커스텀 디렉티브

소개

코어에 기본적으로 포함된 디렉티브 집합(예: `v-model` 또는 `v-show`) 외에도, Vue는 사용자 정의 커스텀 디렉티브를 등록할 수 있도록 지원합니다.

Vue에서는 두 가지 형태의 코드 재사용을 도입했습니다: 컴포넌트와 컴포저블입니다. 컴포넌트는 주요 빌딩 블록이며, 컴포저블은 상태 기반 로직의 재사용에 중점을 둡니다. 반면, 커스텀 디렉티브는 일반적으로 일반 엘리먼트에서 저수준 DOM 접근이 필요한 로직을 재사용할 때 주로 사용됩니다.

커스텀 디렉티브는 컴포넌트의 라이프사이클 훅과 유사한 라이프사이클 훅을 포함하는 객체로 정의됩니다. 훅은 디렉티브가 바인딩된 엘리먼트를 인자로 받습니다. 다음은 Vue가 DOM에 엘리먼트를 삽입할 때 클래스가 추가되는 디렉티브의 예시입니다:

```vue
<script setup>
// 템플릿에서 v-highlight를 사용할 수 있게 함
const vHighlight = {
  mounted: (el) => {
    el.classList.add('is-highlight')
  }
}
</script>

<template>
  <p v-highlight>이 문장은 중요합니다!</p>
</template>
```

이 문장은 중요합니다!

`<script setup>` 에서는 `v` 접두사로 시작하는 카멜케이스 변수는 커스텀 디렉티브로 사용할 수 있습니다. 위 예시에서 `vHighlight` 는 템플릿에서 `v-highlight` 로 사용할 수 있습니다.

할 수 있습니다:

```js
export default {
  setup() {
    /*...*/
  },
  directives: {
    // 템플릿에서 v-highlight를 사용할 수 있게 함
    highlight: {
      /* ... */
    }
  }
}
```

앱 레벨에서 커스텀 디렉티브를 전역 등록하는 것도 일반적입니다:

```js
const app = createApp({})

// 모든 컴포넌트에서 v-highlight를 사용할 수 있게 함
app.directive('highlight', {
  /* ... */
})
```

커스텀 디렉티브를 사용할 때

커스텀 디렉티브는 원하는 기능이 직접적인 DOM 조작을 통해서만 달성될 수 있을 때만 사용해야 합니다.

이의 일반적인 예로는 엘리먼트에 포커스를 주는 `v-focus` 커스텀 디렉티브가 있습니다.

```vue
<script setup>
// 템플릿에서 v-focus를 사용할 수 있게 함
const vFocus = {
  mounted: (el) => el.focus()
}
</script>

<template>
  <input v-focus />
</template>
```

이 디렉티브는 `autofocus` 속성보다 더 유용합니다. 왜냐하면 페이지 로드 시뿐만 아니라, Vue가 엘리먼트를 동적으로 삽입할 때도 동작하기 때문입니다!

더링에도 친화적이므로 이를 권장합니다.

디렉티브 훅

디렉티브 정의 객체는 여러 훅 함수(모두 선택 사항)를 제공할 수 있습니다:

```js
const myDirective = {
  // 바인딩된 엘리먼트의 속성이나
  // 이벤트 리스너가 적용되기 전에 호출됨
  created(el, binding, vnode) {
    // 인자에 대한 자세한 내용은 아래 참고
  },
  // 엘리먼트가 DOM에 삽입되기 직전에 호출됨
  beforeMount(el, binding, vnode) {},
  // 바인딩된 엘리먼트의 부모 컴포넌트와
  // 모든 자식이 마운트된 후 호출됨
  mounted(el, binding, vnode) {},
  // 부모 컴포넌트가 업데이트되기 전에 호출됨
  beforeUpdate(el, binding, vnode, prevVnode) {},
  // 부모 컴포넌트와 모든 자식이
  // 업데이트된 후 호출됨
  updated(el, binding, vnode, prevVnode) {},
  // 부모 컴포넌트가 언마운트되기 전에 호출됨
  beforeUnmount(el, binding, vnode) {},
  // 부모 컴포넌트가 언마운트될 때 호출됨
  unmounted(el, binding, vnode) {}
}
```

훅 인자

디렉티브 훅에는 다음과 같은 인자가 전달됩니다:

`el` : 디렉티브가 바인딩된 엘리먼트. 이 엘리먼트를 통해 직접 DOM을 조작할 수 있습니다.

