Vue 編程風格建議及注意事項

Vue 命名規則

命名的一致性是基本且重要的 — 因為結構化的命名使得程式檔案易於查找和預測

1. 檔案名稱使用大駝峰 (PascalCase) 例如: `MyComponent.vue`
   https://vuejs.org/v2/style-guide/#Single-file-component-filename-casing-strongly-recommended

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
components/
|- mycomponent.vue
|- myComponent.vue

<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
components/
|- MyComponent.vue
```

2. 元件標籤使用大駝峰 (PascalCase) 例如: `<MyComponent></MyComponent>`
   https://vuejs.org/v2/style-guide/#Component-name-casing-in-templates-strongly-recommended

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
<!-- In single-file components and string templates -->
<mycomponent/>
<!-- In single-file components and string templates -->
<myComponent/>
<!-- In DOM templates -->
<MyComponent></MyComponent>

<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
<!-- In single-file components and string templates -->
<MyComponent/>

<!-- In DOM templates -->
<my-component></my-component>
```

3. 元件標籤應避免使用縮寫，儘可能使用完整英文單字.
   https://vuejs.org/v2/style-guide/#Full-word-component-names-strongly-recommended

```
<!-- Ref: https://pablohpsilva.github.io/vuejs-component-style-guide/#/chinese?id=vue-%E7%BB%84%E4%BB%B6%E5%91%BD%E5%90%8D -->
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
<btn-group></btn-group> <!-- 虽然简短但是可读性差. 使用 `button-group` 替代 -->
<ui-slider></ui-slider> <!-- ui 前缀太过于宽泛，在这里意义不明确 -->
<slider></slider> <!-- 与自定义元素规范不兼容 -->

<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
<AppHeader></AppHeader>
<UserList></UserList>
<RangeSlider></RangSlider>
```

4. 相依性高的父子元件，應該要有相同的前置檔名.
   https://vuejs.org/v2/style-guide/#Tightly-coupled-component-names-strongly-recommended

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
components/
|- TodoList/
   |- Item/
      |- index.vue
      |- Button.vue
   |- index.vue

<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
components/
|- TodoList.vue
|- TodoListItem.vue
|- TodoListItemButton.vue
```

5. Prop 名稱格式必須為小駝峰(**camelCase**)，但在標籤內的屬性名均為 `-` 分隔 (**kebab-case**)
   https://vuejs.org/v2/style-guide/#Prop-name-casing-strongly-recommended

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
props: {
  'greeting-text': String
}
<WelcomeMessage greetingText="hi"/>

<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
props: {
  greetingText: String
}
<WelcomeMessage greeting-text="hi"/>
```

6. 統一使用 vue 指令縮寫 `: for v-bind:, @ for v-on: and # for v-slot`
   https://vuejs.org/v2/style-guide/#Directive-shorthands-strongly-recommended

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
<input
  v-bind:value="newTodoText"
  :placeholder="newTodoInstructions"
>

<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
<input
  :value="newTodoText"
  :placeholder="newTodoInstructions"
>
```

Vue 開發基本注意事項

1. Vue 檔案請遵循結構順序: `<template>`, `<script>`, and `<style>`.

https://vuejs.org/v2/style-guide/index.html#Single-file-component-top-level-element-order-recommended

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
<style>/* ... */</style>
<script>/* ... */</script>
<template>...</template>

<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
<template>...</template>
<script>/* ... */</script>
<style>/* ... */</style>
```

2. Javascript 字串請使用單引號 `'` ; 標籤屬性字串請使用雙引號 `"` .
   https://vuejs.org/v2/style-guide/#Quoted-attribute-values-strongly-recommended

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
<input type=text>
<AppSidebar :style={width:sidebarWidth+'px'}>

<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
<input type="text">
<AppSidebar :style="{ width: sidebarWidth + 'px' }">
```

3. 多屬性的元素標籤，請使用一行顯示一個屬性，使程式可讀性更高https://vuejs.org/v2/style-guide/#Multi-attribute-elements-strongly-recommended

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
<img src="https://vuejs.org/images/logo.png" alt="Vue Logo">

<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
<img
  src="https://vuejs.org/images/logo.png"
  alt="Vue Logo"
>
```

