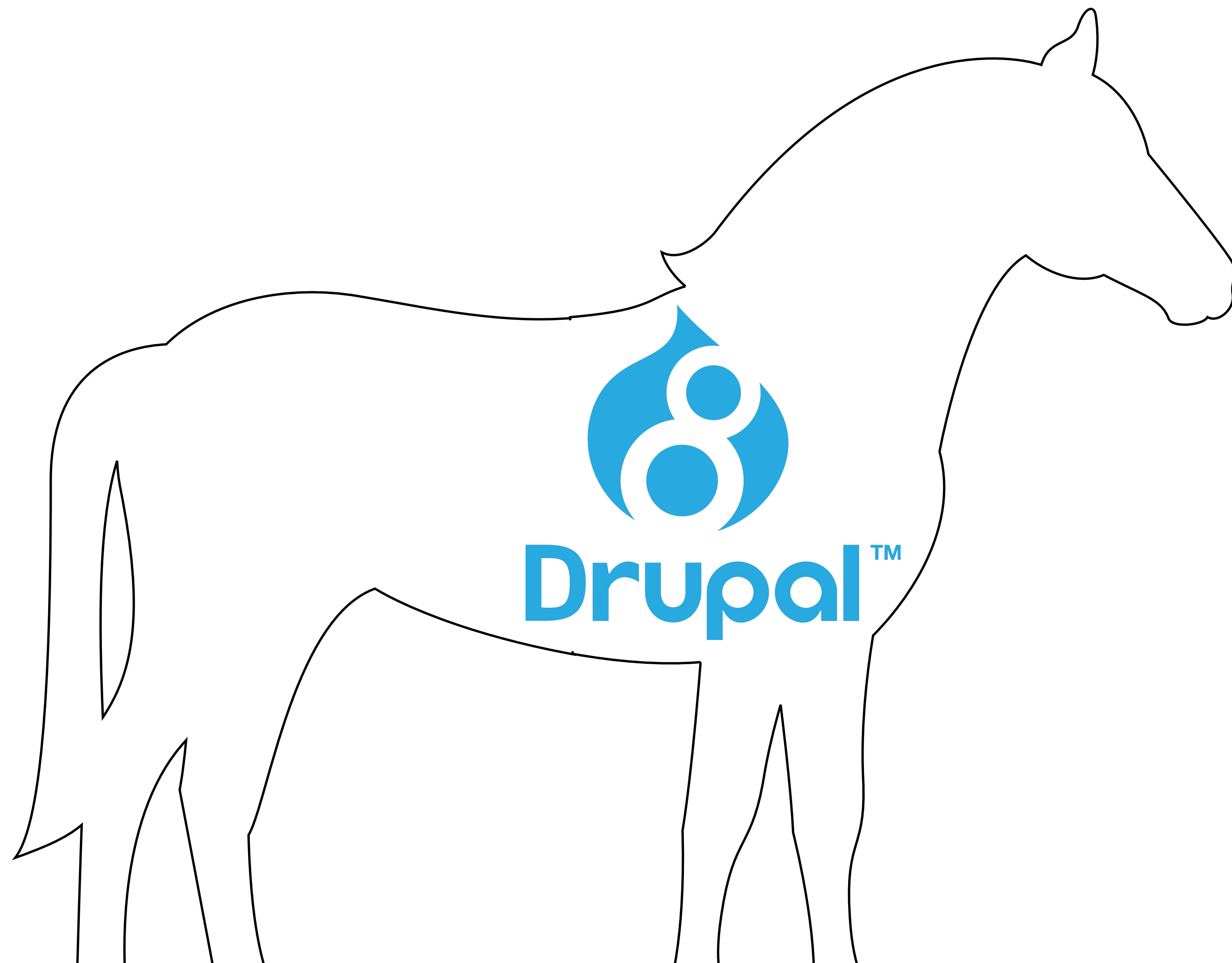
I ❤️

Sass

@JohnAlbin, 2011

TRADITIONAL WEBSITE

ALL-IN-ONE



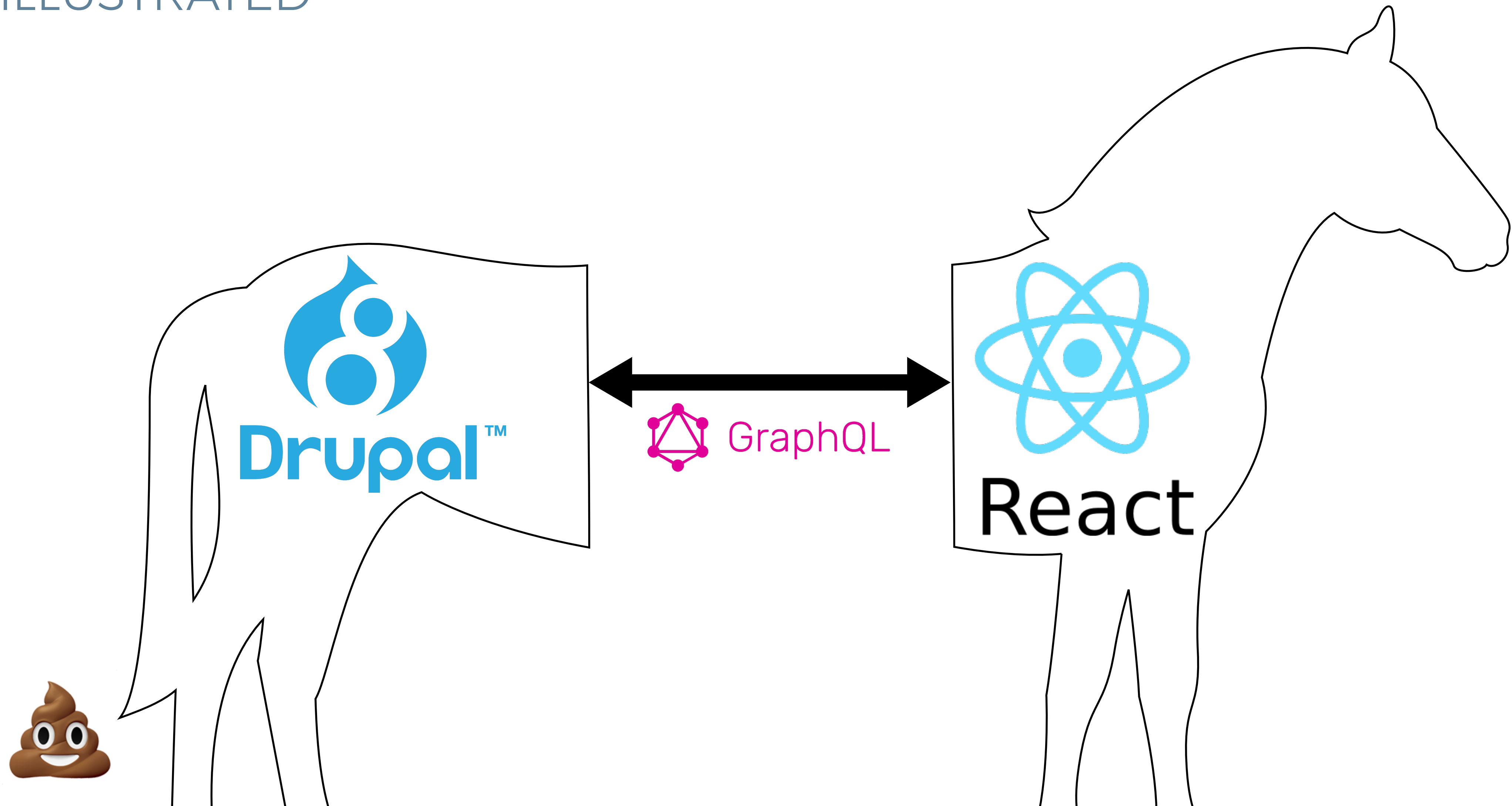
TRADITIONAL WEBSITE

Sass-ified



DECOUPLED WEBSITE

ILLUSTRATED



LET'S EXPLORE CSS-IN-JS



68+ CSS-IN-JS PROJECTS:

