Vue.js Watchers

Topics : Vue
Written on January 11, 2024

In Vue.js, watchers are functions that watch for changes to a specific property in the component's data and perform actions in response to those changes. Watchers are useful for responding to changes in data that may not be directly reactive or when you need to perform asynchronous or expensive operations in response to changes.

Here's a basic example of using a watcher in a Vue component:

```
<template>
<div>
<p>{{ message }}</p>
<input v-model="inputValue" />
</div>
</template>

<script>
export default {
data() {
return {
message: 'Initial message',
inputValue: ''
};
},
watch: {
// Watch for changes to the 'inputValue' property
inputValue(newValue, oldValue) {
console.log(`Input value changed from ${oldValue} to ${newValue}`);
this.updateMessage(newValue);
}
},
methods: {
updateMessage(value) {
// Perform some asynchronous or expensive operation
// For example, an API request
setTimeout(() => {
this.message = `Updated message: ${value}`;
}, 1000);
}
}
};
```

</script>

In this example:

- The `inputValue` property is bound to an input field using `v-model`.
- The `watch` option is used to define a watcher for the `inputValue` property.
- The watcher function is called whenever the value of `inputValue` changes. It receives the new and old values as arguments.
- Inside the watcher, we call the `updateMessage` method, which simulates an asynchronous operation (e.g., an API request) and updates the `message` property after a delay.

Watchers provide a way to react to changes in data and perform actions accordingly. They are especially useful when you need to coordinate changes between multiple properties or when you need to perform side effects based on data changes.