

Vue Styleguide

Styleguide

Motivation

There are some style principles to respect in Vue 3, let us discuss them.

```
var scrollHeight = element.clientHeight + 0.02 * window.innerWidth;  
window.scroll(0, scrollHeight);  
}
```

Styleguide

The official style guide, and therefore also this one here is divided in four categories:

- Priority A: Essential
- Priority B: Strongly Recommended
- Priority C: Recommended
- Priority D: Use with Caution

While A and B are very important to us, priority C and D will only be shallow tackled in this guide.

Styleguide

Sad story:



You're browsing the documentation for v2.x and earlier. For v3.x, [click here](#).

Since this documentation is built in the newer "docs," it is not available anymore... unfortunate, in my opinion, since I liked the read.

Since many things still are likely - It is still considerable.

Styleguide - A: Multi-word component names

Component names should always be multi-word, except for root App components, and built-in components provided by Vue, such as `<transition>` or `<component>`.

Bad:

```
1 export default {  
2   name: 'Todo',  
3   // ...  
4 }
```

Good:

```
1 export default {  
2   name: 'TodoItem',  
3   // ...  
4 }
```

Styleguide - A: Prop definitions

Prop definitions should be as detailed as possible.

Prop definitions should always be as detailed as possible, specifying at least type(s):

Bad:



```
1 // This is only OK when prototyping (uncommitted)
2 props: ['status']
```

Good:



```
1 props: {
2   status: String
3 }
```

Styleguide - A: Keyed v-for

Always use key with **v-for**.

Bad:

```
1 <ul>
2   <li v-for="todo in todos">
3     {{ todo.text }}
4   </li>
5 </ul>
```

Good:

```
1 <ul>
2   <li
3     v-for="todo in todos" :key="todo.id"
4   >
5     {{ todo.text }}
6   </li>
7 </ul>
```

Styleguide - A: Avoid v-if with v-for

There are two common cases where this can be tempting:

- To filter items in a list (e.g. `v-for="user in users" v-if="user.isActive"`). In these cases, replace `users` with a new computed property that returns your filtered list (e.g. `activeUsers`).
- To avoid rendering a list if it should be hidden (e.g. `v-for="user in users" v-if="shouldShowUsers"`). In these cases, move the `v-if` to a container element (e.g. `ul`, `ol`).

Bad:

```
1 <ul>
2   <li
3     v-for="user in users"
4     v-if="user.isActive"
5     :key="user.id"
6   >
7     {{ user.name }}
8 </li>
9 </ul>
```


Styleguide - A: Avoid v-if with v-for

Good:

```
1 <ul>
2   <li
3     v-for="user in activeUsers"
4     :key="user.id"
5   >
6     {{ user.name }}
7 </li>
8 </ul>
```

Styleguide - A: Component style scoping

For applications, styles in a top-level App component and in layout components may be global, but all other components should always be scoped.

Bad:

```
1 <template><button class="btn btn-close">x</button></template>
2
3 <style>
4   .btn-close {background-color: red;}
5 </style>
```

Good:

```
1 props: {
2   status: String
3 }
```

Styleguide - A: Component style scoping

For applications, styles in a top-level App component and in layout components may be global, but all other components should always be scoped.

Good:

```
1 <template><button class="button button-close">x</button></template>
2
3 <!-- Using the `scoped` attribute -->
4 <style scoped>
5   .button {
6     border: none;
7     border-radius: 2px;
8   }
9   .button-close { background-color: red;}
10 </style>
```

Styleguide - B: Component files

Whenever a build system is available to concatenate files, each component should be in its own file.

Bad:



```
1 app.component('TodoList', {  
2   // ...  
3 })  
4  
5 app.component('TodoItem', {  
6   // ...  
7 })
```

Good:



```
1 components/  
2 |- TodoList.vue  
3 |- TodoItem.vue
```

Styleguide - B: Base component names

Base components (a.k.a. presentational, dumb, or pure components) that apply app-specific styling and conventions **should all begin with a specific prefix, such as Base, App, or V.**

Bad:



```
1 components/  
2 |- MyButton.vue  
3 |- VueTable.vue  
4 |- Icon.vue
```

Good:



```
1 components/  
2 |- BaseButton.vue  
3 |- BaseTable.vue  
4 |- BaseIcon.vue
```

Styleguide - B: Single-instance component names

Components that should **only ever have a single active instance** should begin with the **The** prefix, to denote that there can be only one.

Bad:



```
1 components/  
2 |- Heading.vue  
3 |- MySidebar.vue
```

Good:



```
1 components/  
2 |- TheHeading.vue  
3 |- TheSidebar.vue
```

Styleguide - B: Tightly coupled component names


Child components that are tightly coupled with their parent should include the parent component name as a prefix.

Bad:



```
1 components/  
2 |- TodoList.vue  
3 |- TodoItem.vue  
4 |- TodoButton.vue
```

Good:



```
1 components/  
2 |- TodoList.vue  
3 |- TodoListItem.vue  
4 |- TodoListItemButton.vue
```

Styleguide - B: Order of words in component names

Component names should start with the highest-level (often most general) words and end with descriptive modifying words.

Bad:

```
1 components/  
2 |- ClearSearchButton.vue  
3 |- ExcludeFromSearchInput.vue  
4 |- LaunchOnStartupCheckbox.vue  
5 |- RunSearchButton.vue  
6 |- SearchInput.vue  
7 |- TermsCheckbox.vue
```

Good:

```
1 components/  
2 |- SearchButtonClear.vue  
3 |- SearchButtonRun.vue  
4 |- SearchInputQuery.vue  
5 |- SearchInputExcludeGlob.vue  
6 |- SettingsCheckboxTerms.vue  
7 |- SettingsCheckboxLaunchOnStartup.vu
```


Styleguide - B: Self-closing components

Components with no content should be self-closing in single-file components, string templates, and JSX - but never in DOM templates.