CHOOSE ONE

superstyle
styling
reactcss
style-it
css-loader
babel-plugin-css-in-js
styled-jsx
react-free-style
cssx
scope-styles
react-css-modules
react-vstyle
hyperstyles
hiccup-css
react-fela
i-css
Aphrodite
react-jss
electron-css
es-css-modules
react-static-styles
babel-plugin-pre-style
react-inline
css-light
react-styleable
glamorous
cssobj
react-style
css-ns
react-native-web
uranium
styled-components
react-look
react-styled
cxs
react-css-builder
stylable
react-cssom
glamor
react-inline-style
typestyle
react-stylematic
emotion
styletron-react
react-styl
react-css-components
css-constructor
react-theme
stile
stilr
react-inline-style
restyles
react-inline-css
freestyler
smart-css

3 API APPROACHES:

CHOOSE ONE

- CSS in a **object literal** (JavaScript Object)

- CSS in a **template literal** (JavaScript String)

- CSS in a **file** (CSS File)

CSS IN A OBJECT LITERAL

```
// MyComponent.js
import { apiFunc } from 'some-project';

const styles = {
  base: {
    color: '#fff',
    ':hover': {
      backgroundColor: '#0074d9',
    },
  },
  warning: {
    'background-color': someValue(),
  },
};
```

```
// MyComponent.js (cont.)
export const MyComponent = () => (
  <div className={
    apiFunc(styles.base, styles.warning)
  }>
    Some content
  </div>
);
```

CSS IN A TEMPLATE LITERAL

```
// MyComponent.js
import { apiFunc } from 'some-project';

const styles = apiFunc`
  .base {
    color: #fff;

    &:hover {
      background-color: #0074d9;
    }
  }

  .warning {
    background-color: ${someValue()},
  }
`;
```

```
// MyComponent.js (cont.)
export const MyComponent = () => (
  <div className={
    `${styles.base} ${styles.warning}`
  }>
    Some content
  </div>
);
```

CSS IN A FILE

```
/* styles.css */
.base {
  color: #fff;

  &:hover {
    background-color: #0074d9;
  }
}

.warning {
  background-color: var(--imported-value),
}
```

```
// MyComponent.js
import styles from './styles.css';
import classNames from 'classnames';

export const MyComponent = () => (
  <div className={
    classNames(styles.base, styles.warning)
  }>
    Some content
  </div>
);
```

FEATURE COMPARISON

EVALUATION CRITERIA

	locally scoped class	cross- component composition	dead code elimination	nested rulesets	shared JS / CSS variables	multi- platform
CSS in object literal	✓	✓	✓	✓	✓	JS projects only
CSS in template literal	✓	✓	✓	✓	✓	JS projects only
CSS in file	✓	✓	✓	✓	✓	😎

CSS MODULES



LOCALLY-SCOPED CLASS NAMES

OLD RULES FOR BEM NAMING

```
.callToAction { }
```

```
.callToAction--title { }
```

```
.callToAction--link { }
```

callToAction.css

```
.fancyList { }
```

```
.fancyList--title { }
```

```
.fancyList--link { }
```

fancyList.css

LOCALLY-SCOPED CLASS NAMES

NO MORE BEM RULES

```
.Wrapper {}
```

```
.Title {}
```

```
.Link {}
```

callToAction.css

```
.Wrapper {}
```

```
.Title {}
```

```
.Link {}
```

fancyList.css

LOCALLY-SCOPED CLASS NAMES

i.e. ENSURE UNIQUE GLOBAL CLASS NAMES

```
.fdc03d { }
```

```
.79ec33 { }
```

```
.6e4c6d { }
```

callToAction.css

```
.62e171 { }
```

```
.24c42e { }
```

```
.df8be2 { }
```

fancyList.css

LOCALLY-SCOPED CLASS NAMES

MAPPING KNOWN CLASS NAMES TO AUTO-GENERATED ONES

```
// callToAction.js
import styles from './styles.css';
import classNames from 'classnames';

export const CallToAction = () => (
  <article className={styles.Wrapper}>
    <h3 className={styles.Title}>
      Look at me.
    </h3>
    <p>
      <a link="#" className={styles.Link}>
        Now do this.
      </a>
    </p>
    Some content
  </article>
);
```

```
// JS file auto-generated by Webpack
const styles = {
  Wrapper: 'fdc03d',
  Title: '79ec33',
  Link: '6e4c6d',
};

export default styles;
```

