Vue Router로 다중 페이지 만들기 - 완전 가이드

메뉴를 추가하고 여러 페이지를 만들어 연습 예제들을 분리해봅시다!

목차

---

1. 프로젝트 준비

1-1. 새 프로젝트 만들기

터미널에서:

```
cd Desktop/vue-projects
npm create vue@latest
```

프로젝트 이름:

```
✔ Project name: › practice-app
```

질문에 답하기:

```
✔ Add TypeScript? › No
✔ Add JSX Support? › No
✔ Add Vue Router? › Yes  ← 이번엔 Yes! ⭐
✔ Add Pinia? › No
✔ Add Vitest? › No
✔ Add E2E Testing? › No
✔ Add ESLint? › No
✔ Add Prettier? › No
```

⭐ **Vue Router를 Yes로 선택하는 게 중요!**

설치:

```
cd practice-app
npm install
```

실행:

```
npm run dev
```

브라우저: http://localhost:5173/

1-2. 생성된 폴더 구조

```
practice-app/
├── src/
│   ├── assets/          # 이미지, CSS 파일
│   ├── components/      # 재사용 컴포넌트
│   ├── router/
│   │   └── index.js     # 라우터 설정 ⭐
│   ├── views/           # 페이지들 ⭐
│   │   ├── HomeView.vue
│   │   └── AboutView.vue
│   ├── App.vue          # 메인 앱
│   └── main.js
├── package.json
└── ...
```

---

2. Vue Router 설치

2-1. 이미 설치된 경우

프로젝트 생성 시 "Add Vue Router? Yes"를 선택했다면 **이미 설치됨!**

2-2. 기존 프로젝트에 추가하기

만약 이전에 만든 `my-first-vue-app`에 추가하고 싶다면:

Step 1: 터미널에서

```
cd my-first-vue-app
npm install vue-router@4
```

Step 2: 대기

```
added 1 package in 3s
```

---

3. 폴더 구조 만들기

3-1. views 폴더 확인

VS Code에서 왼쪽 사이드바를 보세요:

```
src/
└── views/  ← 이 폴더가 있나요?
```

3-2. views 폴더 만들기 (없다면)

방법 1: VS Code에서

1. `src` 폴더 우클릭
2. "새 폴더" 클릭
3. 이름: `views`

방법 2: 터미널에서

```
mkdir src/views
```

3-3. router 폴더 만들기 (없다면)

방법 1: VS Code에서

1. `src` 폴더 우클릭
2. "새 폴더" 클릭
3. 이름: `router`

방법 2: 터미널에서

```
mkdir src/router
```

---

4. 라우터 설정

4-1. 라우터 파일 만들기

파일 생성: `src/router/index.js`

코드:

```
import { createRouter, createWebHistory } from 'vue-router'
import HomeView from '../views/HomeView.vue'

const router = createRouter({
  history: createWebHistory(import.meta.env.BASE_URL),
  routes: [
    {
      path: '/',
      name: 'home',
      component: HomeView
    },
    {
      path: '/about',
      name: 'about',
      component: () => import('../views/AboutView.vue')
    }
  ]
})

export default router
```

저장!

이해하기: 라우터 구조

```
{
  path: '/counter',        // URL 주소
  name: 'counter',         // 이름 (선택사항)
  component: CounterView   // 보여줄 컴포넌트
}
```

예시:

- `path: '/'` → `localhost:5173/` (홈)
- `path: '/counter'` → `localhost:5173/counter`
- `path: '/about'` → `localhost:5173/about`

4-2. main.js 수정

파일 열기: `src/main.js`

코드:

```
import { createApp } from 'vue'
import App from './App.vue'
import router from './router'  // 라우터 추가 ⭐

const app = createApp(App)

app.use(router)  // 라우터 사용 ⭐
app.mount('#app')
```

저장!

4-3. App.vue 수정

파일 열기: `src/App.vue`

기존 코드를 모두 지우고:

```
<script setup>
// 여기는 비워둡니다
</script>

<template>
  <div id="app">
    <nav class="navbar">
      <div class="container">
        <h1 class="logo">🎓 연습 앱</h1>
        <div class="nav-links">
          <RouterLink to="/">홈</RouterLink>
          <RouterLink to="/about">소개</RouterLink>
        </div>
      </div>
    </nav>

    <main class="main-content">
      <RouterView />
    </main>
  </div>
</template>

<style scoped>
* {
  margin: 0;
  padding: 0;
  box-sizing: border-box;
}

#app {
```

```
  min-height: 100vh;
  background: #f5f7fa;
}

.navbar {
  background: linear-gradient(135deg, #667eea 0%, #764ba2 100%);
  box-shadow: 0 2px 10px rgba(0,0,0,0.1);
}

.container {
  max-width: 1200px;
  margin: 0 auto;
  padding: 0 20px;
  display: flex;
  justify-content: space-between;
  align-items: center;
  height: 70px;
}

.logo {
  color: white;
  font-size: 1.5em;
  font-weight: bold;
}

.nav-links {
  display: flex;
  gap: 20px;
}

.nav-links a {
  color: white;
  text-decoration: none;
  padding: 10px 20px;
  border-radius: 8px;
  font-weight: 500;
  transition: all 0.3s ease;
}

.nav-links a:hover {
  background: rgba(255,255,255,0.2);
}

.nav-links a.router-link-active {
  background: rgba(255,255,255,0.3);
  font-weight: bold;
}

.main-content {
  max-width: 1200px;
  margin: 0 auto;
  padding: 40px 20px;
}
</style>
```

저장!

