Component

Component

- **컴포넌트(Component)**

  Vue.js의 핵심 기능 중 하나로써 재사용 가능한 조각으로 나누어 관리하여
  재사용성이 증가하고, 각자 나누어 관리하기 때문에 코드가 몰리는 현상이 사라져
  가독성과 유지보수성이 좋아짐

Component

- 설명

src/views/Home.vue에 아래와 같이 코드 한줄 추가

```
<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>
        </ul>s
    </nav>
</template>
```

Component

- 설명

src/views/components/ButtonCounter.vue 파일 생성 및 코드 추가

```
<template>
    <div id="btn">
        <br>
        <button @click="count++">
            First : {{ count }}번 클릭되었습니다
        </button>

    </div>
</template>
<script>
export default {
    data() {
        return {
            count: 0
        }
    }
}
</script>
```

Component

- 설명

src/views/ButtonCounterView.vue 파일 생성 및 코드 추가

```
<template>

    <ButtonCounterVue></ButtonCounterVue>
    <ButtonCounterVue></ButtonCounterVue>
</template>
<script>
import ButtonCounterVue from "./components/ButtonCounter.vue";

export default {
    components: {
        ButtonCounterVue
    }
}
</script>
```

Component

- 설명

src/router/index.js에 아래와 같이 코드 추가

```
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'

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: '/404', name: "notFound", component: NotFound},
  { path: '/:pathMatch(.*)*', redirect: "/404"}
];
```

Component

- **부모 컴퍼넌트**

특정 파일에서 import하여 컴포넌트를 사용할 경우 부모-자식 관계가 성립됨
- ButtonCounterView : import하여 컴포넌트를 가져와 사용하므로 부모 컴퍼넌트
- ButtonCounter : 자식 컴퍼넌트

ButtonCounterView

```
<template>
    <ButtonCounterVue></ButtonCounterVue>
</template>
<script>
import ButtonCounterVue from "./components/ButtonCounter.vue";

export default {
    components: {
        ButtonCounterVue
    }
}
</script>
```

ButtonCounter

```
<template>
    <div id="btn">
        <br>
        <button @click="count++">
            First : {{ count }}번 클릭되었습니다
        </button>

    </div>
</template>
<script>
export default {
    data() {
        return {
            count: 0
        }
    }
}
</script>
```

부모 컴퍼넌트로 데이터 전달 ($emit)

$emit

- **부모 컴퍼넌트와 자식 컴퍼넌트 생성 후 테스트**

아래 2개 파일 생성

views/ex1/ChildComponent.vue → 자식 컴퍼넌트로 사용 예정
views/ex1/ParentComponent.vue → 부모 컴퍼넌트로 사용 예정

views/ex1/ParentComponent.vue

```
<template>
    <div>
    <h1>Parent Component</h1>
    <ChildComponent @test="handleIncrement"/>
    <p>Click Count: {{ count }}</p>
    </div>
</template>

<script>
import ChildComponent from './ChildComponent.vue';

export default {
    components: {
        ChildComponent,
    },
    data() {
    return {
        count: 0,
    };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

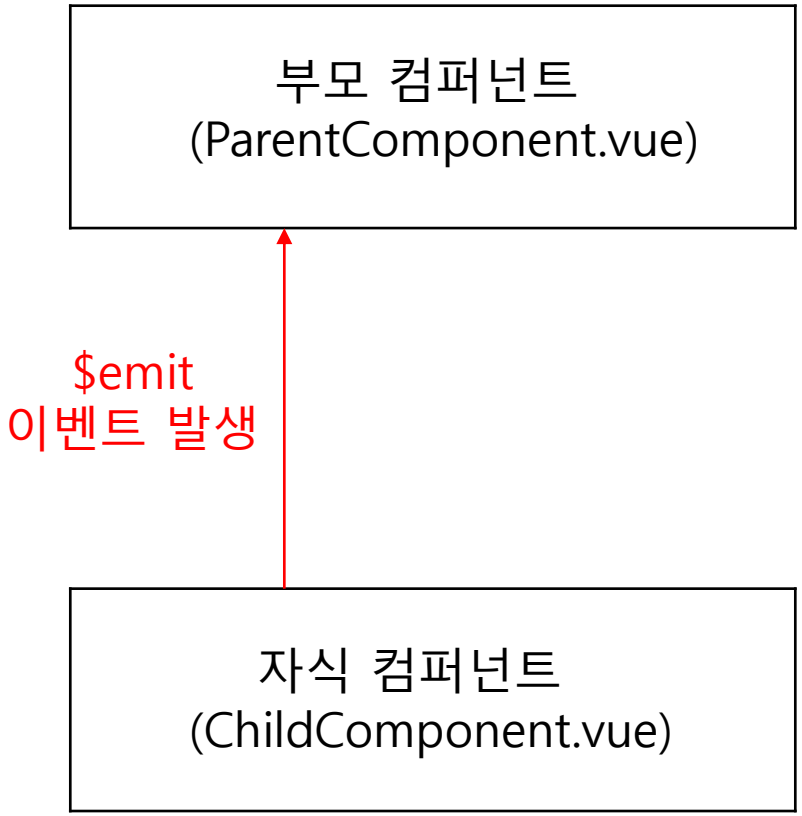
views/ex1/ChildComponent.vue

