Pinia

Pinia

- **Pinia**

Vuex와 마찬가지로 상태 관리하는 라이브러리
VueConf Toronto 2021에서 Avan You가 상태 관리 라이브러리를 Vuex가 아닌 Pinia를 추천 하였으며, Pinia 공식 문서에 따르면 Vuejs가 Pinia를 공식 지원한다고 함
* 그렇다고 Vuex의 지원이 끊기는것이 아니라 그대로 유지됨

- **상태 관리**

여러 컴포넌트 간의 데이터 전달 및 이벤트 통신을 한 곳에서 관리하는 것을 말함

Pinia

- **state**

  여러 컴포넌트가 공유할 데이터를 모아놓은 공간

- **actions**

  데이터 변경, 비동기 작업, API 호출 등을 수행하는 공간

- **getters**

  데이터에 대한 어떠한 연산을 수행하거나, 데이터에 접근하는 등의 작업을 수행하는 공간

Pinia

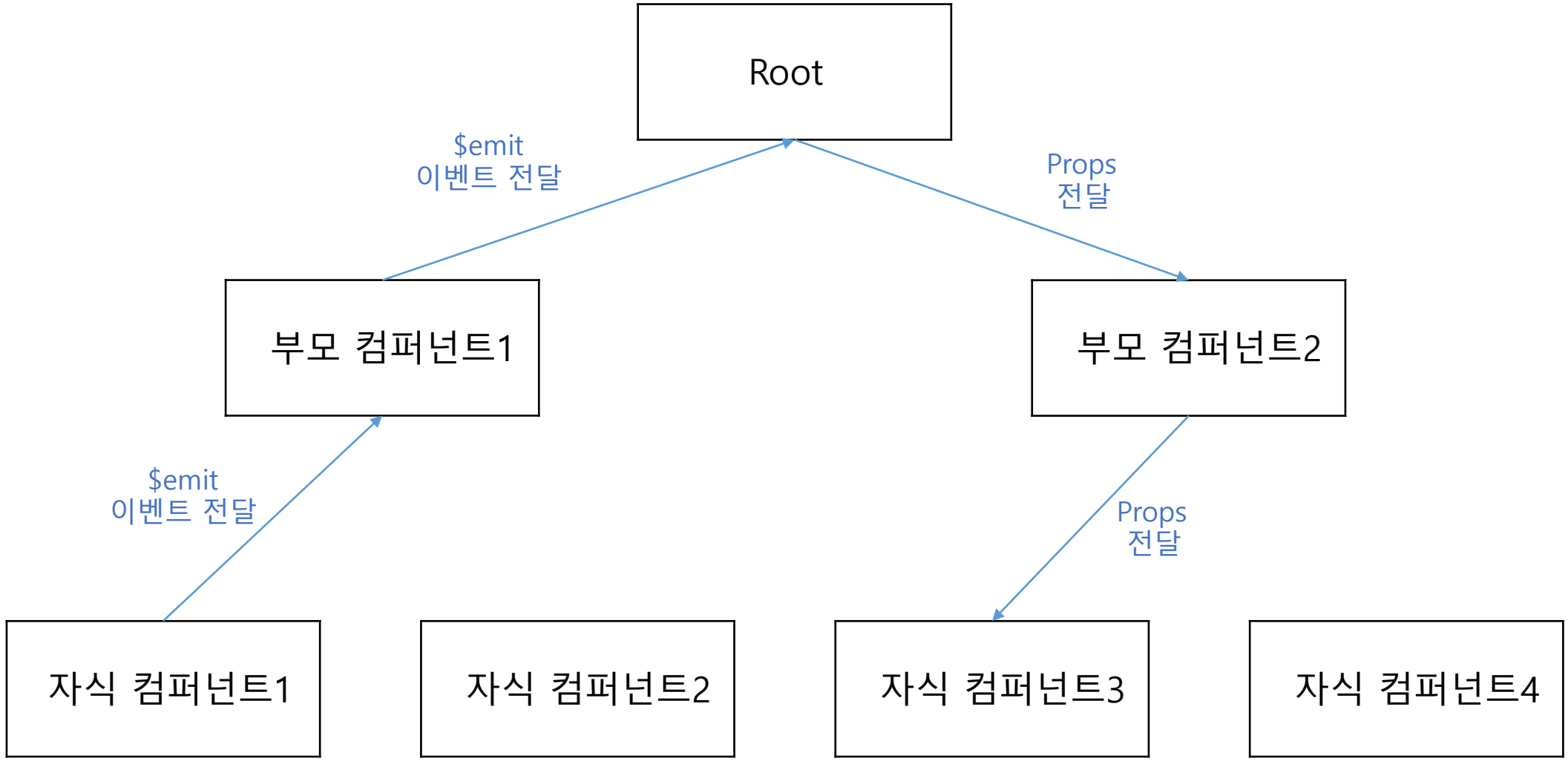
- **기존 컴포넌트 통신 방식의 문제점**

기존 컴포넌트 통신 방식의 경우 props, event emit을 사용하기 때문에 컴포넌트가 많아질수록 중간에 거쳐야 할 컴포넌트가 증가함.