이해하기: 특수 컴포넌트

`<RouterView />`

```
<RouterView />
```

- 현재 경로에 맞는 페이지를 여기에 표시
- 페이지가 바뀌면 내용이 바뀜

`<RouterLink>`

```
<RouterLink to="/counter">카운터</RouterLink>
```

- 페이지 이동 링크
- `<a>` 태그 대신 사용
- 페이지 새로고침 없이 이동!

---

5. 페이지 만들기

5-1. HomeView.vue (홈 페이지)

파일 생성: `src/views/HomeView.vue`

```
<script setup>
// 비워둡니다
</script>

<template>
  <div class="home">
    <h1>🏠 홈 페이지</h1>
    <p class="subtitle">연습 예제들을 선택하세요!</p>

    <div class="cards">
      <RouterLink to="/counter" class="card">
        <div class="card-icon">🔢</div>
        <h3>카운터</h3>
        <p>숫자를 증가/감소시켜보세요</p>
      </RouterLink>

      <RouterLink to="/todo" class="card">
        <div class="card-icon">☑️</div>
```

```
        <h3>할 일 목록</h3>
        <p>할 일을 관리해보세요</p>
      </RouterLink>

      <RouterLink to="/shopping" class="card">
        <div class="card-icon">🛒</div>
        <h3>쇼핑 리스트</h3>
        <p>장바구니를 만들어보세요</p>
      </RouterLink>

      <RouterLink to="/form" class="card">
        <div class="card-icon">📝</div>
        <h3>회원가입 폼</h3>
        <p>폼 입력을 연습해보세요</p>
      </RouterLink>
    </div>
  </div>
</template>

<style scoped>
.home {
  text-align: center;
}

h1 {
  font-size: 3em;
  color: #2c3e50;
  margin-bottom: 10px;
}

.subtitle {
  font-size: 1.2em;
  color: #7f8c8d;
  margin-bottom: 40px;
}

.cards {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
  gap: 30px;
  margin-top: 40px;
}

.card {
  background: white;
  padding: 40px 30px;
  border-radius: 15px;
  text-decoration: none;
  color: inherit;
  box-shadow: 0 4px 6px rgba(0,0,0,0.1);
  transition: all 0.3s ease;
  cursor: pointer;
}
```

```
.card:hover {
  transform: translateY(-5px);
  box-shadow: 0 8px 15px rgba(0,0,0,0.2);
}

.card-icon {
  font-size: 4em;
  margin-bottom: 20px;
}

.card h3 {
  color: #667eea;
  font-size: 1.5em;
  margin-bottom: 10px;
}

.card p {
  color: #7f8c8d;
  font-size: 1em;
}
</style>
```

저장!

5-2. AboutView.vue (소개 페이지)

파일 생성: `src/views/AboutView.vue`

```
<script setup>
// 비워둡니다
</script>

<template>
  <div class="about">
    <h1>📄 소개</h1>

    <div class="content">
      <section class="section">
        <h2>이 앱은 무엇인가요?</h2>
        <p>
          Vue.js를 배우기 위한 연습 앱입니다.
          HTML, CSS, JavaScript의 기초를 실습하면서 배울 수 있습니다.
        </p>
      </section>

      <section class="section">
        <h2>무엇을 배울 수 있나요?</h2>
        <ul>
          <li>✅ Vue.js 기본 개념</li>
          <li>✅ 반응형 데이터 (ref)</li>
          <li>✅ 조건부 렌더링 (v-if)</li>
```

```
            <li>☑ 리스트 렌더링 (v-for)</li>
            <li>☑ 이벤트 처리 (@click)</li>
            <li>☑ 양방향 바인딩 (v-model)</li>
            <li>☑ Vue Router로 페이지 이동</li>
          </ul>
        </section>

        <section class="section">
          <h2>예제 목록</h2>
          <div class="examples">
            <div class="example">
              <h3>🔢 카운터</h3>
              <p>숫자 증가/감소를 배웁니다</p>
            </div>
            <div class="example">
              <h3>☑ 할 일 목록</h3>
              <p>배열과 목록 관리를 배웁니다</p>
            </div>
            <div class="example">
              <h3>🛒 쇼핑 리스트</h3>
              <p>복잡한 데이터 구조를 다룹니다</p>
            </div>
            <div class="example">
              <h3>📝 회원가입 폼</h3>
              <p>다양한 입력 방법을 배웁니다</p>
            </div>
          </div>
        </section>
      </div>
    </div>
  </template>

  <style scoped>
  .about {
    max-width: 800px;
    margin: 0 auto;
  }

  h1 {
    font-size: 3em;
    color: #2c3e50;
    text-align: center;
    margin-bottom: 40px;
  }

  .content {
    background: white;
    padding: 40px;
    border-radius: 15px;
    box-shadow: 0 4px 6px rgba(0,0,0,0.1);
  }

  .section {
    margin-bottom: 40px;
```

```
}

.section:last-child {
  margin-bottom: 0;
}

h2 {
  color: #667eea;
  font-size: 1.8em;
  margin-bottom: 15px;
}

p {
  color: #555;
  font-size: 1.1em;
  line-height: 1.8;
}

ul {
  list-style: none;
  padding: 0;
}

ul li {
  color: #555;
  font-size: 1.1em;
  padding: 10px 0;
  border-bottom: 1px solid #ecf0f1;
}

ul li:last-child {
  border-bottom: none;
}

.examples {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
  gap: 20px;
  margin-top: 20px;
}

.example {
  background: #f8f9fa;
  padding: 20px;
  border-radius: 10px;
  border-left: 4px solid #667eea;
}

.example h3 {
  color: #2c3e50;
  margin-bottom: 8px;
}

.example p {
```

```
  font-size: 0.95em;
  color: #7f8c8d;
}
</style>
```

저장!