CROSS-COMPONENT COMPOSITION

SHARING CSS ACROSS COMPONENTS

```
.TimeStamp {  
  /* Add this class from another file. */  
  composes: TimeStamp from '../BlogPost/styles.css';  
  
  /* properties here */  
}  
  
.List {  
  // Assume this class exists in the global space.  
  composes: ListReset from global;  
  
  /* properties here */  
}  
  
:global(.Loader) {  
  /* properties here */  
}
```

```
// JS file auto-generated by Webpack  
const styles = {  
  // Includes "79ec33" class from BlogPost.  
  TimeStamp: 'fdc03d 79ec33',  
  
  // Includes "ListReset" class.  
  List: '6e4c6d ListReset',  
  
  // No class name transforms.  
  Loader: 'Loader'  
};  
  
export default styles;
```

DEAD-CODE ELIMINATION

AUTOMATIC REMOVAL OF UNUSED CSS

- When you add a component to a page, it adds the HTML, CSS, JS, etc. to the page.
- When you DO NOT add a component to a page, the HTML, CSS, JS, etc. is NOT added to the page.
- The **eslint-plugin-css-modules** plugin will find unused class names in your component.

NESTED RULESETS

NO MORE DUPLICATE SELECTORS

```
/* Old-school CSS */
```

```
.Wrapper { /* properties here */ }
```

```
@media (min-width: 40em) {
```

```
  .Wrapper { /* properties here */ }
```

```
}
```

```
@media (min-width: 60em) {
```

```
  .Wrapper { /* properties here */ }
```

```
}
```

```
.Wrapper:hover, .Wrapper:focus { /* properties here */ }
```

NESTED RULESETS

NO MORE DUPLICATE SELECTORS

```
/* CSS using css-nesting spec */
.Wrapper {
  /* properties here */

  @media (min-width: 40em) { /* properties here */ }

  @media (min-width: 60em) { /* properties here */ }

  @nest &:hover, &:focus { /* properties here */ }
}
```

<https://drafts.csswg.org/css-nesting-1>

WHY @NEST IS NEEDED

BROWSER PERFORMANCE AND LOOK-AHEAD PARSING

```
/* What if we tried Sass-style nesting? */
```