```
<template>
    <div>
        <button @click="handleClick">Increment</button>
    </div>
</template>

<script>
export default {
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
};
</script>
```

$emit

- **$emit**

부모 컴퍼넌트로 이벤트를 발생시키며, 특정 메서드를 실행하거나 매개변수를 전달할 수 있음



$emit

- **$emit**

부모 컴퍼넌트로 이벤트를 발생시키며, 특정 메서드를 실행하거나 매개변수를 전달할 수 있음

```vue
<template>
    <div>
    <h1>Parent Component</h1>
    <ChildComponent @test="handleIncrement"/>
    <p>Click Count: {{ count }}</p>
    </div>
</template>

<script>
import ChildComponent from './ChildComponent.vue';

export default {
    components: {
        ChildComponent,
    },
    data() {
    return {
        count: 0,
    };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

```vue
<template>
    <div>
    <button @click="handleClick">Increment</button>
    </div>
</template>

<script>
export default {
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
};
</script>
```

test라는 이벤트를 발생시키며, 내가 가지고 있는 count의 값을 매개변수로 전달

자식 컴퍼넌트에서 부모 컴퍼넌트로 이벤트를 발생시킴

$emit

- **$emit**

특정 Vue 파일을 import하여 컴포넌트를 사용할 경우 부모 – 자식 관계가 성립됨

```vue
<template>
    <div>
    <h1>Parent Component</h1>
    <ChildComponent @test="handleIncrement"/>
    <p>Click Count: {{ count }}</p>
    </div>
</template>

<script>
import ChildComponent from './ChildComponent.vue';

export default {
    components: {
        ChildComponent,
    },
    data() {
    return {
        count: 0,
    };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

부모 컴퍼넌트에
test 이벤트 전달

```vue
<template>
    <div>
    <button @click="handleClick">Increment</button>
    </div>
</template>

<script>
export default {
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
};
</script>
```

$emit

- **$emit**

특정 Vue 파일을 import하여 컴포넌트를 사용할 경우 부모 – 자식 관계가 성립됨

```vue
<template>
    <div>
    <h1>Parent Component</h1>
    <ChildComponent @test="handleIncrement"/>
    <p>Click Count: {{ count }}</p>
    </div>
</template>

<script>
import ChildComponent from './ChildComponent.vue';

export default {
    components: {
        ChildComponent,
    },
    data() {
    return {
        count: 0,
    };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

이벤트가 발생하면 해당하는 메서드를 호출 및 실행

```vue
<template>
    <div>
    <button @click="handleClick">Increment</button>
    </div>
</template>

<script>
export default {
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
};
</script>
```

$emit

- **$emit**

특정 Vue 파일을 import하여 컴포넌트를 사용할 경우 부모 – 자식 관계가 성립됨

```vue
<template>
    <div>
    <h1>Parent Component</h1>
    <ChildComponent @test="handleIncrement"/>
    <p>Click Count: {{ count }}</p>
    </div>
</template>

<script>
import ChildComponent from './ChildComponent.vue';

export default {
    components: {
        ChildComponent,
    },
    data() {
    return {
        count: 0,
    };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

전달받은 매개변수를 자신의 count 변수에 저장

