Website: https://www.vuedc.io

Twitter: @vuejsdc

Discord:
https://discordapp.com/invite/6MZWP8z

wifi: BNAGuest
password: BePc5309

We’re always looking for speakers!

If you don’t have a topic, that’s okay! We can work with you to come up with one or collaborate together on a talk.

Reach out to us at vuejsdc@gmail.com!

Announcements

July 24th - Vue Native Workshop @ Optoro

August 14th - Topic TBD @ Asymmetrik

Call for Proposals

- Vue London (Deadline: July 26, 2019)
- Vue Toronto (Deadline: June 30, 2019)

Many thanks to
our **gold** sponsor!

Bloomberg
BNA

Using CSS Modules in Vue

Tracey Holinka
Web Application Architect
Bloomberg BNA

What Is CSS?

There is no control flow.

Declarative Domain Specific Language (DSL) for presentation layer of an app or website.

CSS

In the beginning...

Before mobile and media queries

Inline CSS

```
<h1 style="color: #ff0000";>Heading</h1>
```

Internal and External CSS (<style> or .css files)

```
<style>
  h1 { color: #ff0000; }
</style>
```

CSS Processors

Can we have a variable?

Why can't CSS be like JavaScript?

SASS/LESS (preprocessor)
variables, nesting, mixins, etc.

PostCSS (postprocessor)
autoprefixer, future CSS like
variables, nesting, etc.

CSS Methodology

Bringing ~~Scalability~~ Sanity to CSS

OOCSS (Object Oriented CSS)

SMACSS (Scalable and Modular Architecture for CSS)

BEM (Block Element Modifier)

and many more...

Problem Solved!

Maybe Not

Class names must be unique (a nightmare for large projects)

Class names can be long ([block]__[element]--[modifier])

Over qualify selectors to limit scope (.article .meta .author .name) are not performant

The markup isn’t alway readable

Use of `!important`

CSS Modules

Encapsulation for CSS

Uses JavaScript to create dynamic class names that are bound directly to an element and adds a hash to ensure a globally unique name.

Guaranteed unique selectors!

Works Out of the Box with Vue

Article.vue

```
<template>
  ...
    <h1 :class="$style.headline">
      {{ article.title }}
    </h1>
  ...
</template>
```

```
<style module>
...
.headline {
  margin: 0 0 20px;
  font-size: 24px;
  line-height: 29px;
  letter-spacing: -0.32px;
  font-weight: bold;
}
...
</style>
```

Gives You

```
<h1 class="Article_headline_2f7zE">
  Headline for Faker News Article 1
</h1>
```

Component_Classname_Hash

Customize CSS Modules Classes

Vue.config.js

```
module.exports = {
 css: {
   loaderOptions: {
     css: {
       use: ['style-loader', 'postcss-loader'],
       localIdentName: '[name]_[local]_[hash:base64:5]' // default setting
     }
   }
 }
}
```

Downside?

Every element needs a class to be styled.

$style.className isn’t pretty.

End-to-end testing is less straightforward.

Upside?

No more style conflicts

No wasting time creating unique names

Can use SASS and PostCSS

Encourages proper component architecture

Slots & CSS with the component

Some Best Practices

No deeping targeting of classes--a components CSS belongs in the component. Keep classes flat.

Use Composes over SASS extends.

Limit Use of Globals.

Keep non-CSS Modules separate.

Have a well defined and documented styling architecture.

Flat Classes

No deeping targeting of classes--a components CSS belongs in the component. Keep classes flat.

You can add a class to a child component:

```
<template>
...
  <ChildComponent :class="$style.className"/>
...
</template>
```

Keep Classes Flat

```
<template>
 <section :class="$style.author">
   <AuthorBio v-bind="{id:author.id, name:
author.name, avatarUrl: author.avatarUrl, bio:
author.bio}"/>
   <ArticleCard
     v-for="article in author.articles"
     :key="article.id"
     :class="$style.card"
     v-bind="{ article, showAuthor: false,
showImageLead }"
   />
 </section>
</template>
```

```
<style module>
.author {
  margin: 0 16px;
}
.card {
  margin: 16px 0;
}
.author h1 {
  background: orange;
}
</style>
```

H1 is not locally scoped

Use Composes

global.css

```
.fontBase {
  font-size: 16px;
  line-height: 22px;
  letter-spacing: 0;
}
```

ArticleCard.vue

```
<style module>
.summary {
  composes: fontBase from "../assets/globals.css";
  margin: 0 0 16px;
}
</style>
```

Limit Use of Globals

```
* {
 box-sizing: border-box;
}
html {
 text-rendering: optimizeLegibility;
}
body {
 margin: 0;
 background: var(--color-background);
 font-family: 'Open Sans', sans-serif;
 color: var(--color-font);
}
a {
 color: var(--color-link);
 text-decoration: none;
}
```

Keep non-CSS Modules Separate

Article.vue

```
<template>
...
   <ArticleBody
     v-if="!!article.body"
     v-bind="{body: article.body}"/>
...
</template>
```

ArticleBody.vue

```
<template>
   <section
   class="article-body"
   v-html="body"/>
</template>
```

ArticleBody.vue

```
<style lang="postcss">
/* Styles markup delivered by GraphQL */
.article-body {
 & h2 {
   margin: 24px 0 16px;
   font-size: 24px;
   line-height: 29px;
   letter-spacing: -0.32px;
   font-weight: bold;
 }
 ...
 </style>
```

CSS Architecture

Document it!

Design patterns (layout, containers, base elements)

Naming conventions

Applicable programing concepts (open/closed principle and composition over inheritance)

So We’re Done

Styled Components

The future?

CSS in JavaScript using tagged template literals to style a component.

Questions?

If you need support on a day to day, we got you.

Make sure to:

- Check out our Discord channel (https://discordapp.com/invite/6MZWP8z)
- Bookmark our site (https://www.vuedc.io)
- Find talk resources on GitHub (https://github.com/VueDC)