

뷰의 기본은 여기까지

2025년 상반기 K-디지털 트레이닝

# TodoList 예제

[KB] IT's Your Life

html하나에 몽땅 다 넣으니까 복잡해진다.

기능단위로 세분화 시키는 것이 필요하다

파일을 분리해야한다. 소스의 복잡도를 낮춰야한다.

js에서 분리시키는 방법은 모듈화.

기본적인 웹브는 모듈화를 지원하지 않는다.

node에 모듈화를 쓰되

웹브에 배포할 때는 통합시킴

vite, webpack등등으로 통합시킬거고

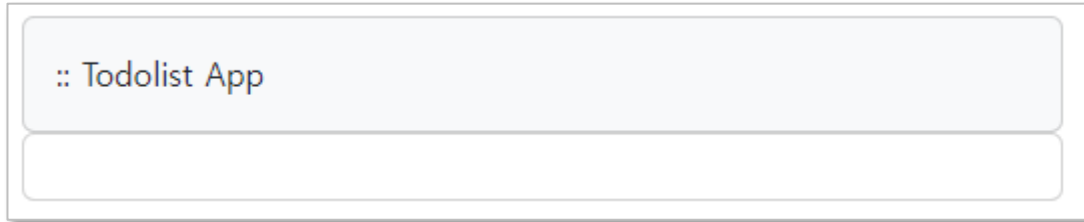
나눌 단위는? 컴포넌트라는 개념이 등장한다. 이 장 다음

컴포넌트 하나당 한개의 파일, 한개의 모듈로 정할거임.

가상돔의 계층구조와 연관이 있다.

# 1 TodoList 예제

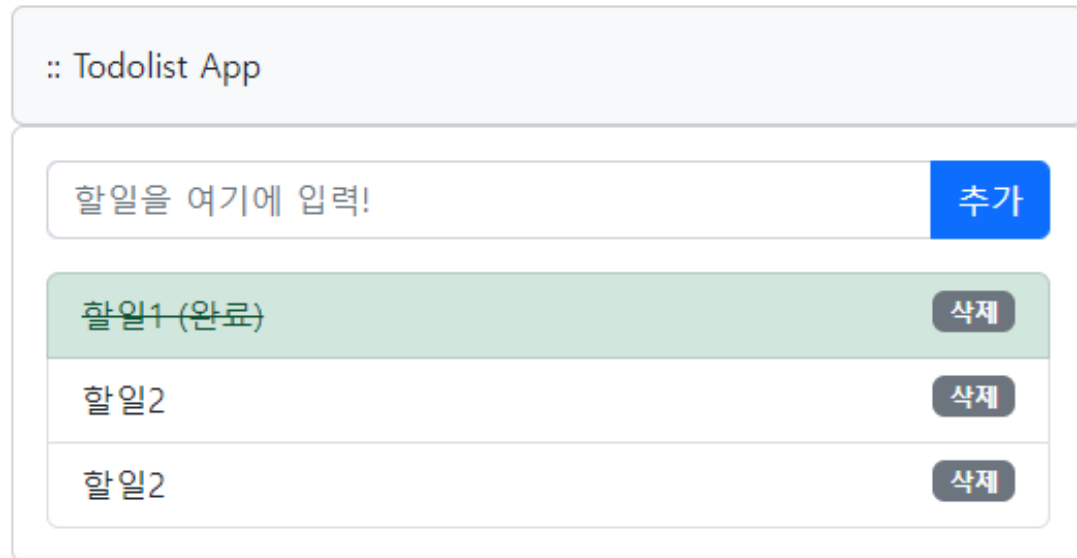
## ✓ 초기 화면 시안



:: Todolist App

0. 앱 모양새 대략적으로 스케치

## ○ 최종 완성 시안



:: Todolist App

할 일을 여기에 입력! 추가

할 일1 (완료)	<span>삭제</span>
할 일2	<span>삭제</span>
할 일2	<span>삭제</span>

CRUD

update는 완료 처리하는 걸로

bootstrap vue 활용

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>06-11</title>
    <link rel="stylesheet" href="https://unpkg.com/bootstrap@5.2.3/dist/css/bootstrap.min.css" />
    <style>
      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 10px;
      }
    </style>
  </head>
  <body>
```

# 1 TodoList 예제

## 06-11.html

```
.panel-borderless {
  border: 0;
  box-shadow: none;
}
.pointer {
  cursor: pointer;
}
</style>
</head>
<body>
  <div id="app" class="container">
    <div class="card card-body bg-light">
      <div class="title">:: Todolist App</div>
    </div>
    <div class="card card-default card-borderless">
      <div class="card-body">
        <!-- 추가 예정 -->
      </div>
    </div>
  </div>
  <script src="https://unpkg.com/vue"></script>
  <script type="text/javascript"></script>
</body>
</html>
```

# 1 TodoList 예제

## ✓ 템플릿 구성하기

:: Todolist App

✓

할일1 (완료)

할일2

할일2

인풋과 버튼이 따로가  
아닌 붙어있음

내가 어떤 데이터를  
제어해야 하느냐를  
생각해보자

데이터를 구분지어야  
하는 ID관리

# 1 TodoList 예제

## 06-12.html

