프론트엔드 개발자를 위한 Vue.js 시작하기

Vue.js 시작하기

Vuex

한국기술교육대학교
온라인평생교육원

학습내용

- State Management
- Vuex Overview

학습목표

- 상태 관리에 대한 개념을 설명할 수 있다.
- Vuex의 기초 사용 방법을 설명할 수 있다.

State Management

State Management

1 상태(State)란

상태(State)의 정의

- 애플리케이션의 데이터
- Component 내에서만 유지되고 관리되어야 하는 데이터를 상태라 표현하지는 않음
- 애플리케이션 전역에서 이용되는 데이터

State Management

2 상태 관리(State Management)란

상태 관리(State Management)의 특징

- 여러 Component로 구성된 애플리케이션에서는 하나의 상태가 하나의 Component에서만 이용되지 않음
- 애플리케이션 관점에서 상태를 **체계적으로 유지 관리**해 주어야 함
- 애플리케이션을 상태 중심으로 설계 개발함으로 **애플리케이션의 데이터와 데이터의 흐름을 명료**하게 할 수 있음

State Management

State Management

3 MVC 구조에서의 데이터 흐름

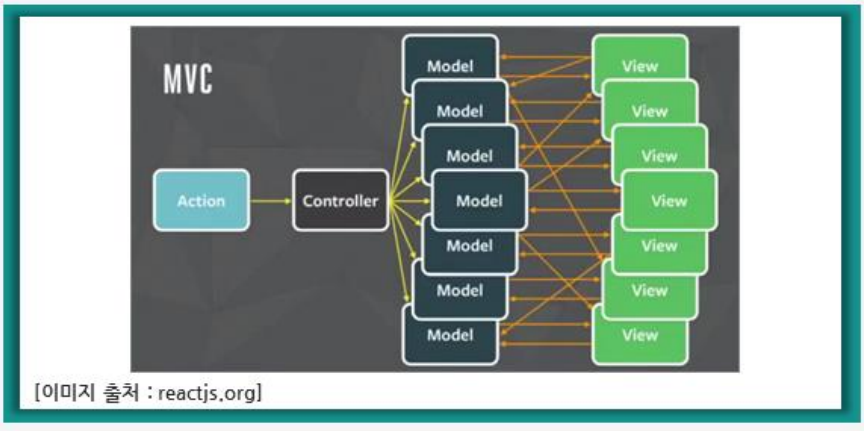
- MVC 모델은 기능의 분리가 초점임
- 애플리케이션의 상태를 중심으로 한 흐름이 아님

State Management

3 MVC 구조에서의 데이터 흐름

- 애플리케이션의 데이터가 양방향으로 이용되어 복잡도가 증가됨

MVC 구조



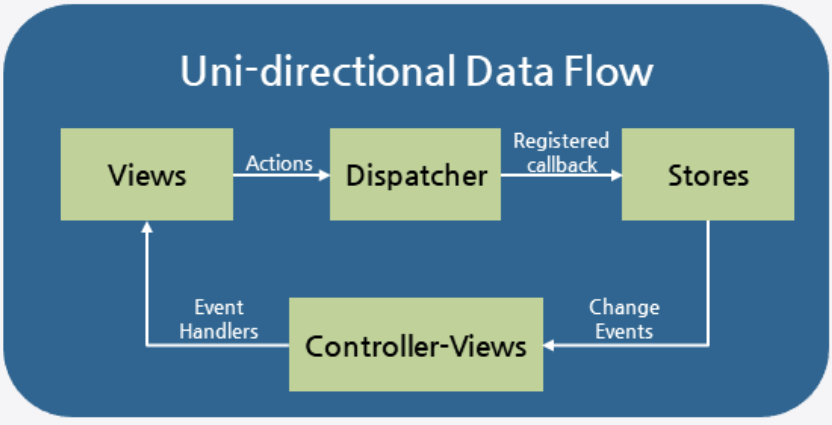
[이미지 출처 : reactjs.org]

State Management

State Management

4 상태 관리 패턴의 데이터 흐름

- 애플리케이션의 데이터는 **단방향으로만 이용**되어야 함

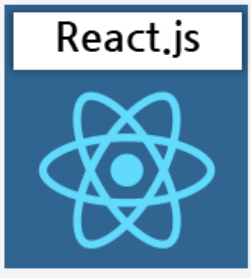


[이미지 출처 : reactjs.org]

