

2025년 상반기 K-디지털 트레이닝

TodoList 예제 리팩토링

[KB] IT's Your Life

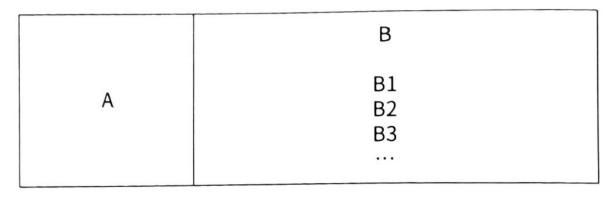
컴포넌트의 단위 기준

재사용성이 높은가 관리의 편의성



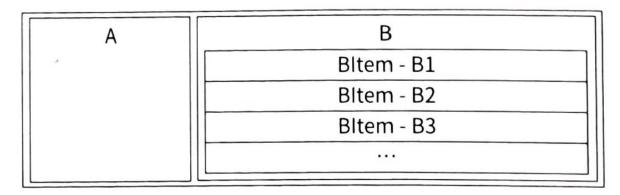
○ 컴포넌트 분할 기준

- 한 번에 변경되는 데이터를 렌더링하는 UI 단위로 컴포넌트를 분할
 - → 변경된 데이터만 다시 렌더링



하나의 컴포넌트 - Blog

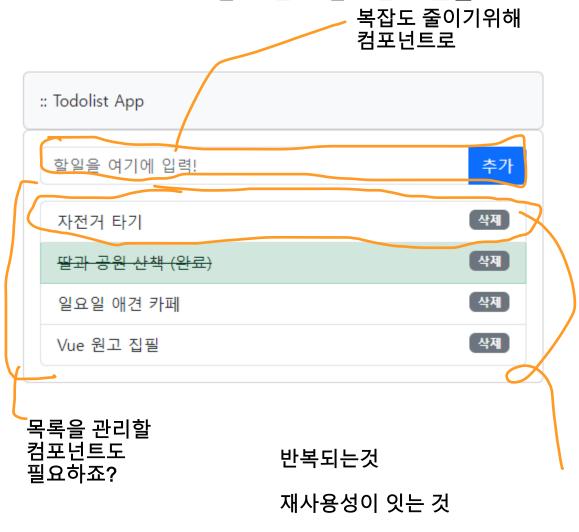




세분화된 컴포넌트

- Blog
- A
- B
 - Bltem

TodoList 앱 화면의 컴포넌트 분할



App



이런게 많죠?

데이터는 누가 관리하느냐

🧿 데이터 관리

화면 단위의 데이터와 메서드 기능을 상위 컴포넌트에 집줍

```
[관리해야 할 데이터]
App이 관리함
{

todoList:[
    { id: 1, todo:"자전거 타기", completed: false },
    { id: 2, todo:"딸과 공원 산책", completed: true },
    { id: 3, todo:"일요일 애견 카페", completed: false },
    { id: 4, todo:"Vue 원고 집필", completed: false },
]
}
```

☑ 메서드 이벤트들

[메서드 목록]

addTodo(todo)
 html에 바로 쓸 때는 click으로 했지만

- deleteTodo(id) 지금은 컴포넌트화하여 사용하므로 사용자정의 이벤트로 만들어 사용해야함

toggleCompleted(id)

[수신 이벤트 목록]