```
...  
<div class="card-body">  
  <div class="row mb-3">  
    <div class="col">  
      <div class="input-group">  
        <input id="msg" type="text" class="form-control" name="msg" placeholder="할일을 여기에 입력!" />  
        <span class="btn btn-primary input-group-addon">추가</span>  
      </div>  
    </div>  
  </div>
```

인풋과 버튼이 붙는다  
한 덩어리가 된다

The screenshot shows a web application titled ":: Todolist App". It features a form with a text input field containing the placeholder text "할일을 여기에 입력!" and a blue button labeled "추가". Below the form is a list of tasks. The first task is "할일1 (완료)" with a green background and a "삭제" button. The second and third tasks are both labeled "할일2" and each has a "삭제" button. A red rectangle highlights the form area.

## 1 TodoList 예제

## 06-12.html

```

<div class="row">
  <div class="col">
    <ul class="list-group">
      <li class="list-group-item list-group-item-success">
        <span class="todo-done pointer">할일1 (완료)</span>
        <span class="float-end badge bg-secondary pointer">삭제</span>
      </li>
      <li class="list-group-item">
        <span class="pointer">할일2</span>
        <span class="float-end badge bg-secondary pointer">삭제</span>
      </li>
      <li class="list-group-item">
        <span class="pointer">할일2 </span>
        <span class="float-end badge bg-secondary pointer">삭제</span>
      </li>
    </ul>
  </div>
</div>

```

...

:: Todolist App

할일을 여기에 입력!

추가

할일1 (완료)

삭제

할일2

삭제

할일2

삭제

# 1 TodoList 예제

## ✓ 데이터 정의

- todo                    텍스트 박스에 사용자가 입력하는 내용을 받기 위한 data
- todoclist              추가한 todo들의 목록, todo 한 건은 다음과 같음
  - id                    todo 한 건의 고유키, 여기서는 timestamp를 이용
  - todo                todo 내용
  - completed        완료 여부 true/false



# 1 TodoList 예제

## ✓ 메서드 정의

- addTodo                    텍스트 박스에 할 일(todo)를 입력하고 엔터를 누르거나 추가 버튼을 누른 경우 todolist에 새로운 todo를 추가
- deleteTodo                삭제 버튼을 클릭하면 해당 id의 할 일(todo)를 찾아서 삭제
- toggleCompleted        할 일(todo) 한 건을 클릭하면 해당 id의 completed 값을 토글

# 1 TodoList 예제

## 06-13.html

```
<script type="text/javascript">
  var ts = new Date().getTime();

  var vm = Vue.createApp({
    name: "App",
    data() {
      return {
        todo: "", v-model로 input처리
        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 },
        ],
      };
    },
  });
```

v-for로 나열

completed로 스타일  
적용여부결정하고

# 1 TodoList 예제

## 06-13.html

```
methods: {  
  addTodo() {  
    if (this.todo.length >= 2) {  
      this.todolist.push({ id: new Date().getTime(), todo: this.todo, completed: false });  
      this.todo = "";  
    }  
  },  
  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;  
  },  
}  
}).mount("#app");  
</script>
```

# 1 TodoList 예제

## ✓ 템플릿 작성


- 스크립트 data와 템플릿 간의 바인딩
  - 데이터 바인딩
  - 이벤트 바인딩
  - 스타일 바인딩

# 1 TodoList 예제

## 06-14.html

```
...
<div class="card card-default card-borderless">
  <div class="card-body">
    <div class="row mb-3">
      <div class="col">
        <div class="input-group">
          <input
            id="msg"
            type="text"
            class="form-control"
            name="msg"
            placeholder="할일을 여기에 입력!"
            v-model.trim="todo"
            @keyup.enter="addTodo"
          />
          <span class="btn btn-primary input-group-addon" @click="addTodo">추가</span>
        </div> <!-- .input-group -->
      </div> <!-- .col -->
    </div> <!-- .row -->
  </div>
</div>
```

## 1 TodoList 예제

 06-14.html 초보자들은 method로 sort를 일일히함 매번 이벤트 추가해서  
고수들은 computed를 활용하여 todomlist 말고 "~ in computed list"로 하겠지

```

<div class="row">
  <div class="col">
    <ul class="list-group">
      <li
        v-for="todoitem in todomlist"
        :key="todoitem.id" key 속성 바인딩 아디
        class="list-group-item"
        :class="{ 'list-group-item-success': todoitem.completed } " 객체로 클래스명 지정
        @click="toggleCompleted(todoitem.id)" 함수를 인자 넘기며
      >
        <span class="pointer" :class="{ 'todo-done': todoitem.completed }"> 객체로 클래스명 지정
          {{todoitem.todo}} {{ todoitem.completed ? "(완료)" : "" }} 내용 변경 반영
        </span>
        <span class="float-end badge bg-secondary pointer" @click.stop="deleteTodo(todoitem.id)"
          >삭제</span>
      </li>
    </ul>
  </div> <!-- .col -->
</div> <!-- .row -->
</div> <!-- .card-body -->
</div> <!-- .card -->

```

...

인자로 넘겨준 정보로  
해당 id의 todoitem의  
completed 변수값  
변경