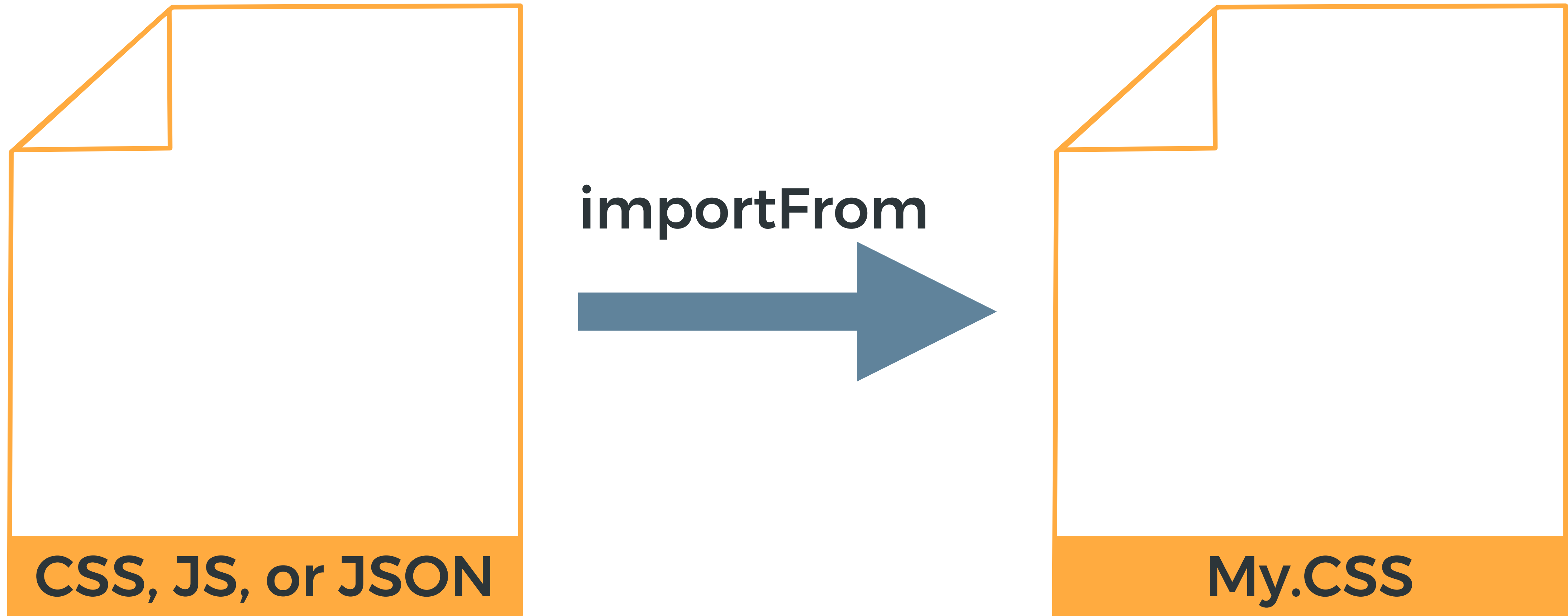
```
.Wrapper {  
  html.js & { /* Won't work! */  
    /* properties here */  
  }  
}
```

```
/* The css-nesting spec does allow  
   simple Sass-style nesting */
```

```
.Wrapper {  
  & span { /* Works! */  
    /* properties here */  
  }  
}
```