`binding` : 다음 속성을 포함하는 객체입니다.

- `value` : 디렉티브에 전달된 값. 예를 들어 `v-my-directive="1 + 1"` 에서 값은 `2` 입니다.
- `oldValue` : 이전 값으로, `beforeUpdate` 와 `updated` 에서만 사용 가능합니다. 값이 변경되지 않았더라도 항상 제공됩니다.
- `arg` : 디렉티브에 전달된 인자(있는 경우). 예를 들어 `v-my-directive:foo` 에서 arg는 `"foo"` 입니다.
- `modifiers` : 수정자가 있는 경우, 이를 포함하는 객체입니다. 예를 들어 `v-my-directive.foo.bar` 에서 modifiers 객체는 `{ foo: true, bar: true }` 입니다.

`dir` : 디렉티브 정의 객체입니다.

`vnode` : 바인딩된 엘리먼트를 나타내는 내부 VNode입니다.

`prevVnode` : 이전 렌더에서 바인딩된 엘리먼트를 나타내는 VNode입니다. `beforeUpdate` 와 `updated` 훅에서만 사용 가능합니다.

예시로, 다음과 같은 디렉티브 사용을 생각해봅시다:

```template
<div v-example:foo.bar="baz">
```

`binding` 인자는 다음과 같은 형태의 객체가 됩니다:

```js
{
  arg: 'foo',
  modifiers: { bar: true },
  value: /* `baz`의 값 */,
  oldValue: /* 이전 업데이트에서의 `baz` 값 */
}
```

내장 디렉티브와 마찬가지로, 커스텀 디렉티브의 인자도 동적으로 사용할 수 있습니다. 예를 들어:

```template
<div v-example:[arg]="value"></div>
```

여기서 디렉티브 인자는 컴포넌트 상태의 `arg` 속성에 따라 반응적으로 업데이트됩니다.

> ⓘ 참고
>
> `el` 을 제외한 이 인자들은 읽기 전용으로 취급해야 하며, 절대 수정해서는 안 됩니다. 훅 간에 정보를 공유해야 한다면, 엘리먼트의 dataset을 통해 공유하는 것이 좋습니다.

함수 단축 표기

커스텀 디렉티브가 `mounted` 와 `updated` 에서 동일한 동작을 하고, 다른 훅이 필요 없는 경우가 많습니다. 이런 경우 디렉티브를 함수로 정의할 수 있습니다:

```template
<div v-color="color"></div>
```

```
  // 이 함수는 `mounted`와 `updated` 모두에서 호출됩니다
  el.style.color = binding.value
})
```

객체 리터럴

디렉티브에 여러 값을 전달해야 한다면, JavaScript 객체 리터럴을 전달할 수도 있습니다. 디렉티브에는 유효한 JavaScript 표현식이라면 무엇이든 전달할 수 있다는 점을 기억하세요.

template

```
<div v-demo="{ color: 'white', text: 'hello!' }"></div>
```

js

```
app.directive('demo', (el, binding) => {
  console.log(binding.value.color) // => "white"
  console.log(binding.value.text) // => "hello!"
})
```

컴포넌트에서의 사용

> ⚠ 권장하지 않음
>
> 컴포넌트에서 커스텀 디렉티브를 사용하는 것은 권장하지 않습니다. 컴포넌트에 여러 루트 노드가 있을 경우 예기치 않은 동작이 발생할 수 있습니다.

컴포넌트에서 사용될 때, 커스텀 디렉티브는 항상 컴포넌트의 루트 노드에 적용됩니다. 이는 속성 전달(Fallthrough Attributes)과 유사합니다.

template

```
<MyComponent v-demo="test" />
```

template

```
<!-- MyComponent의 템플릿 -->

<div> <!-- v-demo 디렉티브가 여기에 적용됩니다 -->
  <span>My component content</span>
</div>
```

티브를 적용하면 무시되고 경고가 발생합니다. 속성과 달리, 디렉티브는 `v-bind="$attrs"` 로 다른 엘리먼트에 전달할 수 없습니다.

GitHub에서 이 페이지 편집

Previous
컴포저블

Next
플러그인