4. 不應該在元件範本內使用複雜的運算邏輯，而應該寫在 methods/computed 設定中.
   https://vuejs.org/v2/style-guide/#Multi-attribute-elements-strongly-recommended

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
<div>
  {{
    fullName.split(' ').map(function (word) {
      return word[0].toUpperCase() + word.slice(1)
    }).join(' ')
  }}
</div>

<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
<!-- In a template -->
<div>{{ normalizedFullName }}</div>
```

```
<!--The complex expression has been moved to a computed property-->
computed: {
  normalizedFullName: function () {
    return this.fullName.split(' ').map(function (word) {
      return word[0].toUpperCase() + word.slice(1)
    }).join(' ')
  }
}
```

5. 元件的 data 選項一定是 function，而非物件。
   https://vuejs.org/v2/style-guide/#Component-data-essential

```
//-----------------------
//--------- 避免 --------
//-----------------------
export default {
  data: {
    foo: 'bar'
  }
}

//-----------------------
//--------- 推薦 --------
//-----------------------
export default {
  data () {
    return {
      foo: 'bar'
    }
  }
}
```

6. 元件中的 props 定義應該儘可能詳盡，包含: 型別、是否必要、欄位驗證規則 等
   https://vuejs.org/v2/style-guide/#Prop-definitions-essential

```
//-----------------------
//--------- 避免 --------
//-----------------------
props: ['status']

//-----------------------
//--------- 推薦 --------
//-----------------------
props: {
  status: {
    type: String,
    required: true,
    validator: function (value) {
      return [
        'syncing',
        'synced',
        'version-conflict',
        'error'
      ].indexOf(value) !== -1
    }
  }
}
```

7. v-for 都應該要有 `key` 的設置.
   https://vuejs.org/v2/style-guide/#Keyed-v-for-essential

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
<ul>
  <li v-for="todo in todos">
    {{ todo.text }}
  </li>
</ul>
```

```
<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
<ul>
  <li
    v-for="todo in todos"
    :key="todo.id"
  >
    {{ todo.text }}
  </li>
</ul>
```

8. v-if 不可以與 v-for 同時使用在同一個標籤上.
   https://vuejs.org/v2/style-guide/#Avoid-v-if-with-v-for-essential

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
<ul>
  <li
    v-for="user in users"
    v-if="user.isActive"
    :key="user.id"
  >
    {{ user.name }}
  </li>
</ul>

<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
<ul v-if="shouldShowUsers">
  <li
    v-for="user in users"
    :key="user.id"
  >
    {{ user.name }}
  </li>
</ul>
```

9. 當使用 v-if + v-else 時，應該配合使用 `key` 屬性，以便識別. (或者填寫明確的註解標籤說明)
   https://vuejs.org/v2/style-guide/index.html#v-if-v-else-if-v-else-without-key-use-with-caution

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
<div v-if="error">
  Error: {{ error }}
</div>
<div v-else>
  {{ results }}
</div>

<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
<div
  v-if="error"
  key="search-status"
>
  Error: {{ error }}
</div>
<div
  v-else
  key="search-results"
>
  {{ results }}
</div>
```

(進階)其他建議

1. 儘可能簡化並拆分運算(computed)屬性.
   https://vuejs.org/v2/style-guide/#Simple-computed-properties-strongly-recommended

```
// 避免
computed: {
  price: function () {
    var basePrice = this.manufactureCost / (1 - this.profitMargin)
    return (
      basePrice -
      basePrice * (this.discountPercent || 0)
    )
  }
}

// 推荐
computed: {
  basePrice: function () {
    return this.manufactureCost / (1 - this.profitMargin)
  },
  discount: function () {
    return this.basePrice * (this.discountPercent || 0)
  },
  finalPrice: function () {
    return this.basePrice - this.discount
  }
}
```