- add-todo : 전달 인자-todo

- delete-todo : 전달 인자-id

- toggle-completed : 전달 인자-id

▽ 컴포넌트

설계 순서

전체구조 파악 컴포넌트 파악

정보전달 방법 파악

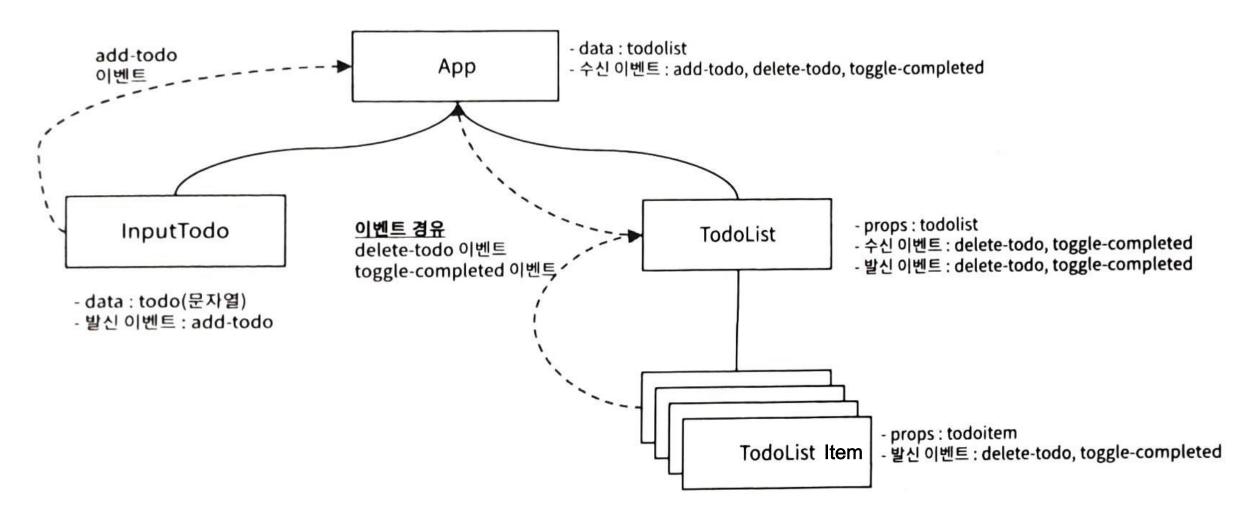
이벤트 파악

[컴포넌트 목록]

- * App 컴포넌트
 - data : todoList 배열
 - methods : addTodo, deleteTodo, toggleCompleted 메서드
 - 수신 이벤트 : add-todo, delete-todo, toggle-completed
- * InputTodo 컴포넌트
 - data : todo 문자열 값
 - 발신 이벤트 : add-todo
- * TodoList 컴포넌트
 - props : todoList 배열(todoList)
 - 발신이벤트 : delete-todo(경유), toggle-completed(경유)
 - 수신이벤트 : delete-todo(경유), toggle-completed(경유)
- * TodoListItem 컴포넌트
 - props : todoItem(todoList 배열의 값 하나)
 - 발신 이벤트 : delete-todo, toggle-completed

6

☑ TodoList 앱 컴포넌트 계층 구조



2 속성과 이벤트를 조합한 리팩토링

💟 프로젝트 만들기

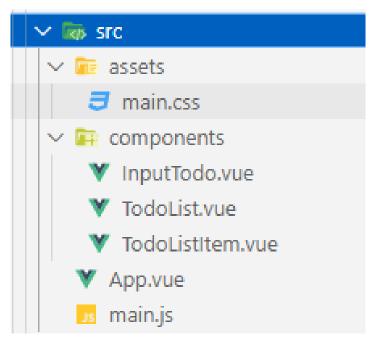
npm init vue todolist-app
--> 모두 No 선택
cd todolist-app
npm install
npm install bootstrap@5
bootstrap버전5설치

○ 컴포넌트 파일 생성

- src/assets/main.css
- src/components/InputTodo.vue
- src/components/TodoList.vue
- src/components/TodoLsitItem.vue

App





2 속성과 이벤트를 조합한 리팩토링

src/main.js

부트스트랩을 사용하기위한 준비조건 전체 컴포넌트에 영향을 줌

```
import 'bootstrap/dist/css/bootstrap.css'
import './assets/main.css' 우리가 자체적으로
만든 스타일 정의는
import { createApp } from 'vue'따로 여기로
import App from './App.vue'

createApp(App).mount('#app')
```

☑ src/assets/main.css 기본 스타일 정의

```
body {
 margin: 0;
 padding: 0;
 font-family: sans-serif;
.title {
 text-align: center;
 font-weight: bold;
 font-size: 20pt;
.todo-done {
 text-decoration: line-through;
.container {
 padding: 10px 10px 10px;
.panel-borderless {
 border: 0;
 box-shadow: none;
.pointer {
 cursor: pointer;
```

src/components/InputTodo.vue

```
<template>
</template>
</script>
    export default {
        name : "InputTodo",
    }
</script>
```

src/components/TodoListItem.vue

```
<template>
</template>
</script>
    export default {
        name : "TodoListItem",
    }
</script>
```

