



컴포넌트 등록

이 페이지는 이미 **컴포넌트 기본**을 읽었다고 가정합니다. 컴포넌트가 처음이라면 먼저 해당 내용을 읽어보세요.

[Vue School의 무료 동영상 강의 보기](#)

Vue 컴포넌트는 템플릿에서 해당 구현체를 찾을 수 있도록 "등록"되어야 합니다. 컴포넌트를 등록하는 방법에는 전역 등록과 지역 등록, 두 가지가 있습니다.

전역 등록

현재 **Vue** 애플리케이션에서 `.component()` 메서드를 사용하여 컴포넌트를 전역적으로 사용할 수 있습니다:

```
import { createApp } from 'vue'

const app = createApp({})

app.component(
  // 등록할 이름
  'MyComponent',
  // 구현체
  {
    /* ... */
  }
)
```

js

SFC를 사용하는 경우, 가져온 `.vue` 파일을 등록하게 됩니다:

```
import MyComponent from './App.vue'
```

js



`.component()` 메서드는 체이닝이 가능합니다:

```
app
  .component('ComponentA', ComponentA)
  .component('ComponentB', ComponentB)
  .component('ComponentC', ComponentC)
```

js

전역으로 등록된 컴포넌트는 이 애플리케이션 내의 모든 컴포넌트의 템플릿에서 사용할 수 있습니다:

```
<!-- 이 코드는 앱 내의 어떤 컴포넌트에서도 동작합니다 -->
<ComponentA/>
<ComponentB/>
<ComponentC/>
```

template

이것은 모든 하위 컴포넌트에도 적용되므로, 이 세 컴포넌트는 서로 내부에서도 사용할 수 있습니다.

지역 등록

전역 등록은 편리하지만 몇 가지 단점이 있습니다:

1. 전역 등록은 빌드 시스템이 사용하지 않는 컴포넌트(즉, "트리 셰이킹")를 제거하지 못하게 합니다. 컴포넌트를 전역으로 등록했지만 앱 어디에서도 사용하지 않는다면, 최종 번들에 여전히 포함됩니다.
2. 전역 등록은 대규모 애플리케이션에서 의존성 관계를 덜 명확하게 만듭니다. 부모 컴포넌트에서 자식 컴포넌트의 구현체를 찾기 어렵게 하여, 너무 많은 전역 변수를 사용하는 것과 비슷하게 장기적인 유지보수에 영향을 줄 수 있습니다.

지역 등록은 등록된 컴포넌트의 사용 범위를 현재 컴포넌트로만 제한합니다. 의존성 관계를 더 명확하게 만들고, 트리 셰이킹에도 더 적합합니다.

`<script setup>` 이 있는 SFC를 사용할 때는, 가져온 컴포넌트를 별도의 등록 없이 지역적으로 사용할 수 있습니다:

```
<script setup>
import ComponentA from './ComponentA.vue'
</script>
```

vue



```
<template>
  <ComponentA />
</template>
```

`<script setup>` 이 아닌 경우에는 `components` 옵션을 사용해야 합니다:

```
import ComponentA from './ComponentA.js'

export default {
  components: {
    ComponentA
  },
  setup() {
    // ...
  }
}
```

js

`components` 객체의 각 속성에서, 키는 컴포넌트의 등록 이름이 되고, 값은 컴포넌트의 구현체가 됩니다. 위 예시는 ES2015 속성 단축 표기를 사용한 것으로, 다음과 동일합니다:

```
export default {
  components: {
    ComponentA: ComponentA
  }
  // ...
}
```

js

지역 등록된 컴포넌트는 하위 컴포넌트에서 사용할 수 없습니다. 이 경우, `ComponentA` 는 현재 컴포넌트에서만 사용할 수 있고, 자식이나 하위 컴포넌트에서는 사용할 수 없습니다.

컴포넌트 이름 표기법

이 가이드 전반에서 컴포넌트를 등록할 때 `PascalCase` 이름을 사용하고 있습니다. 그 이유는 다음과 같습니다:

1. `PascalCase` 이름은 유효한 JavaScript 식별자입니다. 이를 통해 JavaScript에서 컴포넌트를 더 쉽게 가져오고 등록할 수 있습니다. 또한 IDE의 자동 완성 기능에도 도움이 됩니다.
2. `<PascalCase />` 는 템플릿에서 이것이 네이티브 HTML 요소가 아니라 Vue 컴포넌트임을 더 명확하게 보여줍니다. 또한 Vue 컴포넌트와 커스텀 엘리먼트(웹 컴포넌트)를 구분해줍니다.



사항에서 논의한 것처럼, in-DOM 템플릿에서는 PascalCase 태그를 사용할 수 없습니다.

다행히도, Vue는 케밥-케이스 태그를 PascalCase로 등록된 컴포넌트로 해석하는 것을 지원합니다. 즉, `MyComponent` 로 등록된 컴포넌트는 Vue 템플릿(또는 Vue가 렌더링한 HTML 요소) 내에서 `<MyComponent>` 와 `<my-component>` 모두로 참조할 수 있습니다. 이를 통해 템플릿 소스에 상관없이 동일한 JavaScript 컴포넌트 등록 코드를 사용할 수 있습니다.

 [GitHub에서 이 페이지 편집](#)

[< Previous](#)

생명주기 **혹**

[Next >](#)

Props