

Module 3 Day 12

Vue Methods &
Vue Event Handling



Module 3 Day 12

Can you ?

- Implement event handling using Vue using the `v-on` directive
- Use "event modifiers", like `propagation` and `default`, in Vue
- Use the `methods` : property of the Vue object to define methods for a component
- Utilize component methods inside of event handlers

Vue Methods

Before tackling handlers we will introduce one more piece of Vue.JS, the Vue method.

- A Vue method is similar to a function or method in other languages - they are called when needed, optionally taking in parameters and providing some kind of output.
- Just like with the computed section, the methods section is comprised of JavaScript, and is defined in the script section of a Vue component.

Vue Methods vs Computed Properties

Methods and Computed properties were designed for different purposes.

- You use a ***computed*** property to generate derived data based on the data in your JSON model. These are used in the same way as derived properties of a class.
- You use a ***method*** to define processes that manipulate the state of the Model or View itself; they may also return values and accept parameters. Vue methods resemble functions in other languages. Methods can be called (more on this in a minute).

Defining Vue Methods

Vue methods go into their own section, they are a peer of the data and computed sections.

```
<script>
export default {
  name: "product-review",
  data() {
    ...
  },
  computed: {
    ...
  },
  methods: {
    //your methods go here
  }
}
</script>
```

Defining Vue Methods

Vue methods are defined in a similar fashion as computed properties, with successive methods split by a comma:

```
methods: {  
  numberOfReviews(reviews, starType) {  
    return reviews.reduce( (currentCount, review) => {  
      return currentCount + ( review.rating === starType ? 1 : 0);  
    }, 0);  
  },  
  
  addNewReview() {  
    this.reviews.unshift(this.newReview);  
    this.resetForm();  
  },  
  
  resetForm() {  
    this.showForm = false;  
    this.newReview = {};  
  }  
}
```

- Here we have three distinct methods being defined.
- The first method shows that a method can take on parameters and return a value.

Calling Vue Methods

Vue methods work flexibly and can be called in the following contexts:

- Within a v-on directive in the template section (more on this later)
- By a computed property: When we do this, the computed property needs to take a parameter called “vm” which stands for the current Vue instance: i.e. **vm.myMethod()**;
- By another function.

Event Handling Review

- Think back a few lectures ago, we added event listeners to DOM elements so that certain actions might be taken in response to events that take place on the web page.
- The Vue framework provides a directive to do this within components.

The v-on directive

- The v-on directive takes on the following pattern:

v-on: **event**= **'action to take'**

- Here are some examples:

Here we saying: when the user clicks on the span, set the JSON data property to 0.

```
<span class="amount" v-on:click="filter = 0">{{ averageRating }}</span>
```

Here we saying: when the user submits the form, call the method **addNewReivew**

```
<form v-if="showForm === true" v-on:submit.prevent="addNewReview">
```

Event modifiers: prevent

- The v-on directive can be modified with a prevent keyword, which prevents the default behavior of a HTML element from executing:

```
<form v-if="showForm === true"  
v-on:submit.prevent="addNewReview">
```

Note that on the previous example we are overriding the default behavior of the form submission, and instead choosing to handle the scenario ourselves with our own method.

Event Modifiers: stop

- The v-on directive can also be modified with a stop keyword, disabling event bubbling up the DOM.
- For a full overview of Vue events, please see the official documentation:
<https://vuejs.org/v2/guide/events.html>