5-3. CounterView.vue (카운터 페이지)

파일 생성: `src/views/CounterView.vue`

```
<script setup>
import { ref } from 'vue'

const count = ref(0)

function increment() {
  count.value++
}

function decrement() {
  count.value--
}

function reset() {
  count.value = 0
}

function incrementBy(amount) {
  count.value += amount
}
</script>

<template>
  <div class="counter-page">
    <h1>🔢 카운터</h1>

    <div class="counter-box">
      <div class="count-display">{{ count }}</div>

      <div class="button-group">
        <button @click="decrement" class="btn btn-secondary">
          -1
        </button>
        <button @click="reset" class="btn btn-danger">
          리셋
        </button>
        <button @click="increment" class="btn btn-primary">
          +1
        </button>
      </div>
```

```
      <div class="button-group">
        <button @click="incrementBy(5)" class="btn btn-info">
          +5
        </button>
        <button @click="incrementBy(10)" class="btn btn-info">
          +10
        </button>
        <button @click="incrementBy(100)" class="btn btn-info">
          +100
        </button>
      </div>
    </div>

    <div class="info-box">
      <h3>📖 학습 내용</h3>
      <ul>
        <li><code>ref()</code>로 반응형 변수 만들기</li>
        <li><code>@click</code>으로 이벤트 처리하기</li>
        <li><code>{{ }}</code>로 데이터 표시하기</li>
        <li>함수로 로직 분리하기</li>
      </ul>
    </div>
  </div>
</template>

<style scoped>
.counter-page {
  max-width: 600px;
  margin: 0 auto;
}

h1 {
  text-align: center;
  font-size: 2.5em;
  color: #2c3e50;
  margin-bottom: 30px;
}

.counter-box {
  background: white;
  padding: 50px;
  border-radius: 15px;
  box-shadow: 0 4px 6px rgba(0,0,0,0.1);
  text-align: center;
}

.count-display {
  font-size: 5em;
  font-weight: bold;
  color: #667eea;
  margin-bottom: 30px;
  text-shadow: 2px 2px 4px rgba(0,0,0,0.1);
}
```

```
.button-group {
  display: flex;
  gap: 15px;
  justify-content: center;
  margin-bottom: 15px;
}

.button-group:last-of-type {
  margin-bottom: 0;
}

.btn {
  padding: 15px 30px;
  font-size: 18px;
  font-weight: bold;
  border: none;
  border-radius: 10px;
  cursor: pointer;
  transition: all 0.3s ease;
  color: white;
}

.btn:hover {
  transform: translateY(-2px);
  box-shadow: 0 4px 8px rgba(0,0,0,0.2);
}

.btn:active {
  transform: translateY(0);
}

.btn-primary {
  background: #667eea;
}

.btn-secondary {
  background: #95a5a6;
}

.btn-danger {
  background: #e74c3c;
}

.btn-info {
  background: #3498db;
}

.info-box {
  background: #f8f9fa;
  padding: 30px;
  border-radius: 15px;
  margin-top: 30px;
}
```

```
.info-box h3 {
  color: #2c3e50;
  margin-bottom: 15px;
}

.info-box ul {
  list-style: none;
  padding: 0;
}

.info-box li {
  padding: 10px 0;
  color: #555;
  font-size: 1.05em;
}

.info-box code {
  background: #e8e8e8;
  padding: 2px 8px;
  border-radius: 4px;
  font-family: 'Courier New', monospace;
  color: #e74c3c;
}
</style>
```

저장!

5-4. TodoView.vue (할 일 목록 페이지)

