Vue component patterns

권영근

Common

- 기본적인 컴포넌트 등록과 통신 방식
- 주로 많이 사용하는 방식
- Vue는 one-way 데이터 통신을 지향, 서비스가 커지면 vuex 사용

```
  </div>
</template>
```

AppHeader, AppContent (하위 컴포넌트)

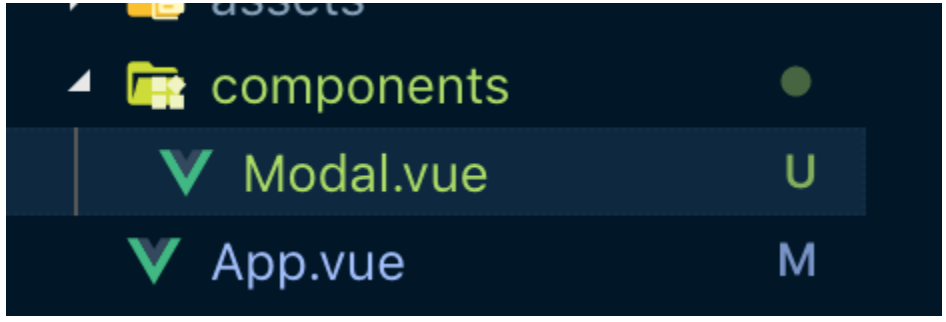
```
<template>
  <h1>{{ title}}</h1>
</template>
```

```
<template>
  <div>
    <ul>
      <li v-for="item in items">{{item}}</li>
```

Slot

- 마크업 확장이 가능한 컴포넌트
- 데이터를 하위 컴퍼넌트로 내리지 않고, 상위컴포넌트(컴포넌트를 등록한 곳)에서 정의한다.
    - 즉, 데이터는 상위컴포넌트에 머물게 된다.
- 언제 쓸까
    - 다양한 표현, 요구사항에 유연하게 대응하기 위해
    - ex) 2번째 아이템에 버튼이 있었으면.. or 모달창의 내용양식을 유연하게 바꾸기위해

modal.vue




```vue
<template>
  <transition name="modal">
    <div class="modal-mask">
      <div class="modal-wrapper">
        <div class="modal-container">
          <div class="modal-header">
            <slot name="header">default header</slot>
          </div>

          <div class="modal-body">
            <slot name="body">default body</slot>
          </div>

          <div class="modal-footer">
            <slot name="footer">
              default footer
              <button class="modal-default-button" @click="$emit('close')">OK</button>
            </slot>
          </div>
        </div>
```

App.vue

```
<div id="app">
  <button id="show-modal" @click="showModal = true">Show Modal</button>
  <modal v-if="showModal" @close="showModal = false">
    <!-- 여기에 요소를 정의한다! -->
    <h1 slot="header">custom header</h1>
    <div slot="body">눈누난나~~~</div>
    <!-- <img slot="body" src="./assets/logo.png" alt="로고" /> -->
  </modal>
</div>
```

Contorlled Component

- 실제 이런 이름을 가진 패턴이 아닌, 캡틴판교님이 지은 이름
- 의존도를 낮추고 결합력을 높이는 방식
- 슬롯이랑 비슷한 목적으로 컴포넌트를 많이 쪼갤때 이용
- 상위에서 데이터를 관리하여 하위 컴포넌트의 결합력을 높임.

App.vue

```vue
<template>
  <div id="app">
    <!-- <input type="checkbox" v-model="checked" /> -->
    <check-box v-model="checked"></check-box>
  </div>
</template>
```

CheckBox.vue

```vue
<template>
  <div>
    <input type="checkbox" :value="value" @click="toggle" />
  </div>
</template>

<script>
export default {
  props: {
    value: Boolean
  },

  methods: {
    toggle() {
      this.$emit("input", !this.value);
    }
  }
};
```

Renderless Component

- 템플릿이 없는 컴포넌트
- 여러가지 유형이 있다.
- 예제는 데이터를 가져오는 상황
- 데이터 요청과 보여주는 영역을 구분하고 싶을 때 사용 (created() 사용하기 싫어!)

App.vue

```
<template>
  <div id="app">
    <fetch-data url="https://jsonplaceholder.typicode.com/users/1">
      <!-- 여기에 데이터가 노출된다! -->

      <div slot-scope="{response, loading}">
        <div v-if="!loading">{{ response}}</div>

        <div v-if="loading">Loading...</div>
      </div>
    </fetch-data>
  </div>
</template>
```

FetchData.vue

```
<script>
import axios from "axios";
export default {
  props: ["url"],
  data() {
    return {
      response: null,
      loading: true
    };
  },
  created() {
    axios
      .get(this.url)
      .then(response => {
        this.response = response.data;
        this.loading = false;
      })
      .catch(error => {
        alert("[ERROR] fetching the data", error);
        console.log(error);
```

코드가 더 있는데 화면을 잡아먹어 자름

Reference

캡틴 판교 - Vue Advanced 강의