Vue.js Emits

Topics : Vue
Written on January 11, 2024

In Vue.js, the `emit` method is used to allow child components to communicate with their parent components by triggering custom events. This mechanism is useful when a child component needs to notify its parent about an event or send data to the parent component.

Here's an overview of how to use `emit` in Vue.js:

Child Component Emits an Event:

1. **Child Component (ChildComponent.vue):**

   ```
   <template>
   <button @click="sendMessage">Send Message to Parent</button>
   </template>

   <script>
   export default {
   methods: {
   sendMessage() {
   // Emitting a custom event named 'message'
   this.$emit('message', 'Hello from child!');
   }
   }
   };
   </script>
   ```

   In this example, when the button is clicked, the `sendMessage` method is called, and it emits a custom event named 'message' using `this.$emit('message', 'Hello from child!')`. The second argument is optional and represents additional data that can be sent with the event.

Parent Component Listens to the Event:

2. **Parent Component:**

   ```
   <template>
   <div>
   <child-component @message="handleMessage"></child-component>
   ```

```
<p>Received Message: {{ receivedMessage }}</p>
</div>
</template>

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

export default {
components: {
'child-component': ChildComponent
},
data() {
return {
receivedMessage: ''
};
},
methods: {
handleMessage(message) {
// Handling the custom event and updating the data in the parent component
this.receivedMessage = message;
}
}
};
</script>
```

In the parent component, the `child-component` is used, and it listens for the custom event named 'message' using `@message="handleMessage"`. When the 'message' event is emitted by the child component, the `handleMessage` method in the parent component is called, and it updates the `receivedMessage` data property.

Passing Data with the Event:

You can pass additional data along with the emitted event. In the child component:

```
this.$emit('message', { text: 'Hello from child!', timestamp: Date.now() });
```

In the parent component:

```
<child-component @message="handleMessage"></child-component>

<script>
methods: {
handleMessage(data) {
console.log(data.text); // 'Hello from child!'
console.log(data.timestamp); // Timestamp when the event was emitted
}
}
</script>
```

Using `v-on` Directive:

You can also use the `v-on` directive as a shorthand for `@` when listening to events:

```
<child-component v-on:message="handleMessage"></child-component>
```