```vue
<template>
    <div>
    <button @click="handleClick">Increment</button>
    </div>
</template>

<script>
export default {
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
};
</script>
```

$emit

- **$emit**

특정 Vue 파일을 import하여 컴포넌트를 사용할 경우 부모 – 자식 관계가 성립됨

```vue
<template>
    <div>
    <h1>Parent Component</h1>
    <ChildComponent @test="handleIncrement"/>
    <p>Click Count: {{ count }}</p>
    </div>
</template>

<script>
import ChildComponent from './ChildComponent.vue';

export default {
    components: {
        ChildComponent,
    },
    data() {
    return {
        count: 0,
    };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

count 전달 및 출력

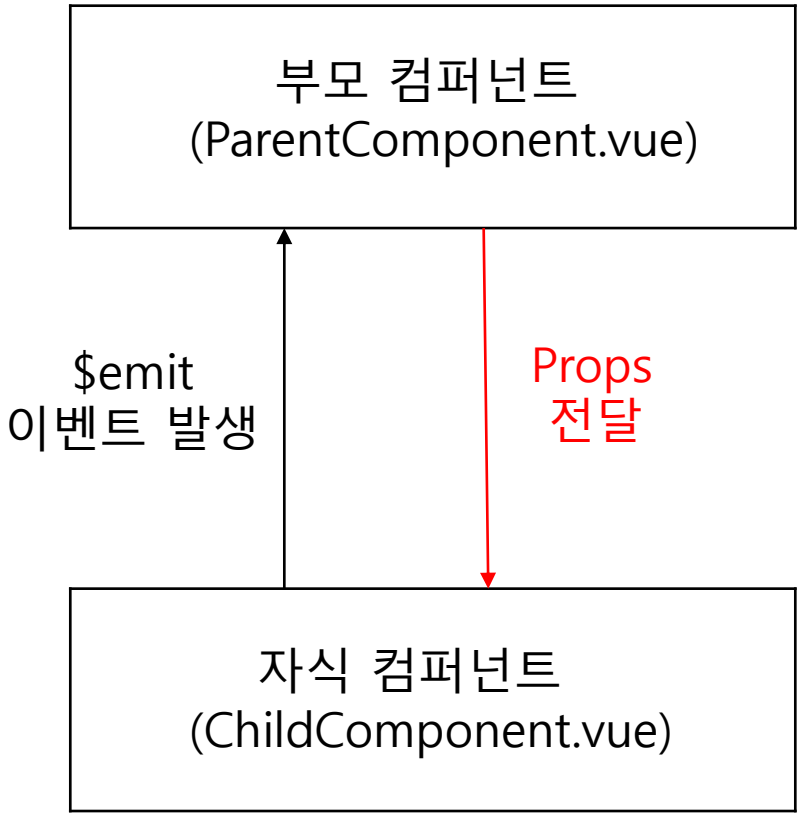
```vue
<template>
    <div>
    <button @click="handleClick">Increment</button>
    </div>
</template>

<script>
export default {
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
};
</script>
```

자식 컴퍼넌트로 데이터 전달 (props)

props

- **props**

부모 컴퍼넌트에서 자식 컴퍼넌트로 데이터를 전달하기 위한 속성



props

- **props**

부모 컴퍼넌트로 이벤트를 발생시키며, 특정 메서드를 실행하거나 매개변수를 전달할 수 있음

```vue
<template>
    <div>
        <h1>Parent Component</h1>
        <ChildComponent :name="name" @test="handleIncrement"/>
        <p>Click Count: {{ count }}</p>
    </div>
</template>

<script>
import ChildComponent from './ChildComponent.vue';

export default {
    components: {
        ChildComponent,
    },
    data() {
        return {
            count: 0,
            name: 'VueJs'
        };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

자식 컴퍼넌트로 바인딩

자식 컴퍼넌트로 보낼 변수 생성

```vue
<template>
    <div>
        <button @click="handleClick">Increment</button>
        <p> {{ name }}</p>
    </div>
</template>