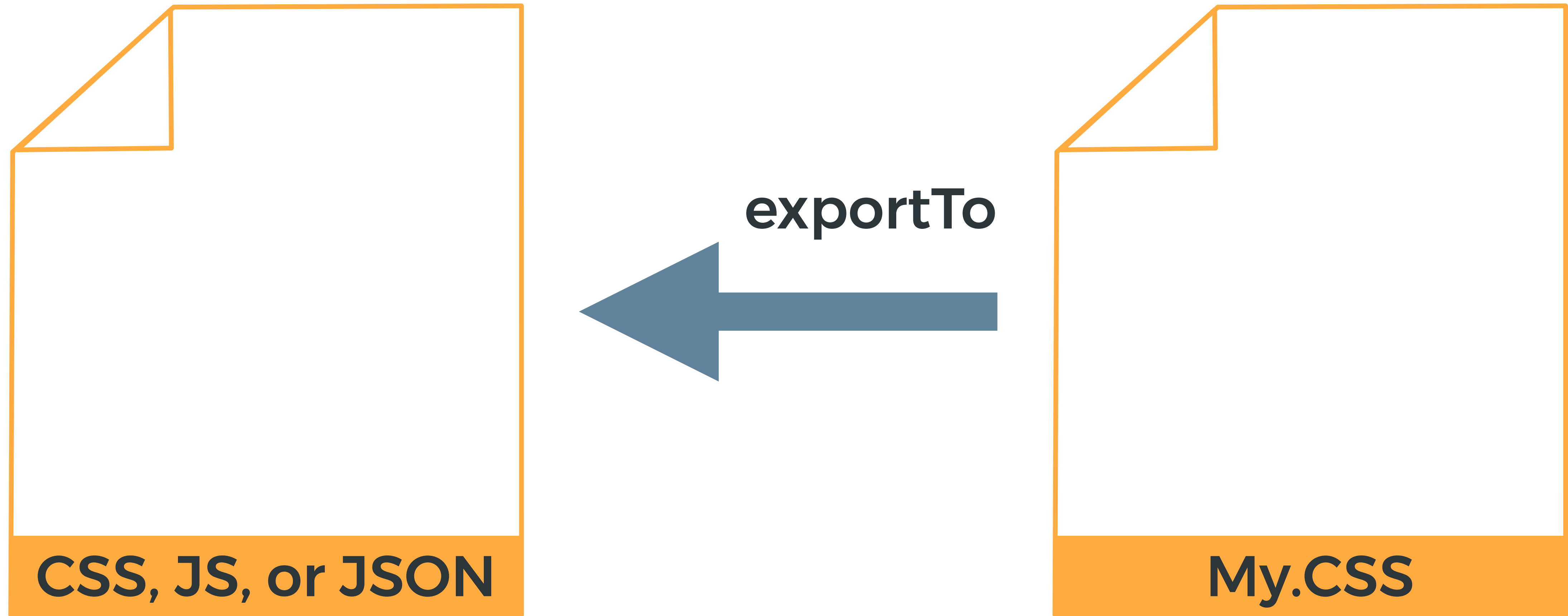

SHARING CSS / JS VARIABLES

WRITE VALUES ONCE, USE THEM EVERYWHERE



SHARING CSS / JS VARIABLES

WRITE VALUES ONCE, USE THEM EVERYWHERE



SHARING CSS / JS VARIABLES

WRITE VALUES ONCE, USE THEM EVERYWHERE

```
/* These values can be exported to CSS, JS, and JSON files. */
```

```
:root {  
  --primary-color: red;  
}
```

```
@custom-media --tablet (min-width: 30em);
```


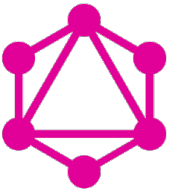
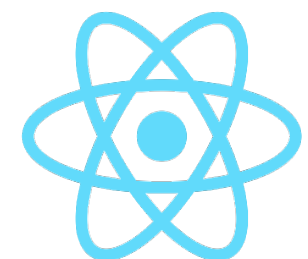
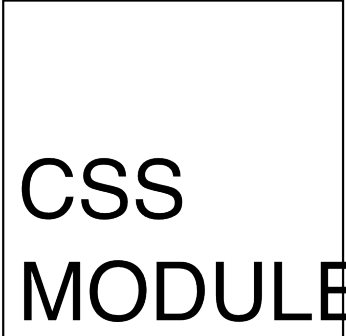
```
@custom-selector :--headings h1, h2, h3, h4, h5, h6;
```

MULTI-PLATFORM SUPPORT

EVERY PLATFORM WORKS WITH CSS FILES

ANY CMS ^{OR} **ANY JS FRAMEWORK** +  CSS MODULES

 +  +  +  CSS MODULES

 +  +  +  CSS MODULES

 +  +  CSS MODULES

CSS Modules

= webpack loader plugin

+ PostCSS



POSTCSS.CONFIG.JS

AMAZEE'S CONFIGURATION (WORK IN PROGRESS)

```
// postcss.config.js
const path = require('path');

module.exports = {
  plugins: {
    'postcss-preset-env': {
      stage: 1,
      browsers: '> 0.5%, last 2 versions, Firefox ESR, not dead',
      exportTo: path.resolve('cssValues.json');
    },
  }
};
```

