비동기 컴포넌트

기본 사용법

거대한 앱에서는 앱을 더 작게 조각내어 나누고, 필요할 때만 서버에서 컴포넌트를 로드해야 할 수 있습니다. 이를 구현하기 위해 **defineAsyncComponent** 함수를 제공합니다:

```js
import { defineAsyncComponent } from 'vue'

const AsyncComp = defineAsyncComponent(() => {
  return new Promise((resolve, reject) => {
    // ...서버에서 컴포넌트를 로드하는 로직
    resolve(/* 로드 된 컴포넌트 */)
  })
})
// ... 일반 컴포넌트처럼 `AsyncComp`를 사용
```

위 예제처럼 defineAsyncComponent 는 Promise를 반환하는 로더 함수를 사용합니다. Promise의 resolve 콜백을 호출해 서버에서 가져온 정의되어 있는 컴포넌트를 반환합니다. 로드가 실패했음을 나타내기 위해 reject(reason) 를 호출할 수도 있습니다.

ES 모듈 동적으로 가져오기도 Promise를 반환하므로, 대부분의 경우 defineAsyncComponent 와 함께 사용합니다. Vite 및 webpack과 같은 번들러에서도 이 문법을 지원하므로 Vue SFC를 가져오는 데 사용할 수 있습니다.

```js
import { defineAsyncComponent } from 'vue'

const AsyncComp = defineAsyncComponent(() =>
  import('./components/MyComponent.vue')
)
```

반환된 AsyncComp 는 페이지에서 실제로 렌더링될 때만 로더 함수를 호출하는 래퍼 컴포넌트입니다. 또한 모든 props를 내부 컴포넌트에 전달하므로, 기존 구현된 컴포넌트를 비동기 래퍼 컴포넌트로 문제없이 교체하여 지연(lazy) 로드를 구현할 수 있습니다.

일반 컴포넌트 처럼, 비동기 컴포넌트도 app.component() 를 통해 전역 등록이 가능합니다:

```js
app.component('MyComponent', defineAsyncComponent(() =>
  import('./components/MyComponent.vue')
))
```

부모 컴포넌트 안에서 직접 선언될수 있습니다.

```vue
<script setup>
import { defineAsyncComponent } from 'vue'

const AdminPage = defineAsyncComponent(() =>
  import('./components/AdminPageComponent.vue')
)
</script>

<template>
  <AdminPage />
</template>
```

로딩 및 에러 상태

비동기 작업에는 필연적으로 로드 및 에러 상태가 포함됩니다. `defineAsyncComponent()` 는 고급 옵션을 통해 이러한 상태 처리를 지원합니다:

```js
const AsyncComp = defineAsyncComponent({
  // 로더 함수
  loader: () => import('./Foo.vue'),

  // 비동기 컴포넌트가 로드되는 동안 사용할 로딩 컴포넌트입니다.
  loadingComponent: LoadingComponent,
  // 로딩 컴포넌트를 표시하기 전에 지연할 시간. 기본값: 200ms
  delay: 200,

  // 로드 실패 시 사용할 에러 컴포넌트
  errorComponent: ErrorComponent,
  // 시간 초과 시, 에러 컴포넌트가 표시됩니다. 기본값: 무한대
  timeout: 3000
})
```

로딩 컴포넌트가 제공되면 내부 컴포넌트가 로딩되는 동안 먼저 표시됩니다. 로딩 컴포넌트가 표시되기 전에 기본 200ms 지연시간이 있습니다. 이는 빠른 네트워크에서 인스턴트 로딩 상태가 너무 빨리 교체되어 깜박임처럼 보일 수 있기 때문입니다.

에러 컴포넌트가 제공되면 로더 함수의 Promise가 reject로 반환될 때 표시됩니다. 요청이 너무 오래 걸릴 때 에러 컴포넌트를 표시하도록 시간 초과를 지정할 수도 있습니다.

지연(suspense) 사용하기

비동기 컴포넌트는 내장 컴포넌트인 `<Suspense>` 와 함께 사용할 수 있습니다. `<Suspense>` 와 비동기 컴포넌트 간의 상호 작용은 **`<Suspense>`** 에 설명되어 있습니다.