<script>
export default {
    props: ['name'],
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
    created() {
        console.log(this.name-child);
    }
};
</script>
```

props

- **props**

부모 컴퍼넌트로 이벤트를 발생시키며, 특정 메서드를 실행하거나 매개변수를 전달할 수 있음

```vue
<template>
    <div>
        <h1>Parent Component</h1>
        <ChildComponent :name="name" @test="handleIncrement"/>
        <p>Click Count: {{ count }}</p>
    </div>
</template>

<script>
import ChildComponent from './ChildComponent.vue';

export default {
    components: {
        ChildComponent,
    },
    data() {
        return {
            count: 0,
            name: 'VueJs'
        };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

자식 컴퍼넌트로 바인딩

자식 컴퍼넌트로 보낼 변수 생성

```vue
<template>
    <div>
        <button @click="handleClick">Increment</button>
        <p> {{ name }}</p>
    </div>
</template>

<script>
export default {
    props: ['name'],
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
    created() {
        console.log(this.name-child);
    }
};
</script>
```

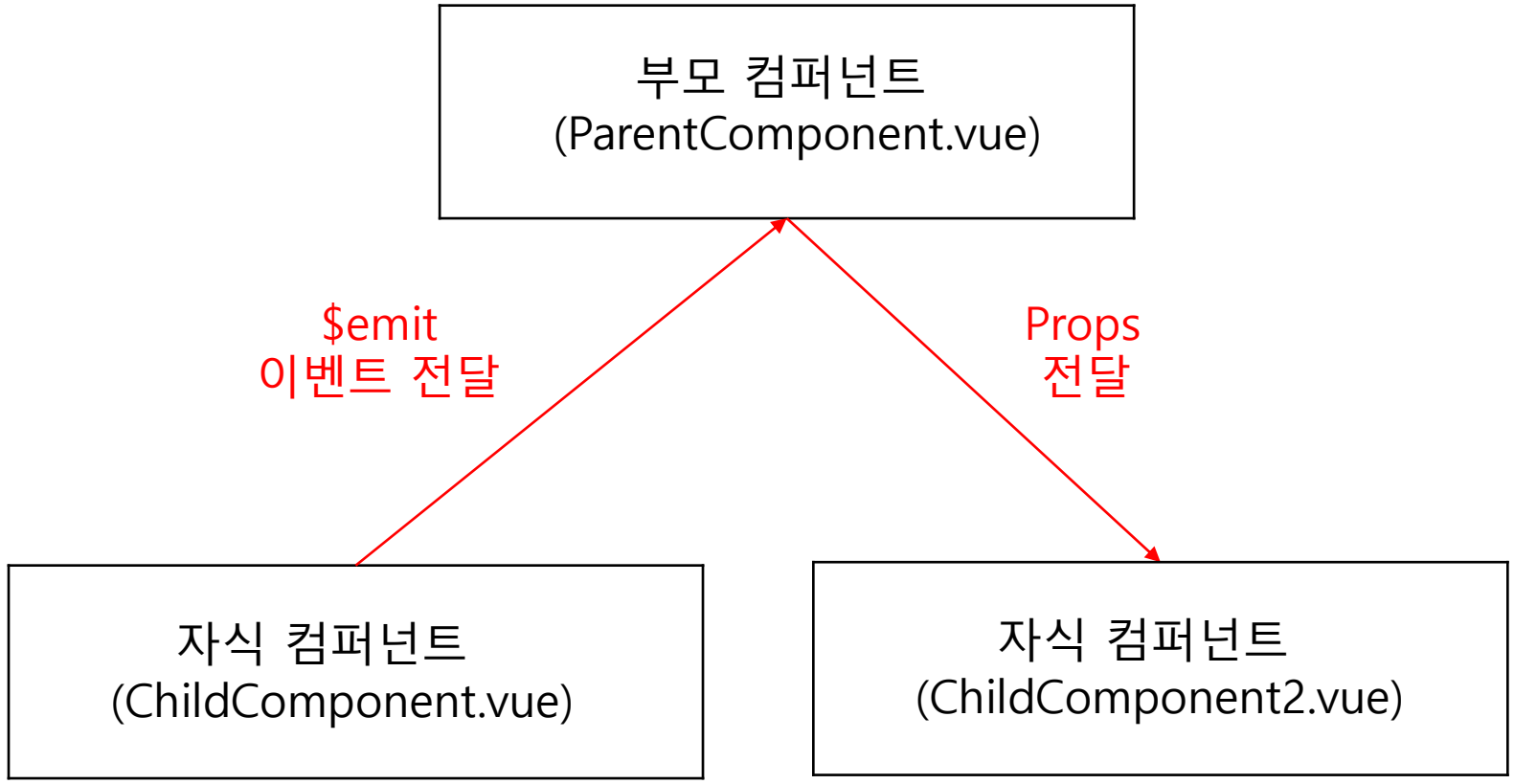
props로 전달받은 데이터 사용

동위 컴퍼넌트로 데이터 전달

동위 컴퍼넌트로 데이터 전달

- **동위 컴퍼넌트로 데이터 전달**

같은 레벨의 컴퍼넌트로 데이터를 전달하기 위해서는 부모 컴퍼넌트를 이용하여 전달해야 함



동위 컴퍼넌트로 데이터 전달

ChildComponent.vue

```vue
<template>
    <div>
        <button @click="handleClick">Increment</button>
        <p> {{ name }}</p>
    </div>
</template>

<script>
export default {
    props: ['name'],
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
    created() {
        // console.log(this.name-child);
    }
};
</script>
```

부모 컴퍼넌트로
이벤트 전달

ParentComponent.vue

```vue
<template>
    <div>
        <h1>Parent Component</h1>
        <ChildComponent :name="name" @test="handleIncrement"/>
        <p>Click Count: {{ count }}</p>
        <hr>
        <ChildComponent2 :count="count"/>
    </div>

</template>

<script>
import ChildComponent from './ChildComponent.vue';
import ChildComponent2 from './ChildComponent2.vue';

export default {
    components: {
        ChildComponent,
        ChildComponent2,
    },
    data() {
        return {
            count: 0,
            name: 'VueJs'
        };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

ChildComponent2.vue

```vue
<template>
    <div>
        <button>Increment</button>