State Management

5 상태 관리 프레임워크

NGXS



Redux, Context

Vuex

Vuex Overview

Vuex Overview

1 Vuex

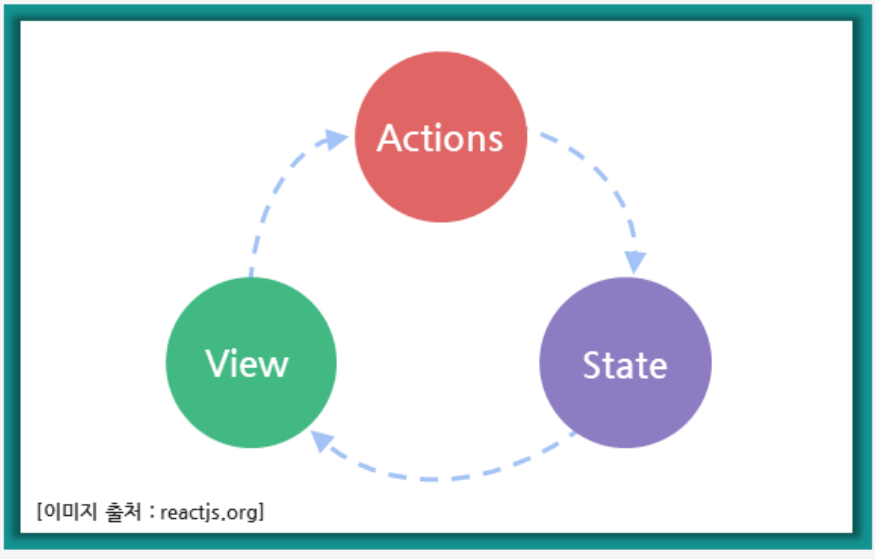
- Vue.js를 위한 상태 관리 프레임워크

Vuex4	Vue.js 3 버전을 위한 프레임워크
Vuex3	Vue.js 2 버전을 위한 프레임워크

```
>npm install vuex@next
```

Vuex Overview

2 One way data flow



Vuex Overview

Vuex Overview

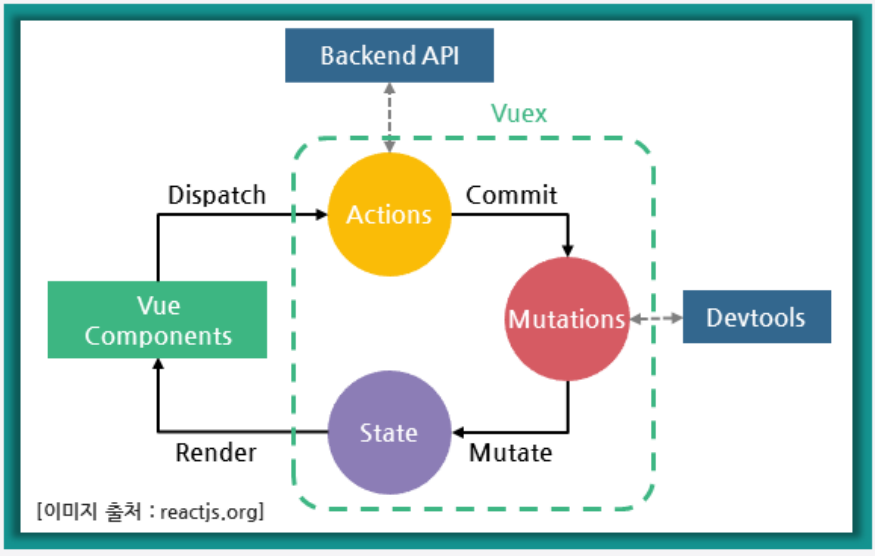
2 One way data flow

state	애플리케이션에서 관리되어야 하는 데이터
view	state가 이용되는 Component
action	state 발생 및 변경을 위한 이벤트

Vuex Overview

3 Vuex의 Flow

- mutation : state 변경



Vuex Overview

Vuex Overview

4 Store Instance

- Store Instance는 state, actions, mutations로 구성됨
- actions과 mutations는 함수로 작성됨
- state는 Object Literal임

Vuex Overview

4 Store Instance

```
const store = createStore({
    state() {
        return {
            items: []
        }
    },
    actions : {
        addItemAction({ commit }, item) {
            item.id=count++
            commit('addItemMutation', item)
        }
    },
    mutations: {
        addItemMutation(state,  item ) {
            state.items.push(item)
        }
    }
})
```

Vuex Overview

Vuex Overview

5 Store Instance 등록

- application instance의 use() 함수로 application에 등록하여 사용함

```
app.use(store)
```

Vuex Overview

6 action 발생

- 애플리케이션 내에서 Vuex의 구성요소는 this.$store 객체로 접근이 가능함
- action은 함수이며 함수 이름을 this.$store.dispatch() 함수에 대입하여 호출함

Vuex Overview

Vuex Overview

6 action 발생

- dispatch 함수를 호출하면서 데이터를 매개변수로 전달할 수 있음

```
this.$store.dispatch('addItemAction', data)
```