파일 생성: `src/views/TodoView.vue`

```
<script setup>
import { ref, computed } from 'vue'

const todos = ref([
  { id: 1, text: '프로그래밍 공부하기', done: false },
  { id: 2, text: '운동하기', done: false },
  { id: 3, text: '책 읽기', done: true }
])

const newTodo = ref('')

function addTodo() {
  if (newTodo.value.trim() !== '') {
    const newId = Math.max(...todos.value.map(t => t.id), 0) + 1
    todos.value.push({
      id: newId,
      text: newTodo.value,
      done: false
    })
```

```
    newTodo.value = ''
  }
}

function removeTodo(id) {
  const index = todos.value.findIndex(t => t.id === id)
  if (index !== -1) {
    todos.value.splice(index, 1)
  }
}

function toggleDone(id) {
  const todo = todos.value.find(t => t.id === id)
  if (todo) {
    todo.done = !todo.done
  }
}

const totalCount = computed(() => todos.value.length)
const doneCount = computed(() => todos.value.filter(t => t.done).length)
const pendingCount = computed(() => todos.value.filter(t => !t.done).length)
</script>

<template>
  <div class="todo-page">
    <h1>☑ 할 일 목록</h1>

    <div class="input-section">
      <input
        v-model="newTodo"
        @keyup.enter="addTodo"
        placeholder="새 할 일을 입력하세요"
        class="todo-input"
      >
      <button @click="addTodo" class="btn-add">추가</button>
    </div>

    <div class="stats">
      <div class="stat">
        <div class="stat-number">{{ totalCount }}</div>
        <div class="stat-label">전체</div>
      </div>
      <div class="stat">
        <div class="stat-number">{{ pendingCount }}</div>
        <div class="stat-label">남음</div>
      </div>
      <div class="stat">
        <div class="stat-number">{{ doneCount }}</div>
        <div class="stat-label">완료</div>
      </div>
    </div>

    <div class="todo-list">
      <div
```

```
        v-for="todo in todos"
        :key="todo.id"
        class="todo-item"
        :class="{ done: todo.done }"
      >
        <input
          type="checkbox"
          :checked="todo.done"
          @change="toggleDone(todo.id)"
          class="checkbox"
        >
        <span class="todo-text">{{ todo.text }}</span>
        <button @click="removeTodo(todo.id)" class="btn-delete">
          🗑️
        </button>
      </div>

      <div v-if="todos.length === 0" class="empty">
        할 일이 없습니다! 🎉
      </div>
    </div>

    <div class="info-box">
      <h3>📖 학습 내용</h3>
      <ul>
        <li><code>v-for</code>로 목록 렌더링하기</li>
        <li><code>v-model</code>로 양방향 바인딩하기</li>
        <li><code>computed</code>로 계산된 값 만들기</li>
        <li>배열 메서드 (<code>push</code>, <code>splice</code>, <code>find</code>)
</li>
      </ul>
    </div>
  </div>
</template>

<style scoped>
.todo-page {
  max-width: 700px;
  margin: 0 auto;
}

h1 {
  text-align: center;
  font-size: 2.5em;
  color: #2c3e50;
  margin-bottom: 30px;
}

.input-section {
  display: flex;
  gap: 10px;
  margin-bottom: 30px;
}
```

```
.todo-input {
  flex: 1;
  padding: 15px;
  font-size: 16px;
  border: 2px solid #ddd;
  border-radius: 10px;
  outline: none;
}

.todo-input:focus {
  border-color: #667eea;
}

.btn-add {
  padding: 15px 30px;
  background: #667eea;
  color: white;
  border: none;
  border-radius: 10px;
  font-size: 16px;
  font-weight: bold;
  cursor: pointer;
}

.btn-add:hover {
  background: #5568d3;
}

.stats {
  display: flex;
  gap: 20px;
  margin-bottom: 30px;
}

.stat {
  flex: 1;
  background: white;
  padding: 20px;
  border-radius: 10px;
  text-align: center;
  box-shadow: 0 2px 4px rgba(0,0,0,0.1);
}

.stat-number {
  font-size: 2.5em;
  font-weight: bold;
  color: #667eea;
}

.stat-label {
  color: #7f8c8d;
  margin-top: 5px;
}
```

```
.todo-list {
  background: white;
  border-radius: 10px;
  padding: 20px;
  box-shadow: 0 2px 4px rgba(0,0,0,0.1);
}

.todo-item {
  display: flex;
  align-items: center;
  padding: 15px;
  border-bottom: 1px solid #ecf0f1;
  transition: background 0.3s ease;
}

.todo-item:last-child {
  border-bottom: none;
}

.todo-item:hover {
  background: #f8f9fa;
}

.todo-item.done .todo-text {
  text-decoration: line-through;
  color: #95a5a6;
}

.checkbox {
  width: 20px;
  height: 20px;
  cursor: pointer;
  margin-right: 15px;
}

.todo-text {
  flex: 1;
  font-size: 1.1em;
}

.btn-delete {
  background: #ffe6e6;
  border: none;
  padding: 8px 12px;
  border-radius: 6px;
  cursor: pointer;
  font-size: 1.2em;
}

.btn-delete:hover {
  background: #ffcccc;
}

.empty {
```

```
  text-align: center;
  padding: 40px;
  color: #95a5a6;
  font-size: 1.2em;
}

.info-box {
  background: #f8f9fa;
  padding: 30px;
  border-radius: 15px;
  margin-top: 30px;
}

.info-box h3 {
  color: #2c3e50;
  margin-bottom: 15px;
}

.info-box ul {
  list-style: none;
  padding: 0;
}

.info-box li {
  padding: 10px 0;
  color: #555;
  font-size: 1.05em;
}

.info-box code {
  background: #e8e8e8;
  padding: 2px 8px;
  border-radius: 4px;
  font-family: 'Courier New', monospace;
  color: #e74c3c;
}
</style>
```

저장!

5-5. 라우터에 페이지 추가

파일 열기: `src/router/index.js`

수정:

```
import { createRouter, createWebHistory } from 'vue-router'
import HomeView from '../views/HomeView.vue'

const router = createRouter({
```

```
  history: createWebHistory(import.meta.env.BASE_URL),
  routes: [
    {
      path: '/',
      name: 'home',
      component: HomeView
    },
    {
      path: '/about',
      name: 'about',
      component: () => import('../views/AboutView.vue')
    },
    {
      path: '/counter',
      name: 'counter',
      component: () => import('../views/CounterView.vue')
    },
    {
      path: '/todo',
      name: 'todo',
      component: () => import('../views/TodoView.vue')
    }
  ]
})

export default router
```

저장!

---

6. 메뉴 네비게이션 만들기

6-1. App.vue 메뉴 확장

파일 열기: `src/App.vue`

`<template>` **섹션의 nav 부분 수정:**

```
<nav class="navbar">
  <div class="container">
    <RouterLink to="/" class="logo">🎓 연습 앱</RouterLink>
    <div class="nav-links">
      <RouterLink to="/">홈</RouterLink>
      <RouterLink to="/counter">카운터</RouterLink>
      <RouterLink to="/todo">할 일 목록</RouterLink>
      <RouterLink to="/about">소개</RouterLink>
    </div>
  </div>
</nav>
```

6-2. 로고를 클릭 가능하게

스타일 추가:

```
<style scoped>
/* ... 기존 스타일 ... */

.logo {
  color: white;
  font-size: 1.5em;
  font-weight: bold;
  text-decoration: none;  /* 추가 */
  transition: opacity 0.3s ease;  /* 추가 */
}

.logo:hover {
  opacity: 0.8;  /* 추가 */
}

/* ... 나머지 스타일 ... */
</style>
```

저장!