        <p> {{ count }}</p>
    </div>
</template>

<script>
export default {
    props: ['count']
};
</script>
```

동위 컴퍼넌트로 데이터 전달

ChildComponent.vue

```vue
<template>
    <div>
        <button @click="handleClick">Increment</button>
        <p> {{ name }}</p>
    </div>
</template>

<script>
export default {
    props: ['name'],
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
    created() {
        // console.log(this.name-child);
    }
};
</script>
```

부모 컴퍼넌트로
이벤트 전달

ParentComponent.vue

```vue
<template>
    <div>
        <h1>Parent Component</h1>
        <ChildComponent :name="name" @test="handleIncrement"/>
        <p>Click Count: {{ count }}</p>
        <hr>
        <ChildComponent2 :count="count"/>
    </div>

</template>

<script>
import ChildComponent from './ChildComponent.vue';
import ChildComponent2 from './ChildComponent2.vue';

export default {
    components: {
        ChildComponent,
        ChildComponent2,
    },
    data() {
        return {
            count: 0,
            name: 'VueJs'
        };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

함수 실행

ChildComponent2.vue

```vue
<template>
    <div>
        <button>Increment</button>
        <p> {{ count }}</p>
    </div>
</template>

<script>
export default {
    props: ['count']
};
</script>
```

동위 컴퍼넌트로 데이터 전달

ChildComponent.vue

```vue
<template>
    <div>
        <button @click="handleClick">Increment</button>
        <p> {{ name }}</p>
    </div>
</template>

<script>
export default {
    props: ['name'],
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
    created() {
        // console.log(this.name-child);
    }
};
</script>
```

부모 컴퍼넌트로
이벤트 전달

ParentComponent.vue

```vue
<template>
    <div>
        <h1>Parent Component</h1>
        <ChildComponent :name="name" @test="handleIncrement"/>
        <p>Click Count: {{ count }}</p>
        <hr>
        <ChildComponent2 :count="count"/>
    </div>

</template>

<script>
import ChildComponent from './ChildComponent.vue';
import ChildComponent2 from './ChildComponent2.vue';

export default {
    components: {
        ChildComponent,
        ChildComponent2,
    },
    data() {
        return {
            count: 0,
            name: 'VueJs'
        };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

값 저장

ChildComponent2.vue