이를 피하기 위해 Event Bus를 사용하는 방법이 나타났으나, 컴포넌트간의 데이터 흐름을 파악하기가 힘들어지는 단점이 생김

Pinia

- 기존 컴포넌트 통신 방식



Pinia

- **기존 컴포넌트 통신 방식의 단점을 보안**

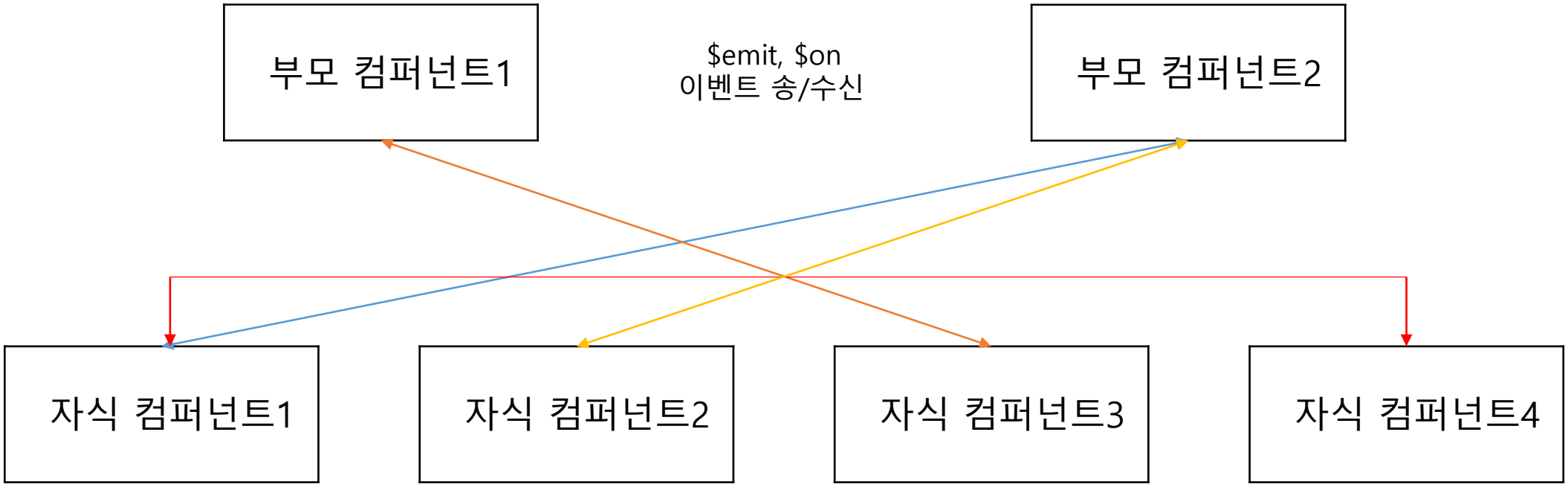
  기존 컴포넌트의 통신 방식의 단점을 보안하기 위해 Event Bus를 사용하는 방법이 나타났으나, 컴포넌트간의 데이터 흐름을 파악하기가 힘들어지는 단점이 생김

- **Event Bus**

  컴포넌트 계층 구조와 상관 없이 컴포넌트 간의 데이터를 전달할 수 있는 방법으로써 event emit(이벤트 송신), event on(이벤트 수신) 두가지를 사용함.

