(디지털컨버전스)스마트 콘텐츠와 웹 응용응용SW개발자 양성과정

강사 - Sanghoon Lee(이상훈)
gcccompil3r@gmail.com
학생 –Joongyeon Kim(김중연)
jjjr69@naver.com

2021년 7월 14일 질문노트

[김중연]

수정 전

```
<div class="home">
    <div id="header">
        <!-- router-link는 어디로 갈지 URL 맵핑해주는 Spring의 Controller 같은 역할
            to: Home은 이 녀석을 클릭했을 때 router/index.js에 등록된 Home으로 가게 만듬
            class: nav-link의 경우 Navigation Bar 같은 역할을 해줄 수 있게 구성함을 의미
            active-class: 링크가 활성화 되었을때 적용된 CSS를 의미함 -->
        <router-link :to="{ name: 'Home' }"
                class="nav-link"
                active-class="active">
            Home
```

to: Home 이라고 적어주셨는데

to 앞에 있는 콜론을 빼도 정상작동을 해서요

혹시 다른 상황에서 콜론을 붙이는 건가요?

수정 후

```
<!-- HTML 요소 파트 작성 - Vue랑 연결 시킬 형식으로 작성해도 무방 -->
<template>
    <div class="home">
        <div id="header">
            <!-- router-link는 어디로 갈지 URL 맵핑해주는 Spring의 Controller 같은 역할
                to= Home은 이 녀석을 클릭했을 때 router/index.js에 등록된 Home으로 가게 만듬 (<- 중요한 개념!)
                class: nav-link의 경우 Navigation Bar 같은 역할을 해줄 수 있게 구성함을 의미
                active-class: 링크가 활성화 되었을때 적용된 CSS를 의미함 -->
            <router-link to="{ name: 'Home' }"
                    class="nav-link"
                    active-class="active">
                Home
            </router-link>
```

Axios의 데이터 주고 받는 과정

- **불러오기** : axios.get(url[, config])
- **입력하기** : axios.post(url[, data[, config]])
- **수정하기** : axios.patch(url[, data[, config]])
- **삭제하기** : axios.delete(url[, config])

actions.js M

```
generateRandomNumber ({ commit }) {
    console.log(commit)

    // axios.get: GET 요청
    // axios.post: POST 요청
    // 특정 URL로 GET 혹은 POST, 그 외의 요청을 보낼 수 있음
    // 보내고 넘겨 받은 데이터는 .then((res)) 절로 수신함
    // .catch((res)) 절은 오류가 발생했을 경우임
    // 어찌 되었든 응답받은 데이터는 res가 가지고 있음
    axios.get('http://localhost:8888/random')
        .then((res) => {
            commit(SUCCESS_GEN_RAND_NUM, parseInt(res.data.randNumber))
        })
        .catch((res) => {
            commit(FAIL_GEN_RAND_NUM, res)
        })
}
```

Test.vue M

```
        <b>random: {{ this.$store.getters.ranodmFromSpring }}</b><br>
        <input type="button" @click="randomNumber()" value="random"/><br>
    </div>
</div>
mplate>
```

mutations.js M

```
//스프링 랜덤 데이터 통신
[SUCCESS_GEN_RAND_NUM] (state, payload) {
    console.log('payload=' + payload)
    state.randomFromSpring = payload
},
[FAIL_GEN_RAND_NUM] () {
    console.log('통신 에러!')
}
```

states.js M

```
export default {
    // TODO
    todoItems: [],
    editingId: 0,
    nextTodoId: 1,
    filter: null,
    // 몬스터
    monsterElements: [],
    nextMonsterId: 1,
    // 스프링과 랜덤 데이터 통신
    randomFromSpring: 0
}
```

getters.js M

```
export default {
    filteredTodoItems (state) {
        if (!state.filter) {
            return state.todoItems
        }
    },
    getMonsterElements (state) {
        return state.monsterElements
    },
    ranodmFromSpring (state) {
        return state.ranodmFromSpring
    }
}
```

Toggletodostatus의 데이터 주고받는 과정??

actions.js M

```
toggleTodoStatus({ commit }, id) {
    commit(TOGGLE_TODO_STATUS, id)
},
```

mutations.js M

```
[TOGGLE_TODO_STATUS] (state, id) {
    // 현재 todoItems 배열에서 id로 들어온 todoItem을 찾는다
    const filtered = state.todoItems.filter(todoItem => {
        return todoItem.id === id
    })

    console.log('filtered:' + JSON.stringify(filtered))

    filtered.forEach(todoItem => {
        todoItem.done = !todoItem.done
    })
},
```

states.js M

```
export default {
    // TODO
    todoItems: [],
    editingId: 0,
    nextTodoId: 1,
    filter: null,
```

Todo.vue M

```
<template>
    <div class="todo">
        <todo-header></todo-header>
        <!-- todo-input 컴포넌트가 emit(addTodo, ~~)를 하면 onAddTodo()가 동작함 -->
        <todo-input v-on:addTodo="onAddTodo"></todo-input>
        <todo-list
            v-on:removeTodo="onRemoveTodo"
            v-on:editTodo="onEditTodo"
            v-on:toggleTodoStatus="onToggleTodoStatus">
        </todo-list>
        <todo-footer v-on:removeAll="onClearAll"></todo-footer>
    </div>
</template>

onToggleTodoStatus (id) {
    this.toggleTodoStatus(id)
    this.save()
},
```

TodoItem.vue M

```
            <input type="checkbox"
                   v-bind:checked="todoItem.done"
                   v-on:change="toggleTodoStatus()"/>
            <button v-on:click="removeTodo">지우기</button>
        </li>
    </div>
</template>

<script>
export default {
    name: 'TodoItem',
    props: {
        todoItem: {
            type: Object
        },
toggleTodoStatus () {
    const id = this.todoItem.id
    console.log('toggleTodoStatus() - id:' +id)

    this.$emit('toggleTodoStatus', id)
},
```

Getters의 역할 여러 컴포넌트에 동일한 computed가 만들어질경우 getters에서 속성을 정의한다

getters.js M

```
export default {
    filteredTodoItems (state) {
        if (!state.filter) {
            return state.todoItems
        }
    },
    getMonsterElements (state) {
        return state.monsterElements
    },
    ranodmFromSpring (state) {
        return state.ranodmFromSpring
    }
}
```