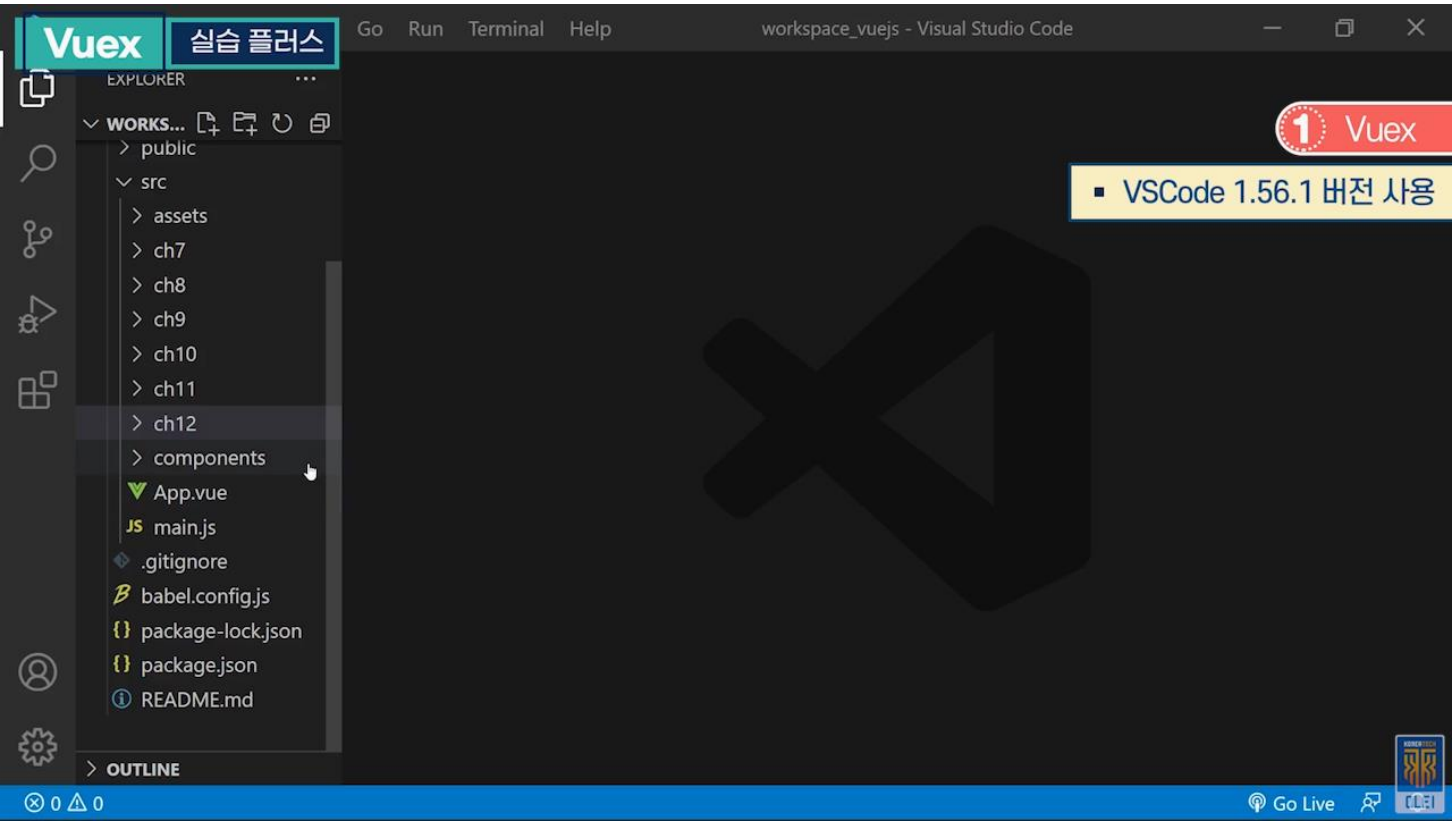
Vuex Overview

7 state 이용

- action 발생에 의해 생성/변경된 state 데이터는 this.$store.state 객체로 획득이 가능함

```
this.$store.state.items
```

실습 플러스



실습단계
src → ch13
이전에 생성한 ch11 폴더의 index.html, main.js, vue.config.js 파일 복사하여 이용
Vuex 테스트가 목적이므로 Vuex 먼저 설치함
vue-test 위치에 설치, npm install로 module 설치
Vuex를 활용할 수 있는 Test.vue 파일 생성
main.js 부분 정의, Vue가 import되어 있음, Vuex도 import
Vuex에 state나 action을 추가하기 위한 정보 포함
state는 count 값 하나 유지, state 변경을 위한 action 선언
increase 함수를 호출해서 state 값 변경
부가적인 업무를 수행할 수 있음, 최종 데이터 변경은 매개변수로 전달되는 commit 이용
Component에서 데이터 변경을 하기 위해 함수를 호출하면 action에서 추가 업무를 진행 후, commit 명령 선언
commit 명령에 의해 increase 함수를 호출하여 매개변수 값을 변경하는 구조
애플리케이션에 Vuex 정보 등록

실습 플러스

실습단계
Vuex를 Component에서 이용, 만들어 놓은 Vue Component 파일에서 정의
Component 함수가 호출됐을 때 Vuex 쪽에 action을 발생 → Vuex에 의해서 이용되는 Component 내장 객체
Vuex에 유지하는 데이터를 가져오기 위해 computed 사용
Vuex 데이터를 얻을 수 있음
현재 Vuex에 유지되고 있는 데이터 출력, 데이터를 변경할 수 있는 버튼 추가
유저가 화면에서 버튼을 눌렀을 때 increase 함수가 호출되고 함수 내부에서 Vuex에 action을 발생시켜서 값 변경
Vue 명령으로 실행
count 값이 출력되어 있고 increase 버튼이 있음