6-3. 모바일 반응형 메뉴 (선택사항)

고급 스타일:

```
<style scoped>
/* ... 기존 스타일 ... */

@media (max-width: 768px) {
  .container {
    flex-direction: column;
    height: auto;
    padding: 15px 20px;
  }

  .nav-links {
    margin-top: 15px;
    width: 100%;
    justify-content: center;
  }

  .nav-links a {
    padding: 8px 15px;
    font-size: 14px;
  }
```

```
}
</style>
```

---

7. 예제 페이지 완성

7-1. ShoppingView.vue (쇼핑 리스트)

파일 생성: `src/views/ShoppingView.vue`

```
<script setup>
import { ref, computed } from 'vue'

const items = ref([
  { id: 1, name: '사과', price: 3000, quantity: 0 },
  { id: 2, name: '바나나', price: 2000, quantity: 0 },
  { id: 3, name: '우유', price: 2500, quantity: 0 },
  { id: 4, name: '빵', price: 1500, quantity: 0 }
])

function increaseQuantity(id) {
  const item = items.value.find(item => item.id === id)
  if (item) item.quantity++
}

function decreaseQuantity(id) {
  const item = items.value.find(item => item.id === id)
  if (item && item.quantity > 0) item.quantity--
}

const totalItems = computed(() => {
  return items.value.reduce((sum, item) => sum + item.quantity, 0)
})

const totalPrice = computed(() => {
  return items.value.reduce((sum, item) => sum + (item.price * item.quantity), 0)
})

const cartItems = computed(() => {
  return items.value.filter(item => item.quantity > 0)
})

function resetCart() {
  items.value.forEach(item => item.quantity = 0)
}
</script>

<template>
  <div class="shopping-page">
    <h1>🛒 쇼핑 리스트</h1>
```

```
      <div class="items-grid">
        <div
          v-for="item in items"
          :key="item.id"
          class="item-card"
          :class="{ selected: item.quantity > 0 }"
        >
          <h3>{{ item.name }}</h3>
          <p class="price">{{ item.price.toLocaleString() }}원</p>
          <div class="controls">
            <button
              @click="decreaseQuantity(item.id)"
              :disabled="item.quantity === 0"
              class="btn-qty"
            >
              -
            </button>
            <span class="quantity">{{ item.quantity }}</span>
            <button
              @click="increaseQuantity(item.id)"
              class="btn-qty"
            >
              +
            </button>
          </div>
        </div>
      </div>

      <div class="cart-section">
        <h2>장바구니</h2>

        <div v-if="cartItems.length === 0" class="empty-cart">
          장바구니가 비어있습니다
        </div>

        <div v-else>
          <div class="cart-items">
            <div v-for="item in cartItems" :key="item.id" class="cart-item">
              <span>{{ item.name }} x {{ item.quantity }}</span>
              <span>{{ (item.price * item.quantity).toLocaleString() }}원</span>
            </div>
          </div>

          <div class="cart-summary">
            <div>총 상품: {{ totalItems }}개</div>
            <div class="total">총 금액: {{ totalPrice.toLocaleString() }}원</div>
          </div>

          <button @click="resetCart" class="btn-reset">
            장바구니 비우기
          </button>
        </div>
      </div>
```

```
    <div class="info-box">
      <h3>📖 학습 내용</h3>
      <ul>
        <li><code>computed</code>로 자동 계산하기</li>
        <li><code>filter</code>, <code>reduce</code> 배열 메서드</li>
        <li><code>:class</code>로 동적 클래스 바인딩</li>
        <li><code>:disabled</code>로 버튼 비활성화</li>
      </ul>
    </div>
  </div>
</template>

<style scoped>
.shopping-page {
  max-width: 1000px;
  margin: 0 auto;
}

h1 {
  text-align: center;
  font-size: 2.5em;
  color: #2c3e50;
  margin-bottom: 30px;
}

.items-grid {
  display: grid;
  grid-template-columns: repeat(auto-fill, minmax(200px, 1fr));
  gap: 20px;
  margin-bottom: 40px;
}

.item-card {
  background: white;
  padding: 25px;
  border-radius: 10px;
  text-align: center;
  border: 2px solid transparent;
  transition: all 0.3s ease;
}

.item-card:hover {
  transform: translateY(-2px);
  box-shadow: 0 4px 8px rgba(0,0,0,0.1);
}

.item-card.selected {
  border-color: #667eea;
  background: #f0f3ff;
}

.item-card h3 {
  color: #2c3e50;
  margin-bottom: 10px;
```

```
}

.price {
  color: #e74c3c;
  font-size: 1.3em;
  font-weight: bold;
  margin-bottom: 15px;
}

.controls {
  display: flex;
  justify-content: center;
  align-items: center;
  gap: 10px;
}

.btn-qty {
  width: 40px;
  height: 40px;
  border: 2px solid #667eea;
  background: white;
  color: #667eea;
  border-radius: 8px;
  font-size: 20px;
  font-weight: bold;
  cursor: pointer;
}

.btn-qty:hover:not(:disabled) {
  background: #667eea;
  color: white;
}

.btn-qty:disabled {
  opacity: 0.3;
  cursor: not-allowed;
}

.quantity {
  min-width: 30px;
  font-size: 1.5em;
  font-weight: bold;
  color: #2c3e50;
}

.cart-section {
  background: white;
  padding: 30px;
  border-radius: 10px;
  box-shadow: 0 2px 4px rgba(0,0,0,0.1);
}

.cart-section h2 {
  color: #2c3e50;
```

```
  margin-bottom: 20px;
}

.empty-cart {
  text-align: center;
  padding: 40px;
  color: #95a5a6;
}

.cart-items {
  margin-bottom: 20px;
}

.cart-item {
  display: flex;
  justify-content: space-between;
  padding: 15px;
  background: #f8f9fa;
  border-radius: 8px;
  margin-bottom: 10px;
}

.cart-summary {
  background: #ecf0f1;
  padding: 20px;
  border-radius: 8px;
  margin-bottom: 20px;
}

.cart-summary div {
  margin-bottom: 10px;
  font-size: 1.1em;
}

.cart-summary .total {
  font-size: 1.5em;
  font-weight: bold;
  color: #e74c3c;
  margin-top: 10px;
  padding-top: 10px;
  border-top: 2px solid #bdc3c7;
}

.btn-reset {
  width: 100%;
  padding: 15px;
  background: #95a5a6;
  color: white;
  border: none;
  border-radius: 8px;
  font-size: 16px;
  font-weight: bold;
  cursor: pointer;
}
```

```
.btn-reset:hover {
  background: #7f8c8d;
}

.info-box {
  background: #f8f9fa;
  padding: 30px;
  border-radius: 15px;
  margin-top: 30px;
}

.info-box h3 {
  color: #2c3e50;
  margin-bottom: 15px;
}

.info-box ul {
  list-style: none;
  padding: 0;
}

.info-box li {
  padding: 10px 0;
  color: #555;
  font-size: 1.05em;
}

.info-box code {
  background: #e8e8e8;
  padding: 2px 8px;
  border-radius: 4px;
  font-family: 'Courier New', monospace;
  color: #e74c3c;
}
</style>
```