```vue
<template>
    <div>
        <button>Increment</button>
        <p> {{ count }}</p>
    </div>
</template>

<script>
export default {
    props: ['count']
};
</script>
```

동위 컴퍼넌트로 데이터 전달

ChildComponent.vue

```vue
<template>
    <div>
        <button @click="handleClick">Increment</button>
        <p> {{ name }}</p>
    </div>
</template>

<script>
export default {
    props: ['name'],
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
    created() {
        // console.log(this.name-child);
    }
};
</script>
```

ParentComponent.vue

```vue
<template>
    <div>
        <h1>Parent Component</h1>
        <ChildComponent :name="name" @test="handleIncrement"/>
        <p>Click Count: {{ count }}</p>
        <hr>
        <ChildComponent2 :count="count"/>
    </div>

</template>

<script>
import ChildComponent from './ChildComponent.vue';
import ChildComponent2 from './ChildComponent2.vue';

export default {
    components: {
        ChildComponent,
        ChildComponent2,
    },
    data() {
        return {
            count: 0,
            name: 'VueJs'
        };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

저장된 값을
자식2로 바인딩

ChildComponent2.vue

```vue
<template>
    <div>
        <button>Increment</button>
        <p> {{ count }}</p>
    </div>
</template>

<script>
export default {
    props: ['count']
};
</script>
```

동위 컴퍼넌트로 데이터 전달

ChildComponent.vue

```vue
<template>
    <div>
        <button @click="handleClick">Increment</button>
        <p> {{ name }}</p>
    </div>
</template>

<script>
export default {
    props: ['name'],
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
    created() {
        // console.log(this.name-child);
    }
};
</script>
```

ParentComponent.vue

```vue
<template>
    <div>
        <h1>Parent Component</h1>
        <ChildComponent :name="name" @test="handleIncrement"/>
        <p>Click Count: {{ count }}</p>
        <hr>
        <ChildComponent2 :count="count"/>
    </div>

</template>

<script>
import ChildComponent from './ChildComponent.vue';
import ChildComponent2 from './ChildComponent2.vue';

export default {
    components: {
        ChildComponent,
        ChildComponent2,
    },
    data() {
        return {
            count: 0,
            name: 'VueJs'
        };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

저장된 값을
자식2로 바인딩

props로
값 받아옴

ChildComponent2.vue

```vue
<template>
    <div>
        <button>Increment</button>
        <p> {{ count }}</p>
    </div>
</template>

<script>
export default {
    props: ['count']
};
</script>
```

동위 컴퍼넌트로 데이터 전달

ChildComponent.vue

```vue
<template>
    <div>
        <button @click="handleClick">Increment</button>
        <p> {{ name }}</p>
    </div>
</template>

<script>
export default {
    props: ['name'],
    data() {
        return {
            count: 0,
        };
    },
    methods: {
        handleClick() {
            this.count++;
            this.$emit('test', this.count);
        },
    },
    created() {
        // console.log(this.name-child);
    }
};
</script>
```

ParentComponent.vue

```vue
<template>
    <div>
        <h1>Parent Component</h1>
        <ChildComponent :name="name" @test="handleIncrement"/>
        <p>Click Count: {{ count }}</p>
        <hr>
        <ChildComponent2 :count="count"/>
    </div>

</template>

<script>
import ChildComponent from './ChildComponent.vue';
import ChildComponent2 from './ChildComponent2.vue';

export default {
    components: {
        ChildComponent,
        ChildComponent2,
    },
    data() {
        return {
            count: 0,
            name: 'VueJs'
        };
    },
    methods: {
    handleIncrement(count) {
        this.count = count
    },
    },
};
</script>
```

저장된 값을
자식2로 바인딩

props로
값 받아옴

ChildComponent2.vue

```vue
<template>
    <div>
        <button>Increment</button>
        <p> {{ count }}</p>
    </div>
</template>

<script>
export default {
    props: ['count']
};
</script>
```

받아온
데이터 사용