Pinia

- 이벤트 버스 통신 방식



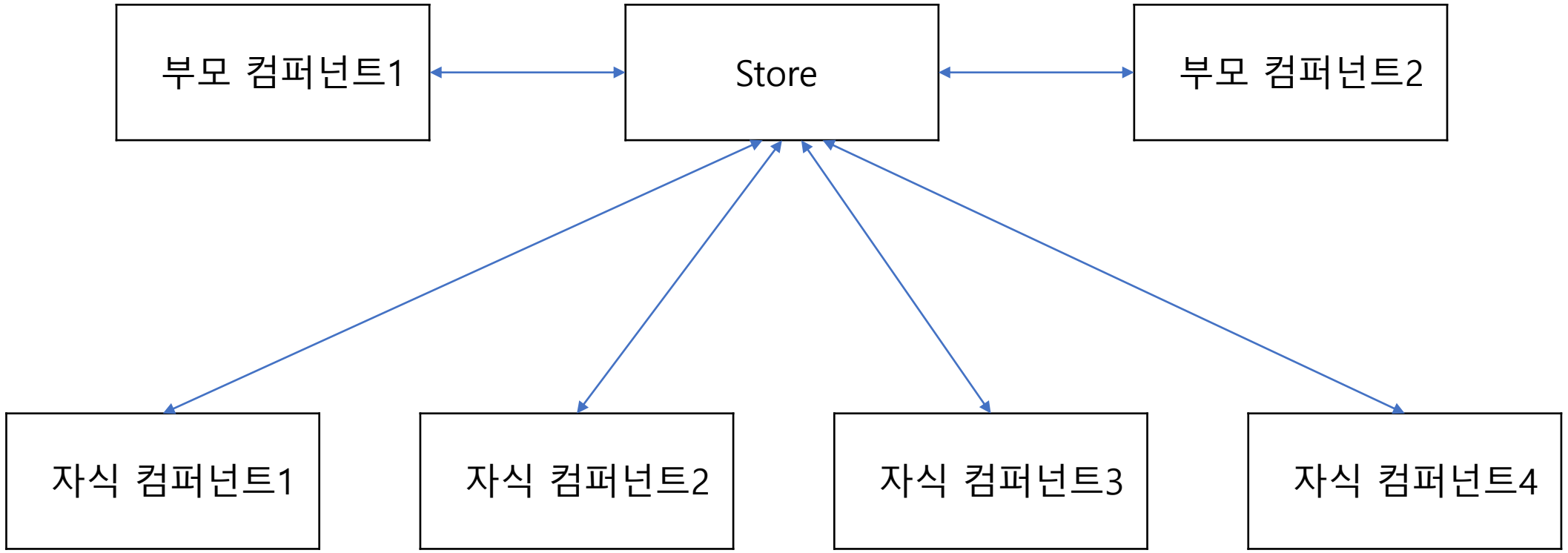
Pinia

- **상태 관리**

  위의 문제들로 인해 데이터 전달 및 이벤트 통신을 한 곳에서 관리하기 위한 중앙 집중식으로 만들어진 것이 상태 관리 개념으로써 Vuex가 등장함

Pinia

- **상태 관리를 활용한 통신 방식**



Pinia 사용해보기

Pinia

- **Pinia 설치**

```
npm install pinia --save
```

```
user@DESKTOP-I9RHACC MINGW64 /d/vuejs/vue-project
$ npm install pinia --save

added 2 packages, and audited 39 packages in 1s

6 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

Pinia

- **Pinia 등록**

src/main.js에 아래와 같이 pinia 등록

```
import { createApp } from 'vue'
import { createPinia } from 'pinia'
import App from './App.vue'
import router from './router/index'

const pinia = createPinia()
const app = createApp(App)

app.use(router)
app.use(pinia)

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

Pinia

- **Pinia 실습을 위한 Home.vue 설정**

pinia router-link 작성

```
<template>
    <nav>
        <ul>
            <li>
                <router-link to="/first">First : {{ firstCount }}</router-link>
            </li>
            <li>
                <router-link to="/Second">Second : {{ secondCount }}</router-link>
            </li>
            <li>
                <router-link to="/ComponentView">component</router-link>
            </li>
            <li>
                <router-link to="/Pinia">pinia</router-link>
            </li>
        </ul>
    </nav>
</template>
```

Pinia

- **Pinia 실습을 위한 router 설정**

import 및 routes 값 설정

```
import { createRouter, createWebHistory } from 'vue-router'
import Home from '../views/Home.vue'
import First from '../views/FirstPage.vue'
import ButtonCounterView from '../views/ButtonCounterView.vue'
import Second from '../views/SecondPage.vue'
import NotFound from '../views/NotFound.vue'

import ex1Parent from '../views/ex1/ParentComponent.vue'
import Pinia from '../views/ex2_pinia/List.vue'

const routes = [
  { path: '/', name:"Home", component: Home},
  { path: '/First', name:"first", component: First},
  { path: '/Second', name:"second", component: Second},
  { path: '/ComponentView', name:"componentView", component: ButtonCounterView},
  { path: '/ex1Parent', name:"ex1Parent", component: ex1Parent},
  { path: '/Pinia', name:"pinia", component: Pinia},
  { path: '/404', name: "notFound", component: NotFound},
  { path: '/:pathMatch(.*)*', redirect: "/404"}
];

const router = createRouter ({
  history: createWebHistory(),
  routes
});

export default router;
```

Pinia

- views/stores/list.js

```
import { defineStore } from "pinia";

export const listStore = defineStore('List',{
    // state : () => ({ list: [] }),
    state() {
        return {
            list: []
        }
    },
    actions: {
        addList(data) {
            this.list.push(data);
        }
    },
    getters: {
        getList(state) {
            return state.list;
        }
    }
})
```

화살표 함수를 사용한 예시

state → vuejs3의 data 역할
list: [] → 배열 형태로 저장

매개변수로 받은 데이터를 list에 추가(push)

state의 값을 전달 받음

state에 있는 list를 반환

Pinia

- views/ex2_pinia/List.vue

```vue
<template>
    <div>
        <input type="text" v-model="text" />
        <button @click="addUserList">추가</button>

        <p v-for="(item, index) in getList" :key="index">
            {{ item }}
        </p>
    </div>
</template>

<script>
import { mapActions, mapState } from 'pinia';
import { listStore } from '../../stores/list';

export default {
    // data: () => ({
    //     text: '',
    // }),
    data() {
        return {
            text: ''
        }
    },
    computed: {
        ...mapState(listStore, ['getList'])
    },
    methods: {
        ...mapActions(listStore, ['addList']),
        addUserList() {
            this.addList(this.text);
            this.text = '';
        }
    }
}
</script>
```

list.js getters의 getList 호출

pinia와 이전 슬라이드에서 만든 list.js 임포트

화살표 함수를 사용한 예시

list.js getters의 getList 호출

list.js actions의 addList 호출