7-2. FormView.vue (회원가입 폼)

파일 생성: `src/views/FormView.vue`

```
<script setup>
import { ref } from 'vue'

const name = ref('')
const email = ref('')
const age = ref(0)
const gender = ref('male')
const hobbies = ref([])
const bio = ref('')
```

```
const agree = ref(false)

function submitForm() {
  if (!agree.value) {
    alert('약관에 동의해주세요!')
    return
  }

  if (name.value.trim() === '' || email.value.trim() === '') {
    alert('이름과 이메일을 입력해주세요!')
    return
  }

  console.log({
    이름: name.value,
    이메일: email.value,
    나이: age.value,
    성별: gender.value,
    취미: hobbies.value,
    자기소개: bio.value
  })

  alert('가입이 완료되었습니다!')
  resetForm()
}

function resetForm() {
  name.value = ''
  email.value = ''
  age.value = 0
  gender.value = 'male'
  hobbies.value = []
  bio.value = ''
  agree.value = false
}
</script>

<template>
  <div class="form-page">
    <h1>📝 회원가입 폼</h1>

    <div class="form-container">
      <div class="form-group">
        <label>이름 *</label>
        <input v-model="name" type="text" placeholder="이름을 입력하세요">
      </div>

      <div class="form-group">
        <label>이메일 *</label>
        <input v-model="email" type="email" placeholder="email@example.com">
      </div>

      <div class="form-group">
        <label>나이</label>
```

```
          <input v-model.number="age" type="number" min="0" max="120">
        </div>

        <div class="form-group">
          <label>성별</label>
          <div class="radio-group">
            <label class="radio-label">
              <input v-model="gender" type="radio" value="male">
              남성
            </label>
            <label class="radio-label">
              <input v-model="gender" type="radio" value="female">
              여성
            </label>
          </div>
        </div>

        <div class="form-group">
          <label>취미</label>
          <div class="checkbox-group">
            <label class="checkbox-label">
              <input v-model="hobbies" type="checkbox" value="게임">
              게임
            </label>
            <label class="checkbox-label">
              <input v-model="hobbies" type="checkbox" value="독서">
              독서
            </label>
            <label class="checkbox-label">
              <input v-model="hobbies" type="checkbox" value="운동">
              운동
            </label>
            <label class="checkbox-label">
              <input v-model="hobbies" type="checkbox" value="음악">
              음악
            </label>
          </div>
        </div>

        <div class="form-group">
          <label>자기소개</label>
          <textarea v-model="bio" rows="5" placeholder="자기소개를 작성하세요">
</textarea>
        </div>

        <div class="form-group">
          <label class="checkbox-label">
            <input v-model="agree" type="checkbox">
            이용약관에 동의합니다 *
          </label>
        </div>

        <div class="button-group">
          <button @click="resetForm" class="btn-reset">초기화</button>
```

```
        <button @click="submitForm" :disabled="!agree" class="btn-submit">
          가입하기
        </button>
      </div>
    </div>

    <div class="result-section">
      <h2>입력된 정보</h2>
      <div class="result-grid">
        <div class="result-item">
          <strong>이름:</strong> {{ name || '-' }}
        </div>
        <div class="result-item">
          <strong>이메일:</strong> {{ email || '-' }}
        </div>
        <div class="result-item">
          <strong>나이:</strong> {{ age }}세
        </div>
        <div class="result-item">
          <strong>성별:</strong> {{ gender === 'male' ? '남성' : '여성' }}
        </div>
        <div class="result-item">
          <strong>취미:</strong> {{ hobbies.join(', ') || '없음' }}
        </div>
        <div class="result-item full-width">
          <strong>자기소개:</strong><br>{{ bio || '-' }}
        </div>
      </div>
    </div>

    <div class="info-box">
      <h3>📖 학습 내용</h3>
      <ul>
        <li><code>v-model</code>로 다양한 입력 받기</li>
        <li>텍스트, 숫자, 이메일, 라디오, 체크박스, 텍스트영역</li>
        <li><code>:disabled</code>로 조건부 버튼 활성화</li>
        <li>폼 유효성 검사하기</li>
      </ul>
    </div>
  </div>
</template>

<style scoped>
.form-page {
  max-width: 700px;
  margin: 0 auto;
}

h1 {
  text-align: center;
  font-size: 2.5em;
  color: #2c3e50;
  margin-bottom: 30px;