src/components/TodoList.vue

```
<template>
</template>
</script>
    export default {
        name : "TodoListI",
    }
</script>
```

```
<script>
 import TodoList from './components/TodoList.vue'
                                                  컴포넌트를 사용하기
 import InputTodo from './components/InputTodo.vue'
                                                  위해 import
 let ts = new Date().getTime()
 export default {
   name: "App",
   components : { InputTodo, TodoList }, 지역으로 지정
   data() {
     return {
       todoList : [
         { id: ts, todo:"자전거 타기", completed: false },
         { id: ts+1, todo:"딸과 공원 산책", completed: true },
         { id: ts+2, todo:"일요일 애견 카페", completed: false },
         { id: ts+3, todo:"Vue 원고 집필", completed: false },
   },
```

```
methods: {
      addTodo(todo) {
        if (todo.length >= 2) {
          this.todoList.push({ id: new Date().getTime(), todo: todo, completed: false });
      deleteTodo(id) {
        let index = this.todoList.findIndex((item) => id === item.id);
        this.todoList.splice(index, 1);
      },
      toggleCompleted(id) {
        let index = this.todoList.findIndex((item) => id === item.id);
        this.todoList[index].completed = !this.todoList[index].completed;
</script>
```

2 속성과 이벤트를 조합한 리팩토링

☑ InputTodo.vue 컴포넌트



○ 엔터를 치거나 추가 버튼을 눌렀을 때 새 할 일 등록되게 함

src/components/InputTodo.vue

```
<template>
    <div class="row mb-3">
        <div class="col">
            <div class="input-group">
                <input type="text" class="form-control"</pre>
                    placeholder="할일을 여기에 입력!"
                    v-model.trim="todo"
                    @keyup.enter="addTodoHandler" />
                <span class="btn btn-primary input-group-addon"</pre>
                    @click="addTodoHandler">추가</span>
            </div>
        </div>
    </div>
</template>
```

src/components/InputTodo.vue

```
<script>
   export default {
       name : "InputTodo",
       data() {
          return { todo : "" } 양방향 바인드 시킬꺼야 todo변수는
                            사용자 이벤트 정의
       emits : ["add-todo"],
       methods : {
                                          클릭시 발생하는 이벤트 핸들러
          addTodoHandler() {
                                          todo내용의 유효성을 검사함
              if (this.todo.length >= 3) {
                  this.$emit('add-todo', this.todo);
                  this.todo = "";
       },
</script>
```

TodoListItem.vue 컴포넌트

○ 제목을 클릭했을 때 완료 여부 toggle 목록 제어. 정형화 되어있는



목록처리 목록관리 component ㄴ 1개 아이템 component

src/components/TodoListItem.vue

```
<script>
export default {
    name: 'TodoListItem',
    props: {
        todoItem: { type: Object, required: true }, todoItem 검사
    },
    emits: ['delete-todo', 'toggle-completed'],
};
</script>
```

src/components/TodoListItem.vue

```
<template>
 <li
   class="list-group-item"
    :class="{ 'list-group-item-success': todoItem.completed }"
    @click="$emit('toggle-completed', todoItem.id)"
    <span class="pointer" :class="{ 'todo-done': todoItem.completed }">
     {{ todoItem.todo }} {{ todoItem.completed ? '(완료)' : '' }}
    </span>
    <span
     class="float-end badge bg-secondary pointer"
     @click.stop="$emit('delete-todo', todoItem.id)"
     >삭제</span
 </template>
```

TodoList.vue 컴포넌트



src/components/TodoList.vue

```
cscript>
import TodoListItem from './TodoListItem.vue';

export default {
  name: 'TodoList',
  components: { TodoListItem },
  props: {
    todoList: { type: Array, required: true },
  },
  emits: ['delete-todo', 'toggle-completed'],
};
</script>
```

src/components/TodoList.vue

```
<template>
 <div class="row">
   <div class="col">
     <TodoListItem ■
        v-for="todoItem in todoList"
         :key="todoItem.id"
         :todoItem="todoItem" 데이터 전잘
                                                         이벤트 중계
         @delete-todo="$emit('delete-todo', $event)"
        @toggle-completed="$emit('toggle-completed', $event)"
       />
     </div>
 </div>
</template>
```