EXPERIMENTAL CSS, TODAY


postcss-preset-env

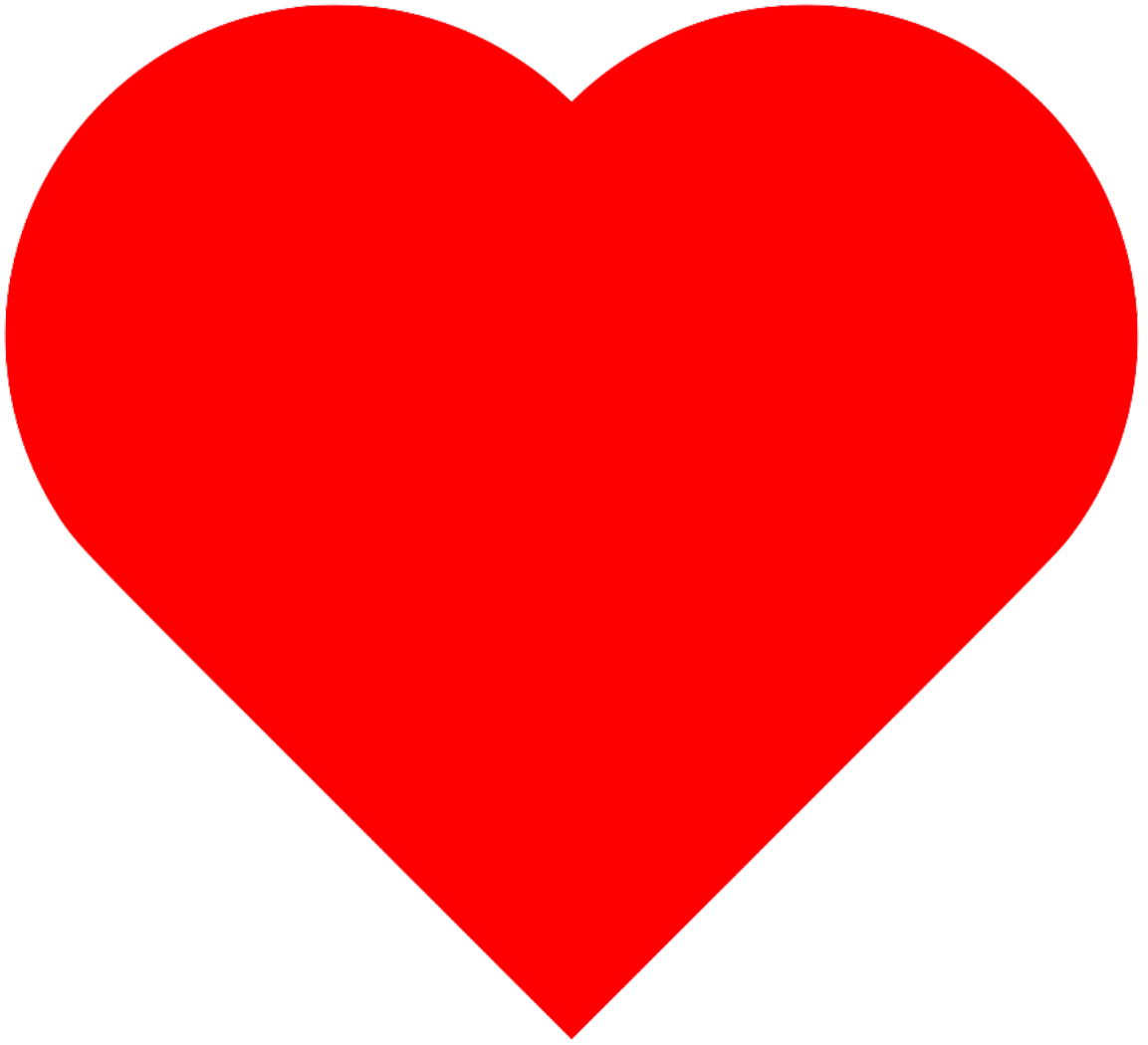


What's next for CSS?

cssdb is a comprehensive list of CSS features and their positions in the process of becoming implemented web standards.

<https://cssdb.org>



I  CSS

@JohnAlbin, 2019

**THANK
YOU**



QUESTIONS AND MAYBE ANSWERS





DrupalCon
SEATTLE 2019
APRIL 8-12

Join us for contribution opportunities

Friday, April 12, 2019

Mentored Contributions

9:00-18:00
Room: 602

First Time Contributor Workshop

9:00-12:00
Room: 606

General Contributions

9:00-18:00
Room: 6A

#DrupalContributions



DrupalCon
SEATTLE 2019
APRIL 8-12

What did you think?

Locate this session at the DrupalCon Seattle website:

<http://seattle2019.drupal.org/schedule>

Take the Survey!

<https://www.surveymonkey.com/r/DrupalConSeattle>

Thank you!