}
```

```
.form-container {
  background: white;
  padding: 40px;
  border-radius: 15px;
  box-shadow: 0 4px 6px rgba(0,0,0,0.1);
  margin-bottom: 30px;
}

.form-group {
  margin-bottom: 25px;
}

.form-group label {
  display: block;
  margin-bottom: 8px;
  font-weight: bold;
  color: #2c3e50;
}

.form-group input[type="text"],
.form-group input[type="email"],
.form-group input[type="number"],
.form-group textarea {
  width: 100%;
  padding: 12px;
  font-size: 16px;
  border: 2px solid #ddd;
  border-radius: 8px;
  box-sizing: border-box;
}

.form-group input:focus,
.form-group textarea:focus {
  outline: none;
  border-color: #667eea;
}

.radio-group,
.checkbox-group {
  display: flex;
  gap: 20px;
  flex-wrap: wrap;
}

.radio-label,
.checkbox-label {
  display: flex;
  align-items: center;
  cursor: pointer;
  font-weight: normal !important;
}

.radio-label input,
```

```
.checkbox-label input {
  margin-right: 8px;
  cursor: pointer;
}

.button-group {
  display: flex;
  gap: 15px;
  margin-top: 30px;
}

.btn-reset,
.btn-submit {
  flex: 1;
  padding: 15px;
  border: none;
  border-radius: 8px;
  font-size: 16px;
  font-weight: bold;
  cursor: pointer;
}

.btn-reset {
  background: #95a5a6;
  color: white;
}

.btn-reset:hover {
  background: #7f8c8d;
}

.btn-submit {
  background: #667eea;
  color: white;
}

.btn-submit:hover:not(:disabled) {
  background: #5568d3;
}

.btn-submit:disabled {
  opacity: 0.5;
  cursor: not-allowed;
}

.result-section {
  background: white;
  padding: 30px;
  border-radius: 15px;
  box-shadow: 0 4px 6px rgba(0,0,0,0.1);
  margin-bottom: 30px;
}

.result-section h2 {
```

```
  color: #2c3e50;
  margin-bottom: 20px;
}

.result-grid {
  display: grid;
  grid-template-columns: repeat(2, 1fr);
  gap: 15px;
}

.result-item {
  padding: 15px;
  background: #f8f9fa;
  border-radius: 8px;
}

.result-item.full-width {
  grid-column: 1 / -1;
}

.result-item strong {
  color: #667eea;
}

.info-box {
  background: #f8f9fa;
  padding: 30px;
  border-radius: 15px;
}

.info-box h3 {
  color: #2c3e50;
  margin-bottom: 15px;
}

.info-box ul {
  list-style: none;
  padding: 0;
}

.info-box li {
  padding: 10px 0;
  color: #555;
  font-size: 1.05em;
}

.info-box code {
  background: #e8e8e8;
  padding: 2px 8px;
  border-radius: 4px;
  font-family: 'Courier New', monospace;
  color: #e74c3c;
}
</style>
```

7-3. 라우터에 모든 페이지 추가

파일 열기: src/router/index.js

최종 코드:

```
import { createRouter, createWebHistory } from 'vue-router'
import HomeView from '../views/HomeView.vue'

const router = createRouter({
  history: createWebHistory(import.meta.env.BASE_URL),
  routes: [
    {
      path: '/',
      name: 'home',
      component: HomeView
    },
    {
      path: '/about',
      name: 'about',
      component: () => import('../views/AboutView.vue')
    },
    {
      path: '/counter',
      name: 'counter',
      component: () => import('../views/CounterView.vue')
    },
    {
      path: '/todo',
      name: 'todo',
      component: () => import('../views/TodoView.vue')
    },
    {
      path: '/shopping',
      name: 'shopping',
      component: () => import('../views/ShoppingView.vue')
    },
    {
      path: '/form',
      name: 'form',
      component: () => import('../views/FormView.vue')
    }
  ]
})

export default router
```

7-4. App.vue 메뉴 최종 버전

파일 열기: `src/App.vue`

최종 `<template>`:

```
<template>
  <div id="app">
    <nav class="navbar">
      <div class="container">
        <RouterLink to="/" class="logo">🎓 연습 앱</RouterLink>
        <div class="nav-links">
          <RouterLink to="/">홈</RouterLink>
          <RouterLink to="/counter">카운터</RouterLink>
          <RouterLink to="/todo">할 일</RouterLink>
          <RouterLink to="/shopping">쇼핑</RouterLink>
          <RouterLink to="/form">폼</RouterLink>
          <RouterLink to="/about">소개</RouterLink>
        </div>
      </div>
    </nav>

    <main class="main-content">
      <RouterView />
    </main>

    <footer class="footer">
      <p>Vue.js 연습 앱 © 2025</p>
    </footer>
  </div>
</template>
```

푸터 스타일 추가:

```
<style scoped>
/* ... 기존 스타일 ... */

.footer {
  background: #2c3e50;
  color: white;
  text-align: center;
  padding: 20px;
  margin-top: 60px;
}

.footer p {
  margin: 0;
}
</style>
```

```
```

저장!