2. 避免直接使用 this.$parent
   https://vuejs.org/v2/style-guide/index.html#Implicit-parent-child-communication-use-with-caution

```
<!----------------------->
<!--------- 避免 -------->
<!----------------------->
Vue.component('TodoItem', {
  props: {
    todo: {
      type: Object,
      required: true
    }
  },
  template: '<input v-model="todo.text">'
})

Vue.component('TodoItem', {
  props: {
    todo: {
      type: Object,
      required: true
    }
  },
  methods: {
    removeTodo () {
      var vm = this
      vm.$parent.todos = vm.$parent.todos.filter(function (todo) {
        return todo.id !== vm.todo.id
      })
    }
  },
  template: `
    <span>
      {{ todo.text }}
      <button @click="removeTodo">
        X
      </button>
    </span>
  `
})
<!----------------------->
<!--------- 推荐 -------->
<!----------------------->
Vue.component('TodoItem', {
  props: {
    todo: {
      type: Object,
      required: true
    }
  },
  template: `
    <input
      :value="todo.text"
      @input="$emit('input', $event.target.value)"
    >
  `
})

Vue.component('TodoItem', {
  props: {
    todo: {
      type: Object,
      required: true
    }
  },
  template: `
    <span>
      {{ todo.text }}
      <button @click="$emit('delete')">
        X
      </button>
    </span>
  `
})
```

3. 建議使用 Vuex 來進行全域狀態管理 (global state management)，避免直接使用 this.$root 或者全域性的事件(global event bus).
   https://vuejs.org/v2/style-guide/index.html#Non-flux-state-management-use-with-caution

▼ 避免

```
new Vue({
  data: {
    todos: []
  },
  created: function () {
    this.$on('remove-todo', this.removeTodo)
  },
  methods: {
    removeTodo: function (todo) {
      var todoIdToRemove = todo.id
      this.todos = this.todos.filter(function (todo) {
        return todo.id !== todoIdToRemove
      })
    }
  }
})
```

▼ 推薦

```
// store/modules/todos.js
export default {
  state: {
    list: []
  },
  mutations: {
    REMOVE_TODO (state, todoId) {
      state.list = state.list.filter(todo => todo.id !== todoId)
    }
  },
  actions: {
    removeTodo ({ commit, state }, todo) {
      commit('REMOVE_TODO', todo.id)
    }
  }
}
```

```
<!-- TodoItem.vue -->
<template>
  <span>
    {{ todo.text }}
    <button @click="removeTodo(todo)">
      X
    </button>
  </span>
</template>

<script>
import { mapActions } from 'vuex'

export default {
  props: {
    todo: {
      type: Object,
      required: true
    }
  },
  methods: mapActions(['removeTodo'])
}
</script>
```

4. ~~Component/instance options order.~~ **https://vuejs.org/v2/style-guide/index.html#Component-instance-options-order-recommended** (Eslint for vue !???)

```
<!-- Ref: https://pablohpsilva.github.io/vuejs-component-style-guide/#/chinese?id=%E7%BB%84%E4%BB%B6%E7%BB%93%E6%9E%84%E5%8C%96 -->
<template lang="html">
  <div class="Ranger__Wrapper">
    <!-- ... -->
  </div>
</template>

<script type="text/javascript">
  export default {
    // 不要忘记了 name 属性
    name: 'RangeSlider',
    // 使用组件 mixins 共享通用功能
    mixins: [],
    // 组成新的组件
    extends: {},
    // 组件属性、变量
    props: {
      bar: {}, // 按字母顺序
      foo: {},
      fooBar: {},
    },
    // 变量
    data() {},
    computed: {},
    // 使用其它组件
    components: {},
    // 方法
    watch: {},
    methods: {},
    // 生命周期函数
    beforeCreate() {},
    mounted() {},
  };
</script>

<style scoped>
```

```
  .Ranger__Wrapper { /* ... */ }
</style>
```

5. ~~Element attribute order.~~
   https://vuejs.org/v2/style-guide/index.html#Element-attribute-order-recommended
6. ~~Private property names. Use module scoping to keep private functions inaccessible from the outside. If that's not possible, always use the $_ prefix~~
   https://vuejs.org/v2/style-guide/#Private-property-names-essential

Ref:

- https://vuejs.org/v2/style-guide
- https://docs.gitlab.com/ee/development/fe_guide/style/vue.html
- https://medium.com/swlh/best-practices-for-writing-vue-apps-component-naming-and-organization-6c1593a251a0