Bad:



```
1 <!-- In single-file components, string templates, and JSX -->
2 <MyComponent></MyComponent>
```

Good:



```
1 <!-- In single-file components, string templates, and JSX -->
2 <MyComponent/>
```

Styleguide - B: Simple expressions in template

Component templates should **only include simple expressions**, with more complex expressions refactored into computed properties or methods.

Bad:

```
1 {{
2   fullName.split(' ').map((word) => {
3     return word[0].toUpperCase() + word.slice(1)
4   }).join(' ')
5 }}
```

Styleguide - B: Simple expressions in template

Good:



```
1 <!-- In a template -->
2 {{ normalizedFullName }}
```



```
1 <!-- In a template -->
2 // The complex expression has been moved to a computed property
3 computed: {
4   normalizedFullName() {
5     return this.fullName.split(' ')
6       .map(word => word[0].toUpperCase() + word.slice(1))
7       .join(' ')
8   }
9 }
```

Styleguide - B: Directive shorthands

Directive shorthands (: for v-bind:, @ for v-on: and # for v-slot) should be used always or never.

Bad:

```
1 <input
2   v-bind:value="newTodoText"
3   :placeholder="newTodoInstructions"
4 >
```

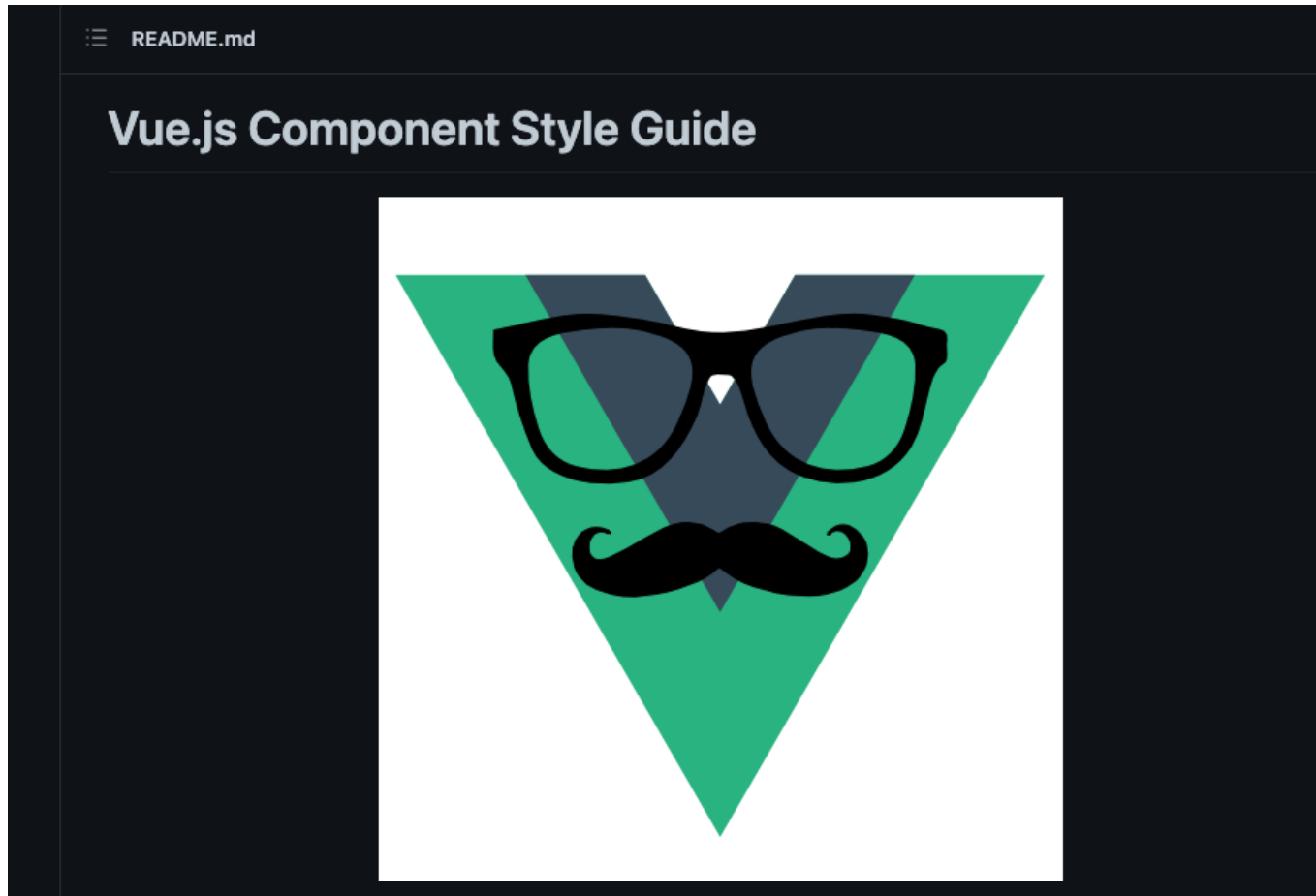
Good:

```
1 <input
2   :value="newTodoText"
3   :placeholder="newTodoInstructions"
4 >
```

Styleguide Conclusion

Most of the things you have seen are obvious but still nice to be heard. There are some more in the styleguide, but I selected the ones worth looking at (personal opinion).

Want more?



Old but still a nice read:

<https://github.com/pablohpsilva/vuejs-component-style-guide>

End



End

That was all for this chapter