---

8. 문제 해결

8-1. 페이지가 안 보여요

증상:

```
404 Not Found
또는
빈 화면
```

해결 방법:

1. 라우터 파일 확인

```
// src/router/index.js
import HomeView from '../views/HomeView.vue'  // 경로 확인!
```

2. 파일 이름 확인

```
views/CounterView.vue  ← 대문자 V!
```

3. 서버 재시작

```
Ctrl + C  (종료)
npm run dev  (재시작)
```

8-2. 메뉴가 작동하지 않아요

증상:

```
클릭해도 페이지가 안 바뀜
```

해결 방법:

1. RouterLink 확인

```
<!-- 올바른 방법 -->
<RouterLink to="/counter">카운터</RouterLink>

<!-- 잘못된 방법 -->
<a href="/counter">카운터</a>  ← 이렇게 하지 마세요!
```

2. RouterView 확인

```
<!-- App.vue에 있어야 함 -->
<RouterView />
```

8-3. router-link-active 스타일이 안 먹혀요

해결 방법:

```
<style scoped>
/* 정확한 클래스 이름 */
.nav-links a.router-link-active {
  background: rgba(255,255,255,0.3);
}

/* 또는 exact 사용 */
.nav-links a.router-link-exact-active {
  background: rgba(255,255,255,0.3);
}
</style>
```

8-4. import 오류

증상:

```
Failed to resolve import
```

해결 방법:

1. 경로 확인

```
// 상대 경로 사용
import HomeView from '../views/HomeView.vue'
```

```
// @ 사용 (설정된 경우)
import HomeView from '@/views/HomeView.vue'
```

2. 파일 존재 확인

```
src/views/HomeView.vue  ← 이 파일이 있나요?
```

8-5. 콘솔 경고

증상:

```
[Vue Router warn]: No match found for location with path "/counter"
```

해결 방법:

라우터에 경로가 등록되어 있는지 확인:

```
{
  path: '/counter',  // 슬래시 있어야 함!
  name: 'counter',
  component: () => import('../views/CounterView.vue')
}
```

---

9. 마무리

🎉 완성!

당신은 이제:

```
☑ Vue Router를 설치할 수 있습니다
☑ 여러 페이지를 만들 수 있습니다
☑ 라우터 설정을 할 수 있습니다
☑ 네비게이션 메뉴를 만들 수 있습니다
☑ 페이지 간 이동을 구현할 수 있습니다
☑ 실용적인 다중 페이지 앱을 만들 수 있습니다
```

📚 배운 내용 정리

Vue Router 핵심 개념

```
<!-- 1. RouterLink: 링크 -->
<RouterLink to="/counter">카운터</RouterLink>

<!-- 2. RouterView: 페이지 표시 영역 -->
<RouterView />

<!-- 3. router-link-active: 현재 페이지 스타일 -->
.router-link-active { }
```

라우터 설정

```
{
  path: '/counter',        // URL
  name: 'counter',         // 이름
  component: CounterView   // 컴포넌트
}
```

다음 단계

1. 더 많은 기능 추가하기

- 페이지 전환 애니메이션
- 중첩 라우트
- 동적 라우트 (예: `/user/:id`)
- 네비게이션 가드

2. 태양계 프로젝트

- Three.js 배우기
- 3D 그래픽 구현
- 실전 프로젝트 완성

3. 스타일 개선

- 각 페이지마다 다른 테마
- 다크 모드 추가
- 애니메이션 효과

최종 체크리스트

```
☑ Vue Router가 설치되었다
☑ router/index.js가 설정되었다
☑ views 폴더에 페이지들이 있다
☑ App.vue에 네비게이션 메뉴가 있다
☑ RouterView가 작동한다
```

☑ 모든 링크가 작동한다
☑ 각 페이지가 제대로 표시된다

🗀 최종 폴더 구조

```
practice-app/
├── src/
│   ├── assets/
│   ├── components/
│   ├── router/
│   │   └── index.js            ☑ 라우터 설정
│   ├── views/
│   │   ├── HomeView.vue        ☑ 홈
│   │   ├── AboutView.vue       ☑ 소개
│   │   ├── CounterView.vue     ☑ 카운터
│   │   ├── TodoView.vue        ☑ 할 일 목록
│   │   ├── ShoppingView.vue    ☑ 쇼핑 리스트
│   │   └── FormView.vue        ☑ 회원가입 폼
│   ├── App.vue                 ☑ 메인 (메뉴 포함)
│   └── main.js                 ☑ 라우터 연결
├── package.json
└── ...
```

축하합니다!

다중 페이지 Vue 앱을 완성했습니다!

이제 각 예제를 독립적으로 연습할 수 있고, 새로운 페이지를 쉽게 추가할 수 있습니다!

계속 실험하고 배워나가세요! 🚀

만든 날짜: 2025년 10월 **대상:** Vue.js 입문자 **소요 시간:** 1-2시간 **다음 단계:** Three.js 배우기!